CHURN DETERMINATION AND PREDICTION IN DIGITAL GAME-BASED LEARNING

DR MUDIARASAN KUPPUSAMY AND MAI KIGUCHI

Submitted to:

LIFE IS TECH INC. JAPAN

ASIA PACIFIC UNIVERSITY OF TECHNOLOGY & INNOVATION (APU)

APRIL 2020

ABSTRACT

Digital game-based learning is a part of Educational Technology (EdTech). It is a digital versionof game-based learning, an approach to facilitating learning with gameplay. The digital game-based learning market has been expanding, and the market revenue is expected to grow by over $24 billion by 2024. One of the common issues in the EdTech market is the higher churnrate. However, because the digital game-based learning market is still in the early stage, few studies related to marketing perspectives. Besides, the education or online gaming industries approach can be only partially applicable to digital game-based learning. A popular approach for addressing a higher churn rate is churn prediction. By using a dataset froma Japanese company providing a digital game-based learning service as a case study, an approachfor the combination of defining churn and churn prediction for digital game-based learning is proposed. There are four objectives for this project. The first one is the determination of churn,which is defined by applying an arbitrary time frame for each user. The determination is done by comparing the recency and average and two standard deviations of user idletime. The second objective is to clarify the churn rate of the Japanese service.

Using the newly created churn definition, the churn rate became evident at 56.77%. The third objective is to develop the best churn prediction model by comparing LR, DT, and RF models. Feature selection, dataset split ratio comparison, and hyperparameter tuning are conducted to build thebest performance models. The LR model has a distinctively highest accuracy of 0.9185, AUC of 0.9225, and an F1-score of 0.9194. The higher F1 score means the recall and precision are well balanced. The higher F1 score means the recall and precision are well balanced. Thus, theresults demonstrate the effectiveness of the proposed approach of churn determination and prediction demonstrated in digital game-based learning.

SECTION 1: INTRODUCTION

Digital game-based learning is one of the narrowed educational technology categories (EdTech). This chapter starts with some important marketing approachesand the digital game-based learning market situation. In the next subsection of the problem statement, EdTech and digital game-based learning are discussed. The subsection of the problem statement follows the research aims, objectives, scope, and significance.

1.1 Background

The importance of the marketing approach in digital game-based learning has been increasing dueto market expansion. Some of the important key indicators of marketing approach, retentionand churn, and churn prediction will be introduced. After that, the market situation of EdTech and digital game-based learning in the world and Japan is explained.

1.1.1 Retention and Churn, and Churn Prediction

Customer churn is a percentage of customers who stopped using a service or products in a certain period (Yang et al., 2018). This is one of the critical metrics for a business to evaluate. It is important to understand the rates and factors of customer churn to focuson the problems. Customer retention is another common metric to be used. It is the percentageof customer relationships, especially maintaining customers using the service or products over time. Addressing the issues by analyzing customer churn leads to higher customerretention. For instance, American Express increased 400% of customer retention by changing customer service to an opportunity to build customer relationships and putting enormous efforts to improve their issues (Clarke & Kinghorn, 2018). The reason for churn is often because of consumer switching. According to CallMiner (2019),85% of adults changed suppliers about 1.81 times on average in a year. These frequent switchescause a tremendous loss to the companies. An estimated loss of companies in the USA is about $136.8 billion per year because of avoidable consumer switching. The highest churning sectoris communications companies (including mobile phones, internet, and landline telephone), followed by banks and property insurance companies. In general, when the business started, the first focus was to acquire new customers because customer acquisition is an apparent and major part of revenue growth (Miller, Vonwiller & Weed, 2016). Another less noticeable but significant factor is customer success, measured by high retention rates. After investing the time and money to acquire a customer, the business loses the acquisition investment and potential revenue from the customer if the customer leaves immediately. Besides, the retention cost for existing users is much lowerthan the cost of acquiring new users (Suh & Alhaery, 2016; Fu et al., 2017). By reducing customer churn, in other words, the increased retention rate is a key to the success of having a robustand faster-growing business. For example, Netflix saved an estimated $1 billion per year by reducing customer churn with its recommendation system (Gomez-Uribe & Hunt, 2015).

Churn prediction has been one of the important topics for customer retention and management because it allows companies to create better marketing strategies to improve user retention (Fu et al., 2017). For example, sending push notifications or e-mails is one of the common ideas to prevent "will be churners" users based on the churn prediction. Therefore, successful churn prediction provides a better retention rate and reduces the cost, and it will benefit stakeholders such as game developers, advertisers, and platform operators.

1.1.2 Education Technology (EdTech) and Digital Game-Based Learning (DGBL) Market in the world and Japan

The education market is huge, with expenditure from governments,parents, corporates, and individuals (HolonIQ, 2019). The education sector is extremely underdigitized, with less than 3% of expenditure is allocated to technology. Nevertheless, it is estimated that the amount spent on digital will grow to $342 billion by 2025 because of the growth of EdTech. EdTech is an industry that integrates education and technology advances. The industry contains wide-ranging services such as online courses, learning (teaching) management, language, coding, Science, Technology, Engineering, and Mathematics (STEM),and Virtual Reality (VR).

One of the narrowed down categories of EdTech is called digital game-based learning (DGBL).The concept of the original non-tech game-based learning (GBL) has gameplay in a learning context, and digital game-based learning (DGBL) is another specific terminology for game-based learning with the use of technology (Perini et al., 2018).
The digital game-based learning market is exceedingly fast-growing. According to a report from Metaari (S. Adkins, 2019), the global five-year compound annual growth rate (CAGR) for the global digital game-based learning market is 33.2%, and the market is maturing.

Additionally, the market revenue is expected to reach over $24 billion by 2024. Technology and science advances have a powerful influence on the digital game-based learning industry. The market growth is highly correlated to the ongoing innovations, which are advances in psychometrics, neuroscience, augmented reality (AR), virtual reality (VR), and artificial intelligence (AI).

Another reason for market expansion on top of the technology innovation is the wide range of customers. The report from Adkins (2019) is split into eight segments: consumers, preschools, primary education, secondary education, tertiary and higher education, federal government agencies, other local government, and corporations and businesses. Revenues willincrease more than double among all eight buying segments over the forecast period, and four of the segments will have four times more revenues. Because of the estimation of market expansion with having many segments and technology innovation, the digital game-based learning market gains attention from investors (Adkins, 2019). Private investments in digitalgame-based learning have been historical high since 2016. $1.7 billion was funded into digital game-based learning companies in 2016 and 2017 combined, and $2.25 billion was invested in 2018.

The market situation in Japan is similar. According to a report from the Nomura research institute (Kumabe, 2018), the EdTech market in Japan has been expanding, the same as the global trends. The market size in 2016 was $1.54 billion (¥169.1 billion), and it is expected to reach about $2.8 billion (¥310.3 billion) by 2023. The market is still at an early stage, so many companies provide EdTech products or services. Based on the market map from Studyplus, Inc. (2018) as shown in Figure 1, there are about 115 services in the EdTech marketin 2018. From the learning contents perspective, there are four big categories in Japan. Compulsory school subjects (Math, Japanese, science, social studies and English), English conversation, programming, and AI and Robotics. Although there are no statistics or market maps specifically for digital-game based learning, some of these services use a digital game-based learning approach. For example, English apps such as Mikan, Manamee, Eipontan,and others are digital game-based learning apps. Some programming services such as Life is Tech, Progate, Codeprep use game-based learning or gamification approaches. On top of the above Japanese services, major game-based learning players such as Code.org, Scratch, Tynker,Code Combat, and Code monkey from the USA had started providing Japanese language supports. Thus, the EdTech and digital game-based learning market in

Japan is expanding and highly competitive.

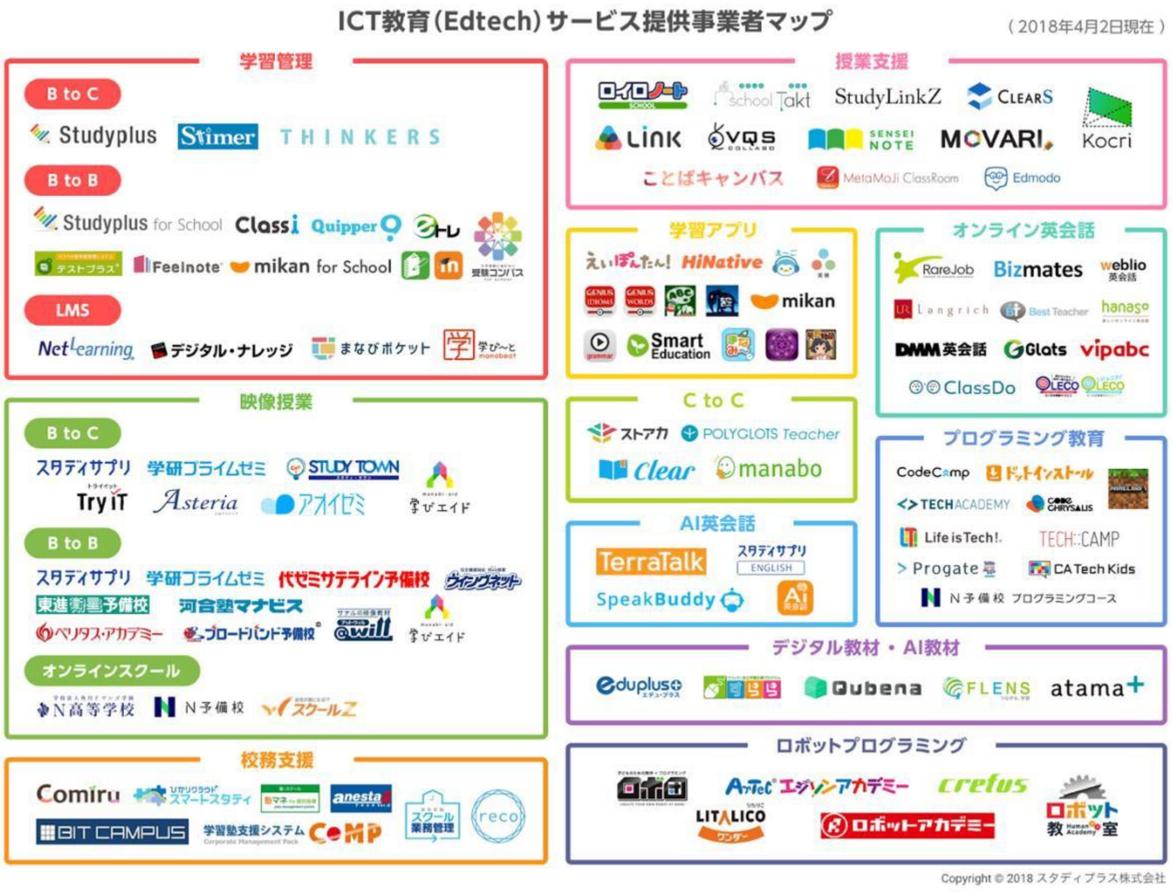


Figure 1. EdTech Market Map in Japan

1.2 Problem Statement

The churn rates in EdTech have been higher in common. The Washington Post reported virtual education in 2019 (Strauss, 2019), noting the difference in graduation rate between normal and virtual schools. The graduation rate of virtual schools is only 50.1%, whereas the overall graduation rate in the United States is 84%. Massive open online courses (MOOCs) have the worst data: only 5% of students finish their courses (Liubov, 2019). Another statistic of subscription business revealed the average churn rate in nine categories. Figure 2 illustrates the churn rates by industry, and education is the third-highest among the ninecategories, which is 10.29% (Recurly Inc., 2019). The churn rate is higher in EdTech in general,and the rate is wide-ranging depending on the report.

The marketing approach is becoming more important to address the issue, especially for the growing market like digital game-based learning. Churn prediction can help address the issues and create strategies for churners (Fu et al., 2017). Nevertheless, there is not yet an established model to address the higher churn by churn prediction in the digital game-based learning industry due to a lack of research in Japan and the world. Hence research is required. Research about online gaming and the education industry can be partly applicable to digital game-based learning because digital game-based learning is a combination of online gaming and education. The education industry has a lot of research on prediction based on student demographics and academic information, but few withuser behaviour data. On the other hand, the online gaming industry has much research about churn prediction with user behaviour, but there are big differences with digital game-based learning.

First, the evaluation time frame is different. The time frame in the online gaming industry is often 7, 14, or 30 days (Fu et al., 2017). At the same time, education uses a longer time frame like the entire course period, from weeks to one year (Bote-Lorenzo & Gómez-Sánchez, 2017; Marquez-Vera, Morales & Soto, 2013). Social interaction is another differencein digital game-based learning. Online gaming or application tends to have more social interaction between users, affecting users' churn (Fu et al., 2017). Digital game-based learning products or services are designed for gameplay, and there is less social interaction during the play than in the online gaming industry. To summarize the points, EdTech, in general, has a higher churn rate but has no established model and no studies on churn prediction in the digital game-based learning industry.

Current models from different industries (gaming and education) cannot be perfectly applicableto digital game-based learning. A Japanese company that provides a digital game-based learning service will be used as a case study in this project. This company has issues with retention analysis because of the above reasons. For that, an approach for churn prediction for digital game-based learning is proposedthen utilized to address the business issues for this Japanese company.

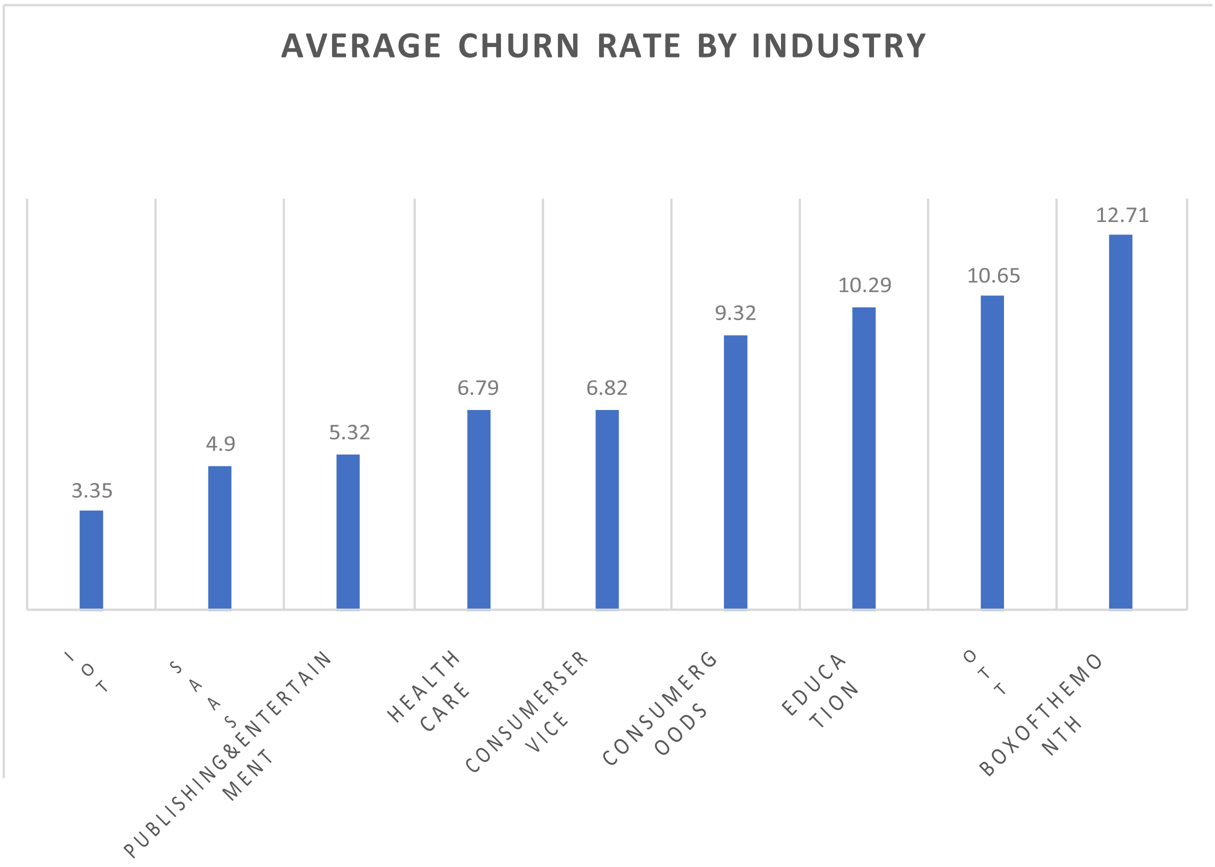


Figure 2. Average Churn Rate by Industry - Adopted from (Recurly Inc., 2019)

1.3 Aim and Objectives

This project aims to design and validate a churn determination and a prediction model for the Japanese digital game-based learning environment. Four objectives are applicable:

1. To identify the determinants of churn in digital game-based learning
2. To clarify the churn rate for the Japanese digital game-based learning product by analyzing datasets provided by a Japanese digital game-based learning company
3. To develop and find the best churn prediction model based on the defined churn using Logistic Regression, Decision Tree, and Random Forest.
4. To validate the model in the context of a Japanese digital game-based learning

1.4 Scope

There are three scopes which are service, period, and method. Service is focused on digital game-based learning service in Japan because a Japanese company provided its digital game-based learning service user data. The company provides one of the biggest digital game-based learning services in Japan. The period is set to about a year and a half. In general, the learning period is about a couple of weeks to one year (Xing et al., 2016; Bote-Lorenzo & Gómez-Sánchez, 2017), and more than one year of data is appropriate for the prediction. The digital game-based learning service had launched in April 2018, and the longest possible period is a year and a half. Besides, the method relies on descriptive and predictive analysis.

1.5 Significance of Research

There are no other studies on churn determination and prediction in digital game-based learning,which will contribute to churn determination and prediction fields and digital game-based learning sectors. In addition, this will help digital game-based learning or gamification companies to conduct better marketing analyses on their services or products.

SECTION 2: LITERATURE REVIEW

The literature review of three types of categories is examined in this chapter. The first category isdigital game-based learning. Its concepts, past studies, research in Japan and current trends areinspected. Next is the definition of churn or retention and the method and evaluation period of retention analysis. These are examined by industries which are education and online gaming. The final category is churn prediction. The digital game-based learning, online gaming, and education researches related to churn prediction are reviewed.

2.1 Digital Game-Based Learning

2.1.1 Concept of Digital Game-Based Learning and Past Studies

The use of gameplay in education is not a new idea, but the use has been limited in nursery school for a long time (Sanchez, 2019). With technological advancement, digital game-based learning was already employed in the 1970s (Gee, 2003). However, the term "digital game-based learning (DGBL)" was made popular in the early 2000s by Gee (Gee, 2003) and Prensky (Prensky, 2001). Digital game-based learning is an approach to facilitating learning with digital games. Although digital game-based learning is quite different from gamification, these two are sometimes confused. The difference is that gamification uses game-like elements such as level, points, and rewards, whereas game-based learning is learning through gameplay (Khan, Ahmad, & Malik, 2017). Digital game-based learning is gaining more attention from educators and researchers (Sanchez, 2019). The reason for widespread attention to digital game-based learning is a combination of three aspects (Information Resources Management Association, 2015). The three aspects are ongoing research by digital game-based learning proponents, today's digital natives who are disengaged from traditional instruction, and increased popularity in the gamingindustry.

There are two main approaches to the digital game-based learning category (Sanchez, 2019). The first approach is based on the concept of Prensky, "Edutainment". Edutainment is entertainment designed to be educational(Prensky, 2001). With the idea of edutainment, digital game-based learning can improve students' motivation or confidence (Sørensen, Meyer & Egenfeldt-Nielsen, 2011). Another approach is based on constructivism and cognitivism (Sanchez, 2019). With this, the educational value of digital game-based learning is the acquired knowledge from decisions and behaviours made by the player to adjust to challenging situations on top of motivating. In addition to the above two approaches, many researchers have evaluated the relationships between learning effectiveness, motivation, different courses, and different gaming design. These researches disclose that digital game-based learning can enlarge learning interests, enhance motivation, and increase performance (Information Resources Management Association, 2015).

2.1.2 Digital Game-Based Learning Research in Japan

Digital game-based learning studies in Japan are very limited, and the area of research is sparsely distributed. Two studies focus on the perspectives on the use of digital games for learning. One study focused on student perspectives on digital game use for English learning in higher education (Bolliger et al., 2015). It surveyed college students, and results are reported that most students have a positive view of the use. Another study analyzed the educator's perspective by questionnaire and e-mail interviews. The result indicates that most educators consider digital game-based learning has a strong impact on students' motivation, but not learning outcomes (Franciosi, 2014).

Some studies focus on recent research findings in the world. For example, one paper reported the current and future development of the educational use of games in informal learning (Fujimoto & Yamada, 2013). The other study introduced an effective approach to digital game-based learning for educators (Fujimoto, 2011). Nonetheless, these works are written by Toru Fujimoto from the University of Tokyo. He also mentioned that research interests in game-based learning in Japan are not as diverse as overseas (Fujimoto, Shigeta & Fukuyama, 2016). Hence, the study of digital game-based learning in Japan is narrow and limited.

2.1.3 Current Trend in Digital Game-Based Learning Research

There are a variety of combinations of subjects, ages, and regions for the research focus; however, the current trend in digital game-based learning can be provided by the three latest meta-analyses. The first meta-analysis focuses on heavy game use in education (Zhonggen, 2019). The secondmeta-analysis scopes for kindergarten to 12th grade (K-12) mathematics education (Byun & Joung, 2018). The third focuses on digital game-based learning in elementary science (Hussein et al., 2019).

Based on the above meta-analyses, the trend in game-based learning is revealed. The publishing trend in digital game-based learning has increased for more than a decade. Hussein et al. (2019) reported that all researches fall into two major categories: motivational and skills acquisitions and knowledge construction and content understanding outcomes, which are the same approaches mentioned by (Sanchez 2019). Byun and Joung (2018) disclosed that the main goal of all research for K-12 mathematics education is to understand how digital game-based learning affects mathematics students' achievement. Zhonggen (2019) examined the meta-analysis in a wide range of research in digital game-basedlearning. It reported three influencing factors, positive and negative effects. Influences include gaming easiness and surprises, relationships between learning attributes and gaming mechanics, types of games, and learners' age. The effective positive findings include improving the learning outcomes and teaching, obtaining cognitive abilities, and facilitating a holistic understanding of concepts.

On the other hand, negative effects are also reported from a couple of research as negative influences on the learning effectiveness from the aggravated mental workload. As clearly seen from the above results, both the past study and the current trend of research mostly focus on the influence of digital game-based learning. Thus, there is no research on marketing-related topics in digital game-based learning yet. The need for the marketing approach in digital game-based learning will likely increase soon, considering the current and expected future market expansion.

2.2 Definition of Churn or Retention, Method and Period of Retention Analysis

As explained earlier in the problem statement section, the churn rate is higher in the EdTech industry in general. However, the definition of churn or retention can be varied and vague from two points of view. The first point is the period used for churn or retention calculation. According to Jill Avery, a seniorlecturer at Harvard Business School, churn rate is measured by month, quarter, or year depending on the industry and products (Gallo, 2014). The introductory period is annual for most companies. Some services that are charged monthly use the churn rate by month orlook at the churn rate monthly due to having a faster churn rate. There is no doubt for the definition of churn rate, which is a percentage of customers who stopped using a service (Yang et al., 2018). Nonetheless, what "churn" or "retention" means differs depending on the services and product.

Therefore, past studies should be reviewed for these perspectives. Due to the lack of research in digital game-based learning, gaming and education research are chosen to be examined to see the definition of churn and how the churn period is chosen because digital game-based learning is a combination of gaming and education.

2.2.1 Definition of Churn and Period of Retention Analysis in the Online Gaming Industry

There is no universally accepted definition of retention or churn because retention analysis is often used for internal use only (Fu et al., 2017; Suh & Alhaery, 2016). Of course, game publishers use metrics to calculate but do not share sensitive, detailed information with others.Though, there is a rough definition in the gaming industry. A study claimed that online games should use a different definition of churn, unlike other industries like telecommunication or financial services (Lee et al., 2018). Instead of using the withdrawal of membership, the inactivity period should be used. More than 13 weeks without any access to the research is defined as churned. Using inactivity is one of the gaming industry's common approaches seen in other literature (Hadiji et al., 2014; Xie et al., 2015, Tamassia et al., 2016).

A work proposed a definition of churn in a different perspective for online casino game services (Suh & Alhaery, 2016). Instead of using a certain time frame that is easy to understand and execute, using an arbitrary time frame for each user is suggested. Hence, it uses three metrics to define churner: each player’s last play, average days betweenplay, and two standard deviations.

However, the common approach of retention analysis is a combination of the login frequency and evaluation period in the gaming industry (Fu et al., 2017). The evaluation period often starts from the first day of the release date or the day a new user joined, and it is often7, 14, or 30 days. There are four retention methods: full retention,classic retention, accumulative retention, and return retention. The method of retention rate calculation is summarized in Table 1. In the research from Fu et al. (2017), full retention in 10days is used to see the overall retention rate. Classifying the method based on Table 1, some other research fits the methods. For example, a study of churn prediction of freemium mobile games uses a return retention method and a one-month evaluation period (Banerjee et al., 2019). Another work also usedreturn retention within 14 days after the last activity (Milošević, Živić & Andjelković, 2017). The four retention methods with user login frequency or the inactive period are often used to define churn in the online gaming industry. Many of the approaches fall into one of the four methods, and the common churn evaluation period is 7, 14, or 30days due to the higher churn in the first month. These methods are applied to both PC andmobile gaming.

Table 1. Retention Rate Calculation Methods - Adopted from Fu et al. (2017)

Method	Definition	Feature
Full retention	Count in if players return every single day during the evaluation period.	Extremely restrictive and not so widespread.
Classic retention	Count in if players return on the evaluation day.	The easiest way to calculate, and it is the most widely used.
Accumulative retention	Count in if players return days during the evaluation period, and higher than the predefined threshold.	Flexible but with a high computational cost.
Return retention	Count in if players return at least once during the evaluation period.	Least restrictive and often produces relatively promising results.

2.2.2 Definition of Churn, and Period of Retention Analysis in the Education Industry

A more simple retention analysis method is used in the education field than in the gaming industry. For offline courses, it is clear that students stopped coming to the class means dropped out (churned). Nevertheless, there was no clear definition of dropout for online courses, same as the gaming industry. At an earlier stage of the dropout research, a study claimed that there is no clear definition of dropout, and it proposed the dropout as students that withdraw from the e-learning courses with financial penalties. Students can drop a course at the add/drop period with fully refunded (Levy, 2007). Recent research on retention management for academic online courses seems to use the same definition of students who stopped coming to the course, which is highly likely not to include the add/drop period dropouts. However, the evaluation period is different depending on the research because the period is the same as the course duration. For example, one academic year was used for dropout analysis (Sorensen & Donovan, 2017). Another one used a fall term of three years (2009, 2010, 2011) at a university for dropout prediction (Bingham & Solverson, 2016).

This is the same with Massive open online courses (MOOCs). MOOCs are a subset of educational technology, and it is a popular field of research these days. Retention analysis on MOOCs uses the same period, course duration, and the same definition of dropout, which means learners who are uncompleted the course. An analysis of dropout causes used one academic year of three semesters (Spring, Fall, Summer) of MIT and Harvard courses and the same dropout definition (Gupta & Sabitha, 2018). Another research that analyzed factors affecting retention also used course duration, six weeks as a period and the same dropout (Hone & El Said, 2016). Some courses with multiple weeks use multiple periods for evaluating dropouts. For instance, a study used eight weeks and checked every week's drop out rate (Xing et al., 2016). Few studies defined and used a different definition ofdropout. For example, a study of analysis on dropout reasons for MOOCs uses the entire course period, and dropout is determined as "not logged in the course for more than 14 days" without finishing the course (Niu et al., 2018). Another work has tried a different approach to the retention analysis of MOOCs (Bote-Lorenzo & Gómez-Sánchez, 2017). It used one semester period data and compared the number of samples based on their defined engagement indicators between chapters. This is a similar concept as dropout or churn. Overall, the popular period is course duration. Still, there are variations from weeks to years, which makes it difficult to determine a certain cutoff like the gaming industry. Onthe other hand, the dropout definition is simpler because if the student did not complete the course, the student is defined as a churner in the educational industry.

2.3 Churn Prediction

Churn prediction has been studied because it is an important topic in various domains (Fu etal., 2017). With the definition of churn, it becomes possible to predict customer churn. For churn prediction in general, common techniques used are machine learning with algorithms such as logistic regression (LR), decision tree (DT), and random forest (RF) (Yang et al., 2018). The details of the studies are summarized in Table 2 and Table 3.

Table 2 contains digital game-based learning and gaming literature. The table has citations, the purpose of the study,variables used for modelling, models (algorithms), and results. Table 3 is literature in education and has one extra column, a data source, because literature in education oftenuses different data sources.

2.3.1 Retention Prediction in Digital Game-Based Learning

Although there is no research on churn or retention prediction in digital game-based learning, one research predicts abandonment in online coding tutorials. The work iscategorized in the gaming sector, but the chosen product for the research teaches programmingconcepts. This is considered a digital game-based learning category. The research focuses onpredicting learners who are likely to complete the next lesson instead of usingthe existing churn prediction approach (Yan, Lee & Ko, 2019). This is because leaving open does not fit the definition of the end of the course, and the tutorial is before the membership registration. It does not fit in the definition of withdrawal of membership. Cumulative features such as idle time, number of execution button clicks, time to spend on reading, andlearner features such as age, gender, registration status, and programming experience are usedfor the prediction. The used machine learning classifiers are Logistic regression (LR), RandomForest (RF), and Gradient Boosting Decision Tree (GBDT), and predicted 61% to 76% of learners who did not complete the next level with an average AUC of 0.68. However, this approachis too specialized for a tutorial before registration and is not applicable to game based-learning products in general.

2.3.2 Churn and Retention Prediction in Online and Mobile Gaming

The churn rate for online gaming industries is relatively high because players do not play the game until the content is exhausted (Fu et al., 2017). There are many approaches or combinations of methods, selected data features, and time duration. However, the common approach is using machine learning with user behaviour data. Some literature mentioned well- established prediction models are Logistic Regression (LR), Decision Tree (DT), Random Forest (RF), and Support vector machines (SVM) (Milošević, Živić & Andjelković, 2017; Liuet al., 2018). Earlier works often used the above common machine learning models. For example, LR, DT, Naive Bayes (NB), and Neural networks predict churn for freemium games (Hadijiet al., 2014). SVM, DT, and LR are used to predict with general event frequency data for the versatility of any game (Xie et al., 2015). The Hidden Markov model was used to predict a major game: a shooter and massively multiplayer online game (Tamassia et al., 2016). For a shorter period of prediction, a one-day churn prediction algorithm is introduced because more than 70% of new users only play a game for one day and stop using it the next day (Milošević,Živić & Andjelković, 2017). The study used user activity features, monetization features, and gameplay style features (e.g. auction usage, spend on training), and five algorithms are used to predict and compare. The algorithms are LR, DT, RF, Gaussian Naive Bayes, Gradient Boosting. For a different approach, a work proposed using an arbitrary time frame to define churn used E-CHAID decision tree algorithm on SPSS, and 60 features of seven categories are fed the algorithm (Suh & Alhaery, 2016). The data categories are recency, frequency, monetary value, length of relationship (e.g. Number of days between the first and last playdates), inter-play (e.g. average number of days between plays), bonuses/reward,demographic. Another approach with a deep neural network (DNN) is proposed with technology advancement. A study proposed a DNN architecture of the inductive semi-supervised embedding model (Liu et al., 2018). The semi-supervised model predicts and compares with state-of-the-art models such as LR, DT, RF, and SVM. Three categories of datafeatures are used for this study; play history (e.g. game title, timestamp of play, wifi connectionstatus, screen brightness), game profile (e.g. game genre, developer, number of downloads, rating), and user information (e.g. device model, region, OS version). Nonetheless, the above studies focus on improving prediction by changing models or the definition ofcalculation.

On the other hand, some recent studies proposed another different approach to customer segmentation. Customer segmentation is a growth-oriented tool that Wendell Smith created. When mass-market strategy became more challenging to succeed due to not satisfying all customers in the market, customer segmentation was proposed to address the issue(Fu et al., 2017). It divides a market into distinct customer groups by similarities to create marketing strategies more efficiently (Smith, 1956). This is considered one of the most effectivetools for marketing to uncover market opportunities, determine customer needs and wants, and disclose potential markets (Yi, 2017). Therefore, customer segment retention analysis can illuminate risks or opportunities. For example, research proposed a churn prediction method by selecting the target and setting a target threshold (Lee et al., 2018). The proposed reason is that the high churn rate and customer lifetimevalue is skewed, so focusing on churn prediction for loyal customers is more cost-effective.

In addition, they are setting a threshold to maximize the expected profit rather than maximizing the accuracy. Hence, the prediction performance results include an expected profit for each model.This study uses data of in-game activities (e.g. the number of days user play, playtime, achievement points, amount of money user gain). Similar to the above study, some research suggests calculating the churn rate for user clusters. A study proposed a framework with the original joint model, and one part of the framework is the prediction of dropout probabilities. The model also used K-mean clustering to segment users and then predicted dropout Probabilities (Banerjee et al., 2019). Another segmentation with a churn prediction model is proposed by Fu et al. (2017). It first creates user segments by stickiness-based FCM, then evaluates the retention trend for each cluster. Engagement, performance, and social features areused. Thus, there are many perspectives and approaches to churn prediction in gaming. The common approach is that all study use user behaviour data for prediction, and the different game category (online or mobile) does not affect the approach that much.

Table 2. Summary of Churn Prediction Literatures in DGBL and Gaming Industries

Citation	Category	Purpose	Variables Used for Prediction	Mo del	Result
(Yan, Lee & Ko, 2019)	DGBL	Prediction of Abandonment in Tutorial	Cumulative features (e.g. idle time,execution button clicks), Learner features (e.g. age, gender, registration status)	Logistic Regression(LR), Random Forest(RF), Gradient Boosting Decision Tree (DGBT)	average AUC: 0.68
(Hadiji et al., 2014)	Online/Mobile Game	Prediction of Churn	Number of sessions, Number of days, average playtime per session, averageplaytime between sessions, etc.	LR, Decision Tree (DT), Naive Bayes (NB), Neural Networks(NNs)	Model: DT F1 Score: 0.916
(Xie et al., 2015)	Online/Mobile Game	Prediction of Disengagement	Event frequency	Support vector machines(SVM), DT, LR	AUC: 0.5 to 0.7
(Tamassia et al., 2016)	Online Game	Prediction of Churn	Activities, performance, achievements	Hidden Markov Model	AUC: 0.77
(Milošević, Živić & Andjelković, 2017)	Mobile Games	Prediction of Early Churn	Activity features (e.g. playtime, sessioncount), Monetization features (e.g. in-gamemoney spend), Gameplay style features (e.g. auction usage)	LR, DT, RF, NB, GradientBoosting	AUC LR: 0.79 DT: 0.67 RF: 0.8 NB:0.78 Gradient Boosting: 0.83

(Suh & Alhaery, 2016)	Online Game	Prediction of Churn	Recency, Frequency, Monetary value, Length of Relationships, Inter-play, Bonuses/Rewards, Demographic	E-CHAID Decision Tree	AUC: 0.88
(Liu et al., 2018)	Mobile Game	Prediction of Churn	Play history (e.g. timestamp of play) Game profiles (e.g. game genre, developer, rating) User information (e.g. device model, OS version)	Proposed semi-supervised deep neural network model, LR, RS, DT, RF, SVM	Best Model: Proposed semi-supervised deep neural network model AUC: 0.82
(Lee et al., 2018)	Online Game	Prediction of Churn	In-game activities (e.g. number of days user play, playtime)	RF, XG Boost (XGB), Generalized Boosting Regression (GBM)	For total customers AUCRF:0.9358 XGB:0.9264 GBM:0.9067
(Banerjee et al., 2019)	Mobile Game	Prediction of Churn	Activity indicators, Activity time, Engagement indicators	Proposed joint model (CEZIJ model)	FP / FN Activity Indicators: 5.86%/4.12% Engagement indicators: 3.54%/1.47%

(Fu et al., 2017)	Online Game	Prediction of Player Lifetime	Engagement features (e.g. login frequency, average playtime), Performance features (e.g. level, coins), Social interaction features (e.g. number of in-game friends)	Stickiness based FCM for clustering NB, Radial basis function (RBF) network for prediction	Stickiness based FCM withNB and RBF has better accuracy

2.3.3 Churn and Retention Prediction in Education

In the education industry, student failure or dropout prediction has been a common topic. Several studies of prediction with machine learning have been reported. Despite havingthe same aim, the data used for prediction is different from the gaming industry. The common data collection is time-consuming questionnaires (Costa et al., 2017). Earlier research has proposed a prediction of academic failure of on-campus with EducationalData Mining (EDA), and it used decision trees and induction rules (Marquez-Vera, Morales &Soto, 2013). It used two survey results and a dataset from the school. Therefore, there are many variables such as demographic data, family information, and academic grades and scores. This study concluded that prediction models performed with relevant accuracy.

Another study to predict student failure also used three survey data (Khobragade & Mahadik, 2015). One is for personal and family-related information (e.g. parents' occupation, income, number of family members), the second is for previous education (scores of multiple subjects from past education), and the third is for academic factors such as marks. The selected 11 variables are fed into NNge, OneR, SimpleCart, Random Tree, and NB to compare the result. Among the five models, NB has the highest accuracy of 87.12%. As for the non-survey data use, a work used non-survey data to predict retention rate from student enrollment data of a university (Bingham & Solverson, 2016). It focused on using demographic data such as gender, race and ethnicity, high school rank instead of academic and social engagement, and LR was chosen as an algorithm. The prediction model performance is proven with an accuracy of 83.2%.

However, these predictions are for on-campus students and not for online courses. A work proposed a prediction of online program dropout (Yukselturk, Ozekes & Türel, 2014). It conducted five surveys for data collection and collected demographic variables such as age, gender, previous online experience, self-efficacy, readiness for online learning, prior knowledge about online program courses, and locus of control. It then used DT, NB, Neural network (NN), K-Nearest Neighbor (KNN) algorithms for prediction. Both AUC and accuracy were the highest with the KNN algorithm, 0.866 AUC and 87% accuracy.A study approached an evaluation of the effectiveness of student failure prediction (Costa et al., 2017). It used non-survey data from two datasets from distance learning and on-campus. On-campus data contains age, gender, civil status, exam performance, number of correct exercises, amount of exercise performed. Distance learning data contains age, gender, civil status, the performance of exams, and assignments. This is different from past research because the data contains user behaviours such as access frequency, participation in the forum, and blog use.

Nevertheless, this isn't easy to apply digital game-based learning services and products because it uses on-campus data, which is not available for commercial services and products. The arrival of MOOCs gave a different approach to predicting churn or dropout by using more user behaviour data like the gaming industry. The research focused on the early prediction of dropout of MOOCs (Xing et al., 2016). C4.5 decision tree, general Bayesian network (GBN), and the ensemble learning method called stacking generalization are compared. The stacking of C4.5 and GBN outperformed the base algorithm alone. The average precision of the stacking model is91.7%, whereas GBN and C4.5 are 89.7% and 89.6%. The average AUC of the stacking model is 90.7%, and GBN and C4.5 are 89.0% and 86.3%.
Another study proposed a prediction model on the decrease of engagement with three different engagement features for MOOCs (Bote-Lorenzo & Gómez-Sánchez, 2017). Video engagement is the average percentage of videos watched, including partially. The averaging percentage of exercises calculates exercise engagement. Assignment engagement is computed by the averaging percentage of the assignments submitted. LR, RF,SVM, and stochastic gradient descent (SGD) are used as prediction algorithms, and for all engagement, SGD outperformed other algorithms. Even though there are many approaches in education, the different data sources (survey, school database, and MOOCs database) have different purposes and churn definitions. Surveydata or on-campus studies are unsuitable for digital game-based learning services or products. The combination of MOOCs and the gaming industry seems the best approach for digital game based-learning.

Table 3. Summary of Churn Prediction Literatures in Education

Citation	Category	Purpose	Data Source	Variables Used for Prediction	Model	Resu lt
(Costa et al., 2017)	Online / On-Campus	Early Prediction of Failure	Online course and school Database	On-campus data (e.g. age, gender, civil status, exam performance) Distance learning data (e.g. access frequency, participation in the forum)	Support vector machines (SVM), DT via J48, Neural Network, Naive Bayes (NB)	Best Model: DT F-measure: 0.82 for the first exam,and 0.79 for the second exam
(Yukselturk, Ozekes & Türel, 2014)	Online	Prediction of Dropout	Survey	Demographic variables, self-efficacy, readiness, prior knowledge, and locus of control	k-Nearest Neighbour (KNN), DT, NB, and Neural Network (NN)	Best Model: KNNAUC: 0.866 Accuracy: 87%
(Bingham & Solverson, 2016)	On-Campus	Prediction of Retention Rate	School database	Gender, residency status, ACT composite score, high school class rank, race/ethnicity, student, etc.	Logistic Regression (LR)	83.2% Accuracy
(Marquez-Vera, Morales & Soto, 2013)	On-Campus	Prediction of Dropout and Failure	Survey, School Database	Scores of each subject, level of motivation, GPA, smoking habits, physical disability, etc.	JRip, NNge, OneR, Prism, Ridor, ADTree, J48, RandomTree, REPTree, SimpleCart	Accuracy Prism: 94.4 ADTree: 96.6 SimpleCart: 96.6
(Khobragade & Mahadik, 2015)	On-Campus	Prediction of Failure	Survey	Personal and family-related (e.g. age, parents occupation), previous education (e.g. scores of multiple subjects of previous education), academic results (e.g. scores)	NNge, OneR, SimpleCart, Random Tree, NB	Accurac y NB: 87.12

(Xing et al., 2016)	MOOCs	Temporal Prediction of Dropout	MOOCs Database	Clickstream (which pages students visited and when or how many times students clicked on certain sources (e.g., syllabus, modules, quizzes, etc.)) quiz scores and discussion forum data	General Bayesian Network (GBN), decision tree (C4.5), Stacking	Model: Stacking AUC: 90.7% Precision: 91.7%
(Bote-Lorenzo & Gómez-Sánchez, 2017)	MOOCs	Prediction of Decrease of Engagement	MOOCs Database	Video engagement Exercise engagement Assignment engagement	LR, stochastic gradient descent (SGD), RF and SVM	Best Model: SGDvideo AUC: 0.81 - 0.894 exercise AUC:0.837 - 0.906 assignment AUC:0.718 - 0.914

2.4 Summary of Literature Review

Previous research in digital game-based learning focuses on evaluating the relationships between learning effectiveness, motivation, different courses, and gaming design.At the same time, Japan's research is more limited and not as diverse as overseas. The only digital game-based learning research for churn prediction specialized for tutorialchurners before registration. This means that there are preliminary researches in digital game-based learning for the churn definition and the common approaches of churn prediction. Because digital game-based learning combines education and the online gaming industry, both gaming and education research are reviewed.

The online gaming industry often applies one of the four retention calculation methods for retention analysis. The four methods are Full retention, classic retention, accumulative retention, and return retention. With these methods, churn is also determined depending on the login timing, and the common evaluation period is 7, 14, or 30 days. On the other hand, the educational definition of churn is simpler than the student has completed or not during the course duration. Nonetheless, the course duration varies in education from weeks to years, and the variation makes it difficult to have a clear evaluation period like the onlinegaming industry. The common approach of churn prediction in the gaming industry uses user activity data for machine learning models. In education, churn prediction for academic purposes uses various data such as surveys, on-campus records, or family information. However, MOOCs have a similar approach of using user activity for churn prediction. The common machine learning models for churn prediction in online gaming and MOOCs are LR and tree-based models. The combination of MOOCs and the gaming industry seems the best approach for digital game based-learning. By considering the current and expected market expansion of the digital game-based learning market, the demands of the marketing approach will presumably be increased. Nonetheless, themarketing approach in digital game-based learning is still in the early phase. Lack of churn definition and preliminary churn prediction research in digital game-based learning makes itdifficult for these companies to have a marketing approach, especially in retention or churn analysis. In addition, there is a gap between online gaming and education approaches, and the one industry knowledge is not wholly applicable to digital game-based learning. Thus, the gapneeds to be filled for digital game-based learning.

SECTION 3: RESEARCH METHOD

Cross-Industry Standard Process for Data Mining (CRISP-DM) is applied for research methodology. The CRISP-DM has six steps to structure and guide the DM process and is used in real environments (Mariscal, Marbán & Fernández, 2010). Because this project is addressing business issues, CRISP-DM is selected. The six steps are business understanding, data understanding, data preparation, modelling, evaluation, and deployment. Python programming language is used for data understanding, preparation, modelling, and evaluation. The reason for choosing Python is that SAS, R, and Python are the most common programming languages. Still, Python is the most used in the technology industry and is the dominant language (69%) among data scientists (Burtch Works, 2018).

Also, all the steps which require a PC are operated on the following system.

- OS: Mac OS Catalina (version 10.15.4)
- CPU: 2.3 GHz Intel Core i5 Quad core
- RAM: 16GB
- Graphics: Intel Iris Plus Graphics 655 1536 MB

Before starting data selection, setting up the PC with installing the required software and packages should be operated.

3.1 Method of Data Understanding

3.1.1 Method of Data Collection and Exploration

User data is provided by the Japanese company, which is primary data. The company gave access to Redash that is an open-source tool to query and visualize the database contents. Two data types are available: structured data from the customer relationship management (CRM) database and semi-structured data in a JSON format. By using SQL through Redash, access to the available data is allowed. Hence, the data collection requires SQLcoding, and the download format is CSV. Data comprehension is crucial before starting any process. Therefore, data about data should be inspected, such as type, acceptable values, and range of values. The statistical and graphical approaches are conducted to see the minimum, maximum, mean, median, standard deviation, distribution and skewness for numeric data. For categorical data, the frequency distribution and skewness are checked. In this part, Python is used for data understanding to grasp the information of each variable. This exploration is done after the data aggregation because the modelling is done on the aggregated values.

3.1.2 Method of Definition of Churn

The majority of the definition of churn is course incompletion in education. Many academic online course durations are fixed weeks and have assignment submission or exams at intervals(Xing et al., 2016; Costa et al., 2017). In this case, churners are users who did not complete thework in the week or by the end of the course. However, commercial, educational content is often composed of a sequence of chapters or levels, and it is especially common in MOOCs (Bote-Lorenzo & Gómez-Sánchez, 2017). A study in digital game-based learning conducted anengagement analysis for each level (Yan, Lee & Ko, 2019). This also implies the generality ofchapters (levels) in game-based learning. The difficulty of using chapters is that there is no timelimitation. This means it is difficult to define if the player dropped out or not. The gaming industry churn definition can be more applicable to address this issue, which usesuser inactivity duration after the last login. By using inactivity duration as a cutoff, churn can be defined. A "churn window (C)" approach, which was claimed by Tamassia et al. (2016), can be used to define the cutoff. If there is no active data for a player during at least C weeks, the player is considered churned. The churn window is set based on the scattered plot of total playtime (sec) vs average absence from the game (in days). In the study set C=4 because the scattered plot density is much higher when the average absence is less or equal to 28 days (which means 4 weeks). Note that the defined churned users can return to the game with this calculation. If the above approach does not work well with the provided dataset, such as the evenly scattered data or no specific cutoff, an arbitrary time frame for each user is calculated to definechurners (Suh & Alhaery, 2016). It uses the number of inactive days from the last play (recency),average days between play (avg between), and standard deviation of days between play (std between). These numbers are used to calculate in the following equation:

- Recency > avg between + 2 * std between

3.2 Method of Data Preparation

3.2.1 Method of Selection of Data

Based on past churn prediction studies that use user behaviour, there are some common categories of variables. Because digital game-based learning has user behaviour data similar to the gaming industry, most data selection is influenced by the gaming industry. Some are referred from MOOCs in education. The majority of research in the gaming industry uses the cumulative number of users such as the number of logins, plays, or clicks for engagement (Hadiji et al., 2014; Milošević, Živić & Andjelković, 2017; Suh & Alhaery, 2016; Liu et al., 2018; Lee et al., 2018; Banerjee et al., 2019; Fu et al., 2017). The average values or duration are commonly used (Suh & Alhaery,2016; Hadiji et al., 2014). User activity log variables such as user id, timestamp, and contents should be selected from the JSON user activity log to calculate these values. In addition, performance features such as levels and scores are often used (Fu et al., 2017; Suh & Alhaery,2016; Xing et al., 2016; Bote-Lorenzo & Gómez-Sánchez, 2017). Levels and number of coins are selected for the performance features. Demographic information is commonly used in education and few in the gaming industry (Yukselturk, Ozekes & Türel, 2014; Costa et al., 2017; Yan, Lee & Ko, 2019; Suh & Alhaery, 2016). So, gender, date of birth, resident area areselected from CRM data. At last, merging data will be conducted in the data integration sectionafter data transformation.

3.2.2 Method of Data Cleaning

Since a company provides the dataset, the selected data is expected to be dirty. Data cleaning is necessary to enhance data quality (Han, Kamber & Pei, 2012). Three types of data should be detected and cleansed: missing, noisy, and inconsistent data. Noisy data means having errors or outliers. Inconsistent data has conflicts in the same variables. For example, having a different data format for the same variables such as “2019/12/25” and “25/12/2019”. Finding possible outliers can be done by data visualization with boxplot using DataFrame.boxplot in the Python *pandas* library. According to the pandas document (2020), the outlies in the box plot are determined by using Interquartile Ranges (IQR). If a datapoint isbeyond IQR * 1.5, it is illustrated as an outlier. The outliers should be judged to be removed ornot based on the variables. For missing values, if the missing percentage is greater than 50% for a variable, skipping using the variable is necessary to avoid the inaccurate result. Else, if thevariable is numerical, the common approaches are filled with single or multiple variables, meanor median. For a normal distribution, the mean can be used. For skewed distribution, the medianshould be used. If the categorical variable, the missing value should be replaced by the mostfrequent value.

3.2.3 Method of Data Aggregation, Transformation, and Integration

For data transformation, Python is used. Discovered inconsistent values and most errors requiredata transformation to correct them (Han, Kamber and Pei, 2012). In addition to errors and inconsistent values transformation, data format and language are expected to be handled. For example, most of the data are expected to be structured, but some of the data may be in semi- structured like JSON, requiring transformation. All categorical data in English and Japanese should be transformed into numerical data. Additionally, data aggregation is expected to create new variables. The standard calculations and transformations are as follows.

- Churn status (Target value)
  - The player is churned or not for each chapter
- Demographic features
  - Age calculated from the date of birth
  - Gender from Japanese to English
  - Prefecture in English retrieved from postal code
- Engagement features
  - The accumulated days of login frequency
  - The duration between the data of the new chapter release and when the userstarted the new chapter
  - The duration between the first and the last login date
  - Accumulated number of repetition of the same contents
- Performance features
  - Total accumulated time consumption of all contents
  - Average time consumption of each login
  - Accumulated time consumption for each chapter
  - Total accumulated inactive time between logins
  - Average inactive time between logins
  - Aggregated number of coins gained
  - Aggregated exp gained

Python library called *pandas* for data manipulation, and analysis was mainly used for the aggregation. The *datetime* library to calculate age, the *regular expression operation* libraryto compare the string, and *NumPy* library to calculate average wait days are used. Additionally,to acquire the 47 Japanese prefectures in English, the API called *postal-code-api* is used with the python *Requests* library, which allows making HTTPS requests. The API returns prefecture, address1, address2, address3, and address4 based on the postal code in English and Japanese (Miyauchi, Hosoya & Urabe, 2019). After the aggregation, the variables with outliers then need to be handled. Unless the value is not important, the variable should be standardized to reduce the impact of outliers. The function called *StandardScaler* in the *scikit-learn* preprocessing library is used for standardization. Thesame library is used for label encoding on the categorical variables, gender and prefecture, to label encode to the numeric values. After data transformation, the demographic dataset from CRM and user activity dataset will be merged into one dataset using the user ID.

3.2.4 Method of Feature Selection

After the data aggregation, transformation, and integration, the variables shown in Table 4 are in the generated dataset. However, feature selection may improve model performance. Many past studies related to churn prediction in gaming and education industries have used manual feature selection, such as using their experience and original calculations depending on the variables. Nevertheless, two past studies, which are the prediction of dropout of MOOCs (Xing et al., 2016) and prediction of the player lifetime (Fu et al., 2017) used Principal Component Analysis (PCA). PCA is one of the common feature selection methods. It combinesinput variables and creates new variables with greater meaning by retaining the most valuable parts of the input variables (Calabrese, 2019). The *PCA* function under the *scikit-learn* Pythonlibrary is used to determine principal components in the code. Hence, two datasets which are generated dataset (as displayed in Table 4) and dataset created by PCA are prepared to see a difference.

Table 4. Data Frame Before Feature Selection

Variable Name	Category of Features	Details
Total_login	Engagement	Total number of logins of the user
entire_period(days)	Engagement	The engagement period. Subtract first_login from last_login.
avr_ch_wait(days)	Engagement	The average period between open and start. Total wait divided by the number of chapters played
replay	Engagement	Total number of replay per user
total_playtime(min)	Performance	Total playtime of the user in minutes
total_inactive(min)	Performance	Total inactive time between logins
average_playtime(min)	Performance	Average playtime per login. Calculated by total_playtime divided by total_login
average_inactive(min)	Performance	The average inactive time between logins. Calculated by total_inactive divided by total_login
ch1_playtime(min)	Performance	Playtime of chapter 1
ch2_playtime(min)	Performance	Playtime of chapter 2
ch3_playtime(min)	Performance	Playtime of chapter 3
ch4_playtime(min)	Performance	Playtime of chapter 4
ch5_playtime(min)	Performance	Playtime of chapter 5
ch6_playtime(min)	Performance	Playtime of chapter 6
ch7_playtime(min)	Performance	Playtime of chapter 7
exp	Performance	Total exp points per user
coins	Performance	Total coins per user
gender	Demographic	Gender in binary 0 or 1 (0=Female and 1=Male)
age	Demographic	Age of the player
prefecture	Demographic	Prefecture from 0 to 47
churn_status	Target Variable	Churn status in binary. 0 or 1 (0=False and 1=True)

3.3 Method of Modeling

Python is used for the modelling part as well. The data needs to be split into training, validation, and test datasets for the modelling. There are three types of common split ratio as follows.

- 80% training, 10% validation, and 10% test
- 75% training, 15% validation, and 15% test
- 60% training, 20% validation, and 20% test

All three patterns of ratios are prepared and compared to see the performance difference of theprediction. The best performance split ratio will be chosen for the final modelling. However, splitting to a fixed ratio can sometimes cause biased results, especially if the dataset size is smaller. Because the dataset can be smaller after the aggregation, 10-fold cross-validation is used for performance comparison during the performance and hyperparameter tuning. *Scikit- learn* library provides to calculate AUC with 10-fold stratified cross-validation. If the target variable is binary, it automatically uses stratified cross-validation. Since the target value of churn prediction is binary, using stratified cross-validation is applied. To allow cross-validation, the dataset is split into two datasets by maintaining the common split ratio. The test dataset is kept aside until the final evaluation.

- 90% training (includes 80% training and 10% validation), and 10% test
- 85% training (includes 70% training and 15% validation), and 15% test
- 80% training (includes 60% training and 20% validation), and 20% test

As for algorithms, the three common algorithms for churn prediction are selected based on the literature. The algorithms are Logistic Regression (Milošević, Živić & Andjelković, 2017; Bote-Lorenzo & Gómez-Sánchez, 2017; Yan, Lee & Ko, 2019; Liu et al., 2018; Hadiji et al., 2014; Xie et al., 2015), Random Forest (Milošević, Živić & Andjelković, 2017; Bote-Lorenzo& Gómez-Sánchez, 2017; Yan, Lee & Ko, 2019; Liu et al., 2018), and Decision Tree (Milošević, Živić & Andjelković, 2017; Xing et al., 2016; Liu et al., 2018; Hadiji et al., 2014; Xie et al., 2015).

This literature uses user behaviour data in both the gaming and education industries. Logistic Regression (LR) is a probabilistic algorithm for binary classification (Milošević, Živić& Andjelković, 2017). LR has been used to predict historical data since the 1980s (Pokorná & Sponer, 2016). The mathematically clear output is one of the reasons for the popularity. The problem of LR is the difficulty with interpretation (Xie et al., 2015).

On the other hand, a Decision Tree (DT) is one of the most interpretable models by a human. DT is a classification algorithm used for supervised learning problems and generates a rule-basedhierarchical tree (Milošević, Živić & Andjelković, 2017). The tree displays the features connected to the terminal nodes, so the interpretation is simpler. Random Forest (RF) is anothertree-based classification algorithm that consists of many decision trees. RF constructs multipledecision trees by randomly sampling, and each tree conduct classification and the result. The RF then collects the outcomes from the trees and select the best result as a final result by voting(Mao & Wang, 2012). These are popular algorithms not only for churn prediction in gaming or education. Yang etal. (2018) also mentioned that the DT, RF and LR are the common techniques used in churn prediction. Therefore, the DT, RF, and LR are used for modelling.

3.4 Method of Evaluation

Using the 20% test data that was kept aside, the model is tested to prove validity. ROC AUC will compare the performance of the models. ROC curve is a probability curveof sensitivity, specificity, and performance measurement for classification (Jeni, Cohn & De La Torre, 2013). AUC is the area under the curve, and a higher AUC close to 1 means a larger area, representing a better classifier. Many studies used AUC as a metric to comparethe prediction models in both gaming and education industries (Milošević, Živić & Andjelković, 2017; Yan, Lee and Ko, 2019; Xie et al., 2015; Tamassia et al., 2016; Suh & Alhaery, 2016; Liu et al., 2018; Xing et al., 2016; Bote-Lorenzo & Gómez-Sánchez, 2017). Using ROC AUC is assumed that the common performance metrics accuracy is affected by the imbalanced data, whereas ROC AUC is not influenced by the imbalanced distribution (Jeni, Cohn & De La Torre, 2013). Because the proportion of the target value *churn_status* (0 and 1) is expected to be imbalanced due to having more churners in general, performance metric ROC AUC fits the case.

As mentioned in the previous subsection, mean AUC was calculated based on the 10-fold stratified cross-validation results. The 10-fold cross-validation is selected because it is the mostpopular evaluation in the past studies related to churn prediction in gaming or education (Tamassia et al., 2016; Yan, Lee & Ko, 2019; Hadiji et al., 2014; Xing et al., 2016).In addition, to improve the performance of each model, hyperparameter tuning is conducted.There are five hyperparameters selected for DT tuning: *criterion*, *splitter*, *max_depth*,*min_samples_split*, and *min_samples_leaf*. These hyperparameters tend to have a strong influence on DT. Similar to RF, three hyperparameters of the *max_depth*, *min_samples_leaf*, and *min_samples_split* are the same as DT. Another hyperparameter *n_estimators* is RF specific.For LR, there are two hyperparameters selected: *penalty* and *C*. These hyperparametersare checked by AUC performance with 10-fold stratified cross-validation to evaluate the difference.

SECTION 4 – IMPLEMENTATION

In this chapter, the whole implementation process is explained in detail.

4.1 Data Collection

Data collection was conducted through Redash web UI by using SQL. There are two types ofdata: semi-structured user log data in a JSON format and structured data from customer relationship management (CRM). Redash web UI allows accessing both datatypes by SQL. However, combining multiple tables data by the join statement and displayingthe results in Redash web UI caused heavy CPU load. The process hangs the Redash web UI due to the limitation of the server specification. In addition, the simple select statements fromthe user activity and lesson log are too huge to even process with the web UI. The following operation was suggested by consultation with an engineer in the Japanese company.

- Smaller table size: download CSV file per table by using Redash
- Big table size: Given in the form of SQL-format dump files

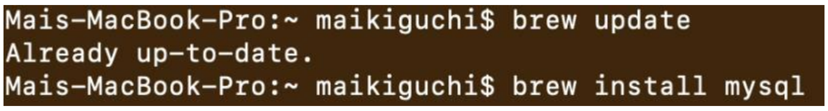
Downloaded data are summarized in Table 5. The learning contents of the Japanese service are split into seven chapters, and the next chapter is accessible after about seven days to a couple of weeks after finishing the previous chapter. Therefore, each player has a different chapter release, start and finish dates. Each user's chapter-related dates are stored in twofiles (ID 1 and 2). ID 1 hold the timestamp of release, start, and finish dates. ID 2 contains chapter 7 finish dates stored in a different table from ID 1. In addition, each chapter contains many small lessons, and players acquire coins and experience points called *exp* by finishing lessons. This obtaining history of *exp* and *coins* are stored in ID 3. If a player repeatsthe same chapter, the history of repetition data, such as chapter number and timestamp, are stored in ID 4. ID 5 file contains user demographic information from CRM.

Table 5. Downloaded CSVs

ID	File Name	Number of Observations	Description
1	player_chapter_waiting	6,973	Chapter related dates such as release, start and finish dates

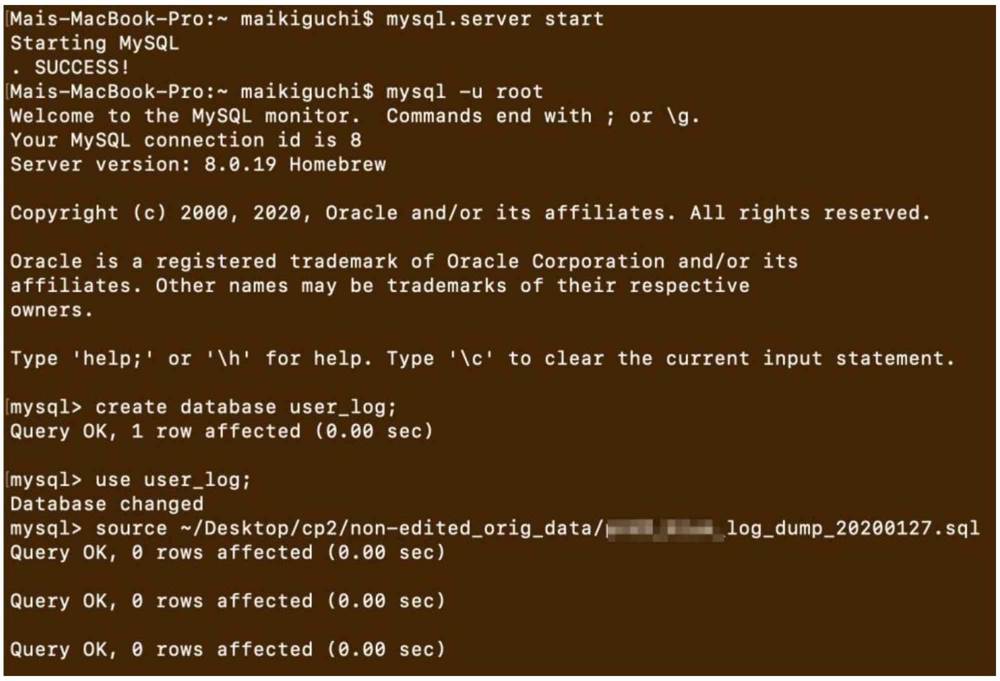
2	ch7_fin	514	Chapter 7 finish date
3	player_exp_coins	478,700	Historical exp and coins acquirement data
4	player_replay	1,496	Historical user replay data
5	crm_players	3,982	CRM data. Contains demographic information such as user name, email, address, birth date, and gender

The other user behaviour related log data are gained in the form of the SQL-format dump file. The given SQL file is compatible with AWS Relational Database Service (RDS) and MySQL.Thus, MySQL server and client are installed on the local computer, as shown in Figure 3. Afterthe installation, a database is created, and the SQL file is loaded using a source query (demonstrated in Figure 4). After the loading, the data is exported to CSV files, as displayed inFigure 5. The exported two files are summarized in Table 6. ID 6 contains all user activity logs except for learning lessons. For instance, logs are recorded when the user moved to a different area, gained a new item, or got access to a new area. The lesson log contains the useractivity log related to learning, such as the start and finish time of a lesson. Therefore, a total of seven CSV files are collected and are aggregated in the next subsections.



```
Mais-MacBook-Pro:~ maikiguchi$ brew update
Already up-to-date.
Mais-MacBook-Pro:~ maikiguchi$ brew install mysql
```

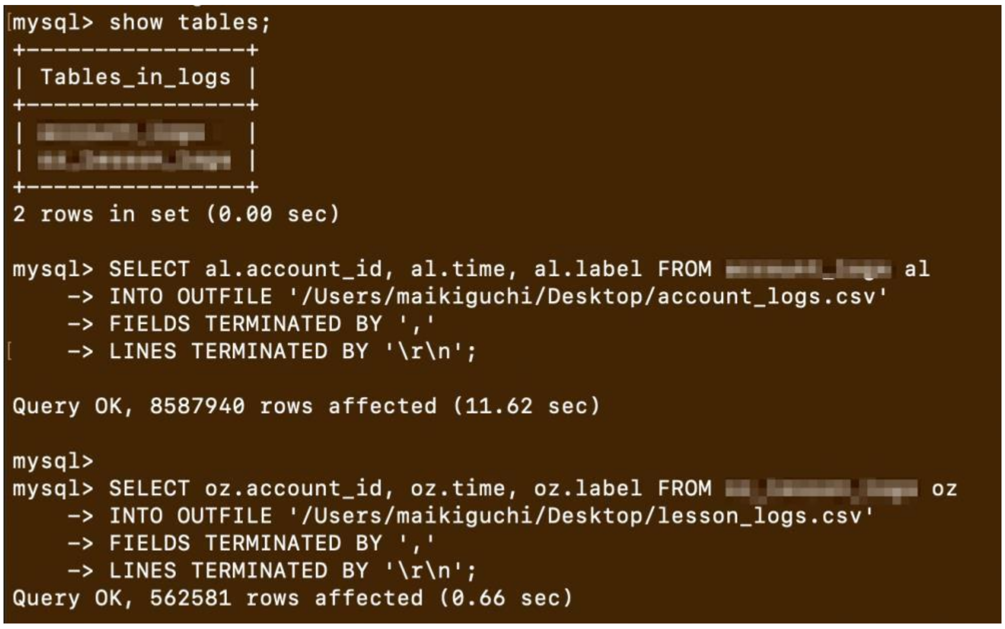
Figure 3. MySQL Installation Command



```
Mais-MacBook-Pro:~ maikiguchi$ mysql.server start
Starting MySQL
. SUCCESS!
Mais-MacBook-Pro:~ maikiguchi$ mysql -u root
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.19 Homebrew

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> create database user_log;
Query OK, 1 row affected (0.00 sec)

mysql> use user_log;
Database changed
mysql> source ~/Desktop/cp2/non-edited_orig_data/[redacted]log_dump_20200127.sql
Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.00 sec)
```

Figure 4. Creating a Database and Loading the SQL file

(a part of the file name is masked due to the information of the development code name)



```
mysql> show tables;
+-----------------+
| Tables_in_logs  |
+-----------------+
| [redacted]      |
| [redacted]      |
+-----------------+
2 rows in set (0.00 sec)

mysql> SELECT al.account_id, al.time, al.label FROM [redacted] al
    -> INTO OUTFILE '/Users/maikiguchi/Desktop/account_logs.csv'
    -> FIELDS TERMINATED BY ','
    -> LINES TERMINATED BY '\r\n';

Query OK, 8587940 rows affected (11.62 sec)

mysql>
mysql> SELECT oz.account_id, oz.time, oz.label FROM [redacted] oz
    -> INTO OUTFILE '/Users/maikiguchi/Desktop/lesson_logs.csv'
    -> FIELDS TERMINATED BY ','
    -> LINES TERMINATED BY '\r\n';
Query OK, 562581 rows affected (0.66 sec)
```

Figure 5. Exportation of the Two Tables into CSV file

(table name is masked due to the information of the production environment)

Table 6. SQL-format dump data

ID	File Name	Number of Observations	Description
6	account_log	8,587,940	User log of all activities except for learning contents related.
7	lesson_log	562,581	User log in the learning contents such as start and finish date-time of a lesson

4.2 Data Preparation

In this section, the implementation of data preparation before modelling is explained. It starts with data aggregation, then EDA and data cleaning, and the last is feature selection. The overview of the data preparation flow is illustrated in Figure 6. All the programming code is writtenin Python, and some graphical plots are by Tableau. The entire Python code is attached in Appendix A as Figure 44.

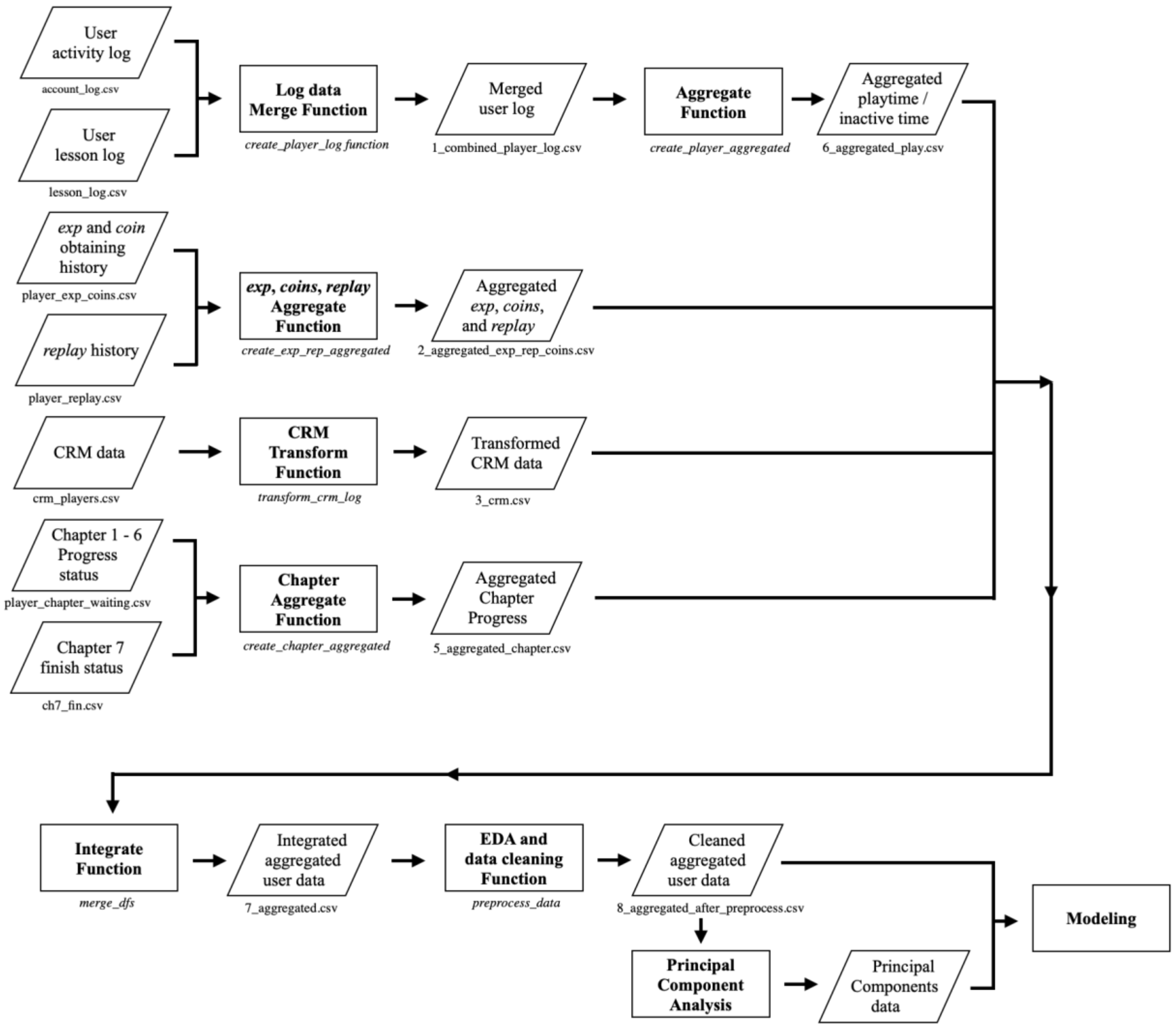

User activity log
account_log.csv
User lesson log
lesson_log.csv
Log data Merge Function
create_player_log function
Merged user log
1_combined_player_log.csv
Aggregate Function
create_player_aggregated
Aggregated playtime / inactive time
6_aggregated_play.csv
exp and coin obtaining history
player_exp_coins.csv
replay history
player_replay.csv
exp, coins, replay Aggregate Function
create_exp_rep_aggregated
Aggregated exp, coins, and replay
2_aggregated_exp_rep_coins.csv
CRM data
crm_players.csv
CRM Transform Function
transform_crm_log
Transformed CRM data
3_crm.csv
Chapter 1 - 6 Progress status
player_chapter_waiting.csv
Chapter 7 finish status
ch7_fin.csv
Chapter Aggregate Function
create_chapter_aggregated
Aggregated Chapter Progress
5_aggregated_chapter.csv
Integrate Function
merge_dfs
Integrated aggregated user data
7_aggregated.csv
EDA and data cleaning Function
preprocess_data
Cleaned aggregated user data
8_aggregated_after_preprocess.csv
Principal Component Analysis
Principal Components data
Modeling

Figure 6. The Overview of Data Preparation Flow

4.2.1 Data Aggregation

There are many steps to complete the data aggregation part, including data transformation anddata integration. Hence, each step is explained in the following subsections.

4.2.1.1 User Activity and Lesson Log Integration

First, ID 6 and ID 7 logs are similar data: a user behaviour log with the *action* and *timestamp*. Both of them only contain the same variables, which are *account_id*, *timestamp*, and*action* of rough category. These two files should be combined to calculate more accurate aggregated values such as playtime and the number of logins. In Python code, a function named *create_player_log* handles this integration. First, both CSV files are imported as a data frame by setting variable names, as shown in Table 7. ID 6 contains the player's activity log, but the *action* named "crm_update_player" is a player's parent log of changing the player’s CRM information. The parent account can manage the player’s information and payments for miners who do not have credit cards. The parent log is excludedbecause this is not the player’s behaviour. After that, two data frames are merged byID and sorted by in the order of ID then timestamp, so the merged data contains all logs related to players. The merged data frame is exported as “1_combined_player_log.csv”. After this process, all player activities and lesson logs are combined into one data. This data willbe used for aggregation to calculate playtime, idle time, and the number of logins in the playerdata aggregation process.

Table 7. Variables in *1_combined_player_log.csv*

Variables	Data Type	Details	Example
id	string	Unique ID for each user	a0b12345-c67d-8901-234e-f5g6789hi01k
timestamp	string	The date and time of action	2018-05-19 10:07:36
action	string	Categorical rough action	lesson_finished

4.2.1.2 *Exp, coins,* and *replay* Data Aggregation

Secondly, *exp*, *coins*, and a number of *replays* need to be aggregated and calculated the total value for each user. The *create_exp_rep_aggregated* function in python code handles this part. ID 3CSV file contains nine variables, and three variables are imported to calculate aggregated *exp* and *coins* (shown in Table 8). The ID 4 CSV file has seven variables, and two are imported(displayed in Table 9). Since the original data contains a history of obtaining *exp* and *coins*, the total values of each user are calculated by using *pandas group_by*. For replay, the *nunique* function calculates how many times the user did replay. Figure 7 shows the part of aggregation code from *create_exp_rep_aggregated* function. These aggregated values (*exp*, *coins*, and *replay*) are merged into one data frame as displayed in Table 10 and stored in “2_aggregated_exp_rep_coins.csv”. The aggregated data of exp, coins, and replay for each user is created from this process. This data will be merged with other aggregated data at the data integration step.

Table 8. Imported Variables from ID 3 player_exp_coins.csv (Before Aggregation)

Variable Name	Data Type	Details	Example
account_id	string	Unique ID for each user	a0b12345-c67d-8901-234e-f5g6789hi01k
exp	int	Gained exp points per lesson	100
coins	int	Gained coins per lesson	200

Table 9. Imported Variables from ID 4 *player_replay.csv* (Before Aggregation)

Variable Name	Data Type	Details	Example
account_id	string	Unique ID for each user	a0b12345-c67d-8901-234e-f5g6789hi01k
START	int	The start time of replay	2018-11-14T04:25:25

```
# aggregate exp and coins per id, then merge them
exp_df.set_index("id")
exp_agg_df = exp_df.groupby(["id"])["exp"].sum()
coins_agg_df = exp_df.groupby(["id"])["coins"].sum()
exp_df = pd.merge(exp_agg_df, coins_agg_df, on="id", how="left")

# count how many times replay, then merge with exp_df
rep_df = rep_df.groupby("id")["replay"].nunique()
exp_agg_df = pd.merge(exp_df, rep_df, on="id", how="left")
```

Figure 7. Python Code: *id*, *exp*, and *replay* Aggregation

Table 10. Variables in 2_aggregated_exp_rep_coins.csv (After Aggregation)

Variable Name	Data Type	Details	Example
id	string	Unique ID for each user	a0b12345-c67d-8901-234e-f5g6789hi01k
exp	int	Total exp points per user	35600
coins	int	Total coins per user	37305
replay	float	Total number of replay per user	2.0

4.2.1.3 CRM Data Transformation

Next, ID 5 CRM data needs to be transformed. The original data have 18 variables and seven imported variables (Table 11). The *transform_crm_log* function deals with thetransformation process. The *gender* is transformed from Japanese to English, and *age* is calculated based on the birthdayin the *calculate_age* function, which uses a *datetime* library. The *prefecture* is retrieved in *postal_2_en_pref* function by using an open-source API. The Python code is presented in Figure 8. The *request* library allows HTTPS requests and the API named *postal-code-api* returns the prefecture information. The returned values are in Japanese and English, and JSON format. So, the English prefecture name is obtained from the returned JSON. After transforming the three variables, test accounts should be removed from the CRM data. The *player_email*, *parent_email*, and *address* are used to differentiate and remove test accounts bycomparing the string. The three variables are then dropped from the data frame, and the remaining variables summarized in Table 12 are stored in "3_crm.csv". The test accounts are saved to "4_test_accounts.csv" for later test account removal processes (as shown in Table 13).This transformation process altered *gender* from Japanese to English, calculated *age* from birthdate, and retrieved English *prefecture* from postal code. These transformed data will be merged with other aggregated data at the later data integration process.

Table 11. Imported Variables from ID 5 *crm_players.csv* (Before Transformation)

Variable Name	Data Type	Details	Example
account_id	string	Unique ID for each user	a0b12345-c67d-8901-234e-f5g6789hi01k
player_email	string	Player's email address	example@example.co.jp
parent_email	string	The email address of parent account (If exist)	example@example.co.jp
gender	string	Gender in Japanese (Male or Female)	男性
birthday	string	Birthday of the player (yyyy/mm/dd)	1997/03/12
postal	string	Postal code of Japanese address	100-0001
address	string	The player's address	東京都千代田区千代田 1-1

```
# get English prefecture name from postal code
def postal_2_en_pref(postal):
    # Using API to get English information
    response = requests.get("https://madefor.github.io/postal-code-
api/api/v1/" + postal[:3] + "/" + postal[4:8] + ".json")
    if response.ok:
        results = response.json()["data"]
        prefecture = results[0]["en"]["prefecture"]
    else:
        prefecture = None

    return prefecture
```

Figure 8. Python Code: Postal Code Transformation Function with API

Table 12. Variables in *3_crm.csv* (After Transformation)

Variable Name	Data Type	Details	Example
account_id	string	Unique ID for each user	a0b12345-c67d-8901-234e-f5g6789hi01k
gender	string	Gender in English (Male or Female)	Male
age	int	Age of the player	36
prefecture	string	Prefecture name in English	Tokyo

Table 13. Variables in 4_test_accounts.csv

Variable Name	Data Type	Details	Example
account_id	string	Unique ID of test accounts	a0b12345-c67d-8901-234e-f5g6789hi01k

4.2.1.4 Chapter Progress Data Aggregation

The next process is chapter progress data aggregation. ID 1 contains the variables shown in Table 14, which has the timestamp of open, start, and finish times for each chapter. The numberof records per user depends on the progress, and it does not have the finish date of chapter 7. Chapter 7 is the last chapter of the learning contents, and the finish date is stored in ID 2 (shownin Table 15). Combining it with the other chapter status is necessary to grasp how many users have finished playing the entire content.

This process is managed in the function *create_chapter_aggregated*. It reorganizes the contentsin ID 1 and ID2 CSV files for each user and calculates average wait days. Also, *timestamp* variables sometimes contain a character T and Z in the string, so these characters are replaced with space. The function outputs the created data frame in CSV named “5_aggregated_chapter.csv” which is summarized in Table 16. After the chapter aggregation process, the test accounts are removed from the files below in the *remove_test_data*function.

- 1_combined_player_log.csv
- 2_aggregated_exp_rep_coins.csv
- 5_aggregated_chapter.csv

After this process, the aggregated chapter progress data without test accounts are generated, and the data will be integrated with other aggregated data in the data integration process.

Table 14. Imported Variables in ID 1 *player_chapter_waiting* CSV file

Variable Name	Data Type	Details	Example
account_id	string	Unique ID for each user	a0b12345-c67d-8901-234e-f5g6789hi01k
chapter	int	Chapter number	3
cleared	string	The date the user finished the previous chapter	2019-03-05T17:49:05
opened	string	The date the chapter is released (available) to the user	2019-03-15T19:00:00Z
begined	string	The date the user started playing the chapter	2019-03-15T23:52:37
days_wait	float	The period between cleared and opened	10.0486
days_begin	float	The period between opened and begined	0.2028

Table 15. Imported Variables in ID 2 *ch7_fin* CSV file

Variable Name	Data Type	Details	Example

account_id	string	Unique ID for each user	a0b12345-c67d-8901-234e-f5g6789hi01k
chapter	int	Chapter number. All chapter is 8 in this CSV file.	8
cleared	string	The date the user finished the previous chapter (means chapter 7)	2019-03-05T17:49:05

Table 16. Variables in *5_aggregated_chapter.csv* (After Aggregation)

Variable Name	Data Type	Details	Example
id	string	Unique ID for each user	a0b12345-c67d-8901-234e-f5g6789hi01k
ch1_fin	string	Timestamp when the user finished chapter 1	2019-02-02 07:57:00
ch2_open	string	Timestamp when chapter 2 is available to the user	2019-02-08 15:44:05
ch2_start	string	Timestamp when the user started chapter 2	2019-02-18 19:00:00
ch2_fin	string	Timestamp when the user finished chapter 2	2019-02-28 19:00:00
ch3_open	Same as ch2_open		
ch3_start	Same as ch2_start		
ch3_fin	Same as ch2_fin		
ch4_open	Same as ch2_open		
ch4_start	Same as ch2_start		
ch4_fin	Same as ch2_fin		
ch5_open	Same as ch2_open		
ch5_start	Same as ch2_start		
ch5_fin	Same as ch2_fin		
ch6_open	Same as ch2_open		
ch6_start	Same as ch2_start		
ch6_fin	Same as ch2_fin		

ch7_open	Same as ch2_open		
ch7_start	Same as ch2_start		
ch7_fin	Same as ch2_fin		
ch2_wait(days)	float	The period between open and start for the chapter	3.0283
ch3_wait(days)	Same as above		
ch4_wait(days)	Same as above		
ch5_wait(days)	Same as above		
ch6_wait(days)	Same as above		
ch7_wait(days)	Same as above		
avr_ch_wait(days)	float	The average period between open and start. Total wait divide by the number of chapters played	0.2028

4.2.1.5 Player Data Aggregation

The last aggregation is conducted on the player log data, which was created in the subsection 4.2.1.1 (User Activity and Lesson Log Integration). The file contains three variables: id, timestamp, and actions, sorted by *id* then *timestamp*. To calculate playtime and number of logins, two timestamps are compared, and the difference is aggregated as playtime or idle time. A function called *create_player_aggregated* operates this player data aggregation. The churn status of each user is also defined in this function. Churn determination's trial and error approach is explained in chapter 5.1 (Churn Determination) with details.The following rules are applied to determine whether the user is still playing or has left.

- If the difference between two timestamps is equal to or less than 60 minutes, consider the user still playing. The difference is treated as playtime.
- If the difference between the two timestamps is greater than 60 mins, consider the playerhas left. The difference is treated as idle time and increment the number of login by one.

In addition, the playtime of each chapter is calculated by using "5_aggregated_chapter.csv" data. For instance, if the timestamp is after the chapter 7 start date, the playtime is considered

As chapter 7 playtime, the variables shown in Table 17 are generated by this process and exported to “6_aggregated_play.csv” Consequently, the data after this process have aggregated playtime, inactive time, number of logins and the entire period of the total average, or by chapter. The data will be integrated with theaggregated data in the previous subsections.

Table 17. Variables in *6_aggregated_play.csv* (After Aggregation)

Variable Name	Data Type	Details	Example
id	string	Unique ID for each user	a0b12345-c67d-8901-234e-f5g6789hi01k
total_playtime(min)	float	Total playtime of the user in minutes	6249.23
total_login	int	Total number of logins of the user	117
total_inactive(min)	float	Total inactive time between logins	659481.57
average_playtime(min)	float	Average playtime per login. Calculated by total_playtime divided by total_login	53.41
average_inactive (min)	float	The average inactive time between logins. Calculated by total_inactive divided by total_login	5636.6
first_login	string	The first login timestamp. This also means the start day and time of chapter 1.	2018-10-04 21:54:32
last_login	string	The last login timestamp.	2020-01-10 05:25:20
entire_period(days)	int	The engagement period. Subtract first_login from last_login.	462
churn_status	string	Boolean churn status. True or False	True
ch1_playtime(min)	float	Playtime of each chapter	522.03
ch2_playtime(min)	float	Playtime of each chapter	1341.38

ch3_playtime(min)	float	Playtime of each chapter	716.85
ch4_playtime(min)	float	Playtime of each chapter	1026.03
ch5_playtime(min)	float	Playtime of each chapter	655.58
ch6_playtime(min)	float	Playtime of each chapter	634.33
ch7_playtime(min)	float	Playtime of each chapter	1353.02

4.2.1.7 Data Integration

After the player data aggregation, all aggregated data and CRM data are merged in the *merge_dfs* function. As presented in Table 18, since the number of observations (unique IDs) is slightly different for each data, the join type needs to be considered. A part of *merge_dfs* function is shown in Figure 9. First, the *exp_agg_df* is left joined to *agg_df* because it is adequate to have users having no *exp*, *coins*, and *replay* data. The *ch_agg_df* has one more observation than agg_df, which means the player only has chapter progress status but no playtime-related data. This is not valuable, so the *ch_agg_df* is left joined to *agg_df*. Lastly, the*crm_df* is left joined to the merged data frame due to having the latest user information than other data. This merged data frame is exported to "7_aggregated.csv". Consequently, this integrated data contains 3,701 players, all types of data, including playtime, obtained score, chapter progress, and demographic related information. The data will be succeeded to the next EDA and modelling process before modelling.

Table 18. Number of Unique ID per File

File Name	Variable Name in the Code	Number of observations	Details

2_aggregated_exp_rep_coins.csv	exp_agg_df	3,623	Less number of users because players who just started playing do not have exp, rep, and coin data until a certain point.
3_crm.csv	crm_df	3,837	The CRM data contains more users because the data contains all users until March 28th, 2020.
5_aggregated_chapter.csv	agg_df	3,702	Number of all users until January 27th, 2020
6_aggregated_play.csv	ch_agg_df	3,701	Number of all users until January 27th, 2020

```
# merge agg and ch_agg
    agg_df = pd.merge(agg_df, exp_agg_df, on="id", how="left")
    agg_df = pd.merge(agg_df, ch_agg_df, on="id", how="left")
    agg_df = pd.merge(agg_df, crm_df, on="id", how="left")
```

Figure 9. Python Code - Merge Data Frames

4.2.2 EDA and Data Cleaning

The dataset is ready for EDA and data cleaning. First, EDA in the *show_plots* function was conducted. All variables for modelling are printed statistical information with the *describe* function of the *pandas* library. The result table is split into two tables as presented in Tables 33-1 and 33-2 in Appendix C. It outputs the number of observations, mean, standard deviation, the percentile of 25%, 50%, and 75%, minimum and maximum values for numeric variables. Categorical variables show the number of observations, the unique number of observations, the most common values (top), and the frequency of the most common values. The *describe* function does not include the number of missing values, so the *isnull* function detects variables with missing values. In addition to the *describe* function, a graphical approach is also operated. Box plot for numeric variables and bar chart for the categorical variables are used. The meaning of the box plot is the whiskers as the most extreme non- outlier data, boxes as the quartiles, median as green horizontal lines, means as green triangularpoints, and fliers as outliers (McKinney & Pandas Development Team, 2020). The results are as shown in Figures 45 until 58 in

Appendix B. Based on the outputs above, data cleaning has been done in the *preprocess_data* function. Imputation of missing values is the first step. Six variables have missing values; *exp*, *coins*, *replay*, *gender*, *age*, and *prefecture*. According to the Japanese company, the observations with missing values are checked manually because there should not be data with missing values. The observations with missing values have the same patterns with no demographic information such as *age*, *gender*, *prefecture,* and no play log after chapter 1. However, some accounts seem to play *replay,* which should be available afterchapter 3—having a replay log but no log after chapter 1 should not exist. It turned out there are144 trial accounts created in terms of marketing for schools and campaigns. These trial accounts lack more than 70% of data and do not have reliable values because it uses debug features for test accounts to skip chapters or lessons to see different components.Therefore, these 144 accounts that do not have a *gender* variable are removed from the dataset. *exp*, *coins*, and *replay* values exist only for players who have played until a certain area in chapter 1, so newer players do not have this data. Hence, these missing values are filled as 0. For the *prefecture*, "Tokyo" is the most frequent prefecture based on the statistical results. NaN is replaced with "Tokyo". The *prefecture* is categorical values, so label-encoded by using *labelEncoder*. The value is set in alphabetical order. After imputation of missing values, *gender* and *churn_status* are label-encoded to the binary values. Female is 0, Male is 1 for gender, and False is 0, and True is 1 for *churn_status*.

Next, highly impossible values in *age* are handled. Highly impossible age is found based on the*describe* output and box plot. The minimum age is 0 and the maximum is 2010. Research about reading ability age was reviewed to determine the lowest possible age for the younger age. According to Japanese research, 60% of four-year-old children can read 80% of Hiragana eventhough the learning level varies depending on the family or day-care environment (Utashiro, Hashimoto & Hayashi, 2015). 80% of the reading ability of Hiragana is not perfect for playingthe service because there are other two more types of characters in Japanese (Katakana and Kanji).

Nevertheless, it is possible to start playing with parents' help. This implies that the ages below 4 should have less reading ability, making it more difficult to play the service. Thus,the ages below 4 are replaced by the median of 26. On the other hand, the ages above 100 are 119, 825, 972, 2002, 2004, 2009, 2010. The oldest age in January 2020 in Japan was 117 yearsold (Beachum, 2020), so the ages above 100 are also replaced with the median 26. Lastly, outliers need to be handled. Based on both statistical and graphical outputs, many variables have outliers. These outliers should not be deleted or replaced because these are the real values, such as playtime. Nonetheless, outliers have a strong impact on the modeling, so standardisation should reduce the impact. The standardization of the variables with outliers is conducted for these variables. The original dirty data frame is overwritten by the cleaned data frame and saved to "8_aggregated_after_preprocess.csv". After this process, the data is clean and is ready to be used for modelling.

4.2.3 Feature Selection

The dataset contains 3,557 observations of 21 variables classified into three categories of engagement, performance, demographic features, and a target variable. The engagement features are *total_login*, *entire_period*, *avr_ch_wait*, and *replay*. The performance features are playtime related variables such as *total_playtime*, inactive variablessuch as *total_inactive*, and experimental variables such as *exp*. This category includes playtime for each chapter. The third category is demographic, *age*, *gender*, and *prefecture*. The target variable is *churn_status*. These 21 variables are summarized in Table 4(in section 3.2.4 Method of Feature Selection). Except for the *churn_status*, 20 variables are processed for feature selection. The *PCA* function under the *scikit-learn* library is used. The number of components (*n_components*) is set to "mle" to find the best number of components by Minka's MLE (Minka, 2000). The generated principal components are 19, and these components are named from "PC-1" to "PC-19". The output of the explained variables is summarized and split into two tables of Table 34-1 and 34-2 in Appendix C. These principal components are stored as *x_features_pc* data frames. As a result, there are three datasets which are *x_features*, *x_features_pc*, and *y_target* as shown in Table 19. The*x_features* is the cleaned aggregated data without PCA, and *x_features_pc* is the principal component of *x_features*. Both datasets are used for modelling to compare the influence inperformance in the next subsection.

Table 19. Data Frame Used for Modeling

Data Frame Name	Details	Variable Name
x_features	20 variables from the original dataset except for ID and churn_status. Independent variables.	Total_login, total_playtime(min), total_inactive(min), average_playtime(min), average_inactive(min), entire_period(days), ch1_playtime(min), ch2_playtime(min), ch3_playtime(min), ch4_playtime(min), ch5_playtime(min), ch6_playtime(min), ch7_playtime(min), avr_ch_wait(days), exp, coins, replay, gender, age, prefecture
x_features_pc	19 principal components. Independent variables.	PC-1, PC-2, PC-3, PC-4, PC-5, PC-6, PC-7, PC-8, PC-9, PC-10, PC-11, PC-12, PC-13, PC-14, PC-15, PC-16, PC-17, PC-18, PC-19
y_target	One variable. Churn status of binary values (1=True and 0=False). Target variable.	churn_status

4.3 Modelling

After the feature selection, data frames are split into training, test, and validation datasets. Therelated past studies do not discuss the data splitting percentage. Hence, three patterns of data splitting are tried on both the original dataset (*x_features*) and PCA selected dataset (*x_features_pc*). The first pattern is 80% training, 10% validation, and 10% test. The second is 70% training, 15% validation, and 15% test. The last is 60% training, 20% validation, and 20% test. The validation data should be merged to training to use stratified 10-fold cross-validation for performance comparison of dataset and hyperparameter tuning. Thus, the splitting ratio of the training and test is modified. For example, 80% training, 10% test, and 10% validation is 90% training and 10% test for the optimization purpose. This splitting was handled in *split_data* to specify the above patterns. The split data are 6 patterns, and the example variable names and its details are summarized in Table 20.

To find out the best dataset and splitting percentage for modelling, creating a classifier and checking the performance with the function *cross_val_score* from the *scikit-learn* library was conducted in the function *DT_cross_val*, *LR_cross_val*, and *RF_cross_val*. The three machinelearning models are chosen based on the popularity in past studies. The training datasets, including the validation dataset part, are used for *cross_val_score* to calculate AUC with 10-fold stratified cross-validation. The only hyperparameter specified for data selection is *random_state* which means the seedused by the random number generator is fixed. DT example is shown in Figure 10. This setting brings the same result every time running the same code. The accuracy is one of the common metrics to compare the performance, but the proportion of*churn_status* (0 and 1) are imbalanced because there are more churners. Therefore, another performance metric ROC AUC is selected because it is not affected by the imbalanced distribution (Jeni, Cohn & De La Torre, 2013).

The 6 data patterns of the dataset and split ratio are sent to each function, and the accuracy and AUC are compared as summarized in Table 21. The bold column shows the highest performance of the same dataset. Four out of six highest AUC are from the same pattern using principal components and the data split ratio of 80% training, 10% test, and 10% validation. Thus, this data pattern (ID 11 and 12 in Table 20) is used for the modelling and hyperparameter optimization for all three machine learning algorithms. The 10% testdataset is put aside until the last modelling evaluation.

Table 20. Split Variables and its Details

ID	**Variable Name Examples**	**Category**	**Dataset Used for Splitting**	**Splitting Percentage**
1	x_train80 for general, x_train90 for cross-validation	Independent Variables	Original	80% Training, 10% Test, 10% Validation
2	y_train10	Target Variable	Original	
3	x_train70 for general, x_train85 for cross-validation	Independent Variables	Original	70% Training,

4	y_train15	Target Variable	Original	15% Test, 15% Validation
5	x_train60 for general, x_train80 for cross-validation	Independent Variables	Original	60% Training,
6	y_train20	Target Variable	Original	20% Test, 20% Validation
7	x_train80_pc for general, x_train90_pc for cross-validation	Independent Variables	Principal Components	80% Training, 10% Test,
8	y_train10_pc	Target Variable	Principal Components	10% Validation
9	x_train70_pc for general, x_train85_pc for cross-validation	Independent Variables	Principal Components	70% Training, 15% Test,
10	y_train15_pc	Target Variable	Principal Components	15% Validation
11	x_train60_pc for general, x_train80_pc for cross-validation	Independent Variables	Principal Components	60% Training, 20% Test,
12	y_train20_pc	Target Variable	Principal Components	20% Validation

```
def DT_cross_val(x_train, y_train, tree_dpt, criteria, split):
    if tree_dpt == "": # no hyperparameter setting. Use default.
        Dt = DecisionTreeClassifier(random_state=1).fit(x_train, y_train)
# train the decision tree classifier
        accuracy = cross_val_score(dt, x_train, y_train, cv=10,
scoring='accuracy')
        roc_auc = cross_val_score(dt, x_train, y_train, cv=10,
scoring='roc_auc')
        print("Accuracy of DT: ", accuracy.mean())
        print("ROC AUC of DT: ", roc_auc.mean())
```
Figure 10. Default Setting in the *DT_cross_val* Function

Table 21. Performance Results for 12 Data Pattern

ML Algorithm	Dataset	Performance Metric	60-20-20%	70-15-15%	80-10-10%
Decision Tree	Original	Accuracy	0.7164	**0.7287**	0.7129
Decision Tree	Original	AUC	0.7106	**0.7213**	0.7081
Decision Tree	Principal Components	Accuracy	0.8370	0.8439	**0.8529**
Decision Tree	Principal Components	AUC	0.8345	0.8410	**0.8492**
Logistic Regression	Original	Accuracy	0.4455	0.6059	**0.6114**
Logistic Regression	Original	AUC	0.7150	0.7207	**0.7222**
Logistic Regression	Principal Components	Accuracy	**0.7309**	0.7207	0.7263
Logistic Regression	Principal Components	AUC	**0.7832**	0.7708	0.7808
Random Forest	Original	Accuracy	0.7730	0.7714	**0.8869**
Random Forest	Original	AUC	0.8617	0.8571	**0.9605**
Random Forest	Principal Components	Accuracy	**0.8848**	0.8809	0.8869
Random Forest	Principal Components	AUC	0.9582	0.9584	**0.9605**

4.4 Summary of Implementation

In chapter 4, the implementation flow and detail in the three sections. The data understanding and collection section are collected from the database and stored into 7 CSV files. The next is the data preparation section. These collected data are then preprocessedincluding data transformation, aggregation and are merged into one file after the process. Thereare 20 independent variables for each player. The independent variables are classified into threedemographic, engagement, and performance features. Feature selection is conducted on the dataset using PCA, and two types of datasets (the original and PCA) are prepared forthe modelling. The last step is modelling. Each of the created datasets is split into three datasets: training, validation, and test by using three types of ratios. The three types of ratio are 80% - 10% - 10%, 70% - 15% - 15%, and 60% - 20% - 20%. These dataset and split ratio patterns are sent to the three machine learning algorithms of DT, RF, and LR. For the performance metrics, accuracy and AUC are used and compared. The selected combination is using principal components and 80% - 10% - 10% split ratio.

SECTION 5: RESULTS AND ANALYSES

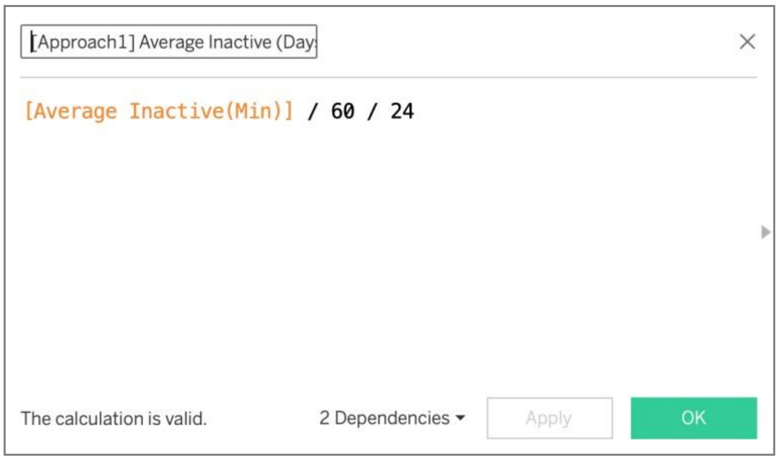
In this chapter, results and analyses are conducted. There are four subsections. The first subsection is churn determination to define a user is a churner or not and identify the churn rate for the case study product to see the validity. The second one is descriptive and retention analysis based on the defined churn. These are specific for the case study services, and it helps the company create new marketing strategies and improve curriculums to increase the retention rate. The third one is hyperparameter optimization because the model created in the previous chapter uses the default setting, and it is possible to have better performance by tuning. The last subsection is about churn prediction and its performance results. There are threemachine learning models: DT, RF, and LR. The results of these models with the besthyperparameters are compared.

5.1 Churn Determination

Churn determination is done by checking a scatter plot of total playtime and average inactive time. If the first plan does not indicate the distinctive churn spot, then the second plan calculates an arbitrary time frame for each user to define churn. The "6_aggregated_play.csv" is imported to Tableau to visualize. Two variables *total_playtime(min)* and *average_inactive(min)* are used. To convert the unit of *average_inactive(min)* from minutes to days, a calculated field named "*[Approach1] Average Inactive (Days)*" was created (Figure 11). The *total_playtime(min)* is set to Rows (axis x), and the created field is set to columns (axis y). Both variables are set to "Dimension" from "Attribute" as indicated in Figure 12. The generated scatter plot is illustrated in Figure 13.The plot is less dense around 100 days, or the more obvious spot is between 150 and 180. However,this is not a convincing cutoff value due to the volume of churners. Even if the cutoff is set at 100days, the number of churned players is 28, covering only 0.75% of the population. And there is no strong reason to set a cutoff on other days.

Next, another calculation of the average inactive has been applied to see the difference. The averageinactive in a day "*[Approach2] Average Inactive(Hours)*" is calculated as shown in Figure 14.Figure 15 displays the scatter plot based on the average. The second plot is harder to find less dense areas after the higher dense hours.

The potential reason for the issue is that digital game-based learning has a long inactive period. The spread-out plot makes it less recognizable than the online gaming industry data. These scatter plot approaches can be concluded that may not be suitable for digital game-basedlearning.



[Approach1] Average Inactive (Day

```
[Average Inactive(Min)] / 60 / 24
```

The calculation is valid.

Figure 11. Screenshot of Calculated Field *[Approach1] Average Inactive (Days)*

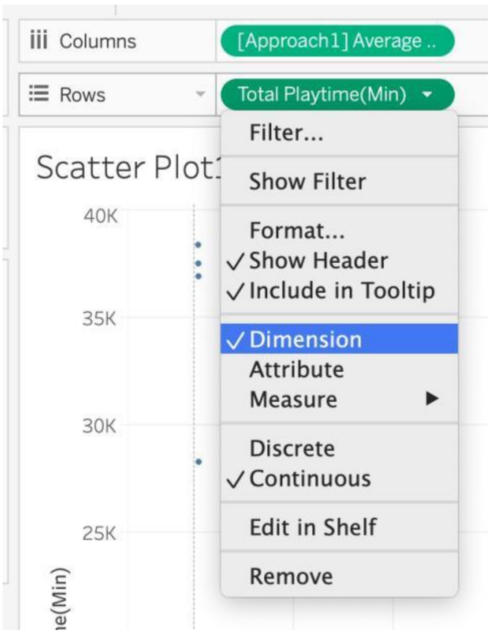


Figure 12. Setting Change to Dimension

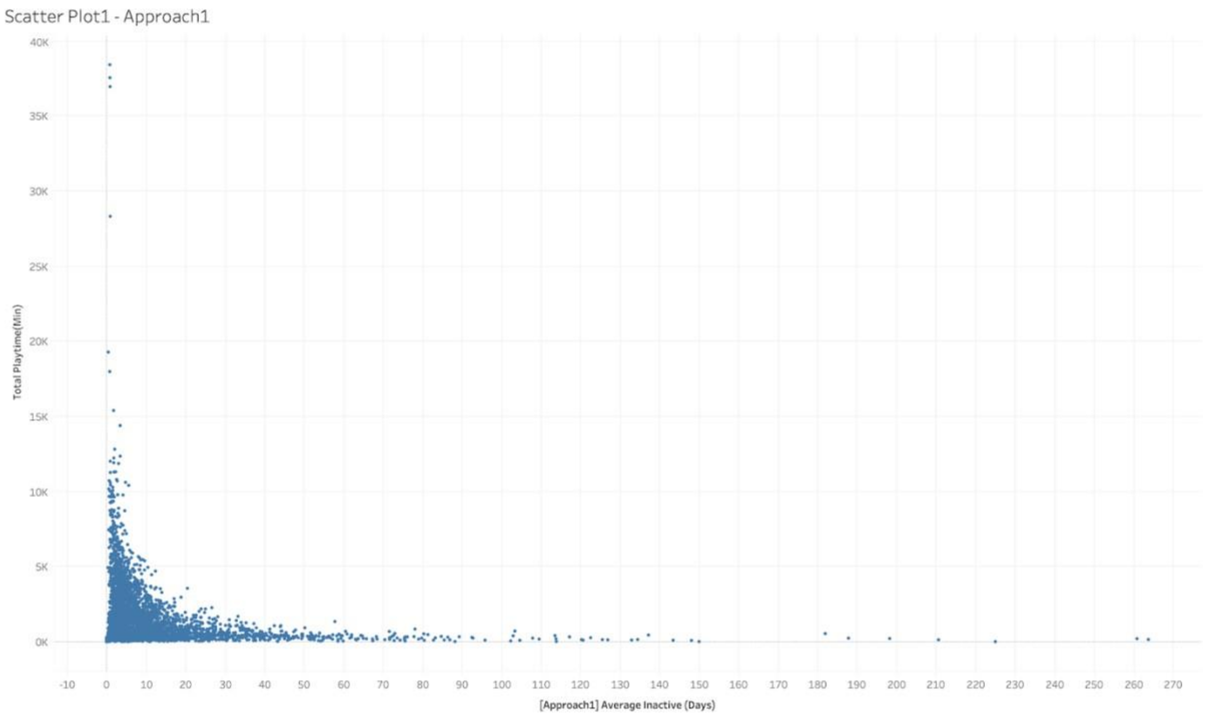
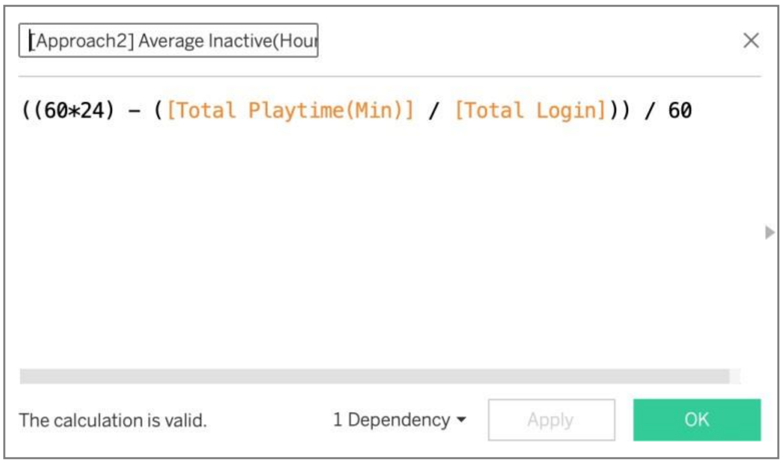


Figure 13. Scatter plot 1 with Average Inactive (Days) and Total Playtime



```
((60*24) - ([Total Playtime(Min)] / [Total Login])) / 60
```

The calculation is valid.

Figure 14. Screenshot of Calculated Field *[Approach2] Average Inactive (Hours)*

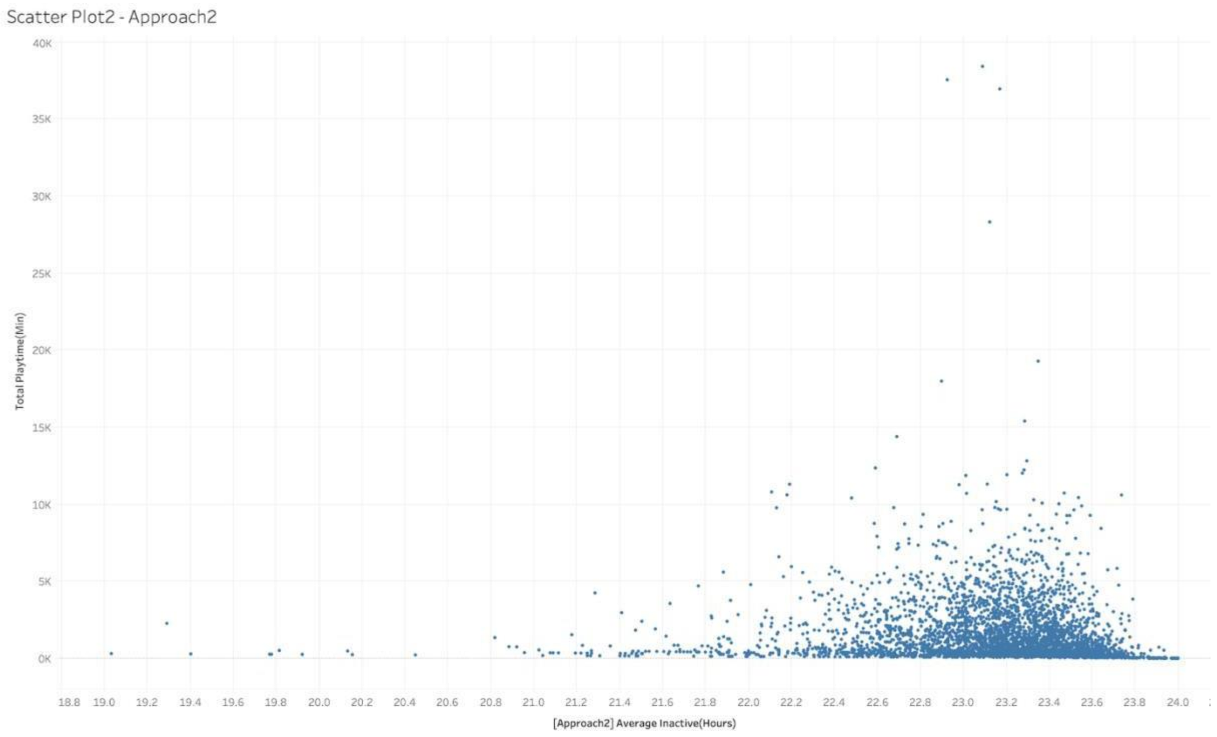


Figure 15. Scatter plot 2 with Average Inactive in a Day (Hours) and Total Playtime

Therefore, the second approach of calculation of arbitrary time frame is conducted. The calculation part is shown in Figure 16, which is a part of the *create_player_aggregated* function. To compute the time frame, three values are required. Recency (*recency*), average inactive time between logins (*avr_inactive*), and standard deviation of inactive (*std_inactive*). Because average inactive had already been calculated in the function *create_player_aggregated*, computation of recency and standard deviation should be added. The *recency* is the time between the last login and the current day. Because the date of the dataset given is January 27th 2020, the current day is set as January 28th 2020. With the current date, the recency is calculated.For the standard deviation for each user, a list named *inactive_list* and Python library called *statistics* are used. The *inactive_list* contains inactive time in seconds between logins for the user for the calculation. The *personal_cutoff* has been determined from the addition of *avr_inactive* and double of *std_inactive*. If the *recency* is bigger than the *personal_cutoff*, the user is defined as churned, and *churn_status* is set to True.

The defined churners are plotted to see the percentage of churners (as plotted in Figure 17). 56.77% of the population are churners based on the value True in *churn_status*. The calculationand the proportion is acceptable compared to the first approach. Hence, it is concluded that theretention rate is 43.23% for the product.

```
if log_idx < (len_log - 1): # if it’s not the last object in a
    listlast = pl_log_df.at[log_idx, "timestamp"]
    avr_play = round((t_play_sec / 60 / t_login), 2)
    current = datetime.strptime("2020-01-28 0:00:00", fmt) # the next
dateof the data collected
    recency = (current - datetime.strptime(last, fmt)).total_seconds()

    if (pl_log_df.at[log_idx, "id"] != pl_log_df.at[log_idx + 1,
        "id"]):log_idx = log_idx + 1

    recency = round((recency / 60), 2)
    try:  # calcurate standard deviation of inactive time
        std_inactive = round(statistics.stdev(inactive_list) /
        60, 2)
    except
        statistics.StatisticsErr
        or:std_inactive = 0

    if ch_idx != []:  # the id is in ch_agg_df
        # ch7 is not finished and total login is not one time,
add theinactive period till now
        if ch_agg_df.at[ch_idx[0], "ch7_fin"] == "0" and t_login
            != 1:t_inactive = (t_inactive + recency)
            avr_inactive = round((t_inactive / 60 / t_login),
                                 2)  # add the time between now and the
last play as inactive
            personal_cutoff = avr_inactive + (2 * std_inactive)
            if recency > personal_cutoff: # determine if the player
ischurned or not
                churn = True
        else:  # finished playing, so not churned
            avr_inactive = round((t_inactive / 60 /
            t_login),
                                 2)  # add the time between now and the
last play as inactive
            churn = False
```

Figure 16. Python Code: Calculation of Churn Determination

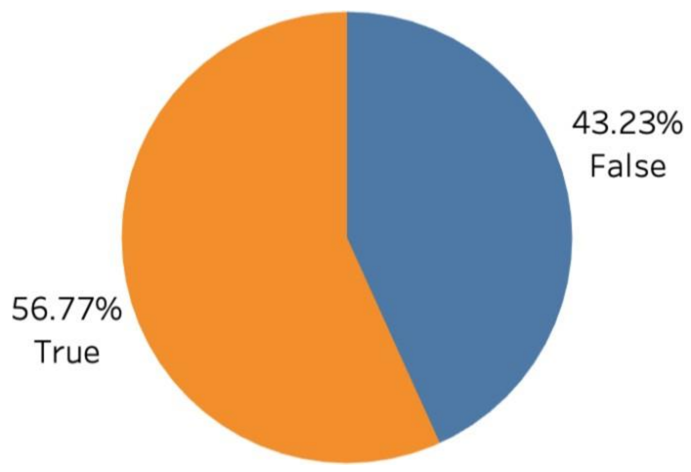


Figure 17. The Proportion of Churners (True) and Non-Churners (False)

5.2 Retention Analysis

In this subsection, descriptive analysis for overview and retention analysis by chapter was conducted with the case study company data. The descriptive analysis is to grasp the overviewof user activities by using the company's data. The dataset used for the analysis is "1_combined_player_log.csv" which contains the combined user activity log and lesson log. For the retention analysis, "8_aggregated_after_preprocess_ret_analysis.csv" was used which was generated to have cleaned aggregated data without label encoding in the function named *preprocess_retention*. Tableau was used for generating new variables and plotting the graphs for both analyses.

5.2.1 Descriptive Analysis of the User Logs

To comprehend the current situation of the case study service, monthly active users (MAU) andpopular hours and days of the week are plotted. The MAU is displayed in Figure 18. The MAU had generally increased during the first year. Then it settled around 1,000. Since the number of users has increased since the service launch, the MAU settlement should be concerned. Next, the popular hour and days of the week are illustrated in Figure 19. The expected popular hour was after office hours until midnight as many web services, and the day of the week is theweekend.

Nevertheless, between 1 PM to 9 PM has bigger access than other hours, and access drops after 8 PM. In addition, there is no difference between the weekdays and weekends unexpectedly. The assumption for the popular hour and day of theweek is that more students than adult workers are. They can access the service earlier after school, and bedtime is also earlier, so the access starts decreasing after 8 PM. Considering that the mean age is 29.6 and the median is 26 (as shown in Table 19-2 in Appendix C), a certain part of the registered age could be the player's parent's age.

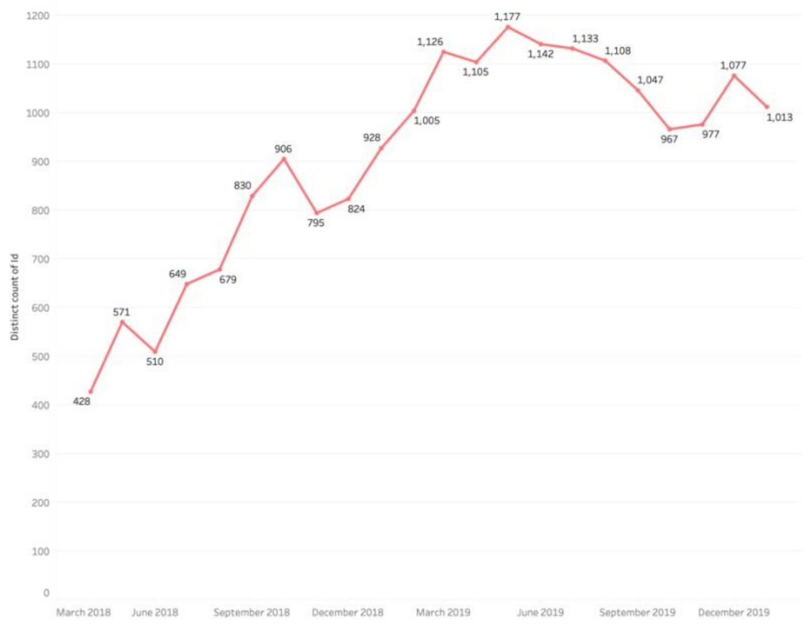


Figure 18. Monthly Active Users (MAU)

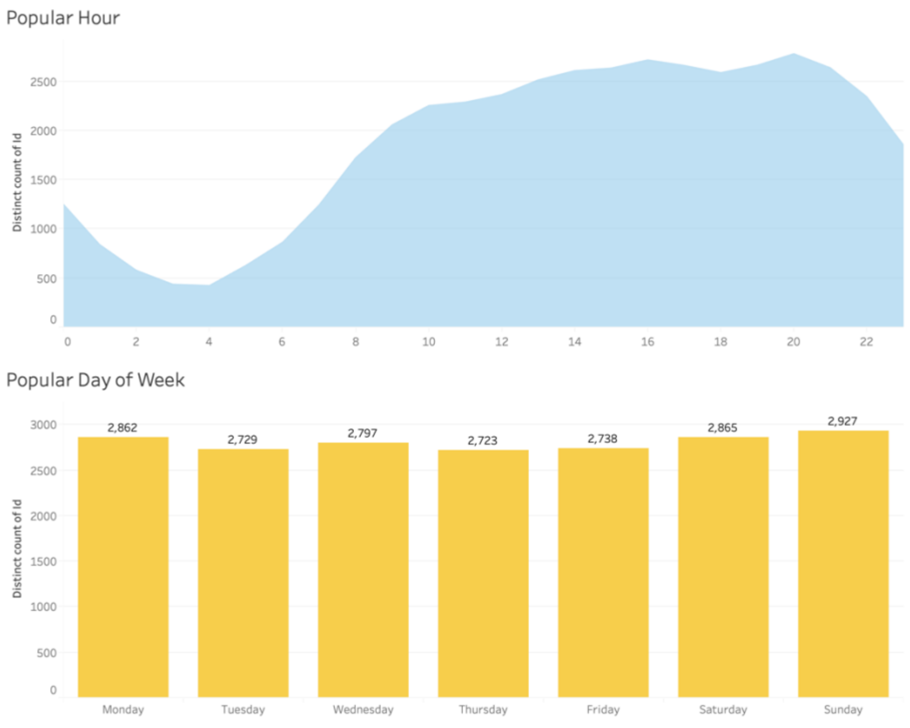


Figure 19. Popular Hour and Day of the Week

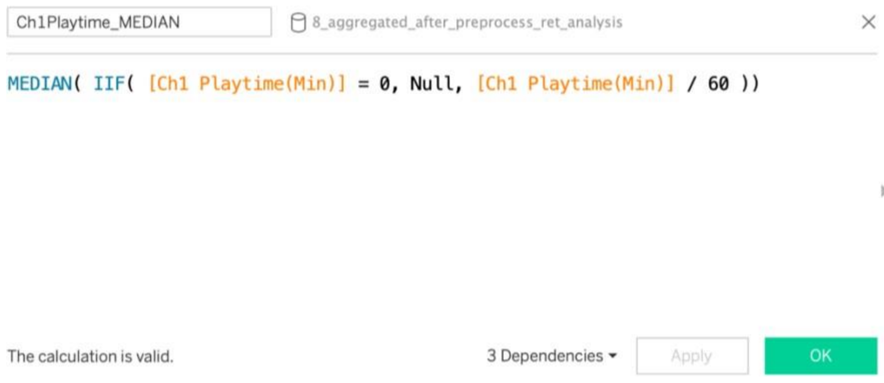
5.2.2 Retention Analysis by Chapter Using the Aggerated Data

After understanding the overview of user behaviour, the chapter's retention analysis was conducted using the newly defined churn status and aggregated data. First, how long the users spend each chapter on average is checked. The calculated fields for each chapter are created to eliminate 0 from the playtime variables and convert from minutes to hours. Figure 20 shows an example in chapter 1. The median of playtime (hour) per chapter is plottedto grasp the general playtime in Figure 21. The reason for using the median is to reduce the impact of the outliers who have an extremely long playtime. The shortest playtimeisspent in chapter 1, and the longest is chapter 2 in general. To see the difference in churn status,the bar chart is created for the median playtime (hour) per chapter by the churn status (Figure 22). Considering that 56.77% are churners and 43.23% are non-churners based on the churn determination, there is a recognizable difference between churners and non-churners. Overall the churners play less than non-churners, but the playtime gap depends on the chapters. Table 22 shows the median playtime (hour) difference for each chapter. Next, to see the completion rate of each chapter, two calculated fields are created for each chapter. Examples of chapter 2 are displayed in Figures 23 and 24. Figure 23 counts the number of finished usersof chapter 2, and Figure 24 calculates the completion percentage of the current chapter from the previous chapter completion. These percentages are then plotted in Figure 25. The completion percentage for non-churners are generally high, but not 100%. This means that a certain percentage of users are currently playing the chapter. This percentage can be used as a benchmark for churners. The difference in completion rate is summarized in Table 23. Bycomparing the results in Tables 22 and 23, the five interesting points are discovered.

- Chapter 1 does not have a big difference between churners and non-churners. So this could be possibly the churners have almost played the entire contents of chapter 1, and stopped coming back.
- Chapter 2 has the longest playtime for both churners and non-churners. However, the playtime difference between them is almost 2 hours. Also, the completion rate decreasedfrom 75.22% of chapter 1 to 50.35% of chapter 2. This means many users had droppedin chapter 2. It may improve by shortening or simplifying the lesson content.
- Chapter 3 has only one hour difference between churners and non-churners, but

the gapof completion rates is huge, which is 17.61%. This implies the churners possibly droppedout later in the chapter lesson. The later contents may need to be examined the necessity of update.

- Chapter 4 has the second-highest playtime for churners and non-churners. And the difference of playtime is larger. The gap indicates that the churners had stopped returning at the earlier lessons in chapter 4. The churners may have lost interest by longer playtime. Same as chapter 2, shortening or simplifying lesson content may boost retention.
- Chapter 5 and 6 have a bigger difference of completion rate and chapter 6 and 7 has bigger playtime difference between churners and non-churners. This may be caused bythe smaller number of records that the latest data can enhance.



Ch1Playtime_MEDIAN

8_aggregated_after_preprocess_ret_analysis

```
MEDIAN( IIF( [Ch1 Playtime(Min)] = 0, Null, [Ch1 Playtime(Min)] / 60 ))
```

The calculation is valid.

Figure 20. Tableau Setting

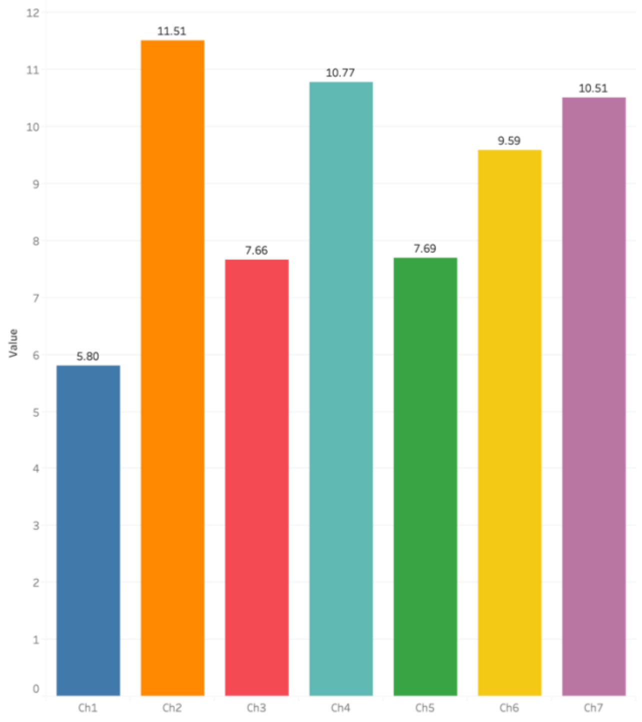


Figure 21. Median Playtime per Chapter (hour)

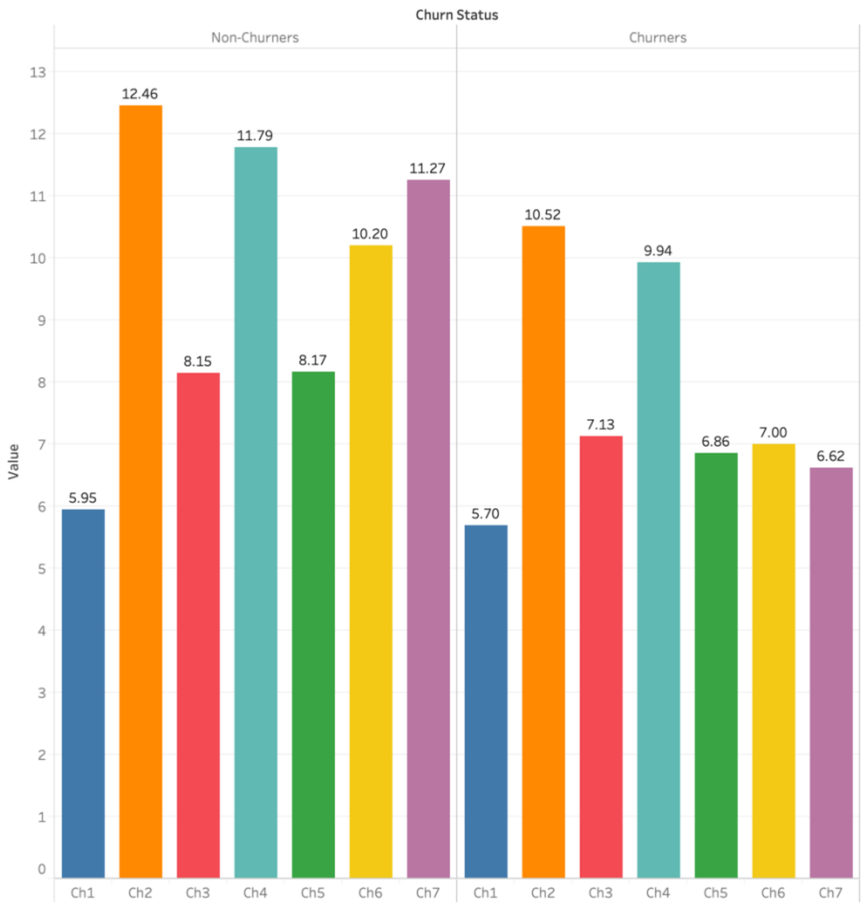
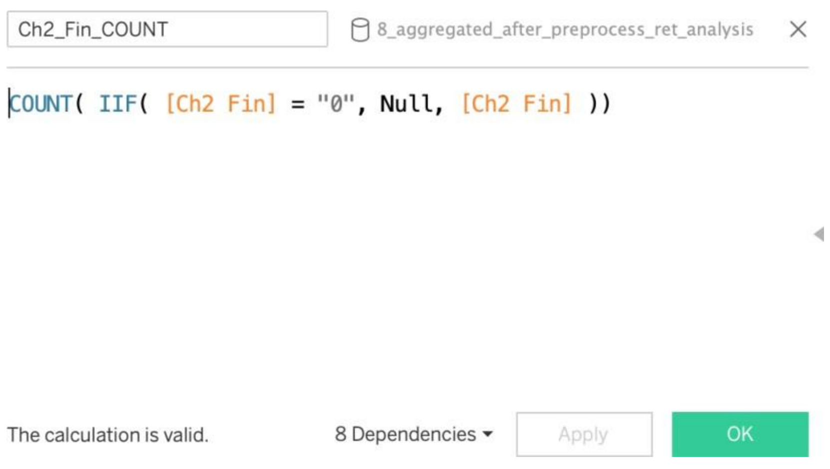


Figure 22. Median Playtime per Chapter by Churn Status (hour)

Table 22. Median Playtime (hour) Difference by Churn Status

	Ch1	**Ch2**	**Ch3**	**Ch4**	**Ch5**	**Ch6**	**Ch7**
Non-Churners	5.96	12.46	8.15	11.79	8.17	10.20	11.27
Churners	5.70	10.52	7.13	9.94	6.86	7.00	6.62
Difference	0.26	1.94	1.02	1.85	1.31	3.2	4.65

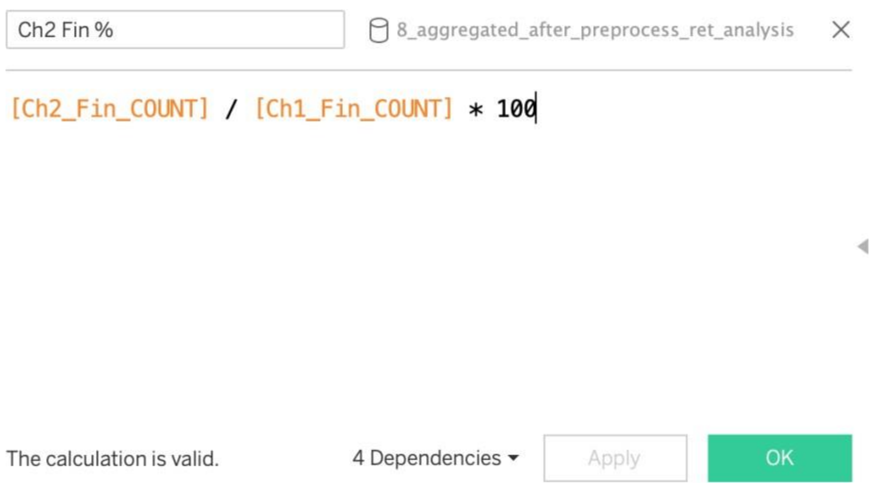

Ch2_Fin_COUNT

8_aggregated_after_preprocess_ret_analysis

```
COUNT( IIF( [Ch2 Fin] = "0", Null, [Ch2 Fin] ))
```

The calculation is valid.

Figure 23. Calculated field of Count Finished Users


Ch2 Fin %

8_aggregated_after_preprocess_ret_analysis

```
[Ch2_Fin_COUNT] / [Ch1_Fin_COUNT] * 100
```

The calculation is valid.

Figure 24. Calculated field of Completion Rate

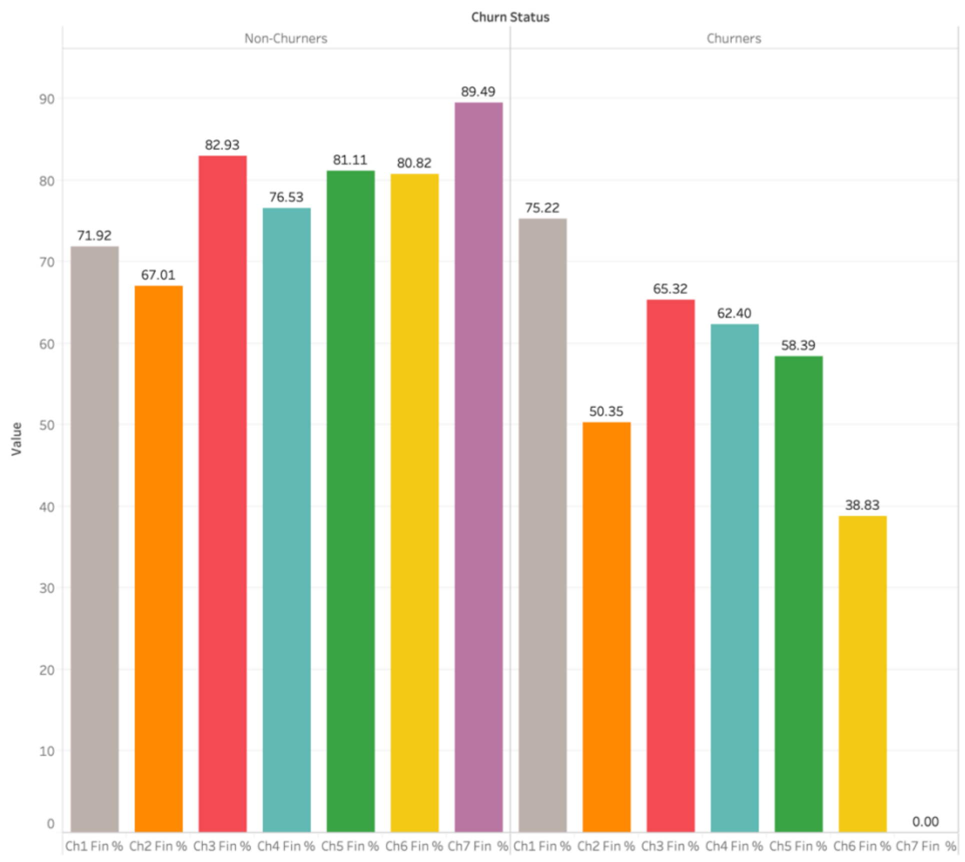


Figure 25. Chapter Completion Rate by Churn Status (%)

Table 23. Chapter Completion Rate Difference by Churn Status

	Ch1	**Ch2**	**Ch3**	**Ch4**	**Ch5**	**Ch6**	**Ch7**
Non-Churners	71.92	67.01	82.93	76.53	81.11	80.82	89.49
Churners	75.22	50.35	65.32	62.40	58.39	38.83	0
Difference	-3.3	16.66	17.61	14.13	22.72	41.99	N/A

5.3 Hyperparameter Optimization

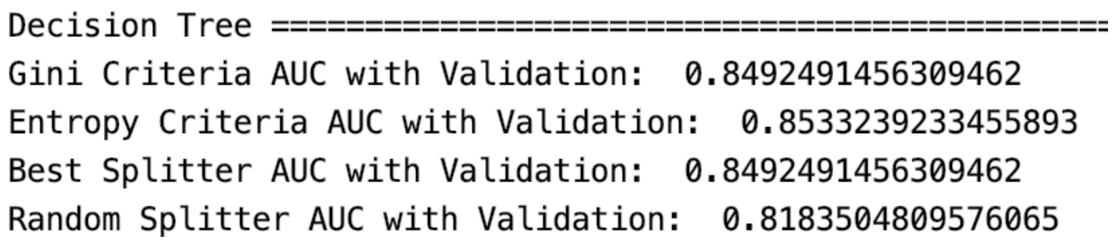
Hyperparameters are the parameters set before learning, and tuning hyperparameters affects the performance outcome of the model. Default settings are used for the previous data selection, and it is important to tune hyperparameters for each model to create a better performance model possibly. For the categorical options of the hyperparameters, the *scikit-learn cross_val_score* function is used to compare AUC with 10-folds stratified cross-validation. For the numeric hyperparameter options, there are two steps. First, plot the mean AUC performance by 10-fold stratified cross-validation with various values and make a general estimation. Second, to discover the best hyperparameter values, check more specific values by using cross_val_score same as the categorical values.

5.3.1 Decision Tree

To optimise the decision tree, five hyperparameters are modified to see the performance difference. The five hyperparameters are *criterion*, *splitter*, *max_depth*, *min_samples_split*, and *min_samples_leaf*. The *criterion* is a function to measure the quality of a split, and there are two criteria: "gini" and "entropy". The default is set to "gini" for criterion. Therefore, both "gini" and "entropy" are run for comparison. The *splitter* is the strategy used to select the split at each node and have two "best" and "random" choices. The "best" is set as the default hyperparameter, and these two choices are set to see thedifference. The *max_depth* is the maximum depth of the tree, and the default is set to "None".For *max_depth*, from 1 to 32 trees are set and see the difference. A part of the code gaining AUC is shown in Figure 26, almost the same as other numerical hyperparameters.

The *min_samples_split* is the minimum number of samples required to split. The default is 2. The 0.01 to 0.5 are set for the comparison, using 1% of data to 50% of data. Thirty evenly spaced values between 0.01 and 0.5 are created and evaluated. The *min_samples_leaf*is the minimum number of samples required to be a leaf node. The default is 1, and 30 valuesbetween 0.01 and 0.5 are set as *min_samples_leaf*. 0.01 and 0.5 mean the same as the *min_samples_split*. These hyperparameter evaluations and outputting the text and graphical plots are conducted in the function named the *DT_find_best_param*. The results of the criteria and splitter are revealed in Figure 27. The "entropy" criterion has better AUC, and the "best" splitter is betterthan "random". So, the criterion needs to change for "entropy" and the splitter is left as default. The *max_depth* plot of AUC and the tree depth is illustrated in Figure 28. After around 6 tree depth, the model performance decreases, and after 10 tree depth, the performance is about the same. This means that a bigger tree depth causes lower performance even though the computing cost is higher. So, between 4 and 8 are set as *max_depth* and tested again to see theAUC with the *DT_cross_val* function. Because the criterion "entropy" has better accuracy, the criterion hyperparameter is set to "entropy" for this. The outcome is summarized in Table 24. Both accuracy and AUC are the highest at *max_depth* 6, so the best value for *max_depth* is set to 6. As for minimum samples of splits and leaf have a similar pattern (as illustrated in Figures 29 and 30). Generally, the higher the hyperparameter value, the lower AUC is output. This is understandable because more data is used for the minimum samples of the splits or leaf. The split should be rougher. For *minimum_sample_leaf*,the best performance is the smallest value, so the default should be used. Nevertheless, the minimum samples of split-plot is not convincing because the AUC reaches a peak of around 5% (around 0.05 in Figure 29). The 5% of data is 142 by considering the number of observations of the training dataset for cross-validation is 2,846. Thus, from 25 to 200 increases by 25 are tested with cross-validation. The result of accuracy and AUC are shown in Table 25. Accuracy and AUC are the highest at *min_samples_split* 50, so DT's best value is set to 50. The summary of hyperparameters and the final best performance values are summarizedin Table 26. The best value for *max_depth* is 6, and *min_samples_split* is 2. Other hyperparameters are preferred as the default setting.

```
# Find the best max_depths ----------------------------
cross_val_results = []
max_depths = np.linspace(1, 32, 32, endpoint=True) # create evenly spaced
32 values between 1 to 32 trees
for max_depth in max_depths:
    dt = DecisionTreeClassifier(random_state=1,
max_depth=max_depth).fit(x_train,y_train)  # train DT dclassifier
    roc_auc = cross_val_score(dt, x_train, y_train,cv=10,scoring='roc_auc')
    cross_val_results.append(roc_auc.mean())
```
Figure 26. DT - AUC calculation for max_depth


```
Decision Tree ============================================
Gini Criteria AUC with Validation:  0.8492491456309462
Entropy Criteria AUC with Validation:  0.8533239233455893
Best Splitter AUC with Validation:  0.8492491456309462
Random Splitter AUC with Validation:  0.8183504809576065
```
Figure 27. DT - Criteria and Splitter Results

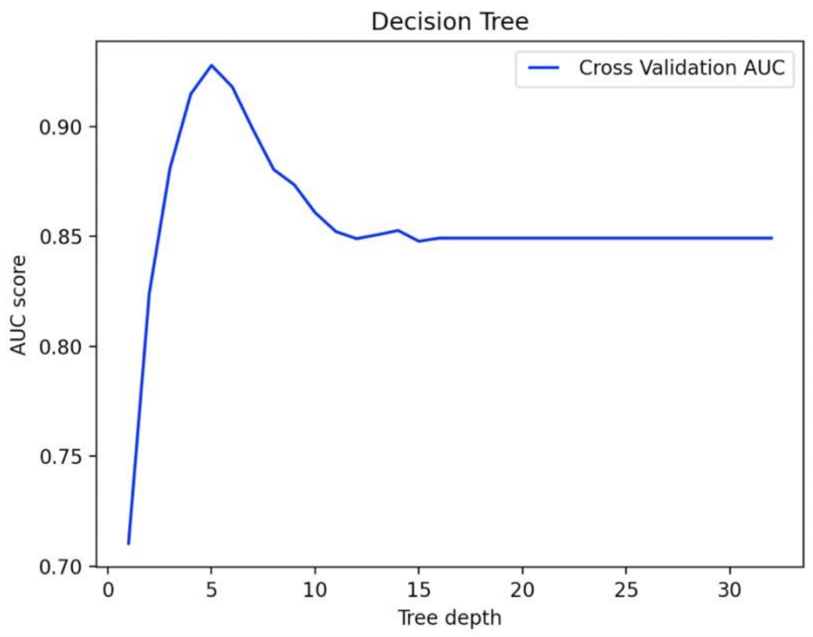


Figure 28. DT - AUC Plot of Tree Depth

Table 24. DT - Maximum Depth and the Performance Difference

max_depth	**4**	**5**	**6**	**7**	**8**
Accuracy	0.8376	0.8472	**0.8616**	0.8613	0.8610
AUC	0.9182	0.9286	**0.9290**	0.9286	0.9289

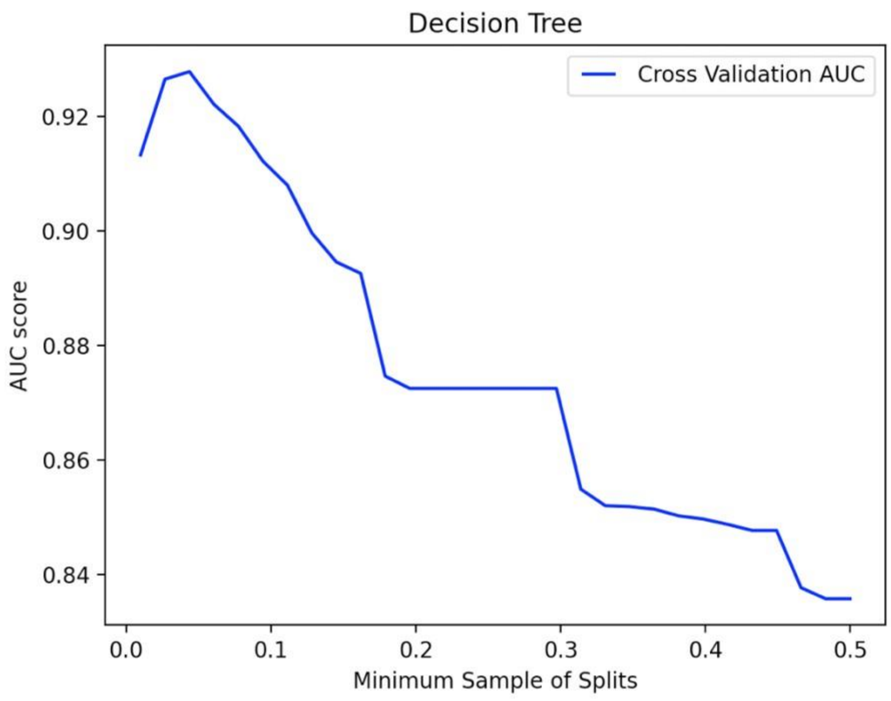


Figure 29. DT - AUC Plot of Minimum Samples of Splits

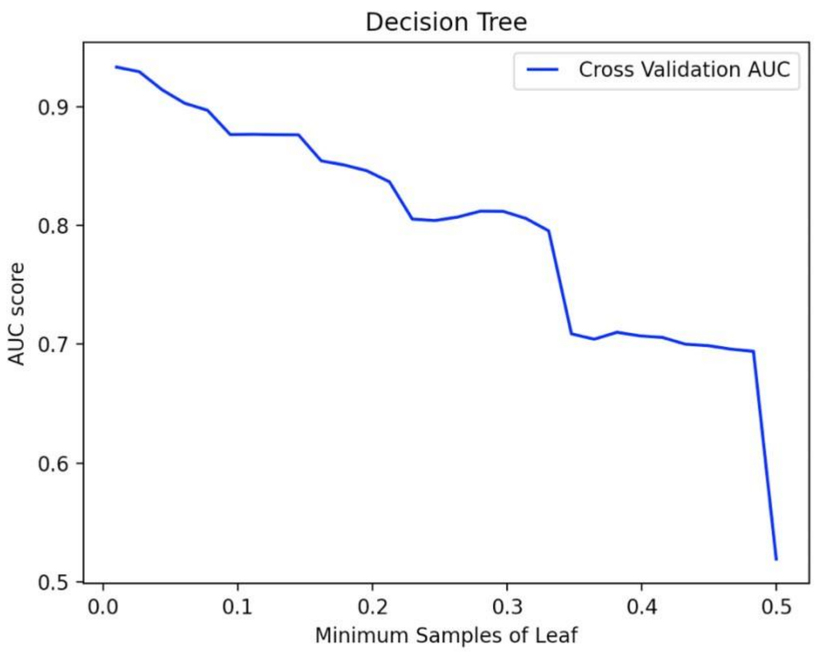


Figure 30. DT - AUC Plot of Minimum Samples of Leaf

Table 25. DT - Minimum Samples of Split

min_split	25	50	75	100	125	150	175	200
Accuracy	0.8626	**0.8632**	0.8588	0.8616	0.8616	0.8557	0.8460	0.8454
AUC	0.9317	**0.9333**	0.9320	0.9329	0.9290	0.9284	0.9258	0.9262

Table 26. DT - Summary of The Hyperparameters and the Best Values

Hyperparameter name	Default	Options	Best Value
criterion	gini	gini, entropy	entropy
splitter	best	best, random	best
max_depth	None	integer values	6
min_samples_split	2	integer or float values	50
min_samples_leaf	1	integer values	1

5.3.2 Random Forest

Next, tuning the random forest model is similar to the decision tree because there are many same parameters. The *max_depth*, *min_samples_split*, *min_samples_leaf* are the same ones, and the different hyperparameter is *n_estimators*. The *n_estimators* is the number of trees in the forest and the default is 100 trees for n_estimators. The other hyperparameters have the same default values as the decision tree: "None” for *max_depth*, 2 for *min_sample_split*, and 1 for *min_sample_leaf*. The function named *RF_find_best_param* handles applying the various values for each hyperparameter and outputs the results. As presented in Figure 31, the number of *n_estimater* is set to 1, 2, 4, 8, 16, 32, 100, 150 to see the difference. The result of AUC with different *n_estimators* is illustrated in Figure 32. The AUC reaches closest to 1 about 100 *n_estimators*. To see the precise value of AUC with cross-validation, the values 25, 50, 75, 100, and 120 are tested again for *n_estimators*. The result is shown in Table 27. The AUC is the highest at the number of *n_estimator* 100, and it does not improve after 100 according to the plot. Having more trees requires more computingcost. Therefore, the best value for *n_estimator* is 100, the same as the default. Next, the performance of the *max_depth* is illustrated in Figure 33. The AUC reaches the maximum of around 10. So the *max_depth* value is set from 7 until 13 and the *n_estimators* are set to 100. The outcome is in Table 28. The *max_depth* at 10, the AUC is the highest which is acceptable by considering the earlier AUC plot. So, 10 is the best value for *max_depth*. The other two hyperparameters *min_samples_split* and *min_samples_leaf* have a similar pattern as the decision tree model. The results are displayed in Figures 34 and 35. Both *min_samples_split* and *min_samples_leaf* have the same pattern in which the AUC decreases as the value increases. Thus, the best value is the smallest value, the same as the default. The hyperparameters and the final setting are summarized in Table 29. The *max_depth* need to be specified to 10, but default is the best for other hyperparameters.

```
# Find the best n_estimators ----------------------------
n_estimators = [1, 2, 4, 8, 16, 32, 64, 100, 150]
cross_val_results = []
for estimator in n_estimators:
    rf = RandomForestClassifier(random_state=1,
n_estimators=estimator).fit(x_train, y_train) # train the random forest
classifier
    roc_auc = cross_val_score(rf, x_train, y_train,cv=10,scoring='roc_auc')
    cross_val_results.append(roc_auc.mean())
```
Figure 31. RF - AUC Calculation for n_estimators

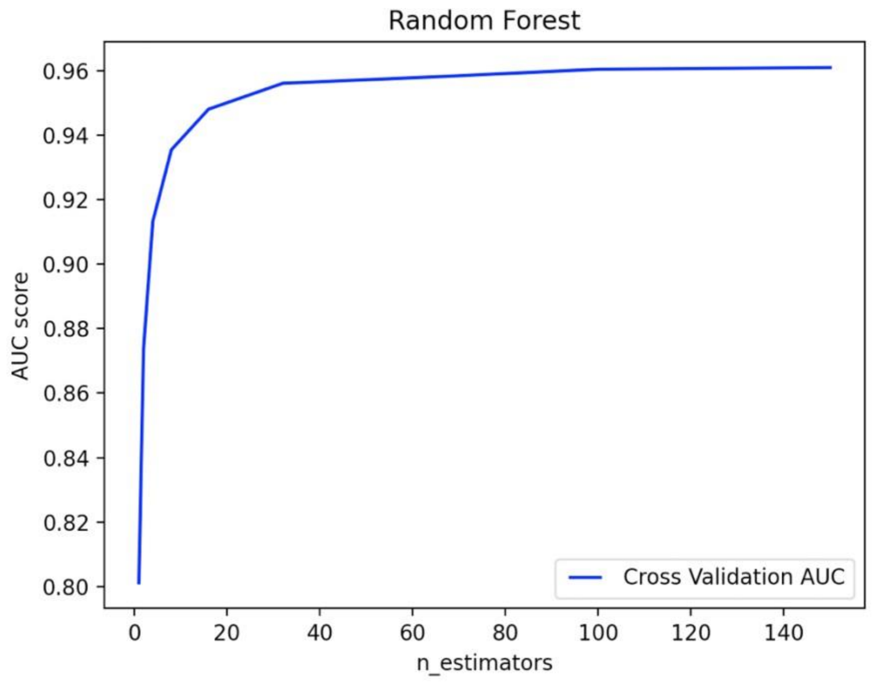


Figure 32. RF - AUC Plot of n_estimators

Table 27. RF - Performance Difference with Different n_estimators

n_estimator	**25**	**50**	**75**	**100**	**125**
accuracy	0.8869	0.8828	0.8850	0.8869	**0.8878**
AUC	0.9605	0.9570	0.9590	**0.9605**	0.9603

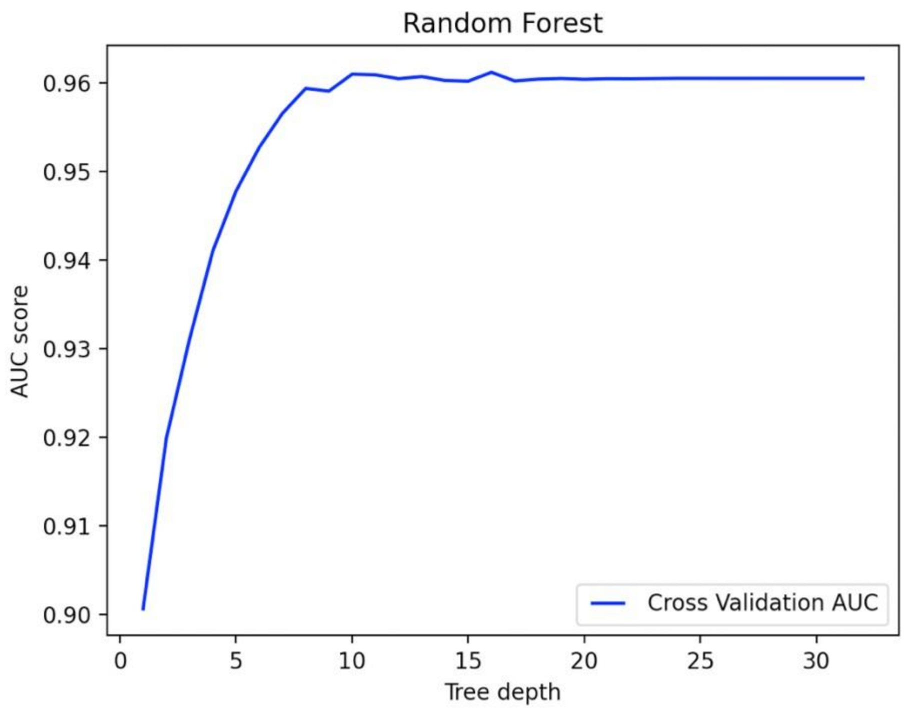


Figure 33. RF - AUC Plot of Tree Depth

Table 28. RF - Performance Difference with Different max_depth

max_depth	**7**	**8**	**9**	**10**	**11**	**12**	**13**
accuracy	0.8779	0.8850	0.8869	**0.8910**	0.8888	0.8844	0.8885
AUC	0.9565	0.9594	0.9591	**0.9610**	0.9609	0.9605	0.9607

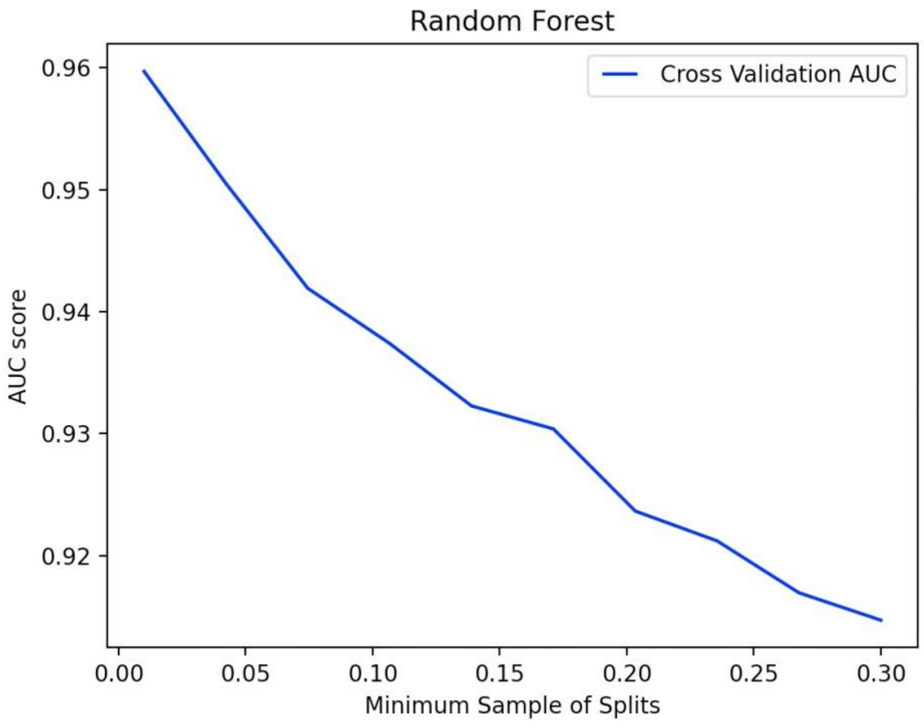


Figure 34. RF - AUC Plot of Minimum Samples of Splits

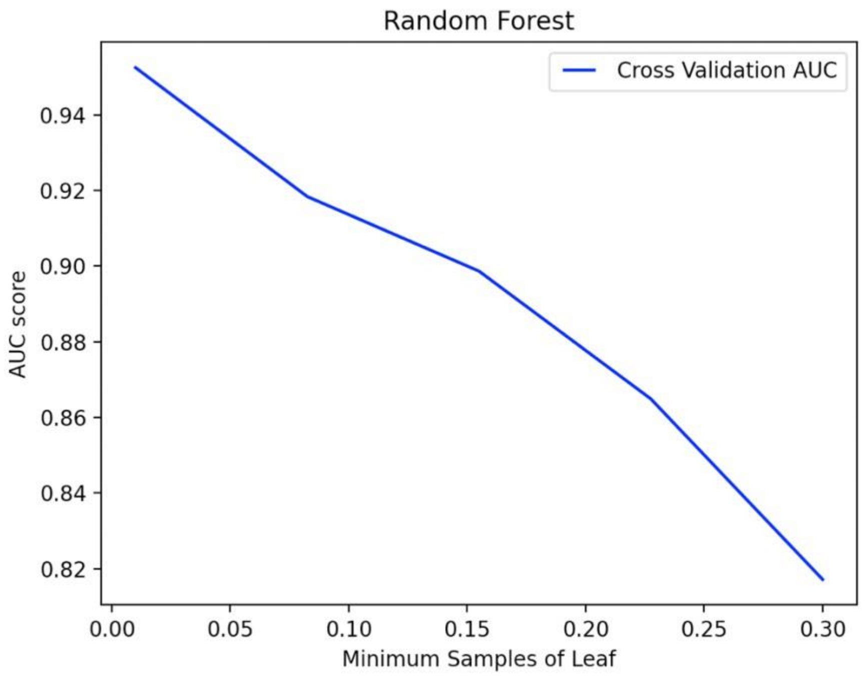


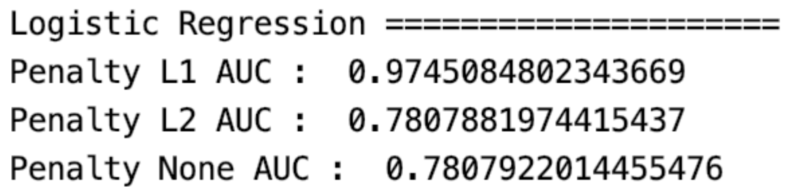
Figure 35. RF - AUC Plot of Minimum Samples of Leaf

Table 29. RF - Summary of The Hyperparameters and the Best Values

Hyperparameter Name	Default	Options	Best value
n_estimator	100	integer values	100
max_depth	None	integer values	10
min_samples_split	2	integer or float values	2
min_samples_leaf	1	integer values	1

5.3.3 Logistic Regression

The logistic regression has two main hyperparameters, which are *penalty* and *C*. The *penalty* isa type of normalization used in penalization, and there are three options. The options are “l1”, “l2”, and “None”, and the default is “l2”. The *C* is an inverse of regularization strength, and the smaller value means stronger regularization. The default is 1.0 for *C*. First, performance with three types of penalty are checked, and the results are displayed in Figure 36. The “l1” has distinctively high AUC, whereas “l2” and None are about 0.19 lower than “l1”. Therefore, “l1” is chosen for the penalty for the dataset. Another hyperparameter *C* is also plotted. Thirty evenly spaced values between 0.01 to 1.5 are set to *C* with default setting andchecked AUC. The resulting plot is illustrated in Figure 37. The AUC reaches the highest around 1.0, and the AUC does not change after that. To make sure what value is the best withprecise value, the *LR_cross_val* function is used to check AUC. The AUC value at 0.8, 0.9, 1.0, 1.1, and 1.2 are evaluated and the results are summarized in Table 30. The AUC is the same for all *C* values and small differences for accuracy. The accuracy with 0.9 is the best, sothe best hyperparameter *C* is chosen as 0.9. The hyperparameters and the final choices are shown in Table 31. The best hyperparametervalues are “l1” for *penalty* and 0.9 for *C*.



```
Logistic Regression ====================
Penalty L1 AUC :  0.9745084802343669
Penalty L2 AUC :  0.7807881974415437
Penalty None AUC :  0.7807922014455476
```

Figure 36. LR - AUC of Different Penalty

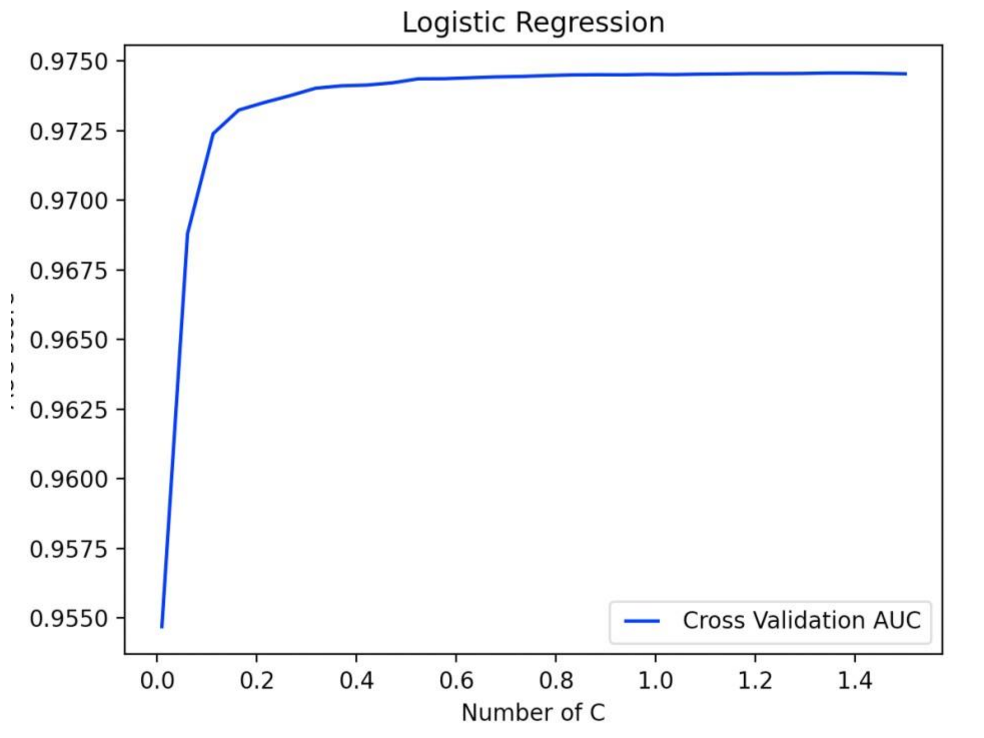


Figure 37. LR - AUC Plot of Hyperparameter C

Table 30. LR - Performance Difference with Different C

C	**0.8**	**0.9**	**1**	**1.1**	**1.2**
Accuracy	0.9206	0.9210	**0.9206**	0.9203	0.9200
AUC	0.9745	0.9745	0.9745	0.9745	0.9745

Table 31. LR - Summary of The Hyperparameters and the Best Values

Hyperparameter name	**Default**	**Options**	**Best Value**
penalty	L2	L1, L2, None	L1
C	1	float value	0.9

5.4 Churn Prediction and Model Evaluation

The final model of each algorithm was created with the best hyperparameters. For the last evaluation, the dataset used for modelling is the same split ratio of 80% training and 10% validation and 10% test. Still, the training and validation sets are combined as a training dataset. Also, the principal components are used. The difference is that the evaluation was conducted in the DT_evaluation, RF_evaluation, and LR_evaluation functions to output other metrics for deeper analysis and validation. These functions output AUC, confusion matrix, and classification report, which outputs accuracy, precision, recall, and F1-score. The reason for checking the other metrics is that the accuracy and ROC AUC does not cover the whole factors behind them. Precision means the ratio of correctly predicted positive cases to the total of predicted positive cases, and recall is the proportion of actual positive observations that are correctly predicted positive observations. Generally, the precision and recall are in tension. So, improvingone metric causes reducing the other. F1-score is a metric to see the balance between precision and recall. According to Power (2011), the ROC analysis ignores recall even though data mining and machine learning consider recall is important. Because the distribution of *churn_status* is not even, these values are inspected to justify the accuracy and AUC. First, the confusion matrix of each model is plotted as illustrated in Figures 38, 39, and 40. TruePositive (TP) correctly predicts an actual churner as a churner, and True Negative (TN) correctly predicts an actual non-churner as a non-churner. The overall prediction results of thethree models can be said that predictions are mostly correct because the number of TP and TNare the more than majority without checking other metrics. The number of TP is greater than TN which is due to the imbalanced target value. This applies to the relationship of False Negative (FN) and False Positive (FP).

Next, by using the classification report, specific values are checked. The reports are shown in Figures 41, 42, and 43. According to the result, the total observation is 356, the churners (1 in the report) are 232, non-churners (0 in the report) are 124. This means that 65.16% of test data are churners higher than the actual percentage of the whole data of 56.7%. Therefore, the results will be more biased. There are “*macro avr*” and “*weighted avr*” in the classificationreport. The “*weighted avr*” considers the imbalanced proportion and calculates the precision, recall, and F1-score with weight. Since the proportion of the target is not balanced, the “*weighted avr*” is chosen to be compared for evaluation.

The summary of the metrics of the three models is presented in Table 32. For precision, recall,and F1-score are taken from “*weighted avr*”. The LR model is the highest accuracy of 0.9185 and AUC of 0.9225, and the F1-score of 0.9194 is also high which means the recall and precision are well balanced. Although DT and RF have good performance of AUC which is 0.8452 and 0.8493, the prediction performance is lower than the LR model. Tree-based algorithm characteristics may cause the similarity of results of DT and RF. Considering the prediction performances of past other studies, the AUC and F1-score are in a higher range. Most studies used AUC for the performance metrics, and the popular range is between the late seventies and early nineties, summarized in Tables 2 and 3. As for theF-1 score, a study conducted churn prediction for freemium games measured F1-score as a performance metrics, and the highest model of F1-score was DT with 0.916 (Hadiji et al., 2014).Thus, the LR model has a great prediction performance even considering the other studies in the gaming and education industries.

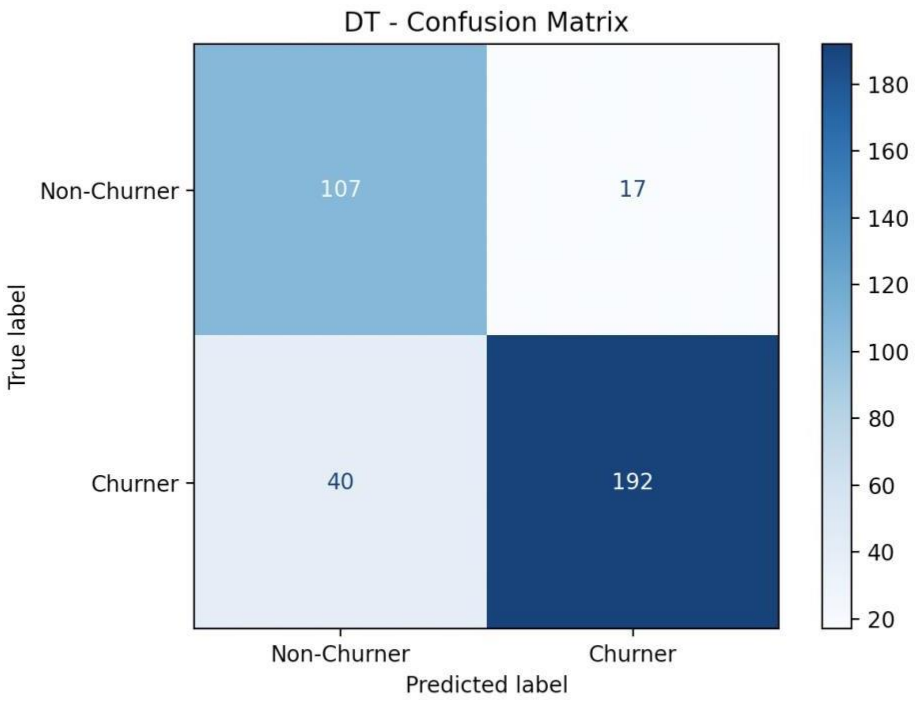


Figure 38. DT - Confusion Matrix

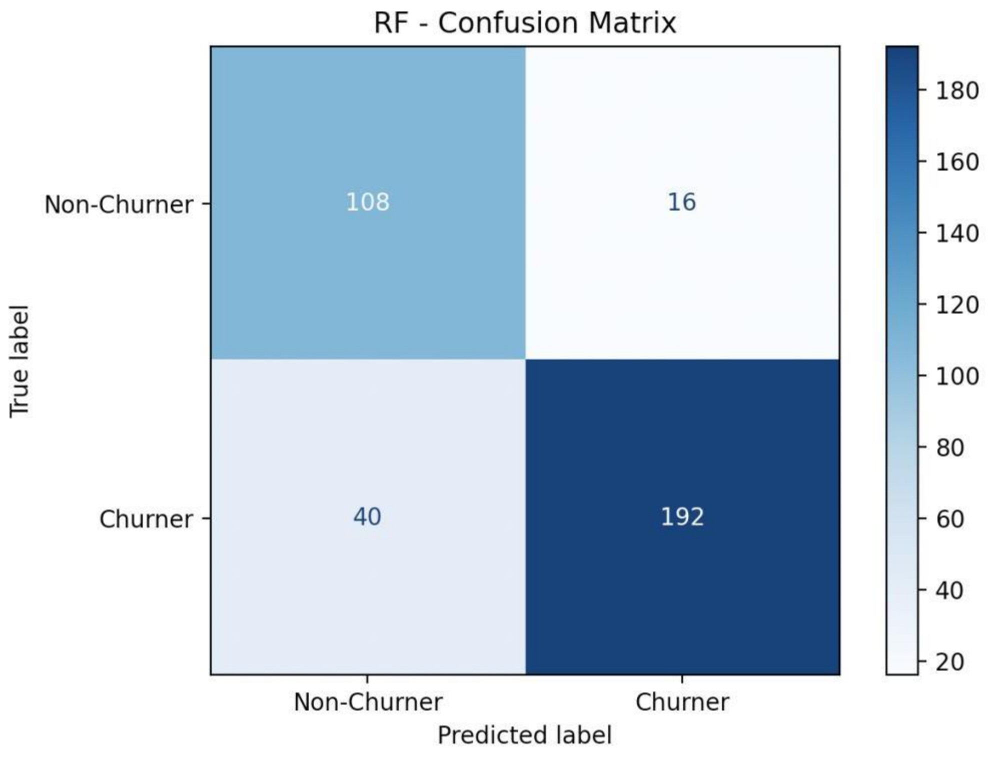


Figure 39. RF - Confusion Matrix

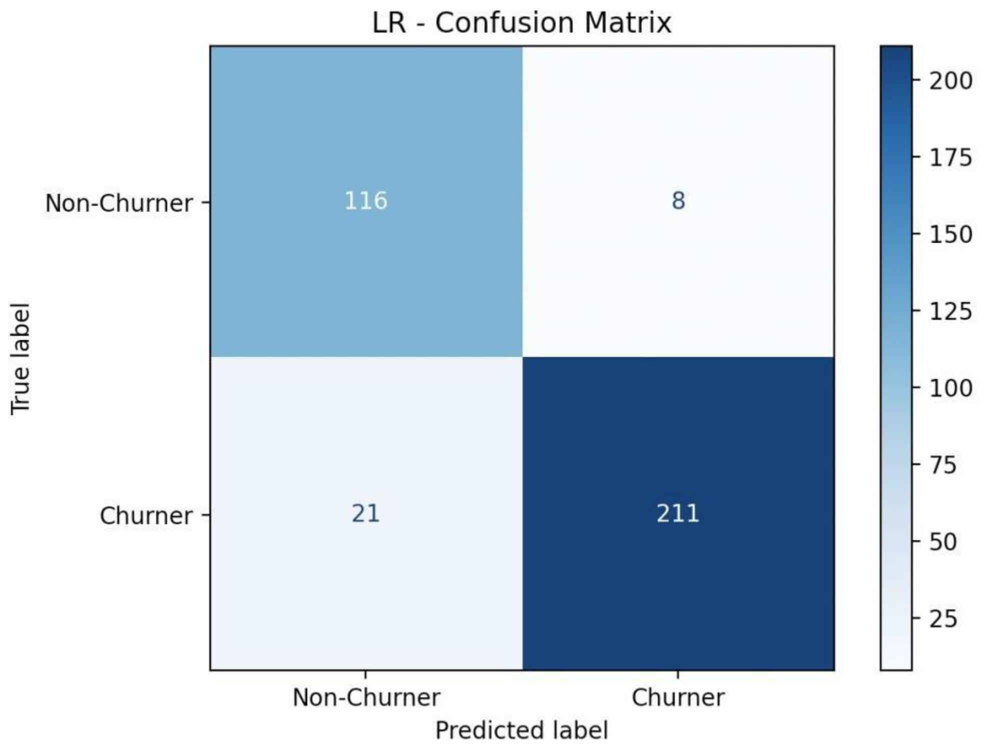
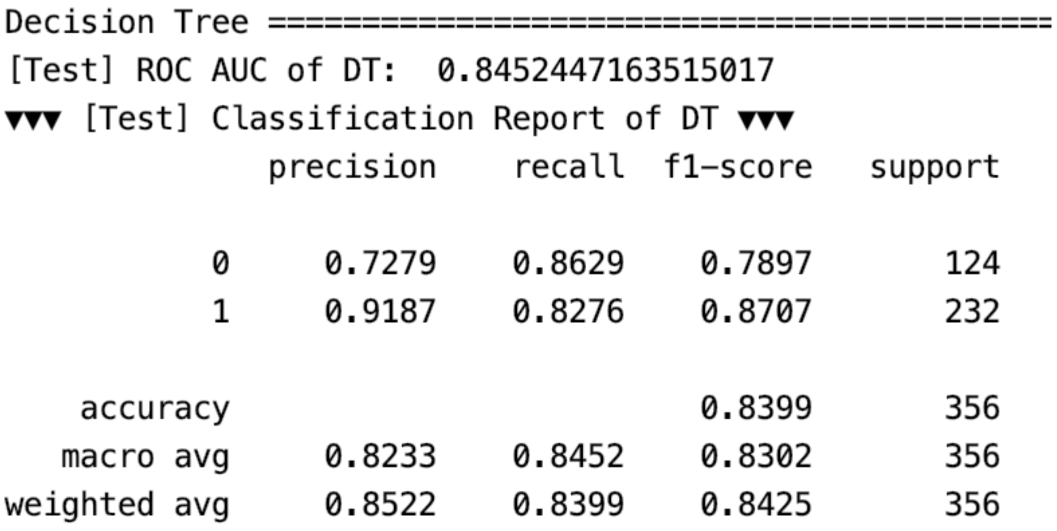
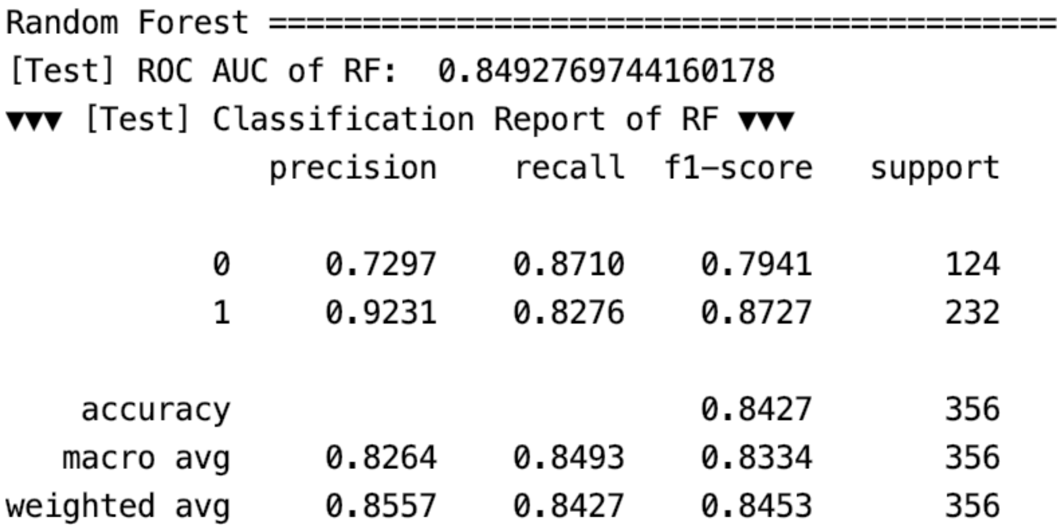


Figure 40. LR - Confusion Matrix



```
Decision Tree ==========================================
[Test] ROC AUC of DT:  0.8452447163515017
▼▼▼ [Test] Classification Report of DT ▼▼▼
              precision    recall  f1-score   support

           0     0.7279    0.8629    0.7897       124
           1     0.9187    0.8276    0.8707       232

    accuracy                         0.8399       356
   macro avg     0.8233    0.8452    0.8302       356
weighted avg     0.8522    0.8399    0.8425       356
```

Figure 41. DT - Prediction Results



```
Random Forest ==========================================
[Test] ROC AUC of RF:  0.8492769744160178
▼▼▼ [Test] Classification Report of RF ▼▼▼
              precision    recall  f1-score   support

           0     0.7297    0.8710    0.7941       124
           1     0.9231    0.8276    0.8727       232

    accuracy                         0.8427       356
   macro avg     0.8264    0.8493    0.8334       356
weighted avg     0.8557    0.8427    0.8453       356
```

Figure 42. RF - Prediction Results

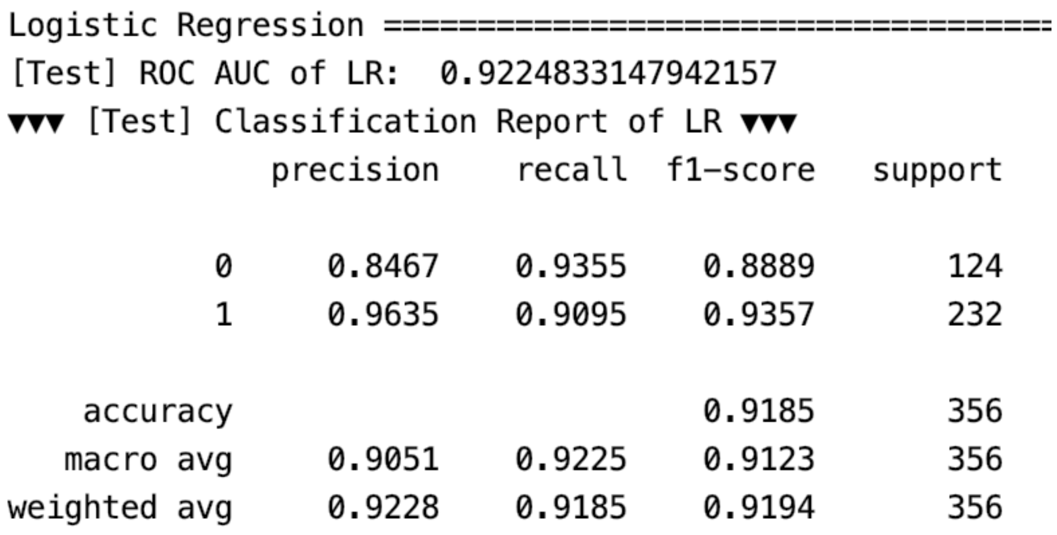

```
Logistic Regression =====================================
[Test] ROC AUC of LR:  0.9224833147942157
▼▼▼ [Test] Classification Report of LR ▼▼▼
              precision    recall  f1-score   support

           0     0.8467    0.9355    0.8889       124
           1     0.9635    0.9095    0.9357       232

    accuracy                         0.9185       356
   macro avg     0.9051    0.9225    0.9123       356
weighted avg     0.9228    0.9185    0.9194       356
```

Figure 43. LR - Prediction Results

Table 32. Summary of Results

Metrics/Algorithms	DT	RF	LR
Accuracy	0.8399	0.8427	0.9185
Precision	0.8522	0.8557	0.9228
Recall	0.8399	0.8427	0.9185
F1-Score	0.8425	0.8453	0.9194
AUC	0.8452	0.8493	0.9225

5.5 Discussion

The combination of churn determination and prediction presented above showed the effectivenessof the approach proposed in digital game-based learning. The defining churn based on the arbitrary time frame for each user can apply to many digital game-based learning services even if there are various course duration or no specified course duration. This isbecause the recency and inactive time between logins of each user should be available or calculatable for most of the services. In addition, this approach allows us to have the flexibility to define churners even if the users are playing in mid-course. Since having a fixed evaluation time and cutoff like the online gaming industry or having a clear definition of a drop-off with various evaluation periods was difficult to apply digital game-based learning, the applicability and flexibility of the proposed churn determination can address the issues. Thus, this determination approach is beneficial to digital game-based learning industry. The churn prediction with the best performance LR model used the three categories of input variables. The categories are demographic, engagement, and performance which are common values for digital game-based learning, online gaming and MOOCs. Although the detailed variables can differ from service to service, the input categories for better churn prediction in digital game-based learning were identified.

A limitation of the proposed approach is that the defining churn is not suitable for the users whojust have started playing. The calculation requires the mean inactive time and standard deviation inactive time, so users with only a couple of logins might lead to less reliable results. The prediction of early churn will not be suitable with this approach. Another limitation is data size.The number of observations used for the churn prediction model is 3,557. This is considered asmaller dataset. The LR was the best performance model with the case study, but other tree-based algorithms had good results. This implies that the other models can improve the performance with different dataset sizes and input variables. Hence, the proposed prediction model may produce different results if the large dataset size.

Considering the above two limitations, the procedure can be further developed. Defining the early churn determination and prediction for shorter courses is one possible future work. As forthe larger dataset, RF may have better performance. Or, different models such as neural networks and deep neural networks can be tested. Another approach with a bigger dataset is to try chapter-based churn prediction, which helps create more specific marketing strategies for each chapter churner.

SECTION 6: CONCLUSION

Digital game-based learning is a narrowed down category of EdTech, and digital version of game-based learning is an approach to facilitate learning with gameplay. The digital game-based learning market has been growing, and the market revenue is expected to reach over $24 billion by 2024. The market expansion has been supported by the innovation of technologysuch as AR, AI, and a variety of customers from preschool to corporations. One of the common issues in the EdTech market is the higher churn rate. Retention and churn rate are common performance key indicators for business. Customer churn is a percentage of customers who stopped using a service or products in a certain period, and retention is the percentage to maintain customers using the service or products over time.

Keeping customers brings more profit because it is less expensive than acquiring new customers,and loyal customers tend to spend more money on the service. Therefore, understanding and analyzing retention and churn is important for a business. The reported churn rate in EdTech is wide-ranging depending on the source, such as a 49.9% churn rate for a virtual school or a 95% churn rate for MOOCs. The only thing clear is that thechurn rate is on a higher side for EdTech compared to other industries. To address the issue, a marketing approach is becoming more important. However, because the market is still in the early stage, there are few studies related to marketing perspectives even though there are great EdTech and digital game-based learning industries. The digital game-based learning market situation in Japan is similar. The market is growing and there are more than 100 services in the EdTech industry. The studies related to the field are extremely limited compared to overseas. In addition, the approach in education or online gaming industries can be only partially applicable to digital game-based learning because of the difference in the definition of churn, evaluation time frame for churners, and social interaction.One of the popular approaches for retention management is churn prediction.

Churn predictionallows companies to create better marketing strategies to improve user retention. Using a dataset from a Japanese company providing a digital game-based learning service as a case study,an approach for churn prediction for digital game-based learning is proposed. There are four objectives for this project. The first one is the determination of churn, which is defined by applying an arbitrary time frame for each user. The calculation compares the recency and average and two standard deviations of user inactive time. The second objective isto clarify the churn rate of the Japanese service. Using the defined churn definition, the churnrate became evident as 56.77% with the case study data. In addition, the descriptive analysis and retention analysis were conducted. The descriptive analysis uncovered the users' MAU and popular dates and time.

The retention analysis reveals the different behaviour of churners and non-churners per chapter. The third objective is to develop the best churn prediction model by comparing LR, DT, and RF models. Feature selection, dataset split ratio comparison, and hyperparameter tuning are conducted to build the best performance models. The LR model has a distinctively highest accuracy of 0.9185, AUC of 0.9225, and an F1-scoreof 0.9194. The higher F1-score means the recall and precision are well balanced. The 0.9225 of AUC is on the higher side than the AUC of past churn prediction studies in onlinegaming and education industries. Consequently, the results indicate the proposed approach's effectiveness in determining churn and churn prediction in digital game-based learning.

REFERENCES

Banerjee, T., Mukherjee, G., Dutta, S. and Ghosh, P. (2019) A Large-Scale Constrained JointModeling Approach for Predicting User Activity, Engagement, and Churn With Application to Freemium Mobile Games. *Journal of the American Statistical Association*. [Online] p. 1-
29. Available from: https://doi.org/10.1080/01621459.2019.1611584 [Accessed 19 December2019].

Beachum, L. (2020) *World'S Oldest Person Breaks Her Own Record By Turning 117*. [Online] The Washington Post. Available from: https://www.washingtonpost.com/world/2020/01/06/worlds-oldest-woman-breaks-her-own- record-by-turning/ [Accessed 23 April 2020].

Bolliger, D., Mills, D., White, J. and Kohyama, M. (2015) Japanese Students' Perceptions of Digital Game Use for English-Language Learning in Higher Education. *Journal of Educational Computing Research*. [Online] 53(3). p. 384-408. Available from: https://journals.sagepub.com/doi/10.1177/0735633115600806 [Accessed 19 December 2019].

Bote-Lorenzo, M. and Gómez-Sánchez, E. (2017) *Predicting the decrease of engagement indicators in a MOOC.* In: Proceedings of the Seventh International Learning Analytics &Knowledge Conference on - LAK '17. [Online] New York, NY, USA: ACM, p. 143-147. Available from: https://dl.acm.org/doi/10.1145/3027385.3027387 [Accessed 19 December2019].

Byun, J. and Joung, E. (2018) Digital game-based learning for K-12 mathematics education:A meta-analysis. *School Science and Mathematics.* [Online] 118(3-4). p. 113-126. Availablefrom: https://doi.org/10.1111/ssm.12271 [Accessed 19 December 2019].

Calabrese, B. (2019) Data Reduction. *Encyclopedia of Bioinformatics and ComputationalBiology*. [Online] 1. p. 480-485. Available from: https://www.sciencedirect.com/science/article/pii/B9780128096338204603 [Accessed 18
April 2020].

CallMiner (2019) *US CallMiner Index: Consumers Switch By Sector, The Reasons* *and theImpact of Call Centers*. [Online] Available from: https://learn.callminer.com/whitepapers/callminerindex-us-consumers-switch-by-sector [Accessed 19 December 2019].

Clarke, D. and Kinghorn, R. (2018) *Experience is everything: here's how to get it right*. [Online] PwC. Available from: https://www.pwc.com/us/en/services/consulting/library/consumer-intelligence-series/future- of-customer-experience.html [Accessed 11 December 2019].

Costa, E., Fonseca, B., Santana, M., de Araújo, F. and Rego, J. (2017). Evaluating the effectiveness of educational data mining techniques for early prediction of students' academicfailure in introductory programming courses. *Computers in Human Behavior*. [Online] 73(8).
p. 247-256. Available from: https://www.sciencedirect.com/science/article/pii/S0747563217300596 [Accessed 19 December 2019].

Fu, X., Chen, X., Shi, Y., Bose, I. and Cai, S. (2017) User segmentation for retention management in online social games. *Decision Support Systems*. [Online] 101(9). p. 51-68.Available from: https://www.sciencedirect.com/science/article/pii/S0167923617301008 [Accessed 19 December 2019].

Fujimoto, T. and Yamada, M. (2013) Review of Educational Games Research for Informal Learning： Research Frameworks and Evaluation Methods. *Japan Society for EducationalTechnology NII-Electronic Library Service J.* [Online] 37(3). p. 343-351. Available from: https://doi.org/10.15077/jjet.KJ00008987694 [Accessed 19 December 2019]. (in Japanese)

Fujimoto, T. (2011) Toward Effective Approaches to Educational Use of Digital Games. *Computer & Education.* [Online] 31. p. 10-15. Available from: https://doi.org/10.14949/konpyutariyoukyouiku.31.10 [Accessed 19 December 2019]. (inJapanese)

Fujimoto, T., Shigeta, K. and Fukuyama, Y. (2016) The Research Trends in Game-Based Learning and Open Education. *Educational technology research.* [Online] 39(1). Availablefrom: https://doi.org/10.15077/etr.41038 [Accessed 19 December 2019].

Gallo, A. (2014) *The Value of Keeping the Right Customers*. [Online] Harvard Business Review. Available from: https://hbr.org/2014/10/the-value-of-keeping-the-right-customers[Accessed 19 December 2019].

Gee, J. (2003) What video games have to teach us about learning and literacy. Basingstoke,UK: Palgrave Macmillan.

Gomez-Uribe, C. and Hunt, N. (2015) The Netflix Recommender System. *ACM Transactionson Management Information Systems.* [Online] 6(4). p. 1-19. Available from: https://dl.acm.org/doi/10.1145/2843948 [Accessed 19 December 2019].

Gupta, S. and Sabitha, A. (2018) Deciphering the attributes of student retention in massive open online courses using data mining techniques. *Education and Information Technologies*.[Online] 24(3). p. 1973-1994. Available from: https://doi.org/10.1007/s10639-018-9829-9 [Accessed 19 December 2019].

Hadiji, F., Sifat, R., Drachen, A., Thurau, C., Kersting, K. and Bauckhaget, C. (2014) *Predicting player churn in the wild.* In: 2014 IEEE Conference on Computational Intelligenceand Games. [Online] Piscataway, New Jersey, US: IEEE. Available from: https://ieeexplore.ieee.org/document/6932876 [Accessed 19 December 2019].

Han, J., Kamber, M. and Pei, J. (2012) *Data mining: Concepts and Techniques.* 3rd ed.Amsterdam, Netherlands: Elsevier/Morgan Kaufmann. p. 83-124.

HolonIQ, H. (2019) *10 charts that explain the Global Education Technology Market.* [Online]Holoniq.com. Available from: https://www.holoniq.com/edtech/10-charts-that-explain-the- global-education-technology-market/ [Accessed 19 December 2019].

Hone, K. and El Said, G. (2016) Exploring the factors affecting MOOC retention: A surveystudy. *Computers & Education.* [Online] 98(6). p. 157-168. Available from: https://www.sciencedirect.com/science/article/pii/S0360131516300793 [Accessed 19 December 2019].

Hussein, M., Ow, S., Cheong, L., Thong, M. and Ale Ebrahim, N. (2019) Effects of Digital Game-Based Learning on Elementary Science Learning: A Systematic Review. *IEEE Access*[Online] 7. p. 62465-62478. Available from: https://ieeexplore.ieee.org/document/8713478 [Accessed 19 December 2019].

Information Resources Management Association (2015) *Gamification: Concepts, Methodologies, Tools, and Applications.* [Online] Hershey, PA, USA: Information ScienceReference. Available from: https://www.igi-global.com/book/gamification-concepts- methodologies-tools-applications/121154 [Accessed 19 December 2019].

Franciosi, S. (2014) Educator Perceptions Of Digital Game-Based Learning In The InstructionOf Foreign Languages In Japanese Higher Education. Michigan, USA: ProQuest LLC, pp.1 - 166.

Jain, S., Shukla, S. and Wadhvani, R. (2018) Dynamic selection of normalization techniques using data complexity measures. *Expert Systems with Applications*. [Online] 106(16). p. 252-
262. Available from:
https://www.sciencedirect.com/science/article/pii/S095741741830232X[Accessed 19 December 2019].

Jeni, L., Cohn, J. and De La Torre, F. (2013) *Facing Imbalanced Data--Recommendations forthe Use of Performance Metrics.* In: 2013 Humaine Association Conference on Affective Computing and Intelligent Interaction. [Online] Piscataway, New Jersey, US: IEEE. p. 245–
251. Available from: https://ieeexplore.ieee.org/document/6681438 [Accessed 14 April 2020].

Khan, A., Ahmad, F. and Malik, M. (2017) Use of digital game based learning and gamification in secondary school science: The effect on student engagement, learning and gender difference. *Education and Information Technologies.* [Online] 22(6). p. 2767-2804. Available from: https://dl.acm.org/citation.cfm?id=3174821 [Accessed 19 December 2019].

Khobragade, L. and Mahadik, P. (2015) Students' Academic Failure Prediction Using DataMining. *International Journal of Advanced Research in Computer and Communication Engineering.* [Online] 4(11). p. 290-298. Available from: https://ijarcce.com/wp- content/uploads/2015/12/IJARCCE-65.pdf [Accessed 19 December 2019].

Kumabe, D. (2018) *Current Situation and Issues of EdTech Market*. [Online] Nri.com. Available from: https://www.nri.com/-
/media/Corporate/jp/Files/PDF/knowledge/publication/it_solution/2018/06/ITSF18060
5.pdf?l a=ja-JP&hash=D06155068CA89444E1336B1784D38EA1FE8D9B0E
[Accessed 19
December 2019]. (in Japanese)

Lee, E., Kim, B., Kang, S., Kang, B., Jang, Y. and Kim, H. (2018) Profit Optimizing ChurnPrediction for Long-term Loyal Customer in Online games. *IEEE Transactions on Games*. [Online] 10. p. 1-1. Available from:
https://ieeexplore.ieee.org/document/8485736 [Accessed 19 December 2019].

Levy, Y. (2007) Comparing dropouts and persistence in e-learning courses. *Computers andEducation.* [Online] 48(2). p. 185-204. Available from: https://www.sciencedirect.com/science/article/pii/S0360131505000096 [Accessed 19
December 2019].

Liu, X., Xie, M., Wen, X., Chen, R., Ge, Y., Duffield, N. and Wang, N. (2018) *A Semi- Supervised and Inductive Embedding Model for Churn Prediction of Large-Scale Mobile Games.* In: 2018 IEEE International Conference on Data Mining (ICDM). [Online] IEEE.

Piscataway, New Jersey, US. Available from: https://ieeexplore.ieee.org/document/8594852[Accessed 19 December 2019].

Liubov, L. (2019) Students churn prediction task in MOOC. *Journal of Computer Science Research.* [Online] 1(1). p. 29-35. Available from: https://ojs.bilpublishing.com/index.php/jcsr/article/view/537 [Accessed 19 December 2019].

Mao, W. and Wang, F. (2012) Chapter 8 - Cultural Modeling for Behavior Analysis and Prediction. *New Advances in Intelligence and Security Informatics* [Online] p. 91-102. Available from: https://www.sciencedirect.com/science/article/pii/B9780123972002000087[Accessed 18 Apr. 2020].

McKinney, W. and Pandas Development Team (2020) *Pandas.Dataframe.Boxplot — Pandas 1.0.3 Documentation.* [Online] Pandas.pydata.org. Available from: https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.boxplot.html [Accessed 18 April 2020].

Mariscal, G., Marbán, Ó. and Fernández, C. (2010) A survey of data mining and knowledge discovery process models and methodologies. *The Knowledge Engineering Review*. [Online] 25(2). p. 137-166. Available from: https://doi.org/10.1017/S0269888910000032 [Accessed 19 December 2019].

Marquez-Vera, C., Morales, C.R. and Soto, S.V. (2013) Predicting School Failure and Dropout by Using Data Mining Techniques. *IEEE Revista Iberoamericana de Tecnologias delAprendizaje.* 8(1). p. 7–14. Available from: https://ieeexplore.ieee.org/document/6461622 [Accessed 19 December 2019].

Miller, A., Vonwiller, B. and Weed, P. (2016) *Grow fast or die slow: Focusing on customersuccess to drive growth.* [Online] McKinsey & Company. Available from: https://www.mckinsey.com/industries/technology-media-and-telecommunications/our- insights/grow-fast-or-die-slow-focusing-on-customer-success-to-drive-growth [Accessed 11 December 2019].

Milošević, M., Živić, N. and Andjelković, I. (2017) Early churn prediction with

personalizedtargeting in mobile social games. *Expert Systems with Applications.* [Online] 83(17). p. 326-
332. Available from:
https://www.sciencedirect.com/science/article/abs/pii/S0957417417303044
[Accessed 19December 2019].

Miyauchi, T., Hosoya, T. and Urabe, H. (2019) *Madefor/Postal-Code-Api.* [Online] GitHub.Available from: https://github.com/madefor/postal-code-api [Accessed 18 April 2020].

Niu, Z., Li, W., Yan, X. and Wu, N. (2018) *Exploring causes for the dropout on massive openonline courses.* In: Proceedings of ACM Turing Celebration Conference - China on - TURC '18. [Online] New York, NY, USA: ACM. p. 47-52. Available from: https://dl.acm.org/citation.cfm?doid=3210713.3210727 [Accessed 19 December 2019].

Perini, S., Luglietti, R., Margoudi, M., Oliveira, M. and Taisch, M. (2018) Learning and motivational effects of digital game-based learning (DGBL) for manufacturing education – The Life Cycle Assessment (LCA) game. *Computers in Industry*. [Online] 102(9). p. 40-49.Available from: https://www.sciencedirect.com/science/article/abs/pii/S0166361517300398 [Accessed 19 December 2019].

Pokorná, M. and Sponer, M. (2016) Social Lending and Its Risks. *Procedia - Social andBehavioral Sciences.* [Online] 220(5). p. 330-337. Available from: https://www.sciencedirect.com/science/article/pii/S1877042816306073 [Accessed 13 December 2019].

Prensky, M. (2001) Digital game-based learning. Mcgraw Hill Book Company: McGraw-Hill.

Recurly Inc (2019) *Benchmarks for Subscription E-Commerce. [*Online] Available from:https://info.recurly.com/research/benchmarks-for-subscription-ecommerce [Accessed 19 December 2019].

S. Adkins, S. (2019) *The 2019-2024 Global Game-based Learning Market.* [Online]Seriousplayconf.com. Available from: https://seriousplayconf.com/newtest/wp-content/uploads/2019/11/Metaari_2019-2024_Global_Game-based_Learning_Market_Executive_Overview.pdf [Accessed 19 December 2019].

Sanchez, E. (2019) *Game-Based Learning*. 2nd ed. Cham, Switzerland: Springer.

Smith, W. (1956) Product Differentiation and Market Segmentation as Alternative MarketingStrategies. *Journal of Marketing*. [Online] 21(1). p. 3-8. Available from: https://www.jstor.org/stable/1247695. [Accessed 19 December 2019].

Sorensen, C. and Donovan, J. (2017) An examination of factors that impact the retention ofonline students at a for-profit university. Online Learning. [Online] 21(3). p. 206-211.
Available from: http://dx.doi.org/10.24059/olj.v21i3.935 [Accessed 19 December 2019].

Sørensen, B., Meyer, B. and Egenfeldt-Nielsen, S. (2011) Serious Games in Education: AGlobal Perspective. Aarhus, Denmark: Aarhus University Press.

Strauss, V. (2019) *New report on virtual education: 'It sure sounds good. As it turns out, it's too good to be true.'.* [Online] Washington Post. Available from: https://www.washingtonpost.com/education/2019/05/29/new-report-virtual-education-it-sure- sounds-good-it-turns-out-its-too-good-be-true/ [Accessed 19 December 2019].

Studyplus, Inc. (2018) *EdTech Market Map in Japan.* [Online] Studyplus. Available from:https://info.studyplus.co.jp/2018/04/02/815 [Accessed 19 December 2019]. (in Japanese)

Suh, E. and Alhaery, M. (2016) Customer Retention: Reducing Online Casino Player Churn Through the Application of Predictive Modeling. *UNLV Gaming Research & Review Journal*.[Online] 20(2). p. 63-84. Available from: https://digitalscholarship.unlv.edu/grrj/vol20/iss2/6/[Accessed 19 December 2019].

Tamassia, M., Raffe, W., Sifa, R., Drachen, A., Zambetta, F. and Hitchens, M. (2016) *Predicting player churn in destiny: A Hidden Markov models approach to predicting player departure in a major online game*. In: 2016 IEEE Conference on Computational Intelligenceand Games (CIG). [Online] Piscataway, New Jersey, US: IEEE. Available from: https://ieeexplore.ieee.org/abstract/document/7860431 [Accessed 19 December 2019].

Utashiro, M., Hashimoto, S. and Hayashi, A. (2015) Research on Acquisition of Hiragana inNormally Developing Children. *Bulletin of Tokyo Gakugei University. Division of comprehensive educational science*. [Online] p. 397-402. Available from: https://ci.nii.ac.jp/naid/110009890163 [Accessed 21 April 2020]. (in Japanese)

Xie, H., Devlin, S., Kudenko, D. and Cowling, P. (2015) *Predicting player disengagement and first purchase with event-frequency based data representation.* In: 2015 IEEE Conferenceon Computational Intelligence and Games (CIG). [Online] Piscataway, New Jersey, US: IEEE. Available from: https://ieeexplore.ieee.org/document/7317919 [Accessed 19 December2019].

Xing, W., Chen, X., Stein, J. and Marcinkowski, M. (2016). Temporal predication of dropoutsin MOOCs: Reaching the low hanging fruit through stacking generalization. *Computers in Human Behavior.* [Online] 58(6). p. 119-129. Available from:

https://www.sciencedirect.com/science/article/pii/S074756321530279X [Accessed 19 December 2019].

Yan, A., Lee, M. and Ko, A. (2019) *Predicting abandonment in online coding tutorials*. In:2017 IEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC). [Online] Piscataway, New Jersey, US: IEEE. p. 191-199. Available from: https://ieeexplore.ieee.org/document/8103467 [Accessed 19 December 2019].

Yang, C., Shi, X., Jie, L. and Han, J. (2018) *I Know You'll Be Back.* Proceedings of the 24thACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '18). London, United Kingdom: ACM. p. 914-922.

Yi, Z. (2017) *Marketing services and resources in information organizations.* 1st ed. Oxford,United Kingdom: Chandos Publishing.

Yukselturk, E., Ozekes, S. and Türel, Y.K. (2014) Predicting Dropout Student: An Application of Data Mining Methods in an Online Education Program. *European Journal ofOpen, Distance and E-Learning.* 17(1). p. 118–133. Available from: https://doi.org/10.2478/eurodl-2014-0008 [Accessed 19 December 2019].

Zhonggen, Y. (2019) A Meta-Analysis of Use of Serious Games in Education over a Decade. *International Journal of Computer Games Technology.* [Online] 2019. p. 1-8. Available from:https://doi.org/10.1155/2019/4797032 [Accessed 19 December 2019].

APPENDIX

Appendix A - Entire Python Code

```
import pandas as pd
import re
import numpy as np
from datetime import datetime
import requests
import statistics
import matplotlib.pyplot as plt
from matplotlib.legend_handler import HandlerLine2D
from sklearn import preprocessing
from sklearn.decomposition import PCA
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report,
roc_curve, auc,plot_confusion_matrix
from sklearn.model_selection import train_test_split, cross_val_score
from sklearn.tree import DecisionTreeClassifier
from sklearn.linear_model import
LogisticRegression

# 
============================================================================
==
=========================
# Functions
============================================================================
==
===============
# import csv files into a dataframe
def import_log_files(path, header):
    if header == "None":  # No header in the csv. attach the specified
    header
df = pd.read_csv(path, header=None, names=["id", "timestamp",
"action"])
    elif header == "Have":  # Have header in the csv. read the file
        df = pd.read_csv(path)
    else:
        df = pd.read_csv(path)
    ret
urn df

# combine user game activity log and user lesson log into one player log
def create_player_log():
    print("***** BEGIN creating player log by merging player and lesson logs
*****")
    # import account_log
    path_al = "/Users/maikiguchi/Desktop/cp2/ORIG_account_logs.csv"
    ac_df = import_log_files(path=path_al, header="None")
index_names = ac_df[ac_df["action"] == "crm_update_player"].index
#removing parent account log
    ac_df.drop(index_names, inplace=True)
    ac_df =
ac_df.reset_index(drop=True)
    # import lesson_log
    path_ll = "/Users/maikiguchi/Desktop/cp2/ORIG_lesson_logs.csv"
    le_df = import_log_files(path=path_ll,
header="None")
    # join the two dataframe, then sort
    pl_log_df = pd.concat([ac_df, le_df], join="outer")
    pl_log_df = pl_log_df.sort_values(by=["id", "timestamp"],
    ascending=True)
    pl_log_df =
pl_log_df.reset_index(drop=True)
pl_log_df.to_csv("/Users/maikiguchi/Desktop/cp2/1_combined_player_log.csv",index=False)
```

```
    print("***** FIN creating player log by merging player and lesson logs
*****")


# create exp, coins, and replay aggregated data
def create_exp_rep_aggregated():
    print("***** BEGIN merging and aggregating exp, coins, and replay
    *****")
    # import exp data from csv
path_exp =
"/Users/maikiguchi/Desktop/cp2/ORIG_player_exp_coins_2020_01_2
7.csv"
exp_df = pd.read_csv(path_exp, usecols=[0, 4, 6], header=0)  # import
onlyspecific columns
    exp_df = exp_df.rename(columns={"account_id":
"id"})
    # import replay data from csv
path_rep =
"/Users/maikiguchi/Desktop/cp2/ORIG_player_replay_2020_01_2
7.csv"
rep_df = pd.read_csv(path_rep, usecols=[1, 2], header=0)  # import
onlyspecific columns
    rep_df = rep_df.rename(columns={"account_id": "id", "START":
"replay"})
    # sort by account id
    rep_df = rep_df.sort_values(by=["id"], ascending=True)
    exp_df = exp_df.sort_values(by=["id"],
ascending=True)
    # aggregate exp and coins per id, then merge them
    exp_df.set_index("id")
    exp_agg_df = exp_df.groupby(["id"])["exp"].sum()
    coins_agg_df = exp_df.groupby(["id"])["coins"].sum()
    exp_df = pd.merge(exp_agg_df, coins_agg_df, on="id",
how="left")
    # count how many times replay, then merge with exp_df
    rep_df = rep_df.groupby("id")["replay"].nunique()
    exp_agg_df = pd.merge(exp_df, rep_df, on="id", how="left")
    exp_rep_df =
exp_agg_df.fillna(0)
exp_rep_df.to_csv("/Users/maikiguchi/Desktop/cp2/2_aggregated_exp_rep_c
oins.csv", index=True)
    print("***** FIN merging and aggregating exp, coins, and replay
*****")


# transform CRM player log (from Japanese to English, calculate
age, getprefecture neme) and remove test accounts
def transform_crm_log():
    print("***** BEGIN transforming CRM data *****")
    # import CRM data
path_crm =
"/Users/maikiguchi/Desktop/cp2/ORIG_crm_players_2020_03_2
8.csv"
tmp_crm_df = pd.read_csv(path_crm, usecols=[0, 1, 3, 11, 12, 13,
14],header=0)  # import only specific columns

    # change the column name from Japanese to English
    crm_df = tmp_crm_df.rename(
columns={"プレイアカウント ID": "account_id", "プレイアカウントメールアドレス
": "player_email", "管理アカウントメールアドレス": "parent_email",
                 "性別": "gender", "生年月日": "birthday", "郵便番号": "postal",
                 "
住所": "address"})
    crm_df = crm_df.dropna(subset=["birthday"])  # drop unfilled test data
    crm_df =
crm_df.reset_index(drop=True)
# create a list of test accounts by email-address, and
transforminformation
```

```
    len_crm = len(crm_df)
    test_users = []
    for i in range(0, len_crm):
        print(i)
        play_adr = crm_df.at[i, "player_email"]
        prnt_adr = crm_df.at[i, "parent_email"]
        gender = crm_df.at[i, "gender"]
        birthday = crm_df.at[i, "birthday"]
        postal = crm_df.at[i, "postal"]
        address = crm_df.at[i, "address"]

        # Gender from Japanese to English
        if gender == "男性":
            crm_df.at[i, "gender"] = "Male"
        elif gender == "女性":
            crm_df.at[i, "gender"] = "Female"

        # Calculate Age by function
        age = calculate_age(birthday)
        crm_df.at[i, "birthday"] = age

        # Convert postal code to prefecture in English
        prefecture = postal_2_en_pref(postal)
        crm_df.at[i, "postal"] = prefecture

        # Check test users by email and address and put them in thetest_user list
        if not pd.isnull(play_adr):
            play = re.search("@lifeistech.co.jp", play_adr)
            if play != None and play_adr != "":
                test_users.append(crm_df.at[i, "account_id"])
        elif not pd.isnull(prnt_adr):
            parent = re.search("@lifeistech.co.jp", prnt_adr)
            if parent != None and (
                    prnt_adr != "ibaraki@lifeistech.co.jp" and prnt_adr !="aoyagi@lifeistech.co.jp" and prnt_adr != "clarkosaka_2019@lifeistech.co.jp"):
                test_users.append(crm_df.at[i, "account_id"])

        address = address.replace(" ", "")
        if re.search(".*南麻布 2-12-3.*", address):
            test_users.append(crm_df.at[i, "account_id"])

    test_usr_df = pd.DataFrame(test_users, columns=["account_id"])
    crm_df = crm_df[~crm_df.account_id.isin(test_usr_df.account_id.values)]# remove accounts in test_usr_df
    crm_df = crm_df.drop(columns=["player_email", "parent_email","address"])  # drop unnecessary columns for analysis
    crm_df = crm_df.rename(columns={"birthday": "age", "postal":"prefecture"})  # change the columns name

    crm_df.to_csv("/Users/maikiguchi/Desktop/cp2/3_crm.csv", index=False)
    test_usr_df.to_csv("/Users/maikiguchi/Desktop/cp2/4_test_accounts.csv",index=False)
    print("***** FIN transforming CRM data *****")
    return test_usr_df


# calculate age
def calculate_age(birthday):
    birth_date = datetime.strptime(birthday, "%Y/%m/%d")
    today = datetime.today()
    age = today.year - birth_date.year - ((today.month, today.day) <(birth_date.month, birth_date.day))

    return age


# get English prefecture name from postal code
def postal_2_en_pref(postal):
    # Using API to get English information
    response = requests.get(
        "https://madefor.github.io/postal-code-api/api/v1/" +
    postal[:3] +"/" + postal[4:8] + ".json")
    if response.ok:
        results = response.json()["data"]
        prefecture = results[0]["en"]["prefecture"]
    else:
        prefecture = None

    return prefecture


# create aggregation data of lesson's chapter progress for each user
def create_chapter_aggregated(unique_id):
    print("***** BEGIN creating chapter aggregated data *****")
    # import chapter data from csv
    path_ch = "/Users/maikiguchi/Desktop/cp2/ORIG_player_chapter_waiting_2020_01_27.csv"
    ch_df = import_log_files(path=path_ch, header="Have")

    # remove unnecessary characters then sort
    ch_df["cleared"] = ch_df["cleared"].str.replace("T|Z", " ")
    ch_df["opened"] = ch_df["opened"].str.replace("T|Z", " ")
    ch_df["begined"] = ch_df["begined"].str.replace("T|Z", " ")
    ch_df = ch_df.sort_values(by=["account_id", "chapter"], ascending=True)
    unique_id_df = pd.DataFrame(unique_id, columns=["id"])
    ch_df = ch_df[ch_df.account_id.isin(unique_id_df.id.values)]
    ch_df = ch_df.reset_index(drop=True)

    # import chapter 7 data from csv
    path_ch7 = "/Users/maikiguchi/Desktop/cp2/ORIG_Ch7_fin_2020_01_27.csv"
    ch7_df = import_log_files(path=path_ch7, header="Have")
    ch7_df["cleared"] = ch7_df["cleared"].str.replace("T|Z", " ")
    ch7_df = ch7_df.sort_values(by=["account_id", "cleared"],ascending=True)
    ch7_df = ch7_df.reset_index(drop=True)

    # create a new dataframe for aggregation
    ch_agg_df = pd.DataFrame(columns=["id", "ch1_fin", "ch2_open","ch2_start", "ch2_fin",
                                      "ch3_open", "ch3_start", "ch3_fin",
                                      "ch4_open", "ch4_start", "ch4_fin",
                                      "ch5_open", "ch5_start", "ch5_fin",
                                      "ch6_open", "ch6_start", "ch6_fin",
                                      "ch7_open", "ch7_start", "ch7_fin",
                                      "ch2_wait(days)", "ch3_wait(days)", "ch4_wait(days)",
                                     "ch5_wait(days)", "ch6_wait(days)","ch7_wait(days)", "avr_ch_wait(days)"])

    # gather chapter info for each id
    len_ch_agg = len(ch_df)
    ch_log_idx = 0
    for unq_idx in range(len(unique_id)):
        print(unq_idx)
        ch1_fin = ch2_op = ch2_st = ch2_fin = ch3_op = ch3_st = ch3_fin =ch4_op = ch4_st = 0
        ch4_fin = ch5_op = ch5_st = ch5_fin = ch6_op = ch6_st = ch6_fin =ch7_op = ch7_st = 0
        ch2_wait = ch3_wait = ch4_wait = ch5_wait = ch6_wait = ch7_wait =avr_wait = 0
215.
        test1 = unique_id[unq_idx]
        test2 = ch_df.at[ch_log_idx, "account_id"]
        while (ch_log_idx < len_ch_agg) and (unique_id[unq_idx] ==ch_df.at[ch_log_idx, "account_id"]):
            if ch_df.at[ch_log_idx, "chapter"] == 2:
                ch1_fin = ch_df.at[ch_log_idx, "cleared"]
                ch2_op = ch_df.at[ch_log_idx, "opened"]
                ch2_st = ch_df.at[ch_log_idx, "begined"]
                ch2_wait = ch_df.at[ch_log_idx, "days_begin"]
```

```
            elif ch_df.at[ch_log_idx, "chapter"] == 3:
                ch2_fin = ch_df.at[ch_log_idx, "cleared"]
                ch3_op = ch_df.at[ch_log_idx, "opened"]
                ch3_st = ch_df.at[ch_log_idx, "begined"]
                ch3_wait = ch_df.at[ch_log_idx, "days_begin"]
            elif ch_df.at[ch_log_idx, "chapter"] == 4:
                ch3_fin = ch_df.at[ch_log_idx, "cleared"]
                ch4_op = ch_df.at[ch_log_idx, "opened"]
                ch4_st = ch_df.at[ch_log_idx, "begined"]
                ch4_wait = ch_df.at[ch_log_idx, "days_begin"]
            elif ch_df.at[ch_log_idx, "chapter"] == 5:
                ch4_fin = ch_df.at[ch_log_idx, "cleared"]
                ch5_op = ch_df.at[ch_log_idx, "opened"]
                ch5_st = ch_df.at[ch_log_idx, "begined"]
                ch5_wait = ch_df.at[ch_log_idx, "days_begin"]
            elif ch_df.at[ch_log_idx, "chapter"] == 6:
                ch5_fin = ch_df.at[ch_log_idx, "cleared"]
                ch6_op = ch_df.at[ch_log_idx, "opened"]
                ch6_st = ch_df.at[ch_log_idx, "begined"]
                ch6_wait = ch_df.at[ch_log_idx, "days_begin"]
            elif ch_df.at[ch_log_idx, "chapter"] == 7:
                ch6_fin = ch_df.at[ch_log_idx, "cleared"]
                ch7_op = ch_df.at[ch_log_idx, "opened"]
                ch7_st = ch_df.at[ch_log_idx, "begined"]
                ch7_wait = ch_df.at[ch_log_idx, "days_begin"]
            else:
                print("ERROR" + ch_df.at[ch_log_idx, "chapter"])

            ch_log_idx = ch_log_idx + 1

        ch7_idx = ch7_df[
            ch7_df["account_id"] == unique_id[unq_idx]].index.values.tolist()# if there is id in ch7 data frame?
        if ch7_idx != [] and ch7_st != 0:  # if above is true frame & ch7_stis not 0
            ch7_fin = ch7_df.at[ch7_idx[0], "cleared"]
        else:
            ch7_fin = 0

        wait_list = [ch2_wait, ch3_wait, ch4_wait, ch5_wait, ch6_wait,ch7_wait]
        len_wait = np.count_nonzero(wait_list)
        if len_wait != 0:
            avr_wait = np.sum(wait_list) / len_wait

        tmp_ch = pd.Series(data=[unique_id[unq_idx], ch1_fin, ch2_op,ch2_st, ch2_fin, ch3_op, ch3_st, ch3_fin,
                                 ch4_op, ch4_st, ch4_fin, ch5_op, ch5_st,ch5_fin, ch6_op, ch6_st, ch6_fin, ch7_op,
                                 ch7_st, ch7_fin, ch2_wait, ch3_wait, ch4_wait,ch5_wait, ch6_wait, ch7_wait, avr_wait],
                           index=ch_agg_df.columns)
        ch_agg_df = ch_agg_df.append(tmp_ch, ignore_index=True)

    ch_agg_df = ch_agg_df.fillna(0)
    ch_agg_df.to_csv("/Users/maikiguchi/Desktop/cp2/5_aggregated_chapter.csv",index=False)
    print("***** FIN creating chapter aggregated data *****")
    return ch_agg_df


# calculate aggregated values from player_log data
def create_player_aggregated():
    print("***** BEGIN creating and calculating aggregated values *****")
    unique_id_df = pd.DataFrame(unique_id, columns=["id"])
    fmt = "%Y-%m-%d %H:%M:%S"
    agg_df = pd.DataFrame(columns=["id", "total_playtime(min)","total_login", "total_inactive(min)",
                              "average_playtime(min)", "average_inactive(min)", "first_login",
                              "last_login", "entire_period(days)","churn_status",
                              "ch1_playtime(min)", "ch2_playtime(min)","ch3_playtime(min)", "ch4_playtime(min)",
                              "ch5_playtime(min)", "ch6_playtime(min)","ch7_playtime(min)"])

    one_hour = 60 * 60  # sec*min
    len_log = len(pl_log_df)
    log_idx = unq_idx = 0

    while unq_idx < len(unique_id):
        t_play_sec = t_inactive = avr_inactive = 0
        ch1_play = ch2_play = ch3_play = ch4_play = ch5_play = ch6_play = ch7_play = 0
        t_login = 1
        churn = False
        first = pl_log_df.at[log_idx, "timestamp"]
        unq_acc_id = "".join(unique_id[unq_idx])
        inactive_list = []
        ch_idx = ch_agg_df[
            ch_agg_df["id"] == "".join(unique_id[unq_idx])].index.values.tolist()  # if the id is inch_agg_df dataframe
        print(unq_idx)

        while (log_idx < len_log - 1) and (unq_acc_id ==pl_log_df.at[log_idx, "id"]) and (
                pl_log_df.at[log_idx, "id"] == pl_log_df.at[log_idx + 1,"id"]):
            t1 = datetime.strptime(pl_log_df.at[log_idx, "timestamp"], fmt)
            t2 = datetime.strptime(pl_log_df.at[log_idx + 1, "timestamp"],fmt)
            td_sec = (t2 - t1).total_seconds()

            if td_sec > one_hour:  # stopped playing, count as inactive time
                t_login = t_login + 1
                t_inactive = t_inactive + td_sec
                if td_sec != 0:
                    inactive_list.append(td_sec)
            elif td_sec <= one_hour:  # keep playing, count as playtime
                t_play_sec = t_play_sec + td_sec
                if ch_idx != []:  # the id is in ch_agg_df, count playtime perchapter
                    if (ch_agg_df.at[ch_idx[0], "ch7_start"] != "0") and (
                            datetime.strptime(ch_agg_df.at[ch_idx[0], "ch7_start"], fmt) <= t1):
                        ch7_play = ch7_play + td_sec
                    elif (ch_agg_df.at[ch_idx[0], "ch6_start"] != "0") and (
                            datetime.strptime(ch_agg_df.at[ch_idx[0], "ch6_start"], fmt) <= t1):
                        ch6_play = ch6_play + td_sec
                    elif (ch_agg_df.at[ch_idx[0], "ch5_start"] != "0") and (
                            datetime.strptime(ch_agg_df.at[ch_idx[0], "ch5_start"], fmt) <= t1):
                        ch5_play = ch5_play + td_sec
                    elif (ch_agg_df.at[ch_idx[0], "ch4_start"] != "0") and (
                            datetime.strptime(ch_agg_df.at[ch_idx[0], "ch4_start"], fmt) <= t1):
                        ch4_play = ch4_play + td_sec
                    elif (ch_agg_df.at[ch_idx[0], "ch3_start"] != "0") and (
                            datetime.strptime(ch_agg_df.at[ch_idx[0],
"ch3_start"], fmt) <= t1):
                        ch3_play = ch3_play + td_sec
```

```
                    elif (ch_agg_df.at[ch_idx[0], "ch2_start"] != "0") and (
                            datetime.strptime(ch_agg_df.at[ch_idx[0],"ch2_start"], fmt) <= t1):
                        ch2_play = ch2_play + td_sec
                    elif datetime.strptime(first, fmt) <= t1:
                        ch1_play = ch1_play + td_sec

            log_idx = log_idx + 1

        if log_idx < (len_log - 1):  # if it's not the last object in a list
            last = pl_log_df.at[log_idx, "timestamp"]
            avr_play = round((t_play_sec / 60 / t_login), 2)
            current = datetime.strptime("2020-01-28 0:00:00", fmt)  # thenext date of the data collected
            recency = (current - datetime.strptime(last,fmt)).total_seconds()

            if (pl_log_df.at[log_idx, "id"] != pl_log_df.at[log_idx + 1,"id"]):
                log_idx = log_idx + 1

            recency = round((recency / 60), 2)
            try:  # calcurate standard deviation of inactive time
                std_inactive = round(statistics.stdev(inactive_list) / 60, 2)
            except statistics.StatisticsError:
                std_inactive = 0

            if ch_idx != []:  # the id is in ch_agg_df
                # ch7 is not finished and total login is not one time, add theinactive period till now
                if ch_agg_df.at[ch_idx[0], "ch7_fin"] == "0" and t_login != 1:
                    t_inactive = (t_inactive + recency)
                    avr_inactive = round((t_inactive / 60 / t_login),
                                         2)  # add the time between now and thelast play as inactive
                    personal_cutoff = avr_inactive + (2 * std_inactive)
                    if recency > personal_cutoff:  # determine if the player ischurned or not
                        churn = True
                else:  # finished playing, so not churned
                    avr_inactive = round((t_inactive / 60 / t_login),
                                         2)  # add the time between now and thelast play as inactive
                    churn = False

            diff = (datetime.strptime(last, fmt) - datetime.strptime(first,
                                                                       fmt)).days  # difference between first and last login
            tmp_ser = pd.Series(
                data=[unq_acc_id, round((t_play_sec / 60), 2), t_login,round((t_inactive / 60), 2),
                      avr_play, avr_inactive, first, last, diff, churn,
                      round(ch1_play / 60, 2), round(ch2_play / 60, 2),round(ch3_play / 60, 2),
                      round(ch4_play / 60, 2),
                      round(ch5_play / 60, 2), round(ch6_play / 60, 2),round(ch7_play / 60, 2)],
                index=agg_df.columns)
            agg_df = agg_df.append(tmp_ser, ignore_index=True)

        unq_idx = unq_idx + 1

    agg_df.to_csv("/Users/maikiguchi/Desktop/cp2/6_aggregated_play.csv",index=False)
    print("***** FIN creating and calculating aggregated values
*****")


# removing test accounts from the data frames
```

```python
def remove_test_data():
    print("***** BEGIN removing test account *****")
    # import test account
    path_test = "/Users/maikiguchi/Desktop/cp2/4_test_accounts.csv"
    test_acc_df = import_log_files(path=path_test, header="YES")

    # remove test data from player log df
    print("... removing test accounts from player log data")
    path_pl = "/Users/maikiguchi/Desktop/cp2/1_combined_player_log.csv"
    pl_log_df = import_log_files(path=path_pl, header="YES")
    pl_log_df = pl_log_df[~pl_log_df.id.isin(test_acc_df.account_id.values)]
    pl_log_df = pl_log_df.sort_values(by=["id", "timestamp"],ascending=True)
    pl_log_df = pl_log_df.reset_index(drop=True)
    pl_log_df.to_csv("/Users/maikiguchi/Desktop/cp2/1_combined_player_log.csv",index=False)

    # remove test data from exp df
    print("... removing test accounts from exp, coins, replay data")
    path_exp = "/Users/maikiguchi/Desktop/cp2/2_aggregated_exp_rep_coins.csv"
    exp_df = import_log_files(path=path_exp, header="YES")
    exp_df = exp_df[~exp_df.id.isin(test_acc_df.account_id.values)]
    exp_df = exp_df.reset_index(drop=True)
    exp_df.to_csv("/Users/maikiguchi/Desktop/cp2/2_aggregated_exp_rep_coins.csv",index=False)

    # remove test data from ch_agg_df
    print("... removing test accounts from chapter agg data")
    path_ch = "/Users/maikiguchi/Desktop/cp2/5_aggregated_chapter.csv"
    ch_agg_df = import_log_files(path=path_ch, header="YES")
    ch_agg_df = ch_agg_df[~ch_agg_df.id.isin(test_acc_df.account_id.values)]
    ch_agg_df = ch_agg_df.reset_index(drop=True)
    ch_agg_df.to_csv("/Users/maikiguchi/Desktop/cp2/5_aggregated_chapter.csv",index=False)

    # remove test data from unique id
    unique_id = pl_log_df["id"].unique().tolist()
    unique_id.sort()
    unique_id_df = pd.DataFrame(unique_id, columns=["id"])
    unique_id = unique_id_df[~unique_id_df.id.isin(test_acc_df.account_id.values)].values.tolist()

    print("***** FIN removing test account *****")
    return ch_agg_df, pl_log_df, unique_id


# merging the dataframes into one for modeling
def merge_dfs():
    print("***** BEGIN merging all datasets *****")
    # import the created exp, coins, replay data
    path_exp = "/Users/maikiguchi/Desktop/cp2/2_aggregated_exp_rep_coins.csv"
    exp_agg_df = import_log_files(path=path_exp, header="YES")

    # import the created crm
    path_crm = "/Users/maikiguchi/Desktop/cp2/3_crm.csv"
    crm_df = import_log_files(path=path_crm, header="YES")
    crm_df = crm_df.rename(columns={"account_id": "id"})

    # import player aggregated data
    path_agg = "/Users/maikiguchi/Desktop/cp2/6_aggregated_play.csv"
    agg_df = import_log_files(path=path_agg,
header="YES")
```

```
    # import chapter aggregated data
    path_ch_agg = "/Users/maikiguchi/Desktop/cp2/5_aggregated_chapter.csv"
    ch_agg_df = import_log_files(path=path_ch_agg, header="YES")

    # merge agg and ch_agg
    agg_df = pd.merge(agg_df, exp_agg_df, on="id", how="left")
    agg_df = pd.merge(agg_df, ch_agg_df, on="id", how="left")
    agg_df = pd.merge(agg_df, crm_df, on="id", how="left")

    agg_df.to_csv("/Users/maikiguchi/Desktop/cp2/7_aggregated.csv",index=False)
    print("***** FIN merging all datasets *****")


# output data frame details such as statistical and graphical information
def show_plots(agg_df):
    # check all variables
    print(agg_df.describe(include="all"))  # statistical text informationfor each variable
    # check which variables have missing values
    print(agg_df.columns[agg_df.isnull().any()])

    # -- numeric variables
    # show box plots for each variable
    box_plot_columns = ["total_login", "total_playtime(min)", "total_inactive(min)", "average_playtime(min)",
                        "average_inactive(min)", "entire_period(days)",
                        "avr_ch_wait(days)", "exp", "coins", "replay", "age"]
    for column in box_plot_columns:
        agg_df.boxplot(showmeans=True, column=column)
        plt.show()

    agg_df.boxplot(showmeans=True,
                   column=["ch1_playtime(min)", "ch2_playtime(min)","ch3_playtime(min)", "ch4_playtime(min)",
                           "ch5_playtime(min)", "ch6_playtime(min)","ch7_playtime(min)"])
    plt.show()

    # -- categorical variables
    # show bar chart
    agg_df["gender"].value_counts().plot(kind="bar")
    plt.show()
    agg_df["prefecture"].value_counts().plot(kind="bar")
    plt.show()


# preprocess the dataset
def preprocess_data(agg_df):
    # -- Missing Value Imputation
    # drop system admin and trial players
    agg_df = agg_df.dropna(subset=["gender"])
    agg_df = agg_df.reset_index(drop=True)

    # [exp, coins, replay] imputation with 0 for the missing value in exp,coins, and replay
    agg_df["exp"] = agg_df["exp"].fillna(0)
    agg_df["coins"] = agg_df["coins"].fillna(0)
    agg_df["replay"] = agg_df["replay"].fillna(0)

    # [prefecture] imputation of the missing value in prefecture, thentransform (Label Encoding) to ordinal value
    agg_df["prefecture"] = agg_df["prefecture"].fillna("Tokyo")
    agg_df["prefecture"].value_counts().plot(kind="bar")  # check aftermodification
```

```
plt.show()
le_pre = preprocessing.LabelEncoder()
agg_df["prefecture"] = le_pre.fit_transform(agg_df["prefecture"])
```

```
    print(list(le_pre.classes_))

    # check if all missing values are imputed
    print(agg_df.columns[agg_df.isnull().any()])

    # -- Transformation (Label Encoding)
    # [gender] transform the gender string to binary value (Female = 0, Male = 1)
    le_gen = preprocessing.LabelEncoder()
    agg_df["gender"] = le_gen.fit_transform(agg_df["gender"])
    print(list(le_gen.classes_))

    # [churn_status] transform churn_status True/False to binary (False = 0,True = 1)
    le_churn = preprocessing.LabelEncoder()
    agg_df["churn_status"] = le_churn.fit_transform(agg_df["churn_status"])
    print(list(le_churn.classes_))

    # -- Replacement of highly impossible values
    # [age] -1 and over 100 years old are replaced with median 26
    print(agg_df["age"].median())  # check median
    agg_df.loc[agg_df["age"] >= 100, "age"] = 26  # non-realistic age
    agg_df.loc[agg_df["age"] <= 3, "age"] = 26  # majority of kids under 3cannot read in Japan, so it"s impossible to start playing
    print(agg_df["age"].describe(include="all"))  # check statistic valuesafter modification
    plt.show()

    # -- Standardization of variables with outliers
    # exclude No outlier variables and categorical variables
    drop_col = ["id", "total_inactive(min)", "entire_period(days)","churn_status", "gender", "age", "prefecture"]
    std_df = agg_df.drop(columns=drop_col)  # drop unnecessary columns
    names = std_df.columns
    scaled_df = preprocessing.StandardScaler().fit_transform(std_df)
    scaled_df = pd.DataFrame(scaled_df, columns=names)

    for name in names:
        agg_df[name] = scaled_df[name]

    agg_df.to_csv("/Users/maikiguchi/Desktop/cp2/8_aggregated_after_preprocess.csv", index=False)
    return agg_df

# preprocess the dataset for retention analysis
def preprocess_retention():
    path_agg = "/Users/maikiguchi/Desktop/cp2/7_aggregated.csv"
    agg_df = import_log_files(path=path_agg, header="YES")

    # -- Missing Value Imputation
    # drop system admin and trial players
    agg_df = agg_df.dropna(subset=["gender"])
    agg_df = agg_df.reset_index(drop=True)

    # [exp, coins, replay] imputation with 0 for the missing value in exp,coins, and replay
    agg_df["exp"] = agg_df["exp"].fillna(0)
    agg_df["coins"] = agg_df["coins"].fillna(0)
    agg_df["replay"] = agg_df["replay"].fillna(0)

    # [prefecture] imputation of the missing value in prefecture, thentransform (Label Encoding) to ordinal value
    agg_df["prefecture"] = agg_df["prefecture"].fillna("Tokyo")
    agg_df["prefecture"].value_counts().plot(kind="bar")  # check
aftermodification

    # -- Replacement of highly impossible values
    # [age] -1 and over 100 years old are replaced with median 26
    agg_df.loc[agg_df["age"] >= 100, "age"] = 26  # non-realistic age
    agg_df.loc[agg_df[
                   "age"] <= 4, "age"] = 26  # majority of kids under 4 cannotread in Japan, so it"s impossible to start playing


    agg_df.to_csv("/Users/maikiguchi/Desktop/cp2/8_aggregated_after_preprocess_ret_analysis.csv", index=False)
    return agg_df

# splitting data into specified percentage
def split_data(train, test, val, x_features, y_target, split_way):
    if split_way == "cross":
        train_ratio = train
        test_ratio = test
        validation_ratio = val
        x_train, x_test, y_train, y_test = train_test_split(x_features,y_target, test_size=(1 - train_ratio),
                                                            random_state=1)
        x_val, x_test, y_val, y_test = train_test_split(x_test, y_test,
                                                        test_size=(test_ratio / (test_ratio + validation_ratio)),
                                                        random_state=1)
        x_train = pd.concat([x_train, x_test])
        y_train = pd.concat([y_train, y_test])
        return x_train, x_val, y_train, y_val


# run decision tree cross_val
def DT_cross_val(x_train, y_train, tree_dpt, criteria, split):
    if tree_dpt == "":  # no hyperparameter setting. use default.
        dt = DecisionTreeClassifier(random_state=1).fit(x_train, y_train)  #train the decision tree classifier
        accuracy = cross_val_score(dt, x_train, y_train, cv=10,scoring='accuracy')
        roc_auc = cross_val_score(dt, x_train, y_train, cv=10,scoring='roc_auc')
        print("Accuracy of DT: ", accuracy.mean())
        print("ROC AUC of DT: ", roc_auc.mean())
    else:  # train with the specified hyper parameters
        dt = DecisionTreeClassifier(random_state=1, max_depth=tree_dpt,criterion=criteria,
                                    min_samples_split=split).fit(x_train,y_train)  # train the decision tree classifier
        accuracy = cross_val_score(dt, x_train, y_train, cv=10,scoring='accuracy')
        roc_auc = cross_val_score(dt, x_train, y_train, cv=10,scoring='roc_auc')
        print("Accuracy of DT: ", accuracy.mean())
        print("ROC AUC of DT: ", roc_auc.mean())


# run logistic regression cross_val
def LR_cross_val(x_train, y_train, c, penalty, solver):
    if c == "":  # no hyperparameter setting. use default.
        lr = LogisticRegression(random_state=1).fit(x_train, y_train)  #train the Logistic Regression classifier
        accuracy = cross_val_score(lr, x_train, y_train, cv=10,scoring='accuracy')
        roc_auc = cross_val_score(lr, x_train, y_train, cv=10,scoring='roc_auc')
        print("Accuracy of LR: ", accuracy.mean())
        print("ROC AUC of LR: ", roc_auc.mean())
    else:  # train with the specified hyper parameters
```

```
        lr = LogisticRegression(random_state=1, C=c, penalty=penalty,solver=solver).fit(x_train,y_train)
        accuracy = cross_val_score(lr, x_train, y_train, cv=10,scoring='accuracy')
        roc_auc = cross_val_score(lr, x_train, y_train, cv=10,scoring='roc_auc')
        print("Accuracy of LR: ", accuracy.mean())
        print("ROC AUC of LR: ", roc_auc.mean())


# run random forest cross_val
def RF_cross_val(x_train, y_train, n_est, tree_dpt):
    if n_est == "":  # no hyperparameter setting. use default.
        rf = RandomForestClassifier(random_state=1).fit(x_train, y_train)  #train the random forest classifier
        accuracy = cross_val_score(rf, x_train, y_train, cv=10,scoring='accuracy')
        roc_auc = cross_val_score(rf, x_train, y_train, cv=10,scoring='roc_auc')
        print("Accuracy of RF: ", accuracy.mean())
        print("ROC AUC of RF: ", roc_auc.mean())
    else:  # train with the specified hyper parameters
        rf = RandomForestClassifier(random_state=1, n_estimators=n_est,max_depth=tree_dpt).fit(x_train,y_train)
        accuracy = cross_val_score(rf, x_train, y_train, cv=10,scoring='accuracy')
        roc_auc = cross_val_score(rf, x_train, y_train, cv=10,scoring='roc_auc')
        print("Accuracy of RF: ", accuracy.mean())
        print("ROC AUC of RF: ", roc_auc.mean())


# find best hyperparameters for the DT model
def DT_find_best_param(x_train, y_train):
    # Find the best criterion ---------------------------
    dt = DecisionTreeClassifier(random_state=1, criterion="gini").fit(x_train,y_train)
    roc_auc = cross_val_score(dt, x_train, y_train, cv=10,scoring='roc_auc')
    print("Gini Criteria AUC with Validation: ", roc_auc.mean())

    dt = DecisionTreeClassifier(random_state=1, criterion="entropy").fit(x_train, y_train)
    roc_auc = cross_val_score(dt, x_train, y_train, cv=10,scoring='roc_auc')
    print("Entropy Criteria AUC with Validation: ", roc_auc.mean())

    # Find the best splitter ---------------------------
    dt = DecisionTreeClassifier(random_state=1, splitter="best").fit(x_train, y_train)
    roc_auc = cross_val_score(dt, x_train, y_train, cv=10,scoring='roc_auc')
    print("Best Splitter AUC with Validation: ", roc_auc.mean())

    dt = DecisionTreeClassifier(random_state=1, splitter="random").fit(x_train, y_train)
    roc_auc = cross_val_score(dt, x_train, y_train, cv=10,scoring='roc_auc')
    print("Random Splitter AUC with Validation: ", roc_auc.mean())

    # Find the best max_depths ---------------------------
    cross_val_results = []
    max_depths = np.linspace(1, 32, 32, endpoint=True)  # create evenlyspaced 32 values between 1 to 32 trees
    for max_depth in max_depths:
        dt = DecisionTreeClassifier(random_state=1, max_depth=max_depth).fit(x_train, y_train)
        roc_auc = cross_val_score(dt, x_train, y_train, cv=10,scoring='roc_auc')
        cross_val_results.append(roc_auc.mean())

    line1, = plt.plot(max_depths, cross_val_results, "b", label="CrossValidation AUC")
    plt.legend(handler_map={line1: HandlerLine2D(numpoints=2)})
    plt.title("Decision Tree")
    plt.ylabel("AUC score")
    plt.xlabel("Tree depth")
    plt.show()

    # Find the best min_samples_split ---------------------------
    cross_val_results = []
    # create evenly spaced 30 values between 1% to 50% of the minimum numberof samples required to split
    min_samples_splits = np.linspace(0.01, 0.5, 30, endpoint=True)
    for split in min_samples_splits:
        dt = DecisionTreeClassifier(random_state=1, min_samples_split=split).fit(x_train,
                                                                                  y_train)  # train the decision tree classifier
        roc_auc = cross_val_score(dt, x_train, y_train, cv=10,scoring='roc_auc')
        cross_val_results.append(roc_auc.mean())

    line1, = plt.plot(min_samples_splits, cross_val_results, "b",label="Cross Validation AUC")
    plt.legend(handler_map={line1: HandlerLine2D(numpoints=2)})
    plt.title("Decision Tree")
    plt.ylabel("AUC score")
    plt.xlabel("Minimum Sample of Splits")
    plt.show()

    # Find the best min_samples_leaf ---------------------------
    val_results = []
    cross_val_results = []
    # create evenly spaced 30 values between 1% to 50% of the minimum numberof samples leaf
    min_samples_leafs = np.linspace(0.01, 0.5, 30, endpoint=True)
    for leaf in min_samples_leafs:
        dt = DecisionTreeClassifier(random_state=1, min_samples_leaf=leaf).fit(x_train,
                                                                                y_train)  # train the random forest classifier
        roc_auc = cross_val_score(dt, x_train, y_train, cv=10,scoring='roc_auc')
        cross_val_results.append(roc_auc.mean())

    line1, = plt.plot(min_samples_leafs, cross_val_results, "b",label="Cross Validation AUC")
    plt.legend(handler_map={line1: HandlerLine2D(numpoints=2)})
    plt.title("Decision Tree")
    plt.ylabel("AUC score")
    plt.xlabel("Minimum Samples of Leaf")
    plt.show()


# find best hyperparameters for the LR model
def LR_find_best_param(x_train, y_train):
    # Find the best penalty ---------------------------
    lr = LogisticRegression(random_state=1, penalty="l1", solver="liblinear").fit(x_train, y_train)  # train the logistic regressionclassifier
    roc_auc = cross_val_score(lr, x_train, y_train, cv=10,scoring='roc_auc')
    print("Penalty L1 AUC : ", roc_auc.mean())

    lr = LogisticRegression(random_state=1, penalty="l2").fit(x_train,y_train)
    roc_auc = cross_val_score(lr, x_train, y_train, cv=10,scoring='roc_auc')
    print("Penalty L2 AUC : ", roc_auc.mean())

    lr = LogisticRegression(random_state=1, penalty="none").fit(x_train,y_train)
    roc_auc = cross_val_score(lr, x_train, y_train, cv=10,scoring='roc_auc')
    print("Penalty None AUC : ", roc_auc.mean())

    # Find the best C --------------------------------------------------------
    cross_val_results = []
    c_params = np.linspace(0.01, 1.5, 30, endpoint=True)  # 30 evenly spaced values between 0.01 to 1.5
    for c in c_params:
        lr = LogisticRegression(random_state=1, C=c, penalty="l1",solver="liblinear").fit(x_train,y_train)
        roc_auc = cross_val_score(lr, x_train, y_train, cv=10,scoring='roc_auc')
        cross_val_results.append(roc_auc.mean())

    line1, = plt.plot(c_params, cross_val_results, "b", label="CrossValidation AUC")
    plt.legend(handler_map={line1: HandlerLine2D(numpoints=2)})
    plt.title("Logistic Regression")
    plt.ylabel("AUC score")
    plt.xlabel("Number of C")
    plt.show()

# find best hyperparameters for the RF model
def RF_find_best_param(x_train, y_train):
    # Find the best n_estimators ----------------------------
    n_estimators = [1, 2, 4, 8, 16, 32, 64, 100, 150]
    cross_val_results = []
    for estimator in n_estimators:
        rf = RandomForestClassifier(random_state=1, n_estimators=estimator).fit(x_train,
                                                                           y_train)
        # train the random forest classifier
        roc_auc = cross_val_score(rf, x_train, y_train, cv=10,scoring='roc_auc')
        cross_val_results.append(roc_auc.mean())

    line1, = plt.plot(n_estimators, cross_val_results, "b", label="CrossValidation AUC")
    plt.legend(handler_map={line1: HandlerLine2D(numpoints=2)})
    plt.title("Random Forest")
    plt.ylabel("AUC score")
    plt.xlabel("n_estimators")
    plt.show()

    # Find the best max_depths ----------------------------
    cross_val_results = []
    max_depths = np.linspace(1, 32, 32, endpoint=True)
    for max_depth in max_depths:
        rf = RandomForestClassifier(random_state=1, max_depth=max_depth).fit(x_train, y_train)
        roc_auc = cross_val_score(rf, x_train, y_train, cv=10,scoring='roc_auc')
        cross_val_results.append(roc_auc.mean())

    line1, = plt.plot(max_depths, cross_val_results, "b",
 label="CrossValidation AUC")
    plt.legend(handler_map={line1: HandlerLine2D(numpoints=2)})
    plt.title("Random Forest")
    plt.ylabel("AUC score")
    plt.xlabel("Tree depth")
    plt.show()

    # Find the best min_samples_split ---------------------------
    cross_val_results = []
    # 1% to 30% of the minimum number of samples required to split
    min_samples_splits = np.linspace(0.01, 0.3, 10,endpoint=True)
    for split in min_samples_splits:
        rf = RandomForestClassifier(random_state=1, min_samples_split=split).fit(x_train,y_train)
        roc_auc = cross_val_score(rf, x_train, y_train, cv=10,scoring='roc_auc')
        cross_val_results.append(roc_auc.mean())

    line1, = plt.plot(min_samples_splits, cross_val_results, "b",label="Cross Validation AUC")
    plt.legend(handler_map={line1: HandlerLine2D(numpoints=2)})
    plt.title("Random Forest")
    plt.ylabel("AUC score")
    plt.xlabel("Minimum Sample of Splits")
    plt.show()

    # Find the best min_samples_leaf ---------------------------
    cross_val_results = []
    # 1% to 10% of the minimum number of samples at the leafs
    min_samples_leafs = np.linspace(0.01, 0.3, 5, endpoint=True)
    for leaf in min_samples_leafs:
        rf = RandomForestClassifier(random_state=1, min_samples_leaf=leaf).fit(x_train,y_train)
        roc_auc = cross_val_score(rf, x_train, y_train, cv=10,scoring='roc_auc')
        cross_val_results.append(roc_auc.mean())

    line1, = plt.plot(min_samples_leafs, cross_val_results, "b",label="Cross Validation AUC")
    plt.legend(handler_map={line1: HandlerLine2D(numpoints=2)})
    plt.title("Random Forest")
    plt.ylabel("AUC score")
    plt.xlabel("Minimum Samples of Leaf")
    plt.show()

# train a DT model and predict with test dataset
def DT_evaluation(x_train, x_test, y_train, y_test, tree_dpt, criteria,split):
    print("Decision Tree ==============================================")
    dt = DecisionTreeClassifier(random_state=1, max_depth=tree_dpt,criterion=criteria, min_samples_split=split).fit(
        x_train, y_train)  # train the decision tree classifier
    dt_pred_test = dt.predict(x_test)  # predict with the test dataset
    false_positive_rate, true_positive_rate, thresholds = roc_curve(y_test,dt_pred_test)
    roc_auc = auc(false_positive_rate, true_positive_rate)
    print("[Test] ROC AUC of DT: ", roc_auc)
    print("▼▼▼ [Test] Classification Report of DT ▼▼▼")
    print(classification_report(y_test, dt_pred_test, digits=4))
    disp = plot_confusion_matrix(dt, x_test, y_test, cmap=plt.cm.Blues,values_format="d",
                                 display_labels=["Non-Churner", "Churner"])
                                 disp.ax_.set_title("DT - Confusion Matrix")
    plt.show()

# train a LR model and predict with test dataset
def LR_evaluation(x_train, x_test, y_train, y_test, c, penalty, solver):
    print("Logistic Regression ==============================================")
    lr = LogisticRegression(random_state=1, C=c,
penalty=penalty,solver=solver).fit(x_train,
y_train)  # train the Logistic Regression classifier
    lr_pred_test = lr.predict(x_test)  # predict with the test dataset
    false_positive_rate, true_positive_rate, thresholds = roc_curve(y_test,lr_pred_test)
    roc_auc = auc(false_positive_rate, true_positive_rate)
    print("[Test] ROC AUC of LR: ", roc_auc)
    print("▼▼▼ [Test] Classification Report of LR ▼▼▼")
    print(classification_report(y_test, lr_pred_test, digits=4))
    disp = plot_confusion_matrix(lr, x_test, y_test, cmap=plt.cm.Blues,values_format="d",
                                 display_labels=["Non-Churner", "Churner"])
    disp.ax_.set_title("LR - Confusion Matrix")
    plt.show()

# train a RF model and predict with test dataset
def RF_evaluation(x_train, x_test, y_train, y_test, n_est, tree_dpt):
    print("Random Forest ==============================================")
    rf = RandomForestClassifier(random_state=1, n_estimators=n_est,max_depth=tree_dpt).fit(x_train,
                                y_train)  # train the random forest classifier
    rf_pred_test = rf.predict(x_test)  # predict with the test dataset
    false_positive_rate, true_positive_rate, thresholds = roc_curve(y_test,rf_pred_test)
    roc_auc = auc(false_positive_rate, true_positive_rate)
    print("[Test] ROC AUC of RF: ", roc_auc)
    print("▼▼▼ [Test] Classification Report of RF ▼▼▼")
    print(classification_report(y_test, rf_pred_test, digits=4))
    disp = plot_confusion_matrix(rf, x_test, y_test, cmap=plt.cm.Blues,values_format="d",
                                 display_labels=["Non-Churner", "Churner"])
    disp.ax_.set_title("RF - Confusion Matrix")
    plt.show()


# ============================================================================================================
# Main -- Preprocessing ============================================================================
# set display number of columns
pd.set_option("display.max_columns", 50)

# -- import and combine account log and lesson log
create_player_log()

# -- create aggregated log for exp, coins, and number of replay time
create_exp_rep_aggregated()

# -- import and transform CRM log and also get test user account list
test_acc_df = transform_crm_log()

# -- import the created crm data
path_crm = "/Users/maikiguchi/Desktop/cp2/3_crm.csv"
crm_df = import_log_files(path=path_crm, header="Have")
crm_df = crm_df.reset_index(drop=True)

# -- import the created player log data
path_plog = "/Users/maikiguchi/Desktop/cp2/1_combined_player_log.csv"
pl_log_df = import_log_files(path=path_plog, header="Have")

# -- check unique id and store in a list
```

```
unique_id = pl_log_df["id"].unique().tolist()
unique_id.sort()

# -- create chapter aggregated log
ch_agg_df = create_chapter_aggregated(unique_id)

# -- remove data of test accounts (in test_acc_df) from
   ch_agg_df,pl_log_df, unique_id, and exp_df
```

```
ch_agg_df, pl_log_df, unique_id = remove_test_data()

# -- calculate aggregated information per players and store them in csv
create_player_aggregated()

# -- merge all aggregated data into one
merge_dfs()

# -- Exploratory Data Analysis (EDA) and preprocess ------------------------------------------------------------------------------
# import the merged df
path_agg = "/Users/maikiguchi/Desktop/cp2/7_aggregated.csv"
agg_df = import_log_files(path=path_agg, header="YES")
agg_df = agg_df[["id", "total_login", "total_playtime(min)","total_inactive(min)", "average_playtime(min)",
                 "average_inactive(min)", "entire_period(days)","churn_status",
                 "ch1_playtime(min)", "ch2_playtime(min)","ch3_playtime(min)", "ch4_playtime(min)",
                 "ch5_playtime(min)",
                 "ch6_playtime(min)", "ch7_playtime(min)","avr_ch_wait(days)",
                 "exp", "coins", "replay", "gender", "age", "prefecture"]]

# -- EDA by checking statistical information
show_plots(agg_df)  # show stats info, box_plots, and bar charts
agg_df = preprocess_data(agg_df)  # do preprocess
show_plots(agg_df)  # check after preprocess

path_agg = "/Users/maikiguchi/Desktop/cp2/8_aggregated_after_preprocess.csv"
agg_df = import_log_files(path=path_agg, header="YES")
feature_cols = ["total_login", "total_playtime(min)", "total_inactive(min)", "average_playtime(min)",
                "average_inactive(min)", "entire_period(days)",
                "ch1_playtime(min)", "ch2_playtime(min)", "ch3_playtime(min)","ch4_playtime(min)", "ch5_playtime(min)",
                "ch6_playtime(min)", "ch7_playtime(min)", "avr_ch_wait(days)",
                "exp", "coins", "replay", "gender", "age", "prefecture"]

x_features = agg_df[feature_cols]  # features
y_target = agg_df["churn_status"] # target

# -- Feature selection by Principal Component Analysis (PCA)
pca = PCA(n_components="mle").fit(x_features)  # mle = select n based oninput
print("Explained Variance: %s" % pca.explained_variance_ratio_)

pc = pca.fit_transform(x_features)

print(pd.DataFrame(pca.components_, columns=x_features.columns,
                   index=['PC-1', 'PC-2', 'PC-3', 'PC-4', 'PC-5', 'PC-6', 'PC-7', 'PC-8', 'PC-9', 'PC-10', 'PC-11',
                          'PC-12', 'PC-13', 'PC-14', 'PC-15', 'PC-16', 'PC-17','PC-18', 'PC-19']))
x_features_pc = pd.DataFrame(data=pc,
                             columns=['PC-1', 'PC-2', 'PC-3', 'PC-4', 'PC-5', 'PC-6', 'PC-7', 'PC-8', 'PC-9', 'PC-10',
                                      'PC-11', 'PC-12', 'PC-13', 'PC-14', 'PC-15', 'PC-16', 'PC-17', 'PC-18', 'PC-19'])
```

```

# Main -- Modeling
===========================================================================
# -- Dataset and split ratio comparison by using 10 K-folds
crossvalidation
# original dataset
# 80% (60% training and 20% validation), 20% test
```

```python
x_train80, x_test20, y_train80, y_test20 = split_data(0.6, 0.2, 0.2,x_features, y_target, "cross")
# 85% (70% training, 15% validation), 15% test
x_train85, x_test15, y_train85, y_test15 = split_data(0.7, 0.15, 0.15,x_features, y_target, "cross")
# 90% (80% training and 10% validation), 10% test
x_train90, x_test10, y_train90, y_test10 = split_data(0.8, 0.1, 0.1, x_features, y_target, "cross")

# features selected by pca method
# 80% (60% training and 20% validation), 20% test
x_train80_pc, x_test20_pc, y_train80_pc, y_test20_pc = split_data(0.6, 0.2,0.2, x_features_pc, y_target, "cross")
# 85% (70% training, 15% validation), 15% test
x_train85_pc, x_test15_pc, y_train85_pc, y_test15_pc = split_data(0.7,0.15, 0.15, x_features_pc, y_target, "cross")
# 90% (80% training and 10% validation), 10% test
x_train90_pc, x_test10_pc, y_train90_pc, y_test10_pc = split_data(0.8, 0.1,0.1, x_features_pc, y_target, "cross")

# --Decision Tree -------------------------------------------------------------------------------
# different percentage with original features
print("Decision Tree ==============================================")
print("Original")
DT_cross_val(x_train80, y_train80, "", "", "")
DT_cross_val(x_train85, y_train85, "", "", "")
DT_cross_val(x_train90, y_train90, "", "", "")

# different percentage with PCA selected features
print("PCA")
DT_cross_val(x_train80_pc, y_train80_pc, "", "", "")
DT_cross_val(x_train85_pc, y_train85_pc, "", "", "")
DT_cross_val(x_train90_pc, y_train90_pc, "", "", "")

# --Logistic Regression -------------------------------------------------------------------------------
print("Logistic Regression ==============================================")
print("Original")
LR_cross_val(x_train80, y_train80, "", "", "")
LR_cross_val(x_train85, y_train85, "", "", "")
LR_cross_val(x_train90, y_train90, "", "", "")

# different percentage with PCA selected features
print("PCA")
LR_cross_val(x_train80_pc, y_train80_pc, "", "", "")
LR_cross_val(x_train85_pc, y_train85_pc, "", "", "")
LR_cross_val(x_train90_pc, y_train90_pc, "", "", "")

# --Random Forest -------------------------------------------------------------------------------
print("Random Forest ==============================================")
print("Original")
# different percentage with original features
RF_cross_val(x_train80, y_train80, "", "") #setting no param = default
RF_cross_val(x_train85, y_train85, "", "")
RF_cross_val(x_train90_pc, y_train90_pc, "", "")

# different percentage with PCA selected features
print("PCA")
RF_cross_val(x_train80_pc, y_train80_pc, "", "")
RF_cross_val(x_train85_pc, y_train85_pc, "", "")
RF_cross_val(x_train90_pc, y_train90_pc, "", "")

# ---- FIND the best parameters and create the best performance model ---------------------------------
# --Decision Tree -------------------------------------------------------------------------------
print("Decision Tree ==============================================")
```

```
# 90% (80% training and 10% validation), 10% test with PCA
DT_find_best_param(x_train90_pc, y_train90_pc)
DT_cross_val(x_train90_pc, y_train90_pc, 6, "entropy", 50)
```

```python

#  Logistic Regression
print("Logistic Regression ========================================")
# 90% (80% training and 10% validation), 10% test with PCA
LR_find_best_param(x_train90_pc, y_train90_pc)
LR_cross_val(x_train90_pc, y_train90_pc, 0.9, "l1", "liblinear")

# Random Forest
print("Random Forest ==============================================")
# 90% (80% training and 10% validation), 10% test with PCA
RF_find_best_param(x_train90_pc, y_train90_pc)
RF_cross_val(x_train90_pc, y_train90_pc, 100, 10)

# ---- Final Evaluation by using the best hyper parameters with the test
dataset
DT_evaluation(x_train90_pc, x_test10_pc, y_train90_pc, y_test10_pc, 6,
"entropy", 50)
LR_evaluation(x_train90_pc, x_test10_pc, y_train90_pc, y_test10_pc, 0.9,
"l1", "liblinear")
RF_evaluation(x_train90_pc, x_test10_pc, y_train90_pc, y_test10_pc, 100,
10)

preprocess_retention()
```

Figure 44. Entire Python Code

Appendix B - Figures of Results

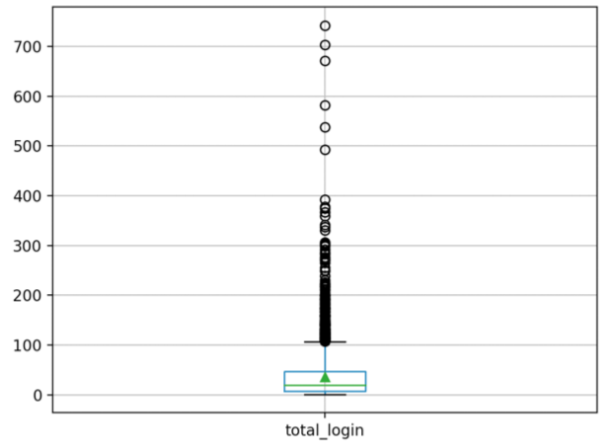


Figure 45. Box Plot of total_login

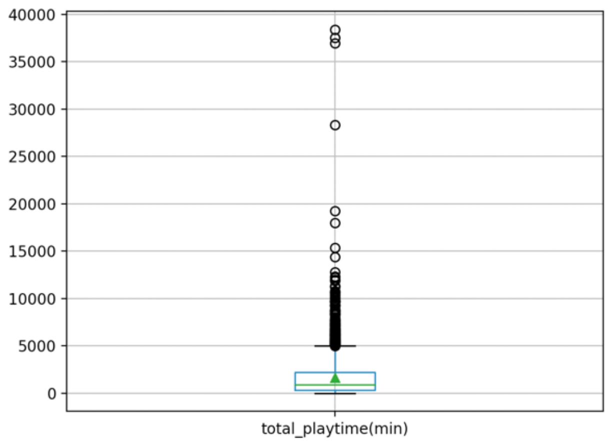


Figure 46. Box Plot of total_playtime

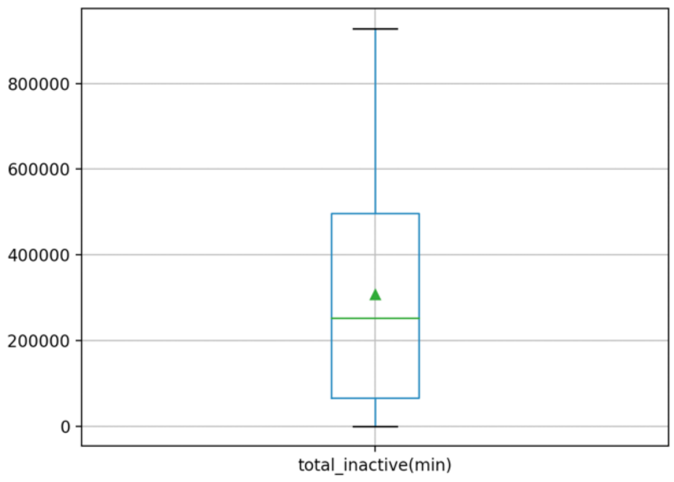


Figure 47. Box Plot of total_inactive

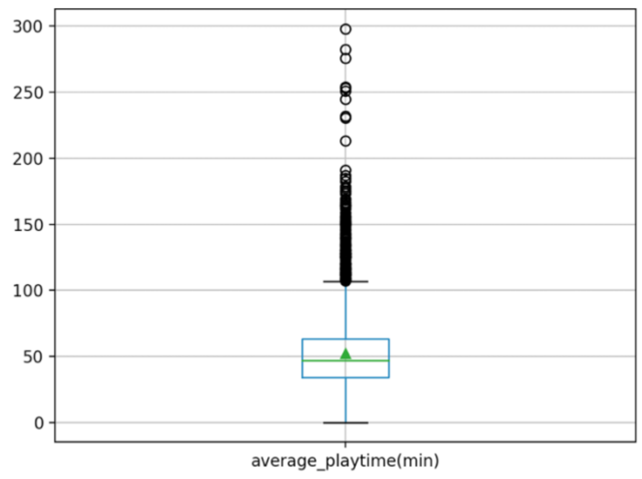


Figure 48. Box Plot of average_playtime

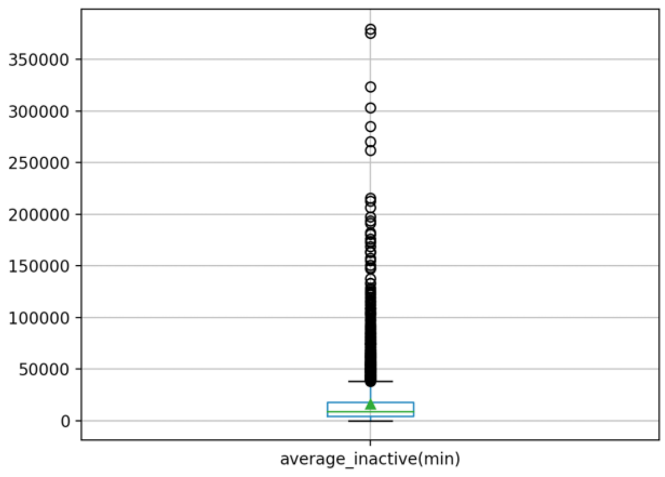


Figure 49. Box Plot of average_inactive

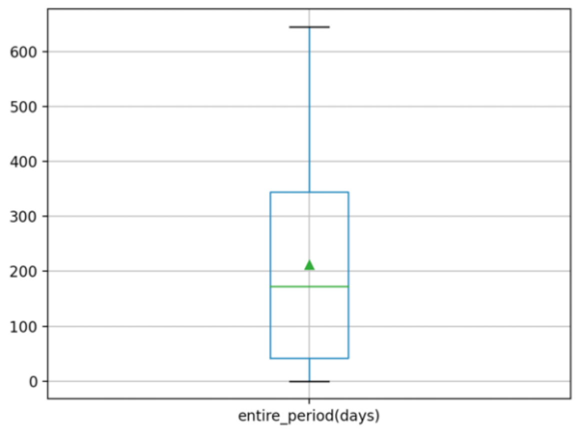


Figure 50. Box Plot of entiere_period

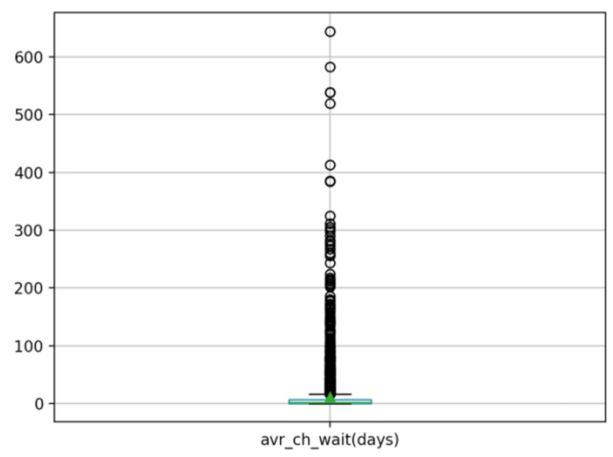


Figure 51. Box Plot of avr_ch_wait

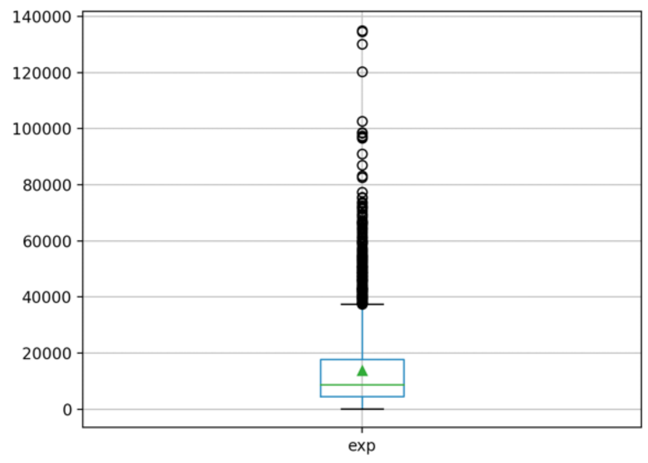


Figure 52. Box Plot of exp

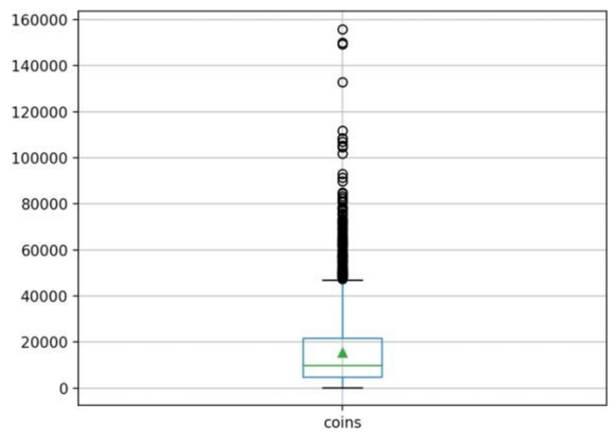


Figure 53. Box Plot of coins

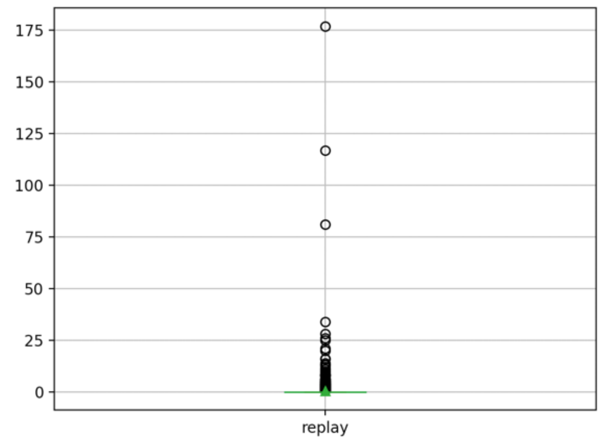


Figure 54. Box Plot of replay

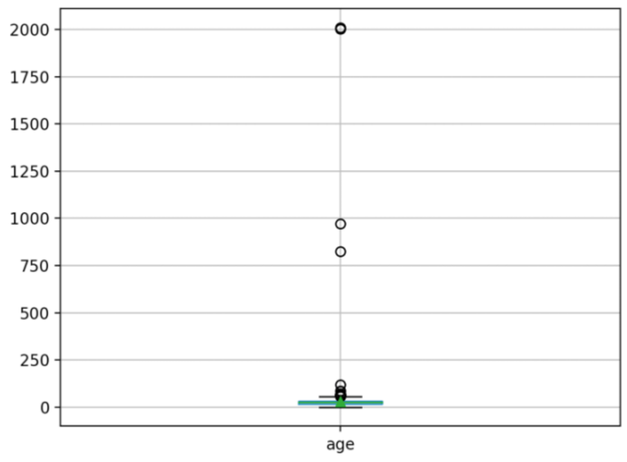


Figure 55. Box Plot of age

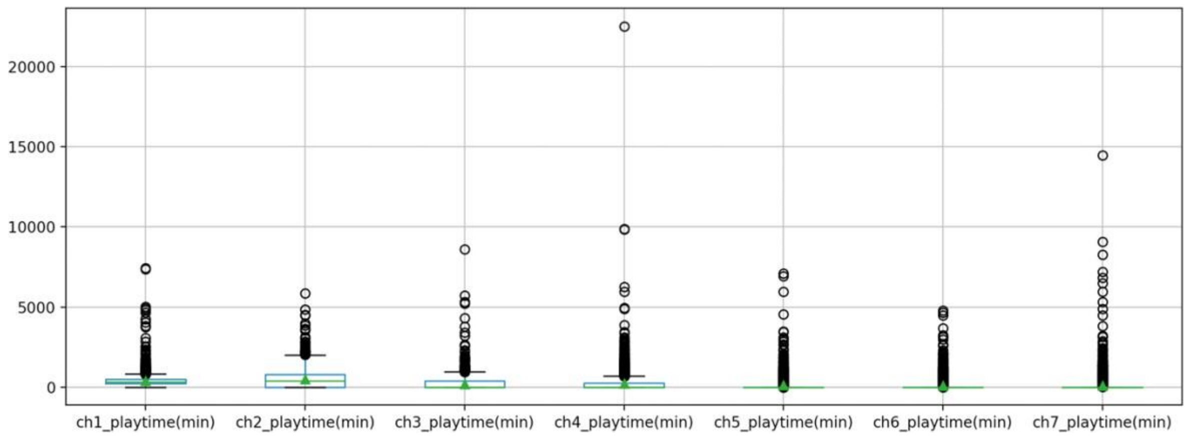


Figure 56. Box Plot of Playtime of Each Chapter

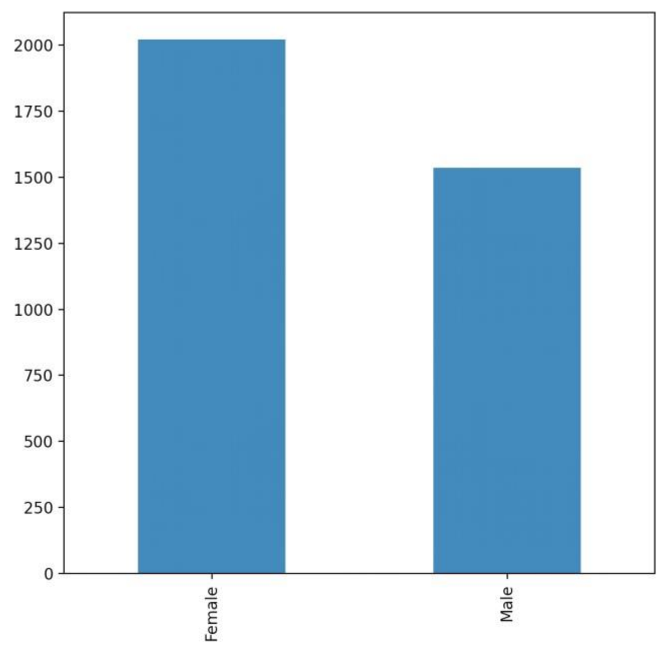


Figure 57. Bar Chart of gender

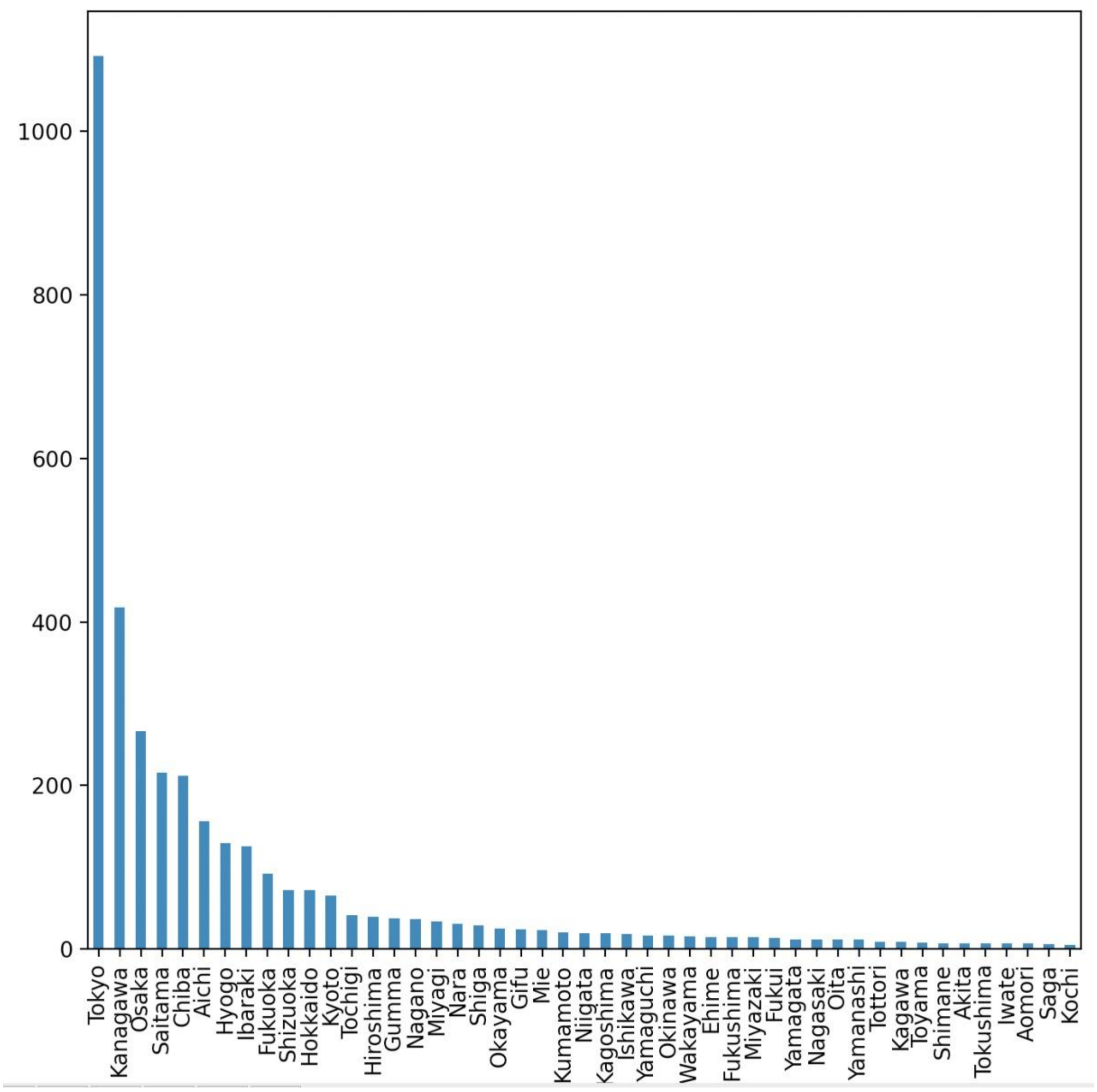


Figure 58. Bar Chart of prefecture

Appendix C - Tables of Results

Table 33 - 1. The Statistical Output of All Variables (1/2)

	total_playtime(min)	total_login	total_inactive(min)	average_playtime(min)	average_inactive(min)	first_login	last_login	entire_period(days)	churatus
count	3701	3701	3701	3701	3701	3701	3701	3701	3701
unique	NaN	NaN	NaN	NaN	NaN	3539	3612	NaN	2
top	NaN	NaN	NaN	NaN	NaN	2019/12/18 16:29	2020/01/10 11:28	NaN	TRU
freq	NaN	NaN	NaN	NaN	NaN	26	9	NaN	2101
mean	1640.14766	36.01378	307397.912	52.085955	16507.7144	NaN	NaN	211.50689	NaN
std	2207.53535	50.265077	255321.919	28.872217	25549.8298	NaN	NaN	178.952966	NaN
min	0	1	0	0	0	NaN	NaN	0	NaN
25%	339.55	700%	66484.59	33.93	4375.9	NaN	NaN	42	NaN
50%	885.82	2000%	251926.17	46.74	9074.17	NaN	NaN	173	NaN
75%	2213.1	4700%	496751.22	63.15	18022.08	NaN	NaN	344	NaN
max	38390.98	742	928384.54	298	379752.63	NaN	NaN	645	NaN

Table 33 - 2. The Statistical Output of All Variables (2/2)

	ch4_playtime(min)	**ch5_playtime(min)**	**ch6_playtime(min)**	**ch7_playtime(min)**	**exp**	**coins**	**replay**	**avr_ch_wait(days)**	**gende**
count	3701	3701	3701	3701	3521	3521	3521	3701	3557
unique	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	2
top	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	Femal
freq	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	20220
mean	234.508522	117.442086	99.966485	93.080057	13753.3684	15485.9046	0.370917	11.431015	NaN
std	642.299392	360.914759	331.141006	492.924226	14314.3139	15841.2305	4.09627	37.920647	NaN
min	0	0	0	0	150	200	0	0	NaN
25%	0	0	0	0	4570	4700	0	0	NaN
50%	0	0	0	0	8670	9920	0	1.4785	NaN
75%	274.37	0	0	0	17700	21760	0	6.457125	NaN
max	22511.83	7105.2	4787.72	14452.2	135010	155850	177	644.5389	NaN

Table 34 - 1. Explained Variables of Principal Components (1/2)

	total_login	total_playtime(min)	total_inactive(min)	average_playtime(min)	average_inactive(min)	entire_period(days)	ch1_playtime(min)	ch2_playtime(min)	ch3_playtime(min)	ch4_playtime(min)
PC-1	2.00E-06	1.00E-06	1.00E+00	-6.69E-07	1.00E-06	0.0007	4.46E-07	0.000002	1.00E-06	0.000001
PC-2	-5.00E-04	0.00023	-7.88E-06	1.23E-03	-0.002	0.0099	2.32E-03	0.002199	7.00E-05	-0.001188
PC-3	0.0102	0.01323	-2.06E-06	5.76E-03	-0.001	0.0061	5.67E-03	0.012157	0.0116	0.007957
PC-4	0.1663	0.18799	-6.05E-04	1.75E-02	-0.096	0.8606	8.42E-02	0.116752	0.1384	0.147524
PC-5	0.2719	0.30871	3.52E-04	3.62E-02	-0.129	-0.506	7.84E-02	0.161545	0.2256	0.281479
PC-6	-0.029	0.03238	-2.08E-05	1.11E-01	0.4419	0.0295	1.32E-01	-0.382636	-0.242	0.168011
PC-7	0.1508	-0.03262	-7.21E-06	-7.75E-01	-0.159	0.011	-5.39E-02	-0.169805	-0.134	0.072236
PC-8	0.0694	0.0559	1.86E-05	7.53E-02	-0.246	-0.026	7.83E-01	0.17535	0.0091	0.024768
PC-9	0.1422	0.02436	7.13E-06	-4.70E-01	-0.128	-0.012	-9.13E-02	0.15487	0.1345	0.062961
PC-10	-0.069	-0.02935	-2.39E-05	1.63E-01	0.0029	0.0344	-4.78E-01	0.11993	0.2582	0.440438
PC-11	0.1205	0.08686	8.32E-07	-2.45E-01	0.7706	-0.004	2.10E-01	0.093908	0.4173	0.02843
PC-12	0.056	0.05772	7.58E-07	2.64E-02	0.098	-0.001	-1.33E-01	0.331262	0.1194	-0.211524
PC-13	-0.14	-0.0956	-3.09E-06	-4.76E-02	-0.119	0.005	1.27E-01	-0.655647	0.2889	0.161819
PC-14	-0.081	0.01833	2.38E-06	6.26E-02	-0.208	-0.002	-2.28E-02	-0.229848	0.6743	-0.39824
PC-15	-0.047	-0.01299	2.87E-06	5.21E-03	-0.034	-0.004	1.01E-02	0.106602	0.0718	-0.140912
PC-16	0.1724	0.04301	-1.46E-06	1.73E-02	-0.034	0.0022	2.07E-02	-0.059972	-0.038	0.202304
PC-17	0.4638	0.08592	8.15E-06	2.14E-01	-0.102	-0.011	-2.69E-02	-0.170329	0.0664	0.377125
PC-18	0.7444	-0.25832	-5.03E-06	1.41E-01	0.0578	0.0068	-1.21E-01	-0.16183	-0.076	-0.409904

PC-19	-0.007	0.00207	2.78E-07	-3.74E-04	0.0051	-5.00E-04	-5.72E-03	0.008591	0.003	-0.001526

Table 34 - 2. Explained Variables of Principal Components (2/2)

	ch5_playtime(min)	**ch6_playtime(min)**	**ch7_playtime(min)**	**avr_ch_wait(days)**	**exp**	**coins**	**replay**	**gender**	**age**	**prefecture**
PC-1	1.00E-06	8.98E-07	5.97E-07	7.69E-07	2.00E-06	2.00E-06	3.49E-07	-3.25E-08	-2.00E-06	-6.73E-07
PC-2	-0.001	4.60E-04	-1.01E-03	-9.65E-04	0.001	0.0012	-1.27E-03	1.20E-03	-0.104	-9.94E-01
PC-3	0.0091	1.01E-02	8.35E-03	1.47E-03	0.0128	0.012	-1.08E-03	-2.74E-03	0.9939	-1.04E-01
PC-4	0.1502	1.50E-01	1.33E-01	-3.16E-02	0.1635	0.1639	8.41E-02	7.86E-03	-0.021	1.14E-02
PC-5	0.2694	2.48E-01	2.29E-01	-7.15E-02	0.2839	0.2834	2.03E-01	-2.73E-04	-0.025	-1.88E-03
PC-6	0.1533	6.46E-02	2.74E-01	2.91E-01	-0.157	-0.149	5.52E-01	-1.61E-02	0.0043	-3.57E-03
PC-7	0.0241	-1.11E-02	8.13E-02	-4.48E-01	-0.139	-0.129	2.24E-01	4.50E-02	0.0098	-2.40E-03
PC-8	-0.171	-3.29E-01	-2.21E-01	-2.78E-02	-0.111	-0.1	2.66E-01	1.07E-02	0.001	2.21E-03
PC-9	-0.065	-7.69E-02	-8.02E-02	8.19E-01	-0.007	-0.01	2.76E-03	1.06E-02	-0.002	-9.19E-04
PC-10	-0.146	-3.06E-01	-3.69E-01	-1.23E-01	-0.041	-0.04	4.40E-01	-2.61E-02	0.0037	-1.34E-03
PC-11	-0.056	-8.19E-03	-2.00E-01	-1.33E-01	0.0109	0.0045	-1.87E-01	1.46E-02	-0.005	-2.80E-04
PC-12	-0.392	-3.56E-01	7.06E-01	-5.77E-02	-0.041	-0.047	9.85E-02	-3.76E-02	-0.002	2.31E-05
PC-13	-0.402	-1.43E-01	1.26E-01	3.37E-02	0.3078	0.3135	-8.56E-02	-4.20E-03	0.0016	-1.18E-04
PC-14	0.3272	4.34E-02	4.93E-02	7.15E-03	-0.269	-0.268	1.36E-01	-7.22E-02	0.0009	-8.36E-04
PC-15	-0.587	7.19E-01	-1.07E-01	-4.31E-03	-0.063	-0.058	2.69E-01	-8.61E-02	0.0002	1.04E-03
PC-16	-0.021	-1.09E-02	2.98E-02	-1.19E-02	-0.113	-0.12	-1.88E-01	-9.26E-01	-0.003	-1.11E-03
PC-17	-0.183	9.63E-02	1.09E-01	7.23E-03	-0.358	-0.347	-3.43E-01	3.53E-01	0.0003	-5.30E-04
PC-18	-0.036	-1.08E-01	-1.91E-01	8.90E-04	0.1577	0.1674	1.94E-01	-3.58E-02	0.0009	-3.28E-04
PC-19	0.0035	-9.50E-04	1.98E-03	2.27E-03	-0.704	0.7096	-7.52E-03	-7.05E-03	0.0004	1.07E-04