Projet NLP

Classification et Génération de Textes de Business Reviews

Rapport de Projet – Groupe 6

Réalisé par :

Angela Saade
Aurélien Daudin
Armand Blin
Baptiste Arnold
Gabriel Monteillard

`Lien vers le dépôt GitHub`

SCIA
2025

Sommaire

1 Introduction

L'analyse automatique des avis clients, ou *business reviews*, est devenue un enjeu majeur pour les entreprises souhaitant mieux comprendre la perception de leurs produits et services.

Dans le cadre de ce projet, nous nous sommes intéressés à deux problématiques complémentaires : la classification et la génération de textes à partir d'un corpus d'avis clients. La première vise à catégoriser automatiquement les reviews selon différentes classes (par exemple, sentiment positif/négatif, catégories de services, etc.), tandis que la seconde consiste à produire de nouveaux avis synthétiques, réalistes et informatifs.

Pour répondre à ces objectifs, nous avons exploré différentes méthodes de représentation des données textuelles. Nous avons ensuite évalué plusieurs modèles de classification et de génération.

Ce rapport présente l'ensemble de notre démarche, depuis la préparation et l'analyse du dataset jusqu'à l'expérimentation et la comparaison des différents modèles. Nous mettons en avant les choix méthodologiques effectués, les résultats obtenus, ainsi que les perspectives d'amélioration pour des applications futures en NLP appliqué aux business reviews.

2 Dataset

2.1 Présentation du dataset

Nous avons fait le choix d'utiliser le YELP dataset. Il fournit une vaste collection de métadonnées d'entreprises, d'avis d'utilisateurs, de données de visites et de photos, permettant aux chercheurs d'explorer diverses tâches en science des données, telles que l'analyse de sentiments, les systèmes de recommandation et l'analyse géospatiale.

Pour ce travail, nous avons voulu nous concentrer uniquement sur les avis de restaurants. Nous avons donc construit un sous-ensemble de données d'avis de restaurants qui est : équilibré entre les différentes catégories de notes par étoiles, stratifié selon le volume d'avis par établissement, et limité à un nombre fixe d'entreprises avec un nombre d'avis approximativement égal.

2.2 Pré-traitement des données

2.2.1 Normalisation

Notre implémentation, présente dans la classe `Tokenizer`, applique plusieurs techniques de normalisation : conversion en minuscules, suppression des caractères spéciaux et des chiffres et lemmatisation. Ces étapes de normalisation permettent d'obtenir un texte plus homogène et réduisent considérablement la complexité du modèle en diminuant la taille du vocabulaire tout en préservant l'information sémantique essentielle.

2.2.2 Tokenisation

1. **Segmentation en tokens** : Nous utilisons la fonction `word_tokenize` de NLTK pour diviser le texte en mots individuels. Cette fonction prend en compte les car-

actéristiques linguistiques spécifiques à l'anglais pour effectuer une segmentation précise.

2. **Filtrage des mots vides** : Les mots vides (stop words) comme "the", "a", "is", qui apparaissent fréquemment mais apportent peu d'information sémantique, sont éliminés à l'aide de la liste prédéfinie de NLTK. Cette opération permet de réduire le bruit et de concentrer l'analyse sur les mots porteurs de sens.

3. **Filtrage par longueur** : Les tokens de longueur inférieure ou égale à 2 caractères sont supprimés, car ils correspondent généralement à des abréviations ou des fragments de mots peu informatifs.

2.3 Analyse exploratoire

2.3.1 Statistiques

Nous avons pu réaliser une analyse exploratoire du dataset global.

Metric	Full Dataset	Restaurant Only
Review	6,990,280	2,615,449
Business	150,346	27,837
Users	1987997	973,227
Time	2005 à 2022	2005 à 2022

Table 1: Analyse globale du dataset

Grade	Total	%	Mean char	Mean Tokens
1 *	304,959	11.66	641.40	122.72
2 *	241,805	9.25	699.00	132.88
3 *	334,952	12.81	659.47	125.19
4 *	672,309	25.71	581.04	109.03
5 *	1,061,424	40.58	447.17	83.00

Table 2: Distribution et longueur moyenne

Token Stat	Moyenne	Médiane	Std dev	Min	Max	Total	Tokens uniques
Valeur	104.35	75.00	96.06	0	1,053	104,353,377	194,532

Table 3: Statistiques des tokens

2.3.2 Elements clés

- **Biai positif** : 66.29% des avis sont positifs.
- **Relation longueur-sentiment** : Les avis négatifs sont 50% plus longs.
- **Vocabulaire** : 195 000 mots uniques mais pas seulement en anglais.
- **Complexité des avis** : 104 tokens, 555 caractères par avis.
- **Plage de longueur** : de 1 à 5 000 caractères.
- **Mot dominant propre au dataset** : "food" est dans le top 20.
- **Période** : 17 ans de données (2005–2022).

3 Modèles de classification

3.1 Naive Bayes

Nous avons entraîné un classifieur *Multinomial Naive Bayes* en utilisant une représentation TF-IDF des critiques de commerces. Le modèle a été évalué pour la classification binaire du sentiment (positif vs négatif) puis pour la prédiction fine des notes (1 à 5 étoiles).

Prédiction du sentiment Sur un jeu restreint, la classification binaire atteint une précision de 88.7%, avec un rappel élevé sur la classe positive (0.96). Les mots les plus discriminants pour chaque sentiment sont bien identifiés, par exemple "great", "food", "good" pour le positif et "better", "didnt", "order" pour le négatif (Figure 1).

Prédiction des étoiles Sur un corpus d'un million de critiques, la prédiction directe des notes est plus complexe : précision globale de 61.5%. Les termes associés aux notes extrêmes sont polarisés ("ordered", "never" pour 1 étoile, "great", "delicious" pour 5 étoiles), alors que le lexique des notes intermédiaires est plus neutre, ce qui explique la confusion du modèle sur ces classes (Figure 2).

Analyse des erreurs La matrice de confusion montre que le modèle détecte bien les avis positifs (1397 vrais positifs pour 57 faux négatifs), mais confond davantage les avis négatifs (184 faux positifs), ce qui illustre un déséquilibre dans la détection des classes (Figure 3).

Exemples

- *"The food was amazing and the service was excellent!"* → sentiment: positif (98.9%), étoiles: 5 (94.9%).
- *"Worst experience ever. The staff was rude and the food was cold."* → sentiment: négatif (99.6%), étoiles: 1 (98.0%).
- *"The ambiance was nice but the food was just okay."* → sentiment: négatif (80.8%), étoiles: 3 (46.3%).

Classe	Precision	Recall	F1-score
Négatif	0.90	0.73	0.81
Positif	0.88	0.96	0.92
Accuracy		**0.887**	

Table 4: Prédiction binaire du sentiment

Classe	Precision	Recall	F1-score
1*	0.68	0.82	0.74
2*	0.46	0.20	0.28
3*	0.47	0.25	0.32
4*	0.47	0.42	0.45
5*	0.68	0.88	0.77
Accuracy		**0.6151**	

Table 5: Prédiction fine de la note (1 à 5)

Ces résultats confirment que la classification du sentiment général est plus robuste que la prédiction exacte de la note, cette dernière étant plus sensible à la subjectivité et à la sémantique fine du texte.

3.2 Régression logistique

Nous avons mis en place un classifieur de type *Logistic Regression*, entraîné sur une représentation TF-IDF des critiques Yelp. Pour la tâche de classification binaire du sentiment, les avis avec une note strictement supérieure à 3.5 ont été considérés comme positifs, les autres comme négatifs.

Prédiction du sentiment Le modèle atteint une précision globale de 88.4% sur le jeu de test. Il présente un bon rappel sur la classe positive (0.96) et une précision élevée sur les deux classes (0.90 pour les négatifs, 0.88 pour les positifs), ce qui se traduit par un bon équilibre des scores f_1 (0.80 pour les négatifs, 0.92 pour les positifs). Ces résultats montrent que le modèle détecte efficacement les avis positifs, bien qu'il tende à confondre certains avis négatifs avec des positifs, comme le suggère un rappel plus faible pour la classe négative (0.72).

Exemples de prédiction :

- "The food was absolutely amazing and the service was great!" → Positif (99.4%)
- "Wow! Yummy, different, delicious." → Positif (98.5%)
- "The service was good but the food was cold." → Négatif (89.7%)

Classe	Precision	Recall	F1-score
Négatif	0.90	0.72	0.80
Positif	0.88	0.96	0.92
Accuracy		**0.884**	

Table 6: Prédiction du classifieur "Logistic Regression"

3.3 Réseau de neurones feedforward

Nous avons implémenté un classifieur basé sur un réseau *feedforward* pour prédire les notes Yelp (1 à 5 étoiles). Le dataset, déséquilibré, contenait une majorité de notes 4 et 5 (60%).

Prétraitement : Tokenisation avec un vocabulaire limité à 10 000 mots. Séquences normalisées à 200 tokens.

Architecture du modèle :

- Couche `Embedding` (dim. 100)
- Moyenne des embeddings (pooling)
- Deux couches `Dense` (256 neurones, ReLU) + `Dropout` (0.5)
- Sortie : couche `Dense` à 5 neurones (1 à 5 étoiles)

Résultats : Accuracy de 60.1%. Bonnes performances sur les classes extrêmes (1 et 5), mais confusions fréquentes entre notes intermédiaires (2 à 4), dont les frontières sémantiques sont plus floues.

Classe	Précision	Rappel	F1-score
1	0.71	0.80	0.75
2	0.40	0.32	0.36
3	0.44	0.39	0.41
4	0.47	0.47	0.47
5	0.72	0.75	0.74
Accuracy		**0.601**	

Table 7: Résultats du modèle feedforward

3.4 Réseau de neurones récurrent

Ce modèle utilise un réseau de neurones récurrent pour prédire les notes Yelp (1 à 5 étoiles). Le jeu de données, déséquilibré (60% de 4 ou 5 étoiles), comprenait environ 10 000 critiques.

Prétraitement : Tokenisation avec vocabulaire limité à 5 000 mots, séquences de 200 tokens. Entraînement sur 10 epochs avec GPU (Kaggle, P100).

Architecture du modèle :

- `Embedding` (dim. 128), `SpatialDropout1D` (0.2)
- Deux couches `Bidirectional LSTM` (64 puis 32 unités)
- `Dense` (32 neurones, ReLU) + `Dropout` (0.3)
- Sortie : `Dense` (5 neurones, softmax)

Résultats : Accuracy de 56.8%. Bonne performance sur la classe majoritaire (5), mais confusions sur les notes intermédiaires. Les limites floues entre 2, 3 et 4 étoiles nuisent à la précision.

Classe	Précision	Rappel	F1-score
1	0.50	0.39	0.44
2	0.35	0.28	0.31
3	0.42	0.38	0.40
4	0.46	0.41	0.43
5	0.69	0.82	0.75
Accuracy		**0.568**	

Table 8: Résultats du modèle récurrent (BiLSTM)

3.5 Transformer

- 2 tâches :
  - Sentiment 3 classes : 0 = négatif (1 ou 2 étoiles), 1 = neutre (3 étoiles), 2 = positif (4 or 5 étoiles).

  - Prédiction de la note (de 1 à 5)

- Modèle : `distilbert-base-uncased`, max_len=128.
- Entraînement : 10 epochs, batch size 16 sur GPU P100, warmup=500, weight_decay=0.01.

Méthode: Pour chacun des résultats présentés, plusieurs weight decay ont été testés et la meilleure des performances du modèle a été retenue sur toutes les époques d'entraînement. Nous avons aussi testé l'augmentation de données et son impact sur les performances (approche bonus) en prenant une proportion de plus en plus grande de notre dataset de restaurants. Pour chaque essai les classes ont été égalisées et le texte nettoyé.

	Train loss	Val loss	Accuracy	Precision	Recall	F1
Sentiments 10k reviews	0.4077	0.4966	0.7930	0.7829	0.7930	0.7871
Note (1-5 étoiles) 10k reviews	0.7152	1.1036	0.5410	0.5307	0.5342	0.5410
Note (1–5 étoiles) 50k reviews	0.7759	0.8750	0.6257	0.6238	0.6251	0.6257
Note (1–5 étoiles) 100k reviews	0.6322	0.6822	0.6898	0.6902	0.6897	0.6886

Table 9: Analyse des performances sur les deux tâches selon différentes tailles du dataset

Exemples de prédiction avec le meilleur modèle:

- "The service was terrible and the food was cold." → 1 star
- "The restaurant was okay, nothing extraordinary." → 3 stars
- "An incredible experience! The best food I've ever tasted." → 5 stars

Analyse:

Classification sentiment vs multi-classes : La prédiction de sentiment surpasse largement la prédiction de notes (79,30% vs 54,10–68,98% d'accuracy)

Impact du volume de données :

- Progression de l'accuracy : 54,10% (10k) → 62,57% (50k) → 68,98% (100k)
- Écart train/validation diminue avec plus de données (0,39 pour 10k → 0,05 pour 100k)
- Meilleure généralisation à grand volume (réduction du surapprentissage)

Prédiction de notes plus complexe :

- Granularité accrue : 5 classes à prédire au lieu de 2, multipliant les possibilités d'erreur
- Frontières conceptuelles floues : distinction entre 3 et 4 étoiles plus subjective et nuancée
- Les attentes varient selon l'établissement, le prix ou la localisation

Conclusion : Le modèle sur 100k offre le meilleur équilibre performance/généralisation, mais la classification par notes reste plus complexe que celle par sentiment.

4 Modèles de génération

4.1 Réseau de neurones feedforward

Nous avons exploré l'utilisation d'un réseau de neurones de type feedforward pour la génération de texte à partir de critiques Yelp. Le modèle a été entraîné sur un sous-ensemble de l'ensemble Yelp, contenant plusieurs milliers de critiques. Les données ont été prétraitées avec une taille maximale de vocabulaire de 5 000 mots. Chaque critique a été tronquée ou complétée par padding à une longueur maximale de 100 tokens.

Le modèle utilise une architecture composée de couches `Embedding`, deux couches `LSTM` avec régularisation par `Dropout`, et une couche finale `Dense` avec une activation `softmax` pour la prédiction du mot suivant. L'entraînement a été réalisé sur 900 époques avec une carte GPU P100 via Kaggle.

Résultats obtenus : Les scores suivants ont été obtenus après l'entraînement, illustrant les performances du modèle en termes de similarité de surface (ROUGE) et de n-grammes (BLEU), ainsi que sa perplexité moyenne.

Métrique	Score
ROUGE-1	0.2386
ROUGE-2	0.0490
ROUGE-L	0.1470

Table 10: Scores ROUGE moyens

Métrique	Score
BLEU-1	0.1270
BLEU-2	0.0599
BLEU-3	0.0388
BLEU-4	0.0253

Table 11: Scores BLEU moyens

Perplexité moyenne du modèle : 36,44

Voici un extrait d'exemple de génération automatique :

> *The restaurant atmosphere was very generous and the atmosphere is good will not knock back the menu just is too hot and each all the staff is slightly hard to order a big teriyaki dish and i think perhaps the main person order...*

Malgré une structure grammaticale fragile, on observe que le modèle est capable de produire des phrases longues, syntaxiquement proches de critiques réelles, avec une certaine cohérence thématique autour de la restauration. Ces résultats restent néanmoins limités par la simplicité de l'architecture et l'absence de mécanismes plus sophistiqués comme l'attention.

4.2 Réseau de neurones récurrent

Nous avons implémenté un modèle de génération de texte basé sur un réseau de neurones récurrent (RNN) avec cellules LSTM pour prédire le mot suivant dans une séquence. Le modèle a été entraîné sur le corpus de critiques de films NLTK. Un vocabulaire de 19 474 mots a été constitué, et les séquences ont été uniformisées à une longueur de 50 tokens.

Architecture du modèle Le modèle comprend :
- Une couche d'*Embedding* (dimension 128)
- Une couche LSTM unidirectionnelle (256 unités)
- Une couche de *Dropout* (0.2)
- Une couche *Dense* avec activation *softmax*

Total des paramètres : 7 891 730

Entraînement et performances L'entraînement sur 10 époques a montré :
- Accuracy d'entraînement : progression de 6.7% à 30.0%
- Accuracy de validation : plafonnement à 14.5%
- Signes de surapprentissage à partir de l'époque 5

Évaluation Les métriques de génération ont produit :

Métrique	Score
BLEU	0.0145
ROUGE-1 F1	0.1160
ROUGE-L F1	0.0917

Table 12: Résultats de génération

Ces scores, bien que numériquement modestes, sont typiques pour la génération de texte non contrainte.

Génération de texte Le paramètre de température permet de contrôler la créativité du modèle :
- Température haute (1.0) : plus diversifié mais moins cohérent
- Température moyenne (0.7) : bon équilibre
- Température basse (0.2) : plus prévisible et conservateur

Limitations et améliorations Limites : surapprentissage, contexte de 50 mots, absence de mécanisme d'attention.
Améliorations possibles : régularisation accrue (dropout 0.3–0.4), embeddings pré-entraînés, modèles bidirectionnels ou Transformer.

4.3 Transformer

Approche principale : Nous avons fine-tuné un modèle de langage causal, tel que `GPT-2`, pour une tâche de génération. Puisque nous disposons de données *labelisées*, nous avons simplement pris une review complète et l'avons coupée. *Exemple* : "Le restaurant est bon" peut être une target, tandis que "Le restaurant" en est l'input. Pour l'entraînement, un paramètre de contexte indique combien de fois une review est découpée. La découpe mot par mot étant irréaliste (trop de données), nous choisissons une granularité plus large.

Modèle : `GPT-2`, avec une tokenization `BPE`, une longueur maximale de 128 tokens, et une fonction de perte par `Cross Entropy`. **Paramètres d'entraînement** : 3 époques, batch size 4, `weight_decay` 0.01, `learning_rate` 5e-5.

Métrique	Score
BLEU	0.1191
ROUGE-L	0.2322

Table 13: 20000 restaurants, 100 reviews/restaurant, len_max 200, contexte 30

Métrique	Score
BLEU	0.1804
ROUGE-L	0.3090

Table 14: 50 restaurants, 50 reviews/restaurant, len_max 150, contexte 100

Exemples de génération : *Input* : "The review is 4 stars for restaurant Cochon"
Output (Table 1) : "The Alligator, the catfish, the grits, Delicious!!! I can't wait to explore the rest of the deliciousness in the Cochon area!"
Output (Table 2) : "Best. Sandwich shop. Ever. Muffaletta was amazing. Homemade

chips. Cash only. Closed on Sundays. That's all you need to do in New Orleans. Can't go wrong with a good sandwich in this place."

Analyse : En utilisant davantage de données propres à un restaurant et en fournissant plus de contexte (cf. Table 2), le modèle apprend plus efficacement à générer des reviews cohérentes. Les modèles comme `GPT-2` comportent un grand nombre de paramètres, ce qui rend leur entraînement coûteux. Cela explique les performances limitées de notre modèle, car il a été entraîné sur un volume restreint de données à cause des limitations GPU. Il genere bien une review mais elle n'est pas assez specifique au restaurent.

Approche expérimentale : Nous avons tenté de fine-tuner un modèle `text2text`, comme `Pegasus`, pour générer un résumé des reviews d'un établissement. En l'absence de données annotées, nous avons envisagé de générer des pseudo-labels à l'aide d'un modèle de summarization tel que `Pegasus`, mais cela était techniquement inenvisageable. Nous avons donc utilisé les reviews les plus courtes du dataset comme pseudo-labels. L'input correspondait aux reviews longues, la target aux courtes, en les faisant correspondre par note (e.g. reviews 5 étoiles entre elles, puis 4, etc.).

Résultats du modèle :

Métrique	Score
BLEU	0.00
ROUGE-L	0.0678

Analyse : **Pegasus** est un modèle avec beaucoup de paramètres donc avec peu de données plus de la pseudo données, il n'est pas étonnant d'avoir ce score.

5 Interface

Nous avons choisi de créer une interface (approche bonus) pour utiliser nos modèles. La première page (4) permet d'afficher les informations d'un restaurant, de tester individuellement et de combiner nos modèles dessus (en générant un avis et en le classifiant, ce qui est une seconde approche bonus). Elle permet aussi sur une seconde page (5) de générer des sous-datasets différents selon des paramètres d'équilibrage et de taille (si on veut par exemple augmenter la donnée, comme ce qui a été utilisé dans la partie transformer pour la classification, ce qui est une troisième approche bonus).

6 Conclusion

Le projet a exploré deux volets complémentaires du NLP appliqué aux avis clients : la classification (sentiment binaire et prédiction fine des étoiles) et la génération de textes synthétiques. Différentes architectures, des méthodes statistiques classiques aux modèles profonds et pré-entraînés, ont été comparées sur un sous-ensemble équilibré du dataset Yelp consacré aux restaurants. Les résultats montrent que les modèles traditionnels restent compétitifs pour la classification binaire, tandis que les Transformers offrent un gain significatif dès que la granularité ou le volume de données augmente. Pour la génération, les modèles à contexte étendu (GPT-2) surpassent nettement les architectures plus simples, mais restent limités par la capacité de calcul et la taille du corpus.

Modèle	Sentiment binaire (acc.)	Note 1–5 étoiles (acc.)
Naive Bayes (TF-IDF)	0.887	0.615
Régression logistique (TF-IDF)	0.884	–
Réseau feedforward (Embedding + pooling)	–	0.601
Réseau récurrent (Bi-LSTM)	–	0.568
Transformer (DistilBERT)	0.793 (3 classes)	0.690 (100 k avis)

Table 15: Benchmark des modèles de classification

Modèle	Perplexité	BLEU-4	ROUGE-L
Feedforward + LSTM	36.44	0.0253	0.1470
RNN-LSTM	–	0.0145	0.0917
GPT-2 fine-tuning (50 restaurants, ctx=100)	–	0.1804	0.3090
Pegasus (pseudo-labels de résumés)	–	0.0000	0.0678

Table 16: Benchmark des modèles de génération

Analyse et synthèse

- **Pipeline réutilisable :** Les étapes de pré-traitement (normalisation, tokenisation) et les représentations textuelles (TF-IDF, embeddings, BPE) ont été partagées entre classification et génération, illustrant l'intérêt de solutions modulaires.
- **Volume de données et fine-tuning :** Les Transformers se distinguent dès que l'on dispose de dizaines de milliers d'exemples, améliorant à la fois la précision des classes multiples et la cohérence des textes générés.
- **Coût/performance :** Les modèles statistiques offrent le meilleur rapport coût/performance pour la classification binaire, alors que les architectures profondes et pré-entraînées sont indispensables pour des tâches plus complexes ou plus créatives.

En conclusion, ce benchmark met en évidence la complémentarité des approches. Le choix du modèle doit être guidé par la nature de la tâche, la granularité requise et les ressources disponibles.

7 Annexes

Table des annexes

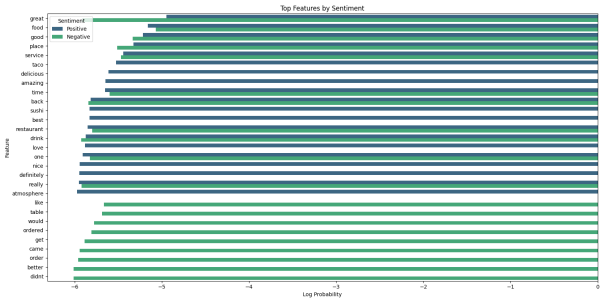

Figure 1: Termes les plus influents pour chaque sentiment.

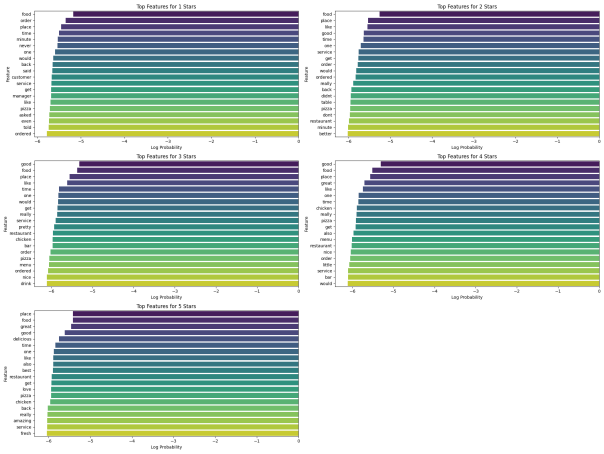

Figure 2: Termes discriminants pour chaque note de 1 à 5 étoiles.

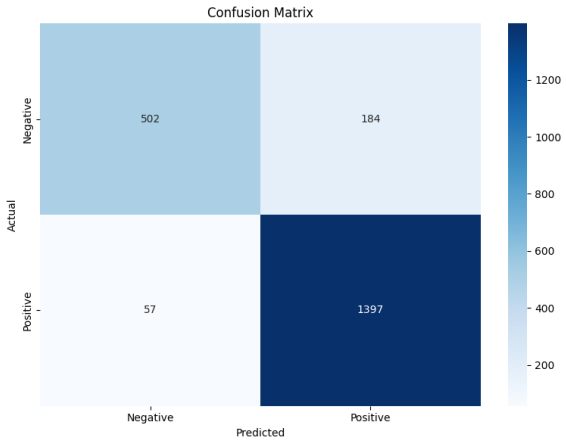

Figure 3: Matrice de Confusion pour le modèle Naive Bayes

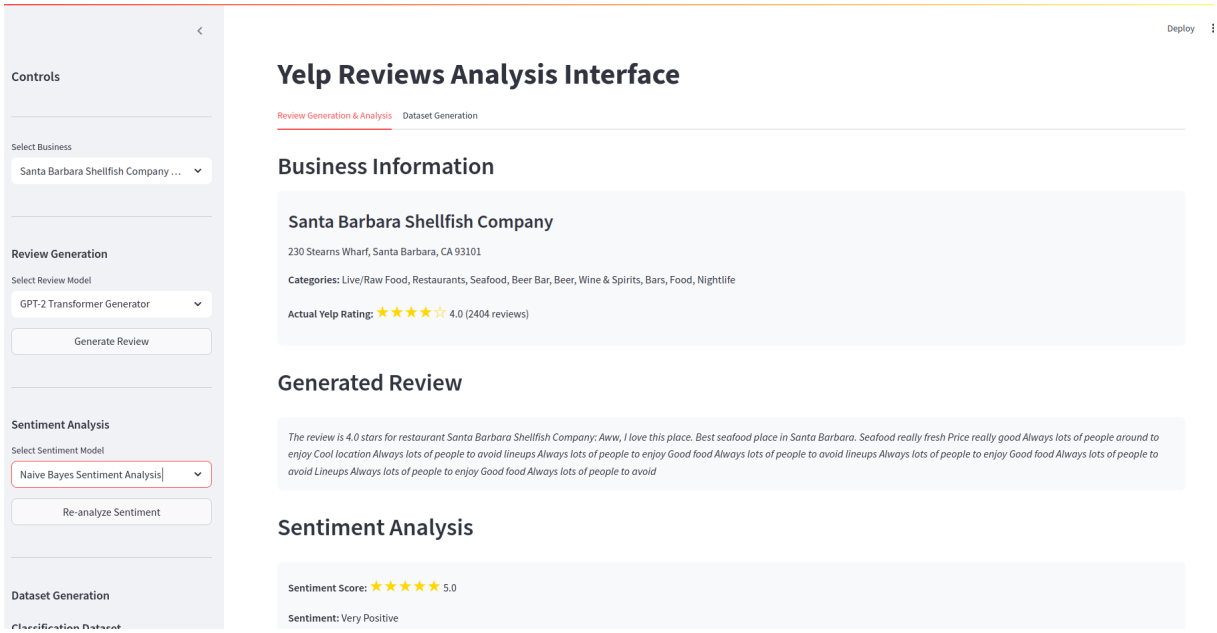

Figure 4: Page de test des modèles

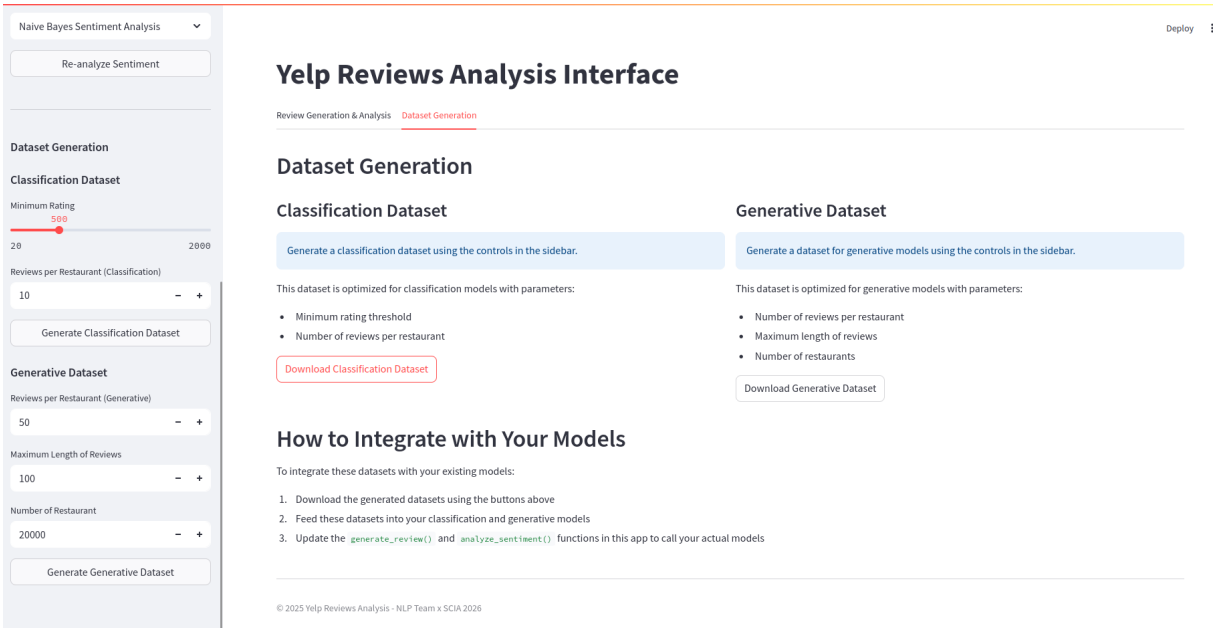

Figure 5: Page pour la génération de custom datasets