DataScientest

Projet

Supply Chain - Satisfaction client

Marie-Anne Bigot-Sazy, Myriam Bris, Deborah Cohen

Table des matières

Introduction

Dans un environnement commercial de plus en plus concurrentiel, la satisfaction client est devenue un levier stratégique majeur pour les entreprises. Dans le secteur du e-commerce, où l'interaction client est principalement numérique, les retours d'expérience partagés en ligne jouent un rôle clé dans l'évaluation de la qualité des services proposés.

Ce projet s'inscrit dans cette dynamique d'écoute et d'analyse des besoins des consommateurs. Il vise à exploiter des données sur des avis clients, afin d'identifier les sources d'insatisfaction et de proposer des leviers d'amélioration concrets pour la chaîne d'approvisionnement. Le jeu de données analysé, constitué de près de 20 000 commentaires relatifs à deux entreprises (ShowRoom et VeePee), couvre une période allant de 2015 à 2021. Il se compose principalement de commentaires textuels accompagnés de notes allant de 1 à 5 étoiles. Ces données présentent un intérêt à la fois quantitatif, via l'analyse des notes et de leur distribution, et qualitatif, à travers l'exploration sémantique des commentaires.

Le rapport se compose de deux grandes parties. La première est dédiée à l'exploration des données : elle détaille la composition de la base, les particularités des variables, la qualité des données ainsi que des analyses statistiques et relationnelles. Cette étape permet de définir deux objectifs principaux : (1) prédire automatiquement le niveau de satisfaction d'un client à partir de son commentaire, et (2) identifier les thématiques récurrentes d'insatisfaction à travers une approche non supervisée.

La seconde partie est consacrée à la mise en œuvre de ces deux axes. Plusieurs modèles de classification supervisée ont été testés, parmi lesquels XGBoost, Random Forest, Gradient Boosting ainsi qu'un réseau de neurones dense (MLP), afin de prédire le sentiment global exprimé dans un avis. En parallèle, des techniques de topic modeling telles que LDA, BERTopic ou encore BERT combiné à des méthodes de clustering ont permis de regrouper les commentaires selon les thématiques dominantes qu'ils abordent.

1. Exploration des données et définition des objectifs du projet

Cette première partie est consacrée à l'exploration de la base de données qui permet de définir les premières pistes de travail possibles.

1.1. Exploration des données

1.1.1. Source

La base de données exploitée dans le cadre du projet est issue de la plateforme ouverte Trustpilot, site Web danois, qui permet aux entreprises commerciales et marchandes de collecter des avis, de les analyser et ainsi de renforcer une image de transparence et de fidéliser leurs consommateurs. Cette plateforme permet également aux consommateurs de partager leur expérience et prendre des décisions éclairées sur leurs prochains achats.

Les données nous ont été transmises par DataScientest. Les données de la plateforme Trustpilot ne sont pas disponibles librement. Il est nécessaire de créer un compte sur la plateforme et de fournir des informations permettant de vérifier l'identité.

1.1.2. Composition de la base

Le jeu de données est composé de 19 863 lignes et 12 colonnes, présentées dans ce tableau.

Nom	Description	Type informatique	Distribution valeurs	Taux de NA
Index	id client	index	Valeur unique	0%
Commentaire	Avis du consommateur	object (str)	Valeur unique	0.15%
star	Notation de 1 à 5	float	Quantitative	1.25%
date	Date du commentaire	object (date)	Catégorielle- sup. à 10 catégories	3.13%
client	Pseudonyme client	object (str)	Catégorielle- sup. à 10 catégories	49.60%
reponse	Réponse de l'entreprise au client	object (str)	Valeur unique	57.72%
source	Site de notation	object (str)	Catégorielle- Binaire	1.27%
company	Nom de l'entreprise notée	object (str)	Catégorielle- 3 à 5 catégories	1.27%
ville	Ville d'habitation du client	object (str)	Catégorielle- 3 à 5 catégories	75.72%
maj		object (date)	Catégorielle- sup. à 10 catégories	99.97%

Nom	Description	Type informatique	Distribution valeurs	Taux de NA
date_commande	Date de la commande	object (date)	Catégorielle- sup. à 10 catégories	66.54%
ecart	Écart entre la date de commande et la date du commentaire	float	Quantitative	66.54%

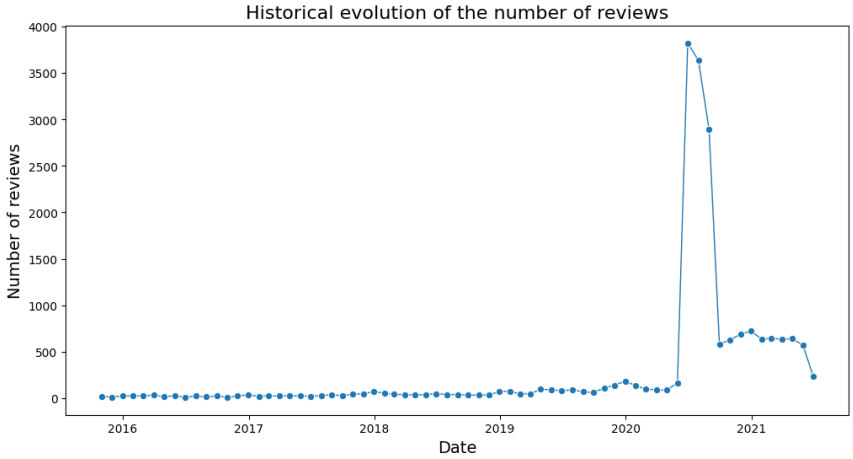
La distribution des variables quantitatives “star” et “ecart” figure ci-dessous. Les modes des variables qualitatives de moins de 5 catégories sont également fournis.

Nom	Type	Distribution des valeurs	Distribution des valeurs - détails
star	float	Quantitative	count : 19863 ;mean : 3,41 ;std : 1,65 ;min : 1 ;0,25 : 1 ;0,5 : 4 ;0,75 : 5 ;max : 5 ; ou qualitative : classe de 1 à 5
company	object	Catégorielle - 3 à 5 catégories	ShowRoom,nan, VeePee
ville	object	Catégorielle - 3 à 5 catégories	TrustPilot, nan, TrustedShop
ecart	float	Quantitative	count : 6686 ;mean : 14,29 ;std : 10,37 ;min : 1 ;0,25 : 9 ;0,5 : 12 ;0,75 : 17 ;max : 304 ;

La variable “star” peut également être interprétée comme variable catégorielle, avec les modes 1,2,3,4 et 5, qui définissent le niveau de satisfaction des clients (la note 5 associée à la plus grande satisfaction du consommateur).

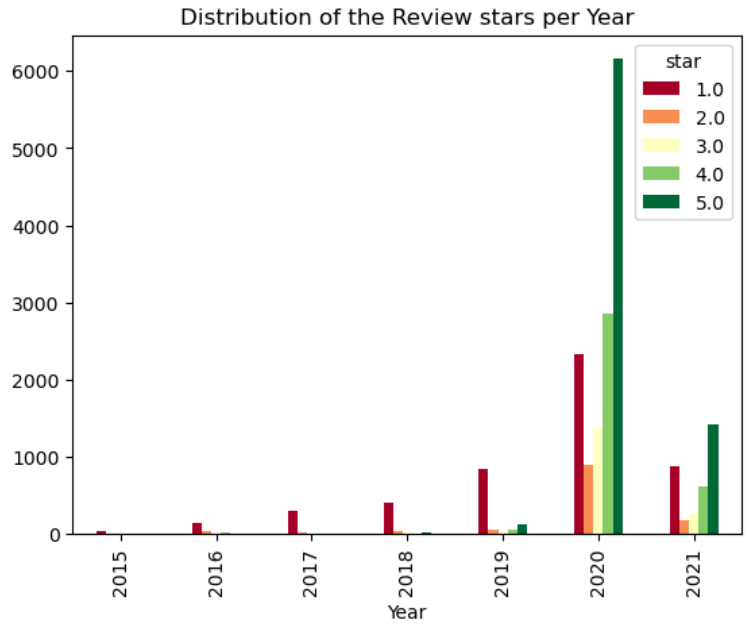
1.1.3. Exploration des attributs et particularités

L’attribut “Date” permet d’évaluer la temporalité des avis en volume et en note, comme illustré sur le graphe ci-après. Les avis sont collectés de 2015 à 2021.



Le nombre d’avis laissés par les consommateurs a soudainement augmenté à l’été 2020 (avec un pic de 3824 avis en juin 2020), correspondant à la période de crise sanitaire de la COVID19. Le nombre d’avis est par la suite redescendu à un niveau bien plus bas mais cependant supérieur à l’avant COVID19 (13 600 avis au total sur 2020 vs. 1123 en 2019, 3381 en 2021).

De plus, à partir de l'année 2019, les consommateurs ont tendance à partager l'ensemble de leur expérience (notes de 1 à 5) alors qu'avant 2019, le partage était principalement négatif (note 1 importante).

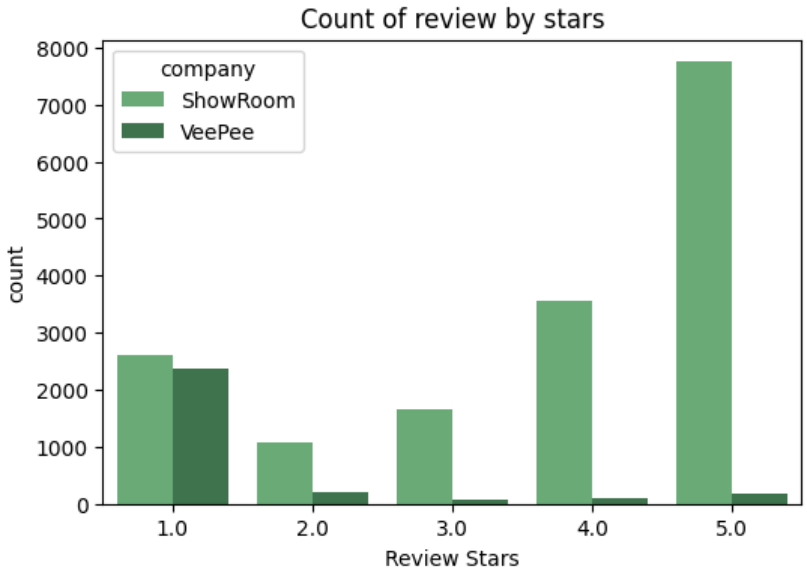


La hausse globale des avis depuis la crise sanitaire s'explique par une montée en popularité des achats en ligne, de plus les entreprises ont accentué leur présence numérique, y compris dans la gestion des avis clients. De façon générale, les consommateurs laissent un avis sur leur achat de manière plus fréquente sur leur achat lors d'une expérience très négative et encore plus lors d'une expérience très positive.

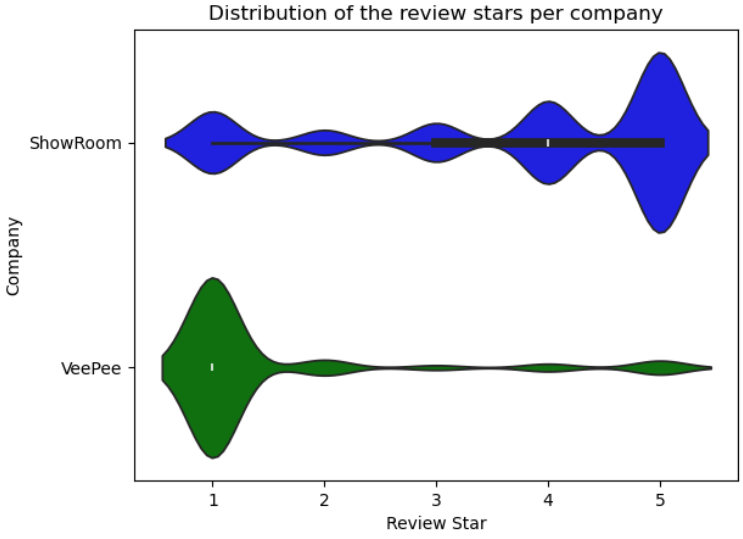
Le graphique suivant représente le nombre de commentaires en fonction de la note. Le nombre d'avis laissés par les consommateurs est plus important pour les notes extrêmes : la plus mauvaise note (1) avec 4959 avis et la meilleure note (5) avec 7748 avis. Pour les notes intermédiaires, l'échantillon d'avis est plus faible : note 2 : 1249 / note 3 : 1683 / note 4 : 3570.

La base de données recense les commentaires issus de deux compagnies : ShowRoom et VeePee. Le graphique suivant représente le nombre d'avis en fonction de la note par entreprise. Deux points sont remarquables :

(1) Le nombre d'avis sur ShowRoom est largement plus important que celui sur Veepee (16294 vs. 2915). Ceci peut être expliqué par la stratégie d'encouragement d'avis adoptée par ShowRoom. Ces derniers proposent également une application très active qui envoie des notifications régulières, ce qui peut aussi encourager les retours d'utilisateurs. Enfin, ShowRoom est actif sur plusieurs canaux pour fidéliser ses utilisateurs, comme des programmes de parrainage. Cela pourrait amener un flux continu d'avis positifs à mesure que les utilisateurs parrainent leurs amis et famille.



(2) Les notes associées à la compagnie VeePee sont plus faibles, comme en témoignent les distributions des notes par compagnie, qui figurent sur le graphe ci-dessous. La note moyenne des avis associés à VeePee est de 1.5, contre 3.8 pour Showroom.



Bien que l'échantillon de données associées à l'entreprise VeePee soit de taille réduite, la nature des avis et des notes s'avère complémentaire à la base données associées à ShowRoom. Aussi la base VeePee couvre une période temporelle plus longue de 2015 à 2021, tandis que la base de données dite ShowRoom est plus ponctuelle : de juin 2019 jusqu'à juin 2021. A ce stade, il est donc décidé de conserver les deux données associées aux deux entreprises.

Enfin, la variable "Commentaire" sous format texte constitue une particularité remarquable de la base de données : son exploitation nécessite la mise en œuvre d'une approche de Text Mining, présentée ci-après.

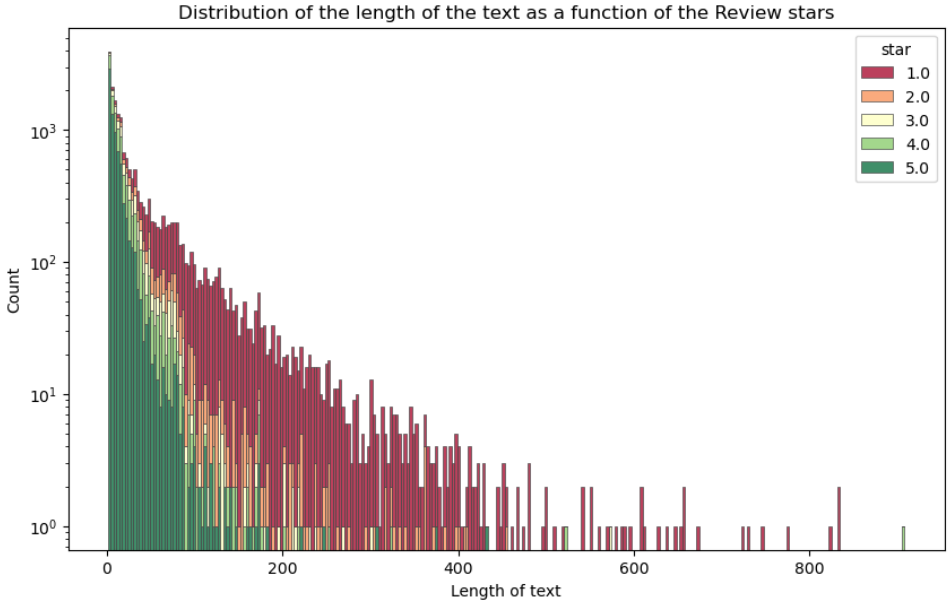
1.1.4. Limitation de la base

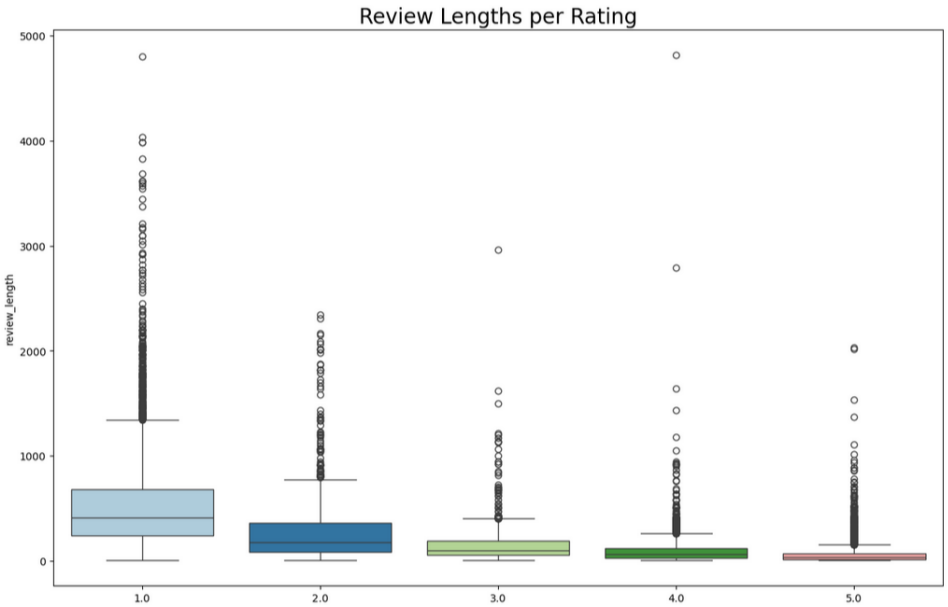
La base de données comprend beaucoup de valeurs manquantes. C'est en particulier le cas des variables "ville", "maj", "reponse", "date_commande" et "ecart", pour lesquelles le taux de NA est supérieur à 50%. Par ailleurs, il est difficile d'introduire une valeur de substitution pour ces variables, pour la plupart qualitatives.

1.2. Relations et tests statistiques

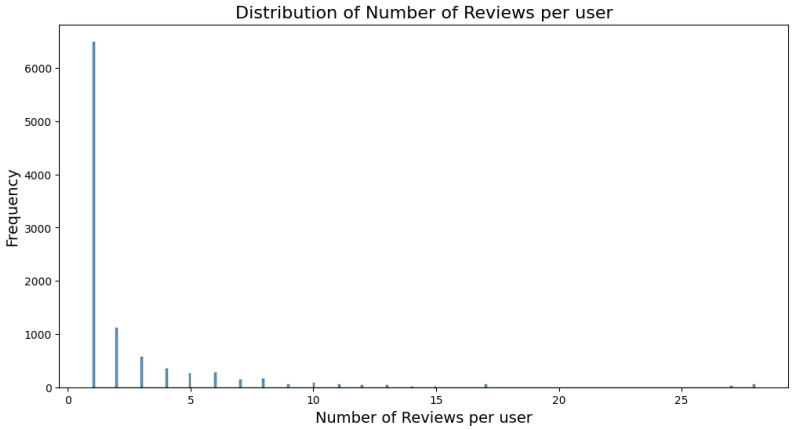
1.2.1. Relations potentielles

La relation entre le commentaire et la note est explorée en première instance. Une attention particulière est d'abord portée à la longueur du commentaire, par un décompte du nombre de mots. D'après la distribution de la longueur du commentaire selon la note, la longueur du commentaire décroît avec la note : moins la note est bonne, plus le commentaire est long. La distribution de la longueur du commentaire selon la note témoigne également de cette tendance, bien que le nombre d' "outliers" semble non négligeable.





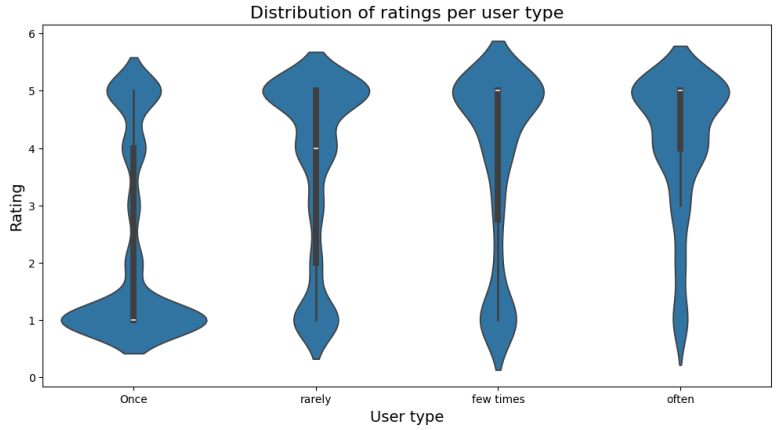
Regardons à présent la fréquence à laquelle les utilisateurs laissent des avis :



La grande majorité des utilisateurs ne donne qu'une seule fois leur avis. D'autre part, il semble exister une corrélation positive entre la fréquence de production d'avis et la note. Plus l'utilisateur laisse d'avis, plus ses avis sont positifs :

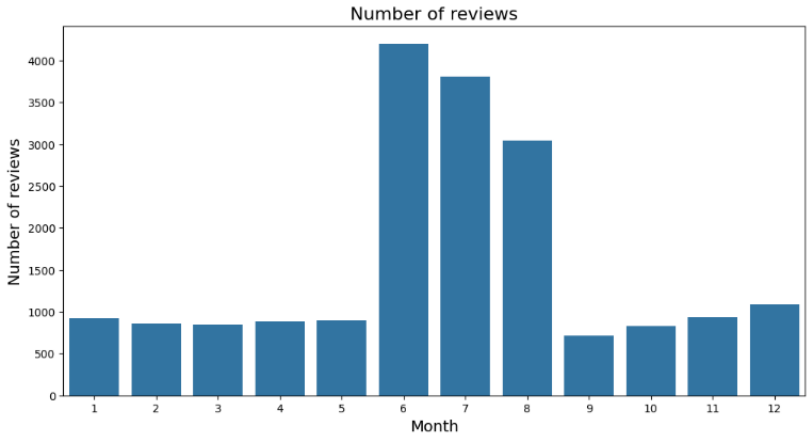
type_user	Rating_Mean
Once	2.301093
Rarely	3.696864
Few times	3.754310
Often	4.085324

Ceci est plus largement visible à travers les distributions des notes par type d'utilisateur :



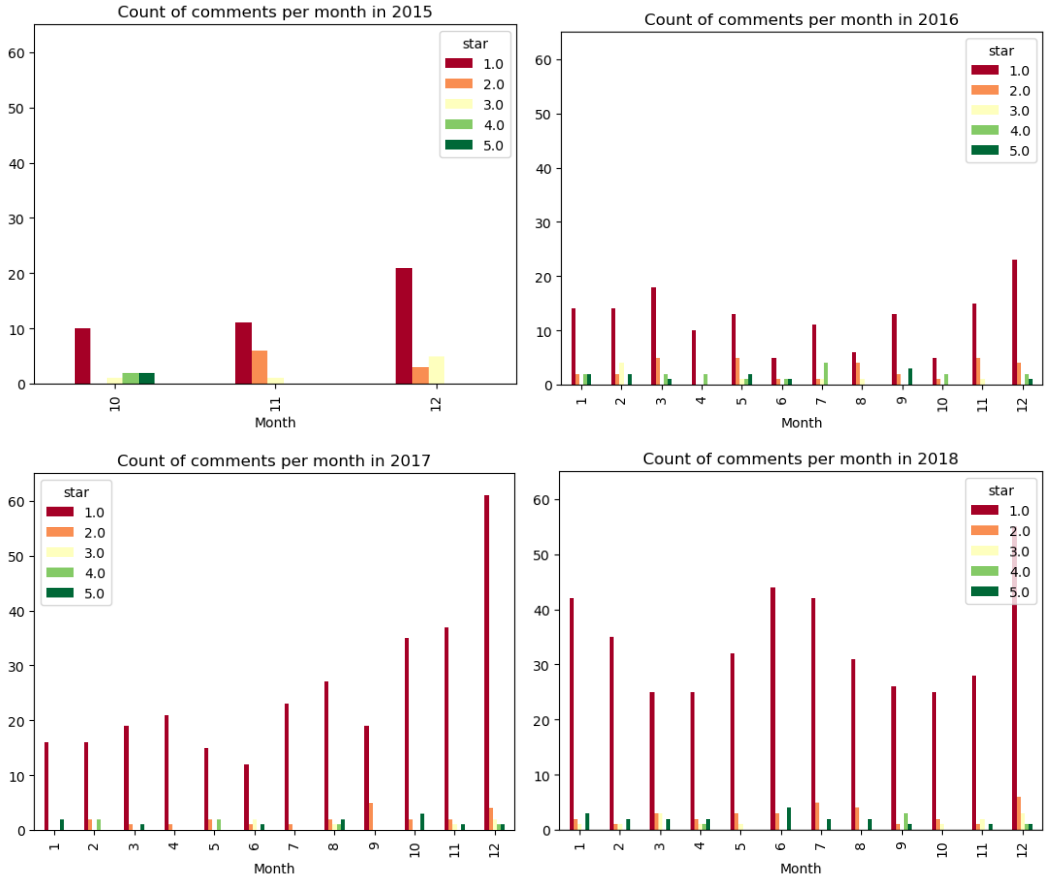
Il est cependant à noter que le nom du client est manquant dans 49% du dataset.

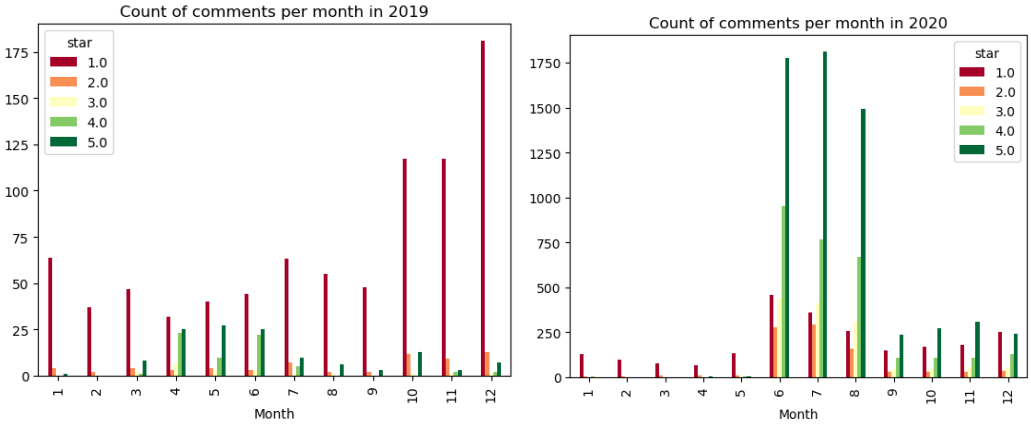
L'influence de la temporalité sur la note est également évaluée, afin de dégager une éventuelle saisonnalité. Le graphique suivant présente le nombre d'avis total par mois.

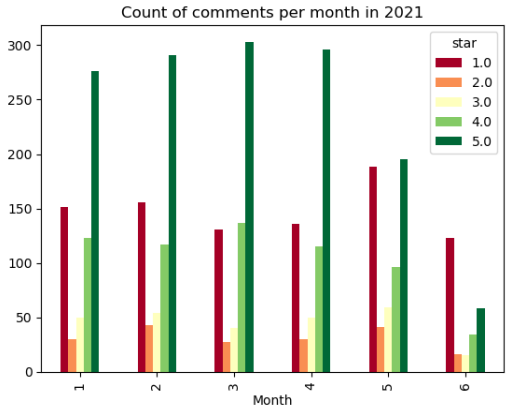


A priori, l'été semble être la période la plus propice au partage de commentaires. Cependant, ce résultat est largement biaisé par le comportement extrême des clients lors de l'été 2020, comme évoqué précédemment. Nous proposons ainsi d'explorer plus en détail la répartition du nombre d'avis par mois sur chaque année.

Le nombre de notes et leurs valeurs sont représentés par mois et par an sur les graphes ci-dessous.



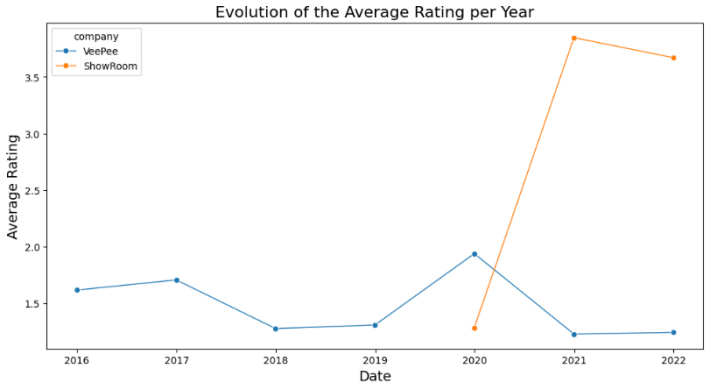




De 2015 à 2019, le nombre d'avis laissés augmente au mois de décembre, en particulier la note 1. En 2020 en revanche, une hausse du nombre de commentaires est remarquable pendant l'été, en particulier des notes positives. Le premier trimestre 2021 est également marqué par une majorité de notes positives, avec un nombre global de notes équivalent à la fin 2020.

Par ailleurs, ces visualisations graphiques confirment qu'avant 2020, les consommateurs laissent une note sur trustiplot principalement négative (notes de 1 et 2) et qu'à partir de 2020, les consommateurs ont plus partagé leur expérience et d'autant plus que cette expérience est positive.

Le graphique suivant présente l'évolution historique des notes par entreprise.



L'augmentation progressive de la satisfaction client concerne en réalité surtout ShowRoom. Cette hausse pourrait refléter une amélioration dans la chaîne de production. En revanche, concernant Veepee, la tendance est plutôt à la baisse (en dehors d'une reprise au moment de la période particulière de la crise sanitaire) ce qui pourrait indiquer des problèmes persistants ou de nouvelles insatisfactions dans la chaîne de production.

Marie-Anne Bigot-Sazy, Myriam Bris, Deborah Cohen

1.2.2. Tests statistiques

Afin de vérifier la relation entre le mois et la valeur de la note en neutralisant les volumes totaux de votes mensuels, un test de student a été réalisé avec l'hypothèse H0 suivante : *la proportion d'une note donnée sur un mois est équivalente au ratio moyen de la même note pour les autres mois*.

Ce test à été réalisé à l'aide de la fonction ttest_1samp du package scipy.stats.
Toutes les notes et tous les mois sont comparés itérativement. Les p-values ne sont pas significativement inférieures au seuil de 5%. L'hypothèse H0 est donc conservée : le ratio d'une note sur un mois donné est équivalent au ratio moyen de la même note sur tous les autres mois.
Un test ANOVA est également réalisé sur les variables mois et note à l'aide des modules statsmodel.api et ols de statsmodels.formula.api, mais les conditions de validité du test ne sont pas vérifiées, en particulier :

- Les variances de chaque groupes ne sont pas équivalentes (test de Levene)
- Les résidus ne sont pas normalement distribués (test d'Agostino et de Pearson)

Par ailleurs, le test non-paramétrique de Kruskal-Wallis (fonction kruspal de la librairie scipy.stats) permet de rejeter l'hypothèse H0 selon laquelle la médiane des notes serait identique sur tous les mois. Un test de Dunn (fonction posthoc_dunn de la librairie scikit_posthocs) révèle que les mois de juin, juillet, août et décembre sont les plus différents. A noter que le test a également été effectué en excluant l'année 2020, et le constat reste le même.
Enfin, nous avons réalisé un test de Kendall, qui ne rejette pas l'absence de relation monotone entre la note et le mois (avec un seuil à 1%). Nous concluons finalement que bien qu'il existe des différences de notes significatives entre les mois, l'influence du mois sur la valeur de la note n'est pas avérée.

Nous avons également étudié l'existence d'une relation entre la variable catégorielle de la note et la variable quantitative de la longueur du commentaire. Nous avons utilisé le test de Spearman avec l'hypothèse H0 selon laquelle il n'existe pas de relation monotone entre ces deux variables. Le test renvoie une p-value inférieure à 5%, l'hypothèse H0 est donc rejetée et nous pouvons donc conclure qu'il existe une relation monotone entre la note et la longueur du commentaire. L'influence de cette variable sur la note sera effectivement évaluée dans le modèle de classification.

La relation entre la note et le nombre d'avis utilisateur a également été testée via le test de Spearman. L'existence d'une relation monotone n'est pas rejetée (p-value < 5%). Cependant, 49% des noms d'utilisateurs étant manquants, il ne sera pas possible d'exploiter ce résultat.

1.3. Pertinence des variables et définition des objectifs

1.3.1. Pertinence des variables

A la lumière de ces premières analyses, les variables les plus pertinentes sont “star” et “Commentaire”. En effet, la variable “star” est identifiée comme étant la variable cible à prédire, et la variable “Commentaire” comporte l’essentiel de l’information à exploiter pour identifier les variables explicatives du modèle.

1.3.2. Définition des objectifs

Sur la base des données disponibles, deux pistes de travail sont identifiées :

- Piste 1 : construire un modèle de Machine Learning basé sur un modèle de classification complexe permettant de prédire la note ou le sentiment perçu par le client (positif ou négatif) à partir de l'analyse du commentaire laissé par le consommateur. La variable cible est donc l’attribut “star” et la variable explicative “Commentaire”.

- Piste 2 : développer un algorithme qui accompagne les entreprises dans l'analyse des avis de leurs consommateurs. Cet algorithme a pour objectif d'identifier les sujets "patterns" fréquemment abordés par les clients et ainsi d'émettre des recommandations auprès des entreprises sur leur supply chain. Les méthodes envisagées sont des méthodes non supervisées appelées "topic modeling". L'un des modèles les plus populaires que nous envisageons est le LDA. La variable commentaire sera exploitée.

Ces deux pistes impliquent un traitement de la variable “Commentaire” qui est présenté dans la partie 2.Pre-Processing.

2. Pre-processing et feature engineering

2.1. Nettoyage de la base

A ce stade de l'analyse, les variables pour lesquelles le taux de NA est supérieur à 50% sont supprimées, à l'exception de la variable "reponse", pour laquelle les valeurs manquantes sont remplacées par "Absence de réponse".
Les variables "Date", "compagny" sont conservées pour information et éventuelle analyse a posteriori. Les lignes en NA sur les variables "Commentaire" et "star" sont supprimées.
Les lignes en doublon sont supprimées.
Bien que l'échantillon de données associées à l'entreprise VeePee soit de taille réduite, la nature des avis et des notes s'avère complémentaire à la base de données associées à ShowRoom. Aussi la base VeePee couvre une période temporelle plus longue de 2015 à 2021, tandis que la base de données dite ShowRoom est plus ponctuelle : de juin 2019 jusqu'à juin 2021. A ce stade, il est donc décidé de conserver les deux données associées aux deux entreprises.
Le tableau suivant résume le traitement des données manquantes et fournit une description plus détaillée de chaque colonne. Les variables traitées étant qualitatives, aucune transformation de normalisation ou de standardisation n'est mise en place.

Nom de la colonne	Gestion des NA
Index	
Commentaire	suppression de la ligne quand la valeur est manquante
star	suppression de la ligne quand la valeur est manquante
date	lignes concernées conservées pour information
client	suppression de la colonne et remplacer par id
reponse	remplacement par "absence de réponse"
source	suppression de la colonne
company	lignes concernées conservées pour information
ville	suppression de la colonne
maj	suppression de la colonne
date_commande	suppression de la colonne
ecart	suppression de la colonne

2.2. Text mining

La variable “Commentaire” fait l’objet de plusieurs traitements. Tout d’abord, elle est traduite en français. En effet, plusieurs langues ont été détectées parmi les commentaires.
Liste ci-dessous non exhaustive des langues détectées à l’aide de la librairie langdetect.

Code	Langue	Nombre commentaires
fr	français	16085
pt	portugais	391
it	italien	377
de	allemand	359
en	anglais	331
es	espagnol	282
cy	gallois	230
id	indonesien	201
ca	catalan	176
ro	roumain	67
lv	letton	47
nl	Néerlandais	43
et	Estonien	27
fi	Finnois	23
lt	Lituanien	21
sk	Slovaque	21
af	Afrikaans	19
vi	Vietnamien	18

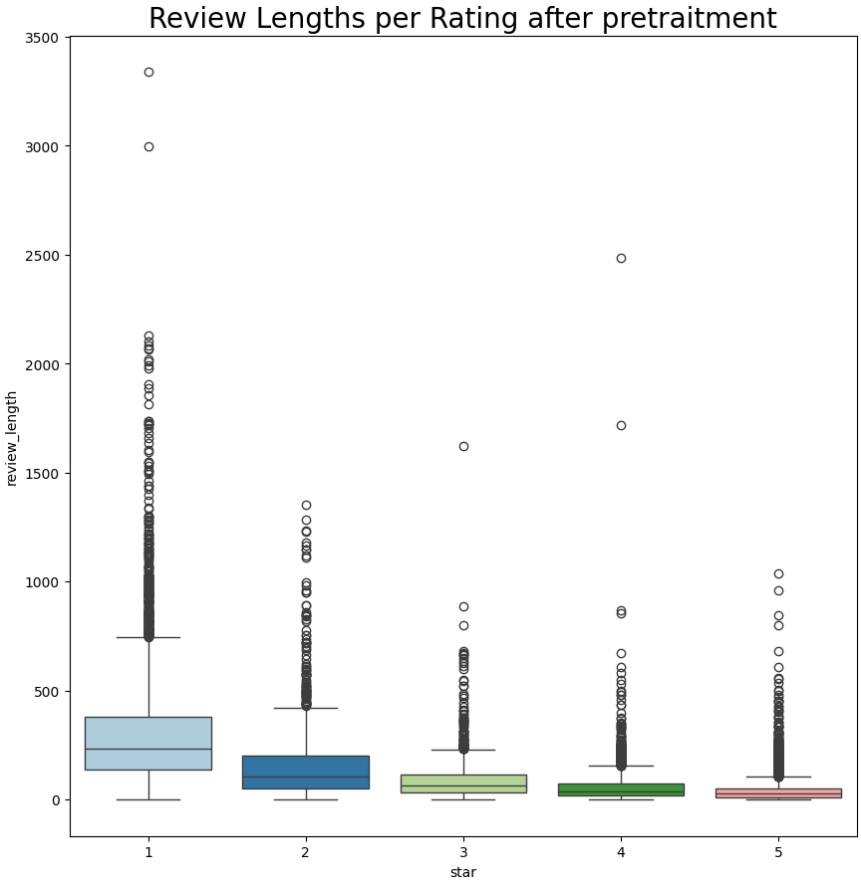
Les commentaires ont été traduits en français en utilisant Translator de la bibliothèque googletrans. On utilise la méthode ROUGE pour évaluer la qualité de la traduction. Nous obtenons un F1-Score élevé : 0.91 ce qui signifie que la traduction est satisfaisante, elle a généré peu d’erreurs.

Les majuscules sont transformées en minuscules. Les chiffres et les espaces multiples sont supprimés à l’aide d’expressions régulières et du package re. Des mots superflus ainsi que de la ponctuation sont supprimés à l’aide de la fonction stopwords du package nltk.

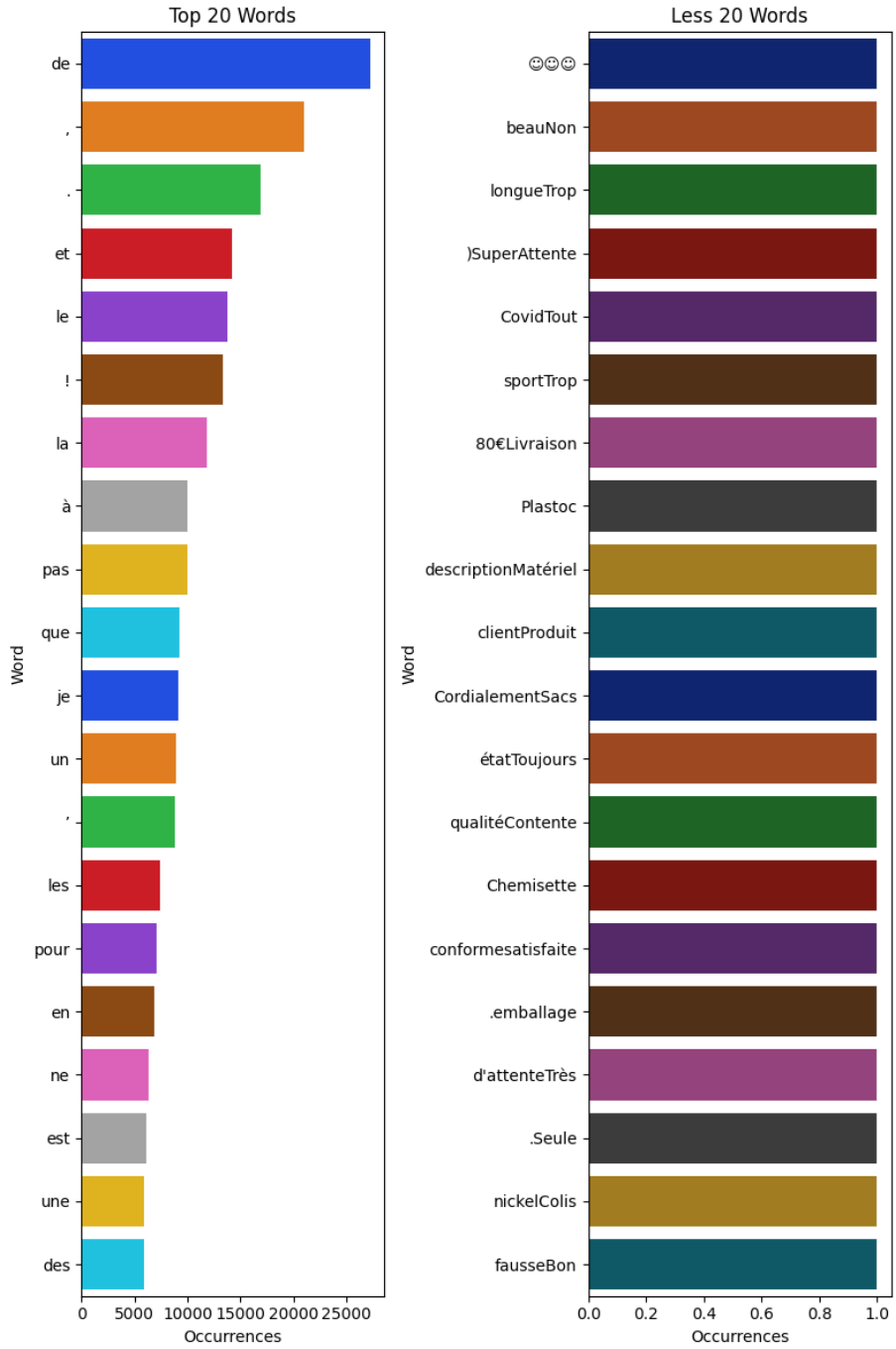
Les commentaires seront par la suite également tokenisés, vectorisés et une normalisation lexicale leur sera appliquée, ceci de différentes manières comme détaillées dans la section suivante.

Ces traitements sont nécessaires aux deux approches proposées au §1..2, i.e. la classification et le topic modeling.

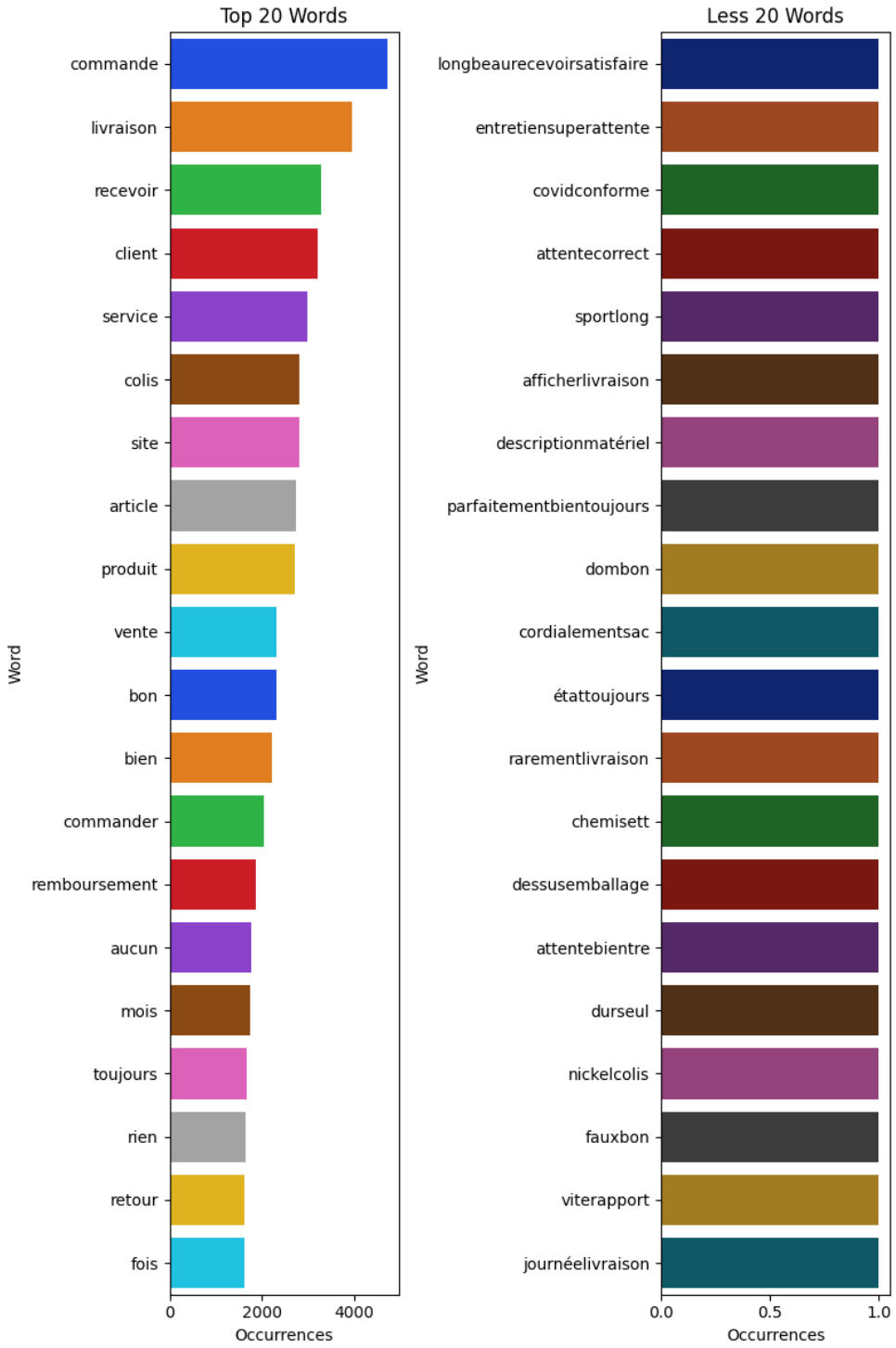
La relation d’ordre entre la taille du commentaire et la note reste similaire avant et après traitement. La longueur du commentaire décroît toujours avec la croissance de la note.


Review Lengths per Rating after pretraitment
review_length
star
0
500
1000
1500
2000
2500
3000
3500
1
2
3
4
5

Nous avons regardé les 20 mots les plus fréquents.
Avant traitement de la cellule commentaire, nous obtenons la liste suivante :



Après traitement de la cellule commentaire (via Spicy pour la normalisation lexicale), nous obtenons la liste suivante :



Le nuage de mots permet également d'avoir un aperçu des mots les plus fréquents:

3. Modélisation

3.1. Modèle de classification

L'objectif de cette partie est de construire un modèle de classification pour prédire la note ou le sentiment du client à partir de la variable commentaire.

3.1.1. Choix du modèle

Dans un premier temps nous travaillons sur la prédiction de la note allant de 1 à 5, puis sur la prédiction du sentiment positif ou négatif du client.
En variables explicatives, nous considérons d'abord le commentaire transformé et vectorisé, puis nous ajouterons la variable correspondant à la taille du commentaire.
Les transformations appliquées à la variable commentaire sont celles mentionnées précédemment dans le paragraphe "Text Mining". Plusieurs méthodes de tokenisation, vectorisation et normalisation lexicale sont testées ici. Nous retiendrons celles offrant la meilleure performance.
Enfin, nous testerons parallèlement plusieurs modèles de classification.

La variables sentiment prend la valeur 1 si la note est égale à 4 ou 5, elle prend la valeur 0 si la note est égale à 1,2 ou 3.

Les différentes transformations testées, de la variable commentaires sont les suivantes :

- Tokenisation
  - Via la fonction word_tokenize du module nltk.tokenize
  - Via la librairie Spacy
- Vectorisation
  - Via la classe *CountVectorizer* du package *sklearn.feature_extraction.text*
  - Via la classe *TfidfVectorizer* du package *sklearn.feature_extraction.text*
- Normalisation lexicale
  - Par racinisation via la classe *PorterStemmer* du package *nltk.stem.snowball*
  - Par lemmatisation via la classe *WordNetLemmatizer* du package *nltk.stem*
  - Via la librairie Spacy
- Filtrage des mots vides via la méthode *stopwords* de la classe *nltk.corpus*

Les différents modèles de classification testés sont les suivants:

- RandomForestClassifier de la librairie sklearn.ensemble : Ensemble d'arbres entraînés indépendamment, la prédiction étant faite par vote majoritaire.
- GradientBoostingClassifier de la libraire sklearn.ensemble : Boosting d'arbres de décision où chaque arbre corrige les erreurs du précédent.
- XGBClassifier de la librairie xgboost : version optimisée du Gradient Boosting avec régularisation et parallélisation.

La répartition des notes et des sentiments étant déséquilibrés nous avons également testé différents types de rééchantillonnage :

- SMOTE de la librairie imblearn.over_sampling
- ClusterCentroids de la librairie mblearn.under_sampling
- BalancedRandomForestClassifier de la librairie imblearn.ensemble

Enfin nous avons tenté de raccourcir le temps d'exécution des modèles en utilisant une PCA.

L'ensemble de données a été divisé en un ensemble d'entraînement et un ensemble de test, à l'aide de la classe train_test_split du package sklearn.model_selection. Nous avons conservé 20% du dataset pour l'ensemble de test. Nous avons également fixé le paramètre random_state à 30.

Nous choisissons la mesure du f1-score pour évaluer la performance des modèles car les données sont déséquilibrées et car nous nous intéressons ici à la performance sur chaque classe. Nous présentons également la mesure de précision qui est également un critère déterminant. Une précision élevée garantit que le modèle identifie correctement les avis positifs, et donc, par extension, qu'il ne classe pas incorrectement des avis négatifs comme positifs. De telles erreurs sont préjudiciables car elles empêchent l'entreprise de prendre connaissance des insatisfactions de ses clients et de mettre en place des actions correctives, ce qui peut conduire à une perte de clientèle.

Les tableaux suivants présentent les résultats obtenus pour la prédiction du sentiment et de la note, en considérant la variable explicative "Commentaire" vectorisée :

Variable cible: Sentiment / Variable explicative: Commentaire	**XGBClassifier**		**GradientBoostingClassifier**		**RandomForestClassifier**	
	précision	**f1-score**	**précision**	**f1-score**	**précision**	**f1-score**
CountVectorizer	0.88	0.88	0.85	0.85	0.87	0.87
TfidfVectorizer	0.88	0.88	0.85	0.85	0.86	0.86
TfidfVectorizer + Racination	0.88	0.88	0.85	0.85	0.86	0.86
TfidfVectorizer + Lemmatisation	0.87	0.87	0.85	0.85	0.86	0.86
TfidfVectorizer + Spacy	0.88	0.88	0.85	0.85	0.86	0.86
TfidfVectorizer + Spacy + Stop words	0.87	0.87	0.84	0.83	0.87	0.87
TfidfVectorizer + Spacy + Stop words + SMOTE	0.87	0.87	0.85	0.84	0.87	0.87
TfidfVectorizer + Spacy + Stop words + ClusterCentroid	0.87	0.87	0.86	0.85	0.83	0.83
TfidfVectorizer + Spacy + Stop words + BalancedRandomForest	0.87	0.87	0.87	0.87	0.87	0.87
TfidfVectorizer + Spacy + Stop words + SMOTE + PCA	0.87	0.86	0.84	0.84	0.85	0.85

Variable cible: Star / Variable explicative: Commentaire	**XGBClassifier**		**GradientBoostingClassifier**		**RandomForestClassifier**	
	précision	**f1-score**	**précision**	**f1-score**	**précision**	**f1-score**
CountVectorizer	0.61	0.63	0.59	0.60	0.60	0.61
TfidfVectorizer	0.61	0.63	0.59	0.60	0.59	0.59
TfidfVectorizer + Racination	0.62	0.63	0.59	0.61	0.65	0.59
TfidfVectorizer + Lemmatisation	0.62	0.64	0.59	0.60	0.59	0.59
TfidfVectorizer + Spacy	0.62	0.64	0.60	0.61	0.59	0.59
TfidfVectorizer + Spacy + Stop words	0.62	0.62	0.57	0.57	0.60	0.60
TfidfVectorizer + Spacy + Stop words + SMOTE	0.62	0.62	0.62	0.60	0.61	0.62
TfidfVectorizer + Spacy + Stop words + ClusterCentroid	0.62	0.60	0.62	0.59	0.62	0.61
TfidfVectorizer + Spacy + Stop words + BalancedRandomForest	0.64	0.62	0.64	0.62	0.64	0.62
TfidfVectorizer + Spacy + Stop words + SMOTE + PCA	0.61	0.61	0.61	0.59	0.59	0.60

Nous observons que les modèles testés sont bien plus performants pour prédire le sentiment que la note. D'autre part, nous notons que le modèle XGBClassifier produit le meilleur f1-score quel que soit l'approche utilisée. En revanche, les différentes méthodes employées pour tokeniser, vectoriser et normaliser renvoient

des performances similaires. A noter également que la réduction de dimension ainsi que le rééchantillonnage ne permettent pas non plus d'améliorer la performance (résultats similaires).

Compte tenu de ces observations, nous proposons de poursuivre avec la variable cible Sentiment, de retenir le modèle XGBClassifier avec les approches suivantes de prétraitements :

- **Tokenisation et normalisation via Spacy**
- **Suppression des stops words**
- **Vectorisation avec TF-IDF**
- **Rééchantillonnage avec SMOTE**

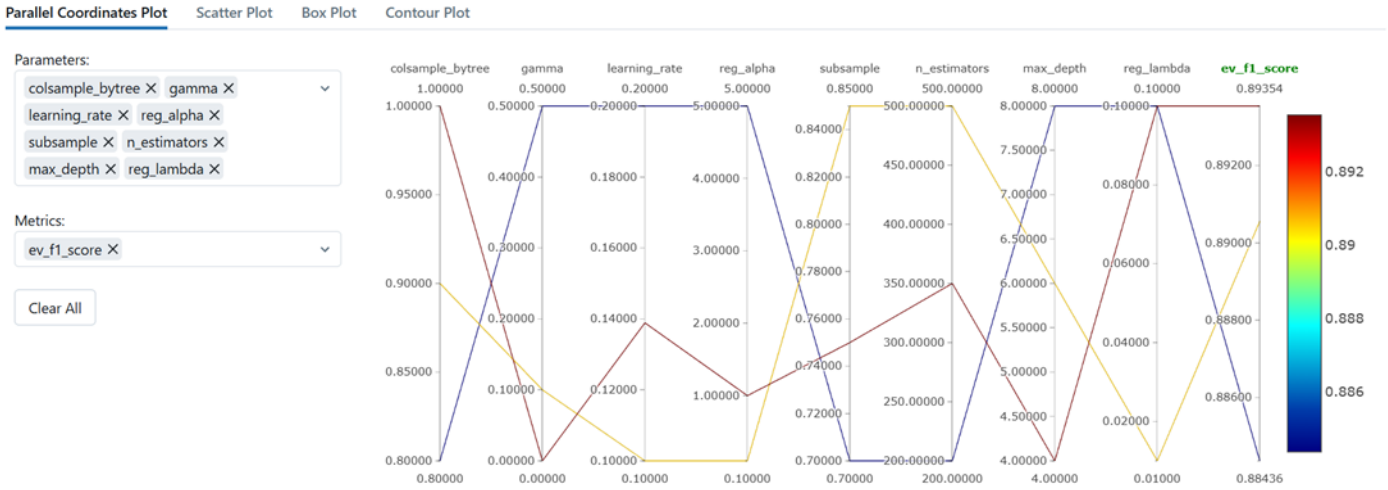
Le modèle a été également testé avec l'ajout de la variable explicative représentant la taille du commentaire. Les résultats obtenus sont similaires à ceux présentés ci-dessus. La variable taille n'apporte donc pas d'information supplémentaire dans la prédiction de la note et du sentiment. Il est donc décidé de ne pas la retenir.

Les hyperparamètres du modèle XGBClassifier ont été optimisés avec la fonction RandomizedSearchCV de la libraire sklearn.model_selection. A noter que RandomizedSearchCV est plus rapide et plus efficace que GridSearchCV lorsqu'il y a un grand nombre de paramètres.
Les meilleurs hyperparamètres trouvés sont les suivants :
{'subsample': 0.75, 'reg_lambda': 0.1, 'reg_alpha': 1.0, 'n_estimators': 350, 'max_depth': 4, 'learning_rate': 0.1388888888888889, 'gamma': 0.0, 'colsample_bytree': 1.0}

Le modèle ainsi construit renvoie un f1-score de 0,89 et une précision de 0,88.

Hyperparamètre	Description
n_estimators	Nombre d'arbres dans le modèle (plus = risque d'overfitting).
learning_rate	Taux d'apprentissage (réduit la contribution de chaque arbre).
max_depth	Profondeur maximale des arbres (plus = risque d'overfitting).
gamma	Seuil de réduction de perte pour ajouter une nouvelle division (plus haut = plus conservateur).
reg_alpha (L1)	Ajoute une pénalité L1 sur les poids des arbres (utile pour la sélection de features).
reg_lambda (L2)	Ajoute une pénalité L2 pour éviter des poids trop grands.
subsample	Pourcentage d'échantillons utilisés par arbre (moins = plus de diversité, mais risque d'underfitting).
colsample_bytree	Ratio de features utilisées pour chaque arbre (utile si dataset avec beaucoup de features).

Deux autres jeux de paramètres ont été testés et ont ete comparés avec le paramétrage ci-dessus via MLFlow. Le premier run adopte une approche conservatrice en augmentant notamment le nombre d'arbres et en réduisant la régularisation :
{"subsample":0.85, "reg_lambda":0.01, "reg_alpha":0.1, "n_estimators":500, "max_depth":6, "learning_rate": 0.1, "gamma": 0.1, "colsample_bytree": 0.9}
Le second run est une approche plus agressive avec notamment plus de régularisation, moins d'arbres, plus de profondeur :
{"subsample": 0.7, "reg_lambda": 10, "reg_alpha": 5, "n_estimators": 200, "max_depth": 8, "learning_rate": 0.2, "gamma": 0.5, "colsample_bytree": 0.8}
Le paramétrage choisi via RandomizedSearchCV reste le meilleur :

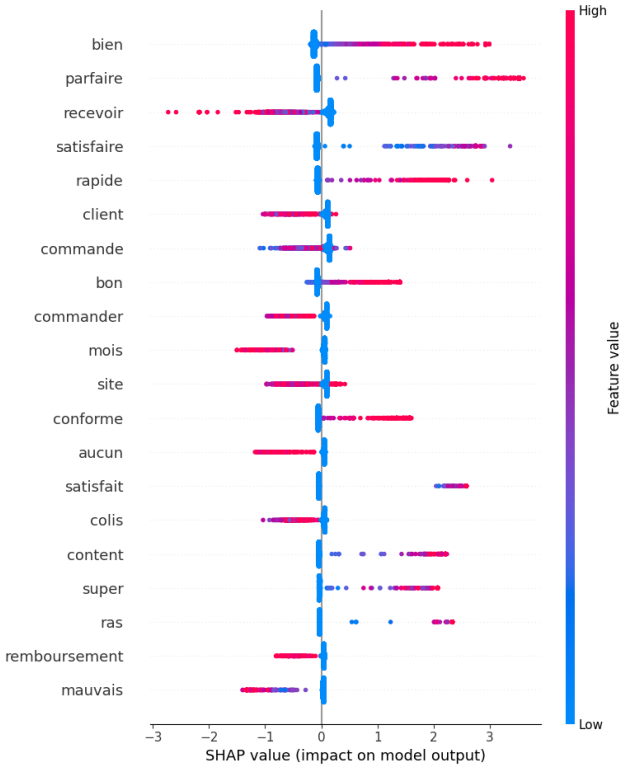


Run Name:	Third_run	first_run_T	Second_run
ev_accuracy	0.861	0.873	0.87
ev_f1_score	0.884	0.894	0.891
ev_precision	0.865	0.879	0.879
ev_recall	0.905	0.909	0.903

3.1.2. Interprétation du modèle de classification

Afin d'interpréter au global les résultats, nous utilisons la librairie Shap. La fonction summary_plot ci-dessous permet de visualiser l'importance des mots dans le modèle de prévision du sentiment. Elle affiche la contribution des variables d'entrée dans la prédiction de chaque observation. Ainsi, les mots présents dans le graphique représentent les 20 mots les plus influents dans la prédiction. Nous pouvons en particulier observer les points suivants :

- L'occurrence élevé des mots « bien », « parfaire », « rapide », « bon », « conforme », contribue à augmenter fortement la prédiction d'un sentiment positif.
- L'occurrence élevé des mots « recevoir », « client », « commander », « mois », « site », « aucun », colis », « remboursement », « mauvais », contribue à réduire fortement la probabilité d'un sentiment positif.
- L'occurrence faible du mot « satisfaire » pousse la prédiction dans une direction négative.

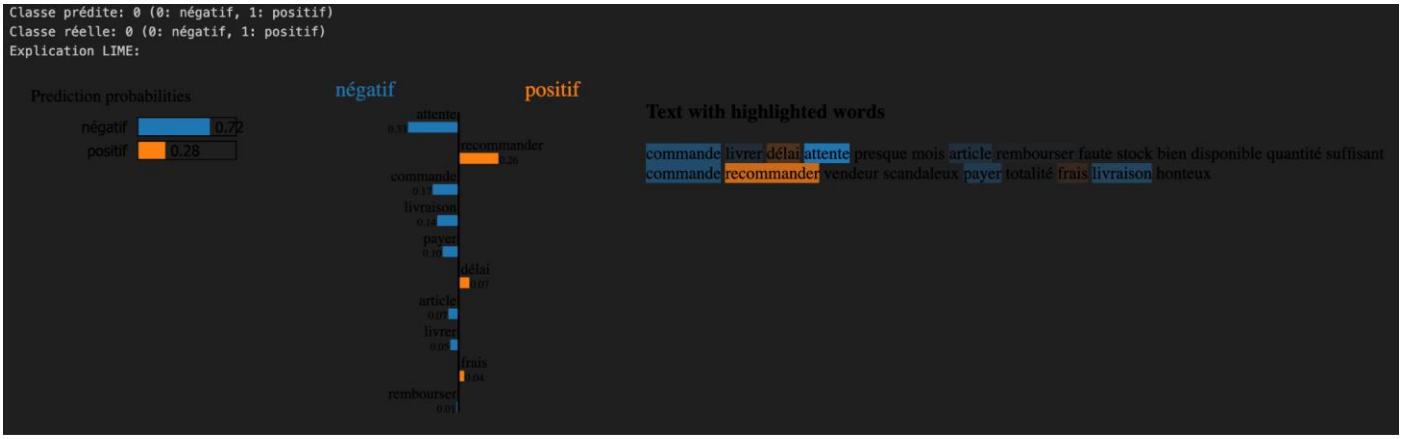


Le modèle de classification des sentiments permet de mettre en évidence que les motifs d'insatisfaction des clients sont liés à plusieurs thématiques :

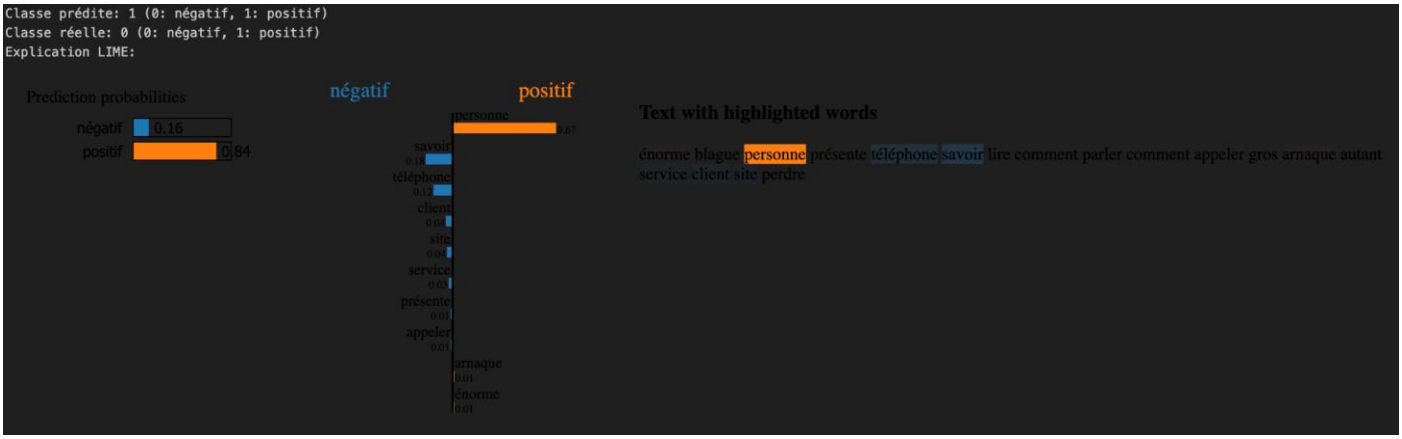
- Le service client, associés aux mots "remboursement", "client", "commande"
- La logistique, dont l'importance est traduite par l'influence du mot "colis"
- L'article : le mot "conformité" renvoie l'importance du produit dans l'expérience d'achat sur le site

Pour analyser le comportement du modèle à l'échelle locale, la librairie LIME est utilisée. Afin d'illustrer les résultats obtenus, prenons l'exemple ci-dessous portant sur trois commentaires.

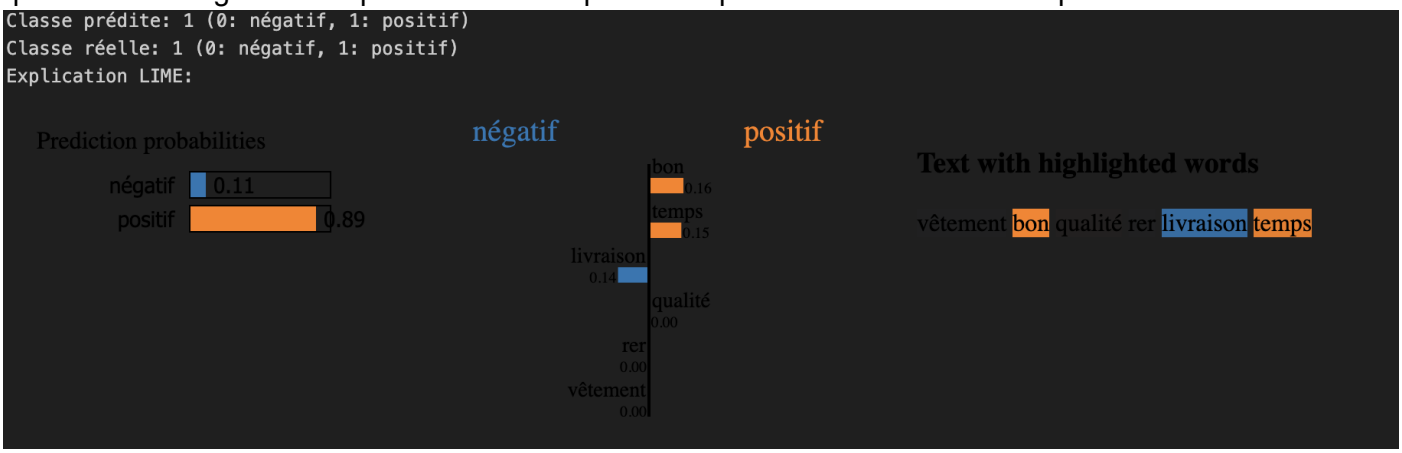
Le premier commentaire analysé est classé correctement par le modèle comme négatif. Les mots les plus influents qui tendent à augmenter la probabilité d'une prédiction négative sont : « attente », « livraison », « commande ». Le mot « recommander » est également important dans la prédiction mais il a en revanche un impact positif sur une prédiction positive.



Le second commentaire est à tort classé comme positif par le modèle, en raison de la forte influence positive du mot "personne", et ce malgré la présence des mots "blague", "arnaque" et "perdre" qui aurait pu influencer plus négativement la note du commentaire.



Le troisième commentaire est classé correctement par le modèle comme positif : les mots les plus influents qui tendent à augmenter la probabilité d'une prédiction positive sont "bon" et "temps"..



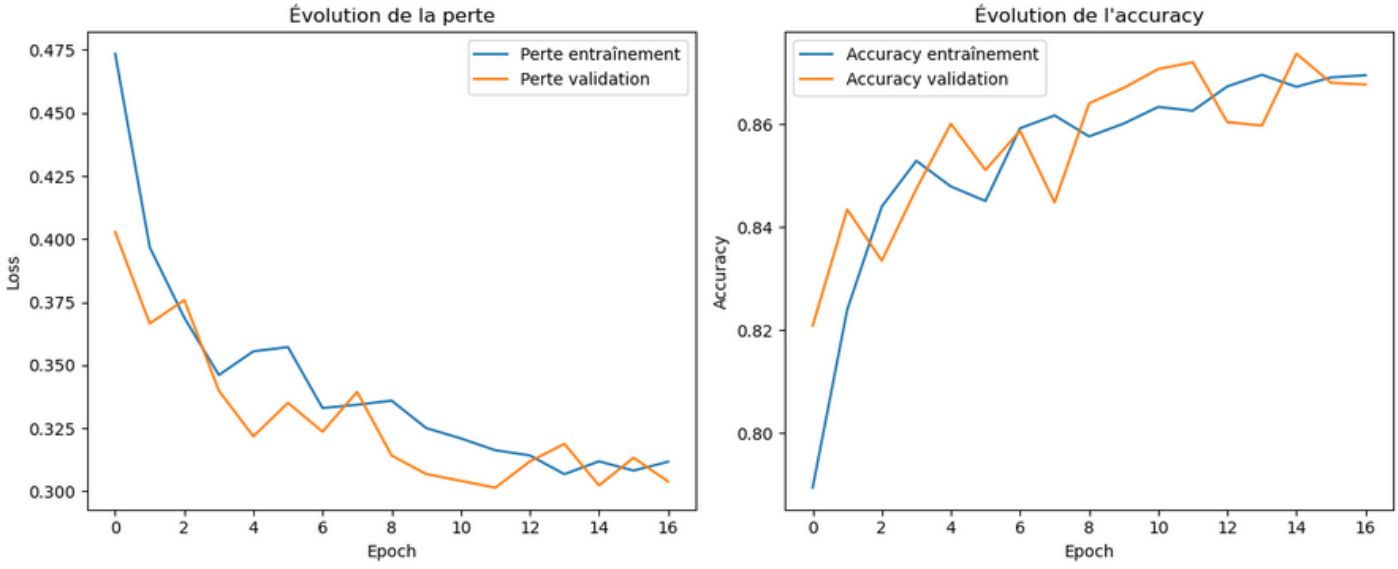
3.2. Classification avec un réseau de neurones - Réseau de neurones dense

Nous avons appliqué la méthodologie suivante :

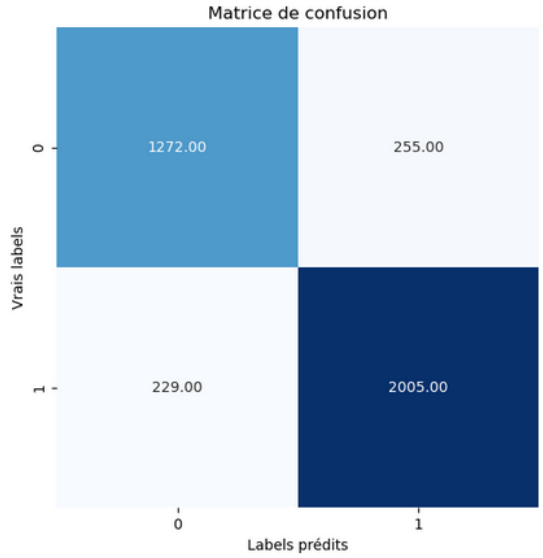
- Prétraitement des données :
  - Les phrases sont encodées à l'aide des modèles CamemBERT et bert-base-multilingual-cased pour obtenir des embeddings vectoriels.
- Préparation des labels :
  - Les avis sont classés en deux catégories : bonnes notes (1) si la note est ≥ 4, sinon mauvaises notes (0) ;
  - Les données sont divisées en ensembles d'entraînement et de test (80% - 20%).
- Construction du modèle :
  - Un réseau de neurones séquentiel est construit avec :
    - Une couche dense de 128 neurones avec activation ReLU ;
    - Une couche de dropout pour régularisation ;
    - Une seconde couche dense de 64 neurones ;
    - Une couche de sortie avec activation sigmoïde pour la classification binaire.
- Entraînement et évaluation :
  - Le modèle est entraîné avec une perte binaire croisée et l'optimiseur Adam (learning_rate=0.001 valeur par defaut) avec 50 epochs et batch size de 128 ;
  - L'évaluation est réalisée sur le jeu de test.

Résultats obtenus :

- La précision du modèle sur le jeu de test est 0.87 ;
- L'évolution de la perte et de l'accuracy sont tracées ;



- Une matrice de confusion et un rapport de classification sont générés pour évaluer la performance en termes de précision, rappel et F1-score.



	precision	recall	f1-score	support
0	0.85	0.83	0.84	1527
1	0.89	0.90	0.89	2234
accuracy			0.87	3761
macro avg	0.87	0.87	0.87	3761
weighted avg	0.87	0.87	0.87	3761

Interprétation :

L'évolution de la perte montre que la perte d'entraînement diminue au fil des époques, ce modèle parvient donc à ajuster ses paramètres pour mieux prédire les étiquettes des données d'entraînement.

La perte de validation baisse globalement et reste relativement proche de la perte d'entraînement. Le modèle généralise de manière satisfaisante les nouvelles données.

L'accuracy d'entraînement augmente progressivement, ce qui est cohérent avec la baisse de la perte : plus le modèle apprend, plus il classe correctement les exemples d'entraînement.

L'accuracy de validation augmente aussi et suit la tendance de l'accuracy d'entraînement. Elle reste relativement proche de la courbe d'entraînement aussi le modèle ne se contente pas de mémoriser les données d'entraînement, mais il généralise sur la validation.

Le modèle a une accuracy de 87%, ce qui indique qu'il classifie correctement la majorité des échantillons :

- Classe 0 (classe négative) :

255 faux positifs : des échantillons réels 0 que le modèle classe à tort en 1.
1272 vrais négatifs : correctement classés en 0.
Précision à ~0.85 et rappel ~0.83 : Le modèle a un peu plus de mal avec la classe 0 qu'avec la classe 1, mais reste performant.

- Classe 1 (classe positive) :

229 faux négatifs : des échantillons réels 1 que le modèle classe à tort en 0.
2005 vrais positifs : correctement classés en 1.
Précision à ~0.89 et rappel à ~0.90 : Le modèle reconnaît bien cette classe.

Pour conclure, le modèle est robuste (87% d'accuracy) et différencie correctement les deux classes, avec une performance légèrement meilleure sur la classe 1.

Nous avons ajouté un nouveau commentaire auquel nous avons appliqué l'étape de preprocessing : "Le produit a été livré cassé, c'est inadmissible".

Nous avons obtenu :

- Probabilité d'avis positif : 0.14 ;

Prédiction : Avis négatif.

Par ailleurs, nous avons comparé trois architectures de modèles avec MLflow :

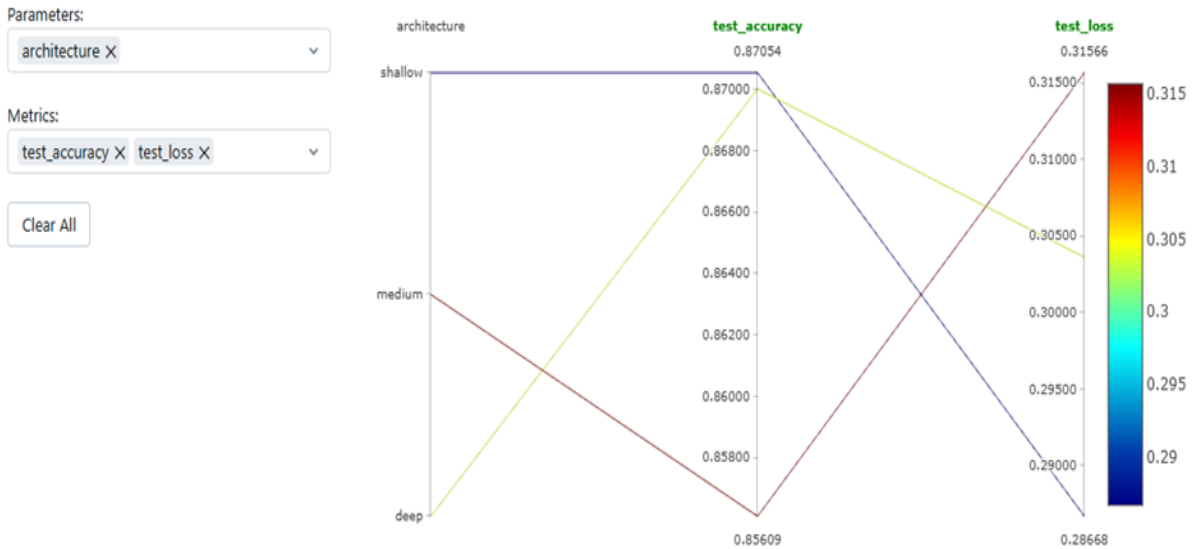
- Un réseau profond :
    - Une couche dense avec 512 neurones et activation relu
    - Une couche dense avec 256 neurones et activation relu
    - Une couche dense avec 128 neurones et activation relu
    - Une couche dense avec 64 neurones et activation relu
    - Une couche de sortie avec 1 neurone et activation sigmoid
- Un réseau intermédiaire :
    - Une couche dense avec 256 neurones et activation relu
    - Une couche dense avec 128 neurones et activation relu
    - Une couche dense avec 64 neurones et activation relu
    - Une couche de sortie avec 1 neurone et activation sigmoid
- Un réseau peu profond :
    - Une couche dense avec 128 neurones et activation relu
    - Une couche dense avec 64 neurones et activation relu
    - Une couche de sortie avec 1 neurone et activation sigmoid.

Nous avons considéré deux métriques : la précision sur les données de test (test_accuracy) et l'erreur sur les données de test (test_loss). La comparaison de ces architectures est présentée dans le graphique ci-dessous. L'echelle de couleur correspond ala valeur de test_loss, avec la couleur bleu pour une erreur faible donc un bon modèle et rouge pour une erreur elevee donc un moins bon modèle. Nous obtenons le meilleur modèle (en bleu) avec un réseau de type shallow, avec :

- test_accuracy ≈ 0.87 ;
- test_loss ≈ 0.29.

Le modèle deep est le moins performant, avec :

- test_accuracy plus basse (~0.86) ;
- test_loss plus élevée (~0.32).



3.3. Modèle de topic-modeling

L'objectif de cette partie est de construire un modèle de topic-modeling pour distinguer les thèmes associés à l'insatisfaction des clients. Les commentaires associés à une note inférieure ou égale à 2 sont analysés au moyen de plusieurs approches.

Nous avons considéré deux types d'apprentissage non supervisé et auto-supervisé :

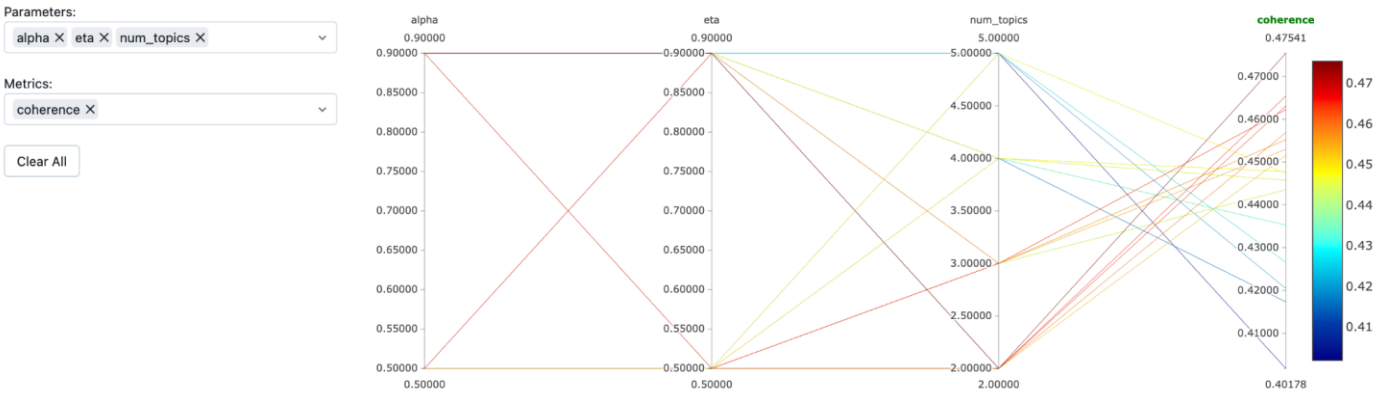
- LDA (Latent Dirichlet Allocation) ;
- BERT (Bidirectional Encoder Representations from Transformers).

3.3.1. Algorithme LDA

L'algorithme LDA est dans un premier temps mis en œuvre pour identifier les thèmes qui sous-tendent l'insatisfaction des clients. La fonction LDAmodel du module gensim.models est d'abord entraînée sur le corpus composé des mots tokénisés et lemmatisés des commentaires. Le nombre d'itérations d'entraînement du modèle est fixé à passes = 30, pour permettre la convergence du modèle tout en limitant l'overfitting. Le nombre de topics, et les hyperparamètres alpha et eta du modèle sont optimisés sur les intervalles suivantes :

- num_topics : 2 à 5
- alpha : [0.5, 0.9] - Alpha contrôle la répartition/densité des sujets dans les commentaires
- eta : [0.5, 0.9] - Eta contrôle la répartition/densité des mots dans les sujets

Les valeurs des hyperparamètres sont prédéfinies plutôt proches de 1 pour ne pas aboutir à des topics de trop petite taille. L'optimisation des hyperparamètres est contrôlée via la mesure de cohérence C_v, qui se concentre sur la similarité sémantique entre les mots les plus probables d'un sujet. Au lieu de simplement compter les cooccurrences de mots dans les documents, C_v utilise des mesures de similarité basées sur des corpus externes. La métrique cohérence C_v est évaluée selon divers jeux d'hyperparamètres différents via MLflow.



L'optimisation du modèle aboutit au jeu de paramètres suivant :

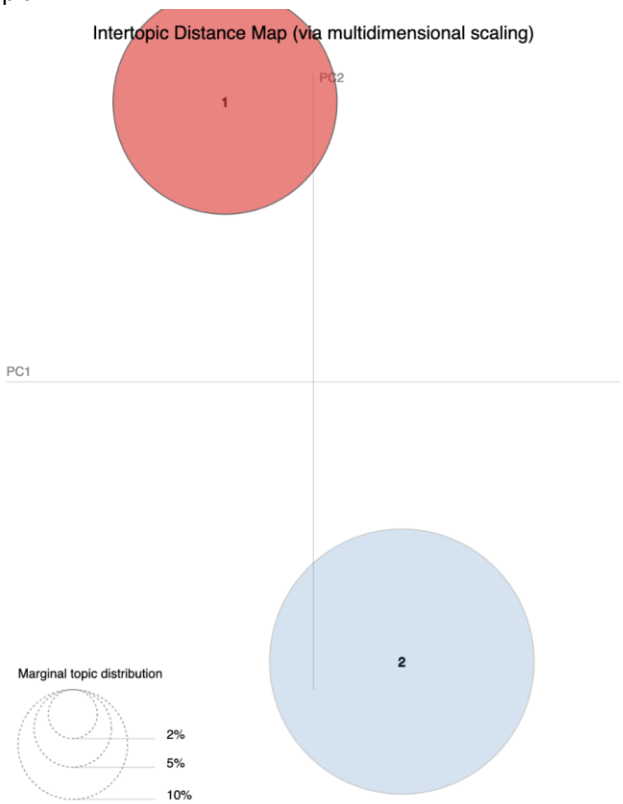
- num_topics : 2
- alpha : 0.9 *Forte densité des sujets dans les commentaires*
- eta : 0.9 *Forte densité de mots par sujet*

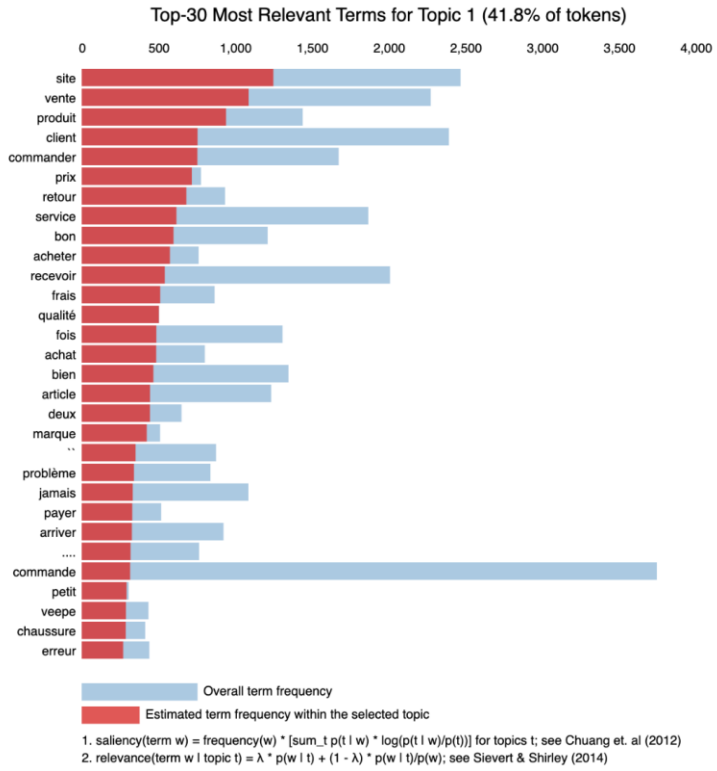
Les résultats sont visualisés graphiquement à l'aide de la librairie pyLDAvis sur la figure ci-dessous.

Les deux topics sont opposés par l'axe des abscisses et ne se superposent pas, témoignant de leur distinction. Les tailles des topics 1 et 2 sont à peu près équivalentes, respectivement 42% et 58%.

Le premier topic semble principalement concerner les problèmes liés à l'expérience client et la qualité du produit d'achat. Les mots les plus significatifs incluent "site", "vente", "client", "commander","prix","service", qui sont plutôt associés à l'expérience d'achat. Les mots prédominants tels que "produits", "article", "qualité", "chaussure", "petit" font référence à l'objet de l'achat et à ses caractéristiques (taille, qualité).

Topic 1





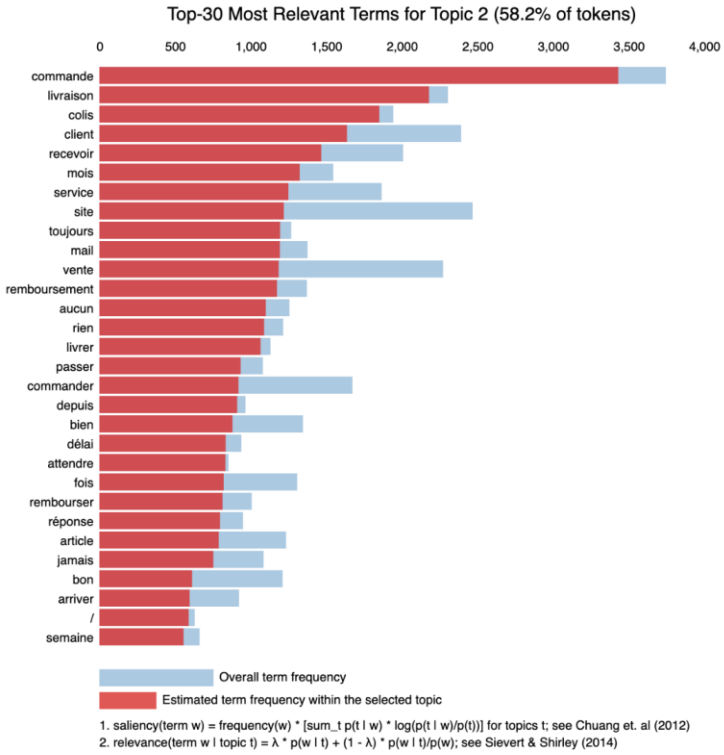
Ce topic met en évidence des problèmes récurrents liés

- à l'expérience client au moment de la commande
- à l'objet de la commande, en termes de conformité ou de qualité du produit

Mots principaux du topic 1

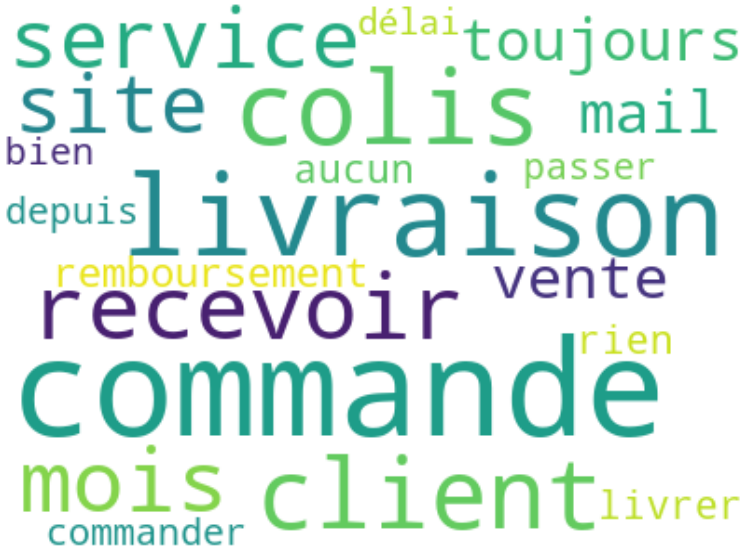
Le second topic semble axé sur la livraison et le service après-vente. La thématique logistique est signalée par la forte occurrence des mots "livraison", "colis", "recevoir", “livrer”. Les mots “client”, “remboursement”, “mail”, “rembourser” font référence au SAV. Les mots “mois”, “délais”, “attendre”,”semaine”,etc. suggèrent de longs délais de livraison ou du processus service après-vente.

Topic 2



Ce topic regroupe toutes les frustrations liées à la chaîne logistique : retards de livraison, erreurs d’expédition, manque de transparence sur le suivi des colis, et problèmes de stock. Contrairement au premier topic, focalisé sur l’acte d’achat et son objet (le produit), le second topic concentre les critiques relatives à la logistique et au service après-vente.

Principaux mots du topic 2



3.3.2. Algorithme BERT

L'algorithme BERT seul ne permet pas d'organiser des textes en groupes sémantiques automatiquement. Il fournit seulement une représentation numérique qui peut être utilisée pour mesurer la similarité entre les textes. Pour identifier des topics ou des clusters à proprement dit, il faut appliquer une méthode de regroupement sur les embeddings en sortie BERT. Dans cette section, nous effectuons une approche non supervisée pour découvrir les thèmes du corpus de commentaires. Pour cela, nous utilisons BERTopic pour générer des **re**présentations vectorielles (embeddings) des textes et et qui applique un algorithme de clustering pour regrouper les textes similaires. Nous avons appliqué BERTopic sur la colonne des commentaires pré-processés (mots tokénisés, lemmatisés et traduits en français). BERT est un algorithme plus performant que LDA pour les tâches qui demande une compréhension du langage.

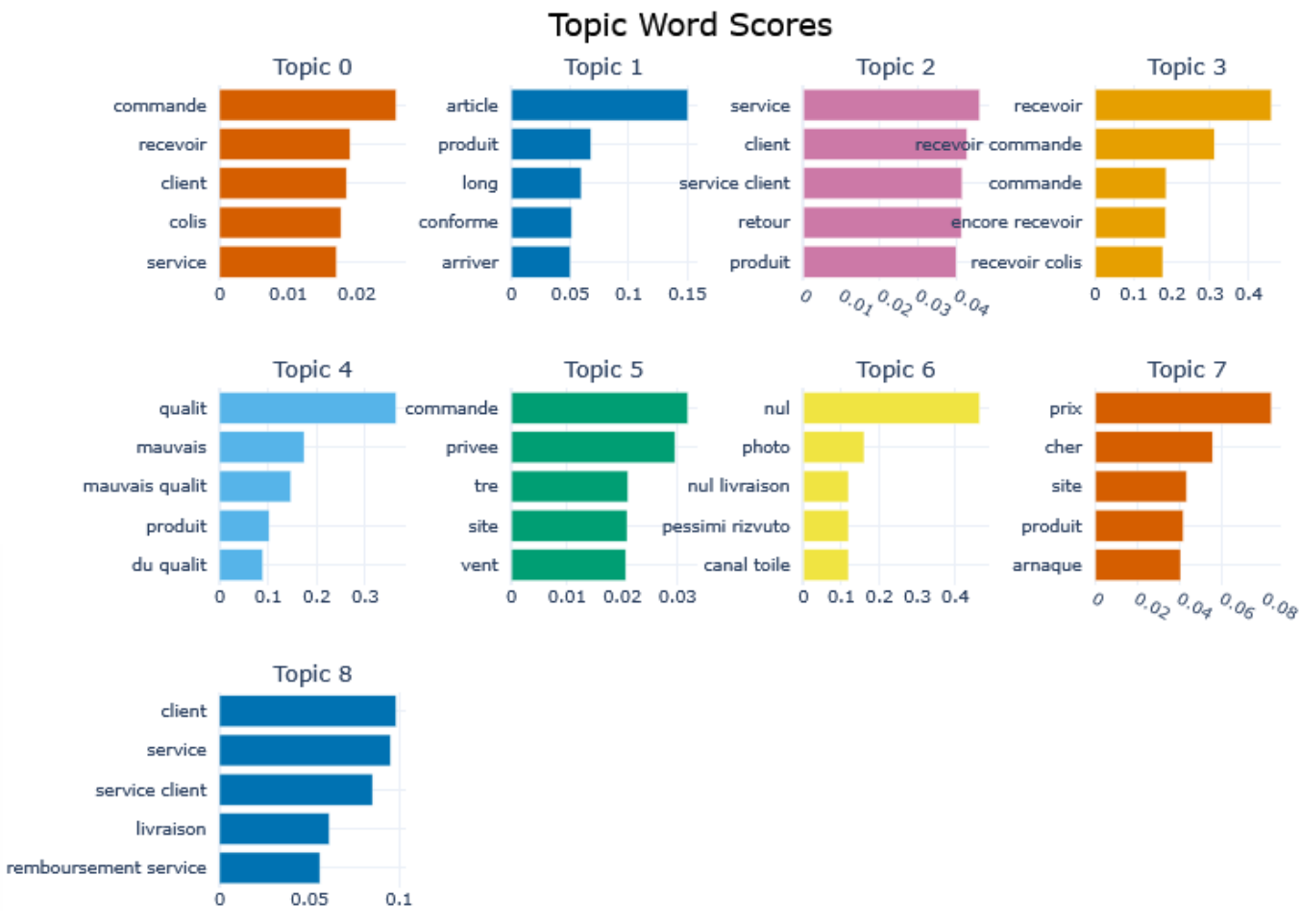
-

Pour améliorer la robustesse de notre modèle, nous avons utilisé deux modèles : CamemBERT qui est pré-entraîné spécifiquement pour la langue français et BERT Multilingual (bert-base-multilingual-cased). Les embeddings des deux modèles sont concaténés avant d'être utilisés dans BERTopic.

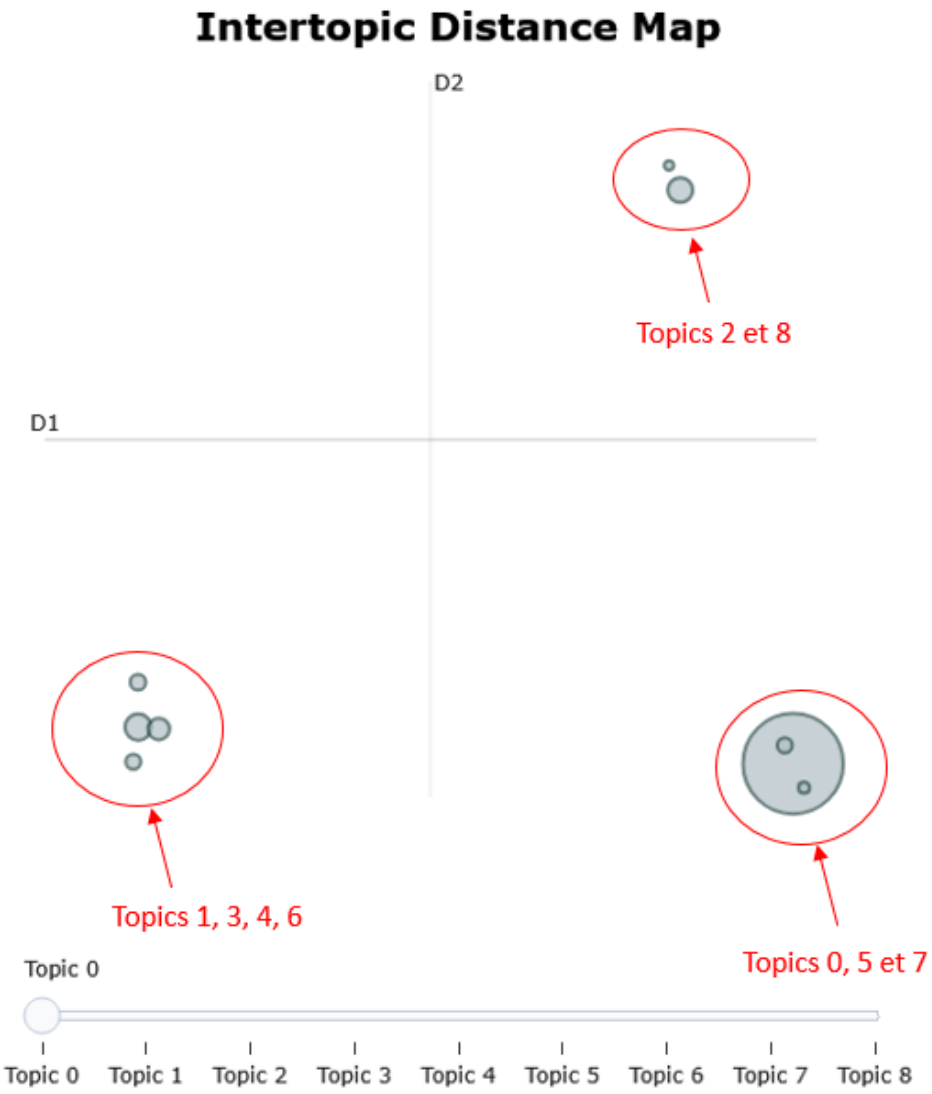
Afin d'obtenir le meilleur modèle BERTopic pour nos données, nous avons appliqué un RandomizedSearchCV avec les valeurs suivantes des paramètres :

- Nombre de topics à extraire : "nr_topics": ["auto", 5, 10, 15] ;
- Taille minimale des topics : "min_topic_size": [5, 10, 15] ;
- Nombre de mots-clés affichés : "top_n_words": [5, 10, 15] ;
- Utilisation d'unigramme et bigramme : "n_gram_range": [(1, 1), (1, 2)].

L'algorithme sélectionne de façon aléatoire 10 combinaisons de paramètres, entraîne BERTopic et évalue les performances de chaque configuration de paramètres selon le score de cohérence. Nous présentons ci-dessous les résultats obtenus avec le jeu de paramètres optimal. Le graphique suivant présente les mots clés dominants par topic. Les mots "commande" et "recevoir" figurent dans plusieurs topics.



Nous utilisons la méthode visualize_topics() de BERTopic qui permet d'analyser les relations entre les topics. Nous obtenons un modèle qui produit trois regroupements bien séparés.



Nous observons des topics proches donc similaires en termes de contenu. Aussi, les topics 0, 5 et 7 se chevauchent (en bas à droite). Le topic 0 présente la plus grande fréquence (78%) par rapport aux autres topics. Les informations associées portent sur :

- Commandes et service client avec les mots clés : "commande", "recevoir", "client", "colis", "service" qui concernent les commandes en ligne et le service client associé à la réception des colis ;

- Privatisation et commerce en ligne. Les mots clés : "commande", "prive", "site", "vente" abordent des aspects liés aux commandes et aux plateformes de vente en ligne ;

- Prix et arnaques. Les mots clés : "prix", "cher", "site", "produit", "arnaque" évoquent les préoccupations des clients concernant les prix élevés et les potentielles arnaques.

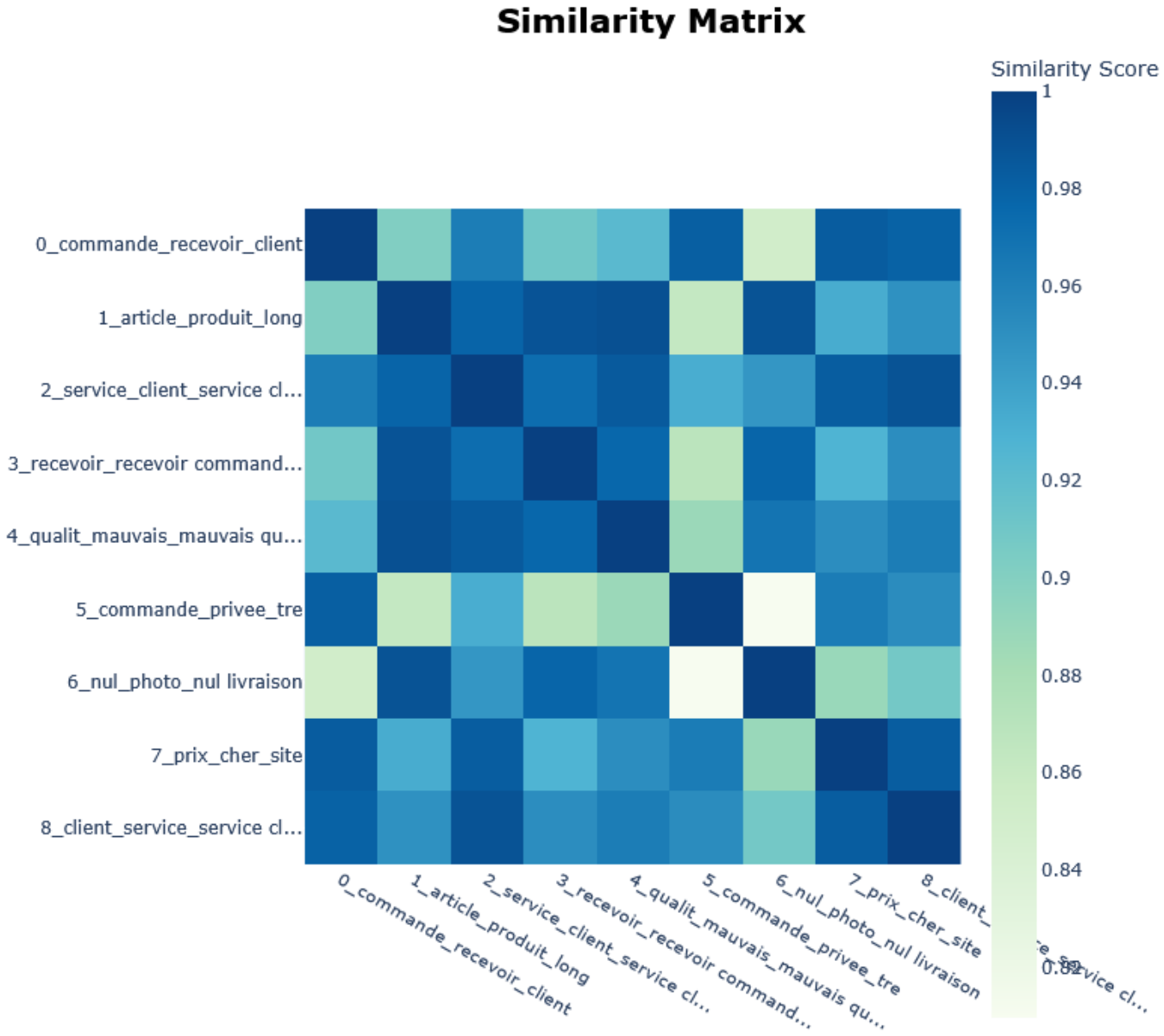
Ces topics mettent en exergue les problématiques liées aux commandes, aux prix et à la fiabilité des plateformes Showroom et Vente Privée de vente en ligne. L'optimisation de la transparence, du service client et des politiques de prix serait essentielle pour améliorer l'expérience client.

Les topics 0, 5 et 7 sont opposés des topics 2 et 8 par rapport à l'axe des abscisses ce qui signifie qu'ils n'ont pas de lien thématique. Ces topics (2 et 8) ont une fréquence de 6%, leurs mots clés portent particulièrement sur le service client, les retours et les remboursements. Les frustrations des clients semblent liées à des commandes non conformes, des problèmes de livraison et des délais de remboursement trop longs. Une

meilleure gestion des retours, une politique de remboursement plus fluide et un service client plus réactif peuvent améliorer la satisfaction.

Les topics 0,5 et 7 sont opposés par rapport à l'axe des ordonnées aux topics 1, 3, 4 et 6 (fréquence 13%). Ces derniers couvrent plusieurs thématiques problématiques du e-commerce : la qualité des produits, des problèmes de réception des commandes et une insatisfaction globale. Il semble pertinent d'améliorer le contrôle qualité, la communication sur les livraisons et la gestion des attentes clients pour limiter l'insatisfaction.

Une autre représentation de la qualité et de la similarité des topics est donnée par la heatmap.



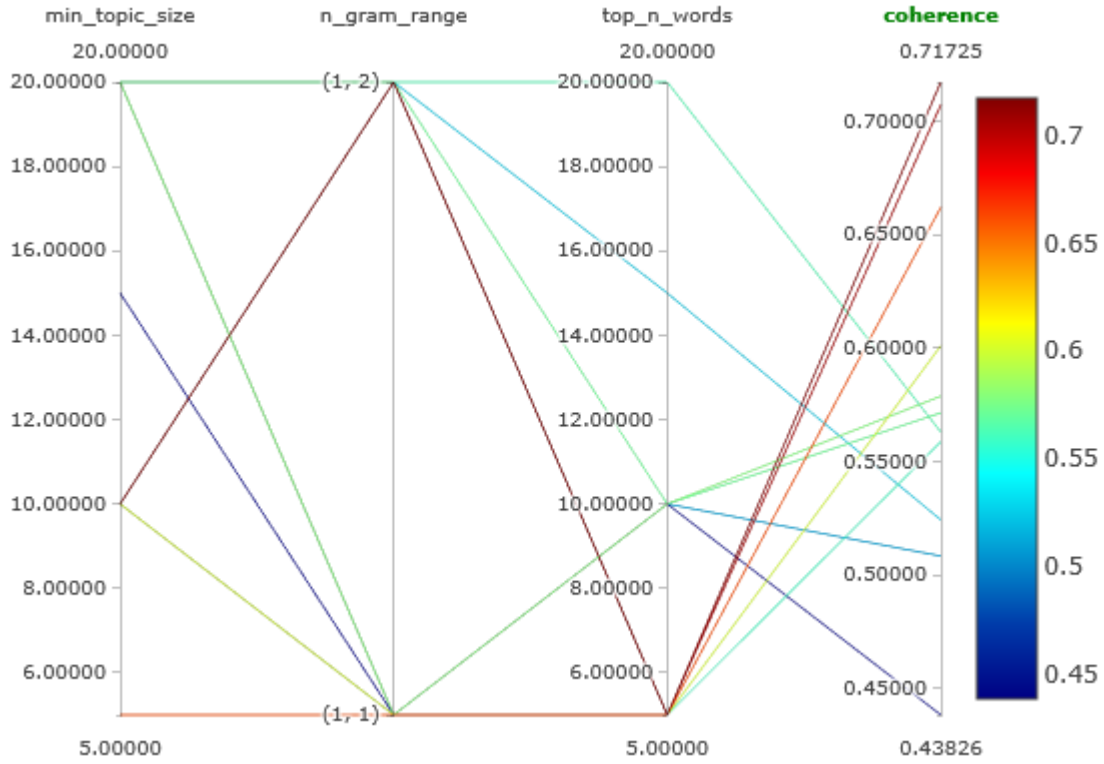
Les clusters de couleur bleu foncé revelent que certains topics sont tres corrélés et pourraient ainsi appartenir à la même catégorie sémantique. Nous retrouvons les regroupements identifiés précédemment :

- Topic 1, Topic 3, Topic 4 & Topic 6 : "Qualité, mauvais, nul, livraison" : association forte entre la mauvaise qualité des produits et les problèmes de livraison;
- Topic 2 & Topic 8 : "Service client, service, remboursement, livraison" : indiquent que ces deux topics traitent du service client et des problèmes de retour/livraison.

- Topic 0, Topic 5 & Topic 7: “Commande, site, recevoir, client, privée, prix, cher”: rassemble des problématiques liées à la plateforme de vente en ligne, aux prix et à la réception des commandes.

A l'inverse, les topic 0 & topic 6 : "Commande, recevoir, client" vs "Nul, photo, livraison" montre que le topic sur la qualité perçue des produits (0) est distinct du topic sur les problèmes de livraison (6).

Dans le graphique suivant, nous avons représenté un Parallel Coordinates Plot généré par MLflow pour comparer différentes configurations du modèle de topic modeling BERT (type BERTopic) en fonction des hyperparamètres et de la métrique de cohérence.



Les paramètres testés incluent la taille minimale des topics (min_topic_size), le nombre de mots-clés affichés par topic (top_n_words) et la plage de n-grammes (n_gram_range). Les meilleurs résultats ont été obtenus avec une combinaison utilisant des unigrammes et bigrammes (n_gram_range = (1,2)), une taille minimale de topic élevée (min_topic_size = 20) et un nombre réduit de mots-clés par topic (top_n_words = 5). Cette configuration a permis d'atteindre une cohérence maximale de 0.717, indiquant une bonne qualité des topics extraits. Ces résultats suggèrent que des topics plus larges et bien définis, combinés à une granularité fine du vocabulaire, permettent une meilleure structuration thématique des commentaires analysés.

3.3.3. Algorithme BERT et Clustering

Dans cette étape, nous avons cherché à améliorer le résultat précédent en appliquant un clustering *après* BERTopic afin de clusteriser les topics entre eux et de personnaliser le processus dans un but par exemple de comparer des algorithmes, imposer un nombre de clusters fixe ou ajouter des contraintes métiers.

L'algorithme de Topic Modeling avec BERT et clustering suit les étapes suivantes :

- Extraction des embeddings BERT : convertir chaque commentaire en un vecteur dense avec un modèle BERT. La méthode de moyenne des Tokens a été préférée à la classification des tokens (CLS) étant plus performante dans le cas de la recherche de similarité et donc l'usage du topic modeling. Comme précédemment, nous utilisons les deux modèles : CamemBERT et BERT Multilingual. Les embeddings des deux modèles sont également concaténés avant utilisation avec Bert.

- Réduction de dimension avec UMAP : réduire les embeddings pour faciliter le clustering et réduire le temps de calcul ;

- Clustering avec deux approches HDBSCAN et KMEANS et combinaison des deux modèles avec Voting : regrouper les commentaires en topics en bénéficiant de la robustesse d'HDBSCAN (gestion des outliers) avec la stabilité de KMeans ;

- Regroupement et interprétation des topics : regrouper les thématiques proches et extraire les mots-clés ;

- Évaluation du modèle et visualisation des topics.

Nous avons optimisé les paramètres d'UMAP, du clustering et BERT en utilisant RandomizedSearchCV pour explorer plusieurs configurations :

- Réduction de dimension UMAP
  - Nombre de dimensions après réduction : n_components : [3, 5, 10]
  - Contrôle la densité locale : n_neighbors : [5, 15, 30]
  - Distance minimale entre points : min_dist : [0.1, 0.5, 0.8]
  - Métrique de distance : metric : [“euclidean”, “cosine”, “manhattan”]
- Clustering
  - HDBSCAN
    - Taille min d'un cluster : min_cluster_size : [5, 10, 20]
    - Seuil de densité pour former un cluster : min_samples : [1, 5 ,10]
  - KMeans
    - Nombre de clusters : n_clusters: [3, 5, 10, 15]

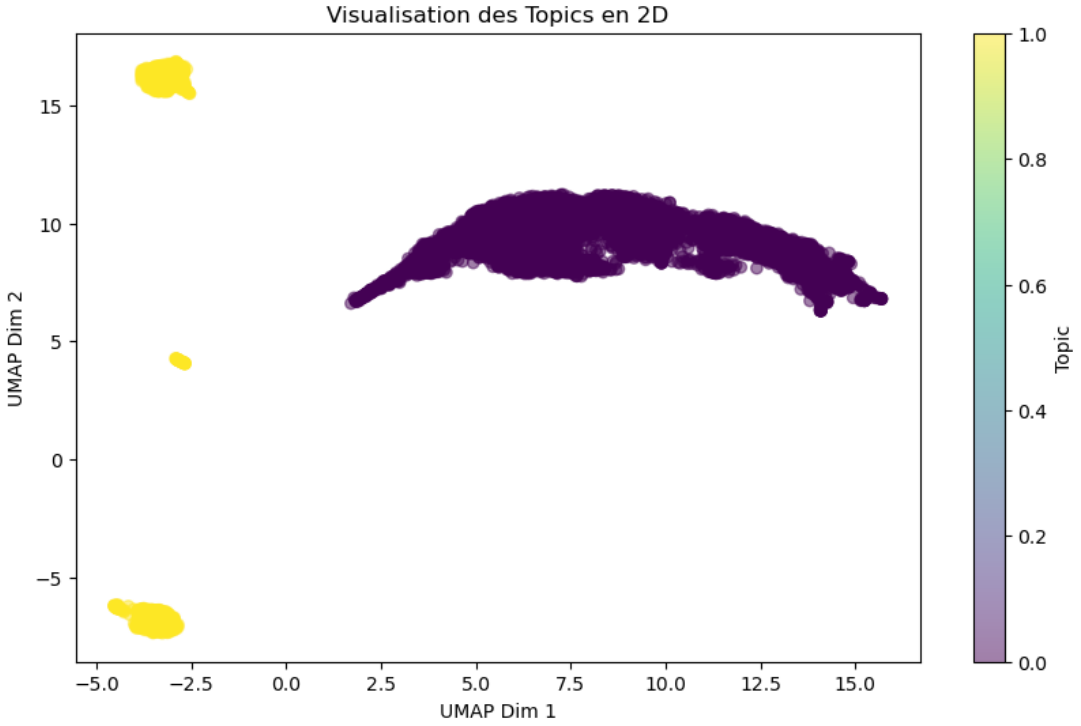
On évalue le clustering avec le score Silhouette.

Le meilleure score de Silhouette, égale à environ 0.9, est obtenu avec la combinaison de paramètres suivante : {'n_neighbors': 30, 'n_components': 5, 'n_clusters': 2, 'min_samples': 5, 'min_dist': 0.8, 'min_cluster_size': 20, 'metric': 'euclidean'}.

L'algorithme a identifié deux clusters :

- 0: 5039 avis ;
- 1: 933 avis.

On projette les embeddings en 2D pour voir la séparation des clusters.



Nous observons :

- Un grand cluster dense en violet (Topic 0) qui contient la majorité du corpus ;
- Des groupes isolés en jaune (Topic 1) : topics distincts, séparés du groupe principal. Ils peuvent correspondre à des thèmes spécifiques identifiables ou des sous-thèmes.

Les wordclouds des principaux mots clés de chaque cluster sont présentés ci-dessous.

Nous identifions des similarités entre les deux clusters. Les mots "commande", "site", "colis", "service client", "remboursement", "recevoir", et "livraison" apparaissent en fréquence importante dans les deux wordclouds. Les commentaires autour de la réception des commandes, les problèmes de livraison et les remboursements sont des thèmes de préoccupation des acheteurs. Nous notons également la présence du mot "rembourser" et de termes négatifs comme "problème", "rien", "aucun", "jamais" ce qui indique que des clients rencontrent des difficultés avec les retours et remboursements.

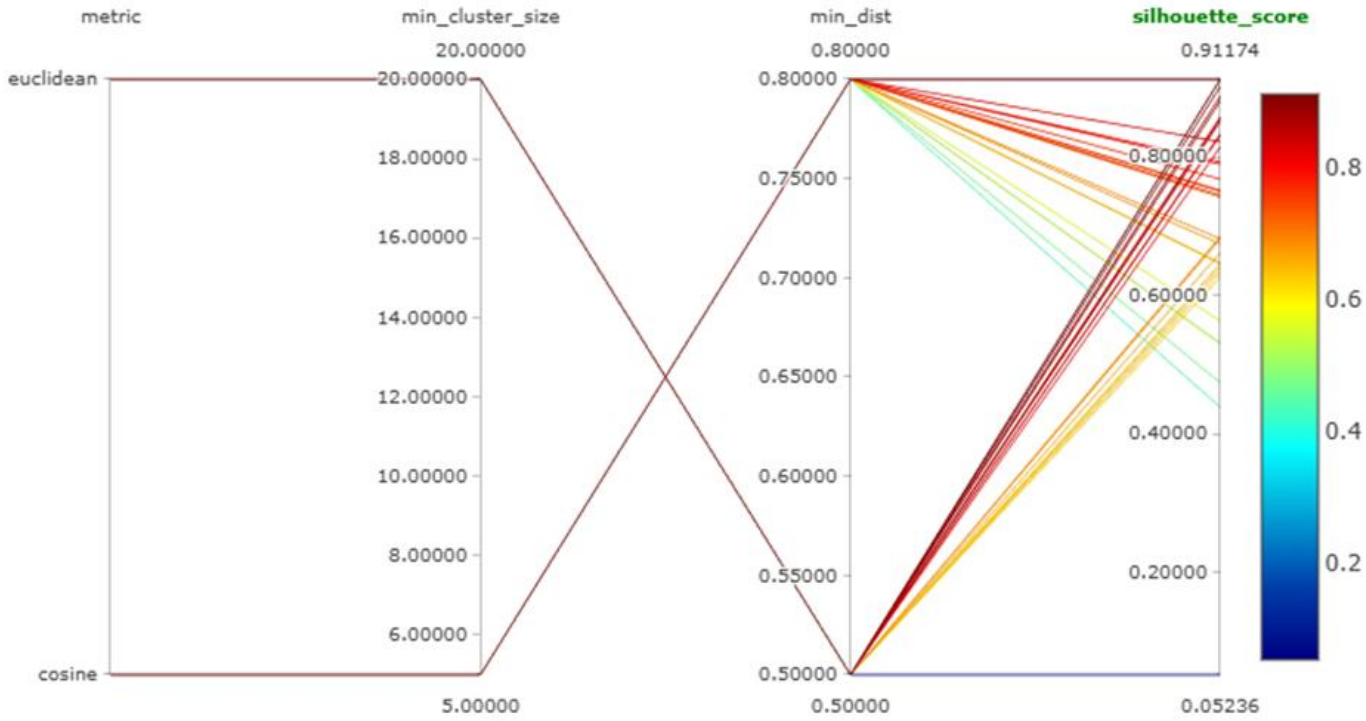
Des disparités sont également identifiables. Le cluster 0 semble insister davantage sur "service client", "remboursement", et des termes associés aux délais comme "mois", "attendre", ce qui correspond a des avis négatifs sur le service après-vente et les délais de remboursement.
Le cluster 1 contient également les mots clés "vente", "acheter", "commander", ce qui pourrait indiquer qu'il regroupe des avis négatifs portant davantage sur l'acte d'achat et la réception du produit.

Par cette analyse, nous pouvons déduire que les commentaires avec une note <3, montrent une forte insatisfaction liée à la gestion des commandes, aux retards de livraison et aux remboursements. Le service client est souvent mentionné et ceci négativement, ce qui signifie qu'il joue un rôle clé dans l'expérience des acheteurs.

Le graphique ci-dessous est un Parallel Coordinates Plot créé par MLFlow, permettant de comparer les hyperparamètres (metric, min_cluster_size et min_dist) et leur impact sur la métrique du score de silhouette. Ce dernier évalue la qualité du clustering.

L'échelle de couleurs représente la valeur du silhouette score. Une couleur rouge correspond à un score élevé et donc à des clusters cohérents. La couleur bleu correspond au contraire à un score faible et donc à des clusters peu cohérents.

Selon le score de silhouette, les meilleurs résultats de clustering sont obtenus avec une distance euclidienne, une taille minimale de cluster relativement grande (min_cluster_size = 20) et une distance minimale entre points : min_dist élevée (autour de 0.8).


metric
min_cluster_size
min_dist
silhouette_score
euclidean
cosine
20.00000
18.00000
16.00000
14.00000
12.00000
10.00000
8.00000
6.00000
5.00000
0.80000
0.75000
0.70000
0.65000
0.60000
0.55000
0.50000
0.91174
0.80000
0.60000
0.40000
0.20000
0.05236
0.8
0.6
0.4
0.2

Conclusions

Conclusions métier

L'utilisation combinée des modèles de classification et de topic modeling permet d'obtenir une vision globale et détaillée des facteurs d'insatisfaction client. Cette analyse approfondie offre à l'entreprise des informations précieuses pour identifier les axes d'amélioration prioritaires et mettre en place des actions correctives efficaces.

Les modèles de classification mis en œuvre permettent d'identifier les principaux facteurs contribuant à l'insatisfaction. Trois catégories majeures se dégagent :

1. **Service client**

   Cette catégorie est fortement associée aux termes « remboursement », « client » et « commande », ce qui suggère que les problèmes liés aux remboursements, à la gestion des commandes et aux interactions avec le service client sont des sources majeures de mécontentement.

2. **Logistique**

   L'importance du mot « colis » met en évidence les problèmes liés à la livraison et à la gestion des colis. Les retards, les pertes ou les dommages subis par les colis sont des facteurs d'insatisfaction significatifs.

3. **Conformité des articles**

   Le terme « conformité » souligne l'importance de la qualité et de la fidélité des produits par rapport à leur description. Les clients expriment leur mécontentement lorsque les articles reçus ne correspondent pas à leurs attentes.

Les modèles de topic modeling permettent de confirmer et d’approfondir l’analyse des motifs de mécontentement en regroupant les problèmes en thématiques spécifiques. Cette approche permet d'identifier des pistes d'amélioration concrètes et à fort impact pour l'entreprise :

1. **Expérience d'achat**

   Cette thématique englobe les problèmes liés à la facilité de commande, aux difficultés de remboursement et à la non-conformité des produits par rapport à leur description. Les clients souhaitent une expérience d'achat fluide, transparente et conforme à leurs attentes.

2. **Logistique**

   Les retards de livraison et les colis endommagés sont les principaux motifs de mécontentement dans cette catégorie. L'amélioration de la fiabilité et de la rapidité de la livraison est cruciale pour satisfaire les clients.

3. **Relation client**

   Cette thématique regroupe les problèmes liés à la réactivité du service client, à la gestion des plaintes, à l'assistance après-vente et à l'efficacité du support. Les clients attendent un service client disponible, compétent et capable de résoudre leurs problèmes rapidement.

Pour répondre aux problématiques ciblées et ainsi améliorer significativement la satisfaction clients, les mesures suivantes peuvent être opportune pour Veepee et showroom :

1. Service et relation Client

Plusieurs pistes peuvent être explorées pour améliorer le processus de remboursement, important motif d'insatisfaction. Les demandes de remboursement peuvent être automatisées et suivies en temps réel par le client. Le nombre de demandes de remboursement traitées dans les délais ainsi que le temps moyen de traitement de demande de remboursement peuvent constituer des indicateurs de suivi intéressants.
Afin d'améliorer la gestion des commandes, les informations relatives aux commandes peuvent être centralisées pour un accès facile par le service client. Un chatbot ou une FAQ peuvent également être intégrés pour répondre aux questions courantes sur les commandes.
Enfin, pour renforcer les interactions avec le service client, les canaux de contacts peuvent être diversifiés (mails, téléphone, chatbot, réseaux sociaux) et un système de ticket peut faciliter le suivi et la résolution des demandes clients. Les agents peuvent également bénéficier de formations destinées à renforcer l'empathie et la résolution proactive des problèmes. Le temps moyen de réponse du service client, le taux de résolution des problèmes dès le premier contact et l'évaluation de la satisfaction client via des enquêtes post-interaction peuvent constituer une base d'évaluation du service client.

2. Expérience d'achat

Pour garantir la conformité des articles et améliorer l'expérience d'achat sur la marketplace, il peut être envisagé de mettre en place un contrôle de qualité strict, incluant un processus de vérification des produits avant leur mise en ligne et l'obligation pour les vendeurs de fournir des descriptions détaillées et illustrées. Un modèle réseau de neurones de reconnaissance d'images et un modèle de langage peuvent être utiles pour vérifier la conformité de l'annonce et des photos promotionnelles aux descriptions détaillées du produit.
En parallèle, la transparence doit être renforcée en encourageant les clients à laisser des avis et des photos des produits reçus, en mettant en place un système de notation des vendeurs basé sur la conformité des produits, et en offrant une garantie de conformité (remboursement ou échange si le produit ne correspond pas à la description).
Pour mesurer l'efficacité de ces actions, il peut être opportun de suivre le taux de réclamations liées à la non-conformité des produits, le taux de retour des produits pour non-conformité, et le nombre d'avis clients mentionnant des problèmes de conformité. Ces indicateurs permettront d'identifier les points d'amélioration et d'ajuster les stratégies pour offrir une expérience d'achat plus fiable et satisfaisante.

3. Logistique

Pour améliorer la gestion des colis et répondre aux problèmes logistiques, il peut être opportun d'auditer régulièrement la performance des partenaires afin de pouvoir évaluer leur fiabilité. La mise en place d'un système de suivi en temps réel (tracking) et l'offre d'options de livraison flexibles (points relais, livraison express) permettront d'améliorer l'expérience client. Pour réduire les retards et les pertes, il peut être recommandé de mettre en place des alertes pour les colis en retard ou perdus, et de proposer des compensations (remboursements partiels, bons de réduction) en cas de problème. Enfin, pour prévenir les dommages aux colis, il faut imposer des standards d'emballage aux vendeurs, former les transporteurs à la manipulation des colis fragiles, et proposer une assurance colis pour les articles de valeur. Les indicateurs de suivi clés peuvent inclure le taux de livraison dans les délais, le nombre de colis perdus ou endommagés, le suivi ciblé du taux de satisfaction des clients concernant la livraison, le temps moyen de traitement des réclamations logistiques, et le nombre de réclamations liées à la logistique par rapport au nombre total de commandes.

Conclusions scientifiques

Optimisation du traitement de texte : de la performance à la complexité de la modélisation

Dans le cadre du traitement automatique du langage naturel, l'efficacité d'un modèle dépend fortement des étapes de pré-traitement et de modélisation.

Pré-traitement : l'avantage de spaCy

En matière de pré-traitement, spaCy se distingue par sa performance supérieure par rapport aux lemmatiseurs plus traditionnels. Sa capacité à effectuer une analyse linguistique complète, incluant la tokenisation, la lemmatisation et l'analyse syntaxique, en fait un outil de choix pour préparer les données textuelles. La précision et la rapidité de spaCy permettent d'obtenir des résultats optimaux pour les étapes de modélisation ultérieures.

Modélisation : équilibre entre performance et complexité

Dans le cadre de notre projet, nous avons considéré deux approches de techniques d'apprentissage automatique : la classification et le clustering (topic modeling) qui présentent des objectifs et des approches différentes. La classification, tâche d'apprentissage, associe des données à des catégories prédéfinies (dans notre cas positif et négatif) à partir d'un ensemble de données étiquetées. Le clustering, tâche d'apprentissage non supervisé, groupe de façon automatique des données similaires sans avoir d'étiquettes préexistantes.

Classification

Nous avons considéré plusieurs traitements des données et algorithmes de machine learning. Nous observons que les modèles, XGBClassifier, GradientBoostingClassifier et RandomForestClassifier sont bien plus performants pour prédire un sentiment (positif ou négatif) que la note (1 à 5). Le degré de satisfaction et la nuance sont plus difficiles à capturer. Nous obtenons que le modèle XGBClassifier produit le meilleur f1-score quelle que soit l'approche utilisée pour tokeniser, vectoriser et normaliser. Cette performance s'explique par le fait que le XGBClassifier (XGBoost Classifier) est une implémentation avancée du Gradient Boosting qui peut souvent surpasser Random Forest et d'autres algorithmes de boosting classiques par la presence des criteres suivants : résistance à l'overfitting, Early Stopping et gestion des classes déséquilibrées.

Enfin, nous avons également testé un modèle de deep learning pour le problème de classification : un réseau de neurones avec une architecture simple. Nous obtenons des performances similaires à celles obtenues avec XGBClassifier. Afin d'optimiser cette approche, il serait intéressant de le tester sur un dataset plus volumineux et avec une architecture mieux configurée (augmentation du nombre de couches, meilleure optimisation des hyperparamètres).

Il est à noter que les réseaux de neurones offrent des performances élevées en TALN, notamment pour la classification de texte. Ils sont plus performants pour des données complexes et ils s'améliorent avec l'ajout de données. De plus, on peut utiliser des modèles pré-entraînés tels que Bert. Néanmoins, leur complexité rend l'interprétation des résultats plus difficile. Comprendre les raisons des prédictions d'un modèle réseau de neurones peut s'avérer complexe, ce qui peut limiter leur utilisation dans les contextes où l'explicabilité est cruciale. L'avantage du machine learning est que le prétraitement des données nécessaire est moins important et il est plus facilement interprétable. Le temps de calcul est aussi plus court, par contre il atteint une limite de performance sur des jeux de données volumineux. De plus, nous avons constaté que la

réduction de dimension ainsi que le rééchantillonnage ne permettent pas non plus d'améliorer significativement la mesure.

Topic modeling

Dans cette étape, nous avions pour objectifs d'extraire automatiquement les thèmes (topics) du corpus de commentaires. Nous avons procédé en deux étapes : un modèle LDA, un modèle BERT seul puis un modèle KMeans après BERT pour réaliser du clustering.

Le modèle LDA permet une approche thématique simple des commentaires et nécessite moins de temps de calcul, mais l'algorithme est également moins performant en matière de compréhension contextuelle profonde : au-delà de trois topics, les groupes se superposent et le modèle peine à faire la distinction entre motifs de satisfaction. Par ailleurs, contrairement à certaines méthodes qui peuvent estimer automatiquement le nombre de groupes (par exemple, des variantes de clustering comme DBSCAN), LDA ne calcule pas automatiquement le nombre optimal de sujets. Cela impose donc une approche empirique de tests de différentes valeurs pour le paramètre "num_topics".

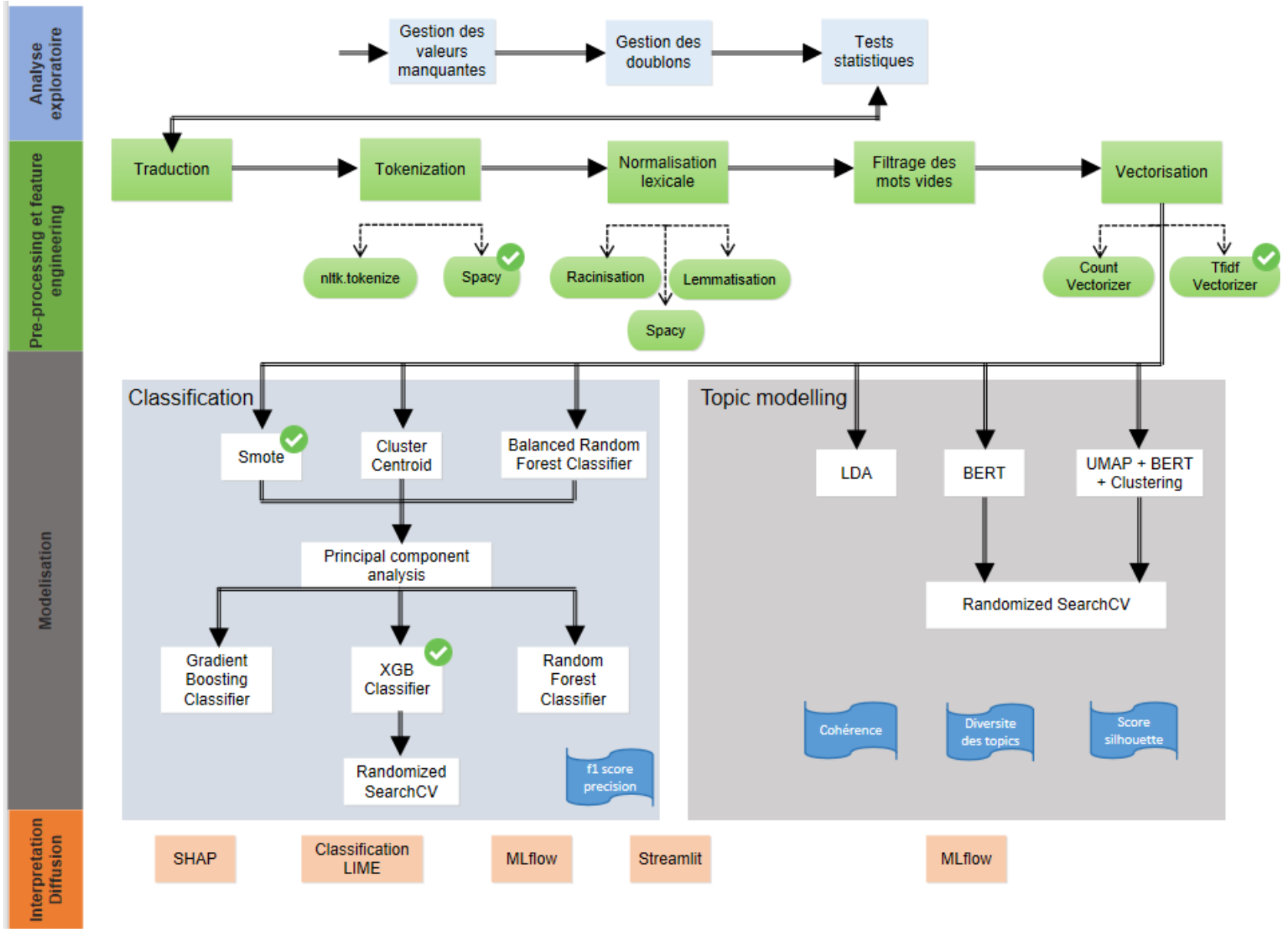
BERT (Bidirectional Encoder Representations from Transformers) est reconnu pour ses performances exceptionnelles dans diverses tâches de TALN. L'utilisation de BERT permet de tenir compte du contexte et pas uniquement de la fréquence des mots et présente une meilleure gestion des synonymes. Cependant, l'entraînement de ce modèle peut être extrêmement long et gourmand en ressources, en particulier pour atteindre les niveaux de performance souhaités. Il est donc essentiel de trouver un compromis entre la durée d'entraînement et la performance attendue.

L'ajout de KMeans après BERT permet de regrouper des textes ou des phrases en clusters sémantiquement similaires basés sur le sens : contrairement par exemple à TF-IDF + KMeans (qui repose sur les fréquences des mots). Nous avons également appliqué une réduction de dimension (UMAP) avant KMeans pour optimiser le clustering. Cette méthode permet de regrouper les avis clients selon leur contenu et ainsi de détecter des problématiques similaires.

Sur notre corpus d'avis, nous obtenons une valeur satisfaisante de coefficient de silhouette de 0.8. Cette approche est simple à interpréter et efficace par sa rapidité et sa performance, elle nous apparaît donc comme une approche intéressante.

Pour conclure, le choix des outils et des modèles dans le traitement automatique des commentaires nécessite une évaluation attentive des compromis entre performance, complexité et ressources disponibles. spaCy offre un avantage significatif pour le pré-traitement, tandis que les modèles BERT et réseaux de neurones présentent des forces et des faiblesses différentes en matière de modélisation.

Annexe - Diagramme de flux des données



Marie-Anne Bigot-Sazy, Myriam Bris, Deborah Cohen