Final Project Presentation

Movies Entertainment

By: Dylan Lam & Eric Nguyen

Course: C S 329E

Time: Friday 2:00-5:00PM

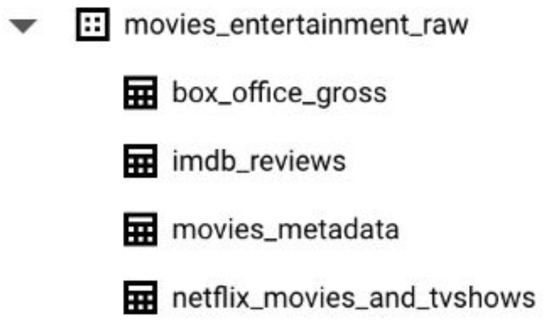
Table of Contents

Overview

- We built a complete end-to-end data pipeline focused on the movies and entertainment industry, leveraging public datasets from Kaggle.
- Our objective was to integrate, clean, and analyze diverse data related to Netflix shows, IMDb reviews, genre metadata, and box office performance.
- We designed our data architecture using Google Cloud Storage for ingestion and BigQuery, while leveraging dbt to structure our pipeline into source, staging, intermediate, and mart layers.
- We implemented ELT best practices, enforced data contracts and constraints, and ensured data quality with automated testing and documentation.
- Our work culminated in two advanced applications: generating a complete lineage and documentation environment on a VM, and implementing fuzzy matching using Gemini embeddings in BigQuery.

Datasets

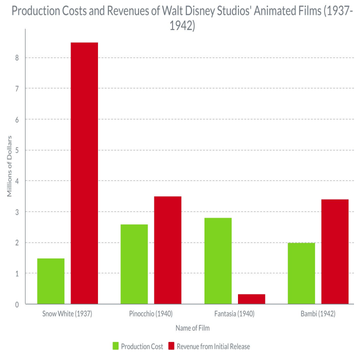
1. **movies_metadata (Kaggle)**
2. **box_office_gross (Kaggle)**
3. **netflix_movies_and_tvshows (Kaggle)**
4. **imdb_reviews (Stanford AI)**



Content

Our project focuses on building a data warehouse with multiple data marts to analyze trends in the movie industry. The datasets we are working with include:

1. **Movies Metadata** – General information on movies, including **title, budget, language, and genre**
2. **Box Office Gross Data** – Revenue figures for theatrical releases
3. **Netflix Movies & TV Shows Data** – Streaming content metadata, including **audience ratings (PG, PG-13, R, etc.)**
4. **Company Information** – Data on **production companies, total releases, and financial performance**



Scope

In-scope (What We Analyze):

- **Fuzzy Entity Resolution via Embeddings**→ used Gemini-generated embeddings to cluster similar or duplicate titles
- **Title Deduplication & Standardization** → used Vector distances to determine entity similarity and prepare for downstream analytics
- **Semantic Similarity Matching -** explored use of ML.GENERATE_EMBEDDING and vector similarity search to match records beyond exact string matches

Out-of-Scope (What We Do Not Analyze):

- **Model Fine-tuning or Training Custom Embeddings** → relied on pre-trained Gemini embedding models and do not train or fine-tune any embedding models ourselves
- **Downstream Actions (Post-Cluster Joins, Aggregations)** → we focused on identifying similarities, not executing downstream transformations using matched clusters
- **Real-time Similarity Matching** → project is batch-based only. We do not explore real-time matching using BigQuery streaming or Vertex AI endpoints.

Challenges

1. Vertex AI + BigQuery Integration
2. Embedding Syntax & Structure
3. Vector Distance Computation at Scale
4. Pipeline Integration

Key Learnings

1. Vector embeddings unlock fuzzy matching at a semantic level–not just syntax.
2. BigQuery ML can now powerfully generate and compare embeddings without needing external APIs.
3. Performance tuning is critical.
4. Debugging across layers requires full-stack visibility and patience.

DEMO

Thank you.