Bayesian Clustering of Player Styles for Multiplayer Games

Aline Normoyle
University of Pennsylvania
3451 Walnut Street
Philadelphia, PA 19104

Shane T. Jensen
The Wharton School
University of Pennsylvania
3730 Walnut Street
Philadelphia, PA 19104

Abstract

With game play data, empirical approaches to clustering are typically based solely on game outcomes, e.g. kills, deaths, and score for each player. In this paper, we investigate a method for clustering players based on how a player's *choices* relate to *outcomes*, or equivalently the latent player styles exhibited by players. Our approach is based on a Bayesian semi-parametric clustering method which has several advantages: the number of clusters do not need to be specified a priori; the technique can work with a very compact representation of each match (e.g. consisting primarily of indicator variables for player choices); a player can belong to multiple clusters and hence can have a hybrid style; and the resulting clusterings often have a straight-forward interpretation. To demonstrate the approach, we apply our method to multiplayer match logs from Battlefield 3 consisting of over 1200 players and 500,000 matches.

Introduction

Clustering is an essential game analysis tool for understanding player strengths and preferences. For example, clustering techniques have been used to identify player preferences for using vehicles over direct combat (Drachen et al. 2012), for taking time to solve puzzles over running through content (Drachen, Canossa, and Yannakakis 2009), for understanding how guilds evolve over time (Thurau and Bauckhage 2010), for quantifying how players differ from an idealized player (Holmgard, Togelius, and Yannakakis 2013), or for associating emotions with game content (Nogueira et al. 2014). Clustering techniques aid designers, developers and producers by enabling them to test assumptions about how players actually interact with their game (El-Nasr, Drachen, and Canossa 2013).

Typically, approaches to player clustering are based on match outcomes such as kills, deaths, and score. However, in this paper we investigate a semi-parametric method for clustering players based on how their choices affect the game outcome (e.g. latent player styles), rather than on their outcomes directly. Such latent player styles do not model the in-game behaviors of players, but rather how the options taken by the player relate to his performance. In our approach, we fit a linear regression model on total score as a function of the character rank, roles, game type, and map chosen by each player for each match. This regression model is used to estimate both global and player-specific weights on each input feature. The set of player-specific weights then define a player's style: how each role/game/map choice relates to their performance. We then use a semi-parametric Bayesian clustering procedure to discover groups of players that have similar weights. These weights are interpretable as player-specific measures of how well that player performs relative to the global performance across all players.

Intuitively, clustering on regression coefficients rather than on outcomes directly has the potential to identify *why* those players are high or low-performing, not just *if* certain players are high or low-performing. Additionally, our approach can work with a very compact representation of the game play data, with each player in a match described only by of a sparse set of indicator variables (for roles/game types/maps chosen) along with the character rank and match score.

There are many clustering procedures that could be used to group players based upon their play styles, with k-means clustering being the most common method. Our use of a model-based semi-parametric Bayesian clustering procedure has two important advantages. First, the number of clusters (unique player styles) does not have to be prespecified. K-means clustering is used as an initialization of our procedure, but the number of clusters can grow (or shrink) as the algorithm proceeds based on whether extra (or fewer) clusters are needed to best fit the observed data.

In addition to allowing the number of clusters to change, our algorithm also allows players to move between clusters. This soft clustering strategy accommodates two types of variability shown by players. In one case, certain players may have a play style that is a true hybrid between two (or more) common play styles shared by many players. In another case, certain players may transition to a new play styles over the course of their gaming career, and so move between play style clusters as we observe more matches for that player.

To demonstrate this idea, we process post-match data from a large dataset of player-versus-player online match data from Battlefield 3, consisting of over 1200 players and 500,000 matches.

Related Work

Existing research has shown many compelling examples of how game analytics can aid game development, maintenance and design (El-Nasr, Drachen, and Canossa 2013). Of these, clustering methods are a subset of analytics techniques which are useful for identifying classifications of players. For example, Sifa et al. (2013) used archetype analysis to understand how player styles evolved during Tomb Raider: Underworld, building on previous work by Drachen, Canossa, and Yannakakis (2009) which modeled styles of game play with self-organizing maps. Drachen et al. (2012) compared the results of k-means and Simplex Volume Maximization (SIVM) in identifying player profiles from the games Tera and Battlefield 2: Bad Company 2. Thurau and Drachen (2011) compared the results of k-means, PCA, archetype analysis, and non-negative matrix factorization for clustering World of Warcraft players according to how their character level changed over time. Thurau and Bauckhage (2010) used k-means and a scalable variant of archetype analysis to identify differences between guilds in World of Warcraft. Nogueira et al. (2014) used a fuzzy clustering technique for modeling how player emotions related to game events. Holmgard, Togelius, and Yannakakis (2013) used a hierarchal clustering method to group players based on how they differed from a "perfect" automated player. Bauckhage et al. (2014) clustered trajectories of players from Quake III to identify hotspots of player activities. The above techniques use a quantitative approach to mine categorizations from a dataset, but qualitative approaches, where categorizations are defined a priori, can also be used. For example, Tychsen and Canossa (2008) define design-based clusterings of players, called personas, which can then be corresponded with empirical metrics.

Our clustering procedure is a quantitative approach closely related to the DP-means procedure of Kulis and Jordan (2012), though our focus is on clustering player-specific regression coefficients (that define latent player styles) rather than means of Gaussian variables. Also, since we revisit the clustering decision for players over multiple iterations, our algorithm does not produce a hard clustering as you would get from DP-means. Rather, players are able to switch clusters (i.e. move between latent player styles) based on new match data from those players.

The Dirichlet process was first introduced in Ferguson (1974) and has become the basis for many successful clustering applications (Griffin and Steel 2006; Teh et al. 2006). See Muller and Quintana (2004) for a good review of DP-based clustering.

Dataset

We demonstrate this method on a dataset of over 500,000 online player-versus-player post-match data taken from over 1200 players of Battlefield 3 from 2012 to 2014. Battlefield 3 is first-person military themed game which allows players to take on a variety of weapons and vehicles in diverse environments across the globe, ranging from tight urban landscapes to open desert. The dataset was provided by EA through the Wharton Customer Analytics Initiative (WCAI).

For each match, this dataset contains information about the player's chosen roles (e.g. assault, support, recon, engineer along with the vehicle roles armored land, unarmored land, helicopter, boat, and jet), match environment (e.g Grand Bazaar, Noshahr Canals, or Operation Metro), and the game type (e.g. conquest, rush, and death match) as well as the player's rank (e.g. a measure of their progression and skill) and match outcome in terms of total score, number of kills and number of deaths. The player may choose more than one role in a single match or might join a match late or quit a match early. We only model player matches for which the player played more than 5 minutes and accumulated more than 100 points (shorter matches with less than 100 points tend to correspond to matches where a player quit early or joined late and thus did not have much time to accumulate points). In this paper, we focus on total score as an outcome variable since it includes points due to both combat and objectives and thus is a good indicator of how players performed overall, regardless of whether their team won or not.

Methodology

Bayesian nonparametric clustering finds subsets of players that all share highly similar parameter values, where parameter values are learned from observed match data. In our approach, we fit a linear regression model on each player's total score as a function of the character rank, roles, game type, and map. The set of player-specific weights from our regression model define a player's latent style: how each role/game/map choice relates to their performance.

The first component of our model involves a predictive regression model of total score for each match m of player j as a function of observed covariates of the match, such as the character rank and taken roles.

$$\begin{aligned}\text{Score}_m = \beta_0 + \beta_1 \text{Rank}_m + \beta_2 \text{Roles}_m + \\ \beta_3 \text{Games}_m + \beta_4 \text{Maps}_m + \ldots + \varepsilon_m \quad (1)\end{aligned}$$

where Rank_m is a integer measuring the character's progression in the game so far and Roles_m, Games_m and Maps_m represent indicator variables for the roles, game type, and map that are utilized in match m. For example, if the player used the helicopter in match m, the corresponding role indicator variable would be 1; otherwise, it is zero. ε_m is the residual of the regression after learning each weight β_i. Score_m is the total score for the player in that match, modeled on a log scale so it is normally distributed.

We based these covariate choices on the information provided but additional (or alternative) covariates could be incorporated into the model if available. The coefficient values $\beta_0, \beta_1, \beta_2, \beta_3, \ldots, \beta_F$, where F is the number of features, represent the global predictive effects (e.g. over all players) of every feature on the outcome variable, Score_m.

The model given in equation 1 provides insight into the relative importance of different covariates, but is insufficient for our goal of accounting for differences among players in terms of play styles. We account for this heterogeneity by

introducing player-specific effects into the model,

$$\text{Score}_{jm} = \beta_{0j} + \beta_{1j}\text{Rank}_{jm} + \beta_{2j}\text{Roles}_{jm} + \beta_{3j}\text{Games}_{jm} + \beta_{4j}\text{Maps}_{jm} + \ldots + \varepsilon_{jm} \quad (2)$$

The intercept parameter β_{0j} captures the overall ability of player j, whereas the collection of coefficients $\beta_j = (\beta_{1j}, \beta_{2j}, \beta_{3j}, \ldots, \beta_{Fj})$ represent the player-specific responses of player j to the covariates of the particular match m. These player-specific indicator variables are non-zero only when the player participated in match m and took on a specific role, map, or game.

We infer these parameters β_j using a Bayesian clustering approach based on the Dirichlet process (DP), which allows for the grouping players into distinct player styles through the clustering of similar behaviors. Intuitively, a Dirichlet process prior can be viewed as a more flexible alternative to traditional random effects models, such as $\beta_j \sim \text{Normal}(\mu, \sigma^2)$, where the player-specific parameters would be shrunk towards a single mean. In contrast, a DP prior allows the player-specific parameters to be shrunk towards multiple centers that are shared by subsets of players.

A desirable consequence of this DP model formulation is that the player-specific parameters β_j for player j are clustered together with other highly similar player-specific parameters β_i from other players. Thus, we will be creating a data-driven grouping of players that exhibit similar in-game behaviors, where those behaviors are defined as the effects on game score of the roles, activities and other covariates taken on by that player. This approach also allows for a subset of players to be left ungrouped from the rest of the population.

Equations 1 and 2 combine to create a sparse linear system $Y_{obs} = X\beta$ which we solve to estimate β. X is a $M \times F$ matrix where each row corresponds to a single player's choices from a single match and each column corresponds to either a global or player input feature.

$$X = [X_{\text{global}} \mid X_{\text{player}_1} \mid \ldots \mid X_{\text{player}_N}]$$

The values of X_{global} are derived from equation 1 and the values of each X_{player_j} are derived from equation 2. Note that for each row, only one set of player-specific values will be non-zero. Also, in cases where a player has never tried a role, map, or game, the corresponding column will be entirely zeros. We remove these all-zero columns from X. The rows of Y_{obs} correspond to the player's score from each match, Score_m, on the log scale to keep quantities normally distributed. The estimated $\hat{\beta}$ is a column vector with the form

$$\hat{\beta}_j = [\hat{\beta}_{\text{global}} \mid \hat{\beta}_{\text{player}_1} \mid \ldots \mid \hat{\beta}_{\text{player}_N}]^T$$

Each $\hat{\beta}_{\text{player}_j}$ is a latent player style for player j.

Bayesian Iterative Clustering

We use a semi-parametric Bayesian clustering procedure to discover groups of players with similar weights. The algorithm is initialized with a starting number of clusters using k-means and then iteratively refined. In each cluster refinement step, we score how well each player fits in each existing cluster as well as how well the player might fit a potential new cluster. We then adjust memberships based on these scores. Figure 1 and Algorithm 1 illustrate and summarize this process.

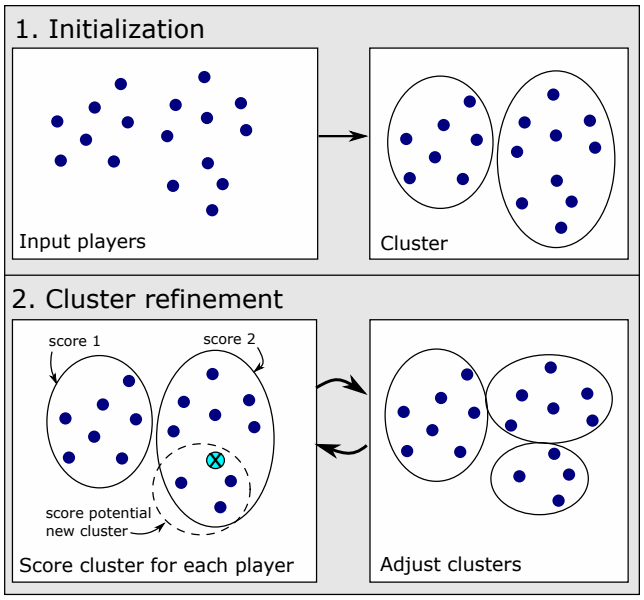

Figure 1: Clustering overview. Each player is represented as a set of weights β learned from a linear regression model relating players' choices to their scores. Our semi-parametric Bayesian cluster procedure contains two steps: First, we initialize our clusters using k-means and then, we iteratively refine these clusters. In each cluster refinement step, we score how well each player fits in each existing cluster as well as how well the player might fit a potential new cluster. Based on these scores, we adjust the memberships of each cluster.

To start, each player will have his own set of coefficients $\hat{\beta}_{\text{player}_j}$ computed to fit $Y_{obs} = X\beta$. As the algorithm progresses, player coefficients will be replaced with their corresponding cluster centers. Note also that because global coefficients will not change, we can compute player coefficients efficiently based only on residual observations Y_{res}

$$Y_{res} = Y_{obs} - X_{\text{global}}\,\hat{\beta}_{\text{global}}$$

where $\hat{\beta}_{\text{global}}$ is the estimate of just the global parameters.

Given player-specific $\hat{\beta}_j$s, we initialize the algorithm using k-means. Because the number of groups are allowed to change, we set the initial number of groups using the heuristic $\sqrt{N/2}$, where N is the number of players (Mardia, Kent, and Bibby 1980). Once clustered, we compute the mean μ and standard deviation Λ over the player-specific coefficients to estimate a multivariate normal distribution $\beta_k \sim MVN(\mu, \Lambda)$ over the starting clusters. This multivariate distribution will be used during cluster refinement to create potential new clusters.

The starting clusters are iteratively refined using the in-

Initialization:

Solve for $\beta = [\beta_{\text{global}} \mid \beta_{\text{player}_1} \mid \ldots \mid \beta_{\text{player}_N}]^T$;
Cluster all β_{player_i} using k-means;
Compute MVN(μ, Λ);
Compute RMSE based on cluster centers;
Set $\sigma^2 = \text{RMSE}^2$;

Cluster refinement:

Foreach iteration t in NumIterations
  Foreach player j in players
    Foreach cluster k in clusters;
      Compute score Q_k;
    Compute score Q_* for potential new cluster;
    Sample player j into cluster based on scores;
  Update clusters
    Update cluster centers;
    Update RMSE and σ^2;
    Remove empty clusters;

Algorithm 1: Bayesian iterative clustering overview.

sample root mean square error (RMSE) to improve each cluster.

$$\text{RMSE} = \sqrt{\frac{\sum(Y_{res} - \hat{Y})^2}{M}}$$

where $\hat{Y} = X_{\text{player}}\hat{\beta}_{\text{player}}$ are scores estimated with player-specific coefficients and M is the number of matches. In our tests, the RMSE usually converges in less than 5 iterations, but we run the algorithm for longer so that we can also analyze players' cluster switching behavior.

During cluster refinement, we revisit the cluster assignments for each player. Each iteration, we quantify how well each cluster center β_k estimates the player's scores with

$$Q_k = \log(n_k) - \frac{1}{\sigma^2}\sum((Y_j - \hat{Y}_{jk})^2)$$

where n_k is the number of members in cluster k, $\sigma^2 = \text{RMSE}^2$ from the previous iteration, Y_j are the observed residual scores for that player, and $\hat{Y}_{jk} = X_{\text{player}_j}\hat{\beta}_k$ are scores estimated with the cluster center. The second term in the equation above corresponds to a normal likelihood that is evaluated on the log scale in order to handle the large values generated by the size of our dataset. The first term in the equation above comes from the Dirichlet process prior.

Additionally, we compute a candidate new cluster β_* sampled from $MVN(\mu, \Lambda)$ and assign it a score similarly

$$Q_* = -\frac{1}{\sigma^2}\sum((Y_j - \hat{Y}_{j*})^2)$$

Players are stochastically assigned to a cluster based on all scores Q. First, scores are renormalized with $Q_k = \exp(Q_k - \max(Q))$. Then, we sample player j into one of the clusters (or the new cluster) with probabilities proportional to $P = (P_1, ..., P_k, P_*)$, each $P_k = Q_k / \sum(Q_k)$. The player-specific β_i is set to the corresponding β_k (or β_*) for the sampled cluster. Note that if β_* is selected, you have created a new cluster with size 1.

Once done looping over all players, we store the player cluster IDs for each player, remove any empty clusters, and adjust the cluster centers to the mean of the member's βs.

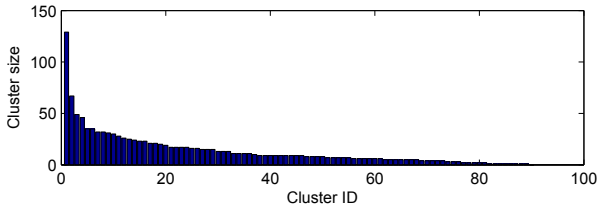

Figure 2: Cluster sizes. The algorithm output 90 clusters with the largest 30 clusters containing 72% of the players

Results

We demonstrate this method on a dataset of 515,605 player-versus-player match logs taken from 1221 players. We use 58 features: an intercept term, player rank, 9 roles, 17 game types, and 30 maps. The intercept term corresponds to β_0 in equation 1 or β_{0j} in equation 2 and in the following sections, we will also refer to β_{0j} as the player coefficient. We initialize the algorithm using k-means with 25 clusters and perform 10 iterations. The in-sample RMSE is reduced from 0.83 to 0.79 during the cluster refinement stage, converging after the third iteration. The algorithm outputs 90 clusters with the largest 30 clusters containing 72% of the players (Figure 2).

With our Matlab implementation, the cluster initialization takes 1.5 minutes and the cluster refinement takes approximately 3 minutes on a 64-bit laptop having 8 GB RAM and 2.5 GHz processors. Our sparse matrix takes 20 MB of disk space and could be optimized to require less space.

We also look at the prediction accuracy of the resulting models using hold-out testing (with the training set consisting of a randomized 90% of each player's matches). A baseline model consisting solely on the single global average has hold-out RMSE of 0.97. The linear regression model, containing both global and player features (41,343 total) has hold-out RMSE of 0.79 (a 19% improvement). The corresponding cluster-based model using the approach in this paper has hold-out RMSE of 0.80 (18% improvement) but only requires a fraction of the features (5220 total, or 13% of the full model).

Latent Player Styles

Recall that we fit a linear regression model on each player's total score for a match as a function of the rank, roles, games, and maps chosen by that player in the match and that the set of player-specific weights from our regression model define a player's style: how each role/game/map choice relates to their performance. Player weights close to zero indicate that the player performs close to average, whereas large weights indicate big differences. For example, a large positive rank indicates that players improve more quickly

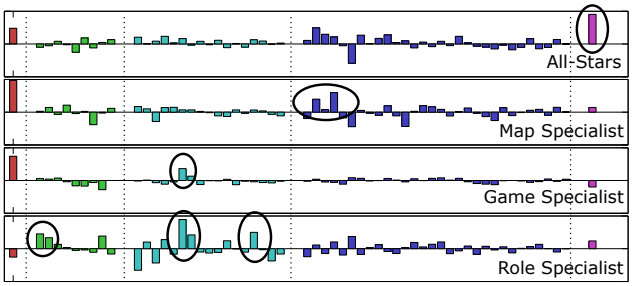

Figure 3: Visualization of four example clusters. The height of each bar represents the weight magnitudes for each cluster center. The colors indicate the type of feature: character rank (red), role (green), game type (cyan), map (blue), and player intercept (purple). Members of "All-Stars" tend to perform above average in every rank, role, game, and map, as indicated by the large coefficient for player. Members of our "Map Specialist" example have a high weight on two maps, Operation Metro and Grand Bazaar, as well as a large weight on rank. Members of our "Game Specialist" example have their highest weights in team death matches. Members of our "Role Specialist" example have their highest weights in assault as well as team death matches.

as they gain ranks than players overall. The largest group computed by our algorithm corresponds to players who perform above average over all. Smaller groups correspond to players who perform best with specific combinations of role, game, and map, or who generally perform below average (usually, composed of low rank characters). Note that roles, maps, and game types are often inter-related since each map will support a subset of game types and roles, e.g. if a player excels at a given map, they often have higher scores for the roles and game types associated with the map as well. Below we give descriptions of several representative clusters, visualized in Figure 3.

All-Stars, 11% Members of this cluster tend to perform above average in every rank, role, game, and map, as indicated by the large coefficient for player (Figure 3, top). In Figure 4 (Top), we summarize the score distributions of a representative member of this cluster. The average log score for our dataset was approximately 8, whereas the player's average log score is consistently close to 9.

Map Specialist, 6% Many clusters have particular map types as their primary drivers of score. In this example, members have a high weight on rank as well as two maps: Operation Metro and Grand Bazaar. (Figure 3, second from top). In Figure 4, we show a representative member of this cluster, who consistently performs well in Operation Metro. The high weight on rank is also reflected in this player's consistently above average scores.

Game Specialist, 4% Many clusters have a game type as the primary driver of score. In this example, members have a high weight on rank and on two types of team death matches. (Figure 3, third from top). Figure 4 shows a representative member of this cluster, who performs significantly better than average in this game type.

Role Specialist, 3% Many cluster types have similar top weights on a role with particular game types and maps that it tends to be well suited for. In this example, members have their highest weights in team death match and assault. (Figure 3, bottom). Figure 4 shows a representative member of this cluster, who performs significantly better than average in one team death match type as well as assault.

Analyzing Player Memberships

During cluster refinement, the transitions players make between clusters is a measure of how well the player fits a particular play style. We distinguish between players whose membership in a group is relatively stable ('sticky" memberships) over memberships which tend to be weak. Sticky members tend to have characteristics very similar to the cluster center. In this analysis, where we run the cluster refinement for 10 iterations, we define players who transition to the same cluster 5 or more times as sticky. With this criteria, 28% of players had stable memberships. For example, the representative players described in the previous section (Figure 4) are sticky members and hence share the closest similarities with their corresponding cluster centers. We define hybrid memberships as those where a player transitions to two groups 4+ times each. With this criteria, 7% of players were hybrids. In some cases, hybrid players belong to similar clusters, e.g. those having a largest weight on the same feature in both. The average number of unique group transitions per player was 4.9. A small group of players (3%) transitioned 9 or more times. The majority of these players (77%) played fewer than 30 games and were beginner rank and hence did not have much information yet to base a classification on.

Discussion and Future Work

This work investigates a method for computing clusters from post match player-versus-player games, using a Bayesian clustering approach. The predictive regression model which is the basis for the player coefficients uses a very compact representation of each match, primarily consisting of indicator variables. In future work, we plan to investigate the effect of additional features, the effect of feature choices on the discovered clusters, and the effect of different outcome variables on the discovered clusters.

This approach doesn't require the number of clusters to be set a priori, and both the computed clusters and the transitions of players provide potential insights for analysis. Although cluster interpretation still requires intuition on the part of the analyst, the resulting weights are straight forward to interpret: they represent differences between the player and the global averages for each feature. Our approach focused on the largest weights for each cluster center and then looked at the profiles of sticky members. This resulted in clusters which segmented players by their strengths and weaknesses according to rank, role, map, and game. Future work will investigate whether such clusterings can help players improve their own performance or help matchmaking through combining players with complimentary skill sets.

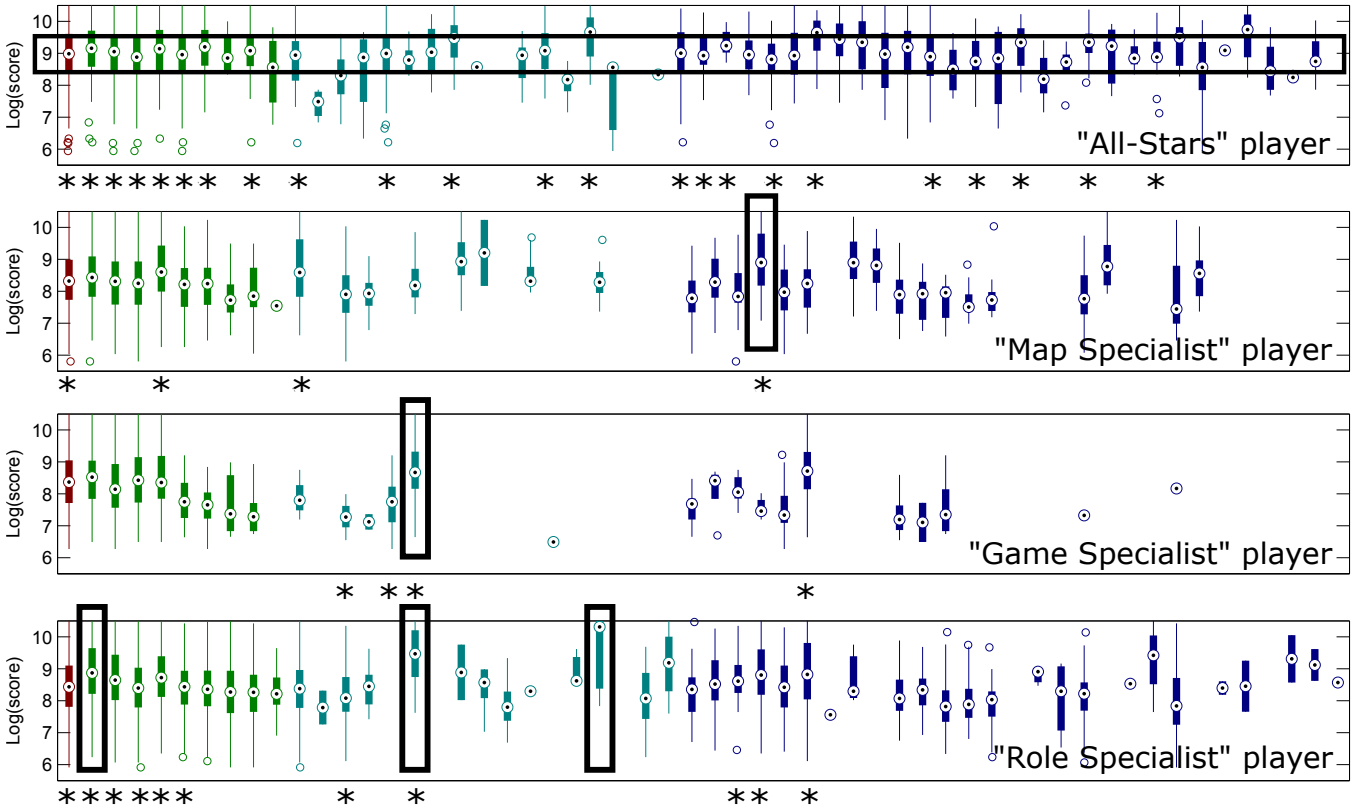

Figure 4: Box plots of player scores. Each player is a "sticky" member of our four cluster examples. From left to right, colors indicate the distribution type: overall (red), by role (green) , by game type (cyan), and by maps (blue). Blank spaces indicate that the player never tried a particular role,game, or map. Stars indicate that the player's average is significantly different from the global average (determined by T-test). From top to bottom, "All-stars" includes members who tend to perform above average in every role, game, and map; "Map specialist" includes members whose best performance is associated with certain maps (with the game types played on those maps also having higher scores); "Game specialist" contains members whose best performance is associated with a game type. "Role specialist" contains players whose scores are best with a single role and associated game types.

Drachen et al. (2012) computed player styles for Battlefield 2: Bad Company 2 using k-means and SIVM. Their analysis, which was based on outcome features (total score over all games, total scores in each role (recon, assault, support, and engineer), duration of time spent in vehicles, etc.), validated the fundamental ways in which players can approach the game, which is independent from the game type and map. We view the approach in this paper as complimentary. In addition to computing clusters based on player strengths and weaknesses on different maps, roles, and game types, we also can gain insights into how well a player fits into a cluster by analyzing how "sticky" their membership is.

Although our dataset is by no means a toy sample, it is still relatively small compared to the millions of players who play Battlefield 3 online. Because the feature set is very sparse, this technique should support data sets with many more players than demonstrated here. However, we also wish to investigate how to adapt the algorithm for an online setting where rounds are processed in order by batches. Such an approach has the potential to work with extremely large numbers of players over long periods of time. Furthermore, because clusters can change, we are interested in how evolving clusters reflect the progression of players over time.

Acknowledgments

We wish to thank Electronic Arts and the Wharton Customer Analytics Initiative (WCAI) for their feedback and support.

References

Bauckhage, C.; Sifa, R.; Drachen, A.; Thurau, C.; and Hadiji, F. 2014. Beyond heatmaps: Spatio-temporal clustering using behavior-based partitioning of game levels. In *Computational Intelligence and Games (CIG)*, 1–8.

Drachen, A.; Sifa, R.; Bauckhage, C.; and Thurau, C. 2012. Guns, swords and data: Clustering of player behavior in computer games in the wild. In *Proceedings of IEEE Computational Intelligence in Games*.

Drachen, A.; Canossa, A.; and Yannakakis, G. 2009. Player modeling using self-organization in tomb raider: Underworld. In *Computational Intelligence and Games, 2009. CIG 2009. IEEE Symposium on*, 1–8.

El-Nasr, M. S.; Drachen, A.; and Canossa, A. 2013. *Game Analytics: Maximizing the Value of Player Data*. Springer.

Ferguson, T. S. 1974. Prior distributions on spaces of probability measures. *Annals of Statistics* 2:615629.

Griffin, J., and Steel, M. 2006. Order-based dependent dirichlet processes. *Journal of the American Statistical Association* 101:179194.

Holmgard, C.; Togelius, J.; and Yannakakis, G. N. 2013. Decision making styles as deviation from rational action: A super mario case study. In *Artificial Intelligence and Interactive Digital Entertainment Conference (AIIDE)*.

Kulis, B., and Jordan, M. I. 2012. Revisiting k-means: New algorithms via bayesian nonparametrics. In *International Conference on Machine Learning (ICML)*.

Mardia, K. V.; Kent, J. T.; and Bibby, J. M. 1980. *Multivariate Analysis*. Academic Press.

Muller, P., and Quintana, F. A. 2004. Nonparametric bayesian data analysis. *Statistical Science* 19:95110.

Nogueira, P. A.; Aguiar, R.; Rodrigues, R. A.; Oliveira, E. C.; and Nacke, L. 2014. Fuzzy affective player models: A physiology-based hierarchical clustering method. In *Artificial Intelligence and Interactive Digital Entertainment Conference (AIIDE)*.

Sifa, R.; Drachen, A.; Bauckhage, C.; Thurau, C.; and Canossa, A. 2013. Behavior evolution in tomb raider underworld. In *Computational Intelligence in Games (CIG)*, 1–8.

Teh, Y.; Jordan, M.; Beal, M.; and Blei, D. 2006. Hierarchical dirichlet processes. *Journal of the American Statistical Association* 101:15661581.

Thurau, C., and Bauckhage, C. 2010. Analyzing the evolution of social groups in world of warcraf;. In *Computational Intelligence and Games (CIG)*, 170–177.

Thurau, C., and Drachen, A. 2011. Introducing archetypal analysis for player classification. In *Foundations of Digital Games Conference, EPEX Workshop*.

Tychsen, A., and Canossa, A. 2008. Defining personas in games using metrics. In *Proceedings of the 2008 Conference on Future Play: Research, Play, Share*, (Future Play '08), 73–80.