BrainStation Data Science Diploma

MyGamePass
Ben Polzin Capstone Report

Introduction

Growing up I always loved video games and played as many great games as I could. As I've grown older, however, life has become more demanding and I just don't have as much time to dedicate to my hobbies as I once did. Between work, family, and friends, time is limited when it comes to entertainment so it is important that time is not wasted. Today there are an estimated 1.2 Million unique video games across all platforms, and that number continues to grow every year. With so much variety of options the amount of choice can be paralyzing. So I asked myself, how might I use data science to find my next favorite game?

Using machine learning and natural language processing, I set out to solve this challenge by developing a recommender system focused on the gamers and their games. Content based filtering and collaborative based filtering were used to recommend games based on descriptive features or user playing habits. This project provides a great highlight of data analysis fundamentals as well as an understanding of more advanced concepts such as Machine Learning Pipelines as well as Natural Language Processing, Funk Singular Vector Decomposition, Cosine Similarity, and model evaluation. Packages demonstrated include pandas, matplotlib, seaborn, sklearn, nltk, and surprise.

Data Gathering and Preprocessing

Steam is the primary digital marketplace for computer video games. A dataset of 27,000 video games from the steam store was acquired from Kaggle as well as a dataset of 200,000 user-game interactions. The initial Steam Store dataset was used as the master dataframe that the project was built around. The steam.csv file contained the "IMDB posting" for each video game listed - data included unique id, name, platforms, developer, publisher, categories, genres, tags, description, ratings, sales, and language. This is excellent data for a content based recommender system. Although the dataset provided was thorough, clean (no duplicates, no nulls) and well-packaged it required extensive preprocessing to be used for modeling.

I began by calculating the percentage of positive reviews and the total review count. This information is critical and provided the backbone of my thresholds for refining recommendations. Additionally, the sales information was vague and reported as a range which was modified and binarized into buckets. Similarly, the platforms (mac/linux/pc) and categories were converted into numeric columns with 'get dummies'.   The dataset was then filtered to only include English games.

The biggest challenge in preprocessing was handling the average hours played column. I first visualized the distribution and realized quickly significant outliers were impacting the data. Using information taken from the data itself and my own domain knowledge I set a maximum cut-off and hard coded any value above that number to the cut-off to eliminate the extreme outliers. Lastly, there were 20,000 games that showed an average playtime of less than 1 hour. To handle this, I chose to remove the games that had an average playtime of less than 1 hour and were deemed "unworthy" based on their ratings (either negative reviews, or no reviews). The prepared master steam data was exported for modeling.

The user data required some modification as well. The dataset included the unique id for each user, the video game and a column with an entry for purchase or play and the hours played. Because it is implied the user purchased the game if they played it, that was removed. Additionally the count of total games played per user was calculated and added to each row, and the count of total users per game.   At this point I chose to filter out the least popular games, as a number of games had only been played by 1 or 2 people and would not be useful in this model. The hours played were converted to a rating scale to be used for the surprise package. The hours played contained significant outliers in this dataset as well, however these were 2.5% of the dataset and were removed.   From here, the hours played were put into 5 equal bins using pandas qcut function to be converted to a rating from 1-5 based on hours played. A conversion to 1-10 rating scale was also performed to test the model. Using these bins, the ratings were added to the dataframe and the prepared data was exported for modeling.

Modeling and Evaluation

Growing up, I loved going to the video game store and when a new game would catch my eye the only thing I had was the information on the back of the box to make my decision. This can be recreated with machine learning and natural language processing and expanded to look at any game in the database, not just what I could find on the shelf as a kid. This is called content based filtering.

I began by simply looking at the highest rated games in the dataset. This is a common "default" recommendation for a cold-start with a new user and helps with our understanding of the data. A minimum total ratings threshold is needed to find the games with a consensus. Humans are notoriously finicky and rarely agree uniformly, particularly on entertainment preferences so it does help to look for a consensus. Additionally, these rating thresholds are used to refine the recommender system based on individual preferences. Some users only play blockbuster hits whereas others are much more likely to search for the lesser known games.

The Natural Language Toolkit (NLTK) was used for Natural Language Processing of the description and non-numeric features of each game. Custom functions were used with tokenizer to remove html tags and prepare the data for the TF_IDF_vectorizer. TF_IDF was chosen as it will put more emphasis on the most unique/descriptive words, not just the most common words in each description.   Words were not counted unless they appeared in 5 or more descriptions, and also thrown out if they appeared in more than 90% of the games descriptions. That will ensure any additional throwaway words that were not included in our stopwords are also ignored.

Once the TF_IDF_matrix was created, a simple cosine similarity is performed to determine the similarity score of each game in the dataset. Using this similarity score, and the rating thresholds we are able to recommend games based on the content of the game alone. This is a great starting point, but there are many opportunities to improve. Topic clustering could be used to help refine the recommendations for a brand new user who has never played video games before, but would like a more personalized recommendation.

Another way I discovered new games as a kid was when my friends would recommend them to me. This can be taken to a new level with machine learning using extensive user data and playing habits. Using the prepared user data, the surprise package was used to perform a FunkSVD to make ratings predictions for every user and every game in the dataset. Matrix Factorization was chosen because an in-memory recommender system is nearly impossible to deploy at scale when talking about hundreds of thousands of games and millions of users.

To begin the Funk Singular Value Decomposition the data needed to be prepared for the surprise package. The surprise package was used for hyperparameter optimization and model evaluation. Using the gridsearch functionality and the evaluation measures the initial model was refined. The Fraction of Concordant Pairs (FCP) was chosen for this model due to the ease of interpretability. A pair is concordant if one user gave a game a high rating and the predicted rating for that game is also high. Once the hyperparameters were optimized, the SVD was used to generate predictions for every user and every game in the dataset. With these predictions, we are able to recommend games to the user based on what similar user profiles have been playing.

A hybrid model would be great to explore where the system finds the top predicted ratings based on similar user profiles, and then also finds the most similar games to those based on content to provide even more great games for a user to enjoy.

Conclusion

The initial goal of this project was to create a system where a user could define their gaming history, select a system and find their new favorite games. As the project evolved, I realized the scope of an individual project compared to a full team at a gaming company and focused on the fundamentals of a recommender system. This work can be iterated upon to refine and improve recommendations, and a particularly beneficial use-case would be to build this tool to focus specifically on parents searching for gift ideas, as well as refine the “cold-start” approach by including Netflix user profiles and a preferences survey to guide recommendations. More detailed user data, demographic data and usage would lead to a vastly improved model. A user who plays only on the weekends may have different preferences than someone who plays every day after work, even if the hours are the same. The possibilities are endless!