Project Design

z5544290 Hongliang Zhang

Steam Game Recommender System Design

1. Scope

1.1 System domain

Domain: Steam game recommendation system for PC gamers
Target users: Anyone looking for PC games, including new users, casual gamers, online gamers etc.

1.2 Display strategy and UI

Recommendation Quantity: 5 games per recommendation session
Interface: Web-based dashboard sign in/sign up with Steam accounts

```
//---------------Home page----------------------------------
[Steam Logo] Welcome, [User]!            Search ▢
┌──────────────────────────────────────┐
│  RECOMMENDED FOR YOU                 │
├──────┬──────┬──────┬──────┬──────────┤
| [Img]| [Img]| [Img]| [Img]| [Img]|
| Title| Title| Title| Title| Title|
| 87%  | $9.99| Deck | 92%  | -70% |
| Like | Wish | More |...   |...   |
└──────┴──────┴──────┴──────┴──────┘
Filters: Genre ˅ Price ˅ OS ˅ Rating ˅



//------------------Below each game interface------------------
Category recommendations:Display by type, price, rating, etc
Similar game recommendations(5 games): Recommend similar games based on games the user already
owns
```

1.3 User interaction and feedback

- User feedback collection, including Game rating (1-5 stars), Like/dislike button, Add to wishlistb and Purchase
- We can also collect user behavior: Game details page browsing time, Frequency of clicking recommended games, Search keywords, Game play time, etc.

1.4 Cold start and Dynamic update

Cold start

- Recommend based on popularity and popular games (e.g. positive_ratio > 90% & user_reviews > 1000)
- Show some tags for new users to choose, and then recommend based on content similarity (type/tag)

Dynamic update

- **Model update frequency**: Retrain the model only when the user's tag changes significantly (purchase/long browsing of game types that have not been played before)
- **Real-time recommendation**: Adjust the recommendation in real time based on the user's current session behavior

1.5 Business Model

- Increase game sales via personalized discovery and get a share
- Developer paid promotion position
- Reduce refunds through better match accuracy

2. Dataset

2.1 Description

Main data set： Steam game data set (`games.csv`)

- **Data source**: Kaggle Steam game data set
  - https://www.kaggle.com/datasets/antonkozyriev/game-recommendations-on-steam?select=games.csv
- **Data size**: About 50,000 records
- **Data fields**:
  - `app_id` : Steam application ID (unique identifier)
  - `title` : Game title
  - `date_release` : Game release date
  - `win/mac/linux` : Supported operating system platforms
  - `rating` : User rating
  - `positive_ratio` : Positive review ratio
  - `user_reviews` : Total number of user reviews
  - `price_final` : Current price
  - `price_original` : Original price

Another data that may be used： `recommendations.csv`

- **Data source**: https://www.kaggle.com/datasets/antonkozyriev/game-recommendations-on-steam?select=recommendations.csv
- **Data size**: About 41 million records
- **Data fields**:
  - `app_id` : Steam application ID (unique identifier)
  - `helpful` : How many users found a recommendation helpful
  - `funny` : How many users found a recommendation funny
  - `date` : Date of publishing
  - `hours` : How may hours played by user
  - `is_recommended` : Is the user recommending the product?
  - `user_id` : User's anonymized ID
  - `review_id` : Autogenerated ID

2.2 Usage

2.2.1 Field application

1. **Key features**:

- `rating/positive_ratio`: the most important indicators for game quality evaluation and user satisfaction
- `user_reviews`: whether a game is popular

2. **Content features**:

- `title`: used for text analysis and game name similarity calculation
- `date_release`: whether a game is new
- `win/mac/linux`: platform compatibility filtering
- `price_final/price_original`: price analysis/filter

The dataset `recommendations.csv` may be used in order to build a complete recommendation system. E.g. `helpful`, `funny`, `is_recommended`, may be used for model training.

2.3 Limitations

- The current dataset only contains static game information and lacks user interaction data **(Solved by using `recommendations.csv`)**
- Prices and ratings may be outdated
- May not include price and availability differences in different regions
- Niche games may have less reviews
- Popular games may have review bombing
- Limited sample of reviews for newly released games

3. Methods

3.1 Method Overview

This project will use a hybrid recommendation system, combining multiple recommendation algorithms to overcome the limitations of a single method. The core methods include content-based recommendation, collaborative filtering, and knowledge-driven recommendation methods.

3.2 Content-Based Filtering

Recommend similar games based on the attributes and features of the games.

3.2.1 Feature Engineering

- **Game feature vector construction**:
  - Price: normalized price range (0-1)
  - Rating: weighted average rating (considering the number of reviews)
  - Platform: binary encoding
  - Title: TF-IDF vector of game title

3.2.3 Similarity

- Use **cosine similarity** to calculate the similarity between games
- Feature weight distribution
  - e.g. rating (0.4) + price (0.1) + text (0.4) + platform (0.05) + time (0.05)

3.3 Collaborative Filtering

3.3.1 User-Based Collaborative Filtering

Find user groups similar to the target user and recommend games that these users like.
Similarity: Use Pearson Correlation Coefficient to calculate the similarity

3.3.2 Item-Based Collaborative Filtering

Recommend based on the similarity of user rating patterns between games.

- 1. Build a game-rating matrix
- 2. Calculate the similarity between games
- 3. Predict the user's rating for unrated games

It should relatively stable, game features change slowly and easy to explain the reason for recommendation.

3.4 Knowledge-Based Recommendation

- Recommend based on the **price range** of the user's historical purchases
- Prioritize the games that supported by the user's device
- Filter low-rated games (e.g. rating < 3.0) and recommend high rated games (e.g. rating > 4.5)

3.5 Hybrid Strategy

1. **Content recommendation**: basic recommendations, solve the cold start problem
2. **Collaborative filtering**: Discover preferences
3. **Knowledge-driven**: Handle boundary

Combine the three algorithms using linear combination:
`score = α × content_score + β × cf_score + γ × knowledge_score`

4. Evaluation

4.1 Recommendation model evaluation indicators

Root mean square error (RMSE): Measures the accuracy of rating predictions
Mean absolute error (MAE): Evaluates the bias
Precision@N (Precision@N): Relevant recommended games / N

- N =5, 10, 15...
  Recall@K (Recall@N): = Relevant recommended games / Games that users actually like
- Measure recommendation coverage
  Normalized Discounted Cumulative Gain (NDCG@K):
- Consider the importance of recommendation position

4.2 Recommendation system evaluation indicators

1. **Click-through rate (CTR)**: umber of clicks/number of impressions
2. **Conversion rate**: the proportion of games added to wishlist/actually purchased
3. **User retention rate**: 7/30-day retention rate

Core indicators

1. **NDCG@10**: Comprehensive consideration of recommendation quality and ranking
2. **Conversion rate**: How many purchases were made

4.3 Computational requirements and real-time performance

- The real-time recommendation generation response time should less than 1s
- Daily update of user preferences, computing time less than 30 minutes
- Concurrent user support

4.4 User Research Design

In order to evaluate the performance of the recommendation system in real scenarios and collect feedback for improvement, the study will recruit several Steam users (divided into three levels according to the number of games owned: novice <10, medium 10-50, and senior >50).

First, the user will use current Steam system to record behavior and satisfaction, then switching to the designed system for comparative testing the next day, and finally cross-validation through A/B testing one week later (50% of users use the new system and 50% use the baseline system to eliminate learning effects).

User behavior data will be collected,including recommendation click patterns, page dwell time, purchase decision path. User feedback will be collect by a questionnaire, covering recommendation quality (interest matching, price rationality, 1-5 point rating of recommendation willingness), system usability (response speed, interface friendliness, explanation clarity) and comparative evaluation (system advantages, function preferences, improvement suggestions).

Finally, data analysis is conducted to optimize the system based on the results.