MONETIZATION ANALYSIS OF A MOBILE APPLICATION: USER BEHAVIOR AND RETENTION FORECASTING

Hanna Trukshanina, Uladzislau Bandarenka, Alevtina Gourinovitch, Associate Professor
EPAM School of Digital Competencies, Savanorių pr. 28, Vilnius

Abstract

The study investigates user monetization behavior and retention dynamics in a mobile game installed between September 21–28, 2023. We analyze platform-specific differences in revenue and user engagement, verify statistical hypotheses, and forecast 30-day retention using a hyperbolic model.

Keywords*: mobile analytics, monetization, retention, cohort analysis, user segmentation.*

Introduction

The mobile app industry is experiencing rapid growth with fierce competition for user attention and revenue. In this dynamic context, understanding how users behave early after app installation is critical to making informed product, marketing, and monetization decisions. The relevance of this study lies in the need for accurate data-driven strategies to retain users and optimize revenue generation during the critical onboarding and engagement stages.

The object of this study is a mobile game for which anonymized user data was collected for the install period from September 21 to September 28, 2023. The dataset includes advertising revenue events and install logs categorized by platform (Android and iOS).

The primary objective of this study is to analyze early user behavior and monetization patterns, and predict future user engagement. To achieve this goal, the following objectives were set:

- Conduct exploratory data analysis (EDA) to study distributions, detect outliers, and identify behavioral trends.
- Calculate and visualize key performance metrics such as Daily Active Users (DAU), Retention, Revenue, ARPU
- Apply clustering algorithms to identify user segments with different behavioral patterns.
- Develop a predictive model for 30-day user retention using hyperbolic regression.

The research methodology includes the use of cohort analysis for temporal segmentation of users and unsupervised learning methods (KMeans, GMM, HDBSCAN, Autoencoder + KMeans) for clustering. Retention prediction is based on a nonlinear approximation approach using hyperbolic decay models and least squares optimization. Statistical metrics such as RMSE and R^2 were used to evaluate the accuracy of the model.

This integrated approach allows for both descriptive and predictive analytics, allowing the study to support not only retrospective assessment but also forward-looking strategic planning. The results of this study offer actionable insights into monetization strategies and user lifecycle optimization, particularly through platform-specific targeting and segmentation.

1. Descriptive Data Analysis

1.1 Dataset Overview

The dataset used in this study includes anonymized information about mobile game users who installed the app from September 21 to September 28, 2023**.** It is divided into two key tables:

1. Advertising Revenue Table**:**
    - install_date: date of installation
    - event_date: date of revenue event
    - event_revenue: revenue generated from the ad
    - platform: Android or iOS
    - user_id: unique identifier
2. Installs Table**:**
    - install_date
    - installs: number of installs per day
    - platform

The dataset reflects event-based ad monetization rather than in-app purchases. The retention, revenue, and engagement metrics were calculated based on interactions captured within these tables. The platform imbalance is notable: Android users make up ~80–90% of the data, which influenced the interpretation of platform-specific trends.

Cohorts were defined as groups of users who installed the application on the same date and platform. For each cohort, the total number of installs was computed.

Findings:

- Android cohorts ranged from 1,000 to 1,800 users.
- iOS cohorts were much smaller and stable (~200 installs daily).
- Due to this imbalance, Android data was more statistically reliable for deeper analysis.

This discrepancy supports treating Android and iOS users as separate analytical segments to avoid skewed interpretation.

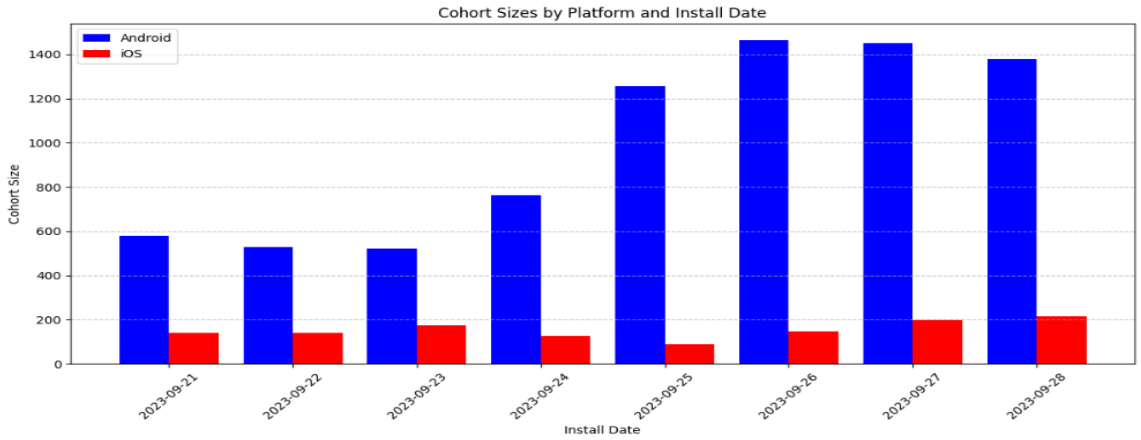

Fig 1. Cohort Sizes by Platform and Install Date
Source: made by Hanna Trukshanina

1.2 Key Metrics Analysis

In order to understand user engagement and monetization efficiency, we calculated and analyzed the key performance indicators (KPIs) over the observation period (Day 0 to Day 9) for both Android and iOS platforms. These metrics include

- Daily Active Users (DAU)
- Revenue, Average Revenue Per User (ARPU)
- Retention

Daily Active Users (DAU) was computed as the number of unique users active on each day after installation. Both platforms exhibited a sharp decline in DAU after Day 1. Android showed a smoother decline curve due to a larger user base, while iOS displayed more variability, attributable to small cohort sizes.

- Android DAU peaked at ~1,300–1,500 users on Day 1 and dropped to ~110–120 users by Day 9.
- iOS DAU dropped from ~170 on Day 1 to fewer than 20 by Day 9.

This indicates a steep early churn rate and highlights the importance of retention mechanisms within the first few days.

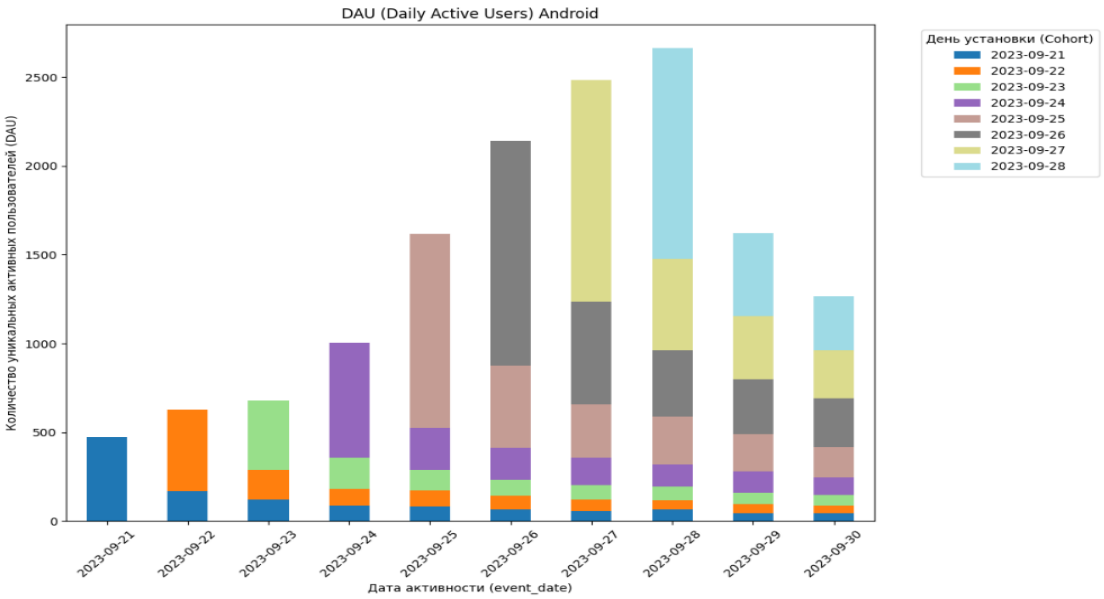

Fig 2. DAU Android

Source: made by Hanna Trukshanina

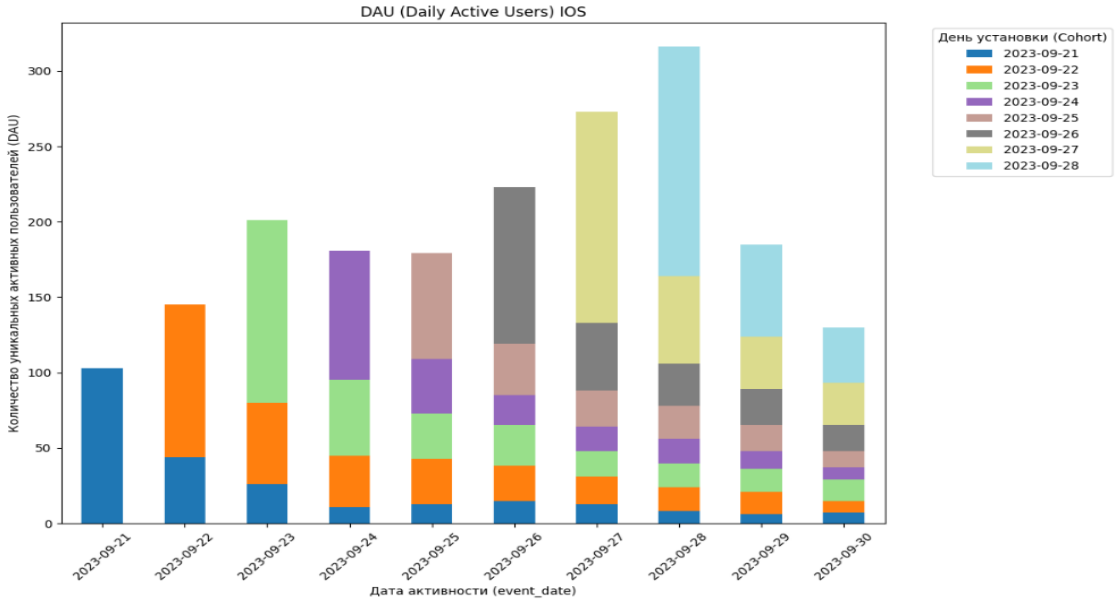

Fig 3. DAU IOS

Source: made by Hanna Trukshanina

ARPU (Average Revenue Per User) is a key indicator reflecting the average income generated by a single active user. It is calculated as the ratio of total revenue generated on a given day to the number of unique daily active users (DAU):

$$\text{ARPU} = \frac{\text{Total Revenue}}{\text{Total Users}}$$

This indicator is crucial for evaluating monetization efficiency at the user level and enables meaningful comparisons between platforms, cohorts, or time periods. It also serves as a guiding metric when testing monetization strategies, such as ad placement or in-app purchase mechanics.

To assess overall monetization efficiency, we calculated the average revenue per user over the entire nine-day period for each platform.

Results:

- Android: $1.10 per user over 9 days
- iOS: $0.94 per user over 9 days

Android demonstrates a higher total ARPU compared to iOS, which may be attributed to higher user activity or more favorable monetization conditions on the platform.

Tracking ARPU on a daily basis provides insight into how users interact with monetized features over the course of their lifecycle and helps identify the most profitable engagement periods.

$$ARPU = \frac{\text{Revenue}}{\text{DAU}}$$

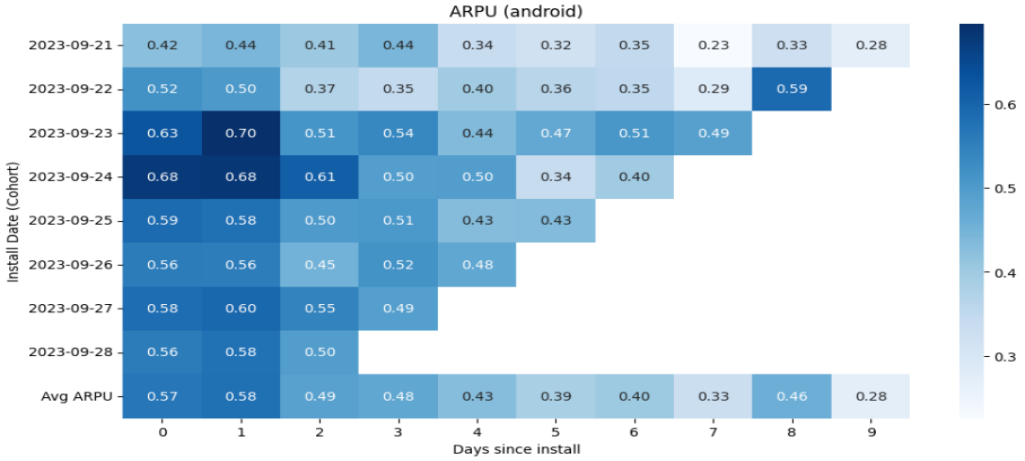

Fig 4. ARPU Android

Source: made by Hanna Trukshanina

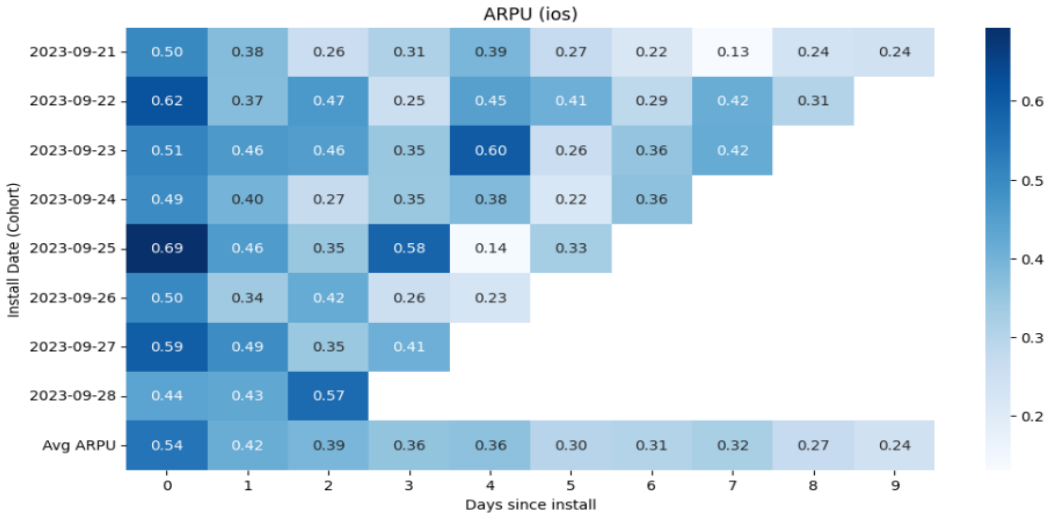

Fig 5. ARPU IOS

Source: made by Hanna Trukshanina

Analyzing daily ARPU dynamics gives insight into user behavior across the early lifecycle:

- Android: ARPU remained stable in the first 3–4 days (~$0.60–0.76), gradually declining to ~$0.35 by Day 9.
- iOS: ARPU started at ~$0.45, peaked at ~$1.30 on one day (due to an outlier), then fluctuated below $0.30 in the late period.

Key Observations:

- The stability of ARPU in the first days indicates strong monetization potential during the initial user engagement phase.

- The decline after Day 4 suggests that monetization efforts should be intensified or diversified for users who remain active in the later stages.
- Platform comparison shows that Android maintains a higher ARPU overall, likely due to greater activity and lower variance. Meanwhile, iOS ARPU is more volatile and sensitive to individual high-revenue users.

Despite its importance, ARPU can be unstable and misleading in some conditions:

- Small cohort sizes (e.g., iOS) can cause single high-revenue users to distort the average.
- Rare but large payments can spike daily ARPU without reflecting the typical behavior of the broader user base.
- Inconsistent user activity across days can cause noticeable ARPU fluctuations, making it harder to distinguish genuine trends.

Therefore, while ARPU is a useful performance metric, it should be interpreted in context and preferably supported by additional indicators such as standard deviation, median values, or user-level segmentation.

Retention Rate is a key metric that reflects the percentage of users who return to the application after a certain number of days following installation. It serves as an indicator of user loyalty, app quality, and long-term engagement potential.

$$\text{Revenue Retention} = \left(\frac{\text{Revenue on Day X}}{\text{Revenue on Day 0}} \right) \times 100$$

To visualize retention dynamics, line charts were used. Each line represents a cohort of users grouped by install date, showing how many users remained active from Day 0 through Day 9. This approach allows us to compare behavioral patterns across cohorts and detect general trends or deviations.

On Android, user retention starts high at approximately 84.2% on Day 0, drops to around 44.5% by Day 1, and continues to decline gradually to 7.6% by Day 9. The curve flattens after the third or fourth day, suggesting that users who stay past this threshold are more likely to remain engaged longer. The retention curves of different Android cohorts are tightly grouped and follow similar trajectories, indicating consistent user behavior and stability across cohorts.

On iOS, retention starts lower, with about 71.5% of users returning on Day 0. By Day 1, retention falls to approximately 41.3%, and by Day 9, only about 5.0% of the cohort remains active. Unlike Android, iOS retention curves are more dispersed, reflecting the smaller cohort sizes and the influence of random variation on user activity patterns.

The general shape of the retention curves is typical for casual mobile games: a sharp drop in the first few days followed by a plateau. The lack of significant divergence between different cohorts suggests that external factors (e.g., marketing campaigns or app updates) did not majorly impact retention during the observed period.

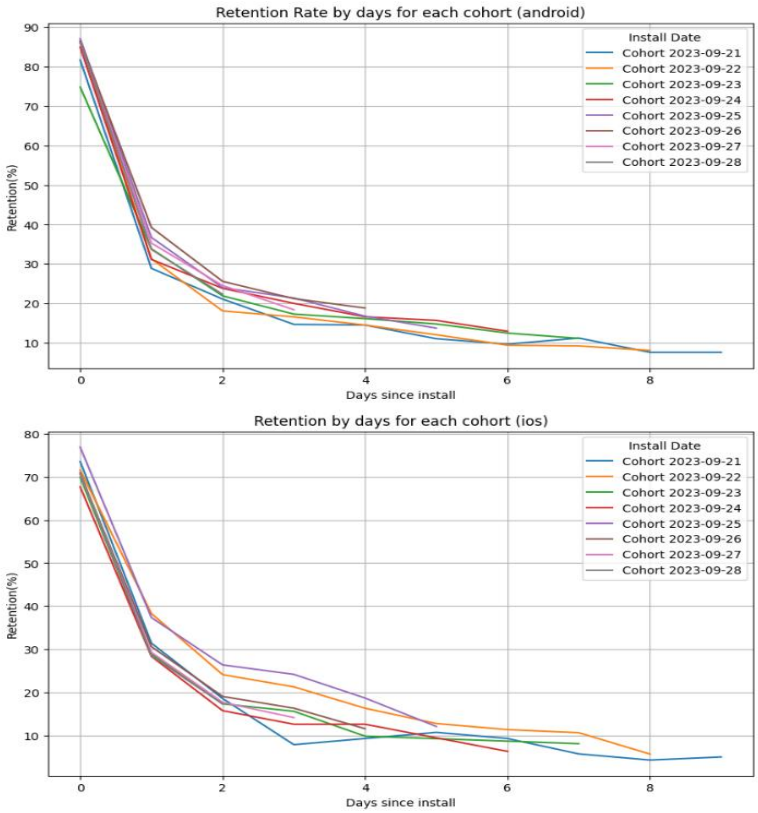

Fig 6. Retetion Rate by days for caohort Android and IOS
Source: made by Hanna Trukshanina

This analysis confirms that early engagement is critical. The majority of user churn occurs within the first 2–3 days, emphasizing the need to optimize onboarding experiences and early gameplay. Users who remain active after Day 4 represent a core audience with higher retention potential and should be the focus of personalized engagement and monetization efforts.

2. Predictive Analysis

2.1 Forecasting User Retention on Day 30

Retention is one of the most important long-term performance indicators for digital products, including mobile apps, web platforms, e-commerce, and even offline services. Across domains, the retention curve typically follows a consistent shape: an initial steep drop in the first days or weeks, followed by a slower decline, and eventually a stabilization period where the curve reaches a plateau. This shape is often referred to as a **"forgetting curve"**, reflecting the rapid loss of users who do not complete onboarding, and the long-term persistence of a loyal core audience.

Understanding and forecasting retention beyond the observed period is critical for estimating **LTV (Lifetime Value)**, making marketing decisions, and projecting revenue. However, in many cases we only have observed values for selected days (e.g., Day 1, Day 7, Day 9), while intermediate or future values such as Day 14 or Day 30 must be estimated. This is where mathematical approximation models become useful.

In this study, we used **curve fitting techniques** to approximate the observed retention trend with a mathematical function. The goal was to select a functional form that accurately captures the shape of the curve and minimizes the error between predicted and actual values. We tested several **hyperbolic models**, starting with simple inverse functions (e.g., A/X) and gradually adding parameters to improve flexibility.

Each model took the general form:

$$\text{retention} = \frac{A}{X+B} + C$$

Where:

- A is a scaling factor determining the starting value (initial retention),

- B adjusts horizontal displacement,
- C controls how quickly the retention curve decays,
- X is the number of days since installation.

To identify the most suitable model, we used the **least squares method** to minimize the squared differences between actual and predicted retention values. The performance of each model was evaluated using two key metrics:

- RMSE (Root Mean Square Error), which quantifies the average prediction error.
- R^2 (coefficient of determination), which indicates the proportion of variance explained by the model.

We trained the model using the first available cohort (September 21, 2023), as it covered the full nine-day observation window. This provided the most complete data for curve fitting. The initial parameters were set to A=80, B=1, C=7, and optimized through iterative minimization.

For **Android**, the optimal parameters were:

$$\text{Retention Rate (X)} = \frac{37.93}{(X + 0.49)} + 4.48$$

- Predicted Day-30 Retention**:** 5.72%
- RMSE**:** 1.09
- R^2**:** 1.00

The fitted curve accurately replicates the retention drop in the first few days and its stabilization around Day 10. The graphical results show high precision and validate the hyperbolic model as a reliable predictor of long-term user behavior on Android.

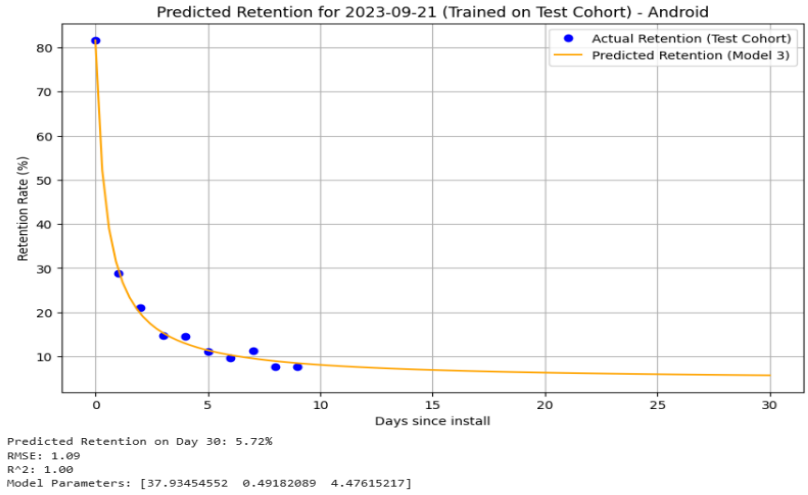

Fig 7. Prediction Retention by Cohrt Android
Source: made by Hanna Trukshanina

For **iOS**, the same model form was used, but with different coefficients:

$$Retention(X) = \frac{86.78}{X + 1.19} - 1.22$$

- Predicted Day-30 Retention: 1.56%
- RMSE: 1.32
- R^2: 1.00

The iOS retention curve also displays a sharp decline early on, followed by gradual stabilization, but at a significantly lower level compared to Android. This suggests differences in user behavior, engagement depth, or the app's appeal across platforms.

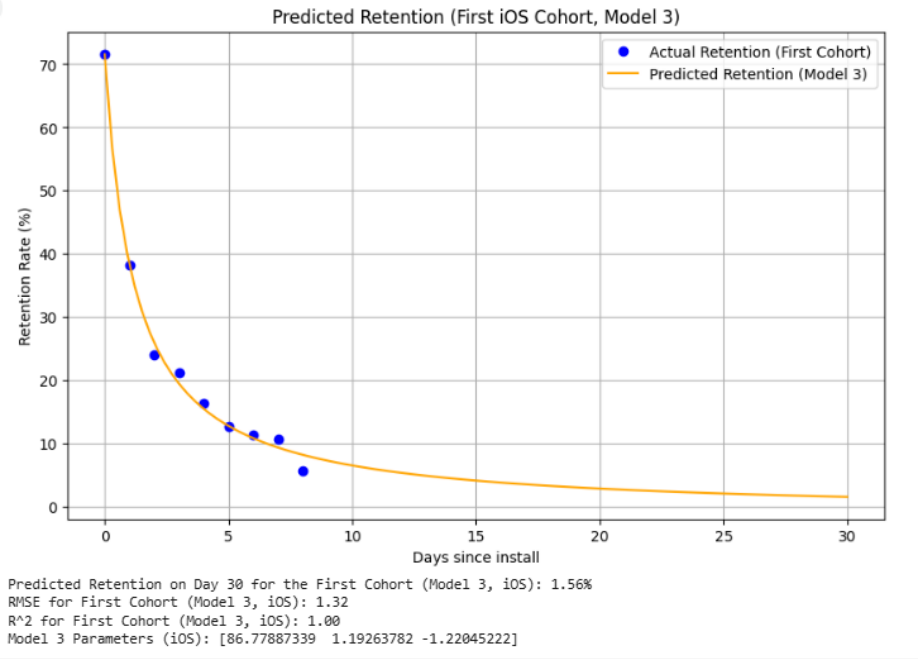

Fig 8. Prediction Retention by Cohrt IOS
Source: made by Hanna Trukshanina

These results confirm that the chosen hyperbolic model offers a precise and interpretable approximation of retention trends. Using the longest available cohort for model training ensures high prediction accuracy due to the broader and more complete dataset.

3. ML in Data Analysis

3.1. Clustering Approach

The main objective of this study is to segment users based on behavioral and revenue metrics. To achieve this, we used **unsupervised machine learning algorithms** — particularly clustering methods — which allow grouping users with similar patterns **without prior labels**.

Since our dataset includes features with different scales and variances (e.g., revenue, frequency, variability), we selected a **diverse set of clustering models** to compare how well they capture user segments in different behavioral dimensions.

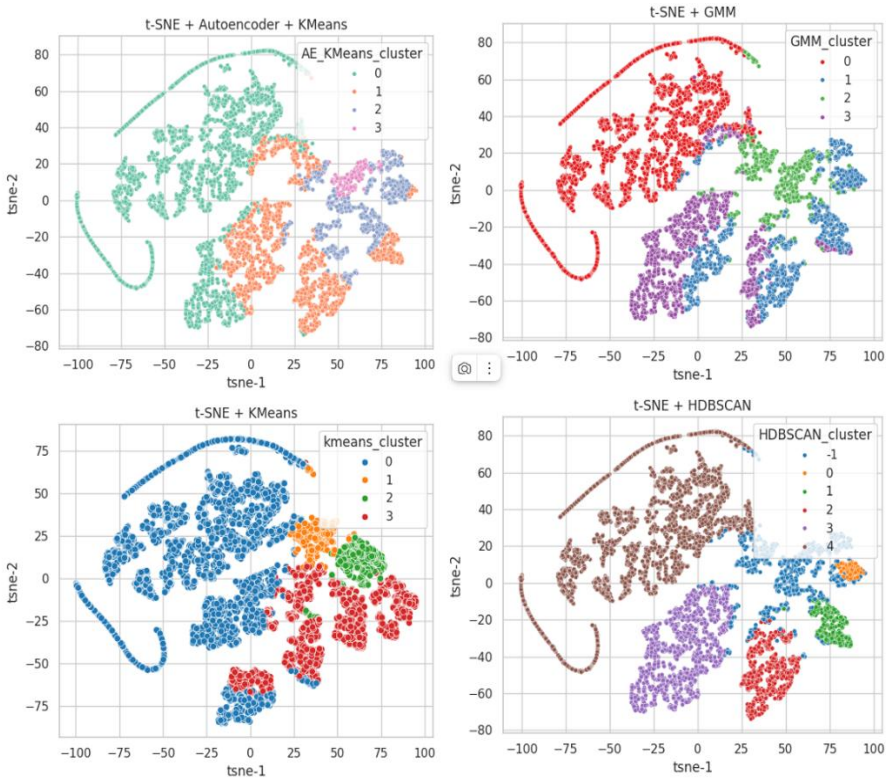

Fig 9. User segmentation by clustering models
Source: made by Uladzislau Bandarenka

3.2. Models Used and Rationale

3.1. 1 Autoencoder + KMeans

A two-step hybrid method combining:

Autoencoder: A neural network trained to compress input data into a latent space and reconstruct it. It reduces noise and dimensionality.

KMeans: A classic partition-based clustering algorithm applied to the latent representations.

Why use it?

Autoencoders capture non-linear relationships and learn a compressed representation of user behavior.

Applying KMeans in this space can yield tighter, more meaningful clusters than in raw feature space.

Benefits:

Handles non-linearity in the data.

Helps denoise and smooth feature irregularities.

Good for high-dimensional or sparse data.

Limitations:

Requires neural network tuning.

Slightly more computationally expensive.

3.2.2. KMeans Clustering

A simple yet powerful clustering algorithm that partitions data into kkk clusters by minimizing the within-cluster variance (inertia):

- Provides a baseline model.
- Efficient and scales well with large datasets.
- Easy to interpret and visualize.

Benefits:

Fast and deterministic.

Works well with spherical, evenly sized clusters.

Limitations:

Assumes equal-size clusters.
Not robust to outliers.
Sensitive to initial centroids.

3.2.3. HDBSCAN (Hierarchical Density-Based Spatial Clustering of Applications with Noise)

A density-based clustering algorithm that can find clusters of varying shapes and sizes and mark noise (outliers):

- Unlike KMeans, HDBSCAN does not require the number of clusters as input.
- Excellent at identifying anomalies and rare patterns.
- Useful for irregular, real-world data.

Benefits:

Handles noise and outliers naturally.
Detects non-convex cluster shapes.
Doesn't assume cluster size/shape.

Limitations:

Requires setting min_cluster_size, which may affect granularity.
Can produce many small clusters depending on data density.

3.2.4. Gaussian Mixture Model (GMM)

What is it?

A probabilistic model assuming that data is generated from a mixture of several Gaussian distributions, each representing a cluster.

Why use it?

Unlike KMeans (which assigns hard cluster labels), GMM provides soft assignments — the probability that a user belongs to each cluster.
Good for modeling elliptical and overlapping clusters.

Benefits:

Captures overlapping clusters.
More statistically grounded.
Each cluster is defined by its mean and covariance.

Limitations:

Assumes the data can be approximated by Gaussian distributions.
Can be sensitive to initialization and outliers.

Table 1

Summary of Methodological Reasoning

Model	Captures Non-Linearity	Robust to Outliers	Learns Probabilistic Membership	Needs Predefined k	Suitable for High-Dim
Autoencoder + KMeans	Yes	Partial	No	Yes	Yes
KMeans	No	No	No	Yes	Limited
HDBSCAN	Partial	Yes	No	No	Yes
GMM	Partial	No	Yes	Yes	Yes

3.2.4. Multiple Models

Using multiple models allows for a more robust analysis. Each model captures different aspects of user behavior:

- Autoencoder + KMeans is ideal for learning complex patterns in compressed spaces.
- KMeans is a reliable baseline that is easy to scale and interpret.
- HDBSCAN is essential for identifying irregular user types or anomalies.
- GMM is valuable for soft clustering where users might not belong to a single clear group.

By comparing results across models, we identify stable user groups that appear in multiple clustering, providing higher confidence in those segments.

Conclusion

This study aimed to analyze user behavior and monetization performance in a mobile application using key product metrics, predictive modeling, and unsupervised learning techniques. The descriptive analysis of metrics such as DAU, revenue, ARPU, and retention revealed platform-specific dynamics and emphasized the importance of early user engagement. Android users showed higher and more stable ARPU and retention values, while iOS users demonstrated greater variability, likely due to smaller cohort sizes and sensitivity to outliers.

To forecast long-term user engagement, it was implemented a hyperbolic approximation model for retention. The selected model showed excellent accuracy, with R^2 values close to 1.00 and low RMSE for both Android and iOS. The predicted retention on Day 30 was 5.72% for Android and 1.56% for iOS, aligning with typical benchmarks for casual mobile apps. These results support the application of retention modeling in LTV forecasting and strategic planning.

In addition, it was applied machine learning techniques (KMeans, GMM, HDBSCAN, Autoencoder + KMeans) to perform behavioral segmentation. The identified clusters helped uncover different user groups based on revenue, activity, and engagement metrics. Practical use cases derived from this segmentation include feedback prioritization, targeted promotions, churn prevention, onboarding optimization, and personalized experiences.

This study was applied a range of unsupervised machine learning clustering methods to segment users based on their behavioral and revenue characteristics. The goal was not only to uncover distinct user groups but also to evaluate the strengths and trade-offs of each clustering algorithm.

By combining classic techniques like KMeans and Gaussian Mixture Models (GMM) with more advanced methods such as Autoencoder + KMeans and HDBSCAN, it is were able to capture diverse patterns in the data: from high-revenue VIPs to rare or irregular users and even outliers. Each model contributed unique insights:

- Autoencoder + KMeans revealed non-linear structures in high-dimensional behavior,
- KMeans provided a fast and interpretable baseline,
- HDBSCAN uncovered noise and non-standard user types,
- GMM offered soft clustering to better understand user overlap.

This multi-model approach enabled us to cross-validate cluster stability and derive more reliable and actionable user segments. These insights can now be translated into real-world strategies such as targeted promotions, churn prevention, onboarding improvements, and personalized experiences.

Ultimately, this analysis highlights how combining machine learning models with domain knowledge empowers more data-driven decisions in user engagement and monetization strategy.

Overall, the integration of exploratory analysis, predictive modeling, and clustering allows for a more comprehensive understanding of user behavior. These insights can support data-driven decision-making in product development, user retention strategies, and monetization optimization.

References

1. Retention Rates for Mobile Apps by Industry: https://www.plotline.so/blog/retention-rates-mobile-apps-by-industry
2. Mobile App Monetization Models, ARPU & Retention Stats That Matte:https://www.winsavvy.com/mobile-app-monetization-models-arpu-retention-stats-that-matter/
3. Charles M. Grinstead. Introduction to Probability / Swarthmore College. – 2009. – 520 p.
4. Dimitri P. Bertsekas. Introduction to Probability, lecture notes, Course 6.041-6.431/ Dimitri P. Bertsekas, John N. Tsitsiklis. – M.I.T, 2000. – 284 p.
5. Sebastian Raschka. Python Machine Learning: Unlock deeper insights into Machine Leaning with this vital guide to cutting-edge predictive analytics – Packt Publishing – 2015. –454 p.