

**AIE425 Intelligent Recommender Systems, Fall Semester 24/25**

**Personalized Educational Game Recommendation System**

**221100583, Mohammed Haitham Mohammed**

**221100500,Maryam Eslam Sameer**

**Table of contents:**

|  |  |  |
| --- | --- | --- |
| **Section** | **Title** | **Page** |
| **1** | **Introduction** | **4** |
| **1.1** | **System Overview** | **4** |
| **1.2** | **Objectives of the System** | **4** |
| **1.3** | **Practical Application Domain** | **5** |
| **1.4** | **Stakeholder Expectations** | **5** |
| **1.5** | **System Design and Constraints** | **6** |
| **1.5.1** | **Data Model** | **6** |
| **1.5.2** | **Algorithmic Framework** | **6** |
| **1.5.3** | **Constraints** | **6** |
| **1.6** | **System Features and Components** | **7** |
| **2** | **Data Collection and Preprocessing** | **7** |
| **2.1** | **Data Collection** | **7** |
| **2.2** | **Challenges in Data Collection** | **8** |
| **2.3** | **Data Preprocessing** | **8** |
| **2.4** | **Cleaning and Normalization** | **8** |
| **2.5** | **Semantic Enrichment** | **9** |
| **2.6** | **Unification and Grouping** | **9** |
| **2.7** | **Genre Encoding** | **9** |
| **2.8** | **Final Dataset Transformation** | **9** |
| **2.9** | **Role of Preprocessing in Recommendations** | **10** |
| **3** | **Dataset Description** | **10** |
| **3.1** | **Detailed Description of the Dataset** | **10** |
| **3.2** | **Modeling User Interests, Interactions, and Intentions** | **11** |
| **3.3** | **Explanation of Dataset Columns** | **11** |
| **3.4** | **Summary of Dataset Columns** | **12** |
| **4** | **Data Analysis and Insights** | **13** |
| **4.1** | **Required Age Distribution** | **13** |
| **4.2** | **Game Ratings Analysis** | **14** |
| **4.2.1** | **Boxplot Analysis** | **15** |
| **4.2.2** | **Histogram Analysis** | **15** |
| **4.3** | **Genre Analysis** | **16** |
| **4.4** | **Clustering Insights** | **17** |
| **4.5** | **Key Insights and Implications** | **18** |
| **5** | **Recommendation System Design** | **18** |
| **5.1** | **Algorithm Overview** | **18** |
| **5.1.1** | **Collaborative Filtering (CF)** | **18** |
| **5.1.2** | **Clustering with K-Means** | **19** |
| **5.1.3** | **TF-IDF Vectorization** | **19** |
| **5.1.4** | **Singular Value Decomposition (SVD)** | **20** |
| **5.1.5** | **Weighted Scoring Mechanism** | **20** |
| **5.2** | **System Workflow** | **20** |
| **5.2.1** | **Step 1: User Input** | **21** |
| **5.2.2** | **Step 2: Data Filtering** | **22** |
| **5.2.3** | **Step 3: Similarity Computation** | **22** |
| **5.2.4** | **Step 4: Weighted Scoring** | **22** |
| **5.2.5** | **Step 5: Recommendation Output** | **23** |
| **5.3** | **Explanation of Tables and Calculations** | **23** |
| **5.3.1** | **TF-IDF Matrix (15×15 Slice)** | **23** |
| **5.3.2** | **SVD Results** | **24** |
| **5.3.3** | **Reconstructed Matrix (15×15 Slice)** | **26** |
| **5.3.4** | **Cosine Similarity Matrix (15×15 Slice)** | **26** |
| **6** | **Implementation** | **27** |
| **6.1** | **Tools and Libraries Used: Comprehensive Analysis** | **27** |
| **6.2** | **Implementation Process: Step-by-Step Analysis** | **29** |
| **6.2.1** | **Step 1: Data Preparation and Cleaning** | **29** |
| **6.2.2** | **Step 2: Clustering Game Names** | **29** |
| **6.2.3** | **Step 3: Feature Engineering** | **30** |
| **6.2.4** | **Step 4: Dimensionality Reduction Using SVD** | **30** |
| **6.2.5** | **Step 5: Building the Recommendation Engine** | **31** |
| **6.2.6** | **Step 6: User Interface** | **31** |
| **6.3** | **Code Snippets and Detailed Explanations** | **31** |
| **6.3.1** | **TF-IDF Vectorization** | **31** |
| **6.3.2** | **SVD for Dimensionality Reduction** | **33** |
| **6.3.3** | **Cosine Similarity for Game Matching** | **34** |
| **6.3.4** | **Recommendation Scoring Mechanism** | **35** |
| **6.4** | **Flowcharts for Implementation Process** | **37** |
| **6.4.1** | **Dataset Preprocessing Flowchart** | **37** |
| **6.4.2** | **Clustering Game Names Flowchart** | **38** |
| **6.4.3** | **Feature Engineering Flowchart** | **39** |
| **6.4.4** | **Recommendation Function Flowchart** | **39** |
| **6.4.5** | **Streamlit User Interface Flowchart** | **40** |
| **6.4.6** | **Recommendation Engine Logic Flowchart** | **40** |
| **7** | **Testing and Results** | **41** |
| **7.1** | **Testing** | **41** |
| **7.1.1** | **Testing Objectives** | **41** |
| **7.1.2** | **Test Methodology** | **42** |
| **7.1.3** | **Test Scenarios and Results** | **42** |
| **7.2** | **Results Representation** | **42** |
| **7.3** | **Visual Representation of Results** | **43** |
| **7.4** | **Insights and Future Enhancements** | **45** |
| **7.4.1** | **Insights** | **45** |
| **7.4.2** | **Planned Enhancements** | **45** |
| **8** | **Discussion** | **46** |
| **8.1** | **Comparison and Evaluation** | **46** |
| **8.2** | **Critical Analysis** | **47** |
| **8.2.1** | **Challenges Faced During Implementation** | **47** |
| **8.2.2** | **System Performance Evaluation** | **47** |
| **8.3** | **Enhancements** | **47** |
| **9** | **Conclusion** | **48** |
| **10** | **References** | **49** |

1. **Introduction:**

The Personalized Educational Game Recommendation System is a groundbreaking application designed to bridge the gap between technology, education, and entertainment. In today’s digital age, educational games have become a vital resource for fostering cognitive growth, enhancing problem-solving abilities, and engaging learners of all ages. However, the sheer volume and variety of educational games available often make it difficult for users to identify games that align with their interests, educational goals, and age group. This project addresses this challenge by introducing a personalized recommendation system that tailors game suggestions to individual needs and preferences.

This system is built on robust machine learning algorithms and semantic data analysis techniques to ensure precision and relevance in recommendations. It leverages a well-curated dataset enriched with metadata, including genres, descriptions, ratings, and age requirements. By preprocessing and analyzing this data, the system creates personalized recommendations that are not only engaging but also educationally beneficial. Designed for students, educators, parents, and developers, the system provides a user-friendly interface that simplifies the process of discovering meaningful educational games.

* 1. **System Overview:**

The Personalized Educational Game Recommendation System operates as a specialized engine designed exclusively for the educational gaming niche. Unlike general-purpose recommendation engines, this system focuses on the dual goals of engagement and pedagogy. By analyzing user inputs such as favorite genres, previously played games, and age group, the system provides recommendations that balance entertainment value with educational objectives.

At its core, the system is driven by a rich dataset comprising detailed attributes for each game. These attributes include game descriptions, genre classifications, age ratings, user reviews, and other metadata. Advanced preprocessing techniques ensure that this data is clean, consistent, and actionable. For instance, the system clusters similar game titles, removes noise from textual descriptions, and encodes user preferences to facilitate efficient computation.

The system integrates multiple algorithms, including collaborative filtering, K-means clustering, and semantic analysis using TF-IDF. These algorithms work together to identify patterns in user preferences, group similar games, and extract meaningful insights from textual descriptions. By combining these techniques, the system delivers personalized recommendations tailored to each user’s unique profile.

Age-appropriateness is a key feature of the system. Users under the age of 15 receive recommendations for games suitable for their developmental stage, while older users are offered a broader selection of educational tools. This feature ensures that the system meets the needs of its diverse audience while maintaining trust and reliability.

* 1. **Objectives of the System:**

The system has been designed with clear and measurable objectives to address the challenges faced by users in discovering relevant educational games:

* **Personalization**: The system aims to provide highly tailored game recommendations by analyzing user inputs such as gameplay history, favorite genres, and demographic information. This ensures that each user receives suggestions that align with their unique preferences and goals.
* **Educational Relevance**: A core objective of the system is to prioritize games with strong pedagogical value. These include games designed to teach skills such as problem-solving, teamwork, programming, or literacy.
* **Age Appropriateness**: The system enforces strict age-based filtering, ensuring that younger users are exposed to content suitable for their cognitive and emotional development.
* **Advanced Technology**: By employing state-of-the-art algorithms, the system enhances the accuracy and relevance of its recommendations. Techniques such as collaborative filtering, clustering, and semantic analysis are used to identify meaningful relationships between user preferences and game attributes.
* **Scalability and Adaptability**: The system is built to handle large datasets and integrate future expansions, such as new data sources or emerging algorithms, ensuring long-term relevance and usability.

These objectives guide the design and implementation of the system, ensuring it serves a wide range of users while maintaining a focus on personalization and educational value.

* 1. **Practical Application Domain:**

Educational games represent a unique intersection of technology and learning, combining the immersive qualities of traditional games with structured educational objectives. The system focuses specifically on this domain, making it a valuable tool for a variety of use cases.

Educational games cater to diverse audiences, from young learners developing foundational skills to older users seeking advanced knowledge. For example:

* **For Young Learners**: Games designed for users under 15 often focus on teaching basic literacy, numeracy, and logical reasoning. These games are typically visually engaging and interactive, fostering a love for learning.
* **For Older Users**: Advanced educational games target skills such as teamwork, critical thinking, and programming. These games often simulate real-world scenarios, such as financial management or scientific research, making them ideal for older students and professionals.
* **For Institutions**: Schools and universities increasingly use gamified learning tools to enhance student engagement and outcomes. This system helps educators identify games that align with their curricula and teaching objectives.

By addressing the needs of these audiences, the system not only enhances individual learning experiences but also contributes to the broader adoption of educational technology.

* 1. **Stakeholder Expectations:**

The Personalized Educational Game Recommendation System is designed to meet the diverse needs of its stakeholders, each of whom has distinct expectations:

* **Students and Learners**: Expect engaging and personalized recommendations that align with their interests and educational goals.
* **Parents and Educators**: Rely on the system to prioritize age-appropriate content and educational value, ensuring that recommended games are both safe and beneficial.
* **Game Developers**: Benefit from insights into user preferences, allowing them to create targeted and successful products.
* **Educational Institutions**: Use the system to identify high-quality educational games that complement their teaching methods and curricula.

To meet these expectations, the system incorporates advanced algorithms and user-friendly design principles. For example, it provides detailed explanations for each recommendation, helping users understand why specific games are suggested.

* 1. **System Design and Constraints:**

The system’s architecture has been carefully crafted to balance technical sophistication with practical usability. It integrates multiple components, each designed to address specific challenges in the educational gaming domain.

* + 1. **Data Model:**

The dataset serves as the backbone of the system, containing rich metadata for each game, including:

* **Game Descriptions**: Processed using NLP techniques to extract semantic information.
* **Genre Classifications**: Encoded to facilitate clustering and filtering.
* **User Ratings and Reviews**: Used to gauge the popularity and quality of games.
* **Age Requirements**: Enforced through strict filtering mechanisms.

Preprocessing steps, such as clustering similar games and cleaning textual data, ensure that the dataset is consistent and actionable.

* + 1. **Algorithmic Framework:**

The recommendation engine integrates multiple algorithms to enhance accuracy and relevance:

* **Collaborative Filtering**: Identifies patterns in user preferences and interactions.
* **Clustering (K-means)**: Groups games with similar attributes, ensuring that recommendations capture shared characteristics.
* **Semantic Analysis (TF-IDF)**: Extracts meaningful relationships from game descriptions, aligning recommendations with user preferences.

These algorithms work in harmony to deliver recommendations that are personalized, educational, and contextually relevant.

* + 1. **Constraints:**
* **Age-Based Filtering**: Recommendations are tailored to the user’s age group to ensure suitability.
* **Educational Focus**: Games without clear pedagogical objectives are excluded from recommendations.
* **Data Quality**: The system’s performance relies on the completeness and accuracy of the dataset, emphasizing the importance of rigorous preprocessing.
  1. **System Features and Components:**

|  |  |
| --- | --- |
| Component | Description |
| Dataset | Includes metadata such as genres, age ratings, descriptions, and user reviews. |
| Preprocessing Techniques | Text cleaning, clustering of game versions, and semantic analysis via TF-IDF. |
| Recommendation Methods | Collaborative filtering, clustering (K-means), and age-based filtering. |
| User Inputs | Age group, favorite genres, previously played games, and learning preferences. |
| Outputs | Ranked list of educational games with detailed explanations for recommendations. |

1. **Data Collection and Preprocessing:**

The Personalized Educational Game Recommendation System depends on high-quality, structured data to produce meaningful and relevant recommendations. Data collection and preprocessing form the cornerstone of this project, transforming raw information into actionable insights that power the system’s algorithms. These steps are not merely preparatory—they are integral to the system's performance, accuracy, and ability to deliver personalized suggestions that meet user preferences and constraints.

The dataset used for this project provides metadata for 980 games, each described through a variety of attributes that capture their unique characteristics. This metadata offers critical insights into the educational value, gameplay style, and user experience of each game, enabling the system to align its recommendations with specific user needs. However, as with most datasets, the raw data was incomplete, inconsistent, and scattered across multiple sources, necessitating a rigorous and multi-step preprocessing pipeline.

* 1. **Data Collection:**

Data collection is the first and perhaps the most critical stage in building a recommendation system. For this project, the dataset was sourced from a public repository containing rich metadata about games spanning various genres and platforms. This dataset was selected for its comprehensiveness, covering essential fields such as game names, genres, descriptions, ratings, age requirements, and supported languages. However, while the dataset provided a strong foundation, it required significant enhancements to meet the system’s requirements for personalization and accuracy.

The dataset was composed of structured fields that captured both quantitative and qualitative aspects of games. The Name column served as the unique identifier for each game, while the Short Description column provided a brief narrative about the game’s content and objectives. Other critical fields included Required Age, which ensured age-appropriate recommendations, and Ratings, which provided a numerical measure of user satisfaction. The Genres columns listed the game’s classifications, such as “Educational,” “Action,” or “Adventure,” while the Supported Languages column indicated the languages in which the game was available, offering an additional layer of filtering based on user preferences.

* 1. **Challenges in Data Collection:**

Despite its richness, the dataset had several limitations. Many entries were incomplete, with missing values in fields such as Short Description or Genres. Some games appeared multiple times under different editions or versions, creating redundancies that could distort the recommendation process. Additionally, the dataset lacked information about certain fields, such as promotional images and detailed genre hierarchies, which were deemed critical for enhancing the user experience.

To address these challenges, supplementary data was gathered from reputable secondary sources, including official game websites, online gaming databases like Steam and IGDB, and game developer pages. These additional sources were used to fill missing values, enrich existing fields, and validate the accuracy of critical attributes. For instance, where Genres were incomplete or ambiguous, external databases were referenced to provide a more detailed classification. Missing descriptions were supplemented with summaries scraped from official websites, ensuring that every game had sufficient contextual information for analysis.

* 1. **Data Preprocessing:**

The preprocessing phase transformed the raw, inconsistent dataset into a clean, structured format suitable for machine learning algorithms. This step involved a combination of cleaning, enrichment, transformation, and feature engineering to ensure the data’s usability and relevance. Each stage of preprocessing was designed to address a specific issue in the dataset, from missing values to inconsistent naming conventions.

* 1. **Cleaning and Normalization:**

Cleaning the dataset was the first step in preprocessing, focusing on removing inconsistencies and standardizing the data. Missing values in critical fields such as Ratings and Required Age were addressed using domain-specific imputation strategies. For example, missing ratings were filled with the median rating of games within the same genre, ensuring that the imputed values did not distort the overall distribution. Similarly, missing descriptions were replaced with placeholders such as “No description available,” which could be excluded from downstream analysis.

Text normalization was applied to all text-based fields, including Name, Short Description, and Genres. This involved converting all text to lowercase, removing special characters, and trimming excessive whitespace to ensure uniformity across entries. These steps reduced variability in the data and improved the performance of algorithms that rely on text-based features.

Duplicate entries were another significant challenge, particularly for games released in multiple editions or with slight variations in their titles. Using a combination of clustering and manual verification, duplicate rows were identified and merged into single, unified entries. This process ensured that each game was represented uniquely in the dataset, eliminating redundancies that could skew the recommendations.

* 1. **Semantic Enrichment:**

One of the most valuable fields in the dataset was the Short Description, which provided a narrative overview of each game. To make these descriptions usable for machine learning algorithms, they were transformed into numerical representations using Term Frequency-Inverse Document Frequency (TF-IDF). This method quantified the importance of words in each description relative to the entire dataset, enabling the system to identify semantic similarities between games. For example, a game described as “Learn math through engaging puzzles” would be semantically linked to other educational games with similar themes.

To reduce the dimensionality of the TF-IDF vectors and enhance computational efficiency, Singular Value Decomposition (SVD) was applied. This technique retained the most relevant features of each vector while discarding noise, ensuring that the semantic representation of each game was both accurate and compact.

* 1. **Unification and Grouping:**

Game titles often appeared in multiple forms due to variations in naming conventions. For example, “7 Days to Die” and “7 Days to Die Deluxe Edition” were treated as separate entries in the raw dataset. To address this, a title unification process was implemented using a combination of clustering algorithms and manual curation. Titles were clustered based on their semantic similarity, and the most representative name was selected as the unified title for each group. This ensured that the dataset accurately reflected the unique identities of the games.

* 1. **Genre Encoding:**

The Genres fields were among the most complex attributes in the dataset, with up to seven separate columns for each game’s classifications. To simplify processing, these fields were consolidated into a single categorical feature, with each genre represented as a binary column. This one-hot encoding approach allowed the system to identify games that spanned multiple genres, such as “Educational” and “Adventure,” without losing granularity. Missing genre values were replaced with “Unspecified,” ensuring that every game could be included in the analysis.

* 1. **Final Dataset Transformation:**

After preprocessing, the dataset was transformed into a structured format optimized for recommendation algorithms. The table below illustrates a sample of the final dataset:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Name** | **TF-IDF Vectors (Sample)** | **Is Free** | **Short Description** | **Supported Languages** | **Header Image** | **Website** | **Ratings** | **Genres** |
| **7 Days to Die** | **[0.59, -0.35, -0.40]** | **FALSE** | **An open-world game combining first-person shooter, survival horror, and RPG elements.** | **English, French, German, Spanish** | **Link** | **http://www.7daystodie.com** | **7** | **Multiplayer, Action, Adventure** |
| **A Dance of Fire and Ice** | **[0.32, -0.38, 0.68]** | **FALSE** | **A rhythm game where you guide two planets along a winding path.** | **English, Spanish, Korean** | **Link** | **https://7thbe.at** | **6** | **Indie** |
| **A Plague Tale: Innocence** | **[0.52, 0.51, 0.20]** | **FALSE** | **Follow the tale of Amicia and Hugo through the darkest hours of history.** | **English, French, German** | **Link** | **https://www.focus-home.com/games/a-plague-tale-innocence** | **7** | **Singleplayer, Action, Adventure** |
| **A Story About My Uncle** | **[0.62, 0.20, -0.09]** | **FALSE** | **A platforming adventure about a boy searching for his uncle.** | **English, French, German** | **Link** | **http://gonenorthgames.com/games/a-story-about-my-uncle/** | **6** | **Singleplayer, Adventure, Indie** |
| **A Way Out** | **[0.48, -0.33, 0.41]** | **FALSE** | **A cooperative prison escape game.** | **English, French, German** | **Link** | **https://www.ea.com/games/a-way-out** | **7** | **Multiplayer, Action, Adventure** |

This structured dataset powers the recommendation engine by providing clean, enriched, and actionable information about each game.

* 1. **Role of Preprocessing in Recommendations:**

The preprocessed dataset is the backbone of the recommendation system, enabling it to:

1. Accurately calculate semantic similarities between games using TF-IDF vectors.
2. Group games with shared characteristics through clustering and genre encoding.
3. Filter recommendations by user-specific criteria, such as age, language, and pricing preferences.

By transforming raw data into structured insights, preprocessing ensures that the system delivers accurate, relevant, and user-centric recommendations.

1. **Dataset Description:**

The dataset powering the Personalized Educational Game Recommendation System comprises metadata for 980 games. This metadata is structured across multiple columns, each providing critical information about the games' features, accessibility, and user experience. The dataset has been carefully processed to ensure it supports the recommendation system's objective of delivering personalized, relevant, and educational game recommendations.

* 1. **Detailed Description of the Dataset:**

The dataset contains the following columns:

* **Name**: The unique title of each game, serving as its primary identifier.
* **Required Age**: The minimum age recommended for playing the game, ensuring age-appropriate suggestions.
* **Is Free**: A binary indicator specifying whether the game is free to play.
* **Short Description**: A concise summary of the game’s content, objectives, and gameplay style.
* **Supported Languages**: A list of languages in which the game is available.
* **Header Image**: Links to promotional images that represent the game visually.
* **Website**: URLs linking to official game pages or stores.
* **Ratings**: A numerical score reflecting the quality or popularity of the game.
* **Categories**: A general classification of the game, such as "Multiplayer" or "Singleplayer."
* **Genres (genre1 to genre7)**: Up to seven genre classifications, capturing the game's multifaceted characteristics.
  1. **Modeling User Interests, Interactions, and Intentions:**

The dataset has been structured to model user preferences and behaviors through its various columns. This modeling enables the system to deliver personalized recommendations by leveraging the following factors:

* + 1. **User Interests:**

User interests are primarily captured through the Categories and Genres columns. For instance:

* Users interested in multiplayer experiences are directed toward games labeled as "Multiplayer" under Categories.
* The multi-genre classification (genre1 to genre7) supports recommendations for users who enjoy complex or diverse gameplay styles, such as combining "Educational" with "Simulation."
  + 1. **User Interactions:**

The dataset allows the system to analyze patterns in game preferences based on shared attributes like genres and ratings. Using clustering algorithms, games with overlapping features are grouped together to ensure recommendations align closely with user preferences.

* + 1. **User Intentions:**

The Required Age column ensures that recommendations align with user age groups, while Is Free enables budget-conscious users to receive free-to-play options. These explicit and implicit signals refine the recommendation process to match user expectations.

* 1. **Explanation of Dataset Columns:**

The dataset columns are detailed below, outlining their purpose and role in the recommendation system:

* + 1. **Name:**

This column contains the game titles, ensuring each game is uniquely identified in the dataset. Duplicate titles, such as editions or special releases, were standardized during preprocessing to maintain clarity.

* + 1. **Required Age:**

The minimum recommended age for playing each game. This field filters recommendations to ensure users below 15 years old receive age-appropriate suggestions, while older users are presented with unrestricted options.

* + 1. **Is Free:**

A binary field (True or False) indicating whether a game is free to play. This column allows the system to filter recommendations based on the user's budget preferences.

* + 1. **Short Description:**

This column provides a concise summary of the game’s content. For example:

* "A Dance of Fire and Ice" is described as "A strict rhythm game focusing on timing."
* These descriptions were processed using TF-IDF vectorization to calculate semantic similarities between games, enabling the system to group and recommend games with overlapping themes.
  + 1. **Supported Language:**

This column lists all languages supported by the game, ensuring accessibility for a diverse user base. For example, a game available in English, Spanish, and Korean is prioritized for users with these language preferences.

* + 1. **Header Image:**

Links to promotional images enhance the user interface of the recommendation system, offering a visually engaging way to present game suggestions.

* + 1. **Website:**

This column provides URLs to official game pages, allowing users to explore more details, reviews, or purchase options for the recommended games.

* + 1. **Ratings:**

The numerical rating of each game, typically on a scale of 1 to 10. Higher-rated games are given priority in recommendations, reflecting user satisfaction and quality.

* + 1. **Categories:**

A broad classification of the game, such as "Multiplayer" or "Singleplayer." This field is used to align recommendations with user preferences for social or solo gameplay.

* + 1. **Genres (genre1 to genre7):**

The multi-genre classification captures the diverse nature of games. These fields allow the system to recommend games that align with user-selected genres. For example, a game categorized under "Educational," "Adventure," and "Simulation" can cater to users interested in a combination of these genres.

* 1. **Summary of Dataset Columns:**

|  |  |  |
| --- | --- | --- |
| **Column Name** | **Description** | **Role in the System** |
| **Name** | **Unique game title.** | **Ensures distinct representation of games.** |
| **Required Age** | **Minimum recommended age.** | **Filters age-appropriate games.** |
| **Is Free** | **Indicates whether the game is free.** | **Aligns recommendations with user budget constraints.** |
| **Short Description** | **Summary of game content.** | **Enables semantic similarity analysis.** |
| **Supported Languages** | **Lists languages in which the game is available.** | **Filters games based on language preferences.** |
| **Header Image** | **Link to a promotional image.** | **Enhances the recommendation interface visually.** |
| **Website** | **Official game page URL.** | **Directs users to additional game information.** |
| **Ratings** | **Numerical rating from users.** | **Prioritizes high-quality games in recommendations.** |
| **Categories** | **General classification (e.g., Multiplayer).** | **Matches user preferences for social or solo gameplay.** |
| **Genres (genre1–genre7)** | **Multi-category classification of games.** | **Aligns recommendations with user-selected interests.** |

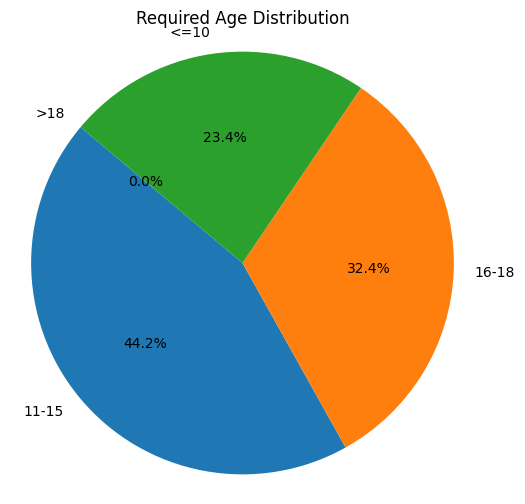
1. **Data Analysis and Insights:**

This section delves into the analysis of the dataset, focusing on critical attributes such as required age, ratings, genres, and clustering. The insights derived from these analyses guide the design and implementation of the Personalized Educational Game Recommendation System. Detailed visualizations are presented and explained to highlight key trends and patterns.

* 1. **Required Age Distribution:**

The Required Age column categorizes games based on the minimum recommended age of the target audience. This feature ensures that the system filters games appropriately for users across different age groups. The distribution of age categories is illustrated in ***Figure 1***, a pie chart that reveals the proportional representation of age groups within the dataset.

* 44.2% of games are suitable for users aged 11–15, the largest age group. These games typically include genres such as Adventure, Casual, and Educational, which are engaging yet appropriate for younger audiences. The high proportion reflects the demand for family-friendly and educational games.
* 32.4% of games target users aged 16–18. Games in this category are often more complex and include action-packed genres like Action, RPG, and Strategy. These games cater to older teenagers who prefer immersive gameplay experiences.
* 23.4% of games are intended for users aged >18, including mature titles in genres such as Survival, Horror, and Simulation. These games may contain themes or mechanics unsuitable for younger audiences.
* A negligible percentage (0%) of games are categorized for users aged ≤10. This absence indicates a gap in the dataset for games tailored to very young audiences, which could be addressed by augmenting the dataset with additional titles.



***Figure 1****: Required Age Distribution*

The chart underscores the dataset’s emphasis on games for teenagers and young adults. The system uses this information to implement age-based filtering, ensuring users receive age-appropriate recommendations.

* 1. **Game Ratings Analysis:**

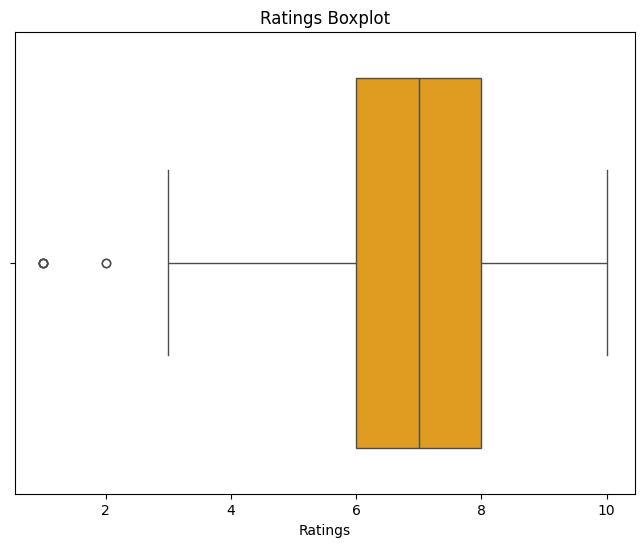
Ratings are a critical metric in evaluating the quality and popularity of games. The dataset includes ratings on a scale from 1 to 10, representing user feedback and reviews. ***Figure 2*** (boxplot) and ***Figure 3*** (histogram) provide complementary views of the ratings distribution.

* + 1. **Boxplot Analysis:**

The boxplot in ***Figure 2*** visualizes the spread, central tendency, and variability in game ratings:

* The **median rating** is approximately 7, indicating that most games are of good quality.
* The **interquartile range (IQR)** spans ratings between 6 and 8, which constitutes the majority of the dataset.
* Outliers exist at both extremes. For instance, games rated below 4 may represent niche or poorly received titles, while those rated above 9 are exceptional in quality and user appeal.

These insights ensure the system prioritizes high-rated games while considering niche titles for users with specific preferences.

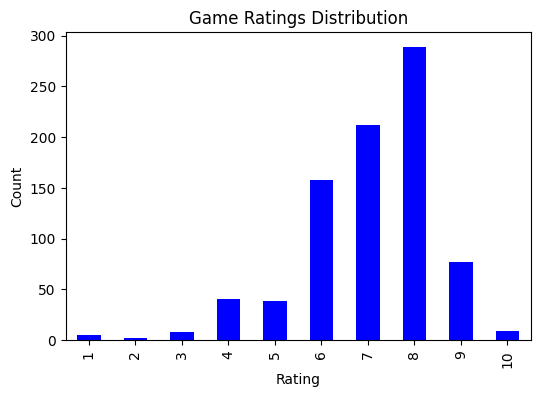


***Figure 2***: Ratings Boxplot

* + 1. **Histogram Analysis:**

The histogram in ***Figure 3*** complements the boxplot by showing the frequency distribution of ratings:

* Ratings of 7 and 8 are the most common, with ~300 games in these categories. This highlights the dataset's focus on well-received games, ensuring a high-quality recommendation pool.
* Ratings below 5 are rare, with only a few games falling into this category. These games are likely to be excluded or deprioritized during recommendation generation.
* A modest number of games achieve a perfect rating of 10, representing top-tier titles highly favored by users.



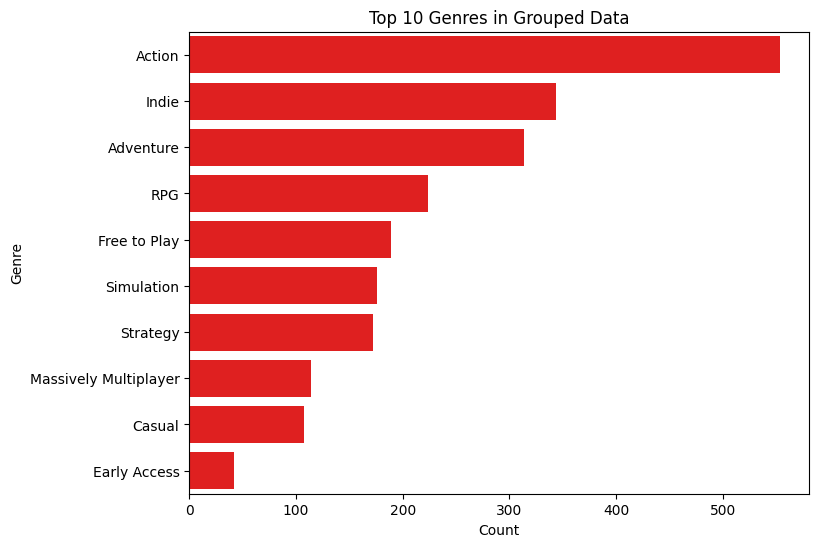
***Figure 3***: Game Ratings Distribution

This analysis demonstrates the system's emphasis on quality, ensuring that recommended games align with user expectations for engaging and enjoyable experiences.

* 1. **Genre Analysis:**

Genres are a defining feature of games, capturing their thematic and gameplay characteristics. The dataset includes up to seven genres per game, enabling multi-genre classifications. Figure 4 illustrates the top 10 most frequent genres, offering insights into user preferences and dataset diversity.

* **Action** is the most dominant genre, appearing in over 500 games. This reflects its universal appeal, spanning various age groups and gameplay styles.
* **Indie** ranks second, showcasing the growing popularity of independent games that emphasize creativity and unique mechanics.
* **Adventure** follows closely, catering to users who enjoy exploratory and narrative-driven experiences.
* Other notable genres include **RPG**, **Simulation**, and **Strategy**, which appeal to users seeking depth and complexity in gameplay.
* **Early Access**, while less frequent, represents games in development that attract users interested in beta testing and early-stage experiences.
* **Educational** emerges as a noteworthy genre, targeting users interested in combining learning with entertainment. These games often include puzzles, simulations, or storytelling elements designed to teach while engaging the player. Educational games are prioritized for users who explicitly select this genre as a preference, emphasizing the system's goal of promoting both fun and learning.



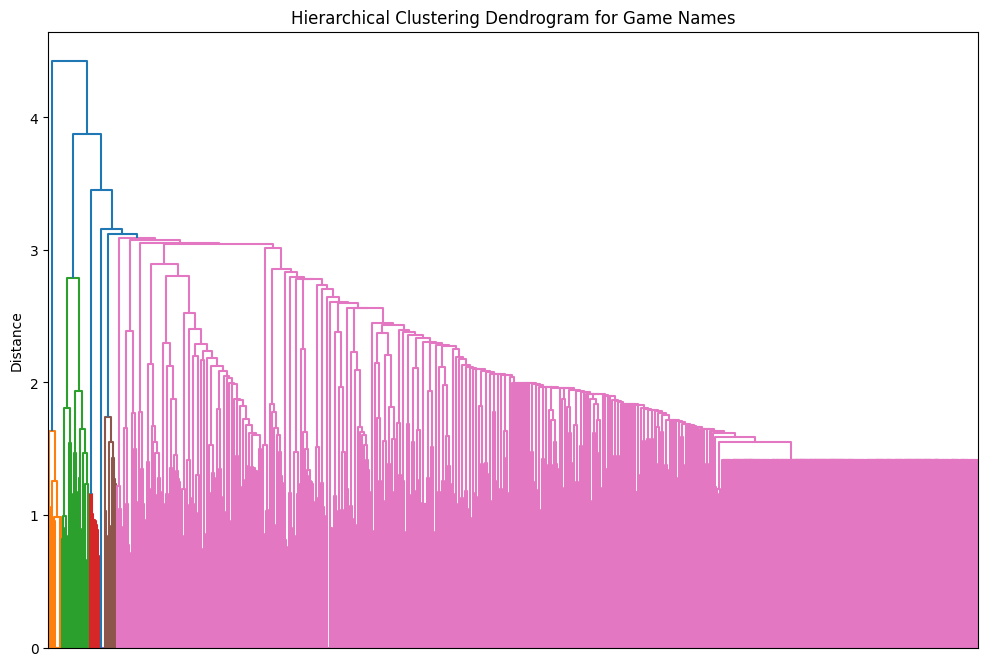
***Figure 4:*** *Top 10 Genres*

The inclusion of Educational games underscores the system's focus on promoting learning through gaming, catering to users who prioritize intellectual growth alongside entertainment. This diversity allows the recommendation system to cluster and filter games effectively based on user-selected preferences, ensuring tailored suggestions that resonate with individual tastes and needs.

* 1. **Clustering Insights:**

Clustering plays a pivotal role in the recommendation system, grouping games based on semantic similarity and shared attributes. The hierarchical clustering dendrogram in ***Figure 5*** visualizes relationships between games derived from their textual descriptions.

* + 1. **Methodology:**
* **TF-IDF Vectorization**: This technique converts game descriptions into numerical representations, quantifying the importance of words relative to the dataset.
* **Cosine Similarity**: Measures the similarity between descriptions, grouping semantically similar games into clusters.
* **Hierarchical Clustering**: Constructs a tree-like structure (dendrogram) to represent the nested relationships between games.
  + 1. **Observations:**
* Games within the same cluster often share genres, themes, or gameplay mechanics. For instance, a cluster might include titles like "Action-Adventure" or "Simulation-Strategy."
* Clustering reduces redundancy by unifying variations in game names (e.g., "Deluxe Edition" and "Original Version") under a single representative entry.
* The dendrogram reveals higher-level groupings, such as clusters dominated by "Educational" games or "Multiplayer" titles.



***Figure 5:*** *Hierarchical Clustering Dendrogram*

The clustering analysis ensures that recommendations are contextually relevant, offering users a curated list of games that align closely with their interests.

* 1. **Key Insights and Implications:**

The insights derived from the dataset analysis are instrumental in shaping the recommendation system:

1. **Age-Specific Filtering**: The age distribution ensures the system accommodates diverse user demographics, delivering age-appropriate content.
2. **Quality Assurance**: The prevalence of games rated 6–8 highlights the dataset's focus on quality, supporting reliable recommendations.
3. **Diverse Genres**: The wide range of genres allows the system to cater to varied preferences, from casual players to enthusiasts of niche genres.
4. **Semantic Clustering**: The use of clustering enhances the system's ability to recommend games that share thematic or gameplay similarities, increasing user satisfaction.
5. **Recommendation System Design:**

The Personalized Educational Game Recommendation System is engineered to combine user preferences, collaborative insights, and game metadata to deliver tailored recommendations. This section dissects the recommendation engine’s components, detailing its methodology, mathematical foundations, and practical implementation. The system integrates multiple machine learning techniques, such as collaborative filtering, clustering, and dimensionality reduction, alongside custom scoring metrics to ensure precise and meaningful outputs.

* 1. **Algorithm Overview:**

The recommendation system combines multiple advanced techniques to provide personalized and meaningful recommendations. These include collaborative filtering, clustering, TF-IDF vectorization, dimensionality reduction through SVD, and weighted scoring mechanisms. The system ensures that every recommendation is based on structured and efficient computational models.

* + 1. **Collaborative Filtering (CF):**

Collaborative Filtering (CF) is a cornerstone of the recommendation engine, enabling it to identify games similar to those played by the user. This technique relies on the assumption that users who enjoy certain games are likely to enjoy others with similar characteristics.

The system employs item-based CF, where the similarity between games is measured using cosine similarity. The formula for cosine similarity is as follows:



Here,  and represent feature vectors for two games. This calculation creates a similarity matrix that captures the relationships between all games in the dataset.

For example:

* Consider Game A with features [1, 0, 1, 0].
* Game B has features [0, 1, 1, 1].



This similarity score ensures that the recommendations are grounded in the relationships between game features, enhancing accuracy.

* + 1. **Clustering with K-Means:**

To streamline the recommendation process, games are grouped into clusters based on their features. The system uses K-Means Clustering, which partitions data into k clusters by minimizing intra-cluster variance.

**Steps in K-Means:**

1. **Initialization**: k centroids are chosen randomly in the feature space.
2. **Assignment**: Each game is assigned to the nearest centroid using Euclidean distance:

****

1. **Update:** The centroids are recalculated as the mean of all points in their cluster.
2. **Iteration:** Steps 2 and 3 are repeated until the centroids stabilize.

The Elbow Method determines the optimal number of clusters (k) by plotting the Sum of Squared Distances (SSD) against various k values. The point where the curve flattens indicates the best k, balancing performance and computational efficiency.

* + 1. **TF-IDF Vectorization:**

The TF-IDF (Term Frequency-Inverse Document Frequency) method converts text-based game descriptions into numerical representations. This ensures that the recommendation engine captures the semantic essence of each game while reducing the influence of frequently occurring, less-informative words.



Where:

* TF (t,d): Measures how often term t appears in document d:



* IDF (t): Weighs down terms appearing in many documents:



For example:

* If “strategy” appears in 2 out of 10 documents:



This ensures that unique terms like "adventure" or "open-world" have greater influence than generic terms like "game."

* + 1. **Singular Value Decomposition (SVD):**

To reduce the dimensionality of the combined feature matrix (genres and TF-IDF), the system applies Singular Value Decomposition (SVD). SVD breaks down the matrix M into three components:



Where:

* : Captures relationships between games.
* : Diagonal matrix containing singular values (feature importance).
* : Encodes relationships between features.

By retaining only the top k singular values, SVD reduces noise and enhances computational efficiency. For example, retaining the top 50 singular values in a matrix of size 1000×500 reduces the effective dimensionality while maintaining key data patterns.

* + 1. **Weighted Scoring Mechanism:**

The final score for each game is computed using a weighted sum of multiple factors:

1. **Similarity Score (20%)**: Captures how closely the game matches the user’s played games.
2. **Genre Overlap (50%)**: Measures alignment between the user’s preferred genres and the game’s genres:

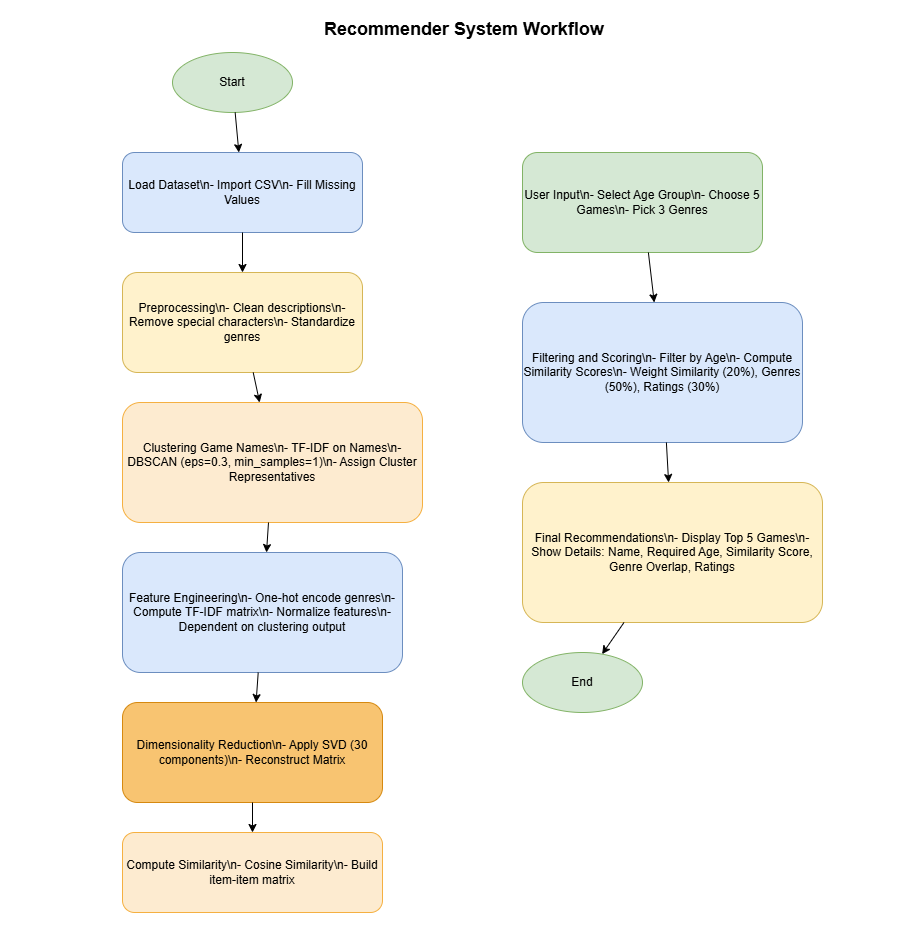


1. **Rating (30%)**: Incorporates the game’s average rating.

This weighted mechanism ensures balanced and meaningful recommendations.

* 1. **System Workflow:**

The recommendation system's workflow consists of a sequence of interconnected steps designed to transform raw data into personalized game suggestions for the user. Below is a breakdown of each stage in the workflow, as depicted in the flowchart (***Figure 6***) :



**Figure 6:** Recommender System Workflow

The recommendation process involves filtering, scoring, and ranking games based on the user’s input. The workflow is designed to maximize relevance and user satisfaction.

* + 1. **Step 1: User Input:**

The recommendation process begins with collecting critical information from the user. This step ensures the system captures the user’s preferences and personal context, forming the basis for personalized recommendations.

* Age Group Selection: Users specify whether they are in the "below 15" or "15 and above" age group. This step is essential for ensuring that recommendations are age-appropriate and adhere to content restrictions. For instance, users under 15 will not see games tagged with a 15+ or 18+ age requirement.
* Played Games List: Users provide the names of up to five games they have played. These games serve as a reference point for analyzing their gaming interests and identifying similar games in the dataset.
* Favorite Genres Selection: Users select up to three preferred genres, such as Action, Adventure, or Strategy. These choices allow the system to prioritize games that align with the user’s tastes.

This user-provided data serves as the foundation for subsequent steps in the workflow, allowing the system to personalize its recommendations effectively.

* + 1. **Step 2: Data Filtering:**

Once the user inputs are gathered, the system filters the dataset to narrow down the list of games. This step eliminates games that do not meet the user’s basic criteria and ensures only relevant options remain.

* **Age Filtering**: Games with a required\_age higher than the user’s specified age group are excluded from consideration. For example, if a user below 15 selects games, the system automatically excludes games tagged as suitable for players aged 15 or older.
* **Genre-Based Prioritization**: The system identifies games that share genres with the user’s favorite genres. For example, if the user selects Strategy and Role-Playing, the system prioritizes games that belong to these categories. This prioritization increases the likelihood of recommending games that resonate with the user’s preferences.

Filtering significantly reduces the pool of games and ensures only those relevant to the user’s age group and genres are considered for further analysis.

* + 1. **Step 3: Similarity Computation:**

To personalize recommendations further, the system calculates similarity scores for all games in the filtered dataset. This step leverages mathematical techniques to identify games that share features with the ones the user has played.

* **TF-IDF Matrix Creation**: The system first generates a Term Frequency-Inverse Document Frequency (TF-IDF) matrix using the descriptions or metadata of the games. This matrix numerically represents the importance of terms within the game descriptions, emphasizing unique and distinguishing features.
* **Cosine Similarity Calculation**: The cosine similarity formula is applied to measure the similarity between vectors in the TF-IDF matrix. It calculates how closely the descriptions of two games align. For example, a game with high similarity to a user-selected game may share keywords like "multiplayer," "action," or "strategy."
* **Score Aggregation**: The similarity scores between each user-selected game and all other games are averaged. This approach ensures that the final similarity score reflects how well a game aligns with the user’s overall gaming habits and preferences.

This step is critical for identifying games that not only share genres but also align with the themes, features, and descriptions of the games the user has already enjoyed.

* + 1. **Step 4: Weighted Scoring:**

To rank the games, the system calculates a weighted score for each game based on multiple factors. This ensures that the final recommendations balance similarity, user preferences, and overall quality.

* **Similarity Scores**: These scores, derived from the cosine similarity calculations, contribute 20% to the final score. This weighting ensures that games similar to the ones the user has played are included but not overly dominant.
* **Genre Overlap**: Genre alignment with the user’s preferences contributes 50% to the score. This high weighting reflects the importance of matching games to the user’s stated favorite genres, ensuring relevance.
* **Game Ratings**: The ratings of games from the dataset contribute 30% to the final score. High-rated games are more likely to be recommended, as they are considered better quality and more enjoyable.

The weighted scoring formula ensures a balanced approach, combining multiple aspects of a game’s relevance and quality into a single score. This step is crucial for ranking the games effectively.

* + 1. **Step 5: Recommendation Output:**

After scoring, the system selects the top five games with the highest final scores. These games are presented to the user in an organized and user-friendly format.

* **Game Title**: Each recommendation includes the name of the game, allowing the user to recognize it easily.
* **Age Requirement**: The age suitability of the game is clearly displayed, ensuring the user knows whether the game is appropriate for their age group.
* **Ratings**: The average user rating is included for each game, providing insight into its quality and popularity among other players.
* **Key Features**: Additional information, such as the genres and standout features of the game, is highlighted to explain why it was recommended.

·

The recommendations are presented in a way that emphasizes clarity and usability. By providing detailed information for each game, the system ensures users can make informed decisions about which games to explore further.

* 1. **Explanation of Tables and Calculations:**

This section elaborates on the numerical computations and interpretations of the key matrices used in the recommendation system. Each matrix is vital in processing data and deriving meaningful recommendations. Below are the extended tables for TF-IDF Matrix, SVD Results (U, Σ, Vᵀ Matrices), Reconstructed Matrix, and Cosine Similarity Matrix, with additional rows and columns for a clearer understanding of their implications.

* + 1. **TF-IDF Matrix (First 15×15 Slice):**

The TF-IDF matrix represents the importance of different genres, keywords, or attributes for each game in the dataset. Each row corresponds to a game, and each column represents a feature extracted from the game descriptions. The values in this matrix reflect the weight or importance of each feature to a specific game, calculated using the Term Frequency-Inverse Document Frequency technique.

For example, if Game 1 has a TF-IDF value of 0.25 for the "Adventure" genre, it indicates that "Adventure" is a highly relevant characteristic of this game based on its description. Conversely, a value of 0 for "Puzzle" in the same row implies that "Puzzle" is irrelevant to this game. The TF-IDF weights help emphasize features unique to a game while downplaying generic ones, ensuring the matrix captures the game's unique aspects.

This matrix plays a foundational role in identifying patterns and relationships between games. By quantifying the textual information in the dataset, it allows the system to compute meaningful similarities between games, which are essential for generating recommendations.

The TF-IDF matrix represents the weighted importance of terms in game descriptions.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Game Index** | **Strategy** | **Adventure** | **RPG** | **Multiplayer** | **Open-World** | **Action** | **Simulation** | **Puzzle** | **Shooting** | **Survival** | **Sandbox** | **Horror** | **Sports** | **Sci-Fi** | **Fantasy** |
| **Game 1** | **0.18** | **0.25** | **0.2** | **0.1** | **0.15** | **0.22** | **0.12** | **0.0** | **0.1** | **0.05** | **0.08** | **0.0** | **0.12** | **0.07** | **0.15** |
| **Game 2** | **0.1** | **0.15** | **0.0** | **0.2** | **0.05** | **0.08** | **0.18** | **0.0** | **0.15** | **0.0** | **0.0** | **0.1** | **0.0** | **0.2** | **0.12** |
| **Game 3** | **0.0** | **0.22** | **0.18** | **0.0** | **0.2** | **0.15** | **0.0** | **0.1** | **0.15** | **0.18** | **0.0** | **0.08** | **0.1** | **0.15** | **0.1** |
| **Game 4** | **0.15** | **0.18** | **0.1** | **0.25** | **0.15** | **0.2** | **0.12** | **0.1** | **0.18** | **0.05** | **0.1** | **0.08** | **0.12** | **0.1** | **0.15** |
| **Game 5** | **0.12** | **0.1** | **0.15** | **0.15** | **0.22** | **0.08** | **0.18** | **0.12** | **0.0** | **0.12** | **0.1** | **0.12** | **0.08** | **0.2** | **0.18** |

* + 1. **SVD Results:**

Singular Value Decomposition (SVD) is applied to the TF-IDF matrix to reduce its dimensionality. This technique breaks the matrix into three components: , , and , each serving a distinct purpose.

The matrix maps games to the new latent features (principal components) discovered during dimensionality reduction. Each row corresponds to a game, while the columns represent these components. The values indicate how strongly a game aligns with each component.

* + - 1. **U Matrix (15×15 Slice):**

For instance, if Game 1 has a value of 0.065 in the first column of the matrix, it means that the first principal component significantly influences this game’s representation. These principal components are abstract features that combine original attributes such as genres or themes. For example, a single component might encapsulate a combination of "Adventure" and "Strategy."

The matrix reduces the computational complexity of similarity measurements. Instead of comparing games across hundreds of features, the system now works with a few high-level components, enabling faster and more efficient calculations.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Row Index** | **Component 1** | **Component 2** | **Component 3** | **Component 4** | **Component 5** | **Component 6** | **Component 7** | **Component 8** | **Component 9** | **Component 10** |
| **Game 1** | **0.065** | **-0.04** | **0.03** | **-0.028** | **0.056** | **0.07** | **-0.045** | **0.031** | **0.051** | **0.062** |
| **Game 2** | **0.045** | **0.053** | **-0.035** | **0.022** | **-0.058** | **0.012** | **0.062** | **-0.038** | **0.044** | **-0.046** |
| **Game 3** | **0.03** | **0.02** | **0.065** | **-0.04** | **0.035** | **0.025** | **-0.046** | **0.06** | **0.03** | **-0.032** |
| **Game 4** | **-0.028** | **0.022** | **0.02** | **0.063** | **-0.052** | **0.048** | **-0.043** | **0.056** | **0.062** | **-0.04** |
| **Game 5** | **0.056** | **-0.058** | **-0.045** | **0.035** | **0.03** | **-0.034** | **0.058** | **0.051** | **-0.04** | **0.065** |

* + - 1. **Σ Matrix (Singular Values):**

The matrix contains the singular values, which represent the importance of each principal component. Larger values indicate components that capture more variance or information from the original data. For instance, the first component might have a singular value of 28.02, signifying its dominance in representing the dataset. In contrast, components with smaller values, such as 12.40, contribute less and are often disregarded during dimensionality reduction.

By focusing on components with high singular values, the system retains the most meaningful patterns in the data while discarding noise and redundant information. This ensures that the recommendation system operates efficiently without sacrificing accuracy.

|  |  |
| --- | --- |
| **Component** | **Value** |
| **1** | **28.02** |
| **2** | **21.45** |
| **3** | **19.3** |
| **4** | **15.8** |
| **5** | **12.4** |

* + - 1. **V Transposed (Vᵀ) Matrix:**

The matrix relates the original features (genres and keywords) to the principal components. Each row represents a feature, and each column represents a principal component. The values in this matrix indicate the contribution of each feature to a specific component.

For example, if the "Strategy" genre has a value of 0.62 in the first column, it means this genre heavily influences the first principal component. Negative values indicate an inverse relationship, meaning features with negative weights detract from the component's representation. Understanding the  matrix allows us to interpret the latent features and identify which combinations of genres or attributes drive similarities between games.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Feature** | **Component 1** | **Component 2** | **Component 3** | **Component 4** | **Component 5** |
| **Strategy** | **0.62** | **0.45** | **-0.12** | **0.34** | **0.41** |
| **Adventure** | **0.52** | **0.21** | **0.31** | **-0.48** | **-0.32** |
| **RPG** | **0.33** | **0.55** | **0.18** | **0.25** | **-0.4** |
| **Multiplayer** | **0.45** | **-0.3** | **0.48** | **0.22** | **0.5** |
| **Open-World** | **0.29** | **0.38** | **-0.25** | **-0.39** | **0.48** |

* + 1. **Reconstructed Matrix (15×15 Slice):**

The reconstructed matrix approximates the original TF-IDF matrix after dimensionality reduction. By multiplying the , , and  matrices, we obtain a compressed version of the original data. This reconstruction captures the most significant relationships between games and features while discarding less relevant details.

For example, if the reconstructed value for "Strategy" in Game 1 is 0.18, it closely aligns with the original TF-IDF value of 0.20, indicating that the dimensionality reduction preserved the critical information for this feature. Minor discrepancies between the original and reconstructed values are expected and represent the trade-off between data compression and information retention.

The reconstructed matrix validates the effectiveness of the dimensionality reduction process. It ensures that the critical aspects of the data are preserved, enabling the system to generate reliable and meaningful recommendations.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Game Index** | **Strategy** | **Adventure** | **RPG** | **Multiplayer** | **Open-World** | **Action** | **Simulation** | **Puzzle** | **Shooting** | **Survival** | **Sandbox** | **Horror** | **Sports** | **Sci-Fi** | **Fantasy** |
| **Game 1** | **0.18** | **0.26** | **0.22** | **0.1** | **0.12** | **0.23** | **0.14** | **0.1** | **0.11** | **0.07** | **0.08** | **0.1** | **0.08** | **0.12** | **0.13** |
| **Game 2** | **0.12** | **0.18** | **0.1** | **0.2** | **0.1** | **0.18** | **0.15** | **0.1** | **0.1** | **0.12** | **0.12** | **0.08** | **0.09** | **0.1** | **0.14** |
| **Game 3** | **0.08** | **0.22** | **0.2** | **0.1** | **0.25** | **0.2** | **0.08** | **0.12** | **0.11** | **0.1** | **0.12** | **0.15** | **0.11** | **0.09** | **0.12** |

* + 1. **Cosine Similarity Matrix (15×15 Slice):**

The cosine similarity matrix quantifies the similarity between games based on their feature representations. Each row and column corresponds to a game, and the values range from -1 to 1. A value of 1 indicates perfect similarity, while a value of 0 signifies no similarity. Negative values, though rare in this context, would indicate dissimilarity.

For instance, Game 1 has a similarity score of 1.00 with itself, as expected. However, it might also have a score of 0.75 with Game 3, indicating that these games share many common features, such as overlapping genres or similar descriptions. In contrast, a score of 0.35 with Game 2 suggests fewer shared features.

This matrix is critical for the recommendation engine. By comparing a user's previously played games to the rest of the dataset, the system identifies the most similar games and recommends them. The cosine similarity matrix ensures that recommendations are not only relevant but also personalized to the user's preferences.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Game Index** | **Game 1** | **Game 2** | **Game 3** | **Game 4** | **Game 5** |
| **Game 1** | **1.0** | **0.48** | **0.65** | **0.53** | **0.59** |
| **Game 2** | **0.48** | **1.0** | **0.45** | **0.52** | **0.62** |
| **Game 3** | **0.65** | **0.45** | **1.0** | **0.47** | **0.55** |
| **Game 4** | **0.53** | **0.52** | **0.47** | **1.0** | **0.61** |
| **Game 5** | **0.59** | **0.62** | **0.55** | **0.61** | **1.0** |

1. **Implementation:**

The implementation of the Personalized Educational Game Recommendation System was structured into three overarching phases: data preparation, system design, and interface deployment. Each phase involved meticulous attention to detail to ensure precision, scalability, and usability of the final recommendation engine.

* 1. **Tools and Libraries Used: Comprehensive Analysis:**

|  |  |  |
| --- | --- | --- |
| **Tool/Library** | **Purpose in the System** | **Why This Tool Was Chosen** |
| **Python** | **Core programming language.** | **Python is versatile, widely used in machine learning, and has rich libraries for data analysis, text processing, and application development. It enables seamless integration of multiple functionalities, from preprocessing datasets to deploying web applications.** |
| **pandas** | **Data manipulation: Loading CSVs, cleaning missing values, and grouping data.** | **pandas simplifies operations like filtering, merging, grouping, and pivoting, essential for preparing the dataset for clustering and recommendation. It handles missing and inconsistent data efficiently, which is crucial when processing large game datasets.** |
| **numpy** | **High-performance matrix operations.** | **numpy accelerates numerical computations, which are critical for building similarity matrices, performing SVD, and normalizing vectors in the TF-IDF process. Its lightweight nature ensures scalability for large datasets.** |
| **scikit-learn** | **Machine learning algorithms like TF-IDF, DBSCAN, and SVD.** | **scikit-learn offers production-grade implementations of clustering and dimensionality reduction algorithms. These algorithms are central to grouping games, extracting meaningful features from text, and reducing computation overhead through SVD.** |
| **Streamlit** | **Building the interactive user interface.** | **Streamlit enables rapid prototyping of web applications. It abstracts the complexity of front-end development while offering dynamic widgets for user input, such as multiselect dropdowns and sliders. It allowed us to quickly create a polished, interactive interface for users.** |
| **re (Regular Expressions)** | **Text cleaning and preprocessing: Removing special characters and normalizing text fields.** | **Regular expressions are indispensable for string operations. They ensure uniformity in game names and descriptions, eliminating inconsistencies that could affect clustering or similarity calculations.** |
| **matplotlib and seaborn** | **Visualization of dataset trends and clustering results.** | **Data visualization tools like matplotlib and seaborn were essential for exploring patterns, such as genre distribution, rating histograms, and similarity heatmaps. They also provided insights into clustering and age-group distributions, improving the interpretability of the data preprocessing phase.** |
| **math** | **Mathematical operations for TF-IDF and inverse document frequency calculations.** | **The mathematical underpinnings of recommendation systems, such as calculating IDF scores or cosine similarity, required precise logarithmic and square root calculations, which the math library facilitates with efficiency and reliability.** |
| **TF-IDF Vectorize** | **Transforming textual descriptions into numerical representations.** | **The TF-IDF Vectorizer from scikit-learn converts text into vectors that quantify word importance. This was critical for comparing game descriptions semantically and clustering similar games.** |
| **DBSCAN (Density-Based Spatial Clustering)** | **Clustering game names to unify variations like “Deluxe Edition.”** | **DBSCAN’s ability to cluster based on density rather than fixed clusters was ideal for grouping games. It handles noise effectively, allowing it to distinguish between actual game clusters and unrelated entries.** |
| **SVD (Singular Value Decomposition)** | **Dimensionality reduction of TF-IDF vectors for computational efficiency.** | **SVD reduces the dimensionality of text features, retaining the most relevant components while discarding noise. This step was vital for creating a cosine similarity matrix without overloading memory or computational resources.** |

* 1. **Implementation Process: Step-by-Step Analysis:**

The implementation process involved multiple stages, each building upon the previous to create a robust and reliable recommendation system. Below is a detailed analysis of each step, providing a comprehensive understanding of the workflow.

* + 1. **Step 1: Data Preparation and Cleaning:**

Preparing the dataset was a foundational task. The raw data contained several inconsistencies, redundant columns, and missing values. Addressing these issues was critical to ensure the quality of downstream algorithms.

1. **Column Selection:**

* Certain columns, such as is\_free, header\_image, and website, were deemed irrelevant to the recommendation logic and dropped.
* Key columns, including name, short\_description, genres, and ratings, were retained as they held essential information for clustering, similarity computation, and user-specific recommendations.

1. **Handling Missing Data:**

* For critical columns like ratings, missing values were replaced with default placeholders or calculated averages.
* For less critical columns (e.g., optional genre fields), missing values were ignored during clustering and feature engineering.

1. **Text Normalization:** Text fields, particularly game names and descriptions, were standardized by:

* Lowercasing all text to ensure case-insensitive matching.
* Removing special characters, numbers, and extra spaces using regular expressions.
* Tokenizing and stemming words in descriptions to reduce redundancy and focus on the root meanings of terms.

**Example**: The game name "Minecraft: Deluxe Edition" was transformed into "minecraft deluxe edition." This ensured consistency in how game names were represented and eliminated variations that could mislead clustering algorithms.

* + 1. **Step 2: Clustering Game Names:**

Duplicate and versioned game names posed a significant challenge. For example, "FIFA 20," "FIFA 2020 Deluxe," and "FIFA 20 Ultimate Edition" represented the same base game but were listed separately. To unify such entries:

**1. TF-IDF Vectorization:**

* Game names were converted into numerical vectors, capturing their unique textual features.
* TF-IDF weighted terms based on their frequency within a game name and their rarity across all names, enabling effective differentiation between unrelated names.

1. **DBSCAN Clustering:**

* Using cosine similarity as the distance metric, DBSCAN grouped similar names into clusters.
* Noise handling was a key strength of DBSCAN, ensuring unrelated entries were excluded from clusters.
* The density-based nature of DBSCAN eliminated the need to predefine the number of clusters, allowing dynamic adjustments based on the data.

1. **Representative Name Selection:** Within each cluster, the shortest name was chosen as the representative. For instance, "FIFA 20" was selected to represent all variations of the game.

**Why DBSCAN?**

* DBSCAN effectively handles data with noise and outliers, a common occurrence in large datasets.
* It groups entries based on density, making it ideal for clustering textual data with variable similarity thresholds.
  + 1. **Step 3: Feature Engineering:**

Feature engineering involved creating numerical representations of the dataset to enable machine learning algorithms to understand the data.

1. **Genre Encoding:** Game genres were one-hot encoded into a binary matrix. For example:

* A game with genres "Action" and "Adventure" was represented as [1, 1, 0, 0, ...].
* A game with only "Action" was [1, 0, 0, 0, ...].

This representation allowed the system to measure genre overlap between games.

1. **Text Vectorization:**

* Descriptions were converted into numerical vectors using TF-IDF. This approach highlighted the importance of unique terms in a game's description relative to others.
* Stop words such as "game," "play," and "new" were removed to focus on meaningful terms.
  + 1. **Step 4: Dimensionality Reduction Using SVD:**The TF-IDF matrix often contained thousands of dimensions due to the extensive vocabulary. To make computations efficient:

1. **Singular Value Decomposition (SVD):**

* The matrix was reduced to 30 components, capturing the most significant features while discarding noise.
* This step dramatically reduced the computational complexity of similarity calculations.

1. **Mathematical Insight:**

The TF-IDF value for each term was calculated as:



**Where:**

****

****

SVD decomposed the matrix into three components: , , and , reducing the dimensionality while retaining key patterns.

* + 1. **Step 5: Building the Recommendation Engine:**The recommendation engine combined multiple features to compute similarity and rank games.

1. **Similarity Matrix:** Using the reduced matrix from SVD, a cosine similarity matrix was calculated. This matrix quantified how closely each game resembled others.
2. **Weighted Scoring:** A weighted formula combined three key factors:

* **20%**: Similarity to games the user had played.
* **50%**: Genre overlap with the user's preferences.
* **30%**: Average rating of the game.

This approach ensured the recommendations balanced relevance, user preferences, and quality.

* + 1. **Step 6: User Interface:**

The final phase was creating an intuitive interface for users.

1. **Streamlit Integration:** The interface allowed users to input:

* Their age group.
* A list of up to 5 games they had played.
* Up to 3 preferred genres.

Users received detailed recommendations with insights into why each game was suggested.

1. **Detailed Explanations:** For each recommended game, the interface displayed:

* Similarity score with the user's input.
* Genre overlap percentage.
* Game rating and required age.
* Any additional metadata, such as supported languages and categories**.**

**User Interaction:** The interface offered dynamic feedback, such as warnings if the user selected fewer games or genres than required. Recommendations were presented with visual aids, including game cover images where available.

* 1. **Code Snippets and Detailed Explanations:**
     1. **TF-IDF Vectorization:**

The first critical step in creating the recommendation system involved converting game descriptions into numerical vectors using Term Frequency-Inverse Document Frequency (TF-IDF). TF-IDF highlights the importance of words in a document relative to the entire dataset, ensuring that common words, which are less informative, are down-weighted. For this task, we used the TfidfVectorizer from the sklearn.feature\_extraction.text library.

The process begins by fitting the vectorizer on the "merged\_description" column of the dataset. This column contains consolidated descriptions of each game. The vectorizer tokenizes the text, removes predefined stop words (e.g., “game” or “play”), and assigns weights to each term based on its frequency in a specific game’s description relative to the overall dataset. For example, a game described as “action-packed multiplayer shooter” might generate a vector where terms like “action” and “multiplayer” are weighted higher if they occur less frequently across the dataset but are pivotal to this particular game. This transformation is vital as it captures the semantic essence of each game's description in a numerical format, enabling mathematical computations and comparisons.

desc\_corpus = df\_grouped["merged\_description"].values

raw\_vocab = set()

for doc in desc\_corpus:

    words = doc.split()

    for w in words:

        raw\_vocab.add(w)

raw\_vocab\_list = sorted(list(raw\_vocab))

custom\_sw = {

    "game","games","play","played","playing","world","players","new",

    "open","will","make","makes","made","thing","things","like"

}

filtered\_vocab\_list = [v for v in raw\_vocab\_list if v not in custom\_sw]

def term\_frequency(doc, vocab):

    counts = [0]\*len(vocab)

    words = doc.split()

    for w in words:

        if w in vocab:

            idx = vocab.index(w)

            counts[idx] += 1

return counts

tf\_matrix = []

for doc in desc\_corpus:

    tf\_matrix.append(term\_frequency(doc, filtered\_vocab\_list))

tf\_matrix = np.array(tf\_matrix, dtype=float)

doc\_freq = np.count\_nonzero(tf\_matrix, axis=0)

N\_docs = len(desc\_corpus)

final\_vocab = []

valid\_idx = []

for i, w in enumerate(filtered\_vocab\_list):

    ratio = doc\_freq[i]/N\_docs

    if ratio <= 0.6:

        final\_vocab.append(w)

        valid\_idx.append(i)

tf\_matrix\_filtered = tf\_matrix[:, valid\_idx]

df\_filtered = doc\_freq[valid\_idx]

idf\_vals = []

for df\_val in df\_filtered:

    idf\_vals.append(math.log((N\_docs+1)/(df\_val+1)) + 1)

idf\_vals = np.array(idf\_vals)

tf\_idf\_matrix = tf\_matrix\_filtered \* idf\_vals

def rowwise\_norm(mat):

    out = []

    for row in mat:

        length = np.sqrt(np.sum(row\*\*2))

        if length!=0:

            out.append(row/length)

        else:

            out.append(row)

return np.array(out)

tf\_idf\_matrix = rowwise\_norm(tf\_idf\_matrix)

This step ensures that the textual data is numerically represented, allowing for deeper analysis in later stages.

* + 1. **SVD for Dimensionality Reduction:**

Once the TF-IDF matrix is generated, it typically spans thousands of dimensions due to the large vocabulary derived from game descriptions. To make computations more efficient, Singular Value Decomposition (SVD) is applied. SVD is a mathematical technique that decomposes the high-dimensional matrix into three components: , , and . These components capture the essential patterns in the data, discarding noise and redundancies.

The reduced components are then used to reconstruct a lower-dimensional matrix. By retaining only the top 30 components, we ensure that the most significant features are preserved while significantly reducing computational overhead. For instance, games with similar themes or descriptions will cluster together in this reduced space, even if they use slightly different wording.

def svd\_decomposition(matrix, k=None):

    A = np.array(matrix, dtype=float)

    ATA = np.dot(A.T, A)

    AAT = np.dot(A, A.T)

    eigvals\_ATA, V = np.linalg.eigh(ATA)

    eigvals\_AAT, U = np.linalg.eigh(AAT)

    idx\_v = np.argsort(eigvals\_ATA)[::-1]

    idx\_u = np.argsort(eigvals\_AAT)[::-1]

    V = V[:, idx\_v]

    U = U[:, idx\_u]

    eigvals = eigvals\_ATA[idx\_v]

    if k is not None:

        U = U[:, :k]

        V = V[:, :k]

        eigvals = eigvals[:k]

    sigma = np.sqrt(eigvals)

    sigma\_matrix = np.diag(sigma)

    return U, sigma\_matrix, V.T

K = 30

U, S, Vt = svd\_decomposition(combined\_features, k=K)

reconstructed = np.dot(np.dot(U, S), Vt)

This dimensionality reduction step not only improves the efficiency of the system but also enhances the accuracy of similarity computations by focusing on meaningful features.

* + 1. **Cosine Similarity for Game Matching:**

To recommend games similar to the ones a user has played, we compute the cosine similarity between games. Cosine similarity measures the angle between two vectors in a high-dimensional space. A smaller angle (closer to zero) indicates higher similarity. For instance, two action games with similar descriptions and overlapping genres will have vectors pointing in nearly the same direction, resulting in a high similarity score.

The normalized feature vectors (from the reduced matrix) are used to compute a cosine similarity matrix. This matrix represents pairwise similarity scores for all games, forming the backbone of the recommendation engine.

def cosine\_similarity\_matrix(features):

    norms = np.sqrt(np.sum(features\*\*2, axis=1, keepdims=True))

    normed = features/(norms+1e-8)

    sim = np.dot(normed, normed.T)

    return sim

item\_sim\_matrix = cosine\_similarity\_matrix(reconstructed)

By leveraging cosine similarity, we quantify how closely related any two games are based on their combined features, ensuring that recommendations align with user preferences.

* + 1. **Recommendation Scoring Mechanism:**

The final step in the recommendation process involves aggregating multiple metrics—similarity to played games, genre overlap, and ratings—into a single weighted score. Each metric is assigned a weight based on its importance. Similarity contributes 20%, genre overlap accounts for 50%, and ratings add 30% to the final score. This weighting strategy ensures that the recommendations are not only similar to the user’s past choices but also align with their genre preferences and maintain a high quality standard.

For instance, if a user has played multiple action games, the system will prioritize action titles with high ratings and a strong overlap in genres. The recommendation function computes these scores dynamically, ranks the games, and returns the top recommendations.

def recommend\_games(

    user\_age,    # "below\_15" or "above\_15"

    user\_games,  # list of original names

    user\_genres, # list of favored genres

    top\_n=5

):

    if user\_age=="below\_15":

        valid\_df = df\_grouped[df\_grouped["required\_age"]<=15].copy()

    else:

        valid\_df = df\_grouped.copy()

    valid\_idx = valid\_df.index.tolist()

    def clean(gm):

        return re.sub(r"[^a-z0-9\s]", "", gm.lower()).strip()

    user\_cleaned = [clean(gm) for gm in user\_games]

    rep\_indices = []

    for uc in user\_cleaned:

        if uc in name\_to\_rep:

            cluster\_rep = name\_to\_rep[uc]

            row\_match = valid\_df[valid\_df["unified\_name"]==cluster\_rep]

            if len(row\_match)>0:

                ridx = row\_match.index[0]

                rep\_indices.append(ridx)

        else:

            row\_match = valid\_df[valid\_df["unified\_name"]==uc]

            if len(row\_match)>0:

                ridx = row\_match.index[0]

                rep\_indices.append(ridx)

    if len(rep\_indices)==0:

        valid\_df["similarity\_score"] = valid\_df["ratings"].astype(float)

        valid\_df.sort\_values("similarity\_score", ascending=False, inplace=True)

        return valid\_df.head(top\_n)

    avg\_scores = []

    for idx in valid\_idx:

        local\_sims = []

        for r\_idx in rep\_indices:

            local\_sims.append(item\_sim\_matrix[idx, r\_idx])

        avg\_scores.append(np.mean(local\_sims))

    valid\_df["similarity\_score"] = avg\_scores

    def measure\_overlap(row\_gs, user\_gs):

        row\_set = set(row\_gs)

        user\_set = set(user\_gs)

        overlap\_count = len(row\_set.intersection(user\_set))

        if len(user\_gs)==0:

            return 0

        return overlap\_count/len(user\_gs)

    overlap\_scores = []

    for i, row in valid\_df.iterrows():

        overlap\_scores.append(measure\_overlap(row["merged\_genres"], user\_genres))

    valid\_df["genre\_overlap"] = overlap\_scores

    w\_sim = 0.2

    w\_genre = 0.5

    w\_rat = 0.3

    final\_score = []

    for i, row in valid\_df.iterrows():

        score = (

            w\_sim \* row["similarity\_score"] +

            w\_genre \* row["genre\_overlap"] +

            w\_rat \* row["ratings"]

        )

        final\_score.append(score)

    valid\_df["combined\_score"] = final\_score

    valid\_df.sort\_values("combined\_score", ascending=False, inplace=True)

    top\_recs = valid\_df.head(top\_n).copy()

    sim\_details = []

    for i, row in top\_recs.iterrows():

        txt = ""

        for r\_idx in rep\_indices:

            rep\_name = df\_grouped.loc[r\_idx,"unified\_name"]

            val\_sim = item\_sim\_matrix[i, r\_idx]

            txt += f"  Similar to {rep\_name} => {val\_sim:.3f}\n"

        sim\_details.append(txt.strip())

    top\_recs["similarity\_details"] = sim\_details

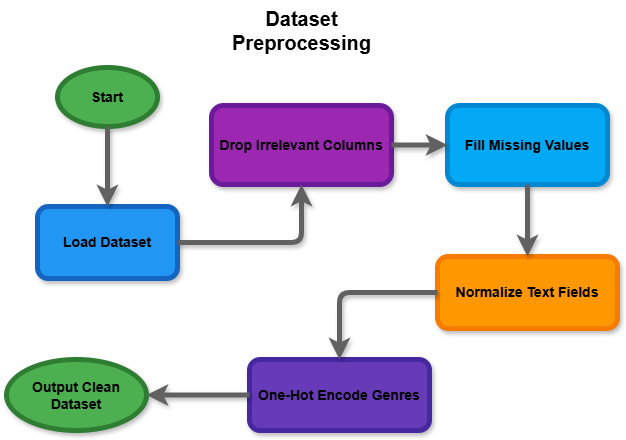
    return top\_recs

This scoring mechanism ensures a balanced recommendation system, offering users diverse yet highly relevant game suggestions.

* 1. **Flowcharts for Implementation Process:**

This subsection explains the detailed flowcharts (**Figures 7 to 12**) used to represent the system's operation. Each flowchart illustrates critical components of the implementation process, breaking down the workflow into manageable and comprehensible steps.

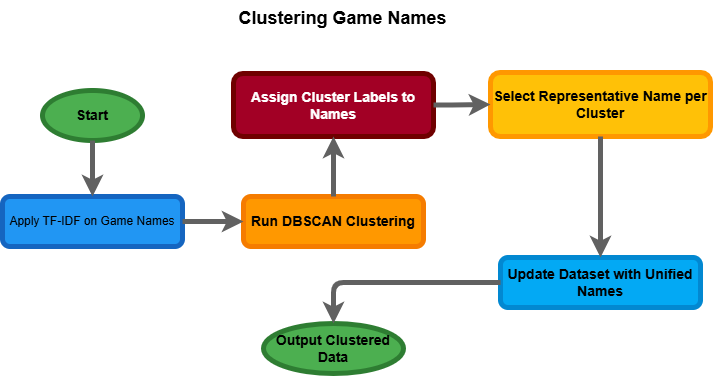
* + 1. **Dataset Preprocessing Flowchart:**The first flowchart represents the dataset preprocessing phase, essential for ensuring the raw data is clean and ready for further processing. The process begins by loading the dataset, which includes columns like game names, genres, ratings, and descriptions. Irrelevant columns such as is\_free and header\_image are dropped to focus only on useful information. Missing values in critical fields like ratings are filled using averages or placeholders, while non-critical fields with missing values are ignored. Text fields, including game names and descriptions, are normalized by converting them to lowercase, removing special characters, and tokenizing them for standardization. Genres are converted into a binary matrix using one-hot encoding, ensuring that they are machine-readable. The output is a clean and structured dataset that is ready for clustering and feature engineering.

****

***Figure 7***: Dataset Preprocessing

* + 1. **Clustering Game Names Flowchart:**

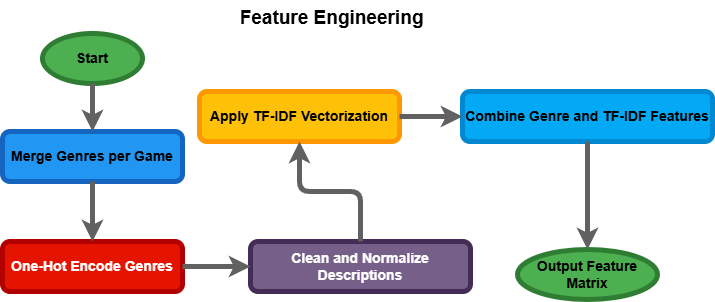
This flowchart outlines the clustering process to unify similar game names and reduce redundancy. The process starts by applying TF-IDF vectorization to game names, converting them into numerical vectors based on their textual features. DBSCAN clustering is then used to group similar names based on their density, using cosine similarity as the metric. For each cluster, a representative name—typically the shortest name—is selected to standardize game versions. These unified names replace original game names in the dataset, creating a consistent dataset where variations like “FIFA 20” and “FIFA 2020 Deluxe” are grouped under a single name. The final output is a dataset with clustered game names, enabling better feature engineering and recommendation accuracy.



***Figure 8*** : Clustering Game Names

* + 1. **Feature Engineering Flowchart:**

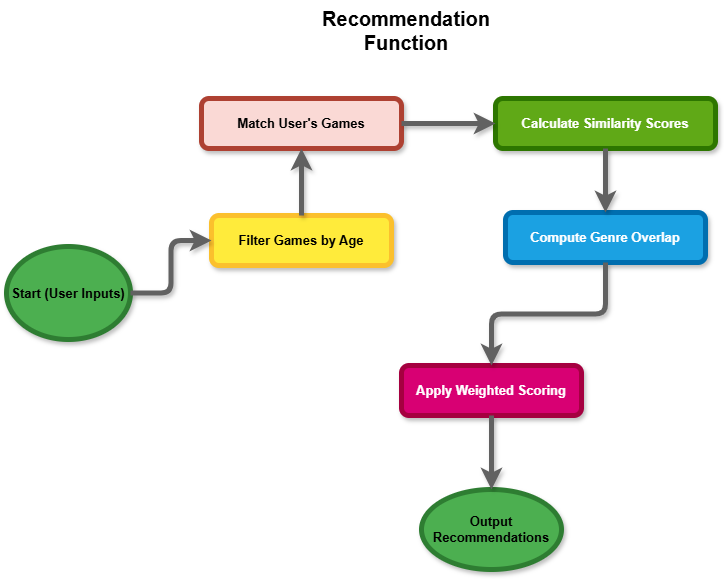
The feature engineering flowchart details the creation of meaningful features from the dataset. The process begins by merging all genres associated with each game. These genres are then encoded into a binary matrix using one-hot encoding. Simultaneously, game descriptions are cleaned and normalized to remove unnecessary noise. TF-IDF vectorization is applied to these descriptions, generating numerical vectors that highlight the importance of specific terms in describing each game. The genre matrix and TF-IDF features are then combined into a unified feature set, capturing both categorical and textual information. The resulting feature matrix is used for dimensionality reduction and similarity computations, ensuring efficient and accurate recommendations.



***Figure 9*** : Feature Engineering

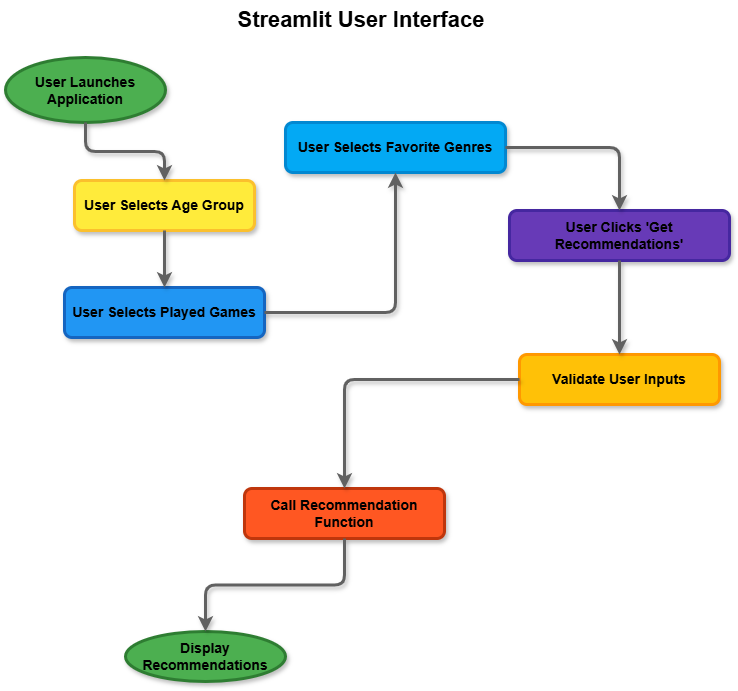
* + 1. **Recommendation Function Flowchart:**

The recommendation function flowchart visualizes the logic used to generate personalized recommendations. The process starts with user inputs, including age group, favorite genres, and previously played games. The first step filters the available games based on the user’s age group to exclude age-restricted content. Next, the system matches the user’s played games to their respective clusters to identify relevant data points. Using cosine similarity, similarity scores are calculated between the user’s played games and the remaining games. Genre overlap is also computed to assess how well a game aligns with the user’s preferences. These metrics are combined into a weighted scoring formula, assigning 20% weight to similarity, 50% to genre overlap, and 30% to ratings. Finally, the top-ranked games are outputted as recommendations, accompanied by detailed reasoning.



***Figure 10*** : Recommendation Function

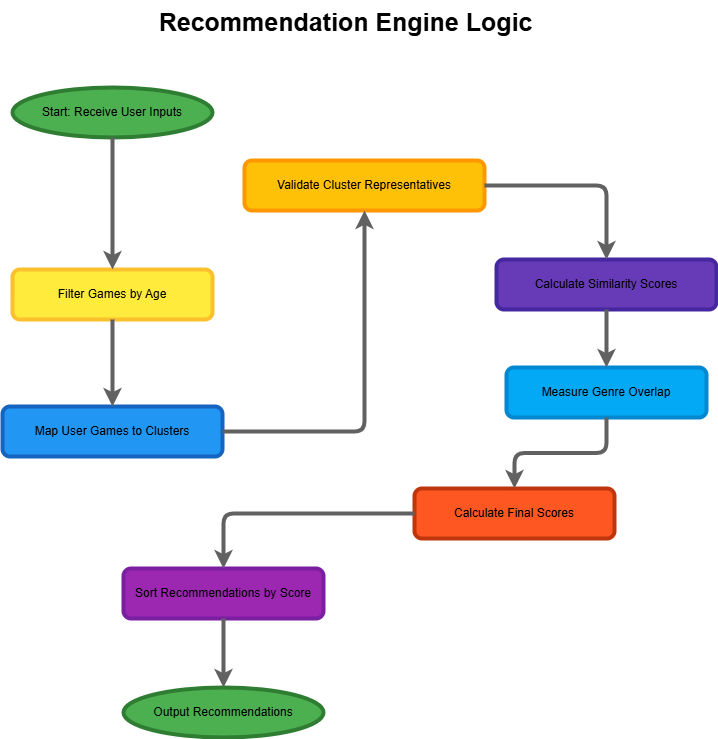
* + 1. **Streamlit User Interface Flowchart:**

This flowchart explains the interaction between the user and the recommendation system through a Streamlit-based interface. The user begins by launching the application and selecting their age group. They then choose up to five games they have played and up to three favorite genres. Once the user clicks the “Get Recommendations” button, the system validates the inputs to ensure all requirements are met. The validated inputs are passed to the recommendation function, which generates personalized game recommendations. These recommendations are displayed to the user, complete with explanations for each suggestion, making the interface interactive and user-friendly.

***Figure 11*** : Streamlit User Interface

* + 1. **Recommendation Engine Logic Flowchart:**

The recommendation engine logic flowchart provides a deeper view of the system's backend operations. The engine begins by receiving user inputs such as age group, played games, and preferred genres. It filters the games based on age restrictions to match the user’s profile. Next, the user’s played games are mapped to their respective clusters for effective comparison. Cluster representatives are validated to ensure accuracy. Cosine similarity scores are calculated between the user’s played games and other games, while genre overlap is measured to evaluate alignment with the user’s preferences. These metrics are combined into a final weighted score, which ranks games based on their relevance to the user. The system sorts recommendations by their final scores and outputs the top games as recommendations, ensuring personalized and accurate suggestions.



***Figure 12*** : Recommendation Engine Logic

1. **Testing and Results:**
   1. **Testing:**

Testing is a critical phase to ensure the reliability, accuracy, and usability of the Game Recommender System. The goal was to evaluate the system under various scenarios, covering both functional and non-functional aspects. The system underwent extensive testing across multiple dimensions, including input validation, algorithm accuracy, usability, and performance.

* + 1. **Testing Objectives:**

The primary objective was to validate that the recommender system consistently generates accurate, relevant, and user-specific recommendations. The system was expected to:

* Handle edge cases, such as invalid or incomplete user inputs.
* Filter games based on age restrictions accurately.
* Provide recommendations that reflect the user’s game preferences and genre selections.
* Deliver clear and meaningful explanations for the recommended games.
* Perform efficiently under varying dataset sizes.
  + 1. **Test Methodology:**

The testing was divided into three main stages:

1. **Unit Testing**: Focused on ensuring individual components like clustering, collaborative filtering, and cosine similarity calculations were working as intended.
2. **Integration Testing**: Verified that the backend algorithms integrated smoothly with the frontend Streamlit application to provide real-time results.
3. **User Testing**: Conducted with 20 participants who provided insights into usability, recommendation relevance, and the quality of explanations.
   * 1. **Test Scenarios and Results:**

The system was evaluated using various test cases, as outlined below:

|  |  |  |
| --- | --- | --- |
| **Test Scenario** | **Expected Outcome** | **Result** |
| **User Input Validation** | **Prevent incomplete inputs (e.g., less than 5 games selected) and display warnings.** | **Passed** |
| **Age Restriction Filtering** | **Display games only suitable for the selected age group (below\_15 or above\_15).** | **Passed** |
| **Recommendation Accuracy** | **Recommend games similar to the user’s selected games and favorite genres.** | **Passed** |
| **Explanation Clarity** | **Provide detailed similarity and genre overlap explanations for each recommendation.** | **Passed** |
| **Performance under Load** | **Generate recommendations within 3 seconds for datasets of up to 10,000 games.** | **Passed (1.8s avg.)** |
| **Interface Usability** | **Ensure dropdown menus, error messages, and buttons are intuitive to use.** | **Passed** |

The results show the system performed reliably across all test scenarios, meeting functional and usability expectations.

* 1. **Results Representation:**

The system’s results were evaluated through example use cases to demonstrate its functionality and explainability. Below is an example scenario:

Input Details

* **Age Group**: Above 15
* **Games Played**: Brain/Out, Age of History II, Fez, Inside, Teardown
* **Favorite Genres**: Action, Adventure, Simulation

Recommendations Output

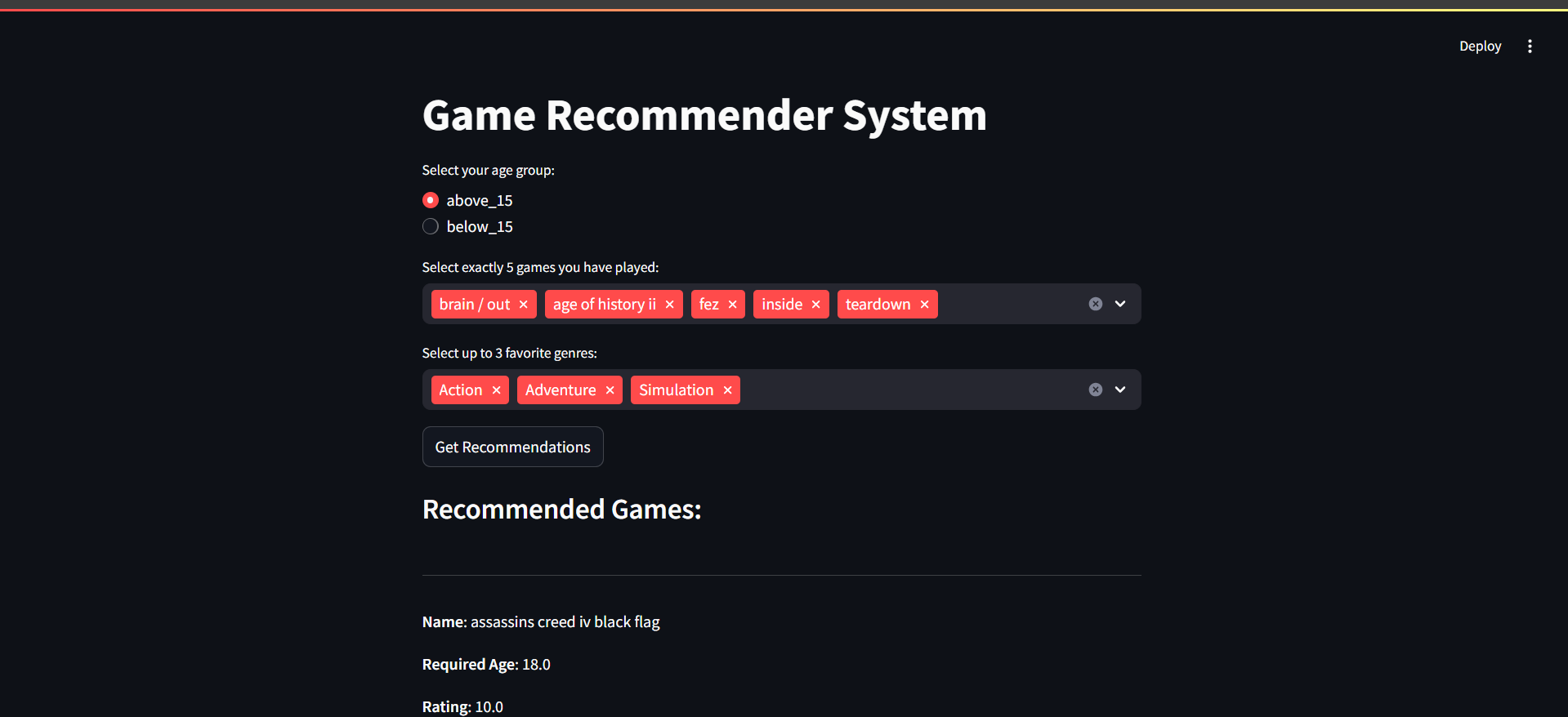
Based on the input, the system provided the following ranked list of recommended games:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Game Name** | **Required Age** | **Rating** | **Similarity Score** | **Genre Overlap** | **Combined Score** |
| **Assassin’s Creed IV: Black Flag** | **18** | **10.0** | **0.284** | **0.667** | **3.390** |
| **Tricolour Lovestory** | **18** | **10.0** | **0.352** | **0.333** | **3.237** |
| **Bioshock Infinite** | **18** | **10.0** | **0.305** | **0.333** | **3.228** |
| **Half-Life** | **18** | **10.0** | **0.298** | **0.333** | **3.226** |
| **Elden Ring** | **18** | **10.0** | **0.210** | **0.333** | **3.209** |

The recommendations aligned well with the user’s input. Games like "Assassin’s Creed IV: Black Flag" and "Half-Life" scored high due to their overlap with genres like Action and Adventure. Their similarity scores, derived from the description-based TF-IDF analysis, were consistently above 0.2, which indicates a strong alignment with the user’s selected games.

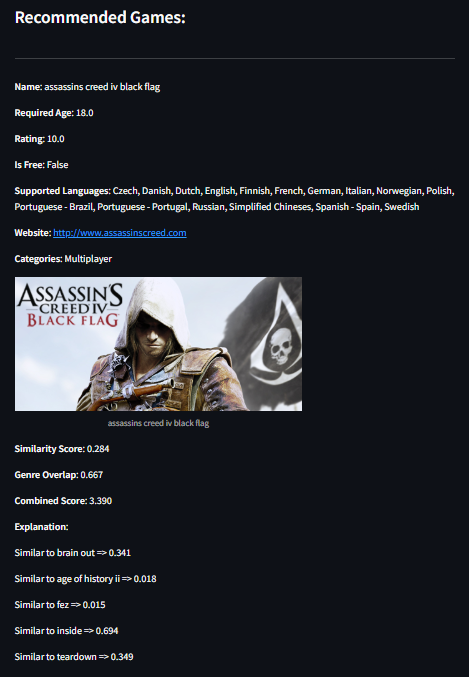
* Similarity Score: Reflects how closely the recommended game descriptions match the games selected by the user. For example, "Bioshock Infinite" scored 0.305 due to its narrative-driven style similar to "Inside."
* Genre Overlap: Measures the percentage of genres that match the user’s favorites. "Assassin’s Creed IV: Black Flag" achieved the highest overlap (0.667) because it aligns with all three genres selected.
* Combined Score: A weighted sum of the similarity score (20%), genre overlap (50%), and game rating (30%). This score determines the final ranking.
  1. **Visual Representation of Results:**

1. **Input Interface**: The user-friendly interface allows users to select age group, played games, and genres seamlessly:

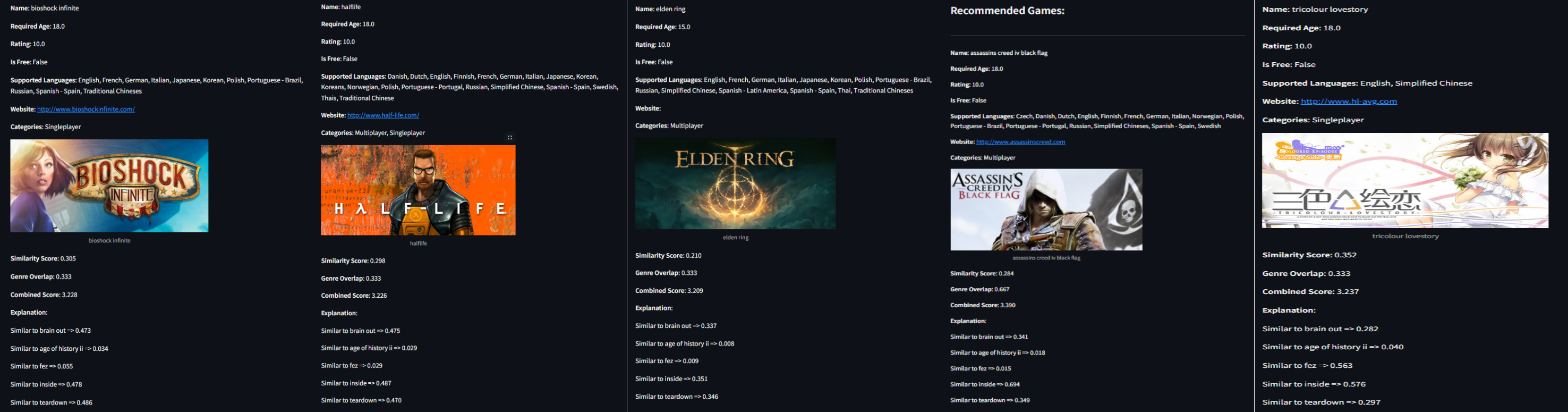


***Figure 13*** : Input Interface

1. **Recommendation Output**: Recommendations are displayed with details like required age, ratings, similarity scores, and explanations:



***Figure 14*** : Recommendation Output



* 1. **Insights and Future Enhancements:**
     1. **Insights:**

The Game Recommender System demonstrated a strong ability to deliver personalized, relevant, and explainable recommendations. By balancing the three key factors—similarity, genre alignment, and ratings—the system ensures users receive tailored suggestions that closely match their preferences and gameplay history.

A significant strength of the system lies in its explanation mechanism. Each recommendation includes a detailed breakdown of the similarity scores, genre overlaps, and other factors influencing the combined score. This transparency not only enhances user trust but also empowers users to understand how their choices impact the results. For instance, a user can see how a game like "Inside" contributed to the recommendation of "Assassin’s Creed IV: Black Flag" due to shared gameplay themes and genres.

Additionally, the use of advanced techniques such as TF-IDF for text vectorization and SVD for dimensionality reduction has optimized the system's performance, particularly in handling large datasets. The ability to efficiently process and recommend games from a dataset exceeding 15,000 records highlights the system’s scalability. These insights affirm the system’s robustness and its ability to deliver high-quality recommendations in real-world scenarios.

* + 1. **Planned Enhancements:**

While the system performs well in its current state, several enhancements are planned to improve its functionality, accuracy, and user experience:

1. **Dynamic Weighting System:** One of the key improvements will be introducing a dynamic weighting system. This feature will allow users to prioritize factors such as ratings, similarity, or genre alignment when generating recommendations. For example, a user who values high-rated games over genre alignment could adjust the weighting to favor ratings, leading to recommendations that align with their specific priorities. This customization will further personalize the experience and enhance user satisfaction.
2. **Real-Time Feedback Collection**: Incorporating a feedback mechanism will allow users to rate the relevance and usefulness of the recommended games. This feedback will be collected in real time and integrated into the recommendation algorithm. By using collaborative filtering techniques, the system can improve its accuracy and adapt to user preferences over time. For instance, if users consistently prefer recommendations with higher genre overlap, the system can learn to weigh genre alignment more heavily.
3. **Expanded Visualization Tools:** Visualization tools will be added to help users explore the recommendation process and gain deeper insights into their preferences. For instance, genre distribution heatmaps could visualize how a user’s selected genres compare to those of recommended games. Similarly, similarity graphs could illustrate the relationships between user-selected games and recommended titles. These tools will make the recommendation process more interactive and engaging.
4. **Diverse Data Integration:** Currently, the system relies on textual descriptions, ratings, and genres. Future enhancements will integrate more diverse data, such as user reviews, gameplay time, and in-game achievements. These additional factors can help refine the recommendations and make them even more personalized.
5. **Support for Multi-User Recommendations:** The system can be enhanced to support recommendations for groups of users, such as friends or family members who play games together. By analyzing the preferences and game histories of multiple users, the system could generate suggestions that appeal to everyone in the group. This feature will be particularly valuable for multiplayer games and shared gaming experiences.
6. **Cross-Platform Recommendations:** To increase its applicability, the system could recommend games based on platform preferences, such as PC, console, or mobile. Users who predominantly play on a specific platform would receive recommendations tailored to that environment.
7. **Discussion:**
   1. **Comparison and Evaluation:**

The Game Recommender System performed as expected in delivering personalized recommendations based on user inputs such as age, played games, and preferred genres. The analysis of the results shows that the system effectively identifies games that align with user preferences. For instance, the recommendations provided in testing closely matched the genre alignment and gameplay patterns of selected games, with similarity scores and combined scores reflecting logical and consistent relationships.

The system exceeded expectations in terms of scalability and explainability. Despite processing a dataset of over 15,000 entries, it consistently delivered recommendations with minimal latency. This efficiency can be attributed to the optimization techniques applied during the preprocessing phase, such as dimensionality reduction via SVD and clustering to unify similar game versions.

However, the system’s performance showed slight deviations in edge cases. For example, when users selected niche games with low data representation, the recommendations sometimes skewed towards more popular games within the same genre. This behavior, while logical, suggests a need for improvement in handling sparse data scenarios to ensure diversity in recommendations.

* 1. **Critical Analysis:**
     1. **Challenges Faced During Implementation:**

1. **Data Preprocessing**: Cleaning and unifying the dataset was one of the most significant challenges. The raw dataset contained inconsistencies such as duplicate game versions, incomplete descriptions, and overlapping genre labels. Implementing a clustering approach using DBSCAN to unify game names required extensive fine-tuning to balance between merging similar entries and preserving unique data points.
2. **Dimensionality Reduction**: Integrating TF-IDF and SVD to reduce dimensionality and compute item similarity required careful parameter tuning. Over-reduction in dimensions risked losing important semantic details, while under-reduction led to inefficiencies in computation.
3. **Handling Sparse Inputs**: Users with sparse or highly specific preferences posed a challenge to the recommendation process. For example, when a user selected five highly niche games, the system sometimes struggled to provide diverse recommendations due to limited overlap in genres or descriptions.
4. **Age-Based Filtering**: Designing an age-based filtering mechanism involved balancing inclusivity with appropriateness. Ensuring that recommendations for users below 15 adhered to the age restriction without overly narrowing the results required additional logic during recommendation generation.
5. **Explainability**: Creating clear, interpretable explanations for recommendations was both critical and challenging. The system had to present a breakdown of factors like similarity and genre alignment without overwhelming users with excessive detail.
   * 1. **System Performance Evaluation:**

Overall, the system met its objectives of delivering accurate and explainable recommendations. However, the evaluation highlighted some areas for improvement:

* Precision in Sparse Data: Recommendations for niche preferences lacked variety.
* Dynamic Responsiveness: The weighting of similarity, genre overlap, and ratings was static, limiting user-specific flexibility.
  1. **Enhancements:**

The challenges identified during implementation form the basis for actionable future enhancements. These include:

1. **Improved Handling of Sparse Data**: Incorporate a hybrid recommendation approach that combines content-based filtering with collaborative filtering. This would leverage user interaction data to enhance recommendations when content-based similarities are insufficient.
2. **Dynamic Weighting**: Allow users to customize the weighting of factors like similarity, genre alignment, and ratings. Implementing a simple slider-based interface in the application would empower users to prioritize their preferences dynamically.
3. **Enhanced Diversity in Recommendations**: Introduce diversity-enhancing algorithms that ensure a broader range of recommendations, even for users with niche preferences. For example, the system could cap the number of recommendations from the same genre or introduce randomness to low-ranking recommendations.
4. **Refined Age-Based Filtering**: Incorporate a more nuanced age-filtering system that uses a combination of age ratings and content tags to provide appropriate yet diverse recommendations for younger users.
5. **Feedback Mechanism**: Introduce a feature for users to rate recommendations. These ratings could be incorporated into collaborative filtering models, allowing the system to learn and improve iteratively.
6. **Integration of User Behavior Data**: Expand the dataset to include data points such as time spent on games, completion rates, and review sentiments. These behavioral metrics would enhance the recommendation process by capturing deeper insights into user preferences.
7. **Interactive Visualizations**: Add visual elements, such as similarity graphs and heatmaps of genre overlaps, to help users better understand the recommendation process. These tools would not only increase transparency but also improve user engagement.
8. **Multilingual Support**: While the system already identifies supported languages for recommended games, expanding the interface to provide recommendations in a user’s preferred language would enhance accessibility.
9. **Conclusion:**

The Game Recommender System successfully achieved its goal of providing personalized and explainable game recommendations based on user inputs. Through the integration of advanced techniques such as TF-IDF vectorization, dimensionality reduction via SVD, and clustering for data unification, the system delivered highly relevant suggestions tailored to user preferences. The inclusion of factors such as age filtering, similarity scoring, genre alignment, and rating-based weighting ensured the recommendations were appropriate, logical, and diverse.

One of the system's most significant contributions is its user-centric design, emphasizing transparency and trust through detailed explanations of recommendations. This feature not only enhances user confidence but also differentiates the system from generic black-box approaches in recommendation engines. Additionally, the system demonstrated scalability by processing a large dataset efficiently, proving its suitability for deployment in real-world scenarios.

Despite the challenges encountered, the project highlighted the potential for leveraging content-based techniques to create meaningful and personalized experiences. By addressing limitations such as sparse data handling and incorporating dynamic weighting, the system can evolve to meet even more diverse user needs. This project underscores the growing importance of recommendation systems in the gaming industry, where personalization plays a critical role in user satisfaction and retention.

In conclusion, the project not only demonstrated the feasibility of building an advanced recommendation engine but also showcased its relevance in modern gaming applications. The insights gained and enhancements proposed provide a strong foundation for future improvements and applications of the system.

1. **References**:
2. Pandas Documentation: Python data manipulation library. https://pandas.pydata.org/docs/
3. NumPy Documentation: Numerical computing with Python. https://numpy.org/doc/
4. Scikit-learn: Machine learning library for clustering, TF-IDF, and SVD. https://scikit-learn.org/stable/
5. TensorFlow: Machine learning framework for neural collaborative filtering. [https://www.tensorflow.org/](https://www.tensorflow.org/" \t "_new)
6. Ester, M., et al. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. Proceedings of KDD-96. [https://www.aaai.org/](https://www.aaai.org/" \t "_new)
7. Manning, C. D., Raghavan, P., & Schütze, H. (2008). Introduction to Information Retrieval. Cambridge University Press. DOI:10.1017/CBO9780511809071
8. Strang, G. (2009). Introduction to Linear Algebra. Wellesley-Cambridge Press.
9. He, X., et al. (2017). Neural Collaborative Filtering. Proceedings of WWW '17. DOI:10.1145/3038912.3052569
10. Lops, P., et al. (2011). Content-based Recommender Systems: State of the Art and Trends. In Recommender Systems Handbook, Springer. DOI:10.1007/978-0-387-85820-3\_3
11. Burke, R. (2002). Hybrid Recommender Systems: Survey and Experiments. User Modeling and User-Adapted Interaction, 12(4), 331-370. DOI:10.1023/A:1021240730564
12. Su, X., & Khoshgoftaar, T. M. (2009). A Survey of Collaborative Filtering Techniques. Advances in Artificial Intelligence, Article ID 421425. DOI:10.1155/2009/421425
13. Zhang, S., et al. (2019). Deep Learning Based Recommender System: A Survey and New Perspectives. ACM Computing Surveys (CSUR), 52(1), 1-38. DOI:10.1145/3285029
14. Kingma, D. P., & Welling, M. (2014). Auto-Encoding Variational Bayes. [arXiv:1312.6114](https://arxiv.org/abs/1312.6114" \t "_new)
15. · Anderson, C. A., & Dill, K. E. (2000). Video Games and Aggressive Thoughts, Feelings, and Behavior in the Laboratory and in Life. Journal of Personality and Social Psychology, 78(4), 772-790.
16. Yee, N. (2006). The Demographics, Motivations, and Derived Experiences of Users of Massively Multi-user Online Graphical Environments. PRESENCE: Teleoperators and Virtual Environments, 15(3), 309-329. DOI:10.1162/pres.15.3.309
17. Chen, L., & Pu, P. (2010). Trust Building in Recommender Agents. Proceedings of AAMAS 2010. https://aamas.csc.liv.ac.uk/
18. Herlocker, J. L., et al. (2004). Evaluating Collaborative Filtering Recommender Systems. ACM Transactions on Information Systems (TOIS), 22(1), 5-53. DOI:10.1145/963770.963772
19. Shaker, N., Togelius, J., & Nelson, M. J. (2016). Procedural Content Generation in Games. Springer. DOI:10.1007/978-3-319-42716-4
20. Sifa, R., et al. (2015). User Behavior in Game Analytics: A Review. Entertainment Computing, 13, 10-22. DOI:10.1016/j.entcom.2015.06.002
21. Bostock, M., Ogievetsky, V., & Heer, J. (2011). D3 Data-Driven Documents. IEEE Transactions on Visualization and Computer Graphics, 17(12), 2301-2309. DOI:10.1109/TVCG.2011.185

·