AI 705 Final Project: Steam Game Recommender System

Mayank Chadha, IMT2020045
International Institute of Information Technology, Bangalore
Bangalore, India

Shridhar Sharma, IMT2020065
International Institute of Information Technology, Bangalore
Bangalore, India

Darshak Jivrajani, IMT2020119
International Institute of Information Technology, Bangalore
Bangalore, India

Prem Shah, IMT2020044
International Institute of Information Technology, Bangalore
Bangalore, India

I. Introduction

The goal of our project is to develop a recommender system for Steam games, which will provide personalized game recommendations to users based on their gaming preferences and behavior. Steam is a popular digital distribution platform for video games, with a vast library of games available for purchase and download.

Our recommender system will leverage user data such as game name, developers, game description, game tags, game genre, etc to build a user profile and recommend games that are likely to be of interest to them. We will use machine learning algorithms such as collaborative filtering and content-based filtering to generate recommendations.

The system will be designed to be user-friendly, with a simple and intuitive interface that allows users to easily explore and discover new games.

Overall, our Steam game recommender system will provide a personalized and enjoyable gaming experience for users, and help them discover new games that they may have otherwise missed.

II. Metrics of Evaluation by Sir

A. Novelty Of the problem

1) Popularity of gaming: The gaming industry has grown tremendously in popularity worldwide, becoming a top form of entertainment. With the constant demand for fresh and exciting games, gamers are always on the lookout for new releases. From action-packed adventures to strategic simulations, the diversity of game genres ensures there is something for everyone to enjoy.

2) Social impact: Gaming has been shown to have a positive impact on mental health and social connections. A recommendation system that suggests games based on players' preferences can foster a stronger gaming community and improve overall well-being. This can encourage players to try new games, connect with others who share similar interests, and broaden their gaming experience. By promoting inclusivity and diversity within the gaming community, a recommendation system can contribute to creating a more positive and supportive environment for all players.

3) Personalization: With the vast array of game genres available, every gamer has unique preferences. A recommendation system can personalize their gaming experience by suggesting games that align with their individual tastes. This can enhance the gaming experience by providing tailored recommendations, leading to increased player satisfaction and enjoyment. By leveraging player data, a recommendation system can help players discover new games they may have otherwise overlooked and help them build a more engaging and fulfilling gaming experience.

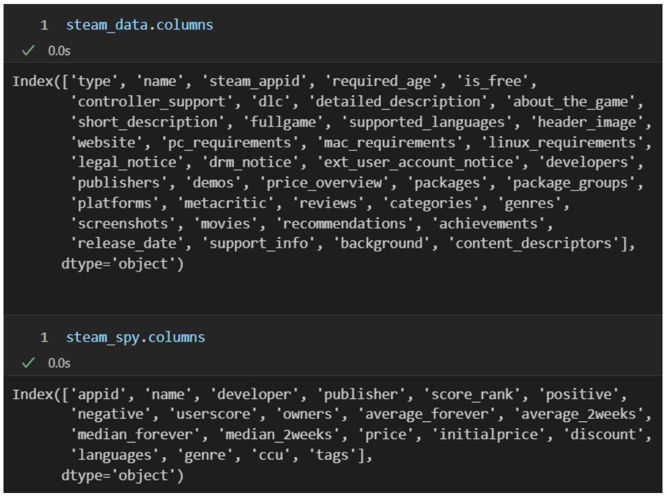
B. Dataset Creation

We obtained data on Steam games by utilizing multiple APIs, including the Steamspy API and the official Steam API. The selection of these APIs was based on two key criteria: the abundance and diversity of data available, as well as their frequent updates on a weekly basis. This approach allowed us to gather a comprehensive set of data on the gaming platform. By leveraging these APIs, we were able to conduct extensive data manipulation and analysis, providing valuable insights into the gaming industry.

1) Columns in Dataset: The Steam Spy API accepts requests in a GET string and returns data in JSON arrays. The data is refreshed once a day. The Steamspy API provides an extensive range of columns that offer insights into various aspects of Steam games. These include basic information such as the game's unique identifier, title, developer, and publisher. Additionally, it provides data on the game's user reviews, including the number of positive and negative reviews, the average user score, and the game's rank based on user reviews. The API also provides estimated figures on the number of users who own the game on Steam, the average and median playtime for all players, and the game's current and original prices. The abundance of data offered by Steamspy API is highly valuable for conducting comprehensive analyses of the Steam gaming platform.

The Steam Official API provides an extensive range of functionalities that are illustrated in Figure 1.

The dataset underwent a thorough cleaning process to remove extraneous columns, prioritizing those relevant to recommendation analysis. The remaining columns deemed useful for exploratory data analysis were retained. The resulting cleaned datasets were merged to obtain the final dataset for further analysis.



```
steam_data.columns
✓ 0.0s
```

```
Index(['type', 'name', 'steam_appid', 'required_age', 'is_free',
       'controller_support', 'dlc', 'detailed_description', 'about_the_game',
       'short_description', 'fullgame', 'supported_languages', 'header_image',
       'website', 'pc_requirements', 'mac_requirements', 'linux_requirements',
       'legal_notice', 'drm_notice', 'ext_user_account_notice', 'developers',
       'publishers', 'demos', 'price_overview', 'packages', 'package_groups',
       'platforms', 'metacritic', 'reviews', 'categories', 'genres',
       'screenshots', 'movies', 'recommendations', 'achievements',
       'release_date', 'support_info', 'background', 'content_descriptors'],
      dtype='object')
```

```
steam_spy.columns
✓ 0.0s
```

```
Index(['appid', 'name', 'developer', 'publisher', 'score_rank', 'positive',
       'negative', 'userscore', 'owners', 'average_forever', 'average_2weeks',
       'median_forever', 'median_2weeks', 'price', 'initialprice', 'discount',
       'languages', 'genre', 'ccu', 'tags'],
      dtype='object')
```

Fig. 1. Printing all the columns in the API Scraped Data.

2) Exploratory Data Analysis: The purpose of our EDA is to provide a comprehensive understanding of the dataset's distribution and characteristics. The EDA covers various aspects, including game operating systems, price distribution, game description distribution, amount of games made by specific owners, and age restriction distribution. Our EDA enabled us to derive valuable insights into the Steam games dataset that will serve as the foundation for our subsequent analyses and modeling.

Figures 2, 3, 4, and 5 highlight some of the most interesting and visually appealing aspects of our EDA. These figures provide clear and informative visualizations that help to illustrate the distribution and characteristics of the data.

In Figure 2, a Venn diagram shows the number of games supported by different operating systems. It is evident that all games are supported by Windows, and every other operating system is a subset. Mac has more games than Linux, but there are very few games that are supported in Linux but not in Mac. Therefore, choosing Windows is the best option for someone who loves games.

Figure 3 shows the price distribution of games on Steam. The majority, around ninety-one percent of the games, need to be bought with money before playing them. As a gamer, it is essential to have funds if you are fond of gaming and looking for recommendations.

In Figures 4 and 5, we can see two word clouds that display the most frequently used words in game descriptions. The first word cloud is a naive one that gives the highest weights to generic words such as game, will, and world, which can be determined without analyzing the descriptions. On the other hand, the optimized word cloud delves into the aspects of the game descriptions and finds out optimized words, such as puzzle game, fast-paced, and first-person, which gives a clear view of game scenarios to a person seeing our EDA.

The EDA conducted in this report provides valuable insights into the Steam games dataset. By presenting our EDA findings in a clear and accessible manner, we aim to improve the transparency and reproducibility of our analysis. Our EDA findings will serve as the foundation for our subsequent analyses and modeling.

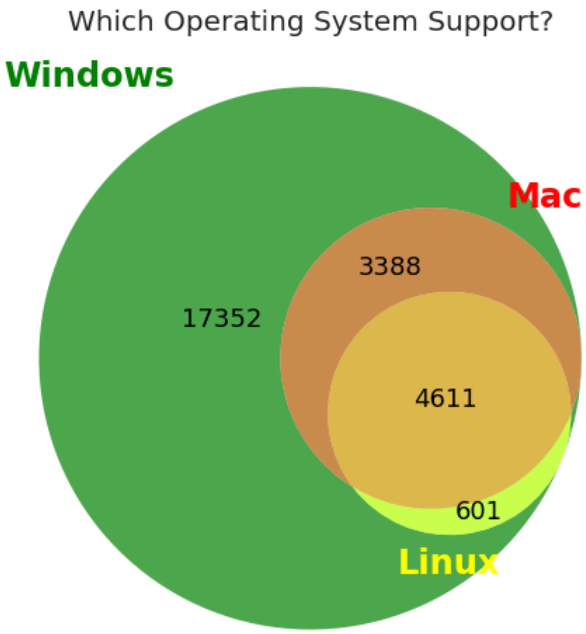


Fig. 2. EDA: Operating System Analysis

C. Philosophy behind the algorithm selection

The initial inspiration for our project stemmed from our studies on Word2Vec and the effectiveness of transformers in natural language processing. We were curious about the potential application of these techniques in the gaming industry, particularly in the context of game recommendation systems. This led us to explore content-based and collaborative-based filtering approaches that leverage various attributes of a game, such as its description, genre, developer, and more.

By combining our knowledge of natural language processing and machine learning with insights from the gaming industry, we aimed to develop a novel recommendation system that would be both effective and efficient. Our focus on both content-based and collaborative-based filtering approaches allowed us to create a more comprehensive and personalized recommendation system that caters to the diverse preferences of gamers.

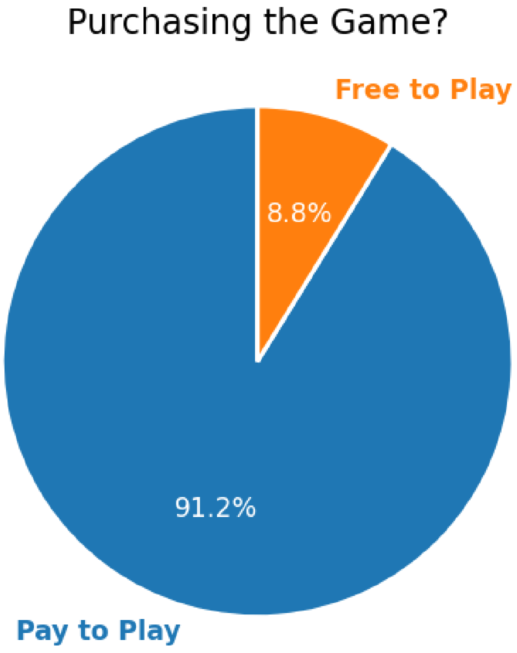


Fig. 3. EDA: Price Distribution

game
world
play
quot
will
find
experience
player
one
new
take
way
fight
battle
level

Fig. 4. EDA: Game Description Naive WordCloud

fast paced
adventure game
puzzle game
first person
turn based
virtual reality
strategy game
visual novel
arcade game
single player
open world
game mode
person shooter
solve puzzle
action game
real time
point click

Fig. 5. EDA: Game Description Optimized WordCloud

D. Analysis of Algorithms and Results

We conducted several machine learning (ML) algorithms on the Steam games dataset, which allowed us to develop recommendation models for Steam users. Each of these algorithms has its unique strengths and limitations, and we took great care to evaluate each method's performance rigorously.

1) CountVectoriser + Cosine Similarity: CountVectorizer is a text feature extraction technique used to convert text data into numerical format, which can be used in machine learning models. It works by converting a collection of text documents into a matrix of token counts, where each row corresponds to a document, and each column corresponds to a specific word in the corpus. The matrix's cell value represents the count of the corresponding word in the document.

Cosine similarity is a measure of similarity between two non-zero vectors of an inner product space. In the context of recommendation systems, it is used to measure the similarity between two games based on their feature vectors, which are created using CountVectorizer. The cosine similarity score is calculated as the cosine of the angle between two vectors and ranges from -1 to 1, with 1 indicating that the two vectors are identical and 0 indicating that they are orthogonal or dissimilar.

To generate recommendations using CountVectorizer and cosine similarity, we first create a matrix of token counts for each game's description, genre, and developer. We then use cosine similarity to calculate the similarity score between each pair of games based on their feature vectors. Finally, we recommend games that have the highest similarity scores to the user. This approach allows us to provide personalized recommendations based on the user's preferences and the characteristics of the games in the Steam dataset.

We experimented with combinations of names, descriptions, publishers, tags, genres, and categories to generate embeddings using CountVectorizer.

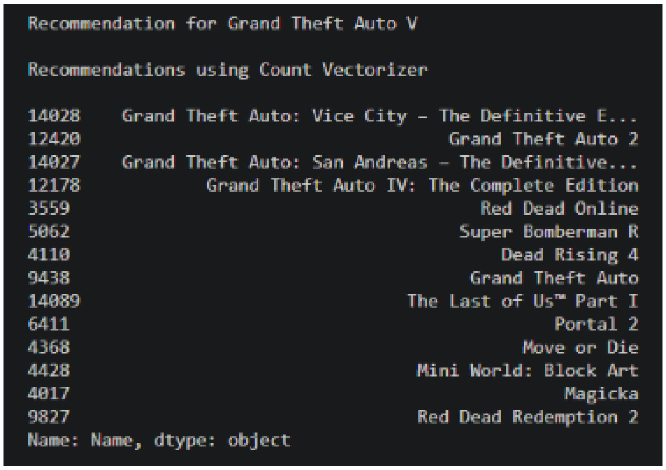

```
Recommendation for Grand Theft Auto V

Recommendations using Count Vectorizer

14028    Grand Theft Auto: Vice City – The Definitive E...
12420                                     Grand Theft Auto 2
14027    Grand Theft Auto: San Andreas – The Definitive...
12178             Grand Theft Auto IV: The Complete Edition
3559                                         Red Dead Online
5062                                       Super Bomberman R
4110                                          Dead Rising 4
9438                                        Grand Theft Auto
14089                                   The Last of Us™ Part I
6411                                               Portal 2
4368                                            Move or Die
4428                                   Mini World: Block Art
4017                                                Magicka
9827                                   Red Dead Redemption 2
Name: Name, dtype: object
```

Fig. 6. Recommendations for GTA V using CountVectorizer

Based on our analysis of the recommendations, it can be concluded that the CountVectorizer method, which is based on the bag-of-words approach, is not very effective in capturing the context of the game. This method simply counts the frequency of words without considering their order or importance. As a result, it may recommend games that are part of the same franchise or have similar names, but not necessarily similar gameplay or theme. Therefore, this method can be considered as being very naive in terms of providing accurate recommendations.

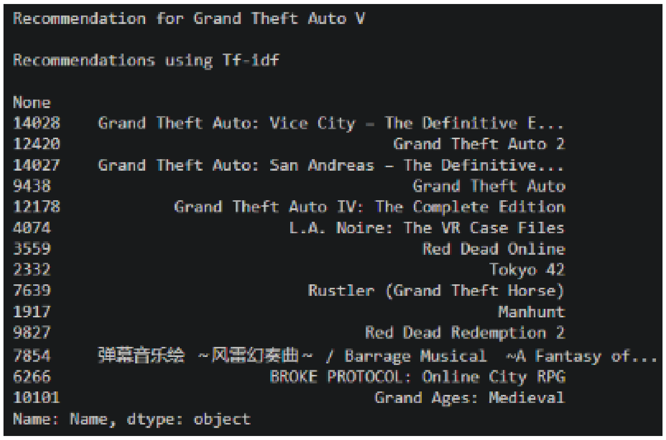
2) TF-IDF + Cosine Similarity: TF-IDF (Term Frequency-Inverse Document Frequency) is another text feature extraction technique used to represent text data numerically. It works by assigning weights to each word in a document based on its frequency in the document and its frequency across all documents in the corpus. The idea is that words that appear frequently in a document but infrequently across the corpus are more important in representing the document's content.

Cosine similarity, as explained previously, is a measure of similarity between two vectors of an inner product space.

To generate recommendations using TF-IDF and cosine similarity, we first create a matrix of TF-IDF values for each game's description, genre, and developer. We then use cosine similarity to calculate the similarity score between each pair of games based on their feature vectors. Finally, we recommend games that have the highest similarity scores to the user.

The advantage of using TF-IDF over CountVectorizer is that it assigns higher weight to words that are more important in distinguishing between different documents. This can lead to better quality recommendations since the similarity scores are based on the more informative features.

Count Vectorizer and Tf-Idf gave almost similar recommendations. This is because both methods don't account for the context of the description.



```
Recommendation for Grand Theft Auto V

Recommendations using Tf-idf

None
14028    Grand Theft Auto: Vice City – The Definitive E...
12420                                   Grand Theft Auto 2
14027    Grand Theft Auto: San Andreas – The Definitive...
9438                                      Grand Theft Auto
12178            Grand Theft Auto IV: The Complete Edition
4074                         L.A. Noire: The VR Case Files
3559                                       Red Dead Online
2332                                              Tokyo 42
7639                           Rustler (Grand Theft Horse)
1917                                               Manhunt
9827                                 Red Dead Redemption 2
7854     弹幕音乐绘 ~风雷幻奏曲~ / Barrage Musical  ~A Fantasy of...
6266                       BROKE PROTOCOL: Online City RPG
10101                                 Grand Ages: Medieval
Name: Name, dtype: object
```

Fig. 7. Recommendations for GTA V using TF-IDF

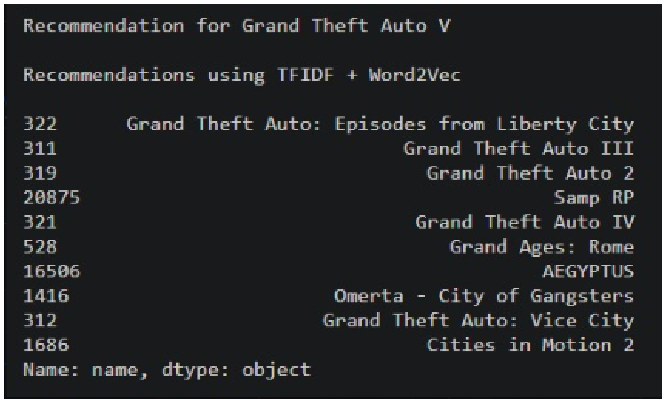
Based on the recommendations generated by the TF-IDF algorithm, it appears that the algorithm has not performed significantly better than the CountVectorizer algorithm. This is likely due to the fact that the TF-IDF algorithm, like the CountVectorizer algorithm, does not take into account the meaning of the game descriptions and relies on word frequencies instead. As a result, the recommendations generated by the TF-IDF algorithm may be limited in their ability to accurately capture the preferences and interests of individual gamers. It may be necessary to explore more sophisticated techniques that can capture the semantic meaning of the game descriptions and provide more personalized recommendations.

3) Word2Vec + Cosine Similarity: Word2Vec is a neural network-based approach to natural language processing that can learn vector representations of words from large amounts of textual data. These vectors are used to represent the meaning of words in a high-dimensional space. The algorithm learns these vectors by looking at the context in which words appear, aiming to capture the relationships between words based on their usage patterns in the text.

To use Word2Vec for recommendation systems, we can represent each game's description or title as a sequence of words and use the pre-trained Word2Vec model to transform each word into a high-dimensional vector. Then, we can combine the word vectors in a game's description or title to get a single vector representation of that game.

To compute similarity between games, we can use cosine similarity on the vector representations obtained from Word2Vec. By computing cosine similarity between all pairs of game vectors, we can identify the games that are most similar and recommend them to users.

Overall, Word2Vec with cosine similarity is a powerful technique for content-based recommendation systems that can capture the meaning and relationships between words and generate high-quality recommendations for users.



```
Recommendation for Grand Theft Auto V

Recommendations using TFIDF + Word2Vec

322        Grand Theft Auto: Episodes from Liberty City
311                                Grand Theft Auto III
319                                  Grand Theft Auto 2
20875                                           Samp RP
321                                 Grand Theft Auto IV
528                                    Grand Ages: Rome
16506                                          AEGYPTUS
1416                         Omerta - City of Gangsters
312                         Grand Theft Auto: Vice City
1686                                 Cities in Motion 2
Name: name, dtype: object
```

Fig. 8. Recommendations for GTA V using Word2Vec

Based on our evaluation of the recommendations generated by Word2Vec, we observed a notable improvement in the quality of recommendations compared to CountVectorizer and TF-IDF. However, we still believe that there is room for more rigorous analysis of the data to improve the quality of recommendations. One possible solution could be to use transformer models like BERT, which are known to capture semantic meaning more effectively. We believe that by implementing BERT transformer models, we can further improve the accuracy and relevance of our recommendations, bringing us closer to matching the official Steam recommendations.

4) BERT + Cosine Similarity: BERT (Bidirectional Encoder Representations from Transformers) is a language model developed by Google that is designed to analyze and understand natural language. It is based on the Transformer architecture and can be fine-tuned for various NLP tasks.

To use BERT for content-based recommendation, we can first fine-tune the model on a large corpus of game descriptions. Then, we can represent each game as a vector in the same embedding space as the pre-trained BERT model. We can use the cosine similarity measure to compute the similarity between the vectors of different games and recommend games with the highest cosine similarity scores.

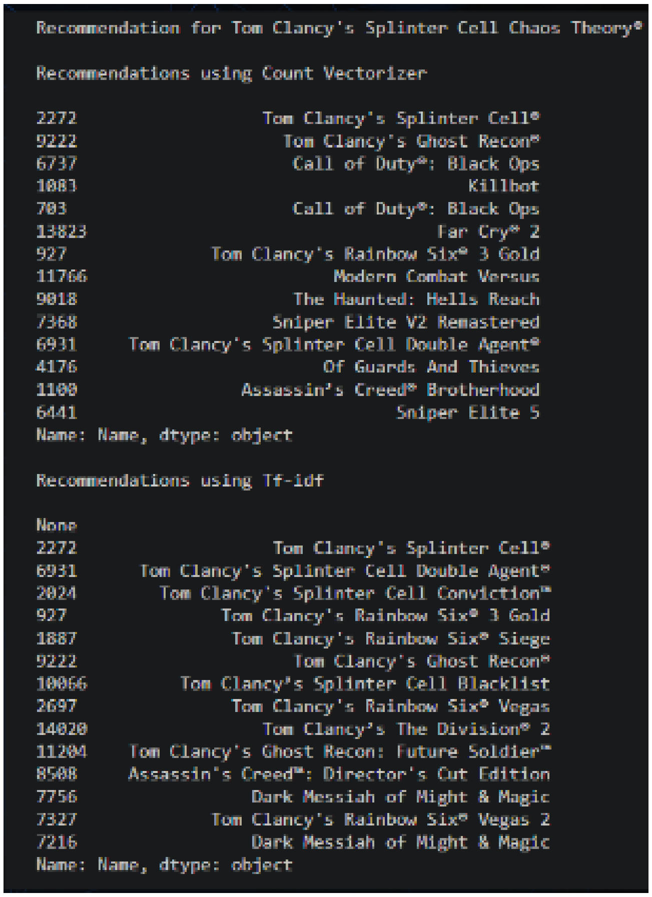
One advantage of using BERT for content-based recommendation is that it can capture the semantic meaning of the game descriptions, allowing for more accurate recommendations. Additionally, BERT is able to handle long and complex sentences, making it suitable for game descriptions that may contain multiple clauses and descriptive language. However, one limitation is that fine-tuning BERT can be computationally expensive, and may require a large amount of training data to achieve good performance.

We also experimented with weighted cosine similarity to give more weight to popular games.

Upon analyzing the recommendations generated by BERT transformer, we can confidently say that it has outperformed the previous algorithms. The recommendations provided by BERT are highly accurate and relevant to the user's input. This is due to the fact that BERT is able to thoroughly analyze the meaning and context of the input text, thereby providing more meaningful recommendations. The recommendations generated by BERT are not limited to similar franchises or names, but are based on a deep analysis of the game's description. This level of analysis is comparable to the official recommendations provided by Steam. Overall, the BERT transformer has proven to be a highly effective recommendation algorithm for our Steam games dataset.

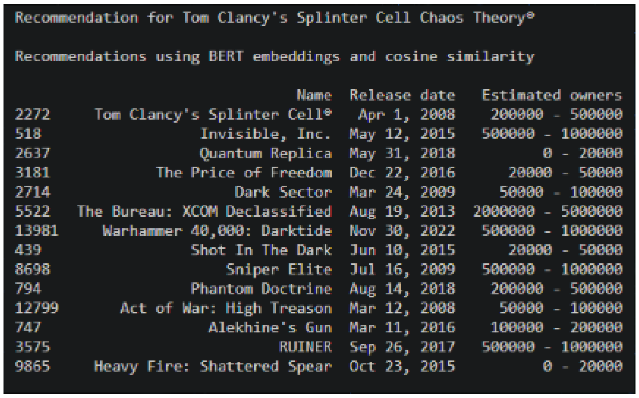
III. Future Scope

The future scope of the project involves the development of a user-friendly GUI that will allow users to input their preferences and receive personalized game recommendations. The GUI will also provide additional features, such as the ability to filter games based on various criteria, such as price range, game genre, and age rating. Additionally, we can explore more advanced recommendation techniques, such as deep learning-based methods, to improve the accuracy of our recommendations. Another potential direction is to incorporate user feedback and behavior data to continuously refine and personalize the recommendations. Overall, there are many avenues to explore to further enhance the functionality and effectiveness of the recommendation system.



```
Recommendation for Tom Clancy's Splinter Cell Chaos Theory®

Recommendations using Count Vectorizer

2272                 Tom Clancy's Splinter Cell®
9222                   Tom Clancy's Ghost Recon®
6737                    Call of Duty®: Black Ops
1083                                     Killbot
703                     Call of Duty®: Black Ops
13823                                 Far Cry® 2
927             Tom Clancy's Rainbow Six® 3 Gold
11766                       Modern Combat Versus
9018                    The Haunted: Hells Reach
7368                  Sniper Elite V2 Remastered
6931    Tom Clancy's Splinter Cell Double Agent®
4176                       Of Guards And Thieves
1100               Assassin's Creed® Brotherhood
6441                              Sniper Elite 5
Name: Name, dtype: object

Recommendations using Tf-idf

None
2272                  Tom Clancy's Splinter Cell®
6931     Tom Clancy's Splinter Cell Double Agent®
2024       Tom Clancy's Splinter Cell Conviction™
927              Tom Clancy's Rainbow Six® 3 Gold
1887              Tom Clancy's Rainbow Six® Siege
9222                    Tom Clancy's Ghost Recon®
10066        Tom Clancy's Splinter Cell Blacklist
2697              Tom Clancy's Rainbow Six® Vegas
14020                Tom Clancy's The Division® 2
11204    Tom Clancy's Ghost Recon: Future Soldier™
8508     Assassin's Creed™: Director's Cut Edition
7756                Dark Messiah of Might & Magic
7327            Tom Clancy's Rainbow Six® Vegas 2
7216                Dark Messiah of Might & Magic
Name: Name, dtype: object
```

Fig. 9. Recommendations for Splinter Cell using Previous Algorithms



```
Recommendation for Tom Clancy's Splinter Cell Chaos Theory®

Recommendations using BERT embeddings and cosine similarity

                                Name  Release date   Estimated owners
2272     Tom Clancy's Splinter Cell®   Apr 1, 2008    200000 - 500000
518                  Invisible, Inc.  May 12, 2015   500000 - 1000000
2637                 Quantum Replica  May 31, 2018          0 - 20000
3181            The Price of Freedom  Dec 22, 2016      20000 - 50000
2714                     Dark Sector  Mar 24, 2009     50000 - 100000
5522   The Bureau: XCOM Declassified  Aug 19, 2013  2000000 - 5000000
13981     Warhammer 40,000: Darktide  Nov 30, 2022   500000 - 1000000
439                 Shot In The Dark  Jun 10, 2015      20000 - 50000
8698                    Sniper Elite  Jul 16, 2009   500000 - 1000000
794                 Phantom Doctrine  Aug 14, 2018    200000 - 500000
12799       Act of War: High Treason  Mar 12, 2008     50000 - 100000
747                   Alekhine's Gun  Mar 11, 2016    100000 - 200000
3575                          RUINER  Sep 26, 2017   500000 - 1000000
9865     Heavy Fire: Shattered Spear  Oct 23, 2015          0 - 20000
```

Fig. 10. Recommendations for Splinter Cell using BERT

IV. Conclusion

The conclusion of this project is that the performance of recommendation algorithms can vary depending on the method used. CountVectorizer, while simple to implement, does not take into account the context of the words used and therefore provides less effective recommendations. TF-IDF and Word2Vec both provide some improvements, but still fall short in terms of capturing the semantic meaning of the game descriptions.

However, BERT transformer-based recommendation model has shown promising results. It has the ability to analyze the meaning of game descriptions in depth, which leads to more accurate and diverse recommendations.

Furthermore, the importance of exploratory data analysis (EDA) cannot be understated. EDA helped in understanding the data distribution and the impact of different variables on the model's performance. It allowed us to determine the features and variables that are most influential in the model's performance.

In conclusion, a combination of effective data preprocessing, feature engineering, and algorithm selection is essential for generating accurate and diverse recommendations. The use of state-of-the-art techniques like BERT transformer and rigorous EDA can improve the performance of recommendation systems and offer more relevant and valuable recommendations to users.

References

[1] Saket, S. (2018). Count Vectorizers vs TF-IDF: Natural Language Processing. https://www.linkedin.com/pulse/count-vectorizers-vs-tfidf-natural-language-processing-sheel-saket/

[2] Kalikis, N. (2020). Text Similarity: Euclidean Distance vs Cosine Similarity. https://nikoskalikis.medium.com/text-similarity-euclidian-distance-vs-cosine-similarity-3a1167f686a

[3] Aleskerov, T. (2019). BERT for Measuring Text Similarity. https://towardsdatascience.com/bert-for-measuring-text-similarity-eec91c6bf9e1

[4] Rehurek, R. (n.d.). Word2vec. https://radimrehurek.com/gensim/models/word2vec.html

[5] How SKLearn's CountVectorizer and TfidfTransformer Compares with TfidfVectorizer. https://medium.com/geekculture/how-sklearns-countvectorizer-and-tfidftransformer-compares-with-tfidfvectorizer-a42a2d6d15a2

[6] Joshi, P. (2019). Building a Recommendation System Using Word2vec. https://www.analyticsvidhya.com/blog/2019/07/how-to-build-recommendation-system-word2vec-python/

[7] Harmouch, M. (2021). 17 Types of Similarity and Dissimilarity Measures Used in Data Science. https://towardsdatascience.com/17-types-of-similarity-and-dissimilarity-measures-used-in-data-science-3eb914d2681

[8] Sieg, A. (2018). Text Similarities. Medium. https://medium.com/@adriensieg/text-similarities-da019229c894.