NLP Project 2 Insurance Review

YOUSAF Fajer

SOMMAIRE

INTRODUCTION

Dans le cadre du projet du module *Advance machine Learning and NLP,* il nous a été mis à disposition deux jeux de données contenant des avis sur des assurances françaises, un jeu de train et un jeu de test qui ne contiennent pas les notes attribueés par les utilisateurs. L' objectif de ce projet est de modéliser et d'étudier les notes attribuées aux différentes assurances en fonction des différents variables du départ et des variables que l'on va créer.

Pour ce faire nous avons d'abord étudié les les avis des bénéficiaires et avec une approche d'apprentissage non supervisé, on a pu ainsi classifier les commentaires. Puis dans un deuxième temps, on a définit une approche de régression et testé plusieurs algorithmes et modèles. Pour chacun d'entre eux nous avons apprécié leur performances, afin de conclure sur le meilleure modèle nous permettant de prédire au mieux la target : *note.*

Présentation du dataset

On souhaite prédire la variable **Note** dans le jeu de données test mais avant cela nous allons travailler sur le jeu de données avisassurance_train.

Pour cela, on dispose du jeu de données d'entraînement data qui contient 24105 observations et 5 variables qui sont toute qualitatives :

- **date** : il contient la date de publication de l'examen et la période d'expérience de l'examen, et il doit être nettoyé.
- **auteur** : l'identifiant du client
- **avis** : le texte avec l'avis
- **assureur** : le nom de l'assurance
- **produit** : le type d'assurance

La première étape est de créer un modèle non supervisé pour mieux comprendre les critiques et créer des segmentations qu'on pourra interpréter. Ensuite, il faudra créer différents modèles pour prédire le nombre d'étoiles d'une critique à l'aide de l'ensemble de données d'entraînement

Il s'agit ici d'un problème de régression avec comme métrique RMSE.

	date	note	auteur	avis	assureur	produit
0	06 septem...	5	brahim--k-131532	Meilleurs assurances, prix, solutions, écoute,...	Direct Assurance	auto
1	03 mai 20...	4	bernard-g-112497	je suis globalement satisfait , sauf que vous ...	Direct Assurance	auto
2	21 mars 2...	5	virginie-t-107352	Prix tres abordable plusieurs options s'offren...	Direct Assurance	auto
3	10 juin 2...	4	boulain-f-116580	je satisfait du service, une réponse très rapi...	L'olivier Assurance	auto
4	29 janvie...	1	ouaille31-51798	Client depuis plus de 25 ans, très déçu de cet...	Matmut	auto
...	...	...	...	...	...	...
24100	22 mars 2...	1	hophop-107522	Assurance moto chez la mutuel des motards en F...	Mutuelle des Motards	moto
24101	06 décemb...	1	tzl-81680	Même les demandes les plus simples n'aboutisse...	Allianz	habitation
24102	14 avril ...	1	jmr-72500-110395	En décembre 2019, j'ai souscrit à un contrat C...	Cegema Assurances	sante
24103	11 juille...	3	cris-77532	Je suis assurer à la gmf depuis plus de 15 ans...	GMF	auto
24104	19 janvie...	1	jesse-51459	Bonjour\r\nMon ami vient de se faire voler sa ...	AMV	moto

24105 rows × 6 columns

Présentation du dataset

On a ensuite regardé la description du dataset grâce à la fonction data.describe(include="all").T

Etant donnée la majorité des variables sont qualitatives, on s'aperçoit que toutes les valeurs ne sont pas remplies.

On peut voir qu'il y a 1 valeur manquante dans chacune des variables avis et auteur donc on remplace ces données manquantes par 'None'.

En ce qui concerne la target Note, qui est la seule variable quantitative,, la moyenne est d'environ 2,85 et la médiane est de 3.

Les variables assureur et produit ont respectivement 56 et 13 valeurs uniques chacune.
Par la suite nous pourrions manipuler ces variables et visualiser l'évolution de notes en fonction de ses variables.

	count	unique	top	freq	mean	std	min	25%	50%	75%	max
date	24105	1949	08 avril ...	101	NaN	NaN	NaN	NaN	NaN	NaN	NaN
note	24105.0	NaN	NaN	NaN	2.847583	1.531368	1.0	1.0	3.0	4.0	5.0
auteur	24104	23676	mm-53953	4	NaN	NaN	NaN	NaN	NaN	NaN	NaN
avis	24104	24069	Intervention supprimée à la demande de l'inter...	10	NaN	NaN	NaN	NaN	NaN	NaN	NaN
assureur	24105	56	Direct Assurance	5896	NaN	NaN	NaN	NaN	NaN	NaN	NaN
produit	24105	13	auto	14077	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Création de nouvelles variables

On va créer de nouvelles variables qui vont nous donner des indications mathématiques sur les commentaires :

- Le nombre de mot dans chaque commentaire
- Le nombre de caractère dans chaque commentaire
- Le nombre de phrase dans chaque commentaire
- La taille moyenne des mots dans chaque commentaires
- La taille moyenne des phrase dans chaque commentaire

Ces nouvelles variables sont des variables quantitatives et nous verrons par la suite comment elles participeront à l'amélioration des modèles lors de l'apprentissage supervisé.

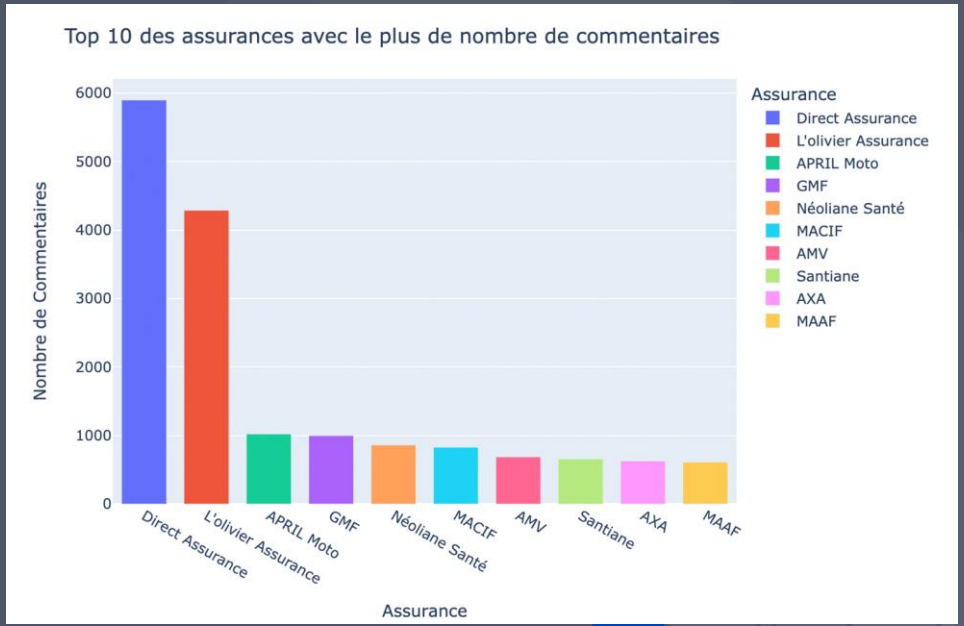
Data Visualisation

Nombre de commentaires par assurance

Nous nous sommes d'abord intéressés aux assurances présentes dans le dataset. Grâce à un histogramme on a pu observer quelles sont les 10 assurances avec le plus grand nombre de commentaires/avis présentes dans notre dataset.

On voit que deux assurances se distinguent : Direct assurance et Olivier Assurance avec chacun 5896 et 4288 avis dans notre dataset.

On peut donc affirmer que le nombre de commentaires n'est pas distribué de manière uniforme parmis les assurances.

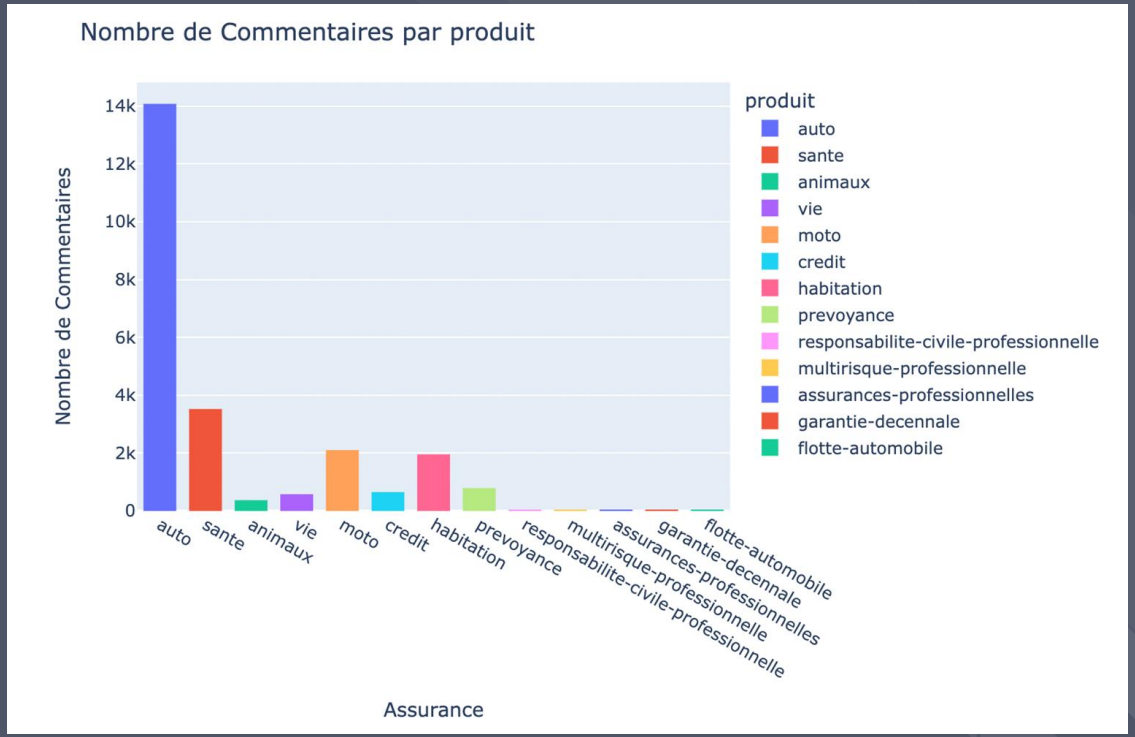


Data Visualisation

Nombre de commentaire par produit d'assurance

Nous nous sommes maintenant intéressés aux différents produits représentés dans le dataset. Grâce à un histogramme on a pu observer quelle est la répartition du nombre de commentaires/avis en fonction des 13 produits présent dans ce dataset.

On peut observer que les auto sont les plus majoritaires avec plus de 14 000 commentaires laissés. Ils sont ensuite suivis par la santé (3525 avis), moto (2105 avis) puis habitation (1956 avis).
Les autres produits sont présents que très peu de fois (inférieur à 800 avis laissés).



Data Visualisation

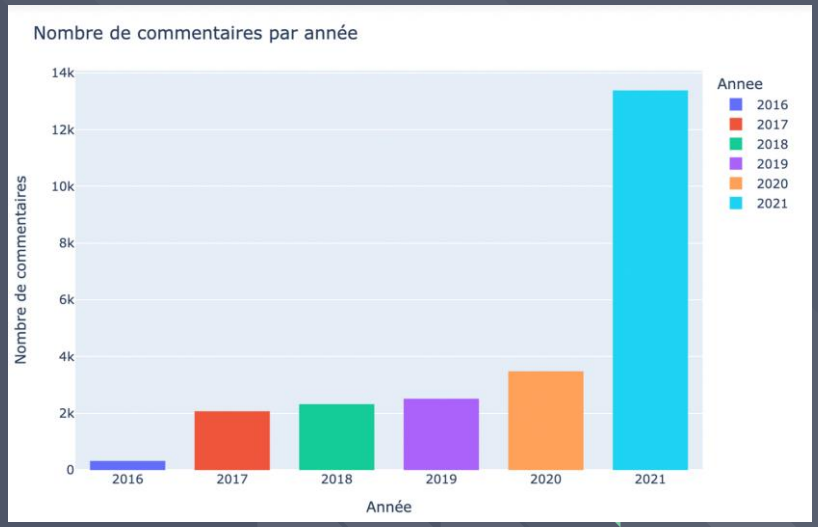
Variables temporelles : l'année de la publication du commentaire

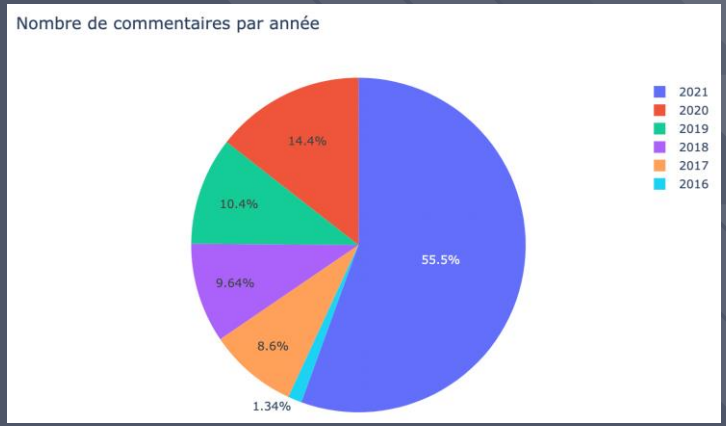
La variable date était sous la forme suivante ; `06 septembre 2021 suite à une expérience en septembre 2021`

Il a fallu modifier la date et séparer cette variable en nouvelles variables que l'on pourra exploiter grâce au tokenizer. On crée donc 5 nouvelles colonnes à partir de cette variable :

- data['Jour']=jour
- data['Mois']=mois
- data['Annee']=annee
- data['mois_siniste']=mois_siniste
- data['annee_siniste']=annee_siniste

On va d'abord observer les années de publication des commentaires et grâce à un histogramme et un diagramme circulaire on peut voir que la majorité des commentaire du dataset sont apparus en 2021 (55,5% des commentaires).





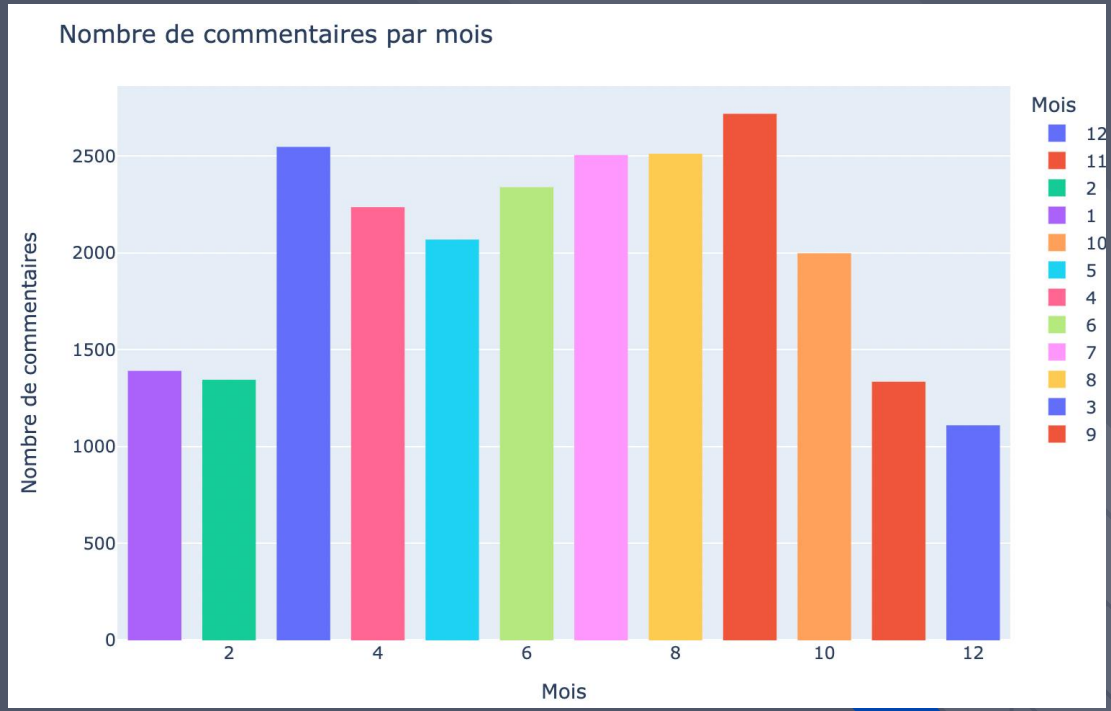
Data Visualisation

Variables temporelles : l'année de la publication du commentaire

On va maintenant transformer la variable des mois en valeurs numériques pour ensuite observer la répartition des commentaires par mois

Grâce à cet histogramme on voit qu'il y a deux pics, un au mois de mars et un autre au mois de septembre.

Ces pics peuvent être expliqués par le fait qu'au mois de février il y a les vacances d'hiver et qu'avant septembre il y a les vacances d'été donc les gens décide de publier des commentaires qu'à leur retour de vacances.



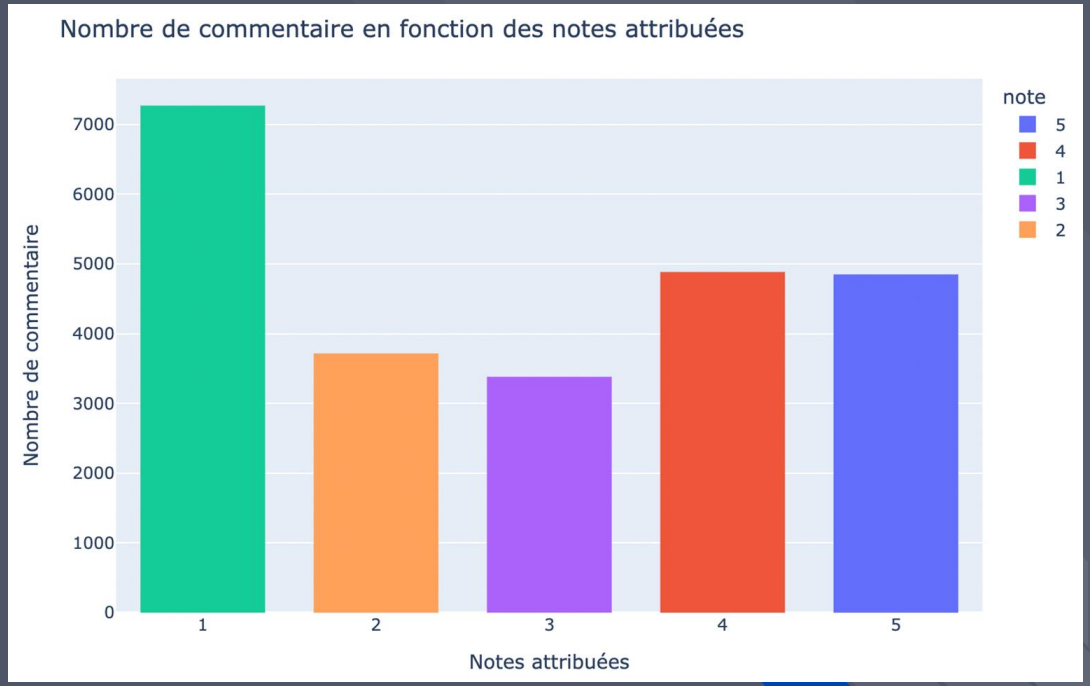
Data Visualisation

Nombre de commentaire par note attribués

Les personnes qui laissent des avis vont ensuite donner une note suite à leur expérience avec l'assurance. Cette note peut prendre une valeur entre 1 et 5.

On crée un histogramme qui donne la distribution des notes.

La note qui a été la plus attribuée est la note ⅕ donc c'est la note la plus basse.
Généralement lorsque les gens sont mécontents de leur assurances, is n'hésitent pas à attribuer une note basse.
Mais lorsqu'ils sont satisfaits, ils hésitent généralement entre le 4 et le 5. C'est pourquoi on voit un équilibre quasi-parfait entre les notes 4 et 5.



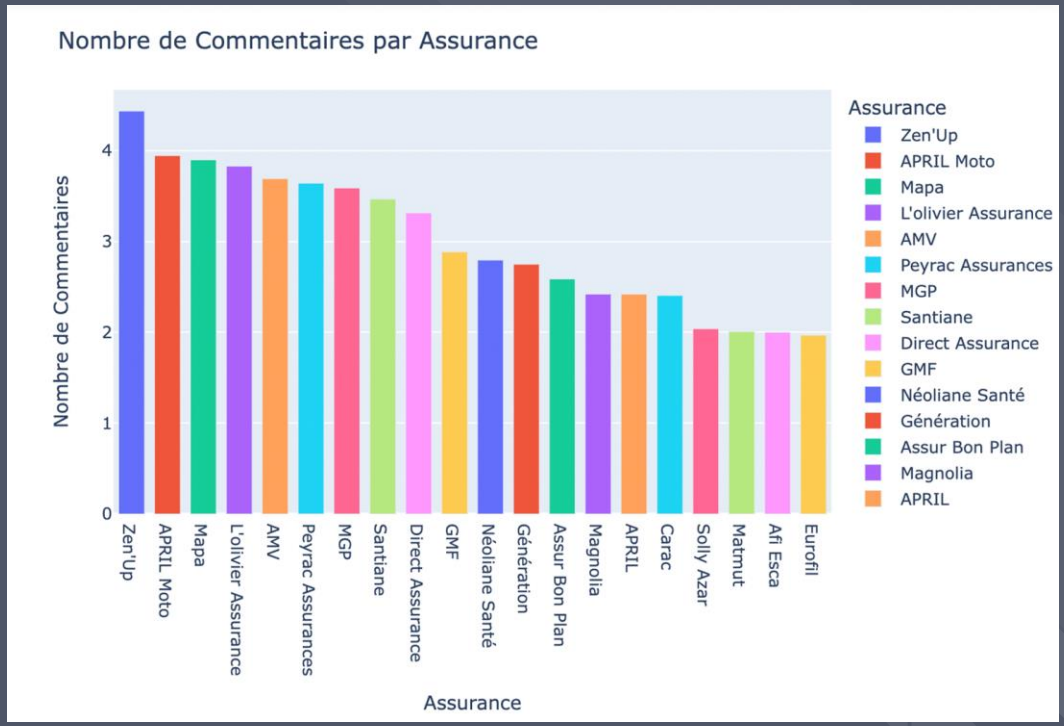
Data Visualisation

Note moyenne par assurance

Pour afficher le graphique suivant on a dû créer une nouvelle table qui nous donne la note moyenne par assureur on a donc utilisé les deux variables note et assureur.

On trouve sur cet histogramme les 20 assurances avec la moyenne sur les notes la plus élevée. L'assurance qui a la meilleur note est Zen'Up avec une note moyenne de 4,44/5 et qui dans le dataset a 245 avis laissés donc ce qui correspond à environ 1% des avis de tous le dataset.

On voit aussi qu'en tout il y a seulement 13 assurances qui ont une note moyenne supérieur à la moyenne ce qui représente 23% des assureurs. Donc au moins ¾ des assureurs présents dans le dataset ont des moyennes de note inférieur à 2.5/5.



Data Cleaning

Le data cleaning va se dérouler en plusieurs étapes :
La première étape est de mettre tous les mots en caractères minuscules.

Supprimer les caractères spéciaux

- les mentions
- les url
- les symboles
- les hyperliens
- la ponctuation

Créer une liste avec les mots parasites

On va importer une liste qui contient les stopword français et qu'on va agrémenter de mots que l'on considère comme parasites mais qui ne se trouve pas dans la liste de départ. Puis de cette liste on va supprimer les mots suivants car ils ne nous seront utile pour notre analyse :
'ne','pas','plus','personne','aucun','ne','pas','plus','personne','aucun',
'ni','aucune','rien''ni','aucune','rien'

L'étape d'après va être de créer de nouvelles colonnes sans les caractères spéciaux et sans les mots parasites une sous forme de liste et une autre sous forme de chaîne de caractères.
Pour une partie de la suite de notre étude on va regrouper les commentaires par assureur.

Récupérer la fréquence de chaque mot

On va Identifier la fréquence relative de chaque mot et stocker cette information dans un dictionnaire puis dans un liste pour effectuer un tri ensuite. J'ai ensuite afficher les 30 mots les plus fréquents dans les commentaires et on voit que ces mots seuls ne vont pas donner d'indication sur le sentiment du commentaire, c'est lorsqu'ils sont assemblés qu'ils vont donner cette indication.

```
[('chez', 55),
 ('assurance', 55),
 ('rien', 54),
 ('plus', 54),
 ('dune', 54),
 ('dossier', 54),
 ('toujours', 53),
 ('mois', 53),
 ('après', 53),
 ('contrat', 53),
 ('non', 53),
 ('répond', 53),
 ('client', 52),
 ('documents', 52),
 ('dit', 52),
 ('jamais', 52),
 ('prix', 52),
 ('problème', 52),
 ('service', 52),
 ('demande', 52),
 ('aucun', 52),
 ('suite', 52),
 ('plusieurs', 52),
 ('nai', 52),
 ('nest', 52),
 ('téléphone', 52),
 ('payer', 51),
 ('ans', 51),
 ('faut', 51),
 ('pendant', 51)]
```

Nuage de mots

On a créé une fonction qui va afficher le nuage de mots des commentaires par assurance après qu'ils soient été nettoyer.

On a affiché le nuage de mots des 20 mots les plus fréquents pour les 9 assurances avec le plus de commentaires laissés car afficher les 56 assurances aurait été illisible à la lecture.

On voit que les mots : contrat, plus, plu, tarif et prix se retrouvent dans plusieurs nuages de mots.

Apprentissage Non Supervisé

VECTORISATION

Pour l'apprentissage non supervisé on a décidé de d'abord de faire une analyse de sentiments qui sera utile dans la suite pour lancer les modèles supervisé et prédire la note.

On a aussi effectuer d'autres manipulations comme la vectorisation grâce à Tfidf Vectorizer avec lequel on a réussi à avoir une matrice tfidf avec tous les mots et leurs scores dans tous les commentaires. Avant d'appliquer le vectorizer on a d'abord créé une liste de mots qui sera normaliser en faisant un dédoublonnement des mots avec la même racine et pour cela on a importé depuis la librairie nltk.stem.snowball le FrenchStemmer.
On a initialisé un CountVectorizer pour compter le nombre de mots (fréquence des termes) et ainsi on a pu voir qu'il y a 43119 mots différents dans les commentaires. Ensuite Pour avoir un aperçu de l'aspect des valeurs IDF, on les affiche en triant les valeurs par ordre croissant ainsi on pourra voir quelle sont les mots avec le score le plus élevé.

	tfidf
direct	0.390744
assurance	0.378761
prix	0.328903
service	0.255430
plus	0.218839
...	...
fuirraison	0.000000
fuirprélevé	0.000000
fuirproscrire	0.000000
fuirproblèmes	0.000000
œuvres	0.000000

43119 rows × 1 columns

Apprentissage Non Supervisé

LDA

L'allocation de Dirichlet latente est la technique de modélisation de sujets le but est de définir les différents thèmes présent dans les commentaires.

Pour cela, on a créé un dictionnaire de termes de nos commentaires qu'on va transformer en matrice, et c'est sur cette matrice qu'on va appliquer le modèle.

Pour le modèle on a fait appel à la librairie Gensim qui contient une implémentation du modèle LDA. On a choisi comme paramètres 3 topics et on a choisi 10 pour le paramètre passes et qui correspond à combien de fois l'algorithme est censé passer sur l'ensemble du corpus et enfin on a choisis 40 mots par topics.

Avec ces paramètres on obtient le résultat suivant :

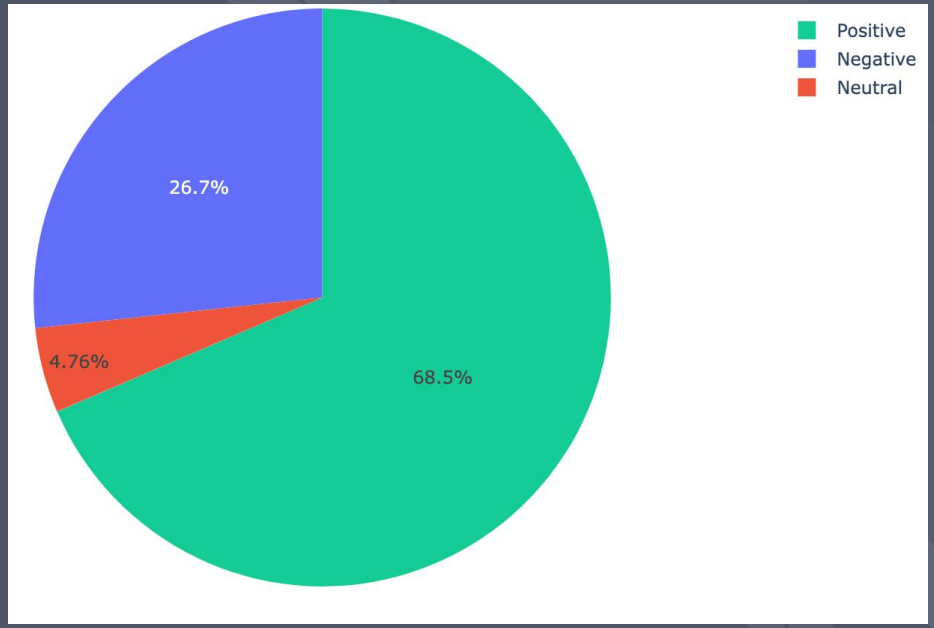
```
]: print(ldamodel.print_topics(num_topics=3, num_words=40))

[(0, '0.015*"plus," + 0.013*"mois," + 0.013*"contrat," + 0.010*"toujours," + 0.010*"mutuell
e," + 0.009*"demande," + 0.008*"dossier," + 0.007*"réponse," + 0.007*"après," + 0.006*"servic
e," + 0.006*"téléphone," + 0.006*"assurance," + 0.006*"mail," + 0.006*"rien," + 0.005*"compt
e," + 0.005*"jamais," + 0.005*"documents," + 0.005*"aucune," + 0.005*"client," + 0.005*"reç
u," + 0.005*"courrier," + 0.005*"jours," + 0.004*"nai," + 0.004*"jour," + 0.004*"nest," + 0.0
04*"aucun," + 0.004*"envoyé," + 0.004*"plusieurs," + 0.004*"suite," + 0.004*"carte," + 0.004
*"dit," + 0.004*"personne," + 0.004*"quil," + 0.003*"temps," + 0.003*"non," + 0.003*"euros,"
+ 0.003*"chez," + 0.003*"joindre," + 0.003*"dun," + 0.003*"faut,"'), (1, '0.018*"plus," + 0.0
16*"assurance," + 0.014*"sinistre," + 0.011*"ans," + 0.011*"véhicule," + 0.011*"chez," + 0.00
8*"voiture," + 0.007*"mois," + 0.006*"charge," + 0.006*"non," + 0.006*"assuré," + 0.006*"rie
n," + 0.005*"après," + 0.005*"accident," + 0.005*"payer," + 0.005*"suite," + 0.005*"aucun," +
0.005*"jamais," + 0.005*"contrat," + 0.004*"dun," + 0.004*"euros," + 0.004*"dit," + 0.004*"ma
if," + 0.004*"expert," + 0.004*"quil," + 0.004*"assureur," + 0.004*"bonus," + 0.003*"année,"
+ 0.003*"macif," + 0.003*"axa," + 0.003*"aucune," + 0.003*"client," + 0.003*"lexpert," + 0.00
3*"malgré," + 0.003*"sinistres," + 0.003*"cher," + 0.003*"nest," + 0.003*"garage," + 0.003*"p
rendre," + 0.003*"personne,"'), (2, '0.046*"prix," + 0.035*"service," + 0.031*"satisfait," +
0.027*"assurance," + 0.017*"rapide," + 0.013*"simple," + 0.011*"client," + 0.010*"tarifs," +
0.010*"site," + 0.010*"merci," + 0.009*"bonne," + 0.009*"tarif," + 0.008*"tres," + 0.008*"dir
ect," + 0.008*"devis," + 0.008*"qualité," + 0.008*"contrat," + 0.007*"rapport," + 0.007*"plu
s," + 0.007*"services," + 0.007*"ligne," + 0.007*"efficace," + 0.006*"téléphone," + 0.006*"pr
atique," + 0.006*"merci" + 0.006*"facile," + 0.005*"assurance" + 0.005*"chez," + 0.005*"nivea
u," + 0.005*"rapidité," + 0.005*"accueil," + 0.005*"lécoute," + 0.005*"garanties," + 0.004*"r
este," + 0.004*"cher," + 0.004*"lolivier," + 0.004*"internet," + 0.004*"agréable," + 0.003*"p
rise," + 0.003*"écoute,"')]
```

Analyse de sentiment

Pour l'analyse de sentiment, depuis la librairie vaderSentiment_fr on a importé SentimentIntensityAnalyzer. On va initialiser une nouvelle variable qui sera remplie de la polarity_scores et qui donne les 3 scores pour chaque commentaire : positivité, neutralité et négativité.

Puis on a créé une nouvelle variable sentiment_vader qui nous donne le sentiment présent dans le commentaire.

On va ensuite afficher un diagramme circulaire qui nous donne les pourcentages de chaque sentiment.
On peut donc voir que le sentiment prédominant est le positif avec un taux à 68,5% suivi du sentiment négatif avec un taux de 26,7% et enfin le sentiment neutre avec un taux à 4,8%.



Apprentissage Supervisé

Avant de procéder à la création de modèles, nous devons transformer nos données qualitatives en valeurs entières. Par conséquent, un OneHotEncoding a été effectué sur les variables que nous avons traité en amont.

Par la suite, sur la base de l'analyse que nous avons faite, nous pouvons lancé les modèles de régression. Pour cela, commençons par définir X et y. X correspondant au dataset auquel on retire les variables qualitatives qu'on n'a pas pu encoder comme le commentaire car cela aurait créé des milliers de colonnes en plus et y désigne la target donc la note attribuée.

Puisque la prédiction de *note* est un problème de classification mais on peut aussi lancer des modèles de régressions et arrondir le résultat pour avoir un entier en résultat, et pour cela nous avons expérimenté les modèles suivants :

- DecisionTreeRegressor
- BaggingRegressor
- GradientBoostingRegressor
- RandomForestRegressor

Pour cela, nous avons divisé l'ensemble de données en deux parties : l'une pour l'apprentissage et l'autre pour la prédiction. Le seuil de partage n'est pas fixé et est utilisé comme un hyperparamètre.

Pour tuner nos modèles, nous avons fait plusieurs tests en variant les hyperparamètres spécifiques et en utilisant une technique simple de validation croisée pour chaque algorithme.

Apprentissage Supervisé

Après l'obtention de 84 modèles nous sélectionnons celui dont le RMSE sur le test est le plus bas. Le meilleur modèle est le Bagging Regressor avec comme split 50% et le nombre d'estimateurs fixé à 60 avec lequel on obtient un RMSE de 0.398.

	Methode	Modèle	Paramètres	RMSE_train	RMSE_test	R2_train	R2_test
0	BaggingRegressor	(DecisionTreeRegressor(random_state=1973955297...	{'split': 0.5, 'n_estimators': 60.0}	0.400412	0.398196	0.931061	0.932926
1	BaggingRegressor	(DecisionTreeRegressor(random_state=791304662)...	{'split': 0.4, 'n_estimators': 60.0}	0.399863	0.401982	0.931182	0.932030
2	BaggingRegressor	(DecisionTreeRegressor(random_state=1469510583...	{'split': 0.4, 'n_estimators': 50.0}	0.399391	0.405950	0.931344	0.930682
3	BaggingRegressor	(DecisionTreeRegressor(random_state=55281235),...	{'split': 0.5, 'n_estimators': 50.0}	0.403231	0.406356	0.930087	0.930149
4	BaggingRegressor	(DecisionTreeRegressor(random_state=2047538114...	{'split': 0.5, 'n_estimators': 40.0}	0.408072	0.406788	0.928399	0.930000
5	BaggingRegressor	(DecisionTreeRegressor(random_state=679699144)...	{'split': 0.6, 'n_estimators': 40.0}	0.412677	0.407702	0.926615	0.929603
6	BaggingRegressor	(DecisionTreeRegressor(random_state=987723123)...	{'split': 0.6, 'n_estimators': 30.0}	0.410772	0.412637	0.927291	0.927889
7	BaggingRegressor	(DecisionTreeRegressor(random_state=411284261)...	{'split': 0.6, 'n_estimators': 20.0}	0.426971	0.424978	0.921444	0.923511
8	BaggingRegressor	(DecisionTreeRegressor(random_state=2127178618...	{'split': 0.4, 'n_estimators': 10.0}	0.468097	0.461502	0.905692	0.910412

Nous avons aussi lancé les modèles de classification suivants : *Logistic Regression, Decision Tree,KNN, Naive Bayes, LDA et SVC*
Ces modèles donnent un score RMSE trop élevée. Le meilleur RMSE est de 1.2397091535374898 atteint avec le LDA avec un slip de 50%

Conclusion