CSE 158 - Assignment 2

Cecilia Hong, Tianze Guan, Runchu Xu, Xiaoshuo Yao, Zeqian Min

November 2021

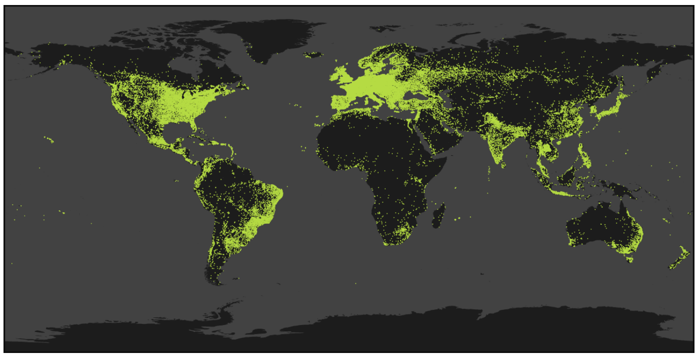


Figure 1: Steam user download distribution 11/30/2021

Abstract−Video games have been an increasingly important form of entertainment since the new millennium. In contrary to the traditional CDs, PC users today are more likely to get their games from online game distributors. Such behavioral change produces a large amount of data. In this article, we made a predictive analysis on the relationship between a user's past behavior and the time he might want to put on a new game.

1 Introduction

As the network infrastructure continuously evolved through the past decades, online application stores have replaced the traditional CDs and became the major way of getting new software. Beside the convenience on the user's side, the data produced by such online software stores, even from after-purchase user behaviors, provide the stores a chance to better serve the users with the software they might like. And game stores (or distributors), which have access to plenty of user behavior data and a high user loyalty, is one of the most prominent areas where user data can be used to recommend future games to the user.

In this article, we focused on a specific online game distributor, which is Steam. There's plenty of data analysis and predictive models proposed on steam user data [1][2][3]. But in our opinion, they did not touch one of the core aspects of user data from a game distributor which is the game time. And we consider this type of data as one of the most important advantages of online game distributors when compared to traditional game stores since obtaining this kind of data is not possible without real-time monitoring user's activity. In order to utilize such data, we proposed a model that take game genre and stats related to game genre along with user's past behavioral data as features to predict a user's game time on a particular game.

2 Data

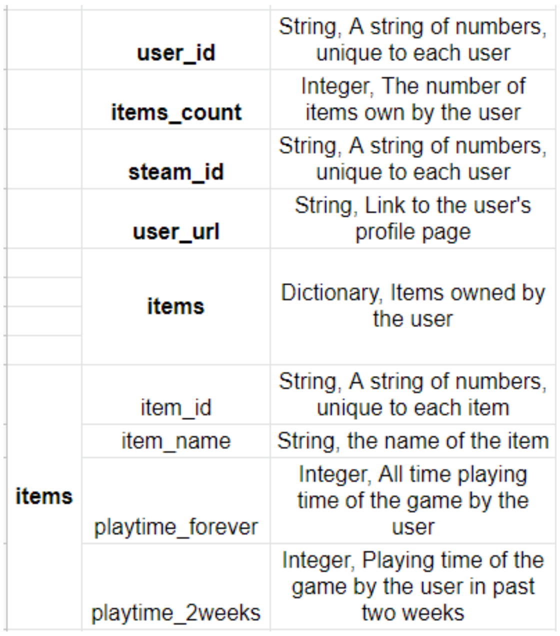
2.1 Dataset

The data set we use comes from professor McAuley's collection of data sets. More specifically, *Version 1: User and Item Data* and *Version 2: Item Metadata*. We also did explorations data analysis on other data sets in the steam data collections but ultimately decided they are not suitable for our task.

2.2 EDA

2.2.1 EDA on User and Item Data

In User and Item Data, there are 88310 unique users and for each user, the data set contains the following information.



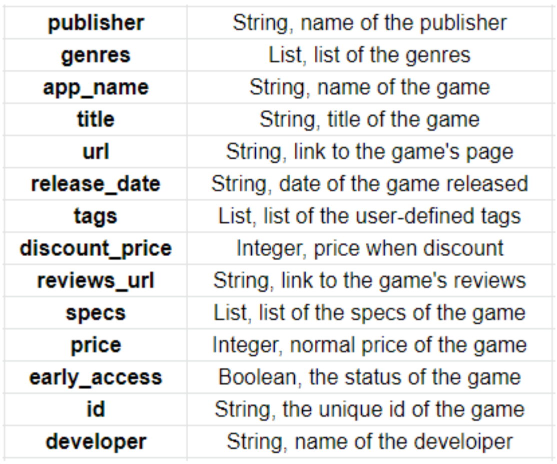
	user_id	String, A string of numbers, unique to each user
	items_count	Integer, The number of items own by the user
	steam_id	String, A string of numbers, unique to each user
	user_url	String, Link to the user's profile page
	items	Dictionary, Items owned by the user
items	item_id	String, A string of numbers, unique to each item
	item_name	String, the name of the item
	playtime_forever	Integer, All time playing time of the game by the user
	playtime_2weeks	Integer, Playing time of the game by the user in past two weeks

Figure 2: Contents in User and Item Data set

In the data set, the user with the longest cumulative gaming time has a gaming time of 4660393 minutes, which means that particular user spent nearly 9 years on gaming. And the average gaming time of the data set is 57857.34 minutes while median sits at 26921.5 minutes. In terms of time spent on individual games, the maximum is 642773 minutes while the mean is at 991 minutes and median is at 34 minutes. Putting all these features together we can see that this data set is extremely skewed. Such screwed data set can bring a lot of challenge when we work on the predictive model.

2.2.2 EDA on Item Metadata

In Item Metadata, there are 32135 unique games and for each games, the data set contains the following information.



publisher	String, name of the publisher
genres	List, list of the genres
app_name	String, name of the game
title	String, title of the game
url	String, link to the game's page
release_date	String, date of the game released
tags	List, list of the user-defined tags
discount_price	Integer, price when discount
reviews_url	String, link to the game's reviews
specs	List, list of the specs of the game
price	Integer, normal price of the game
early_access	Boolean, the status of the game
id	String, the unique id of the game
developer	String, name of the develoiper

Figure 3: Contents in Items Metadata

In our analysis, we plan to use the "genres" feature from the metadata to construct the feature array, so we dropped 3285 games with no genre listed from the user-item dataset.

2.2.3 EDA on other datasets

Version 1: Review Data contains reviews made by users to different games. We thought this data can be useful at first but when we look at the data alongside the *Version 1: User and Item Data*, we fond that there's nearly 90000 users in User and Item Data while there's only 20000 of them have made reviews so they appears in *Version 1: Review Data*.
Version 2: Review Data contains nearly 8 million item reviews so it appears to be useful in the first look. But there's several problems exist within the data set. Firstly, the user identifier used in the data set is username, instead of user id. So we can't match the reviews to users in *Version 1: User and Item Data* because the username may changes and we don;t know if the detests are obtained around the same period. Secondly, the review data set does not include the opinion toward the game (i.e. recommended or not), which undermines the usability of the data set by a large margin. It is possible to process the text of the reviews and figure out if the user would recommend the game, but that is a huge topic and probably enough for another research report.
Bundle Data on the other hand, is not relevant to our predictive task.

3 Predictive Task

3.1 Task and baseline

We decided to predict the time a user will play a game based on the user and item information. In the dataset, we have access to the games each user has. Starting from there, we can access the game's genres, price, tags, etc. based on the common item id feature in both data sets. As the most straightforward feature, the average play time for each user and game are used as the baseline for this predictive task. In the baseline, we split the dataset into half and half to make a training and testing dataset and put the data into the linear regression model. The MSE we got was 11301 hours. One important thing we observed from the baseline model is that there exist negative prediction values, which is not possible for the actual game play time. Also, there exist some extreme game play time that over 100,000 minutes. These outliers will significantly boost the MSE and making the average gameplay hour goes to 7.3 hours, and that should be the main reason for the previous MSE.

3.2 Data cleaning

We did some data cleaning based on those outliers. We first decided to remove the training data that with more than 1,000 minutes as the play time, since we think 1,000 minutes is close to the up limit that a user will spend on one game, and most of the playtime in the dataset is within 1,000 minutes. After this step, the MSE goes below 50,000. Also, we decided to use gradient boost instead of linear regression to build the model.

3.3 Feature

The features we included are basically two parts: user's feature and item's feature.

3.3.1 User's features

- One hot encoding of the average playtime by user for each genre. There are 22 genres in total, so this feature will contain 22 entries. The game that with 0 playtime is not counted since each user will only play a small part of all the games in one genre. This feature can indicate the genres of game the user prefers.
- Average game play time of the user. This feature did not included genres, so this feature represents whether this user will spend large amount of time on Steam.

3.3.2 Item's features

- One hot encoding of game genres. The game that with 0 playtime is not counted. Only 1 and 0 will appear in the 22 entries, representing the genres of game.
- Average game play time of all users. The game that with 0 playtime is not counted. This feature basically can tell whether this game is durable to play or not.
- Price of the game.
- Number of reviews the item get. This feature can help to identify whether this game is popular or not.
- Average play time of top 3 Most similar items using Jaccard similarity. Tags are used here instead of genres since there are limited numbers of genres and will lower the precision of similarity. (This feature is not included in the final model since our hardware cannot support the amount of calculation over the whole train set, but it did help with MSE when we do that on a smaller train set)

4 Model

4.1 Choice of model

4.1.1 Baseline model

Linear regression model with user's average play time and game's average playtime as features.

4.1.2 Prediction models

1. Linear regression model.
2. Regularized regression model with alpha = 1.0.
3. Gradient boost regression model with random-state=0

4.2 optimization

The optimization of the model relies heavily on whether to include certain feature, as we designed many types of features to capture the properties of user, game, and their interactions. Due to the size of the data set and limitation in our machine's computing power, we decided to start from using average user play time and average game play time as baseline, and add features one by one and record the change in MSE. For model tuning, we tried different learning rate, n estimators, random state for our gradientboosting regressor. It turned out that n estimator makes the biggest difference, the higher the value, the smaller the MSE, while the run time of gets larger significantly. We finally decided to use:

1. learning rate = 0.2
2. n estimators = 150
3. random state = 2

4.3 model comparison

In all the cases, no matter how we split the data set or adjust the feature, the linear regression model and the regularized regression perform nearly the same on test MSE. Gradient boost regression model outperforms the other two. The best MSE we got is 30874.67 for the gradient boost model, 34365.54 for the linear regression model. So, we adapt gradient boost regression for this assignment.

4.4 unsuccessful attempts

Some features we proposed in the Feature section turned out to be inefficient, which is indicated by a small or even imperceptible change in MSE, including:

price of the game, with change in MSE of ≤ 0.1 number of reviews the item get, with change in MSE of 0.01

4.5 strengths and weaknesses

The only strength of linear model that we used as baseline is being fast, usually 10 times faster then gradientboosting, even when its n estimators is set to a very small value. It allows us to test out the relative effectiveness of different features and finally train the most effective ones using gradientboosting. Unlike linear model, gradientboosting regressor is very robust against over-fitting, thus always yields a smaller MSE. It is even stronger than other regularized models like random forest and decision tree regressor, as it uses a more optimized way of building the decision tree and calculating the results. The only weakness of gradientboosting is that it has less hyperparameters for tuning comparing to XGboost.

5 Literature

The datasets we used are from the collection of Professor Julian McAuley's research lab. In general, these datasets contain reviews from the Steam video game platform, and information about which games were bundled together. We extracted individual user's average play time spent on all owned games and games in each genre as well as the price and the number of reviews received on each game and other data, in order that we can predict a user's play time on an unseen game. We used about 75 percent of data to train the models and the rest of data to validate and test the models.

One other similar dataset contains columns: user id, game title, behavior name, behavior value, which give us information about if a game is purchased and how long a game is played. They created the scoring system for each game by defining the score as average hours played for the game multiplied by sum likes fraction add average hours across games multiplied by minimum number of likes fraction. Based on the scoring system, they used a restricted Boltzman machine to develop a stochastic ANN to generate construct recommendations. Also, with the same dataset, other researchers implemented recommending games to a user based on what games they have played and how many hours they have played it by using the K-Nearest Neighbor algorithm based on a rating system from the user's hours.

The results are similar but not quite the same. Our implementation uses features on the playtime on games of the same genre and how many hours the user has spent on this genre of games. At the same time, the KNN implementation only bases on the time played on the same game. This causes that their model tends to favor users who have one game in common with the given user when they have the exact same number of hours played.

6 Results and Conclusion

We have shown a method to estimate the personalized playing time regarding on a specific Steam game. We developed a time predict model which used the trained features of review numbers of the game, one-hot encoding of the game genres, one-hot encoding of average time spent on each genre by the user, and the price of the game in order to learn personalized playing time over s Steam game. As a result, we got 32047 minutes (8.9019 in hours) on out test MSE, which means that in general, our model will predict the personal playing time on a game off by less than 3 hours.

After doing an experiment on each feature and compare the resulting MSE on the test set, we found that the feature of review number of the game is the most important feature for our model as it made the biggest different on out test MSE, whereas the price of the game seems to have the lease impact on our model.

As future work, we intend to adapt these approaches on predicting the personalized playing time to generate the personal recommendations for the games. Base on the predicted playing time for the current user, we can pick the games that have the top playing time to make recommendations to the current user. In addition, if a user want to explore new games out of the genre which the user usually played in, we can use the model to calculate the expected playtime of popular games in the specific genre to make a more personalized recommendation.

7 Works Cited

Dataset:

[1] Wang-Cheng Kang, Julian McAuley, *Self-attentive sequential recommendation*, ICDM, 2018

[2] Mengting Wan, Julian McAuley, *Item recommendation on monotonic behavior chains*, RecSys, 2018

[3] Apurva Pathak, Kshitiz Gupta, Julian McAuley, *Generating and personalizing bundle recommendations on Steam*, SIGIR, 2017

Literature Review:

[4] Tamber, *"Steam Video Games."*, Kaggle, 9 Mar. 2017, https://www.kaggle.com/tamber/steam-video-games.