A Computational Approach to the Study of Player and Group Performance in Massively Multiplayer Online Games

A DISSERTATION
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Kyong Jin Shim

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
Doctor of Philosophy

Professor Jaideep Srivastava

December, 2011

Acknowledgements

It is with great pleasure and deep sense of gratitude that I thank all those who have helped and inspired me during my doctoral study at the University of Minnesota - Twin Cities.

First and foremost, I want to thank my advisor Professor Jaideep Srivastava for his help, support, and guidance throughout my research and studies. His energy and enthusiasm in everything he does continue to motivate me. I feel extremely fortunate to have him as not only my dissertation advisor but also as a lifelong teacher, mentor, and friend. Professor John Carlins, Professor Theodore Christ and Professor Yongdae Kim deserve thanks as my dissertation committee members. Their valuable feedback helped me look at the big picture and look for applications of my research work in a variety of other fields.

Many of my colleagues and friends from DMR Lab helped enrich my experience at the University. In particular, I would like to thank Nishith Pathak, Muhammad A. Ahmad, and Senthilkumar Krishnamoorthy for our fruitful collaboration on various projects. I would like to thank Colin DeLong and Nishith Pathak for our collaboration on the HaloFit project. I would also like to thank David Kuo-Wei Hsu for our fruitful research collaboration on a number of publications. I want to thank Prasanna Desikan for his constructive feedback on my research. Both David and Prasanna, as my lab seniors, have provided support and encouragement in many forms. In moments of doubt, they were the two strong pillars for me. Furthermore, I would like to thank our external collaborators Professor Dmitri Williams, Professor Noshir Contractor, and Professor Marshall Scott Poole for providing valuable advice to all graduate students, including myself, as part of Virtual World Observatory project.

My deepest gratitude goes to my family for their unconditional love and support throughout my life. This dissertation is simply impossible without them. I am deeply indebted to my parents, Dr. Y. S. Shim and Mrs. Y. S. Kwon, for tirelessly supporting me through all the life changing events, for their confidence and trust in me, and for encouraging and inspiring me to continue my journey in searching and finding my own path in life. I am deeply indebted to my elder sister Annie for being my safe haven since we moved to the USA sixteen years ago, for being the best sister anyone could ask for, and for being my best friend in the world. I thank my brother-in-law for his support and for providing valuable advice during difficult times. My baby niece Sophia has brought so much joy and happiness to my lonely, sometimes dry and stressful doctoral journey. All my stress, concerns, and worries melt away when I see her big baby smile. I would also like to thank all of my friends for their love and support. In particular, I am very grateful to my friends Alan and Lana for always standing by me through good and bad times. Last but not least, I want to thank my two lovely feline friends, USA and Barley, for being part of my life and for giving me so much love and support with their meows.

Dedication

To my family, friends, and colleagues who have mentored me along the way.

Abstract

The market for video games skyrocketed over the past decade. Massively Multiplayer Online Games (MMOGs) have become increasingly popular and have communities comprising over 47 million subscribers by the year 2008. With their increasing popularity, researchers are realizing that video games can be a means to fully observe an entire isolated universe. Each action is logged, and the level of granularity and completeness with which information is collected is unmatched by any real-life experimental setup. They serve as unprecedented tools to theorize and empirically model the social and behavioral dynamics of individuals, groups, and networks within large communities.

Virtual world applications usually have a thin-client architecture, practically all user actions are captured in the click-stream logged at the server. This dataset contains a comprehensive record of every user's in-network activities, accomplishments, interactions, economic status, etc. A brief record of the user's side information (i.e. profile data) is also stored. It is common for popular social networking and collaborative systems to have hundreds of thousands of users generating copious amounts of data based on the many different activities they are participating in at any given time. The data also has a temporal component which is often an integral part of the analysis and introduces further relationships that must be accounted for. Thus, while providing an exciting new tool for the social sciences, the virtual worlds also present a set of difficult and novel computational challenges.

In the gaming community today, there is a growing interest in understanding player behaviors both inside and outside the gaming space. Game companies are interested in finding out how their games are played, if they are being played as intended, how the different game mechanics are being played out and how the different game playing patterns lead to a high level of satisfaction and entertainment for customers. Retrospective analyses after the game launch on existing game features can reveal information on which features enhance player's gaming experience and to which demographic segments they especially appeal to. Features negatively correlated with gaming experience can be considered for removal while those positively correlated with gaming experience can be further enhanced. For new game features, prospective analyses before the game launch

can reveal information on which features might appeal to certain player population segments with a certain level of confidence and user-oriented testing can focus on these features for further validation.

This thesis work presents the first comprehensive quantitative analysis of an important aspect of MMOG game play, namely player and group performance. While there are many different game genres (i.e. action, shooter, action-adventure, adventure, role-playing, and simulation) and many dimensions comprising players' game-play experience, in certain game genres such as MMOGs, close connection has been reported between player enjoyment and completing challenges and mastering skills. A systematic study of individual game player characteristics, group composition and characteristics, social interactions amongst the group members, and game environments can reveal a great deal about what are the recipes for success in achieving various objectives in the game. Broadly, this thesis work seeks to develop 1) Player performance metrics and prediction models, 2) Player activity prediction model, 3) Player enjoyment prediction model, and 4) Group performance metrics and prediction models. Lastly, we contribute a single, generic framework for player and group behavior analysis that is applicable to other MMOGs with minimal configuration changes.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Background

The market for video games skyrocketed over the past decade. In the United States alone, the video game industry in 2009 generated almost $20 billion USD in sales [60]. Furthermore, according to [45], estimated 97% of the teenage population and 53% adult population are regular game players. Massively Multiplayer Online Games (MMOGs) have become increasingly popular and have communities comprising over 47 million subscribers by the year 2008. MMOGs are online spaces providing users with comprehensive virtual universes, each with its own unique context and mechanics. They range from the fantastical world of elves, dwarfs, and humans to space faring corporations and mirrors of our world. Large numbers of users interact and role-play via the in-game mechanics. With their increasing popularity, researchers are realizing that video games can be a means to fully observe an entire isolated universe. Each action is logged, and the level of granularity and completeness with which information is collected is unmatched by any real-life experimental setup. They serve as unprecedented tools to theorize and empirically model the social and behavioral dynamics of individual players, groups, and networks within large communities.

1.2 Game Logs

Virtual world applications usually have a thin-client architecture, practically all user actions are captured in the click-stream logged at the server. This data contains a comprehensive record of every user's in-network activities, accomplishments, interactions, economic status, etc. A brief record of the user's side information (i.e. profile data) is also stored. It is common for popular social networking and collaborative systems to have hundreds of thousands of users generating copious amounts of data based on the many different activities they are participating in at any given time. One key research challenge is developing analysis methods that can analyze relationships while scaling to terabytes of data. The data also has a temporal component which is often an integral part of the analysis and introduces further relationships that must be accounted for. Thus, while providing an exciting new tool for the social sciences, the virtual worlds also present a set of difficult and novel computational challenges.

1.3 Applications

1.3.1 Opportunities for Social Sciences and Military

The data collected provide an excellent means of studying human social behavior with respect to the complete context of the environment for a large population. They serve as unprecedented tools to theorize and empirically model the social and behavioral dynamics of individual players, groups, and networks within large communities. For instance, in recent years, educational research has found virtual environments to be a sound venue for studying learning, collaboration, social participation, literacy in online space, and learning trajectory at the individual level as well as at the group level. MMOGs in particular are of great interest to social scientists that study social interactions and social structures because of a variety of social systems present in those games. Interestingly, networks generated by in-game processes are closely analogous to the real world. For example, money transactions are similar in the game world and the real world [8]. In contrast, role-playing networks differ from any real-world networks because this is a unique in-game phenomenon. Yet another popular video game genre is First-Person-Shooter (FPS). This category of games involves gun and weapon-based combat fighting, and the game play is through the first perspective. While some game titles are restricted to single person modes (in which case, the player would play against bots or computer-controlled characters), others offer multiplayer gaming experience either exclusively or jointly with single person modes. FPS games are available on a variety of platforms: PC, Xbox, PlayStation, and Nintendo. FPS games are highly popular. As of December 2010, Call of Duty: Black Ops sold 13.7 million unites since its release in early November 9, 2010, making it the best selling video game ever in the U.S. [106]. And, Halo: Reach since its release in mid September 2010 sold over 7.8 million copies [107]. FPS games are of great interest to social scientists in a similar manner, but in particular, the U.S. military has found a great use of such games for recruiting soldiers and training soldiers in marksmanship and stealth [103, 104, 105]. The military also uses MMOGs for team coordination and other types of training [104]. MMOGs provide an immersive environment where any of a variety of training scenarios can be built in. Training sessions can be recorded and replayed/rerun to see how making different choices would lead to different outcomes. Understanding and explaining social behavior as well as individual player/soldier-level behavior can fundamentally alter the functioning of professional organizations, battlefield scenarios, and multi-player professional teams.

1.3.2 Opportunities for Commercial Vendors

The business climate today is very competitive as well as volatile that companies can no longer let their business decisions guided by intuition alone. Companies lacking an intelligent strategy for engaging and retaining their customers will not succeed. Many E-commerce vendors in general are heavily leveraging insights derived from the data collected about how their consumers browse, shop, and interact online. For instance, consumer click stream data can provide information useful for the company to understand how certain ways of arranging items and visualizing product information on their website successfully leads to purchasing behavior. The video games market is no exception when it comes to customer engagement and retention. The term 'game analytics' is often heard in many tech forums in recent years. There is a growing interest in the gaming community in understanding player behaviors both inside and outside the gaming space. Game companies are interested in finding out how their games are played, if they are being played as intended, how the different game mechanics are being played out and how the different game playing patterns lead to a high level of satisfaction and entertainment for customers. Retrospective analyses after the game launch on existing game features can reveal information on which features enhance player's gaming experience and to which demographic segments they especially appeal to. Features negatively correlated with gaming experience can be considered for removal while those positively correlated with gaming experience can be further enhanced. For new game features, prospective analyses before the game launch can reveal information on which features might appeal to certain player population segments with a certain level of confidence and user-oriented testing can focus on these features for further validation.

1.3.3 Opportunities for Computer Science

As much as game logs bring exciting new opportunities for research and commercial purposes with its detailed and temporal tracking of a variety of activities going on in the game space, it poses new challenges for computer scientists with its massive amount of data containing complex networked relationships spanning multiple entities. Our research uses knowledge discovery as a core mechanism to analyze MMOG logs and build models of social interactions in games and understand how users' relationships are affected by the variety of environmental factors present in each world. MMOGs provide mechanisms to foster social activities among users, letting them form groups, guilds, corporations, and so on and tackle collaboration-oriented tasks, such as raiding a dungeon for gold. Data collected from these mechanisms provide an excellent means of studying human social behavior with respect to the complete context of the environment. Such research is infeasible for real-life activities because it is impossible to track and record complete information on a large population. Knowledge discovery is a key computational approach to realizing this promise, and in the process, it creates opportunities to push the frontiers of knowledge discovery itself.

1.4 This Thesis

This thesis work presents the first comprehensive quantitative analysis of an important aspect of MMOG game play, namely player and group performance. While there are many different game genres (i.e. action, shooter, action-adventure, adventure, role-playing, and simulation) and many dimensions comprising players' game-play experience, in certain game genres such as MMOGs, close connection has been reported between player enjoyment and completing challenges and mastering skills. A systematic study of individual game player characteristics, group composition and characteristics, social interactions amongst the group members, and game environments can reveal a great deal about what are the recipes for success in achieving various objectives in the game. Broadly, this thesis work seeks to develop 1) Player performance metrics and prediction models, 2) Player activity prediction model, 3) Player enjoyment prediction model, and 4) Group performance metrics and prediction models. Lastly, we contribute a single, generic framework for player and group behavior analysis that is applicable to other MMOGs with minimal configuration changes.

1.5 Research Contributions

This thesis work is the first comprehensive analysis of an important aspect of MMOG game play, namely player and group performance. In summary, the research contributions of this thesis are as follows:

- We propose two novel individual player performance prediction models, namely Neighbor-Weight-1 and Neighbor-Weight-2. In the former model, the future performance prediction for each game player is solely dependent upon their past performance. In the latter model, the feature representation includes more behavioral attributes of high granularity. The future performance prediction for each game player is dependent upon their past activities, some of which are solo activities while others are group activities such as apprenticing and mentoring. While the existing player monitoring tool offers a retrospective look at each player's past activities (in aggregates), the key contribution we make in this research work is the addition of prospective and predictive analytics component to the tool.

- We propose a novel player activity prediction model, namely Player-Activity-Pred. While our player performance prediction methods use aggregated behavioral data (i.e. total number of monster kills a player performed over a given time period, etc.), this method takes a deeper dive and models each player's activities as a sequence of events. Next, it uses sequence alignment algorithms from Bioinformatics domain (where the algorithms were first developed to compare biological sequences). We formulate this as a binary classification problem and show that the proposed model can fairly accurately predict player's future activity (i.e. whether or not they will continue to play significant in-game activities in the future) based on past behavioral sequences. The key contribution we make in this research work is the addition of high granularity (sequence of activity events) player behavior analysis and prediction model to the existing player monitoring tool.

- We propose two novel player enjoyment prediction models, namely Player-Fun-Pred and Player-Quit-Pred. Player enjoyment is extremely important for commercial game development. While many games today provide in-game tutorials to help newcomers ramp up quickly in the early stage of the game as well as in-game assistance throughout the game to help identify tasks to perform to gain rewards, little is understood about the relationship between in-game player performance and player enjoyment in MMOGs. We analyze the impact of game difficulty and player performance on player enjoyment. Further, we investigate how player motivation(s), combined with player performance, play a critical role in determining player enjoyment. The key contribution we make in this research work is the analysis of two important dimensions of game play experience in MMOGs, namely motivation and enjoyment, and the analysis of how player performance combined with player motivation can fairly accurately predict player enjoyment.

- We propose a comprehensive performance management tool for measuring and reporting operational activities of groups. While there are many game-related forums and blogs where game players and group members share tips and anecdotal stories about group plays, the MMOG gaming community is lacking a systematic, data-driven group performance management tool, which this thesis work proposes. This study uses performance data of game players and groups to build performance prediction models for task performing groups. We propose three novel group performance prediction models, namely Group-Composition-Pred, Group-Familiarity-Pred, and Group-All-Pred. The first model uses feature representation scheme where the future performance prediction for a given group is based on group composition (i.e. level diversity, class diversity). The second model uses feature representation scheme where the prediction depends on the degree at which group members are familiar with one another from their past in-game interactions. The third model, Group-All-Pred, combines the previous two models into one. The key contribution we make in this research work is that we introduce the first data-driven group performance management tool to the MMOG gaming community. The tool we develop not only provides retrospective analyses but also prospective and predictive analyses on group performance.

- Lastly, we contribute an implementation of a game player and group behavior analysis framework to the MMOG gaming community. The framework is written entirely using open-source tools and components. The core part of the framework consists of all of the aforementioned analysis and prediction modules, all written in Java programming language. It uses Weka [29], an open source data mining suite, for certain algorithms, JFreeChart [97] for charting, and JasperReports [98] for report generation. The framework also uses Hibernate [99] to configure data models, and the data connection module currently supports MySQL database and Microsoft SQL Server 2005 or above.

In summary, this thesis work presents the first comprehensive quantitative analysis of player and group performance, which is an important aspect of MMOG game play. Our work builds into the existing player performance management tool several analysis and prediction models with respect to player performance, player activity, and player enjoyment. This thesis work also develops the first group performance management tool for MMOGs. Lastly, we develop and contribute to the MMOG gaming community a single, generic player and group behavior analysis framework for MMOGs.

1.6 Organization of the thesis

This thesis is organized as follows:

- Chapter 1 introduces the thesis overview and discusses background and motivation for this thesis work [70, 65].
- Chapter 2 discusses the game dataset used in this thesis work.
- Chapter 3 describes player performance models and metrics [64].
- Chapter 4 discusses the effect of mentoring on player performance [66, 67].
- Chapter 5 presents player performance prediction methods [64, 69].
- Chapter 6 describes player activity prediction models and methods [72].
- Chapter 7 examines the relationship between player performance and player enjoyment. It presents models for predicting player enjoyment [68].
- Chapter 8 describes group performance models and metrics [71].
- Chapter 9 presents group performance prediction methods [71].
- Chapter 10 describes the implementation of all analysis and prediction models discussed in previous chapters in a single, generic analysis framework.
- Chapter 11 presents a final discussion of the analyses presented in this thesis work.

Chapter 2

Dataset

This section explains in details the EverQuest II game dataset used in this thesis work.

2.1 Massively Multiplayer Online Games

2.1.1 Introduction

MMOGs are online spaces providing users with comprehensive virtual universes, each with its own unique context and mechanics. They range from the fantastical world of elves, dwarfs and humans to space faring corporations and mirrors of our world. Large numbers of users interact and role-play via the in-game mechanics. With their increasing popularity, researchers are realizing them as a means to fully observe an entire isolated universe. Each action is logged, and the level of granularity and completeness with which information is collected is unmatched by any real life experimental setup. The data collected provides an excellent means of studying human social behavior with respect to the complete context of the environment, for a large population. Since MMOGs usually have a thin-client architecture, all player actions are captured in the click-stream logged at the server. Along with the players' behaviors, often times a brief record of their real life profile is also stored. And it is common for popular virtual worlds to have hundreds of thousands of players generating copious amounts of data at any given time.

Our research has focused on data from Sony's popular MMOG, EverQuest II. The data primarily consist of three types; avatar characteristics, player demographics and in-game behavior. Avatar characteristics provide data on players' virtual avatars in the game. This includes the avatars' gender, race, faction, health and damage statistics, and more. Player demographics include real life age, gender, geographic location, and more. Survey results from a small sample of players are also available, and it includes response on multiple facets such as feelings towards the game and other users in the game, psychological fitness, and real life behavior outside the game. The most unique are the in-game behavior logs, which contains each user's activities in the game. Anytime a significant event occurs, i.e. the user kills a monster, completes a quest, forms a group with other users, engages in trade and so on, a record of the same is written to the in-game behavior logs. The behavior logs are time-stamped and can be parsed to uncover a wealth of information. For our research, the most important information gleaned from the logs is about user relationships. Players can socially interact with each other using a variety of methods provided by the game mechanics. They can form groups to tackle monsters, go on quests together, trade with each other, express different levels of trust, create and join guilds, mentor other players, and more. Each of these can be represented as a social network among players, which are then studied to understand how they are impacted by, and affect, both the players and the environment. Following are examples of the types of relationship networks that can be constructed from game logs.

- Combat_Network - Team formation is a common occurrence in many MMOGs as the games are designed to encourage social interactions. For instance, certain quests require avatars, each with a different set of skills. Team members must rely on each other while playing their respective set of capabilities and abilities during combats. Studying the formation of task-oriented groups is an important first step to understanding dynamics of team collaboration. Team formation is highly influenced by the common interests of players on challenging tasks. Combat team performance is positively correlated with group size and group level. Over 12 million teams form monthly, making games such as EverQuest II an excellent venue for studying human and organizational behaviors.

- Mentoring_Network - Mentoring in MMOGs is the phenomenon where an experienced player helps a novice player gain skills. Mentors level down to the level of their apprentices in order to teach them combat skills, transmit knowledge about different aspects of the game, and perform one or more tasks with the apprentices. Mentors gain achievement points from successfully completing tasks with their apprentices. Apprentices also gain XP points (plus bonus XP for participating in mentoring) from, for instance, killing monsters but more importantly, they attain valuable lessons and experience by interacting with their mentors. A skilled and knowledgeable mentor will help his apprentice pick up the game quickly and efficiently. A player can have multiple mentors in a session. He can interact with one mentor at a time. When the mentor levels down, his abilities will also scale down to that level.

- Trust Network - Trust networks are difficult to study, since mapping trust relationships in the real world is very difficult. Online interactions, however, provide a unique opportunity to study trust relationships. In some MMOGs such as EverQuest II, players can buy virtual houses and grant access to other players to their houses with varying levels of access. Thus, trust in these games is described with respect to a commodity. Analysis of these trust networks reveals that their properties are similar to trust networks in other domains.

- Guilds - MMOGs are highly social games. By game design, players can play some parts of the game by themselves. For certain classes, it is easier to play combats alone because they are better equipped with armor. Other parts of the game require group playing - requiring a group of players that can complement each other's skills. Some combat groups are temporary. Many games have in-game group/player search feature (using basic demographic information such as player level and class). Using this search feature, ad-hoc combat groups form. While many such combat groups are temporary (meaning that players play together and do not necessarily get together again in the future and therefore, their relationship is need-based), there are persistent groups. The definition of "persisting" is twofold. An implicit form of persistent group is such that the game logs indicate that the same group of players play together for a prolonged period of time without being part of an in-game guild. Typically, in combat groups, there is one player that acts as a group leader. The leader player is responsible for searching for players to join his group. He is the manager of player recruitment. Guild is an explicit form of grouping, where there is a leader who is responsible for player recruitment and other management work. Guilds are large persistent groups, and players in guilds can learn from each other and share their experience in a community setting. Players belonging to guilds can play solo combats as usual and they can also get together with other players that may not necessarily be part of the same guild. However, they can participate in guild-specific activities and move up the ladder (attain higher ranks) in their community. Guild members often get together to tackle large combats or raids.

Figure 2.1: Sony Online Entertainment - EverQuest II

MMOGs offer fantastic game play experience with immersive graphics and storylines. Typically, the virtual worlds that the games offer are organized into multiple ”zones”. The zones can be organized by storylines and organized further for players of varying player levels. For instance, the zone that a new game character enters will typically have enemies whose levels are low. Also, new players will typically be guided by in-game quests, which provide a great means by which new players can learn different aspects of the game while building experience. As the player completes quests and move up the rank, he can then move into other zones with stronger (higher level) enemies.

- Game players can customize their in-game characters via "classes". Different classes are equipped with different strengths and skill sets.
- NPCs or non-player-characters are computer controlled characters (called monsters) that players can fight against and attain points from, upon successful kills.
- Dungeons are locations on game maps, and typically they have a lot monsters for players to come and attack. Players must be at a certain level, in order to be able to access certain dungeons. Some dungeons are exceptionally difficult, requiring a careful coordination of a large number of players with complementary skill sets.

2.2 EverQuest II Game Mechanics

2.2.1 Monster Kills and Point System in EverQuest II

EverQuest II is rich in types of task players can perform with monster kills being one of the most popular. In monster kills, each monster has a level and a tier. The two values indicate the difficulty of a monster. The higher the two values, the more difficult or challenging it is for a given player to kill the monster. The monster level increase is not a monotonic function (i.e., monster level 17 is not necessarily difficult than monster level 16 because difficulty is an aggregate function of monster levels and tiers). In successfully killing the monster, a player obtains points. The amount of points assigned is minimally dependent upon three factors: 1) monster's level, 2) monster's tier, and 3) player's level.

Table 2.1 shows performance data from killing a baby dune cobra. This example shows two different baby cobras: one having level 13 and tier 5 and the other having level 15 and tier 5. Two players of levels 16 and 19, respectively, performed the first task and obtained scores of 52 and 12. In performing the same task, the player with a lower level obtains more points. The same trend is shown in the second example where three players performed the same task, and the player with the lowest level obtains the highest points amongst the three. These examples illustrate how EverQuest II rewards adjusted points based on task difficulty and player skill level.

Figure 2.2: Sony Online Entertainment - EverQuest II - Character and Monster

Table 2.1: Monster Level and Tier

Monster	M-Level	M-Tier	Player Level	Points
Baby dune cobra	13	5	16	52
Baby dune cobra	13	5	19	12
Baby dune cobra	15	5	13	141
Baby dune cobra	15	5	21	27
Baby dune cobra	15	5	22	12

2.2.2 Point-Scaling System in EverQuest II

In EverQuest II, there is a concept of Ding Points, which is the amount of points one needs to obtain in order to move from one level to the next higher level [56]. For instance, to move from Level 2 to Level 3, one needs to obtain 1,000 points whereas 20,000 points are required to move from Level 73 to 74. The amount of ding points increases as one advances to the next level. As players gain more experience with the game and advance to higher levels, the types of task they can perform increase and the task difficulty also increases. The higher the task difficulty, the higher the potential point gain. Does this increase in point gain scale well with the increase in task difficulty?

Numerous online and offline posts and articles within gaming circles report that the relative player rating systems and point-scaling systems are not perfect, often causing distress among advanced players who assert that the systems over-reward low level players and under-reward high level players. Sony's EverQuest II is no exception in that it adopts a relative point-scaling system which requires that higher level players accumulate relatively more points than lower level players in order to advance to the next level [56].

2.2.3 Tasks in EverQuest II

EverQuest II is rich in types of task players can perform with monster kills being one of the most popular. In monster kills, each monster has a level and a tier. The two values indicate the difficulty of a monster. The higher the two values, the more difficult or challenging it is for a given player to kill the monster. The monster level increase is not a monotonic function (i.e., monster level 17 is not necessarily difficult than monster level 16 because difficulty is an aggregate function of monster levels and tiers). In successfully killing the monster, a player obtains points. The amount of points assigned is minimally dependent upon three factors: 1) monster's level, 2) monster's tier, and 3) player's level. Table 2.1 shows performance data from killing a baby dune cobra. This example shows two different baby cobras: one having level 13 and tier 5 and the other having level 15 and tier 5. Two players of levels 16 and 19, respectively, performed the first task and obtained scores of 52 and 12. In performing the same task, the player with a lower level obtains more points. The same trend is shown in the second example where three players performed the same task, and the player with the lowest level obtains the highest points amongst the three. These examples illustrate how EverQuest II rewards adjusted points based on task difficulty and player skill level. In addition to monster kills, other sources of experience points exist in the game such as alternate achievement points (AA) which can be obtained from quests, named mobs, and discovery experience. A player can gain more experience points by having another player mentor him. The mentor levels down to the level of the apprentice. The apprentice receives a five percent bonus to adventuring experience points.

2.2.4 Archetypes, Classes, and Sub-classes in EverQuest II

Selection of character type (i.e. archetype, classes, sub-classes, race, etc.) is considered an important decision as it defines the basis of opportunities and choices of roles and tasks within the game [109]. In EverQuest II, there are four archetypes where each archetype consists of three classes each of which in turn consists of two sub-classes [56]. Figure 2.3 shows average performance of five sub-classes in the month of March, 2006. Performance at each player level is defined as a function of play time at each player level.

Fury sub-class is of priest archetypes. Fury characters specialize in healing, and their primary task as a member of a raid team is to heal other members in combats. Fury sub-class is favorite as a solo character, but it is also effective in team plays (i.e. monster raids, quests). On the other hand, berserker sub-class is of fighter or warrior archetype. It is considered a pure class of fighters, and berserker characters can make use of any weapon possible to fight monsters. They are considered well-rounded as solo

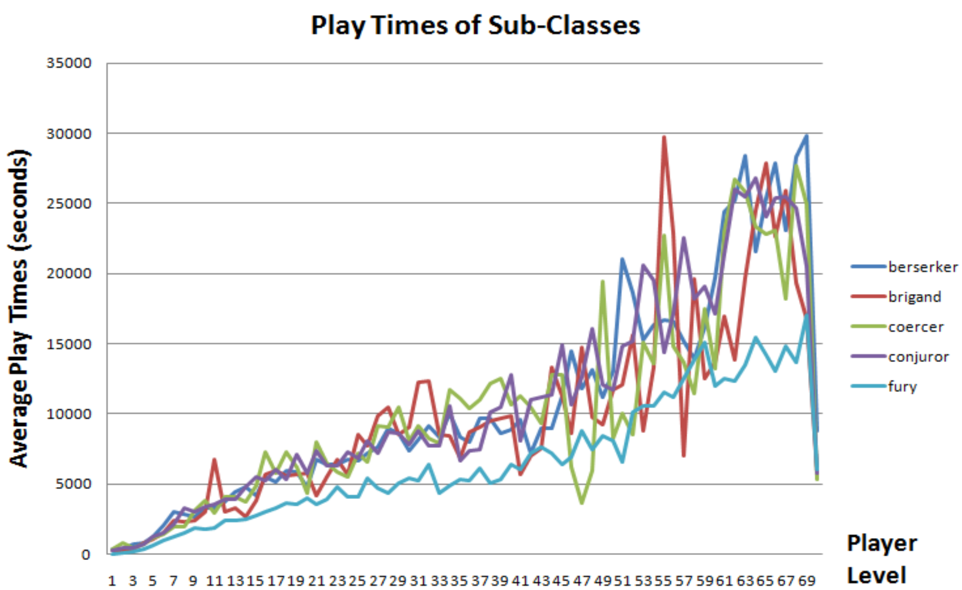


Figure 2.3: Average Play Times of Five Sub-classes in EverQuest II

players or team players. They possess and use heavy armors and can sustain injuries for a long time. In raid groups, berserker character often times play tanks, confronting vicious monsters up front whereas other character play as supporters and healers.

Figure 2.3 shows that players of fury sub-class spend relatively less amount of play time in order to progress to the next level. This trend is consistent across all 70 player levels. There can be several reasons as to why berserker characters, on average, progress to the next level slower than fury characters. One possible explanation might be that berserker characters in general may not be performing activities that would amount to experience point gain. For instance, it is recommended that a player explores a zone that he plans on questing. Zoning does not lead to experience point gain. Yet another explanation might be that sub-classes that progress relatively slower may be performing tasks whose experience point gain is not substantial. For instance, mentoring system in EverQuest II allows a player to level down to mentor a lower level player. The experience point gain for the mentor can be substantially small, however, it allows the apprentice to gain more experience points and the mentor to perform tasks that are no longer accessible to players at his current level. Multiple online resources are available today that show how to level up fast [108], and there can be numerous other explanations as to why certain sub-classes progress relatively slow. The rich dataset we have is expected to allow our analysis to reveal information at the level of granularity appropriate to answer these questions, and it is our future direction to explore these research questions.

2.3 EverQuest II Game Dataset

The study uses nine months' worth of player activity data from January 1, 2006 to September 11, 2006. The dataset contains over 170 million player-to-task records where over 93 million of them are monster kills and quest related tasks. The dataset contains 62,881 distinct players across player levels 1 through 70. Since then, Sony Online Entertainment has added an additional ten levels to the game, making 80 the maximum level one can reach. In a more recent release, Sentinel's Fate, the game maker raised the level cap to 90. All of the characters and their activity data has been extracted from XP table in the EverQuest II database housed at National Center for Supercomputing Applications (NCSA) at the University of Illinois. The dataset contains at the minimum the following information about game players and their characters: character id, character sub-class, race, task, timestamp of task completion, group size (whether a given character grouped with one or more other characters in completing a task), average group level (if a given character played with one or more other characters, this value represents the average of player levels of all characters in that group), experience (XP) points, and location (location in which the task was completed).

2.3.1 In-Game Logs

This study uses nine months' worth of player and team activity data from 'Guk' server (Player-versus-Environment or PvE), 'Antonia Bayle' server (Role-Playing or RP), and 'Nagafen' server (Player-versus-Player or PvP), between January 2006 and September 2006. These three game servers are selected because they represent three different ways to play the game. Therefore, this study provides valuable insights into a wide variety of game genres. The dataset contains the following information about game players and their characters: class, race, task, timestamp of task completion, group size (whether a given character grouped with one or more other characters in completing a task), average group level (if a given character played with one or more other characters, this value represents the average of player levels of all characters involved in that group), experience (XP) points, location (location in which the task was completed), and mentoring and apprenticeship activities. Table 2.2 summarizes this dataset.

Table 2.2: Dataset Overview

	Guk	Antonia Bayle	Nagafen
# of tasks	205+ million	227+ million	308+ million
# of solo tasks	58+ million	69+ million	80+ million
# of group tasks	148+ million	157+ million	228+ million
# of monster kills	98+ million	114+ million	151+ million
# of monster kills (solo)	26+ million	24+ million	25+ million
# of monster kills (group)	71+ million	79+ million	116+ million
# of quests	4.6+ million	6.3+ million	8.5+ million
# of quests (solo)	3+ million	4.6+ million	5.7+ million
# of quests (group)	1.5+ million	1.7+ million	2.7+ million
# of PvP fights	N/A	N/A	4.5+ million

2.3.2 Survey Data

This study uses survey data collected from 7,000 game players in EverQuest II in 2006. The survey data provide information about players' motivations and enjoyment. These data are joined with in-game behavioral data, from the game's game logs, mentioned in the above section. It is noted that while the survey data is a static snapshot taken at a single time point in September 2006, when joined with the in-game behavioral data, the resulting data is enhanced with player activity data across all nine months from January 2006 through September 2006, which includes all of the past in-game activities. Chapter 7 explores in detail survey variables concerning player motivation and enjoyment.

2.3.3 Session Extraction

Our preliminary analysis shows that the total amount of time between a player logs into a game and logs out of the game does not reflect the actual amount of time that the player spent performing tasks or socializing. A player can log in and leave the game without explicitly logging out of the game, hence creating one or more chunks of what we refer to as "inactive" or "idle" time. We programmatically weave one or more active sessions from the game's performance data. Any chunk of time that exceeds 30 minutes without any activity is considered an inactive or idle time, and it is excluded from the total amount of play time computed for each player.

2.3.4 Group Extraction

With respect to group combat, the game logs show which of the remaining players completed a task. The information pertaining to dead players is recorded separately. Hence, group extraction seeks to weave this information to create a more complete picture of who all comprise a given group and who died.

2.3.5 Mentor Network

This dataset contains mentoring networks from three different game servers; 'Guk', 'Antonia Bayle', and 'Nagafen'.

Table 2.3: Dataset Overview

	Guk	Antonia Bayle	Nagafen
Total # characters	64,102	102,177	130,472
Total # characters as mentors (%)	18.4%	14.2%	9.2%
Total # characters as apprentices (%)	28.4%	23.7%	18.0%

The portion of the dataset concerning the 'Guk' server contains 11,419 mentors and 15,091 apprentices over a nine month period. It contains 4,006,860 mentor-apprentice interactions over 2,592,005 monster kills. Of these, over 67% of the monster kills were performed by two-player (one mentor and one apprentice) teams, and over 20% of the monster kills were performed by three-player (two mentors and one apprentice) teams.

Four-player (three mentors and one apprentice) teams comprise some 8% of the monster kills and five-player (four mentors and one apprentice) teams comprise some 3% of the monster kills. The other two servers show similar statistics.

Chapter 3

Player Performance: Models and Metrics

This chapter examines an important aspect of MMOG game play, namely player performance. Player performance has been found to have a close connection to player enjoyment in other game genres [58, 80, 38, 81]. This thesis work is the first comprehensive quantitative analysis of player performance in MMOGs. In this chapter, we define player performance in MMOGs and propose performance metrics with respect to individual game players in the game. We report three major findings. First, we show that the game's point-scaling system overestimates performances of lower level players and underestimates performances of higher level players. We present a novel point-scaling system based on the game's player performance data that addresses the underestimation and overestimation problems. Second, we present a highly accurate predictive model for player performance as a function of past behavior. Third, we show that playing in groups impacts individual performance and that individual player characteristics alone are insufficient in explaining an individual player's performance. The next chapter examines in detail the effect of mentoring activities, as both mentors and apprentices, on player performance.

3.1 Background

Today, there are many different game genres (based on game play interaction); action, shooter, action-adventure, adventure, role-playing, and simulation [95]. Each is further categorized into single-player mode and multi-player mode. Depending on the genre and the type, games can present a wide variety of game mechanics (i.e. point scoring or reward system, point-and-shoot, keyboard button-driven), task types (i.e. monster kills, player-versus-player, problem solving, strategy planning), and modes of interaction either with the in-game computer-driven bots or with other gamers. While there are many dimensions comprising player's game-play experience, in certain game genres, a close connection has been reported between completing tasks/challenges and mastering skills and player enjoyment [58, 80, 38, 81]. In social sciences, the flow theory states that most enjoyment (i.e. positive experiences) is achieved when one masters tasks or challenges that are in the flow zone, where tasks are not too easy and not too challenging or difficult [14, 42]. The flow theory has been explored in the context of game play experience [63, 33, 75, 11, 51, 90].

As shown in Figure 3.1 [93], there are certain games that are easy to ramp up in while others are very difficult to tackle. In the chart, as depicted in the top left corner, some games are notoriously difficult to learn in the beginning. It is not an exaggeration to say that the falling stick figures in the chart are depicting potential churners in the games.

The EverQuest II game logs contain information about when individuals signed up for the game (subscribe) for the first time and when they leave (unsubscribe). Figure 3.2 shows the total number of EverQuest II churners over the period of nine months between January, 2006 and September, 2006. Some 80% of the churners occur in the first 23 game levels. Some of the commercially most successful MMOG vendors (in terms of the total number of subscribers, acceptance by a wide array of demographics, etc.), as reflected in Figure 3.1, have learning or leveling up curves that are gradually increasing in its difficulty.

Game companies are sensitive about public acceptance of game difficulty, especially during the beta testing before they go public. Even after the production launch, in the first couple of months, it is no surprise that game community managers go online and

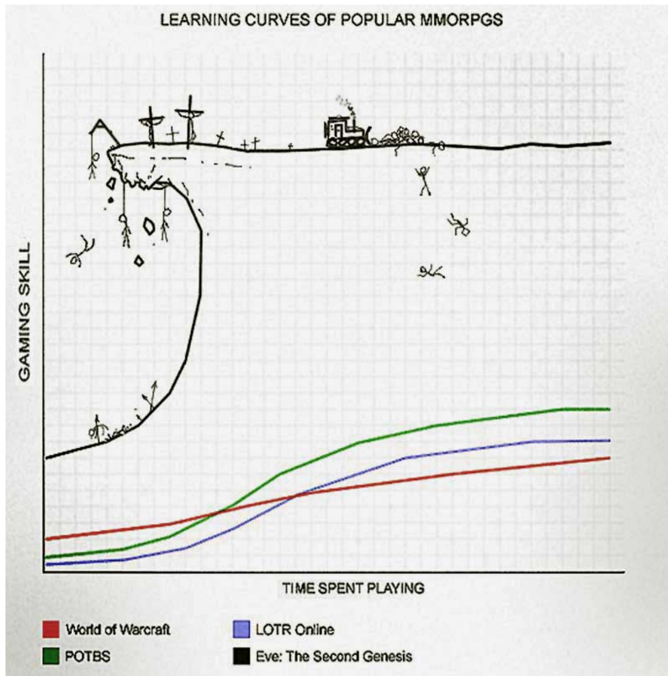


Figure 3.1: MMOG Learning Curve

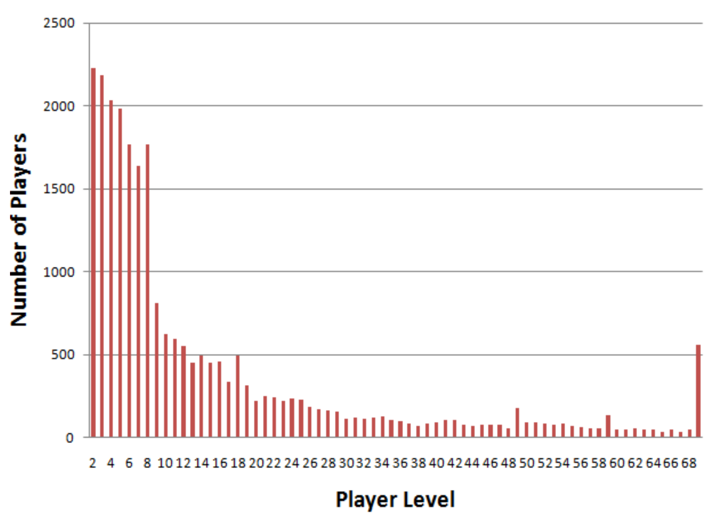
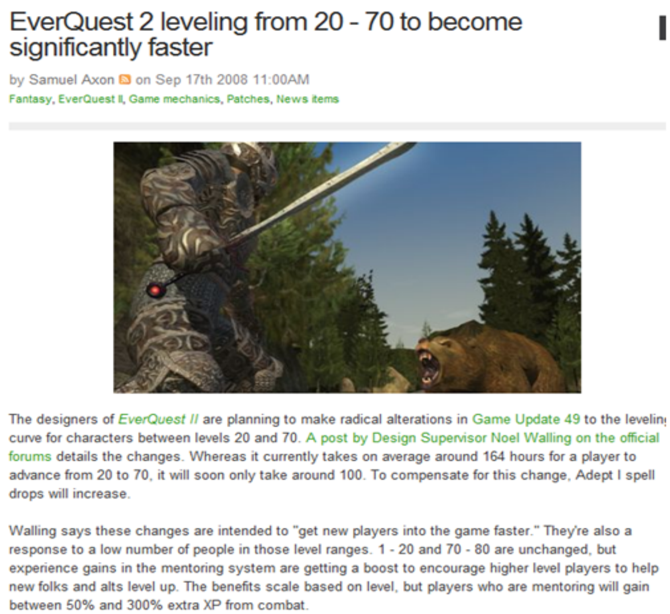


Figure 3.2: EverQuest II - Churners

check on game blogs, collect and analyze customer survey, and general feedback about game features and how much and in what way game players like or dislike the game. As shown in Figure 3.3, sensible game companies do and will respond to public feedback. If leveling up is difficult and game players write about it, sensible game companies will incorporate it into the next releases or put in game patches for game balancing. Especially, as shown in the case of EverQuest II, if a majority of the churners occur in early levels, any feedback concerning these early levels will be taken seriously and promptly.



EverQuest 2 leveling from 20 - 70 to become significantly faster

by Samuel Axon on Sep 17th 2008 11:00AM

Fantasy, EverQuest II, Game mechanics, Patches, News items

The designers of *EverQuest II* are planning to make radical alterations in Game Update 49 to the leveling curve for characters between levels 20 and 70. A post by Design Supervisor Noel Walling on the official forums details the changes. Whereas it currently takes on average around 164 hours for a player to advance from 20 to 70, it will soon only take around 100. To compensate for this change, Adept I spell drops will increase.

Walling says these changes are intended to "get new players into the game faster." They're also a response to a low number of people in those level ranges. 1 - 20 and 70 - 80 are unchanged, but experience gains in the mentoring system are getting a boost to encourage higher level players to help new folks and alts level up. The benefits scale based on level, but players who are mentoring will gain between 50% and 300% extra XP from combat.

Figure 3.3: EverQuest II - Leveling Up - Game Companies Respond

From the game players' perspective, before the beta release or production release, there are already lots of talks about what all the new features are in the release that is coming up. Game companies typically post information about these new features on the game website in a form of videos (cinematic introduction, game play videos) and blog or forum entries. One important aspect of this buzz amongst the public concerns tackling new adventures and leveling up in the game as shown in Figure 3.3. Players lucky enough to be selected beta players get the first sneak peek preview of the game (as the game is still being developed and game features added and removed). Even after the production launch, the very first ones to play the game and are successful at leveling up fast share their insights online. Given the nature of EverQuest II-like games, most such information sharing concerns such topics as class selection, NPCs, quests, locations on the game maps, where interesting adventures are, etc.

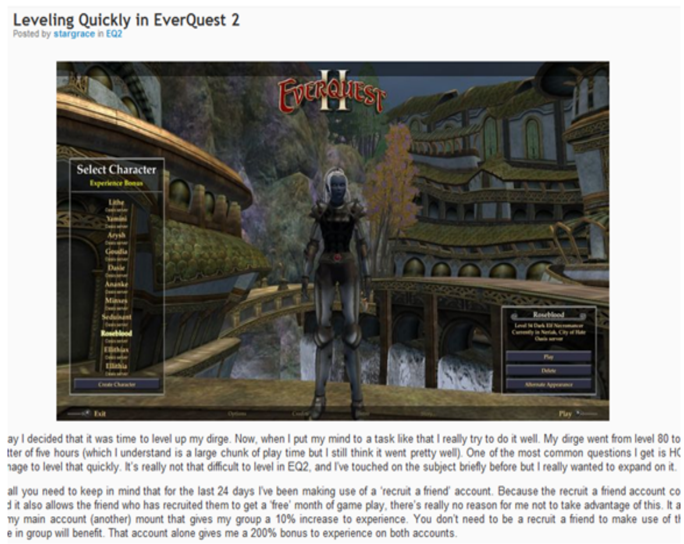


Figure 3.4: EverQuest II - Leveling Up - User's blog

Thus, player performance is an important aspect of MMOG game play. From game companies' perspective, player performance is important because it is directly related to customer retention. Understanding how to measure player performance and what factors impact player performance (i.e. class selection, tasks they perform, task difficulty, their social activities such as mentoring) is important. Furthermore, understanding not only where players come from but where they are going in terms of performance can help game companies act proactively rather than reactively.

Many MMOGs have a player performance management or monitoring tool. Figures 3.5, 3.6, and 3.7 show screenshots from EverQuest II's character dashboard.

While the existing performance management tools provide a quick, convenient way for players to have a peek into their characters, they lack the ability to show 1) how what they have done in the game relate to their performance and 2) how they will progress

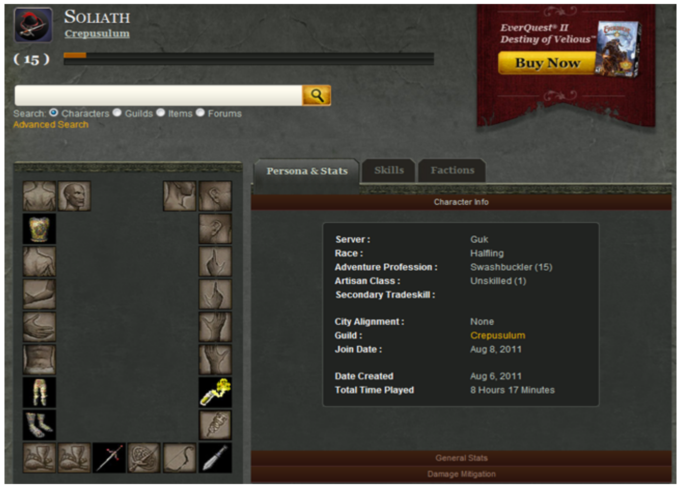
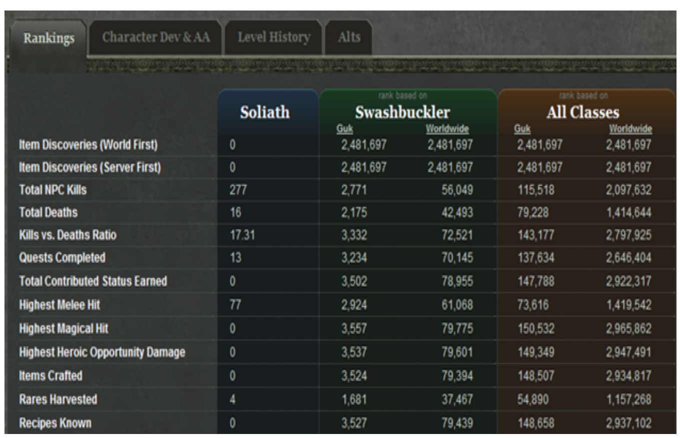


Figure 3.5: EverQuest II - Leveling Up - User's blog



	Soliath	Swashbuckler (rank based on)		All Classes (rank based on)	
		Guk	Worldwide	Guk	Worldwide
Item Discoveries (World First)	0	2,481,697	2,481,697	2,481,697	2,481,697
Item Discoveries (Server First)	0	2,481,697	2,481,697	2,481,697	2,481,697
Total NPC Kills	277	2,771	56,049	115,518	2,097,632
Total Deaths	16	2,175	42,493	79,228	1,414,644
Kills vs. Deaths Ratio	17.31	3,332	72,521	143,177	2,797,925
Quests Completed	13	3,234	70,145	137,634	2,646,404
Total Contributed Status Earned	0	3,502	78,955	147,788	2,922,317
Highest Melee Hit	77	2,924	61,068	73,616	1,419,542
Highest Magical Hit	0	3,557	79,775	150,532	2,965,862
Highest Heroic Opportunity Damage	0	3,537	79,601	149,349	2,947,491
Items Crafted	0	3,524	79,394	148,507	2,934,817
Rares Harvested	4	1,681	37,467	54,890	1,157,268
Recipes Known	0	3,527	79,439	148,658	2,937,102

Figure 3.6: EverQuest II - Leveling Up - User's blog

Figure 3.7: EverQuest II - Leveling Up - User's blog

in the future. Thus, in this thesis work, we present the first comprehensive quantitative analysis of player performance.

3.2 Performance Metrics in Other Domains

In Operations Research, performance has been studied extensively for decades and has resulted in the development of quantitative measures to optimize assembly line production, customer satisfaction, and employee retention. One area of interest in Operations Research is manufacturing strategy, defined in terms of capabilities and resources, and how they are linked to manufacturing performance. Assembly line balancing problems are an example and have been the subject of rigorous research for decades. Although most of the existing literature is focused on single-product assembly lines, recent work in this area has explored the measurement and optimization of performance in team-oriented assembly lines. The main objective of assembly line balancing problem is to maximize efficiency through minimization of idle time [3, 24, 15]. Along the same line, prior literature exists that discusses more complex systems. Mixed model assembly lines are one such example, and it assembles several models of the same product on the same line [9, 18, 47, 77].

In much the same way manufacturing performance is measured in terms of performance, quality and inventory [40], we define performance in EverQuest II as a function of productivity and quality. We define productivity as a measure of how many tasks a given player completes and how many points he/she gains as a result of completing the task(s) in a given time duration. In Operations Research, quality is discussed in terms of defects. In a similar manner, we define quality as success ratio, which is a measure of how successful a given player is at completing a task. In the game, there are multiple types of tasks a player can perform. Monster kills and quests are two prominent types of task in EverQuest II. In the case of monster kills, success ratio is formulated as (number of successful attempts) / (number of successful attempts + number of unsuccessful attempts).

3.3 Impact of Group Formation on Performance

Manufacturing plants over the years have adopted the formation of work teams as a practice [6, 37, 54, 59]. Many companies have adopted team approaches to produce high quality products and services which would lead to improved customer satisfaction [4]. Additionally, huge cost savings coupled with quality improvements have been reported in numerous studies [26, 34, 41, 61, 62, 79]. A more recent study conducted empirical studies on the impact of team formations at workplaces on manufacturing performance over an extended period of time [2].

As is the case in manufacturing, team formation is a common occurrence in many MMOGs [88, 86, 87]. The games are designed to encourage social interactions in such a way that certain quests must be done as a group. Not only that, certain quests require players each with a different set of skills. In order to successfully complete a given quest, the team members must collaborate and rely on one another. In EverQuest II's monster kills, a player can choose to group with and collaborate with one or more players in killing monsters. Such grouping behaviors are often observed in the case of killing difficult or vicious monsters. Also, novice players can team up with more advanced players to get familiarized with the game via the game's mentorship system.

The present study investigates the impact of team formation on individual players' performances: Is teaming up necessarily better or worse for individual players? Does the answer to this question vary depending on participating players' levels?

3.4 Performance-Based Point-Scaling System

An experimental study is conducted to evaluate the existing Ding Points-based Point-Scaling System. The main objective of any point-scaling system should be to raise or lower expectations in terms of performance based on the player's level so that advancing from Level i to Level $i+1$ carries the same amount of difficulty throughout the different levels after factoring in player skill. One way to verify that this condition is currently being met in the game is to measure the average time spent by players to advance from Level i to Level $i+1$. The reasoning behind the measurement of time spent working on tasks is that time spent is generally proportional to task difficulty. If players at Level i are spending substantially more time than what is expected in the entire distribution from Level i to Level $i+1$, it is an indication that the game's ding points-based point-scaling system is imposing expectations too high, and that it is not fair for players at Level i.

3.5 Performance Metrics

3.5.1 Productivity as a Performance Measure

Upon completing a task, a player obtains a certain amount of points, which we call Experience (XP) points. Table 3.1 shows an example of performance data of four players A, B, C, and D.

Case 1 - Differing Number of Tasks, Same XP Points, Same Time Durations

Let us take performance data of Player A and B. Both players gained 570 XP points in 26 minutes. However, Player A completed four tasks to achieve this score whereas Player B completed only three tasks to achieve the same.

Case 2 - Same Number of Tasks, Same XP Points, Differing Time Durations

Let us take performance data of Player B and C. Both players gained 570 XP points by completing three tasks. However, Player B took less time than Player C to achieve this score.

Case 3 - Same Number of Tasks, Differing XP Points, Differing Time Durations

Let us take performance data of Player A and D. Both players completed four tasks. However, Player A achieved a much higher XP score than Player D. Additionally, Player A took more time than Player D achieve higher XP score.

Given the above use cases, the most reasonable measure of productivity in the game is XP points gain.

The reasoning behind leveraging only XP points, not number of tasks, is that as shown in Use Case 1, the two players took different paths (one choosing three relatively difficult tasks and the other choosing one easy task and three relatively difficult tasks) but they both achieved the same XP points at the end of the day, in 26 minutes. For Player B, the 26 minute duration might have been more intensive than that of Player A because he faced more difficult monsters but one can make a similar argument by saying that Player A kept himself busy by completing one more task than Player B.

Table 3.1: Task Performance and Experience Points

	Task	Task	Task	Task	Total
Player A	1	2	3	4	
(Level 25)	(50)	(200)	(120)	(200)	570 XP
Time spent	3 min	8 min	5 min	10 min	26 mins
Player B	5	6	7		
(Level 25)	(150)	(200)	(220)		570 XP
Time spent	6 min	9 min	11 min		26 mins
Player C	5	6	7		
(Level 25)	(150)	(200)	(220)		570 XP
Time spent	4 min	8 min	10 min		22 mins
Player D	10	11	12	13	
(Level 25)	(50)	(40)	(40)	(50)	180 XP
Time spent	3 min	1.25 min	1 min	2 min	9.25 mins

Apart from the number of tasks, the two primary reasonings behind using XP points are 1) XP points reflect task difficulty and 2) tasks with higher difficulty levels take more time. Hence, coupled with time measure (time taken to complete a set of tasks), XP points gain can provide a good measure of a player's productivity. The performance of Player K as a function of Productivity (Efficiency Index) is defined as the following:

$$EfficiencyIndex_k = \frac{\sum_{i=1}^{N} XP_i}{\sum_{j=1}^{M} ST_j}$$

where

XP = Experience points

N = Total number of tasks completed by Player K

ST = Session time

M = Total number of sessions during which Player K completed tasks

3.5.2 Quality as a Performance Measure

In Operations Research, quality is discussed in terms of defects. In a similar manner, in EverQuest II, quality is defined as success ratio, which is a measure of how successful a given player is at completing a task. We refer to this success ratio as Success Index. It is formulated as follows:

$$SuccessIndex_k = \frac{(\text{\# of successful attempts})}{(\text{\# of successful attempts + \# of unsuccessful attempts})}$$

Success ratio is specific to a task and performing of that task, and therefore, the aforementioned Performance formula can be adjusted to account for task-specific quality. Hence, the performance of Player K as a function of productivity and quality is as follows:

$$AdjustedEfficiencyIndex_k = \frac{\sum_{i=1}^{N} (XP_i \times Q_i)}{\sum_{j=1}^{M} ST_j}$$

where

XP = Experience points

N = Total number of tasks completed by Player K

ST = Session time

M = Total number of sessions during which Player K completed tasks

Q = Quality or success ratio associated with completing Task i

3.6 Experiments and Results

3.6.1 Dataset

The dataset used in this study is described in Chapter 2. We further refined the dataset, and it contains 62,881 distinct players across player levels 1 through 70. The reasoning behind this filtering is that some of our experiments require data points where the game players have played multiple levels and their activities across multiple levels are logged in the game log. Not only that, due to the fact that the original dataset is a nine months snapshot, we had to filter in only those data points where each game player's entire history at a particular player can be traced. Hence, we end up with 62,881 distinct data points and that is still a substantially large dataset size.

3.6.2 Evaluation of Ding Points-Based Point-Scaling System

An experimental study is conducted to evaluate the existing Point-Scaling System. The game's ding points are indicative of player level difficulty or how much effort is needed to move from Level i to Level $i + 1$. Another source of player level difficulty is the game's performance data. From the performance data, we extract session time, which is indicative of how much effort is actually being spent to move from Level i to Level $i + 1$. We compare this against the ding points-based point-scaling system to see how well the ding points-based point-scaling system reflects the actual player performance.

Figure 3.8 is meant to compare ding point system and the actual performance of the players. The x-axis shows the player levels while the y-axis is a ratio. The green bars show the ratios between ding points required to move from level i to level $i+1$ while the red bars show the ratios between amounts of time spent in moving from level i to level $i+1$ divided by the maximum time spent. From this figure, it can be observed that up

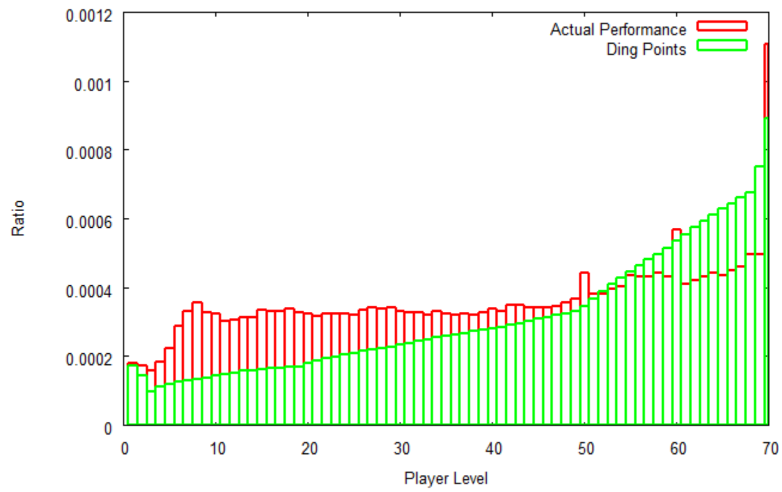


Figure 3.8: Point-Scaling System versus Actual Time Spent

until Level 49, the actual time spent by players performing tasks is more than expected. This could potentially mean that the tasks performed by players up until Level 49 were more challenging than expected as time spent increases with an increasing level of task difficulty. Between Levels 50 and 55, the actual time spent is well in accordance with what is expected. Beyond Level 55 up until Level 68, the actual time spent is well below what is expected. This could potentially mean that the tasks performed were not challenging enough as time spent decreases with a decreasing level of task difficulty. Game developers can use the above two pieces of information to do the following.

First, they could lower standards/expectations for players at levels below 49 by decreasing the amount of ding points required to move up to the next level. Players will need to either complete less number of tasks or complete less challenging tasks. Alternatively, the score adjustment formula can reduce the penalty imposed on lower level player.

Secondly, they could raise the standards/expectations for players at Levels 55 through 68 by increasing the amount of ding points required to move up to the next level. Players will respond by trying more challenging tasks as increasing challenge level positively correlates with increasing experience points necessary to move up to the next level. Alternatively, players will need to complete more tasks if they opt for not trying more challenging tasks as completing more tasks will result in experience points gain necessary to move up to the next level. Another way for the game developers to respond to this finding is that in EverQuest II, experience points are adjusted based on the player level and perhaps, the score adjustment formula as discussed in Section II-A can reduce the reward amount imposed on higher level player.

Analyzing the game's player performance data reveals information valuable in understanding at what rate players advance through completing tasks of different difficulty levels. The results from this analysis can be used to establish a measure of how difficult it is to move from Level i to Level $i + 1$. This measure would be similar to the existing Ding Points-based Point-Scaling System, however, as the study reveals, this is not completely reflective of actual player performance.

This observation has practical consequences for game development since it is known that one of the main attractions for game players in any game is the challenge associated with the game [19]. Understanding the balance between keeping players relatively busy and challenged versus bombarding them with difficult tasks is valuable in devising training routines for soldiers and novices.

3.6.3 Impact of Group Formation on Player Performance

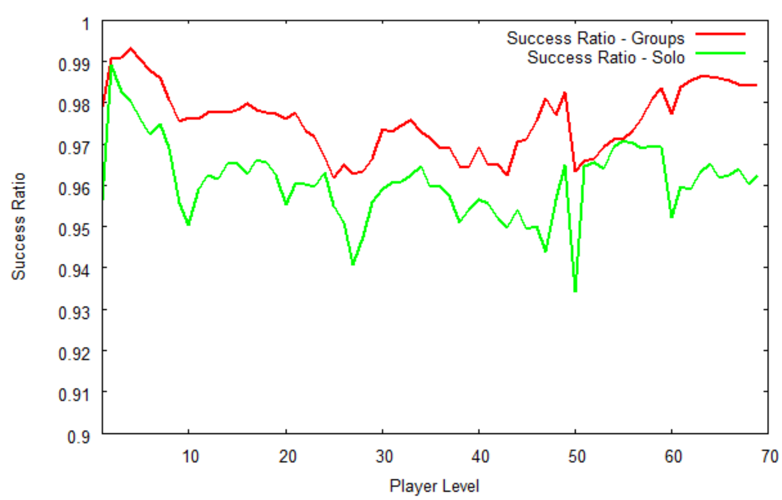


Figure 3.9: Success Ratio - Playing Solo versus Playing in Groups

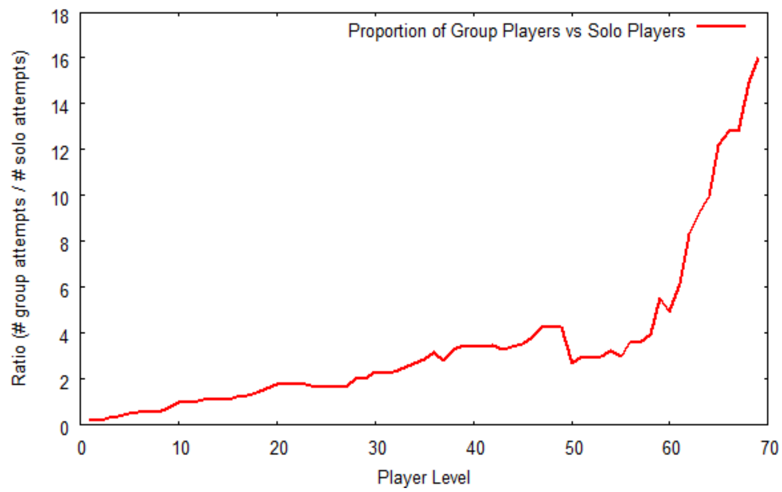


Figure 3.10: Proportion of Group Players versus Solo Players

To evaluate the impact of group formation on individual players' performances, we computed each player's quality measure only in the context of monster kills. We then aggregated over all players at each level, and the resulting Figure 3.9 shows that in most levels, individual players' success ratios are higher when they played in groups. Figure 3.10 shows a trend that as the player level increases, the proportion of players playing in groups increases. Higher level players tend to fight more difficult monsters which presents the necessity to group with other players in order to successfully kill the monsters. Additionally, players reaching higher levels are more inclined to join guilds or raid groups. Moreover, a player's higher level status attracts other players to group with him.

3.6.4 Evaluation of Performance Metrics

In this set of experiments, we evaluate the first Performance Metric. Player performance as a function of productivity and/or quality reveals a given player's rate at which he or she advances through player levels by completing tasks of different difficulty levels and/or by grouping with other players. Given the player's past performance, it is possible to predict his or her future performance.

At each player level (Level i), we select N players and compute their Efficiency Index scores. The game's existing ding points-based point-scaling system dictates that there is a fixed amount of points to be gained between any Level i and Level $i+1$. Given a player's Efficiency Index score and the fixed amount of points between Level $i+1$ and Level $i+2$, we can compute the total session time (play time) and this becomes our prediction as to how fast this player will advance to the next higher level in the future.

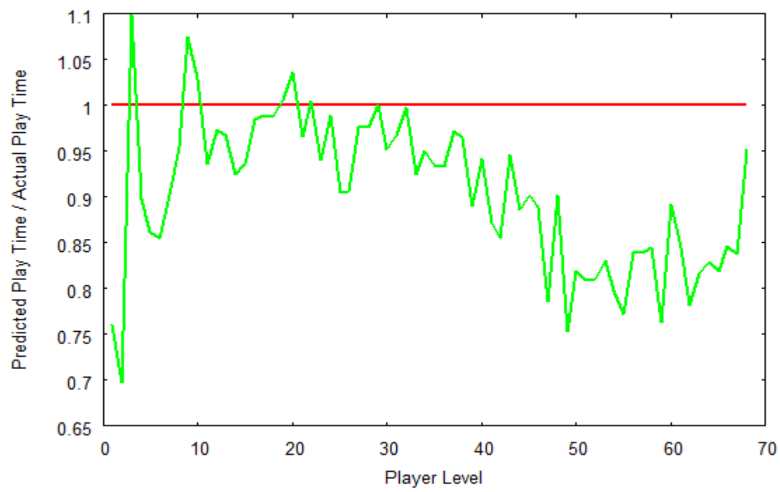


Figure 3.11: Ratio of Predicted Play Time and Actual Play Time by Player Level

We compare the predicted play time against the actual play time. We take the ratio between the two and observe at each player level what is the margin of prediction error is. Figure 3.11 shows that for Levels 2 and 3, our method underestimated the actual play time. Between Levels 4 and 48, the margin of prediction error stays well within 18% boundary. Beyond Level 48, the margin of error increases and players' performances become less predictable. For higher level players, our method tends to underestimate the actual play time.

Coupling this finding with Figure 3.9 and Figure 3.10 reveals an interesting pattern. As the player level increases, group formation becomes a more common occurrence. And playing in groups leads to higher success ratio at the individual player's level. There is a tradeoff between playing solo versus playing in groups. From timing perspective, playing solo allows a given player to advance faster than it would if he were to play in groups potentially due to process loss or coordination overhead incurred by having to get multiple players together. From the perspective of successful task completion and success ratio, playing in groups serves as an advantage in that the chance of getting a given task done is higher for a given individual player in this setting.

Analyzing historical performance data of a player is expected to yield information valuable in understanding his learning trajectory over time. To further evaluate Efficiency Index, we conducted an experiment. In one use case, we used the immediate past performance data (from Level $i+1$ to $i+2$) as a predictor for future performance (from Level $i+2$ to $i+3$). In the other use case, we used a more distant past performance data (from Level i to $i+1$) as a predictor for future performance (from Level $i+2$ to $i+3$).

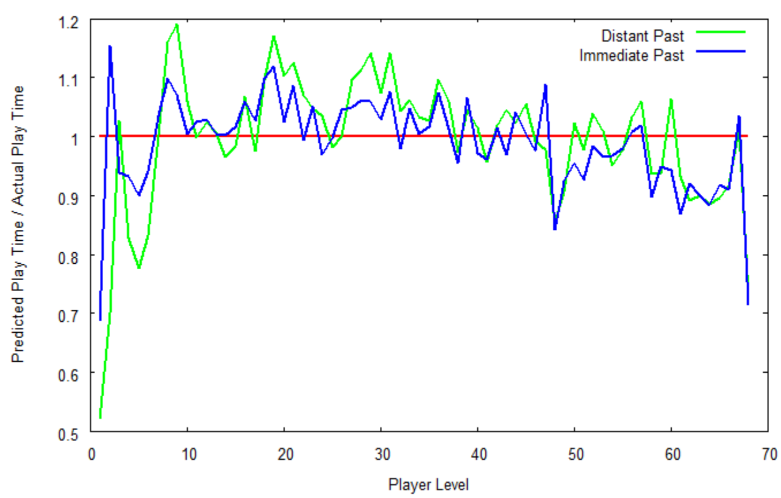


Figure 3.12: Predicted Play Time versus Actual Play Time (Distant past as a predictor of future performance)

Figure 3.12 shows that the margin of prediction error is larger when we use a more distant past performance data as predictor for future performance. It is evident from the results that the quality of performance data as a predictor for future performance decays with an increasing distance on the player level scale. Our finding indicates that in order to incorporate more distant performance data into future performance prediction, some sort of a weight assignment or decay function must be applied in such a way that the most weight is given to the most immediate past.

Our findings indicate that our proposed Efficiency Index is suitable for predicting individual players' future performances in absence of impact of group formation. These findings call for more thorough and comprehensive studies on different types of group formations (homogeneous, heterogeneous, social interactions amongst the group members, etc.) and their impact on individual players' performances. This topic is covered in later chapters.

We conducted a similar experiment to evaluate Performance Metric 2. In this experiment, we used only monster kills (either as single events or as part of quests) for analysis because these tasks are readily amenable to analysis in terms of success and failure. Game logs clearly mark information about success and failure in such events. Failure in this context would mean the death in the game, while failure cannot be readily described for most other task types. For instance, suppose a player embarks on a quest (which is a task). A typical quest consists of multiple sub-tasks such as monster kills, finding items, visiting certain locations on a map, and etc. The game logs clearly mark when players start a particular quest and completes each sub-task. However, sometimes, players finish some sub-tasks and just give up, and thereby, not completing the quest. One might argue that in such cases, these players failed at completing this quest. Once players start a quest, it gets entered into their in-game journals. Unless players explicitly remove the quest from their journals, the quest continues to remain there. Removal of quests from in-game journals is not an activity logged in the game logs made available during this study, hence, we focus on monster kills where success (killing an NPC) and failure (getting killed by an NPC) are explicitly marked in the game logs.

The results were not significantly different from the Efficiency Index evaluation results. In monster kills, taking quality into consideration for performance metric does not lead to better predictions.

3.7 Conclusion

This chapter analyzes EverQuest II's player performance data to devise individual player performance metrics. First, our analysis reveals that the game's existing ding points-based point-scaling system is in general well in accordance with the actual player performance observed in the game's historical performance data. It also reveals that the level of granularity that the performance data offers can potentially lead to fine tuning of the existing point-scaling system. Secondly, the proposed performance metrics define performance as a function of productivity and/or quality. Our findings demonstrate that a given player's past performance can be used as a predictor for his future performance. Our findings also indicate that the proposed performance metrics yield less than optimal predictions about individual players' future performances in higher levels where group formation increasingly becomes a common occurrence. Additionally, the study reveals that in a certain type of task (i.e. monster kills), the quality aspect of individual player performance plays an insignificant role in predicting a player's future performance. Future directions include studying different ways of defining quality in all types of task in the game to devise more generalizable individual player performance metrics, conducting a more thorough and comprehensive analysis on the impact of group compositions and social interactions on individual player performance, investigating individual learning trajectory over a larger time span, and developing group performance metrics.

Chapter 4

Player Performance: Effect of Mentoring

This study presents a detailed analysis of the effects of mentoring on player performance and uses player performance metrics to measure and quantify the effects of mentoring on player performance. Specifically, we investigate and report findings on social diversity in mentoring network in EverQuest II. We examine three different game servers from the EverQuest II game logs. In all three servers, the results from our analyses suggest that increase in social diversity in terms of characters and classes encountered moderately negatively correlates with player performance. The findings from this study are expected to be a useful addition to the existing game system by providing a close-up look at how mentors and apprentices interact. The outcome of the social interactions between mentors and apprentices as reflected by performance metrics can be used to enhance the existing group and player recommendation feature of the game by systematically and accurately matching up mentors with apprentices so that their respective objectives are fulfilled. Our contributions include a better understanding of performance metrics used in the game and a foundation of recommendation systems for mentors and apprentices.

4.1 Mentor System in Games

The mentor system acts as a channel for both the mentor and the apprentice to interact with each other in such a way that they both obtain what they need or want. EverQuest

II, Age of Conan, and Vanguard are some of the games that offer mentor system. In some other games, though there does not exist an explicit mentor system, mentoring and apprenticing activities take place in such a way that the guild or community create characters, equip them with appropriate gear in any way and amount they want, and teach inexperienced new members about character types and what all they can do with different character types. One of the most useful aspects of mentor system is that it allows game players to interact with each other and perform tasks together without being restricted to their respective players levels. Often times, high level players play in a different space than low level players in that monsters that low level players may find interesting to attempt (because the challenge level is appropriate for the low level players) may be too easy or boring to attempt for high level players. In some cases, high level players may not come across such easy targets at all. For two friends of greatly differing player levels to hang out, mentor system is there for them in games like EverQuest II.

In mentor system, higher level players act as mentors to lower level players. Mentors level down to the level of their apprentices in order to teach them combat skills, transmit knowledge about different aspects of the game, and perform one or more tasks with the apprentices. This is almost analogous to an in-class one-on-one teaching between a teacher and a student. Mentors gain achievement points from successfully completing tasks with their apprentices. Apprentices also gain XP points (plus bonus XP for participating in mentoring) from, for instance, killing monsters but more importantly, they attain valuable lessons and experience by interacting with their mentors. A skilled and knowledgeable mentor will help his apprentice pick up the game quickly and efficiently. A given player can have multiple mentors in a session. He can interact with one mentor at a time. When the mentor levels down, his abilities will also scale down to that level. In EverQuest 2, one must be at least at Player Level 8 to become a mentor and at Player Level 7 to become a apprentice. When playing the game, in order to participate in mentoring, the apprentice candidate must have XP Gain turned on.

4.2 Effects of Mentoring on Player Performance

This study presents a detailed analysis of the effects of mentoring on player performance and uses player performance metrics to measure and quantify the effects of mentoring on player performance. Techniques for assessing and projecting individual player performance has been well studied in [64, 69]. In EverQuest II, there are many different types of networks where social interactions are abundant. In this study, we analyze the combat network in EverQuest II where we focus on combat-oriented activities where the skill assessment and projection models have already been built [64, 69]. Embedded within the combat network is yet another network called mentorship network. This network consists of two or more players teaming up together to kill monsters where one or more team members (of higher level) can mentor a team member (of lower level).

While there are numerous studies on the effects of mentoring on mentors and apprentices [1, 43, 83, 46], little is understood about the effects of mentoring on player performance in highly social-driven games such as MMOGs. In this study, broadly we examine the effects of mentoring on player performance of both the mentor and the apprentice. How much mentoring is happening in the game and how much does it help players? First, we analyze the mentorship network and show basic statistics of who all are participating in mentoring activities. Next, we examine and report our findings on the effects of participating in mentorship activities as both/either mentors and/or apprentices on player performance. We use the performance metrics to measure the magnitude of the effects of mentoring on player performance. The next section describes different player performance metrics.

4.3 Mentor Network

As described in Chapter 2, the study uses nine months worth of player and team activity data from 'Guk' server (Player-versus-Environment or PvE), 'Antonia Bayle' server (Role-Playing or RP), and 'Nagafen' server (Player-versus-Player or PvP), between January 2006 and September 2006.

4.4 Mentoring Activities in EverQuest II

In this section, we examine mentoring activities in EverQuest II. We examine both the apprentice population and the mentor population.

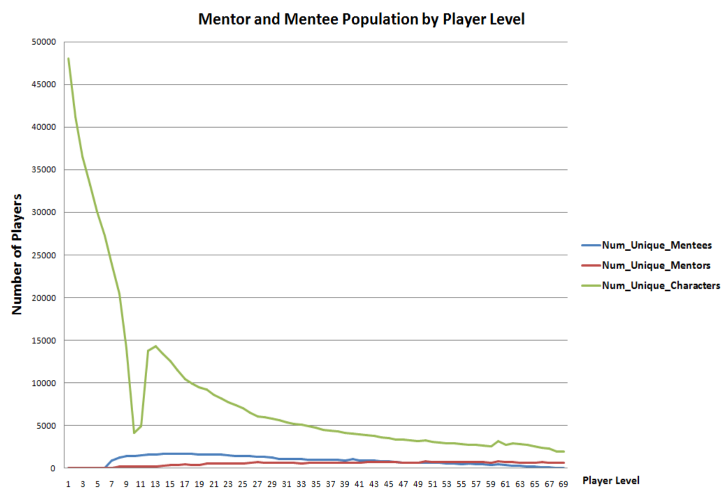


Figure 4.1: Number of Mentors and Apprentices by Player Level (Frequency)

Figure 4.1 shows that as the player level increases, the total number of players decreases, yet the total number of mentors steadily increases whereas the total number of apprentices slightly increases but steadily decreases after Player 15.

Figure 4.2 magnifies Figure 4.1 and shows a steep increase in apprentice population in lower levels followed by a steady decline as the player level increases. It also shows a steady increase in mentor population.

Figure 4.3 shows mentor population and apprentice population ratios. It shows that as the player level increases, the ratio of players participating in mentoring as mentors increases. However, the ratio of players participating in mentoring as apprentices drastically increases from Player Level 7 to Player Level 10, and then it drops by over 20% from Player Level 10 to Player Level 12. It steadily increases until Player Level 40 and takes a steady drop afterwards.

Figure 4.4 shows that for those players participating in mentoring as apprentices, between Player Level 7 and Player Level 12, 50% or more of their combat activities

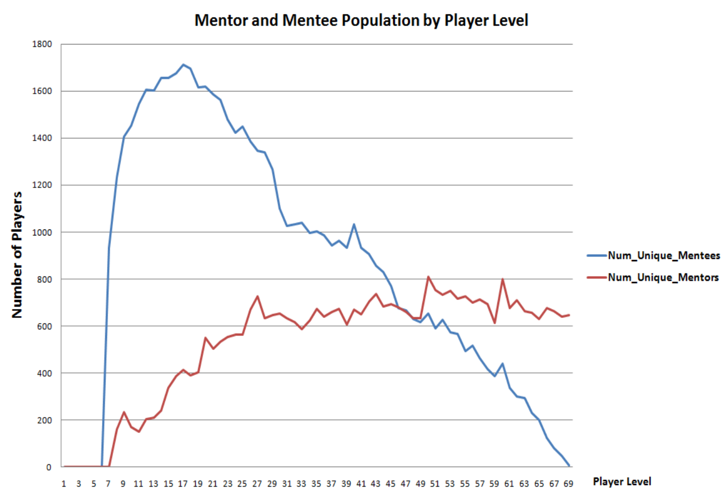


Figure 4.2: Number of Mentors and Apprentices by Player Level (Frequency)

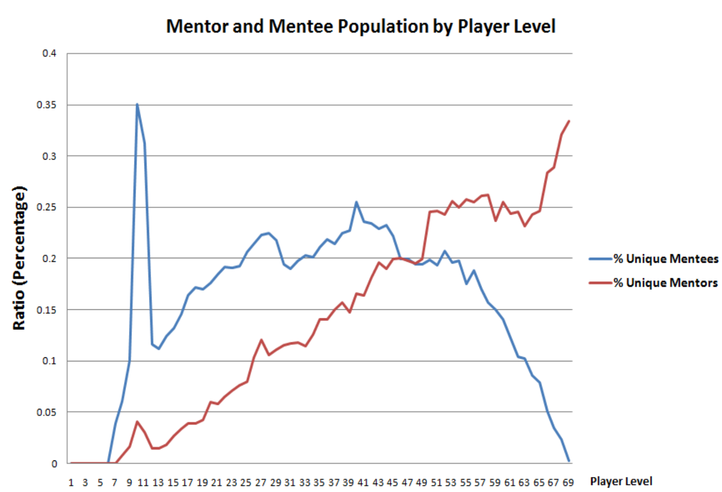


Figure 4.3: Number of Mentors and Apprentices by Player Level (Ratio)

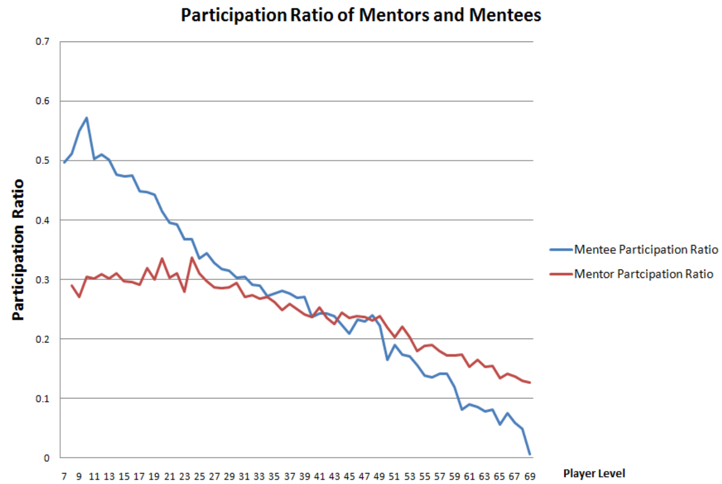


Figure 4.4: Mentor and Apprentice Participation Ratio

are accompanied by one or more mentors. Their participation ratio gradually decreases as they advance to higher levels. For those players participating in mentoring as mentors, until Player 30, some 30% of their combat activities involve mentoring lower level players. Their participation ratio steadily decreases afterwards.

Figure 4.5 shows, at each player level, a boxplot of Mentor-to-Apprentice player level difference. As the player level increases, the median level difference increases. Additionally, as the player level increases, the inter-quartile range also increases. The median line within each boxplot gradually becomes less and less equidistant from the hinges, indicating that the distribution of the level differences at each player level becomes skewed with an increasing player level. The boxes increasingly become shifted to the low end, indicating that the distributions become more positively skewed. Also, as the player level increases, the increasing size of the box signifies a distribution with a thinner peak. Also, as the player level increases, more outliers are observed towards the high end. These extreme values represent mentor-apprentice pairings where the mentor's player level is significantly larger than that of the apprentice.

Figure 4.6 shows the distribution of mentor population and apprentice population across the 40 different classes. The figure shows the raw frequency of total number of players. The distribution of the player population is not equal across the 40 classes.

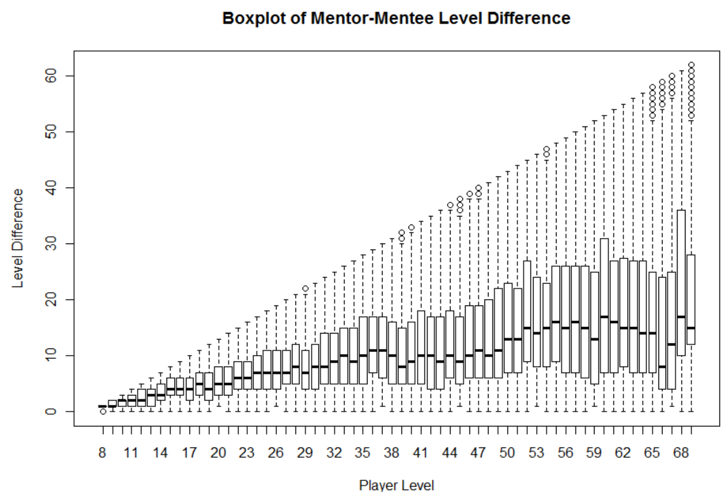


Figure 4.5: Boxplots of Player Level Difference between Mentor and Apprentice

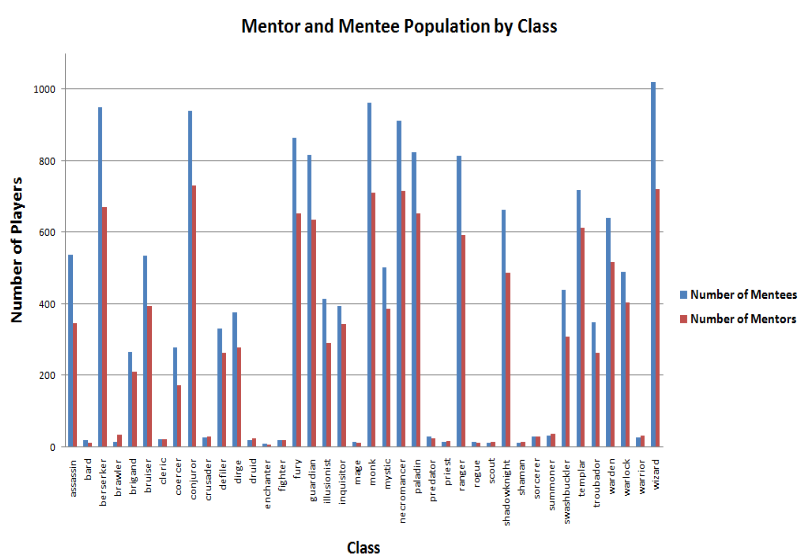


Figure 4.6: Mentor and Apprentice Population by Class (Frequency)

"Assassin", "Berserker", "Conjuror", Fury", "Guardian", and "Monk" are some of the most popular classes.

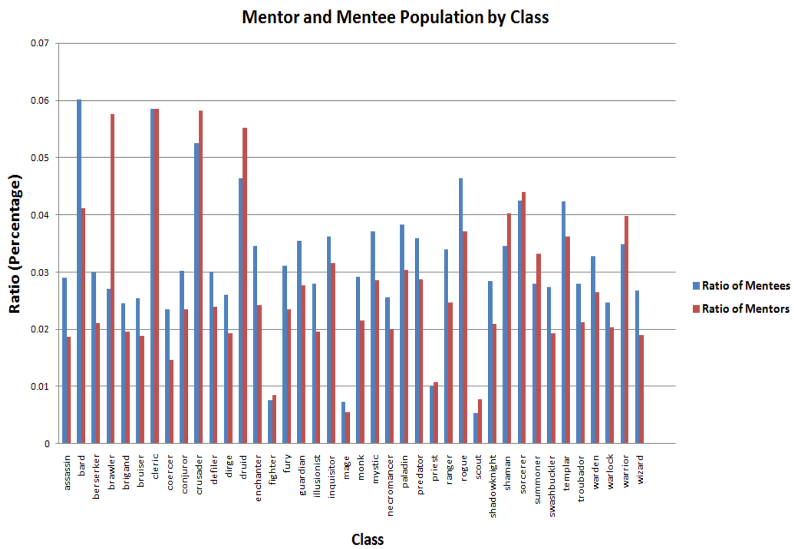


Figure 4.7: Mentor and Apprentice Population by Class (Ratio)

Next, we examine what percentage of the players in each of the 40 classes participate in mentoring. Figure 4.7 shows the distribution of mentor population and apprentice population across the 40 different classes. The figure shows the ratio of mentors or apprentices belonging to each of the 40 different classes. "Bard", "Brawler", "Cleric", "Crusader", and "Druid" classes stand out with a comparatively (to other classes) larger percentage of their players participating in mentoring.

Figure 4.8 shows boxplots of participation ratios of apprentices across the 40 different classes. Take "Assassin" class, for instance. 20% of their combat activities are accompanied by one or more mentors. The "Cleric" class (7th boxplot from the left) characters on average have over 65% of their combat activities accompanied by one or more mentors. Likewise, the "Bard" class (2nd boxplot from the left) characters on average have over 80% of their combat activities accompanied by one or more mentors. The figure shows that the apprentice participation ratio differs from one class to another, though, it appears that the majority of the classes have participation ratios lower than 50%. The "Shaman" class (10th from the right) stands out with an average

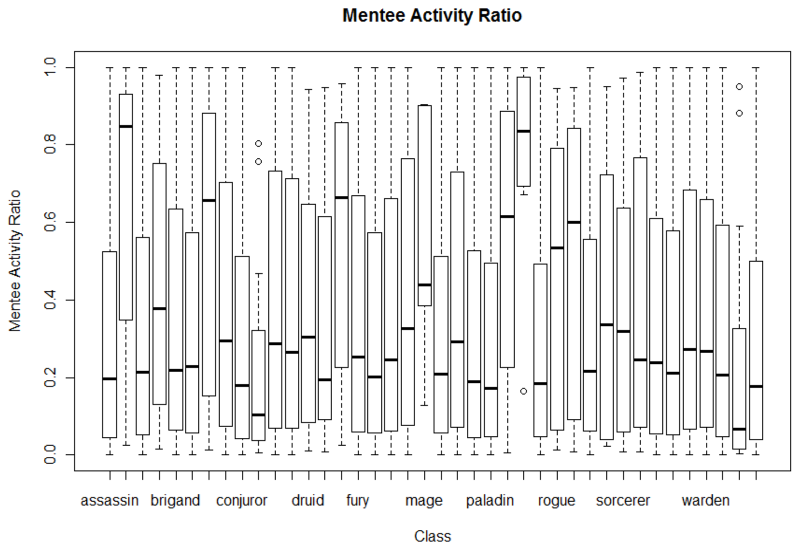


Figure 4.8: Apprentice Participation Ratio by Class

mentoring participation ratio of some 50%.

Figure 4.9 shows boxplots of participation ratios of mentors by different classes. The fighter, mage, priest, and scout classes are omitted in this figure due to their small dataset sizes. There are a few other classes whose dataset sizes are small but present in this figure. For those classes, there is not enough variation in the data distribution, hence, they appear as horizontal disks (i.e. the "Bard" class appears second from the left as a disk). Take "Predator" class (14th boxplot from the right), for instance. About 30% of their combat activities involve mentoring other players.

Next, we examine the impact of participating in mentoring on player performance.

Figure 4.10 shows the four segments of the player population based on which we perform a comparison of player performance.

Figure 4.11 shows a comparison of Combat Efficiency Indexes of the four segments of the player population. The blue line represents the Combat Efficiency Index of players that do not participate in mentoring at all. In low levels, from Player Level 1 and Player level 6, the Combat Efficiency Index of non-participating players gradually decreases but starts ramping up at Player Level 7 and it steadily increases. Between Player Levels 7 and 30, we observe that the Combat Efficiency Index of non-participating players is

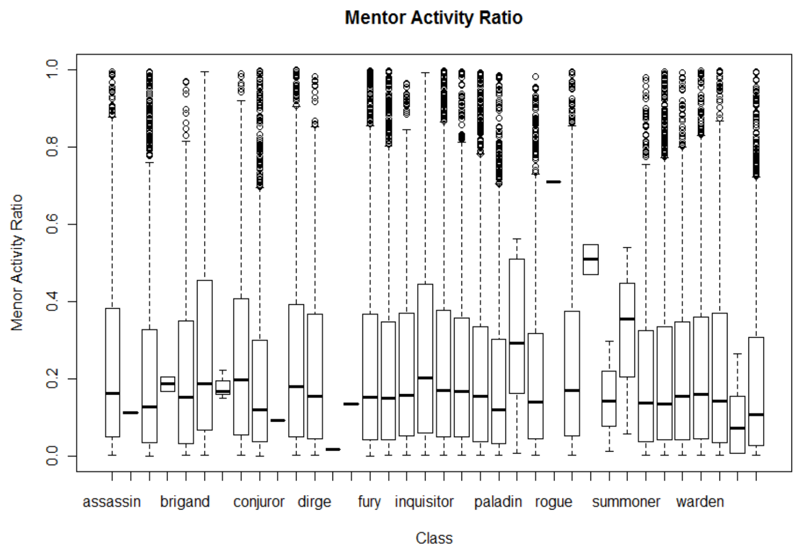


Figure 4.9: Mentor Participation Ratio by Class

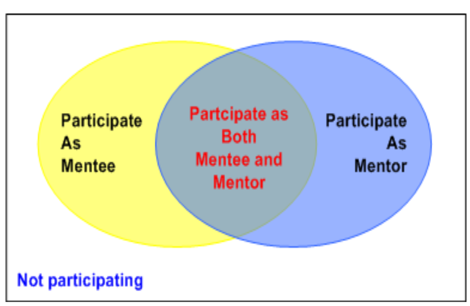


Figure 4.10: Mentoring Population Segmentation

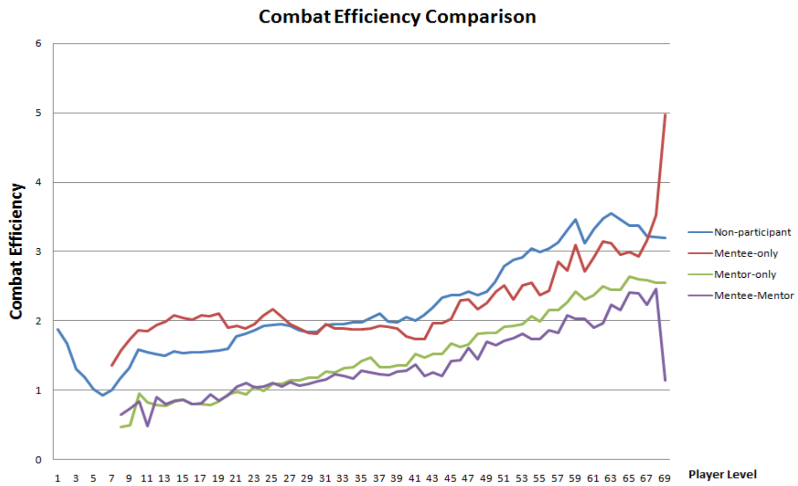


Figure 4.11: Comparison of Combat Efficiency Index by Player Level

higher than that of mentor-only (players that do not apprentice but only mentor others) and apprentice-mentor (players that mentor others and also apprentice). Throughout the game play, through the ultimate level 70, their Combat Efficiency Index steadily rises and stays well above mentor-only and apprentice-mentor. Initially between Player Levels 7 and 27, the Combat Efficiency Index of apprentice-only is higher than that of non-participating players. However, beyond Player Level 27, it steadily increases, but the Combat Efficiency Index of non-participating players surpasses it. Our findings indicate that mentoring leads to decreased Combat Efficiency. Given the same amount of time, apprentice-only players and non-participating players attain more XP points.

Figure 4.12 shows a comparison of Combat Success Indexes of the four segments of the player population. In lower players levels, non-participating players tend to achieve lower Combat Success Index compared to those that participate in mentoring. However, the Combat Success Indexes of the four population segments gradually converge as the player level increases. Mentor-only and apprentice-mentor players achieve higher Combat Success.

Earlier, we reported that non-participating players and apprentice-only players achieve higher Combat Efficiency. Figure 4.13 shows that that finding is true across most of the 40 classes. Participating in both mentoring and apprentice activities leads to decreased

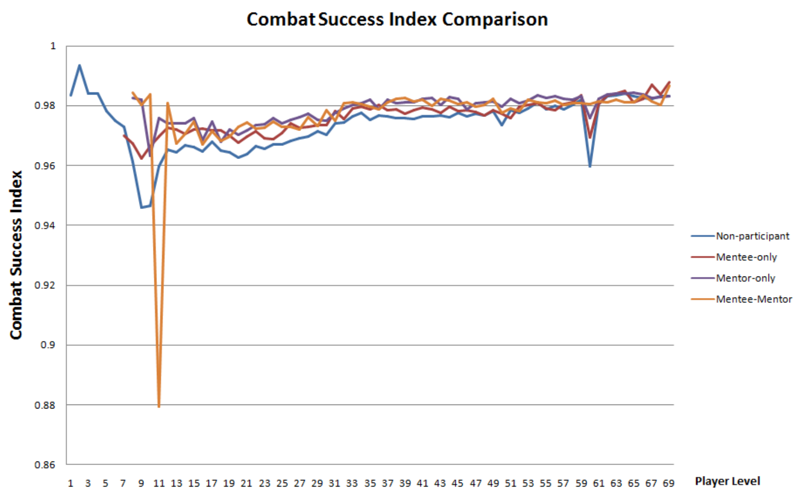


Figure 4.12: Comparison of Combat Success Index by Player Level

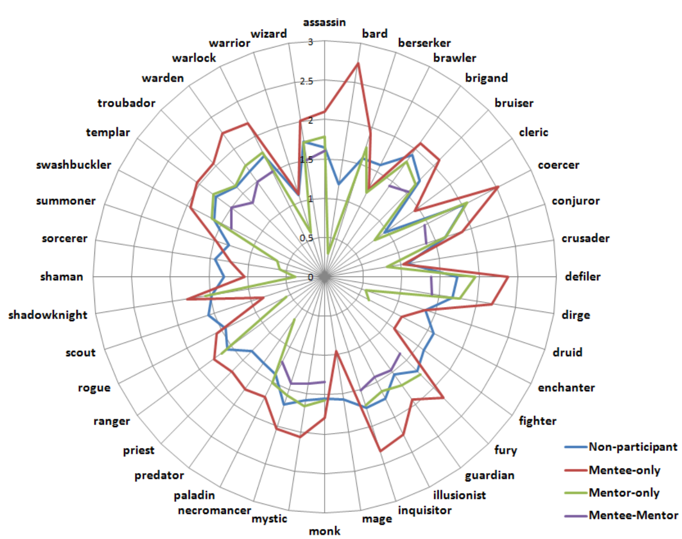


Figure 4.13: Comparison of Combat Efficiency Index by Class

Combat Efficiency Index.

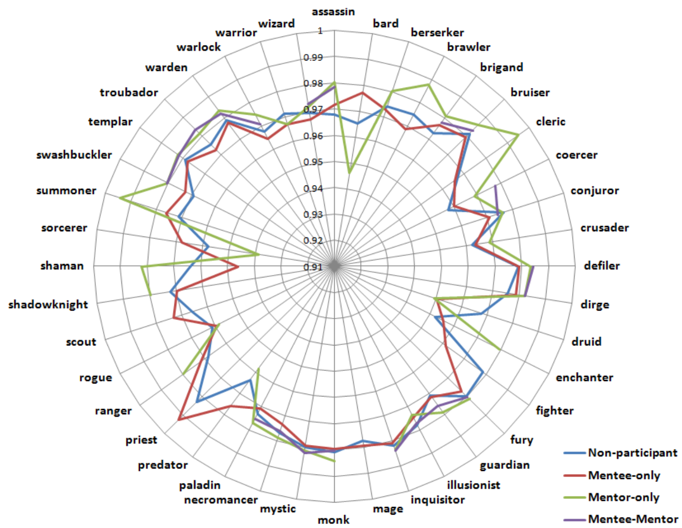


Figure 4.14: Comparison of Combat Success Index by Class

We reported earlier that non-participating players achieve lower Combat Success Index compared to those that participate in mentoring. Figure 4.14 shows that that finding is true across most of the 40 classes. Participating in mentoring leads to increased Combat Success.

4.5 Relationship Between Past Mentoring and Apprenticeship Behavior and Future Behavior

In this section, we examine the relationship between past mentoring or apprenticeship behavior and future behavior. Are players that mentored in the past more likely to mentor again in the future? Are players that apprenticed in the past more likely to become mentors in the future? In the following analysis, we define 'past' as the immediate previous player level and 'future' as the current player level. For instance, for a given player at Level 25, we want to examine how his mentoring/apprenticeship behavior at Level 24 correlates with his mentoring/apprenticeship behavior at Level 25. Is past mentoring behavior indicative of future mentoring behavior?

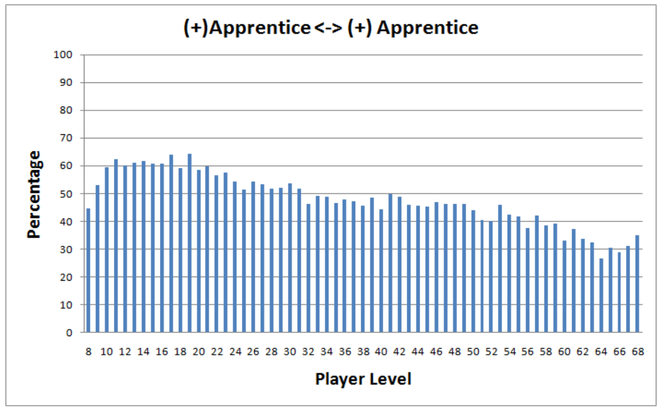


Figure 4.15: Apprenticed in the past, Apprentice in the future

Figure 4.15 shows what percentage of players who apprenticed in the past ended up apprenticing again in the future. Over 47% of the past apprentices continue to apprentice in the future.

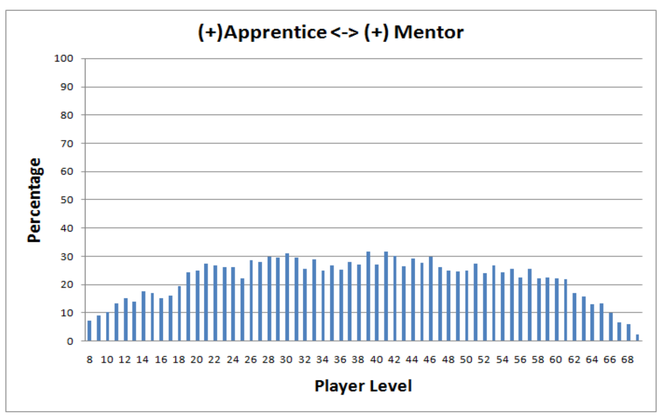


Figure 4.16: Apprenticed in the past, Mentor in the future

Figure 4.16 shows what percentage of players who apprenticed in the past ended up mentoring in the future. Only overall 22% of the past apprentices become mentors.

Figure 4.17 shows what percentage of players who mentored in the past ended up mentoring again in the future. Overall, over 63% of past mentors continue to mentor in

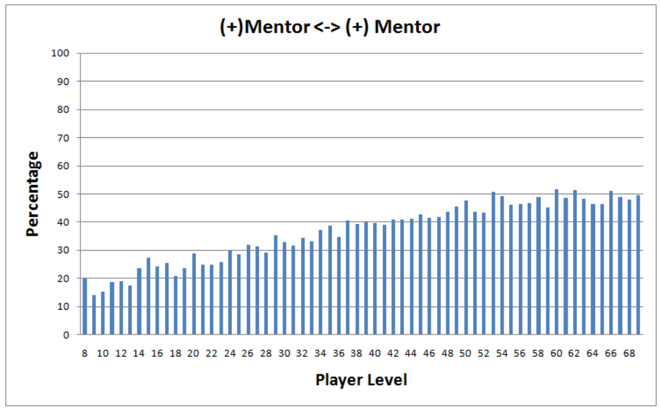


Figure 4.17: Mentored in the past, Mentor in the future

the future.

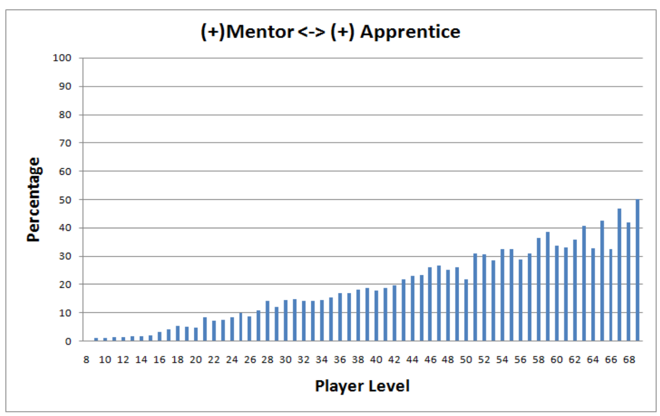


Figure 4.18: Mentored in the past, Apprentice in the future

Figure 4.18 shows what percentage of players who mentored in the past ended up apprenticing in the future. Overall, less than 20% of past mentors apprentice in the future.

Next, for each of the four cases, we examine the correlation between the past behavior and the future behavior. We define 'behavior' as the proportion of tasks completed at each player level either as a mentor or as an apprentice. Next, we perform linear regression and we use the slope as a correlation indicator. Figure 4.19

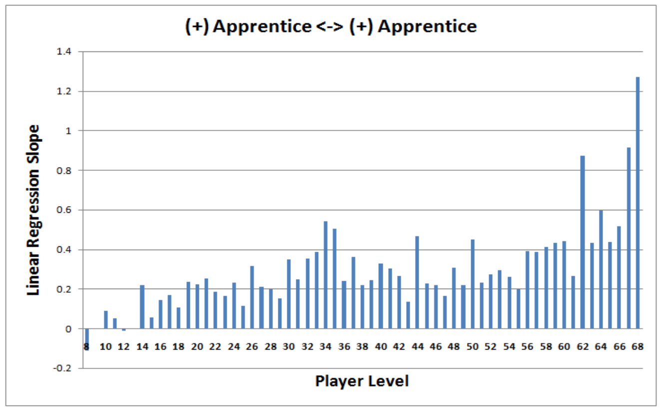


Figure 4.19: Correlation between Past Apprenticing and Future Apprenticing

shows that with the exception of some low player levels, in most player levels, the past apprenticing behavior is positively correlated with the future apprenticing behavior. As the player level increases, the correlation becomes stronger.

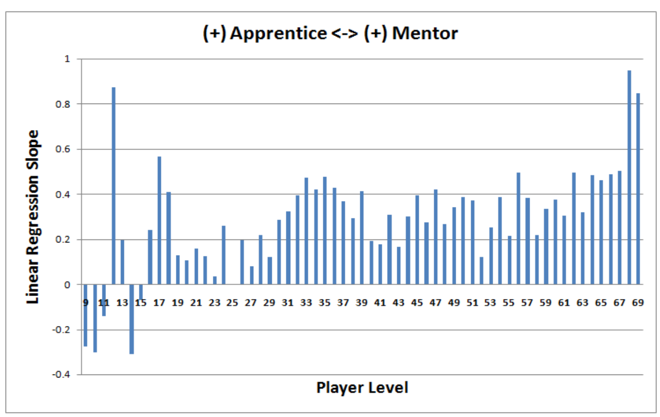


Figure 4.20: Correlation between Past Apprenticing and Future Mentoring

Figure 4.20 shows that with the exception of some low player levels, in most player levels, the past apprenticing behavior is positively correlated with the future mentoring behavior.

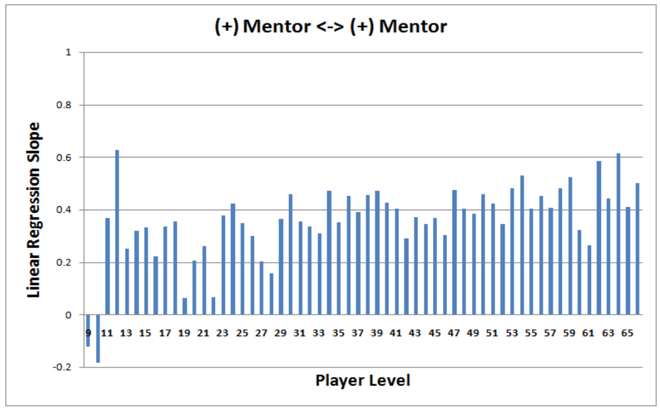


Figure 4.21: Correlation between Past Mentoring and Future Mentoring

Figure 4.21 shows that with the exception of some low player levels, in most player levels, the past mentoring behavior is positively correlated with the future mentoring behavior.

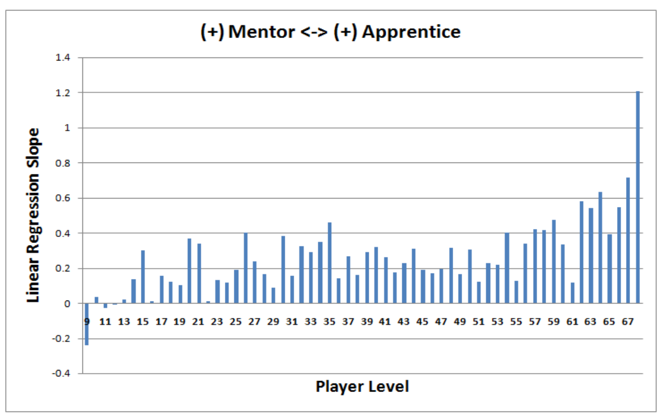


Figure 4.22: Correlation between Past Mentoring and Future Apprenticing

Figure 4.22 shows that with the exception of some low player levels, in most player levels, the past mentoring behavior is positively correlated with the future apprenticing behavior.

4.6 Effect of Mentoring on Future Player Performance

In this section, we examine the effect of past mentoring/apprenticeship behavior on future player performance. Here, we use Combat Efficiency Index as a performance measure. What happens when a player quits apprenticeship and goes solo in the future? How does this affect his combat efficiency? Similarly, what happens when a player newly picks up on apprenticeship and starts getting mentored. How does this affect his performance? In the following analysis, we define 'past' as the immediate previous player level and 'future' as the current player level.

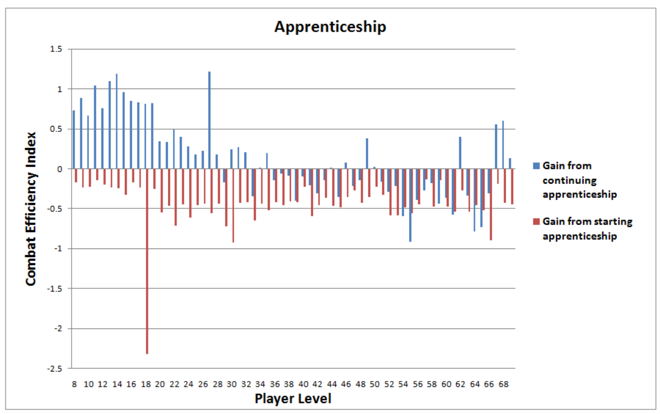


Figure 4.23: Apprenticeship: Combat Efficiency Gain/Loss

Figure 4.23 shows that in low player levels, it is beneficial (in terms of Combat Efficiency) for past apprentices to continue to apprentice in the next level. This effect starts minimizing at Player Level 14 and with a few exceptions, it is no longer beneficial to continue. The figure also shows that when players, who did not apprentice in the past, apprentice in the future, it leads to Combat Efficiency loss.

Figure 4.24 shows that for past mentors, it is beneficial (in terms of Combat Efficiency) to discontinue mentoring in the next level. Also for those players that have not mentored in the past, it is beneficial (in terms of Combat Efficiency) to not mentor in the next level.

In summary, our findings indicate that past mentoring and apprenticeship activities can affect future performance. Hence, it is important, when modeling player's past

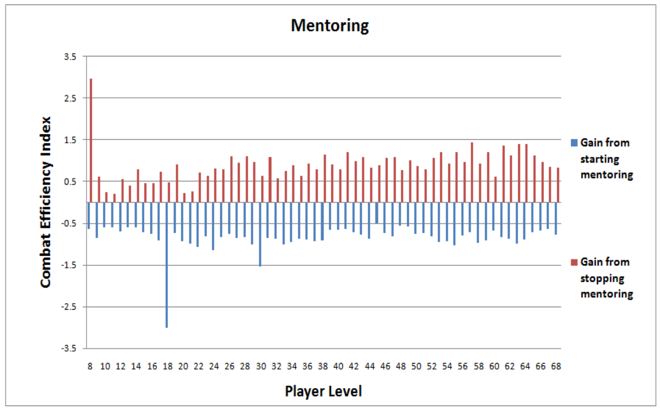


Figure 4.24: Mentoring: Combat Efficiency Gain/Loss

behaviors for the purposes of predicting his or her future behavior, the modeling techniques take into account past mentoring and apprenticeship activities. The next chapter incorporates these findings into future performance prediction models.

4.7 Effect of Diversity in Mentoring Network

In this section, we investigate and report findings on social diversity in mentoring network in EverQuest II. We examine three different game servers from the EverQuest II game logs. In all three servers, the results from our analyses suggest that increase in social diversity in terms of characters and classes encountered moderately negatively correlates with player performance.

4.7.1 Diversity in Mentoring Network

Character Social Diversity

For each player, we compute social diversity score with respect to different characters he/she socializes with in mentoring network. We term this "Character Diversity," and it is computed as the following:

$$log_2 \frac{(\text{\# unique apprentices})}{(\text{\# mentoring sessions})}$$

Likewise, for apprentices, we compute Character Social Diversity score as:

$$log_2 \frac{(\text{\# unique mentors})}{(\text{\# apprenticing sessions})}$$

Class Social Diversity

For each player, we compute social diversity score with respect to different classes [56] he/she socializes with in mentoring network. We term this "Class Diversity," and it is computed as the following:

$$log_2 \frac{(\text{\# unique apprentice classes})}{(\text{\# mentoring sessions})}$$

Likewise, for apprentices, we compute Class Social Diversity score as:

$$log_2 \frac{(\text{\# unique mentor classes})}{(\text{\# apprenticing sessions})}$$

4.7.2 Player Performance and Social Diversity

First, we report our findings on the relationship between player performance and social diversity. For 'Guk' server, we report Adjusted R-squared and P-value below: Figure 4.25 shows that player performance is moderately negatively correlated with Character Diversity. As the Character Diversity value decreases, player performance moderately increases.

Figure 4.26 shows that player performance is moderately negatively correlated with Class Diversity. As the Class Diversity value decreases, player performance moderately increases.

Figure 4.27 and Figure 4.28 are showing a similar trend of moderate correlation.

For 'Antonia Bayle' server, we report Adjusted R-squared and P-value below:

For 'Nagafen' server, we report Adjusted R-squared and P-value below:

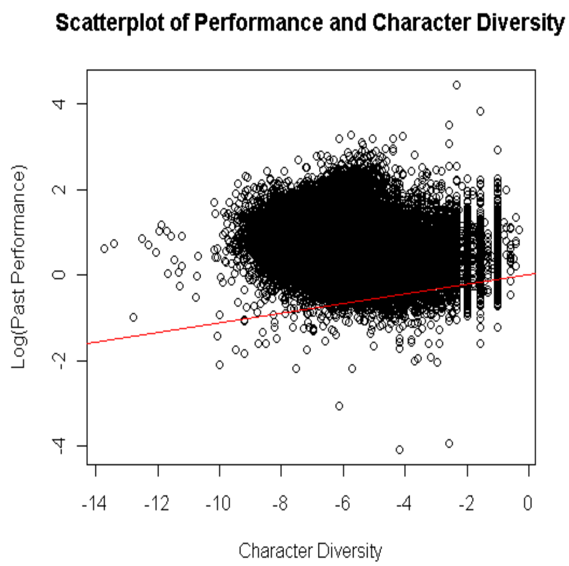


Figure 4.25: Correlation between Player Performance and Character Diversity ('Guk' server, mentors), Adjusted R-squared: 0.538, P-value < 2.2e-16

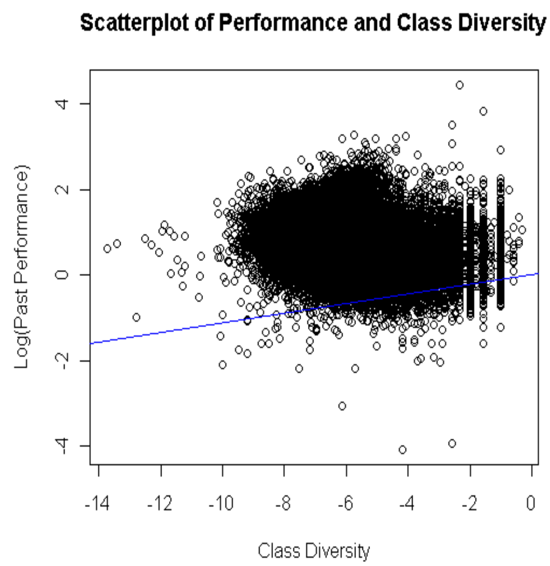


Figure 4.26: Correlation between Player Performance and Class Diversity ('Guk' server, mentors) Adjusted R-squared: 0.5387, P-value < 2.2e-16

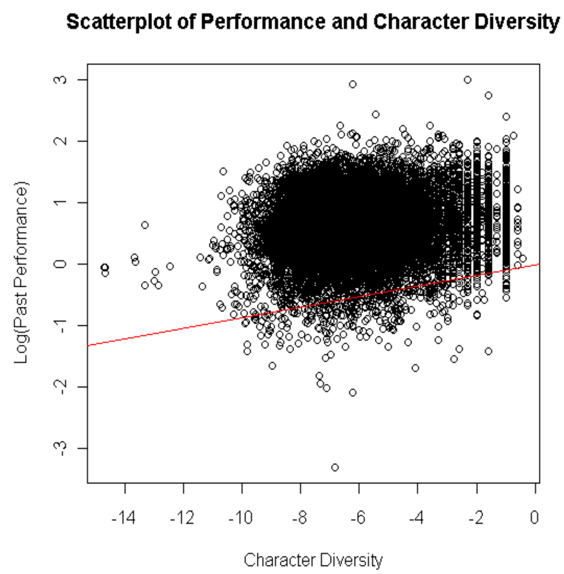


Figure 4.27: Correlation between Player Performance and Character Diversity ('Guk' server, apprentices), Adjusted R-squared: 0.4367, P-value < 2.2e-16

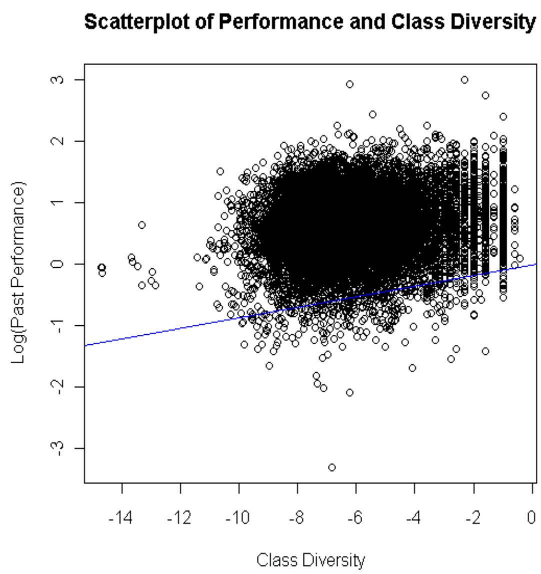


Figure 4.28: Correlation between Player Performance and Class Diversity ('Guk' server, apprentices), Adjusted R-squared: 0.5387, P-value < 2.2e-16

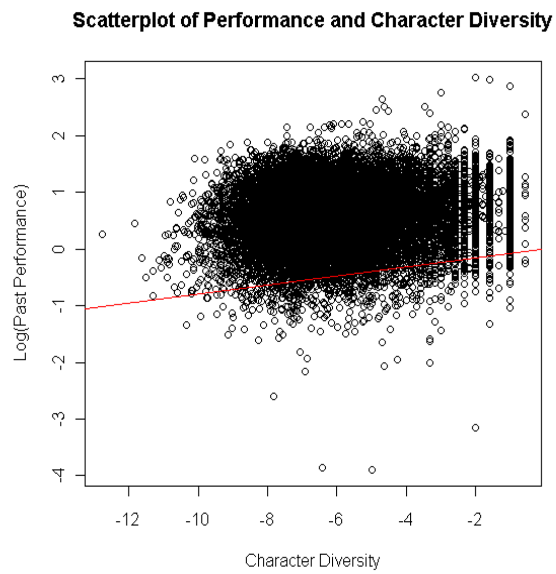


Figure 4.29: Correlation between Player Performance and Character Diversity ('Antonia Bayle' server, mentors), Adjusted R-squared: 0.394, P-value < 2.2e-16

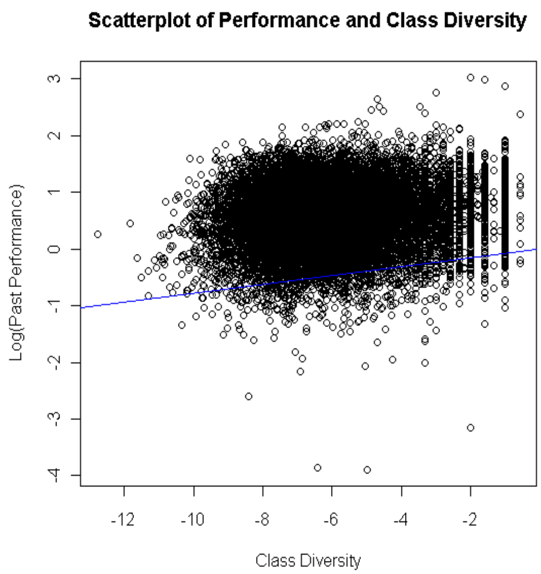


Figure 4.30: Correlation between Player Performance and Class Diversity ('Antonia Bayle' server, mentors) Adjusted R-squared: 0.3934, P-value < 2.2e-16

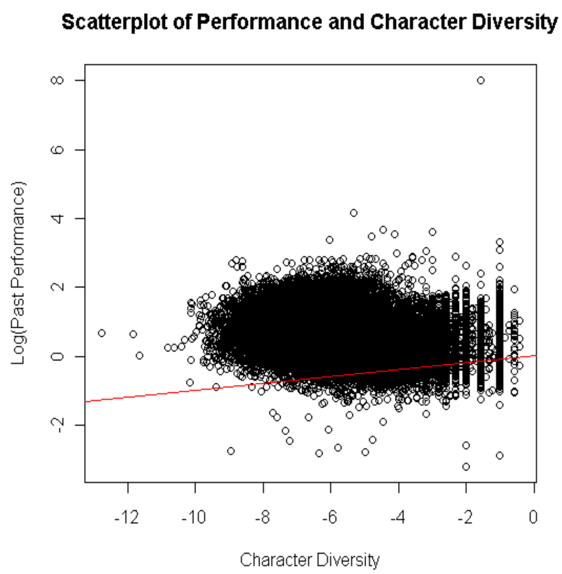


Figure 4.31: Correlation between Player Performance and Character Diversity ('Antonia Bayle' server, apprentices), Adjusted R-squared: 0.4915, P-value $< 2.2e\text{-}16$

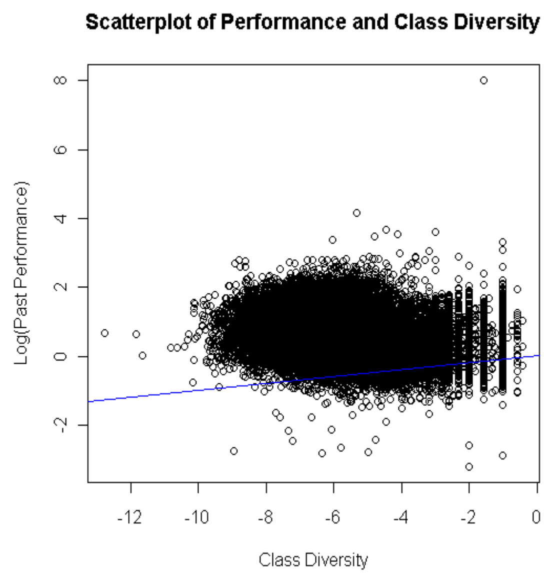


Figure 4.32: Correlation between Player Performance and Class Diversity ('Antonia Bayle' server, apprentices), Adjusted R-squared: 0.4923, P-value < 2.2e-16

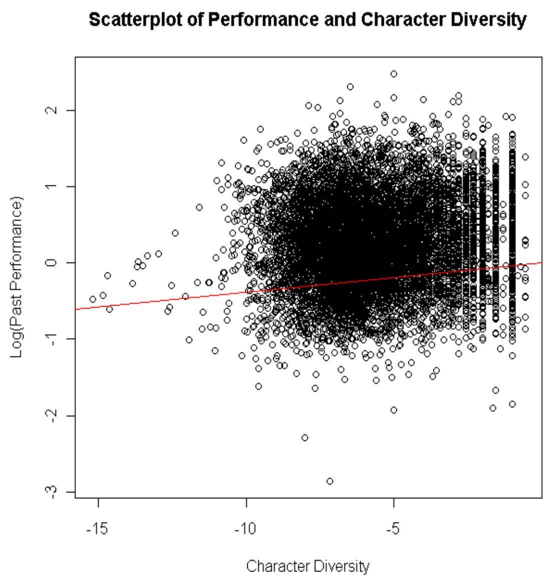


Figure 4.33: Correlation between Player Performance and Character Diversity ('Nagafen' server, mentors), Adjusted R-squared: 0.4663, P-value $< 2.2e-16$

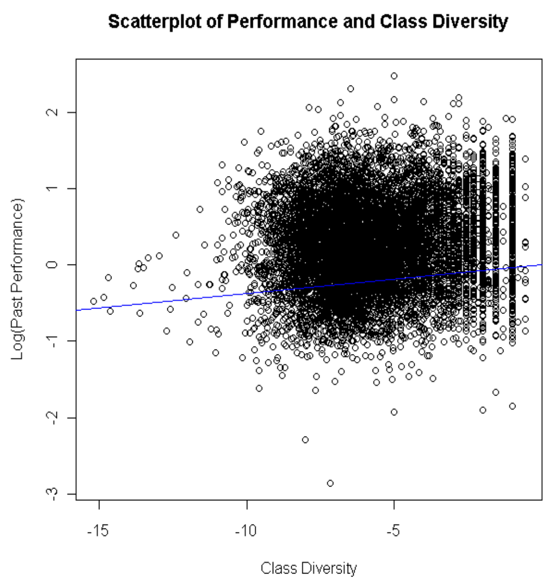


Figure 4.34: Correlation between Player Performance and Class Diversity ('Nagafen' server, mentors) Adjusted R-squared: 0.4662, P-value < 2.2e-16

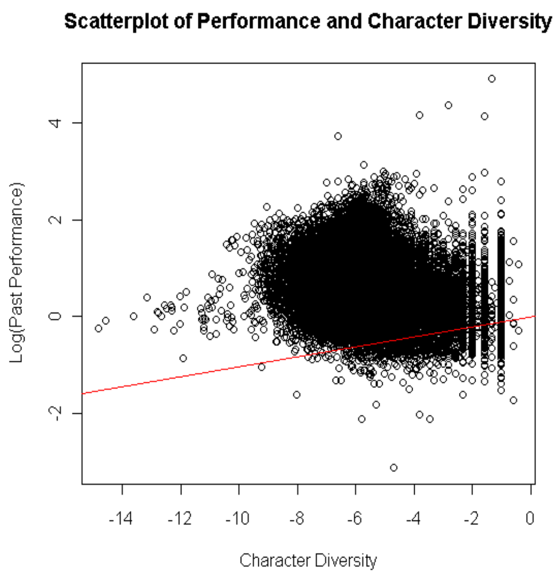


Figure 4.35: Correlation between Player Performance and Character Diversity ('Nagafen' server, apprentices), Adjusted R-squared: 0.1217, P-value < 2.2e-16

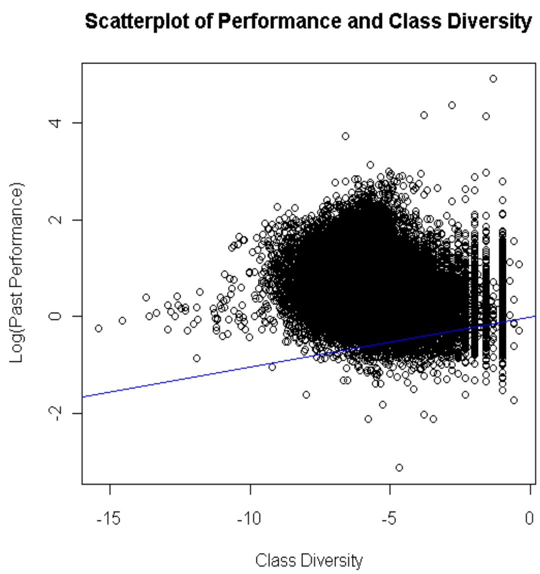


Figure 4.36: Correlation between Player Performance and Class Diversity ('Nagafen' server, apprentices), Adjusted R-squared: 0.1215, P-value < 2.2e-16

In summary, our findings suggest that when it comes to mentoring and apprenticing, it is beneficial for a given player to interact with focused few rather than interacting with many. Also, our findings indicate that when players interact with many different types of character class, their performance decreases. The reasoning behind this is because by game design (as discussed in Chapter 2 and also in Appendix C), there are largely four different types of characters and each type is equipped with different capabilities such as armor types, etc. For instance, players of fighter or warrior type would benefit most from interacting with similar classes. From modeling perspective, it is important, when modeling player's past behaviors for the purposes of predicting his or her future behavior, the modeling techniques take into account Character Diversity and Class Diversity with respect to mentoring and apprenticeship. The next chapter incorporates these findings into future performance prediction models.

4.8 Conclusion

In this study, we examine the effects of mentoring on player performance in EverQuest II. As shown in Figure 4.4, participation rate is very low across all player levels. This brings up an interesting question; how can the game attract more players to participate in mentoring, as a mentor and as an apprentice. Our findings show that participating in mentoring as a mentor may not appeal to achievement-oriented players whose primary objective is to achieve XP points as fast as possible and move up to higher levels fast. Similarly for apprentices, as Figure 4.11 shows, until Player Level 25 or so, it is beneficial to participate in mentoring as an apprentice in terms of Combat Efficiency. However, as player level increases, its effect diminishes. In summary, our findings indicate that past mentoring and apprenticeship activities can affect future performance. Hence, it is important, when modeling player's past behaviors for the purposes of predicting his or her future behavior, the modeling techniques take into account past mentoring and apprenticeship activities. The next chapter incorporates these findings into future performance prediction models.

In summary, our findings suggest that when it comes to mentoring and apprenticing, it is beneficial for a given player to interact with focused few rather than interacting with many. Also, our findings indicate that when players interact with many different types of character class, their performance decreases. The reasoning behind this is because by game design (as discussed in Chapter 2 and also in Appendix C), there are largely four different types of characters and each type is equipped with different capabilities such as armor types, etc. For instance, players of fighter or warrior type would benefit most from interacting with similar classes. From modeling perspective, it is important, when modeling player's past behaviors for the purposes of predicting his or her future behavior, the modeling techniques take into account Character Diversity and Class Diversity with respect to mentoring and apprenticeship. The next chapter incorporates these findings into future performance prediction models.

Chapter 5

Player Performance: Prediction Models

Earlier chapters examined player performance metrics and various factors affecting player performance. First, we examined how player's past performance can indicate his or her future performance. We showed that player's past performance, up to a certain level into the past, can be used to predict his or her future performance. Additionally, we showed that the predictive power decays with each increasing distance into the past, hence, we proposed decaying weight function for modeling player's past performance. Second, we examined an important aspect of game play in MMOGs, namely, mentoring and apprenticeship. This newly introduced game feature became widely popular across almost all MMOGs in the gaming community over the last few years, yet, no systematic data-driven studies were done previously that analyze its relationship with and impact on player performance. We showed in the previous chapter that past mentoring and apprenticeship activities can affect future performance. We also showed that social diversity can also impact player performance.

Overall, we propose a comprehensive performance management tool for measuring and reporting operational activities of game players. The prediction models we build in this chapter provide a projection of player's future performance based on his past performance and past activities both at the individual level (solo playing) and social (group playing including mentoring and apprenticing). While many games today provide in-game "how to started" guides to help newcomers ramp up quickly in the early stage of the game as well as in-game assistants throughout the game to help identify tasks to perform to gain rewards, it lacks accurate player performance prediction systems which can model not only player's past performance but social interactions which can influence player performance. In this chapter, we develop and evaluate player performance prediction models based on player's past performance and social diversity in mentoring network. Our findings provide a foundation for a customized performance management system and mentor/apprentice recommendation system during game play where its primary objective is to evaluate and suggest mentoring-based social interactions in order to optimize player performance.

In this chapter, we introduce a family of individual player performance prediction models, namely MARCEL-1, MARCEL-2, Adjusted-MARCEL-2, LinRegress-1, Neighbor-Weight-1 and Neighbor-Weight-2. The first four are variations of PECOTA [73] and MARCEL [76], two most popular baseball home run prediction methods. Next, we propose two novel individual player performance prediction models, namely Neighbor-Weight-1 and Neighbor-Weight-2. In the former model, the future performance prediction for each game player is solely dependent upon their past performance. In the latter model, the feature representation includes more behavioral attributes of high granularity. The future performance prediction for each game player is dependent upon their past activities, some of which are solo activities while others are group activities such as apprenticing and mentoring. We evaluate and compare these methods and show that Neighbor-Weight methods outperform other existing methods.

5.1 Objective

In this chapter, we introduce a family of individual player performance prediction models, namely MARCEL-1, MARCEL-2, Adjusted-MARCEL-2, LinRegress-1, Neighbor-Weight-1 and Neighbor-Weight-2. The first four are variations of PECOTA [73] and MARCEL [76], two most popular baseball home run prediction methods. Next, we propose two novel individual player performance prediction models, namely Neighbor-Weight-1 and Neighbor-Weight-2. In the former model, the future performance prediction for each game player is solely dependent upon their past performance. In the latter model, the feature representation includes more behavioral attributes of high granularity. The future performance prediction for each game player is dependent upon their past activities, some of which are solo activities while others are group activities such as apprenticing and mentoring. We evaluate and compare these methods and show that Neighbor-Weight methods outperform other existing methods.

5.2 Player Performance Prediction - Existing Methods

5.2.1 Baseball Home Run Prediction

Prediction of future performance of humans has long been studied in various disciplines over the years. Most notably, it has been well studied in sports. Baseball has a long history of record keeping and statistical analyses that dates back to the nineteenth century. Batting average, RBIs, and home runs are some of the many statistics being kept track of today. There exists an enormous amount of public and private interest in the projection of future performance. Major league teams rely on the past statistics of a given player in deciding whether to acquire him or not and for how many seasons under the assumption that his past success is a good indicator of his future success. PECOTA [73] and MARCEL [76] are widely known methods in baseball home run prediction.

PECOTA [73] is considered a very sophisticated method for home run prediction in baseball. For a given ball player at the age of X, the method uses a nearest neighbor analysis of both minor and major league players from the past that exhibited similar performance at age X. It uses the historical performance of these past players to predict the given player's future performance. MARCEL [76] uses data from the three immediate past seasons of a given ball player, and it assigns more weight to more recent seasons. One drawback of this approach is that prediction models solely based on individual players cannot be generalized to the global population. A variation of the MARCEL approach attempts to regress predictions to the global population mean. One drawback of this approach is that prediction models built on the global population can become too coarse.

5.2.2 Using Home Run Prediction Methods for Game Player Performance Prediction

We consider game player levels in EverQuest II similar to seasons in baseball. Players perform tasks, gain points, and move up to the next level as ball players would attain different types of achievement (i.e. home runs, single, double, triple hits, run batted in, etc.) in each season and proceed to the next season. Unlike in baseball where there is not necessarily a fixed number of home runs, triples, doubles, etc. required to move to the next season, EverQuest II employs a point scaling system where there exists a fixed number of experience points at each level in order to move up to the next level. Because the ding point is a fixed constant, we measure a game player's total play time at each level and uses it as a performance measure in this study.

5.3 Player Performance Prediction in MMOGs

In this study, we develop performance prediction models for game players in EverQuest II. The objective is to predict a given player's play time at level i, a future state, based on his past performance at levels $i-1$, $i-2$, and so forth, where performance at any level is measured as the total play time spent at that level. Play time in EverQuest II excludes any idle periods where being idle is defined as any contiguous time blocks of 30 minutes or beyond.

Our contributions are two-fold. First, we develop novel player performance techniques, namely Neighbor-Weight-1 and Neighbor-Weight-2, that outperform the two existing methods from baseball home run prediction problem. Second, we incorporate high granularity operational data about mentoring and apprenticeship into feature representation and show that inclusion of these high granularity data about players' social activities leads to improved prediction accuracy.

5.4 Dataset

The dataset used in this study is described in Chapter 2. One additional filtering criterion exists. First, our dataset is a snapshot from January 1, 2006 through September 11, 2006. Hence, for Player Level 1 (minimum level) and Player Level 70 (maximum

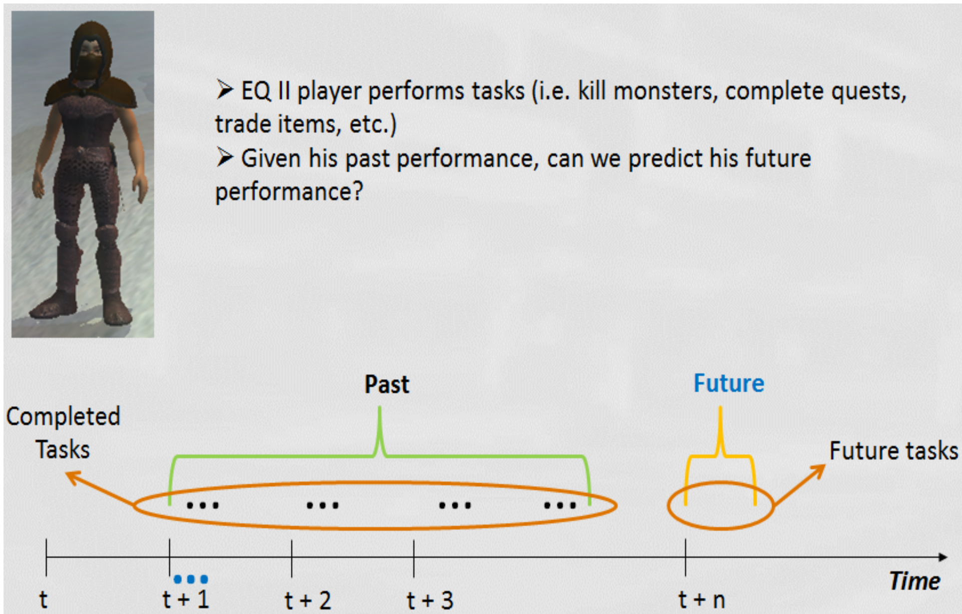


Figure 5.1: Player Performance Prediction

level), we cannot assume that we have complete information about player performance data pertaining to these two levels. Hence, we exclude these two levels. Second, our experiments seek to use past three levels worth of performance data as training data for future performance prediction. Hence, future performance prediction is performed for Player Levels 5 through 69, where for the former, levels 2 through 4 serve as the ”past” performance data which constitute the training dataset for prediction models. Additionally, we perform sampling based on standard deviation. First, we take all the data points at each player level and transform it using logarithm in order to convert our data into a normal distribution (near normal). Second, we filter in all data points within two standard deviations. We convert these values back into their raw format and these constitute the final dataset used in this experiment. The reasoning behind this type of sampling is that some of the existing algorithms (MARCEL variations as well as Linear Regression) are sensitive to outliers and thus is our evaluation metric. We further describe how a future extension of this research seeks to explore outlier detection in the context of player performance, in Chapter 11. Below, we show the distributions (after sampling) of player performance metrics, Efficiency Index and Success Index, across three game servers.

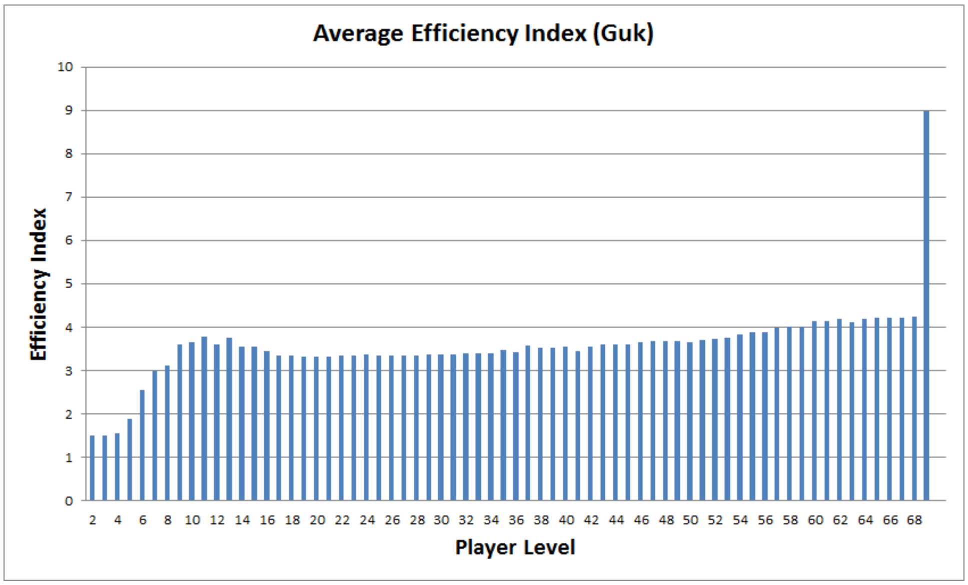


Figure 5.2: Efficiency Index - Guk

5.5 Player Performance Prediction Methods

In this chapter, we introduce a family of individual player performance prediction models, namely MARCEL-1, MARCEL-2, Adjusted-MARCEL-2, LinRegress-1, Neighbor-Weight-1 and Neighbor-Weight-2.

5.5.1 MARCEL-1

MARCEL-1 works in the following manner. For each player, it looks at his performance at a previous level, say $i-1$. It then takes the global population average at level $i-1$. Next, it sees how far away this players performance was from the global population average at level $i-1$. We call this distance δ. The method then takes the global population average at level i. The prediction then becomes this global population average at level i plus the δ value computed previously. In Decaying-Weight scheme, it

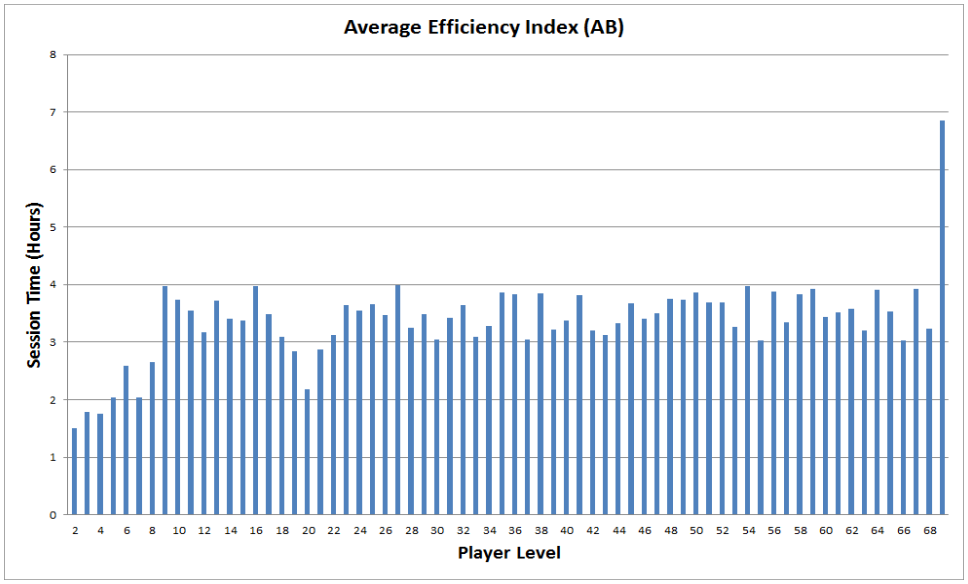


Figure 5.3: Efficiency Index - Antonia Bayle

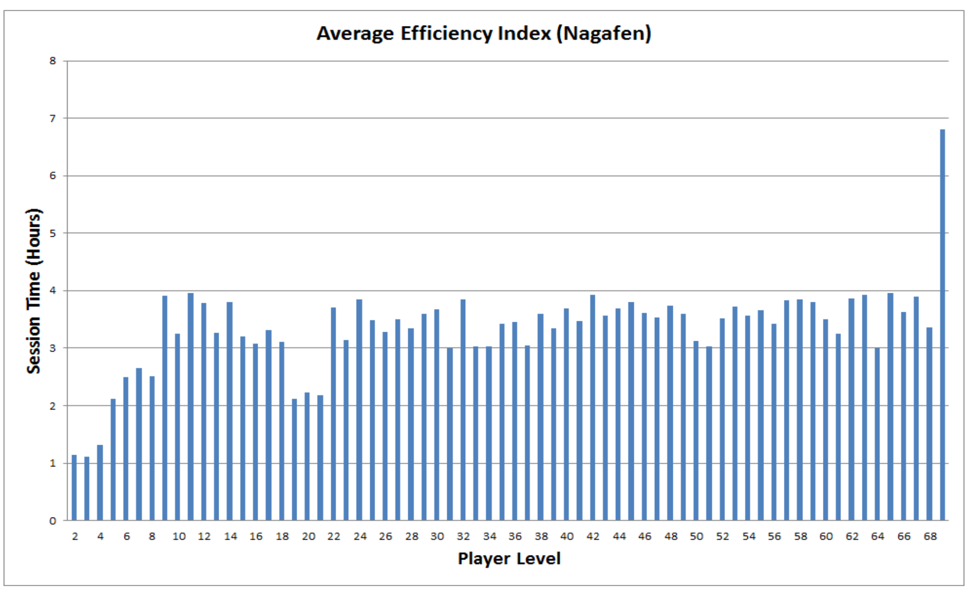


Figure 5.4: Efficiency Index - Nagafen

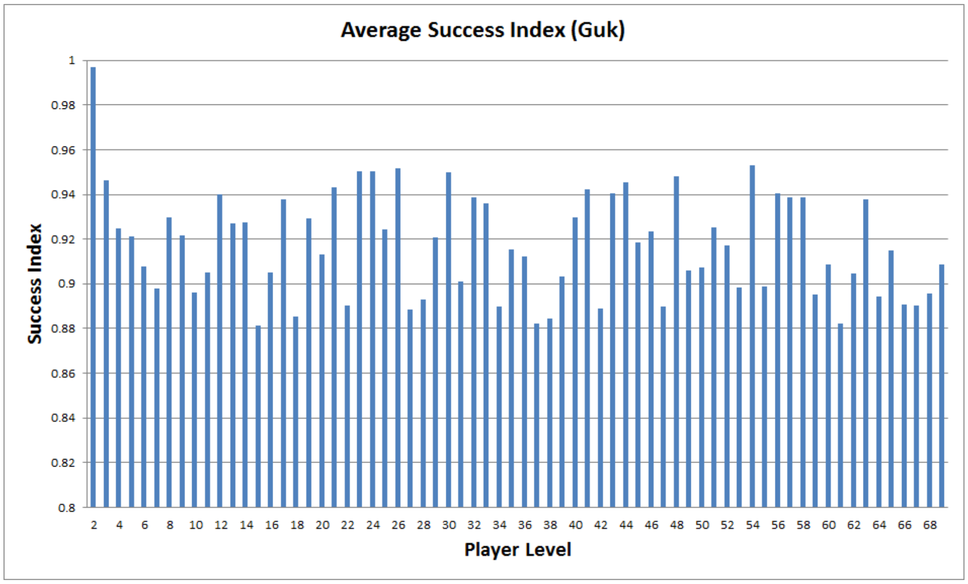


Figure 5.5: Success Index - Guk

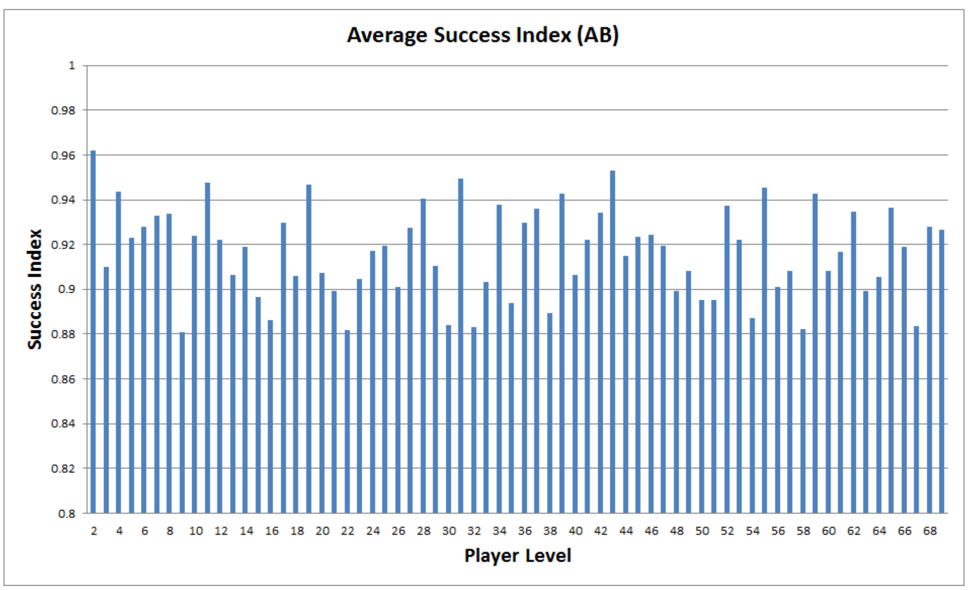


Figure 5.6: Success Index - Antonia Bayle

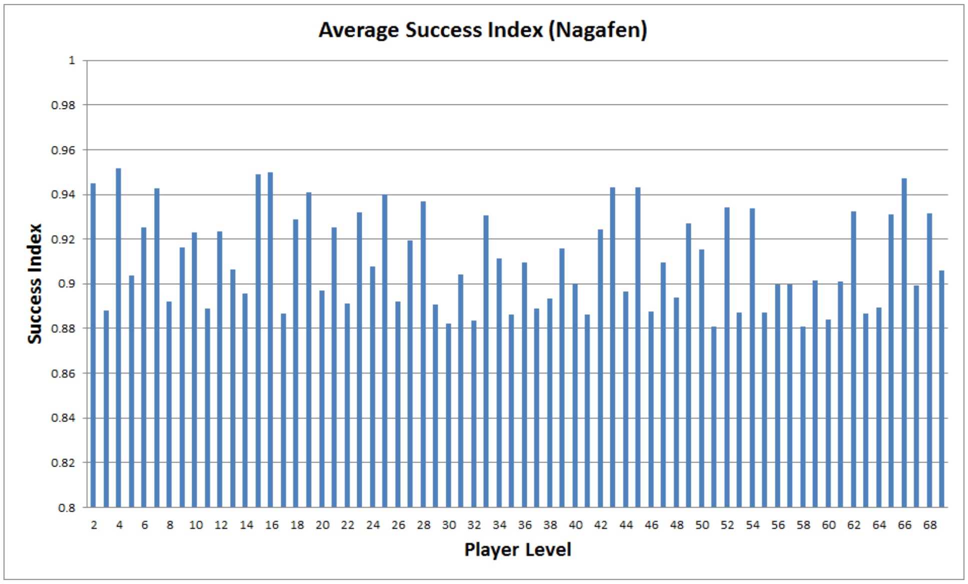


Figure 5.7: Success Index - Nagafen

gives the most weight to the most recent level. In Even-Weight scheme, it gives equal importance to all past levels.

We take the absolute value of the difference between the global population average at each past level and player J's performance because after weighting each such δ, we will add up the differences over multiple past levels. The final prediction then always decrements δ from the global population average at level I. Per our evaluation measure Root-Mean-Squared-Error, whether we increment or decrement this δ from the global population average at level I does not affect the results. Contextually, the final prediction shall then give the prediction range rather than a single numeric value. However, for evaluation purposes we keep the procedure as outlined in Algorithm 1.

5.5.2 MARCEL-2

MARCEL-2 works in the following way. It takes each player, looks at his past performance, takes the average and that becomes the final prediction. In Decaying-Weight scheme, it gives the most weight to the most recent level. In Even-Weight scheme, it gives equal importance to all past levels.

Algorithm 1 MARCEL-1: Calculate Predicted Performance for Player J at Level I

$player\_levels = 70$
$num\_players[]$ (array of player numbers at each level)
$total\_metrics[]$ (array of player metric at each level)
$player\_specific\_metric[][]$ (array of individual player metric at each level)
$avg\_player\_metrics[]$ (temporary array initialized to zero, array of average metric at each level)
for $i = 1$ **to** $player\_levels$ **do**
  $avg\_player\_metrics[i] = total\_metrics[i] \div num\_players[i]$ (array of average metric across all players at each level)
end for
T (predicted metric at level I)
P (number of previous levels)
$weights[]$ (array of weights)
$A = 0, M = 0, N = I - P, temp$ (temporary variables)
while $N < I$ **do**
  $temp = avg\_player\_metrics[N] - player\_specific\_metric[J][N]$ (compute δ at each past level)
  $temp = temp \times weights[M]$
  $A = A + |temp|$
  $N \leftarrow N + 1$
  $M \leftarrow M + 1$
end while
$T = avg\_player\_metrics[I] - A$

Contextually, the final prediction for a given player is the average performance across his or her one or more past levels.

5.5.3 Adjusted-MARCEL-2

This method is only applicable to Efficiency Index prediction. This is a modified version of MARCEL-2, where the final prediction is adjusted per the global population average. Unlike in baseball (where from season to season, the various statistics such as RBI, home runs and their absolute values do not change), in MMOGs, there exists point-scaling systems where as players move from one level to the next, expectation increases. It gradually becomes more difficult to move up to the next level. Our analysis in Chapter 3 has found an interesting phenomenon, where in fact, it is in the lower levels that it is difficult to move up and in higher levels its easier to move up than what

Algorithm 2 MARCEL-2: Calculate Predicted Performance for Player J at Level I

$player\_levels = 70$
T (predicted player metric at level I)
P (number of previous levels)
$player\_specific\_metric[][]$ (array of individual player metric at each level)
$weights[]$ (array of weights)
$A = 0, M = 0, N = I - P, temp$ (temporary variables)
while $N < I$ **do**
  $temp = player\_specific\_metric[J][N] \times weights[M]$
  $A = A + |temp|$
  $N \leftarrow N + 1$
  $M \leftarrow M + 1$
end while
$T = A$

the game company intended. Thus, the game's point-scaling system is not reflective of actual player behavior. Hence, we use the novel point-scaling system suggested in Chapter 3. Due to the fact that performance expectation in terms of XP points varies from one level to the next, we adjust the predicted future performance score in the following manner.

Contextually, the final prediction for a given player is the average performance across his or her one or more past levels while the performance at each past level is adjusted per the expectation at level I.

5.5.4 LinRegress-1

LinRegress-1 performs linear regression and predicts the future performance of a given player. For different weighting schemes, we were only able to perform even-weight scheme because there was no straightforward way of applying weight using linear regression. In this case, we perform multiple linear regression with multiple independent variables; performance metrics from multiple past levels.

Our preliminary data analysis of the game data reports that play times at each player level exhibit a skewed distribution. Figure 5.8 shows the distribution of level 15 players by their session times.

Since linear regression assumes linearity of the relationship between dependent and

Algorithm 3 Adjusted-MARCEL-2: Calculate Predicted Performance for Player J at Level I

$player\_levels = 70$
T (predicted player metric at level I)
P (number of previous levels)
$player\_specific\_metric[][]$ (array of individual player metric at each level)
$num\_players[]$ (array of player numbers at each level)
$total\_metrics[]$ (array of player metric at each level)
$avg\_player\_metrics[]$ (temporary array initialized to zero, array of average metric at each level)
for $i = 1$ **to** $player\_levels$ **do**
  $avg\_player\_metrics[i] = total\_metrics[i] \div num\_players[i]$ (array of average metric across all players at each level)
end for
$weights[]$ (array of weights)
$A = 0, M = 0, N = I - P, temp$ (temporary variables)
while $N < I$ **do**
  $temp = player\_specific\_metric[J][N] \times weights[M] \times (\frac{avg\_player\_metrics[I]}{avg\_player\_metrics[N]})$
  $A = A + |temp|$
  $N \leftarrow N + 1$
  $M \leftarrow M + 1$
end while
$T = A$

independent variables and assumes that variables have normal distributions, we transformed our dependent and independent variable(s) by taking the logarithm of each (i.e. power model) to achieve linearity. Hence, we modify Algorithm 4 in such a way that the regression equation looks like the following:

$$log(y) = b_0 + b_1 log(x_1) + b_2 log(x_2) + ... + b_n log(x_n)$$

In order to compute the error metric (i.e. Root-Mean-Squared-Error), at the end of the procedure, we perform back transformation to restore the dependent variable to its original and hence into its non-transformed measurement scale. This way, we can compare the prediction results with other algorithms apple-to-apple. Hence, the predicted value (future player performance) is computed in the following manner:

Algorithm 4 LinRegress-1: Calculate Predicted Performance for Player J at Level I

T (predicted player metric at level I)
P (number of previous levels)
$(Slope[], Intercept) = LinearRegressionTrain(total\_metrics, I, P)$
$temp = 0$ (temporary variable)
for $i = 1$ **to** P **do**
  $temp = temp + (Slope[i] \times player\_specific\_metric[J][I - i])$
end for
$temp = temp + Intercept$
$T = temp$

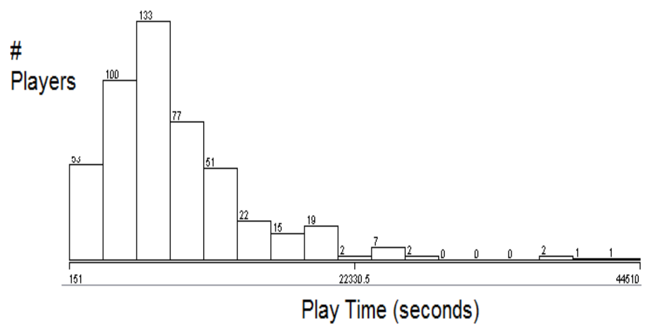


Figure 5.8: Distribution of Level 15 Players by Play Time (March, 2006)

$$\hat{y} = (10 \times b_0) + b_1 log(x_1) + b_2 log(x_2) + ... + b_n log(x_n)$$

And the modified LinRegress-1 algorithm is shown below:

5.5.5 Neighbor-based Method

As demonstrated in [73], a neighbor search-based method works in the following manner.

The Neighbor-based method uses a specified distance function to find K nearest neighbors, where "nearest" is defined as having shared similar past performance patterns as the given player J. The intuition behind this approach is that players that have similar past performance patterns are likely to behave in a similar manner in the future. Hence, unlike in LinRegress-1 where a single linear regression equation is derived from

Algorithm 5 Modified-LinRegress-1: Calculate Predicted Performance for Player J at Level I

T' (logarithm of predicted player metric at level I)
P (number of previous levels)
$(Slope[], Intercept) = LinearRegressionTrain(total\_metrics, I, P)$
$temp = 0$ (temporary variable)
for $i = 1$ **to** P **do**
  $temp = temp + (Slope[i] \times log(player\_specific\_metric[J][I-i]))$
end for
$temp = temp + (Intercept \times 10)$
$T = temp$

Algorithm 6 Neighbor-based: Calculate Predicted Performance for Player J at Level I, given K

T (predicted player metric at level I)
P (number of previous levels)
$total\_metrics[]$ (array of player metric at each level)
$player\_specific\_metric[][]$ (array of individual player metric at each level)
$Y[]$ (array of k nearest neighboring players)
$Y[] = NearestNeighborSearch(K, Euclidean, total\_metrics, J, I, P)$
$temp = 0$ (temporary variable)
for $i = 1$ **to** K **do**
  $temp = temp + player\_specific\_metric[Y[i]][I]$
end for
$temp = temp/K$ (average metric across K neighbors)
$T = temp$

the data about the global population, Neighbor-based method will select K players who share similar past performance patterns with player J. Once these K neighbors are identified, the prediction (future performance of player J) is computed as the average performance across all K players at the future level I.

The Euclidean distance function used in Algorithm 6 is formulated as below:

Given two points in Euclidean m-space, $\underset{\sim}{p}$ and $\underset{\sim}{q}$, where $\underset{\sim}{p} = (p_1, p_2, ..., p_m)$ and $\underset{\sim}{q} = (q_1, q_2, ..., q_m)$, the distance between the two points is given by:

$$d(p, q) = d(q, p) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + ... + (q_m - p_m)^2}$$

where
p_1 = Player P's performance metric at level $(I-1)$
p_2 = Player P's performance metric at level $(I-2)$
and so on.

5.5.6 Our Method - Neighbor-Weight-1

We propose a novel neighbor-based method as shown in Algorithm 7. Our earlier analysis in Chapter 3 showed that the predictive power deteriorated as we go further into the past of a given player. Hence, we introduce a weight assignment scheme into the existing Neighbor-based method.

Our novel weighting-based distance function ($WeightFunction$) used in Algorithm 7 is formulated as below:

Given two points in Euclidean m-space, $\underline{p}$ and $\underline{q}$, where $\underline{p} = (p_1, p_2, ..., p_m)$ and $\underline{q} = (q_1, q_2, ..., q_m)$, the distance between the two points is given by:

$$d(p,q) = d(q,p) = \sqrt{U_1 \cdot (q_1 - p_1)^2 + U_2 \cdot (q_2 - p_2)^2 + ... + U_m \cdot (q_m - p_m)^2}$$

where
p_1 = Player P's performance metric at level $(I-1)$
p_2 = Player P's performance metric at level $(I-2)$
and so on, and

U represents weights given to each past performance metric, such that:
$U_1 = U_2 ... = U_3$ in Even-Weight scheme and
$U_1 > U_2 ... > U_3$ in Decaying-Weight scheme.

Next, based on the nearest-neighbor distance computed, top K neighbors are selected such that shorter the distance, the closer and the more similar player q is to player p. We use the distance measures directly to compute the predicted future performance in the following manner:

$$Prediction_{i,I} = \sum_{j=1}^{k} \frac{(\sum_{l=1}^{k} d_{i,l}) - d_{i,j}}{\sum_{l=1}^{k} d_{i,l}} \times PerformanceMetric_{j,I}$$

The intuition behind this formula is that we take the inverse of the distance measure and use that directly to assign weight to the neighbor in question. Hence, the most weight is given to the neighbor whose distance from player J is the shortest.

Given $WeightFunction$ and the above formula for averaging up the performance metrics of K neighbors per the distance measure computed by $WeightFunction$, the Neighbor-Weight-1 method is shown in Algorithm 7.

Algorithm 7 Neighbor-Weight-1: Calculate Predicted Performance for Player J at Level I, given K

T (predicted player metric at level I)
P (number of previous levels)
$total\_metrics[]$ (array of player metric at each level)
$player\_specific\_metric[][]$ (array of individual player metric at each level)
$Y[]$ (array of k nearest neighboring players)
$D[][]$ (array of distance measures between players)
$(Y[], D[]) = NearestNeighborSearch(K, WeightFunction, total\_metrics, J, I, P)$
$TotalDistance = 0$
for $i = 1$ **to** K **do**
  $TotalDistance = TotalDistance + D[J][i]$
end for
$temp = 0$ (temporary variable)
for $i = 1$ **to** K **do**
  $temp+ = ((TotalDistance - D[J][i])/TotalDistance) \times player\_specific\_metric[i][I]$
end for
$T = temp$

5.5.7 Our Method - Neighbor-Weight-2

In Neighbor-Weight-1 method, the future performance prediction for each game player is solely dependent upon their past performance metrics, excluding other social activities such as mentoring. We devise a variation of this method and develop Neighbor-Weight-2, where the feature representation includes more behavioral attributes of high granularity. The future performance prediction for each game player is dependent upon their past activities, some of which are solo activities while others are group activities such as apprenticing and mentoring, whose importance on player performance prediction has been described in Chapter 4.

In order to incorporate the aforementioned variables as independent variables into the distance function, we introduce a novel weight assignment scheme into the existing Neighbor-based method.

Our novel weighting-based distance function ($WeightFunctionMulti$) used in Algorithm 8 is formulated as below:

Given two points in Euclidean m-space, $\underline{\text{p}}$ and $\underline{\text{q}}$, where
$\underline{\text{p}} = (p_1, p_2, ..., p_m)$ and
$\underline{\text{q}} = (q_1, q_2, ..., q_m)$ and

$p_1 =$ Player P's past at level $(I-1)$
$p_2 =$ Player P's past at level $(I-2)$
and so on, and

Each such p_i or q_j is a $1 \times Z$ matrix, where Z is the total number of independent variables and each cell in a vector represents a particular performance metric such as 1) Efficiency Index, 2) Ratio of combats done as a mentor, 3) Ratio of combats done as an apprentice, and etc.

The distance between the two points p and q is given by:

$$
\begin{aligned}
&(d(p,q))^2 = (d(q,p))^2 = \\
&U_1 \cdot C_1 \cdot (q_{1,1} - p_{1,1})^2 + U_1 \cdot C_2 \cdot (q_{1,2} - p_{1,2})^2 + ... + U_1 \cdot C_z \cdot (q_{1,z} - p_{1,z})^2 \\
&+U_2 \cdot C_1 \cdot (q_{2,1} - p_{2,1})^2 + U_2 \cdot C_2 \cdot (q_{2,2} - p_{2,2})^2 + ... + U_2 \cdot C_z \cdot (q_{2,z} - p_{2,z})^2 \\
&+... \\
&+U_m \cdot C_1 \cdot (q_{m,1} - p_{m,1})^2 + U_m \cdot C_2 \cdot (q_{m,2} - p_{m,2})^2 + ... + U_m \cdot C_z \cdot (q_{m,z} - p_{m,z})^2
\end{aligned}
$$

and
$$d(p,q) = \sqrt{(d(p,q))^2} = \sqrt{(d(q,p))^2}$$

U represents weights given to each past performance metric, such that:

$U_1 = U_2 ... = U_3$ in Even-Weight scheme and

$U_1 > U_2 ... > U_3$ in Decaying-Weight scheme and

C represents correlation coefficients between the dependent variable and each independent variable. The intuition behind this is to assign more importance to higher correlating (with the dependent variable) independent variable(s).

Given $WeightFunctionMulti$, the Neighbor-Weight-2 method is shown in Algorithm 8.

Algorithm 8 Neighbor-Weight-2: Calculate Predicted Performance for Player J at Level I, given K

T (predicted player metric at level I)
P (number of previous levels)
$total\_metrics[]$ (array of player metric at each level)
$player\_specific\_metric[][]$ (array of individual player metric at each level)
$Y[]$ (array of k nearest neighboring players)
$D[][]$ (array of distance measures between players)
$C[][]$ (array of correlation coefficients between dependent variable and each independent variable)
$(Y[], D[]) = NearestNeighborSearch(K, WeightFunctionMulti, total\_metrics, J, I, P, C)$

$TotalDistance = 0$
for $i = 1$ **to** K **do**
  $TotalDistance = TotalDistance + D[J][i]$
end for
$temp = 0$ (temporary variable)
for $i = 1$ **to** K **do**
  $temp+ = ((TotalDistance - D[J][i])/TotalDistance) \times player\_specific\_metric[i][I]$
end for
$T = temp$

5.6 Experiments and Results

In this section, we discuss experiments and results.

5.6.1 Evaluation of Predictions

First, we discuss the evaluation metric used in this study, namely Root-Mean-Squared-Error. We formulate player performance prediction as a numeric prediction problem. Root-Mean-Squared-Error (RMSE for short) is known to be a good measure of accuracy for numeric prediction models such as regression models. In some literature, prediction coverage is used, however, such evaluation metrics cannot be readily translated into contextual explanations in the gaming context in which this study is taking place. RMSE is computed in the following way:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(\hat{y}-y)}{N}}$$

where

$\hat{y}$ represents the predicted value and
y represents the actual value and
N denotes the sample size (total number of players at a given player level I)

For each prediction a given method computes, we compute the difference between the predicted and the actual (these are called residuals), square it (in order to remove the directionality for later summation), and sum the squared differences across all data points (predictions across all players at a given player level I), and take the average by dividing it by the sample size N. We then take the square root of this average, and this single number (RMSE) collectively tells how much our predictions (across all data points) are overall. Mean Absolute Error (MAE) is sometimes preferred over the RMSE as an evaluator for numeric prediction models because it is known to be less sensitive to extreme values or outliers. MAE is computed as follows:

$$MAE = \frac{\sum_{i=1}^{N}|(\hat{y}-y)|}{N}$$

MAE is the average of the absolute value of the residuals (errors). It is very similar to RMSE. However, RMSE is the error measure more commonly used in many applications, and hence, we use RMSE as the primary evaluation metric especially for interpretation in the gaming context.

5.6.2 Experimentation Setup

We perform player performance prediction over all players at each player level. First, we take all the data and show how the aforementioned prediction methods perform in terms of prediction accuracy. Second, for the purposes of comparing 1) the feature representation scheme without social variables and 2) the one with social variables, we construct a separate dataset in such a way that it consists only of those players that have participated in mentoring. This way, we can evaluate the effect of including and excluding mentoring-related social variables in our prediction. Third, we perform the same set of experiments on datasets that are further classified by character classes per findings reported in Appendix C.

5.6.3 Comparison of Prediction Methods - Efficiency Index

First, we compare prediction methods for Efficiency Index prediction. Figure 5.9 shows the prediction accuracy in RMSE (y-axis) across player levels 5 through 69. And Table 5.1 summarizes the prediction results by aggregating the RMSE measures across player levels 5 through 69. Table 5.1 shows that overall, Neighbor-Weight-based methods where K is set to six outperform all other prediction methods. However, it is worthwhile to look at Figure 5.9 for more details. Figure 5.9 shows that 1) error margin in terms of RMSE increases with increasing player level and 2) Neighbor-Weight-1 outperforms other methods in the majority of player levels.

Table 5.1: Prediction of Efficiency Index - Guk - RMSE

Method	RMSE (aggregated over levels 5 through 69)	Game Session Time (in minutes)
MARCEL-1	2.9937	200.410
MARCEL-2	3.0212	199.469
Adjusted-MARCEL-2	2.8013	190.917
Modified-LinRegress-1	2.7584	190.915
Neighbor-Weight-1 (Even-Weight)	1.7077	117.332
Neighbor-Weight-2	1.6389	113.092

(Even-Weight)		
Neighbor-Weight-1 (Decaying-Weight)	1.6664	116.654
Neighbor-Weight-2 (Decaying-Weight)	1.3782	109.557

Figure 5.10 shows the comparison between Neighbor-Weight-1 and Neighbor-Weight-2. As discussed in Chapter 4, players can apprentice from player level 7 and they can mentor from player level 8. In Figure 5.10, we show the prediction accuracy between the two methods; Neighbor-Weight-1 which does not include mentoring and other social variables and Neighbor-Weight-2 which includes these variables. As discussed in Chapter 4, mentoring activities are abundant between player level 7 and 33 or so. As also shown in Table 5.1, Neighbor-Weight-2 slightly outperforms Neighbor-Weight-1. It shows that inclusion of these mentoring-related and other social variables positively contribute to prediction of Efficiency Index performance metric, especially between levels 7 and 33 where mentoring activities are especially abundant.

Suppose that a game company or game players were to look at our analysis results. It takes a bit of time to put all the numbers into perspective. Figure 5.11 shows our prediction accuracy (RMSE) in terms of game session time (in minutes). One can tweak this to view it in hours or seconds. Also as shown in Table 5.1's third column, our best performing method, Neighbor-Weight-2 on average is off by 113.092 minutes, which is significantly smaller than 190.915 minutes that the best performing existing method Modified-LinRegress-1 produces. Figure 5.11 also shows that in higher levels, Neighbor-Weight-based methods comparatively performs better. The intuition behind this finding is that while 1) MARCEL-1, Adjusted-MARCEL-2 and Modified-LinRegress-1 are affected by the global population average and 2) MARCEL-2 solely depends on a given player's past performance patterns, Neighbor-Weight-based methods rely on the closest K neighbors of each given player.

Again, it is worthwhile to look at this at each player level. For instance, Figure 5.12 shows how much our prediction is off in terms of percentage (%) from the average. It shows that in early player levels up until level 33, our prediction is off by about 22% with the exception of levels 5 through 7.

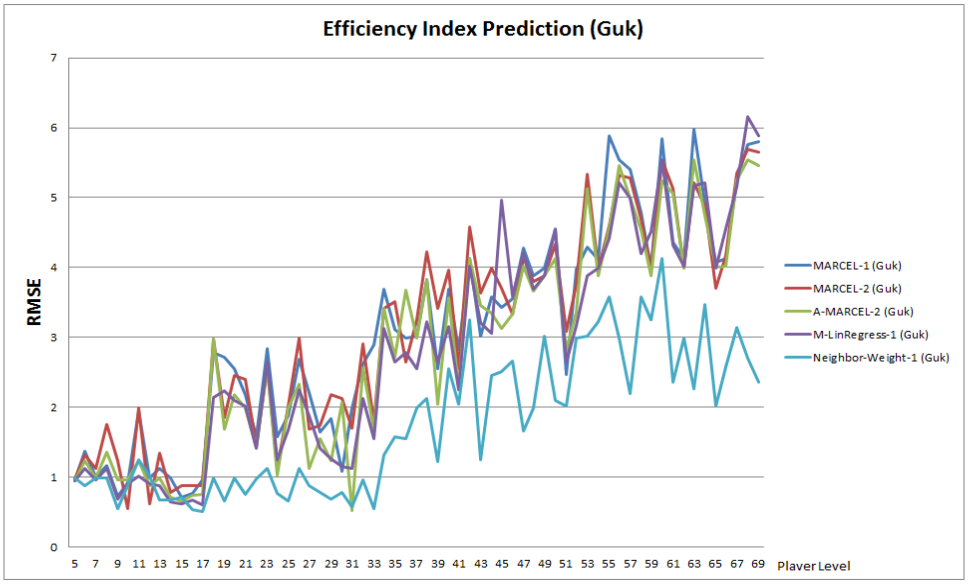


Figure 5.9: Efficiency Index Prediction - Guk

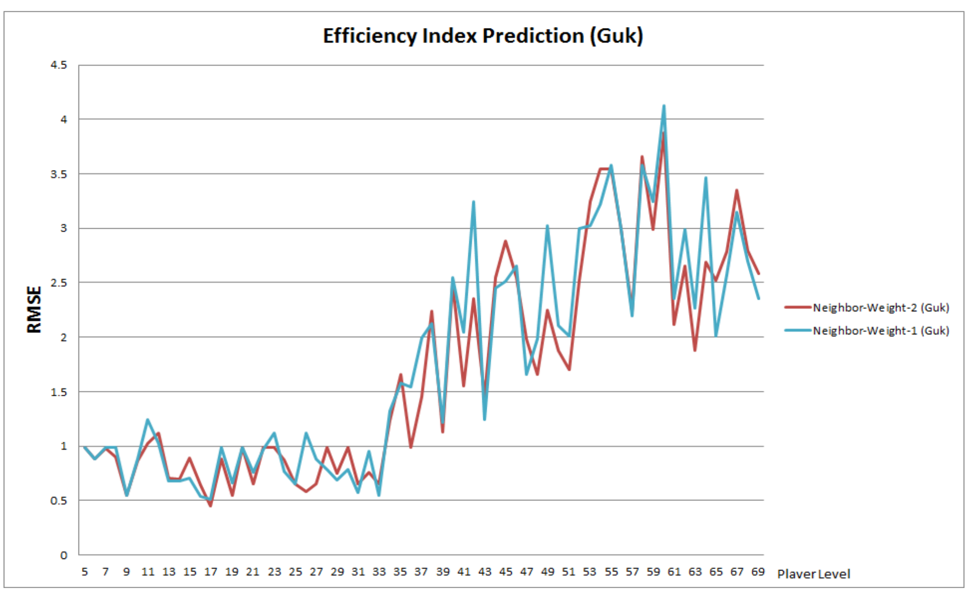


Figure 5.10: Efficiency Index Prediction - Guk - Inclusion of Mentoring and other Social Variables

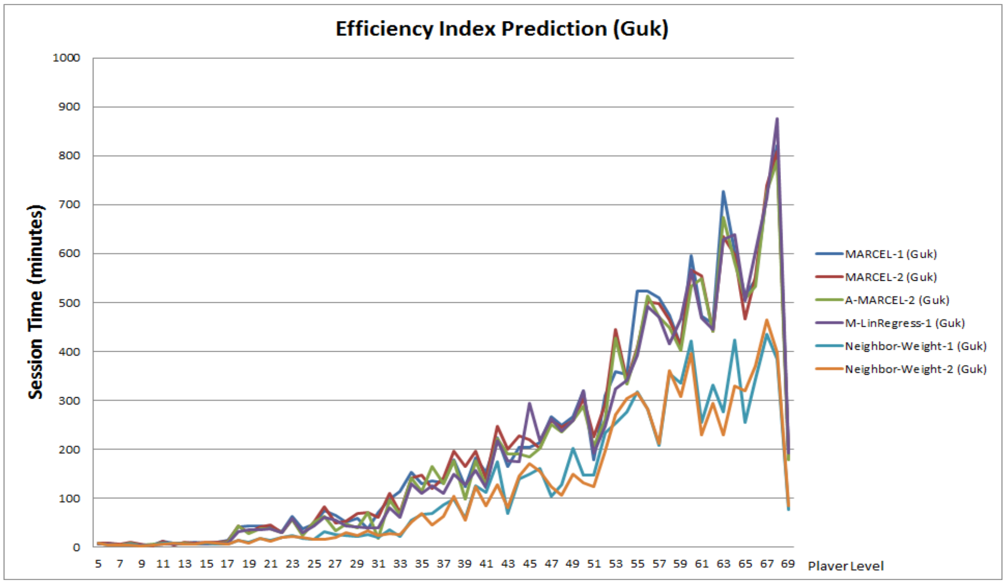


Figure 5.11: Efficiency Index Prediction - Guk - Session Time in Minutes

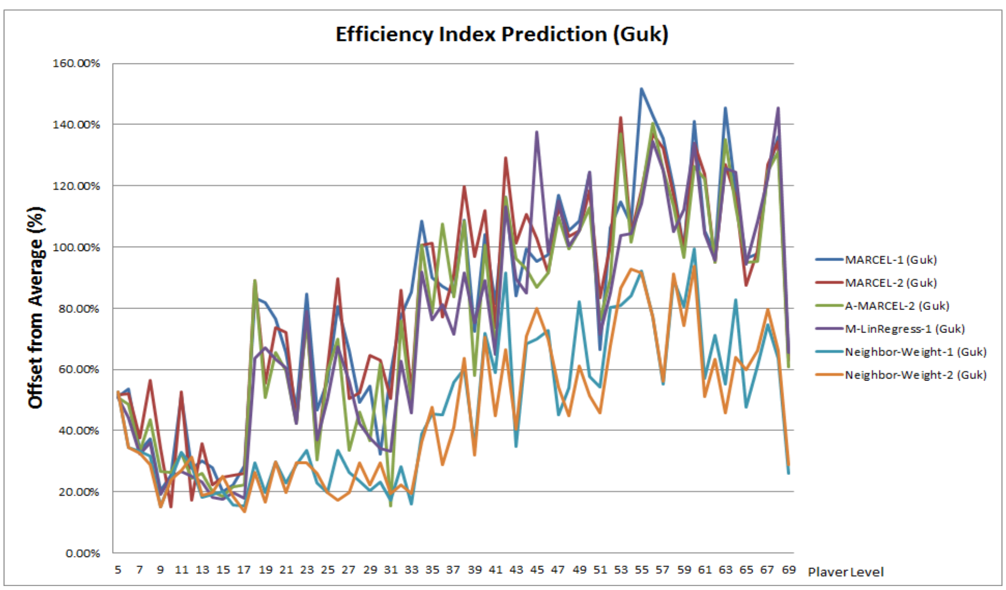


Figure 5.12: Efficiency Index Prediction - Guk - Offset from Average (%)

Next, we examine the effect of weighting schemes used in this study. We take the top two performing methods, Neighbor-Weight-1 and Neighbor-Weight-2 and compare how the two weighting schemes, namely Even-Weight scheme and Decaying-Weight scheme, perform in use with these two prediction methods. Table 5.1 shows that Neighbor-Weight-2 with Decaying-Weight scheme overall outperforms all other methods. Figure 5.13 shows how Neighbor-Weight-2 performs using either Weight scheme, across players levels 5 through 69. In almost all player levels, both Neighbor-Weight-1 and Neighbor-Weight-2 methods lead to better prediction (lower RMSE values) when they use Decaying-weight scheme. Figure 5.14 further shows in the context of gaming session time (in minutes) how well Neighbor-Weight-based methods perform when using Decaying-weight scheme.

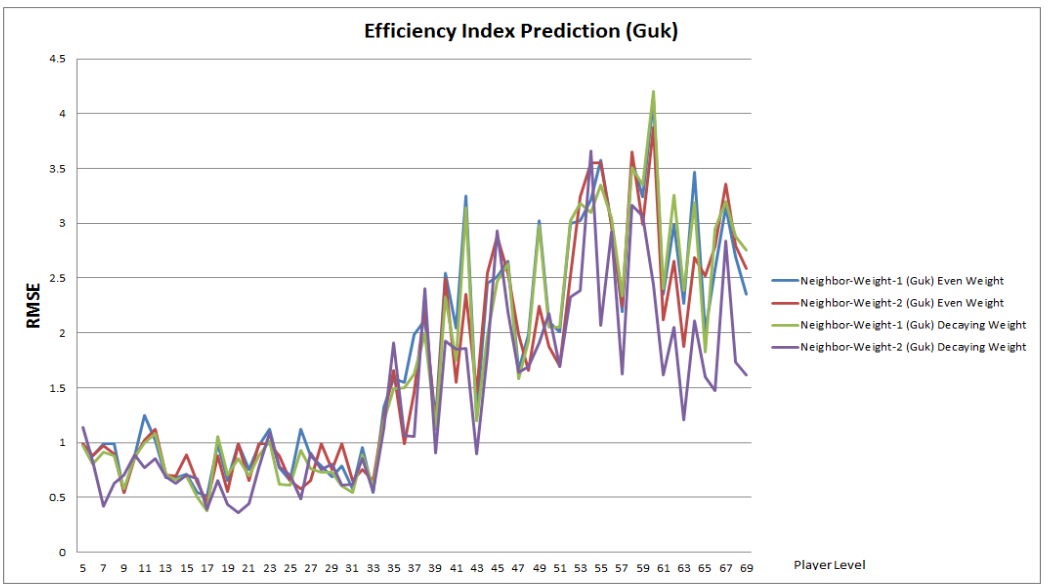


Figure 5.13: Efficiency Index Prediction - Guk - Comparison of Weight Schemes

5.6.4 Comparison of Prediction Methods - Success Index

First, we compare prediction methods for Success Index prediction. Table 5.2 shows the prediction accuracy in RMSE measures aggregated across player levels 5 through 69. For 'Guk' server, the average Success Index ranges between 88.12% and 99.7% across

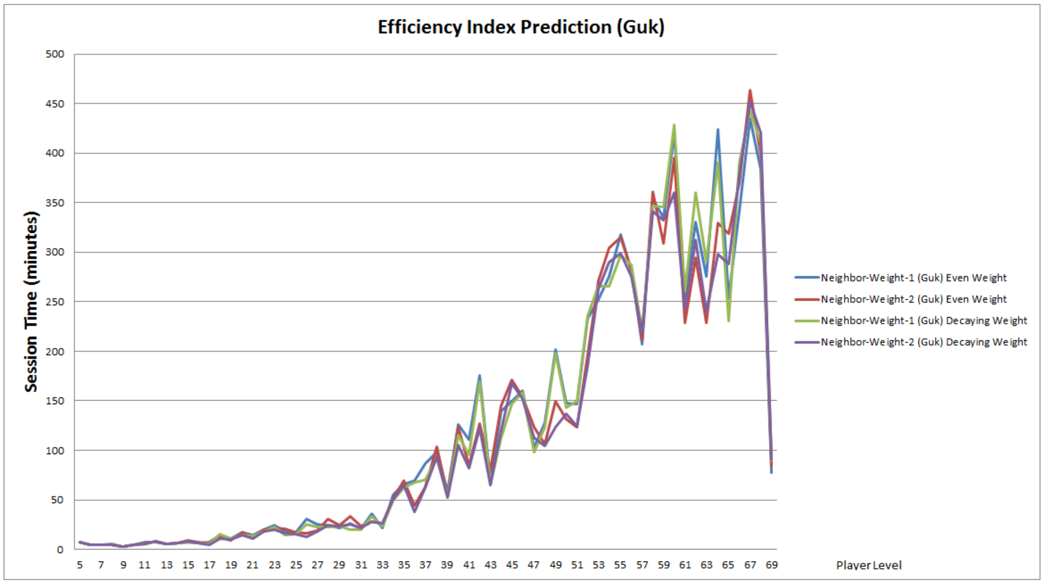


Figure 5.14: Efficiency Index Prediction - Guk - Comparison of Weight Schemes - Session Time in Minutes

player levels 5 through 69. Compared to Efficiency Index player performance metric, Success Index is less varied and thus, overall, all of the prediction methods produce error margins (in terms of offset %) minimal compared to that of Efficiency Index prediction problem.

Table 5.2: Prediction of Success Index - Guk - RMSE

Method	RMSE (aggregated over levels 5 through 69)	Success Index (%)
MARCEL-1	0.01077	1.077 %
MARCEL-2	0.01041	1.041 %
Modified-LinRegress-1	0.01020	1.020 %
Neighbor-Weight-1 (Even-Weight)	0.01069	1.069 %
Neighbor-Weight-2 (Even-Weight)	0.01012	1.012 %

Neighbor-Weight-1 (Decaying-Weight)	0.00982	0.982 %
Neighbor-Weight-2 (Decaying-Weight)	0.00965	0.965 %

Table 5.2 shows that 1) overall, all methods perform comparatively well, 2) Neighbor-Weight-based methods outperform other existing methods, and 3) Neighbor-Weight-based methods employing Decaying-Weight scheme outperforms those using Even-Weight scheme.

5.7 Conclusion

In this chapter, we show several prediction methods (MARCEL and its variations, Linear Regression, and Neighbor-Weight methods) for player performance prediction. For both Efficiency Index prediction and Success Index prediction, our experimental results show that our novel prediction methods (Neighbor-Weight-based methods) outperform existing methods. First, we show that by assigning more importance to more recent past performance metrics and less importance to distant past (Even-Weight scheme versus Decaying-Weight scheme), it leads to better prediction accuracy (lower RMSE values). Next, we show that inclusion of mentoring-related and other social variables from the game logs (such as grouping information) leads to better prediction.

Prediction models we propose in this study are expected to be a useful addition to many existing player performance monitoring tools by 1) providing a projection of a given player's future performance given his or her past performance and 2) showing how player's social activities such as mentoring and grouping can affect his or her performance. Game player performance data such as that of EverQuest II is rich of not only outcome data (i.e. number of monsters killed, number of experience points gained, number of deaths occurred, number of quests completed in a given time duration) but also process data, from which we can construct a progression of a given player's performance at any given time point. Existing player performance monitoring tools can be greatly enhanced to dynamically capture player performance progression, provide instant feedback on player's progress, and recommend tasks tailored towards a given player's objectives of playing the game (performance-oriented tasks vs. social activity-oriented).

Systematic studies of game player performance is expected to yield the following contributions. First, analysis of player performance in different dimensions (i.e. player demographics, archetypes, classes, sub-classes) can help game developers understand whether their games and game characters are being played as intended. Second, benefits for game players are two fold. a) Game players can not only have a view of their past and current performance but also they can have a view of their projected future performance. b) A recommendation engine can be built to recommend character types and tasks to players in order to meet certain objectives (i.e. move up to the next level as fast as possible, play safe by attempting easy tasks, play aggressively by attempting challenging tasks, play tasks that encourage grouping with other players). Third, players can have a view of performances of other players for the purposes of forming quest or raid teams.

Chapter 6

Player Activity Prediction

We propose a novel player activity prediction model, namely Player-Activity-Pred. While our player performance prediction methods use aggregated behavioral data (i.e. total number of monster kills a player performed over a given time period, etc.), this method takes a deeper dive and models each player's activities as a sequence of events. Next, it uses sequence alignment algorithms from Bioinformatics domain (where the algorithms were first developed to compare DNA sequences). We formulate this as a binary classification problem and show that the proposed model can fairly accurately predict player's future activity (i.e. whether or not they will complete at least one significant activity such as monster kill or quest) based on his or her past behavioral sequence. The key contribution we make in this research problem is the addition of high granularity (sequence of activity events) player behavior analysis and prediction model to the existing player monitoring tool.

This study proposes a sequence alignment-based behavior analysis framework (SABAF) developed for predicting inactive game players that either leave the game permanently or stop playing the game (stop performing significant events in the game) for a prolonged period of time. Sequence similarity scores and derived statistics form profile databases of inactive players and active players from the past. SABAF uses global and local sequence alignment algorithms and a unique scoring scheme to measure similarity between activity sequences. SABAF is tested on the game player activity data of EverQuest II. SABAF consists of the following key components: 1) sequence alignment-based player profile databases, 2) feature selection schemes and prediction model building, and 3) decision support model for determining inactive players.

6.1 Sequence Alignment Based Analysis of Player Behavior in MMOGs

In recent years, many customer relationship management tools have been developed to understand, predict, and prevent customers from leaving permanently. This can be a costly process for the company as it leads to negative financial consequences. Numerous data analysis approaches such as Support Vector Machines as well as a variety of behavior profiling methods have been developed in the past. This study proposes a behavior analysis framework based on activity sequence alignments, namely Sequence Alignment-based Behavior Analysis Framework (SABAF). SABAF is designed to predict inactive game players that either leave the game permanently or stop playing the game (stop performing significant events in the game) for a prolonged period of time. Early detection of potentially inactive players allows companies to pro-actively plan out intervention strategies in an attempt to retain the customer basis, for instance, by providing special customized or personalized offers and services, free in-game items, or any type of incentives for sticking around.

SABAF consists of the following key components: 1) sequence alignment-based customer profile databases, 2) feature selection schemes and prediction model building, and 3) decision support model for determining inactive players. This section discusses methods and experiments that test the effectiveness of SABAF on the game logs from EverQuest II (spanning over eight months). The results show that by choosing appropriate feature selection schemes and classification algorithms and experimentally adjusting the parameters, inactive players can be readily detected. This study provides comparisons between SABAF and the baseline method. This study aims to show that the data analysis methods based on sequence alignment methods can be successfully applied in inactivity prediction of game players. Additionally, we show that the novel feature selection schemes based on activity sequence alignments combined with selection of proper classification algorithms lead to inactivity prediction coverage higher than that achieved by only using aggregated activity information (i.e. total number of monsters killed, total number of quests completed, total instances of mentoring apprentices, and so forth) as demonstrated in previous chapters.

6.2 Customer Relationship Management and Inactivity Analysis

The Customer Relationship Management domain has seen numerous tools developed to improve customer acquisition and retention and increase sales. Saturated markets and intensive competition have led companies in virtually all industries to pay special attention to retaining existing customer basis as numerous studies have shown that acquisition of new customers can be costly, a process which is many-fold more expensive than retaining the existing customers [13, 27]. One of the key components in the CRM tooling is that of predictive modeling and classification for prediction of inactive customers. Inactivity prediction models often deal with a large amount of customer data where one type of customer information is customer's activity over time. Various statistical and data analysis methods have been developed to enable timely detection of potentially inactive customers followed by strategic and effective customer retention efforts [28, 50, 5].

6.3 Sequence Alignment

Sequence alignment is a well-studied method for quantifying and visualizing similarity between sequences. One of the most prominent uses of sequence alignment has been in biological sciences where the technique has been used to compare genetic materials such as DNA, RNA, and protein sequences [23]. One well-known application of sequence alignment is searching against biological databases to find specific genes or motifs [22] as well as studying phylogenetic relationships via multiple sequence alignment [7]. Sequence alignment techniques operate in a global, semi-global, or local context. In DNA and RNA sequences, nucleotides in one sequence are aligned against those of the other sequence. In protein sequences, amino acids of one sequence are aligned against those of the other sequence. Depending on how similar a pair of nucleotides or amino acids are, a score is assigned for each nucleotide-to-nucleotide or amino acid-to-amino acid pair (or gaps if gaps are inserted during alignment). The sum of the individual scores amounts to a sequence similarity measure, indicating how similar the two sequences are.

This study uses this same concept to align sequences of player activities. Given a pair of players, SABAF aligns their activity sequences, assigns scores to each activity-to-activity pair using match, mismatch and gap penalty scores, and then computes the similarity score. By aligning an activity sequence of a player (whose future inactivity is unknown) with activity sequences of known active players and inactive players, SABAF predicts whether the given activity sequence is indicative of inactivity behavior in the future. A number of factors predispose sequence alignment algorithms for use in inactivity prediction, i.e. capability to identify high level patterns embedded within the alignment and a manageable number of parameters to tweak in order to suit different types of data. Recent studies have applied sequence analysis method to time-series human behavior [57, 30, 31].

6.4 EverQuest II Game Mechanics

Chapter 2 describe the game play mechanics of EverQuest II in details.

6.4.1 Game Subscriptions and Inactivity

There is no one unique definition of inactivity. EverQuest II requires a monthly fee to play the game (as of May 2010, the monthly fee is $14.99/month in US currency). A subscribing user creates an account in order to play the game. The subscription fee is per account. Using the account, the player can create one or more in-game characters. The purchase price of the game includes a free play period of 30 days. Optionally, extended services such as an online item database or guild hosting websites can be purchased by subscribers. As part of a free trial, players can download and play the game for free. In this study, we define inactivity as an event in which 1) a subscribing user explicitly requests to discontinue the service or 2) a subscribing user stops performing significant in-game activities (i.e. monster kills, quests, recipes, trading) regularly for a prolonged period of time. If a player is seen playing at least two significant in-game activities for two consecutive weeks, we say that he remains active.

6.5 Sequence Alignment-based Behavior Analysis Framework

6.5.1 Dataset

The study uses over eight months worth of player activity data on 'Guk' server (Player-versus-Environment) from January 1, 2006 to September 11, 2006. Chapter 2 discusses in detail what all data are available from the game logs. Figure 6.1 shows the number of inactive players at each player level. 70% of such players occur in the first 15 levels, 80% occur in the first 23 levels, and 90% occur in the first 42 levels.

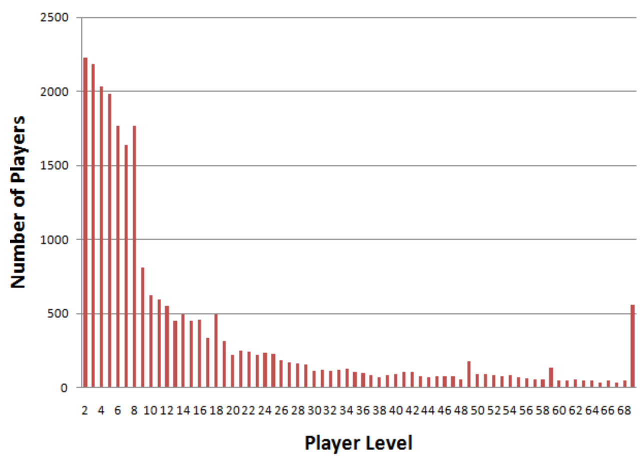


Figure 6.1: Number of Inactive Players

6.5.2 System Overview

The objective of SABAF is to predict whether a given player at a certain level i will 1) leave the game permanently in the future, for instance, at level $i + 1$ or 2) stop playing the game (and thereby producing no new significant activities) for more than 30 days. Figure 6.2 shows the workflow of SABAF. The next three sub-sections discuss in details the three key components which comprise the SABAF system.

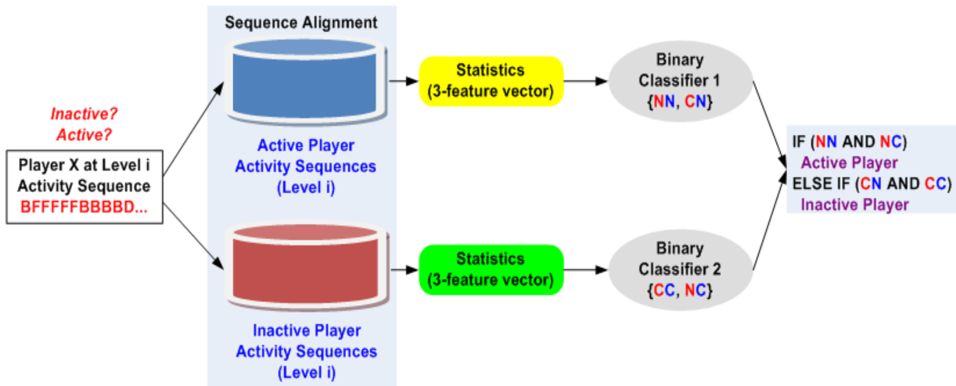


Figure 6.2: SABAF Workflow

6.5.3 Adapting Sequence Alignment to Player Behavior

Figure 6.3 shows two sample player activity sequences. Each letter represents a certain task completed in the game; 'M' denotes a monster kill, 'D' denotes a death event, 'Q' denotes a quest, 'T' denotes a mentoring event. There are well over 20 different activities kept track of in the game logs. From left to right, it shows that player X completed two monster kills and died next to come back and complete another monster kill.

X: MMDMDDDMMDMQDDQ

Y: TMTMMTMTTQDDMD

Figure 6.3: Player Activity Sequences

Next, we demonstrate how we take activity sequences of game players and transform it so that we can leverage scoring matrices used in sequence alignment. Figure 6.4 shows one example of sequence transformation. In the figure, P1 represents some player and the alphabet letters appearing after the colon representing his activity sequence. The top sequence is the original sequence directly taken from the game logs. Letter 'T' represents an act of mentoring and 'M' represents a successful completion of monster kill. In the example, a 'T' followed by an 'M' in essence represents a single activity, namely a player 'killing a monster while mentoring'. In order to use scoring matrices, we must represent players' activity sequences in such a way that each letter represents a single activity. Hence, we take 'TM' and transform it into a single letter. Figure 6.5 shows the complete mapping.

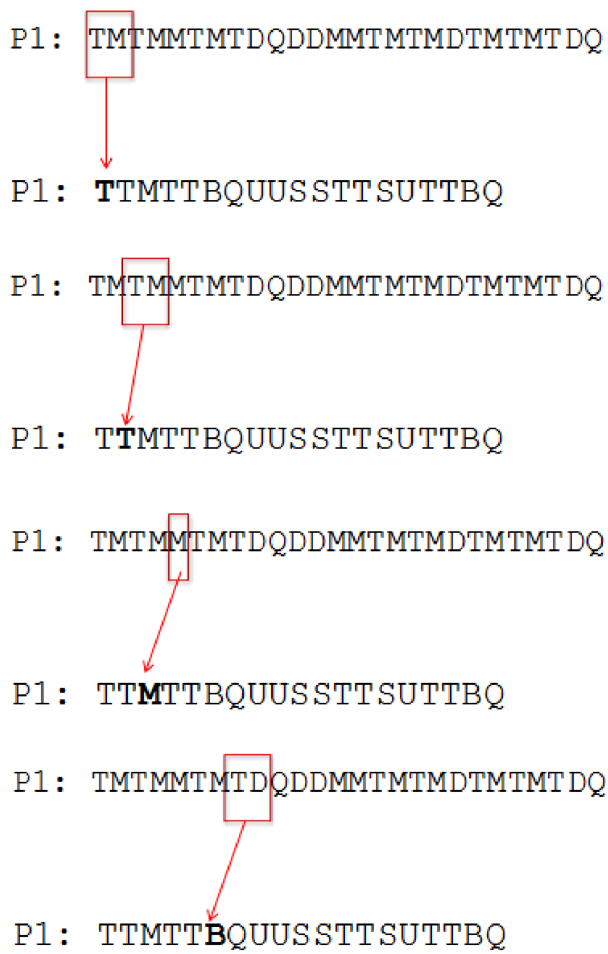


Figure 6.4: Transforming Behavioral Sequences

Figure 6.6 shows a scoring matrix we developed for EverQuest II game player sequence alignment, based on the dataset used in this thesis work. This scoring matrix can be extended and/or modified per future changes in the game design or mechanics.

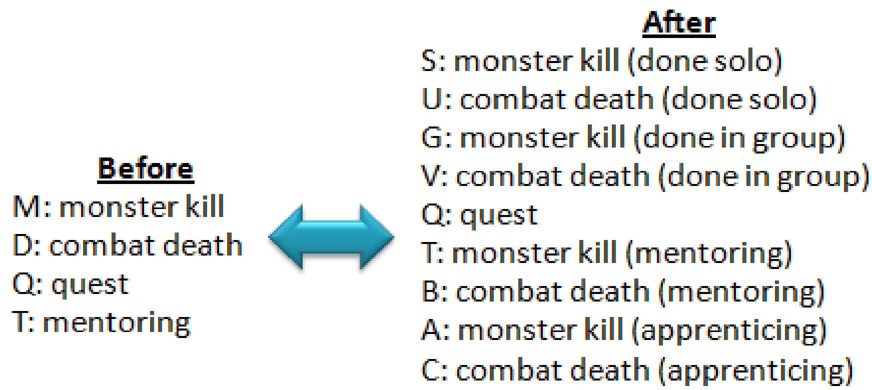


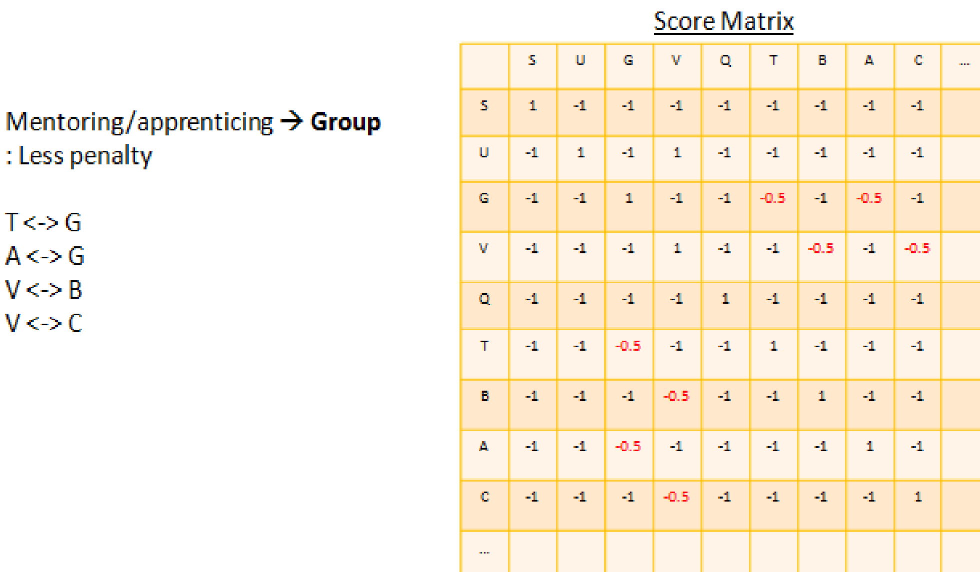
Figure 6.5: Transforming Behavioral Sequences - Mapping

In other words, the scoring matrix and the scores for aligning activities must be carefully designed in order to reflect the underlying game play experience. In the figure, we show that mentoring/apprenticing is a 'grouping' activity, hence we penalize this 'mismatch' less than we would for complete mismatches such as 'monster kill (done solo)' and 'combat death (done solo)'. A gap is penalized as '-1', and each mismatch is penalized as '-1'.

6.5.4 Part 1 - Profile Database Construction and Sequence Alignment

The first component of SABAF builds sequence alignment profile databases. It builds two types of profile database, one containing global sequence alignments and the other containing local sequence alignments. Subsequently, in each type, it builds two profile databases, one for inactive players and the other for active players. Each profile database is segmented by player level. At each level i, it takes the players who become inactive at level $i+1$, and it creates for each player an activity sequence based on their activities at level i.

Likewise, at each level i, it takes the players who stay active at level $i+1$ and creates for each player an activity sequence based on their activities at level i. Hence, it ends up with N activity sequences belonging to the Active group and M activity sequences belong to the Inactive group. Algorithm 9 outlines the steps taken to create two profile databases each segmented by player level.



	S	U	G	V	Q	T	B	A	C	...
S	1	-1	-1	-1	-1	-1	-1	-1	-1	
U	-1	1	-1	1	-1	-1	-1	-1	-1	
G	-1	-1	1	-1	-1	-0.5	-1	-0.5	-1	
V	-1	-1	-1	1	-1	-1	-0.5	-1	-0.5	
Q	-1	-1	-1	-1	1	-1	-1	-1	-1	
T	-1	-1	-0.5	-1	-1	1	-1	-1	-1	
B	-1	-1	-1	-0.5	-1	-1	1	-1	-1	
A	-1	-1	-0.5	-1	-1	-1	-1	1	-1	
C	-1	-1	-1	-0.5	-1	-1	-1	-1	1	
...										

Figure 6.6: EverQuest II - Sequence Alignment Scoring Matrix

Algorithm 9 SABAF Profile Database Construction

Data: Player Activity Sequences (P) at levels (L)
Result: Sequence Alignment-Based Profile Databases
for $l \in L$ **do**
  $N \longleftarrow \emptyset$
  $S \longleftarrow \emptyset$
  for $P[i] \in P$ **do**
    for $P[j] \in P$ **do**
      if $i \neq j$ **then**
        $N \longleftarrow Global(P[i], P[j])$
        $S \longleftarrow Local(P[i], P[j])$
      end if
    end for
    $ComputeStatisticsInsertDB(N)$
    $ComputeStatisticsInsertDB(S)$
  end for
end for

In Algorithm 9, Global function executes the Needleman-Wunsch global sequence alignment algorithm [52] and Local function executes the Smith-Waterman local sequence alignment algorithm [74]. At the core of all sequence alignment methods is an idea of assigning a score to an alignment, which we described earlier.

A player statistic consists of the mean, median, and standard deviation of the alignment scores obtained from aligning his activity sequence at a particular level against all other players at the same level. SABAF executes Algorithm 9 to construct 1) active players vs. active players, 2) active players vs. inactive players (same as inactive players vs. active players), and 3) inactive players vs. inactive players profile databases.

In order to evaluate the SABAF system, a baseline dataset is created. Two baseline profile databases are created; one for active players and one for inactive players. Each profile database is segmented by player level. At each level i, SABAF takes the players who become inactive at level $i+1$ and creates for each player an activity vector of size 25, where each column represents one of the 25 different activities in the game. Each element in the vector represents the frequency of each activity. Likewise, at each level i, SABAF takes the players who stay active at level $i+1$ and creates for each player an activity vector.

Algorithm 10 outlines the steps of creating the two baseline profile databases.

Algorithm 10 Baseline Profile Database Construction

```
Data: Player Activity Sequences (P) at levels (L)
Result: Baseline Profile Databases
for l ∈ L do
  for P[i] ∈ P do
    ComputeFrequencyInsertDB(P[i])
  end for
end for
```

6.5.5 Part 2 - Feature Selection and Predictive Model Construction

SABAF uses the open source project Weka [29] to construct classifiers. The Experiments and Results sections show comparative performances of different schemes.

6.5.6 Baseline Classifiers

In building the baseline classifiers, SABAF uses the 25-feature vector as the main feature set with two labels, one representing active players and the other representing inactive players. A binary classifier is built for each player level. Each binary classifier classifies a given input (25-feature vector of a given player) into either the active players bucket or the inactive players bucket.

6.5.7 SABAF Classifiers

With respect to feature representation, SABAF uses three different schemes; one using only the global sequence alignment-based profiles, one using only the local sequence alignment-based profiles, and one using both the global sequence alignment-based profiles and the local sequence alignment-based profiles.

Next, SABAF builds two classifiers. Each SABAF classifier receives as input a three-feature vector computed by aligning the activity sequence of a given player at level i (whose inactivity at level $i+1$ is unknown) against 1) all known active players' activity sequences at level i and 2) all known inactive players' activity sequences at level i.

The final decision on deciding whether the given input is of an active player or of an inactive player is made based on the outputs from both of these classifiers. The first classifier (Classifier 1) is built using the NN (active players vs. active players) profile database and the CN (inactive players vs. active players) profile database. And the second classifier (Classifier 2) is built using the CC (inactive players vs. inactive players) profile database and the NC (active players vs. inactive players) profile database.

6.5.8 Part 3 - Inactivity Prediction

Figure 6.2 shows the final decision making process of the SABAF system with respect to inactivity prediction. When Classifier 1 outputs NN and Classifier 2 outputs NC, the system outputs "active player" as the final decision. Likewise, when Classifier 1 outputs CN and Classifier 2 outputs CC, the system outputs "inactive player" as the final decision. The correctly identified inactive players are considered True Positives, the correctly identified active players True Negatives, the active players incorrectly identified as inactive players False Positives, and the inactive players incorrectly identified as active players False Negatives.

6.6 Experiments and Results

6.6.1 Evaluation

Evaluation focuses on the effects of 1) baseline versus SABAF feature selection schemes and 2) different classification algorithms on the True Positive and True Negative percentages. True Positives are those cases where SABAF correctly labels inactive players as inactive players. True Negatives are those cases where SABAF correctly labels active players as active players. False Positives are those cases SABAF labels active players as inactive players. False Negatives are those cases SABAF labels inactive players as active players.

Often times, companies attempt to provide some type of incentives or free goodies in an effort to prevent potentially inactive customers from leaving their service completely. In other words, False Positives could mean to the company more staff and personnel time as well as revenue loss due to having to provide incentives and free goodies to the customers that really are not potentially inactive customers. False Negatives could mean to the company future revenue loss due to permanent leaving of potentially inactive customers in the future. In either case, it leads to revenue loss for the company and therefore, this study focuses on maximizing both True Positive and True Negative percentages and thereby minimizing both False Positives and False Negatives. This study performs ten-fold cross validation on the game dataset and reports findings below.

Table 6.1 shows the logics for determining True Positive, True Negative, False Positive, and False Negative cases in use with SABAF binary classifiers.

Table 6.1: Logics for Determining TP, TN, FP, and FN cases

Input	Classifier 1 Output	Classifier 2 Output	Decision	Type
Inactive Player	CN	CC	Inactive Player	TP
Inactive Player	NN	NC	Active Player	FN
Active Player	NN	NC	Active Player	TN

Active Player	CN	CC	Inactive Player	FP

Furthermore, as Figure 6.1 shows, a majority of the inactive players in the game belong to player levels 2 through 23 or so. In addition to reporting the overall coverage across all the player levels, this study also reports prediction coverages for those player levels where the majority of the inactive players occur.

6.6.2 Comparison of Feature Selection Schemes and Classification Algorithms

This study evaluates the different feature selection schemes (baseline, global & local, global-only, and local-only) and different classification algorithms. First, the overall inactivity prediction coverage measures (measured as True Positive percentage) across players levels 2 through 69 are reported.

Table 6.2: Overall Inactivity Prediction Coverage (True Positive Percentage) across Levels 2 through 69

Algorithm	Best Feature Scheme	TP Percentage (Levels 2 to 69)
JRip	Baseline	0.547
J48	Baseline	0.501
SVM (RBF kernel)	Global & Local (SABAF)	0.903
SVM (Linear kernel)	Global & Local (SABAF)	0.915
Decision Table	Baseline	0.505
AdaBoost	Baseline	0.622
Logistic Regression	Baseline	0.539
Naive Bayes	Baseline	0.500
Neural Network	Baseline	0.594

Table 6.1 shows that the Support Vector Machine (using Linear kernel) classifier combined with one of the SABAF feature selection schemes (Global and Local sequence alignments) produces the highest overall inactivity prediction coverage (measured as

True Positive percentage). The overall coverage of this classifier using the Global and Local sequence alignment information, is 0.915 which is significantly larger than that produced by the same algorithm using the baseline feature selection scheme (0.527). In player levels beyond 20, the baseline scheme performs better than the SABAF feature representation schemes.

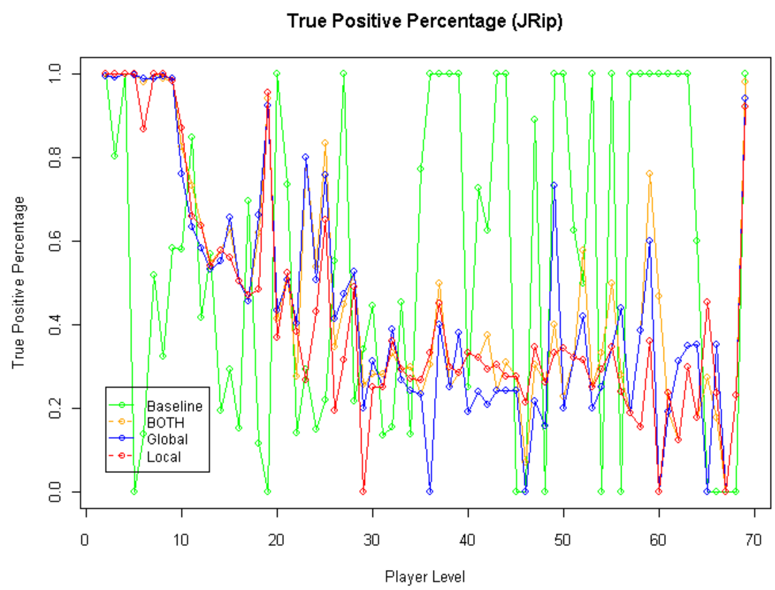


Figure 6.7: Inactivity Prediction Coverage (True Positive Percentage) - Comparison of Algorithms and Feature Selection Schemes. "BOTH" indicates the feature selection scheme of combining Global and Local sequence alignment information.

In most of the classification algorithms, all of the three SABAF feature selection schemes produce comparative prediction coverage measures with the exception of SVM (RBF kernel) and SVM (Linear kernel). Additionally, no one feature extraction scheme or no one classification algorithm works best across all the player levels. For instance, the SVM (Linear kernel) classifier produces the highest overall True Positive percentage, however, in player levels 11, 20, 28, 43 and 44, the baseline feature selection scheme produces a considerably higher coverage.

Nearly 80% of the inactive players occur in player levels 2 through 23. Table 6.3 shows the comparative performances of different classification algorithms and feature

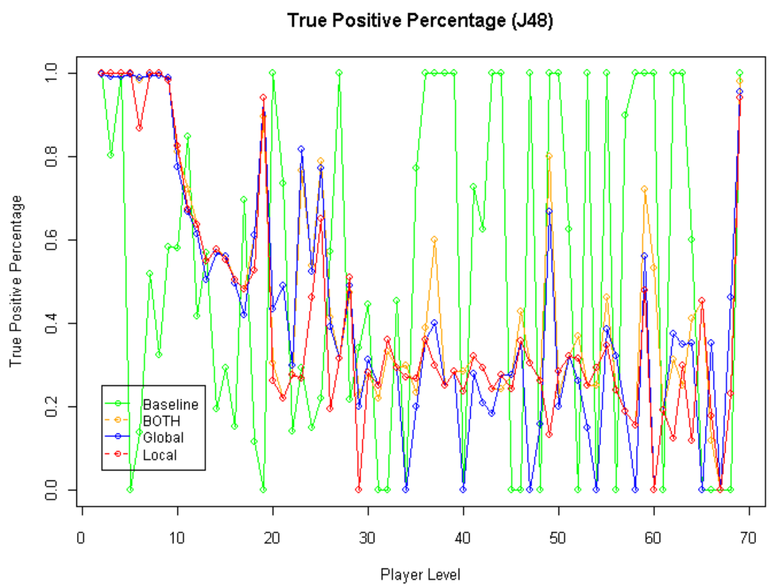


Figure 6.8: Inactivity Prediction Coverage (True Positive Percentage) - Comparison of Algorithms and Feature Selection Schemes. "BOTH" indicates the feature selection scheme of combining Global and Local sequence alignment information.

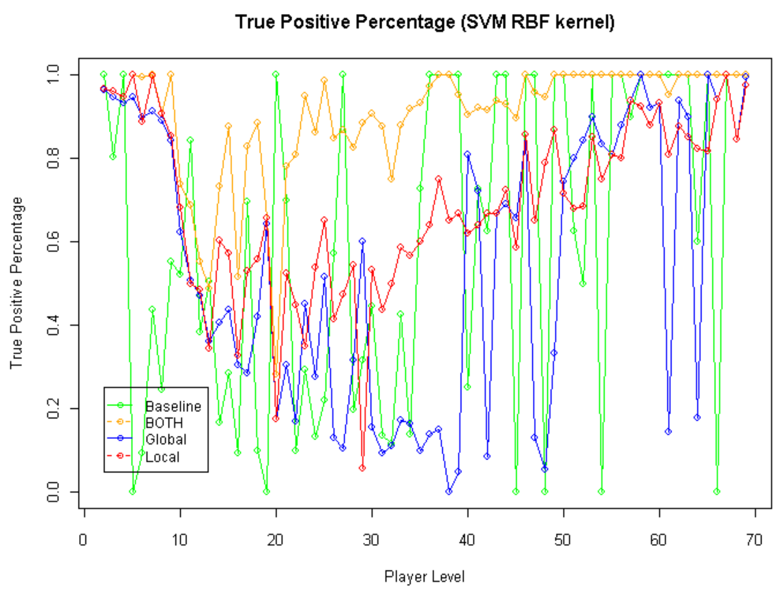


Figure 6.9: Inactivity Prediction Coverage (True Positive Percentage) - Comparison of Algorithms and Feature Selection Schemes. "BOTH" indicates the feature selection scheme of combining Global and Local sequence alignment information.

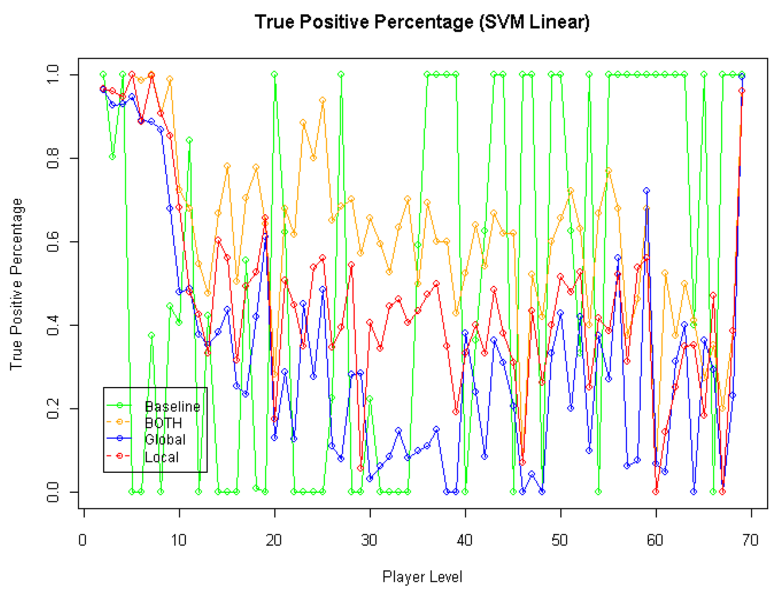


Figure 6.10: Inactivity Prediction Coverage (True Positive Percentage) - Comparison of Algorithms and Feature Selection Schemes. "BOTH" indicates the feature selection scheme of combining Global and Local sequence alignment information.

selection schemes on this 80% of the inactive player population.

Table 6.3: Inactivity Prediction Coverage (True Positive Percentage) across Levels 2 through 23 (80% of Inactive Player Population)

Algorithm	Best Feature Scheme	TP Percentage (Levels 2 to 23)
JRip	Global & Local (SABAF)	0.747
J48	Global (SABAF)	0.712
SVM (RBF kernel)	Global (SABAF)	0.798
SVM (Linear kernel)	Global & Local (SABAF)	0.972
Decision Table	Global & Local (SABAF)	0.730
AdaBoost	Global & Local (SABAF)	0.533
Logistic Regression	Global & Local (SABAF)	0.509
Naive Bayes	Global & Local (SABAF)	0.506
Neural Network	Global & Local (SABAF)	0.600

With respect to the first 23 player levels (comprising 80% of inactive player population), in most of the classification algorithms, the SABAF feature selection schemes lead to inactivity prediction coverage measures higher than that produced by using the baseline feature selection scheme.

Overall, the SABAF system performs better than the baseline in lower levels (23 player levels), covering some 80% of the inactive player population. However, the low True Positive percentage in higher levels means potentially inactive players missed out. In order to achieve an even higher overall coverage with an improved True Positive percentage in higher levels, it is best to use the baseline methods in higher levels while using the SABAF methods in lower levels.

Next, we evaluate True Negative percentages produced by different feature scheme and algorithm combinations.

While the baseline methods consistently produce near 100% True Negative percentages as the player level goes up, the True Negative percentages produced by the SABAF methods decrease, indicating that the SABAF methods in higher levels generate more

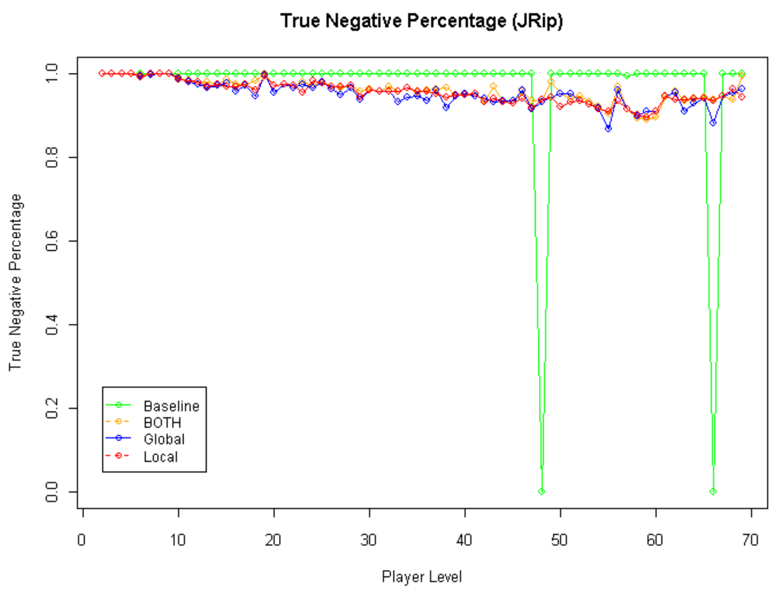


Figure 6.11: Inactivity Prediction Coverage (True Negative Percentage) - Comparison of Algorithms and Feature Selection Schemes. "BOTH" indicates the feature selection scheme of combining Global and Local sequence alignment information.

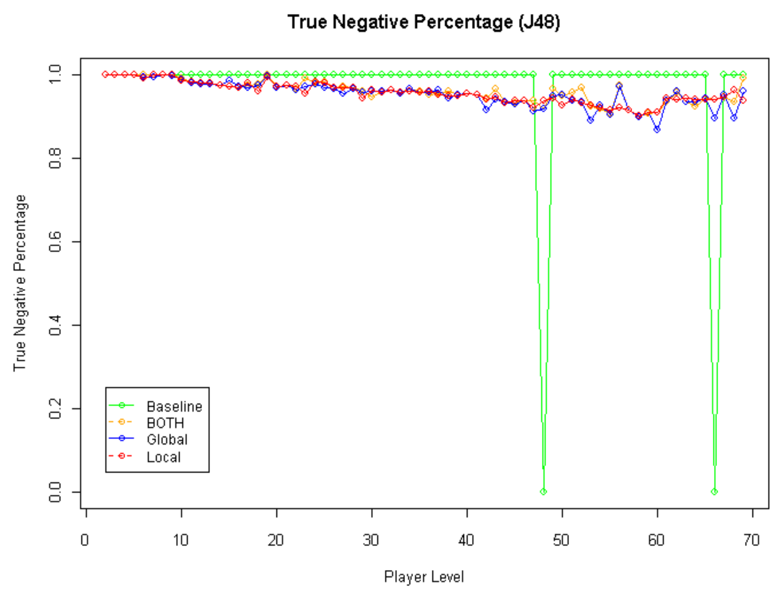


Figure 6.12: Inactivity Prediction Coverage (True Negative Percentage) - Comparison of Algorithms and Feature Selection Schemes. "BOTH" indicates the feature selection scheme of combining Global and Local sequence alignment information.

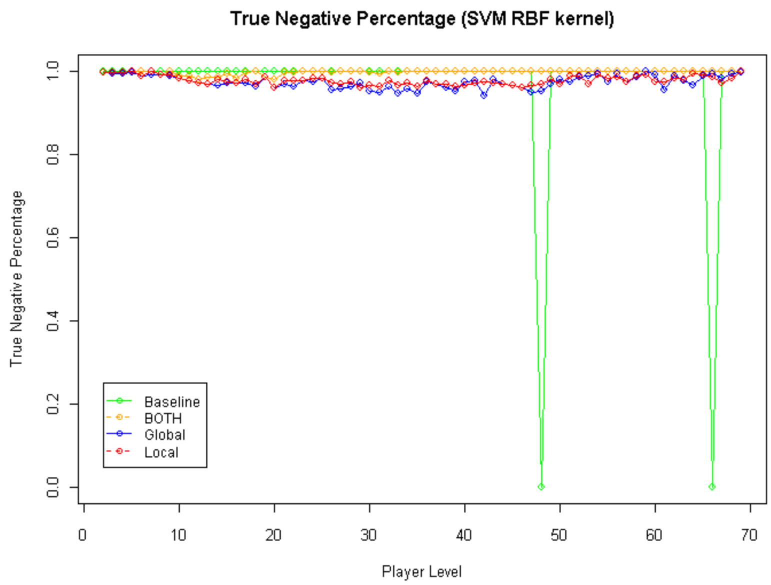


Figure 6.13: Inactivity Prediction Coverage (True Negative Percentage) - Comparison of Algorithms and Feature Selection Schemes. "BOTH" indicates the feature selection scheme of combining Global and Local sequence alignment information.

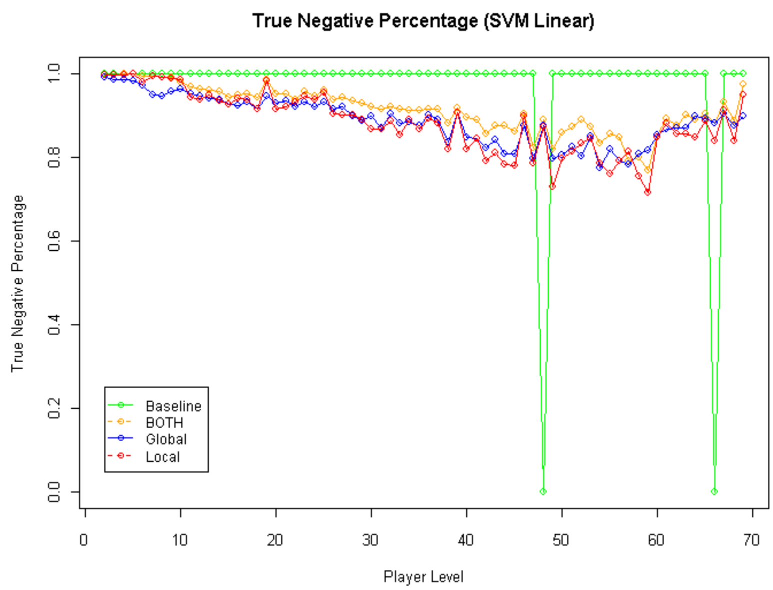


Figure 6.14: Inactivity Prediction Coverage (True Negative Percentage) - Comparison of Algorithms and Feature Selection Schemes. "BOTH" indicates the feature selection scheme of combining Global and Local sequence alignment information.

False Positives than the baseline method.

6.7 Conclusion

This chapter introduces a sequence alignment-based behavior analysis framework (SABAF) developed for inactivity prediction. Sequence similarity scores and derived statistics form profile databases of known inactive players and active players from the past. The proposed system uses global and local sequence alignment algorithms and a unique scoring scheme to measure similarity between activity sequences of game players. SABAF is tested on the game player activity data of EverQuest II. The system consists of the following key components: 1) sequence alignment-based customer profile databases, 2) feature selection schemes and prediction model building, and 3) decision support model for determining inactive players. The results show that by choosing appropriate feature selection schemes and classification algorithms, inactive players can be readily detected. This study provides comparisons between the SABAF system and the baseline method.

Our findings indicate that the Support Vector Machine (using Linear kernel) classifier combined with the Global & Local feature selection scheme produces the highest overall True Positive percentage. In most of the classification algorithms, all of the three SABAF feature selection schemes produce comparative prediction coverages with the exception of Logistic Regression, SVM (RBF kernel), and SVM (Linear kernel). We report that no one feature extraction scheme or no one classification algorithm works best across all the player levels. For instance, the SVM (Linear kernel) classifier produces the highest overall True Positive percentage, however, in player levels 11, 20, 28, 43 and 44, the baseline feature selection scheme produces a considerably higher coverage. With respect to the first 23 player levels (comprising 80% of inactive player population), in most of the classification algorithms, the SABAF feature selection schemes lead to inactivity prediction coverage higher than that produced by using the baseline feature selection scheme.

Overall, the SABAF system performs better than the baseline in lower levels (23 player levels), covering some 80% of the inactive player population. However, the low True Positive percentage in higher levels mean potentially inactive players missed out. In order to achieve an even higher overall coverage with an improved True Positive percentage in higher levels, it is best to use the baseline methods in higher levels while using the SABAF methods in lower levels. In terms of True Negatives, while the baseline methods consistently produce near 100% True Negative percentages as the player level goes up, the True Negative percentages produced by the SABAF methods decrease, indicating that the SABAF methods in higher levels generate more False Positives than the baseline method.

Chapter 7

Player Performance, Motivation, and Enjoyment

Enjoyment is a vital component in the business model of the game industry. Despite research on their relationship to the success or failure of a game, little attention has been paid to the effect of player performance on player enjoyment. This study investigates how player motivation, player performance, and player enjoyment are connected in EverQuest II. It investigates the impact of task difficulty and player performance on player enjoyment. Estimation of task difficulty and player performance was performed by the analysis of the game's operational data (i.e. game logs), while assessment of player enjoyment was based on a survey. Our findings indicate that the correlations do not fully conform to the flow theory and suggest that the knowledge of player motivations is critical in accurately predicting player enjoyment. In this chapter, we propose two novel player enjoyment prediction models, namely Player-Fun-Pred and Player-Quit-Pred. The key contribution we make in this research is the analysis of two important dimensions of game play experience in MMOGs, namely motivation and enjoyment, and the analysis of how player performance combined with player motivation can fairly accurately predict player enjoyment.

7.1 Introduction

Player enjoyment is extremely important for commercial game development. While many games today provide in-game tutorials to help newcomers ramp up quickly in the early stage of the game as well as in-game assistants throughout the game to help identify tasks to perform to gain rewards, little is understood about the relationship between in-game player performance and player enjoyment. Also, the role of player motivation(s) in determining player enjoyment has not been studied in detail. In this section, building on concepts from the flow theory [14], we focus on the impact of game difficulty and player performance on player enjoyment. Further, we investigate how player motivation(s), combined with player performance, play a critical role in determining player enjoyment. We use operational data of game players in EverQuest II in conjunction with survey data to relate player performance to player enjoyment as well as player motivation. Our findings provide a foundation for a customized task recommendation system during game play where its primary objective is to automatically identify player enjoyment level and suggest in-game activities that will help players stay in the flow zone.

7.2 Background

7.2.1 Player Motivations

An explosive growth in video game sales and the emergence of different game genres and types designed to appeal to all sorts of demographics over the last decade have created a venue for new research problems in social sciences. The wide spread of the Internet in late 90's led to many studies on Internet addiction. In a similar manner, with the wide spread of video games, researchers in recent years have investigated game addiction [82, 35]. In particular, studies on MMOGs report that individuals playing such games are driven by the need to substitute human connections in real life, unachievable in real life under normal circumstances. In the game's persistent, virtually connected fantasy world, individuals via their in-game avatars create inter-personal relationships with others or other virtual beings (i.e. characters driven by other human players and computer-driven bots). There exist several different motivations for playing MMOGs, and broadly, they are categorized into 1) Achievement, 2) Socialization, and

3) Immersion [92]. And each category is further divided into multiple sub-components [92]. Detailed analysis of player motivations and player demographics is in EverQuest II is reported in [89].

Achievement-oriented players thrive to make progress, level up quickly, understand game mechanics towards progressing fast, and compete. Socialization-oriented players thrive to interact with other gamers (i.e. chatting, teaming up, helping, and joining organized groups such as guilds), form sustaining relationships with other gamers, and collaborate with others. Immersion-oriented players thrive to explore the game's virtual fantasy world, wander around and discover new locations and new monsters or tasks to complete, enjoy following the story, and customize their characters (i.e. styling their characters' outer appearances).

7.2.2 Player Performance and Enjoyment

Today, there are many different game genres (based on game play interaction); action, shooter, action-adventure, adventure, role-playing, and simulation [95]. Each is further categorized into single-player mode and multi-player mode. Depending on the genre and the type, games can present a wide variety of game mechanics (i.e. point scoring or reward system, point-and-shoot, keyboard button-driven), task types (i.e. monster kills, player-versus-player, problem solving, strategy planning), and modes of interaction either with the in-game computer-driven bots or with other gamers. While there are many dimensions comprising player's game-play experience, in certain game genres, a close connection has been reported between completing tasks/challenges and mastering skills and player enjoyment [58, 80, 38, 81].

7.3 Research Problems

The flow theory states that most enjoyment (i.e. positive experiences) is achieved when one masters tasks or challenges that are in the flow zone, where tasks are not too easy and not too challenging or difficult [14, 42]. The flow theory has been explored in the context of game play experience [63, 33, 75, 11, 51, 90]. Figure 7.1 depicts the principles of the flow theory. When the task difficulty is comparatively more difficult than the skill level of the individual, he/she can experience anxiety or frustration (negative experience). In the other extreme, boredom (negative experience) can be experienced by the individual if the task difficulty is comparatively easier than his/her skill level.

As described in an earlier section, in the case of MMOGs, there exist several different motivations, broadly categorized into 1) Achievement, 2) Socialization, and 3) Immersion [92]. While not being completely mutually exclusive, the primary objectives or goals pursued by individuals in each category appear to differ from one another. Specifically, Achievement-oriented players are expected to derive more enjoyment from challenging tasks than players in the other two categories. It is noted that it is not a mutually exclusive categorization, wherein, a particular individual can be of each of the three types at varying degrees. In a later section, we present statistics on the EverQuest II players with respect to their motivation indicators.

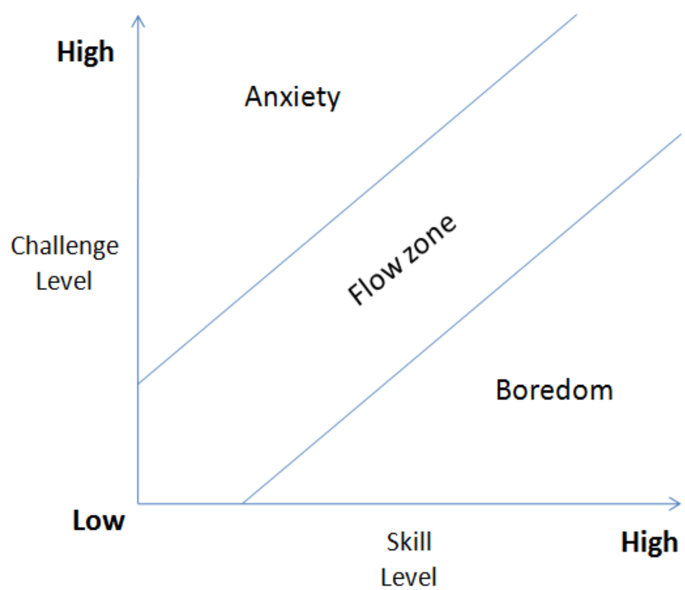


Figure 7.1: Flow Theory

In this study, we explore the following research questions in detail and seek to provide empirical understanding and evidence for an accurate model of player experience from the perspective of player enjoyment.

- 1) How does task difficulty affect player enjoyment?

- 2) How do the different motivations affect players in terms of enjoyment in the game in relation to task difficulty?

7.4 In-Game Logs

This study uses nine months' worth of player and team activity data from 'Guk' server (Player-versus-Environment or PvE), 'Antonia Bayle' server (Role-Playing or RP), and 'Nagafen' server (Player-versus-Player or PvP), between January 2006 and September 2006. These three game servers are selected because they represent three different ways to play the game. Therefore, this study provides valuable insights into a wide variety of game genres. Details of all the features available in the game logs of EverQuest II are described in Chapter 2.

7.5 Survey Data

This study uses survey data collected from 7,000 game players in EverQuest II in 2006. The survey data provide information about players' motivations and enjoyment. These data are joined with in-game behavioral data, from the game's game logs, mentioned in the above section. It is noted that while the survey data is a static snapshot taken at a single time point in September 2006, when joined with the in-game behavioral data, the resulting data is enhanced with player activity data across all nine months from January 2006 through September 2006, which includes all of the past in-game activities. There are three survey variables used in this study. In this exploratory study, we consider the 'fun' variable and the 'quit' variable to be measuring player enjoyment. The correlation between the 'Fun' variable and the 'Quit' variable is shown below (Table 7.1).

Table 7.1: Correlation between Fun and Quit

Server	Correlation Coefficient
Guk	0.358
Antonia Bayle	0.423
Nagafen	0.387

The low correlation between the two variables indicates that the players who do not enjoy the game do not necessarily want to quit the game. Likewise, it indicates that the players who enjoy the game can still want to quit the game. This suggests that player 'fun' and player (tendency to) 'quit' must be looked at separately.

7.6 Analysis of Survey Variables

7.6.1 Fun

This variable measures how much players have enjoyed playing this game (Table 7.2).

Table 7.2: 'Fun' survey variable

Value	Meaning
1	Not at all
2	Not much
3	Somewhat
4	Very much

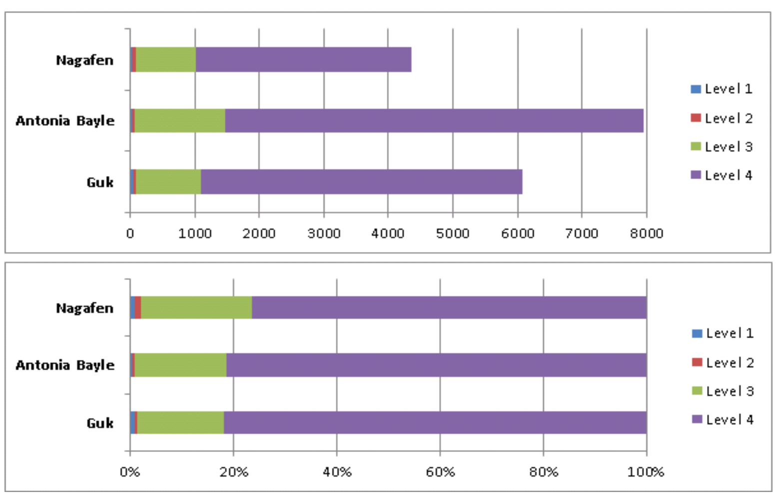


Figure 7.2: 'Fun' survey variable

As shown in Figure 7.2, a majority of the respondents responded positively, stating that they are somewhat to very much enjoying the game.

7.6.2 Quit

This variable measures how much players are thinking about quitting the game (Table 7.3).

Table 7.3: 'Quit' survey variable

Value	Meaning
1	Yes, very soon
2	Maybe
3	Some day, but not soon
4	I have no plans to quit at all

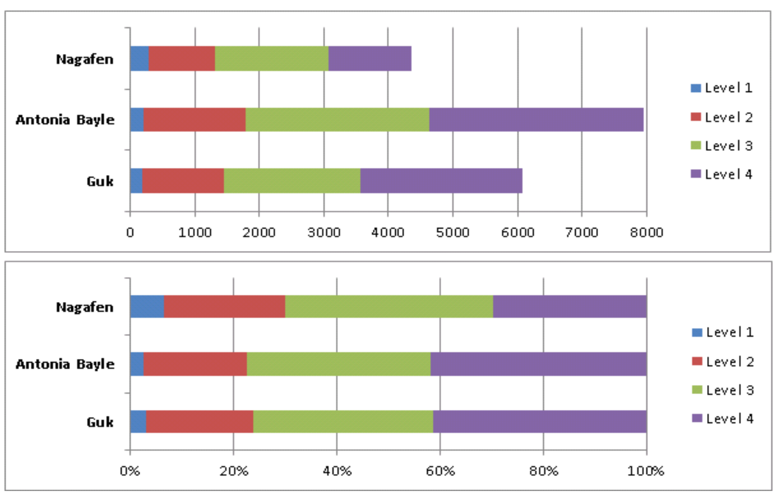


Figure 7.3: 'Quit' survey variable

Figure 7.3 shows that a majority of the respondents responded positively, indicating that they are likely to not quit the game.

7.6.3 Player Motivations

There are 11 variables measuring how important the following game-play aspects are to the respondents, and they are categories into; Achievement (A), Socialization (S), and Immersion (I). Each variable is measured between 1 and 5, where 1 indicates "Not important at all" and 5 indicates "Extremely important" (Table 7.4).

Table 7.4: Player Motivation Survey Variables

Variable	Game-Play Aspect
A-1	Leveling, acquiring great items and gear, becoming powerful.
A-2	Figuring out the game mechanics, planning my characters development, and optimizing my character.
A-3	Competing with other players in terms of combat, crafting ability, or the economy.
S-1	Chatting with and getting to know other players.
S-2	Developing deep and meaningful relationships with other players.
S-3	Having a character that can solo well and work independently.
S-4	Being part of a team.
I-1	Exploring the world and knowing things (stories, locations of NPCs, etc.) that most other players don't know about.
I-2	Role-playing and having interesting background stories for your character.
I-3	Customizing your characters to make them look distinctive, stylish, and unique.
I-4	Escaping from the real world and leaving behind some real-life problems and worries.

Figures 7.4 through 7.6 show the raw number and percentage of responses (ranging from 1 to 5) for each of the 11 Player Motivation questions. Responses whose values are either 4 or 5 are considered "positive" for this discussion. The following are our findings:

- (A-1) PvP (68%) > PvE (62%) > RP (48%)

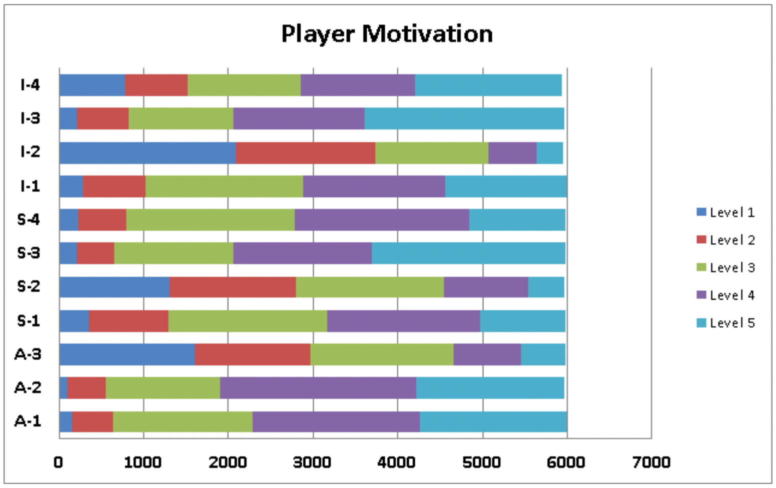


Figure 7.4: Player Motivation, 'Guk' (PvE)

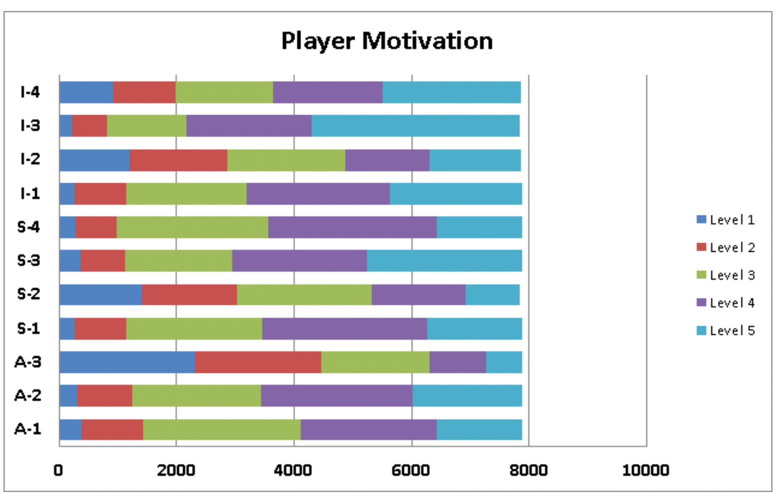


Figure 7.5: Player Motivation, 'Antonia Bayle' (RP)

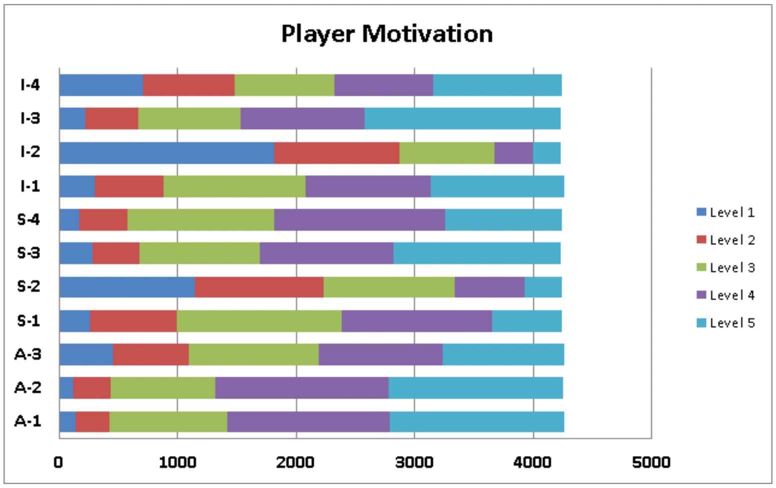


Figure 7.6: Player Motivation, 'Nagafen' (PvP)

- (A-2) PvP (69%) > PvE (68%) > RP (57%)
- (A-3) PvP (49%) > PvE (22%) > RP (19%)
- We find that on average, respondents from PvP servers are shown to be more achievement-oriented than those from PvE server. RP server's respondents show the lowest level of achievement orientation.
- (S-1) RP (57%) > PvE (48%) > PvP (43%)
- (S-2) RP (32%) > PvE (24%) > PvP (21%)
- (S-3) PvE (67%) > RP (62%) > PvP (59%)
- (S-4) PvP (58%) > RP (55%) > PvE (53%)
- We find that RP server respondents respond more positively about chatting, getting to know, and developing meaning relationships with other players. PvP server respondents respond more positively about being part of a team, working collaboratively rather than soloing.
- (I-1) RP (59%) > PvE (52%) = PvP (52%)

- (I-2) RP (38%) > PvE (16%) > PvP (13%)
- (I-3) RP (72%) > PvE (66%) > PvP (63%)
- (I-4) RP (54%) > PvE (52%) > PvP (46%)
- We find that consistently RP server respondents respond more positively to immersion-oriented questions. In particular, a significantly greater percentage of RP server respondents responded positively to I-2 than in other servers.

7.7 Player Performance and Task Difficulty

We model player performance as a measure of Skill versus Challenge. Skill is defined as player level, and Challenge is defined as task difficulty. Details of different task types, point-scaling system, and other game mechanics of EverQuest II are described in Chapter 2 and Chapter 3. As far as task types, we focus on monster kills since the majority of the tasks completed by players in the game are monster kills on all three game servers. Task difficulty in this case is the monster level. From 'Nagafen', a PvP server, we also extract PvP fights. In this case, task difficulty is the player level of the opponent player. In Figures 7.7 through 7.14, we explore Skill versus Challenge statistics across all data points. In the figures, the black line represents the mean. The red line represents one standard deviation above the mean, and the blue line represents one standard deviation below the mean.

A comparison of Figures 7.7, 7.9, 7.11 and 7.13 reveals that task difficulty increases linearly to skill level and that PvP fights show a wider variation in terms of task difficulty. Also, a comparison of Figures 7.8, 7.10, 7.12 and 7.14 shows that the average task difficulty in PvP fights (i.e. Skill Level of the opponent player) is higher than that in monster kills. Figures 7.8, 7.10 and 7.12 show a very similar pattern of monster task difficulty. This indicates that human-controlled characters exhibit a wider variation of skill level and on average, they are easier to tackle than monsters.

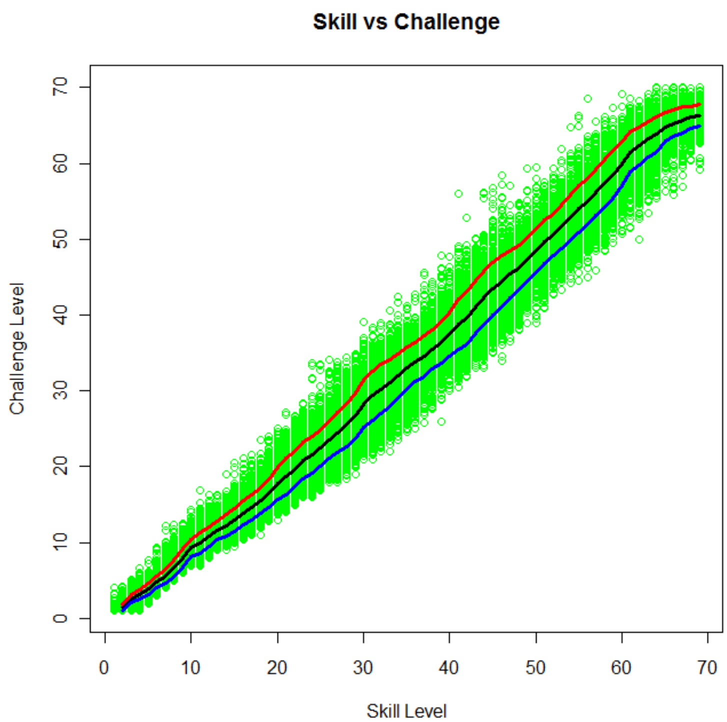


Figure 7.7: Skill vs. Challenge (Monster Kills), 'Guk' (PvE)

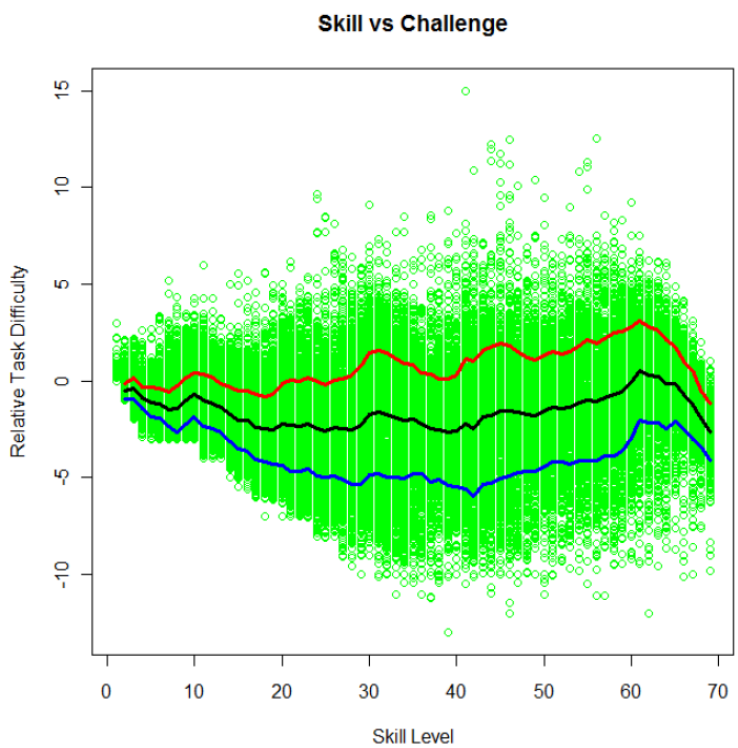


Figure 7.8: Skill vs. Challenge (Monster Kills), 'Guk' (PvE)

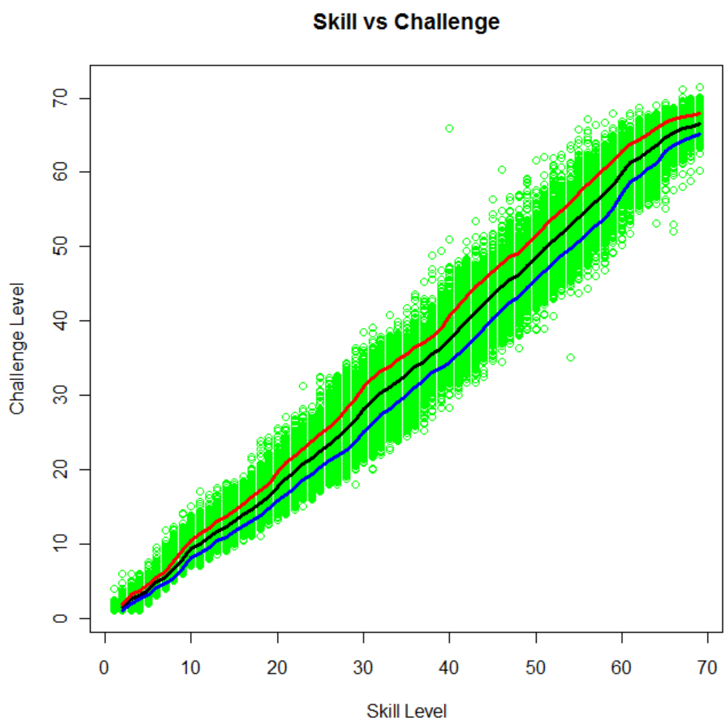


Figure 7.9: Skill vs. Challenge (Monster Kills), 'Antonia Bayle' (RP)

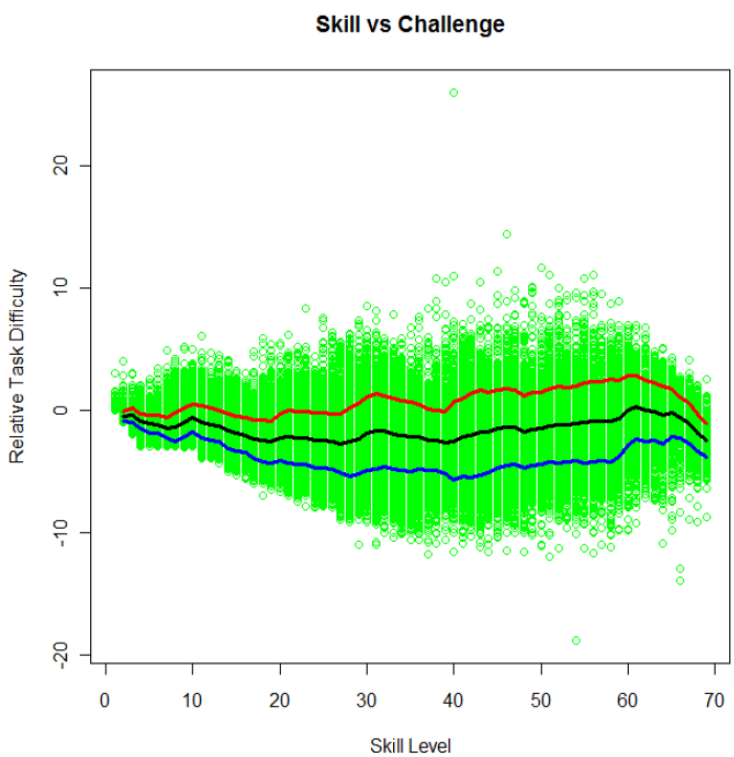


Figure 7.10: Skill vs. Challenge (Monster Kills), 'Antonia Bayle' (RP)

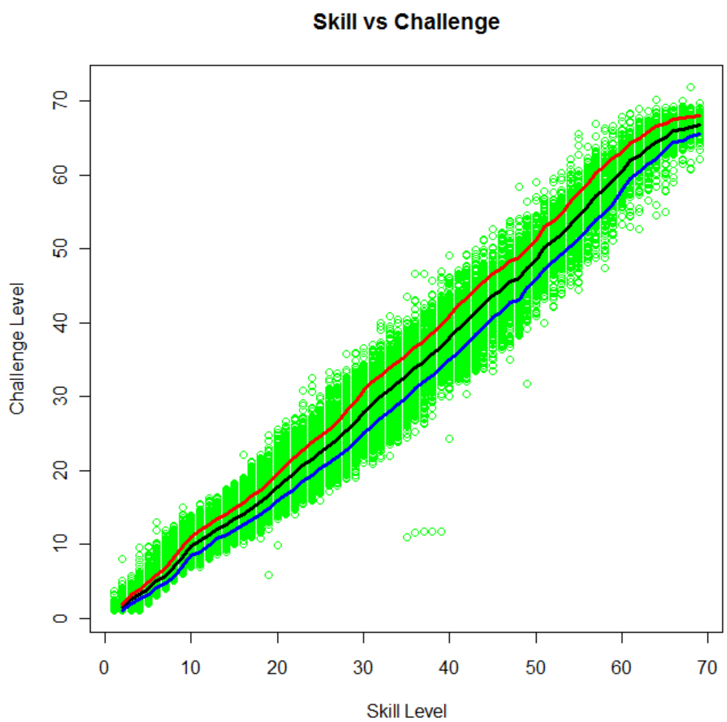


Figure 7.11: Skill vs. Challenge (Monster Kills), 'Nagafen' (PvP)

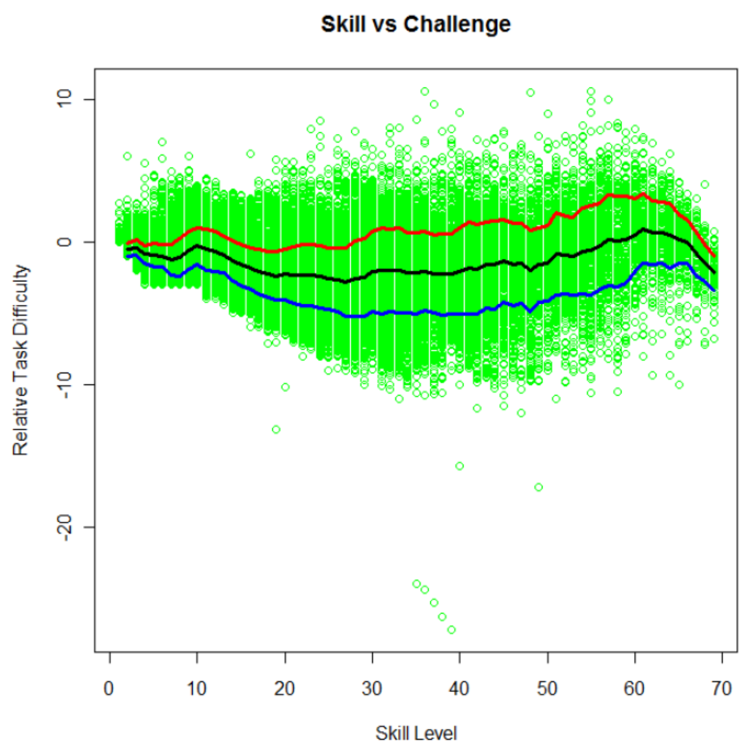


Figure 7.12: Skill vs. Challenge (Monster Kills), 'Nagafen' (PvP)

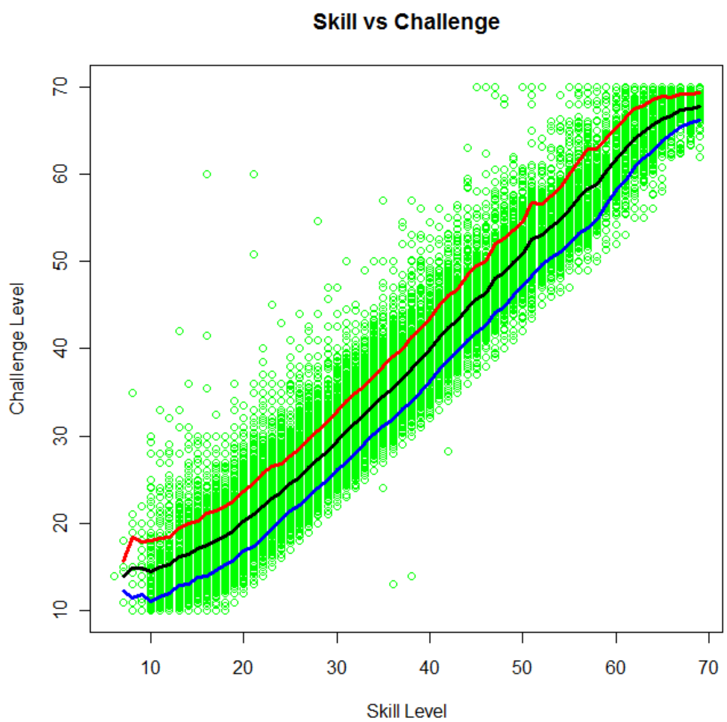


Figure 7.13: Skill vs. Challenge (PvP Fights), 'Nagafen' (PvP)

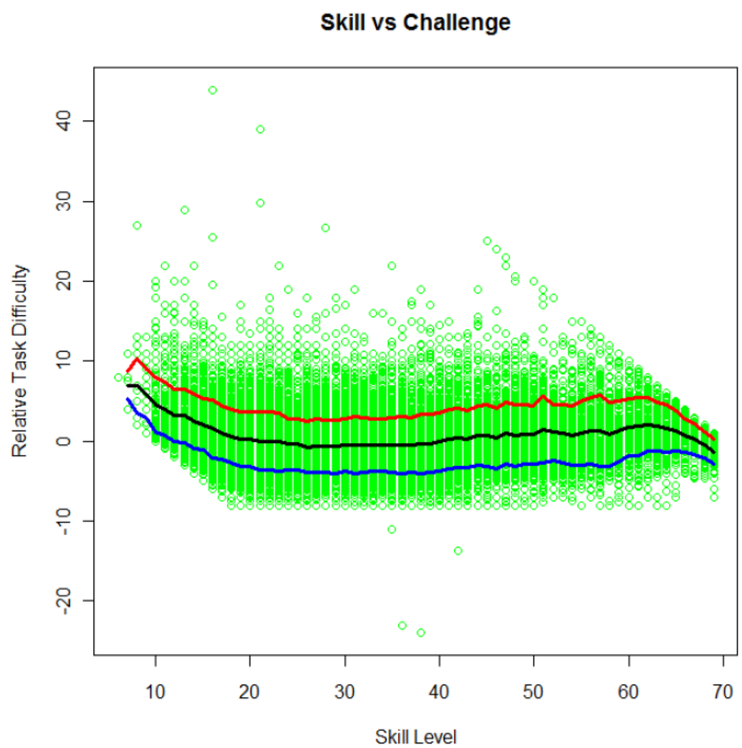


Figure 7.14: Skill vs. Challenge (PvP Fights), 'Nagafen' (PvP)

7.8 Predicting Player Enjoyment - Quit

This section describes our empirical results from a large-scale experiment for predicting levels of players' intention to quit. We formulate this as a multi-class classification problem, where the objective is to predict the 'Quit' level (which, in this experiment, is treated as a categorical variable, i.e. 1, 2, 3, 4). We use Weka [29] to evaluate various classification algorithms on our dataset. We use standard 10-fold cross-validation.

7.8.1 Feature Representation

- 1) Scheme #1 (S1): In this feature representation scheme, we only use the Skill vs. Challenge data.
- 2) In this case, we combine both Skill vs. Challenge data and Player Motivation data.

7.8.2 Sampling

As shown in Figures 7.2 and 7.3, the number of data points is not uniform across the four values (labels) we intend to predict. The SMOTE algorithm [84], a widely employed technique for imbalanced classification, is used to over-sample the comparatively rare classes. As a result, we sample a total of 5,000 data points from each class (label). We take 10 such samples.

7.8.3 Results

Our experimentation takes 10 samples and for each sample, we perform 10-fold cross-validation. Table 7.5 shows the prediction results, averaged up across all 10 samples over 10-fold cross-validation.

Table 7.5: Prediction Results (F-measure in %) for 'Quit' - server 'Guk'

	Guk (monster kills)	
Algorithm	S1	S2
Naive Bayes	33.8	41.3

Bayesian Network	36.7	51.2
Logistic Regression	32.3	51.2
J48	41.2	98.1
Random Forest	42.2	96.6
REP Tree	40.3	85.0
Random Tree	41.3	93.5
LAD Tree	32.1	36.0
Decision Table	37.3	85.6
Decision Table & Naive Bayes	38.0	98.1
JRip	32.8	96.1
SVM (Radial)	39.2	53.8
Neural Network	35.8	66.4
SMO	32.0	43.0

Table 7.6: Prediction Results (F-measure in %) for 'Quit' - server 'Antonia Bayle'

	Antonia Bayle (monster kills)	
Algorithm	S1	S2
Naive Bayes	32.4	38.5
Bayesian Network	32.9	46.0
Logistic Regression	32.9	38.4
J48	36.8	97.7
Random Forest	37.8	94.5
REP Tree	37.4	81.0
Random Tree	37.2	91.5
LAD Tree	26.9	40.0
Decision Table	33.7	82.1
Decision Table & Naive Bayes	33.9	97.5
JRip	19.3	93.0
SVM (Radial)	35.4	53.8

Neural Network	33.6	58.4
SMO	33.0	38.9

Table 7.7: Prediction Results (F-measure in %) for 'Quit' - server 'Nagafen'

	Nagafen (monster kills)		Nagafen (PvP fights)	
Algorithm	S1	S2	S1	S2
Naive Bayes	27.2	37.5	31.0	39.8
Bayesian Network	29.8	43.2	32.4	45.8
Logistic Regression	31.2	37.6	33.4	39.6
J48	30.8	97.2	34.5	96.9
Random Forest	27.1	97.3	33.2	97.7
REP Tree	29.8	87.6	34.3	91.3
Random Tree	26.9	94.8	32.4	96.0
LAD Tree	25.4	38.4	27.3	39.9
Decision Table	31.3	91.2	34.4	94.3
Decision Table & Naive Bayes	31.4	96.9	34.3	97.0
JRip	18.3	90.1	21.1	69.8
SVM (Radial)	28.1	45.1	33.9	48.2
Neural Network	30.0	56.7	33.5	64.5
SMO	30.4	38.1	33.1	38.5

Tables 7.5, 7.6 and 7.7 show the prediction results in terms of F-measure (in percentage). Overall, our results show that across all of the classification algorithms, the predictive models built on Feature Representation Scheme #2 perform significantly better than those based on Feature Representation Scheme #1. In other words, Skill versus Challenge information alone is not sufficient for accurately predicting players' intention to quit, a finding that does not conform to the flow theory.

For instance, for 'Guk' server (PvE), the best performing predictive model leveraging only Skill versus Challenge information achieves only 42.2% F-measure whereas the inclusion of Player Motivations information leads to 98.1% F-measure. A similar trend is observed in prediction results for other servers. Our results indicate that for 'Nagafen' server (PvP), it is strongly evident that Skill versus Challenge information alone leads to poor prediction results.

Earlier, Play Motivations data revealed that PvP server respondents responded more positively about being part of a team, working collaboratively rather than soloing. Although not particularly specific to only 'Nagafen' server (because there are a lot of group plays in MMOGs in general), one possible explanation is that for those players largely participating in group plays, achievement may not necessarily be the only source of enjoyment with the game. Players come into the game with different motivations, and therefore, in order to accurately predict players' intention to quit, predictive models must take into consideration player motivations.

7.9 Predicting Player Enjoyment - Fun

This section describes our empirical results from a large-scale experiment for predicting players' fun level. Similarly, we formulate this as a multi-class classification problem, where the objective is to predict the 'Fun' level (which, in this experiment, is treated as a categorical variable, i.e. 1, 2, 3, 4). We use Weka [29] in the experiment. Results reported in this section are also obtained from standard 10-fold cross-validation.

7.9.1 Feature Representation

The same schemes used in 'Quit' level prediction experiment are also utilized in this experiment.

7.9.2 Sampling

The same over-sampling technique used in 'Quit' level prediction experiment is also utilized in this experiment, for the same reasoning (i.e. to deal with imbalanced classification).

7.9.3 Results

Our experimentation takes 10 samples and for each sample, we perform 10-fold cross-validation. Table 7.8 shows the prediction results, averaged up across all 10 samples over 10-fold cross-validation.

Table 7.8: Prediction Results (F-measure in %) for 'Fun' - server 'Guk'

	Guk (monster kills)	
Algorithm	S1	S2
Naive Bayes	63.4	76.1
Bayesian Network	72.4	79.6
Logistic Regression	63.5	76.7
J48	75.7	99.3
Random Forest	74.5	98.7
REP Tree	75.2	94.0
Random Tree	74.3	97.4
LAD Tree	62.0	78.0
Decision Table	70.7	76.3
Decision Table & Naive Bayes	74.2	97.4
JRip	62.1	91.7
SVM (Radial)	71.8	83.3
Neural Network	71.3	92.2
SMO	64.3	77.0

Table 7.9: Prediction Results (F-measure in %) for 'Fun' - server 'Antonia Bayle'

	Antonia Bayle (monster kills)	
Algorithm	S1	S2
Naive Bayes	50.1	57.7
Bayesian Network	68.1	79.2

Logistic Regression	50.1	69.3
J48	72.6	98.3
Random Forest	51.4	97.5
REP Tree	72.1	91.1
Random Tree	71.5	95.4
LAD Tree	51.4	69.5
Decision Table	66.1	71.5
Decision Table & Naive Bayes	70.2	97.7
JRip	52.5	85.0
SVM (Radial)	65.6	79.1
Neural Network	62.7	86.8
SMO	48.7	70.1

Table 7.10: Prediction Results (F-measure in %) for 'Fun' - server 'Nagafen'

	Nagafen (monster kills)		Nagafen (PvP fights)	
Algorithm	S1	S2	S1	S2
Naive Bayes	45.1	57.2	47.5	60.1
Bayesian Network	57.8	74.4	67.8	78.9
Logistic Regression	44.9	63.4	48.4	96.3
J48	63.0	97.8	47.5	92.8
Random Forest	63.7	98.1	69.8	98.8
REP Tree	61.5	92.8	68.8	95.4
Random Tree	63.0	96.3	69.1	97.9
LAD Tree	39.8	66.0	45.7	68.7
Decision Table	63.0	85.7	66.3	71.2
Decision Table & Naive Bayes	39.8	97.3	69.5	98.0
JRip	62.1	84.5	49.2	87.2
SVM (Radial)	55.1	78.5	63.4	79.2
Neural Network	71.8	86.8	55.3	90.9

SMO	44.8	64.3	44.8	64.3

Tables 7.8, 7.9 and 7.10 show the prediction results in terms of F-measure (in percentage). Overall, predictive models built only on Skill versus Challenge information produce prediction results significantly better than those produced in 'Quit' level prediction experiments. This finding somewhat conforms to the flow theory, wherein players attain positive experiences (i.e. enjoyment) when tasks are in the flow zone (meaning that tasks are not too easy and not too challenging). We observe that when combined with Player Motivations information, prediction results significantly improve.

7.10 Conclusion

This chapter explores the relationship between player performance and player enjoyment. First, we report that players' flow state (in terms of player performance) is not a significant predictor for their intention to quit, and that information about their play motivations significantly improves the prediction results. Second, our findings indicate that players' flow state can be a predictor for their fun level, however, inclusion of Player Motivations information can further improve the prediction results. An extension to the present exploratory study is to investigate the underlying contextual explanations (i.e. analyzing J48 trees) as to how different motivations, Skill vs Challenge, and player enjoyment are linked. While this study shows that the knowledge of players' motivations is critical in predicting their enjoyment with the game (i.e. fun, quit), it is noted that in order to dynamically detect players' flow states while they play the game (i.e. for in-game task recommendation), predictive models which do not rely on survey data are desired. Hence, one future direction is to investigate players' in-game behaviors and identify in-game behavioral patterns indicative of player motivations. The results from this analysis can then supplement the player motivation variables used in this study. Lastly, as noted earlier, social activities are abundant in MMOGs, meaning that accurate predictive models must take into consideration various types of social interaction. In Chapter 5, we successfully demonstrated that inclusion of social variables such as mentoring, apprenticeship, and grouping in combats and quests led to improved player performance prediction accuracy. Hence, one future direction is to investigate 1) how different motivations drive players' social interaction in the game, 2) what is the effect of different types and varying degrees of social interaction on player performance and player enjoyment.

Chapter 8

Group Performance: Models and Metrics

This chapter examines an important aspect of MMOG game play, namely group performance. Although it has been shown that some players choose to play solo for much of their EverQuest II experience, it is commonly accepted that grouping is the heart of many MMOGs including EverQuest II. A group of up to a certain size can get together and work as a team. Many MMOGs have some tools built-in for allowing players to find other players to group with. One drawback of many of such tools is that the match-making system relies on basic demographic information such as class and level. As a result, sometimes players cannot even find a group. Even worse, players can end up grouped with players who do not get along with them. Thus, some players feel that it is too difficult to find a group and they are forced to solo. Group playing takes more than simple profile matching. Especially in higher level raid combats, group playing becomes much about coordination and inter-player interactions where each plays a different role given their respective capabilities and abilities (as described in Appendix C). This thesis work is the first comprehensive quantitative analysis of group performance in MMOGs. In this chapter, we define group performance in MMOGs and propose performance metrics with respect to combat groups in EverQuest II. Further, we examine how group composition in terms of group size and the average player level have any associations with group performance. We further examine other aspects of group composition, namely inter-player familiarity, which measures to what degree two or more group members know each other from past combats. This study then discusses the relationship between inter-player familiarity and group performance.

8.1 Background

In this study, we propose a comprehensive performance management tool for measuring and reporting operational activities of groups. This study uses performance data of game players and groups in EverQuest II to build performance prediction models for combat performing groups. The prediction models provide a projection of task performing group's future performance based on the past performance patterns of participating players on the group as well as group characteristics. While the existing game system lacks the ability to predict group-level performance, the prediction models proposed in this study are expected to be a useful addition with potential applications in player and group recommendations. First, we present group performance metrics that can be generalized to all types of games with the concept of point gain, leveling up, and session or completion time. Second, we show that larger or more advanced groups do not necessarily achieve higher group performance than smaller or less advanced groups. Third, in the next chapter, we present novel group performance prediction methods based on the past performance patterns of participating players and group characteristics and social interactions.

8.2 Group Formation in EverQuest II

Manufacturing plants over the years have adopted the formation of work groups as a practice [6, 37, 54, 59]. Many companies have adopted group approaches to produce high quality products and services which would lead to improved customer satisfaction [4]. Additionally, huge cost savings coupled with quality improvements have been reported in numerous studies [26, 34, 41, 61, 62, 79]. A more recent study conducted empirical studies on the impact of group formations at workplaces on manufacturing performance over an extended period of time [2].

As is the case in manufacturing, group formation is a common occurrence in many MMOGs [88, 86, 87]. The games are designed to encourage social interactions in such a way that certain quests must be done as a group. Not only that, many combats and quests require multiple players playing different roles. In order to successfully complete a given combat, the group members must collaborate and rely on one another. In EverQuest II, players can form a group of up to six players. They can then go on combats and quests together and share the XP points. In higher levels, players can form raid groups. Raids monsters are larger and more vicious that raid groups sometimes can include 12, 24 or more players all playing together. Also, novice players can group up with more advanced players to get familiarized with the game via the game's mentoring system, and this aspect of the game was discussed in detail in Chapter 4. Our analysis of the EverQuest II game logs shows that over 12 million groups form monthly, where a majority of them are casual or temporary groups.

8.3 Impact of Diversity and Inter-Player Familiarity on Group Performance

Diversity and familiarity amongst members of task performing teams and its impact on group performance (i.e. outcome of performing a task as a group) have been widely studied in social sciences domain [36, 32]. Many MMOG group tasks resemble real world team or group tasks. For instance, let us take an example from [36]. An MMOG combat group can be compared to a software project team. For the former, one desirable outcome is to kill a monster so that XP points can be collected and shared amongst the group members. Specifically, it is desirable to work efficiently as a group so that more XP points can be collected in less time (higher efficiency). Another desirable outcome is to sustain as less number of deaths as possible because one less player means prolonged combat time or it can also jeopardize the lives of other players. The quality of a given combat can be defined by how many players died during the combat. For a software project team, one desirable outcome is code completion. When a project is completed, for instance in a contractual situation, there is revenue gain from completing the project. Specifically, it is desirable for the software development and testing group to complete a given software project as fast as possible in the least amount of time possible (higher efficiency). Another desirable outcome is that the deployed code is of high quality where quality is defined by the total number of post-production bugs. The more post-production bugs found, the more maintenance time and revenue loss. Given this analogy, this study seeks to study the impact of diversity and familiarity on group performance. Before we embark on this research question, we first define group performance metrics below.

8.4 Group Performance Metrics

8.4.1 Efficiency Index

Group Efficiency Index is slightly different from Player Efficiency Index. The intuition behind Group Efficiency Index is the following. Given two groups with the same group composition (diversity and familiarity), given the same task, the one that completes the task faster (less session time) is said to be more efficient. This group will then attain higher Group Efficiency Index than the other group. As discussed in Chapter 2, the game logs do not explicitly log when a group of players form a group. It only logs when a task is completed. Hence, the time at which this group of players presumably got together and started a task has to be inferred. First, we take each group member and looks for his or her last activity within the last 30 minutes. The completion time of this last activity is then presumed to be the start time of this new task for this player. We repeat this procedure for all other group members. We then take the average across these presumed start times and the average time then becomes the time this group of players got together. We then take the difference between this start time and the logged group task completion time, and that becomes the session time for this group task.

8.4.2 Success Index

Success Index is a measure of what fraction of the group has survived and successfully completed a given task. Success Index is computed as follows:

$SuccessIndex_i = \frac{T_i - D_i}{T_i}$
where

D = Total number of dead members on Group i

T = Total number of members on Group i

8.4.3 Level Diversity Index

Level Diversity within a group is a measure of the average level difference amongst the group members. For a group of size N, Level Diversity Index is computed as follows:

Algorithm 11 Calculate Level Diversity Index for Group J of size N

```
M = 2 (we take a pair at a time for comparison)
temp = 0
GM[] (group members and their player levels)
for i = 1 to N do
  for j = (i + 1) to N do
    temp = temp + |GM[i] − GM[j]|
  end for
end for
Combo = N! / (M! × (N − M)!)
temp = temp/Combo
LevelDiversityIndex = temp
```

8.4.4 Class Diversity Index

We use one of the most popular diversity indexes, i.e. Simpson's Diversity Index. A diversity index is a statistic, where the computed value is used to assess the diversity of a given population, where each data point (in our case, game players) belongs to a specific type (in our case, character class). We compute Class Diversity Index as follows:

$$ClassDiversityIndex = 1 - \sum_{i=1}^{C}(p_i)^2$$

where

p_i is the ratio or fraction of all players that belong to the i-th class.

This value ranges from zero to one, where the computed values nearing zero indicate low diversity and values nearing one indicate high diversity. In MMOGs, a high diversity value would come up when a combat group consists of many different classes. In fact, this is desirable and typical because each class is equipped with different abilities and capabilities. For instance, in a raid team, there have to be healers that can heal wounded warriors while warriors themselves take a turn and rotate (freshened up ones go face the monster face-to-face while wounded ones get healed up in the back).

8.4.5 Inter-Player Familiarity Index

The simplest form of inter-player Familiarity Index measures how many of the group members have interacted in the past. It is a binary decision; flag it as 'yes' if they have interacted (in combats) at least once and flat it as 'no' otherwise. Algorithm 12 computes FamiliarityIndex-1, whose value is equal to one if all of the group members have interacted with one another in the past and zero if none of them interacted in the past.

Algorithm 12 FamiliarityIndex-1: Calculate Familiarity Index for Group J of size N

$M = 2$ (we take a pair at a time for comparison)
$temp = 0$
$Interaction[][]$ (array containing info about whether two players interacted in the past)
for $i = 1$ **to** N **do**
  for $j = (i + 1)$ **to** N **do**
    if $Interaction[i][j] == true$ **then**
      $temp++$
    end if
  end for
end for
$Combo = \frac{N!}{M! \times (N-M)!}$
$temp = temp/Combo$
$FamiliarityIndex1 = temp$

The next form of inter-player Familiarity Index measures how many of the group members have interacted in the past and incorporates the intensity (frequency) of the interaction(s). The frequency of interactions between two group members is computed over the past 30-day period. All interactions which took place beyond 30 days from the current time are discarded.

Algorithm 13 computes FamiliarityIndex-2, where 1) raw number of interactions over the past 30-day period is computed and normalized by the total number of possible combinations and 2) the computed value if penalized further by how many of the group members interacted with one another. The reasoning behind this second step is the following. Suppose that there is a group of four individuals, A, B, C, and D. A and B have interacted in the past 20 times and C and D have interacted in the past 28 times. A total of 48 interactions is normalized by a total of 24 combinations, resulting in Familiarity Index value of 2. Now, suppose that there is a group of four individuals, E, F, G, and H. E and F have interacted 8 times, E and G 10 times, E and H 10 times, F and G 8 times, F and H 6 times, and G and H 6 times. A total of 48 interactions is normalized by a total of 24 combinations, resulting in Familiarity Index value of 2. However, the former group is to be penalized further as A has never interacted with C nor D. Likewise, B has never interacted with C nor D. In the latter group, everyone knows each other from past interactions whereas in the former group, from each person's perspective, there are always two strangers. In order to penalize such cases, we normalize the computed Familiarity Index using the FamiliarityIndex-1 value (ratio).

Algorithm 13 FamiliarityIndex-2: Calculate Familiarity Index for Group J of size N

$M = 2$ (we take a pair at a time for comparison)
$temp = 0, freqsum = 0$
$Interaction[][]$ (array containing info about whether two players interacted in the past)
$InteractionFreq[][]$ (array containing info about how often two players interacted in the past)
for $i = 1$ **to** N **do**
  for $j = (i + 1)$ **to** N **do**
    if $Interaction[i][j] == true$ **then**
      $temp++$
      $freqsum+ = InteractionFreq[i][j]$
    end if
  end for
end for
$Combo = \frac{N!}{M! \times (N-M)!}$
$temp = temp/Combo$
$freqsum = freqsum/Combo$
$FamiliarityIndex2 = freqsum \times temp$

8.5 Conclusion

In this chapter, we define group performance metrics and present reasonings behind each metric. In the next chapter, we examine the relationship between 1) Level Diversity and Efficiency, 2) Familiarity and Efficiency, 3) Level Diversity and Success, and 4) Familiarity and Success. Next, we propose regression-based prediction methods for group performance.

Chapter 9

Group Performance: Prediction Models

In this chapter, we examine group performance metrics and various factors affecting group performance. First, we examine how group composition in terms of group size and the average player level can impact group performance. Lastly, we showed how inter-player familiarity can impact group performance. Next, we propose three novel group performance prediction models, namely Group-Composition-Pred, Group-Familiarity-Pred, and Group-All-Pred. The first model uses feature representation scheme where the future performance prediction for a given group is based on group composition (i.e. group size, level diversity). The second model uses feature representation scheme where the prediction depends on the degree at which group members are familiar with one another from their past in-game interactions. The third model, Group-All-Pred, which combines the previous two models into one. While there are many game-related forums and blogs where game players and group members share tips and anecdotal stories about group plays, the MMOG gaming community is lacking a systematic, data-driven group performance management tool, which this thesis work proposes. This study uses performance data of game players and groups to build performance prediction models for task performing groups. The key contribution we make in this research problem is that we introduce the first data-driven group performance management tool to the MMOG gaming community. The tool we develop not only provides retrospective analyses but also prospective and predictive analyses on group performance.

9.1 Dataset

The dataset used in this study is described in Chapter 2. As discussed in Chapter 2, the game logs do not explicitly log when a group of players form a group. It only logs when a task is completed. Hence, the time at which this group of players presumably got together and started a task has to be inferred. First, we take each group member and looks for his or her last activity within the last 30 minutes. The completion time of this last activity is then presumed to be the start time of this new task for this player. We repeat this procedure for all other group members. We then take the average across these presumed start times and the average time then becomes the time this group of players got together. We then take the difference between this start time and the logged group task completion time, and that becomes the session time for this group task. Additionally, when a player dies during combat, the remaining group (upon completing a task) will have its activity logged in the game logs without the information about dead group member(s). For our analyses, we had to weave this missing information by searching in the vicinity of where the group information was logged, in the game logs.

9.2 Experiments and Results

First, we examine the impact of group playing on player performance.

9.2.1 Impact of Group Size on Player Performance

Is it necessarily true that the more the better? Do large groups necessarily achieve high efficiency? Our results indicate that this statement is true up to a certain point. The below charts summarize our results.

The red line in Figure 9.1 shows that Efficiency Index increases up until group size of six and then it starts declining. The blue line plots the Efficiency Index computed for solo players for the same tasks that the group players reflected in the red line performed. The chart shows that some of the tasks can be done with higher efficiency by playing solo.

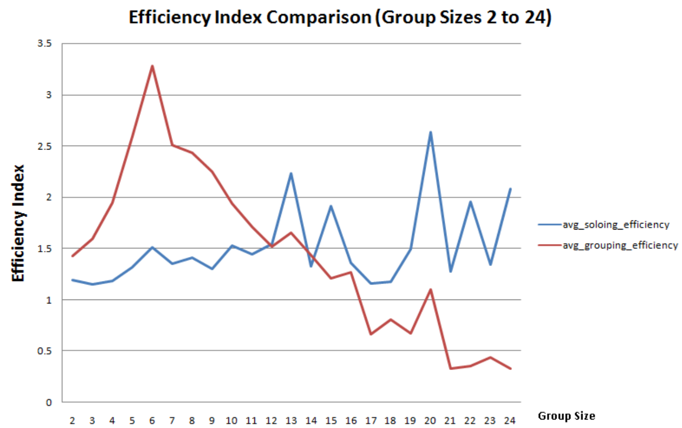


Figure 9.1: Efficiency Index - Varying Group Sizes

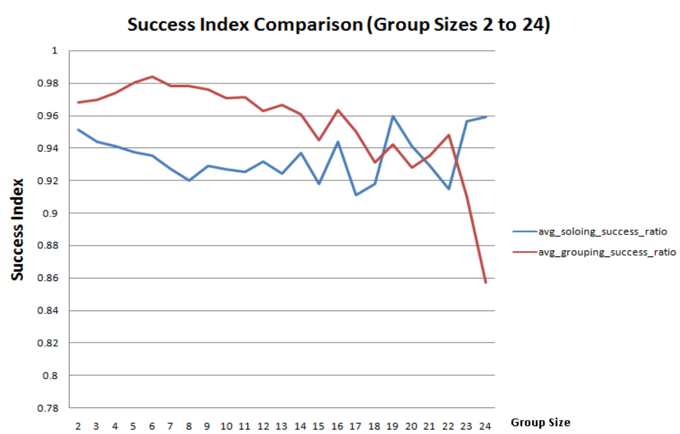


Figure 9.2: Success Index - Varying Group Sizes

Figure 9.2 shows that up until group size of six, it becomes increasingly safer to play as part of a group than playing solo. However, beyond the group size of six, the Success Index declines dramatically. The blue line plots the Success Index computed for solo players for the same tasks that the group players reflected in the red line performed. The chart shows that overall, it is still safer to play as part of a group, but as the group size becomes large beyond the size of six, we observe more occurrence of death(s) on the group.

Next, we examine factors influencing group performance. We take the following two as dependent variables: 1) Group Efficiency Index and 2) Group Success Index. And we take 1) Group Level Diversity Index and 2) Group Familiarity Index as independent variables. Their definitions are detailed in Chapter 8. We seek to understand the relationship between 1) diversity and familiarity and 2) efficiency and success of combat groups in EverQuest II.

9.2.2 Impact of Individual Player Performance on Group Performance

We performed an analysis to investigate the impact of individual player performance on group performance. Suppose that there exists a combat group of size N and the group is about to go tackle a monster. Suppose that each individual player has a good track record of being efficient and safe. When these excellent players are put together in a group, will they work cohesively well together? We take groups of size 3, 4, 5, and 6 and performed a correlation analysis. For each group, we take the average performance across all group members and performs a correlation analysis against each group performance metric. Table 9.1 shows that the average Player Efficiency Index has very little correlation with Group Efficiency Index. Table 9.2 shows that the average Player Success Index has minor correlation with Group Success Index. In summary, mere aggregation or summarization of individual player performance metrics is insufficient for explaining group performance metrics.

Table 9.1: Impact of Player Efficiency Index on Group Efficiency Index

Group Size	Correlation Coefficient

3	0.2657
4	0.1870
5	0.1373
6	0.0732

Table 9.2: Impact of Player Success Index on Group Success Index

Group Size	Correlation Coefficient
3	0.3891
4	0.4354
5	0.4036
6	0.5176

9.2.3 Impact of Task Difficulty on Group Performance

In our analysis, we examine monster kills and the task difficulty is defined as a function of monster kills. Figure 9.3 and Figure 9.4 show monster kills at various monster levels in the first week of September 2006. Figure 9.4 shows that the mean group level in group plays is in majority close to the level of the monster. As players level up, the monsters that they encounter become more vicious. Suppose that a player levels up to Level 40 and encounters Level 40 monsters. Our findings indicate that for the same task difficulty, smaller groups can take slightly longer session time to kill a given monster than larger groups, but the difference is negligible. Figure 9.3 shows that the majority of the Level 40 monster kills are performed by groups of size four, however, we see a good number of solo players. A further look into this group of solo players reveals that a majority of them are players of levels more advanced than 40.

Another investigation reveals that a majority of the higher level solo players that attempt similar level monsters are of fighter/warrior classes with heavy armors that often plays tanks in organized raid combats. We do not have many data points that show very high level players attempting to kill monsters whose levels are way below their levels. In the few cases we have observed such data points, we have found that due to the nature of the game's point scaling system [64], the XP point gain in such cases would be very minimal. Perhaps because of the low challenge level, subsequently low XP point gain, and lack of entertainment in attempting mediocre monsters, players and groups do not target tasks whose difficulty is way lower than their levels. The implication of this finding on our prediction models is that there is not much variation in Efficiency Index values due to task difficulty and that task difficulty, given the dataset we have in this analysis, would not be a good independent variable to use in our prediction models.

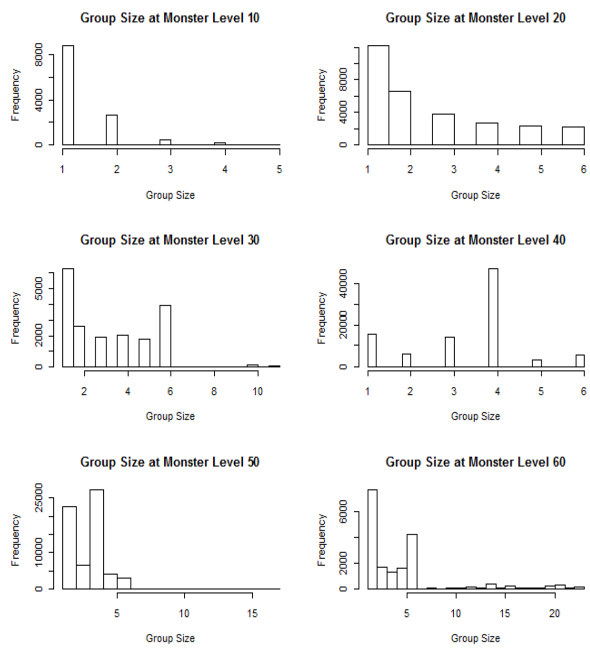


Figure 9.3: Task Difficulty and Group Size

9.2.4 Impact of Diversity and Familiarity on Group Performance

Table 9.3 shows correlation coefficients from regression analysis. It shows that Group Level Diversity Index is highly negatively correlated with both Group Efficiency Index and Group Success Index, and it is more so than Group Size. This indicates that groups where the skill difference amongst players is high are more likely to be not only inefficient but also sustaining deaths. Inter-Player Familiarity has higher correlation with Group

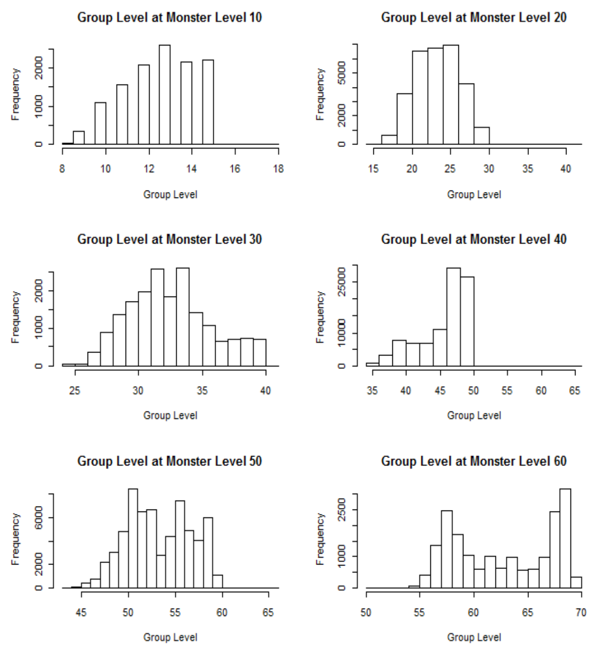


Figure 9.4: Task Difficulty and Group Level

Success Index than it is with Group Efficiency Index. This indicates that groups where members know each other from past interactions are more likely to play safely but not necessarily efficiently. Also, Class Diversity has higher correlation with Group Success Index than it is with Group Efficiency Index. This suggests that groups with a variety of skill sets are likely to play safely but not necessarily efficiently.

Table 9.3: Impact of Diversity and Familiarity on Group Performance - Guk

Variables	Correlation Coefficient (aggregated over all levels)
Level Diversity & Efficiency	-0.7472
Level Diversity & Success	-0.7053
Class Diversity & Efficiency	0.5200
Class Diversity & Success	0.7406
Familiarity-1 & Efficiency	0.5235
Familiarity-1 & Success	0.7645
Familiarity-2 & Efficiency	0.5527
Familiarity-2 & Success	0.7626
Group Size & Efficiency	0.6268
Group Size & Success	0.6410

9.2.5 Prediction of Group Performance

Based on the findings so far, we report three different regression-based models for group performance prediction. We propose three novel group performance prediction models, namely Group-Composition-Pred, Group-Familiarity-Pred, and Group-All-Pred. The first model uses feature representation scheme where the future performance prediction for a given group is based on group composition (i.e. group size, level and class diversity). The second model uses feature representation scheme where the prediction depends on the degree at which group members are familiar with one another from their past in-game interactions (Inter-Player Familiarity Index). The third model, Group-All-Pred, which combines the previous two models into one.

Table 9.4: Summary of Regression Models - Prediction of Efficiency Index

Model Name	P-value	R-squared value
Group-Composition-Pred	less than 2.2e-16	0.6627
Group-Familiarity-Pred	less than 2.2e-16	0.5363
Group-All-Pred	less than 2.2e-16	0.6478

Table 9.5: Summary of Regression Models - Prediction of Success Index

Model Name	P-value	R-squared value
Group-Composition-Pred	less than 2.2e-16	0.7357
Group-Familiarity-Pred	less than 2.2e-16	0.7568
Group-All-Pred	less than 2.2e-16	0.7439

Tables 9.4 and 9.5 summarize the three regression models for the prediction of Group Efficiency Index and Group Success Index. Our results show that for Group Efficiency Index prediction, Group-Composition-Pred is the best performing model. As shown in Table 9.3, class diversity has no significant impact on Group Efficiency Index. Further, past interactions amongst group members have no significant impact on Group Efficiency Index. However, our findings suggest that both Class Diversity and Familiarity are positively correlated with Group Success Index. With respect to Class Diversity, by game design, this finding is expected as the game mechanics encourage players of diverse classes to work together. With respect to inter-player familiarity, this finding conforms to findings from social sciences domain as discussed in [36, 32].

9.3 Conclusion

In this chapter, we examine group performance in EverQuest II. First, we examined the impact of group playing on player performance. First, we report that as player level increases, it becomes more efficient to play as part of a group. Second, we report that large groups do not necessarily achieve high efficiency. Our results show that six is the golden number for the number of group members, as efficiency (at the individual player level) starts degrading beyond the group size of six. Next, we examine group performance. First, we show that Group Level Diversity Index is highly negatively correlated with both Group Efficiency Index and Group Success Index, and it is more so than Group Size. This indicates that groups where the skill difference amongst players is high are more likely to be not only inefficient but also sustaining deaths. Second, we show that Inter-Player Familiarity has higher correlation with Group Success Index than it is with Group Efficiency Index. This indicates that groups where members know each other from past interactions are more likely to play safely but not necessarily efficiently. Third, Class Diversity has higher correlation with Group Success Index than it is with Group Efficiency Index. This suggests that groups with a variety of skill sets are likely to play safely but not necessarily efficiently. Based on these findings, we built three regression-based prediction models and used R-squared value as the evaluation metric. We report that Group-Composition-Pred is the best performing method for Group Efficiency Index prediction and its feature representation scheme is solely based upon group compositional data such as level and class diversity as well as group size. Next, we report that for Group Success Index prediction, all three models perform comparatively well with the ability to explain 73% to over 75% of the variance in the data. Group-Familiarity-Pred is the highest performing method, however, in absence of players' past data (which can be the case for early level players), the other two models perform comparatively well.

Chapter 10

Implementation of MMOG Player and Group Behavior Analysis Framework

In this chapter, we contribute an implementation of a game player and group behavior analysis framework to the MMOG gaming community. The framework is written entirely using open-source tools and components. The core part of the framework consists of all of the aforementioned analysis and prediction modules, all written in Java programming language. It uses Weka [29], an open source data mining suite, for certain algorithms, JFreeChart [97] for charting, and JasperReports [98] for report generation. The framework also uses Hibernate [99] to configure data models, and the data connection module currently supports MySQL database and Microsoft SQL Server 2005 or above.

10.1 Overview

As part of this thesis work, we architected and implemented a behavior analysis framework for player and group behaviors in Massively Multiplayer Online Games. In this chapter, we discuss the overall system architecture.

10.2 Architecture

Figure 10.1 below shows the overview of our behavioral analysis framework system. First, we need to fetch data from data source. Currently, our system accepts MySQL, Oracle, and MS SQL Server 2005 or above as the data source. The system also accepts CSV files and DDL's including SQL statements for local data loading. The next component is the Extract-Transform-Load component. For this, we have taken three different approaches; 1) raw SQL statements stored as a SQL script file, an XML configuration file where users can specify data loading information (location of CSV files and SQL script file), and an independently executable Java program for kicking off data loading jobs; 2) in Phase 2 development, we use MS SQL Server 2008 as the destination/local data source, for which we use MS SQL Server SSIS program for ETL loading; 3) Pentaho data integration, which is freely available and comes with a suite of applications for data loading. In order for our algorithms to run, we need data to be aggregated at the individual player level and for group performance algorithms to run, we need to extract groups and also perform data aggregation. Collectively, we refer to this as a data aggregation module and the web-based user interface allows users to kick off data aggregation jobs and also monitor the progress of these jobs. Once data aggregation is done, we can then execute our algorithms. Most of the algorithm executions require parameters to be specified. Users can directly edit XML configuration file as shown in Figure 10.2 to specify job details or they can use the web-based user interface to fill out a form and submit a job. For all "test" jobs ("train" will build classification models and save the models and "test" will take an input file containing one or more entities i.e. players, groups and perform prediction), the Java written modules will load output data into a local datamart. Once data are all processed and new data become available in the datamart, which can then be viewed via the web-based user interface.

10.3 Warehouse Database and ETL Pipeline

10.3.1 Data Modeling

The original EverQuest II game logs contain a lot more information than what has been used in this thesis work. For the purposes of building this analysis framework per this

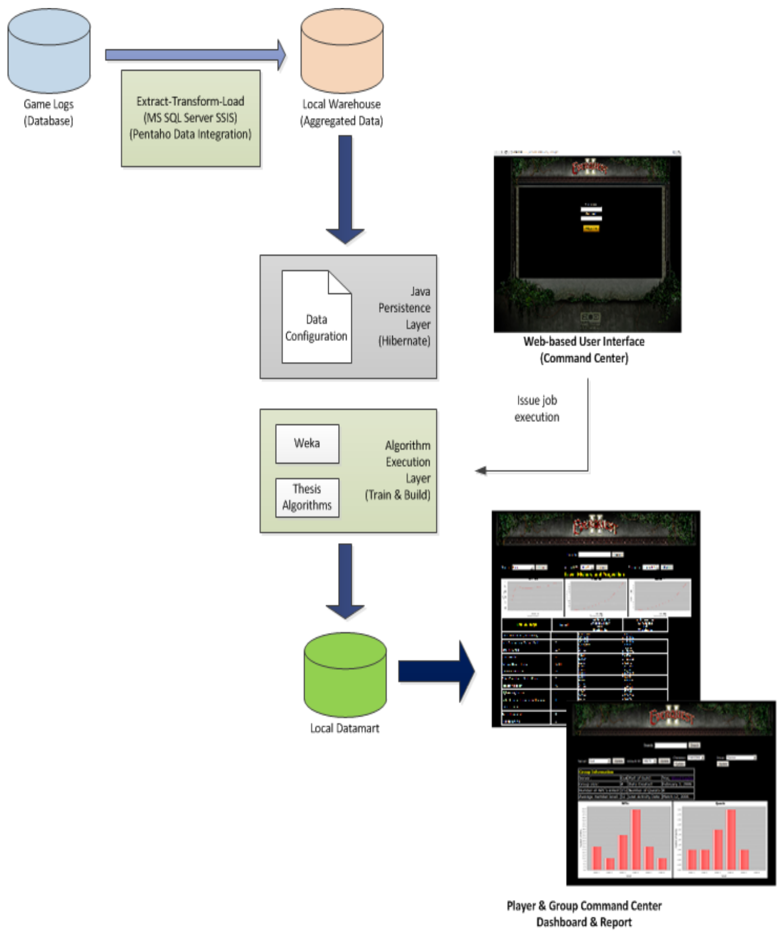


Figure 10.1: Behavior Analysis Framework - System Overview

```xml
<request>
 <job-source-host>pacman.cs.umn.edu</job-source-host>
 <job-source-username>kjshim</job-source-username>
 <job-submit-time>2011-10-24 14:09:51</job-submit-time>
 <job-category>ipp</job-category>
 <job-algorithm>nw-2</job-algorithm>
 <job-type>test</job-type>
 <job-output>
  <param>com.shim.ipp.models</param>
 </job-output>
 <job-params>
  <param>3</param>
  <param>decaying</param>
  <param>0.5;0.25;0.25</param>
  <param>6</param>
  <param>10</param>
 </job-params>
 <job-source>
  <param>pacman.cs.umn.edu</param>
  <param>3306</param>
  <param>bogus</param>
  <param>bogus123</param>
  <param>vwe</param>
  <param>xp_guk_2006_ordered_duplicate_analysis2_summary6</param>
  <dependent>efficiency_i_f1</dependent>
  <independent>efficiency_i_p1;ratio_mentor_i_p1;ratio_group_i_p1;</independent>
  <independent>efficiency_i_p2;ratio_mentor_i_p2;ratio_group_i_p2;</independent>
  <independent>efficiency_i_p3;ratio_mentor_i_p3;ratio_group_i_p3;</independent>
 </job-source>
</request>
```


Figure 10.2: Behavior Analysis Framework - Request XML Configuration File

thesis work, we took only the data relevant to this research and created a separate back-end warehouse database. Figure 10.3 shows an Entity-Relationship diagram depicting the data model of the warehouse database used in this thesis work. The main entities are game accounts, characters, tasks, etc. The Activity table is the main table which contains records of all in-game activities performed by game players. Activities include monster kills, quests, mentoring, apprenticing, and etc.

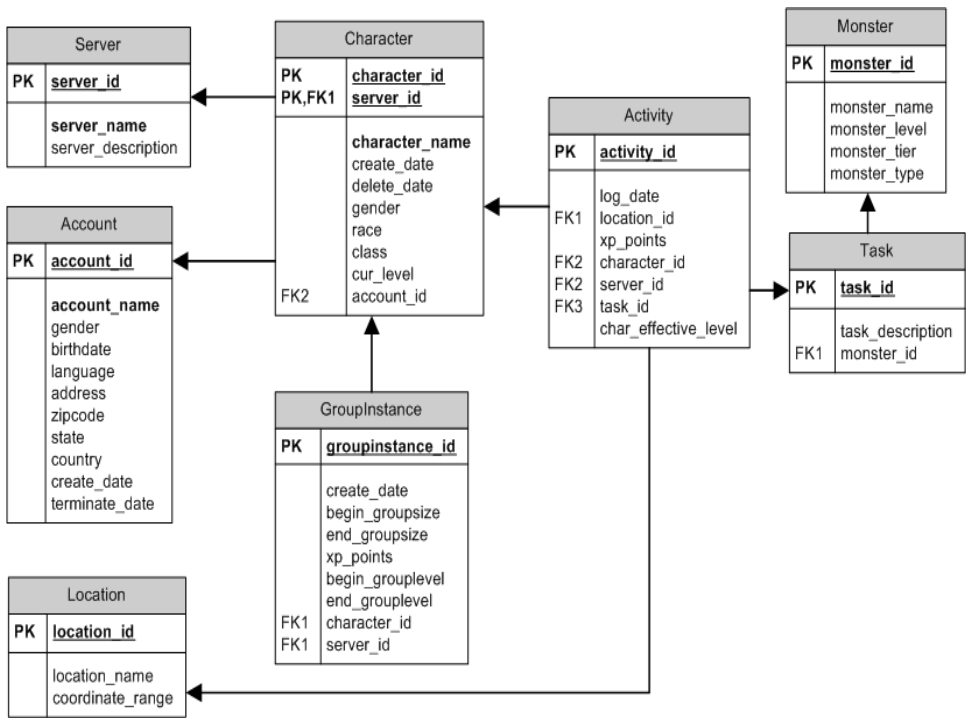


Figure 10.3: Behavior Analysis Framework - E-R Diagram of Warehouse Database

10.3.2 Implementation

In Phase 1, we used an open source MySQL Server Community Edition 5.0 [96] on a Dell PowerEdge R510 rack server with the following specifications; 2 CPU's (Intel Xeon X5560 2.8Ghz 8M Cach), 24 GB Memory (1333MHz Dual Ranked RDIMMs for 2 Processors), Ubuntu OS, 14 TB SATA hard drive. In Phase 2, we created a replica of this warehouse database in Microsoft SQL Server 2008 Enterprise Edition also on a

Dell PowerEdge R510 rack server with the same specifications, except the OS: Windows Server 2008 R2.

The original game logs came in two different formats: 1) de-normalized data dumps in flat tables, 2) flat files in delimited format. The framework we built had to be flexible enough to allow for these two different input file types. In Phase 1 where we used MySQL as the destination warehouse database, we wrote a Java program and used JDBC to connect directly to the source database (i.e. Oracle database), properties files to store data loading SQL statements, and a Java program which can be triggered manually or via crontab command in Linux/Unix OS. Later on, we replaced this layer with Hibernate 3.x, in order to better handle real-time data pulls. In Phase 2, we used the Microsoft SSIS software to set up ETL packages, in which we specified a set of SQL statements to pull data from the source and load onto the destination warehouse database. SSIS packages can then be triggered either manually from Java programs (both locally and remotely) or via DOS command line (locally). These jobs can also be scheduled to get trigger on a regular basis or at a set time of choice.

10.4 Analysis Programs

All of the algorithms developed in this thesis work have been implemented in Java programming language and have been integrated into the Behavior Analysis Framework. Inner-workings of all of the algorithms have been discussed in previous chapters. We use Weka [29] 3.6 JAR file as part of this analysis framework and for certain algorithms such as K-nearest neighbor, we use a modified version of their implementation, especially the two novel distance functions discussed in previous chapters on player performance.

10.5 User Interface

10.5.1 Player Command Center

The first part of the Player Command Center allows users to browse through game characters. Users can use the search box to search by character ID number (i.e. 30291840) of character ID name (i.e. 'Soliath'). The search operation will then pull up all entries matching the search criteria. For instance, the same named characters can exist on multiple servers, in which case, the search will show all characters. Users can then click on each and further view the character information of the selected game character. Alternatively, users can start with Game Server selection. For instance, users can choose in the left-most drop-down menu the server 'Guk' and click on Update button. It will then narrow down the search space of accounts. Users can use the middle drop-down menu to select a particular account and click on Update. It will then pull up all characters owned by the selected account. Users can use the right-most drop-down menu to select a particular game character. Clicking up Update button will pull up all information about the selected game character and display it in the main screen as shown in Figures 10.4 and 10.5.

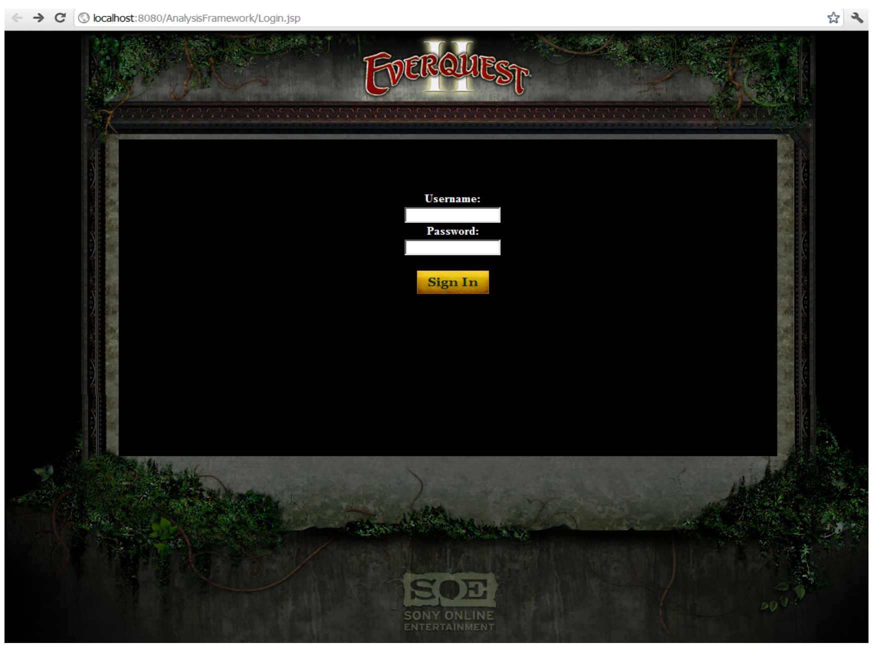


Figure 10.4: Behavior Analysis Framework - Login Screen

10.5.2 Group Command Center

This is the main panel for users to browse through combat groups on various game servers. Users can use the search box to search by group name. The search operation

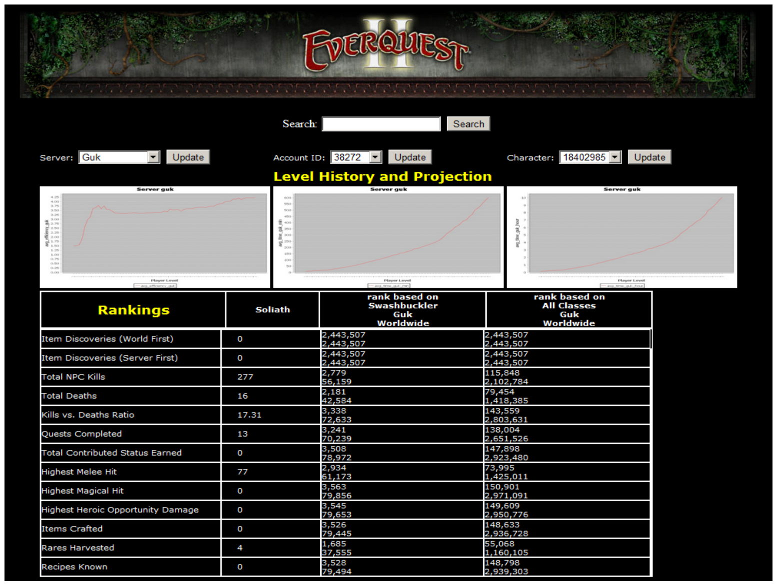


Figure 10.5: Behavior Analysis Framework - Player Command Center

will then pull up all combat groups matching the search criteria. As is the case for player characters, the same named characters can occur on multiple servers, in which case, the search will pull up all groups. Users can then click on each and further view the group information of the selected group. Alternatively, users can start with Game Server selection. For instance, users can choose in the left-most drop-down menu the server 'Guk' and click on Update button. It will then narrow down the search space of accounts. Users can use the middle drop-down menu to select a particular account and click on Update. Users can use the middle drop-down menu to select a particular account and click on Update. It will then pull up all characters owned by the selected account. Users can use the right-most drop-down menu to select a particular game character. Next, it will pull up all combat groups this character has been part of. Users can then select a particular group and click on Update button to view the group information in the main section of the interface. Figure 10.6 shows the Group Command Center panel.

Figure 10.6: Behavior Analysis Framework - Group Command Center

Chapter 11

Conclusion and Discussion

Chapter 3 analyzes EverQuest II's player performance data to devise individual player performance metrics. First, our analysis reveals that the game's existing ding points-based point-scaling system is in general well in accordance with the actual player performance observed in the game's historical performance data. It also reveals that the level of granularity that the performance data offers can potentially lead to fine tuning of the existing point-scaling system. Secondly, the proposed performance metrics define performance as a function of productivity and/or quality. Our findings demonstrate that a given player's past performance can be used as a predictor for his future performance. Our findings also indicate that the proposed performance metrics yield less than optimal predictions about individual players' future performances in higher levels where group formation increasingly becomes a common occurrence. Additionally, the study reveals that in a certain type of task (i.e. monster kills), the quality aspect of individual player performance plays an insignificant role in predicting a player's future performance. Future directions include studying different ways of defining quality in all types of task in the game to devise more generalizable individual player performance metrics, conducting a more thorough and comprehensive analysis on the impact of group compositions and social interactions on individual player performance, investigating individual learning trajectory over a larger time span, and developing group performance metrics.

In Chapter 4, we examine the effects of mentoring on player performance in EverQuest II. As shown in Figure 4.4, participation rate is very low across all player levels. This brings up an interesting question; how can the game attract more players to participate in mentoring, as a mentor and as an apprentice. Our findings show that participating in mentoring as a mentor may not appeal to achievement-oriented players whose primary objective is to achieve XP points as fast as possible and move up to higher levels fast. Similarly for apprentices, as Figure 4.11 shows, until Player Level 25 or so, it is beneficial to participate in mentoring as an apprentice in terms of Combat Efficiency. However, as player level increases, its effect diminishes. In summary, our findings indicate that past mentoring and apprenticeship activities can affect future performance. Hence, it is important, when modeling player's past behaviors for the purposes of predicting his or her future behavior, the modeling techniques take into account past mentoring and apprenticeship activities. The next chapter incorporates these findings into future performance prediction models.

In Chapter 5, we show several prediction methods (MARCEL and its variations, Linear Regression, and Neighbor-Weight methods) for player performance prediction. For both Efficiency Index prediction and Success Index prediction, our experimental results show that our novel prediction methods (Neighbor-Weight-based methods) outperform existing methods. First, we show that by assigning more importance to more recent past performance metrics and less importance to distant past (Even-Weight scheme versus Decaying-Weight scheme), it leads to better prediction accuracy (lower RMSE values). Next, we show that inclusion of mentoring-related and other social variables from the game logs (such as grouping information) leads to better prediction. Prediction models we propose in this study are expected to be a useful addition to many existing player performance monitoring tools by 1) providing a projection of a given player's future performance given his or her past performance and 2) showing how player's social activities such as mentoring and grouping can affect his or her performance. Game player performance data such as that of EverQuest II is rich of not only outcome data (i.e. number of monsters killed, number of experience points gained, number of deaths occurred, number of quests completed in a given time duration) but also process data, from which we can construct a progression of a given player's performance at any given time point. Existing player performance monitoring tools can be greatly enhanced to dynamically capture player performance progression, provide instant feedback on player's progress, and recommend tasks tailored towards a given player's objectives of playing the game (performance-oriented tasks vs. social activity-oriented). Systematic studies of game player performance is expected to yield the following contributions. First, analysis of player performance in different dimensions (i.e. player demographics, archetypes, classes, sub-classes) can help game developers understand whether their games and game characters are being played as intended. Second, benefits for game players are two fold. a) Game players can not only have a view of their past and current performance but also they can have a view of their projected future performance. b) A recommendation engine can be built to recommend character types and tasks to players in order to meet certain objectives (i.e. move up to the next level as fast as possible, play safe by attempting easy tasks, play aggressively by attempting challenging tasks, play tasks that encourage grouping with other players). Third, players can have a view of performances of other players for the purposes of forming quest or raid teams.

Chapter 6 introduces a sequence alignment-based behavior analysis framework (SABAF) developed for inactivity prediction. Our findings indicate that the Support Vector Machine (using Linear kernel) classifier combined with the Global & Local feature selection scheme produces the highest overall True Positive percentage. In most of the classification algorithms, all of the three SABAF feature selection schemes produce comparative prediction coverages with the exception of Logistic Regression, SVM (RBF kernel), and SVM (Linear kernel). We report that no one feature extraction scheme or no one classification algorithm works best across all the player levels. For instance, the SVM (Linear kernel) classifier produces the highest overall True Positive percentage, however, in player levels 11, 20, 28, 43 and 44, the baseline feature selection scheme produces a considerably higher coverage. With respect to the first 23 player levels (comprising 80% of inactive player population), in most of the classification algorithms, the SABAF feature selection schemes lead to inactivity prediction coverage higher than that produced by using the baseline feature selection scheme.

Overall, the SABAF system performs better than the baseline in lower levels (23 player levels), covering some 80% of the inactive player population. However, the low True Positive percentage in higher levels mean potentially inactive players missed out. In order to achieve an even higher overall coverage with an improved True Positive percentage in higher levels, it is best to use the baseline methods in higher levels while using the SABAF methods in lower levels. In terms of True Negatives, while the baseline methods consistently produce near 100% True Negative percentages as the player level goes up, the True Negative percentages produced by the SABAF methods decrease, indicating that the SABAF methods in higher levels generate more False Positives than the baseline method.

One drawback of the SABAF system and our best performing methods (Support Vector Machines) is that while the prediction accuracy is significantly higher than other methods, it is a black-box approach and thereby, underlying reasonings behind why certain players are likely to be active versus inactive cannot be analyzed. With millions of dollars in stake for designing and executing marketing strategies (such as customer intervention strategies), often times it is desirable to see beforehand what all factors lead to potential inactivity in terms of game play. Once game companies understand what are the factors comprising potential inactivity in the future, data-driven marketing strategies with a certain level of confidence (i.e. likelihood of a certain strategy working towards particular population segments) can be carefully designed and executed with minimized task. For such studies, often times companies resort to such algorithms as J48 or JRip which can produce rule sets with independent variables and their values explicitly identified (white-box approach) and their relationships to the dependent variable (i.e. inactivity in the future) also identified. Such rule sets can be directly interpreted by game designers or even marketing personnel.

An extension to the current work involves segmenting the players by character class. A previous study [70] reports that the selection of class at character creation limits the character to certain activities as reflected in player-to-task interaction records in the game logs, hence, the activity signatures are different from one class to another. The current implementation of the SABAF system lumps all classes into one bucket due to the fact that segmentation by class leads to buckets too small for algorithms such as Support Vector Machines to train on. Yet another addition to this study is to leverage a variety of social networks in EverQuest II (i.e. housing trust network, raid group network, and guild network) to further segment the player population based on social interactions over time and perform sequence alignment within each segment.

Chapter 7 explores the relationship between player performance and player enjoyment. First, we report that players' flow state (in terms of player performance) is not a significant predictor for their intention to quit, and that information about their play motivations significantly improves the prediction results. Second, our findings indicate that players' flow state can be a predictor for their fun level, however, inclusion of Player Motivations information can further improve the prediction results. An extension to the present exploratory study is to investigate the underlying contextual explanations (i.e. analyzing J48 trees) as to how different motivations, Skill vs Challenge, and player enjoyment are linked. While this study shows that the knowledge of players' motivations is critical in predicting their enjoyment with the game (i.e. fun, quit), it is noted that in order to dynamically detect players' flow states while they play the game (i.e. for in-game task recommendation), predictive models which do not rely on survey data are desired. Hence, one future direction is to investigate players' in-game behaviors and identify in-game behavioral patterns indicative of player motivations. The results from this analysis can then supplement the player motivation variables used in this study. Lastly, as noted earlier, social activities are abundant in MMOGs, meaning that accurate predictive models must take into consideration various types of social interaction. In Chapter 5, we successfully demonstrated that inclusion of social variables such as mentoring, apprenticeship, and grouping in combats and quests led to improved player performance prediction accuracy. Hence, one future direction is to investigate 1) how different motivations drive players' social interaction in the game, 2) what is the effect of different types and varying degrees of social interaction on player performance and player enjoyment.

In Chapters 8 and 9, we examine group performance in EverQuest II. We define group performance metrics and present reasonings behind each metric. First, we examined the impact of group playing on player performance. First, we report that as player level increases, it becomes more efficient to play as part of a group. Second, we report that large groups do not necessarily achieve high efficiency. Our results show that six is the golden number for the number of group members, as efficiency (at the individual player level) starts degrading beyond the group size of six. Next, we examine group performance. First, we show that Group Level Diversity Index is highly negatively correlated with both Group Efficiency Index and Group Success Index, and it is more so than Group Size. This indicates that groups where the skill difference amongst players is high are more likely to be not only inefficient but also sustaining deaths. Second, we show that Inter-Player Familiarity has higher correlation with Group Success Index than it is with Group Efficiency Index. This indicates that groups where members know each other from past interactions are more likely to play safely but not necessarily efficiently. Third, Class Diversity has higher correlation with Group Success Index than it is with Group Efficiency Index. This suggests that groups with a variety of skill sets are likely to play safely but not necessarily efficiently. Based on these findings, we built three regression-based prediction models and used R-squared value as the evaluation metric. We report that Group-Composition-Pred is the best performing method for Group Efficiency Index prediction and its feature representation scheme is solely based upon group compositional data such as level and class diversity as well as group size. Next, we report that for Group Success Index prediction, all three models perform comparatively well with the ability to explain 73% to over 75% of the variance in the data. Group-Familiarity-Pred is the highest performing method, however, in absence of players' past data (which can be the case for early level players), the other two models perform comparatively well.

Lastly, we contribute an implementation of a game player and group behavior analysis framework to the MMOG gaming community. The framework is written entirely using open-source tools and components. The core part of the framework consists of all of the aforementioned analysis and prediction modules, all written in Java programming language. It uses Weka [29], an open source data mining suite, for certain algorithms, JFreeChart [97] for charting, and JasperReports [98] for report generation. The framework also uses Hibernate [99] to configure data models, and the data connection module currently supports MySQL database and Microsoft SQL Server 2005 or above.

References

[1] Allen, T. D., Poteet, M. L., & Burroughs, S. M. (1997). The mentor's perspective: A qualitative inquiry and future research agenda. Journal of Vocational Behaviour, 51, 70-89.

[2] Banker, R. D., Field, J. M., Schroeder, R. G., & Sinha, K. K. (Aug., 1996). Impact of Work Teams on Manufacturing Performance: A Longitudinal Field Study. The Academy of Management Journal, Vol. 39, No. 4, pp. 867-890.

[3] Baybars, I. (1986). A survey of exact algorithms for the simple assembly line balancing problem. Mgmt. Sci. 32, 909-932.

[4] Boyett, J. H. & Conn, H. P. (1991). Workplace 2000: The revolution reshaping American business. New York: Penguin.

[5] Burez, J., Van den Poel, D. (2008) "Handling class imbalance in customer churn prediction", Expert Systems with Applications, In Press, available online 16 May 2008.

[6] Business Week. (1994). The team player: no more "same-ol'-same-ol'." October 17, 95-96.

[7] Carrillo, H., and Lipman, D. The multiple sequence alignment problem in biology. SIAM Journal of Applied Math 48(5): 1073-1082, 1988.

[8] Castronova, E. (2005). Synthetic worlds: The business and culture of online games. Chicago: University of Chicago Press.

[9] Chakravarty, A. K. & Shtub, A. (1985). Balancing mixed model lines with in-process inventories. Mgmt. Sci. 31, 9.

[10] Charles, D. et al. 2005. Player-centred game design: Player modelling and adaptive digital games. In Proceedings of DiGRA 2005 Conference: Changing Views - Worlds in Play.

[11] Chen, J. 2007. Flow in games (and everything else), Communications of the ACM, 50, n. 4 (April 2007), 31-34.

[12] Chen, H., Zhang, Y., Bainbridge, W. S., Shim, K. J., Pathak, N., Ahmad, M. A., DeLong, C., Borbora, Z., Mahapatra, A., Srivastava, J. "AI, Virtual Worlds, and Massively Multiplayer Online Games," IEEE Intelligent Systems, January/February 2011, pp. 80-89.

[13] Coussement, K., Van den Poel, D. (2006) "Churn Prediction in Subscription Services: an Application of Support Vector Machines While Comparing Two Parameter-Selection Techniques", Working Paper 2006/412, Ghent University.

[14] Csikszentmihalyi, M.: Flow: The psychology of optimal experience. Harper Row, New York (1990).

[15] Dar-El, E. M. (1991). Design of assembly systems "DAS", ICPR. 1991, Workshop 1, 18-23 August 1991, Hefei, China.

[16] Datta, P., Masand, B., Mani, D.R. and Li, B., Automated cellular modeling and prediction on a large scale, Artif. Intell. Rev., vol. 14, no. 6, pp. 485502, 2000.

[17] Drachen, A., Canossa, A. and Yannakakis, G. N.. 2009. Player modeling using self-organization in tomb raider: underworld. In Proceedings of the 5th international conference on Computational Intelligence and Games (CIG'09). IEEE Press, Piscataway, NJ, USA, 1-8.

[18] Fremery, F. (1991). Model-mix balancing: more flexibility to improve the general results. In: Production Research: Approaching the 21st. century, London.

[19] Gao, Y. (2004). Appeal of online computer games: A user perspective. The Electronic Library, 22, 74-78.

[20] Gee, J. P. (1999). An introduction to discourse analysis: Theory and method. New York: Routledge.

[21] Gee, J. P. (2003). What videogames have to teach us about learning and literacy. New York: Palgrave Macmillan.

[22] Gelfand, M., Mironov, A., and Pevzner, P. Gene recognition via splices sequence alignment. Proc. Natl. Acad. Sci. USA, 93:9061-9066, 1996.

[23] Goad, W. and Kanehisa, M. Pattern recognition in nucleic acid sequences: a general method for finding local homologies and symmetries. Nucleic Acids Research, 10: 247-263, 1982.

[24] Gosh, S. & Gagnon, R. J. (1989). A comprehensive literature review and analysis of the design, balancing and scheduling of assembly systems. Int. J. Prod. Res. 27, 637-670.

[25] Granger, C. W. J. (1996), "Can We Improve the Perceived Quality of Economic Forecasts?" Journal of Applied Econometrics, 11, 455-473.

[26] Gupta, Y.P. & Ash, D. (1994). Excellence at Rohm and Haas Kentucky: A case study of work team introduction in manufacturing. Production and Operations Management, 3(3): 186-200.

[27] Hadden, J., Tiwari, A., Roy, R., Ruta, D. (2005) "Computer assisted customer churn management: State-of-the-art and future trends", Computers & Operations Research, 34, pp. 2902-2917.

[28] Hadden, J., Tiwari, A., Roy, R., and Ruta, D. (2006) "Churn prediction using complaints data," in Proceedings Of World Academy Of Science, Engineering and Technology.

[29] Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I. H. (2009); The WEKA Data Mining Software: An Update; SIGKDD Explorations, Volume 11, Issue 1.

[30] Hamuro, Y., Kawata, H., Katoh, N. and Yada, K., "A Machine Learning Algorithm for Analyzing String Patterns Helps to Discover Simple and Interpretable Business

Rules from Purchase History," Progress in Discovery Science, LNAI 2281, Springer, pp.565-575, 2002.

[31] Hamuro, Y., Katoh, N., Ip, E. H., Cheung, S. L., Yada, K., "Combining Information Fusion with String Pattern Analysis: A New Method for Predicting Future Purchase Behavior," V. Torra(ed.), Information Fusion in Data Mining, Studies in Fuzziness and Soft Computing, Vol.123, Springer, pp.161-187, 2003.

[32] Harrison, D. A., Mohammed, S., McGrath, J. E., Florey, A. T., Vanderstoep, S. W. "Time matters in team performance: Effects of member familiarity, entrainment, and task discontinuity on speed and quality". Personnel Psychology; Autumn 2003; 56, 3; ABI/INFORM Global pg. 633.

[33] He, S., Gao, Y., Yang, J., Fu, Y., Liu, X. Creating Challengeable and Satisfactory Game Opponent by the Use of CI Approaches. International Journal of Advancements in Computing Technology, Volume 2, Number 1, March 2010.

[34] Hitchcock, N. A. (1993). Employee participation paves way to product perfection. Modern Material Handling, 48(6): 44-45.

[35] Hsu, S. H., Wen, M.-H., Wu, M.-C. Exploring user experiences as predictors of MMORPG addiction. Computers & Education. Volume 53, Issue 3, November 2009, Pages 990-999.

[36] Huckman, R. S., and Staats, B. R. "Fluid Tasks and Fluid Teams: The Impact of Diversity in Experience and Team Familiarity on Team Performance." Knowledge Creation Diffusion Utilization (2011) : n. pag.

[37] Ichniowski, C., Shaw, K., & Prennushi, G. (1994). The impact of human resource management practices on productivity. Working paper 014, Columbia University, Carnegie Mellon University, and World Bank.

[38] Jansz, J., Tanis, M.: Appeal of playing online first person shooter games. Cyber Psychology and Behavior 10, 133-136 (2007).

[39] Jewels (2002, December 9). But in the end, they're still nothing more than video games. Jive Magazine.

[40] Kaplan, R. (Oct 1983). Measuring Manufacturing Performance: A New Challenge for Managerial Accounting Research, The Accounting Review, Vol. 58, No. 4., pp. 686-705.

[41] Katzenbach, J. R. & Smith, D. K. (1993). The wisdom of teams: Creating the high-performance organizations. New York: HarperCollins.

[42] Klimmt, C.: Dimensions and determinants of the enjoyment of playing digital games: A three-level model. In: Copier, M., Raessens, J. (eds.) Level Up: Digital Games Research Conference, pp. 246-257. Faculty of Arts, Utrecht University, Utrecht (2003).

[43] Kram, K. E. (1996). A relational approach to career development. In D. T. Hall, & Associates (Eds.), The career is dead-long live the career: A relational approach to careers (pp. 132-157). San Francisco: Jossey-Bass.

[44] Lave, J. & Wenger, E. (1991). Situated learning: Legitimate peripheral participation. Cambridge: Cambridge University Press.

[45] Lenhart, A., Jones, S., & Macgill, A. (2008). Pew Internet Project Data Memo: Over half of American adults play video games. Washington, D.C.: Pew Internet & American Life Project.

[46] Liu, D., Liu, J., Kwan, H. K., Mao, Y. "What can I gain as a mentor? The effect of mentoring on the job performance and social status of mentors in China," Journal of Occupational and Organizational Psychology (2009), 82, 871-895.

[47] Macaskill, J. L. C. (1972). Production-line balances for mixed-model lines. Mgmt. Sci. 19, 423-434.

[48] MacQueen, J. B. (1967): "Some Methods for classification and Analysis of Multivariate Observations, Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability", Berkeley, University of California Press, 1:281-297.

[49] Manufacturing Studies Board. (1986). Human resources practices for implementing advanced manufacturing technology. Washington, DC: National Academy Press.

[50] Morik, K. and Kopcke, H., "Analysing customer churn in insurance data: a case study," in PKDD 04: Proceedings of the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases, (New York, NY, USA), pp. 325336, Springer-Verlag New York, Inc., 2004.

[51] Nacke, L. and Lindley, C. A. 2008. Flow and immersion in first-person shooters: measuring the player's gameplay experience. In Proceedings of the 2008 Conference on Future Play: Research, Play, Share (Toronto, Canada). Future Play '08. ACM, New York, NY, pp.81-88.

[52] Needleman, S. B. and Wunsch, C. D. (1970). "A general method applicable to the search for similarities in the amino acid sequence of two proteins". Journal of Molecular Biology 48 (3): 4433.

[53] New York Times. (1994). A new labor design at Levi Strauss. october 13: C1-C2.

[54] Osterman, P. (1994). How common is workplace transformation and who adopts it? Industrial and Labor Relations Review, 17: 173-188.

[55] Pelleg, D. and Moore, A. W. (2000) X-means: Extending K-means with Efficient Estimation of the Number of Clusters. In Proceedings of the Seventeenth International Conference on Machine Learning (ICML '00), Pat Langley (Ed.). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 727-734.

[56] Prima Development. Everquest II: Official Strategy Guide (Prima Official Game Guides).

[57] Prinzie, A., Van den Poel, D., "Incorporating sequential information into traditional classification models by using an element/position-sensitive SAM," DECISION SUPPORT SYSTEMS, Volume 42, Issue 2, pages 508-526, Nov 2006.

[58] Ryan, R., Rigby, C.S., Przybylski, A.: The motivational pull of video games: A self-determination theory approach. Motivation and Emotion 30, 347-363 (2006).

[59] Safizadeh, M. H. (1991). The case of workgroups in manufacturing operations. California Management Review, 33(4): 61-82.

[60] Satariao, A. (2010). Online game system to start in June at $14.95 a month: Bloomberg.

[61] Schilder, J. (1992). Work teams boost productivity. Personnel Journal, 71: 67-71.

[62] Sheinkman, J. (1993). Amalgamated clothing and textile workers union. Quality Progress, 26(9): 57-58.

[63] Sherry, J. L.: Flow and media enjoyment. Communication Theory 14, 328-347 (2004).

[64] Shim, K. J., Ahmad, M. A., Pathak, N. , Srivastava, J. "Inferring Player Rating from Performance Data in Massively Multiplayer Online Role-Playing Games (MMORPGs)," Proceedings of the IEEE Social Computing (SocialCom-09). Workshop on Social Intelligence in Applied Gaming. Vancouver, Canada, August 29-31, 2009.

[65] Shim, K. J., Damania, S., Delong, C., Srivastava, J., "Player and Team Performance in EverQuest II and Halo 3," Potentials, IEEE , vol.30, no.5, pp.21-26, Sept.-Oct. 2011.

[66] Shim, K. J., Hsu, K. W., Srivastava, J. "Effects of Mentoring on Player Performance in Massively Multiplayer Online Role-Playing Games (MMORPGs)," The 2011 International Conference on Advances in Social Networks Analysis and Mining, Kaohsiung, Taiwan, July 25-27, 2011.

[67] Shim, K. J., Hsu, K. W., Srivastava, J. "Modeling Player Performance in Massively Multiplayer Online Role-Playing Games: The Effects of Diversity in Mentoring Network," The 2011 International Conference on Advances in Social Networks Analysis and Mining, Kaohsiung, Taiwan, July 25-27, 2011.

[68] Shim, K. J., Hsu, K. W., Srivastava, J. "An Exploratory Study of Player Performance, Motivation, and Enjoyment in Massively Multiplayer Online Role-Playing Games," Proceedings of the IEEE Social Computing (SocialCom-11). Boston, Massachusetts, USA, October 9-11, 2011.

[69] Shim, K. J., Sharan, R. , Srivastava, J. "Player Performance Prediction in Massively Multiplayer Online Role-Playing Games (MMORPGs)," Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD-10). Hyderabad, India, June 21-24, 2010.

[70] Shim, K. J., Srivastava, J. "Behavioral Profiles of Character Types in EverQuest II," Proceedings of the IEEE Conference on Computational Intelligence and Games (CIG-10). Copenhagen, Denmark, August 18-21, 2010.

[71] Shim, K. J., Srivastava, J. "Team Performance Prediction in Massively Multiplayer Online Role-Playing Games (MMORPGs)," Proceedings of the IEEE Social Computing (SocialCom-10). Minneapolis, Minnesota, USA, August 20-22, 2010.

[72] Shim, K. J., Srivastava, J. "Sequence Alignment Based Analysis of Player Behavior in Massively Multiplayer Online Role-Playing Games (MMORPGs)," Proceedings of the IEEE International Conference on Data Mining (ICDM-10). Workshop on Domain-Driven Data Mining.

[73] Silver, N. (2003). "Introducing PECOTA." Baseball Prospectus, 2003: 507-514.

[74] Smith, T. F. and Waterman, M. S. (1981). "Identification of Common Molecular Subsequences". Journal of Molecular Biology 147: 195197.

[75] Sweetser, P., Wyeth, P. 2005. GameFlow: A model for evaluating player enjoyment in games. ACM Computers in Entertainment 3, 3.

[76] Tango, T. (2004). "Marcel The Monkey Forecasting System." Tangotiger.net, March 10, 2004. URL http://www.tangotiger.net/archives/stud0346.shtml.

[77] Thomopoulos, N. T. (1970). Mixed model line balancing with smoothed station assignments. Mgmt. Sci. 16, 593-603.

[78] Thurau, C., Bauckhage, C. and Sagerer, G. Combining self organizing maps and multilayer perceptrons to learn bot-behaviour for a commercial game," in GAME-ON, 2003, pp. 119-123.

[79] Tjosvold, D. (1991). Team organization: An enduring competitive advantage. New York: Wiley.

[80] Vorderer, P., Hartmann, T., Klimmt, C.: Explaining the enjoyment of playing video games: The role of competition. In: Marinelli, D. (ed.) ICEC conference proceedings 2003: Essays on the future of interactive entertainment, pp. 107-120. Carnegie Mellon Press, Pittsburgh (2006)

[81] Vorderer, P., Klimmt, C., Ritterfeld, U.: Enjoyment: At the heart of media entertainment. Communication Theory 14, 388-408 (2004).

[82] Wan, C. S., Chiou, W. B. Psychological motives and online games addiction: a test of flow theory and humanistic needs theory for Taiwanese adolescents. Cyberpsychol Behav. 2006 Jun;9(3):317-24.

[83] Wanberg, C. R., Welsh, E. T., & Hezlett, S. A. (2003). Mentoring research: A review and dynamic process model. Research in Personnel and Human Resources Management, 22, 39-124.

[84] Wang, J., Xu, M., Wang, H., Zhang, J. "Classification of Imbalanced Data by Using the SMOTE Algorithm and Locally Linear Embedding," In Proceedings of the 8th international Conference on Signal Processing (2006).

[85] Williams, D. (2005). Bridging the methodological divide in game research. Simulation & Gaming, 36(4), 447-463.

[86] Williams, D. (2006). Groups and goblins: The social and civic impact an online game. Journal of Broadcasting and Electronic Media, 50(4), 651-670.

[87] Williams, D., Caplan, S., & Xiong, L. (2007). Can you hear me now? The impact of voice in on online gaming community. Human Communication Research, 33(4), 427-449.

[88] Williams, D., Ducheneaut, N., Xiong, L., Zhang, Y., Yee, N., & Nickell, E. (2006). From tree house to barracks: The social life of guilds in World of Warcraft. Games & Culture, 1(4), 338-361.

[89] Williams, D., N. Yee & S. Caplan (2008). Who Plays, How Much, and Why? A Behavioral Player Census of Virtual World. Journal of Computer Mediated Communication.

[90] Yannakakis , G. N., "How to Model and Augment Player Satisfaction: A Review," in Proceedings of the 1st Workshop on Child, Computer and Interaction, ICMI'08, Chania, Crete, October, 2008. ACM Press (invited paper).

[91] Yee, N. MMORPG Hours vs. TV Hours. http://www.nickyee.com/daedalus/archives/000891.php.

[92] Yee, N. (2007). Motivations of play in online games. CyberPsychology & Behavior, 9, 772-775.

[93] EVE Online, the Social Learning Curve, and Shooting People in the Face with Love. http://mmoreporter.com/2011/06/07/eve-online-the-social-learning-curve-and-shooting-people-in-the-face-with-love/

[94] MMORPG Information(s). http://mmorpginformations.blogspot.com/2010/12/game-review-everquest-ii.html

[95] Video Game Genres. http://en.wikipedia.org/wiki/Video_game_genres

[96] MySQL. http://www.mysql.com/

[97] JFreeChart. http://www.jfree.org/jfreechart/

[98] JasperReports. http://jasperforge.org/projects/jasperreports

[99] Hibernate. http://www.hibernate.org/

[100] Apache Hadoop. http://hadoop.apache.org/

[101] Apache Hive. http://hive.apache.org/

[102] Apache Mahout. http://mahout.apache.org/

[103] "U.S. Army's First Person Shooter Video Game Number One Recruiting ROI". http://socialmediarecruitment.com/blog/2010/03/03/u-s-armys-first-person-shooter-video-game-number-one-recruiting-roi/

[104] "U.S. military turns to the gaming world for training technology". http://www.makibie.com/blog/technology/us-military-turns-to-the-gaming-world-for-training-technology/

[105] ”US Troops Training on Video Games.” http://www.npr.org/templates/story/story.php?storyId=17308466

[106] ”’Call of Duty: Black Ops’ Is Best-Selling Video Game Ever in U.S.”. http://www.hollywoodreporter.com/news/call-duty-black-ops-is-166625

[107] Halo: Reach. http://gamrreview.vgchartz.com/sales/35024/halo-reach/

[108] http://www.eq2-guides.com/archives/98

[109] http://pc.gamespy.com/pc/everquest-ii/564411p1.html

[110] http://www.crystalkeep.com/eq2/EQ2Recipes.pdf

Appendix A

Glossary

Table A.1: Glossary of Terms

Term	Description	Comments
Account	Refers to user account created by a human user. When purchasing or signing up for a particular game, a human user must create an online account via game website.	Account has attributes such as first name, last name, street address, state, zip code, country, gender, birthdate, etc.
Player	Refers to human user who has created an online account and is eligible for creating one or more game characters.	
Character	A player can create one or more characters. Player plays a character in the virtual game space. It is also referred to as avatar in a broader game context.	

Level	Character starts off at Level 1. As he obtains more points, he reaches the next milestone of moving up to the next level. One of the goals of many of these games is to level up so that players can experience more game content and challenges.	
Group	Refers to a set of characters that play together in a given session. One or more characters can team up and complete a task or more together.	This is different from guild in that this is a transient instance of a set of characters playing together, meaning that this persists only for some duration (i.e. session).
Task	Refers to any significant activity in the game which leads to point gain for character.	
Monster	Refers to one type of task wherein character kills an in-game character (non-person-character or NPC) and attains points.	
Quest	Refers to one type of task wherein character is given one or more missions (i.e. go to a particular location on a game map, complete a monster kill, etc.). Upon completing each task, character is given points.	
Experience Point	XP for short. When a character completes a task, he attains points. Points add up and when it reaches the next milestone, character moves up to the next level.	

Guild	Refers to a persistent group of characters that form a social group. It is led by one or more players and often they take on combat tasks such as monster kills.	
Server	There are multiple game servers in multiple geographical locations. When player signs onto the game through the game client or console, he must choose a server to join.	

Appendix B

Player Modeling in Massively Multiplayer Online Games

Player modeling is a key aspect of game design. This chapter presents the results of an explorative study on player modeling in EverQuest II, a popular massively multiplayer online role-playing game (MMORPG) developed by Sony Online Entertainment. This study focuses on two aspects of MMOG game play; achievement and socialization. We develop a system for analyzing dissimilar patterns of game playing behavior in EverQuest II using dimensionality reduction and unsupervised learning. We conduct experiments on data collected from the game, and the results indicate that, across different types of players, different behaviors correspond to different stages of the level progression.

B.1 Introduction

Player modeling is a key aspect of game design. This chapter presents the results of an explorative study on player modeling in EverQuest II, a popular massively multiplayer online role-playing game (MMORPG) developed by Sony Online Entertainment. This study focuses on two aspects of MMOG game play; achievement and socialization. We develop a framework for analyzing dissimilar patterns of game playing behavior in EverQuest II using dimensionality reduction and unsupervised learning.

There is a growing interest in the gaming community in understanding player behaviors both inside and outside the gaming space. Game companies are interested in finding out how their games are played, if they are being played as intended, how the different game mechanics are being played out and how the different game playing patterns lead to a high level of satisfaction and entertainment for customers. Retrospective analyses after the game launch on existing game features can reveal information on which features enhance player's gaming experience and to which demographic segments they especially appeal to. Features negatively correlated with gaming experience can be considered for removal while those positively correlated with gaming experience can be further enhanced. For new game features, prospective analyses before the game launch can reveal information about which features might appeal to certain player population segments with a certain level of confidence and user-oriented testing can focus on these features for further validation.

This study proposes a framework for studying dissimilar game play behaviors in EverQuest II. The framework consists of PCA-based K-means clustering algorithm and a system equipped with a user interface, which can be used by game professionals to understand whether various game mechanics are being played as intended and if not, how they are being played by whom. Our experimental results show that different stages of the player level progression exhibit different behavior patterns across multiple clusters of game players.

B.2 Player Modeling

One key aspect of game design and testing is player modeling. Each game player experiences the game world differently and leverages game mechanics to fulfill their own objectives; whether it be achievement, socialization, or immersion [92]. Analyzing players' game behavior data can yield information useful for many purposes.

First, analysis of players' behavioral data from game logs can reveal information useful for game designers understand whether certain game mechanics are being played as intended. Second, game designers can leverage this information to modify existing game features or design new features that appeal to certain player population segments. Additionally, more recently, adaptive games leverage player behavioral data in order to adapt to the actions of game players for the purposes of creating a more enjoyable gaming experience [10].

Game players can also benefit from this large scale analysis of game player behaviors. For instance, while many games today provide in-game ”how to get started” guides to help beginners get familiar with the game, there exists a high demand for a more automated and prospective way of recommending in-game tasks to maximize player performance, satisfaction and enjoyment. Findings from this study provide a foundation for a customized task recommendation system during game play where its primary objective is to automatically identify play patterns and suggest in-game activities. This study uses operational and process-oriented performance data of game players in EverQuest II.

B.3 Contribution

This study investigates dissimilar patterns of playing behavior in EverQuest II. We take in-game activities of game players and then group them into different playing behaviors. The first step is to apply dimensionality reduction to the various in-game activity variables. The next step is to perform clustering on game players. This study uses Principal Component Analysis (PCA) for dimensionality reduction and K-means for clustering [48]. Previous studies report usage of unsupervised learning techniques for player behavior modeling [17, 78].

B.4 Dataset

In this study, a variety of in-game behaviors were aggregated at each player level.

- Session time (which excludes any idle time exceeding 30 minutes); total and also aggregated by task types (monster kills, questing, deaths, combat bonus, mentoring, apprenticing, recipes).
- Number of tasks and amount of earned points; total and also aggregated by task types.

- Number of monster kills and amount of earned points while mentoring or apprenticing.
- Number of tasks and amount of earned points, while soloing versus grouping.
- Average group size, total and also aggregated by task types.
- Average group level, total and also aggregated by task types.
- Average performance; 1) Efficiency Index [64], 2) Success Index [64].
- Monster kill challenge level; i.e. average monster level.
- Number of mentoring instances in monster kills.
- Number of apprenticing instances in monster kills.
- Average level difference between mentor and apprentice.
- Average character affinity in mentoring.
- Average character affinity in apprenticing.
- Average class affinity in mentoring.
- Average class affinity in apprenticing.

Each game player is represented as a vector of 58 behavioral attributes as outlined above. In the next section, we describe our PCA-based K-means clustering approach to finding clusters of dissimilar game play behaviors.

B.5 Dimensionality Reduction and Clustering

The first step is to apply dimensionality reduction to the various in-game activity variables. The objective of this step is to remove any redundant information which might be present in more than one feature, which can occur because two or more variables can have a high correlation. Below, we elaborate our approach which combines the results from the PCA analysis and K-means clustering algorithm [48] to find dissimilar behavior patterns among game players with respect to various in-game behaviors discussed in Section III. We use the Weka [29] package for this experiment. We use the package's X-means algorithm [55] to find k, the number of clusters.

Algorithm 14 PCA-based K-means clustering to find clusters of dissimilar game play behaviors

Require: R $\longleftarrow$ N by M matrix, where N $\longleftarrow$ number of players and M $\longleftarrow$ number of in-game behavioral attributes.
{PCA function returns an M by V matrix, where V $\longleftarrow$ number of eigenvectors}
E $\longleftarrow$ PCA(R)
U $\longleftarrow$ eigenvector of first principal component, a 1 by M matrix
{Execute XMeans algorithm, X $\longleftarrow$ maximum number of clusters, Y $\longleftarrow$ minimum number of clusters, using the Euclidean distance function formulated as below. Given two points in Euclidean m-space, $\underset{\sim}{p}$ and $\underset{\sim}{q}$, where $\underset{\sim}{p} = (p_1, p_2, ..., p_m)$ and $\underset{\sim}{q} = (q_1, q_2, ..., q_m)$, the distance between the two points is given by:}
$d(p, q) = d(q, p) = \sqrt{U_1 \cdot (q_1 - p_1)^2 + U_2 \cdot (q_2 - p_2)^2 + ... + U_m \cdot (q_m - p_m)^2}$

B.6 Experiments and Results

A dataset is prepared for each player level. The reasoning behind this experimental design is that the game's point-scaling factor is not consistent across all player levels. Hence, we start with low granularity player-level specific patterns and reach generalizations across player level segments. We perform Algorithm 14 on each player level-specific dataset.

B.6.1 Number of Clusters

Our results show that game play behaviors partition into two to four clusters across all the player levels. Below, we summarize our findings in terms of dissimilar game play behaviors between clusters. Amongst the 58 in-game variables, the most distinguishing were found to be player performance in terms of efficiency, grouping behavior, challenge level, mentoring and apprenticing behavior. It must be noted that while the aforementioned in-game variables are most significantly distinguishing, there exists, within each cluster, some level of variation attributed to other less significantly distinguishing variables such as task-specific aggregated statistics (i.e. in some cases, a high number of recipes performed is seen to be negatively correlated with efficiency). Below, we describe the top five most significantly distinguishing variables and how they manifest in different stages of the game's player level progression.

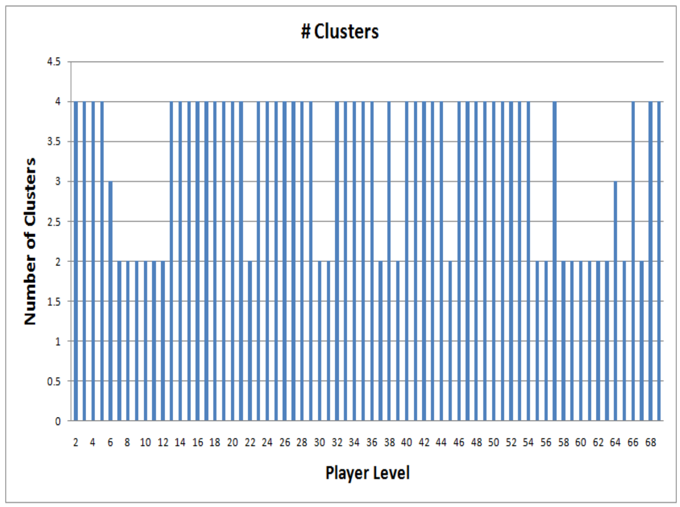


Figure B.1: 'Guk' server, number of clusters

B.6.2 Levels 2 to 6

Players at levels 2 through 6 are broadly clustered into two distinct groups and the following are the most distinguishing features:

- Efficiency: There are two groups of players, one with high efficiency and one with low efficiency.
- Grouping behavior: High performing players tend to solo most of the time. The amount of session time spent positively correlates with the number of tasks done as a group. It also positively correlates with the group size. The higher the group size, the higher the session time. The higher the group level, the higher the session time.

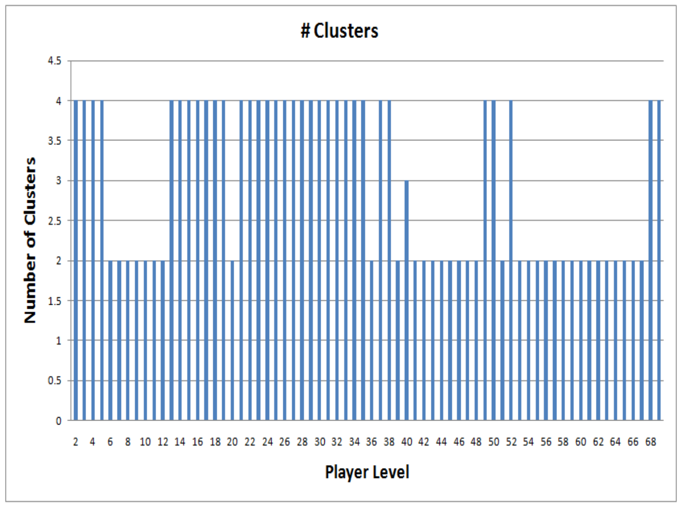


Figure B.2: 'Antonia Bayle' server, number of clusters

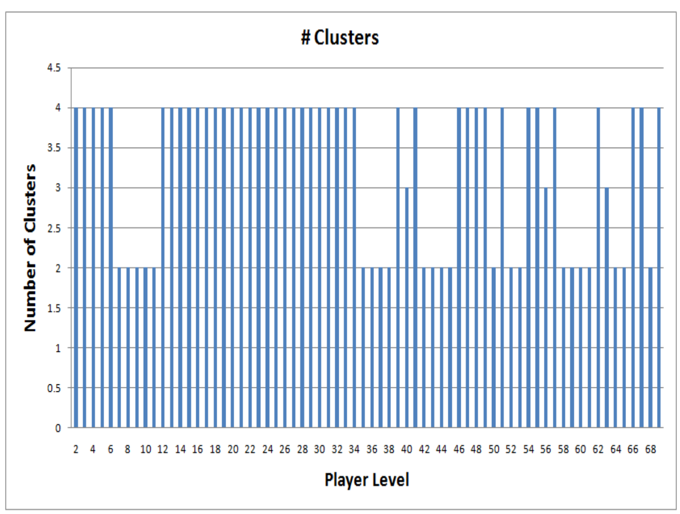


Figure B.3: 'Nagafen' server, number of clusters

B.6.3 Levels 7 to 12

- Efficiency: One group is more efficient as they spend comparatively less time than the other group to move up to the next level.

- Grouping behavior: The amount of session time spent negatively correlates with the number of tasks done as a group. It also negatively correlates with the group size. High performing players stay in groups of average level close to their own level. Low performing players tend to participate in groups whose average level is much lower than their own level. High performing players form groups of size 2 or above where as low performing players tend to play solo.

- Apprenticing: High performing players log a substantially high number of monster kills accompanied by a mentor. High performing players have had 90% of their monster kills accompanied by a mentor whereas low performing players went solo most of the times.

- Challenge level: High performing players average monster kills where the average monster level is slightly higher than that achieved by low performing players.

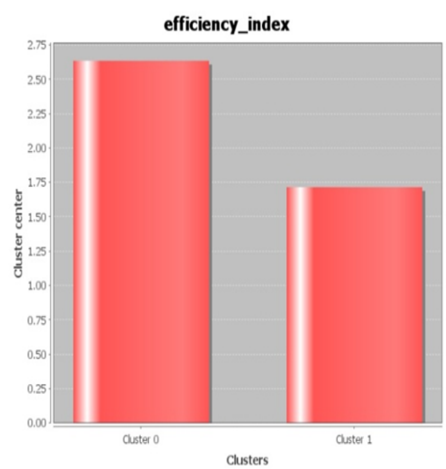


Figure B.4: Player Performance (efficiency), Level 12, 'Guk' server

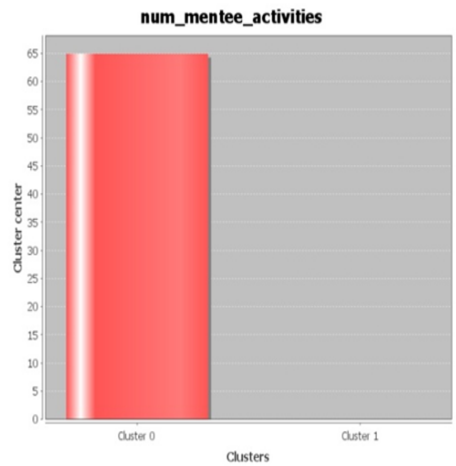


Figure B.5: Number of Apprenticing Activities, Level 12, 'Guk' server

Figures B.4 and B.5 show two clusters identified by Algorithm 14 for Level 12 players. It shows that a high number of apprenticing activities is positively correlated with efficiency.

Between player levels 13 and 20, behavioral patterns change from one level to the next. Below, we describe several representative cases from 'Guk' server.

B.6.4 Levels 13 to 20

- Efficiency: There are broadly two groups of players, one with high efficiency and one with medium efficiency.

- Grouping behavior: Average group size ranges between 1.25 and 2. High performing players tend to form groups of size 2. We start seeing a lot of group plays. High performing players tend to form groups with higher average group level, however, only to a certain extent.

- Mentoring: High performing players tend to log a higher number of mentoring activities.

- Apprenticing: In certain levels, high performing players tend to log a high number of apprenticing activities. In others, a high number of apprenticing activities is negatively correlated with efficiency.

B.6.5 Levels 21 to 39

- Efficiency: There are broadly two groups of players, one with high efficiency and one with medium efficiency.
- Grouping behavior: Average group size ranges between 1.75 and 2.25. High performing players tend to form groups of size 2 or above. There are a lot of group plays occurring in this level range. High performing players tend to form groups with higher average group level.
- Mentoring: High performing players tend to log a higher number of mentoring activities.
- Apprenticing: In certain levels, high performing players tend to log a high number of apprenticing activities. In others, a high number of apprenticing activities is negatively correlated with efficiency.

B.6.6 Levels 40 to 50

- Efficiency: There are three groups of players, one with high efficiency, one with medium efficiency, and one with low efficiency.
- Grouping behavior: Average group size ranges between 2 and 2.8. High performing players tend to form groups of size 3 or above. There are a lot of group plays in this level range. High performing players tend to form groups with higher average group level.
- Mentoring: Low performing players tend to log a substantially higher number of mentoring activities.
- Apprenticing: High performing players tend to log a high number of apprenticing activities.

B.6.7 Levels 51 to 54

- Efficiency: There are three groups of players, one with high efficiency, one with medium efficiency, and one with low efficiency.
- Grouping behavior: Average group size ranges between 2 and 3.15. High performing players tend to form groups of size 3 or above. There are a lot of group plays in level range. High performing players tend to form groups with higher average group level.
- Mentoring: Low performing players tend to log a substantially higher number of mentoring activities.
- Apprenticing: High performing players tend to log a high number of apprenticing activities.

B.6.8 Levels 55 and above

- Efficiency: There are three groups of players, one with high efficiency, one with medium efficiency, and one with low efficiency.
- Grouping behavior: Average group size ranges between 3.75 and 4.25. There is no distinct pattern across the clusters. There are a lot of group plays in high levels. High performing players tend to form groups with higher average group level.
- Mentoring: Low performing players tend to log a substantially higher number of mentoring activities.
- Apprenticing: High performing players tend to log a high number of apprenticing activities.

Figures B.6 and B.7 show four clusters identified by Algorithm 14 for Level 69 players. It shows that the higher the number of mentoring activities, the lower the efficiency.

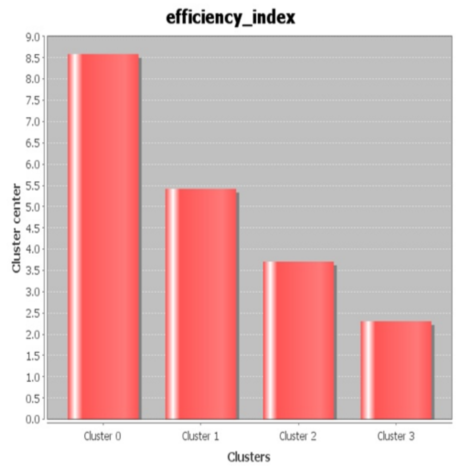


Figure B.6: Player Performance (efficiency), Level 69, 'Guk' server

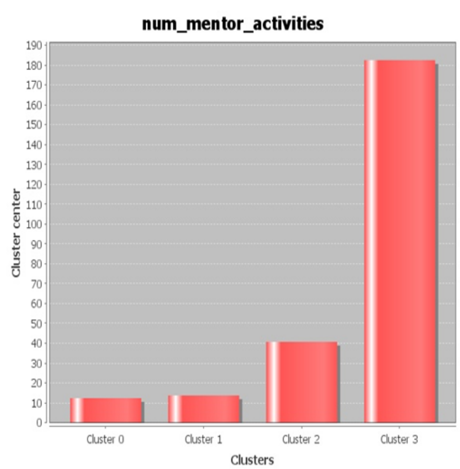


Figure B.7: Number of Mentoring Activities, Level 69, 'Guk' server

B.7 System

The proposed framework consists of the Java-written back-end program and the servlet-based user interface as shown in Figure B.8. The back-end program is scheduled to run on a regular basis to extract behavioral data from the game logs, aggregate it, and make it available for PCA analysis and clustering analysis. Image files are produced and stored on the server for the User Interface application to query against. The user interface allows for easy viewing of similar and dissimilar behavior patterns across different clusters of game players. One can choose to view only achievement-related variables or socialization-related variables.

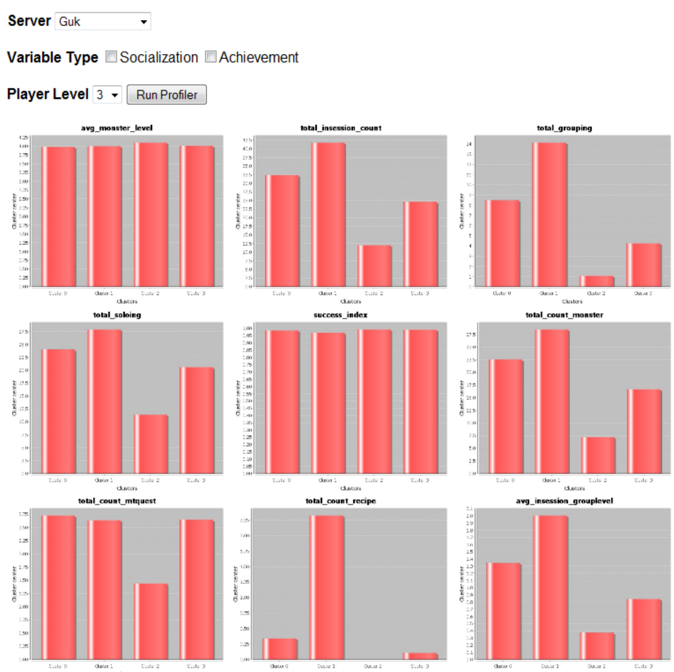


Figure B.8: Behavior Profiling User Interface

B.8 Conclusion

This study proposes a framework for studying dissimilar game play behaviors in EverQuest II. The framework consists of PCA-based k-means clustering algorithm and a system equipped with a user interface, which can be used by game professionals to understand whether various game mechanics are being played as intended. Our experimental results show that game play behaviors partition into two to four clusters across all the player levels. Amongst the 58 in-game variables, the most distinguishing were found to be player performance in terms of efficiency, grouping behavior, challenge level, mentoring and apprenticing behavior. We report that different stages of the player level progression exhibit different behavior patterns across multiple clusters of game players.

Appendix C

Behavioral Profiles of Character Types in EverQuest II

This chapter examines different types of playing behaviors by observing their play patterns and performance over time. Insight into different types of learning patterns can reveal information about numerous different ways a game can be played and potentially provide clues as to why certain players stop playing or drop out completely whereas other players continue with the game. It also examines behavior profiles of different character types in MMOGs in relation to player performance.

C.1 Background

This study examines behavioral profiles of different character types in EverQuest II, a popular massively multiplayer online game (MMOG) developed by Sony Online Entertainment. The study uses the game's player activity data to construct behavioral profiles of online game players for the applications of normal behavior recognition and anomaly detection. The behavioral profiles give insight into not only what players do in the game to level up but also how they perform on different types of task and how they work with other players. The behavioral profiles this study reports provide insight into online game player behavior, and it provides information useful for choosing a character type suitable for one's objectives in playing the game. In this study, we develop a framework for automatic behavior profiling. The proposed framework consists of the following key components; 1) segmentation analysis of historical player behaviors, 2) behavior profiling of input users (new players whose behaviors we want to label, and 3) recommendation of tasks based on input users' objectives.

This study uses operational and process-oriented performance data of game players in EverQuest II to analyze player behaviors in the game universe, segment players by their play style or play behaviors, and recommend the next course of action to a player based on his objective(s). Player objectives vary from advancing through the game as fast as possible to becoming an integral part of a guild, a community of players that participate in all sizes and difficulties of quest activities in the game. Analyzing the past behaviors of a large number of game players provide an insight into a variety of character types that exist in the game. The next section discusses in depth what are the typical character types or roles observed in many massively multiplayer online games. We incorporate the identified behavioral signatures into a performance management tool and a task recommendation system for use by EverQuest II players. The built tool and system are expected to be a valuable addition to the existing game system.

C.2 MMOGs and Character Types

In many massively multiplayer online games (MMOGs), selection of character type is very important as it will have a large influence on a player's game play experience. Exact names and specific abilities and capabilities for different character types may vary from one game to another. Take EverQuest II, for instance. As delineated in details in subsequent sections, there is a concept of archetype, class, and sub-class. Also, there is a concept of (character) race and (character) gender. Something similar is observed in other games such as World of Warcraft, for instance. Across many such games, there are several broad categories of characters and they are 1) tanks, 2) damage dealers, 3) casters, 4) crowd control, utility managers, and 5) healers.

Typically, it is the toughest and strongest classes in the game that play tanks. In EverQuest II, they are warrior, shadowknight, and paladin. They wear heavy armors and charge right into the monster (enemy). They run into the monster head on so that the monster's attention is on them. In a quest or group play, this allows other players perform their respective tasks. Tanks generally have a good level of health, strong armors, and combat abilities. Tanks may not impose direct attacks on the monster. Rather, they are designed to take on a lot of damage imposed by the monster while other combat players (i.e. supporters) impose direct damage on the monster. It is generally considered that skillful tanks do not let the monster's attention away from them and onto other players on the team.

Damage dealers specialize in imposing a lot of damage on the monster. In EverQuest II, they are beastlord, berserker, monk, ranger, and rogue. They are referred to as Damage Per Second or DPS characters, and they come with abilities to drop huge amounts of damage on the enemy in a short period of time. While it is the tanks that hold the attention of the monster, it is the DPS characters that act as supporters. They go behind or around the monster, surround it, and produce large amounts of damage. DPS characters come in two flavors, melee and ranged. The latter can impose damages from a distance. They usually do not have heavy, strong armors that can protect them in close combats. Also, their health levels are not as high as close-combat characters. DPS characters are favored as solo play characters as their abilities and capabilities are very effective in destroying monsters. However, because their armors may not necessarily be tough and strong as those of tanks, occurrence of combat death may be higher than tanks.

Healers specialize in keeping the healths of team players high in combat situations. In EverQuest II, they are cleric, druid, and shaman. In combats, they are in the background, on stand-by, keeping track of everyone's health level. They can exercise what is called "buffs" which helps enhance everyone's offense and defense abilities. While tanks go head-on against the monster, healers must be there to keep the tanks' health levels high. In almost all quests, healers are seen as they play an integral role. Healers do not have heavy, strong armors to protect themselves very well. They are not designed to fight but to heal others.

Yet another category of characters include casters. In EverQuest II, they are wizard, magician, and necromancer. Mostly, these characters are designed for group activities. They help in combat situations in a variety of ways. As is the case with healers, some caster characters can also exercise buffs to increase the damage amount and close-combat characters' health levels. Other caster characters take on damage imposed by the monster while tanks and other supporters are healing.

The last category of characters include crowd control and utility characters. In EverQuest II, they are enchanter and bard. These characters can control the monster, distract it away from the tanks, healers, supporters, etc. Often times, this is performed to allow enough time for combat characters to heal and to prevent the monster from damaging important characters such as healers during critical times.

In summary, there are broadly six categories of characters in MMOGs. Character type selection in the beginning of the game play is crucial as each character type comes with a different set of abilities and capabilities. However, in playing MMOGs, player experience is also shaped by the way a given player plays the character and also by the way he socializes with other players.

C.3 Archetypes, Classes, and Sub-classes in EverQuest II

Selection of character type (i.e. archetype, classes, sub-classes, race, etc.) is considered an important decision as it defines the basis of opportunities and choices of roles and tasks within the game [109]. In EverQuest II, there are four archetypes where each archetype consists of three classes each of which in turn consists of two sub-classes [56]. Figure **??** shows average performance of five sub-classes in the month of March, 2006. Performance at each player level is defined as a function of play time at each player level.

Fury sub-class is of priest archetypes. Fury characters specialize in healing, and their primary task as a member of a raid team is to heal other members in combats. Fury sub-class is favorite as a solo character, but it is also effective in team plays (i.e. monster raids, quests). On the other hand, berserker sub-class is of fighter or warrior archetype. It is considered a pure class of fighters, and berserker characters can make use of any weapon possible to fight monsters. They are considered well-rounded as solo players or team players. They possess and use heavy armors and can sustain injuries for a long time. In raid groups, berserker character often times play tanks, confronting vicious monsters up front whereas other character play as supporters and healers.

Figure C.1 shows that players of fury sub-class spend relatively less amount of play time in order to progress to the next level. This trend is consistent across all 70 player levels. There can be several reasons as to why berserker characters, on average, progress

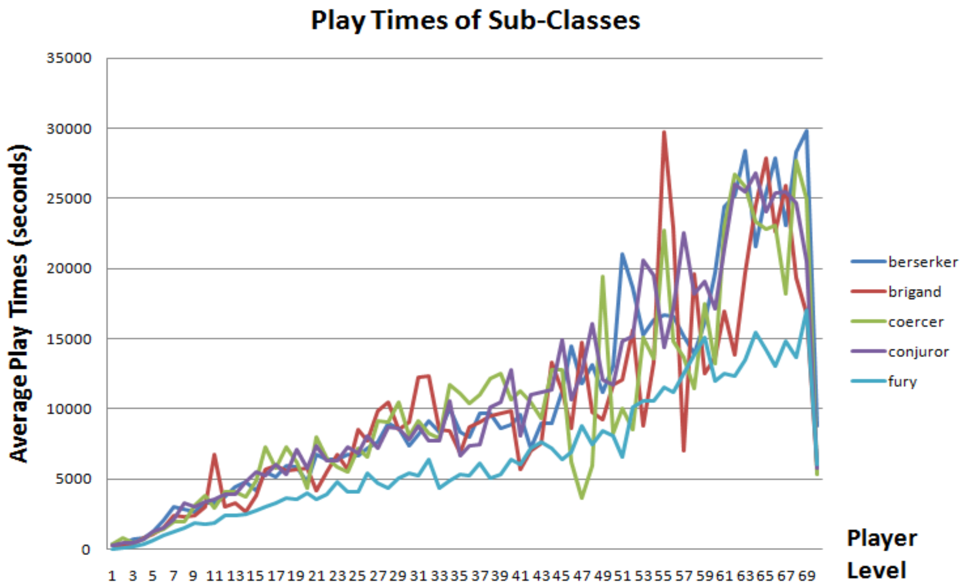


Figure C.1: Average Play Times of Five Sub-classes in EverQuest II

to the next level slower than fury characters. One possible explanation might be that berserker characters in general may not be performing activities that would amount to experience point gain. For instance, it is recommended that a player explores a zone that he plans on questing. Zoning does not lead to experience point gain. Yet another explanation might be that sub-classes that progress relatively slower may be performing tasks whose experience point gain is not substantial. For instance, mentoring system in EverQuest II allows a player to level down to mentor a lower level player. The experience point gain for the mentor can be substantially small, however, it allows the mentee to gain more experience points and the mentor to perform tasks that are no longer accessible to players at his current level. Multiple online resources are available today that show how to level up fast [108], and there can be numerous other explanations as to why certain sub-classes progress relatively slow. The rich dataset we have is expected to allow our analysis to reveal information at the level of granularity appropriate to answer these questions, and it is our future direction to explore these research questions.

C.4 Behavioral Profiles of Character Types

C.4.1 Dataset

The study uses over eight months worth of player activity data from January 1, 2006 to September 11, 2006. The dataset contains over 170 million player-to-task records where over 93 million of them are monster kills and quest related tasks. The dataset contains 62,881 distinct players across player levels 1 through 70. Since then, Sony Online Entertainment has added an additional ten levels to the game, making 80 the maximum level one can reach. In a more recent release, Sentinel's Fate, the game maker raised the level cap to 90. All of the characters and their activity data has been extracted from XP table in the EverQuest II database housed at National Center for Supercomputing Applications (NCSA) at the University of Illinois. The dataset contains at the minimum the following information about game players and their characters: character id, character sub-class, race, task, timestamp of task completion, group size (whether a given character grouped with one or more other characters in completing a task), average group level (if a given character played with one or more other characters, this value represents the average of player levels of all characters in that group), experience (XP) points, and location (location in which the task was completed).

C.4.2 Session Extraction

Our preliminary analysis shows that the total amount of time between a player logs into a game and logs out of the game does not reflect the actual amount of time that the player spent performing tasks or socializing. A player can log in and leave the game without explicitly logging out of the game, hence creating one or more chunks of what we refer to as "inactive" or "idle" time. In the present study and also in previous studies [64, 69], we programmatically weave one or more active sessions from the game's performance data. Any chunk of time that exceeds 30 minutes without any activity is considered an inactive or idle time, and it is excluded from the total amount of play time computed for each player.

C.4.3 Methods

In this study, we construct behavioral profiles of online game players for the applications of normal behavior recognition and anomaly detection. We develop a framework, which we call Performance Management Tool and Recommendation System, for automatic behavior profiling. The proposed framework consists of the following key components: 1) segmentation analysis of historical player behaviors, 2) behavior profiling of input users (new players whose behaviors we want to label, 3) recommendation of tasks based on input users' objectives

The behavioral patterns we learn from the historical game logs get updated monthly as the new data dump becomes available from the vendor. It accepts a user account number as an input and it searches for all of the characters that have been created under that account. The user can choose one or more characters to examine. The tool compares the given player's performance, as defined as player efficiency, player busyness, success ratio, and other measures, against the measures of 1) the global population and 2) the subset of the population belonging to the same archetype/class/sub-class as the given player. Further, the tool projects how fast the given player will advance to the next level, given his past behaviors. The system recommends future tasks for the player, based on the patterns learned from the historical data of the global/subset population.

C.5 Experiments and Results

As described in the Motivation section, many MMOGs have game mechanics by which characters assume certain roles within the game. Depending on the archetype, class, and/or sub-class of the character, the set of abilities and capabilities the character can exercise may differ from one type to another. In the following sections, we report our findings.

C.5.1 Player Efficiency Index

A previous study [64] defines player performance in EverQuest II as a function of XP point gain over play time (referred to as session time). We refer to this measure as player efficiency. Given two players, the one with a larger point gain, given the same amount of time as the other player, is considered more efficient. Figure C.2 shows boxplots of player efficiency (computed as total XP point gain divided by session time) across the 24 different sub-classes observed in our experimental dataset. The player efficiency value is an aggregate over all player levels (1 through 70). Table C.1 in Appendix A lists all of the 24 sub-classes. In Figure C.2, the left most boxplot represents Class 1 (assassin) and the right most boxplot represents Class 40 (wizard).

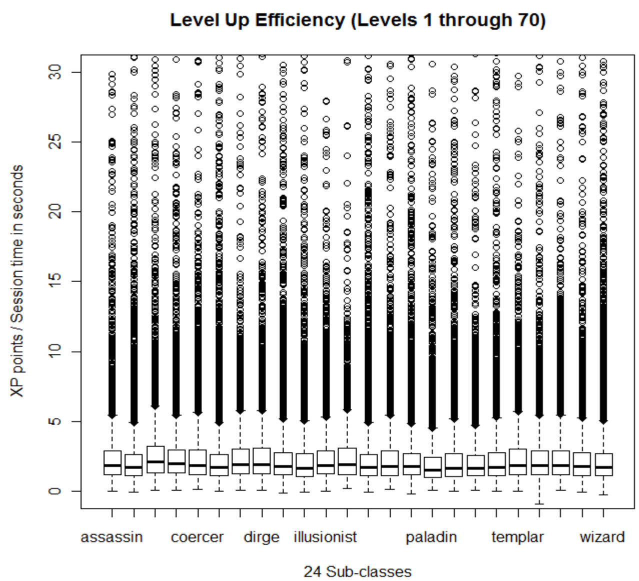


Figure C.2: Box Plots - Player Efficiency - Aggregated Over 70 Player Levels

Figure C.2 aggregates player efficiency across the 70 player levels, and hence, it does not provide a good visibility into distinct patterns in player behaviors. Hence, we further our analysis by examining player efficiency for each of the 24 sub-classes at each of the 70 player levels.

Figure C.3 plots all of the players and their player efficiency across the 70 player levels. As the player level increases, the number of outliers decreases.

In the fighter archetype, bruiser sub-class has the highest player efficiency where as paladin sub-class has the lowest player efficiency. The rest of the sub-classes exhibit similar level of player efficiency.

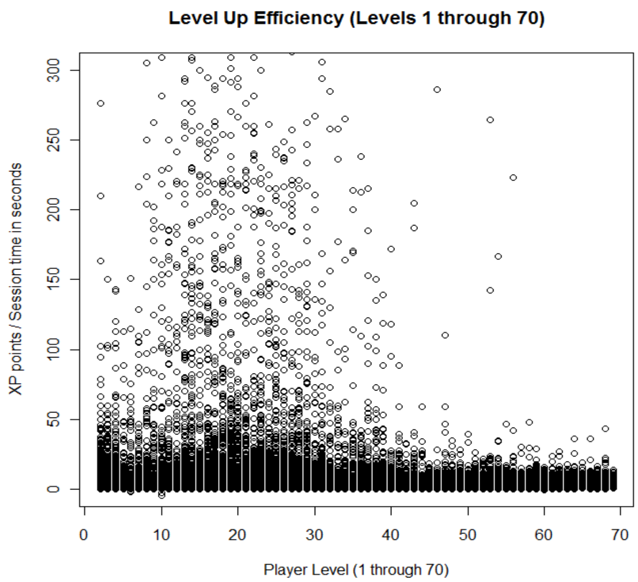


Figure C.3: Player Efficiency - Across 70 Player Levels

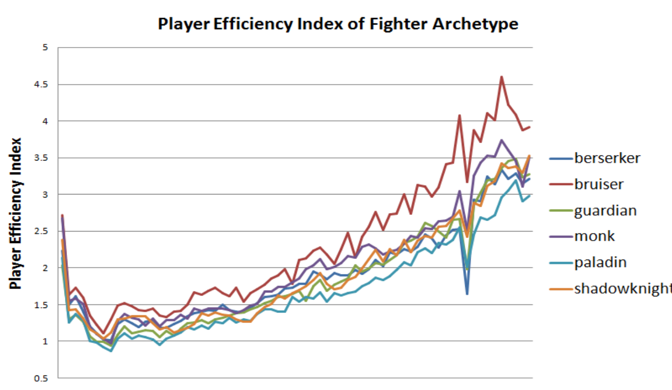


Figure C.4: Player Efficiency Index - Fighter Archetype

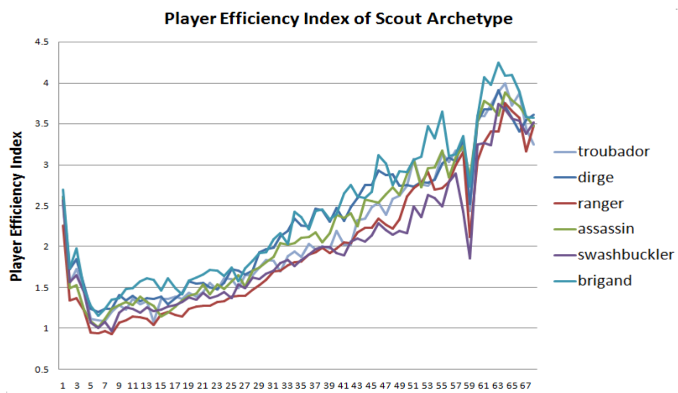


Figure C.5: Player Efficiency Index - Scout Archetype

In the scout archetype, brigand sub-class has the highest player efficiency where as ranger and swashbuckler sub-classes exhibit the lowest player efficiency. The rest are in the middle zone inbetween the highest and the lowest.

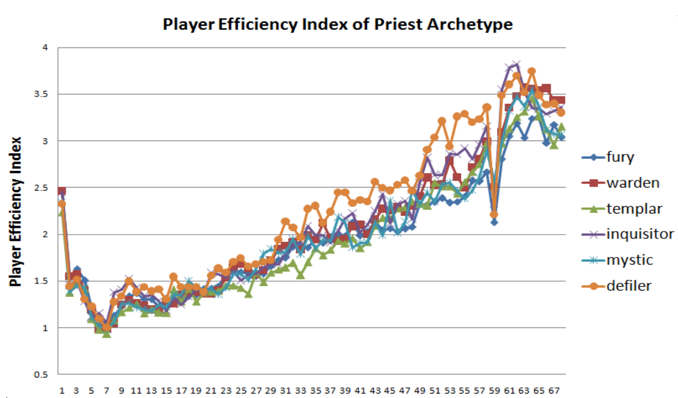


Figure C.6: Player Efficiency Index - Priest Archetype

In the priest archetype, defiler sub-class has the highest player efficiency where as templar sub-class has the lowest player efficiency up until level 43, and fury sub-class exhibit the lowest player efficiency from level 43 onwards.

In the mage archetype, coercer sub-class has the highest player efficiency whereas warlock and wizard exhibit the lowest player efficiency in higher levels, and conjuror

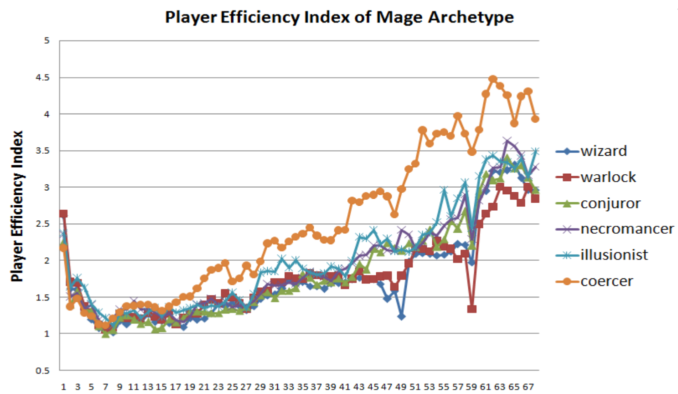


Figure C.7: Player Efficiency Index - Mage Archetype

sub-class has the lowest player efficiency up until level 42.

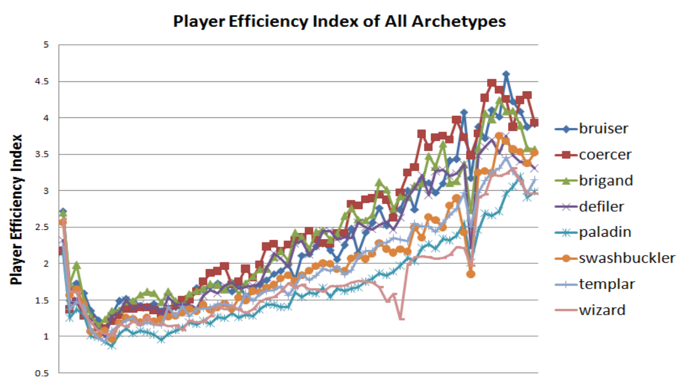


Figure C.8: Player Efficiency Index - All Archetypes

From each archetype, we take the highest and the lowest sub-classes and chart across all of the four archetypes. Coercer (mage), bruiser (fighter), brigand (priest), and defiler (scout) sub-classes exhibit comparatively higher player efficiency whereas wizard (mage) and paladin (fighter) sub-classes stay in the lower range.

In this subsection, we examined player efficiency across the 24 sub-classes in EverQuest II. Our analysis shows that there is an overall trend with respect to how fast particular sub-classes in the game advances throughout the game.

C.5.2 Player Busyness Index and Task Types

In this subsection, we examine Player Busyness. Player Busyness is a function of the total number of activities over play time (session time). Player Busyness Index is intended to show how actively a player is involved in game play.

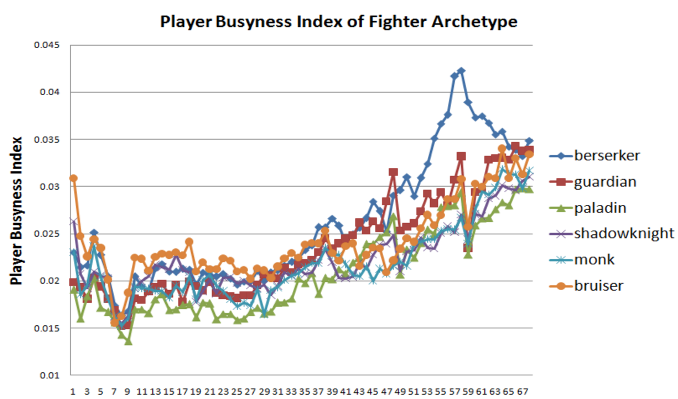


Figure C.9: Player Busyness Index - Fighter Archetype

In the fighter archetype, paladin sub-class exhibits the lowest player busyness throughout the game. In levels 1 through 40, bruiser sub-class has the highest player busyness. Beyond level 40, however, guardian starts ramping up and berserker sub-class quickly ramps up, showing the highest player busyness in higher levels.

In the scout archetype, ranger sub-class exhibits the lowest player busyness throughout the game. From level 35 onwards, all of the sub-classes exhibit a similar level of player busyness.

In the priest archetype, all sub-classes exhibit a similar level of player busyness in lower levels. However, from level 37 and onwards, mystic sub-class drops down a bit and fury sub-class ramps up quickly and becomes very busy. All other sub-classes continue to exhibit a similar level of player busyness in higher levels.

In the mage archetype, all sub-classes exhibit a similar level of player busyness in levels 1 through 11. From level 12 and onwards, they start to diverge. Warlock sub-class ramps up quickly and so does wizard sub-class.

From each archetype, we take the highest and the lowest sub-classes and chart across

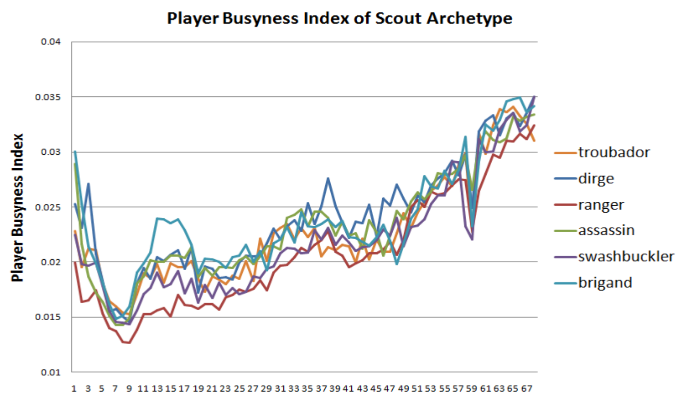


Figure C.10: Player Busyness Index - Scout Archetype

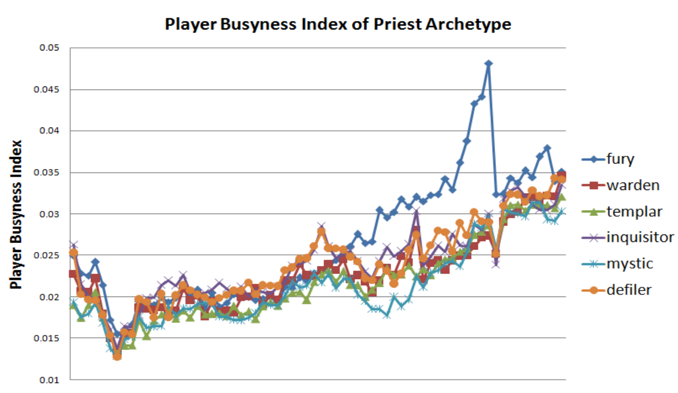


Figure C.11: Player Busyness Index - Priest Archetype

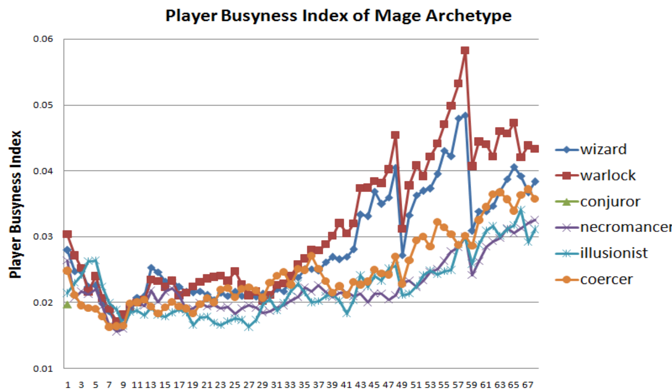


Figure C.12: Player Busyness Index - Mage Archetype

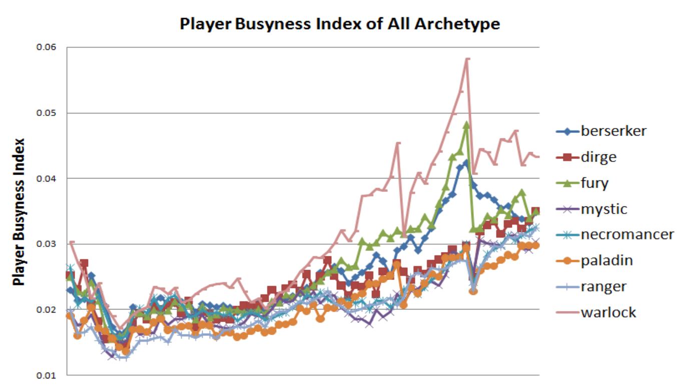


Figure C.13: Player Busyness Index - All Archetypes all of the four archetypes. Warlock (mage), fury (priest), and berserker (fighter) sub-classes exhibit comparatively higher player busyness whereas all the other sub-classes stay in the lower range.

Next, we examine types of task performed by the three busiest sub-classes (warlock, fury, and berserker) and the three least busy sub-classes (mystic, paladin, and necromancer). There are numerous other activities, many of them social oriented, such as trading, participating in player housing activities, guild-specific activities (which can be a source of XP point gain), etc. In this study, we examine 1) combat oriented activities (such as monster kills, deaths, questing), 2) recipe activities [110] (crafting, use of ingredients), and 3) mentoring activities.

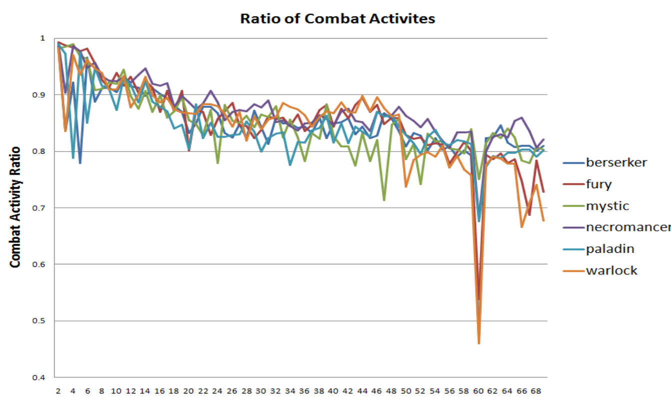


Figure C.14: Task Types - Ratio of Combat Activities

Figure C.14 shows that all of the six sub-classes show a high level of combat activities in lower levels. As the player level increases, the ratio of combat related activities decreases. Take mystic (priest), for instance. Especially from level 38 onwards, its combat activity ratio starts decreasing. However, Figure C.15 shows that mystic sub-class's ratio of recipe activities starts increasing. Mystic sub-class gradually spends more time performing recipe activities and less time performing combat activities.

Figure C.14 also shows that warlock (mage) and fury (priest) exhibit a significant drop in their participation in combat activities. However, Figure C.16 shows that these two sub-classes quickly ramp up on mentoring.

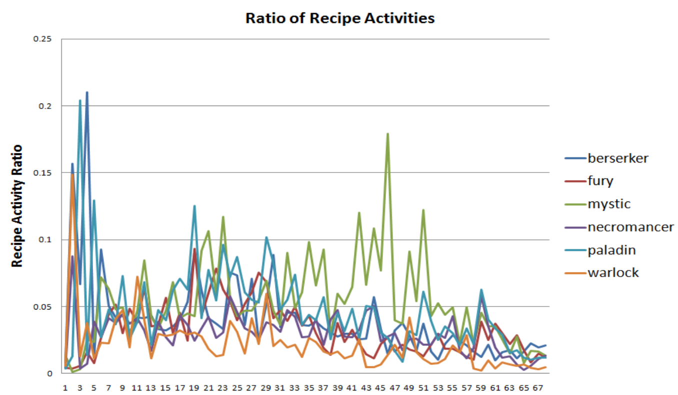


Figure C.15: Task Types - Ratio of Recipe Activities

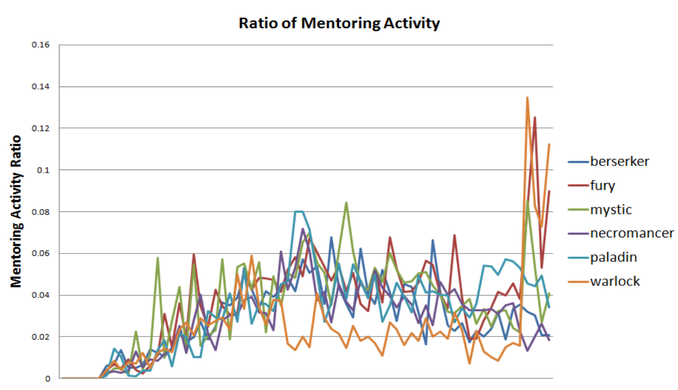


Figure C.16: Task Types - Ratio of Mentoring Activities

In this subsection, we examined player busyness across the 24 sub-classes in EverQuest II. Our analysis shows that there is an overall trend with respect to the changes in task types as players advance throughout the game. While certain sub-classes continue to maintain a relatively high level of combat activity ratio as they level up, others start spending more time on other types of activity such as mentoring.

C.5.3 Grouping versus Soloing

In this subsection, we examine ratios of group activities versus solo activities.

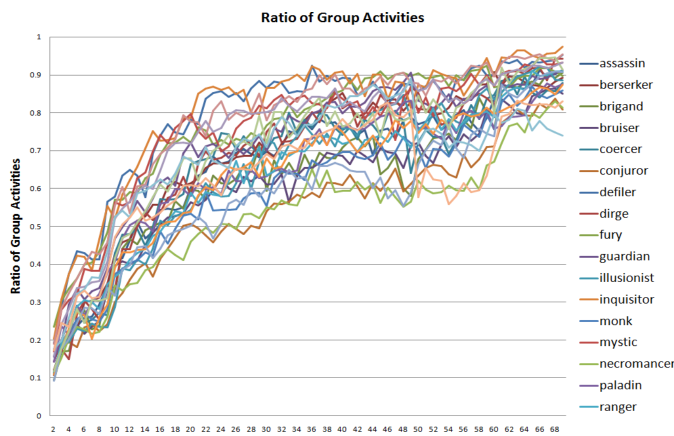


Figure C.17: Ratio of Grouping Activities

Figure C.17 shows the ratio of group activities across all of the 24 sub-classes from level 1 to 70. We observe that as the player level increases, the ratio of group activities increases. In higher levels, many combat activities such as quests are performed as a group as many monsters are too vicious for one single high level player to tackle. Furthermore, as discussed in the Background section, it is often the case that specific roles are required for killing vicious monsters. While tanks go up against the monster straight head-on, supporters are needed to attack from one or more sides, and healers are needed to help fighters/warriors recuperate.

Figure C.18 shows the average group size across all of the 24 sub-classes from level 1 to 70. We observe that as the player level increases, the average group size also increases. This finding is in alignment with the game mechanics that many combat tasks in higher

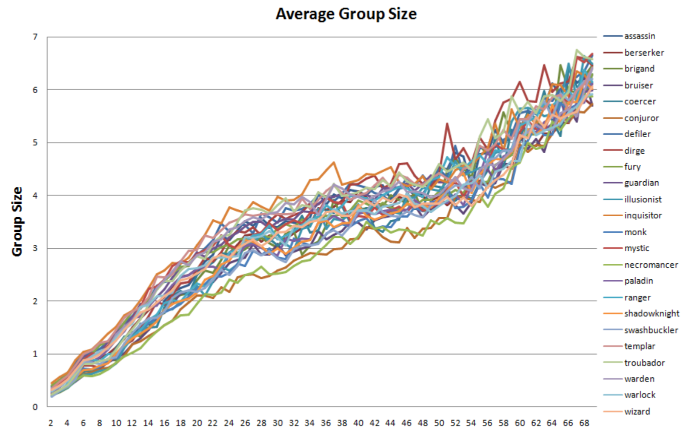


Figure C.18: Average Group Size

levels require grouping with other players.

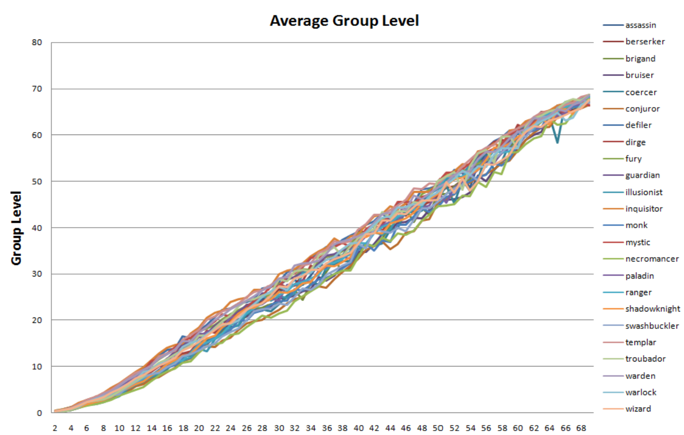


Figure C.19: Average Group Level

Figure C.19 shows the average group level across all of the 24 sub-classes from level 1 to 70. We observe a near linear relationship wherein as the player level increases, the average group level also increases. Our preliminary analysis shows that players typically attempt monsters whose levels are within zero to two levels away. When players attempt monsters whose levels are beyond their own levels, grouping activities are often observed wherein they are joined by one or more players whose levels are equal to or higher than the monster level. However, it is also observed that groups of level i players kill monsters whose levels are beyond i. Figure C.19 is in alignment with these reasonings.

C.5.4 Success Ratio

In this subsection, we examine success ratio of players across all of the 24 sub-classes from level 1 to level 70. A previous study [64] introduces quality as a performance measure. It is referred to as success ratio, and it is a measure of how successful a given player is at completing a task. It is formulated as (number of successful attempts) / (number of all attempts).

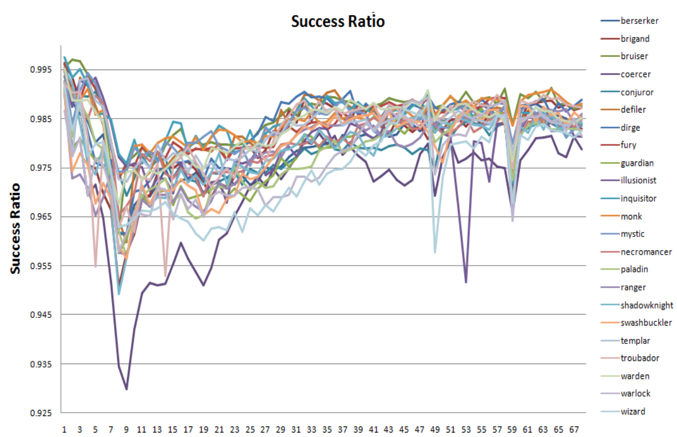


Figure C.20: Success Ratio

Figure C.20 shows the success ratio (in combat activities) across all of the 24 sub-classes from level 1 to 70. We observe in levels 1 through 9 that the success ratio drops by as large as 6.5%. However, beyond level 9, the success ratio gradually increases and stabilizes in higher levels. In higher levels, many combat activities such as quests are performed as a group as many monsters are too vicious for one single high level player to tackle. Furthermore, as discussed in the Background section, it is often the case that specific roles are required for killing vicious monsters. Solo playing is often observed in lower levels and sometimes in higher levels as well. Soloing is considered challenging in completing certain tasks due to the difficult nature of those tasks. However, per game mechanics, certain sub-classes are known to find soloing more challenging than others. Coercer is one of them. Figure C.20 shows that the success ratio of coercer (priest) is relatively lower than other sub-classes. This finding we report is in alignment with a general game knowledge that coercer sub-class faces the most challenges while soloing. Coercer sub-class is known to be most useful in group plays where their primary abilities are charm, damage over time, buffs, and crowd control. They are also known to have a limited selection of armor as they have access to only very light armor such as robes.

C.5.5 Performance Management Tool and Task Recommendation System

The Methods section describes the three key components in the built Performance Management Tool and Task Recommendation System. While the specifics of the game mechanics and design may vary from one game to another, the overall architecture of 1) behavioral profile construction from historical game logs and iteratively updating it with newer data, 2) automatic behavior profiling of input users, and 3) task recommendation for future use can be applied to any games wherein segmentation analysis based on demographics or play styles can be performed. Here, we showcase a couple of screenshots from our application.

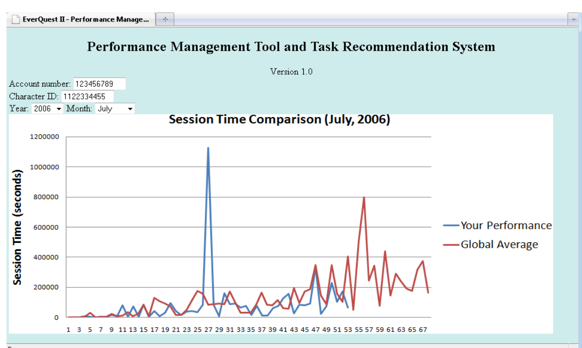


Figure C.21: Performance Management Tool and Task Recommendation System for EverQuest II

C.6 Conclusion

The present study examines behavioral profiles of game players in EverQuest II. First, we examine player efficiency, defined as a function of total XP point gain over session time. Our analysis shows that there is an overall trend with respect to how fast particular sub-classes advance throughout the game. Second, we examine player busyness across the 24 sub-classes in the game. Our analysis shows that there is an overall trend with respect to the changes in task types as players advance throughout the game. While certain sub-classes continue to maintain a relatively high level of combat activity ratio as they level up, others start spending more time on other types of activity such as mentoring. Third, we examine ratios of group activities versus solo activities. We report that as the player level increases, the ratio of group activities increases. In higher levels, many combat activities such as quests are performed as a group as many monsters are too vicious for one single high level player to tackle. We also observe that as the player level increases, the average group size also increases. This finding is in alignment with the game mechanics that many combat tasks in higher levels require grouping with other players with similar or different abilities and capabilities with respect to game play. Additionally, as the player level increases, the average group level increases linearly to the player level. Lastly, we report that as player level increases, success ratio increases beyond a certain level and this trend is observed across all of the 24 sub-classes. We incorporate the identified behavioral signatures into a performance management tool and a task recommendation system for use by EverQuest II players. The built tool and system are expected to be a valuable addition to the existing game system. Future directions include performing the same set of analyses using demographic information such as in-game gender and in-game race. Further, we plan to incorporate results from social network analyses performed on housing trust network, guilds, combat teams, and trade network.

C.7 List of Classes in EverQuest II

Table C.1: EverQuest II - 40 classes observed in January 2006 - September 2006 dataset

Index	Name of Class
1	assassin
2	berserker
3	brigand
4	bruiser
5	coercer
6	conjuror
7	defiler
8	dirge
9	fury
10	guardian
11	illusionist
12	inquisitor
13	monk
14	mystic
15	necromancer
16	paladin
17	ranger
18	shadowknight
19	swashbuckler
20	templar
21	troubador
22	warden
23	warlock
24	wizard