Fuzzy Systems and Data Mining IX
A.J. Tallón-Ballesteros and R. Beltrán-Barba (Eds.)

doi:10.3233/FAIA231096

Game Recommendation System

Man-Ching Yuen [a,1] , Chi-Wai Yung [a] , Wing-Fat Cheng [a] ,
Hon-Pong Tsang [b], Chi-Ho Kwan [b] , Chun-Lok Chan [b], Po-Yi Li [b]
[a] *Department of Information Technology, Vocational Training Council, China*
[b] *Department of Applied Data Science, Hong Kong Shue Yan University, China*

Abstract. Recommendation systems are widely adopted in many areas to provide better services to customers. As there are many games stored in online games platforms, people may be confused when choosing or buying the game that is suitable for them. Many game platforms would like to have a game recommendation system, so that it can automatically recommend the right games to their customers. However, there are a lot of difficulties in developing game recommendation systems. First, it is difficult to collect and organize data on customers' behavior. Second, the user interface needs to be easier to use for customers, such as the charts displayed that customers are interested in. In this paper, we have developed a games recommender system with complete functional recommending features. By using machine learning techniques and applying data visualization on our system, we build a recommendation system that can showcase flexible outcomes with the same element as the user input, which can give the user more choice when finding the games they want.

Keywords. Recommendation system, game, machine learning, web scraping and data visualization.

1. Introduction

Nowadays, there are loads of games platforms such as PS4 store, Steam, Xbox etc. However, not all the games platforms have a built-in recommender system to help users to find games they desire [1, 2, 3, 4, 5]. It is because some games platforms do not have enough data visualization to show the current trend of games clearly. Furthermore, some people think that most of the existing recommender systems can not satisfy the users. They may face difficulties when trying to find a suitable game to play using the existing recommender systems.

Based on these reasons, we want to build a recommendation system to users who want to play games which suit their taste. We will investigate the relationship between each of the categories and the analytics of the trend of the selected game platform. So, we can find out the most popular games for each type of genre. By entering different elements of the game that the user is interested in, the recommender system can list out the game that the user may prefer.

Our goal of this project is trying to build a recommender system that is uniquely different from other recommender systems. We hope to improve the accuracy, efficiency to about 10% better than the existing recommender system. And to make our system

[1] Corresponding Author, Man-Ching YUEN, Department of Applied Data Science. Hong Kong Shue Yan University, Hong Kong, China; E-mail: mcyuen@hksyu.edu.

uniquely different from the other existing work, we are trying to develop a system to have unique features like showing trends by using data visualization and can showcase flexible outcomes with the same element as the user input rather than the other system to have the same output every time. We may consider choosing one of the online game platforms for datasets. We will scrap the data including the basic information, categories and system requirements to develop our game recommendation system.

The organization of this paper is as follows. Section 2 presents the related work. Section 3 describes our game recommendation system with the data analytical framework. Section 4 shows the experimental result analysis. Section 5 draws out the conclusion.

2. Related Work

There are a number of existing works. Content-based filtering and collaborative filtering are usually used for recommendation systems [6, 7, 8]. In the following, we describe some existing works and highlight the improvement in our game recommendation system.

1) *Games Finder* - The website is one of the databases of online games (Games Finder) [9]. You can click the game icon. Then, it will display other similar games. However, it does not have any data visualization. You can input your favorite game on the website. Our model also has the function to get recommendations which use content-based filtering. Moreover, we have designed a dashboard to show distribution of reviews in different genres.
2) *Quantic Foundry* - By entering 3 game titles, this website will show a list of games that are similar with those you enter, and the platform you can buy the game from [10]. However, the input items are quite simple and without data visualization. Our system adopts data visualization techniques to enhance customer experiences.
3) *Deep Visual Semantic Multimedia Recommendation Systems (D_VSMR)* –The proposed approach employs content-based techniques to expand users' profiles based on the visual content of games [11]. However, the features extracted by the system might not be equally representative for all users.
4) *Social-aware Contextualized Graph Neural Recommender System (SCGRec)* – It proposed using user personalized data (such as social connections) to improve the game recommendation [12]. However, due to the high complexity, frequent update on social media data in the system is not possible.

3. Our System

3.1. Overview

Our first step is to apply web scraping on STEAM (A video game digital distribution platform), the scraped data will be stored in a .JSON file. Then, the data will be cleaned and used for data visualization. After that, content-based filtering and collaborative filtering may be used for designing the recommendation system. Finally, the Flask application will be used for demonstration of our final prototype.

3.2. System Architecture

Figure 1 shows the system architecture of our game recommendation system.

1. Build a model with data
   - *MongoDB Database* - MongoDB is a source-available cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with optional schemas. MongoDB is developed by MongoDB Inc. and licensed under the Server-Side Public License (SSPL).
   - *Machine Learning Algorithms (Content-based filtering)* – We use both TF-IDF and Cosine similarity in our system. We use Python to implement our system.
2. UI Client
   - *Web Application (Flask)* - We use Flask API to run our recommendation system, Data Visualization and the basic function of a client.
   - *Server*

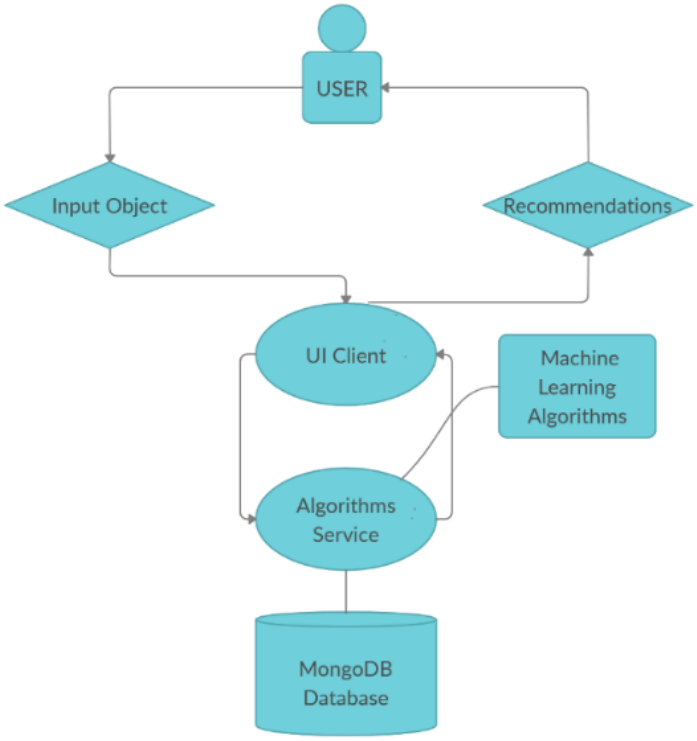

Figure 1. The system architecture of our game recommendation system.

3.3. Data Collection and Preparation

To acquiring data, we have created 3 programs to apply web scraping on the STEAM platform to acquire data into datasets. Three of them are URLs of the game, game information and comments from some games.

1. *Scraping all the URLs of the video games*. The input and the output are as shown in Figures 2 and 3. Figure 2 shows the list of games on the webpage we want to scrap. We need to scroll the page with "Page Down" button many times if we want to see all the games. The code in Figure 3 shows we used the web driver from selenium to scroll the page. Selenium is used mostly in this program. After that, we need to find the URLs of all the games by XPath and store them into a list.

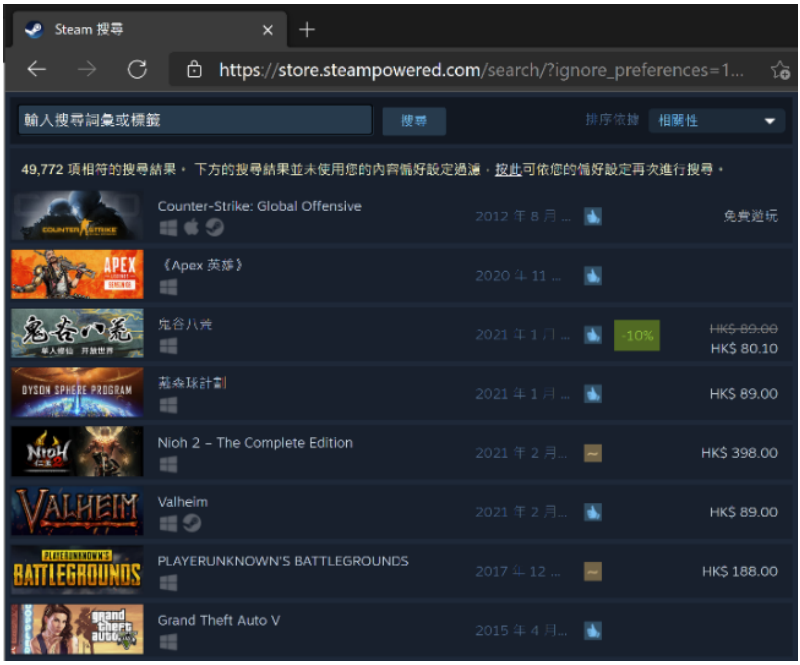

Figure 2. The input of scraping all the URLs of the video games.

```
    for x in tqdm(range(20000)):    # for loop is created to scroll to the bottom of the page
        driver.execute_script("window.scrollBy(0,document.body.scrollHeight)")    # Scrolling the page by 20000 times

  0%|          | 61/20000 [00:00<01:10, 283.65it/s]
The Webdriver is scrolling ...
100%|          | 20000/20000 [46:54<00:00,  7.11it/s]
```

Figure 3. The output of scraping all the URLs of the video games.

2. *Scraping the information of the games*. By reading the .pkl file, which have been mentioned above, we have to scrap the information of the game that we need. Figure 4 shows a website of one of the games. Useful data for content-based filtering have been scraped such as Game Title, Genre, User Tags, Overview and so on. A single game data will be stored in a dictionary and all the data we acquire will be stored into

"game_data" which is a list. All of the data is exported into a .JSON file (output_2F.JSON). To speed up the process of scraping the game information, we found that we can scrap the information by running eight programs at the same time. This has benefits to update our datasets more frequently.

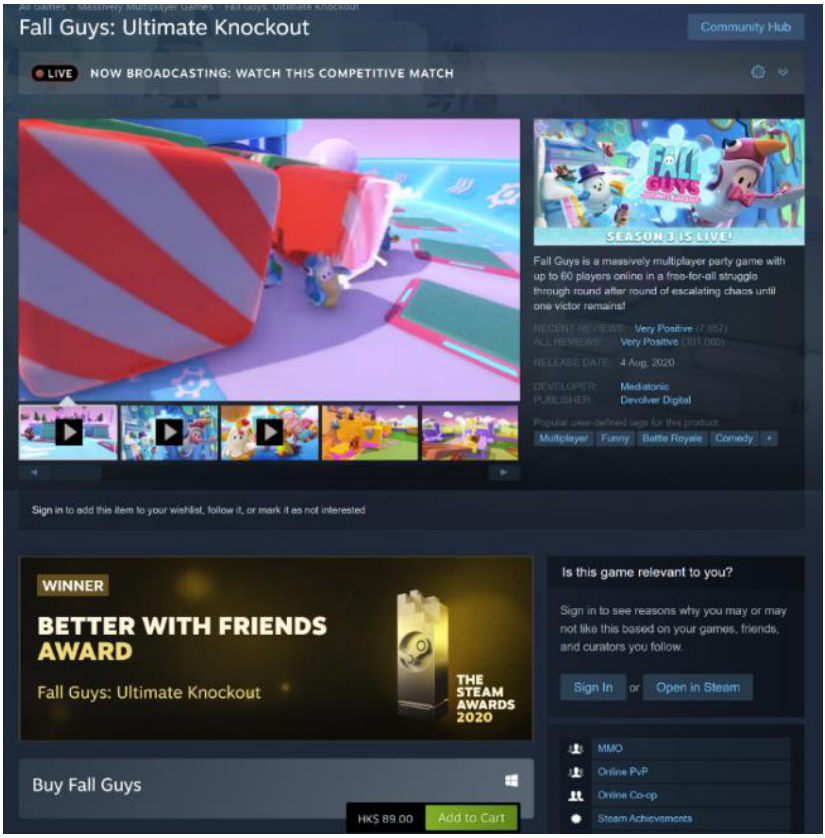

Figure 4: A website of one of the games with useful data for content-based filtering.

3. *Scraping the comments of the games*. The scraped data in this part is for building the recommendation system using content-based filtering. We are not using collaborative filtering because we found out that the scrapping time will increase sharply when scraping loads the comments in each game. If we scrap less comments in each game, it is hard to find the games which have positive comments from the same player. It is because the STEAM platform has many players. Figure 5 shows one of the websites about the game's comments. Three types of data are scraped, the id of the game, users' id which is highlighted in yellow color and the comments which are highlighted in green color. By getting the game id from output_1F.pkl, we have to change all the URLs to the comments page. Like scraping the URLs of the game, we need to scroll down the page to try to scrap the greatest number of comments. A JSON file will be created which stores the scraped comments.

4. *Data Preparation*. Some of the values will be modified during web scraping. For example, to make the analysis process easier, '()' and ',' are replaced by empty space so that the data type of this information is integer.

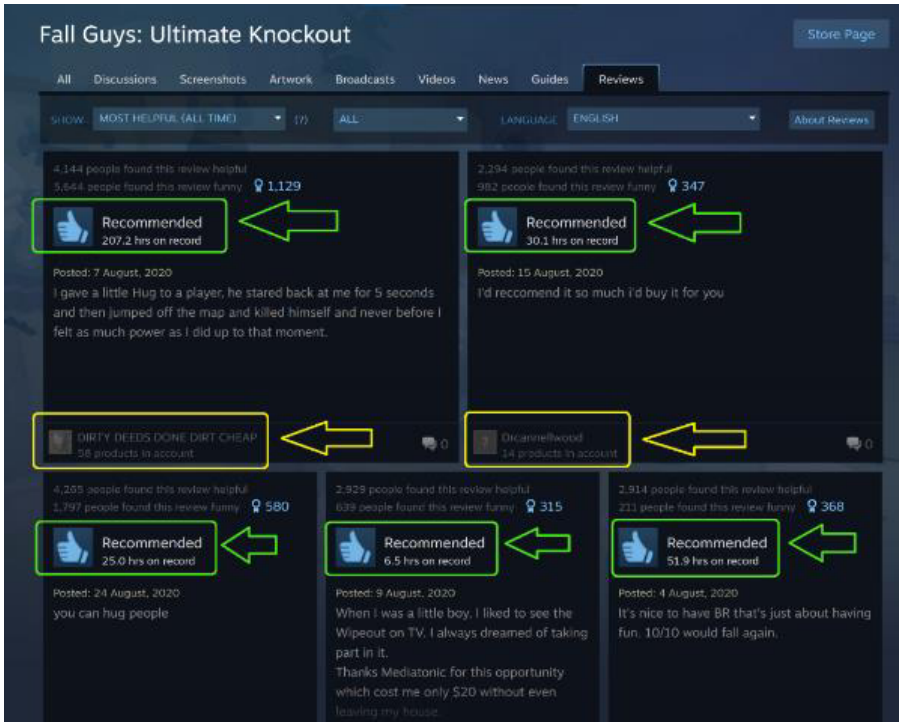

Figure 5. One of the websites about the game's comments.

3.4. Data Modelling / Algorithm

In this model, we decided not to split data into training parts and testing. As the content-based filtering is calculating their similar value by using matrix. We use **cosine similarity** as a model to build the game recommender system with Python. **cosine similarity** measures the similarity between two vectors of an inner product space. It is measured by the cosine of the angle between two vectors and determines whether two vectors are pointing in roughly the same direction.

```
from sklearn.metrics.pairwise import cosine_similarity
cosine_sim = cosine_similarity(tfidf_matrix, tfidf_matrix)
```

Compute **Term Frequency-Inverse Document Frequency** (TF-IDF) vectors for each document. TF-IDF is the frequency of a word occurring in a document, down-weighted by the number of documents in which it occurs. This is done to reduce the

importance of words that frequently occur in plot overviews and, therefore, their significance in computing the final similarity score.

```
from sklearn.feature_extraction.text import TfidfVectorizer
```

```
tfidf_matrix = tfidf.fit_transform(df['ABOUT THE GAME'])
# Construct the required TF-IDF matrix by fitting and transforming the data
```

3.5. Data Presentation

Figure 6 shows the sunburst chart with no selection input. Figure 7 shows the sunburst chart with the 'Very Positive' cell in 'Recent Reviews' path selected. Figure 8 shows the sunburst chart with the 'Action' cell in 'Genre' path selected AFTER 'Very Positive' cell in 'Recent Reviews' path was selected.

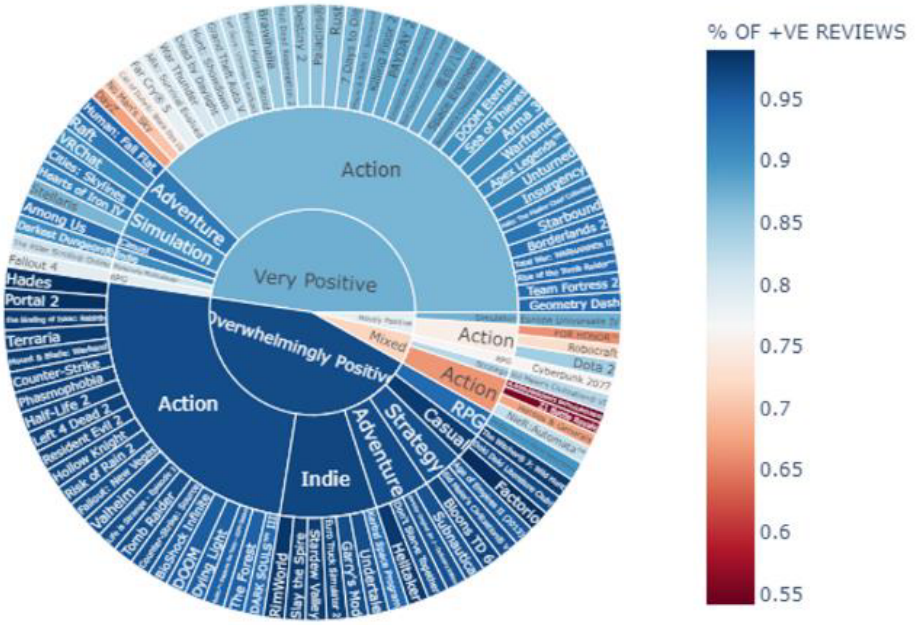

Figure 6. The sunburst chart with no selection inputted.

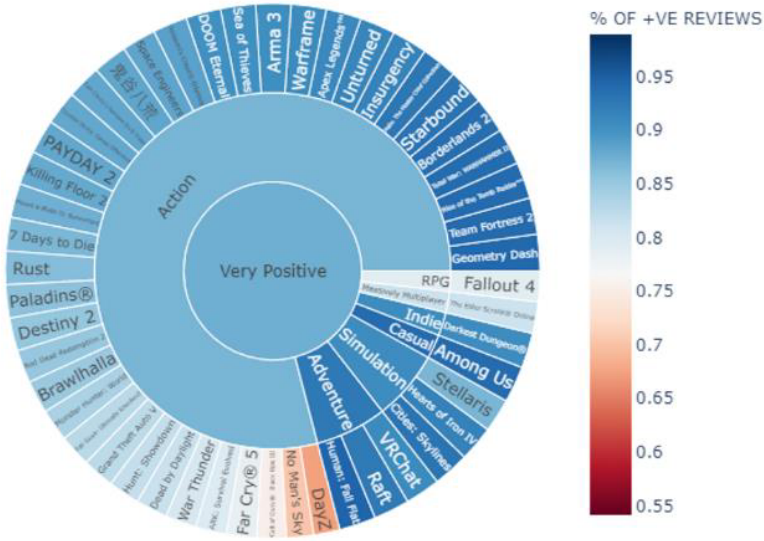

Figure 7. The sunburst chart with the 'Very Positive' cell in 'Recent Reviews' path selected.

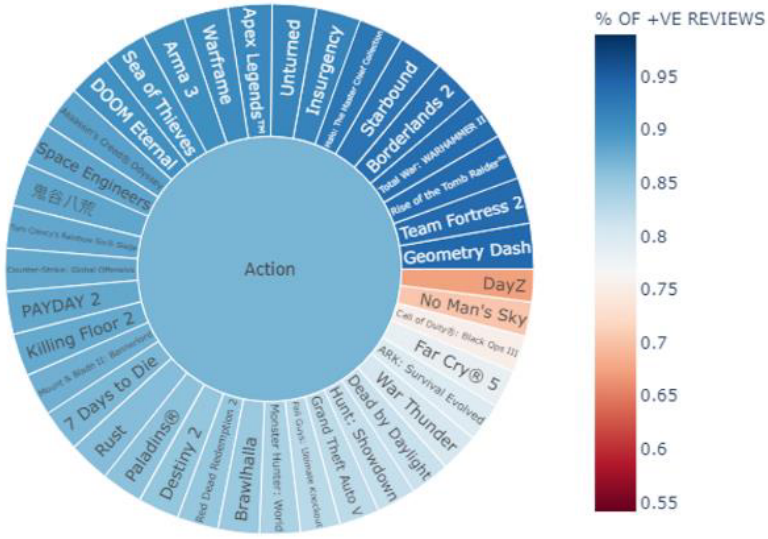

Figure 8. The sunburst chart with the 'Action' cell in 'Genre' path selected AFTER 'Very Positive' cell in 'Recent Reviews' path was selected.

We design and develop interactive dashboards by using Python and Tableau. Figure 9 shows the dashboard that is built with four basic graphs with the data we obtained. Figure 10 shows the dashboard with one cell selected. Bottom left shows the top-rated game which matches the element selected in that graph. Figure 11 shows the dashboard with two cells selected. Bottom left shows the top-rated game which matches the elements selected in those graphs. Figure 12 shows the dashboard with only the game's cell selected.

In this project, we used the Flask framework to run our recommendation system. It includes the basic UI for user input and out and some data visualization charts. Our business value is how much profit can be earned in the game selling. If the company wants to launch a first-person shooting (FPS) game in the STEAM, we can help this game evaluate whether it can make money. We can use the dashboard to check whether the FPS game is one of the most popular games which is played by many people. The company can analyze the dashboard and have a better consideration on launching the games.

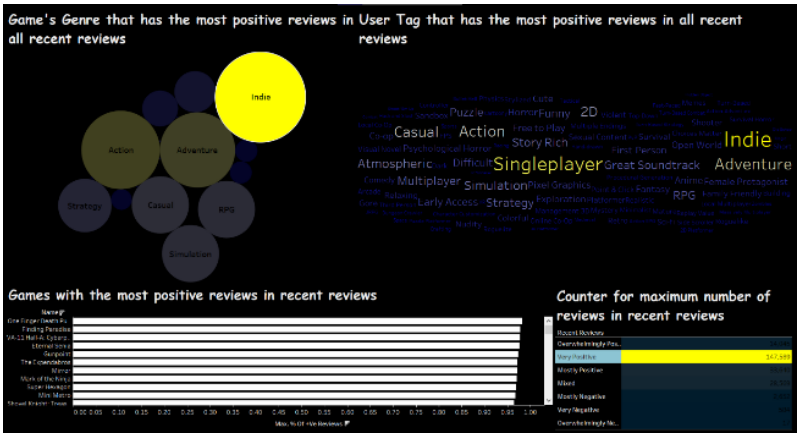

Figure 9. The dashboard.

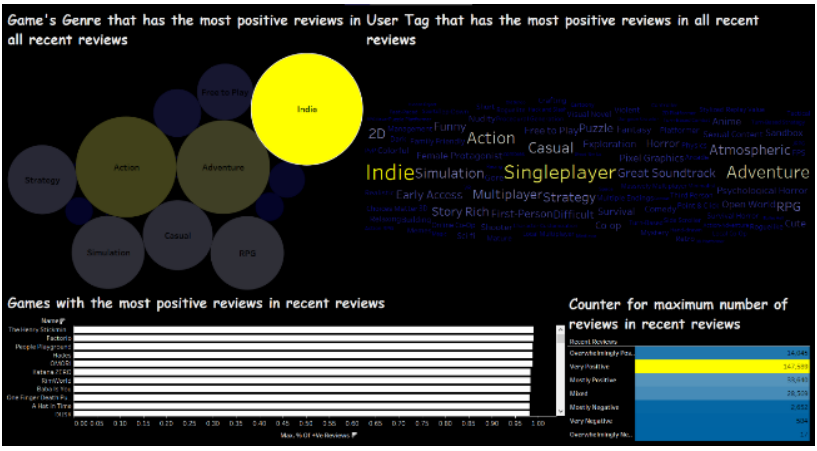

Figure 10. The dashboard with one cell selected.

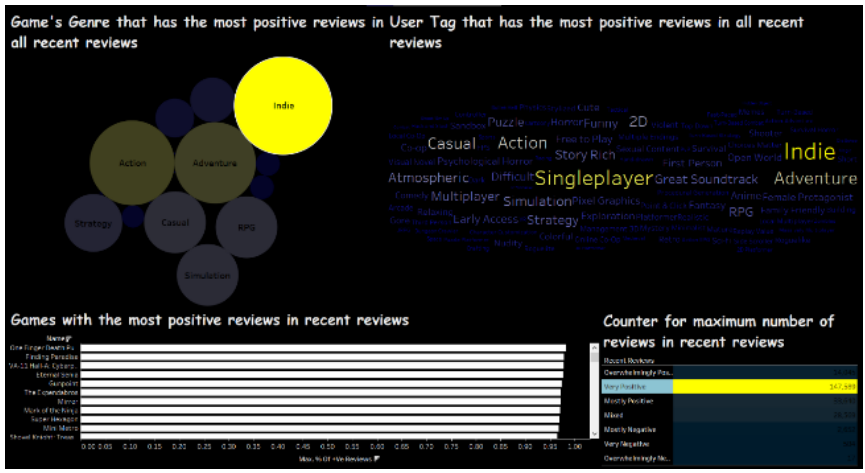

Figure 11. The dashboard with two cells selected.

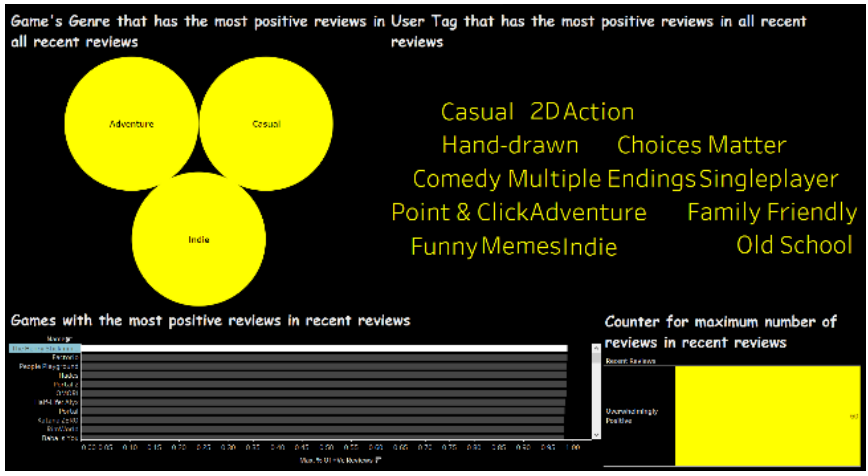

Figure 12. The dashboard with only game's cell selected.

3.6. Website

3.6.1 System Design

A website is created on localhost server. By running the program of the Flask framework, the browser will be opened and go to the home page which is mentioned in the following part. HTML, CSS are used to create the content and layout.

3.6.2 *Interface Design*

In our website, three interfaces are designed as shown in Figure 13. Figure 14 shows the home page of our game recommendation system. The home page provides a simple background and mainly in blue color. A search box is created for the users to input the full name of games. By clicking the submission button, the users page shows the recommended results or the page which shows the results are not found. A bottom left container shows our summary of the website. Figures 15 and 16 show the sunburst chart by scrolling down the home page. By scrolling down the page, you can see the sunburst chart which shows the percentage of positive reviews by recent reviews as shown in Figure 15. The chart is interactive to users by clicking the genre of the game and the recent review like the figure shown in Figure 16. Figure 16 shows the sunburst chart after a user selected the 'Very Positive' in the 'Genre' path. By selecting this path, it shows all the games and its genre which has a 'Very Positive' recent review in overall review.

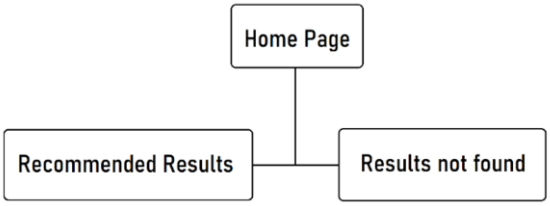

Figure 13. Three interfaces in our website.

Figure 14. Home page of our game recommendation system.

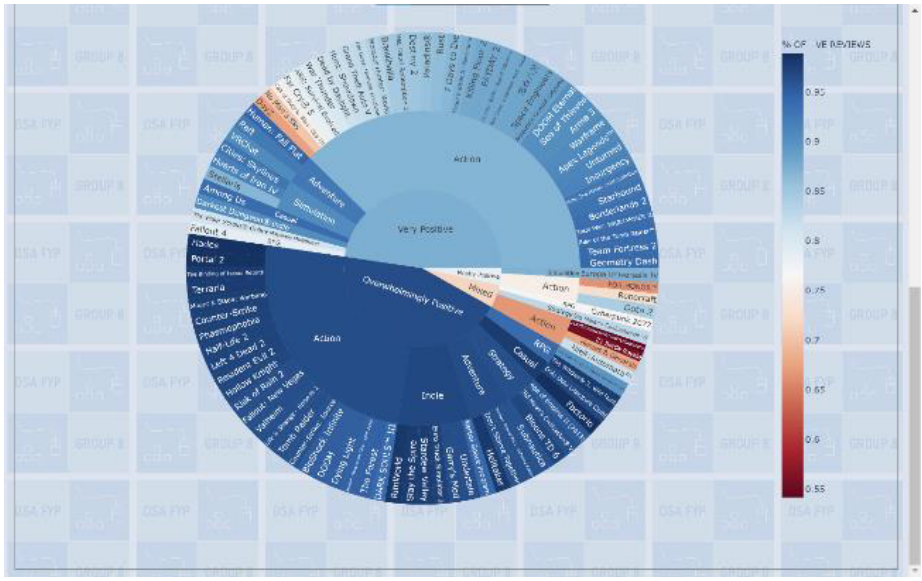

Figure 15. The sunburst chart which shows the percentage of positive reviews by recent reviews.

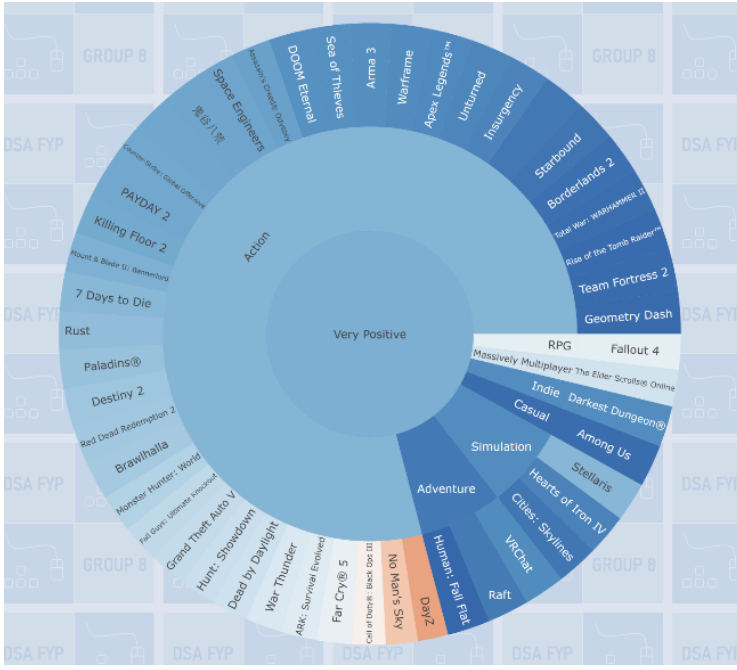

Figure 16. The chart is interactive to users by clicking the genre of the game and the recent review.

4. Preliminary Result on Performance Evaluation

4.1. Performance Metrics

The model can recommend games with the same taste as the user based on the game's attribute information. For example, the user chooses “Counter-Strike: Global Offensive” (objective-based, multiplayer first-person shooter) as input data of the model.

There is a similarity score to compare similarities between different games and “Counter-Strike: Global Offensive”. If the similarity of two games is high, the score is closer to 1. We expect our model to accurately recommend a similar game according to the similarity score (1).

$$similarity(A,B) = \frac{A \cdot B}{\|A\| \times \|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \times \sqrt{\sum_{i=1}^{n} B_i^2}} \tag{1}$$

Figure 17 shows the similarity score to compare similarities between different games and “Counter-Strike: Global Offensive”. If the score is closer to 1, it is more similar between two games.

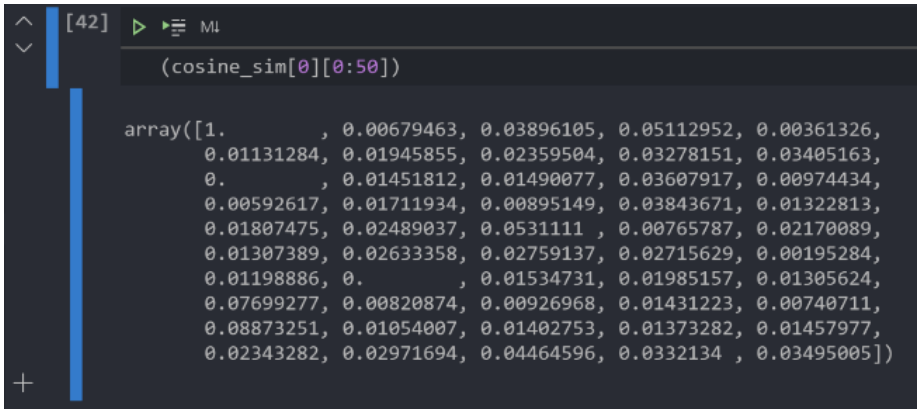

Figure 17. The similarity score.

4.2. Preliminary Results

In Figure 18, the page of recommendation results provides a simple background and mainly in purple color. ‘Cities: Skylines’ is chosen as an example; it will show the top five results related to ‘Cities: Skylines’ with their genres. And display like a food menu. The link of the title of the game can be clicked to go to the website from STEAM to check further details. The page displays the error message if the recommended results are not found.

Figure 18. The page of recommendation results.

5. Conclusion

In this project, we have developed a games recommender system with complete functional recommending features. We applied data visualization to our solution and can successfully make the recommending system more appealing to the user, which completes our goal of making a recommending system that feels more refreshing to the existing one. Also, we achieve our goal of making a system that can showcase flexible outcomes with the same element as the user input, which can give the user more choice when finding the game they want.

We successfully apply machine learning, data science skills such as web scraping, data preparation, machine learning algorithms, data visualization and Flask application and build a working recommendation system, these skills will surely help us furthermore in the future with our data science work. And we hope that our recommender system can successfully help those people who want to find their favorite games.

References

[1] Cheuque, G., Guzmán, J., & Parra, D. (2019, May). Recommender systems for Online video game platforms: The case of STEAM. In Companion Proceedings of The 2019 World Wide Web Conference (pp. 763-771).

[2] Hannula, R., Nikkilä, A., & Stefanidis, K. (2019, September). GameRecs: Video Games Group Recommendations. In European Conference on Advances in Databases and Information Systems (pp. 513-524). Springer, Cham.

[3] Pathak, A., Gupta, K., & McAuley, J. (2017, August). Generating and personalizing bundle recommendations on steam. In Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 1073-1076).

[4] Yuen M. C., Chan S. L., Leung H. T., Wu P. L., Yip P. Y. (2019). "A System for Collecting and Analyzing Data from Existing Game Selling Platforms", Proceedings of The 2019 15th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD 2019), July 2019, Kunming, China.
[5] Dokoupil, P., & Peska, L. (2023, June). The Effect of Similarity Metric and Group Size on Outlier Selection & Satisfaction in Group Recommender Systems. In Adjunct Proceedings of the 31st ACM Conference on User Modeling, Adaptation and Personalization (pp. 296-301).
[6] Yang S., Korayem M., AlJadda K., Grainger T., Natarajan S. (2017). Combining content-based and collaborative filtering for job recommendation system: A cost-sensitive statistical relational learning approach, Knowl.-Based Syst. 136 (2017) 37–45.
[7] Yuen M. C., King I., Leung K. S. (2015) "TaskRec: A Task Recommendation Framework in Crowdsourcing Systems", Neural Processing Letters Volume 41 Number 2, 2015.
[8] Yuen M. C., King I., Leung K. S. (2021) "Temporal Context-Aware Task Recommendation in Crowdsourcing Systems", Knowledge-Based Systems, Elsevier.
[9] Games Finder. https://gameslikefinder.com/
[10] Quantic Foundry. https://apps.quanticfoundry.com/recommendations/gamerprofile/videogame/
[11] Ikram F., Farooq H., and Nawaz W.. (2022). Multimedia Recommendation System for Video Game Based on High-Level Visual Semantic Features. Sci. Program. 2022 (2022). https://doi.org/10.1155/2022/6084363
[12] Yang L., Liu Z., Wang Y., Wang C., Fan Z., and Yu P. S. (2022). Large-scale Personalized Video Game Recommendation via Social-aware Contextualized Graph Neural Network. In Proceedings of the ACM Web Conference 2022 (WWW '22). Association for Computing Machinery, New York, NY, USA, 3376–3386. https://doi.org/10.1145/3485447.3512273