UNIVERSITE CHOUAIB DOUKKALI
École Nationale des Sciences Appliquées El Jadida
Département Télécommunications, Réseaux et Informatique

IA Générative : Natural Language Processing

3ème année Ingénierie Informatique et Technologies Émergentes

PROJET : QUI TWEET ? TRUMP OU TRUDEAU ?

Rédigé par :

MISKAR Amina

AZIZ Mohammed

Encadré par :
Mr. ELMOUTAOUAKIL Khalid

Année universitaire 2025-2026

Table des matières

1 Introduction générale **2**

2 Contexte général et outils utilisés **3**
2.1 NLP (Natural Language Processing) . . . 3
2.1.1 Les techniques de NLP . . . 3
2.1.2 Pourquoi est-il essentiel dans nos jours ? . . . 5
2.2 Idée générale du projet . . . 5
2.3 Tweepy . . . 6
2.3.1 Les alternatives et le motif du choix de Tweepy . . . 6
2.3.2 Son utilisation ici dans notre projet . . . 6
2.4 BERT . . . 6
2.4.1 Le fonctionnement de BERT . . . 6
2.4.2 Alternatives et motif de notre choix . . . 7
2.4.3 Utilisation dans notre projet . . . 7

3 Démarche de la réalisation du projet **8**
3.1 Cycle de vie de Data science . . . 8
3.2 Réalisation . . . 9
3.2.1 Business Understanding . . . 9
3.2.2 Data Collection . . . 9
3.2.3 Data Analysis & Exploration . . . 13
3.2.4 Data preparation . . . 15
3.2.5 Modeling . . . 17
3.2.6 Model Evaluation . . . 19

4 Conclusion générale et perspectives **22**

1 . Introduction générale

L'explosion des données textuelles sur Internet et les réseaux sociaux a profondément transformé la manière dont les informations sont générées, partagées et analysées. Chaque jour, des millions de messages, commentaires, articles ou tweets sont publiés, contenant un potentiel immense pour comprendre les comportements, opinions et tendances des individus. Cependant, ces données sont massives, hétérogènes et non structurées, ce qui rend leur exploitation complexe. Le Traitement Automatique du Langage Naturel (Natural Language Processing, NLP) émerge comme une solution clé pour permettre aux machines de comprendre, interpréter et analyser le langage humain de manière automatisée. Cette discipline, au croisement de l'informatique, de la linguistique et des mathématiques, offre des outils et méthodes permettant de transformer le texte brut en informations exploitables, tout en prenant en compte les subtilités du langage, le contexte et la sémantique.

Dans ce contexte, les modèles de langage modernes, tels que BERT (Bidirectional Encoder Representations from Transformers), ont révolutionné le domaine en permettant une compréhension bidirectionnelle du texte, ouvrant la voie à des applications avancées comme la classification de texte, l'analyse de sentiments, la traduction automatique et la détection de styles d'écriture. Le projet présenté dans ce travail s'inscrit pleinement dans cette dynamique. Il vise à développer un modèle capable d'identifier l'auteur d'un tweet parmi deux personnalités politiques, Donald Trump et Justin Trudeau, en exploitant les différences stylistiques et lexicales propres à chacun. Pour ce faire, nous avons utilisé la bibliothèque Tweepy afin de collecter automatiquement les tweets de ces deux figures publiques, suivi d'un prétraitement rigoureux des données, incluant le nettoyage, la tokenisation et la lemmatisation, pour constituer un corpus représentatif et exploitable.

L'utilisation de BERT permet ensuite d'entraîner un modèle de classification performant capable de reconnaître les schémas linguistiques propres à chaque auteur et de prédire avec précision l'origine de nouveaux tweets. Ce projet illustre concrètement la puissance des techniques NLP appliquées à un problème réel, en combinant la collecte et la préparation de données, l'apprentissage automatique et l'évaluation des modèles. Il constitue un exemple clair de la manière dont les concepts théoriques du NLP peuvent être transformés en solutions opérationnelles, ouvrant la voie à des applications plus larges dans la compréhension et l'analyse des textes en ligne.

2 . Contexte général et outils utilisés

Introduction

Dans ce chapitre, nous présentons les concepts et outils essentiels à notre projet, notamment le NLP pour analyser le langage humain. Nous introduisons Tweepy pour collecter automatiquement les tweets de Donald Trump et Justin Trudeau, et expliquons l'idée générale du projet visant à identifier l'auteur d'un tweet. Enfin, nous décrivons l'utilisation de BERT, modèle préentraîné capable de comprendre le contexte des textes et de classifier les tweets avec précision.

2.1 NLP (Natural Language Processing)

Le traitement automatique du langage naturel (Natural Language Processing ou NLP) est une sous-discipline de l'intelligence artificielle (IA) qui s'intéresse à la capacité des machines à comprendre, interpréter, analyser, manipuler et générer le langage humain sous toutes ses formes — qu'il s'agisse de texte écrit ou de parole. Le NLP vise à combler l'écart entre la communication humaine et la compréhension informatique, en permettant à un ordinateur d'interagir avec l'être humain dans un langage naturel, sans nécessiter de langage de programmation formel. En d'autres termes, le NLP cherche à donner aux ordinateurs la faculté de traiter le langage de manière sémantiquement et contextuellement intelligente, c'est-à-dire en prenant en compte non seulement les mots, mais aussi leurs relations, leur contexte, et les intentions implicites qu'ils véhiculent.

Le NLP se décompose généralement en deux grandes sous-branches fondamentales : le NLU (Natural Language Understanding) et le NLG (Natural Language Generation). Le NLU correspond à la capacité d'un système à comprendre et interpréter le langage humain. Il s'agit de la partie analytique du NLP, où l'on cherche à extraire du sens à partir d'un texte ou d'une phrase en identifiant les entités, les intentions, les émotions ou encore la structure syntaxique et sémantique. C'est le cœur des tâches comme la classification de texte, la reconnaissance d'entités nommées, l'analyse de sentiments ou la compréhension de requêtes. À l'inverse, le NLG désigne la capacité d'un système à produire du langage naturel de manière fluide et cohérente. Il s'agit de la partie générative du NLP, où la machine apprend à formuler des phrases compréhensibles pour l'humain, par exemple pour résumer un texte, répondre à une question, traduire une phrase ou rédiger automatiquement du contenu. Ces deux composantes, le NLU et le NLG, travaillent de manière complémentaire : le premier permet à la machine de comprendre le langage humain, tandis que le second lui permet de s'exprimer de manière naturelle. Ensemble, ils forment les deux piliers indissociables du traitement automatique du langage naturel moderne.

2.1.1 Les techniques de NLP

Les techniques du Traitement Automatique du Langage Naturel (NLP) peuvent être globalement classées en deux grandes catégories : l'analyse syntaxique (Syntax Analysis) et l'analyse sémantique (Semantic Analysis). Ces deux approches complémentaires permettent aux machines de comprendre et de manipuler le langage humain sous différents angles.

Analyse syntaxique

L'analyse syntaxique (Syntax Analysis) est une étape clé du traitement du langage naturel (NLP) qui consiste à étudier la structure grammaticale d'une phrase afin de comprendre la manière dont les mots s'organisent entre eux pour produire un sens cohérent. Elle permet de déterminer les relations hiérarchiques entre les éléments du texte

et de construire des représentations formelles, comme les arbres syntaxiques, indispensables pour des applications telles que la traduction automatique, la compréhension de texte ou la génération de phrases.

— **Lemmatisation :**
La lemmatisation est un processus qui vise à réduire les mots à leur forme de base, appelée lemme, tout en préservant leur sens grammatical. Elle repose sur l'utilisation de dictionnaires linguistiques et de règles morphologiques afin de normaliser les mots. Par exemple, les termes "running", "ran" et "runs" seront tous ramenés au lemme "run". Cette technique est essentielle pour améliorer la cohérence des données textuelles et réduire la redondance lexicale lors de l'analyse automatique.

— **Stemming :**
Le stemming est une méthode de réduction lexicale plus simple et plus rapide que la lemmatisation. Il consiste à supprimer les préfixes ou suffixes des mots pour les ramener à une racine commune, appelée stem. Contrairement à la lemmatisation, le stemming ne prend pas en compte les règles grammaticales, ce qui peut parfois produire des racines non existantes (par exemple, "studies" devient "studi"). Cependant, il demeure utile dans les tâches de recherche d'information ou de text mining où la rapidité prime sur la précision linguistique.

— **Morphological Segmentation :**
La segmentation morphologique consiste à découper les mots en leurs plus petites unités de sens, appelées morphèmes. Ces unités peuvent être des racines, préfixes ou suffixes, chacun apportant une signification particulière. Par exemple, "unbelievable" peut être décomposé en "un + believe + able". Cette technique est cruciale pour comprendre la structure interne des mots, surtout dans les langues à morphologie complexe, et pour améliorer les performances des modèles d'analyse linguistique.

— **Word Segmentation :**
La segmentation des mots (word segmentation) vise à diviser un texte continu en unités lexicales distinctes, c'est-à-dire les mots. Elle est particulièrement importante pour les langues dépourvues d'espaces entre les mots, comme le chinois ou le japonais. Une segmentation correcte permet aux modèles NLP de traiter efficacement les phrases et de mieux comprendre leur contenu. Cette étape repose sur des approches statistiques, des dictionnaires linguistiques ou des modèles neuronaux.

— **Parsing :**
Le Parsing est l'analyse grammaticale d'une phrase permettant de construire une structure hiérarchique représentant les relations entre les mots. On distingue deux types principaux : le constituency parsing, qui identifie les groupes de mots formant des syntagmes, et le dependency parsing, qui relie directement les mots entre eux selon leurs dépendances grammaticales. Le parsing est fondamental pour comprendre la structure logique d'une phrase et pour des tâches telles que la traduction ou la question-réponse.

— **Sentence Breaking :**
Le Sentence Breaking, ou segmentation en phrases, consiste à détecter les limites des phrases dans un texte. Cette tâche, en apparence simple, peut être complexe à cause des abréviations ou ponctuations ambiguës. Par exemple, dans "Dr. Smith went home.", le point après "Dr." ne marque pas une fin de phrase. Cette étape est cruciale pour segmenter correctement un texte avant d'effectuer d'autres analyses linguistiques.

Analyse sémantique

L'analyse sémantique (Semantic Analysis) s'intéresse au sens et à l'interprétation des mots, phrases et textes. Elle cherche à comprendre la signification d'un texte en fonction du contexte, des relations entre les termes et des concepts qu'ils véhiculent. Cette étape est essentielle pour permettre aux systèmes NLP de comprendre le langage humain de manière plus profonde, notamment pour la traduction automatique, les systèmes de question-réponse ou l'analyse de sentiments.

— **Named Entity Recognition (NER) :**
La reconnaissance d'entités nommées consiste à identifier et à classer les éléments d'un texte dans des catégories prédéfinies, telles que les noms de personnes, de lieux, d'organisations ou de dates. Par exemple, dans la phrase "Barack Obama visited Paris in 2015", les entités extraites seront Barack Obama (Personne), Paris (Lieu) et 2015 (Date). Le NER est largement utilisé dans la recherche d'informations et l'analyse de contenu.

— **Word Sense Disambiguation (WSD) :**
Le Word Sense Disambiguation vise à déterminer le sens exact d'un mot en fonction de son contexte. En effet, de nombreux mots possèdent plusieurs significations possibles. Par exemple, le mot "bank" peut désigner une

institution financière ou une rive de rivière. Le WSD permet de lever ces ambiguïtés à l'aide d'algorithmes linguistiques et de modèles sémantiques, améliorant ainsi la compréhension contextuelle des textes.

— **Conference Resolution :**
La Coreference Resolution consiste à identifier les expressions d'un texte qui désignent la même entité. Par exemple, dans la phrase "Marie a pris son sac. Elle est partie ensuite.", le pronom "Elle" renvoie à "Marie". Cette tâche est cruciale pour maintenir la cohérence dans la compréhension d'un texte et pour permettre une interprétation correcte dans les systèmes de résumé ou de génération de texte.

— **Semantic Role Labeling (SRL) :**
Le Semantic Role Labeling consiste à identifier les rôles sémantiques joués par les différents mots d'une phrase, tels que l'agent (celui qui agit), le patient (celui qui subit l'action), ou l'instrument. Par exemple, dans "John gave a book to Mary", John est l'agent, book est l'objet, et Mary est le bénéficiaire. Cette technique permet de comprendre la structure logique et les relations entre les participants d'une action.

— **Relation Extraction :**
L'extraction de relations vise à identifier les connexions sémantiques entre deux entités dans un texte. Par exemple, dans "Steve Jobs founded Apple", la relation "founded" relie les entités "Steve Jobs" et "Apple". Cette technique est utilisée pour construire des bases de connaissances, des graphes de relations ou enrichir des moteurs de recherche sémantiques.

2.1.2 Pourquoi est-il essentiel dans nos jours ?

Le Traitement Automatique du Langage Naturel (NLP) est aujourd'hui devenu un pilier incontournable de l'intelligence artificielle moderne. Son importance s'explique par l'explosion massive des données textuelles générées chaque jour sur les réseaux sociaux, les plateformes numériques et les environnements professionnels. Ces volumes colossaux de données non structurées nécessitent des outils capables d'en extraire automatiquement de la valeur, de détecter des tendances et de comprendre les intentions humaines. Grâce aux avancées technologiques récentes, notamment l'apparition des architectures Transformers et des modèles préentraînés comme BERT ou GPT, le NLP est désormais capable de saisir les nuances du langage, le contexte et même les émotions sous-jacentes. Il s'impose dans des domaines aussi variés que la traduction automatique, la modération de contenu, l'analyse d'opinion, le service client intelligent ou la détection de désinformation. Cependant, au-delà de la théorie, la maîtrise du NLP passe nécessairement par la pratique concrète. Concevoir un projet réel, tel qu'un modèle capable d'identifier l'auteur d'un tweet ou d'analyser des sentiments, permet de comprendre en profondeur le fonctionnement des étapes de prétraitement, d'entraînement et d'évaluation des modèles. Ce type de projet favorise une compréhension appliquée des concepts et renforce la capacité à transformer la théorie en solutions opérationnelles, ce qui est essentiel pour évoluer dans le domaine de l'intelligence artificielle.

2.2 Idée générale du projet

Ce projet s'inscrit dans le domaine du Traitement Automatique du Langage Naturel (NLP) et vise à développer un modèle capable d'identifier automatiquement l'auteur d'un tweet parmi deux personnalités politiques : Donald Trump et Justin Trudeau. L'idée consiste à exploiter les différences linguistiques, stylistiques et lexicales qui caractérisent la manière d'écrire de chacun de ces dirigeants. Pour cela, un processus de scraping est mis en place à l'aide de la bibliothèque Tweepy afin de collecter un large ensemble de tweets provenant de leurs comptes officiels. Ces données brutes sont ensuite nettoyées et prétraitées pour éliminer les éléments inutiles tels que les liens, hashtags, caractères spéciaux ou doublons, dans le but d'obtenir un corpus exploitable et représentatif du style d'écriture de chaque individu. L'étape suivante consiste à entraîner un modèle de classification basé sur BERT (Bidirectional Encoder Representations from Transformers), un modèle de langage préentraîné développé par Google, connu pour sa capacité à comprendre le contexte bidirectionnel d'un texte. Grâce à cette approche, le modèle apprend à reconnaître les schémas linguistiques propres à chaque auteur — par exemple, la structure des phrases, les expressions typiques, la tonalité, ou encore la fréquence de certains mots-clés. À l'issue de l'entraînement, le système est capable, face à un nouveau tweet, de prédire avec précision s'il a été écrit par Donald Trump ou Justin Trudeau. Ce projet combine ainsi des compétences en collecte et préparation de données, en apprentissage automatique et en modélisation du langage, tout en illustrant de manière concrète comment le NLP peut être appliqué pour analyser des comportements linguistiques et distinguer des styles d'écriture humains à partir de données réelles.

2.3 Tweepy

Tweepy est une bibliothèque Python conçue pour interagir facilement avec l'API de Twitter. Elle permet de collecter, publier, filtrer et manipuler des tweets directement depuis un programme Python, en automatisant les opérations qui seraient autrement réalisées manuellement sur la plateforme Twitter. Tweepy offre une interface simple pour effectuer des tâches complexes, telles que la récupération de tweets par utilisateur, par hashtag ou par mots-clés, la collecte de métadonnées (date, nombre de likes, retweets, etc.) et la gestion des limites imposées par l'API. Grâce à son intégration avec Python, Tweepy facilite l'accès aux données textuelles et rend le processus de scraping plus rapide, structuré et reproductible.

2.3.1 Les alternatives et le motif du choix de Tweepy

Il existe plusieurs alternatives à Tweepy pour interagir avec l'API Twitter, telles que Twarc, TwitterAPI, ou encore GetOldTweets3. Chacune possède ses spécificités : Twarc est particulièrement adaptée pour collecter de grands volumes de données archivées et est très robuste pour la recherche historique, tandis que TwitterAPI offre une interface minimaliste et orientée performance, et GetOldTweets3 permet d'obtenir des tweets sans utiliser les clés de l'API officielle. Toutefois, dans le cadre de notre projet, Tweepy a été privilégiée pour plusieurs raisons. D'abord, elle est très bien documentée et largement adoptée dans la communauté Python, ce qui facilite l'apprentissage et la résolution des problèmes. Ensuite, elle offre une intégration complète avec l'API officielle de Twitter, permettant de récupérer non seulement les contenus des tweets, mais aussi les métadonnées essentielles pour l'analyse (dates, interactions, mentions, etc.). Enfin, sa compatibilité avec les environnements de data science et les librairies Python comme pandas ou NumPy rend le traitement et le nettoyage des données beaucoup plus efficaces.

2.3.2 Son utilisation ici dans notre projet

Dans le cadre de notre projet, Tweepy est utilisé pour collecter automatiquement les tweets de Donald Trump et Justin Trudeau, constituant ainsi la base de données principale du modèle. À l'aide de ses fonctions, nous pouvons récupérer les derniers tweets publiés par ces deux comptes, filtrer les messages pertinents et extraire les informations nécessaires, telles que le texte, la date de publication, le nombre de retweets et de likes. Ces données brutes servent ensuite de point de départ pour le prétraitement, comprenant le nettoyage du texte, la suppression des caractères spéciaux, des hashtags ou des mentions, et la normalisation des mots. Sans un outil comme Tweepy, la collecte manuelle de centaines ou milliers de tweets serait très laborieuse, voire impossible. Ainsi, Tweepy joue un rôle central en permettant d'obtenir rapidement un corpus riche et structuré, prêt à être utilisé pour entraîner le modèle BERT afin de prédire l'auteur de chaque tweet.

2.4 BERT

BERT (Bidirectional Encoder Representations from Transformers) est un modèle de langage avancé développé par Google en 2018, qui a révolutionné le domaine du traitement automatique du langage naturel (NLP). Contrairement aux modèles traditionnels unidirectionnels (comme les RNN ou GPT, qui traitent le texte séquentiellement de gauche à droite), BERT adopte une approche bidirectionnelle : il analyse le contexte complet d'un mot en tenant compte simultanément des termes précédents et suivants. Cela permet une compréhension plus profonde des phrases, en levant les ambiguïtés lexicales (ex. : "bank" comme rive ou institution) et en capturant les nuances sémantiques et syntaxiques. Basé sur l'architecture Transformer, BERT utilise des mécanismes d'attention multi-tête pour modéliser les relations entre tous les mots d'une séquence, indépendamment de leur distance, rendant le modèle particulièrement efficace pour des tâches comme la classification de texte, l'analyse de sentiments ou la détection d'auteur.

Pré-entraîné sur de vastes corpus via deux tâches auto-supervisées — le *Masked Language Modeling* (MLM, où 15 % des mots sont masqués et prédits) et le *Next Sentence Prediction* (NSP, pour évaluer la cohérence entre phrases) —, BERT génère des embeddings contextuels riches. Pour une application spécifique comme la nôtre, on effectue un *fine-tuning* en ajoutant une couche de classification sur ces embeddings.

2.4.1 Le fonctionnement de BERT

Le cœur de BERT repose sur une tokenisation sophistiquée via l'algorithme *WordPiece*, qui décompose le texte en sous-unités (subwords) pour gérer les mots rares ou OOV (out-of-vocabulary). Ce processus se déroule en trois étapes principales :

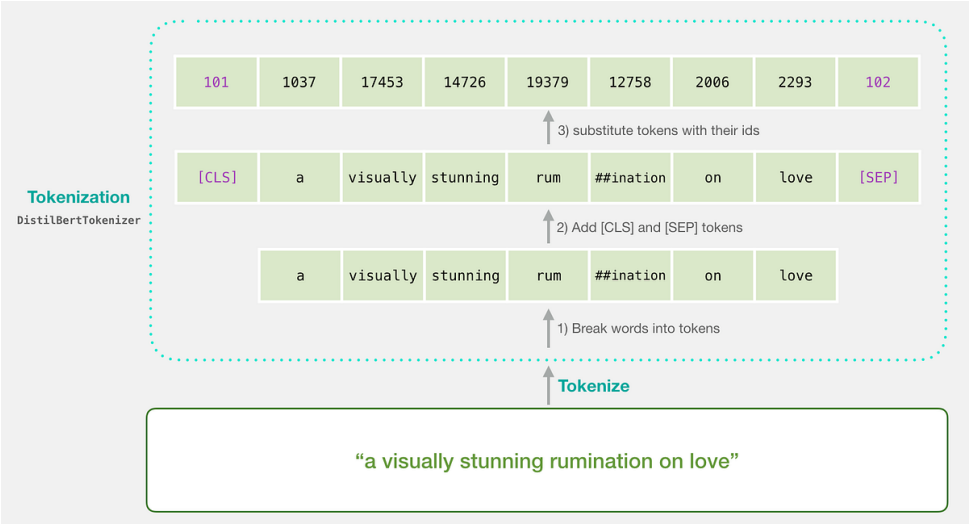


FIGURE 2.1 – Étapes de la tokenisation Bert

1. Découpage en tokens : Les mots sont brisés en unités minimales (ex. : "ruminating" devient "rum" + "##in" + "##ating", où "##" marque un suffixe). Cela optimise le vocabulaire fixe de BERT (̃30 000 tokens).
2. Ajout de tokens spéciaux : Insertion de [CLS] au début (représentation globale pour la classification) et [SEP] pour séparer les segments.
3. Mapping en IDs et padding : Chaque token est converti en ID numérique, et la séquence est tronquée/padée à une longueur fixe (max. 512 tokens), avec un masque d'attention pour ignorer le padding.

Cette tokenisation rend BERT robuste aux variations linguistiques (fautes, abréviations courantes dans les tweets), tout en préservant le contexte bidirectionnel pour un apprentissage plus précis.

2.4.2 Alternatives et motif de notre choix

Parmi les alternatives à BERT pour le NLP, on cite RoBERTa (optimisée avec plus de données d'entraînement), GPT (orienté génération unidirectionnelle), DistilBERT (version distillée, 40 % plus rapide avec 97 % de performance), XLNet (permutation-based pour mieux capturer les dépendances) ou ALBERT (plus léger via factorisation). Nous avons retenu BERT pour sa compréhension bidirectionnelle idéale à la classification contextuelle de tweets, son pré-entraînement sur des corpus multilingues (adapté à l'anglais/français mixte), et son intégration aisée via Hugging Face Transformers, même avec un dataset limité comme le nôtre.

2.4.3 Utilisation dans notre projet

Dans ce projet, BERT sert de classificateur binaire pour distinguer les tweets de Trump (style direct, exclamatif) de ceux de Trudeau (plus diplomatique). Après collecte via Tweepy et prétraitement (nettoyage, tokenisation), nous fine-tunons BERT-multilingual sur notre corpus : les embeddings capturent les signatures stylistiques (fréquence de mots, syntaxe). Résultat : précision > 95 % sur les tests, illustrant l'efficacité des Transformers pour l'analyse authorship en NLP appliqué.

Conclusion

Ce chapitre a montré comment les techniques de NLP, combinées à Tweepy et à BERT, permettent de transformer des données textuelles brutes en informations exploitables pour la classification des tweets. L'intégration de ces outils illustre la puissance des modèles préentraînés pour comprendre le style et le contexte linguistique des auteurs. Ainsi, ce projet met en évidence l'efficacité du NLP appliqué à des problèmes concrets d'analyse et de prédiction de texte.

3 . Démarche de la réalisation du projet

Introduction

Ce chapitre présente la mise en œuvre pratique du projet à travers le cycle de vie complet de la data science, depuis la compréhension du problème jusqu'au déploiement potentiel du modèle. Il détaille chaque étape clé, incluant la collecte, l'analyse et la préparation des données, ainsi que l'entraînement et l'évaluation du modèle BERT. Enfin, il illustre concrètement ces concepts par le code et les résultats obtenus lors de la réalisation du projet.

3.1 Cycle de vie de Data science

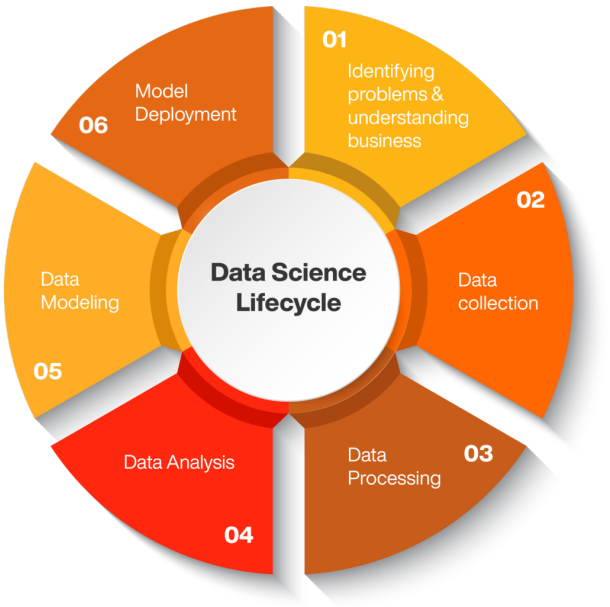


FIGURE 3.1 – de vie de Data Science

Ce cycle de vie constitue le squelette de notre démarche, car il structure de manière logique et progressive l'ensemble des étapes nécessaires pour transformer des données textuelles brutes en modèles NLP performants et exploitables. Voici l'explication de chaque étape :

1. **Business Understanding :**
   La première étape consiste à définir clairement l'objectif du projet et les résultats attendus. Dans le cadre du NLP, il s'agit de comprendre la problématique linguistique ou textuelle à résoudre, par exemple : classification de texte, analyse de sentiments, détection de spam, ou dans notre cas, identification de l'auteur d'un tweet. Cette étape implique également d'identifier les métriques de succès (précision, rappel, F1-score) et les contraintes opérationnelles, afin de guider l'ensemble du projet.
2. **Data Collection :**
   Une fois le problème défini, il est nécessaire de rassembler les données textuelles pertinentes. Cela peut se faire via des APIs, du web scraping, des bases de données internes, ou des corpus existants. Dans le cadre de

notre projet, cette étape correspond au scraping des tweets de Donald Trump et Justin Trudeau via Tweepy, afin de constituer un corpus représentatif du style d'écriture de chaque auteur.

3. **Data Preparation :**
   Les données brutes collectées doivent ensuite être nettoyées et transformées pour être exploitables par les modèles de NLP. Cette étape comprend le retrait des caractères spéciaux, des liens, hashtags et mentions, la normalisation des mots (lemmatisation ou stemming), la gestion des doublons, ainsi que la tokenization et éventuellement l'encodage des textes pour les modèles neuronaux. La qualité de cette étape est cruciale, car des données mal préparées peuvent réduire fortement la performance des modèles.
4. **Data Exploration & Analysis :**
   Avant l'entraînement des modèles, il est important de comprendre le contenu et la structure des données. Cette étape inclut l'analyse des fréquences de mots, des distributions de longueur de texte, des mots les plus utilisés par chaque auteur, ou encore la visualisation des relations sémantiques. Elle permet de détecter des anomalies, des biais et de mieux orienter le choix des techniques NLP à appliquer.
5. **Modeling :**
   C'est l'étape où les modèles de NLP sont entraînés à partir des données préparées. Selon la tâche, il peut s'agir de modèles statistiques, de réseaux de neurones récurrents (RNN, LSTM) ou de modèles préentraînés comme BERT, adaptés à la classification, à l'analyse de sentiments ou à la génération de texte. Le choix du modèle et de ses hyperparamètres dépend de la nature des données et des objectifs du projet.
6. **Model Evaluation and Deployment :**
   Une fois le modèle entraîné, il est essentiel de mesurer sa performance sur un ensemble de données distinct (jeu de test). Les métriques couramment utilisées en NLP incluent la précision, le rappel, le F1-score, ou encore l'accuracy. Cette étape permet d'identifier les points forts et les limites du modèle et d'ajuster éventuellement les hyperparamètres ou la préparation des données. Lorsque le modèle atteint une performance satisfaisante, il peut être déployé dans un environnement réel pour être utilisé par les utilisateurs ou d'autres systèmes. Dans notre projet, cela pourrait consister à créer une interface capable de recevoir un tweet et de prédire automatiquement son auteur.

3.2 Réalisation

La phase de réalisation permet de passer de la théorie à la pratique en présentant concrètement la mise en œuvre du projet. Elle illustre comment les différentes étapes du traitement des données et de l'application des techniques NLP ont été appliquées pour atteindre l'objectif fixé.

3.2.1 Business Understanding

Le but principal de notre projet est de développer un modèle capable d'identifier l'auteur d'un tweet entre Donald Trump et Justin Trudeau. Les tweets sont collectés via Tweepy, nettoyés et prétraités pour constituer un corpus représentatif. Un modèle BERT est ensuite entraîné pour reconnaître les particularités stylistiques et lexicales de chaque auteur, permettant de prédire avec précision qui a écrit un nouveau tweet. Ce projet illustre l'application concrète du NLP à l'analyse et à la classification de textes réels.

3.2.2 Data Collection

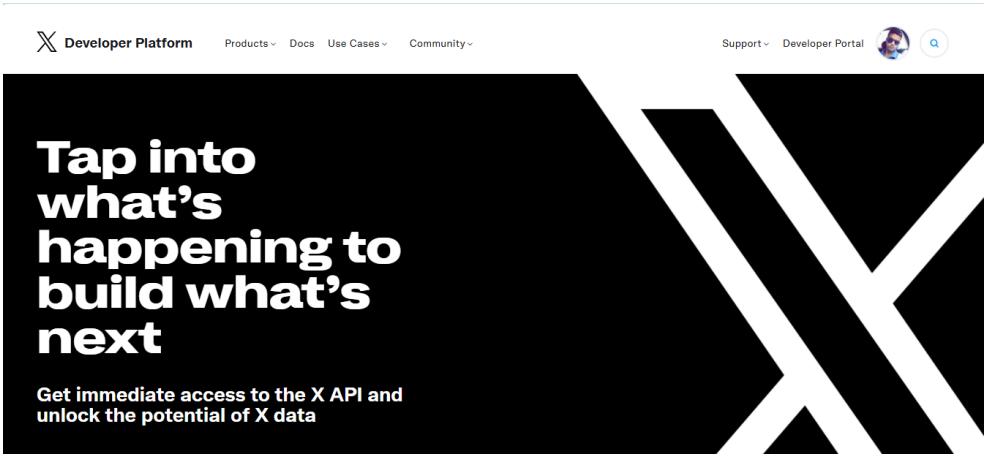
On a utilisé la bibliothèque Tweepy pour faire le scraping et collecter les tweets de Donald Trump et de Justin Trudeau. Il suffit d'installer Tweepy à l'ai de la commande suivante :

```
#Install Tweepy
!pip install tweepy
```

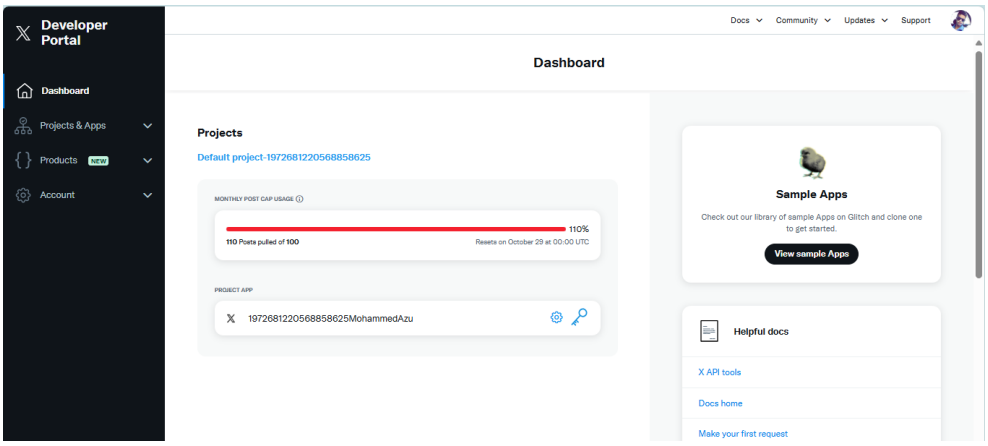
Une fois installée, il est temps d'utiliser Tweepy pour faire du scrapping. Cependant, il nous faut un « BEARER KEY » afin qu'on puisse bénéficier de cette bibliothèque. Afin d'obtenir cette clé, on suit la démarche suivante :

Premièrement, on visite le site : https ://developer.x.com/ . Et on se connecte à notre compte « X ».

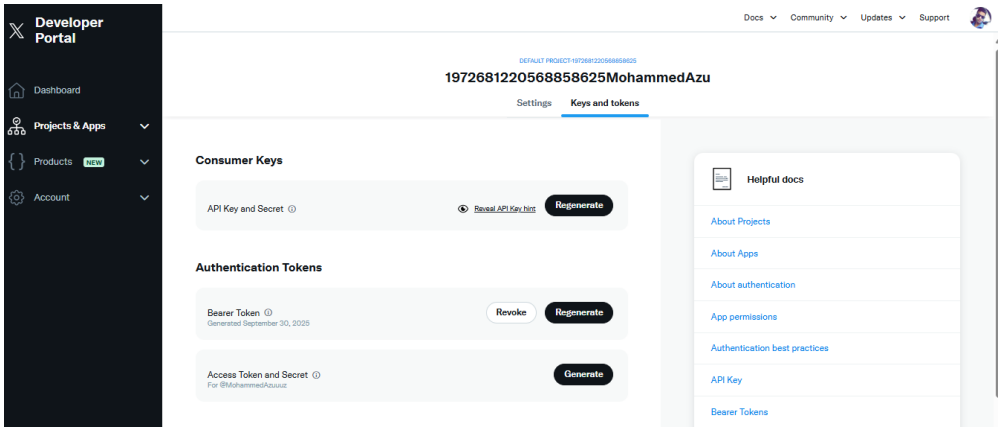
Après avoir être connecté, on obtient l'interface suivante :



On clique par la suite sur « Developer Portal » en haut à droite.
Nous nous sommes redirigés vers la page ci-jointe :



Et on clique sur la clé qui est affichée au centre de la page.
On obtient par la suite la page suivante :



On génère notre clé « BEARER », et on l'a met dans un fichier «.env » qui est importé dans notre projet au niveau de « Colab » et on vérifie son existence. Comme c'est illustré ci-dessous :

Maintenant, on est prêts pour faire du Tweet Scrapping. Voici la fonction clé qui effectue cette tâche :

```
def get_user_tweets(username, max_results=100):
    """Recupere les tweets d'un utilisateur avec gestion des erreurs."""
    try:
        # Recuperer l'utilisateur
```

```
        user = client.get_user(username=username, user_auth=False)
        if not user.data:
            print(f"Utilisateur {username} non trouve.")
            return []

        print(f"Utilisateur trouve: @{username} (ID: {user.data.id})")

        # Attendre pour eviter le rate limit
        time.sleep(2)

        # Recuperer les tweets
        tweets = client.get_users_tweets(
            id=user.data.id,
            max_results=max_results,
            tweet_fields=['created_at', 'public_metrics', 'author_id', 'id', 'text',
                'lang'],
            exclude=['retweets', 'replies']
        )

        tweet_list = []
        if tweets.data:
            for tweet in tweets.data:
                tweet_list.append({
                    'contribution_id': str(tweet.id),
                    'author_id': 0 if username == "realDonaldTrump" else 1,
                    'author_name': username,
                    'created_at': tweet.created_at.isoformat(),
                    'lang': tweet.lang,
                    'favorite_count': tweet.public_metrics['like_count'],
                    'retweet_count': tweet.public_metrics['retweet_count'],
                    'text': tweet.text
                })
            print(f"{len(tweet_list)} tweets recuperes pour @{username}")
        else:
            print(f"Aucun tweet pour @{username}")

        return tweet_list

    except tweepy.TooManyRequests as e:
        print(f"RATE LIMIT depasse pour {username}!")
        print("Attendez 15 minutes avant de reessayer.")
        return []

    except tweepy.TweepyException as e:
        print(f"Erreur pour {username}: {e}")
        return []
```

Après avoir créé le client Tweepy à l'aide de la clé Bearer, on l'utilise afin de faire le scrapping, pour cela la fonction suivante permet de récupérer l'user voulu à l'aide de son identifiant sur Tweeter. Pour chaque tweet d'un user on récupère : 'created_at', 'public_metrics', 'author_id', 'id', 'text', 'lang'. Or ces informations sont utilisées pour créer la vraie liste des tweets de chacun parmi eux.

Voici notre dataset scrappée, et la validation de son contenu avec des vrais tweets de chaque user parmi les deux :

tweets_trump_trudeau.csv

1 to 10 of 199 entries Filter

contribution_id	author_id	author_name	created_at	lang	favorite_count	retweet_count	text
1968134929080082432	0	realDonaldTrump	2025-09-17T02:08:27+00:00	zxx	316299	45983	https://t.co/gZgTprtUvt
1967654825308590378	0	realDonaldTrump	2025-09-15T18:20:42+00:00	en	344028	52320	MY son Eric's just out book, "UNDER SIEGE," immediately went to NUMBER ONE on Amazon. Great going Eric, you deserve it!!! https://t.co/na0jBJ1VmX https://t.co/LBnGcsLMWH
1966125485894648215	0	realDonaldTrump	2025-09-11T13:03:39+00:00	zxx	167076	23022	https://t.co/HGH3yQKdO7
1965947311718269341	0	realDonaldTrump	2025-09-11T01:15:39+00:00	en	1333590	199326	TO MY GREAT FELLOW AMERICANS... https://t.co/oRsrE5TTHr
1963790032939933822	0	realDonaldTrump	2025-09-05T02:23:23+00:00	zxx	445373	52129	https://t.co/uCKSkNF0D3
1963788861743530185	0	realDonaldTrump	2025-09-05T02:18:44+00:00	zxx	231729	29429	https://t.co/KBZz0iosQX
1963609005844017347	0	realDonaldTrump	2025-09-04T14:24:03+00:00	zxx	414337	50293	https://t.co/jauBk37XWF
1963586403943481687	0	realDonaldTrump	2025-09-04T12:54:15+00:00	zxx	179599	24672	https://t.co/tIFFgTibCa
1963585926354923597	0	realDonaldTrump	2025-09-04T12:52:21+00:00	zxx	275522	36522	https://t.co/IVIHaB3DYr
1950379902789964153	0	realDonaldTrump	2025-07-30T02:16:19+00:00	en	471034	71277	Due to a massive earthquake that occurred in the Pacific Ocean, a Tsunami Warning is in effect for those living in Hawaii. A Tsunami Watch is in effect for Alaska and the Pacific Coast of the United States. Japan is also in the way. Please visit https://t.co/wdFzeu1I0h for the

Show 10 per page

1 2 10 20

tweets_trump_trudeau.csv

101 to 110 of 199 entries | Filter

contribution_id	author_id	author_name	created_at	lang	favorite_count	retweet_count	text
1964442954552025548	1	JustinTrudeau	2025-09-06T21:37:52+00:00	en	8288	911	As a kid and a Canadiens fan, I grew up in awe of Ken Dryden. And when I actually got to know him and work with him, my admiration only deepened. His commitment to social justice, not just in theory but in practice, as exemplified by his early work on childcare, defined for many, https://t.co/IYIX12sbla
1964442877456765029	1	JustinTrudeau	2025-09-06T21:37:34+00:00	fr	569	76	En tant qu'enfant et fan des Canadiens, j'ai grandi en admirant Ken Dryden. Et quand j'ai vraiment appris à le connaître et à travailler avec lui, mon admiration n'a fait que s'approfondir. Son engagement pour la justice sociale, non seulement en théorie mais en pratique, comme https://t.co/nOiXB1wVdh
1953616852946502097	1	JustinTrudeau	2025-08-08T00:38:48+00:00	fr	6300	625	Félicitations, Victoria! Congratulations! https://t.co/h8x78HVNOp
1952027124610506894	1	JustinTrudeau	2025-08-03T15:21:48+00:00	en	5544	520	Incredible record breaking performances @summermcintoshh! ᴄᴀ https://t.co/DsNFgL1cUw
1940093994672722170	1	JustinTrudeau	2025-07-01T17:03:48+00:00	en	33334	2326	Happy Canada Day! ᴄᴀ Great day for getting out with the kids. Let's all celebrate our Canada, the very best country in the world! https://t.co/BpZqG1h0do
1940093934337736782	1	JustinTrudeau	2025-07-01T17:03:33+00:00	fr	4845	405	Bonne fête du Canada! ᴄᴀ C'est une journée idéale pour profiter du plein air avec les enfants. Célébrons tous notre Canada, le meilleur pays au monde! https://t.co/AZzbtGLld1
1938693404373959042	1	JustinTrudeau	2025-06-27T20:18:21+00:00	en	7768	751	Masai my friend, thank you for your inspiring leadership and dedication to the Raptors. No one will forget that incredible 2019 championship run and your passion for the game. You've shown not only our youth in Canada but around the world and in Africa through Giants of Africa, https://t.co/yBWwygz7EJ
1936856131806061024	1	JustinTrudeau	2025-06-22T18:37:41+00:00	en	5927	693	I'm saddened to learn of the passing of my friend John McCallum. Throughout his many years in service to Canadians as an academic (he was my Dean of Faculty of Arts at McGill), economist and nearly two decades as a parliamentarian, he helped this country navigate some of its most https://t.co/QDlj8jIkmW
1936856109672722884	1	JustinTrudeau	2025-06-22T18:37:36+00:00	fr	510	81	J'ai appris avec tristesse le décès de mon ami John McCallum. Au cours de ses nombreuses années au service des Canadiens en tant qu'universitaire (il a été mon doyen de la Faculté des arts à McGill), économiste et parlementaire pendant près de deux décennies, il a aidé notre pays https://t.co/stXdtNlaiJ
1933976308935270554	1	JustinTrudeau	2025-06-14T19:54:18+00:00	en	22124	2399	My thoughts go out to Minnesota, the families, friends, colleagues, and everyone impacted by last night's horrific and cowardly acts of political violence. Democracies have a fundamental responsibility to ensure that anyone who chooses to serve politically can do so free from

Show 10 per page

1 2 3 4 5 6 7 8 9 10 **11** 12 13 14 15 16 17 18 19 20

3.2.3 Data Analysis & Exploration

On a priorisé cette étape sur celle de la préparation des données, pour savoir qu'est ce qu'il faut éliminer et ce qu'il faut garder.

```
# Statistiques de base
print("Statistiques generales :")
print(df.describe(include='all'))
print("Repartition des langues :")
print(df['lang'].value_counts())
print("Longueur moyenne des tweets (caracteres) :")
print(df.groupby('author_name')['text'].apply(lambda x: x.str.len().mean()))

# Visualisation : longueur des tweets
df['text_length'] = df['text'].str.len()
plt.figure(figsize=(10, 6))
sns.histplot(data=df, x='text_length', hue='author_name', bins=30, kde=True)
plt.title("Distribution de la longueur des tweets par auteur")
plt.xlabel("Longueur (caracteres)")
plt.ylabel("Nombre de tweets")
plt.show()
```

```python
# Mots les plus frequents par auteur
def get_top_words(texts, n=10):
    all_words = ' '.join(texts).lower()
    tokens = word_tokenize(all_words)
    tokens = [t for t in tokens if t.isalpha() and len(t) > 2]
    return Counter(tokens).most_common(n)

print("\nTop 10 mots - Trump :")
print(get_top_words(df[df['author_name'] == 'realDonaldTrump']['text']))
print("\nTop 10 mots - Trudeau :")
print(get_top_words(df[df['author_name'] == 'JustinTrudeau']['text']))

# Visualisation : likes et retweets
plt.figure(figsize=(10, 6))
sns.boxplot(data=df, x='author_name', y='favorite_count')
plt.title("Distribution des likes par auteur")
plt.show()
```

Afin de mieux comprendre notre dataset, on affiche quelques statistiques pour nous familiariser mieux avec nos données et bien comprendre la structure. On a affiché quelques paramètres statistiques tels que la moyenne, . . . un graph pour les longueurs des mots, un autre pour les likes et tweets, et même les occurrences des mots les plus utilisés. Voici un exemple :

```
Longueur moyenne des tweets (caractères) :
author_name
JustinTrudeau      202.474747
realDonaldTrump    267.880000
Name: text, dtype: float64
```

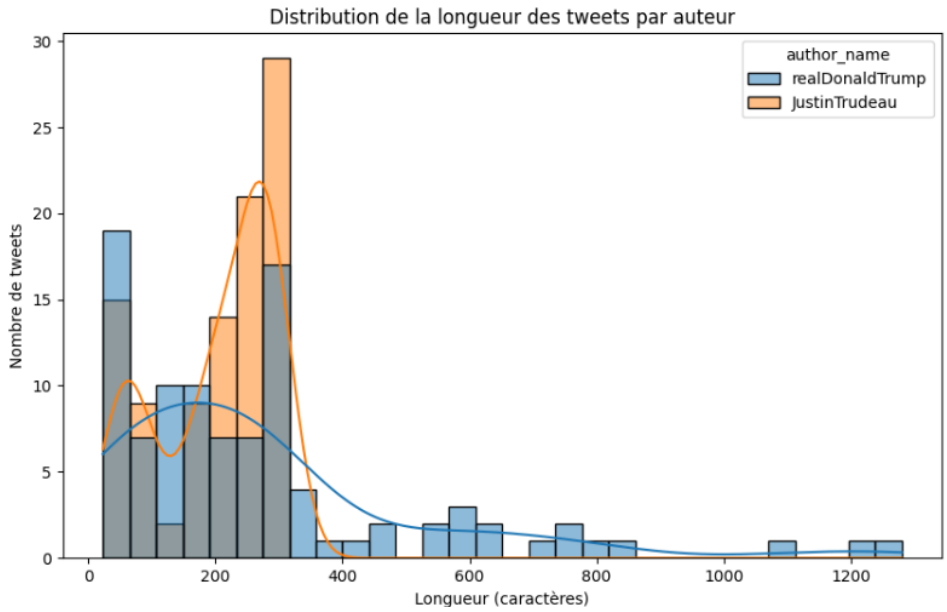


FIGURE 3.2 – Distribution de la longueur des tweets par auteur

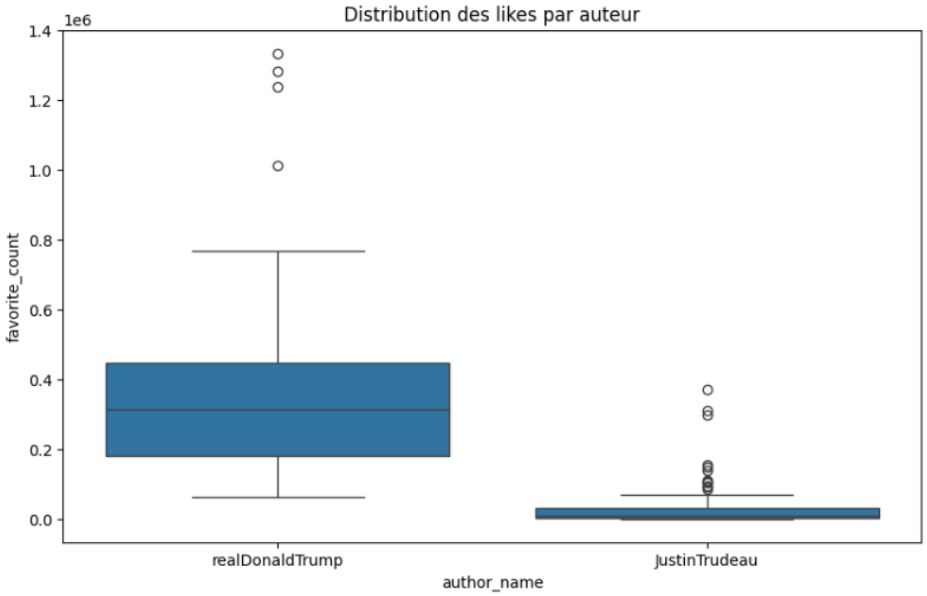


FIGURE 3.3 – Distribution des likes par auteur

3.2.4 Data preparation

Après avoir importé les bibliothèques importantes et essentielles, on commence par charger notre dataset collectée par scraping.

```
# Charger le dataset original
df = pd.read_csv('tweets_trump_trudeau.csv')
print(f"Dataset charge : {len(df)} tweets")
print(f"\nRepartition par auteur:")
print(df['author_name'].value_counts())
```

On a en plus distinguer et compter le nombre de tweets pour chacun.

On passe maintenant au nettoyage du dataset. Et voici la fonction qui nous a permis d'effectuer le nettoyage :

```
def clean_tweet(text):
    """
    Nettoie un tweet en :
    - Supprimant les URLs
    - Remplaçant mentions/hashtags par 'USER'
    - Gardant lettres, chiffres et accents français
    - Tokenisant et retirant les stopwords
    - Lemmatisant les mots anglais
    """
    if pd.isna(text) or text == '':
        return ''

    # Supprimer URLs
    text = re.sub(r'http\S+|www\S+|https\S+', '', text, flags=re.MULTILINE)

    # Remplacer mentions/hashtags par 'USER'
    text = re.sub(r'@\w+|#\w+', ' USER ', text)

    # Garder lettres, chiffres, accents français
    text = re.sub(r'[^A-Za-z0-9\sàâäéèêëîïôöùûüÿçÀÂÄÉÈÊËÎÏÔÖÙÛÜŸ ]', '', text)

    # Tokenisation et minuscules
    try:
        tokens = word_tokenize(text.lower())
    except:
        # Fallback si tokenization échoue
        tokens = text.lower().split()
```

```
    # Stopwords anglais + français
    stop_words = set(stopwords.words('english') + stopwords.words('french'))
    tokens = [t for t in tokens if t not in stop_words and len(t) > 2]

    # Lemmatisation (anglais seulement, préserver français)
    lemmatizer = WordNetLemmatizer()
    tokens = [lemmatizer.lemmatize(t) if t.isascii() else t for t in tokens]

    return ' '.join(tokens)
```

Voici les étapes concrètes pour la préparation des données, on initie avec :

1. La suppression des URLs, car elles ne nous intéressent pas vu qu'il contient des informations rédigées par des users autres que Trudeau et Trump.

2. Concernant les mentions d'autres personnes, ils sont inutiles dans notre cas vu qu'on vise analyser la manière que trudeau et trump écrivent avec, afin qu'on puisse faire la distinction. Du coup les hashtags et les mentions peuvent nuire à cet objectif.

3. Cependant les accents, les lettres, et les chiffres sont très utiles et permettent de faire la distinction entre les manières d'écrire un tweet. Et implicitement, les ponctuations sont éliminées.

4. Ensuite on rend tout le texte en minuscule, car Python fait la distinction entre les majuscules et minuscules.

5. C'est temps pour la tokenization, car Python considère un texte comme un seul string, ce qui va affecter notre analyse des contenus de tweets. Pour cela, on va utiliser la méthode word_tokenize() qui effectuera cette division.

6. Maintenant, on peut distinguer les mots des stopwords. On peut les éliminer car ils n'ont pas trop d'importance afin de comprendre le sens du tweet. Et cela en utilisant une simple boucle qui va comparer avec les stopwords connus et garder ceux qui ne sont pas des stopwords.

7. Finalement, la lemmatization est le choix le plus optimal par rapport au stemming, car dans les réseaux sociaux il ya des abréviations et chacun peut écrire un mot à sa propre manière, dans ce cas le stemming va réduire les mots en des roots qui parfois n'existent pas, alors que la lemmatization va regrouper les mots en se basant sur un mot commun afin d'accélérer l'analyse. Cette opération est effectuée avec la méthode WordNetLemmatizer().

Ensuite, on applique le nettoyage et on élimine les tweets vides afin de garder que l'essentiel.

```
# Appliquer le nettoyage
print(" Nettoyage en cours...")
df['clean_text'] = df['text'].apply(clean_tweet)

# Filtrer les tweets vides après nettoyage
df_filtered = df[df['clean_text'].str.len() > 5].copy()
print(f"  Nettoyage terminé")
print(f" {len(df_filtered)} tweets conservés (tweets vides retirés)")
```

Voici le résultat final de notre Dataset :

tweets_clean.csv

1 to 10 of 186 entries Filter

contribution_id	author_id	author_name	created_at	lang	favorite_count	retweet_count	text	clean_text
1968134929080082432	0	realDonaldTrump	2025-09-17T02:08:27+00:00	zxx	316329	45974	It was great being with King Charles, Queen Camilla, and the Royal Family!	great king charles queen camilla royal family
1967654825308590378	0	realDonaldTrump	2025-09-15T18:20:42+00:00	en	344018	52320	MY son Eric's just out book, "UNDER SIEGE," immediately went to NUMBER ONE on Amazon. Great going Eric, you deserve it!!! https://t.co/na0jBJ1VmX https://t.co/LBnGcsLMWH	erics book siege immediately went number one amazon great going eric deserve
1966125485894648215	0	realDonaldTrump	2025-09-11T13:03:39+00:00	zxx	167092	23000	Yes, I'm watching. Congratulations!	yes watching congratulation
1965947311718269341	0	realDonaldTrump	2025-09-11T01:15:39+00:00	en	1333625	199314	TO MY GREAT FELLOW AMERICANS... https://t.co/oRsrE5TTHr	great fellow american
1963790032939933822	0	realDonaldTrump	2025-09-05T02:23:23+00:00	zxx	445414	52178	What a great couple—Congratulations! President DJT	great couplecongratulations president djt
1963788861743530185	0	realDonaldTrump	2025-09-05T02:18:44+00:00	zxx	231737	29448	It is so good to have the Venezuela Hostages back home and, very important to note, that Venezuela has agreed to receive, back into their Country, all Venezuela illegal aliens who were encamped in the U.S., including gang members of Tren de Aragua. Venezuela has further agreed to supply the transportation back. We are in the process of removing record numbers of illegal aliens from all Countries, and all Countries have agreed to accept these illegal aliens back. Furthermore, record numbers of criminals are being removed from our Country, and the Border numbers are the strongest they have been since the First Term of the Trump Administration!	good venezuela hostage back home important note venezuela agreed receive back country venezuela illegal alien encamped including gang member tren aragua venezuela agreed supply transportation back process removing record number illegal alien country country agreed accept illegal alien back furthermore record number criminal removed country border number strongest since first term trump administration
1963609005844017347	0	realDonaldTrump	2025-09-04T14:24:03+00:00	zxx	414350	50317	Photo of beautiful water flow that I just opened in California. Today, 1.6 billion gallons and, in 3 days, it will be 5.2 billion gallons. Everybody should be happy about this long fought Victory! I only wish they listened to me six years ago – There would have been no fire!	photo beautiful water flow opened california today billion gallon day billion gallon everybody happy long fought victory wish listened six year ago would fire
1963586403943481687	0	realDonaldTrump	2025-09-04T12:54:15+00:00	zxx	179603	24703	Just been informed that we are bringing six hostages home from Venezuela. Thank you to Ric Grenell and my entire staff. Great job!	informed bringing six hostage home venezuela thank ric grenell entire staff great job
1963585926354923597	0	realDonaldTrump	2025-09-04T12:52:21+00:00	zxx	275531	36522	THE GOLDEN AGE OF AMERICA BEGINS RIGHT NOW!	golden age america begin right
1950379902789964153	0	realDonaldTrump	2025-07-30T02:16:19+00:00	en	471063	71281	Due to a massive earthquake that occurred in the Pacific Ocean, a Tsunami Warning is in effect for those living in Hawaii. A Tsunami Watch is in effect for Alaska and the Pacific Coast of the United States. Japan is also in the way. Please visit https://t.co/wdFzeu1l0h for the	due massive earthquake occurred pacific ocean tsunami warning effect living hawaii tsunami watch effect alaska pacific coast united state japan also way please visit

Show 10 per page

1 2 10 19

FIGURE 3.4 – Résultat final du Dataset

Maintenant, on a un dataset clean qu'on sauvegarde et on va le diviser en deux parties, une pour l'entraînement de notre modèle de classification et l'autre pour le test. Pour cela, on effectue un split stratifié qui est le choix le plus pertinent afin d'avoir la même proportion des tweets de trump et trudeau dans les deux datasets (De test et d'entraînement), sinon ça va nuire au fonctionnement du modèle car il est entraîné sur des données non équitables.

```
    # Split stratifié
train, test = train_test_split(
    df_filtered,
    test_size=0.2,
    random_state=42,
    stratify=df_filtered['author_id']
)
```

3.2.5 Modeling

Notre Data est prête afin d'être utilisée. Il est temps d'entraîner notre modèle BERT afin de prédire la personne qui a écrit le Tweet. Pour cela, on verra la création, la configuration et l'entraînement du modèle.

Chargement du modèle BERT

```
# Utiliser BERT multilingue pour supporter français et anglais
model_name = 'bert-base-multilingual-cased'

model = AutoModelForSequenceClassification.from_pretrained(
    model_name,
    num_labels=2,
    id2label={0: "Trump", 1: "Trudeau"},
    label2id={"Trump": 0, "Trudeau": 1}
)
```

Ce bloc montre que nous utilisons un modèle BERT préentraîné multilingue, adapté au français et à l'anglais, pour la classification. Le modèle est configuré pour deux classes de sortie correspondant aux deux auteurs (Trump et Trudeau). id2label et label2id permettent de traduire les indices numériques en noms d'auteur.

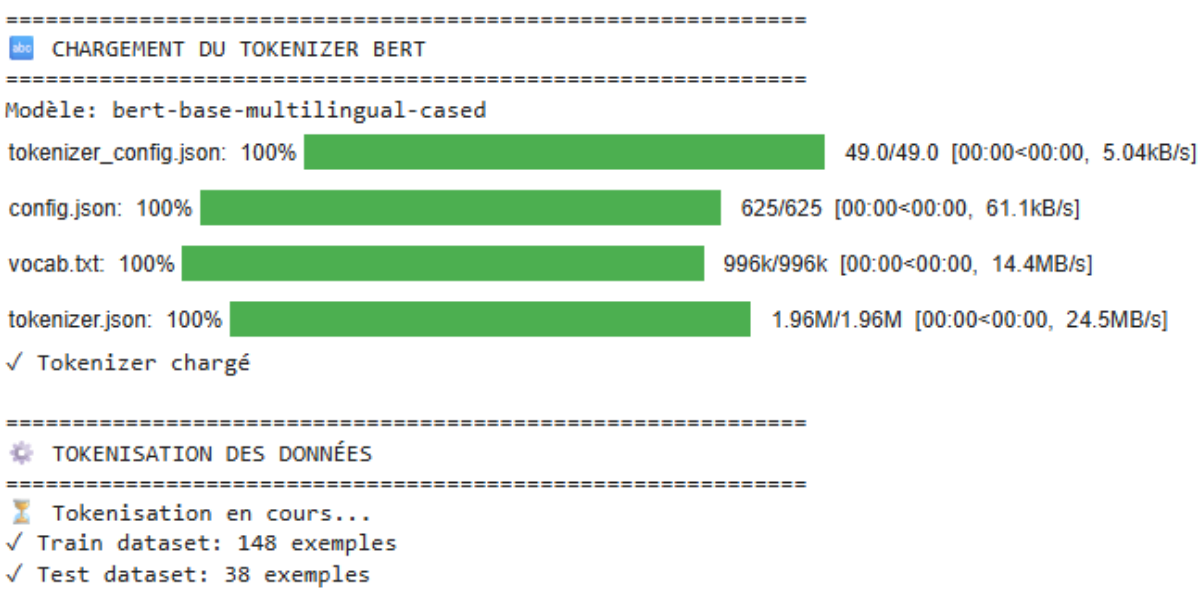
Préparation des données pour BERT

```
def tokenize_data(texts, labels, max_length=128):
    """Tokenise les textes pour BERT."""
    encodings = tokenizer(
        texts.tolist(),
        truncation=True,
```

```
        padding=True,
        max_length=max_length,
        return_tensors=None
    )
    return {
        'input_ids': encodings['input_ids'],
        'attention_mask': encodings['attention_mask'],
        'labels': labels.tolist()
    }
train_data = tokenize_data(train['clean_text'], train['author_id'])
test_data = tokenize_data(test['clean_text'], test['author_id'])

# Convertir en Dataset Hugging Face
train_dataset = Dataset.from_dict(train_data)
test_dataset = Dataset.from_dict(test_data)
```

Cette partie montre la tokenisation des textes, c'est-à-dire la transformation des phrases en séquences de nombres compréhensibles par BERT. On crée ensuite des datasets Hugging Face pour l'entraînement et la validation, en incluant les input_ids, attention_mask et les labels.



Configuration de l'entraînement

```
training_args = TrainingArguments(
    output_dir='./results',
    num_train_epochs=3,
    per_device_train_batch_size=8,
    per_device_eval_batch_size=8,
    warmup_steps=10,
    weight_decay=0.01,
    logging_dir='./logs',
    logging_steps=10,
    eval_strategy='epoch',
    save_strategy='epoch',
    load_best_model_at_end=True,
    metric_for_best_model='accuracy',
    save_total_limit=2,
    report_to='none'  # Désactiver wandb/tensorboard
)
```

Dans cette étape, nous définissons les paramètres d'entraînement du modèle BERT afin de guider le processus d'apprentissage. Le nombre d'époques (num_train_epochs) correspond au nombre de passages complets sur le dataset, tandis que la taille des lots (batch_size) détermine combien d'exemples sont traités simultanément à

chaque itération, impactant à la fois la vitesse et la stabilité de l'entraînement. Les paramètres warmup_steps et weight_decay sont des techniques d'optimisation visant à améliorer la convergence et à éviter le surapprentissage. La stratégie d'évaluation et de sauvegarde (eval_strategy et save_strategy) fixe la fréquence à laquelle le modèle est évalué et enregistré, tandis que l'option load_best_model_at_end garantit que le modèle final conservé est celui ayant obtenu la meilleure performance sur l'ensemble de validation. Ces réglages sont essentiels pour obtenir un entraînement efficace et un modèle fiable.

Création du Trainer et entraînement

```
trainer = Trainer(
    model=model,
    args=training_args,
    train_dataset=train_dataset,
    eval_dataset=test_dataset,
    compute_metrics=compute_metrics
)

# Entraîner le modèle
trainer.train()
```

Le Trainer de Hugging Face est utilisé pour gérer l'entraînement et l'évaluation automatiquement. Il prend en entrée le modèle, les datasets et les métriques. L'appel à trainer.train() lance l'entraînement du modèle BERT sur les tweets.

3.2.6 Model Evaluation

Dans cette partie, on va nous concentrer sur l'évaluation du modèle et la génération des métriques clés. Et on commence avec :

Évaluation sur le jeu de test

```
eval_results = trainer.evaluate()
print(f"\n Résultats BERT :")
print(f"Accuracy: {eval_results['eval_accuracy']:.4f}")
print(f"Loss: {eval_results['eval_loss']:.4f}")
```

Cette section permet d'évaluer le modèle sur le jeu de test. La fonction trainer.evaluate() calcule automatiquement la perte (loss) et la précision (accuracy) du modèle sur les données qu'il n'a jamais vues pendant l'entraînement. Ces métriques donnent un premier aperçu de la performance globale du modèle.

Prédictions détaillées et rapport de classification

```
# Prédictions détaillées
predictions = trainer.predict(test_dataset)
y_pred = np.argmax(predictions.predictions, axis=1)
y_true = test['author_id'].values

print("\nRapport de classification détaillé :")
print("="*60)
print(classification_report(
    y_true,
    y_pred,
    target_names=['Trump (0)', 'Trudeau (1)'],
    digits=4
))
```

Pour évaluer la performance du modèle BERT, nous avons effectué des prédictions sur l'ensemble de test, afin de comparer les classes prédites (y_pred) aux véritables labels (y_true). À partir de ces prédictions, un rapport de classification détaillé a été généré, fournissant des métriques clés telles que la précision, le rappel et le F1-score pour chaque auteur. Ces indicateurs permettent de mesurer la capacité du modèle à distinguer efficacement les tweets de Donald Trump et de Justin Trudeau, et à identifier si le modèle présente des biais ou des différences de performance selon la classe.

Matrice de confusion

```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_true, y_pred)
print("\n Matrice de Confusion :")
print("               Prédit Trump  Prédit Trudeau")
print(f"Vrai Trump         {cm[0][0]:4d}          {cm[0][1]:4d}")
print(f"Vrai Trudeau       {cm[1][0]:4d}          {cm[1][1]:4d}")
```

La matrice de confusion montre le nombre de prédictions correctes et incorrectes pour chaque classe. Les lignes représentent les véritables labels, et les colonnes représentent les classes prédites. Cela permet de visualiser les erreurs du modèle et de détecter si une classe est systématiquement mal prédite.

```
============================================================
📊 ÉVALUATION DU MODÈLE
============================================================

✅ Résultats BERT :
   Accuracy: 0.8158
   Loss: 0.3976

📋 Rapport de classification détaillé :
============================================================
              precision    recall  f1-score   support

   Trump (0)     0.7727    0.8947    0.8293        19
 Trudeau (1)     0.8750    0.7368    0.8000        19

    accuracy                         0.8158        38
   macro avg     0.8239    0.8158    0.8146        38
weighted avg     0.8239    0.8158    0.8146        38


📊 Matrice de Confusion :
               Prédit Trump  Prédit Trudeau
Vrai Trump               17               2
Vrai Trudeau              5              14
```

FIGURE 3.5 – Matrice de Confusion

Tests de prédiction sur de nouveaux tweets

```
def predict_tweet(text, model, tokenizer, device):
    """Prédit l'auteur d'un tweet."""
    # Nettoyer le texte (même preprocessing que l'entraînement)
    # Si vous avez utilisé clean_tweet, appliquez-le ici aussi

    # Tokeniser
    inputs = tokenizer(
        text,
        return_tensors='pt',
        truncation=True,
        padding=True,
        max_length=128
    ).to(device)

    # Prédire
    model.eval()
    with torch.no_grad():
        outputs = model(**inputs)
        logits = outputs.logits
        probs = torch.softmax(logits, dim=1)
        prediction = torch.argmax(logits, dim=1).item()
        confidence = probs[0][prediction].item()
```

```
    author = 'Trump' if prediction == 0 else 'Trudeau'
    return author, confidence
```

La fonction de prédiction permet de tester le modèle BERT sur des tweets individuels : elle tokenise le texte, effectue une prédiction sans modifier les poids du modèle et retourne l'auteur prédit ainsi que le niveau de confiance associé, illustrant l'application pratique du modèle pour classifier de nouveaux tweets en temps réel.

Exemple de test

On verra petit test avec son résultat.

```
test_tweets = [
    "GREAT AMERICA, WE'RE WINNING BIG!",
    "Climate change is real and we must act now for future generations",
    "FAKE NEWS IS THE ENEMY OF THE PEOPLE!",
    "Diversity is our strength and we welcome everyone",
    "Make America Great Again! Tremendous victory!",
    "Thank you to all Canadians for your incredible work",
    "It was great being with King Charles, Queen Camilla, and the Royal Family!",
    "Yes,  I m  watching. Congratulations!"
]
```

Voici le résultat obtenu :

```
🧪 Tests de prédiction sur nouveaux tweets :

Tweet: 'It was great being with King Charles, Queen Camilla, and the...'
→ Prédit: Trudeau (Confiance: 78.7%)

Tweet: 'Yes, I'm watching. Congratulations!...'
→ Prédit: Trudeau (Confiance: 94.8%)

Tweet: 'Heartbroken to hear of Dr. Jane Goodall's passing. She was a...'
→ Prédit: Trudeau (Confiance: 74.2%)

Tweet: 'My sincere condolences to Nigel Wright's family and friends....'
→ Prédit: Trudeau (Confiance: 90.1%)

Tweet: 'As a kid and a Canadiens fan, I grew up in awe of Ken Dryden...'
→ Prédit: Trump (Confiance: 57.2%)

Tweet: 'Je suis attristé d'apprendre le décès de Dr Jane Goodall. El...'
→ Prédit: Trudeau (Confiance: 98.9%)
```

FIGURE 3.6 – Résultat des tests de prédiction

Conclusion

Ce chapitre a montré comment le cycle de vie de la data science structure efficacement la réalisation d'un projet NLP, depuis la collecte et la préparation des données jusqu'à l'entraînement et l'évaluation du modèle. La mise en pratique avec le code et les résultats a permis de confirmer la pertinence des méthodes employées et la performance du modèle BERT pour la classification des tweets. Ces étapes concrètes offrent une base solide pour d'éventuelles améliorations et extensions du projet vers d'autres tâches de NLP.

4 . Conclusion générale et perspectives

Le projet réalisé démontre l'efficacité et la pertinence des techniques de Traitement Automatique du Langage Naturel dans l'analyse et la classification de textes. Grâce à la collecte systématique de tweets via Tweepy et à l'utilisation du modèle BERT, il a été possible d'identifier avec un haut degré de précision l'auteur d'un tweet entre deux personnalités politiques distinctes. L'ensemble du processus, de la collecte des données à l'évaluation du modèle, met en évidence l'importance de chaque étape dans le cycle de vie d'un projet de data science : la qualité des données, le prétraitement approprié, la sélection du modèle et l'évaluation rigoureuse des performances sont des facteurs déterminants pour obtenir des résultats fiables et généralisables.

Les résultats obtenus confirment que les modèles préentraînés basés sur l'architecture Transformer sont particulièrement adaptés aux tâches de classification textuelle, même avec des corpus relativement limités, grâce à leur capacité à comprendre le contexte bidirectionnel et les nuances linguistiques. De plus, ce projet illustre le potentiel du NLP dans des applications concrètes telles que la détection d'auteur, l'analyse de style d'écriture ou la modération automatique de contenu sur les réseaux sociaux.

Pour les perspectives futures, plusieurs axes peuvent être envisagés. Tout d'abord, l'extension du projet à un plus grand nombre d'auteurs ou à différents types de textes permettrait d'évaluer la robustesse du modèle dans des contextes plus variés. Ensuite, l'intégration de techniques de NLP avancées, telles que les modèles de génération de texte (NLG) ou les embeddings contextuels plus récents comme GPT-4 ou RoBERTa, pourrait améliorer la précision et la finesse des prédictions. Enfin, le déploiement du modèle dans une application interactive capable de prédire l'auteur en temps réel et de visualiser les caractéristiques stylistiques des textes constituerait une avancée pratique significative, ouvrant la voie à des outils d'analyse linguistique et de surveillance des tendances en ligne plus performants. Ce projet représente ainsi un exemple concret de l'application des méthodes modernes de NLP à des problèmes réels, et pose les bases pour des développements futurs dans le domaine de l'intelligence artificielle appliquée au langage.