SORBONNE UNIVERSITE

Faculté des Sciences et Ingénerie

Master Informatique - M1

Parcours Données Apprentissage Connaissance DAC

RITAL

Bag-of-Words project

Compte rendu

Auteurs

Ben Kabongo
Sofia Borchani

Encadrant

Nicolas Thome

Mars 2023

Dans ce projet, nous nous sommes concentrés sur l'utilisation de la technique Bag-of-words pour analyser deux types de données différentes : les discours de deux anciens présidents français et les avis sur des films.

Dans la première partie de notre projet, nous avons cherché à déterminer avec précision le locuteur d'un discours de président, en utilisant la méthode du Bag-of-words. Plus spécifiquement, nous avons analysé des discours de deux présidents français notables, Jacques Chirac et François Mitterand. En effectuant une analyse de texte approfondie de leurs discours, nous avons pu mettre en évidence des caractéristiques distinctes qui nous ont permis de déterminer avec précision le locuteur d'un discours donné.

Dans la seconde partie de notre projet, nous avons appliqué la même technique du Bag-of-words pour analyser des avis sur des films, en nous concentrant sur la prédiction de leur tonalité (positive ou négative). Nous avons collecté un grand nombre d'avis sur des films de différentes sources et avons utilisé des méthodes de prétraitement pour nettoyer et normaliser les données. Ensuite, nous avons appliqué différentes techniques de modélisation pour prédire avec précision si un avis donné était positif ou négatif.

Dans ce rapport, nous détaillerons les méthodes et les résultats de nos analyses, en fournissant des exemples concrets de nos résultats.

Table des matières

Table des figures

Liste des tableaux

1 Locuteurs

1.1 Prise en main des données

Dans cette partie, nous avons effectué quelques analyses exploratoires des données.

1.1.1 Distribution des classes

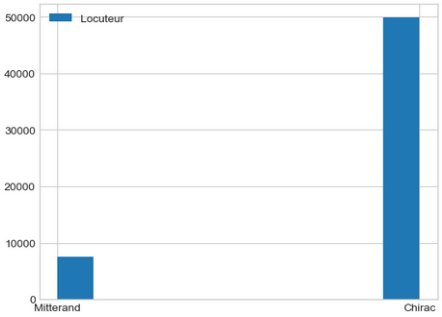

FIGURE 1 – Distribution des classes - Locuteurs

Il y a un fort déséquilibre entres les classes.

1.1.2 Longueur des documents

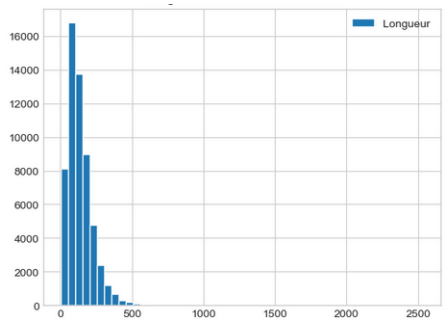

FIGURE 2 – Longueur des documents - Locuteurs

La longueur des documents varie entre 4 et 2500. La majorité des documents ont une longueur allant d'une dizaine de caractère à 500 caractères.

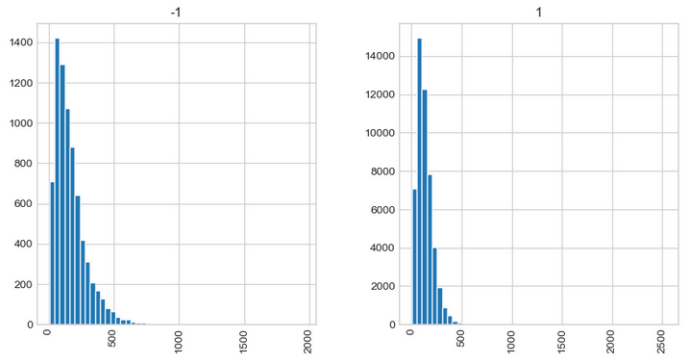

FIGURE 3 – Longueur des documents par classes - Locuteurs

1.2 Prétraitements

Nous avons établi diverses fonctions de prétraitement :

1.2.1 Suppression de la ponctuation

delete_punctuation qui supprime la ponctuation dans un texte en remplaçant tous les symboles de ponctuation par des espaces, puis en fusionnant tous les espaces consécutifs en un seul.

1.2.2 Remplacement des majuscules

replace_maj_word qui remplace les mots en majuscules d'un texte par un jeton spécifique <MAJ>.

1.2.3 Suppression des chiffres

delete_digit qui supprime tous les chiffres d'un texte en utilisant une expression régulière pour trouver tous les nombres (0-9) et en les remplaçant par une chaîne vide.

1.2.4 Conservation de la première ligne

first_line qui retourne la première ligne d'un texte.

1.2.5 Conservation de la dernière ligne

last_line qui retourne la dernière ligne d'un texte.

1.2.6 Suppression des balises

delete_balise qui supprime toutes les balises HTML d'un texte en utilisant une expression régulière pour rechercher des motifs commençant par "<" et se terminant par ">".

1.2.7 Stemming

stem qui effectue la racinisation (ou "stemming" en anglais) des mots dans un texte en utilisant le stemmer français de NLTK.

1.2.8 Lemmatisation

lemmatize qui effectue la lemmatisation des mots dans un texte en utilisant le lemmatiseur WordNet de NLTK.

1.3 Extraction du vocabulaire

Dans cette partie, nous allons présenter les résultats d'étude du vocabulaire.

1.3.1 Vocabulaire initial

Le vocabulaire comporte 28524 mots différents.

FIGURE 4 – Vocaublaire initial - Locuteurs

1.3.2 Odds ratio

L'odds ratio est une mesure statistique qui est souvent utilisée pour évaluer l'association entre la fréquence des mots dans deux corpus de textes appartenant à deux classes différentes.

Plus précisément, l'odds ratio mesure le rapport des chances de trouver un mot donné dans l'un des corpus par rapport à l'autre. Cela peut aider à déterminer quels mots sont les plus associés à chaque classe.

FIGURE 5 – 100 mots les plus discriminants au sens de odds ratio - Locuteurs

1.3.3 Loi de Zipf

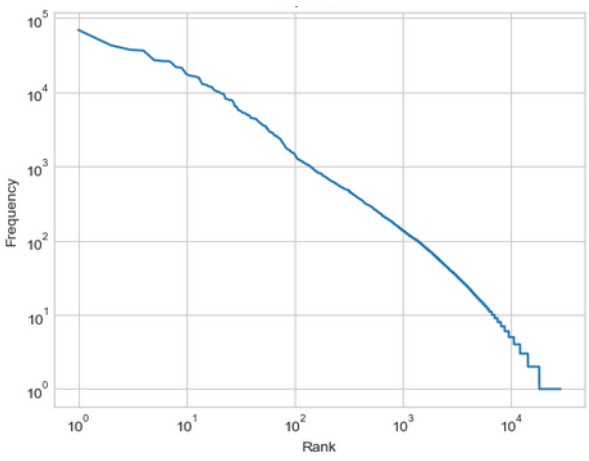

FIGURE 6 – Ziph law - Locuteurs

La loi de Zipf est une loi statistique empirique qui décrit la distribution des fréquences des mots dans un

texte donné. Elle stipule que dans un corpus de texte donné, la fréquence d'un mot donné est inversement proportionnelle à son rang de fréquence.

En d'autres termes, si l'on classe tous les mots d'un texte en fonction de leur fréquence décroissante, le mot le plus fréquent apparaît environ deux fois plus souvent que le deuxième mot le plus fréquent, trois fois plus souvent que le troisième, et ainsi de suite.

La distribution de la fréquence des mots du corpus des locuteurs suit effectivement la loi de Zipf.

1.3.4 WordClouds du corpus

FIGURE 7 – Mots les plus fréquents sans les stop words - Locuteurs

FIGURE 8 – Mots les plus fréquents - Mitterand

FIGURE 9 – Mots les plus fréquents - Chirac

1.4 Etude de temps et comparaison

Dans cette partie, nous allons présenter les résultats d'études des différents classifieurs : présenter les performances de chacune et conclure sur un choix de classifieur pour la suite de cette partie.

Nous avons fixé quelques tailles de vocabulaire variant entre 10 et 100000. Les données ont été divisées en données d'apprentissage et de test, pour chaque taille de vocabulaire donnée.

Dans la suite, nous avons regardé pour différents classifieurs : le temps d'apprentissage, l'accuracy, le f1-score et l'accuracy balancée en fonction de la taille du vocabulaire.

Nous avons considéré la régression logistique comme classifieur de référence.

1.4.1 Régression logistique

Vocabulary Size	Learning Time	Accuracy	F1 Score	Balanced Accuracy
10	25.91	0.60	0.73	0.52
100	44.48	0.69	0.80	0.58
1000	103.49	0.75	0.84	0.63
10000	194.94	0.79	0.87	0.66
20000	402.78	0.80	0.87	0.66
40000	352.23	0.80	0.87	0.67
50000	482.92	0.80	0.87	0.66
80000	711.42	0.80	0.87	0.66
100000	847.25	0.80	0.87	0.66

TABLE 1 – Régression Logistique Résultats - Locuteurs

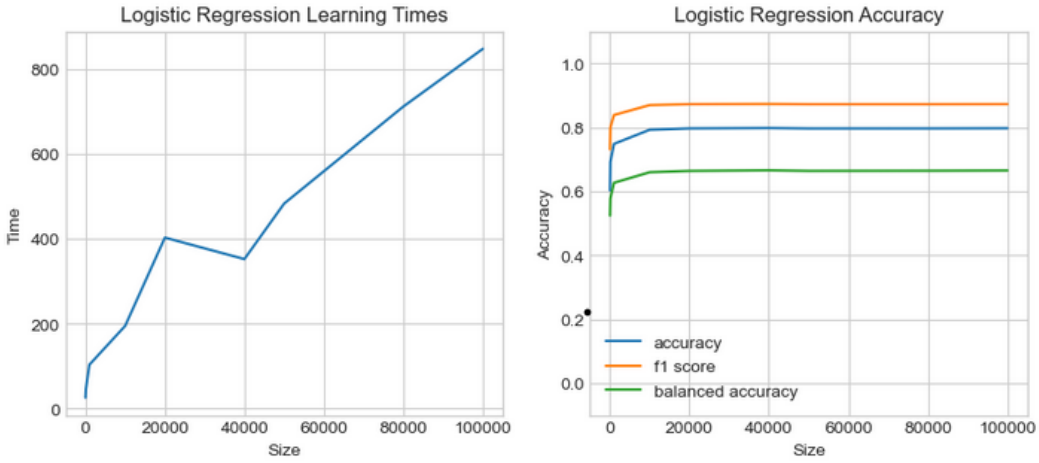

FIGURE 10 – Régression logistique - Locuteurs

1.4.2 Naives Bayes

Le modèle Naive Bayes s'entraine relativement plus rapidement que le modèle de Regression Logistique, cependant on constante que les performances du modèle de Regression Logistique sont nettement meilleures que celles du modèle Naive Bayes.

Vocabulary Size	Learning Time	Accuracy	F1 Score	Balanced Accuracy
10	3.91	0.31	0.37	0.52
100	1.24	0.60	0.72	0.56
1000	1.45	0.72	0.82	0.62
10000	1.90	0.78	0.86	0.66
20000	1.99	0.77	0.86	0.66
40000	2.30	0.76	0.85	0.66
50000	2.54	0.76	0.84	0.66
80000	3.16	0.75	0.83	0.65
100000	3.38	0.74	0.83	0.65

TABLE 2 – Naive Bayes Résultats - Locuteurs

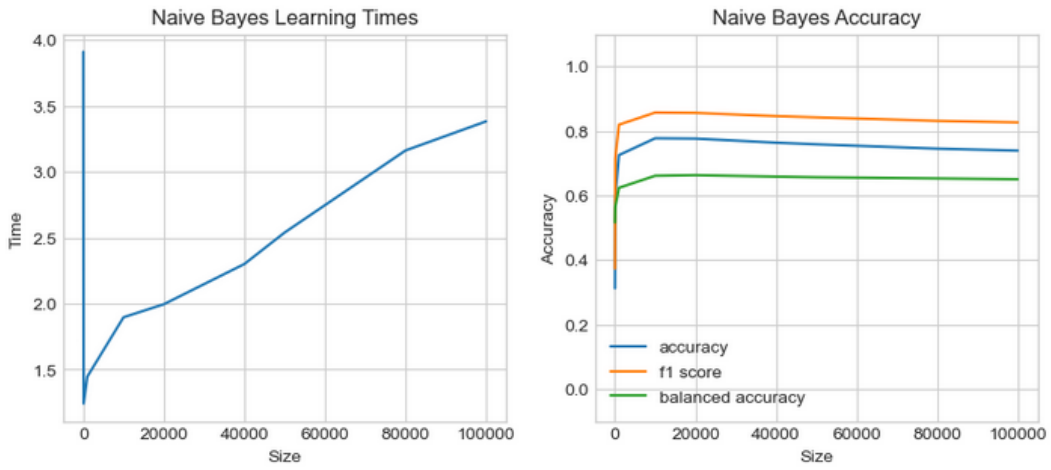

FIGURE 11 – Naive Bayes - Locuteurs

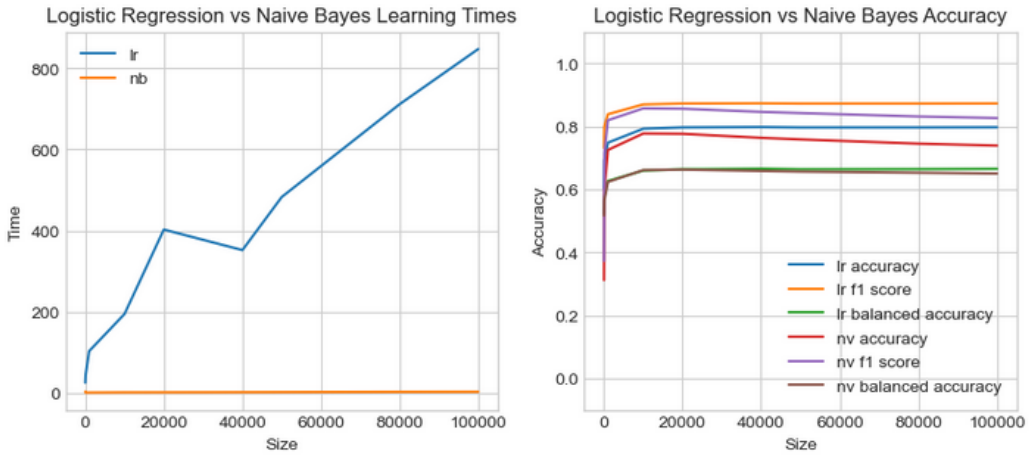

FIGURE 12 – Régression Logistique vs Naive Bayes - Locuteurs

1.4.3 SVM

Le temps d'apprentissage du SVM est beaucoup plus long que le temps d'apprentissage de la regression logistique. Cependant, les performances des deux modèles sont presque équivalentes.

Vocabulary Size	Learning Time	Accuracy	F1 Score	Balanced Accuracy
10	2452.59	0.68	0.80	0.53
100	4081.51	0.69	0.80	0.58
1000	7278.06	0.75	0.84	0.63
10000	15214.56	0.79	0.87	0.65
20000	18212.55	0.79	0.87	0.66
40000	20603.91	0.79	0.87	0.66
50000	21901.52	0.79	0.87	0.66
80000	23215.14	0.79	0.87	0.66
100000	23639.96	0.79	0.87	0.66

TABLE 3 – Support Vector Machine Résultats - Locuteurs

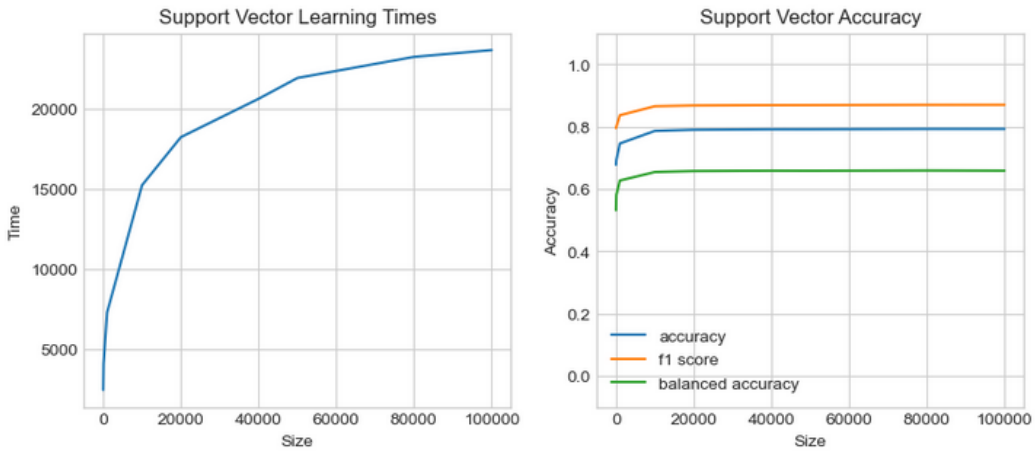

FIGURE 13 – Suport Vector Machine - Locuteurs

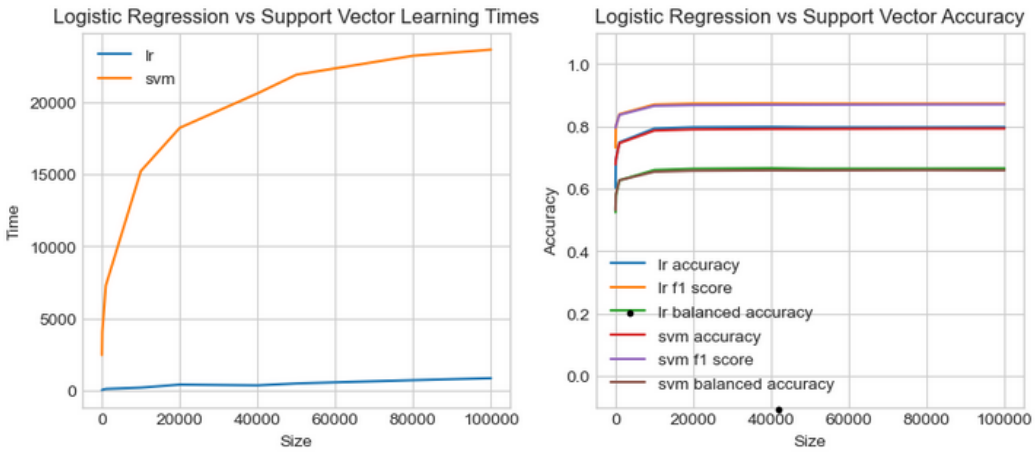

FIGURE 14 – Régression Logistique vs Suport Vector Machine - Locuteurs

1.5 Etude de la validation croisée

1.5.1 Validation croisée / Train-Test split

Dans cette partie, nous avons comparé les résultats de la validation croisée à ceux d'un train-test split simple, sur base des métriques d'accuracy, précision, rappel et f1-score. Nous avons constaté une différence peu significative au niveau des résultats.

Métrique	Train/Test	Validation Croisée
Accuracy	0.771	0.767
Precision	0.761	0.760
Recall	0.784	0.781
f1-score	0.773	0.771

TABLE 4 – Train-Test split vs Validation croisée - Locuteurs

1.5.2 Validation croisée : stabilité

Dans cette partie, nous avons étudié la stabilité de la validation croisée, sur base des métriques accuracy, précision, rappel et f1-score.

Au regard des résultats, nous avons conclu que la validation croisée est bel et bien stable.

Métrique	Accuracy	Précision	Recall	F1-Score
Moyenne	0.7761	0.7719	0.7840	0.7778
Écart-type	0.0010	0.0014	0.0016	0.0010

TABLE 5 – Stabilité de la validation croisée - Locuteurs

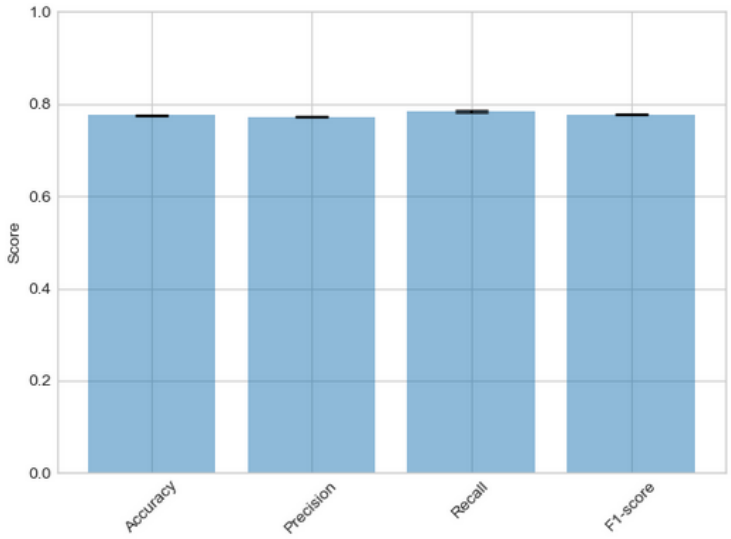

FIGURE 15 – Stabilité de la validation croisée

1.6 Variantes de BoW et Machine Learning

1.6.1 Expérimentations

Dans cette partie nous avons mené differentes expérimentations en variant un certain nombre de paramètres. Les vectoriseurs utilsés sont CountVectorizer et TfidfVectorizer, le classifieur utilisé est LogisticRegression. On utilise également RandomUnderSampler pour rééquilibrer les classes en apprentissage.

Pour CountVectorizer et TfidfVectorizer, nous avons regardé les paramètres suivants :

— **lowercase** : conversion en minuscule ou non du corpus.

— **binary** : BoW binaire ou non.

— **stopwords** : liste des stopwords à supprimer.

— **max_df** : seuil de fréquence maximum pour les mots qui apparaissent dans le texte.

— **min_df** : seuil de fréquence minimum pour les mots qui apparaissent dans le texte.

— **max_features** : taille du vocabulaire.

— **ngram_range** : intervalle des ngrammes.

Différents jeux de test :

Légende

1. CountVectorizer
2. TfidfVectorizer
3. CountVectorizer : suppression de la ponctuation
4. TfidfVectorizer : suppression de la ponctuation
5. CountVectorizer : suppression des chiffres
6. TfidfVectorizer : suppression des chiffres
7. CountVectorizer : suppression des balises
8. TfidfVectorizer : suppression des balises
9. TfidfVectorizer : première ligne
10. TfidfVectorizer : dernière ligne
11. TfidfVectorizer : stemming
12. TfidfVectorizer : remplacement des majuscules

V	Accuracy	F1-Score -1	F1-Score 1	AUC
1	0.7933	0.49	0.87	0.8604
2	0.7965	0.49	0.87	0.8580
3	0.8035	0.50	0.88	0.8708
4	0.8054	0.50	0.88	0.8670
5	0.8031	0.50	0.88	0.8706
6	0.8044	0.50	0.88	0.8669
7	0.8028	0.50	0.88	0.8710
8	0.8058	0.50	0.88	0.8674
9	0.8024	0.50	0.88	0.8640
10	0.7969	0.49	0.87	0.8601
11	0.7950	0.49	0.87	0.8571
12	0.8044	0.50	0.88	0.8662
13	0.7973	0.49	0.87	0.8604
14	0.7759	0.46	0.86	0.8415
15	0.7850	0.48	0.86	0.8610
16	0.7712	0.46	0.85	0.8393
17	0.7947	0.49	0.87	0.8559
18	0.8105	0.51	0.88	0.8729
19	0.7642	0.42	0.85	0.7955
20	0.7690	0.47	0.85	0.8611

TABLE 6 – Locuteurs Résultats

13. TfidfVectorizer : binary
14. TfidfVectorizer : ngram_range=(2, 2)
15. TfidfVectorizer : ngram_range=(1, 2)
16. TfidfVectorizer : ngram_range=(2, 3)
17. TfidfVectorizer : max_features=5000
18. TfidfVectorizer : suppression de la ponctuation, min_df=10, max_df=0.75, binary , ngram_range=(1, 2)
19. TfidfVectorizer, suppression de la ponctuation, première ligne, suppression des balises, max_df=0.5, binary , ngram_range=(3, 3)
20. TfidfVectorizer : suppression de la ponctuation, suppression des chiffres, stemming, binary , ngram_range=(1, 3)

1.6.2 Grid search : recherche plus exhaustive des paramètres optimaux

Les différentes expérimentations détaillées dans le point précédent ont été un indicateur sur l'agencement des paramètres qui fonctionnaient le mieux.

Dans cette partie, nous allons détailler la procédure du grid search que nous avons mis en place pour le choix des paramètres optimaux pour ce problème.

Le nombre de paramètres, de classifieurs, de vectoriseurs étant assez grand, la combinaison des paramètres peut ainsi très vite devenir très exponentielle. En fonction de petites expériences menées et expliquées plus haut, nous avons dû faire différents choix.

Nous avons testé quelques classifieurs tels que les SVM, les réseaux de neurones ; le temps d'exécution s'est averé très long et parfois pour des résultats moins bien que la régression logistique. Pour ce qui est du classifieur, notre choix s'est donc porté sur la **régression logistique**.

Nous avons également préféré choisir **TfidfVectorizer**, plutôt que CountVectorizer : en général les résultats du premier vectoriseur étaient mieux que le second. De plus le paramètre *use_idf* du TfidfVectorizer permet d'utiliser ou non la partie inverse document frenquency.

Nous avons démontré également que la validation croisée et le train/test split avaient des résultats presque

similaires. Ainsi, pour aussi gagner du temps, pour chaque test de combinaison de paramètres, nous avons choisi de faire du **train/test split**.

Le problème portant sur des données dont les classes ne sont pas équilibrées, nous avons utilisé **RandomUnderSampler** pour rééquilibrer les données en phase d'apprentissage pour chaque combinaison de paramètres.

Enfin, nous avons combiné différents paramètres de TfidfVectorizer, de LogisticRegression et différentes fonction de préprocessing pour le choix de la meilleure combinaison.

Nous avons décidé que la meilleure combinaison était celle qui maximisait le score f1 de la classe minoritaire.

Paramètres de la régression logistique :

C	0.1	1	10	100
Penality	None	L2		

TABLE 7 – Régression logistique - Locuteurs paramètres

Paramètres du TfidfVectorizer :

Stopwords	Français	None				
max_df	0.5	0.6	0.7	0.8	0.9	1.0
min_df	2	3	5	10		
ngram_range	(1, 1)	(1, 2)	(1, 3)	(2, 2)	(2, 3)	
binary	False	True				
lowercase	False	True				
use_idf	False	True				
sublinear_tf	False	True				
max_features	None	2 000	5 000	10 000	20 000	50 000

TABLE 8 – Locuteurs Vectorizer - paramètres

Résultats qui maximisent le score f1 de la classe minoritaire :

Prétraitement	**Valeur**
Suppression de la ponctuation	True
Suppression des chiffres	True
Remplacement des majuscules	True
Régression logistique	**Valeur**
C	1
Penality	None
TfidfVectorizer	**Valeur**
Stopwords	Français
max_df	0.5
min_df	5
ngram_range	(1, 3)
binary	True
lowercase	True
use_idf	True
sublinear_tf	True
max_features	None
Résultats	**Score**
F1 score classe -1	**0.520128087831656**
F1 score	0.8871679036248251
AUC	0.7961797917256521
Accuracy	0.8172951319341636
Balanced accuracy	0.7961797917256521

TABLE 9 – Résultats - F1 Mitterand maximum

La modification des paramètres de la régression logistique pour cette combinaison des paramètres ne modifie

pas grandement le résultat. C'est également le cas de quelques paramètres du vectoriseur tel que le nombre de features qui, fixé à 20 000 ou 50 000 donne des résultats similaires.

Résultats qui maximisent l'AUC :

Prétraitement	**Valeur**
Suppression de la ponctuation	True
Remplacement des majuscules	True
Régression logistique	**Valeur**
C	1
Penality	None
TfidfVectorizer	**Valeur**
Stopwords	Français
max_df	0.9
min_df	2
ngram_range	(1, 2)
binary	True
lowercase	False
use_idf	True
sublinear_tf	True
max_features	None
Résultats	**Score**
F1 score classe -1	0.5160714285714285
F1 score	0.8827220599372498
AUC	**0.7981452662877283**
Accuracy	0.8111991639815379
Balanced accuracy	0.7981452662877283

Table 10 – Résultats - AUC maximum

2 Movies

2.1 Prise en main des données

Dans cette partie, nous avons effectué quelques analyses exploratoires des données.

2.1.1 Distribution des classes

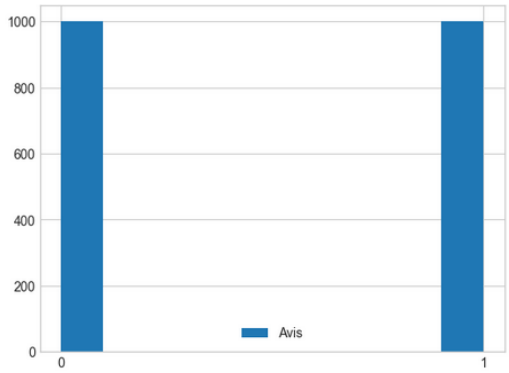

FIGURE 16 – Distribution des classes - Movies

La répartition des classes est équilibrée.

2.1.2 Longueur des documents

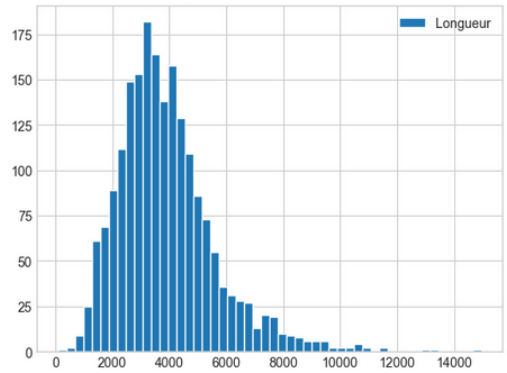

FIGURE 17 – Longueur des documents - Movies

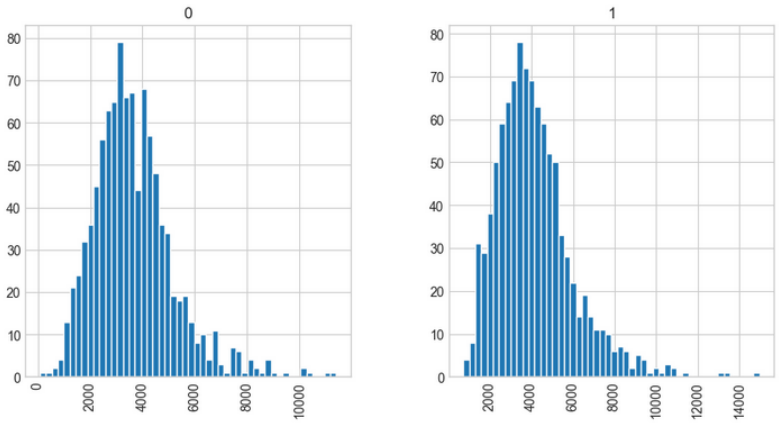

FIGURE 18 – Longueur des documents par classes - Movies

2.2 Prétraitements

Les prétraitements faits pour cette partie du projet correspondent à ceux faits dans la partie précédente. Quelques adaptations ont été faites : le corpus de la première partie est en français, celui de la seconde en anglais.

2.3 Extraction du vocabulaire

Dans cette partie, nous allons présenter les résultats d'étude du vocabulaire.

2.3.1 Vocabulaire initial

Le vocabulaire comporte 39659 mots différents.

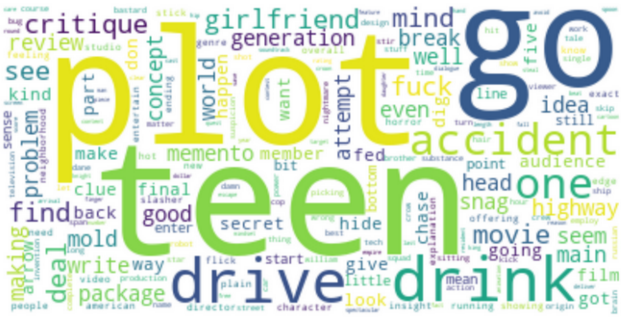

FIGURE 19 – Vocaublaire initial - Movies

2.3.2 Odds ratio

FIGURE 20 – 100 mots les plus discriminants au sens de odds ratio - Movies

2.3.3 Loi de Zipf

La distribution des fréquences des mots du corpus suit bel et bien la loi de Zipf.

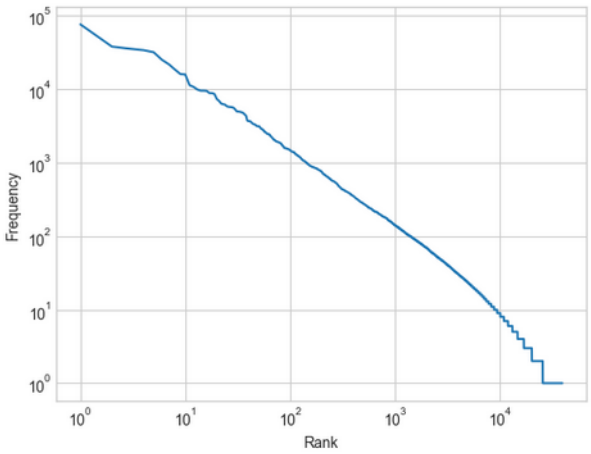

FIGURE 21 – Ziph law - Movies

2.3.4 WordClouds du corpus

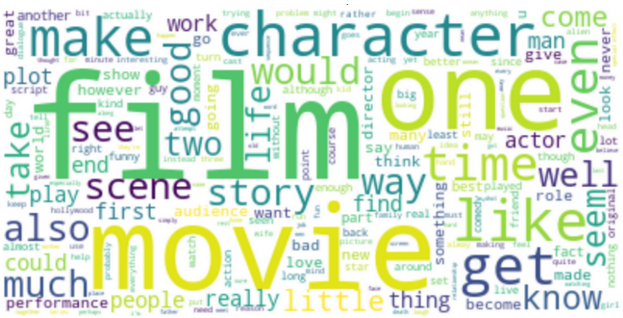

FIGURE 22 – Mots les plus fréquents sans les stop words - Movies

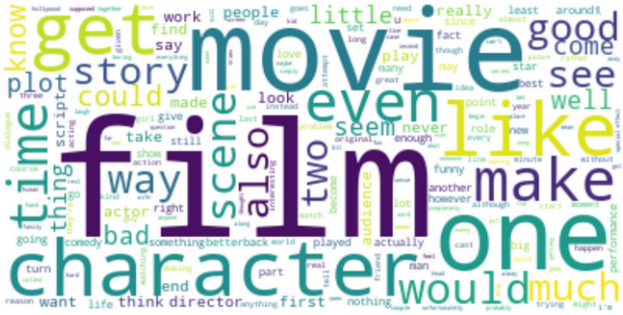

FIGURE 23 – Mots les plus fréquents - Negatif

FIGURE 24 – Mots les plus fréquents - Positif

2.4 Etude de temps et comparaison

Dans cette partie, nous allons présenter les résultats d'études des différents classifieurs : présenter les performances de chacune et conclure sur un choix de classifieur pour la suite de cette partie.

Pour ce faire, nous avons établi des tailles de vocabulaire allant de 10 à 100 000, en divisant les données en ensembles d'apprentissage et de test pour chaque taille de vocabulaire.

Nous avons ensuite examiné plusieurs indicateurs de performance pour différents classifieurs, tels que le temps d'apprentissage, l'accuracy, le f1-score et l'accuracy balancée en fonction de la taille du vocabulaire.

Nous avons considéré la régression logistique comme classifieur de référence.

2.4.1 Régression Logistique

Vocabulary Size	Learning Time	Accuracy	F1 Score	Balanced Accuracy
10	65.86	0.64	0.63	0.64
100	116.90	0.73	0.73	0.73
1000	289.26	0.79	0.79	0.79
10000	784.67	0.85	0.85	0.85
20000	1533.49	0.86	0.86	0.86
40000	1640.81	0.87	0.87	0.88
50000	2051.39	0.87	0.87	0.87
80000	2839.97	0.87	0.87	0.87
100000	3477.19	0.87	0.87	0.87

TABLE 11 – Régression Logistique Résultats - Movies

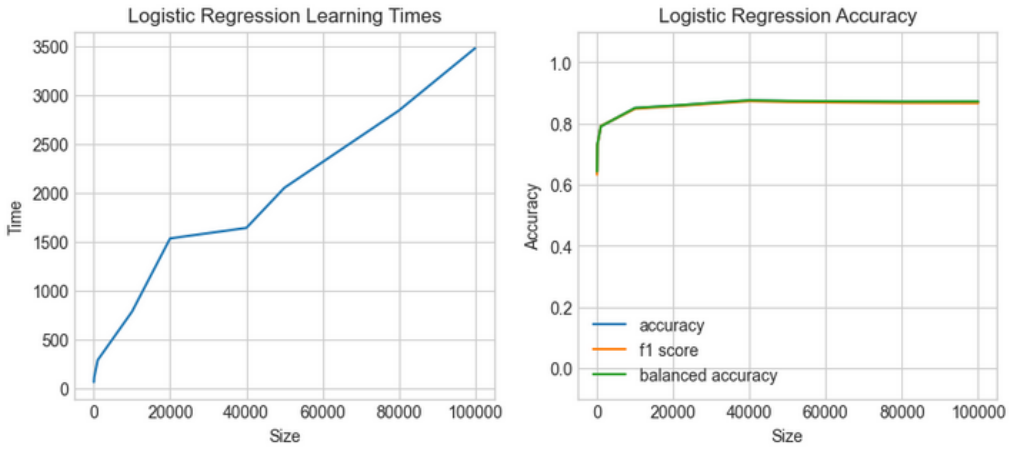

FIGURE 25 – Régression Logistique - Movies

2.4.2 Naives Bayes

Le modèle Naive Bayes s'entraine relativement plus rapidement que le modèle de Regression Logistique, on constante que les performances du modèle de Regression Logistique et celles du modèle Naive Bayes sont semblables.

Vocabulary Size	Learning Time	Accuracy	F1 Score	Balanced Accuracy
10	2.99	0.64	0.64	0.64
100	2.05	0.67	0.69	0.67
1000	3.99	0.75	0.76	0.75
10000	6.55	0.85	0.85	0.85
20000	8.98	0.84	0.84	0.84
40000	13.96	0.85	0.85	0.85
50000	14.96	0.85	0.85	0.85
80000	18.95	0.86	0.85	0.86
100000	19.72	0.85	0.85	0.85

TABLE 12 – Naive Bayes Résultats - Movies

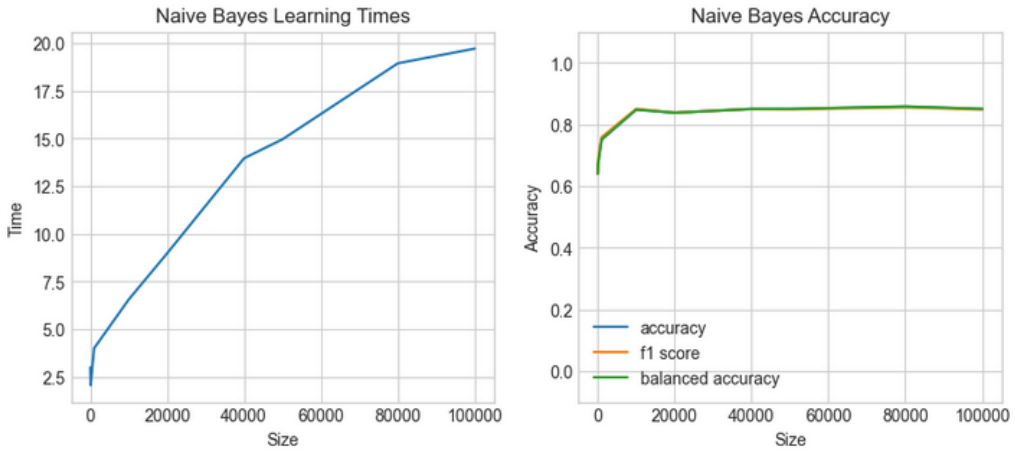

FIGURE 26 – Naive Bayes - Movies

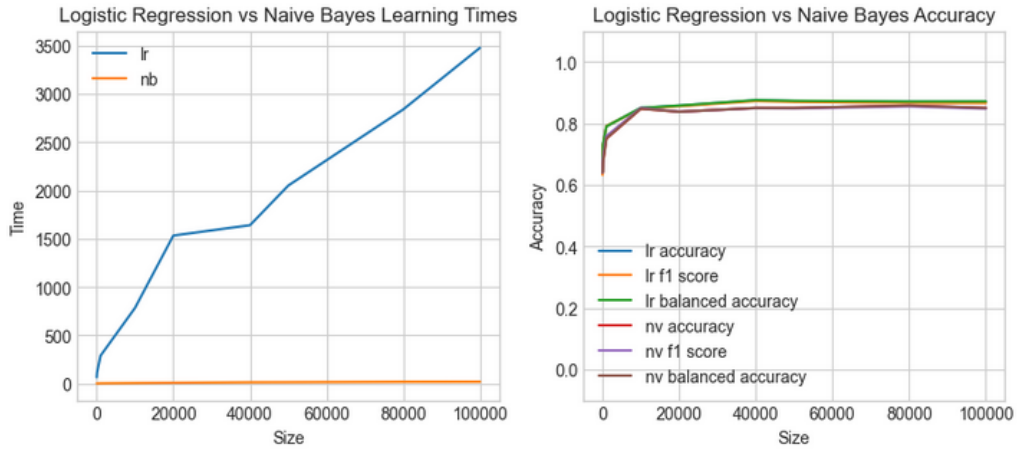

FIGURE 27 – Régression Logistique vs Naive Bayes - Movies

2.4.3 SVM

Le temps d'apprentissage du SVM est beaucoup plus long que le temps d'apprentissage de la regression logistique. Cependant, les performances des deux modèles sont presque équivalentes bine que celles de la Regression Logistique soient légèrement meilleures que celles du SVM.

Vocabulary Size	Learning Time	Accuracy	F1 Score	Balanced Accuracy
10	139.97	0.67	0.64	0.67
100	1603.18	0.72	0.70	0.72
1000	6753.02	0.78	0.76	0.78
10000	12327.69	0.79	0.78	0.80
20000	15554.81	0.79	0.77	0.80
40000	18689.86	0.79	0.77	0.80
50000	19878.94	0.78	0.77	0.79
80000	21860.02	0.78	0.77	0.79
100000	23474.90	0.78	0.77	0.79

TABLE 13 – Support Vector Machine Résultats - Movies

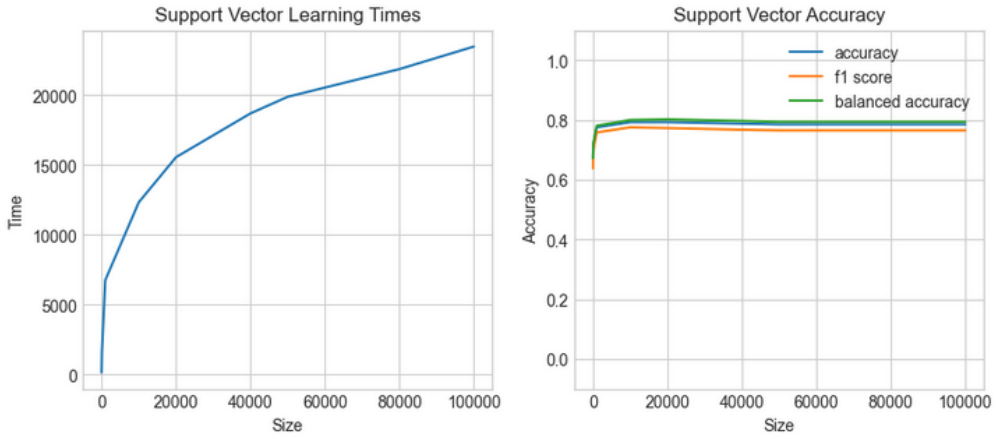

FIGURE 28 – Suport Vector Machine - Movies

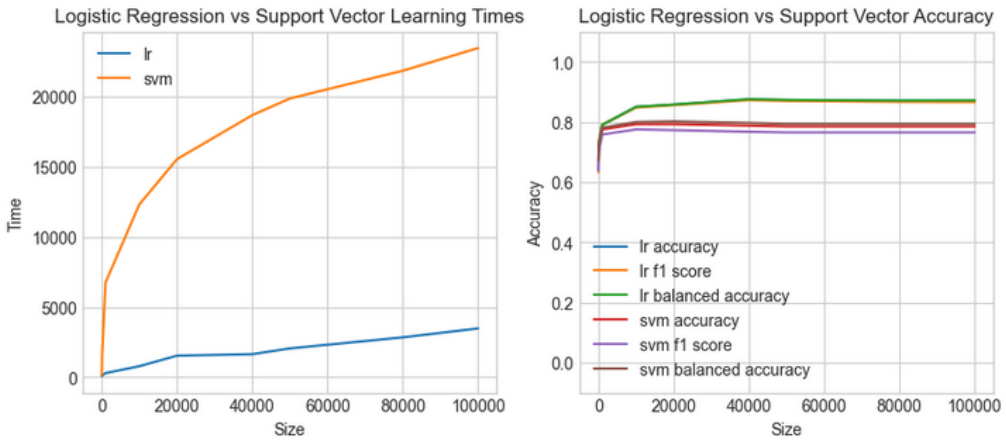

FIGURE 29 – Regression Logistique vs Suport Vector Machine - Movies

2.5 Validation croisée

2.5.1 Validation croisée / Train-Test split

Dans cette partie, nous avons effectué une comparaison entre les résultats obtenus grâce à la validation croisée et ceux obtenus grâce à un train-test split, en utilisant des métriques telles que l'accuracy, la précision, le rappel et le f1-score. Nous avons remarqué que la différence entre ces deux méthodes était peu significative au niveau des résultats obtenus.

Métrique	Train/Test	Validation Croisée
Accuracy	0.85	0.8345
Precision	0.8279	0.8444
Recall	0.8856	0.8200
f1-score	0.8558	0.8317

TABLE 14 – Train-Test split vs Validation croisée - Movies

2.5.2 Validation croisée : stabilité

Dans cette partie, nous avons examiné la stabilité de la validation croisée en utilisant des métriques telles que l'accuracy, la précision, le rappel et le f1-score.

À la lumière des résultats obtenus, nous avons pu conclure que la validation croisée était effectivement stable.

Métrique	Accuracy	Précision	Recall	F1-Score
Moyenne	0.8384	0.8444	0.8314	0.8373
Écart-type	0.0054	0.0040	0.0083	0.0059

TABLE 15 – Stabilité de la validation croisée - Movies

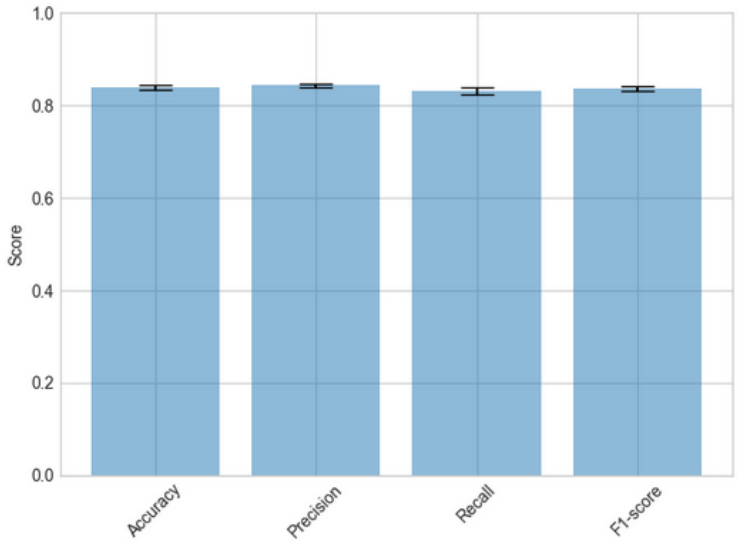

FIGURE 30 – Stabilité de la validation croisée - Movies

2.6 Variantes de BoW et Machine Learning

2.6.1 Expérimentations

Dans cette partie, nous avons mené diverses expérimentations en variant un certain nombre de paramètres. Les vecteurs de texte ont été obtenus à l'aide de deux vectoriseurs couramment utilisés, à savoir le CountVectorizer et le TfidfVectorizer.

Différents jeux de test :

V	Accuracy	F1 score	AUC
1	0.8525	0.8513	0.9188
2	0.8575	0.8564	0.9141
3	0.8375	0.8387	0.9300
4	0.85	0.8514	0.9418
5	0.84	0.8407	0.9192
6	0.8425	0.8467	0.9051
7	0.86	0.8620	0.9269
8	0.865	0.8669	0.9343
9	0.8525	0.8513	0.9188
10	0.8575	0.8564	0.9141
11	0.8475	0.8486	0.9191
12	0.8625	0.8661	0.9257
13	0.7425	0.7541	0.8415
14	0.83	0.8341	0.9045
15	0.8825	0.8810	0.9482
16	0.8875	0.8888	0.9502
17	0.8925	0.8905	0.9519
18	0.9	0.9009	0.9544
19	0.8525	0.8513	0.9196
20	0.855	0.8542	0.9148
21	0.9	0.8979	0.9539
22	0.8925	0.8927	0.9528

TABLE 16 – Movies Résultats

Légende :

1. CountVectorizer
2. TfidfVectorizer
3. CountVectorizer, binary
4. TfidfVectorizer, binary
5. CountVectorizer, stop_words
6. TfidfVectorizer, stop_words
7. CountVectorizer, binary, stop_words
8. TfidfVectorizer, binary, stop_words
9. CountVectorizer, lower
10. TfidfVectorizer, lower
11. CountVectorizer, max_df=.75
12. TfidfVectorizer, max_df=.75
13. CountVectorizer, lowercase, binary, ngram_range=(1, 2), max_features=1000
14. TfidfVectorizer, lowercase, binary, ngram_range=(1, 2), max_features=1000
15. CountVectorizer, lowercase, binary, ngram_range=(1, 2), max_features=40_000
16. TfidfVectorizer, lowercase, binary, ngram_range=(1, 2), max_features=40_000
17. CountVectorizer, lowercase, binary, ngram_range=(1, 3), max_features=40_000
18. TfidfVectorizer, lowercase, binary, ngram_range=(1, 3), max_features=40_000
19. CountVectorizer, suppression de la ponctuation
20. TfidfVectorizer, suppression de la ponctuation
21. CountVectorizer, suppression de la ponctuation
22. TfidfVectorizer, suppression de la ponctuation

2.6.2 Grid search : recherche plus exhaustive des paramètres optimaux

Afin de trouver les paramètres optimaux, nous avons fait une recherche plus exhaustive. Nos choix se sont encore portés sur la régression logistique et le vectoriseur TfidfVectorizer, dont nous avons étudié différentes

combinaisons des paramètres.

Paramètres de la régression logistique :

C	0.1	1	10	100
Penality	None	L2		

Table 17 – Régression logistique - Movies paramètres

Paramètres du TfidfVectorizer :

Stopwords	English	None				
max_df	0.5	0.6	0.7	0.8	0.9	1.0
min_df	2	3	5	10		
ngram_range	(1, 1)	(1, 2)	(1, 3)	(2, 2)	(2, 3)	
binary	False	True				
lowercase	False	True				
use_idf	False	True				
sublinear_tf	False	True				
max_features	None	2 000	5 000	10 000	20 000	50 000

Table 18 – Movies Vectorizer - paramètres

Résultats qui maximisent le score f1 :

Prétraitement	**Valeur**
Suppression de la ponctuation	True
Suppression des chiffres	True
Remplacement des majuscules	True
Suppression des balises	True
Régression logistique	**Valeur**
C	1
Penality	None
TfidfVectorizer	**Valeur**
Stopwords	None
ngram_range	(1, 3)
binary	True
lowercase	True
use_idf	True
sublinear_tf	True
max_features	40_000
Résultats	**Score**
F1 score	0.9032258064516129
AUC	0.95475
Accuracy	0.9025

Table 19 – Résultats Movies - F1 maximum