Videogame Recommendation System Using Steam Reviews

Gerardo Gómez Argüelles, Tirdod Behbehani, Oliver Tausendschön

March 26th, 2025

Introduction

In recent years, video game distribution platforms like Steam have experienced explosive growth, offering thousands of titles to millions of players worldwide. While this provides users with countless choices, it also poses a challenge: how to efficiently connect players with games that align with their interests. Traditional recommendation systems often rely on numeric ratings or simplistic preference indicators. However, user-generated textual reviews contain a wealth of nuanced information opinions, sentiments, and detailed feedback that can significantly enhance the recommendation process if analyzed properly.

Our project leverages text mining techniques to build a recommendation system that uses Steam reviews as the primary data source. The system takes as input textual reviews scraped from Steam for each game, accompanied by game metadata such as game title, genre, and wether the review is positive or negative. Additionally, it incorporates a set of manually written user reviews from a simulated user, capturing both positive and negative impressions of previously played games. Based on these inputs, the output is a ranked list of game recommendations, where each game is scored according to its similarity to the user's profile, which is derived from the positive reviews provided by the user.

Specifically, we collect and aggregate user reviews for each game, apply a comprehensive text preprocessing pipeline (tokenization, removal of stopwords, lemmatization, etc.), and transform these processed texts into numerical vectors via TF–IDF. By separately modeling positive and negative reviews, we aim to capture not just what users like about certain games, but also what they dislike, allowing for a more granular understanding of user preferences.

A key component of this approach is the creation of a "user profile," derived from the TF–IDF features in the user's own reviews. This profile enables us to calculate similarity scores between the user's preferences and the aggregated features of the games. The final recommendation step incorporates both positive and negative similarities, ensuring that games with characteristics similar to what the user enjoys are ranked higher, while those sharing disliked traits are penalized. By focusing on textual data rather than traditional numerical ratings alone, our system aims to deliver more tailored, context-rich game recommendations ultimately helping players discover titles that genuinely resonate with their tastes.

1 Related Work

Game recommendation systems have been extensively studied, with methods broadly falling into collaborative filtering (CF), content-based filtering (CBF), and hybrid approaches. Early work in collaborative filtering [Adomavicius and Tuzhilin, 2005] leverages user ratings to find similarities between users or items. While CF can uncover complex patterns from large communities, it typically suffers from the cold-start problem, especially for new or less popular games where rating data is sparse.

To mitigate such limitations, content-based filtering methods incorporate textual descriptions, tags, or metadata about games.[Lops et al., 2011] provide an overview of content-based recommender systems, emphasizing the importance of extracting discriminative features from text. These approaches excel at recommending items similar to those a user already enjoys, but they can be limited by the breadth and quality of the available textual content.

Recent advances have focused on textual analysis of user-generated reviews. [Ni et al., 2019] propose a

framework for justifying recommendations by extracting fine-grained aspects from reviews, thus providing interpretable insights into why an item is recommended.

Despite these advances, many existing approaches rely primarily on users' own historical data or high-level product metadata to generate recommendations. As a result, they often underutilize the wealth of textual feedback available from the broader user community. This limitation is particularly significant in the gaming domain, where detailed reviews frequently highlight both the strengths and shortcomings of a title.

In contrast, our work builds on content-based and hybrid approaches by explicitly separating positive (recommended) and negative (not recommended) reviews for each game, in this case Steam's metadata simplifies our work by clearly indicating whether a review is positive or negative. This helps to capture a richer representation of user sentiment across the entire community. Rather than treating user feedback as a monolithic block, we aggregate and analyze the distinct positive and negative characteristics of each game as expressed by a diverse set of players. This distinction provides a more granular way to align individual user preferences with specific game attributes, ultimately improving both the accuracy and explainability of our recommendations.

2 Dataset

Our dataset was built by scraping reviews from Steam for various games, resulting in a collection of 95 games and a total of 40,649 reviews. Each game is accompanied by its metadata, including the title, genre, and a binary indicator provided by Steam, which specifies whether the user recommends the game (a positive review) or not (a negative review). Notably, the dataset spans 15 different game genres, with 53 unique combinations, reflecting a diverse selection of titles.

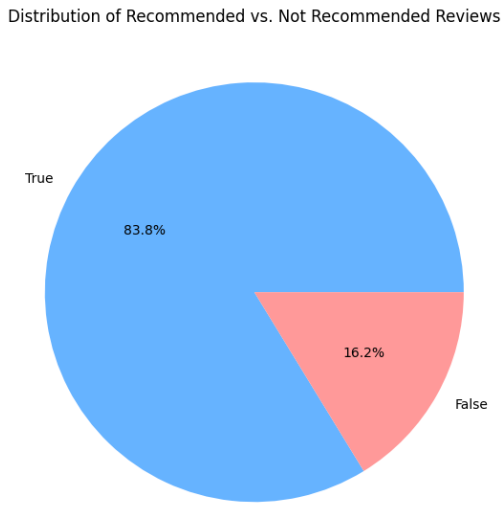

Figure 1: Distribution of Recommended vs. Not Recommended Reviews.

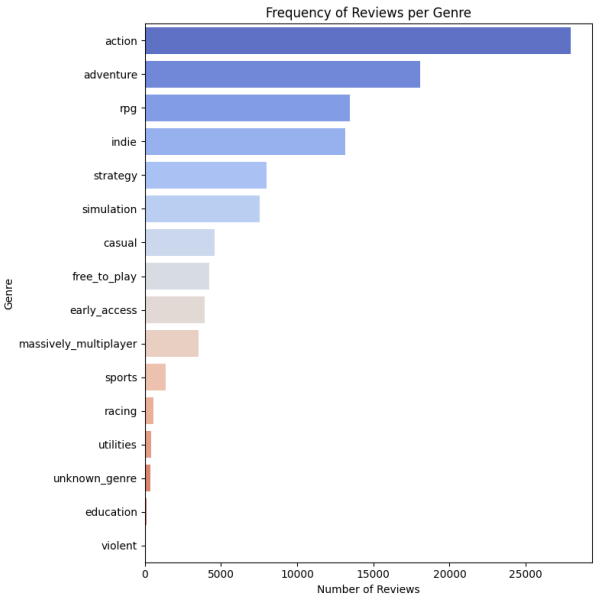

Figure 2: Frequency of Reviews per Genre.

A closer examination of the recommended variable reveals a significant imbalance: there are 34,045 positive reviews compared to only 6,604 negative reviews. As shown in Figure 1, this skew may initially raise concerns regarding data balance, yet it also mirrors a common phenomenon observed in user-generated content on platforms like Steam, where gamers tend to post positive feedback more frequently than negative. Such behavior may stem from the bias of satisfied customers being more inclined to leave a review, or perhaps from a general positivity bias in the gaming community.

For instance, consider a popular title with hundreds of positive reviews highlighting its engaging gameplay and stunning visuals, alongside only a handful of negative remarks centered around minor issues. Such examples underscore the practical importance of handling imbalanced data. Our recommendation system must carefully weigh these sentiments to accurately capture both what players love and what they might dislike about a game.

In addition, Figure 2 shows that *action* is the most prevalent genre in our dataset, followed by *adventure* and *rpg*. This genre distribution suggests that certain game types might dominate user attention, potentially influencing the performance of our recommendation system if not carefully accounted for.

3 Pipeline

3.1 Data Extraction

Since our work depends on user reviews, it's important to obtain accurate data for each review. We use Steam's API, which provides all the necessary information. This approach allows us to collect the data outlined in the Dataset Section above (Game Title, Reviews, Recommended, Genre).

3.2 Data Preprocessing

Our data preprocessing phase was designed to transform raw game reviews into a clean and uniform format that would enable effective TF-IDF analysis and reliable recommendations. Initially, our focus was on text cleaning using lemmatization with spaCy. This allowed us to reduce words to their base forms, consolidating similar words like "running" into "run", thereby reducing redundancy in our dataset. In addition, we applied a filter to keep only alphabetic words, effectively removing numbers and symbols that typically do not contribute meaningful sentiment or descriptive information.

During our preprocessing, we encountered a common challenge in the gaming community: the pervasive use of emojis and HTML-like tags. Although these elements are frequently used by gamers, they do not add useful information and are often overly repeated. To address this, we incorporated an extra regex function into our pipeline specifically to remove emojis and HTML-like tags. This adjustment ensured that our analysis would focus solely on the substantive textual content. Additionally, we later introduced a lowercasing step to standardize the text, ensuring that variations such as "Game" and "game" were treated as the same token.

After applying these preprocessing steps (lemmatization, alphabetic filtering, removal of emojis and HTML-like tags, and lowercasing) we created a new "cleaned review" column for each review. This column contained the processed text that was then ready for further analysis. In the subsequent steps, before performing TF-IDF, we aggregated all reviews for each game into a single document. Importantly, we maintained a separation based on sentiment by creating one aggregated document for positive (recommended) reviews and another for negative (not recommended) reviews. This dual-corpus, or equivalently dual-document approach per game, allowed us to capture the distinct characteristics of both positive and negative sentiments, ensuring that our TF-IDF vectors accurately reflected the aspects of a game that resonated well with users versus those that did not.

Furthermore, we observed that terms from the game titles themselves, such as *witcher*, *wild*, and *hunt* in *The Witcher Wild Hunt*, or *ark*, *survival*, and *ascended* in *ARK: Survival Ascended*, were frequently repeated within the reviews of their respective games. This repetition caused these words to appear disproportionately in the TF-IDF results, despite offering limited analytical value. To mitigate this bias, we implemented a procedure to remove all individual words from each game's title within its own reviews. This step ensured that our TF-IDF vectors would not be dominated by game-specific proper nouns and would instead highlight more meaningful and content-driven terms that reflect users' actual experiences and opinions.

3.3 Games: Feature Extraction (TF - IDF)

To explain our process for extracting features using TF-IDF, we start by clearly defining our corpus and the individual documents within it. It is important to note that this process is executed twice, once for reviews marked as positive and once for reviews marked as negative, allowing us to capture the distinct characteristics associated with each sentiment group.

3.3.1 Defining the Corpus and Documents

To extract key characteristics of games using TF-IDF, we define:

- **Corpus:** The entire dataset of all game reviews.

- **Documents:** Each game's cleaned and concatenated reviews are treated as a single document.

Example Corpus (All Reviews)

Game (Document)	Reviews (Text)
The Witcher 3	"Nice!", "Amazing story, deep RPG, incredible open world..."
Cyberpunk 2077	"My son loves it!", "Buggy, but great story and immersive city..."
Dark Souls 3	"Challenging combat, rewarding experience.", "Difficult but..."

Table 1: Example of how the corpus is structured, where each document represents a game. The column Reviews includes all reviews of that game.

For each document (i.e., a game), we apply the TF-IDF to extract the most representative terms from its reviews. This helps us identify which words are frequent in a specific game's reviews. After computing the TF-IDF matrix for each document, we extract the top 10 words with the highest scores. These represent the most characteristic features of that game's positive and negative sentiment profile.

Example TF-IDF Output

Game (Document)	"story"	"open world"	"combat"	"buggy"
The Witcher 3	0.45	0.50	0.10	0.02
Cyberpunk 2077	0.38	0.20	0.12	0.55
Dark Souls 3	0.10	0.05	0.48	0.05

Table 2: Example of a TF-IDF matrix: each game is a document and each column is a word.

Figure 3: Word clouds illustrating the top terms for positive (left) and negative (right) reviews of *The Witcher 3: Wild Hunt*.

Figure 3 demonstrates how the most frequent words differ between positive and negative reviews. For instance, players often highlight terms like *story*, *character*, *quest*, or *monster* when expressing positive feedback. On the other hand, negative reviews may feature words such as *loot*, *control*, or *movement*. Notably, some terms like *combat* appear in both clouds, indicating that combat can be either an appealing or a frustrating element depending on a player's personal preferences. This reinforces the importance of separating positive and negative reviews to capture the full spectrum of sentiments and tailor recommendations more accurately.

3.4 Creating a User Profile

Each user has a preference profile based on the games they like and dislike. We create:

- **Positive Profile:** Extract TF-IDF words from games the user liked.
- **Negative Profile:** Extract TF-IDF words from games the user disliked.

For this project, we created an artificial user profile based on our own gaming experiences, incorporating both likes and dislikes to closely mimic real-world scenarios. By writing six distinct reviews, some positive and some negative, we can later evaluate whether the resulting recommendations align with our personal preferences. Although we could have attempted to collect real user data, hardware limitations prevented us from storing the extensive volume of Steam reviews required for a fully authentic dataset. Additionally, we did not have access to Steam user IDs, which further limited our ability to gather comprehensive user metadata. Consequently, crafting our own reviews provided a practical solution that ensures the user profile contains enough data for meaningful testing. With more powerful hardware and access to genuine user identifiers, this limitation could be overcome; however, for our purposes, the artificial user profile sufficiently demonstrates the core concepts of our recommendation system.

Again, we create two separate TF-IDFs, one containing all positive reviews and one containing all negative reviews. However, for the user profile, the setup is slightly different. Since each review written by the user corresponds to a single game, we have exactly one review per game. Instead of keeping these reviews separate, we concatenate all reviews from games the user liked into one large document, and we do the same for the games they disliked. This results in exactly two documents—one representing positive sentiment and the other representing negative sentiment—which together form the entire corpus when building the user's TF-IDF profile.

We apply the same TF-IDF pipeline used for the games, with one key difference in how the TF-IDF values are computed. After performing text cleaning, stopword removal, and vectorization using both unigrams and bigrams, we calculate the TF-IDF scores. This process enables us to identify the terms that best represent the features a user likes versus those they dislike.

This leads to the following structure:

- The **game TF-IDF matrix** has one row per game (aggregated reviews), and each column corresponds to a word or n-gram.
- The **user TF-IDF profile** is made up of two vectors — one from the positive document and one from the negative document.

An example of the User TF-IDF for positive reviews could look like below.

Example TF-IDF Output for positive User Reviews

User Review Type (Document)	**"story"**	**"open world"**	**"graphics"**	**"buggy"**
Positive Reviews	0.45	0.50	0.40	0.05

Table 3: TF-IDF scores for selected terms in the two user documents: one containing all positive reviews, and one containing all negative reviews.

This design enables a direct comparison between each game's textual profile, compiled from a large number of reviews, and a user profile that highlights the terms most frequently associated with games they have liked or disliked, using cosine similarity as the metric for alignment.

3.5 Creating Recommendations (Similarity Scores)

In order to measure how closely each game's feature profile aligns with the user's preferences, we employ **cosine similarity**. This metric calculates the cosine of the angle between two vectors, as shown in the following formula:

$$\text{cosine\_similarity}(\vec{A}, \vec{B}) = \frac{\vec{A} \cdot \vec{B}}{\|\vec{A}\| \, \|\vec{B}\|} \tag{1}$$

where

- $\vec{A}$ is the user's TF-IDF vector (positive or negative).
- $\vec{B}$ is the TF-IDF vector representing a game's review features.
- $\vec{A} \cdot \vec{B}$ is the dot product of the two vectors.
- $\|\vec{A}\|$ and $\|\vec{B}\|$ are the magnitudes (Euclidean norms) of the vectors.

A higher cosine similarity indicates that the game shares more key terms with the user's preferred features, regardless of differences in overall word counts. As a result, it is a particularly effective way to compare TF-IDF vectors, focusing on the pattern of terms rather than their magnitude. This helps us rank and recommend games that best match the user's stated interests.

Game	Positive Similarity Score	Negative Similarity Score	Recommended?
Elden Ring	0.92	0.10	Yes
Horizon Zero Dawn	0.87	0.15	Yes
Cyberpunk 2077	0.70	0.85	No
Assassin's Creed Valhalla	0.78	0.90	No

Table 4: Example of how cosine similarity scores are used to make recommendations.

3.6 Improving the System by Adding Game Features (Genre)

To refine our predictions beyond the TF-IDF similarities, we leverage the genre booleans present in both the user's reviews and the game metadata. First, we compute a user feature profile by averaging the genre columns from all games that the user has positively reviewed. This yields a numerical vector indicating how much each genre matters to the user. For instance, if the user has recommended many action and RPG titles, the corresponding entries in the vector will be higher.

We then merge this user feature profile with the game metadata. By selecting the relevant columns for each game and converting them into a matrix, we can compare each row (i.e., each game's genre vector) to the user's feature vector via a dot product. A higher dot product value implies stronger alignment between the user's preferred genres and a given game's genres.

Final Score Computation

To create a single ranking metric, we incorporate this "feature match score" into the existing positive and negative similarity scores. Specifically, we use a weighting parameter λ to determine how much emphasis to place on genre alignment:

$$\text{Final\_Score} = \big(\text{Positive\_Similarity} \; - \; \text{Negative\_Similarity}\big) + \lambda \times \text{Feature\_Match\_Score}. \tag{2}$$

By adjusting λ, we can balance textual similarity (based on TF-IDF) against genre matching. Through experimentation, we found $\lambda = 0.5$ to yield intuitive results, although this choice may vary with different user preferences or datasets.

When we rerank the recommendations using this final score, we see noticeable changes in the top results. Certain titles that were previously absent may now appear if they align strongly with the user's preferred genres.

4 Results

To evaluate the effectiveness of our recommendation system, we analyze three key elements: (1) the user's preferred genres, (2) the top positive words that define the user's taste, and (3) the resulting top recommendations after incorporating genre matching.

4.1 User Profile: Preferred Genres

In Figure 4, we visualize the user's top five genres by their mean score. This score is computed by averaging the genre booleans across all positively reviewed games. The height of each bar indicates how much the user tends to favor each genre, with *RPG* being the highest-scoring category.

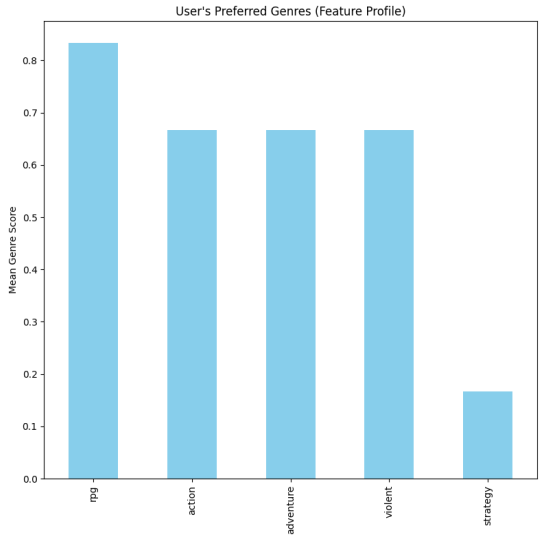

Figure 4: User's Preferred Genres

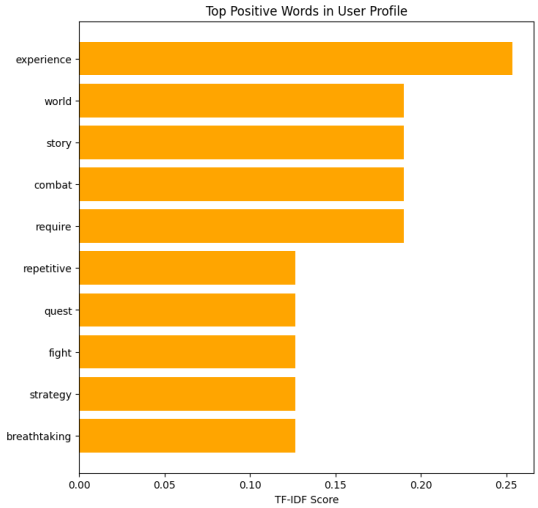

Figure 5: Top Positive Words in User Profile

4.2 User Profile: Top Positive Words

We also examine the most influential words in the user's positive reviews. Figure 5 shows the top ten terms by TF-IDF score, revealing the specific attributes that most resonate with the user. Words like *experience* and *world* appear prominently, suggesting that immersive gameplay and expansive environments are important factors.

4.3 Top Recommended Games

After combining the TF-IDF similarities (both positive and negative) with the genre match score, we arrive at a final ranking of games. Figure 6 presents the top ten recommended titles based on the *Final Score*. Games such as *Kingdom Come: Deliverance II* and *Warhammer 40,000: Space Marine 2* rank highly, indicating a strong alignment with the user's preferred genres and review-derived keywords.

As shown in the figures above, incorporating both textual and genre-based preferences significantly influences the final recommendations. While *RPG* and *adventure* genres remain strong predictors of user satisfaction, certain titles with matching attributes also benefit from high textual similarity scores. These results underscore the importance of balancing content-based features (like keywords) with broader metadata (like genres) to produce more nuanced and personalized game suggestions.

5 Performance

Measuring performance in this system is not as straightforward as in ordinary ML applications, so this shall shortly be described in the section below.

Most traditional recommendation systems rely on collaborative filtering or content-based filtering to suggest items to users. [AI, 2024]. Often, systems analyze user behaviour, such as past purchases to predict which items a user might like. The most common evaluation methods for these systems include Precision@K, Recall@K, Mean Reciprocal Rank (MRR), and Normalized Discounted Cumulative Gain (NDCG). These metrics assess how many of the recommended items were relevant to the user based on prior knowledge, like whether the user liked the game or not.

However, in our case, we build recommendations using a TF-IDF-based model that already leverages positive and negative user reviews. This means that a direct evaluation using traditional metrics could

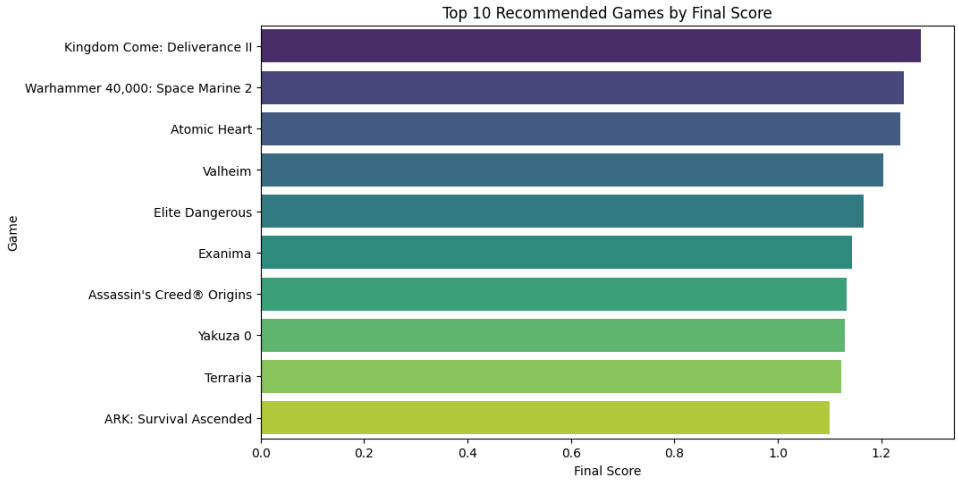

Figure 6: Top 10 Recommended Games by Final Score

introduce bias. Since the model is trained on the user's own reviews, the system naturally recommends games similar to those the user has already reviewed positively. As a result, evaluating the system by checking how many previously liked games appear in the recommendations does not provide a meaningful measure. It would be a meaningless feedback loop, not accurately improving the recommendations. Instead, a better approach is to assess the ranking quality of recommended games by analyzing the distribution of similarity scores, measuring feature alignment with the user's preferences, and comparing positive versus negative similarity scores.

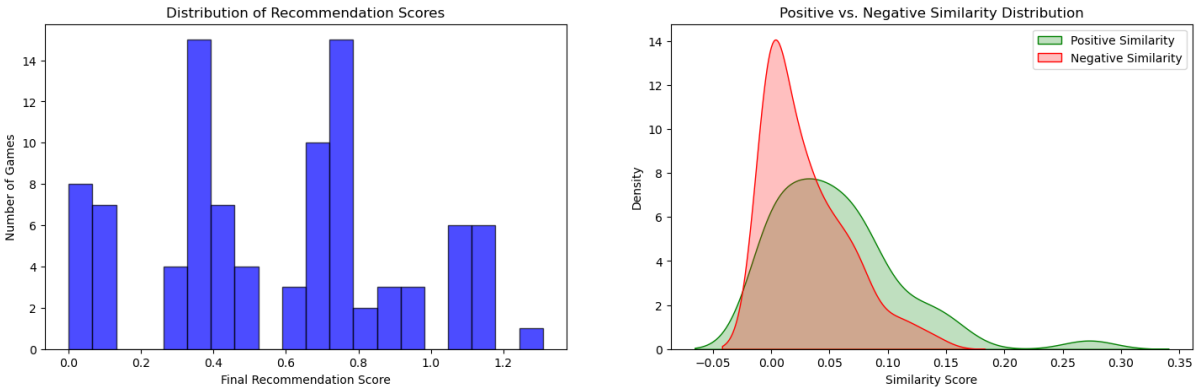

Figure 7: Comparison of Image and Cosine Similarity Visualization

While these analyses provide valuable insights into the system's behavior, they should not be misinterpreted as absolute performance measures. For instance, a lower cosine similarity score does not necessarily indicate a poor recommendation—it might suggest that the system is diversifying recommendations rather than strictly adhering to past preferences. Similarly, strong correlations between user preferences and recommendations indicate alignment, but they do not guarantee user satisfaction. Therefore, these analyses should be used as diagnostic tools rather than definitive performance metrics.

We quickly realize that we need some kind of additional information to improve our model by using performance metrics. Therefore, we decided to finally implement an adaptive user feedback system.

5.1 Adaptive Personalization. Leveraging Additional User Feedback

To further refine the recommendation system, we introduced an adaptive approach that incorporates user feedback into the model. This step enhances personalization by dynamically adjusting the recommendation scores based on newly provided preferences.

Updating the User Profile

Initially, the user profile was derived from positively and negatively reviewed games by computing TF-IDF representations of key terms. Additionally, the user's genre preferences were extracted by averaging the boolean genre indicators of positively reviewed games.

Now to refine this system, we can imagine the following: The user is given the option to like or dislike the recommendation. This might sound complex, but all we have to do is updating the feature profile to integrate this.

- If the user **likes** a recommended game, the feature score is **strengthend** in the user profile.
- If the user **dislikes** a recommended game, its influence is **reduced**, ensuring that similar games receive a lower score in future recommendations.

It is important to note that this only applies to the feature scores, not to the text scores to keep things simple. The purpose of this is not to have the most soffisticated system but to understand the limitations and potential.

Recomputing Similarities and Final Score

After updating the user profile, the recommendation process is rerun:

1. **TF-IDF Profiles:** The user's positive and negative term representations are recomputed.
2. **Cosine Similarity:** Each game's text representation is compared to the updated user profile.
3. **Feature Matching:** The dot product between the game's genre features and the user's preferences is recalculated.
4. **Final Score Update:** The new recommendation score integrates positive similarity, penalizes negative similarity, and incorporates genre matching.

Benefits of this integration

This iterative process allows the system to adapt to the user. Additionally, we get some kind of information of how well the algorithm performs, successfully improving performance. We dynamically adjust weights so the system ensures that recommendations are dynamic and long term fruitful.

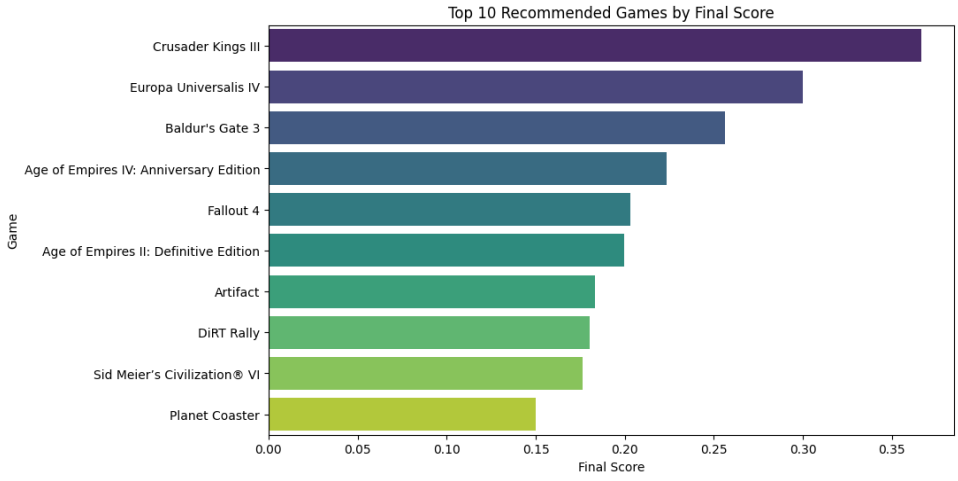

Figure 8: Top 10 Recommended Games by Final Score After Adaptive Feedback

Results

As we can see in the code and in Figure 8, based on the gamer's past reviews and preferences, the top recommended games for him/her are Crusader Kings III (which can be disregarded here as we manually liked the recommendation), **, Europa Universalis IV, Age of Empires IV: Anniversary Edition, and Baldur's Gate 3**. After reading about these games, we can objectively say that they align well with

preference for strategy, adventure, and RPG elements, while also incorporating action and immersive gameplay. Of course, here we mainly prioritize feature importance instead of review importance, but this could be extended to ultimately have a long term robust system.

6 Discussion

Our results demonstrate that combining text-based features (via TF-IDF) with game metadata (particularly genres) can significantly enhance recommendation quality. By separately modeling positive and negative reviews, the system captures not only what users like about certain titles, but also which attributes they find off-putting. This two-pronged approach to user preference modeling, one for liked features and one for disliked features, enables more nuanced, sentiment-aware recommendations.

Moreover, the adaptive feedback mechanism shows how even a small amount of additional user input can reshape the final ranking. For instance, after indicating approval for specific recommended titles (e.g., *Crusader Kings III*) and disapproval for others (e.g., *Rust*), the system quickly recalibrates the user profile, increasing the weight of relevant genres and penalizing attributes tied to disliked games. As a result, the updated recommendations shift toward titles that better match the user's evolving tastes.

Despite these encouraging findings, several limitations remain. The artificial user profile may not fully represent the complexity of real-world gaming preferences, and the number of reviews collected is constrained by both hardware limitations and Steam restrictions. Additionally, while our TF-IDF approach captures valuable textual insights, more advanced natural language processing techniques, such as LLMS could further refine the system's understanding of user preferences. Finally, evaluating recommendation quality poses its own challenges: traditional metrics (e.g., Precision@K, Recall@K) may not be ideal for systems already trained on the user's own reviews, requiring more nuanced methods that account for adaptive and sentiment-aware models.

Overall, our findings suggest that a hybrid strategy, incorporating both textual analysis of reviews and structured metadata, offers a promising path for personalized game recommendations. The adaptive feedback loop, in particular, highlights the system's potential to grow more accurate over time, ultimately improving the user's experience by continually aligning recommendations with their evolving interests. Furthermore, if Steam were to grant access to additional metadata, such as the number of hours a user has played a game and other engagement metrics currently used in their recommendation systems, the integration of review-based insights could further enhance these systems. This extra layer of data could help refine user profiles and lead to even more precise recommendations, demonstrating the considerable potential of leveraging textual feedback alongside traditional usage statistics.

References

[Adomavicius and Tuzhilin, 2005] Adomavicius, G. and Tuzhilin, A. (2005). Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17(6):734–749.

[AI, 2024] AI, N. (2024). Recommender systems: Metrics to evaluate your model. Accessed: March 20, 2025.

[Lops et al., 2011] Lops, P., de Gemmis, M., and Semeraro, G. (2011). Content-based recommender systems: State of the art and trends. In *Recommender Systems Handbook*, pages 73–105. Springer.

[Ni et al., 2019] Ni, J., Li, J., and McAuley, J. (2019). Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 188–197.