Recommend a Steam Game to User

University of California, San Diego
CSE 158A

Introduction

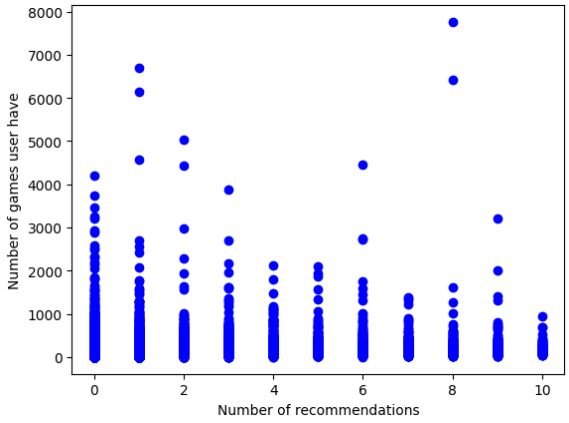
Within this evolving world, there a piece of entertainment that evolved with the times is the idea of games. Games are enjoyable "tasks" that people would indulge in to pass their time. However, games have gradually advanced from a stick and wheel to that of technologically innovations. Nowadays, when we hear "games" we think of "videogames" or digital games, where one would have a console/device that would allow them to interact with a digital world where they indulge in their games. One of the more popular platforms to play games is known as Steam, where users can digitally purchase games and launch it through Steam. When browsing through Steam, there are many chances where user would come across recommendations of high compatibilities to the games that the user is playing. The recommendations within this platform sparked the objective of this paper, in which it is to see if we can come up with an imitation (or better) predictive model for recommending games to a user. I will be using the Steam datasets from Professor McAuley's datasets in my quest to derive an accurate game prediction model.

Dataset Overview

Looking into the scope of the Steam dataset, I will mostly be using the "Version 1: Review Data", "Version 1: User and Item Data", and "Version 2: Item metadata". The data within the "Review Data" dataset will provide me with all reviews that a user have given, and if they would recommend the game that they'd reviewed. This dataset will contain 25799 user data. The "User and Item Data" dataset contains basic user properties, and a list of games that the user own. Within the list, each game will have basic identification info, an amount of playtime in hours for the entirety of owning the game and the amount within a 2-week span. This dataset has a size of 60000 (though there is more, I'd placed a cap on it). The "Item metadata" dataset will allow me to grab all necessary information about a game, such as the title, genre(s), tag(s), publisher, developer, price/discounted price, and if the game is in early development. The dataset has a size of 32135. A keynote that I'd identified when looking at the datasets is that user can buy and use a software from Steam as well, not just games. Additionally, there are some data objects within the listed datasets where some attributes may be missing. (e.g., "Price": TBD)

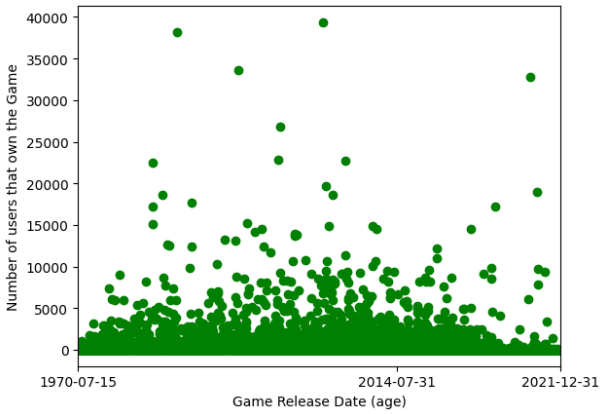
Exploratory Analysis

An interesting finding that I've seen when looking through reviews that users would give is that the maximum recommendations that a user would give is 10, no matter how much games they own. When comparing the number of games that a user own vs how much recommendations a user would give, most users tend to not give recommendations even if they have more than 4,000 games.

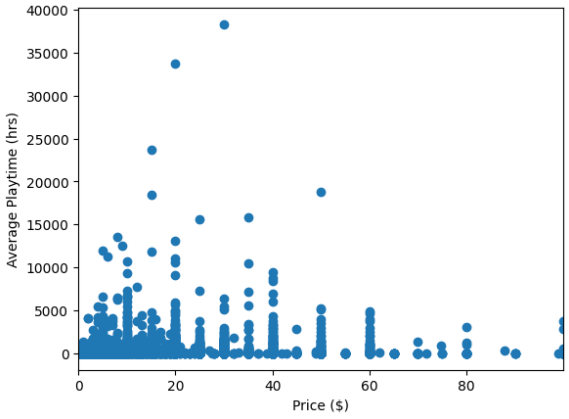


Furthermore, as the numbers of games a user own decrease, the number of recommendations would increase. This inverse relationship provides us with the idea that users are very keen, but sensitive when giving recommendations and that most people just play games, without bothering to give reviews/recommendations. Another interesting finding was when I measured the release date of a game to the number of users that would own that game. For this comparison, I eliminated any

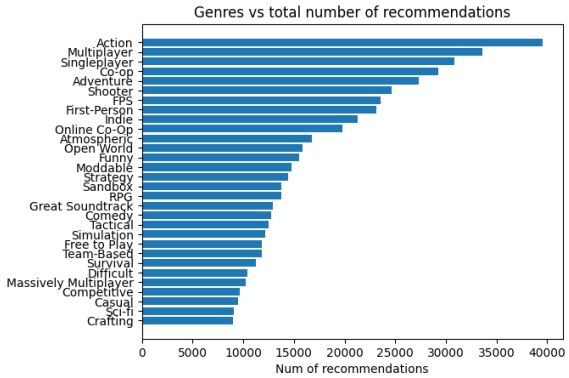
games that did not have a release date or that the format of the date was not in accordance with "yyyy-mm-dd". (Some date values were "Oct 2021" or "TBD")



It turns out that most users owned games that had a release date older than the year 2014. While a sparse number of users owned more recently released games. This comparison does not consider the game's pricing (free vs $59.99 vs discounted prices), but it helps me understand that ages of games does not define its potential to be bought. In continuation of the data analysis, I started to wonder if the pricing of a game affects the amount of playtime a user would have on the game. I've always assumed that many people would put down a lot more time on a game that they've spent a lot more money on.



Surprisingly, this graph illustrates the idea that most people would buy "affordable" games and put a lot of time in it, while others who spent more than $60 did not play their game as much. (Personally, witnessing a playtime of over 5,000+ hours are unheard of, but I am not surprised). It can be concluded from this graph that the lower/reasonably priced games are the ones that get the most playing time. Games below $20 tend to have a higher play-rate than other games priced at $20+. As I dug deeper into the properties of games, I wanted to see which genres have one of the highest # of recommendations. The graph shown below will contain the top 30 game genres that users have given recommendations to. Some games have multiple genres, in which if the games were recommended then the count would increase by one for each genre.



The graph above demonstrates that action and adventure are one of the top 5 unique genres that received the highest recommendations. The tags of multiplayer, single player, and co-op describes the unique features of the game, where a user would enjoy playing with others or just by themselves. There is a slight, but greater number of users that prefer to play multiplayer games. Given the ideas above, it can be assumed that if a game's genre is action, then most users would play and recommend it.

Predictive Task

The predictive task that I will be solving is if I can build a model that would be able to recommend certain games for user to buy/play. First, I'm planning to build a model that uses the similarity metrics where it'll compare a set of users for a game to another set of users for each game within the dataset that the user does not own. If there does not exist a similarity between

the items, then the user would be recommended a game based on the top genre that the user play and the number of recommendations the game has. These recommendations will be done based on the first half of game library, while the second half will be used to validate whether the recommendations were accurate. This means that I will be predicting/recommending a game based on the games in the first half of a user's library and verifying whether the prediction is a part of the second half of the user's game library. Within my model I will be incorporating the features of a game's genre, the amount of playtime user for the genre, the number of reviews a game has, the number of recommended reviews a game has, and the price of the game. A baseline that I will be using is recommending a game based on the highest number of recommended reviews for the most popular genre in the user's library. For the baseline, I first had to create a data structure that kept track of all genres of games reviewed in the "Version 1: Review Data" and count the number of times that game has been reviewed. A requirement for the count is that the reviewed game ID must exist in "Version 2: Item metadata", since going through this dataset is the only way to get the genre of the game. An issue that I've ran into was that some games may not have genre, tags, or both. To fix this issue, I simply ignore those games. I can also apply another filter where we count only the reviews that recommends a game. I'd decided not to apply the filter and got a data structure containing the most reviewed game for a particular genre. I then created a data structure that kept track of all games that user own(gamesPerUser) and iterate through the first half of this structure to find the top game genre for all games in the user's library. After finding the top genre for the user, I will recommend to the user the top reviewed game for that genre. If the top reviewed game exists in the first half of the game library for user, then I will use the most recommended game for the next top genre. An edge case that was considered is if the user does not own any games, in which the most owned game by user would be recommended. I then validated the recommendation to that of the second half the "gamesPerUser" data to validate the recommendation, in which I got an accuracy of only 20%. A failed attempt that I'd made for the base model was applying the idea of using next most reviewed game for the user's top game genre. This reduced my accuracy to 7.5%. Additionally, I tried changing the data structure from getting the top reviewed game to the top reviewed AND recommended game for the specific genre. This did not seem to help the baseline model, since accuracy was reduced to 4.2%.

Model

When attempting to use Jaccard similarity, I ran across a major issue with time complexity(overhead). The unsupervised model was running extremely slow. Not only was it slow, but the accuracy was also significantly smaller, less than 1% accuracy, than the baseline model. I tried using different similarity functions like Cosine similarity and Pearson correlation, but this did not solve the issue. I started to look around the web on how other methods of recommending games, specifically on how Steam games are recommended. Most of the articles that I've encountered involved using Bayesian ranking to recommend Steam games, so I decided to choose the Bayesian ranking model. As it turned out, after implementing the Bayesian ranking model (using the Tensorflow code from Chapter 5 workbook) I was able to successfully make somewhat accurate predictions of whether a game should be recommended to a user or not. The model follows the equation from lecture when deriving the score of each game compared.

$$\max \ln \sigma(\gamma_u \cdot \gamma_i - \gamma_u \cdot \gamma_j)$$

After testing the model with my test data, I was able to get an overall accuracy of about 71%. Initially, I had issues with the scalability of the dataset, which led me to split the dataset for training and testing. I chose this model because compared to using the similarity function, this

ran much smoother, and the recommendations were more accurate. I did consider other models such as the Latent Factor model, but I wasn't sure how predicting a real value would help me make a recommendation, since a user review would involve a binary recommendation value. I also considered a regression model, but the size of a data entry would be extremely big due to the large number of features, which would then cause an overhead. Using a Bayesian ranking model makes more sense, due to its efficiency and simplicity. The logistic regression model will be good for incorporating features to the prediction, but there were a lot of overhead when using this model. Using the similarity function as a recommender made the model very simple, but again the overhead kept it from running well. The Bayesian ranking model on the other hand is good for binary predictions and is efficient. Since Bayesian ranking focuses only on the interactions between user and game/item, it does not consider the features of the game and type of games that a user would enjoy. I would optimize this model by incorporating the features of the games (price, genres, tags, publisher) when preselecting the games that COULD be recommended to the user. To do so, I would need to find out for each user their favorite genre and the "depth" of their wallet (Avg amount of money they would spend on a game). Their favorite genre will depend on the type of games they have in their Steam library, if they'd made a recommended review for a game, and the amount of playtime they have on the game. A higher weight will be associated with the games that were recommended by user. The higher the playtime on a game, the higher the weight for the game. This will help evaluate which genre is the user's favorite genre. Then we can evaluate the average price that a user would pay for a game based on the games in their Steam library. This average will serve as the maximum price, and with the user's favorite genre in mind, games can be selected based on the genre and price. This would then represent a list of potential games that COULD be recommended to a user would be fed to the Bayesian model. From there, the Bayesian ranking model will give a score for each game and once sorted the first game should be the top recommendation to be given to the user based on their preferences.

Literature

I used the datasets given from the professor's website. To reiterate from the dataset overview, I used 3 datasets, one was for each user's library, each user's reviews, and each game properties. For the prediction/recommendation task, I used the "Version 1: User and Item Data" as the interaction data (user, game pairs). The datasets containing the reviews and game metadata were used to pull out features for each user and each game. One literature "Steam Recommendation Systems", written by Albert Yefeng Liang used a dataset from an API called Steam spy. From the article, this dataset was used to identify user's preference, game popularity, similarity between games based on description and the quality of the game. The popularity of a game is based on the number of users that own the game. The quality of the game is taken based on the number of positive reviews, in which this property was not include in my own dataset to use. A technique that was used to study this type of data was making HTTP requests to view the description of each game. A TF-IDF model was used on the description of the game and a linear-kernel was used to compute a "game's descriptions TF-IDF weights with other game's descriptions TF-IDF weights" (Liang, Albert). A similar conclusion from existing work to my own finding is using the playtime as a means of "rating" since user can only make a binary choice of "recommend" or "not recommended". Furthermore, the algorithm/model that was used in this literature's recommendation was the Spark ALS algorithm. Although Spark ALS differs from Bayesian Ranking model, it still follows the idea of scoring a game and then sorting the ranked games in descending order. The pre-evaluation of data from this article differs from my own. Another literature that approached the Steam recommendation problem

was “Video Game Recommendation System”, written by Robert R. In this article, Robert used centered cosine similarity to make recommendations. Like the first article, Robert used the playtime data of a user as a substitute for rating. The dataset that he used was from Kaggle. One generic conclusion that was brought to Robert’s attention when doing an analysis on the data was that some games would have a very large playtime for one to few individuals, although those games weren’t popular amongst the majority of user who played it for “less than a few hours”. (Robert, R.) With the modified data based on popularity, Robert found correlations between games using Sklearn tools. It seems like a simple model was implemented with no state-of-the-art methods, but the conclusions were very fair and accurate.

Conclusion & Results

When testing the Bayesian model, I fed the model with game data for each user where 50% were games that they own that were not used in training and 50% were games that they do not own. The result that I got from this test was that about 71% of the overall recommendations were correct for all users (before optimizations). An alternative model was using the baseline model (unsupervised), in which my current model dominates over it. The significance of the results is that the Bayesian model could give reasonable scores to games that had potential to be recommended for a user. Given unseen data to the model does not negatively affect it. Using the review data, price of game, average user’s playtime, and game genres/tags worked very well, in which it optimized the data prior to feeding it into the model. The other properties of game’s did not work well, such as “Publishers” and “Developers”. The model’s parameters are the relationship between a user and two games (whether one is more positive than the other). The proposed model succeeds because I was able to modified the training data, and effectively utilize the Bayesian Ranking code from the class’s workbook to fit with my recommendation process. Others failed due to scaling and bad representation of user-game interactions. All in all, the Bayesian Ranking model was used for this assignment, and it was able to give me a sufficient accuracy.

Works Cited

Liang, A. (2021, December 15). *Steam Recommendation Systems - Towards Data Science*. Medium. https://towardsdatascience.com/steam-recommendation-systems-4358917288eb

R., R. (2022, January 6). *Video Game Recommendation System - Web Mining [IS688, Spring 2021]*. Medium. https://medium.com/web-mining-is688-spring-2021/video-game-recommendation-system-b9bcb306bf16