Projet Recofilm

Système de recommandation de films

Datascientest - Promo FEV 24 MLOps
Aurore Gailhard, Charles de Valois, Edouard Gobinet
Soutenance du 02/07/2024

Sommaire

1 - Contexte

Catégorie	Détails
Problématique	Offre personnalisée de contenu pour améliorer l'engagement et la satisfaction des utilisateurs sur une plateforme de streaming.
Commanditaire	Entreprise opérant dans le secteur du streaming vidéo, telle que Netflix ou un service similaire cherchant à optimiser ou à redéfinir son système de recommandation.
Utilisateurs/Administrateurs	Equipe Data/Engineering de la plateforme de streaming: alimentation et optimisation de l'interface utilisateur, gestion du cycle de vie des modèles de ML, de la maintenance et de la disponibilité de l'application.
Support	Accès via une API qui permettra de récupérer les recommandations de films et de renvoyer des données sur les interactions des utilisateurs.

1 - Objectifs

Catégorie	Détails
Amélioration du moteur de recommandation	- Collaborative filtering - Content based filtering - Modèle hybride combinant les deux
Architecture Robuste et Évolutive	- PostgreSQL pour la gestion des données - FastAPI pour l'interfaçage API - Docker pour la conteneurisation et le déploiement
Testing et monitoring	- Pytest pour l'intégration des tests unitaires - Airflow pour l'orchestration - MLFlow/Grafana pour le monitoring

2 - Modèles

Content Based

- Données:
  → tags
  → genres

- Matrice TF-IDF:

Quantifier l'importance des genres et des tags en fonction de leur fréquence

- Similarité cosinus:

Mesurer la similitude entre le film recherché et les autres films

Collaborative Filtering (SVD)

- Données:
  → userId
  → ratings
  → movieId

- Evaluation du modèle:

Validation croisée avec les mesures de RMSE et MAE pour quantifier l'efficacité du modèle.

- Entraînement et enregistrement du modèle:

Entraînement sur l'ensemble des données

Hybrid

Combinaison des 2 modèles:

- Normalisation des scores
- Facteur de pondération pour équilibrer les deux méthodes
- Enregistrement dans MLflow et dans une table Recommendations

2 - API FastAPI

Authentification et sécurité:

- Utilisation de JSON Web Token → Généré lors de la connexion avec un user_id valide
- Durée de vie de 10 minutes
- Nécessaire pour valider chaque requête ultérieure → sera passé dans l'en-tête

Fonctionnalitées:

- Modèle chargé au démarrage de l'api → GET"/"
- Authentification avec un user_id → POST "/login"
- Recommandations Collaborative Filtering → GET"/recommendations"
- Recommandations Hybrides → POST"/hybrid"
- Cold-Start → POST "/genre_recommendations"

Surveillance et analyse:

Enregistrement des requêtes et du status-code dans un fichier log

2 - BDD

Création d'une base Postgres à partir des fichiers CSV

src > data > db > initialize_database.pgsql

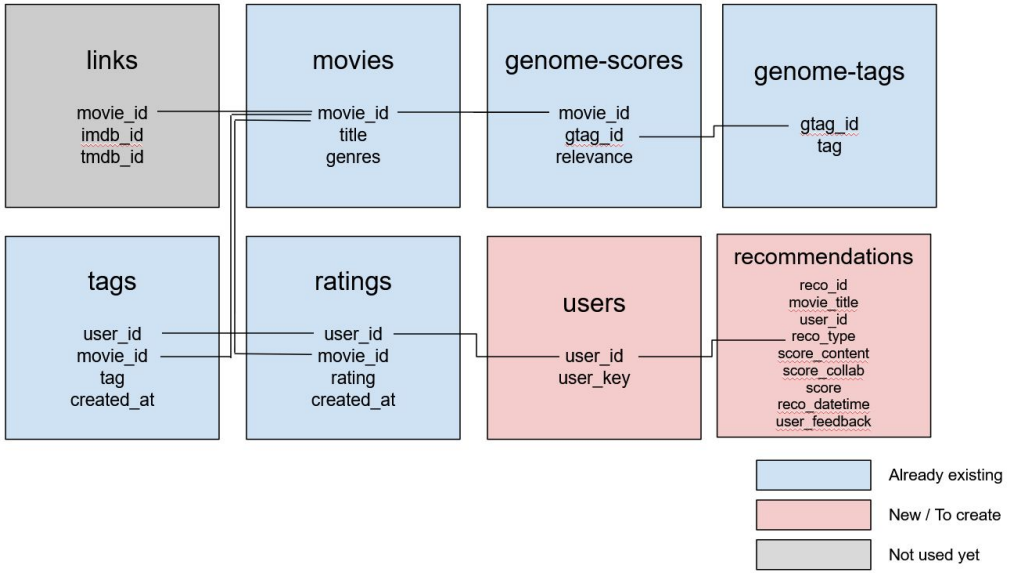
```sql
CREATE TABLE movies (
    movie_id    INTEGER PRIMARY KEY,
    title       VARCHAR(255),
    genres      TEXT
);

CREATE TABLE users (
    user_id     INTEGER PRIMARY KEY,
    user_key    TEXT
);

CREATE TABLE ratings (
    user_id     INTEGER,
    movie_id    INTEGER,
    rating      DECIMAL(2, 1) CHECK (rating >= 0.0 AND rating <= 5.0),
    created_at  TIMESTAMP,
    FOREIGN KEY (movie_id) REFERENCES movies(movie_id),
    FOREIGN KEY (user_id) REFERENCES users(user_id)
);

CREATE TABLE links (
    movie_id    INTEGER,
    imdb_id     INTEGER,
    tmdb_id     INTEGER,
    FOREIGN KEY (movie_id) REFERENCES movies(movie_id)
);

CREATE TABLE tags (
    user_id     INTEGER,
    movie_id    INTEGER,
    tag         TEXT,
    created_at  TIMESTAMP,
    FOREIGN KEY (movie_id) REFERENCES movies(movie_id),
    FOREIGN KEY (user_id) REFERENCES users(user_id)
```

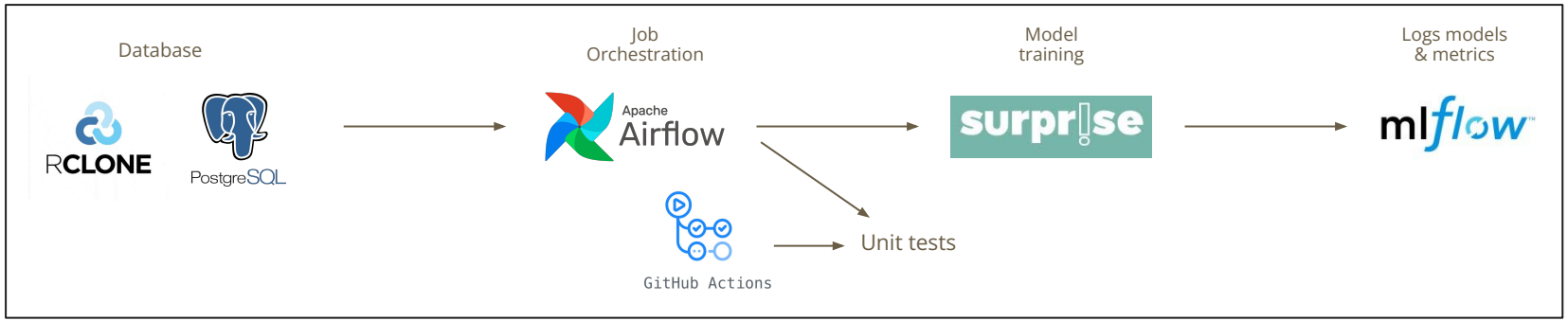


2 - Stack & Conteneurisation

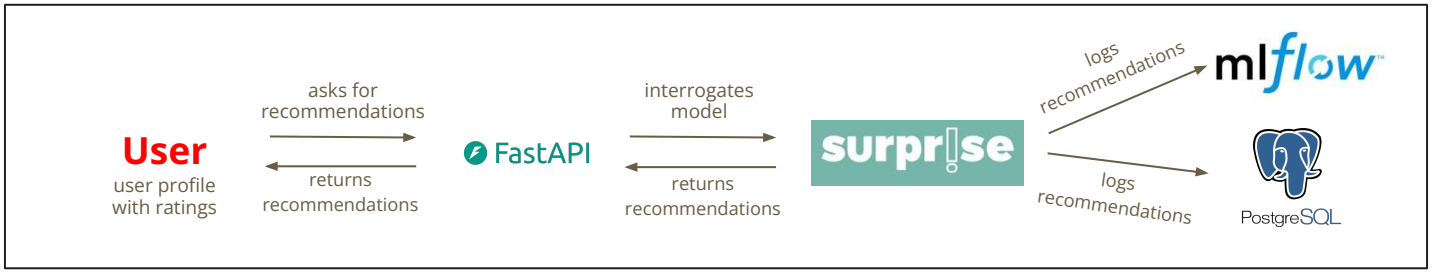
Services containerized with docker

Services monitored with Prometheus Grafana

Model training



User recommendation



3 - Démo

4 - Conclusion

Conclusion:

- Faisabilité et efficacité de la mise en place d'un système de recommandation de films
- Amélioration de la pertinence et de la personnalisation des recommandations de films pour les utilisateurs avec un modèle hybride
- Architecture fiable et adaptable assurant la scalabilité et la maintenance de l'application

Ouverture:

- Application Streamlit
- Gestion d'un modèle plus gros avec les 20M de ratings/Réentrainement partiel
- Insertion de nouveaux ratings et de nouveau users dans la database
- Déclenchement partiel de l'entraînement du modèle en fonction du feedback user ou des actions users
- Utilisation de Kubernetes et déploiement sur le cloud
- Monitoring de l'activité API et ajout d'alertes dans Grafana
- Monitoring du décrochage de la performance du modèle dans MLFlow pour déclencher des actions correctives
- Amélioration de la sécurité en utilisant Github Secrets
- Gestion du cold start: demande du genre préféré