Sequential Pattern Mining: Big Data Analysis in Mobile Gaming

A Technical Report Submitted to the Faculty of the School of Engineering and Applied Science
University of Virginia • Charlottesville, Virginia

In Partial Fulfillment of the Requirements of the Degree Bachelor of Science, School of Engineering

Selwyn Hector

Spring 2020

Independent Research Collaborators: Prof Natasha Foutz, Prof Yangfeng Ji, Prof Jingjing Li

On my honor as a University Student, I have neither given nor received unauthorized aid on this assignment as defined by the Honor Guidelines for Thesis-Related Assignments

Signature _____ Date _____

Selwyn Hector

Approved _____ Date _____

Yangfeng Ji, Department of Computer Science

Sequential Pattern Mining: Big Data Analysis in Mobile Gaming

Introduction

Data Science is one of the fastest-growing job areas in the world. According to Forbes, the field is projected to have over 700,000 openings in 2020.[1] The University of Virginia feels the field is promising enough to open an entirely new school to allow students to study it. Hardware has rapidly progressed in recent years and companies are capturing more data than ever. Data science allows companies to analyze these influxes and discover new information they might not have ever been able to discover. It is an interdisciplinary field that unites computer science and statistics.

This technical report will focus on a data science project done in collaboration with the McIntire School of Commerce. Data was collected from a popular free-to-play MMORPG game. The game features offline and online play where users control characters, level them up, and then take on quests either individually or in groups. Players also have a stamina system that limits how many activities they can complete each day. While the game is free, players can pay money to reset their stamina and obtain items more quickly. The company that developed the game reached out to the university because they needed help analyzing their data. They had many files of data on user activity and wanted to discover which activities they should promote and develop to make the most money. They also wanted to discover what patterns lead to users participating in group quests and joining guilds. This was under the assumption that users would play longer if playing online with friends.

Datasets

The initial dataset provided was a single 20-gigabyte log file. This log file contained 323,319,283 lines and each line detailed a single action of a single character. This is clearly too much data for any human to make sense of so it was necessary to utilize data science for proper analysis. Each line contained a character id which acted as the primary key to identify the player's character. It also contained an actiontype that represented what the character was doing along with a timestamp. The action types were AchievementMake, FatigueChange, FriendMake, GuildJoin, Levelup, Login, Logout, PVE, PVP, QuestComplete, SkillChange, and Trade. The social actions that the company was particularly interested in were GuildJoin, FriendMake, and Trade. The dataset represented a month of data from a random sample of players.

Along with the log file, there was also a table that connected characters to users. A user can have multiple characters linked to a single account so the data must have a unique identifier for each. While characters are identified by a character id, users are identified by a pop id and the table simply mapped character ids to pop ids. Finally, there was a table that represented user purchases. Each line would give the pop id for the user making the purchase along with the price advertised, the price they paid, and the date of the transaction. From these three sets of data, there was a lot of information to be discovered.

Analysis Process

From the beginning of the project, the goal was always to use sequential pattern mining to break down and understand the data. According to Fournier-Viger, sequential pattern "consists of discovering interesting subsequences in a set of sequences, where the interestingness of a

subsequence can be measured in terms of various criteria such as its occurrence frequency, length, and profit." [3] In simple terms, sequential pattern mining algorithmically finds patterns in large sets of data. It requires the data to be time ordered and organized into independent sequences with some meaning. This set of meaningful sequences is known as a sequence database. From a sequence database, sequential pattern mining algorithms can discover the most frequent patterns and more importantly which patterns lead to certain actions. For example, a sequential pattern mining algorithm can take 100 sequences as input and predict that a Trade will likely lead to a LevelUp and SkillChange.

The project began with a lot of trial and error to determine which sequential pattern mining algorithms were most appropriate for the dataset. The SPMF library, the most popular open-source library for performing pattern rule mining, has over 200 different algorithms available. Initially, the team was provided a sample sequence database that had converted user data into sequences based on the day they occurred. The three the team agreed upon to use algorithms from a subset of sequential pattern mining known as sequential rule mining. Sequential rule mining takes probability into account along with frequency. According to Kour, sequential rules are in form X -> Y such that X and Y are sets of items, and if the actions in the set of items from X occur then the actions of set Y items will occur with some probability[4]. Sequential Rule Mining not only finds which patterns are most common but also makes strong predictions on how actions influence and predict following actions. Each sequential rule has a confidence value and support value that accompany the rule. The support is the number of sequences that exhibit the pattern and the confidence is a statistical measure of the likelihood given as a percentage.

The sequential rule mining algorithms ultimately selected were ERMiner, TNS, and TRuleGrowth. They all take sequence databases as input and output a set of sequence rules. They also take minimum support and minimum confidence as input. These are the minimum values for support and confidence needed for a rule to be output. ERMiner is the most efficient generic rule mining algorithm and takes a text file representation of the sequence database along with minimum support and minimum confidence as an input. TRuleGrowth requires those parameters but also requires a window size argument that restricts which rules can be generated. The larger the window size the more rules can be considered. Finally, the TNS algorithm is unique because it not only finds sequential rules but finds the k most likely rules. Therefore, given a k of 100, TNS will extract the top 100 rules with the highest confidence.

In order to transform the raw log file of timestamped character actions into a sequence database, the team had to determine how to create sequences and which sequences would be included. It was decided the definition of a valid sequence would be a set of consecutive actions in between a character's login and logout. So the raw log file of just actions had to be scanned line by line and character ids were mapped to sequences in a new file. Interestingly, there were a number of actions that did not occur between a login and a logout so those were dropped. After many discussions and looking at the size of the population, it was decided the dataset would be further restricted by only accepting characters that had 10 or more of these valid sequences. This would represent the most active users who had enough samples to draw conclusions from. These restrictions shrunk the data set from about 4.5 million characters to just over 100,000, creating a data set that could be loaded and analyzed much more easily.

After deciding on the dataset and sequence definitions, basic statistics were taken for the entire dataset like the average sequence length and the average number of sequences per character. These were also taken for samples of sequences that contained certain actions like Trade or PVE. By sampling certain sequences and running statistics, the team found that some actions like PVE were much more frequent than average while actions like GuildJoin rarely occurred. The dataset then had to be prepared to be used for the sequential rule mining algorithms. The algorithm libraries required sequences to be coded as numbers instead of strings as text so the sequences were coded so that each action is represented by a unique number. Figure 1 shows the output of a TNS algorithm being run on the data. The arrow indicates the behavior coded on the left-hand side leads to the behaviors on the right-hand side, unordered. So for a sequence, if the activity represented by 9 occurs then the activities represented by 4 and 8 will likely occur later on as supported by 100,000 sequences.

```
9 ==> 4,8 #SUP: 100000 #CONF: 1.0
9 ==> 2,4,8 #SUP: 89152 #CONF: 0.89152
9 ==> 1,4,8 #SUP: 83982 #CONF: 0.83982
```

Figure 1: Sample TNS algorithm output

The team decided that TNS was the best algorithm overall to draw predictions from since it outputs high confidence rules. The TNS algorithm was run on every sample and results were stored.

The final goal was to take these outputs and prepare them for clustering analysis. According to Jain, clustering is the process of organizing objects into groups whose members are

similar in some way[4]. When clustering is complete users with similar spending habits will be grouped together and their commonalities can be analyzed to find the strongest indicators. Clustering required first linking the TNS rules back to users. This was done by scanning each user's sequences and matching the rules from the TNS outputs back to user sequences. For example, If a TNS rule is 1,2 ⇒ 3 and a user pattern is 5, 1,2 then that user pattern is counted as matching the TNS rule because the left side of the TNS rule is a user subsequence. These rules were generated in a number of combinations including counting the number of matches versus averaging out all the matches TNS rule confidence values. Another set of options was the size of the item insert. Item insert represents the space allowed between consecutive items in the user pattern. For example, 1,4,3 as a user pattern would not match a TNS rule of 1,3 ⇒ 6 with an item insert of 0 but would match for an item insert of 1. Larger item inserts expand the number of user patterns that can be matched to TNS rules. Using different combinations gave the team many options for looking at how well an individual user is described by the set of TNS rules. Each user's set of actions on each day were analyzed with this process and numerical values were stored in a table with a user id and date.

Lastly purchasing data was incorporated into this table. This was done by taking the table that represented users' TNS values and adding data on whether the user had made any purchases on that day. In the final analysis, users will be clustered by their spending activity and how well their own sequences correlated to the TNS rules. This will allow the team to determine how much the TNS rules indicated and what rules are most successful at predicting spending.

Tools Used

A majority of programs written to analyze the data were written in Python. Python, along with R, is one of the premier programming languages for data science with very well-maintained libraries. The main libraries used were Numpy, Pandas, and Matplotlib. Pandas and Numpy are libraries that efficiently allow the program to upload, view, and export. They can also easily perform common operations like joins between datasets, data segmentation, and statistical calculations. Matplotlib is a charting library that allows for these data sets to be visualized with a wide range of customization options.

Along with Python C++ was used in certain smaller cases such as altering the input sequence database data. Also, the SPMF library written in Java was used to perform all sequential pattern mining algorithms described. In terms of storage, Amazon Web Services (AWS) was used to store long term data and share it with the gaming company. Google Drive was also by the UVA team to coordinate and share documents. The UVA Research Computing cluster, Rivanna, was essential for running the programs which required large amounts of memory and could sometimes take hours to process.

Conclusion

Due to NDA constraints, the official results of the analysis cannot be reported. However, the clustering algorithms were successful in categorizing the data. Also, a number of other suitable machine learning processes can be applied to the generated data such as linear

regressions and neural network deep learning. Ultimately sequential pattern mining proved to be very successful at analyzing discrete, time-ordered behaviors and helping to predict future behavior. The project was also a very good full-scope example of taking a massive and unclear data set and transforming it into smaller, well-understood parts. Data Science allows for massive data collection to be profitable and for businesses to discover hidden trends.

References

Columbus, L. (2017, May 14). IBM Predicts Demand For Data Scientists Will Soar 28% By 2020. Retrieved May 1, 2020, from https://www.forbes.com/sites/louiscolumbus/2017/05/13/ibm-predicts-demand-for-data-scientists-will-soar-28-by-2020/#5de07ecb7e3b

Fournier-Viger, P., & Lee, C.-H. (2017, March 8). An Introduction to Sequential Pattern Mining. Retrieved April 28, 2020, from http://data-mining.philippe-fournier-viger.com/introduction-sequential-pattern-mining/

Fournier-Viger, P. (2020). SPMF: A Java Open-Source Data Mining Library. Retrieved May 3, 2020, from https://www.philippe-fournier-viger.com/spmf/

Jain, P. (2019, December 11). Clustering Clearly Explained. Retrieved from https://towardsdatascience.com/clustering-clearly-explained-5561642ec20c

Kour, A. "Sequential Rule Mining, Methods and Techniques: A Review" *International Journal of Computational Intelligence Research,* Vol. 13, NO. 7, pp. 1709-1715, 2017.