MOVIELENS DATASET ANALYSIS REPORT - STAGE 1

Feature Engineering & Exploratory Data Analysis

Author: Adu Morenikeji Toluwalope

Date: October 2025
Dataset: MovieLens ml-latest-small
HNG Internship - Data Analytics Track

Executive Summary

This comprehensive report details the first stage of the MovieLens dataset analysis, focused on Feature Engineering and Exploratory Data Analysis (EDA). Utilizing a robust Python 3.11 environment with key libraries like Pandas and Seaborn, the project successfully transformed raw data into a highly informative dataset suitable for advanced machine learning models. The core achievement was the creation of 8 high-utility features and the discovery of 6 critical, actionable insights into user behavior and movie characteristics, providing a strong foundation for the development of a high-performance recommendation system.

Key Results: - Processed 100,836 ratings from 610 users on 9,724 movies - Created 8 meaningful features for recommendation systems - Identified significant patterns in user behavior and movie characteristics - Generated actionable insights for recommendation algorithm design

1. Dataset Overview

1.1 Data Sources

The MovieLens ml-latest-small dataset consists of four main files:

Dataset	**Records**	**Description**
ratings.csv	100,836	User ratings (userId, movieId, rating, timestamp)

movies.csv	9,742	Movie metadata (movieId, title, genres)
tags.csv	3,683	User-generated tags
links.csv	9,742	External database links

1.2 Data Quality Assessment

1. No duplicate records were found in primary datasets
2. No missing values in core rating and movie data
3. Clean data quality with consistent formatting
4. Date range: March 1996 to September 2018 (22+ years of data)

2. Feature Engineering

2.1 Methodology

Created 8 new features to enhance the dataset for recommendation systems:

2.2 Feature Descriptions

Feature	Method / Technique	Range / Average / Coverage	Key Value / Insight
1: Release Year	Extracted from movie titles using regex parsing	100,818 movies (99.98% success); Range: 1902–2018	Enables time-based recommendations and temporal analysis

2: Genre Count	Counted genres per movie (split by space delimiter)	Range: 0–10; Average: 2.7 genres	Indicates movie complexity and wide appeal
3: Movie Age at Rating	Calculated difference between release and rating year	Average: 13.3 years	Captures user preferences for new vs. classic films
4: Temporal Features	Extracted rating hour, day, and month	Peak Activity: 8:00 PM, Monday	Supports time-aware recommendation delivery
5: Popularity Score	Counted total ratings per movie	Range: 1–329; Average: 58.8 ratings	Addresses cold start and long-tail problems
6: Average Movie Rating	Computed mean rating per movie	Overall Average: 3.50 / 5.0	Serves as a quality measure for filtering
7: User Activity Level	Counted total ratings per user	Range: 20–2,698; Average: 603.9 ratings	Indicates user engagement and data reliability
8: Decade Categories	Grouped movies into decade-based categories	Pre-1950, 1950s–2010s+	Enables era-based clustering and trend analysis

3. Exploratory Data Analysis

3.1 Key Statistics

1. Total Users: 610
2. Total Movies: 9,724
3. Total Ratings: 100,836
4. Average Rating: 3.50/5.0

Rating Distribution: Positively skewed (61.2% are 3.5+ stars)

4. Six Key Insights

4.1 Insight 1: Rating Patterns

Finding: Bias of positive rating - 61.2% of ratings are 3.5 stars or higher - Most frequent rating is 4.0 stars - Implication: Bias in the ratings chosen by the users is positive, which creates the selection bias.

4.2 Insight 2: Movie Popularity Distribution

The most popular movie has 329 ratings - The average rating on a movie is 58.8 ratings - Implication: Long tail problem that necessitates specialized algorithmic content processing to serve niche content.

4.3 Insight 3: Genre Preferences

Finding: Drama prevails, multi-genre films are prevalent - Median number of genres per movie: 2.7 - Conclusion: Multi-genre films can be used to reach a wider audience, allowing them to recommend cross-genre.

4.4 Insight 4: Temporal Rating Behavior

Finding: he obvious trends regarding the rating activity: The highest rating activity: 8:00 PM (the evening leisure time) - The most active day: Monday - Implication: Temporal features can be used to optimize the time and delivery of recommendations.

4.5 Insight 5: User Engagement Patterns

Discovery: Large range in the level of activity of the users -Most active user: 2,698 ratings - average user: 603.9 ratings -Heavy users (100 or more ratings): 84,313 instances -Implication: The level of activity of users should be used to inform the recommendation effectiveness and the choice of the algorithm. 4.6 Insight 6: Age and Quality of Movies.

4.6 Insight 6: Movie Age and Quality

Finding: Movie ratings over the eras - Top rated decade: 1950s (3.85 average ranking) - Average age at which films are rated: 13.3 years old - Insight: Survivorship bias with older movies could be observed, only good films can be popular with time.

5. Recommendation System Applications

5.1 Content-Based Filtering

1. Applicable Features: Genre Count, Decade Categories, Release Year, and Average Movie Rating.
2. Strategy: CBF can leverage movie content features to find similarities between items. For a new user (cold start), the system can recommend items based on features of their first rated movie.
3. Use Cases: "Movies similar to X" recommendations, genre-based filtering, and providing era-specific recommendations (e.g., "The best of the 1990s").

5.2 Collaborative Filtering

Applicable Features: - User activity levels - Temporal rating patterns - Popularity scores

Use Cases: - User-user similarity calculations - Item-item collaborative filtering - Activity-weighted recommendations

5.3 Hybrid Systems

Combined Approach: - Balance popularity with personalization - Use temporal features for timing - Leverage user activity for confidence scoring - Apply genre diversity for exploration

5.4 Cold Start Mitigation

Strategies: - Use movie features for new users - Leverage popularity scores for new items - Apply temporal patterns for timing optimization

6. Technical Implementation

6.1 Data Processing Pipeline

Data Loading: Multi-file CSV ingestion

Quality Checks: Duplicate and missing value detection

Feature Engineering: 8 new feature creation

Data Integration: Merge operations with validation

Export: Enhanced dataset generation

6.2 Feature Engineering Code Structure

```python

Example feature creation

df['release_year'] = df['title'].apply(extract_year) df['genre_count'] = df['genres'].apply(lambda x: len(x.split('|'))) df['movie_age_at_rating'] = df['rating_year'] - df['release_year'] ```

### 6.3 Performance Metrics

Processing Time: < 5 seconds for full dataset

Memory Usage: Efficient pandas operations
```

Feature Coverage: 99.98% success rate for year extraction

7. Conclusions and Future Work

7.1 Key Achievements

1. Data Preparation: Achieved cleaned and integrated multi-file dataset
2. Feature Engineering: added 8 meaningful features for recommendation systems
3. Insight Generation: found 6 actionable insights about user behavior
4. System Readiness: Improved dataset prepared for advanced analytics

7.2 Recommendation System Readiness

The newly improved dataset now integrates innovative collaborative filtering techniques, content-based recommendation systems, hybrid approaches, and temporal as well as popularity-conscious methodologies to enhance the overall recommendation performance.

7.3 Future Enhancements

1. Advanced NLP: Process user tags for semantic features
2. External Data: Integrate IMDb/TMDb metadata via links
3. Deep Learning: Feature preparation in neural collaborative document filtering.
4. Streaming recommendation Design features.

7.4 Business Impact

1. Enhanced Recommendations:Ease of use by adding superior features.
2. Cold Start Solutions: New user/item processing capabilities.

3. Personality: The confidence of the activity-based recommendations.
4. Temporal Optimization: Time sensitive recommendation delivery.

8. Deliverables

8.1 Technical Specifications

1. Programming Language: Python 3.11
2. Key Libraries: pandas, numpy, matplotlib, seaborn
3. Dataset Size: 100,836 records with 15 features
4. Processing Environment: Windows 11, Jupyter Notebook