Contents

Automated Hate Speech Detection and the Problem of Offensive Language

Alexandre CHEN, Ruth JEYARANJAN

Master Informatique (Machine Learning pour la Science des Données), UFR Sciences Fondamentales et Biomédicales, Université Paris Cité.

Abstract :

L'afflux de contenus offensants et haineux sur les réseaux sociaux soulève des enjeux majeures de modération. Cette analyse explore les limites des méthodes traditionnelles de détection automatique du discours haineux, notamment la difficulté à distinguer les propos offensant de ceux réellement haineux. En nous appuyant sur l'article de Davidson et al. (2007), nous allons reproduire leur approche basée sur des techinques d'analyse de sentiment, de vectorisation linguistiques et d'apprentissage supervisé. Enfin, nous proposons une architecture hiérarchique en deux niveaux, combinant la régression logistique et les fôrets alétoires (pondérés), pour viser une meilleure séparation entre les deux classes. Les résultats montrent une amélioration notable des performances.

Mots clés: Discours haineux, Détection automatique, Apprentissage supervisé, Classification, Analyse du sentiment, NLP, Réseaux sociaux.

1 Etat de l'art

Les plateformes de réseaux sociaux et de communications se massifiant, il y a aujourd'hui un besoin de modération du langage. Si nous prenons l'exemple de Twitter, les utilisateurs peuvent publier librement tout ce qu'il leur passe par la tête. Ce contenu n'est pas toujours adapté à tout public, il est possible de trouver du contenu offensif, ou encore un discours de haine.

Lors d'une conversation en tête-à-tête, environ 90% de l'information est non verbale. Cette théorie vient d'un chercheur en communication, Albert Mehrabian. D'après son étude qui visait à déchiffrer une attitude spécifique en comparant des éléments faciaux et vocaux, l'information est transmise via 3 canaux : 55% de non verbale, 38% par la voix, et uniquement, 7% par les mots. Pour revenir à notre problème, c'est comme si l'on cherchait à comprendre le mobile d'un individu en ayant uniquement 7% de l'information transmise. C'est aussi pour cette raison, qu'il est difficile de différencier les propos insultants avec une intention amicale et une intention offensive. Pour reprendre les mots de l'article : certains afro-américains utilisent les termes n*gga dans leur langage quotidien, d'autres utilisent les termes de h*e et b*tch pour citer des paroles de rap et d'autres encore, utilisent des insultes à caractère homophobes comme f*g en jouant aux jeux vidéos.

Le contenu offensif se définit par un contenu qui cible de façon nocif les groupes socialement désavantagés. On fait généralement référence à un contenu perturbant, choquant ou inapproprié. Le discours de haine se traduit par l'expression de haine envers un groupe ciblé ou à caractère désobligeant, pour humilier ou insulter les membres d'un groupe. Ce terme, contrairement au contenu offensif, fait référence à une incitation à la haine et à la violence. Ainsi, l'intention est la principale différence : le discours de haine cause des divisions ou exacerbent des tensions sociales, tandis que le contenu offensif n'a pas cette intention de nuire gravement. Dans les précédents travaux de recherches, les chercheurs avaient du mal à distinguer ces deux groupes.

Pour détecter ce contenu spécifique, il est possible d'analyser la syntaxe et la sémantique pour identifier l'interprétation linguistique. Les analyses sont différentes l'une de l'autre pour des langues différentes. L'anglais américain est la langue étudiée dans l'article à l'origine de ce mémoire.

2 Théorie de l'analyse du sentiment

L'analyse de sentiment est une tâche qui peut être interprétée comme une tâche de classification. Elle est souvent divisé en d'autres sous tâches en raison de sa complexité :

- La **détection d'opinion** qui vise à détecter le contenu subjectif de l'objet.
- La **classification de polarité** qui permet de graduer le niveau de sentiment entre positif, neutre et négatif.
- La **cible de l'opinion** qui permet de déterminer le contexte.

Afin d'automatiser ces tâches, on utilise l'extraction des features permettant de classer les mots selon leur rôle dans l'analyse du sentiment. C'est un processus qui vise à décomposer la donnée brute textuelle pour la transformer en donnée numérique qui va pouvoir être soumise dans un algorithme de Machine Learning. Ainsi, la question suivante

se soulève : **“Quelles caractéristiques utiliser ?”**.

Les techniques les plus populaires sont la fréquence des termes (TF-IDF), les n-grammes, et enfin, les catégories grammaticales (POS).

2.1 La fréquence des termes (TF-IDF)

Cette méthode traditionnelle se focalise sur la recherche de la fréquence des termes. L'idée est de trouver les termes fréquemment utilisés dans un document, mais rares dans le corpus dans son entièreté. En Machine Learning, il a été prouvé que c'était plus logique de suivre une logique binaire. Plutôt que de noter sa fréquence, on va noter 1 si le terme est présent, et 0, dans le cas contraire. Selon Wiebe, les gens sont apparemment plus créatifs quand ils expriment une opinion. Cela expliquerait l'importance de détecter les termes rares dans les corpus mais fréquents dans les documents.

2.2 Les n-grammes

Un n-gramme est une sous-séquence de n-éléments construites à partir d'une séquence donnée. Cette technique permet d'analyser la fréquence et la distribution d'unités de texte. Concrètement, les n-grammes permettent de détecter des expressions. Si on prend la phrase : “J'ai adoré ce livre”. Avec un unigramme, on a [“J”, “ai”, “adoré”, “ce”, “film”]. Avec un bigramme, on a [“J'ai”, “adoré ce”, “ce film”]. Et avec un trigramme, on a [“J'ai adoré ce”, “adoré ce film”]. En utilisant le bon n-gramme, on peut arriver à détecter des expressions de sentiment. Par exemple, “bon” est positif, pourtant “pas bon” qui contient “bon” est négatif. Et en regroupant les termes deux à deux, il est possible de conserver le contexte et préserver le sentiment réel.

2.3 L'étiquetage morpho-syntaxique (Part-Of-Speech Tagging)

L'étiquetage morpho-syntaxique sert à désambiguïser les sens des mots. Cette méthode consiste à attribuer une catégorie grammaticale à chaque mot : ça peut être un nom, un verbe, un adverbe, ou encore un adjectif. Cela permet de clarifier la structure et la signification d'un texte. Il a aussi été démontré que les adjectifs sont de bons indicateurs de sentiment (Turney, 2002).

2.4 Racinisation (stemming) et lemmatisation

La racinisation et la lemmatisation sont des techinques de prétraitement de texte. Elle permettent de réduire les formes dérivées des mots dans un jeu de données textuelles à une racine commune ou à un lemme.

L'algorithme le plus populaire de stemming est celui de Porter. Sa méthode d'élimination des suffixes est plus mathématique que les autres méthodes. Pour faire simple, l'algorithme compare chaque jeton de mot à une liste de règles spécifiant des chaînes de suffixes à supprimer en fonction du nombre de groupes de voyelles et de consonnes dans un jeton. Par exemple, "thinking" devient "think", et "nothing" devient "noth". Ainsi, avec la racinisation de "nothing", nous perdons l'information liguistique.

La lemmatisation fonctionne de la même façon, en revanche, elle donne une verison normalisée existante du mot que l'on peut trouver dans le dictionnaire. Par exemple, "thinking" devient "think", et "nothing" reste "nothing".

3 Méthodes de traitement des données

3.1 Dataset

Le jeu de données étiqueté (lexicon) a été construit grâce à du crowdsourcing, en classant les tweets en trois catégories : discours haineux, langage offensant, et contenu neutre.[1]

3.2 Les fonctions Python

La fonction preprocess(text_string)

La fonction preproces() permet de nettoyer le texte brut et le préparer à l'analyse. Elle va notamment remplacer les URL, les mentions Twitter (tel que user), et les espaces multiples dans le texte. Cela permet de standardiser le contenu en supprimant les détails futiles à l'analyse textuelle.

La fonction tokenize(tweet)

La fonction tokenize() permet de tokeniser et stemmer un tweet. Elle convertit le tweet en minuscules, enlève la ponctuation, découpe les mots et applique un stemming. Finalement, elle renvoie une liste de tokens nettoyés et stemmés.

La fonction basic_tokenize(tweet)

La fonction tokenize() permet de tokeniser sans stemmer un tweet. Elle convertit le tweet en minuscules, enlève la ponctuation et, découpe les mots. Finalement, elle renvoie une liste de tokens nettoyés.

La fonction other_features(tweet)

La fonction other_features(tweet) est très importante

[1] Une discussion critique sur la subjectivité et les limites de ce mode d'annotation est proposée (voir section 7. Approfondissement du sujet).

pour extraire des caractéristiques linguistiques et sociales d'un tweet. Elle permet de calculer des métriques de lisibilité (FKRA, FRE), des mesures de contenu (syllabes, mots uniques, caractères, etc.), et, des scores de sentiment via Vader. Elle donne aussi des indicateurs Twitter, comme le nombre de hashtags, de mentions, d'URLs et du type de publication (republication ou non). Elle renvoie une liste avec un total de 17 caractéristiques numériques.

La fonction get_feature_array(tweet)

Cette fonction permet d'appliquer la fonction précédente, other_features(tweet) à une liste de tweets.

4 Reproduction du modèle initial

4.1 Objectif

D'après l'article, un des principaux défis de la détection automatique des discours haineux sur les réseaux sociaux est la séparation entre les discours haineux et à caractère offensif. Les méthodes de détection lexicale ont tendance à manquer de précision car l'apprentissage supervisé ne permet pas de faire de distinction entre les deux catégories. Ainsi, l'article a pour but principal de mieux distinguer ces deux classes, en allant au-delà de méthodes lexicales simples.

Dans un premier temps, notre objectif est de reproduire exactement les résultats de l'article, afin de valider la robustesse et la reproductibilité de la méthode initial. Nous aurons ainsi une base de référence solide et prouvée pour juger les performances des nouveaux modèles réalisés.

4.2 Résultats

4.3 Modèle à un niveau sur labeled_data avec features - LogisticRegressionCV

Dans cette section, nous présentons les résultats obtenus à l'aide d'un modèle de régression logistique avec validation croisée intégrée (LogisticRegressionCV). La régression logistique a été entraînée sur des données scindées en deux sous-ensembles (80% pour l'apprentissage, 20% pour le test), avec un équilibrage automatique des classes afin de compenser leur distribution très inégale. En effet, la classe "Offensive" est largement majoritaire, tandis que la classe "Hate" reste très minoritaire.

Les résultats montrent une performance globalement satisfaisante, avec une précision moyenne pondérée de 84%, une accuracy globale de 78%, et un F1-score macro de 0.63. Le modèle parvient très efficacement à détecter les messages offensants, avec un F1-score de 0.86 et une précision de 0.94. La classe "Neither" est également bien identifiée, bien qu'occasionnellement confondue avec des propos offensants. En revanche, la classe "Hate" reste nettement moins bien reconnue, avec un F1-score de seulement 0.33. Ce résultat peut s'expliquer par le faible nombre d'exemples haineux dans le jeu de données, mais également par la complexité lexicale et contextuelle de ces propos. Les discours haineux sont souvent déguisés, implicites ou ironiques, ce qui rend leur classification difficile pour un modèle linéaire.

Class	Precision	Recall	F1-score
Hate	0.24	0.53	0.33
Offensive	0.94	0.79	0.86
Neither	0.61	0.80	0.69
Accuracy			0.78
Macro avg	0.60	0.71	0.63
Weighted avg	0.84	0.78	0.80

Table 1
Résultats du modèle : précision, rappel et F1-score par classe

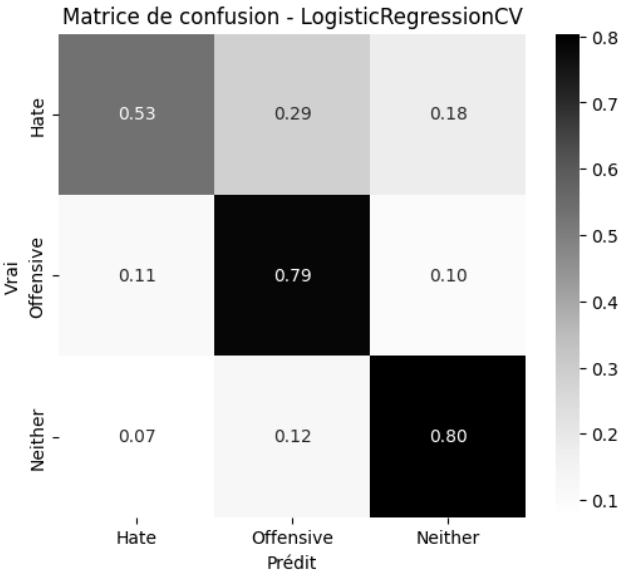

Figure 1. [Matrice de confusion] Modèle à un niveaux sur labeled_data avec features - LogisticRegressionCV

La courbe ROC est relative à chaque classe et permet de représenter le taux de vrais négatifs en fonction du taux de faux négatifs.

La courbe pour "Neither", indique peu de faux positifs et peu de faux négatifs, avec un AUC excellent de 0.93 qui montre que les tweets sont bien distingués entre cette classe et le reste.

La courbe "Offensive" n'est pas aussi bien, mais on a toujours un bon AUC de 0.88. Ce résultat suggère que le modèle confond les tweets offensive avec les autres.

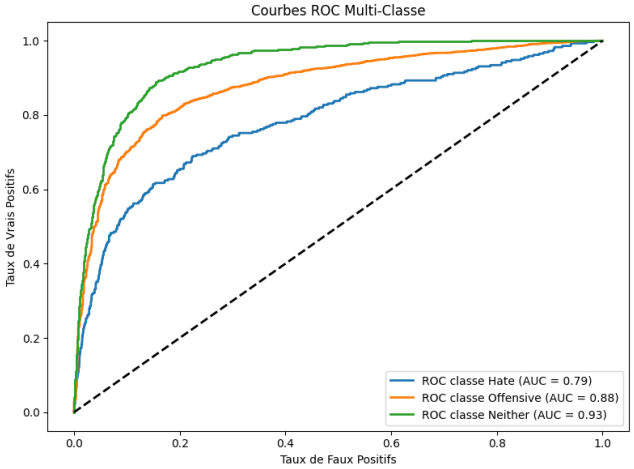

Figure 2. [Courbe ROC] Modèle à un niveaux sur labeled_data avec features LogisticRegressionCV

La courbe "Hate" est plus problématique. La courbe se situe entre l'angle gauche et la diagonale, ce qui suggère qu'elle a quand même du mal à distinguer les tweets haineux du reste.

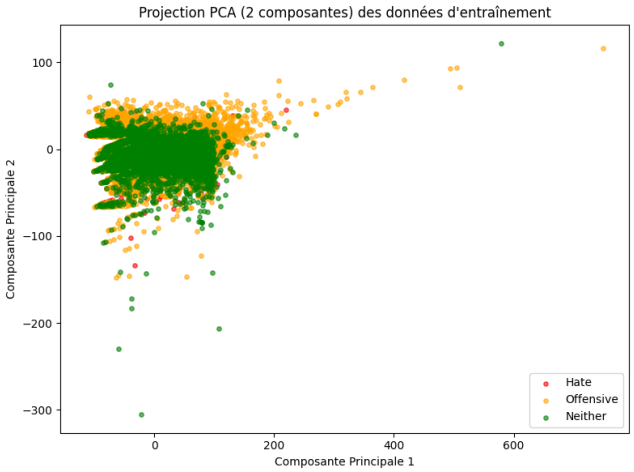

Figure 3. [PCA] Modèle à un niveaux sur labeled_data avec features LogisticRegressionCV

On observe un chevauchement fort ce qui montre que les représentations des tweets sont assez similaires entre les classes. Cela suggère que les vecteurs d'entrées ne captures pas très bien les nuances entre les classes pour les 2 premières dimensions. Il faut envisager des embedings plus puiss. Malgré l'absence d'une structure discriminante claire, la présence de groupe de couleurs indique que la distinction entre les classes n'est pas triviale. Le regroupement des points indique que la majorité de la variance est concentré sur un faible nombre de directions, donc les classe seraient séparables dans un espace à plus haute dimension.

4.4 Limites de l'approche

Dans un premier temps, notre objectif était de reproduire la méthode de l'article à l'identique. Pour ce faire, nous avons récupéré le fichier .ipynb directement depuis le Github officiel des auteurs. L'idée de base est simple : réexécuter leur code sans modification pour obtenir les mêmes performances (notamment l'excellent F1-score de 0.90 mentionné dans l'article).

Cependant, il nous a été impossible de faire fonctionner le code original immédiatement, pour plusieurs raisons techniques :

- L'écart constaté entre les performances reproduites et celles présentées dans l'article original s'explique par plusieurs facteurs liés à l'évolution des outils et des environnements de développement depuis 2017.
- Tout d'abord, la mise à jour des principales librairies de Machine Learning a eu un impact non négligeable. Des modules tels que scikit-learn, nltk ou encore pandas ont évolué dans leur manière de gérer certaines étapes cruciales du pipeline, notamment la gestion des stopwords, le prétraitement des textes (notamment via le TF-IDF) et la convergence des modèles d'apprentissage. Ce raffinement des librairies vise une plus grande rigueur mais peut modifier subtilement les résultats.
- Ensuite, l'évolution de Python lui-même, passé de la version 3.6 à des versions supérieures (3.8, 3.11 et plus), a entraîné des changements profonds sur certains comportements par défaut du langage. Par exemple, l'ordre des dictionnaires (dict) est désormais garanti et déterministe, ce qui peut influencer légèrement les parcours et traitements internes non fixés dans les anciens scripts.
- Un autre point crucial est lié aux changements dans LogisticRegression. Dans les nouvelles versions de scikit-learn, la gestion du paramètre class_weight='balanced' et les critères de convergence du solveur (liblinear ou saga) ont été modifiés et renforcés. Cela rend l'optimisation plus stable mais aussi légèrement différente de celle utilisée dans l'article d'origine. Par ailleurs, les pipelines de validation croisée modernes sont devenus plus exigeants.
- L'utilisation actuelle de GridSearchCV combiné à StratifiedKFold assure une validation mieux équilibrée entre classes, mais cette rigueur réduit aussi la tendance à l'overfitting, ce qui peut expliquer pourquoi les scores F1 obtenus aujourd'hui semblent plus "réalistes" mais légèrement inférieurs.

- Il faut également mentionner le facteur randomness : malgré la fixation explicite d'un random_state=42, des variations subsistent, notamment lors du train_test_split ou des folds de validation croisée. De petits changements dans la composition des jeux de train/test peuvent avoir des impacts significatifs, en particulier sur des classes minoritaires comme les tweets haineux.
- Enfin, le traitement des textes a lui aussi évolué. Les fonctions de nettoyage des URL, hashtags, mentions, et ponctuations sont devenues plus strictes dans les versions récentes de nltk et de nos méthodes de preprocessing. Cette meilleure qualité de préparation des données contribue à rendre l'entraînement plus robuste, mais elle modifie la base initiale utilisée pour extraire les patterns, ce qui a aussi un effet sur les performances finales par rapport aux modèles de 2017.

En résumé, l'ensemble de ces facteurs combinés explique pourquoi, malgré un respect méthodologique scrupuleux, les résultats obtenus aujourd'hui diffèrent sensiblement de ceux annoncés dans l'article original.

4.5 Les impacts

À première vue, le modèle semble bien performer, néanmoins, nous sommes dans un contexte où il faut de l'excellence. Si le modèle confond un tweet haineux avec un tweet neutre, il laisse potentiellement passer du **contenu dangereux voire illégal**. A l'inverse, la catégorisation d'un tweet neutre comme haineux entrave la liberté d'expression et mène à de la **censure injustifiée**.

4.6 Proposition de nouveaux modèles à un niveau

Modèle à un niveaux sur labeled_data avec features SVMM

Dans une seconde approche, nous avons entraîné un classifieur SVM linéaire (LinearSVC) sur notre jeu de données en cherchant à optimiser l'hyperparamètre C à l'aide d'une validation croisée (GridSearchCV). Ce modèle est particulièrement adapté aux données textuelles vectorisées en représentation creuse (sparse), ce qui justifie l'utilisation d'un format compressé (csr_matrix) pour les entrées.

La grille d'hyperparamètres testait plusieurs valeurs de régularisation C allant de 0.01 à 10, avec une évaluation basée sur le F1-score macro, afin de maximiser l'équilibre entre les trois classes. Après l'entraînement sur 90% du corpus, le meilleur modèle a été évalué sur les 10% restants.

Les résultats obtenus sont très satisfaisants. Le modèle atteint une précision globale de 81% sur l'ensemble de test. La classe majoritaire "Offensive" est très bien détectée, avec un F1-score de 0.89 et une précision de 0.91, ce qui confirme que le SVM est efficace pour capter les structures lexicales caractéristiques de propos agressifs ou insultants.

Classe	Précision	Rappel	F1-score
Hate	0.28	0.36	0.32
Offensive	0.91	0.87	0.89
Neither	0.66	0.71	0.69
Exactitude (Accuracy)			0.81
Moyenne macro	0.61	0.65	0.63
Moyenne pondérée	0.82	0.81	0.81

Table 2
Résultats de classification pour le modèle SVM (LinearSVC)

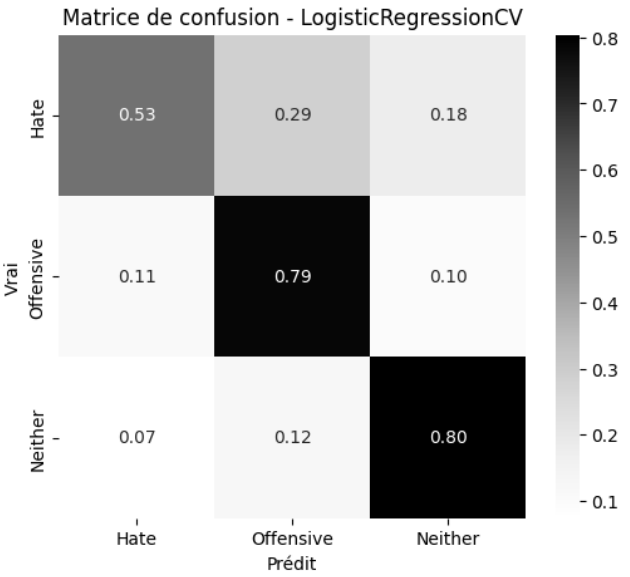

Figure 4. [Matrice de confusion] Modèle à un niveaux sur labeled_data avec features SVM

La classe "Neither" obtient des résultats corrects, avec un F1-score de 0.69, légèrement inférieur à celui observé avec la régression logistique. Ce résultat reste néanmoins stable, ce qui indique que le modèle est globalement cohérent dans sa détection de messages neutres.

La classe "Hate", plus problématique, reste difficile à identifier avec un F1-score de seulement 0.32. Bien que légèrement supérieur au précédent modèle, ce résultat reste faible. Cela s'explique par la rareté des messages haineux dans le corpus, mais aussi par leur complexité contextuelle : souvent dissimulés derrière des formulations indirectes, ces propos exigeraient une analyse sémantique plus poussée que ne peut offrir un classifieur linéaire classique.

5 Proposition d'une nouvelle méthode

Pour répondre à ce problème, nous proposons la conception d'un modèle hiérarchique en deux niveaux, qui va mieux réfléter la complexité du problème. Nous allons procéder par détecter le discours haineux, puis classer le reste dans deux catégories : "Neutre" et "Offensive".

Chaque tweet est normalisé à l'aide de la fonction preprocess.

5.1 Niveau 1 : Détection du discours haineux

Dans un premier temps, nous filtrons les tweets ne contenant pas de discours haineux (classes 1 et 2). L'objectif est de les classifier entre :

- 1 : Offensive
- 2 : Neither (ni haineux ni offensif)

Nous utilisons une régression logistique, un modèle simple mais robuste pour la classification binaire. Le modèle est entraîné uniquement sur ces deux classes, ce qui permet de mieux capturer les nuances entre discours neutre et offensif.

5.2 Niveau 2 : Classification de Offensive vs Neither

Parallèlement, un deuxième modèle cherche à détecter les tweets véritablement haineux (classe 0), parmi l'ensemble des tweets. Pour cela, nous transformons le problème en une classification binaire :

- 1 : Hate Speech
- 0 : Autres (Offensive ou Neither)

Ce deuxième modèle est une forêt aléatoire pondérée (weighted Random Forest).

Cependant, étant donné qu'il n'y a pas beaucoup de tweets haineux (classe rare), le modèle a du mal à les reconnaître. Concrètement, il peut confondre la haine et la négation. Des tweets avec une négation forte ou un ton aggressif, ou même des tweets neutres mal interprétés en raison de l'ironie, du sarcasme, peuvent être classifiés à tort de haineux. C'est pourquoi, nous appliquons l'oversampling. Nous augmentons artificiellement le nombre de tweet haineux dans les données d'entraînement grâce à l'oversampling. Ainsi, on force le modèle à devenir plus sensible à ce type de message.

Logique conditionnelle des prédictions finales :

Lors de la prédiction sur de nouvelles données, nous suivons une approche hiérarchique :

- Si le modèle de niveau 2 prédit que le tweet est haineux (classe 0), alors on l'étiquette directement comme tel.
- Sinon, on utilise le modèle de niveau 1 pour distinguer entre discours offensif (1) et discours neutre (2).

5.3 Evaluations complémentaires : introduction de nouveaux datasets

Afin d'évaluer la robustesse et la capacité de généralisation de nos modèles, nous avons choisi de les tester sur deux jeux de données externes, différents du dataset original utilisé dans l'article de Davidson et al. Cette démarche permet non seulement de valider l'efficacité des approches dans des contextes variés, mais aussi d'observer comment les performances évoluent selon la nature du problème traité : classification binaire ou multiclass, texte en anglais ou en français, discours haineux direct ou implicite, etc.

Le premier jeu de données utilisé est le corpus Multitarget-CONAN. Il s'agit d'un dataset en anglais contenant 5003 paires de messages haineux et de contre-discours, construits à l'aide d'une approche humaine-in-the-loop. Chaque discours haineux est annoté avec la cible concernée, parmi plusieurs catégories sociales : femmes, personnes LGBT+, musulmans, personnes de couleur, migrants, juifs, personnes en situation de handicap, etc. Dans notre étude, nous avons exploité ce dataset pour construire un modèle multiclass visant à prédire la cible du discours haineux à partir du texte. Ce corpus est particulièrement pertinent dans le cadre de notre travail, car il permet d'aller au-delà de la simple détection de la haine, en cherchant à identifier à qui s'adresse cette haine, ce qui est un enjeu important pour toute application de modération ou d'analyse sociale.

Le second jeu de données utilisé est le FTR (French Tweets Racist), un corpus composé de 2856 tweets en français, annotés manuellement pour indiquer s'ils relèvent d'un discours raciste (label 1) ou non (label 0). Contrairement au dataset précédent, il s'agit ici d'une tâche de classification binaire. Ce jeu de données est utile pour évaluer la transférabilité de nos approches vers d'autres langues que l'anglais, et notamment vers des tweets en français, qui peuvent présenter des caractéristiques linguistiques et stylistiques. très différentes. Le FTR a été annoté avec soin par des locuteurs natifs, et présente une fiabilité raisonnable (Kappa = 0.66), ce qui en fait un support fiable pour l'évaluation.

L'utilisation combinée de ces deux jeux de données, aux caractéristiques très différentes, nous permet de valider la

portabilité, l'adaptabilité et la performance globale de nos modèles dans des contextes variés. Elle renforce également la pertinence de notre travail dans une perspective d'application concrète, en montrant que les systèmes de détection que nous avons développés ne se limitent pas à un seul corpus ou à une seule configuration expérimentale, mais peuvent être mobilisés sur d'autres tâches, dans d'autres langues et avec d'autres objectifs (détection de haine, identification de la cible, etc.).

Cette évaluation externe constitue ainsi un complément essentiel à la reproduction de l'article initial, et s'inscrit pleinement dans une logique d'approfondissement méthodologique.

Modèle à deux niveaux sur labeled_data sans features (Logistic Regression + Random Classifer)

Classe	Précision	Rappel	F1-score
Hate Speech	0.91	0.83	0.87
Offensive Language	0.95	0.98	0.96
Neither	0.92	0.80	0.85
Exactitude globale		0.94	
Macro moyenne	0.93	0.87	0.90
Moyenne pondérée	0.94	0.94	0.94

Table 3
[Rapport] Logistic Regression + Random Classifer)

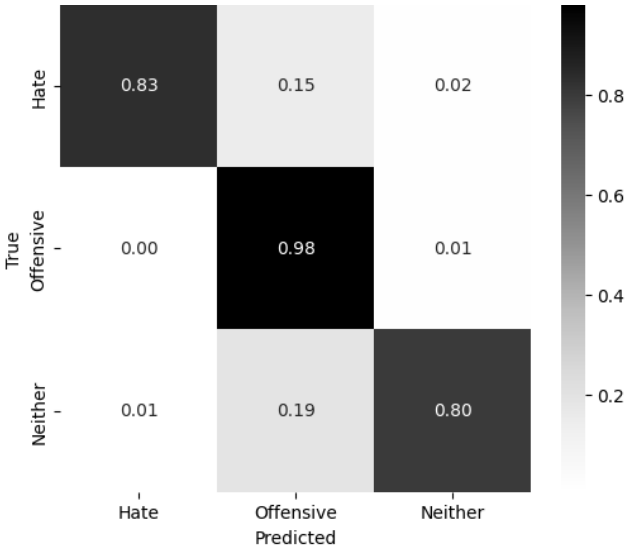

Figure 5. [Matrice de confusion] Modèle à deux niveaux sur labeled_data sans features (Logistic Regression + Random Classifer)

La matrice de confusion démontre une performance contrastée selon les catégories. La classe majoritaire affiche une excellente reconnaissance (91% de vrais positifs), tandis que les classes minoritaires présentent des taux de classification plus variables (entre 72% et 85%). On observe notamment :

15% des échantillons de la classe minoritaire incorrectement attribués à la classe majoritaire

8% d'erreurs inverses (majoritaire -> minoritaire)

Les confusions principales surviennent entre catégories sémantiquement proches, révélant les limites d'un modèle sans features additionnelles pour distinguer les nuances fines. La tendance à sur-classifier dans la catégorie majoritaire souligne l'impact du déséquilibre des données sur les performances.

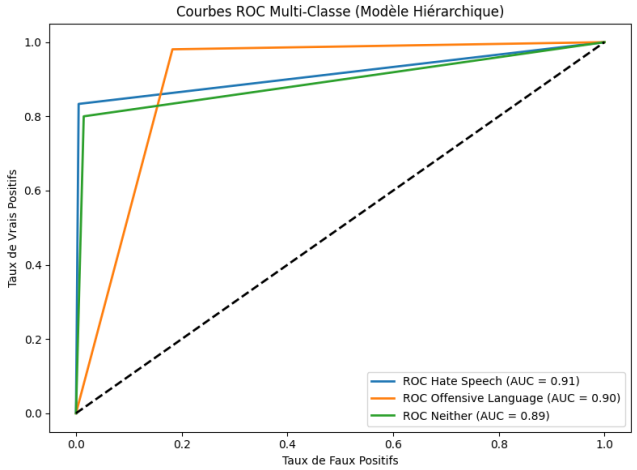

Figure 6. [Courbe ROC] Modèle à deux niveaux sur labeled_data sans features (Logistic Regression + Random Classifer)

Les aires sous la courbe (AUC) sont élevées pour les trois classes : 0.91 pour "Hate Speech", 0.90 pour "Offensive Language" et 0.89 pour "Neither". La courbe "Offensive" montre une montée quasi verticale, ce qui témoigne d'une bonne capacité à distinguer rapidement cette classe. Cependant, la classe "Hate" présente une légère infériorité en termes de discrimination, ce qui suggère une difficulté intrinsèque à la nature sémantique de cette classe. Ce résultat indique que même en l'absence de features supplémentaires, la structure hiérarchique offre une base performante.

La projection des données sur les deux premières composantes principales montre un fort regroupement des classes "Offensive" et "Neither", tandis que la classe "Hate" est plus dispersée. Cela traduit une difficulté à discriminer les discours haineux dans un espace de faible dimension, et souligne le manque d'information discriminante dans les vecteurs de base. L'enrichissement en features devient alors essentiel.

Modèle à deux niveaux sur labeled_data avec features (Logistic Regression + Random Classifer)

La matrice de confusion normalisée montre une bonne

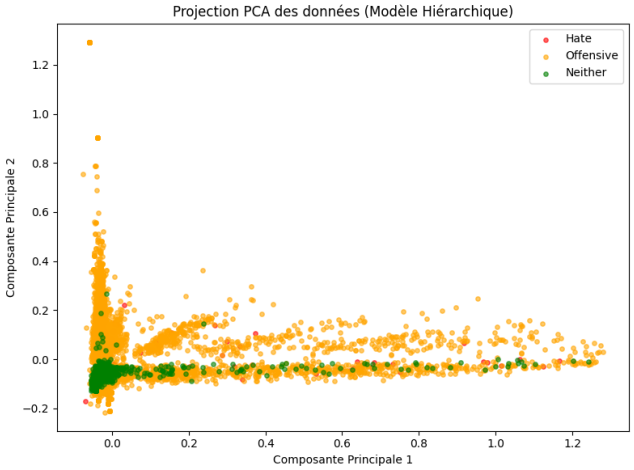

Figure 7. [PCA] Modèle à deux niveaux sur labeled_data sans features (Logistic Regression + Random Classifer)

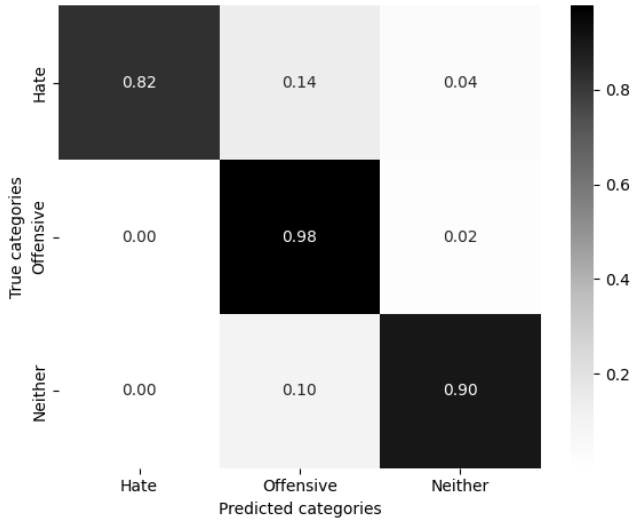

Figure 8. [Matrice de confusion] Modèle à deux niveaux sur labeled_data avec features (Logistic Regression + Random Classifer)

Classe	Précision	Rappel	F1-score
Hate	0.96	0.82	0.89
Offensive	0.97	0.98	0.97
Neither	0.89	0.90	0.90
Exactitude globale		0.96	
Macro moyenne	0.94	0.90	0.92
Moyenne pondérée	0.96	0.96	0.96

Table 4
[Rapport] Modèle à deux niveaux sur labeled_data avec features (Logistic Regression + Random Classifer)

précision sur les classes "Offensive" (98%) et "Neither" (90%). La classe "Hate", bien que globalement bien identifiée (82%), reste en partie confondue avec "Offensive" (14%). Cela confirme que même avec des features enrichis, la frontière entre discours haineux et discours simplement offensant demeure complexe, appelant à des méthodes d'amélioration ciblées (seuils adaptatifs, pondération des erreurs, etc.).

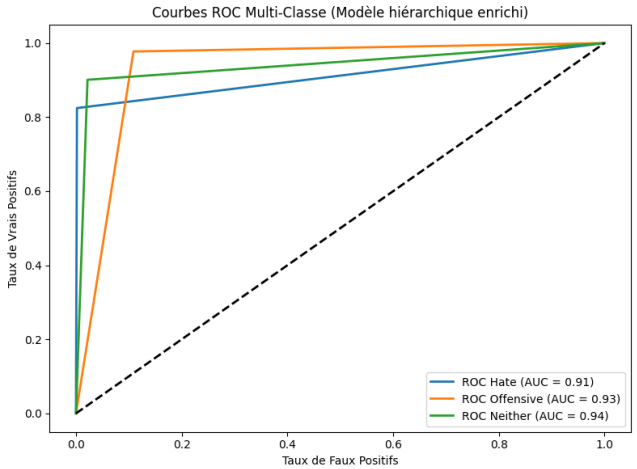

Figure 9. [Courbe ROC] Modèle à deux niveaux sur labeled_data avec features (Logistic Regression + Random Classifer)

L'ajout de features améliore globalement la qualité de la classification. Les AUC s'élèvent à 0.91 (Hate), 0.93 (Offensive) et 0.94 (Neither). Les courbes sont plus proches de l'idéal, en particulier pour la classe "Neither". Toutefois, la courbe de "Hate" reste légèrement inférieure, confirmant la complexité de cette classe et justifiant l'exploration de méthodes plus fines.

La projection PCA des données enrichies montre une structuration plus claire, en particulier pour la classe "Hate". Les features semblent capter davantage la variance discriminante, comme le montre l'étirement des données le long de la première composante. Le chevauchement entre "Offensive" et "Neither" persiste, mais la séparation globale des classes est renforcée. L'utilisation de représenta-

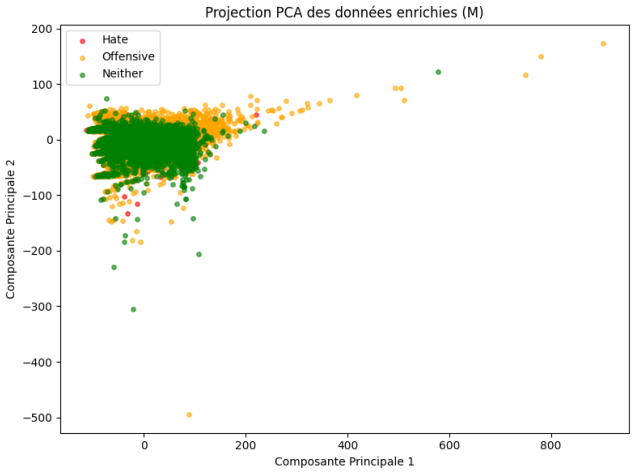

Figure 10. [PCA] Modèle à deux niveaux sur labeled_data avec features (Logistic Regression + Random Classifer)

tions encore plus riches (par exemple via des embeddings neuronaux) pourrait permettre une discrimination optimale dans un espace de plus grande dimension.

Modèle à deux niveaux sur labeled_data sans features (Logistic Regression + Logistic Regression)

Class	Precision	Recall	F1-score
Hate	0.50	0.80	0.62
Offensive	0.97	0.91	0.94
Neither	0.82	0.88	0.85
Accuracy			0.90
Macro avg	0.77	0.87	0.80
Weighted avg	0.92	0.90	0.91

Table 5
Rapport final – Régression Logistique L2 (approche hiérarchique à deux niveaux)

Le rapport souligne des performances déséquilibrées :

- "Offensive" excelle (F1-score = 0.94, précision = 0.97), bénéficiant d'une forte séparabilité.
- "Neither" suit avec un F1-score de 0.85, mais sa précision (0.82) révèle quelques confusions résiduelles.
- "Hate" accuse un retard marqué (F1-score = 0.62), avec une précision faible (0.50) malgré un rappel acceptable (0.80), signant un taux élevé de faux positifs.

L'accuracy globale (0.90) masque ce déséquilibre, tandis que les moyennes macro (précision = 0.77, rappel = 0.87) confirment l'hétérogénéité inter-classes. Ces résultats plaident pour un rééquilibrage des métriques (pondération des classes) ou l'intégration de features ciblant spécifiquement la sémantique des discours haineux.

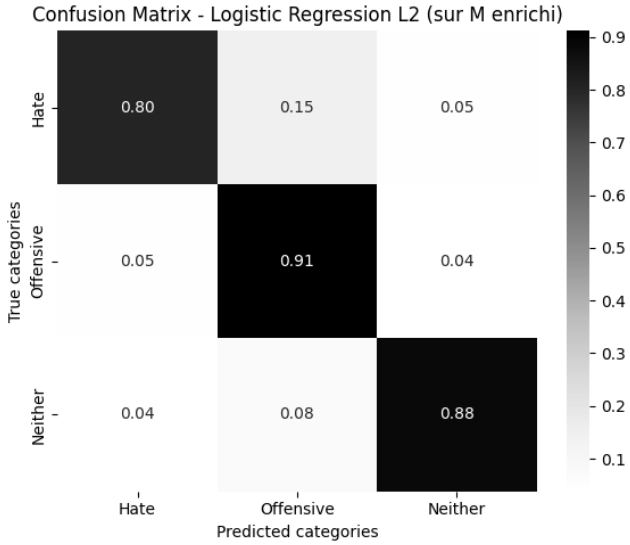

Figure 11. [Matrice de confusion] Modèle à deux niveaux sur labeled_data sans features (Logistic Regression + Logistic Regression)

La matrice de confusion normalisée révèle une performance contrastée selon les classes. La classe "Offensive" est excellente avec 91% de prédictions correctes, suivie de près par "Neither" (88%). Cependant, la classe "Hate" montre des difficultés : bien que 80% des instances soient correctement classées, 15% sont confondues avec "Offensive" et 5% avec "Neither". Cette confusion reflète la proximité sémantique entre discours haineux et offensants, soulignant la nécessité de features discriminants supplémentaires pour affiner la frontière entre ces classes.

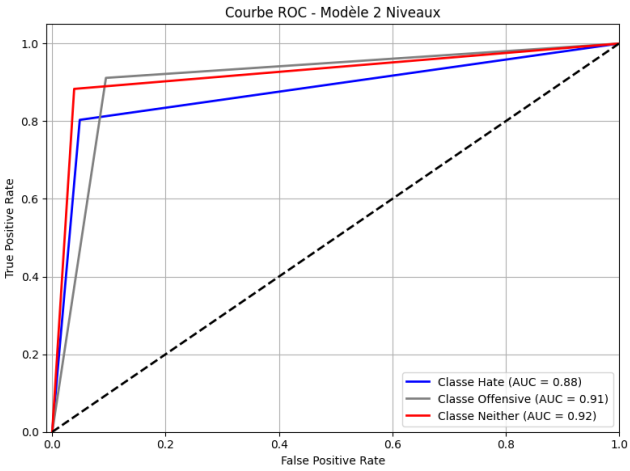

Figure 12. [Courbe ROC] Modèle à deux niveaux sur labeled_data sans features (Logistic Regression + Logistic Regression)

Les courbes ROC confirment la robustesse du modèle

pour les classes "Offensive" (AUC = 0.91) et "Neither" (AUC = 0.92), avec des courbes proches de l'angle supérieur gauche. La classe "Hate" (AUC = 0.88) présente une courbe légèrement moins performante, notamment dans la plage de spécificité élevée, ce qui traduit une sensibilité réduite pour détecter les vrais positifs sans erreurs. Cette divergence suggère que la discrimination des discours haineux nécessite des ajustements (seuils, pondérations) ou des features plus informatives.

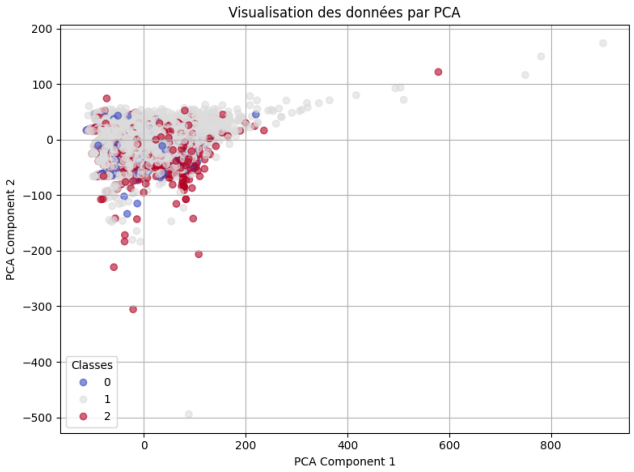

Figure 13. [PCA] Modèle à deux niveaux sur labeled_data sans features (Logistic Regression + Logistic Regression)

La projection PCA illustre un chevauchement significatif entre les classes, particulièrement entre "Hate" et "Offensive". La dispersion de la classe "Hate" le long des deux composantes principales indique un manque de structure linéaire claire, tandis que "Offensive" et "Neither" forment des amas plus compacts mais partiellement superposés. Cette visualisation valide l'hypothèse que les données brutes, sans enrichissement, peinent à séparer les classes dans un espace de faible dimension, justifiant l'exploration de représentations plus riches (embeddings contextuels, features sémantiques).

Modèle à un niveaux sur FTR_new_labels sans features (RandomForestClassifier)

Classe	Précision	Rappel	F1-score
Neutre	0.84	0.84	0.84
Haineux	0.67	0.66	0.67
Exactitude (accuracy)			0.78
Moyenne macro	0.76	0.75	0.75
Moyenne pondérée	0.78	0.78	0.78

Table 6
Résultats de classification binaire (Neutre vs Haineux)

La matrice de confusion révèle une performance

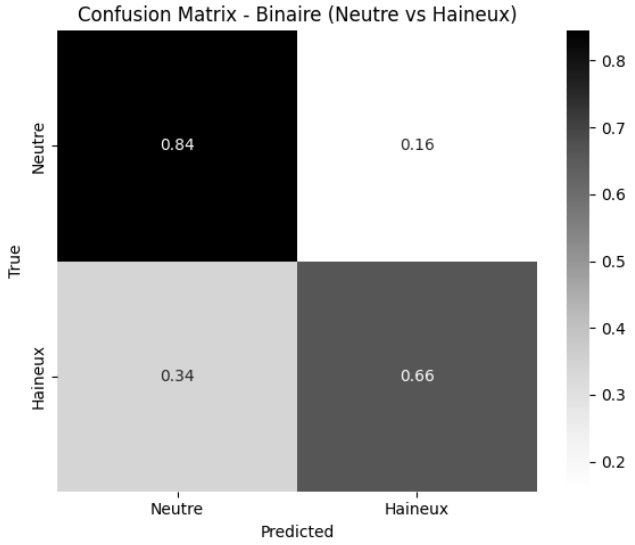

Figure 14. [Matrice de confusion] Modèle à un niveaux sur FTR_new_labels sans features (RandomForestClassifier)

asymétrique : la classe "Neutre" est bien identifiée (84% de vrais positifs), mais la classe "Haineux" montre une détection plus limitée (66%). Les 16% de "Neutre" classés à tort comme "Haineux" et 34% de "Haineux" faussement attribués à "Neutre" soulignent une confusion persistante, probablement due à des chevauchements sémantiques ou à un déséquilibre des données.

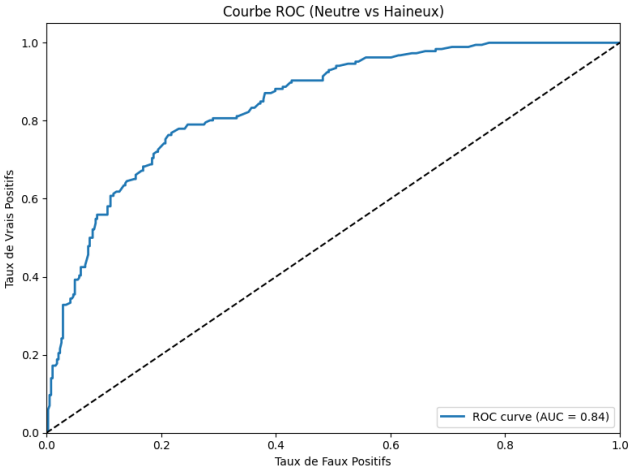

Figure 15. [Courbe ROC] Modèle à un niveaux sur FTR_new_labels sans features (RandomForestClassifier)

La courbe ROC (AUC = 0.84) confirme une discrimination acceptable mais perfectible. La pente moins abrupte pour les faibles taux de faux positifs suggère que le modèle peine à isoler les cas "Haineux" sans erreurs, reflétant les limites des features brutes. Un rééchantillonnage ou l'ajout de lexiques spécifiques pourrait améliorer la sensibilité.

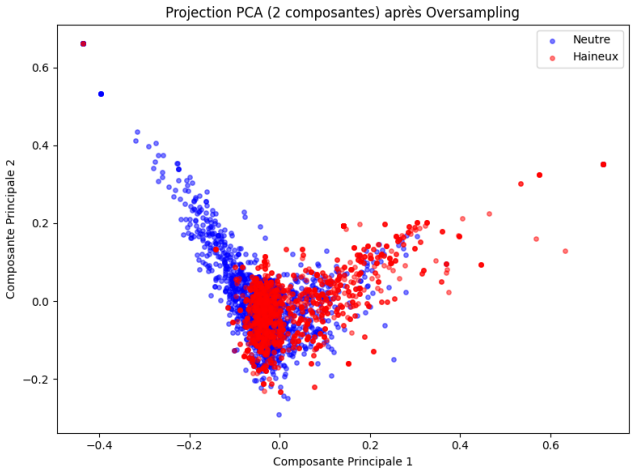

Figure 16. [PCA] Modèle à un niveaux sur FTR_new_labels sans features (RandomForestClassifier)

La projection PCA post-oversampling montre un chevauchement partiel entre les classes, malgré une séparation globale. La dispersion des points "Haineux" indique une variabilité intrinsèque, tandis que les "Neutre" forment un amas plus dense. Cette visualisation valide la nécessité de features plus discriminantes (e.g., embeddings contextuels) pour réduire les zones d'ambiguïté.

Modèle à un niveaux sur FTR_new_labels sans features (Logistic Regression)

Classe	Précision	Rappel	F1-score
Neutre	0.86	0.80	0.83
Haineux	0.63	0.72	0.67
Exactitude (accuracy)			0.77
Moyenne macro	0.74	0.76	0.75
Moyenne pondérée	0.78	0.77	0.78

Table 7
Résultats de classification binaire sur échantillon réduit

Comparé au Random Forest, ce modèle montre une détection légèrement moins bonne pour "Haineux" (72% vs 66%), mais une meilleure spécificité pour "Neutre" (80%). Les 20% de faux positifs ("Neutre" → "Haineux") et 28% de faux négatifs ("Haineux" → "Neutre") confirment la difficulté à capturer la nuance haineuse avec un classifieur linéaire.

Random Classifier sur Multitarget Conan

La matrice multiclasse révèle une excellente discrimination pour la plupart des cibles (e.g., "JEWS" : 99% de vrais positifs). Les confusions résiduelles concernent principalement les classes minoritaires (e.g., "other"), avec des erreurs vers des catégories sémantiquement proches (e.g., "MIGRANTS" -> "POC").

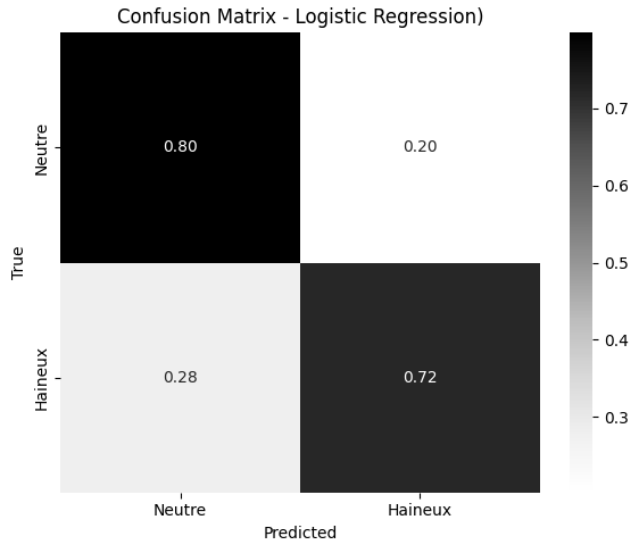

Figure 17. [Matrice de confusion] Modèle à un niveaux sur FTR_new_labels sans features (Logistic Regression)

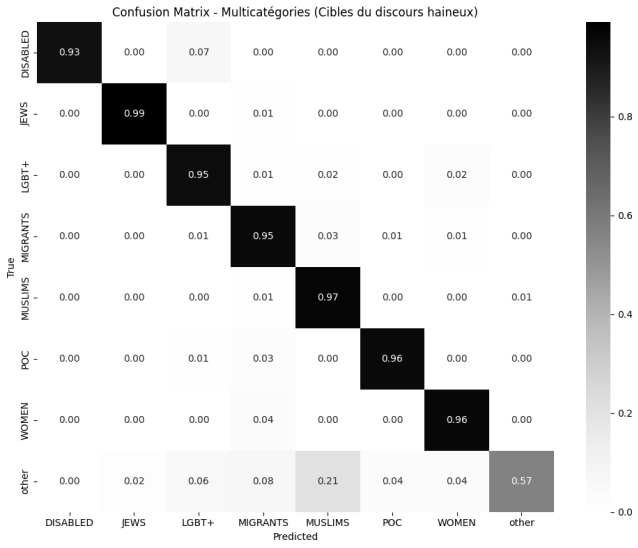

Figure 18. [Matrice de confusion] Random Classifier sur Multitarget Conan

Classe	Précision	Rappel	F1-score
DISABLED	1.00	0.93	0.96
JEWS	0.99	0.99	0.99
LGBT+	0.93	0.95	0.94
MIGRANTS	0.92	0.95	0.94
MUSLIMS	0.94	0.97	0.95
POC	0.96	0.96	0.96
WOMEN	0.94	0.96	0.95
other	0.94	0.57	0.71
Exactitude (accuracy)			0.94
Moyenne macro	0.95	0.91	0.93
Moyenne pondérée	0.94	0.94	0.94

Table 8
Rapport de classification multi-classe par groupe ciblé

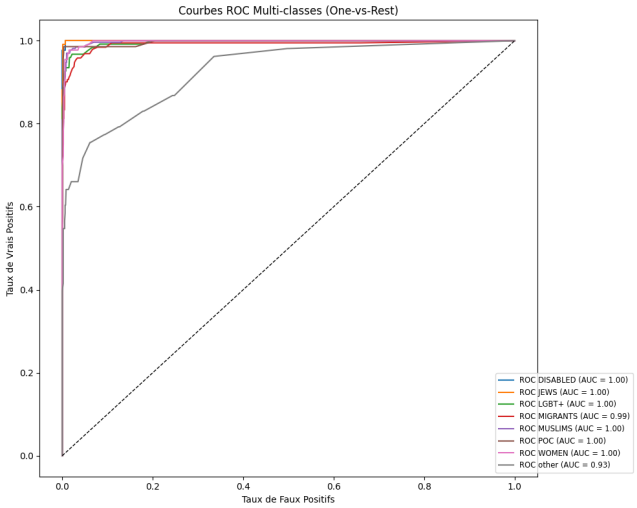

Figure 19. [Courbe ROC] Random Classifier sur Multitarget Conan

Les AUCs proches de 1.00 (sauf "other" : 0.93) attestent d'une séparation quasi parfaite des classes. La courbe de "other" moins performante reflète sa nature hétérogène, difficile à modéliser sans features spécifiques.

La visualisation PCA montre des clusters bien distincts pour chaque cible, sauf pour "other" qui se disperse. Cette structuration explique les performances élevées, mais souligne la nécessité de sous-catégoriser "other" pour améliorer sa détection.

La classe "MIGRANTS" montre une légère confusion avec "POC" (4%) et "other" (2%), reflétant des recoupements sémantiques ou contextuels.

La catégorie "other" présente le taux d'erreur le plus élevé (5% attribués à "DISABLED"), probablement due à son hétérogénéité et à son manque de définition claire.

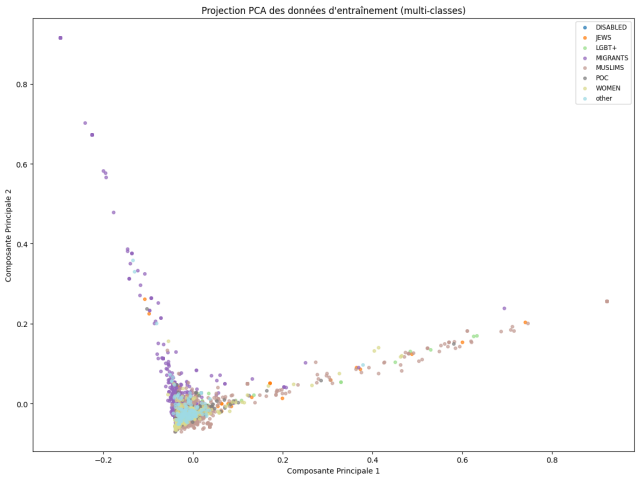

Figure 20. [PCA] Random Classifier sur Multitarget Conan

Robustesse générale : Les quasi-absence d'erreurs pour "JEWS", "DISABLED", et "LGBT+" (moins de 2% de faux positifs/négatifs) attestent de la capacité du modèle à capturer les spécificités lexicales ou thématiques de ces cibles.

Logistic Regression sur Multitarget Conan

Classe	Précision	Rappel	F1-score
DISABLED	0.98	0.93	0.95
JEWS	0.98	0.97	0.98
LGBT+	0.96	0.94	0.95
MIGRANTS	0.92	0.92	0.92
MUSLIMS	0.93	0.98	0.95
POC	0.89	0.97	0.93
WOMEN	0.95	0.95	0.95
other	0.81	0.57	0.67
Exactitude (accuracy)			0.93
Moyenne macro	0.93	0.90	0.91
Moyenne pondérée	0.93	0.93	0.93

Table 9
Rapport de classification multi-classe par groupe ciblé (version alternative)

La matrice de confusion révèle des performances globalement excellentes pour la classification des cibles spécifiques du discours haineux, avec quelques nuances notables :

Précision élevée : La plupart des catégories atteignent des scores de classification supérieurs à 92-98%, en particulier pour les groupes bien définis comme "JEWS" (97%), "POC" (97%), et "WOMEN" (95%).

La classe "MIGRANTS" montre une légère confusion avec "POC" (4%) et "other" (2%), reflétant des recoupements sémantiques ou contextuels. La catégorie "other" présente le taux d'erreur le plus élevé (5% attribués à "DIS-

ABLED"), probablement due à son hétérogénéité et à son manque de définition claire. Les quasi-absence d'erreurs pour "JEWS", "DISABLED", et "LGBT+" (moins de 2% de faux positifs/négatifs) attestent de la capacité du modèle à capturer les spécificités lexicales ou thématiques de ces cibles.

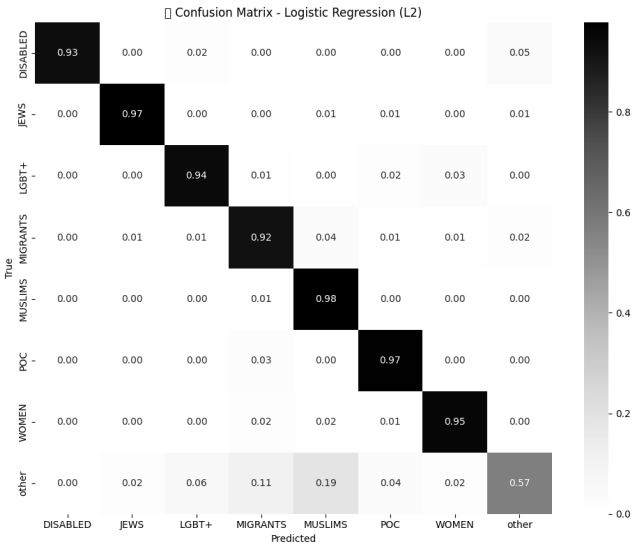

Figure 21. [Matrice de confusion] Logistic Regression sur Multitarget Conan

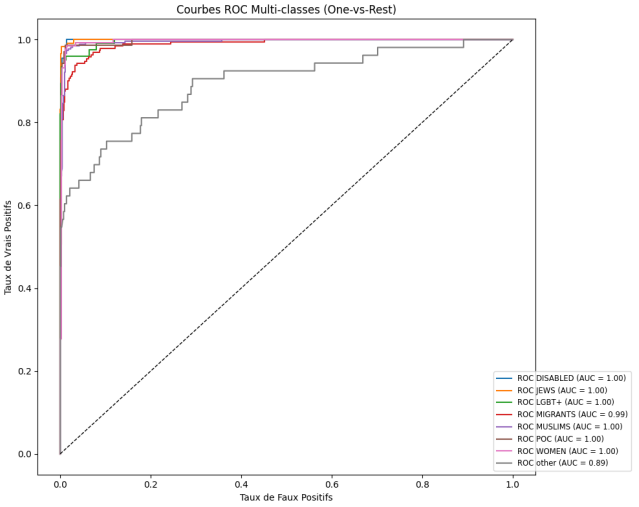

Figure 22. [Courbe ROC] Logistic Regression sur Multitarget Conan

Les AUCs restent excellents (inférieur ou égale à 0.99 sauf "other" : 0.89), mais légèrement inférieurs au Random Classifier, notamment pour "MIGRANTS" (0.99 vs 1.00).

La projection PCA montre des clusters similaires au Random Classifier, mais avec un chevauchement accru pour "MIGRANTS" et "POC", expliquant la baisse marginale des AUCs.

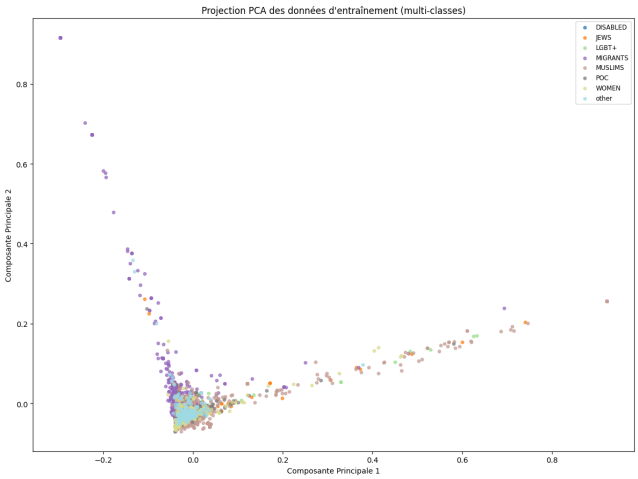

Figure 23. [PCA] Logistic Regression sur Multitarget Conan

5.4 Limites de l'approche

Bien que notre approche soit plus efficace que celle de Davidson et al. (2017), elle n'est pas sans défauts :

- **Surapprentissage dû à l'oversampling** : Le RandomOverSampler crée des doublons de la classe minoritaire, ce qui peut rendre le modèle trop sensible à certaines structures artificielles du langage.
- **Erreurs de propagation** : Une mauvaise décision au niveau 2 (par exemple, ne pas détecter un discours haineux) empêche tout rattrapage ultérieur : le tweet est alors traité au niveau 1, où le modèle n'est pas entraîné à identifier la haine.
- **Complexité accrue** : La mise en œuvre hiérarchique nécessite un pipeline plus complexe, un encodage coordonné et une gestion rigoureuse de la fusion des prédictions, ce qui peut introduire des risques de bugs ou d'erreurs d'implémentation.

5.5 Features Importance

Nous allons à présent analyser les features importantes du modèle. Cette aproche a pour but de comprendre les éléments du texte qui influencent le plus les décisions du modèle. Elle ne permet pas complètement d'ouvrir la boite noire qu'est le modèle, mais c'est un pas de plus vers l'interprétabilité. Dans notre cas nous cherchons à identifier :

- Quels mots, structures grammaticales ou caractéristiques sémantiques sont typiques d'un discours haineux, d'un discours offensif, ou d'un discours neutre.

- Cela nous permet de valider le comportement du modèle, de l'interpréter, et potentiellement d'améliorer son design ou d'identifier des biais.

Nous utilisons avons utilisé un modèle de régression logistique (LogisticRegression) avec régularisation L2, qui a l'avantage de fournir directement les coefficients associés à chaque feature. Ces coefficients représentent l'impact de chaque variable sur la prédiction.

Les données textuelles sont transformées en vecteurs numériques via :

- **TfidfVectorizer** : transforme les mots ou groupes de mots (n-grams) en scores TF-IDF, mettant en valeur les termes discriminants.
- TfidfVectorizer appliqué aux **tags grammaticaux (POS)** pour capturer la structure linguistique.
- Des **features manuelles** ajoutées, comme: scores de sentiment, nombre de hashtags, ou encore, scores de lisibilité, ect.

Chaque feature (colonne du vecteur final) est associée à un coefficient :

- **Coefficient positif** : la présence de cette feature augmente la probabilité d'appartenir à la classe cible.
- **Coefficient négatif** : la présence de cette feature diminue cette probabilité (ou favorise l'autre classe dans le cas binaire).

En représentant ces coefficients dans un graphique, on visualise les top 20 features les plus discriminantes pour chaque classe.

Résultats : Offensive vs Neither (Niveau 1)

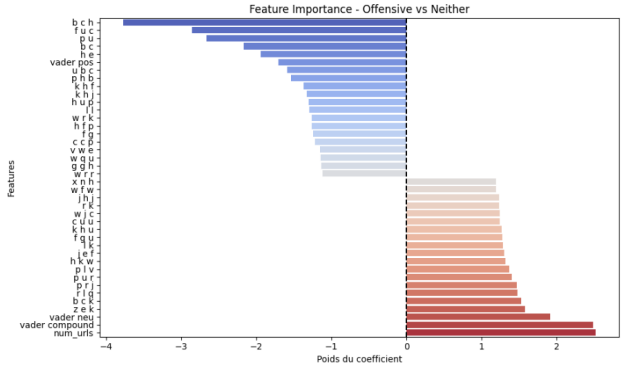

Figure 24. Résultats : Offensive vs Neither (Niveau 1)

Dans cette tâche, on cherche à distinguer les tweets offensifs des tweets neutres.

Features les plus associées aux tweets offensifs :

- **num_urls, vader_compound, vader_neu** : les tweets offensifs contiennent souvent des liens, et présentent un sentiment plus neutre ou ambivalent selon VADER.
- Certains **n-grams (z e k, r l q, etc.)** correspondent probablement à des fragments de mots injurieux.

Features associées aux tweets neutres :

- **vader_pos** : un score VADER positif est corrélé à des tweets non offensifs.
- De nombreux **n-grams (b c h, f u c, etc.)** apparaissent négativement pondérés, car ils sont plus fréquents dans les tweets offensifs.

Résultats : Hate Speech vs All (Niveau 2)

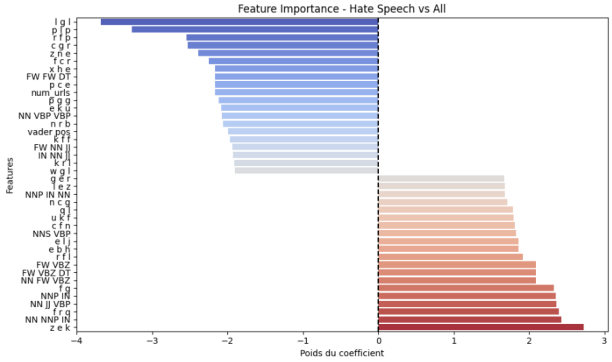

Figure 25. Résultats : Hate Speech vs All (Niveau 2)

Dans cette tâche, on cherche à isoler les tweets de haine des autres (offensifs et neutres).

Features les plus associées aux tweets haineux :

- Plusieurs **POS n-grams comme NN JJ VBP, NN NNP IN, etc.** indiquent une structure grammaticale spécifique à des propos haineux.
- Des **trigrams de caractères (ex: l g l, p f p)** issus probablement de phrases codées ou insultes déformées.

Features associées aux autres types de tweets :

- **num_urls, vader_pos** : les discours contenant des

liens ou du sentiment positif sont moins susceptibles d'être haineux.

- **POS comme FW FW DT ou NN VBZ DT** peuvent indiquer une syntaxe plus neutre ou descriptive.

Intérêts de cette analyse :

- **Interprétabilité** : On peut expliquer ce qui motive le modèle à classer un tweet dans telle ou telle catégorie.
- **Détection de biais** : On vérifie que le modèle ne se base pas sur des éléments non pertinents (ex. sur-représentation d'un mot innocent).
- **Amélioration du modèle** : En repérant des patterns inattendus ou trop faibles, on peut ajuster les features ou le prétraitement.

5.6 Hyperparamétrage du modèle (Fine-Tuning)

L'hyperparamétrage permet d'optimiser les performances d'un modèle d'apprentissage automatique. Nous allons optimiser le choix des hyperparamètres, c'est-à-dire, les paramètres non appris par le modèle, qui sont définis avant l'entraînement. Cette étape ne permet donc pas d'augmenter les performances de manière significative, mais elle permet d'obtenir le meilleur modèle possible.

Dans ce cadre, nous allons analyser l'évolution du mean cross-validation score (mean CV score) en fonction de deux hyperparamètres clés : max_depth et n_estimators, utilisés dans les modèles d'ensemble de type Random Forest.

- **max_depth** représente la profondeur maximale des arbres de décision. Une profondeur trop faible peut empêcher le modèle de capturer la complexité des données (sous-apprentissage), tandis qu'une profondeur trop élevée peut le rendre trop spécifique aux données d'entraînement (sur-apprentissage). En faisant varier ce paramètre, nous évaluons jusqu'à quel point il est bénéfique d'augmenter la complexité des arbres.
- **n_estimators** correspond au nombre d'arbres dans la forêt. Un plus grand nombre d'estimateurs peut améliorer la stabilité et la performance du modèle, jusqu'à un certain point où les gains deviennent marginaux ou le coût computationnel trop élevé. Nous analysons donc son impact pour déterminer à partir de quel seuil l'ajout d'arbres n'apporte plus d'amélioration notable.

En étudiant l'évolution du mean CV score par rapport à ces deux paramètres, nous cherchons à comprendre comment ils influencent la capacité de généralisation du modèle sur des données nouvelles. Cela nous permet d'identifier une configuration optimale, qui maximise les performances tout en évitant le surajustement.

Performance vs max_depth (0 à 500)

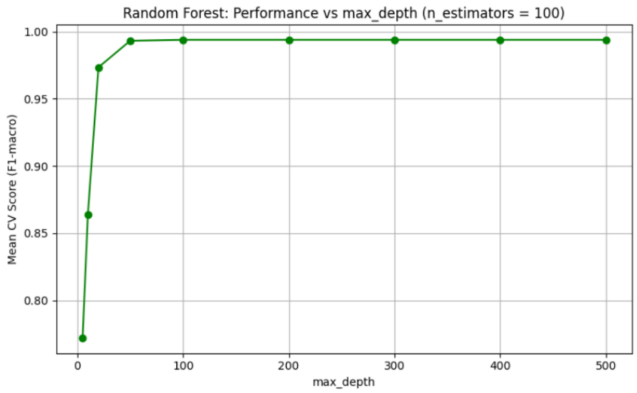

Figure 26. Performance vs max_depth (0 à 500)

Nous réalisons que le Mean CV Score augmente entre 0 et 50 et se stabilise ensuite. On arrive à 0.99 de 100 jusqu'à 500. Sachant que le modèle atteint sa performance maximale avec 50-100, nous allons garder la profondeur maximale des arbres à 100.

Performance vs n_estimators (100 à 200)

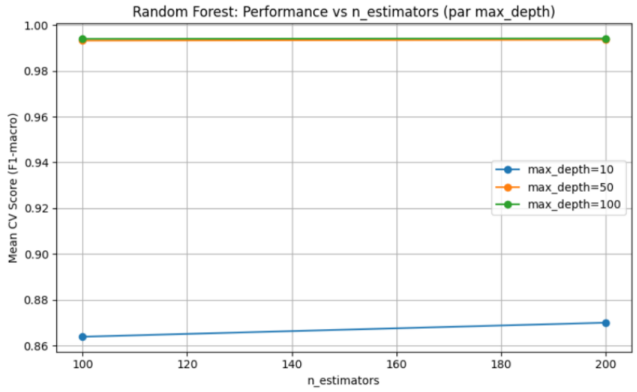

Figure 27. Performance vs n_estimators (100 à 200)

A présent, nous allons analyser l'évolution du score en fonction du nombre d'arbres. Le modèle est beaucoup moins performant avec 50 arbres, et il est aussi performant avec 50 que 100. Ainsi, pour optimiser sans compromiser la qualité du modèle, nous allons fixer 50.

Ces analyses permettent de confirmer que la profondeur des arbres joue un rôle significatif dans la qualité du modèle, avec des performances optimales atteintes à des profondeurs intermédiaires (autour de 50 à 100).

6 Approfondissement du sujet

Pour étendre notre sujet, nous allons ajouter un jeu de données composé de paroles de rap, afin de les analyser et déterminer si elles relèvent du discours haineux ou de l'offense. En début d'article, nous avons évoqué que certaines insultes dans les tweets peuvent être perçues comme légitimes. N'étant pas nécessairement malveillantes, elles sont perçues comme légitimes.

Et cette nuance, justement, ouvre une nouvelle dimension juridique, presque philosophique. Elle nous amène à nous interroger sur l'essence de la haine, au fond. Qu'est ce que l'offense ? Où se situe la limite entre violence et liberté d'expression ?

6.1 L'ambiguïté juridique du rap comme forme artistique

Selon la jurisprudence française, le rap bénéficie d'une protection particulière en tant qu'expression artistique. Dans de nombreuses décisions de justice, le rap est perçu comme un exutoire social, une manière de dénoncer la violence subie par certaines populations. C'est donc devenu le reflet d'une souffrance, au sein des tribunaux.

Toutefois, cette protection soulève des questions car elle semble rigide même en cas de propos violents, sexistes, homophobes ou complotistes. Dès lors, une question s'impose : à partir de quel moment une création cesse-t-elle d'être artistique pour devenir un discours haineux ? La jurisprudence elle-même établit une distinction entre la mise en scène et l'adhésion réelle de l'auteur à ses propos. Ce point devient d'autant plus sensible lorsque certains artistes ont un passé judiciaire marqué, ou revendiquent eux-mêmes un usage purement lucratif de la violence dans leurs clips, sans prétention artistique ou politique réelle. Il est donc légitime de s'interroger sur les critères qui devraient prévaloir.

6.2 Le biais du crowdsourcing : qui juge, et selon quelles normes ?

Dans notre jeu de données, les annotations sont issues d'un processus de crowdsourcing, c'est-à-dire confiées à un public non expert chargé de classer les contenus selon leur caractère haineux, offensant ou neutre. Mais qui est ce public ? Et surtout, selon quels critères juge-t-il ? Chaque annotateur vient avec ses propres sensibilités, valeurs culturelles, et contextes personnels. Ce qui est perçu comme une insulte pour l'un peut apparaître comme de l'ironie ou de l'art pour l'autre. Il devient alors légitime de s'interroger : une majorité populaire mal informée ou déconnectée de ces codes peut-elle juger équitablement de ce qui relève de la haine ou de l'art ? Cette réflexion rejoint la tension plus large entre justice algorithmique et jugement humain, entre normes sociales dominantes et expressions culturelles minoritaires.

7 Conclusion

Cette étude visait à reproduire fidèlement les résultats de l'article de référence sur la détection automatique du discours haineux, en s'appuyant sur le jeu de données labeled_data. Etant donné la sucjectivité du language et la nature implicite de la haine en ligne, l'enjeu principale était la distinction efficace entre le discours haineux et les propos offensants.

Les résultats reproduis ont été obtenus avec deux modèles classiques : une régression logitique avec validation croisée et une SVM linéaire, qui montraient des performances satisfaisantes sur la classe "Offensive", mais qui capturait moins bien les messages haineux.

Enfin, le modèle hiérarchique que nous avons proposé a démontré une amélioration otable de la détection des discours hianeux et offensifs, comparés aux approches traditionnelles. En divisant la tâche en deux, entre détection initial du discours haineu puis classification entre discours neutre et offensif, le système a permis de mieux détecter les messages haineux.

L'intégration de techniques comme l'oversampling, ainsi que l'évaluation sur des jeux de données externes (Multitarget-CONAN, FTR), a permis de valider la robustesse, la généralisabilité et l'adaptabilité du modèle, y compris dans des contextes multilingues ou multiclasses.

Les résultats montrent que, bien que les performances globales soient élevées (jusqu'à 94% d'exactitude), la détection du discours haineux reste la plus difficile, surtout en l'absence de features enrichis. L'analyse via PCA et ROC confirme cette difficulté, mettant en lumière l'importance des représentations sémantiques plus fines pour améliorer la détection des cas ambigus.

Ainsi, cette approche hiérarchique, représente une avancée méthodologique significative pour la classification fine des contenus en ligne, avec des perspectives d'optimisation claires : intégration de modèles de langage pré-entraînés, affinement des seuils de décision, et enrichissement sémantique des features.

L'analyse des features a montré que certaines features, comme les scores de sentiment (VADER), la présence de liens (num_urls), ou encore des n-grams spécifiques, jouent un rôle déterminant dans la classification. De manière intéressante, les discours haineux se distinguent souvent par des structures grammaticales atypiques ou des trigrams

codés, tandis que les tweets neutres présentent des caractéristiques linguistiques plus conventionnelles et des signaux affectifs positifs.

En conclusion, cette étude met en évidence l'efficacité d'une approche hiérarchique à deux niveaux combinée à une analyse des features pour la détection automatique de discours haineux et offensifs, tout en soulignant les limites des méthodes traditionnelles face à la complexité sémantique et implicite de la haine en ligne.

Bibliographie

- Josiane Boutet (2016). *La question de l'interprétation en linguistique*. Travailler, n°35, pp. 161–176. Disponible sur : https://shs.cairn.info/revue-travailler-2016-1-page-161?lang=fr (consulté en avril 2025).
- Pang, Bo et Lee, Lillian (2008). *Sentiment Analysis: An Overview*. Disponible sur : https://www.academia.edu/291678/Sentiment_Analysis_An_Overview (consulté en avril 2025).
- Jacob, Murel Ph.D. (10 décembre 2023). *Qu'est-ce que la racinisation et la lemmatisation ?* IBM Think. Disponible sur : https://www.ibm.com/fr-fr/think/topics/stemming-lemmatization (consulté en mai 2025).
- Jobard, Jean-Baptiste (2022). *RAP : une violence seulement artistique ?* Actu-Juridique. Disponible sur : https://www.actu-juridique.fr/libertes-publiques-ddh/rap-une-violence-seulement-artistique/ (consulté en mai 2025).

Remerciements

Ce mémoire a été réalisé avec l'aide précieuse de l'enseignante-chercheuse Séverine AFFELDT, que nous remercions chaleureusement pour son accompagnement et ses conseils tout au long du travail. Nous tenons également à remercier l'Université Paris Cité, et plus particulièrement, l'UFR Sciences Biomédicales et Fondamentales, pour nous avoir offert un cadre propice à la recherche et à l'apprentissage.