INTL JOURNAL OF ELECTRONICS AND TELECOMMUNICATIONS, 2026, VOL. 72, NO. 1, PP. 1-9
Manuscript received November 17, 2025; revised January, 2026. doi: 10.24425/ijet.2026.157873

Framework for intelligent prediction of the mobile players retention

Piotr Bilski, and Adrian Bilski

***Abstract*—The paper presents the novel approach for the analysis of the mobile game players' data to predict retention, i.e. duration of remaining inside the game. The definition and measures of evaluating retention from the mobile game perspective are described. The architecture of the implemented system is presented with the detailed explanation of the subsequent (specifically data collection and processing) modules. Three specific problems related to the retention analysis (focused on the gameplay duration and making financial transactions using actual currency) are presented. Task-oriented evaluation measures (to learn about the accuracy of the proposed approach) are described. Experiments regarding the efficiency of the proposed approach using the data set from the My Spa Resort mobile game from CherryPick Games company are presented, proving usefulness of the approach. Future prospects of the solution and technical limitations are described.**

***Keywords*—mobile gaming; retention analysis; artificial intelligence; player analysis**

I. Introduction

MOBILE games designed for the freemium business model are currently popular due to the ubiquitous existence of handheld devices able to run them. They are run by Google Android or Apple iOS operating systems and maintain constant connection with the game server. The latter allows for storing the user data, keeping the application updates and adjusting the gameplay to the individual performance and abilities. The intrinsic aspect of such games is that they are free to download and play, but contain the additional paid features, accessible after making the transaction with the real-world currency (for instance, to buy in-game resources or switch off the commercials). Because gameplay in most cases depends on the Internet connection, it is also easy to record behavior of the users and their actions within the game (independent from the operating system used) [1]. This allows for profiling gamers and determine their preferences or reaction to the specific features through data engineering (analysis of player-oriented information collected in the global repository). Because the game is developed and may be updated constantly, this gives the opportunity to modify the gameplay and test new elements on-line.

The business model of mobile games relies on two factors: users spending their money on virtual assets and staying in-game long enough to be presented with commercials (which benefit the game owner). Both depend on the overall game quality and its attractiveness for the user. It is difficult to design the gameplay optimally and for that purpose the separate module analyzing users' (direct or indirect) feedback would have to be designed [2]. Assuming the game is completed (or its updates are independent of the user feedback), it is important to evaluate the players' retention and isolate the most attractive ones, i.e. staying in-game for the longest duration possible and/or spending their money. The separate task is the isolation of the users making payment multiple times, as they are the most devoted to the game and their preferences should be addressed by the developer in the first place. This would enable maximization of profits from the created product.

The paper presents a generic approach to evaluating and predicting the gamers' retention in mobile free-to-play games. The detailed aims, i.e. determining the future behavior of the player, identification of the most prominent gamers regarding in-game payments and the gamers most eagerly paying, are presented. The possibly universal software architecture of the system operating on the collected data is described. The algorithmic details of data processing are iterated, showing the crucial steps in determining the retention-oriented aspects. Experimental results proving efficiency of the approach are discussed, using actual data extracted from the My Spa Resort game from the CherryPick Games company (https://cherrypickgames.com/). The work is an extension of experiments presented in [3] (where players' preliminary analysis is discussed) and refers to the data set refined from the "raw" data collected from the cloud storage. Two main problems are solved in the following way:

- identification of the most probable behavior for the selected gamer based on the analysis of historical data of previous players.
- Discovery of the most prominent features for the most attractive users from the game developers from an economic standpoint.

Based on the preliminary analysis of data and selection of applicable algorithms for specific tasks the complete system for the holistic retention prediction was prepared.

The structure of the paper is as follows. In Section 2 state of the art regarding the gamers' retention analysis is discussed. Section 3 covers the proposed solution, deployed in the company infrastructure. In Section 4 the detailed tasks executed on the collected data are presented. Section 5 describes the analyzed game data set. In Section 6 experimental results are

This text summarizes a part of research within a EU co-funded project POIR.01.01.01-00-092 7/17 - Cherrylytics.

Piotr Bilski is with Warsaw University of Technology, Poland (e-mail: piotr.bilski@pw.edu.pl)

Adrian Bilski is with Warsaw University of Life Sciences, Poland (e-mail: adrian_bilski@sggw.pl)

demonstrated, showing the actual benefits of the applied solution to the game vendor. Section 7 contains conclusions with prospects for the presented approach.

II. State of the Art

Player retention and churn prediction in video games (particularly in free-to-play and subscription-based models) has emerged as a critical research area at the intersection of game analytics, machine learning, and behavioral modeling. High competition in the gaming industry makes player retention not only a matter of engaging game design but also a strategic lever for monetization and loyalty building.

Early approaches to churn prediction in gaming often relied on simple heuristics, such as thresholds for periods of player inactivity. In [4] it was shown that such simplistic methods, while easy to implement, fail to capture the nuanced behavioral patterns that precede churn. Their work on The Settlers Online compared four labeling approaches and eight classification algorithms, concluding that Random Forest models using sliding windows achieved AUC values above 0.99 and prediction accuracy over 97%, outperforming prior studies that typically reached 60–90% accuracy. Importantly, they emphasized the distinction between churn and disengagement, recommending tailored labeling strategies for each.

Other research extended this focus to rapid retention prediction. In [5] short-term retention forecasting in mobile free-to-play (F2P) titles was investigated, introducing a heuristic modeling approach derived from heavily pruned decision trees. While such methods traded off predictive granularity for speed and ease of deployment, they offered viable solutions for smaller studios and early post-launch analytics.

In [6] supervised machine learning (ML) models were presented for churn prediction in a multiple-choice storytelling F2P game, exploring different observation and prediction windows (1–7 days). Their findings showed that prediction accuracy varied from 66% to 95%, depending on time horizons, underscoring the importance of balancing short-term responsiveness with long-term prediction stability.

In [7] churn in mobile F2P games was addressed, tackling dataset imbalance issues and employing behavioral feature extraction from player logs. Their model, evaluated via cross-validation, achieved high predictive performance, making it suitable for real-world deployment.

In [8] a generalizable approach for predicting disengagement and first purchases using event-frequency-based data representation were proposed. Unlike game-specific feature engineering, their method required no prior knowledge of event semantics, enabling application across different genres. Their experiments demonstrated that “disengagement” as a concept could be more discerning than “churn” in certain contexts.

In [9] a churn prediction framework leveraging Graph Neural Networks (GNNs) on social interaction networks in games was introduced. Their method modeled players as nodes connected by in-game social activities (e.g., friend lists, guild memberships, cooperative events), with graph structures enriched by activity-based features. Experiments showed that integrating social influence into churn models significantly outperformed traditional feature-only approaches, highlighting the role of community dynamics in player retention.

Recent studies have shifted from binary churn modeling toward broader engagement forecasting. The Authors of [10] framed engagement prediction as a multi-class problem (High, Medium, Low), using demographic, behavioral, and in-game performance features. XGBoost emerged as the most effective model, achieving 91% accuracy and offering actionable feature importance insights.

A special segment within churn-related research concerns so-called “white whales” - a small fraction of high-value players who generate a disproportionately large share of revenue in F2P ecosystems. Losing even a few of these players can have an outsized impact on a game’s financial performance. Studies on this subject emphasize that churn prediction for this segment requires tailored models, as their behavioral signals often differ from the broader player base [7,8]. Accurately identifying and retaining white whales is therefore a priority for data-driven live operations teams.

In [11] retention and monetization measurement was approached through the Mean Cumulative Function (MCF), which accommodates censored data—common when many players are still active during analysis. MCF generalized standard retention rates, enabling unbiased estimates of lifetime value and playtime while supporting statistical comparisons between game versions.

In [12] time-varying survival analysis to link game design elements with retention outcomes was applied, addressing the scalability challenges of handling massive time-dependent datasets from AAA titles such as Far Cry 4. Their model incorporated time-varying covariates and coefficients, making it suitable for identifying when and how specific design elements impact quitting behavior.

Retention is also influenced by adaptive game design. In [13] a dynamic programming framework using Q-learning to optimize quality adjustments in subscription-based games was introduced, factoring in players’ memory of past experiences and network externalities. Their simulations showed convergence toward optimal quality levels, revealing that high network externality intensity could paradoxically reduce the benefits of quality increases.

In [14] a dynamic difficulty adjustment algorithm for real-time pattern prediction using accumulated playtime and session counts was proposed. Their method was designed for scalability to millions of concurrent users and successfully applied to multiple datasets, supporting personalization and monetization strategies.

The presented overview shows that the interest in retention analysis is growing, as it may significantly affect the financial benefits developers take from the game. Individual approaches require adjusting the well-known algorithms for classification and prediction. This justifies applications of new approaches, especially based on the actual in-game data.

III. Data Collection and Analysis System

The solution to the presented problems is the system presented in Fig. 1. It operates next to the server-side in the cloud environment and is divided into data collection and data analytics modules. The former is critical as it is responsible for extracting new records from the Big Data infrastructure. The “raw” data records are processed to form the data set which is further used to extract knowledge about the behavior of the players. The implemented software layer is the add-on to the API provided by the game, which operates locally on mobile computers (phones or tablets), uploading to the database all in-game events predefined by the creators. Their actual set depends on the actual games, but it is assumed the following aspects of the game are true:

a. Each event is related to the identifiable player’s action inside the game, for instance, purchasing the building, producing some goods, reaching the specified destination, etc.
b. Events are related to the specific board or level and therefore can be grouped together based on this criterion
c. It is possible to distinguish between the events of different types (so they can be further aggregated)
d. It is possible to put the events in sequence, depending on the timestamps, i.e. moments of occurring the particular event.

The retention measurement [15] requires information about the direct and indirect engagement of the player in the game environment. In most cases time-based parameters are considered (i.e. events taking place in the particular moment). The presented approach considers creating features based on the global time indicators (specific action, like constructing the building, collecting the crops, etc., and its timestamp). This allows for determining what and when the user is doing. Additionally, economic-related events are considered, such as making payments for virtual assets (such as additional ammo, diamonds, etc.) with the real currency. This includes both the fact of making purchases and the amount of spent money.

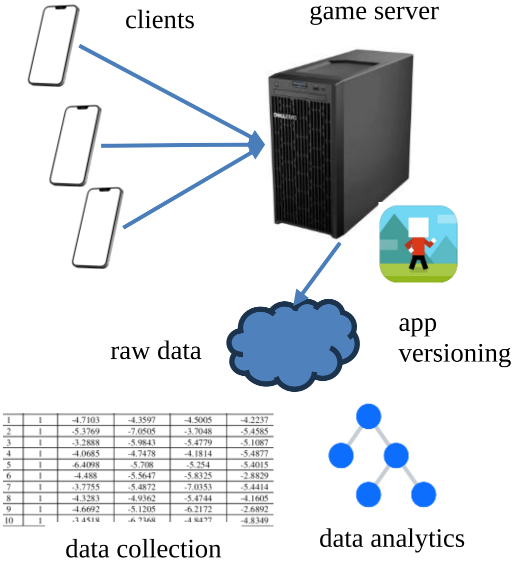
From the retention analysis perspective, it is essential to learn about the overall game duration of each user and the existence of real-currency payments. The latter are relatively rare, but the most proficient economically. It may also be important to isolate the players spending their money multiple times, as their interest in the gameplay is maximal and should be analyzed separately.

Additional detail to be considered is the application version. As mobile games are updated during their lifetime, it may be crucial to know in which version the particular events take place. The app version influences the game environment, determining difficulty level, events in the game, existence of challenges and assets. This may be an additional element for aggregation of data stored in the cloud database.

The presented “raw” events are collected in the Big Data database, which should be periodically (for instance, on the daily basis) queried for new records, and then fed to the analytics module. From them, the actual attributes representing the selected player may be created. It is assumed that the player information should be aggregated according to the particular level of the game advancement. This way vectors summarizing behavior of the gamer pj at the selected level or on a specific board (represented by the discrete l variable indicating the in-game advancement) may be created, as follows:

$$v_p(l) = \{l \quad a_1 \quad \cdots \quad a_m \quad r\} \tag{1}$$

where ai are attributes representing the aggregated behavior of the player at this level (number of buildings constructed, amount of experience points gathered, etc.), and r is the set of parameters representing the information about the retention. The latter includes data about spending actual money inside the game or duration of staying inside the gameplay.



1	1	-4.7103	-4.3597	-4.5005	-4.2237
2	1	-5.3769	-7.0505	-3.7048	-5.4585
3	1	-3.2888	-5.9843	-5.4779	-5.1087
4	1	-4.0685	-4.7478	-4.1814	-5.4877
5	1	-6.4098	-5.708	-5.254	-5.4015
6	1	-4.488	-5.5647	-5.8325	-2.8829
7	1	-3.7755	-5.4872	-7.0353	-5.4414
8	1	-4.3283	-4.9362	-5.4744	-4.1605
9	1	-4.6692	-5.1205	-6.2172	-2.6892
10	1	-3.4518	-6.2368	-4.8427	-4.8349

Fig. 1. Structure of the system for the players’ retention analysis.

The dataset containing all information about the players is then represented in the following form:

$$D = \{\boldsymbol{p}_1 \quad \cdots \quad \boldsymbol{p}_m\} = \begin{bmatrix} v_1(l_1) \\ v_1(l_2) \\ \vdots \\ v_p(l_n) \end{bmatrix} \tag{2}$$

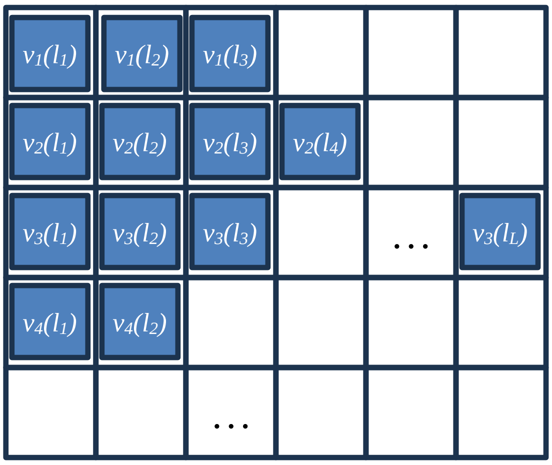
where each player $\boldsymbol{p}_j$ is a set of points in the m-dimensional space. This way it is possible to freely process all their subsets, focusing on the particular gameplay fragments. The condition is that the points can be put in the specific sequence, i.e. based on them it is possible to evaluate the levels that have not yet been achieved.

Analysis of the retention requires information about the activity of the player [17], which defines whether he/she is still playing the game or has already quit (this is determined by the timestamp of the most recent activity in the game). The dataset is supposed to be processed in the input-output scheme (in the supervised way). Attributes defined as output ones may change, depending on the task. Different machine learning algorithms may be used to bind input with output (i.e. expected retention $\overline{r_e}$):

$$\overline{r_e}(\boldsymbol{p_i}) = f(D) \tag{3}$$

calculated as estimation of levels based on the player’s performance so far. This way it would be possible to predict any user based on the available historical data (sets of vectors (2) as in Fig. 2).

The in-game progress is illustrated using the sequence of points (levels), which forms the trajectory. Such patterns can be compared through distance calculation. This allows for treating the player’s history as the time series, where each point is the aggregated advancement at the particular level. The simplest approach for similarity calculation would be to find the closest players through their pattern’s comparison. Due to the multidimensional representation of each vector, the standard implementation of k nearest neighbor classifier cannot be used, requiring the tailored approach. Approaches used for the time series calculation are usually aimed at finding repeating patterns, which does not happen in the typical game, as the progress is usually the increasing curve with the varying steepness.



$v_1(l_1)$	$v_1(l_2)$	$v_1(l_3)$			
$v_2(l_1)$	$v_2(l_2)$	$v_2(l_3)$	$v_2(l_4)$		
$v_3(l_1)$	$v_3(l_2)$	$v_3(l_3)$		…	$v_3(l_L)$
$v_4(l_1)$	$v_4(l_2)$				
		…			

Fig. 2. Data structure for the analysis of the gamers' retention.

IV. LABORATORY TEST STAND

The My Spa Resort game is one of the popular titles produced and maintained by the CherryPick Games company. The user's task is to expand on the development of the spa resort by building infrastructure and preparing multiple assets to attract new clients. The business model of the game includes farming and exploiting in-game resources, for example preparing potions for the particular treatment. Virtual resources are steadily gathered; therefore, it is so important to collect them gradually. The game allows for customization of spa facilities, interaction with staff and clients, and social features like visiting and trading with other players. Through these activities, the users build and maintain their resorts, aiming to optimize resources and client satisfaction while fostering social connections with other players. The game is available for Android and iOS system, though it can also be run on emulators (such as Bluestack [16]). Since the initial installation, multiple versions were published for update, which creates diversity in gamers reception. In the experiments presented, these versions are ignored, but in the future experiments the relation between the application version and the players behavior should be investigated.

Though the game is alive, i.e. multiple users are coming in and going out, the tests were performed on the fixed data set. It was prepared for these experiments and contains 69101 players (identified by the unique automatically generated alphanumeric string, for instance: "eu-west-1:00b278f0-6b67-4e3b-b3ad-ea61786d8447v1").

The set contains data aggregated from the Goodle BigQuery database through the SQL queries run once a day. The minimum level of the users was 1, while the maximum one – 485 and statistical information about the maximum level achieved by the players present in the dataset is in Fig. 4. Users mostly stay at relatively low levels, i.e. up to 10 (quickly losing interest into the game). The most interesting ones would be the small subset of gamers that reach levels between 30 and 50. One player has reached level 485 and is treated as the anomaly (is it is difficult to make any conclusions about his/her behavior).

Fig. 3. Myspa Resort game board.

The actual form of the record generated from the raw data contained the following attributes:

u_l– user level, for which the vector of attributes $v_p(l)$ is created

a_v – identifier of the application version, for which data were collected (in the format major_version.minor_version.variant, for instance: "0.2.17")

n_{xp} – experience points collected at the level l (by performing operations such as farming, building construction, mining resources, creating potions, etc.)

n_e – number of atomic events executed by the player at this level (related to the user operations)

n_p – number of transactions with physical currency made by the user at the level l

n_{cb} – number of constructed buildings – number of buildings constructed in the spa resort in this level

g_t – physical game time (in ms), calculated as the timestamp difference between events and the beginning and the end of the level.

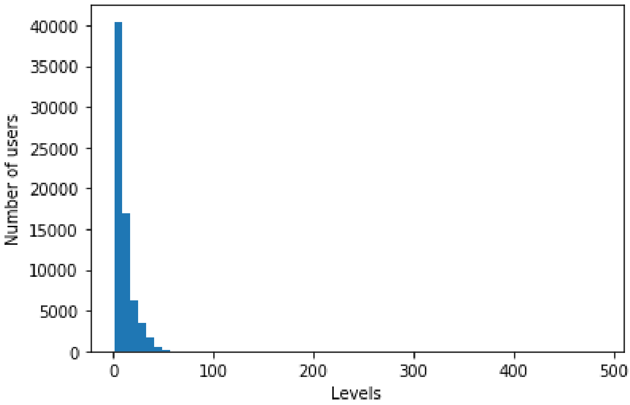


Fig. 4. Statistical distribution of maximum levels reached by the players.

No information about the players background (like nationality, sex, age, etc.) are collected or processed.

Computations have been performed using Python language (version 3.13), which is currently data science standards. For the data set processing. The set of specialized functions regarding the players' retention analysis was prepared using numpy, pandas and scikit learn packages. The presented data set was implemented as DataFrame type. Duration of operations performed by the system was measured on the local workstation, equipped with the Intel Xeon processor E5-1650 v.2 (3.5 GHz) with 6 cores and 32 GiB RAM.

V. SOLVED TASKS

The retention analysis requires solving the defined tasks. Three problems to analyze have been shown and to be separately resolved. Each may provide suggestions for the developers to modify the gameplay or adjust parameters of the program to attract more users or maintain their interest.

A. Prediction of the players' final level

The aim here is to learn about the expected retention of the selected player depending on his/her historical data. It is assumed that multiple players behave similarly at the particular levels. Therefore based on their past behavior it is possible to predict how they will act in the future, especially how long will they remain in the game.

For the data analytics the output variable is the status of the player (i.e. "active" or "inactive"), which is used to divide the dataset *D* into two subsets, according to the these statuses (here *A* and *I*). The first one ("active" players) is the one to be processed for prediction, as we want to learn how it will behave and (the most importantly) when their interest in the game will end. The second subset contains all past achievements and stands for the base of prediction knowledge. The activity status (with the value "0" standing for "inactive" and "1“ – for "active") was calculated based on the difference between timestamps of the most recent in-game action and the current date and the threshold θ being two previous weeks of inactivity):

$$r_e(\boldsymbol{p_i}) = \begin{cases} 1 & if \quad t_{now} - t_{last} > \theta \\ 0 & \quad otherwise \end{cases} \tag{4}$$

where t_{now} is the current timestamp indication, while t_{last} is the most recent activity timestamp. Based on that, 11313 of users were considered active and 57788 – inactive.

The prediction is based on the assumption that the part of the trajectory identical or similar to two gamers determines the similarity between them in the future. Therefore the analysis of all vectors inside the active player and corresponding ones by the inactive player, allows for retrieving set of performances in the remaining boards or levels.

In this case the input (explanatory) variables include (like amount of experience points gained or the number of buildings constructed), while the output (explained) variable is the activity indicator. The task is to show the predicted values of parameters in the incoming levels and the maximum activity level predicted (see the algorithm below).

The proposed algorithm searches all entries of the set *A* and returns the subset *k* of the most similar ones. The similarity measure is the value of the closest distance $d(\boldsymbol{p}_x, \boldsymbol{p}_y)$ between two players. Because each point in the trajectory is *m*-dimensional, considering all of them simultaneously requires one of two approaches:

a. Comparing one-dimensional trajectories (one for each attribute) separately and collecting *k* the most similar for each. Overall, similar is the player the most frequently on k lists (if there are more than one equally frequent, their positions matter).

b. Calculating a single trajectory based on the distances between the whole points. In this case the attributes must be scaled (for instance, through z-scoring or min-max [18]) or selected (data reduction).

Algorithm 1 Maximum level estimation

```
input: $\boldsymbol{p_a} \in A, I$
for each $\boldsymbol{p_i} \in I$:
        $\boldsymbol{d_p} = d(\boldsymbol{p_i}, \boldsymbol{p_a})$
$\overline{l_{max}}(\boldsymbol{p_a}) = l_{max} : min(\boldsymbol{d_p})$
return $\overline{l_{max}}(\boldsymbol{p_a})$
```

Examples of players' history regarding three attributes (i.e. n_{xp}, n_e and n_{cb}) for the low-level user ("beginner") and medium level ("experienced") are shown in Fig. 5-7 and 8-10, respectively. In each case values in the last level are omitted, as they do not represent it as a whole (this level was not completed). The number of experience points collected during the game increases with the curve steepness (as subsequent levels require more actions that generate them). The number of events behaves similarly, minor discrepancies are related with the random gameplay behavior and different amount of resources collected through the preceding levels. These numbers skyrocket on the latter levels, but after reaching the required level of advancement. On the other hand, the number of constructed buildings is not closely related to the specific level, as the promotion between levels is possible through different actions and depends on the users' applied strategy. Also, further into the game, new buildings become available.

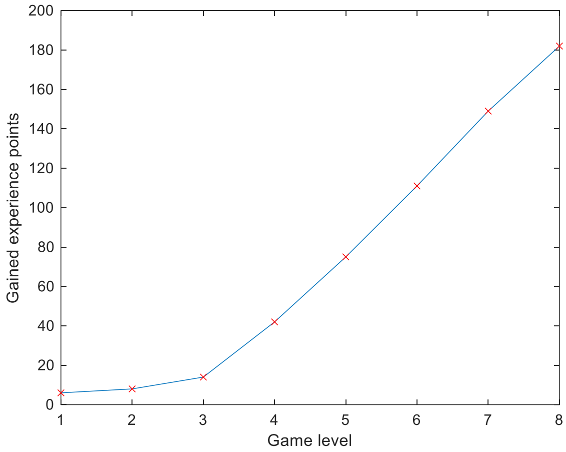


Fig. 5. Trajectory of the obtained experience points for the "beginner" player.

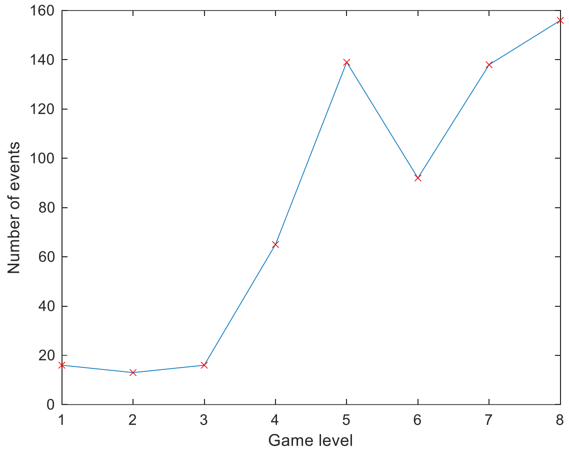


Fig. 6. Trajectory of the occurring events for the “beginner” player.

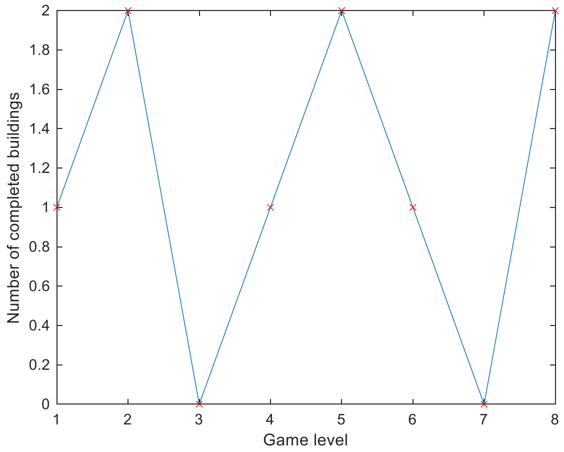


Fig. 7. Trajectory of the constructed buildings for the “beginner” player.

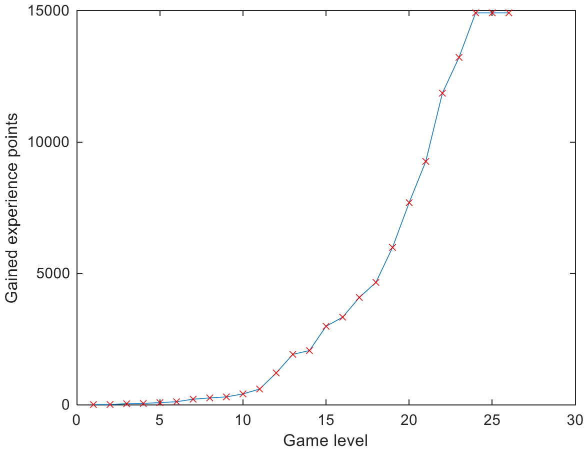


Fig. 8. Trajectory of the obtained experience points for the “experienced” player.

During the trajectory calculation an important issue is what should be the minimum and maximum level used for comparison. Usually the former could be the first one, but in practice initial levels are the most similar and repeatable (there is little freedom of choice), therefore their information capacity is low. For the presented game variability between different users is in Fig. 11.

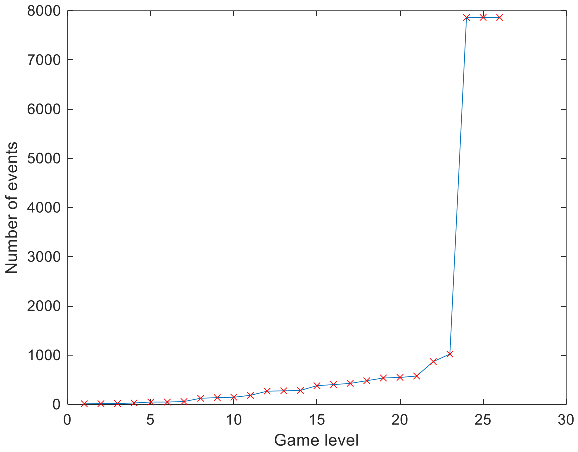


Fig. 9. Trajectory of the occurring events for the “experienced” player.

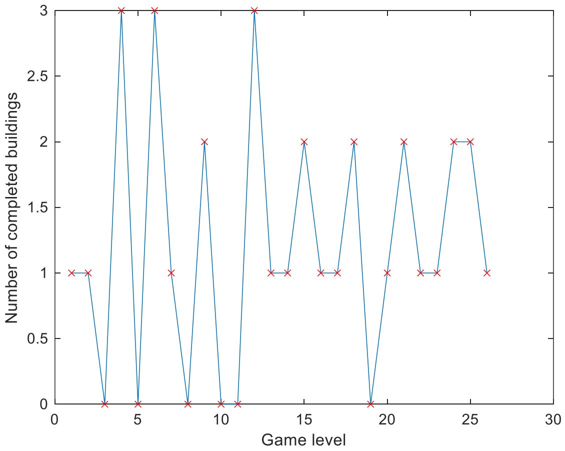


Fig. 10. Trajectory of the constructed buildings for the “experienced” player.

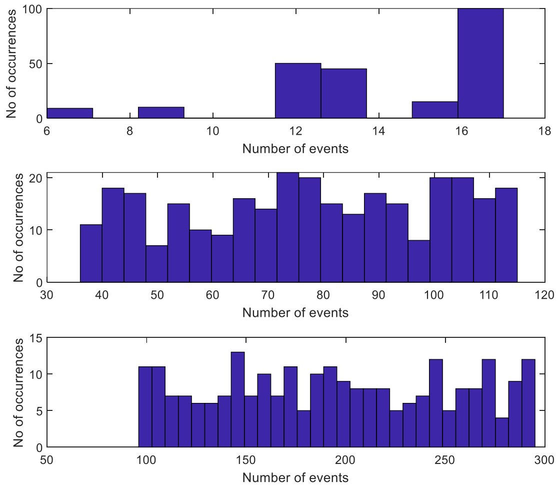


Fig. 11. Variability of occurring events in levels 2 (a), 5 (b) and 10 (c).

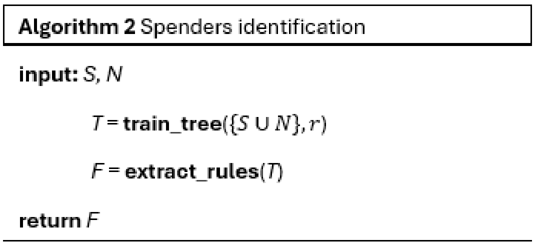
B. *Detection of spending players*

In this task the key attribute is the fact of payments made by the player. The aim is to determine what specific behavior distinguishes the paying players (so-called spenders) from all other players (by default, less attractive from the financial point of view). In the process each record referring to the player is treated as the separate data point. The task to solve is then the binary case classification with the need to learn about the reasons of distinguishing one from the other. The data set *D* is now divided into two subsets, i.e. *S* (spenders) and *N* (non-spenders), disregarding whether the users are active or not (here the "active" boolean attribute is replaced with the "spending" attribute):

$$r(\boldsymbol{p}_i) = \begin{cases} 1 & if \quad \sum_L a_{np} > 0 \\ 0 & otherwise \end{cases} \tag{5}$$

Also, the n_b is excluded from attributes. Based on this criterion, 1488 users have been isolated, making overall 8043 payments (which shows that some group of users were making multiple transactions).

To learn about the specific spender traits, explainable-AI may be used, such as decision tree, with the structure of the classifier easy to read by the human being. The obtained information should include the attributes selected as the important for distinguishing between spenders and all other players with the details about the parameters' values. The operation consists of the decision tree generation (**train_tree**()) and analysis of the decision tree structure, which is the combination of rules, leading to distinguishing between spenders and non-spenders. The function **actract_rules**() produces the structure *F* containing the particular attributes and their values allowing for distinguishing between these two groups of users. Each complete rule allows for isolating a subset of spenders.



Algorithm 2 Spenders identification

```
input: $S$, $N$
        $T$ = train_tree($\{S \cup N\}, r$)
        $F$ = extract_rules($T$)
return $F$
```

The important parameter here is the height of the tree, determining the number of nodes and attribute-value combinations used to classify the users. Though the full adjustment of the tree to the data is considered undesired (i.e. the effect overfitting), here it could be accepted if the extracted knowledge would be easy to follow. Otherwise, the height must be optimized.

The additional challenge in this task is the limited number examples forming the data set, i.e. high imbalance between categories (1488 spenders are around 12% of the whole gamers population in the set *D*). This is the general problem of mobile games, as most users refrain from spending money and financial benefits from them are only through the long duration of in-game participation and watching commercials.

C. Detection of the multiple spending players

The aim is to determine the variability among spenders. Specifically, the task is to distinguish between the players making only one payment and multiple payments. The rationale behind this task is that a single payment may be the random operation of lesser importance. On the other hand, multiple spenders are people devoted to the game enough to be treated separately. Their feedback and behavior should be analyzed carefully, as they bring most of the money. In this case the main problem is the relatively small number of users being the focus of the analysis.

The algorithm for analyzing multiple spenders behavior is the same as in Section *5.B*, but with different data. Now, the set is compressed to spenders only, so *S* becomes *D* and decomposed into two subsets: multiple spenders (*M*) and single spenders (*S*). The binary value of the explained variable is now according to the following criterion:

$$r(\boldsymbol{p}_i) = \begin{cases} 1 & if \quad \sum_L a_{np} > 1 \\ 0 & otherwise \end{cases} \tag{6}$$

The number of users belonging to the set *M* is 378, which (considering the relatively small number of players in *D*) makes this task even more challenging. However, using the decision tree is justified even in sparse data cases (where neural models would be difficult to use).

VI. EXPERIMENTAL RESULTS

The described framework operates on the server collecting and processing data delivered by the players. Despite its practical usages, the system was also tested on the available data sets, but with the changes regarding details presented in Section 5. The in-game duration prediction was evaluated based on all inactive players, some of which are selected to the predicted subset (so their actual performance can be verified).

A. Prediction of the players' final level

To correctly measure the retention, the accuracy evaluation function has to be proposed. Considering two expected results (i.e. the projected trajectory and the expected final level), two elements are required. The first one is the distance between trajectories built on the future levels. Because the single value is required, weighted RMSE was used:

$$f_{et} = \frac{1}{L} \sum_{L > l_{now}} \sqrt{\sum_M w_i \cdot (a_{ik} - a_{il})^2} \tag{7}$$

The second part of the evaluation is the difference between the predicted $\hat{l}_{max}$ and actual l_{max} values of the maximum level reached. The computed value is the integer (i.e. both negative and positive) with "0" being the ideal case (the exact final, level of user's activity has been predicted).

$$f_{el} = \hat{l}_{max} - l_{max} \tag{8}$$

The accuracy is then two-dimensional, with the option of weighing both components.

The experiments consisted in applying the presented measures to the evaluation subset which was randomly selected from the original set *D* through the hold-out cross-validation (in the 7:3 ratio). Table 1 contains averaged results of the retention prediction in terms of RMSE (7) and (8) for different starting l_{start} and ending l_{end} historical levels. The f_{el}^- and f_{el}^+ measures show the average difference between the predicted and actual final levels where the game ended earlier than and later than predicted, respectively. The RMSE required setting up weights to scale the attributes in the following way:

$$\boldsymbol{w} = \{\mathrm{w_{xp}} = 10^{-3}, \mathrm{w_{ne}} = 1, \mathrm{w_{nb}} = 1, \mathrm{w_{bc}} = 1, \mathrm{w_{gt}} = 10^{-7}\}$$

TABLE I
TRAJECTORY ESTIMATION RESULTS WITH THE PREDICTED FINAL LEVEL

l_{start}	l_{end}	$RMSE$	f_{el}^{-}	f_{el}^{+}	f_{el}
2	8	0.183	-1.78	1.52	-0.21
2	6	0.274	-1.97	1.62	-0.17
5	9	0.071	-1.66	1.55	-0.13
5	12	0.068	-1.43	1.49	-0.10

Example of the prediction result for the trajectory is in Fig. 12, where blue and red lines represent the historical analyzed player's data and the closest pattern from the set D, respectively. Also, green and black dotted lines represent the predicted and actual trajectory, respectively.

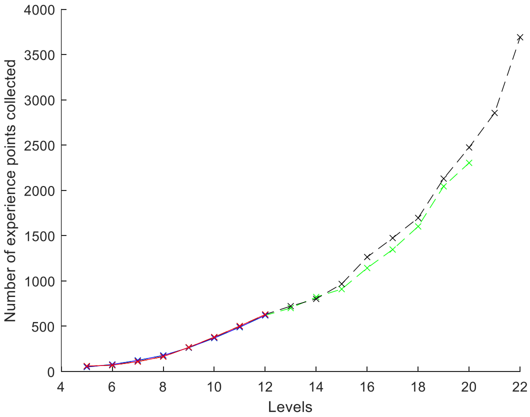


Fig. 12. Variability of occurring events in levels 2 (a), 5 (b) and 10 (c).

Results show that based on both historical trajectories it is possible to predict the behavior of the selected gamer with limited accuracy. From the practical standpoint it was considered close enough to be usable in practice. The more challenging task is the adjustment of the gameplay to make the game more attractive for the users, which is however, a separate problem.

B. *Prediction of the players' final level*

The analysis of the decision trees generated from the dataset was difficult not only due to the small amount of available data, but also the complexity of the given task. Both categories are intertwined, therefore the complete tree constructed to separate them is complicated (with the height of 25). Therefore the knowledge extracted from its structure is difficult to read and of low usability. To simplify the process, the maximum height was set to 4, which facilitates analysis of rules and makes possible to present it visually. Example of such a tree is in Fig. 13 (without the leaves, as they are not important for the feature selection). From the rules it can be stated that the most important feature distinguishing the spenders from non-spenders is the number of obtained experience points, which are visibly higher for the former. The second attribute is the number of events, then the game time. More experience leads to more events, which in turn lengthens playtime and, consequently, increases the probability of being a spender). It means that the spenders are definitely more devoted to the game, remaining inside the game for longer amounts of time.

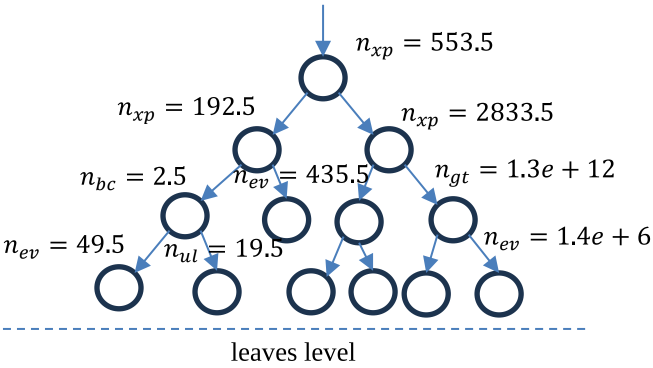


Fig. 13. Structure of the decision tree for distinguishing spenders from non-spenders.

C. *Multiple spenders analysis*

The experiments are similar to the ones presented in Section *6.B* with the similar decision tree structure as before. Multiple spenders are also difficult to be distinguished from the single spenders, but due to the smaller number of examples to process, the tree height was limited to 3. Results show that the number of collected experience points is thill the main factor for the binary classification, but with much higher values. The second attribute is the game time and the number of buildings constructed. The latter usually require large amount of resources, creating the opportunity to quickly get advance through the actual currency.

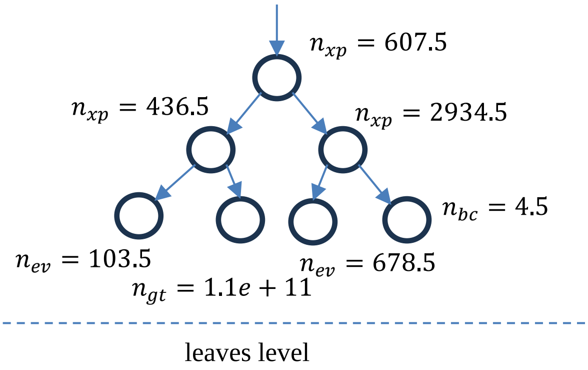


Fig. 14. Structure of the decision tree for distinguishing multiple spenders from single spenders.

D. *Identified limitations*

The problem with the results evaluation may be the determination of the activity status. The presented data set was processed in the specific time instant, but the future behavior of still existing players remains unknown. Analysis of the closed set from the timelapse perspective may show changes in the players status. For instance, the user may be considered as inactive for some time, but later return to the game. Because the inactivity time is not considered in the research, it is unknown how much time should be monitored to determine if the user is inactive. The second issue is the fact that the duration of staying at the particular level may be caused by the external factors, not related to the gameplay itself (for instance, the user did not have the time for playing and had a long pause while remaining at the same level – this will significantly increase the value of the parameter).

Another problem is the size of the analyzed data. Though the used data set is of limited size, the daily updates (when more users are joining the game) may cause the set becoming large and difficult to process in the Real Time or at least in the time conditions needed. The structures created for the analysis are not only CPU demanding, but also memory hungry. This problem must be investigated in the future., either by optimizing the software (for instance, by exchanging Python with C++) or extending the processing platforms.

VII. CONCLUSIONS

The following paper presented the framework for the analysis of game players' data to determine their retention. The obtained results allow for determining, which users are the most interesting regarding financial potential for the developer. On one hand, the users staying long in the game are attractive because they are presented multiple commercials. On the other hand. The players spending actual money for in-game assets are the source of the direct income. In both cases such users are active within the gameplay and construct relatively large number of buildings and graduate between he levels faster. To confirm these statements on the larger number of users, a new data set would have to be constructed.

The system is adjusted to the specific game, but can easily be extended to other titles or genres, assuming that it will be level- (or board-) based game with content easy to aggregate and represent separate users. One of the challenges is the time required for creating the dataset (because of changing statuses of the players' activity it must always be created from scratch). Even though such operations are executed in the cloud, the presented algorithms pose a challenge. Therefore optimization of the software efficiency should be considered.

The future research should include application of additional algorithms for the feature extraction and distinguishing the most devoted players from all others. This includes deep learning, currently becoming a standard, though the successful implementation would require larger data sets. Also, due to the fact that the set of users is dynamically changing, a new data set would have to be prepared for knowledge extraction.

REFERENCES

[1] Y.-D. Shi, H.-X. Feng, J. Liu, Y.-L. Ma, Q.-Xiao, "Mobile online game experience and community interaction: Mechanisms affecting user satisfaction," Acta Psychologica, Vol. 251, 2024, 104591, ISSN 0001-6918, https://doi.org/10.1016/j.actpsy.2024.104591.

[2] A. Khoziasheva, "Applying user-centred techniques and expert feedback to refine an AI-based app for addressing mobile gaming addiction in adolescents," Dialogues in Health, Vol. 6, 2025, 100220, ISSN 2772-6533, https://doi.org/10.1016/j.dialog.2025.100220 .

[3] P. Bilski, R. Łabędzki, and A. Bilski, „Prediction of the mobile game players' payments-related retention from the Big Data perspective," International Journal of Electronics and Telecommunications, 2024, Vol. 70, No. 4, pp. 1081-1087, https://doi.org/10.24425/ijet.2024.152510.

[4] K. Rothmeier, N. Pflanzl, J. A. Hüllmann, and M. Preuss, "Prediction of Player Churn and Disengagement Based on User Activity Data of a Freemium Online Strategy Game," Univ. Münster & Univ. Leiden, 2020.

[5] A. Drachen, E. T. Lundquist, Y. Kung, P. Rao, R. Sifa, J. Runge, and D. Klabjan, "Rapid Prediction of Player Retention in Mobile Free-to-Play Games," in Proc. 12th AAAI Conf. Artif. Intell. Interactive Digital Entertainment, 2016.

[6] K. Mustac, K. Bacic, L. Skorin-Kapov, and M. Suznjevic, "Predicting Player Churn of a Free-to-Play Mobile Video Game Using Supervised Machine Learning," Appl. Sci., vol. 12, no. 2795, 2022, doi: https://doi.org/10.3390/app12062795.

[7] S.-K. Lee, S.-J. Hong, S.-I. Yang, and H. Lee, "Predicting Churn in Mobile Free-to-Play Games," ETRI & Hongik University, 2018.

[8] H. Xie, S. Devlin, D. Kudenko, and P. Cowling, "Predicting Player Disengagement and First Purchase with Event-Frequency Based Data Representation," in Proc. 2015 IEEE Conf. Computational Intelligence and Games (CIG), 2015, doi: https://doi.org/10.1109/CIG.2015.7317919.

[9] Y.-J. Han, J. Moon and J. Woo, "Prediction of Churning Game Users Based on Social Activity and Churn Graph Neural Networks," IEEE Access, vol. 12, pp. 101971-101984, 2024, doi: https://doi.org/10.1109/ACCESS.2024.3429559.

[10] T. M. U. Sree, P. Sasikumar, S. Ayesha, M. S. Amzad Basha, and M. M. Sucharitha, "Predicting Player Engagement in Online Gaming: A Machine Learning Approach," in Proc. 2024 IEEE North Karnataka Subsection Flagship Int. Conf. (NKCon), 2024.

[11] M. Viljanen, A. Airola, A.M. Majanoja, and J. Heikkonen (2017). "Measuring Player Retention and Monetization Using the Mean Cumulative Function," IEEE Transactions on Games. https://doi.org/10.1109/TG.2020.2964120.

[12]] T. Allart, G. Levieux, M. Pierfitte, A. Guilloux, and S. Natkin, "Design Influence on Player Retention: A Method Based on Time-Varying Survival Analysis," Ubisoft & CNAM, 2018

[13] L. Pang, Z. Hu, and Y. Liu, "How to Retain Players through Dynamic Quality Adjustment in Video Games," in Proc. 5th Adv. Inf. Technol., Electron. Autom. Control Conf. (IAEAC), 2021.

[14] V. Chigateri, W. P. Puthran, G. Attigeri, S. Kolekar, and S. Vobugari, "Player Pattern Prediction Using Action Logs of Players," in Proc. 2nd Global Conf. Advancement in Technology (GCAT), 2021.

[15] M. Viljanen, A. Airola, T. Pahikkala and J. Heikkonen, "Modelling user retention in mobile games," 2016 IEEE Conference on Computational Intelligence and Games (CIG), Santorini, Greece, 2016, pp. 1-8, doi: https://doi.org/10.1109/CIG.2016.7860393.

[16] Qifan Yang, Zhenhua Li, Yunhao Liu, Hai Long, Yuanchao Huang, Jiaming He, Tianyin Xu, and Ennan Zhai. 2019. Mobile Gaming on Personal Computers with Direct Android Emulation. In The 25th Annual International Conference on Mobile Computing and Networking (MobiCom '19). Association for Computing Machinery, New York, NY, USA, Article 19, 1–15. https://doi.org/10.1145/3300061.3300122.

[17] M. Sužnjević, M. Miklec and S. Poštić, "Case Study of Player Behavior in a Mobile Free-to-Play Interactive Story Telling Game," IEEE Transactions on Games, vol. 15, no. 3, pp. 450-459, Sept. 2023, doi: https://doi.org/10.1109/TG.2022.3221607.

[18] C. H. Tan, K. C. Tan and A. Tay, "Dynamic Game Difficulty Scaling Using Adaptive Behavior-Based AI," IEEE Transactions on Computational Intelligence and AI in Games, vol. 3, no. 4, pp. 289-301, Dec. 2011, https://doi.org/10.1109/TCIAIG.2011.2158434.