NLP – Modération de Chat

Nakache Eithan
Ziane Camil
Hadj-Said Samy
Perez Jason
Six Briac
Bellamy Baptiste

Sujet

Modération en temps réel des chats sur une plateforme de contenu, par l'intermediaire de modèle NLP

Table des matières

1. Introduction

Ce projet se propose de développer des modèles du traitement du language naturel dédiés à la modération en temps réel des chats sur une plateforme de contenu. L'objectif principal est d'assurer la modération des messages des utilisateurs afin de prévenir la diffusion de contenus inappropriés, tels que les insultes, les spams et autres formes de communications indésirables.

Dataset

Le dataset choisit pour ce projet est le **Jigsaw Toxic Comment Classification**. Ce dataset recense un grand nombre de commentaires anglais, provenant de Wikipedia, labellisés par des humains en fonction de leur toxicité. Ce dernier provient d'une compétition Kaggle et peut être obtenu à partir du lien suivant : jigswaw toxic comment classification challenge.

Objectif

Plusieurs modèles de machine learning et de deep learning seront développés et entraînés dans le but de prédire la pertinence et l'acceptabilité des messages. Une fois ces modèles développés, nous procéderons à une évaluation rigoureuse de leurs performances par comparaison mutuelle. L'ultime étape de ce projet consistera en la création d'une preuve de concept sous la forme d'une interface web. Cette interface permettra de modérer en direct les messages échangés sur un chat de live streaming, démontrant ainsi l'applicabilité pratique de nos recherches.

2. Présentation du jeu de données

Objectifs et Financement du Dataset

Le dataset de classification des commentaires toxiques de JIGSAW a été créé pour promouvoir la recherche sur la détection de la toxicité dans les commentaires en ligne. Il vise à identifier des comportements indésirables tels que les commentaires toxiques. Ce corpus a été collecté dans le but de développer des technologies et des méthodes capables de modérer automatiquement ces types de contenu nuisible sur les plateformes en ligne.

Le projet a été financé par Jigsaw (anciennement connu sous le nom de Google Ideas) et Google, dans le cadre d'un concours organisé sur la plateforme Kaggle. Ce partenariat a non seulement mis à disposition les ressources nécessaires, mais a également encouragé la communauté globale des data scientists à résoudre ce problème urgent de modération des contenus toxiques sur Internet.

Contexte et Caractéristiques des Données du Dataset

Les commentaires inclus dans le dataset proviennent des pages de discussion de Wikipedia. Ces discussions sont menées en anglais par des contributeurs qui échangent sur les améliorations à apporter aux articles et sur les modifications nécessaires. Ces échanges sont caractéristiques des interactions collaboratives typiques sur Wikipedia, où les utilisateurs débattent de la véracité, de la neutralité et de la complétude des articles.

Le format du texte est écrit et prend la forme de communications en ligne non formelles mais structurées. Cela signifie que les commentaires, bien que rédigés dans un cadre informel, suivent une certaine structure logique et sont orientés vers des objectifs spécifiques de collaboration et de modification de contenu.

Démographie des auteurs

Les informations démographiques spécifiques sur les auteurs des commentaires ne sont pas fournies.

Processus de collecte

Le dataset comprend environ 160 000 commentaires pour l'entraînement et 60 000 commentaires pour le test. Ces données ont été extraites dans le but de représenter divers comportements toxiques, bien que la méthode exacte d'échantillonnage n'ait pas été spécifiée.

Étant issues de plateformes ouvertes, les questions de consentement sont gérées dans le cadre des normes de Wikipedia concernant la publication de commentaires publics. Toutefois, les détails spécifiques concernant le consentement des auteurs ne sont pas divulgués.

En ce qui concerne le prétraitement, les données ont été anonymisées et les informations personnellement identifiables ont été supprimées pour protéger la vie privée des utilisateurs.

Processus d'annotation

Le dataset est structuré autour de plusieurs catégories d'annotations qui permettent de définir la nature de la toxicité des commentaires. Celles-ci incluent : **Toxique**, **Très toxique**, **Obscène**, **Menace**, **Insulte**, **Haine identitaire**. Ces catégories ont été choisies pour couvrir un large éventail de comportements toxiques potentiellement rencontrés dans les commentaires en ligne.

La méthode d'annotation repose sur l'intervention de multiples annotateurs pour chaque commentaire. Cette approche vise à maximiser la fiabilité des annotations. Le recours à plusieurs annotateurs permet de réduire les biais individuels et d'améliorer la précision générale des données annotées, assurant ainsi que les modèles de machine learning entraînés avec ce dataset peuvent fonctionner de manière efficace et équitable.

Distribution

Le dataset est disponible à des fins de recherche non commerciales. Les utilisateurs doivent généralement accepter des conditions d'utilisation qui limitent l'utilisation commerciale et la redistribution.

3. Statistiques descriptives du jeu de données

Division du dataset

Le dataset a été divisé en trois parties : entraînement, validation et test. Voici la répartition du nombre de commentaires dans chacune de ces parties :

Catégorie	**Nombre de commentaires**
Train	127,656
Validation	31,915
Test	63,978

TABLE 3.1 – Répartition du nombre de commentaires

Répartition des labels

Les commentaires toxiques sont minoritaires dans l'ensemble des données. En effet il y a **10.2%** de commentaire globalement non-toxique. Cela peut poser des problèmes lors de l'entraînement des modèles, car les classes minoritaires peuvent être sous-représentées et donc mal apprises. Il y a aussi une répartition inégale au sein des labels de toxicité On peut remarquer que la somme des pourcentages n'est pas égale à 100% car un commentaire peut avoir plusieurs labels. On est donc dans un problème de classification multi-labels.

Label	**Pourcentage**
toxic	94.3%
severe_toxic	9.8%
obscene	51.9%
threat	3.1%
insult	48.2%
identity-hate	8.6%

TABLE 3.2 – Répartition des labels sur les commentaires globalement toxiques

Corrélation entre les labels

On peut remarqué que les labels de toxicité sont fortement corrélés entre eux. On peut dès à présent anticiper une difficulté du modèle à distinge un commentaire toxique d'un commentaire obscene (**74%** de corrélation). Cela représente un point à prendre en compte lors de la conception du modèle. La matrice de corrélation est inclus dans l'appendice A.

Longueur des commentaires

La distribution de la longueur des commentaires est très variée. En effet l'écart type est bien plus élevé que la moyenne. La longueur moyenne des commentaires est de **395** caractères, avec un écart-type de **593** caractères. Cela peut poser des problèmes lors de la conception du modèle. Il est donc important de prétraiter les données pour normaliser la longueur des commentaires.

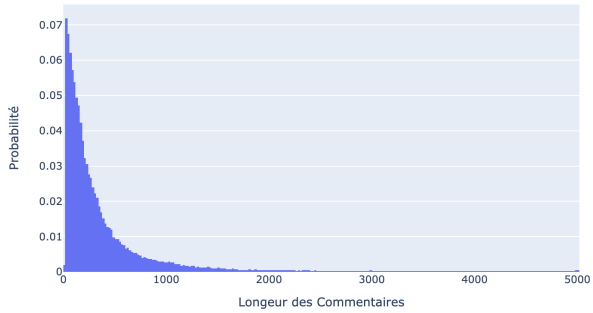

FIGURE 3.1 – Distribution de la longueur des commentaires

Distribution des mots

Dans le jeu de données, il y a un total de environs 180 000 mots uniques. Sans surprises, les mots les plus fréquents sont les mots de liaison et les mots vides. Mais en vue de la source du dataset les mots **articles**, **wikipedia** et **pages** apparaissent aussi très souvent dans les commentaires (top 26, 30, 31 respectivement). En effet les utilisateurs peuvent citer des sources pour appuyer leurs propos. Dans les commentaires toxiques, on trouve des insultes, des mots vulgaires et des mots discriminatoires. On peut visualiser ces derniers en utilisant WordCloud. Cela représente une représentation visuelle des mots les plus utilisés. En annexe A, on peut voir une représentation WordCloud générée à partir des commentaires du jeu de données filtré selon le type de toxicité.

4. Tokenizer

4.1 Pré-traitement du jeu de données

Le pré-traitement des données est une étape essentielle dans le processus d'analyse de texte. Il vise à nettoyer, transformer et préparer les données brutes afin de les rendre exploitables pour l'entraînement des modèles. Dans cette section, nous appliquerons différentes techniques de pré-traitement sur notre jeu de données afin de le rendre apte à être utilisé dans nos modèles prédictifs.

4.1.1 Tokenisation à base d'expressions régulières (RegexTokenizer)

La tokenisation est le processus de division du texte en unités plus petites appelées "tokens". La tokenisation à base d'expressions régulières utilise des règles basées sur des motifs d'expressions régulières pour diviser le texte en tokens.

Implémentation de la tokenisation à l'aide de la classe RegexTokenizer

Nous commençons par importer la classe RegexTokenizer et appliquer la tokenisation sur notre jeu de données.

4.1.2 Tokenisation byte-pair encoding (TikToken)

Le byte-pair encoding (BPE) est une méthode de tokenisation qui découpe le texte en sous-unités de texte appelées "tokens" en utilisant un algorithme de compression de données.

Implémentation de la tokenisation à l'aide de la bibliothèque TikToken

Nous appliquons la tokenisation BPE sur notre jeu de données en utilisant la bibliothèque TikToken.

4.1.3 Méthodes de normalisation du texte

La normalisation du texte est une étape cruciale du pré-traitement des données textuelles. Elle vise à uniformiser le texte en le mettant en minuscules, en supprimant les mots vides (stop words) et en lemmatisant les mots.

Suppression des stop words

Les stop words sont des mots courants qui n'apportent pas beaucoup de valeur sémantique au texte. Nous les supprimerons de notre jeu de données.

Lemmatisation

La lemmatisation consiste à réduire les mots fléchis ou dérivés à leur forme de base ou racine. Cela permet de normaliser le texte et de réduire la dimensionnalité de l'espace des features.

Mise en minuscules

La mise en minuscules permet d'uniformiser le texte en convertissant toutes les lettres en minuscules. Cela permet d'éviter les doublons dus à la casse.

5. Régression Logistique

Choix des Métrics

Avant de débuter l'analyse avec le modèle, il a été décidé de sélectionner des métriques d'évaluation telles que le rappel macro et le F1-score macro pour les labels *toxique*. En utilisant un modèle de régression linéaire, la compréhension des phrases peut être difficile et entraîner une précision réduite, surtout avec une labellisation imparfaite et des faux positifs. Cependant, il est crucial de maximiser le rappel pour détecter un maximum de phrases toxiques, quitte à accepter quelques erreurs. Le choix de la pondération macro a été privilégié pour une évaluation globale de la performance, en donnant le même poids à chaque classe. Ainsi, lors de chaque optimisation des paramètres, la métrique de rappel macro a été maximisée.

Étape de la construction du modèle

La classification *multilabel* a été simplifiée en classification binaire par la création d'un label unique nommé *overall toxic*, considéré comme vrai si au moins un des labels toxiques est présent. Deux méthodes de featurisation et douze méthodes de pré-traitement ont été explorées. Pour chaque featurisation, divers hyper-paramètres ont été optimisés à l'aide de la bibliothèque `Optuna`. Enfin, un modèle de régression linéaire *multilabel* a été entraîné sur les meilleures méthodes de pré-traitement et de featurisation.

5.0.1 TF-IDF

La featurisation *TF-IDF* est une méthode qui transforme les mots en vecteurs numériques, attribuant un poids à chaque mot selon sa fréquence dans le document et dans l'ensemble du corpus. Cette technique ne prend pas en compte la séquence des mots ni les relations entre eux. Malgré cette limitation, elle se révèle très efficace lorsqu'elle est utilisée avec des modèles de régression linéaire.

Premier Résultats

Étant donné le déséquilibre du jeu de données, le paramètre `class_weight` a été ajusté à `balanced` pour compenser cette disparité. Des tests ont été réalisés sur les douze variantes de pré-traitement du jeu de données. Voici les résultats obtenus pour le modèle de régression logistique sans optimisation des hyperparamètres :

TABLE 5.1 – Score Macro de la Régression Logistique TF-IDF, sur le jeu de validation. (3 meilleurs)

Technique de Pré-Traitement	Précision	Rappel	Score F1	Support
BPE Tokenize 2	**0.826**	**0.911**	**0.862**	31915
BPE Tokenize 1	0.822	0.910	0.859	31915
GPT Tokenize 1	0.820	0.910	0.857	31915
Baseline	0.790	0.887	0.8318	31915

Optimisation des hyper-paramètres

Afin d'optimiser les performances du modèle, une optimisation des hyper-paramètres a été réalisé sur la meilleure méthode de pré-traitement *BPE Tokenize 2*. Les meilleurs paramètres obtenues sont une régularisation de type *L1* avec un paramètre de régularisation C égal à *0.60*.

TABLE 5.2 – Score Macro de la Régression Logistique TF-IDF Optimisé, sur le jeu de validation. (3 meilleurs)

Technique de Pré-Traitement	Précision	Rappel	Score F1	Support
BPE Tokenize 2	**0.826**	**0.916**	**0.863**	31915
GPT Tokenize 1	0.820	0.915	0.859	31915
BPE Tokenize 1	0.821	0.914	0.860	31915
Baseline	0.784	0.890	0.825	31915

Résultat sur le jeu de test

TABLE 5.3 – Score Macro de la Régression Logistique TF-IDF, sur le jeu de test

Technique de Pré-Traitement	Précision	Rappel	Score F1	Support
BPE Tokenize 2	**0.706**	**0.900**	**0.751**	63978
Baseline	0.695	0.869	0.737	63978

La meilleur technique de pré-traitement reste le *BPE Tokenize 2* avec ces paramètres de régularisation. On remarque néanmoins que les résultats sur le jeu de test sont moins bons que sur le jeu de validation.

Interprétation des résultats

Faux positifs

L'examen des faux positifs révèle la présence de termes offensants, confirmant ainsi une labellisation imparfaite. Un nuage de mots des faux positifs est disponible en annexe B.

Faux négatifs

Il est à noter que certains termes négatifs sont plus fréquents dans le jeu de données de test que dans celui d'entraînement, ce qui contribue aux erreurs du modèle. Par exemple, le terme "boob" apparaît 1000 fois dans le jeu de données de test contre 32 fois dans celui d'entraînement. Un nuage de mots des faux négatifs est également disponible en annexe B.

Shap Values

Les *Shap Values* révèlent les mots ayant le plus d'impact sur le modèle. Sans surprise, les termes les plus influents sont les plus vulgaires. On remarque aussi que les mots comme "please" ou "thank" ont un impact négatif sur la prédiction de toxicité. Un aperçu des 20 mots avec les plus grandes *SHAP Values* est présenté en annexe B.

5.0.2 Word2Vec

La featurisation *Word2Vec* est une technique qui permet de convertir les mots en vecteurs de nombres. Elle permet de réduire la dimensionnalité des données et de capturer les relations sémantiques entre les mots.

Comparaison Pré-Entraîné et Entrainé à partir de zéro

Une comparaison a été effectué entre la representation *word2vec* d'un modèle pré-entraîné et celle d'un modèle entraîné à partir de notre jeu de données. Le modèle entraîné à partir de zéro repose sur 200 dimensions pour être comparable au modèle pré-entraîné. Il en ressort que la regréssion logistique est nettement meilleur avec un modèle pré-entraîné. Les moins bons résultats du modèle entrainé à partir de zéro peuvent s'expliquer par un manque de données pour exprimer la sémantique des mots.

Ce tableau comparatif soulève plusieurs points. L'importance des pré-traitements des données est plus marquée que pour la featurisation *TF-IDF*. De plus, les pré-

TABLE 5.4 – Score Macro de la Régression Logistique Word2Vec, sur le jeu de validation

Technique de Pré-Traitement	Précision	Rappel	Score F1	Support
Pré-Entraîné				
Word Tokenize 1	**0.744**	**0.884**	**0.790**	31915
BPE Tokenize 2	0.740	0.880	0.784	31915
Baseline	0.637	0.810	0.654	31915
Entraîné				
Word Tokenize 3	**0.677**	**0.863**	**0.712**	31915
BPE Tokenize 1	0.674	0.862	0.708	31915
Baseline	0.650	0.809	0.676	31915

traitement *GPT Tokenizer* sont moins performants avec le modèle *word2vec* pré-entraîné que les pré-traitement *word tokenize*. Cela s'explique par le fait que le modèle *word2vec* pré-entraîné attend des mots et non des *byte tokens*.

Résultat après optimisation des hyper-paramètres

La régression logistique a été optimisée en utilisant la meilleure technique de pré-traitement *Word Tokenize 1*, vectorisée par le modèle *word2vec* pré-entraîné. Lors de l'optimisation, il n'a pas été possible de tester la régularisation L1 car le modèle prenait trop de temps à converger. Le meilleur ensemble de paramètres inclut une régularisation *L2* avec un C égale à *27.26*.

Résultat sur le jeu de test

TABLE 5.5 – Score de la Régression Logistique Word2Vec Optimisé, sur le jeu de test

Technique de Pré-Traitement	Précision	Rappel	Score F1	Support
Word Tokenize 3	0.673	**0.870**	0.706	63978
Word Tokenize 1	**0.680**	0.859	**0.716**	63978
Baseline (Sans Pré-traitement)	0.637	0.810	0.655	63978

On peut noter que les résultats sont moins bons qu'à partir d'une featurisation TF-IDF.

Interprétation des résultats

Faux positifs

On peut tirer les mêmes conclusions qu'avec *TF-IDF*.

Faux negatifs

Les faux négatifs sont cependant plus nombreux qu'avec la featurisation *TF-IDF*. Ils présentent également des différences. On remarque que certains mots longs, qui sont des insultes, sont mal prédits. Par exemple, le modèle n'a pas identifié `motherfucker` comme toxique. Cela révèle deux problèmes : le tokenizer ne sépare pas les mots composés et le modèle *Word2Vec* n'a pas été entraîné sur ces mots plus longs. Un ensemble de données plus grand et un entraînement de *Word2Vec* avec un tokenizer entraîné sur le jeu d'entraînement auraient été nécessaires pour une meilleure compréhension globale de la sémantique.

5.0.3 Régression Multilabel

En raison de la nette différence de performance entre le modèle *Word2Vec* et *TF-IDF*, la featurisation TF-IDF a été choisie pour le modèle *multilabel*. Le prétraitement choisi est *BPE Tokenize 2*, car il a obtenu les meilleurs scores en termes de macro et de F1-score rappel. Un modèle est entraîné par label.

Voici ici le score pour chaque label sans optimsation des hyper-paramètres.

Table 5.6 – Scores de Précision, Rappel, et F1-Score par Label, d'un modèle de Régression Logistique TF-IDF pour chaque label, sur le jeu de test

Label	**Précision**	**Rappel**	**F1-Score**	**Support**
toxic	0.44	0.91	0.59	6090
severe_toxic	0.14	0.87	0.24	367
obscene	0.48	0.87	0.62	3691
threat	0.21	0.78	0.33	211
insult	0.40	0.87	0.55	3427
identity_hate	0.25	0.84	0.38	712
overall_non_toxic	0.99	0.87	0.93	57735
macro avg	0.41	0.86	0.52	72233
weighted avg	0.88	0.88	0.86	72233

Optimisation des hyper-paramètres

Apres optimisation des hyperparmaètre pour chaque label on obtient les résultats suivants :

TABLE 5.7 – Scores de Précision, Rappel, et F1-Score par Label, d'un modèle de Régression Logistique TF-IDF Optimisé pour chaque label, sur le jeu de test

Label	**Précision**	**Rappel**	**F1-Score**	**Support**
toxic	0.42	0.93	0.58	6090
severe_toxic	0.10	0.92	0.19	367
obscene	0.43	0.91	0.59	3691
threat	0.09	0.90	0.16	211
insult	0.37	0.90	0.53	3427
identity_hate	0.17	0.91	0.28	712
overall_non_toxic	0.99	0.84	0.91	57735
macro avg	0.37	0.90	0.46	72233
weighted avg	0.87	0.86	0.84	72233

On remarque que pour le label 'threat', en améliorant le rappel, la précision a nettement diminué. Cela est dû au fait qu'il y a très peu de d'échantillon correspondant à ce label dans le jeu de données. De plus, le modèle de régression logistique utilisant *TF-IDF* ne prend pas en compte la sémantique des mots. En conlusion, le modèle de régression logistique sera efficace pour identifier les éléments toxiques (rappel macro à *0,90*), mais moins efficace pour déterminer précisément le type de toxicité présent (précision macro à *0.37*).

A. Appendix

A.1 Corrélation entre les labels

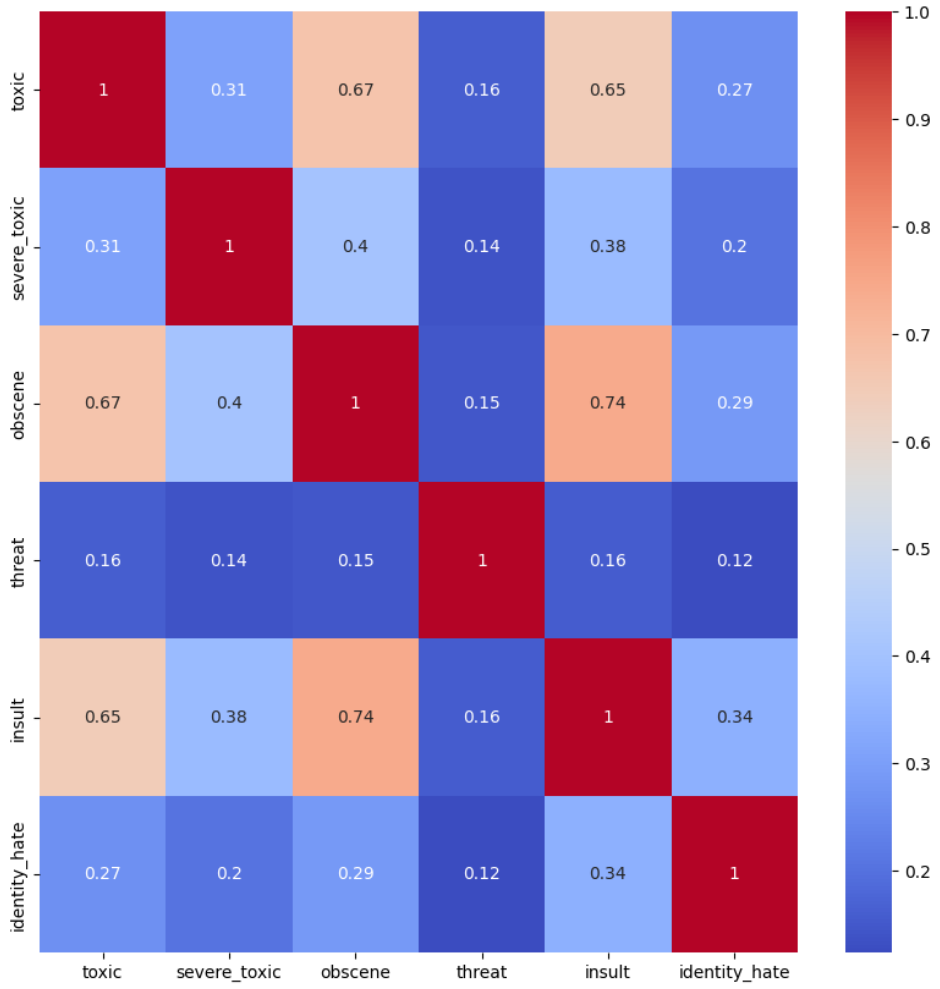

FIGURE A.1 – Matrice de corrélation des labels du dataset

A.2 Distribution des mots

(a) Word Cloud – Toxique

(b) Word Cloud – Haine identitaire

(c) Word Cloud – Obscene

(d) Word Cloud – Insult

FIGURE A.2 – Word Clouds

B. Appendix

B.1 Word Cloud Faux Positifs

FIGURE B.1 – Word Cloud Faux Positifs – Régression Logisitique TF-IDF Optimisé, sur le jeu de test. Pré-traitement : BPE Tokenize 2

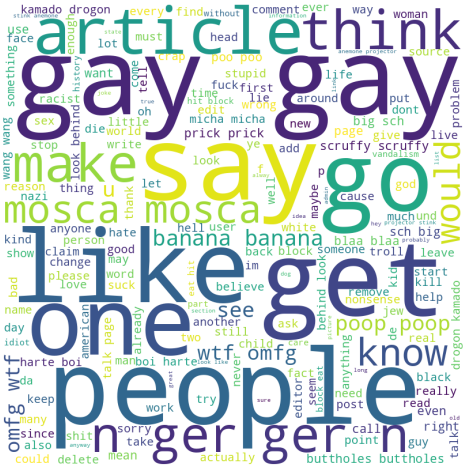

FIGURE B.2 – Word Cloud Faux Positifs – Régression Logisitique TF-IDF Optimisé, sur le jeu de test. Pré-traitement : Word Tokenize 1

B.2 Word Cloud Faux Négatifs

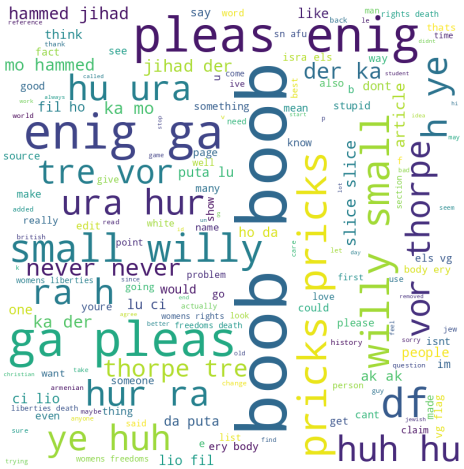

FIGURE B.3 – Word Cloud Faux Négatifs – Régression Logisitique TF-IDF Optimisé, sur le jeu de test. Pré-traitement : BPE Tokenize 2

FIGURE B.4 – Word Cloud Faux Négatifs – Régression Logisitique Word2Vec Optimisé , sur le jeu de test. Pré-traitement : Word Tokenize 1

B.3 Shap Values

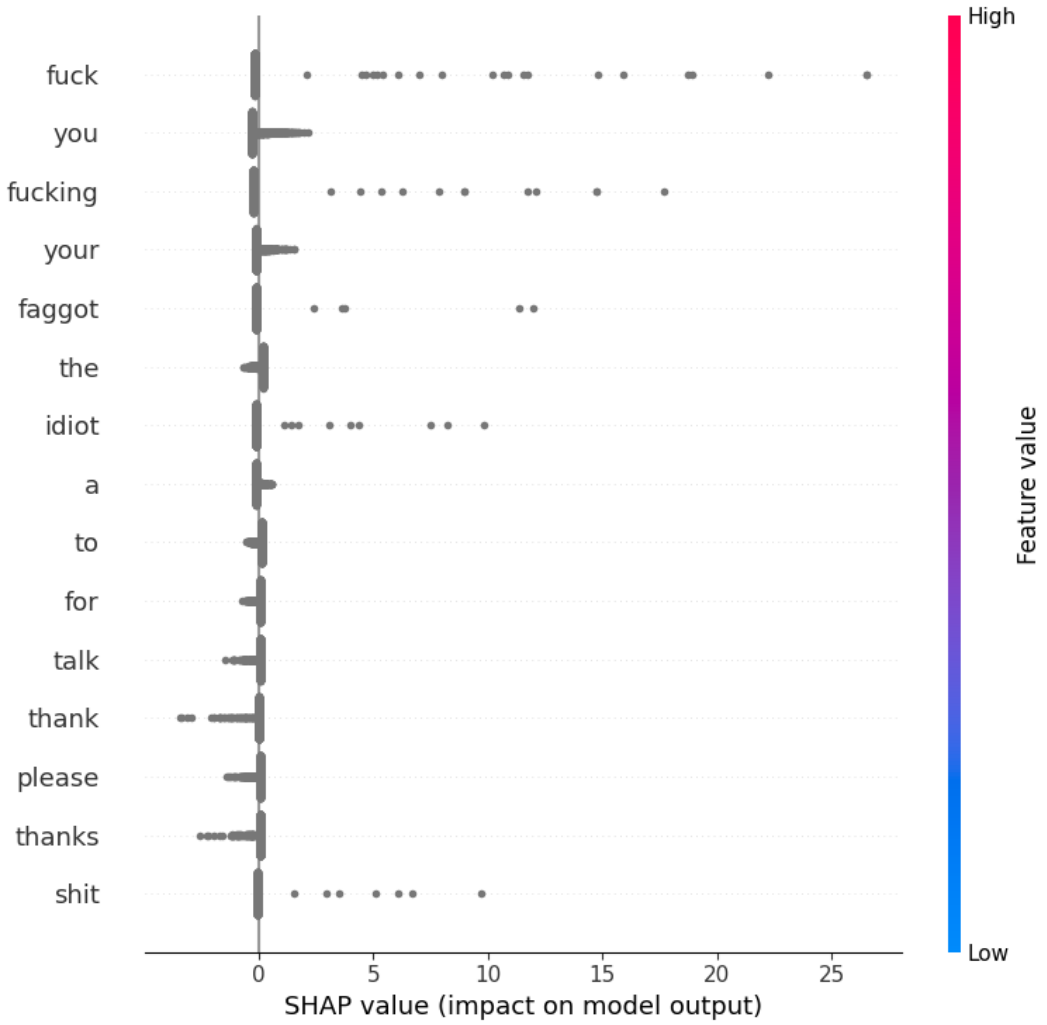

FIGURE B.5 – Shap Values – Régression Logisitique TF-IDF Optimisé, sur le jeu de test. Pré-traitement : BPE Tokenize 2