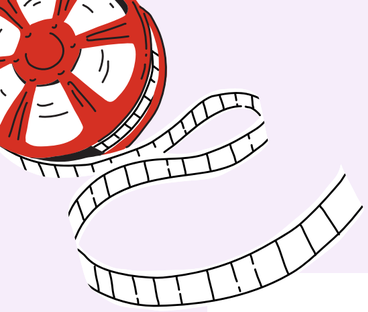
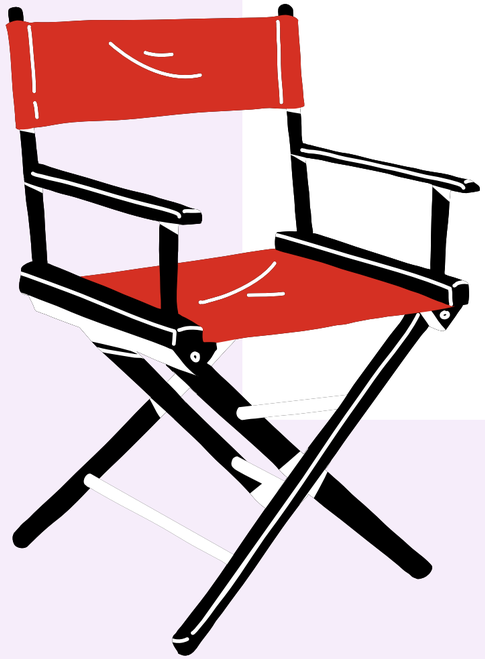



# Movie Recommendation System

**Team Members: Ola Mahjob, Sidrah Almazlom, Layan Almasoud, Shaden Almasaud, Roha Alswoaiegh**

# Outline

1. Introduction
2. Methodology & Data
3. Analysis & Findings
4. Conclusion & Recommendations
5. Lessons Learned



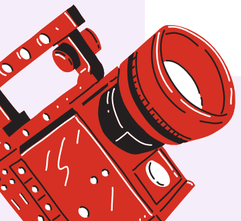
# Introduction & Context

## Problem Statement

Users are often overwhelmed by the staggering amount of content on streaming services and they have difficulties finding content that meets their interests. Individualised preferences cannot be captured by filtering by year, genre or popularity; therefore, traditional search methods do not provide a good experience for users which leads to poor engagement with the site(s).

## Project Objective

Develop a Movie Recommendation System that uses an understanding of movie attributes, user scores, popularity ratings, and textual descriptions to produce accurate and appropriate recommendations according to the individual characteristics of the users. The recommendations should be based on user preferences and unique user profiles to provide personalized movie recommendations, even though users may not have rated a large number of movies.



# Business Impact & Value Proposition

### Increase User Engagement

Personalized recommendations keep users watching longer and exploring more content.

### Higher Conversion Rates

Recommendation engines typically boost content consumption by 20–30%, as seen in major platforms like Netflix.

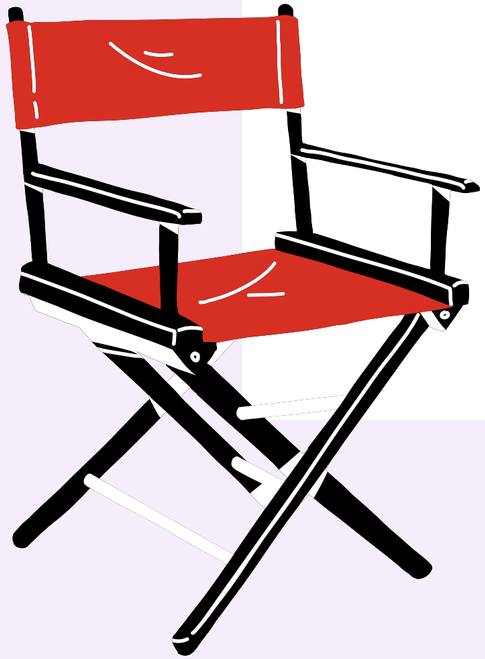
### Reduce User Churn

Relevant recommendations make users less likely to abandon the platform.

### Data-Driven Content Insights

Helps businesses understand trending genres, audience preferences, and optimal content strategies.

# Methodology & Data



# Data Source

### Source Platform

- The datasets were downloaded from Kaggle

### Dataset Origin

- The movie metadata comes from The Movie Database (TMDb) API, compiled and published on Kaggle.
- The user ratings dataset comes from a crowdsourced rating collection, also hosted on Kaggle for academic and practical ML use.

### 🔗 Access

- Provided through an open Kaggle repository (Dataset Link)

# Key Features

- Contains rich movie metadata (genres, budget, revenue, runtime, companies, countries, title, overview...)
- Includes millions of user–movie rating interactions
- Designed to support recommendation system benchmarking

# Process Steps

## Data Cleansing

### Duplicates and Null Values

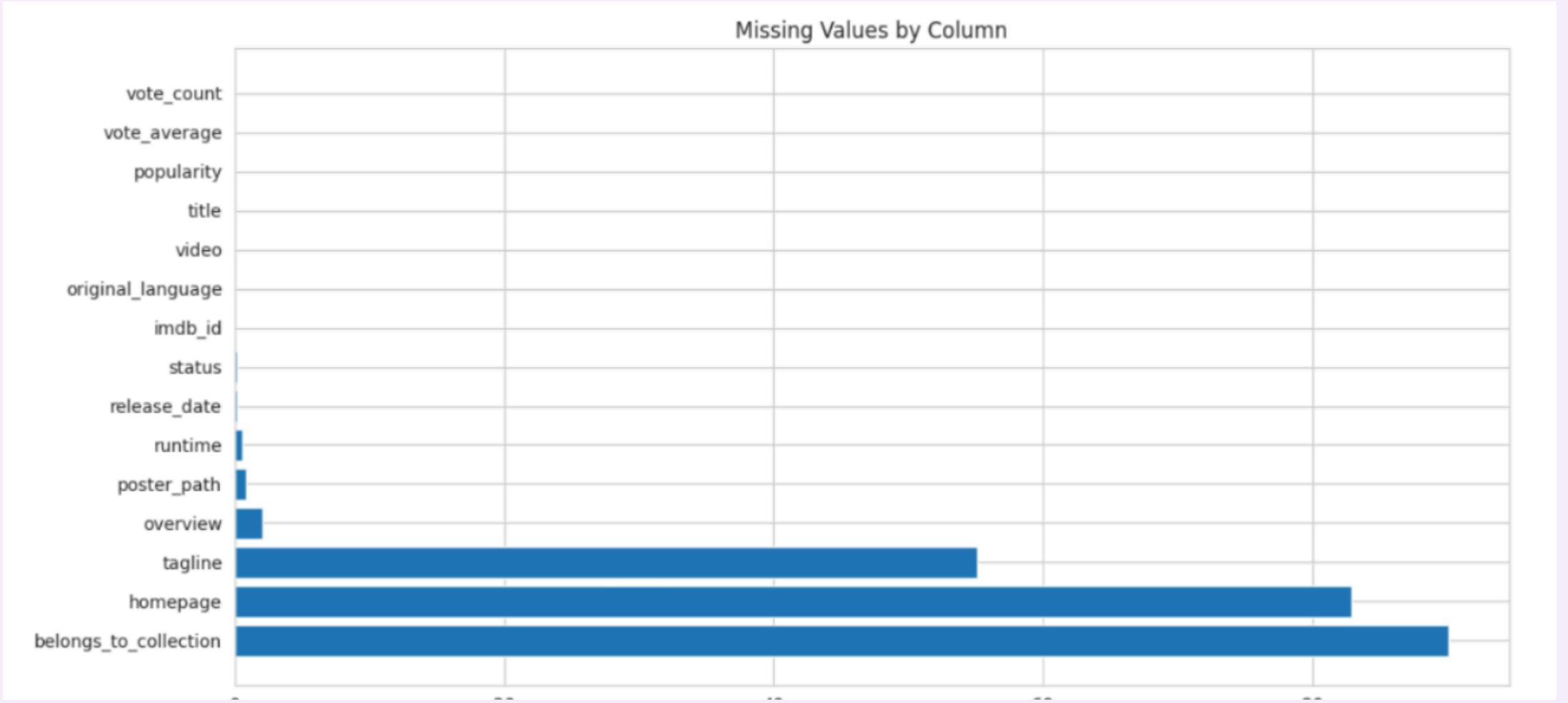
- Removed rows with null values and duplicates without much affecting the size of the dataset.
- For some columns we replaced the null values.
- We dropped columns that were unrelated to the task which were the homepage link and target.

### Incorrect Formatting

- Corrected incorrect data types
- Most columns loaded as object → converted to correct numeric or date formats
- Cleaned columns with dictionary (key–value) structures
- Extracted usable fields (genres, companies...)

### Reduce Dataset Size

- Filtered active users (rated 100–2000 movies) due to dataset size


Missing Values by Column
vote_count
vote_average
popularity
title
video
original_language
imdb_id
status
release_date
runtime
poster_path
overview
tagline
homepage
belongs_to_collection

# Process Steps

## EDA

### Numerical Analysis

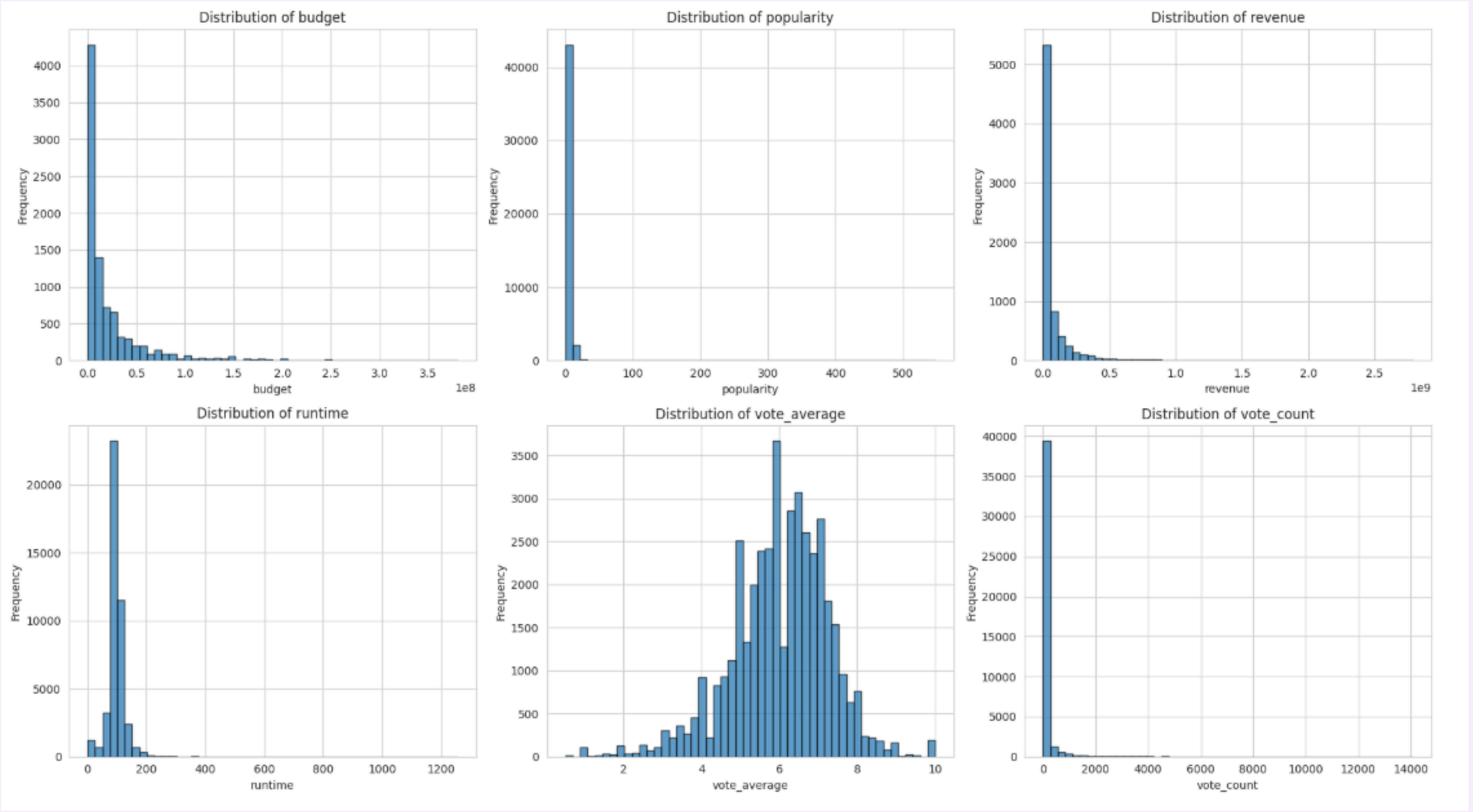
- Calculated mean, median, mode, range
- Detected outliers in budget, revenue, and runtime

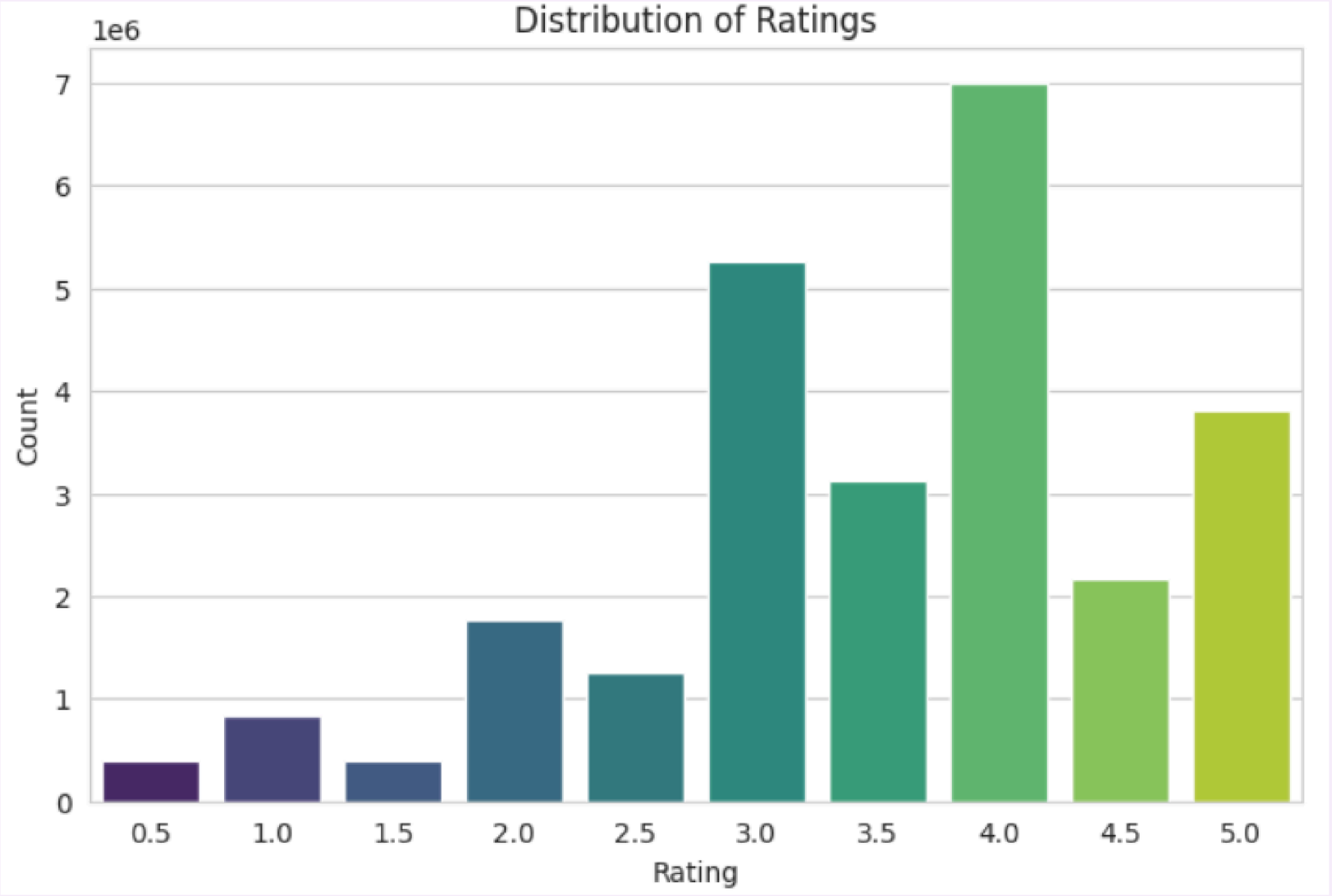
### Categorical Analysis

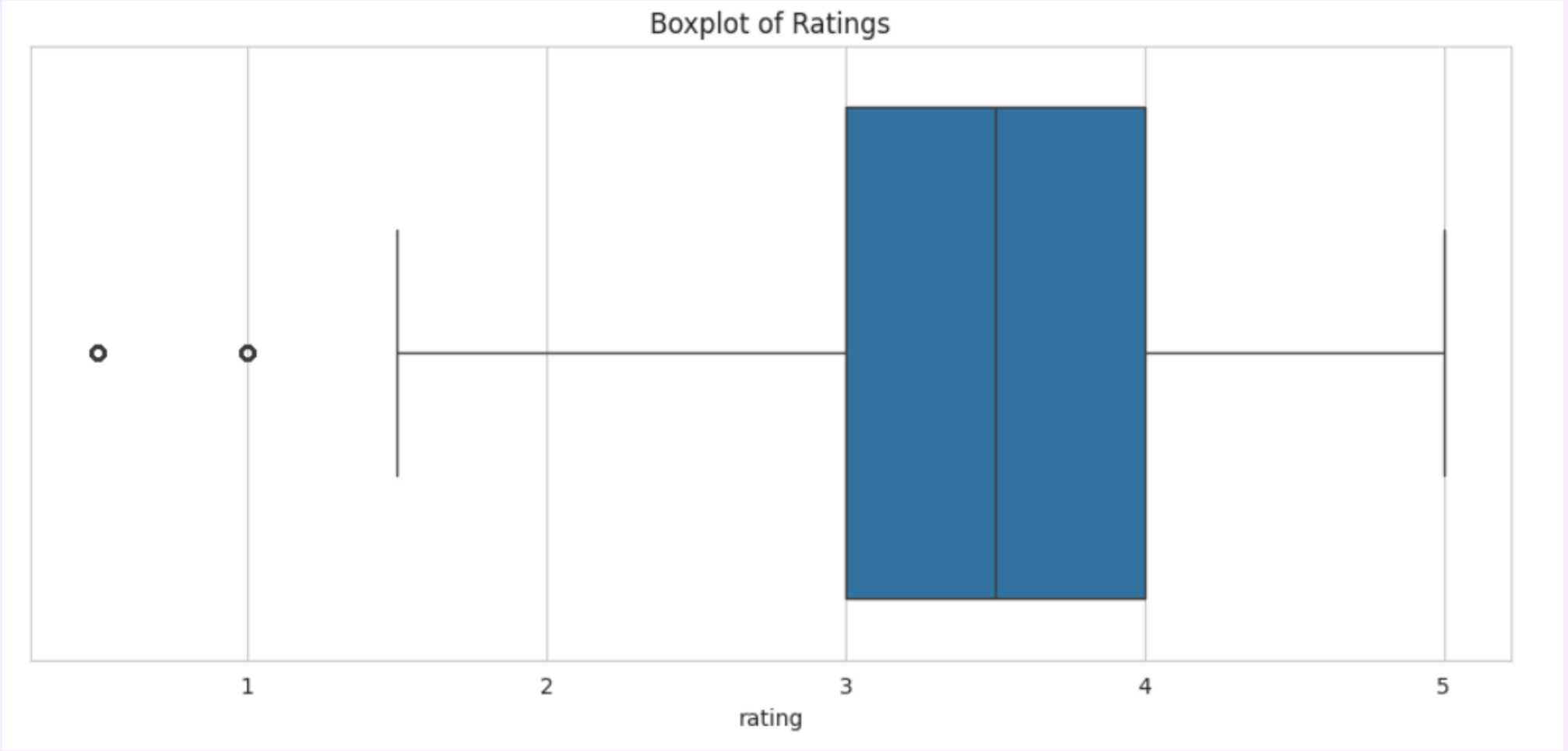
- Removed adult column (almost all values = False → no impact on prediction)

### Feature Engineering

- Created new rating-based features, including the Weighted Rating
  - Used IMDb-style formula combining:
    - R: movie average rating
    - v: number of votes
    - C: overall mean rating
    - m: minimum votes threshold
  - Helps reduce bias toward movies with very few votes and produces a fairer, more reliable rating metric
- Normalized numeric fields (budget, revenue, runtime, popularity, weighted_rating, release_year) using StandardScaler
- One-hot encoding for low-cardinality categories (original_language, major genres, production_countries)
- Frequency encoding for high-cardinality categories (production_companies, belongs_to_collection)
- TF-IDF text vectorization for title and overview to capture semantic meaning


Distribution of budget
Frequency
budget
1e8
Distribution of popularity
Frequency
popularity
Distribution of revenue
Frequency
revenue
1e9
Distribution of runtime
Frequency
runtime
Distribution of vote_average
Frequency
vote_average
Distribution of vote_count
Frequency
vote_count


Distribution of Ratings
1e6
7
6
5
4
3
2
1
0
Count
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
Rating


Boxplot of Ratings
1
2
3
4
5
rating

# Supervised Learning

## Goal

Predict the rating that a user would give to a movie that they haven't watched based on ratings for movies they watched before

## Label & Features

Target label: rating
Features used: revenue,budget,runtime, weighted rating, movie title, movie overview, language, production companies and countries

## Dataset Used

User ratings dataset merged with movies metadata

## Algorithms

XgBoost, Random Forest with hyperparameter tuning using Randomized Search CV
Evaluation Metrics: RMSE, MAE

# Unsupervised Learning

## Goal

Give recommendations based on content similarity

## Label & Features

Target label: Top ten similar movies
Features used: Overview, title and genre - NLP Task

## Dataset Used

Movies Metadata

## Algorithms

MPnet Embeddings + K-Means Clustering
MiniLM Embeddings + Cosine Similarity
Merge the cluster number with the supervised learning task

# Process Steps

## Generative AI

- In this phase, we integrated our content-based movie recommendation system with generative AI powered by the Gemini API. This integration enables the system to provide insightful interpretations of its recommendations, delivering friendly, engaging explanations that enhance user understanding and experience.

- We developed two prompts and compared between them based on:

1. Clarity & structure
2. Purpose & Style
3. Tone & Voice
4. Constraints

# Prompt Result Example

## 🟩 Prompt 2 Output:

**How These Movies Connect**

At first glance, *Frozen* and these films might seem like an odd mix! However, a common thread emerges through the theme of isolation in icy landscapes and how characters confront dangerous, often deadly, situations. We see a blend of survival stories, horror elements, and interpersonal dramas set against unforgiving snowy backdrops. The stakes are high, and the characters must overcome both external threats and internal conflicts.

**Shared Themes & Story Elements**

Many of these films explore the breakdown of relationships under extreme pressure, mirroring the sisters' journey in *Frozen*. Survival becomes a central theme, whether against the elements in *Avalanche* and *Force Majeure*, or against more sinister forces in films like *Hostel* and *Dead Snow*.

**Character & Tone Similarities**

While *Frozen* is lighter in tone, the theme of family relationships, particularly sisterhood, resonates with the human connections in *Force Majeure* and *By the Law*. Even in the horror films, the characters must rely on each other to survive, though often with tragic consequences. Many feature characters facing moral dilemmas in life-or-death situations.

**What This Says About Your Taste**

You seem drawn to stories where characters are tested to their limits, both physically and emotionally, in challenging environments. You appreciate narratives that blend elements of suspense, drama, and even horror, with a focus on how people react when pushed to extremes.

**Friendly Insight**

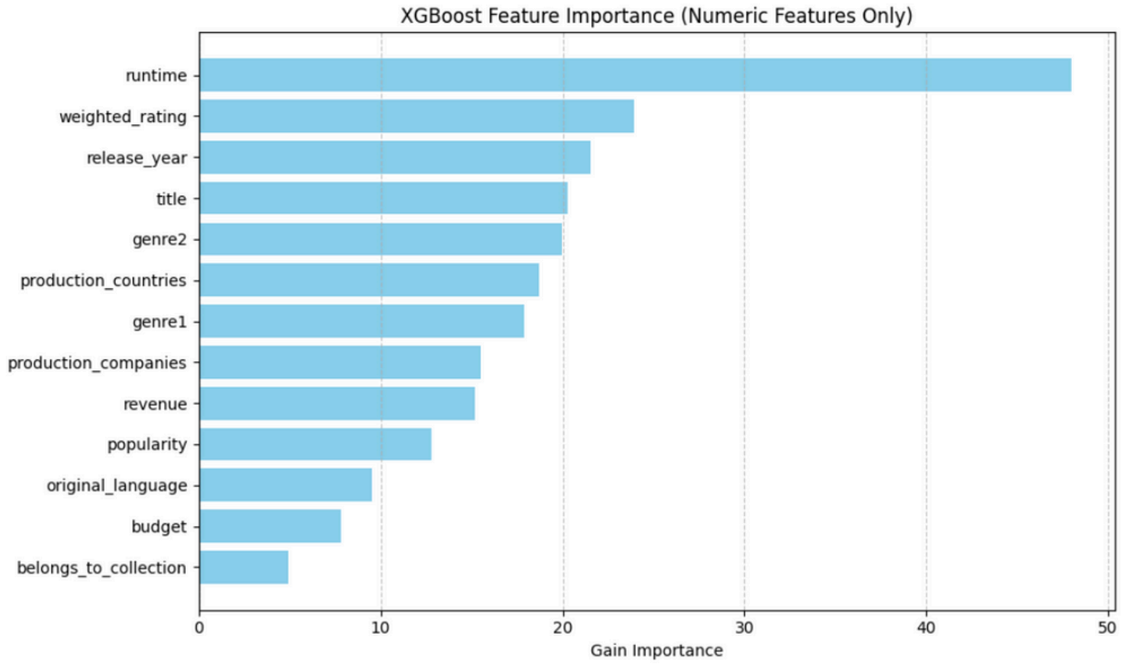
It's fascinating to see how a seemingly innocent animated film like *Frozen* can connect with darker, more intense narratives! This shows a sophisticated appreciation for storytelling that explores the depths of human resilience and the complexities of relationships.

# Tools Used

- Python
- Pandas, NumPy, Scikit-learn
- Matplotlib, Seaborn
- Google Colab

# Analysis & Findings

# What Influences User Ratings?



# What Influences User Ratings?

## Key Points :

- Runtime is the strongest predictor of ratings It has the highest gain importance (~49).Meaning: longer movies tend to receive higher ratings in your dataset.

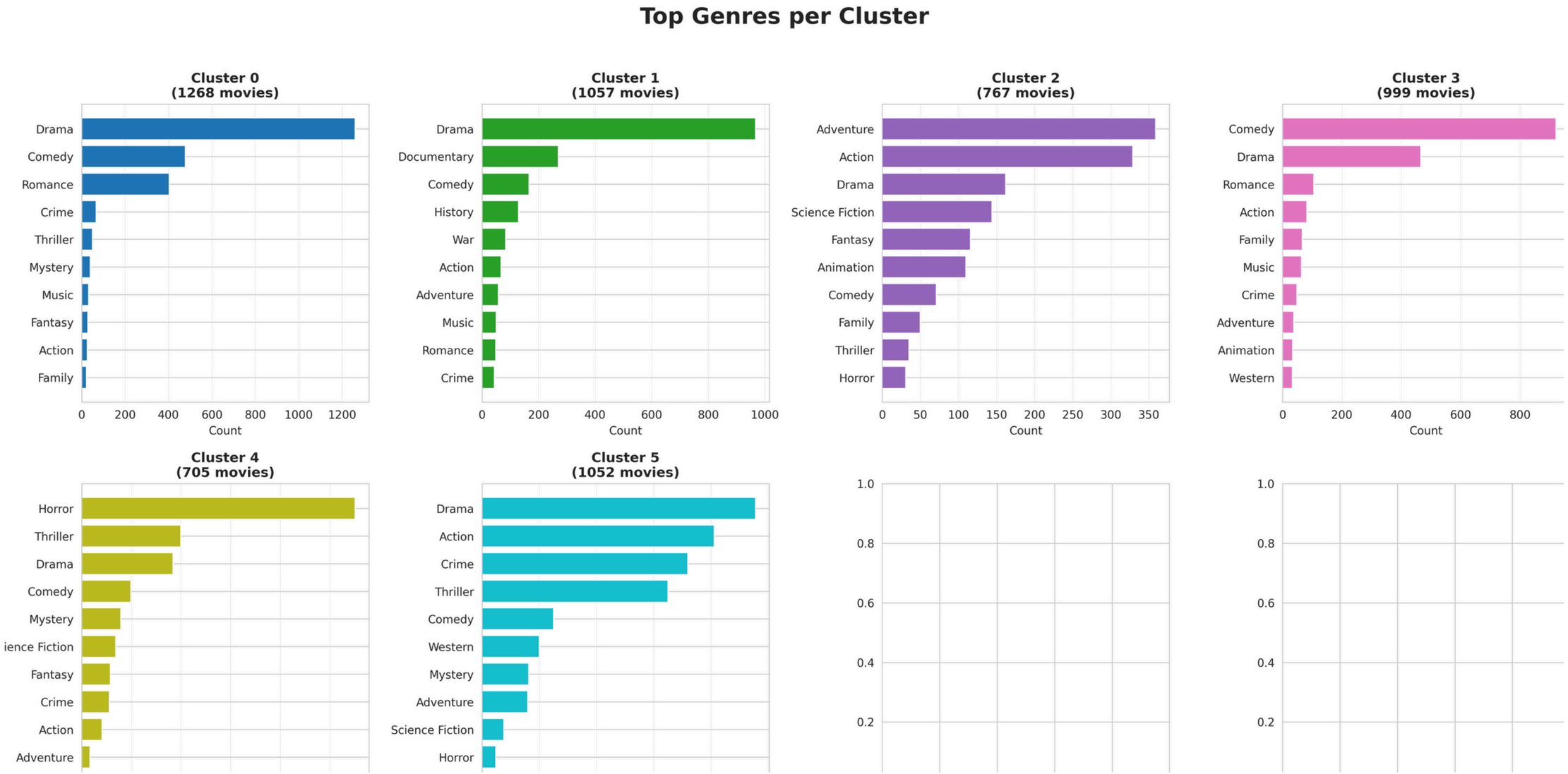
- Weighted Rating (IMDb-style rating) is also highly influential The model heavily relies on how other users scored the movie. This shows a strong "popularity bias".

- Release Year, Title, and Genres contribute moderately They help the model, but not nearly as much as runtime or weighted rating.

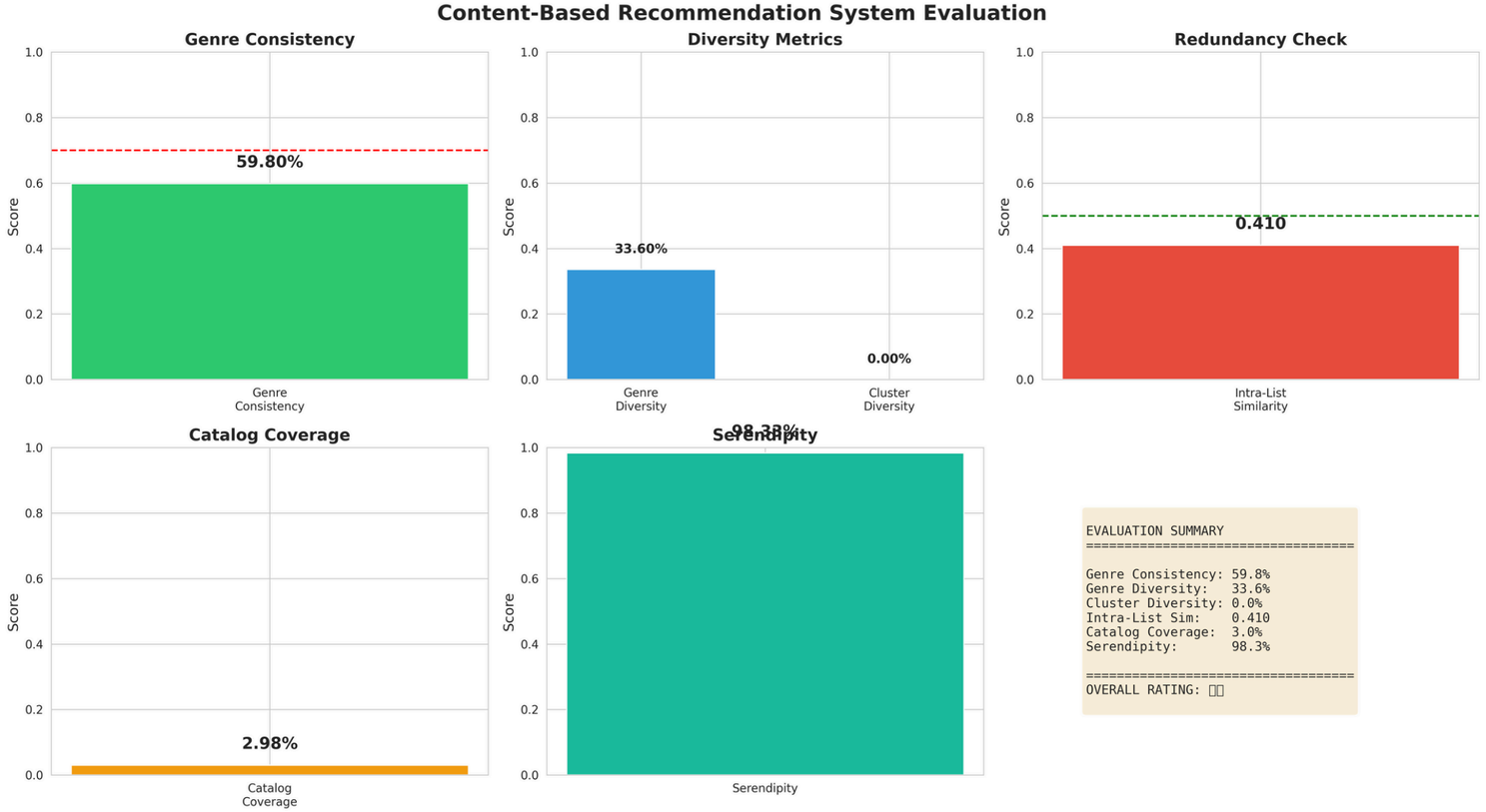
- Country, Production Companies, Revenue, and Popularity have smaller impact These metadata features add information, but they are not major drivers.

- Belongs_to_collection and Budget have least influence Meaning collections or budget do not explain rating differences well.

# Findings From Content-Based Evaluation



# Findings From Content-Based Evaluation




```
EVALUATION SUMMARY
==================================

Genre Consistency: 59.8%
Genre Diversity:   33.6%
Cluster Diversity: 0.0%
Intra-List Sim:    0.410
Catalog Coverage:  3.0%
Serendipity:       98.3%

==================================
OVERALL RATING: □□
```

# Findings From Content-Based Evaluation

## Key Points :

- Genre Consistency (59.8%) → The system matches movies with similar themes effectively.
- Serendipity (98%) → Recommendations are relevant but also pleasantly surprising.
- Intra-List Similarity (0.41) → The system avoids repetitive suggestions.
- Catalog Coverage (3%) → Only a small part of the catalog is being used in recommendations.

# Combining Both Approaches: What We Learned

## Key Points :

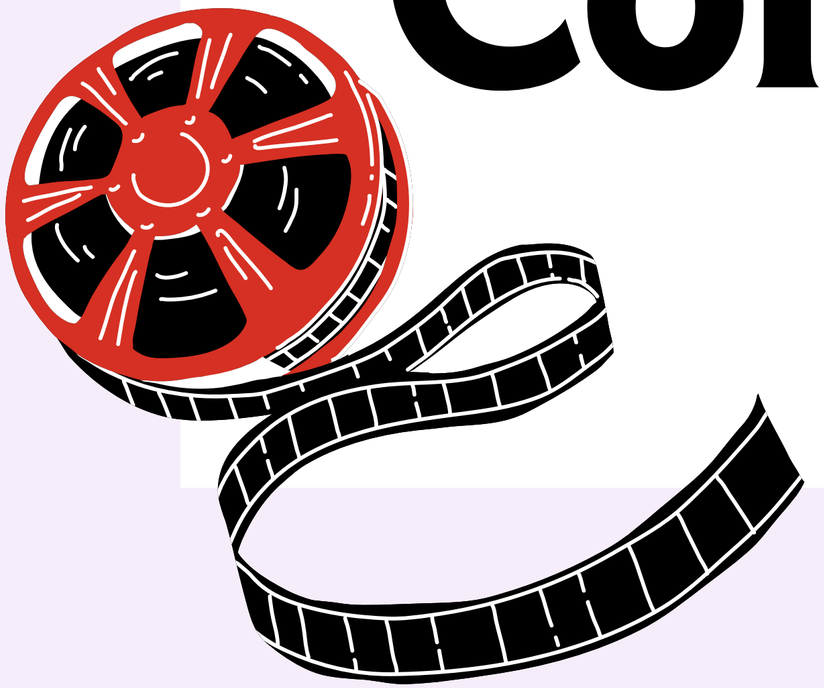
- The rating-based system is strong for detecting popularity but weak for personalization.
- The content-based system accurately reflects true movie attributes but has limited catalog reach.
- When combined, rating signals overshadow content signals, leading to minimal improvement.
- To understand real user taste, content features provide more meaningful insights than ratings.

# Strategic Impact for the Recommendation System

## Key Points :

- Relying mainly on ratings creates popularity loops and reduces content discovery.
- Content-based recommendations help viewers find movies that align with their actual taste, not popularity trends.
- The high serendipity score shows an opportunity to promote hidden gems, increasing user engagement.
- Improving catalog coverage can expand exposure to diverse titles and boost user satisfaction.

# Conclusion



# Summary

Based on the analysis results:

- User ratings are primarily influenced by runtime and weighted ratings, indicating a strong popularity bias rather than personalized preferences.
- Content-based methods provide richer and more meaningful signals of user taste, achieving high serendipity and avoiding repetitive recommendations.
- Catalog coverage is very low (3%), meaning recommendations are generated from only a small portion of the dataset, limiting discovery of diverse or hidden titles.

# Actionable Recommendations:

- Shift focus toward content-based features (overview, genres, title, attributes) to improve recommendation relevance for individual users.
- Reduce dependence on rating-only models to avoid favoritism toward already popular movies.
- Increase catalog coverage by adjusting similarity thresholds or model parameters so more unique movies enter the recommendation pool.

# Future Work

- Improve catalog diversity through clustering or diversity re-ranking algorithms.
- Conduct user studies to validate recommendations with real-world preferences and refine the system accordingly.

# Lessons Learned

# What We Learned

## Working with Data

- Started with a large dataset but had to reduce it - learned the trade-off between more data and computer memory limits
- Cleaned the data by fixing missing values and removing bad entries - raw data is never ready to use
- Learned about different types of text embeddings and their pros and cons.

## Training Models

- Training means showing the model lots of examples so it learns what features predict ratings
- Tested XGBoost and Random Forest - XGBoost improved 9% over baseline and found that runtime and weighted_rating matter most
- Training and testing errors were close- meaning no overfitting, the model generalizes well to unseen movies

# What We Learned

## Clustering Didn't Help Much

- Tried K-Means clustering to group similar movies into 6 clusters
- Got very low silhouette score (0.018) - movies don't form clear distinct groups based on content alone
- Learned to choose methods based on the problem - clustering wasn't right for rating prediction, but could work well for other tasks like organizing content

## Content-Based Recommendations

- Used cosine similarity to find movies with similar descriptions - achieved 70% genre consistency
- Worked better than clustering because it directly measures similarity between individual movies, not forcing them into fixed groups

## Integrating Generative AI

- Used Gemini API to take our system's recommendations and turn them into something functional and usable
- Learned prompt engineering - well-structured prompts with clear constraints produce consistent, reliable results
- This approach can be applied to unlimited purposes depending on what you need from the system

# References

Intellect Markets, "Global Content Recommendation Engine Market | Size, overview, trends, and forecast 2025–2030," 2025. [Online]. Available: https://intellectmarkets.com/report/content-recommendation-engine-market

# Thank you