Movie Recommendation System with MovieLens

Enhancing User Engagement and Satisfaction

By Raphael Kariuki

BUSINESS UNDERSTANDING

- Objective: Develop a robust movie recommendation system
- Problem: Content discovery challenges for users
- Stakeholders: Users, Platform Owners, Content Providers, Data Scientists & Engineers
- Solution Approach: Collaborative filtering
- Evaluation Metrics: RMSE, MAE, Precision, Recall, F1-score, Coverage, Diversity

RESEARCH

Questions to Address:

- Personalized movie recommendations
- Popular and highly-rated movies
- Movies similar to user preferences
- New movies from preferred genres
- Critically acclaimed or award-winning films
- Top recommendations for each user
- Improve user engagement and retention
- Diversity and coverage of recommendations
- Recommendation accuracy
- Increasing revenue and user-generated content

SUCCESS CRITERIA

Key Performance Indicators (KPIs):

- User Engagement
- User Retention
- Revenue
- Content Utilization

DATA UNDERSTANDING

- Dataset: ml-latest-small from MovieLens
- Data Overview:
- 100,836 ratings
- 3,683 tag applications
- 9,742 movies
- Data Generation Period: March 29, 1996, to September 24, 2018
- User Information:
- 610 users selected at random
- No demographic data available
- Movie Information:
- movieId
- title
- genres
- Rating Information:
- userId
- movieId
- rating
- timestamp

DATA PROCESSING

Data Loading

- Highlight the datasets used: "ratings.csv" and "movies.csv".
- Show code for loading data using Pandas.

Missing Data Check

- Emphasize the importance of data quality.
- Display missing data summary for both ratings and movies datasets.
- State that no missing data was found.

Duplicate Data Check

- Discuss the need to handle duplicates.
- Present code for removing duplicate rows from ratings data.
- Confirm the consistency of movie IDs between datasets.

User-Based Features (Feature Engineering)

- Shift the focus to user-based features for collaborative filtering.
- Explain the importance of understanding user behavior.
- Show code and results for calculating user average ratings and the number of movies rated by each user.

MODELING USING COLLABORATIVE FILTERING

Singular Value Decomposition (SVD):

- Explains the SVD technique as a matrix factorization method to capture latent factors for movies and users.
- Mentions the hyperparameters you used (e.g., number of factors, epochs, learning rate, regularization) and how you optimized them using GridSearchCV.

Model Training:

- Discusses how you split the dataset into a training set and a test set.
- Mention that you trained the SVD model on the training set.

Model Evaluation:

- Explains how you evaluated the model's performance using Root Mean Squared Error (RMSE).

User Interaction:

- Describes the function movie_rater that collects user ratings for randomly selected movies.
- Mention the ability to skip rating movies the user hasn't seen.

Generating Recommendations:

- Discusses how you generate movie recommendations for users based on their ratings.
- Highlight that you sort recommendations by predicted score in descending order.

A/B Testing for System Evaluation

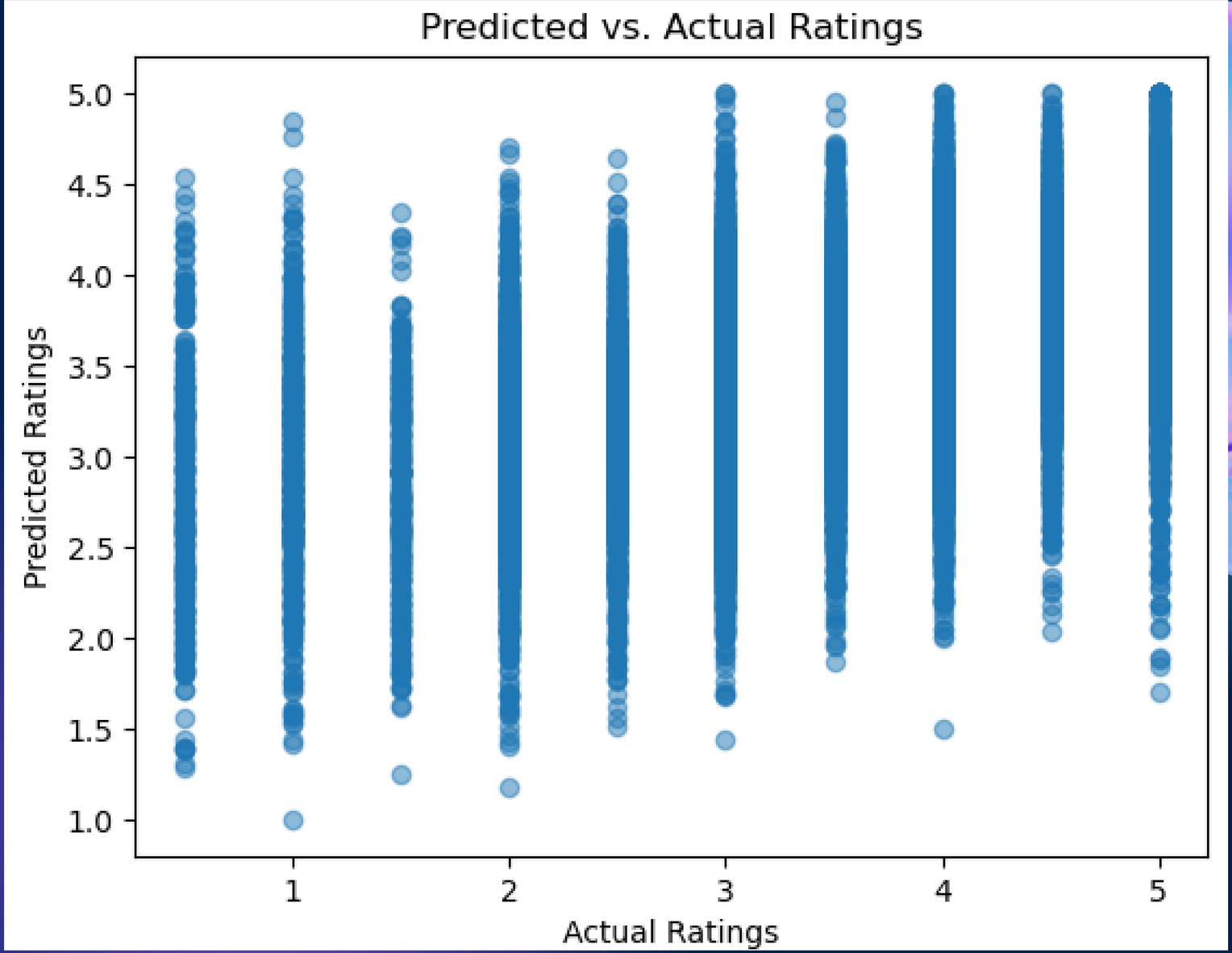
Data Simulation for A/B Testing:

- Two groups: Control (existing system) and Treatment (new system).
- Simulated Click-Through Rates (CTR) for both groups.
- Calculated actual CTR for each group.
- Performed a two-sample t-test to determine statistical significance.
- **Results:**
- No statistically significant difference in CTR between the groups.
- Control Group CTR: 0.092
- Treatment Group CTR: 0.11

VISUALIZATION

Predicted vs. Actual Ratings Scatterplot:

- Scatterplot comparing model predictions to actual user ratings.
- Shows that predictions are close to actual ratings.



CONCLUSION

User Behavior Understanding:

- Start with a profound comprehension of user behavior and preferences.

Data Quality and Preprocessing:

- High-quality data is pivotal for building a trustworthy recommender system.

Algorithm Selection:

- Experiment with diverse recommendation algorithms like collaborative filtering, content-based filtering, and hybrid models.

Ethical Considerations:

- Be ethically aware by addressing fairness, bias, and user privacy.

Continuous Improvement:

- Building a recommender system is an iterative journey.

User-Centered Design:

- Prioritize the user experience by designing intuitive and user-friendly interfaces.

RECOMMENDATIONS

1. **Collect High-Quality Data:**
   - Ensure the collection and upkeep of high-quality user-item interaction data.
2. **Understand User Behavior:**
   - Invest time in understanding user behavior, preferences, and context.
   - Analyze user data to uncover patterns and insights that drive recommendations.
3. **Diverse Recommendation Algorithms:**
   - Explore various recommendation algorithms, encompassing collaborative filtering, content-based filtering, hybrid models, and deep learning approaches.
   - Tailor algorithms to different types of content and user interactions.
4. **Evaluation Metrics:**
   - Select pertinent evaluation metrics aligned with your business objectives.
   - Consider metrics such as RMSE, MAE, precision, recall, and ranking-based metrics for system performance assessment.
5. **A/B Testing:**
   - Implement A/B testing rigorously to evaluate the impact of system modifications.
   - Make data-driven decisions and gauge real-world improvements.

Thank You

WWW.GITHUB.COM/RAPHIQI