



Université Mohammed V de RABAT

ECOLE NATIONALE SUPÉRIEUR D'INFORMATIQUE

ET D'ANALYSE DES SYSTÈMES



Filière : Génie de la data (GD)

Détection de profils frauduleux

RÉALISÉ PAR : — AARBAT FARAH
— AZZIZ CHAIMAA
— GUIGUI SALMA

DEVANT LE JURY — PR S. EL FKIHI
— PR. Y. TABII

ANNÉE UNIVERSITAIRE : 2024/2025

Résumé

Avec l'essor des réseaux sociaux, les faux comptes sont devenus un problème majeur, faussant les statistiques, diffusant de la désinformation et participant à des activités frauduleuses. Ce projet propose une approche basée sur l'apprentissage automatique et le traitement du langage naturel (NLP) pour développer un modèle de détection des faux utilisateurs, capable de s'adapter aux comportements de plus en plus sophistiqués de ces comptes. En analysant les données de profil, incluant le texte descriptif et les métriques de compte, le projet vise à créer une solution fiable et évolutive pour identifier les faux utilisateurs sur de grands ensembles de données.

Abstract

With the rise of social networks, fake accounts have become a major issue, distorting statistics, spreading misinformation, and engaging in fraudulent activities. This project proposes an approach based on machine learning and natural language processing (NLP) to develop a model for detecting fake users, capable of adapting to the increasingly sophisticated behaviors of these accounts. By analyzing profile data, including descriptive text and account metrics, the project aims to create a reliable and scalable solution to identify fake users across large datasets.

Table des matières

Table des figures	7
Introduction Générale	8
I Cadre théorique	9
1 Contexte générale	10
1.1 Introduction	10
1.2 Description du Projet	10
1.3 Problématique	10
1.4 Objectifs du Projet	10
1.5 Conclusion	11
2 Cadre Théorique et revue de littérature	12
2.1 Introduction	12
2.2 Fondements théoriques et techniques utilisées	12
2.2.1 Introduction au cadre théorique	12
2.2.2 État de l'art de la détection de faux comptes sur les réseaux sociaux	12
2.2.3 Techniques avancées de prétraitement et d'augmentation de données	13
2.2.4 Synthèse et justification des choix pour ce projet	14
2.3 Conclusion	14
II Cadre pratique	15
3 Ingénierie et traitement des données	16
3.1 Introduction	16
3.2 Jeu des données	16
3.2.1 Source des données	16
3.2.2 Description du jeu de données	16
3.3 Visualisation des données	17
3.3.1 Distribution des utilisateurs	18
3.3.2 Distribution des langues	18
3.3.3 Distribution de valeurs numériques	19
3.3.4 Corrélation des valeurs numériques	19
3.4 Nettoyage de données	20

3.4.1	Prévention du sur-apprentissage	20
3.4.2	Valeurs manquantes	20
3.5	Augmentation de données	22
3.6	Prétraitement des données	23
3.6.1	Normalisation des données textuelles	23
3.6.2	Vectorisation des descriptions textuelles	24
3.6.3	Encodage des variables catégorielles	24
3.6.4	Mise à l'échelle des variables numériques	24
3.7	Conclusion	24
4	Réalisation et résultats	25
4.1	Introduction	25
4.2	Choix des modèles de classification	25
4.2.1	Random Forest	25
4.2.2	Gradient Boosting	25
4.2.3	Régression Logistique	25
4.2.4	K-Nearest Neighbors (K-NN)	26
4.2.5	Support Vector Machine (SVM)	26
4.2.6	Arbre de Décision	26
4.3	Accuracy	26
4.4	Évaluation des Modèles de Classification	27
4.4.1	Analyse des performances :	27
4.4.2	Optimisation des hyperparamètres	27
4.4.3	Application sur nos modèles	28
4.5	Conclusion	28
	Conclusion générale	29
	Webographie	30

Table des figures

3.1	Logo kaggle	16
3.2	Jeu de données	17
3.3	Nombre de comptes réels et faux	18
3.4	Visualisation des données manquantes	18
3.5	Visualisation des valeurs numériques	19
3.6	Corrélation entre les variables numérique	20
3.7	Visualisation des données manquantes	21
3.8	Visualisation des données manquantes après traitement	21
3.9	Répartition des comptes réels et Fake après augmentation des données	22
3.10	Résultat de la normalisation des descriptions textuelles	23
4.1	Evaluation des modèles	26
4.2	Résultat du hyperparameters tuning	28

Introduction Générale

Avec la montée en puissance des réseaux sociaux et des plateformes en ligne, les faux comptes sont devenus un problème majeur. Ces comptes peuvent fausser les statistiques des utilisateurs, diffuser de la désinformation et même s'engager dans des activités frauduleuses. La détection et la suppression des faux comptes sont désormais essentielles pour préserver l'intégrité des plateformes, assurer la sécurité des utilisateurs et garantir l'exactitude des mesures d'engagement numérique. Les méthodes traditionnelles de détection des faux utilisateurs reposent souvent sur des systèmes basés sur des règles, qui peuvent peiner à suivre les comportements de plus en plus sophistiqués des faux comptes. Ce projet vise à tirer parti de l'apprentissage automatique pour développer une solution plus robuste et adaptable à la détection des faux utilisateurs.

La détection automatique et à grande échelle des faux comptes reste un défi. Les faux comptes sont souvent conçus pour imiter les utilisateurs légitimes, ce qui les rend difficiles à identifier avec des règles simples ou une vérification manuelle. Ce projet répond au besoin d'un modèle de classification efficace capable de distinguer les vrais utilisateurs des faux en s'appuyant sur les informations de profil disponibles. En analysant les données de profil, y compris le texte descriptif et les métriques de compte, nous avons appliqué des techniques de traitement du langage naturel (NLP) et d'apprentissage automatique (ML) pour enrichir la qualité des données et optimiser la performance de notre modèle de prédiction. Notre objectif est de développer un modèle fiable applicable à de grands ensembles de données, capable de détecter efficacement les faux utilisateurs.

Première partie

Cadre théorique

Contexte générale

1.1 Introduction

L'essor des réseaux sociaux s'accompagne de nouveaux défis, notamment la détection de profils frauduleux. Ces faux comptes, souvent utilisés à des fins malveillantes, posent des risques pour la sécurité des utilisateurs. Dans ce chapitre, nous présenterons le contexte actuel de cette problématique, les enjeux associés, et les approches existantes pour identifier ces profils frauduleux.

1.2 Description du Projet

Ce projet porte sur la détection de profils frauduleux en utilisant des données de Twing, une source de données liée à Twitter. Face à la croissance rapide des réseaux sociaux, les faux comptes sont de plus en plus fréquents et présentent divers risques pour la sécurité. L'objectif est de développer une méthode efficace pour identifier ces comptes frauduleux, en exploitant des techniques d'apprentissage automatique adaptées aux caractéristiques des données.

1.3 Problématique

La prolifération de profils frauduleux sur les réseaux sociaux représente un défi majeur pour la sécurité et la crédibilité des informations partagées. Ces comptes peuvent être utilisés pour diffuser de fausses informations, usurper des identités et manipuler l'opinion. La problématique ici est de concevoir une méthode capable de détecter avec précision ces faux profils, tout en réduisant les erreurs affectant les comptes authentiques.

1.4 Objectifs du Projet

Les objectifs de ce projet incluent la conception d'un modèle performant capable de distinguer les faux profils des vrais au sein des données Twing. En parallèle, il s'agit d'identifier les caractéristiques distinctives des comptes frauduleux, spécifiques à cette source de données. L'objectif final est d'évaluer l'efficacité de la détection sur un ensemble de données réelles, sans développer un système complet.

1.5 Conclusion

Ce chapitre a permis de poser les bases du projet, en présentant son contexte, ses enjeux, et ses objectifs. Les sections suivantes détailleront les méthodes et les outils envisagés pour atteindre ces objectifs.

Cadre Théorique et revue de littérature

2.1 Introduction

La détection de faux comptes sur Twitter est un enjeu majeur pour assurer la qualité des informations en ligne. Ce chapitre explore les approches théoriques et les recherches existantes en matière de traitement automatique du langage (NLP) et d'apprentissage automatique pour l'identification de comptes "fake". Il fournit un aperçu des techniques de prétraitement, d'augmentation de données et de classification utilisées dans ce projet, en s'appuyant sur les avancées de la littérature.

2.2 Fondements théoriques et techniques utilisées

2.2.1 Introduction au cadre théorique

Détection de faux profils : La détection de faux profils sur les réseaux sociaux, en particulier sur Twitter, est devenue un domaine crucial pour préserver l'intégrité de l'information en ligne. Les faux comptes peuvent diffuser de la désinformation, influencer des opinions ou propager des campagnes de spam. La détection de ces comptes nécessite une analyse approfondie des comportements et des caractéristiques textuelles des profils.

Concepts clés en NLP et classification : Dans ce projet, nous utilisons des techniques de traitement automatique du langage naturel (NLP) et de classification supervisée. Le NLP permet de traiter et d'analyser le texte pour extraire des caractéristiques pertinentes, tandis que la classification supervisée est utilisée pour distinguer les comptes "fake" des comptes "réels" en se basant sur ces caractéristiques.

2.2.2 État de l'art de la détection de faux comptes sur les réseaux sociaux

La détection de faux comptes sur les réseaux sociaux repose sur plusieurs approches principales, qui se répartissent en trois catégories : l'analyse comportementale, les techniques de traitement du langage naturel (NLP) et les algorithmes d'apprentissage supervisé pour la classification. Voici un aperçu des approches et des méthodes utilisées.

- **Analyse comportementale** : Cette approche consiste à analyser les comportements en ligne des comptes. Des métriques comme la fréquence de publication f_{post} ou l'intervalle moyen entre deux tweets Δt_{post} peuvent indiquer des comportements suspects. Un compte qui publie de façon intensive et à des intervalles réguliers pourrait être un signe d'automatisation.
- **Traitement du langage naturel (NLP)** : Le NLP permet de traiter le contenu textuel des profils et des tweets, en extrayant des caractéristiques sémantiques et syntaxiques pour la classification. Les techniques courantes incluent :
 - **Tokenisation** : Le texte est divisé en unités appelées "tokens" $T = \{t_1, t_2, \dots, t_n\}$, facilitant l'analyse des mots individuellement.
 - **Filtrage des Stopwords** : Les mots sans valeur informative (ex. "le", "de", "et") sont supprimés, afin de ne conserver que les termes significatifs.
 - **Stemming** : Cette technique réduit les mots à leur racine, par exemple "running" devient "run", pour limiter la variation lexicale.
- **Apprentissage supervisé pour la classification** : Pour la classification des comptes, plusieurs modèles supervisés sont utilisés. Voici les principaux algorithmes couramment appliqués dans la détection de faux comptes :
 - **Régression logistique** : Estime la probabilité qu'un compte soit "fake" ou "réel". La probabilité d'appartenance à la classe "fake" pour un vecteur de caractéristiques X est donnée par :

$$P(Y = 1|X) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X_1 + \dots + \beta_p X_p)}}$$

où $\beta_0, \beta_1, \dots, \beta_p$ sont les paramètres du modèle.

- **Forêts aléatoires** : Utilise un ensemble de K arbres de décision construits sur des sous-ensembles aléatoires des données. La prédiction finale est déterminée par un vote majoritaire.
- **Machines à vecteurs de support (SVM)** : Ce modèle sépare les données en classes en utilisant un hyperplan optimal, maximisant la marge entre les deux classes, "fake" et "réel".
- **Word embeddings** : Utilisé dans les modèles avancés comme Word2Vec ou GloVe pour générer des représentations vectorielles denses pour chaque mot, capturant les relations contextuelles dans un espace vectoriel de dimension réduite \mathbb{R}^n .

2.2.3 Techniques avancées de prétraitement et d'augmentation de données

Dans la détection de faux comptes, certaines étapes avancées sont appliquées pour préparer les données et améliorer la performance des modèles :

- **Prétraitement multilingue** : Sur Twitter, les descriptions de profil et les tweets peuvent être multilingues. Il est donc essentiel d'adapter le prétraitement aux langues détectées, en appliquant des stopwords et des stemmers spécifiques à chaque langue.
- **TF-IDF (Term Frequency-Inverse Document Frequency)** : Utilisé pour pondérer l'importance de chaque mot dans le document en fonction de sa fréquence

dans l'ensemble des documents. La formule est donnée par :

$$\text{TF-IDF}(t, d) = \text{TF}(t, d) \times \log \frac{N}{\text{DF}(t)}$$

où $\text{TF}(t, d)$ est la fréquence du mot t dans le document d , N est le nombre total de documents, et $\text{DF}(t)$ est le nombre de documents contenant t .

- **Augmentation de données** : Pour compenser le déséquilibre des classes (peu de comptes "fake" par rapport aux comptes "réels"), l'augmentation de données est appliquée en insérant des mots aléatoires dans les descriptions de la classe minoritaire. Cette technique permet de générer des exemples supplémentaires, rendant l'ensemble de données plus équilibré. Si $N_{\text{fake}} < N_{\text{real}}$, l'augmentation continue jusqu'à obtenir un équilibre approximatif entre D_{fake} et D_{real} .

2.2.4 Synthèse et justification des choix pour ce projet

À la lumière de l'état de l'art et des techniques spécifiques à la détection de faux comptes, les choix méthodologiques de ce projet sont justifiés comme suit :

- **Prétraitement et extraction de caractéristiques textuelles** : La combinaison de TF-IDF et d'une approche de stemming multilingue permet de capturer efficacement les informations textuelles nécessaires pour la classification.
- **Modèles de classification** : Les modèles choisis, y compris la régression logistique et les forêts aléatoires, sont réputés pour leur performance en classification binaire et sont adaptés à la détection de faux comptes.
- **Équilibrage des classes** : L'augmentation de données par insertion de mots aléatoires permet de résoudre le problème de déséquilibre de classes, ce qui est essentiel pour améliorer les performances du modèle.

2.3 Conclusion

Ce chapitre a présenté les bases théoriques et les avancées en matière de détection de faux comptes sur les réseaux sociaux. Nous avons exploré les concepts de NLP et de classification supervisée, ainsi que les techniques de prétraitement et d'augmentation de données pour gérer les défis liés aux données textuelles et au déséquilibre des classes. Les choix méthodologiques, comme l'utilisation du TF-IDF et de modèles de classification avancés, sont justifiés par l'état de l'art et offrent une base solide pour les étapes d'implémentation et d'évaluation des performances qui suivront.

Deuxième partie

Cadre pratique

Ingénierie et traitement des données

3.1 Introduction

Ce chapitre offre une vue d'ensemble et une introduction à notre ensemble de données. Nous commencerons par décrire notre base de données, puis nous nous attarderons sur la visualisation et le processus de nettoyage des données.

3.2 Jeu des données

3.2.1 Source des données

Les données sont acquises à partir de kaggle contenant deux fichiers csv l'un pour les utilisateurs réels et l'autre pour les utilisateurs frauduleux.



FIGURE 3.1 – Logo kaggle

3.2.2 Description du jeu de données

Le jeu de données utilisé pour ce projet contient divers attributs associés aux comptes utilisateurs. Ces attributs ont été sélectionnés pour leur capacité à différencier les utilisateurs réels des utilisateurs frauduleux, en se basant sur les profils, les mesures d'activité des comptes, ainsi que sur les données textuelles descriptives. Les principales colonnes comprennent :

- **statuses_count** : Le nombre total de publications ou de mises à jour effectuées par l'utilisateur. Cette métrique peut aider à identifier les comptes inactifs, souvent associés à des profils frauduleux.

- **followers_count** : Le nombre de followers associés à chaque compte utilisateur. Les comptes frauduleux ont tendance à avoir un faible nombre de followers par rapport à leur activité, car ils s’engagent généralement dans des comportements visant à paraître légitimes sans développer un véritable engagement.
- **friends_count** : Le nombre de comptes suivis par l’utilisateur. Des ratios disproportionnés entre les amis et les followers peuvent parfois indiquer des comportements automatisés ou de type spam.
- **favourites_count** : Le nombre total de publications marquées comme favoris par l’utilisateur. Les comptes frauduleux peuvent afficher des comptages de favoris anormalement faibles ou irréguliers, car ces comptes sont souvent moins engagés dans les comportements habituels des utilisateurs.
- **lang** : La langue principale utilisée sur le compte. La langue peut être pertinente, car certains comportements peuvent être corrélés avec des comptes spam ou automatisés dans certaines régions.
- **listed_count** : Le nombre de fois où l’utilisateur a été ajouté aux listes d’autres utilisateurs, ce qui peut indiquer sa fiabilité ou son influence.
- **default_profile** : Un indicateur booléen indiquant si l’utilisateur a personnalisé son profil. Les utilisateurs frauduleux conservent souvent les paramètres par défaut, tandis que les utilisateurs légitimes ont tendance à personnaliser leur profil.
- **geo_enabled** : Un attribut booléen indiquant si le compte est géo-activé, ce qui peut être moins fréquent dans les profils frauduleux.
- **description** : La description du profil de l’utilisateur. Ce champ textuel est traité à l’aide de techniques de traitement du langage naturel (NLP), car les comptes frauduleux peuvent inclure des descriptions génériques ou riches en mots-clés, contrairement aux comptes réels qui contiennent souvent des textes plus personnalisés.
- **is_fake** : Une étiquette indiquant si le compte est classé comme frauduleux ou réel. C’est la variable cible pour le modèle de classification.

statuses_count	followers_count	friends_count	favourites_count	listed_count	lang	default_profile	geo_enabled	description
413	274	20	80	4	0	it	0.0	free brain
2286	35	15	321	0	0	en	1.0	~ #2C7A95
887	230	5	93	0	0	it	1.0	Psichiatra Milanista Collezionista Musica Rock...
1519	55	21	614	0	0	en	1.0	Student, likewatch TV and Basketball game!
1080	155	45	173	10	0	it	1.0	Thank god it's friday.
2740	3	0	42	0	0	en	1.0	a blogger amodel a warrior
284	707	27	59	1	1	it	1.0	Un modo per vincere lo trovo sempre!
241	294	30	96	4	0	en	0.0	Faccio cose, vedo gente
2055	50	18	333	0	0	en	1.0	nu
84	34917	1161	194	2	28	en	1.0	actress, singer, artist ... now what?

FIGURE 3.2 – Jeu de données

3.3 Visualisation des données

La visualisation des données permet de mieux comprendre la distribution et les relations entre les différentes variables du jeu de données. Dans ce projet, plusieurs graphiques ont été générés pour analyser les attributs des utilisateurs réels et frauduleux.

3.3.1 Distribution des utilisateurs

Cette figure est un graphique en barres verticales qui compare le nombre de comptes réels et de faux comptes .

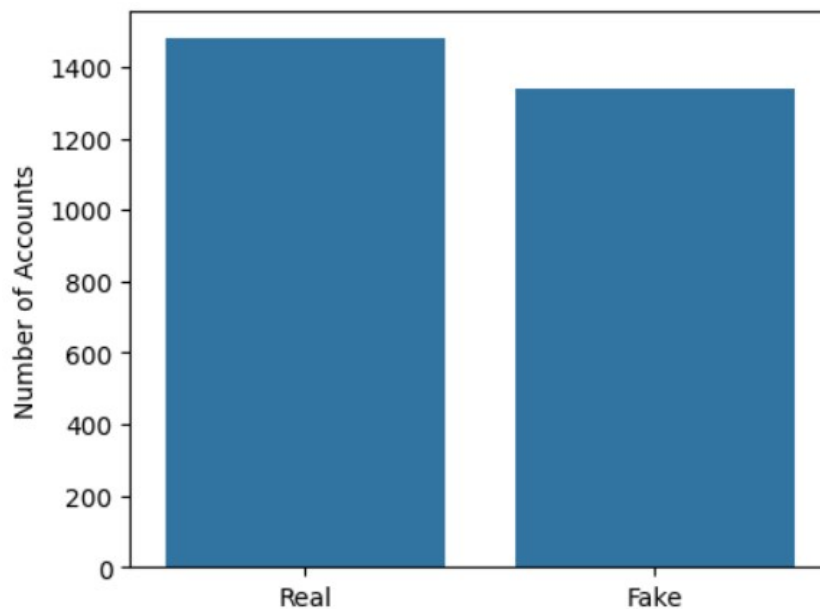


FIGURE 3.3 – Nombre de comptes réels et faux

Ce graphique met en évidence un déséquilibre significatif entre les comptes authentiques et les comptes frauduleux. Cette disparité importante sera corrigée par l'augmentation de notre jeu de données.

3.3.2 Distribution des langues

Ce graphique est un histogramme qui représente la distribution des langues utilisées dans la description dans l'ensemble de données.

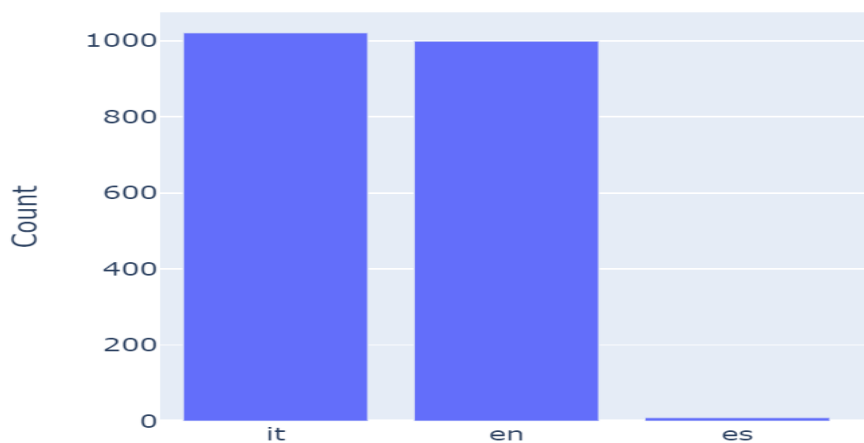


FIGURE 3.4 – Visualisation des données manquantes

3.3.3 Distribution de valeurs numériques

Ce graphique est un diagramme à barres qui compare la distribution moyenne de trois variables numériques **nombre d'abonnés, d'amis et de statuts**, entre deux catégories d'utilisateurs : les utilisateurs réels et les faux utilisateurs.

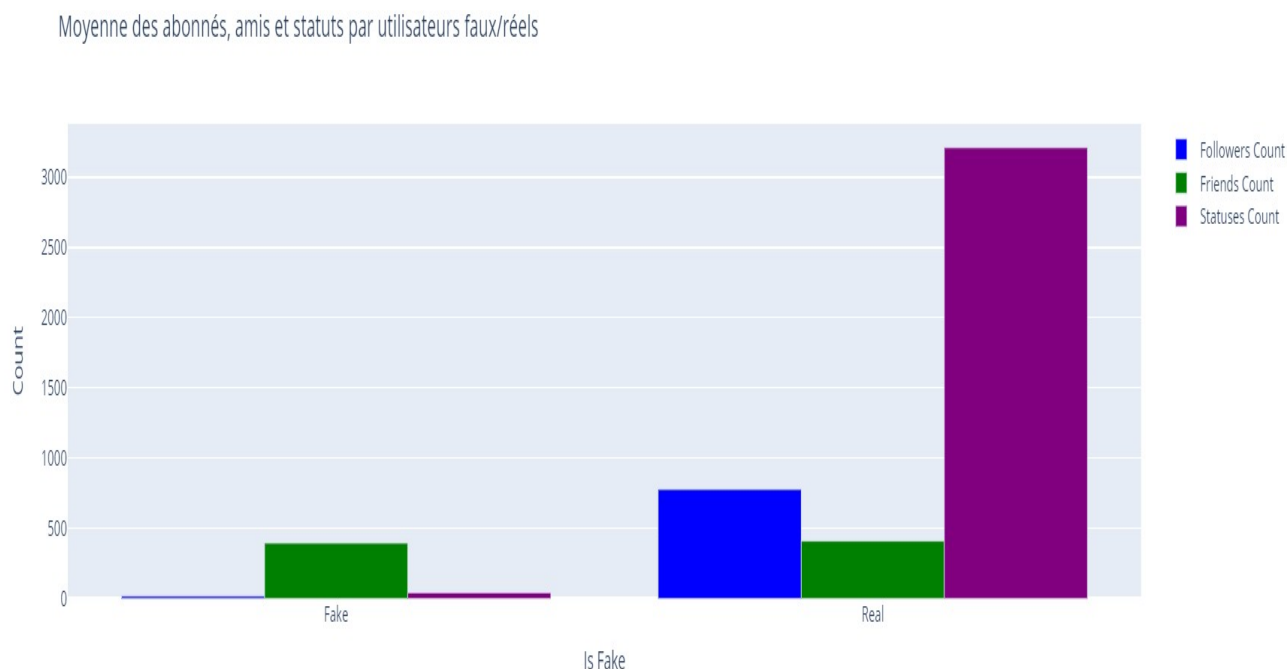


FIGURE 3.5 – Visualisation des valeurs numériques

Ce graphique suggère que le nombre d'abonnés, d'amis et de statuts peuvent être utilisés comme des indicateurs pour distinguer les comptes réels des faux comptes. Les comptes réels ont tendance à avoir un nombre plus élevé d'abonnés, d'amis et de statuts.

3.3.4 Corrélation des valeurs numériques

Cette carte de chaleur représente les corrélations entre différentes variables numériques liées à des comptes d'utilisateurs sur un réseau social. Chaque case de la carte correspond à la corrélation entre deux variables, et la couleur de la case indique l'intensité et le sens de cette corrélation.

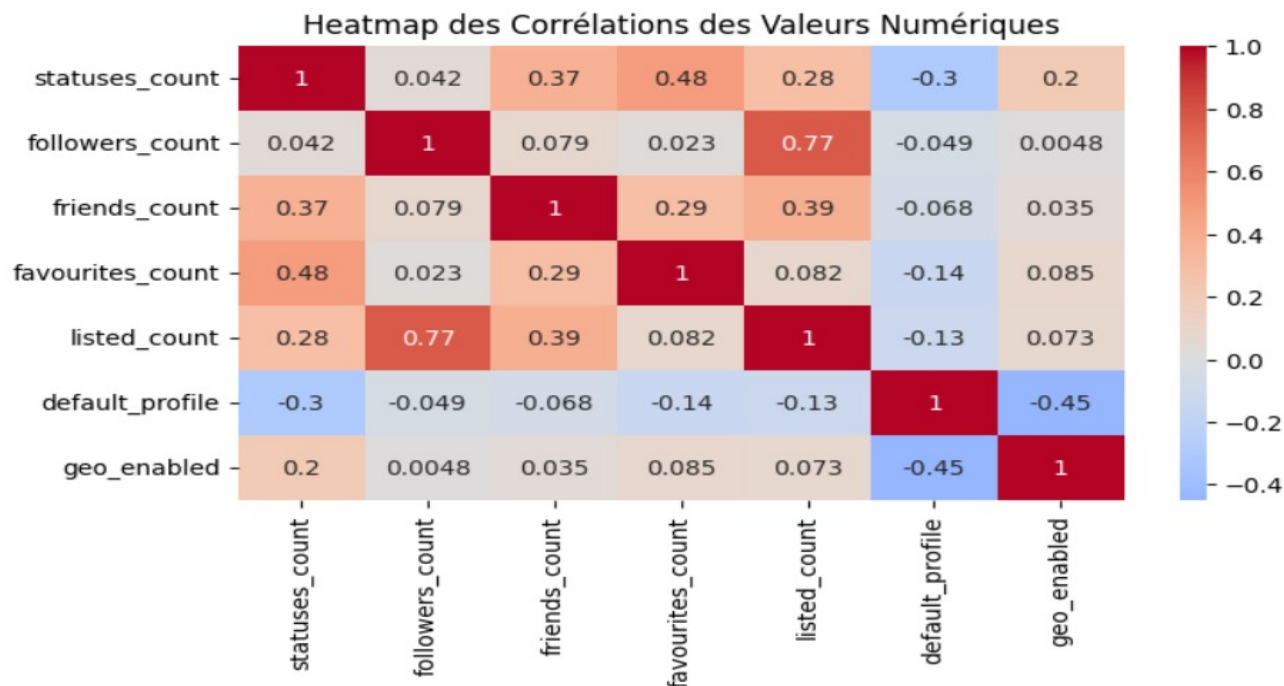


FIGURE 3.6 – Corrélation entre les variables numérique

3.4 Nettoyages de données

3.4.1 Prévention du sur-apprentissage

En phase de nettoyage de données de ce projet, nous avons d'abord divisé notre ensemble de données en données d'entraînement et de test. Cette étape est cruciale car le nettoyage et le prétraitement des données vraies et fausses ensemble pourraient entraîner un sur-apprentissage. Si les données de test subissent les mêmes transformations que les données d'entraînement, cela pourrait amener le modèle à performer anormalement bien sur l'ensemble de test, conduisant à des métriques de performance biaisées.

3.4.2 Valeurs manquantes

La figure ci-dessous présente une visualisation des valeurs manquantes dans notre jeu de données pour les différentes variables.

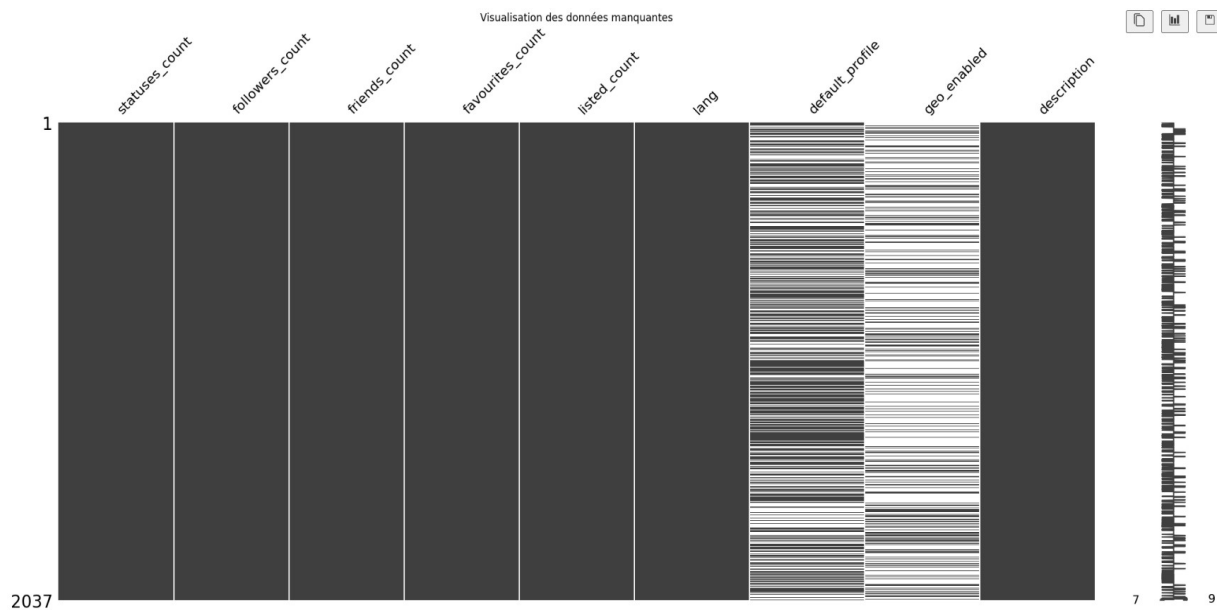


FIGURE 3.7 – Visualisation des données manquantes

Pour l'ensemble de l'entraînement, nous avons traité les valeurs manquantes en remplaçant les entrées NaN dans les colonnes 'default_profile' et 'geo_enabled' par 0. Cela a été fait pour s'assurer que l'absence d'un profil par défaut ou d'un paramètre de géolocalisation est traitée comme une caractéristique binaire, indiquant que le profil n'est pas le profil par défaut ou qu'il manque de géolocalisation.

Un processus de nettoyage similaire a été appliqué à l'ensemble de test (X_{test}), où les valeurs manquantes dans les mêmes colonnes ont été remplacées par 0 et les colonnes 'verified' et 'protected' ont été supprimées. Cela garantit la cohérence entre les deux ensembles de données tout en empêchant les fuites de données et le sur-apprentissage potentiel.

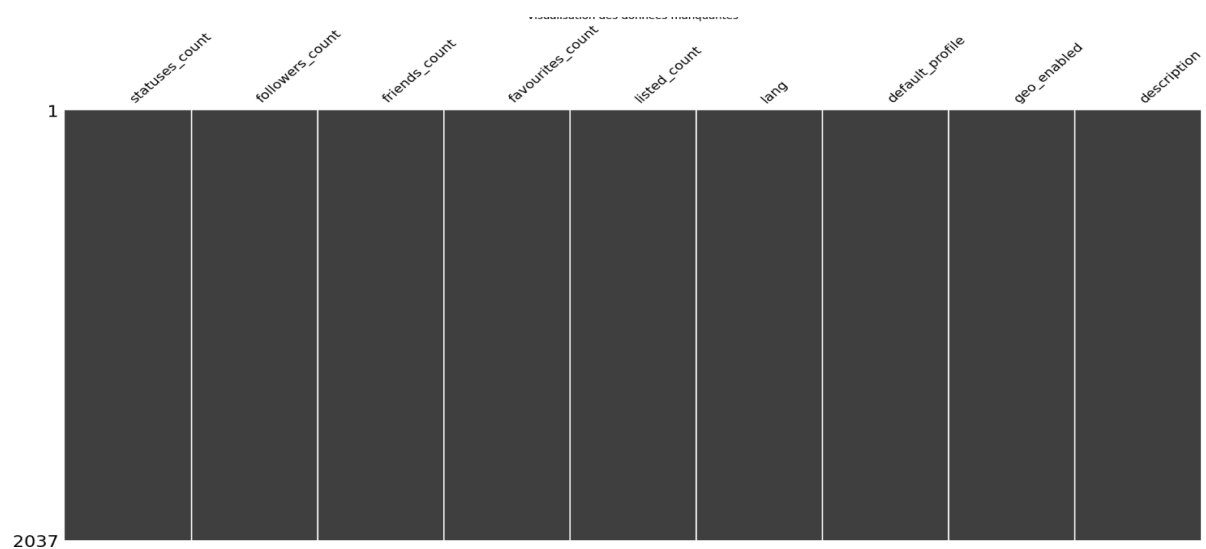


FIGURE 3.8 – Visualisation des données manquantes après traitement

3.5 Augmentation de données

Dans cette étape, une technique d'augmentation de données a été appliquée pour équilibrer les classes de l'ensemble d'entraînement. Cette augmentation est nécessaire en raison d'un déséquilibre entre le nombre d'utilisateurs réels et le nombre d'utilisateurs "fake".

- **Ajout de mots aléatoires** : Insertion des mots aléatoires dans les descriptions textuelles de la classe minoritaire. Cette étape utilise un dictionnaire de mots (`wordnet`) pour choisir des mots aléatoires et les insérer à des positions aléatoires dans le texte original. Ce processus permet de diversifier les exemples de la classe minoritaire tout en préservant la structure linguistique.
- **Calcul de la différence de comptes entre classes** : Le nombre d'utilisateurs réels et d'utilisateurs "fake" est comparé, et la différence entre les deux est calculée. Cette différence détermine le nombre de nouvelles descriptions à générer pour la classe minoritaire.
- **Augmentation des descriptions de la classe minoritaire** : Si la classe minoritaire est celle des utilisateurs "fake", des lignes de cette classe sont sélectionnées aléatoirement, et des mots aléatoires sont insérés dans leurs descriptions pour créer de nouvelles observations. Cette étape est répétée jusqu'à ce que le nombre de lignes dans chaque classe soit équilibré.
- **Création d'un nouvel ensemble de données équilibré** : Les nouvelles descriptions augmentées sont ensuite ajoutées à l'ensemble de données d'entraînement original pour créer un ensemble équilibré. Enfin, cet ensemble équilibré est mélangé aléatoirement pour éviter toute influence de l'ordre des données.

Ce processus d'augmentation de données aide à atténuer le déséquilibre entre les classes et améliore ainsi la capacité du modèle à généraliser sur les deux classes.

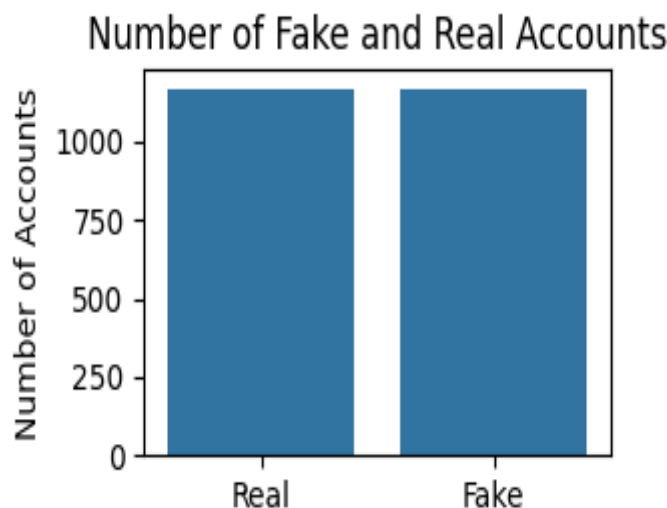


FIGURE 3.9 – Répartition des comptes réels et Fake après augmentation des données

Le nombre de comptes dans chaque classe est désormais presque égal, ce qui réduit le déséquilibre initial et permet une meilleure performance des modèles d'apprentissage.

3.6 Prétraitement des données

3.6.1 Normalisation des données textuelles

Dans notre jeu de données seul variables textuelles : **Description** à laquelle on doit appliquer les techniques du traitement du langage naturel. La fonction de normalisation des documents standardise le texte brut afin d'améliorer l'efficacité des modèles. Les étapes principales sont :

- **Suppression des caractères spéciaux et mise en minuscules** : Nettoyage du texte en supprimant les caractères non alphabétiques et en convertissant en minuscules.
- **Tokenisation** : Division du texte en tokens (mots) pour faciliter le traitement.
- **Filtrage des stopwords** : Suppression des mots courants (stopwords) pour chaque langue (anglais, italien, espagnol), conservant ainsi les mots significatifs.
- **Stemming** : Réduction des mots à leur racine selon la langue, par exemple "running" devient "run".
- **Reconstitution du document** : Les tokens nettoyés sont réunis pour former le document final normalisé.

Les valeurs manquantes dans la colonne **description** sont remplacées par des chaînes vides. Ensuite, la fonction de normalisation est appliquée à chaque description, en tenant compte de la langue, et stockée dans la colonne **processed_description** pour les étapes suivantes.

	description	processed_description
2621	http://t.co/QqJ2aiKi is coming..	httpcoqjaiki come
837	In the future everyone will be world-famous fo...	the futur everyon will be worldfamous for minutes
1646	Sharp, sensitiv, woman of the South. Married w...	sharp sensitiv woman south marri id
2310	Artist Development, Artist Management, Marketi...	artist develop artist manag market promot even...
2751	Chasing a tale of a loe-drawn distant memor.	chase tale loedrawn distant memor
...
286	Passione per innovazione,musica progressive ro...	passion innovazionemus progress rock saggistic...
754	Love the life you live, live the life you love...	lov the lif you liv liv the lif you lov bob ma...
1325	Chi ama davvero ama il mondo intero, non solta...	ama dawver ama mond inter soltant individu par...
462	â€ŽDon't settle down and sit in on place. Move...	dont settl sit place move around nomad make da...
279	Between Central London & Canary Wharf...	central london canari wharf

510 rows × 2 columns

FIGURE 3.10 – Résultat de la normalisation des descriptions textuelles

Cette figure présente un aperçu des données après le prétraitement textuel. La colonne **description** contient le texte d'origine, tandis que la colonne **processed_description** montre le texte après normalisation. Ce processus comprend la suppression des caractères spéciaux, la mise en minuscules, le filtrage des mots vides (stopwords) et l'application du stemming pour réduire les mots à leur racine. Le texte transformé est ainsi simplifié, permettant une analyse plus efficace pour les étapes de modélisation.

3.6.2 Vectorisation des descriptions textuelles

Pour extraire les informations textuelles, nous avons utilisé le `TfidfVectorizer` avec une limite de 1000 caractéristiques. Cette technique de pondération nous permet de représenter chaque description de manière numérique en fonction de l'importance des termes dans chaque texte par rapport au corpus global.

3.6.3 Encodage des variables catégorielles

La variable *lang*, représentant la langue, est encodée en format one-hot, permettant de transformer chaque catégorie en une colonne binaire.

3.6.4 Mise à l'échelle des variables numériques

Pour les autres variables numériques comme `statuses_count`, `followers_count`, `friends_count`, `favourites_count`, `listed_count`, `geo_enabled`, et `default_profile`, nous avons appliqué une standardisation à l'aide du `StandardScaler`. Cette étape aligne les variables sur une même échelle, ce qui est souvent nécessaire pour optimiser les performances de certains algorithmes de machine learning.

3.7 Conclusion

Au cours de ce chapitre, notre attention s'est portée sur la visualisation des diverses variables présentes dans notre ensemble de données, ainsi que sur leur nettoyage, deux étapes préliminaires essentielles à la mise en œuvre de la plupart des algorithmes d'apprentissage automatique.

Réalisation et résultats

4.1 Introduction

Ce chapitre se concentre sur la mise en œuvre de différents modèles de machine learning pour la classification des données et l'évaluation de leurs performances. L'objectif de cette partie est de décrire le processus d'implémentation, en abordant la sélection des modèles, leur entraînement, et l'évaluation des performances. Ce chapitre comprend l'explication de chaque modèle choisi, ainsi que les étapes de configuration, d'entraînement, de prédiction et de comparaison des performances.

4.2 Choix des modèles de classification

Nous avons sélectionné plusieurs modèles, chacun apportant des avantages spécifiques à la classification. Les modèles utilisés incluent :

4.2.1 Random Forest

Random Forest est un algorithme d'ensemble qui utilise une combinaison de nombreux arbres de décision pour effectuer des prédictions. Chaque arbre est entraîné sur un sous-ensemble aléatoire des données et des caractéristiques, ce qui améliore la diversité des arbres et réduit le risque de surapprentissage (overfitting). La décision finale est prise par un vote majoritaire des arbres (classification) ou par la moyenne des prédictions (régression).

4.2.2 Gradient Boosting

Gradient Boosting est un modèle d'ensemble itératif qui construit des arbres de décision faibles en série, chaque nouvel arbre cherchant à corriger les erreurs des arbres précédents. Ce processus d'ajustement progressif produit un modèle puissant qui minimise les erreurs de classification en pondérant davantage les erreurs passées.

4.2.3 Régression Logistique

La régression logistique est un modèle linéaire largement utilisé pour la classification binaire. Il estime les probabilités d'appartenance à chaque classe en utilisant une fonction logistique appliquée à une combinaison linéaire des caractéristiques. Le modèle est

interprétable et rapide à entraîner, ce qui le rend adapté pour obtenir rapidement une première évaluation de la séparation des classes.

4.2.4 K-Nearest Neighbors (K-NN)

Le modèle K-NN classe un profil en fonction de la proximité de ses voisins dans l'espace des caractéristiques. Pour chaque nouvelle instance, K-NN cherche les K voisins les plus proches et assigne la classe majoritaire parmi eux. Cet algorithme est non paramétrique et est efficace pour des jeux de données simples.

4.2.5 Support Vector Machine (SVM)

Le SVM est un modèle qui crée une frontière de décision optimale entre les classes en maximisant la marge entre les points de données les plus proches des classes opposées, appelés vecteurs de support. Il peut utiliser différentes fonctions de noyau pour transformer les données et gérer les problèmes non linéaires, comme le noyau gaussien (RBF).

4.2.6 Arbre de Décision

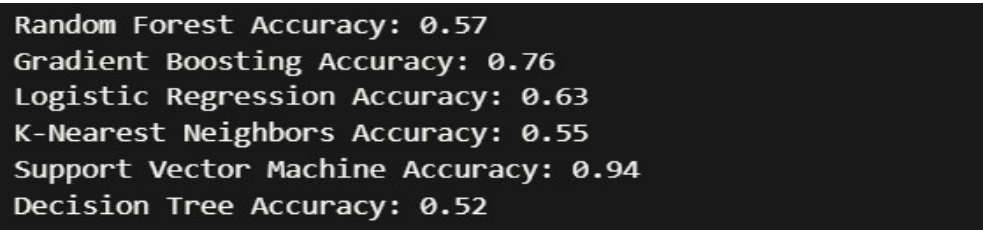
Un arbre de décision divise les données en sous-groupes en fonction de règles basées sur les caractéristiques. Chaque division est choisie pour maximiser la séparation des classes, mesurée par des critères tels que l'entropie ou le Gini. Ce modèle est interprétable et visuellement compréhensible, ce qui permet de suivre le processus de décision du modèle.

4.3 Accuracy

Dans un contexte de classification, **Accuracy** représente la proportion de prédictions correctes (les vrais positifs et les vrais négatifs) par rapport au nombre total de prédictions effectuées. Mathématiquement, elle est définie comme suit :

$$\text{Exactitude} = \frac{\text{Nombre de Prédictions Correctes}}{\text{Nombre Total de Prédictions}} \quad (4.1)$$

En d'autres termes, elle mesure la fréquence à laquelle le classificateur fait des prédictions correctes.



```
Random Forest Accuracy: 0.57
Gradient Boosting Accuracy: 0.76
Logistic Regression Accuracy: 0.63
K-Nearest Neighbors Accuracy: 0.55
Support Vector Machine Accuracy: 0.94
Decision Tree Accuracy: 0.52
```

FIGURE 4.1 – Evaluation des modèles

La visualisation indique que le modèle qui atteint la performance maximale est le SVM

4.4 Évaluation des Modèles de Classification

Dans le cadre de la détection de faux profils sur Twitter, différents modèles de classification supervisée ont été testés pour évaluer leur capacité à distinguer les comptes "fake" des comptes réels. Les résultats montrent que certains modèles, comme la **Machine à Vecteurs de Support (SVM)** et le **Gradient Boosting**, obtiennent des performances supérieures par rapport à d'autres modèles tels que les **arbres de décision** et les **forêts aléatoires**.

4.4.1 Analyse des performances :

- **Machine à Vecteurs de Support (SVM)** : Ce modèle obtient une précision élevée (0.94), ce qui peut être attribué à sa capacité à maximiser la marge entre les classes dans un espace de haute dimension. Dans le contexte de la détection de faux profils, SVM est efficace pour gérer des données textuelles et comportementales qui contiennent de nombreuses caractéristiques, en particulier lorsqu'il existe des différences subtiles entre les classes.
- **Gradient Boosting** : Ce modèle présente également une bonne performance (0.76). Le Gradient Boosting est un ensemble de modèles faibles (arbres de décision) qui corrige les erreurs de prédiction des précédents. Cette approche est particulièrement utile pour des données déséquilibrées ou contenant des caractéristiques non linéaires, comme c'est souvent le cas avec les profils Twitter.
- **Forêts Aléatoires et Arbres de Décision** : Ces modèles ont montré des performances plus faibles (0.57 et 0.52 respectivement). Les forêts aléatoires sont souvent performantes sur des données avec une structure forte ou une variabilité importante, mais elles peuvent être moins efficaces pour capturer des motifs fins dans les données textuelles, comme les subtilités linguistiques ou comportementales associées aux faux profils.
- **Régression Logistique et K-Nearest Neighbors (KNN)** : La régression logistique obtient une précision modérée (0.63) car elle est adaptée aux relations linéaires, mais elle peut manquer de flexibilité pour capter les relations complexes présentes dans les données. Le modèle KNN, qui se base sur la proximité des données, atteint également une précision modérée (0.55), mais il peut être sensible à la dimensionnalité élevée des données textuelles, ce qui peut expliquer sa performance inférieure.

En conclusion, les résultats démontrent que les modèles capables de capturer des relations non linéaires et de maximiser les marges entre les classes, comme SVM et Gradient Boosting, sont mieux adaptés à la détection de faux profils sur Twitter, où les différences entre classes peuvent être subtiles et non linéaires. Ces modèles sont donc privilégiés dans le contexte de ce projet.

4.4.2 Optimisation des hyperparamètres

- **Recherche d'hyperparamètres** : Utilisation de `GridSearchCV` pour tester plusieurs combinaisons d'hyperparamètres sur différents modèles et trouver les meilleures configurations pour chaque algorithme.
- **Validation croisée** : Utilisation de `StratifiedKFold` pour évaluer les modèles de manière robuste en maintenant l'équilibre des classes à chaque pli.

- **Normalisation des données** : Application de `StandardScaler` pour standardiser certaines variables numériques avant d'entraîner des modèles sensibles à l'échelle des données.
- **Pipeline de traitement** : Construction de pipelines pour intégrer facilement les étapes de prétraitement et le modèle, permettant d'appliquer les transformations de manière cohérente à chaque itération.
- **Comparaison des modèles** : Enregistrement et comparaison des meilleurs scores et paramètres pour chaque modèle afin d'identifier l'algorithme le plus performant.

4.4.3 Application sur nos modèles

```
Summary of Best Parameters and Scores:  
Random Forest: Best Params = {'classifier_max_depth': None, 'classifier_min_samples_split': 5, 'classifier_n_estimators': 100}, Best Score = 0.5252136752136752  
Gradient Boosting: Best Params = {'classifier_learning_rate': 0.2, 'classifier_max_depth': 7, 'classifier_n_estimators': 300}, Best Score = 0.5076923076923077  
Logistic Regression: Best Params = {'classifier_C': 100, 'classifier_solver': 'liblinear'}, Best Score = 0.5183760683760683  
K-Nearest Neighbors: Best Params = {'classifier_metric': 'euclidean', 'classifier_n_neighbors': 5, 'classifier_weights': 'distance'}, Best Score = 0.5264957264957265  
Support Vector Machine: Best Params = {'classifier_C': 10, 'classifier_gamma': 'scale', 'classifier_kernel': 'linear'}, Best Score = 0.5188034188034187  
Decision Tree: Best Params = {'classifier_criterion': 'gini', 'classifier_max_depth': None, 'classifier_min_samples_split': 2}, Best Score = 0.5222222222222223
```

FIGURE 4.2 – Résultat du hyperparameters tuning

4.5 Conclusion

Au cours de ce chapitre, nous nous sommes concentrés sur la formation et l'évaluation de différents modèles de prédiction, avec la sélection du meilleur modèle qui s'est portée sur le SVM.

Conclusion générale

Ce projet a exploré l'utilisation de l'apprentissage automatique et du traitement du langage naturel (NLP) pour détecter les faux comptes sur les réseaux sociaux en se basant sur les caractéristiques des profils. En analysant des données variées, allant des métriques de comptes aux descriptions textuelles, le modèle développé a démontré une bonne capacité à distinguer les utilisateurs légitimes des faux utilisateurs. L'approche adoptée, intégrant des techniques de prétraitement de texte et des modèles de classification, s'est révélée efficace pour répondre au problème croissant de la prolifération de faux comptes, avec des performances prometteuses pour une utilisation à grande échelle.

Les résultats obtenus montrent que les modèles de machine learning peuvent significativement contribuer à améliorer la précision de la détection des faux utilisateurs. En utilisant des informations diverses, telles que le nombre de followers, la personnalisation du profil et le contenu textuel, notre modèle permet d'identifier des tendances et des comportements distinctifs propres aux faux comptes. Ce système adaptable offre une solution viable pour améliorer la sécurité et la fiabilité des plateformes en ligne.

Les perspectives d'amélioration incluent l'intégration de données supplémentaires pour affiner les résultats et l'exploration d'autres techniques de traitement de texte et de deep learning pour des analyses plus avancées. En somme, ce projet montre que les technologies de l'IA et du NLP peuvent contribuer de manière efficace à relever les défis posés par les faux comptes dans le monde numérique, offrant ainsi des solutions évolutives pour le maintien de l'intégrité des plateformes en ligne.

Webographie

Dpcumentation scikit-learn : <https://scikit-learn.org/stable/index.html>

Documentation Plotly : <https://plotly.com/python/>

Documentation sur NLTK : <https://www.nltk.org/>

Github : <https://github.com/>

Fake News Detection using Machine Learning models : <https://prabhitha3.medium.com/fake-news-detection-using-machine-learning-models-4475f62c0836>