Résumé du Projet 1 : Analyse des Sentiments sur Avis Clients

1.Objectif principal:

Le projet vise à analyser les sentiments exprimés dans les avis clients en utilisant des modèles pré-entraînés, avec des étapes clés allant de la préparation des données à la visualisation des résultats. ## Étapes du Projet :

1. Préparation des Données :
   - Chargement d'un fichier JSONL contenant des avis clients.
   - Extraction des 200 premières données pour simplifier l'analyse. - Normalisation et tokenisation des textes.
2. Génération des Embeddings :
   - Utilisation du modèle `all-MiniLM-L6-v2` via SentenceTransformers pour encoder les textes en vecteurs.
   - Alternative : TF-IDF pour les cas où les dépendances de SentenceTransformers posaient problème.
3. Clustering (Optionnel) :
   - Proposition d'algorithmes comme KMeans ou DBSCAN pour regrouper les avis, mais non pleinement implémenté.
4. Analyse des Sentiments :
   - Chargement et expérimentation avec des modèles pré-entraînés (`nlptown/bert-basemultilingual-uncased-sentiment`, `cardiffnlp/twitter-roberta-base-sentiment`).
   - Association des prédictions à des labels numériques (1-5 étoiles).
   - Calcul de la corrélation de Pearson entre les notes réelles et les prédictions.
5. Visualisation :
   - Utilisation de Matplotlib pour comparer les distributions des notes réelles et prédites.

Problèmes Rencontrés et Solutions :

- Fichier JSONL vs JSON :
  - Problème : Erreur de lecture initiale.
  - Solution : Utilisation du module `json` pour lire ligne par ligne et charger les données.
- Dépendances Python (Numpy, Pytorch) :
  - Problème : Erreurs liées à l'indisponibilité de bibliothèques (e.g., Numpy non installé correctement).
  - Solution : Réinstallation ciblée via `pip` et ajustements pour réduire la dépendance à certaines bibliothèques.
- Modèle Alternatif :
  - Problème : Besoin d'évaluer un autre modèle pour confirmer la robustesse.
  - Solution : Implémentation du modèle `cardiffnlp/twitter-roberta-base-sentiment` et comparaison des prédictions.
- Manque de Mémoire (MacBook sans Xcode) :
  - Problème : Limitation matérielle pour installer certaines dépendances (Xcode).

- Solution : Remplacement par des alternatives légères (TF-IDF au lieu d'embeddings, simplifications des modèles).

Conclusions :

Le projet explore eAicacement plusieurs méthodes pour analyser et visualiser les sentiments des avis clients. Malgré des contraintes techniques, les ajustements apportés (modèles alternatifs, solutions légères) ont permis de mener à bien l'analyse.

Résumé du Projet 2

1. Objectif principal:

Développer une chaîne de récupération et génération de réponse (RAG) en utilisant un modèle de langage Ollama, avec une base de données vectorielle existante pour répondre aux requêtes des utilisateurs.

2. Problèmes rencontrés:

Problèmes avec la solution OpenAI :

- Quota dépassé : Erreur RateLimitError liée à un quota d'API insuAisant, empêchant l'utilisation d'OpenAI.
- TypeError : Erreurs de type (expected string or buAer) lors de l'appel de l'API pour générer des embeddings, probablement à cause de mauvais formats de données.

- Erreur d'intégration du modèle LLM (Ollama):

- Problème : ValidationError et TypeError lors de l'intégration du modèle.
- Solution : Correction des arguments et utilisation des versions d'API appropriées. -

Problème de base vectorielle:

- Problème : Erreurs liées à la récupération de documents dans la base vectorielle.
- Solution : Vérification de la structure des documents et du format des embeddings.

- Absence de fichier .txt:

- Problème : Pas de fichier texte pour alimenter la base de données.

- Solution : Remplissage direct de la base vectorielle avec des documents pertinents.

- Erreur de embedding function:

- Problème : Absence de fonction d'embedding dans Chroma.
- Solution : Vérification et génération des embeddings associés aux documents.

3. Méthodes utilisées:

- Ollama pour le LLM :

Utilisation du modèle stablelm2.

- Chroma pour la base vectorielle :

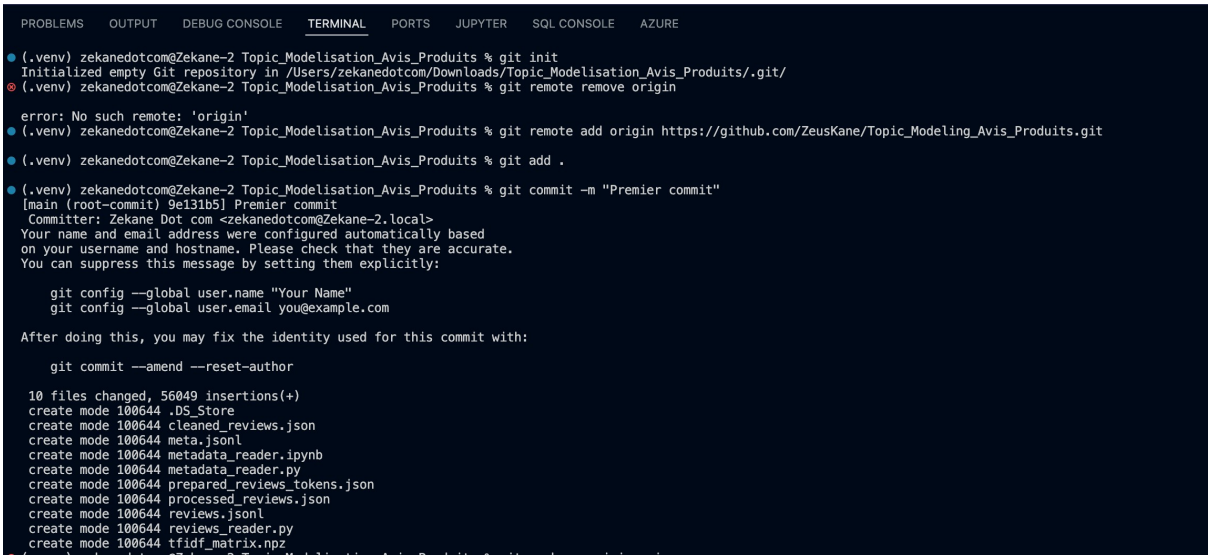
Gestion des embeddings et récupération des documents pertinents.

- Langchain pour l'intégration :

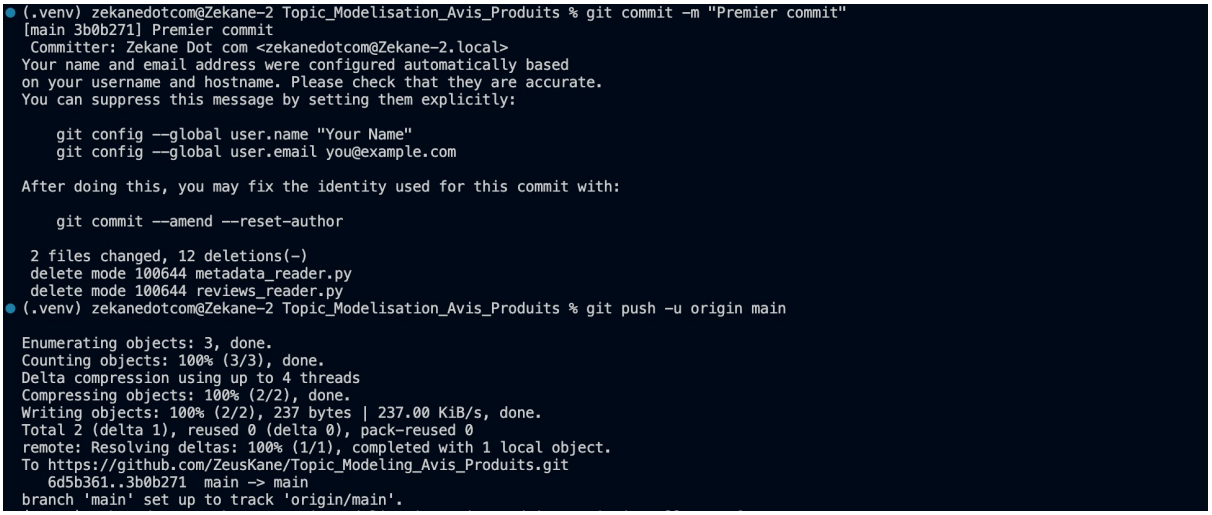
Création de la chaîne RAG et gestion des requêtes utilisateur.

3.Etape Github :

Sur cette partie je mets les push à partir de mon local vers github.



Après cela on aperçoit ici la dernière commit.



Ci-dessous le lien vers mon repo GitHub :

https://github.com/ZeusKane/projets_sds_nlp-genai

Mail : serignekane01@gmail.com