Modeling Player Retention in Madden NFL 11

Ben G. Weber
UC Santa Cruz
Santa Cruz, CA
bweber@soe.ucsc.edu

Michael John
Electronic Arts, Inc.
Redwood City, CA
MJohn@ea.com

Michael Mateas
UC Santa Cruz
Santa Cruz, CA
michaelm@soe.ucsc.edu

Arnav Jhala
UC Santa Cruz
Santa Cruz, CA
jhala@soe.ucsc.edu

Abstract

Video games are increasingly producing huge datasets available for analysis resulting from players engaging in interactive environments. These datasets enable investigation of individual player behavior at a massive scale, which can lead to reduced production costs and improved player retention. We present an approach for modeling player retention in *Madden NFL 11*, a commercial football game. Our approach encodes gameplay patterns of specific players as feature vectors and models player retention as a regression problem. By building an accurate model of player retention, we are able to identify which gameplay elements are most influential in maintaining active players. The outcome of our tool is recommendations which will be used to influence the design of future titles in the Madden NFL series.

Introduction

One of the major recent trends in video games is the use of telemetry in order to record data and statistics about players. Due to the widespread popularity of games, there are now huge repositories of player data available for analysis. Telemetry has been shown to be useful in game development as well as post release, enabling developers to track bugs, detect cheaters, and balance an in-game economy (Zoeller 2010). By mining game telemetry, developers have been able to reduce production costs, by identifying unused features, and improve player retention, by identifying features correlated with active players. For these reasons, collecting and analyzing data is quickly becoming a critical aspect in the process of producing large-scale video games.

We explore the application of mining game telemetry to aid Electronic Arts in the game design process. The purpose of our tool is to assist designers in answering questions about the gameplay patterns of players. Specifically, the goal of our tool is to address the following questions: which gameplay elements are most strongly correlated with player retention, and what is the optimal win ratio for maximizing player retention? The intended impact of answering these questions is to improve player retention, which will increase potential for year-to-year purchases and secondary revenue sources, such as in-game purchases.

Our domain of analysis is gameplay data collected from Xbox 360 players in *Madden NFL 11*, a popular football game[1]. The task of the tool is to identify correlations between gameplay features and the number of games played. Our tool performs this task by building regression models of player retention and by analyzing how modifying the inputs to the model impacts the expected number of games played. It utilizes several machine learning algorithms to identify which features have the most significant effect on retention. The output of the tool is evaluated by an analyst who provides the developer with recommendations for future game designs.

The tool we developed was part of a pilot study at Electronic Arts investigating the application of AI to analytics-driven design. Our tool is deployed as a server-side system that performs off-line analysis of players, and the end users are analysts at Electronic Arts. The impact of our work is design recommendations that will be incorporated in future iterations of Madden NFL.

Game Telemetry

Game telemetry is the transmission of data from a game executable for recording and analysis (Zoeller 2010). Externally, developers are using data collected from game telemetry to provide additional services to players. For example, the *World of WarCraft Armory* provides a web interface with statistics about players and guilds. Internally, developers are using telemetry datasets for tasks including bug identification, cheat detection, asset selection, and game balancing. Hullett et al. (2011) show how telemetry data can be used to identify unused game assets and modes in *Project Gotham Racing 4*, leading to reduced production costs for future iterations of the series. Zoeller (2010) presents the application of telemetry to the development process in order to improve game development workflow.

There are two ways in which telemetry is used to transmit data to a server. In a *streamed data* approach, data is sent to the server when an in-game trigger occurs. For example, Madden NFL 11 sends a message to the server when a player displays the on-screen help dialog. On the server side, streamed data is usually logged directly to text files,

[1]Madden NFL 11 was developed by EA Tiburon and published by EA Sports. Trademarks belong to their respective owners.

which serve as telemetry logs. In a *session-based data* approach, summaries are sent to a server when a gameplay session has completed. In Madden NFL 11, a summary of play-by-play results is sent to the server at the completion of a game. Game telemetry can be generated by a variety of sources: the quality assurance group during development, beta testers prior to release, and players once the game has been released.

One of the main challenges in using game telemetry is dealing with the scale of the data. The datasets collected from players can be massive, reaching terabytes in size. Game telemetry datasets are large because they contain logs of individual player behavior. To analyze data of this scale, distributed approaches such as Map-Reduce-Merge (Yang et al. 2007) have been used to distribute and analyze datasets on a large cluster. In our approach, we analyze a subset of telemetry using a representative sampling of players.

Related Work

Our tool is at the intersection of two research areas: computer-assisted game design, and game telemetry. There is an extensive literature on game design, but the use of tools for supporting the design process has only recently received attention (Nelson and Mateas 2009). Nelson and Mateas claim that tools for supporting game design need to reason about game mechanics, which are fundamentally different from current domains for which there are design support tools. Smith et al. (2010) address this problem by representing game mechanics as a formal logic. Our work differs from this research direction, because we are focusing on mining information from a large volume of human-generated playtesting as opposed to automated analysis of games using machine playtesting.

One of the main uses of game telemetry is player modeling, which includes recognizing player goals, identifying distinct types of players, and modeling player engagement. Research on goal recognition in games varies widely, because different genres of games have unique player objectives. In the domain of real-time strategy games, a player may have the goal of executing a specific strategy. Weber and Mateas (2009) present an approach to goal recognition in this domain by modeling strategy recognition as a classification problem. They grouped strategies into distinct classes and trained several classifiers on thousands of replays from professional players. In a football game, the player has a goal of executing a specific play. Laviers et al. (2009) explore an approach to play recognition using support vector machines to identify plays based on football player positions and trajectories. Their system was applied to the task of calling play switches in response to identifying specific defensive plays. The system uses a telemetry dataset generated by simulating all permutations of offense and defense play combinations.

Recent work has explored the application of data mining techniques to identify different types of players. Drachen et al. (2009) applied emergent self-organizing maps to player behavior in *Tomb Raider: Underworld* and indentified four unique styles of gameplay. Thurau and Bauckhage (2010) investigated the evolution of social groups in *World of Warcraft* and identified archetypical player types. They applied convex-hull matrix factorization to reduce high dimensionality data to eight intuitively interpretable classes of players.

Yannakakis (2008) identifies approaches for modeling player satisfaction in games, and discusses three types of telemetry: play-game interaction data, physiological data, and qualitative data. Physiological data includes data such as heart rate and EMG, while qualitative data refers to less quantifiable metrics such as *challenge* and *flow*. Models of player satisfaction have been applied to optimizing player engagement by adapting the difficultly level of the game (Yannakakis and Hallam 2009). Our tool differs from this work, because currently only play-game interaction data can be collected from a shipped game.

Game telemetry has also been utilized in systems that learn by demonstration. *Darmok* is an online case-based planner that learns to play real-time strategy games by analyzing replays of expert demonstrations (Ontañón et al. 2010). The system builds a case library by extracting sets of actions from expert demonstrations, in the form of replays, into cases with annotated goals. During runtime, *Darmok* employs a delayed case adaptation mechanism to reuse the case library in new game situations.

Feature Selection

We represent the goal of identifying the most influential gameplay features as a feature selection problem. Regression can be used to perform feature selection by building a statistical model of the data, evaluating the characteristics of the model, and identifying the most pertinent features (Suard, Goutier, and Mercier 2010).

Consider a dataset containing telemetry recorded from n players. Each player's actions are encoded as a feature vector with dimensionality j of the form:

$$x =< x_1, x_2, x_3, \ldots, x_j >$$

where the features are normalized and each vector has a corresponding label, y. Our algorithm starts by building a regression model, f, which maps a set of input features to a predicted label:

$$y = f(x)$$

Next, the regression model is used to find the *unique effect*, g, of each feature, k:

$$g(k, v) = f(x)$$

given the restriction:

$$x_j = \begin{cases} v & : j = k \\ x_j & : otherwise \end{cases}$$

where g is a function that represents the result of holding all features fixed except for x_k, which is set to v. The function g enables the impact of a specific feature to be evaluated by setting k to a constant and evaluating the output of $g(k, v)$ over the range $v = [0, 1]$. In the special case of linear regression, the following property holds:

$$\frac{d}{dv} g(k, v) \propto \beta_k$$

where β_k is the regression coefficient of feature k. We measure the feature impact, h_k, of a feature as follows:

$$\mu_k = \int_0^1 g(k,v)dv$$

$$h_k = \int_0^1 |g(k,v) - \mu_k|dv$$

where μ_k is the mean value of the unique effect, computed by finding the average value over the range $[0,1]$, and h_k is the absolute difference between the unique effect and mean value over the range $[0,1]$. This measure of impact enables the identification of the most significant features for a regression model.

We use an ensemble approach to feature weighting by summing the impact of each feature across models from multiple regression algorithms. Given m regression models, the overall impact of a feature, h_k, is computed as follows:

$$h_k = \sum^m h_k^i$$

where h_k^i is the impact of a feature determined by regression model i. The resulting feature impacts are sorted in descending order and the features with the highest impacts are identified as the most significant features.

Application to Madden NFL 11

Madden NFL 11 is a commercial American football game released for the Xbox 360 and PlayStation 3 game platforms. While Madden NFL is already a successful franchise, the cost of development greatly increases each year. Due to financial constraints, the designers need to determine where to allocate resources to maximize revenue potential.

Previously, the developers relied on evaluation from a small set of players in order to determine where to focus design attention. Feedback from players was acquired using expensive, in-house player studies limited to less than one hundred players. While these studies provide the developers with qualitative feedback, they are subject to individual player bias.

The role of our tool is to assist developers in understanding the behavior of players in Madden NFL 11 at a much larger scale. The main task of the tool is to identify the common patterns between players that stop playing the game. On average, the active player base decreases by 10 percent each week after release. By identifying the gameplay aspects that have the most significant impact on player retention, the tool enables developers to incorporate feedback from the player base into future iterations of the game.

Building a tool for identifying the features with the highest impact consisted of the following steps: collect data, develop a representation for encoding player behavior, build regression models for predicting the number of games played, and analyze the output of the regression models.

Data Collection

The first phase of the project consisted of collecting gameplay data from the Madden NFL 11 player base. We collected data from the release date of the game on August 10$^{\text{th}}$, 2010 to November 1$^{\text{st}}$, 2010. While Madden NFL 11 telemetry provides both streamed data and session-based data, we focused our investigation of the session-based data. Each session corresponds to a single game played, and contains a summary of every play in the game. For each play, the summary provides information about the starting conditions of the play, the formations and playcalls executed by each team, a subset of the actions executed during the play, and the outcome of the play. Since our tool was applied to a game that was already released, we had no control over the telemetry data being collected by the system.

The dataset contains gameplay information collected from over one million players. For our analysis we used a random sampling of 25,000 players.

Player Representation

The next phase of the project was to develop a suitable representation for encoding knowledge about a player, given the player's complete gameplay history in Madden NFL 11. We chose a feature vector representation which is used to capture a player's mode preferences, control usage, performance, and playcalling style. In total, there are 46 features which describe a player in our representation. Our system creates a feature vector for each of the sampled players. The label assigned to each vector is the number of games played, which we use as a metric for measuring player retention.

Mode preference features describe a player's preferences for different game modes. Our representation includes two features for each mode, which capture the usage ratios of the different game modes as well as the player's win ratio for each mode. In Madden NFL 11 there are eight different game modes which include several singleplayer and multiplayer variants. For multiplayer modes, an additional features is included which specifies the ratio of opponent quits and disconnects.

Control usage features encode a player's competency of the controls in Madden NFL 11. It includes two types of features: features describing the player's usage of pre-snap commands, such as audibles for changing the current play, and intra-play commands, such as controlling the path of a specific player during a play.

Performance features describe the ability of the player to make successful plays and gain an advantage over opponents. Features in this category include turnovers (changes in possession), average yards gained, average yards allowed, ratio of possession, and ratios of down conversions. For turnovers, our representation contains two features for both interceptions and fumbles, which correspond to offense and defense roles.

Playcalling features are used to describe a player's playcalling preferences. In Madden NFL 11, players can either manually choose a play to execute or use *Gameflow* which automatically selects a play for the player based on the current game situation. Our representation includes a feature that records the ratio of manual versus *Gameflow* playcalling. There is also a feature that describes the player's ratio of running versus passing plays. In order to capture the variety of plays called by a player, the representation also includes features for play diversity. Play diversity is defined as the ra-

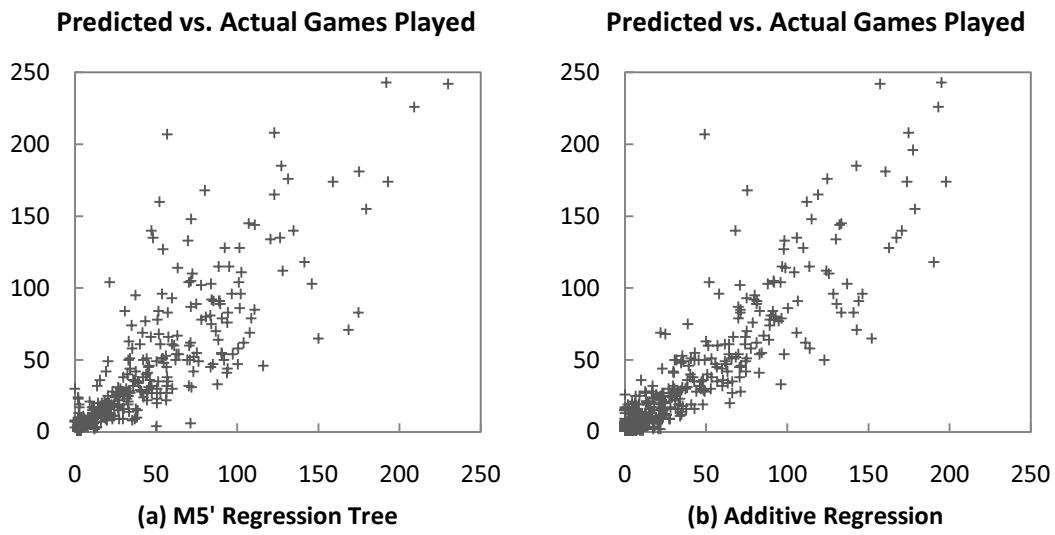

Figure 1: Predicted versus actual number of games played for the (a) M5' regression tree and (b) additive regression models.

tio of unique plays called to the total number of plays called, and there is a feature for *Offense Play Diversity* and *Defense Play Diversity*.

Regression Models

During the third phase of the project, we built regression models for predicting the number of games a player played, given the player's feature vector description. To assist in this process, we used the *WEKA* machine learning toolkit (Witten and Frank 2005), which provides several regression algorithms. Our tool utilizes the linear regression, M5' regression tree (Wang and Witten 1997), and additive regression (Friedman 2002) implementations provided by *WEKA*. We used the resulting regression models to evaluate the unique effect and impact of each feature.

Results

Our tool was applied to two experiments. In the first experiment, we used the tool to identify the most significant features given the complete feature set. We first evaluated the *marginal effect* of each feature, by measuring the correlation between the feature and the number of games played. While we identified *Offense Play Diversity* as the most significant feature, the correlation coefficient was less than 0.1.

During the first portion of the experiment, we investigated the accuracy of regression models for predicting the number of games played given our player representation. The accuracy of the regression models using 10-fold cross validation are shown in Table 1 and scatter plots of the M5' and additive regression predictions are shown in Figure 1. *ZeroR* is a baseline regression algorithm that always predicts the mean value of the distribution. While the M5' regression tree had the smallest mean error, the additive regression model had the smallest root-mean squared error (RMSE) and largest correlation coefficient. The best regression model is capable of predicting the number of games played with a correlation coefficient of 0.9.

Table 1: Accuracy of the regression models for predicting the number of games played by a player.

	ZeroR	Linear Regression	M5'	Additive Regression
Mean Error	37.1	20.9	**12.2**	12.6
RMSE	55.3	35.8	26.3	**24.4**
Correlation	0.02	0.77	0.88	**0.90**

In the second portion of the experiment, we measured the impact of each feature. The dataset was split into training and testing folds of even size. We used the training fold to build regression models and the testing fold to analyze the unique effect of each feature. To compute the unique effect of a feature, $g(k, v)$, all values in the testing dataset are held fixed, except for the value of the feature being evaluated, which is set to v. For each feature, we measured the output value of the regression model over the range $v = [0, 1]$. A visualization of $g(k, v)$ for the additive regression model is shown in Figure 2.

We then used the $g(k, v)$ functions generated by each regression model to compute the impact, h_k, of each feature. The features with the highest impacts are shown in Table 2. Overall, the *Offense Play Diversity* and *Defense Play Diversity* features had the most significant impact on the predicted number of games played. Features corresponding to making successful plays, including *Interceptions Caught* and *Sacks Made*, also had a large impact. Additionally, the *Online Franchise Win Ratio* and *Multiplayer Win Ratio* features were highly influential in maintaining player retention.

In the second experiment, our goal was to identify win ratios for maximizing player retention in each game mode. To achieve this task, we built regression models using a subset

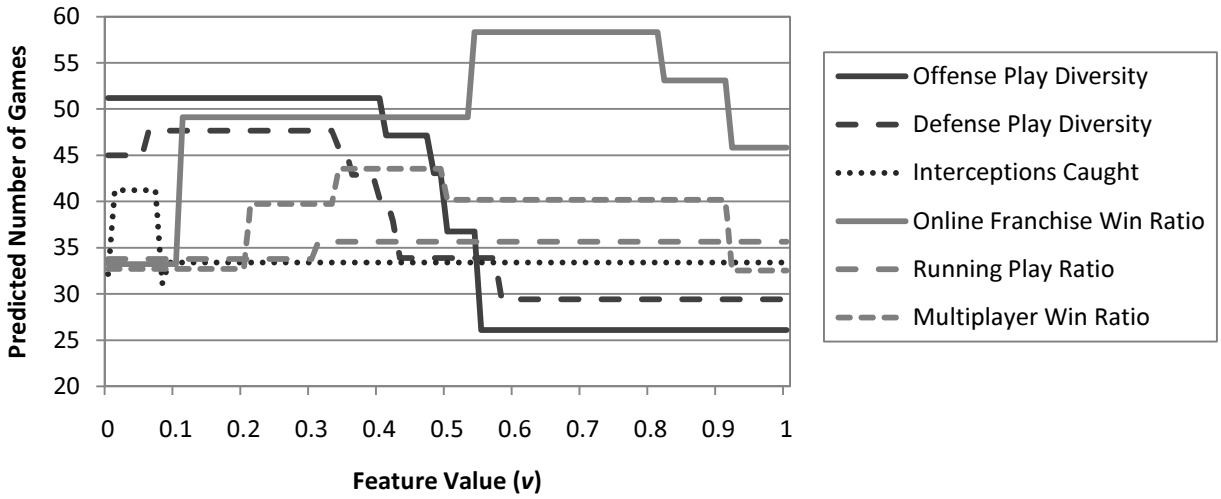

Figure 2: The predicted number of games played for different feature values based on the additive regression model. The lines show the value of $g(k, v)$ for the highest impact features, k, over the range $v = [0, 1]$.

Table 2: Features our tool identified as having the highest impact on player retention. The direction indicates whether the feature was positively or negatively correlated with the number of games played.

Feature	Correlation Direction	Impact (h_k)
Offense Play Diversity	$(-)$	55.4
Defense Play Diversity	$(-)$	34.2
Interceptions Caught	$(+)$	24.6
Online Franchise Win Ratio	$(+)$	15.7
Running Play Ratio	$(+)$	10.1
Multiplayer Win Ratio	$(+)$	9.3
Sacks Made	$(+)$	8.4
Defense Audibles	$(+)$	6.9
Peer Disconnects	$(-)$	6.3
In Possession	$(+)$	5.1

of the features and evaluated the impact of the win ratio feature for each game mode. The feature vector was limited to the features specifying the usage ratio and win ratio of each mode, resulting in 16 features. The complete dataset was used for the training dataset in the second experiment. An artificial dataset, consisting of a single instance, was used for the testing dataset. The artificial instance simulates a player that plays precisely one game mode.

For each game mode, we measured the impact of win ratios over the range $v = [0, 1]$. Results for the M5' regression model are shown in Figure 3. The results show that different game modes have different win ratios for maximizing player retention. For the *Singleplayer* and *Multiplayer* modes where the player competes head-to-head versus a computer or human opponent, 40%-60% is the optimal win ratio for maintaining player retention. For the *Superstar* mode, in which the player manages a player, and *Franchise* mode, in which the player manages a team, a win ratio of 60%-80% maximizes player retention

Conclusion

We presented a tool that uses game telemetry to assist in the process of game design. The tool enables developers to identify which features are most influential in maximizing player retention. To achieve this task, our approach builds regressions models of player retention, evaluates the characteristics of the models, and identifies the most significant features. We used our tool to identify the most influential features for maximizing player retention in Madden NFL 11.

Based on our analysis, we were able to provide the designers with several recommendations. First, playbooks should be simplified because players that use a large variety of plays generally play fewer games and are less successful in Madden NFL 11. Second, the controls should be clearly presented to players, because knowledge of the controls had a larger impact on retention than winning. Finally, provide players with the correct challenge, because the optimal win ratio is different for each game mode.

Our tool was deployed as an off-line analytics system as part of a pilot study at Electronic Arts. The tool demonstrates that AI techniques can be incorporated into analytics-driven game development. Rather than replacing current approaches for acquiring player feedback, the system is being used in conjunction with player studies to provide designers with both qualitative and quantitative feedback. One of the main impacts of our tool is that Electronic Arts is now investing additional resources in AI-driven telemetry analysis and exploring how to scale up our tool to larger datasets.

While we developed the tool specifically for analyzing player retention in Madden NFL 11, our technique for measuring feature impact is generalizable to other games. The data collection and feature analysis components in our workflow are domain independent and directly applicable to other

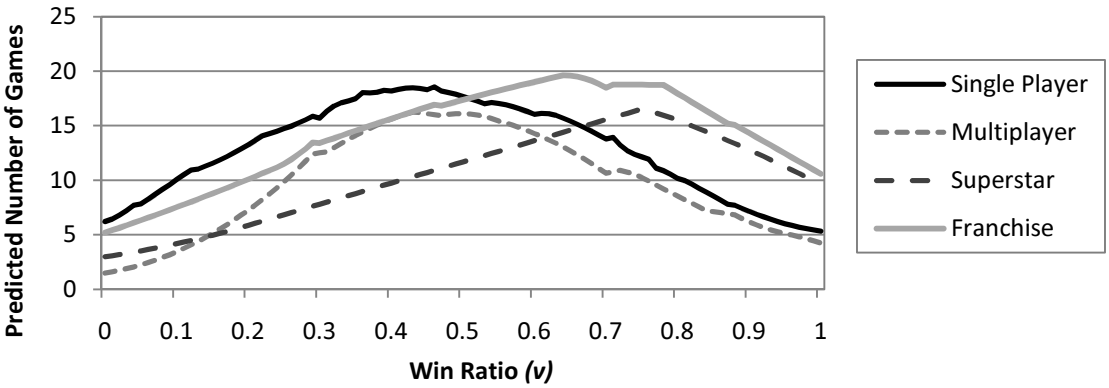

Figure 3: Predicted number of games played, $g(k, v)$, from the M5' model for a single game mode, k, with a win ratio of v.

titles at Electronic Arts. The game specific aspect of our tool is the feature selection and encoding process, which is unique to each game.

There are several directions for future work. One possible research direction is to investigate approaches for dealing with player behavior data at larger scales. We used a sample size of 25,000, but only captured the behavior from a small percentage of the player base. Another research direction is to identify additional features for representing player behavior. This could be accomplished by encoding more gameplay statistics or by recording additional features in the telemetry data. Finally, future work could investigate the use of additional regression algorithms for analysis.

Acknowledgments

This material is based upon work supported by Electronic Arts, Inc. and the National Science Foundation under Grant Number IIS-1018954. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

Drachen, A.; Canossa, A.; and Yannakakis, G. N. 2009. Player modeling using self-organization in Tomb Raider: Underworld. In *Proceedings of the IEEE Symposium on Computational Intelligence and Games*, 1–8. IEEE Press.

Friedman, J. 2002. Stochastic gradient boosting. *Computational Statistics and Data Analysis* 38(4):367–378.

Hullett, K.; Nagappan, N.; Schuh, E.; and Hopson, J. 2011. Data Analytics for Game Development. In *Proceedings of the ICSE Workshop on Games and Software Engineering*, To appear.

Laviers, K.; Sukthankar, G.; Molineaux, M.; and Aha, D. W. 2009. Improving offensive performance through opponent modeling. In *Procceedings of the Artificial Intelligence and Interactive Digital Entertainment Conference*, 58–63.

Nelson, M., and Mateas, M. 2009. A requirements analysis for videogame design support tools. In *Proceedings of the Conference on Foundations of Digital Games*, 137–144.

Ontañón, S.; Mishra, K.; Sugandh, N.; and Ram, A. 2010. On-Line Case-Based Planning. *Computational Intelligence* 26(1):84–119.

Smith, A. M.; Nelson, M. J.; and Mateas, M. 2010. LUDOCORE: A Logical Game Engine for Modeling Videogames. In *Proceedings of the IEEE Conference on Computational Intelligence and Games*, 91–98. IEEE Press.

Suard, F.; Goutier, S.; and Mercier, D. 2010. Extracting relevant features to explain electricity price variations. In *Proceedings of the Conference on the European Energy Market*, 1–6. IEEE Press.

Thurau, C., and Bauckhage, C. 2010. Analyzing the Evolution of Social Groups in World of Warcraft. In *Proceedings of the IEEE Conference on Computational Intelligence and Games*, 170–177. IEEE Press.

Wang, Y., and Witten, I. 1997. Inducing model trees for continuous classes. In *Proceedings of the European Conference on Machine Learning*, 128–137.

Weber, B., and Mateas, M. 2009. A Data Mining Approach to Strategy Prediction. In *Proceedings of the IEEE Symposium on Computational Intelligence and Games*, 140–147. IEEE Press.

Witten, I. H., and Frank, E. 2005. *Data Mining: Practical machine learning tools and techniques*. San Francisco, California: Morgan Kaufmann.

Yang, H.; Dasdan, A.; Hsiao, R.; and Parker, D. 2007. Map-reduce-merge: simplified relational data processing on large clusters. In *Proceedings of the SIGMOD Conference on Management of Data*, 1029–1040. ACM.

Yannakakis, G. N., and Hallam, J. 2009. Real-time game adaptation for optimizing player satisfaction. *IEEE Transactions on Computational Intelligence and AI in Games* 1(2):121–133.

Yannakakis, G. N. 2008. How to model and augment player satisfaction: A review. In *Proceedings of the ICMI Workshop on Child, Computer and Interaction*. ACM.

Zoeller, G. 2010. Game Development Telemetry. In *Proceedings of the Game Developers Conference*.