Analyzing Player Retention and Monetization Strategies in Mobile Games

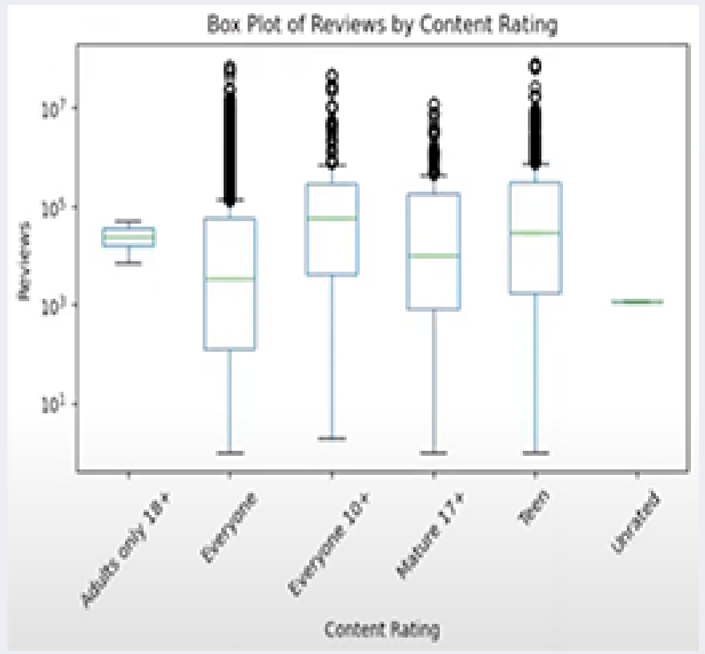
Parker Brandt, Jai Shourie, Andrew Chu, Ivan Ng

Abstract

This research takes a deep dive into player retention and monetization strategies within the mobile gaming industry. With the mobile gaming industry expanding rapidly and becoming more competitive, game developers must understand how to keep players engaged. We used data science to investigate how players behave, what keeps them returning, and how games may generate more revenue. By studying how people play games and what they think, we want to identify evident actions that game developers can take to make their games more enjoyable and lucrative.

Introduction

The mobile gaming industry is thriving, providing endless entertainment to consumers and opportunities for game developers. However, keeping players engaged and convincing them to spend money may be difficult. Our initiative dives into these issues, attempting to determine what makes gamers remain and how to leverage that into profit. We're analyzing game statistics, player feedback, and how people spend money on games to identify patterns and valuable behaviors. This isn't just generating money; it's about creating games that people like playing. We believe that by understanding the player users, we can create mobile games more enjoyable and successful. Our main goals are to discover what keeps players coming back, to determine which methods of making money work best, and to recommend how to create games that are both entertaining and profitable.



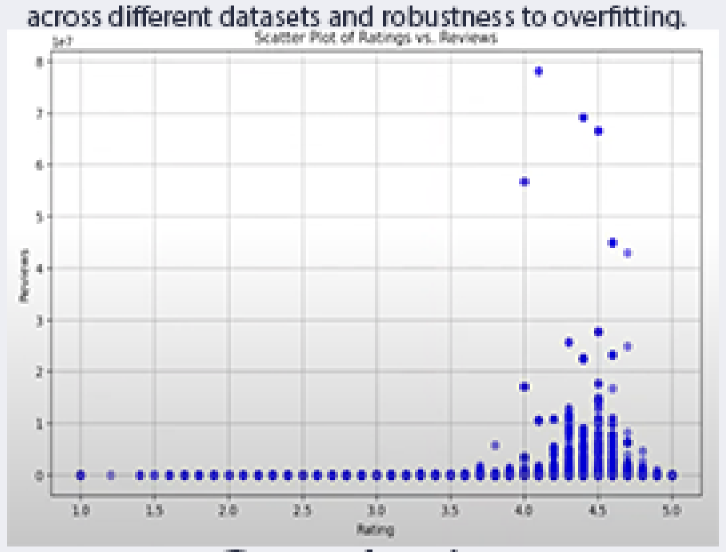
Objectives

Data Acquisition and Data Preparation: Obtain dataset related to the research of player retention and monetization strategies in mobile games. Goal is to clean, structure, and preprocess the data to ensure consistency and reliability for analysis, providing the groundwork for understanding player behaviors and game monetization strategies

EDA: Through exploratory data analysis uncover the trends and connections in the dataset, specially focusing on the impact of gaming elements, user engagement, and user retention and expenditure

Model Selection: Find a predictive model that will be chosen based on its ability to properly forecast user engagement and monetization efforts

Model Training and Evaluation: Metrics like R-squared values, MSE (Mean Squared Error), and MAE (Mean Absolute Error) can be used to evaluate the efficacy of the trained model. Utilized cross-validation method to confirm the model's prediction, ensuring its applicability across different datasets and robustness to overfitting.



Conclusion

Our project was only partly successful in achieving its goals. We aimed to predict game installs using factors like developer, app-store rating, genre, and price. Although we built a functional model, its accuracy was lower than desired. The complexity of predicting game success lies in the many qualitative and immeasurable factors involved, such as player enjoyment, which requires deep psychological analysis to gauge accurately.

Despite these challenges, our models and data-driven approach achieved an MSE of 0.2, enabling developers and marketers to better tailor their games to target audiences. Future improvements could focus on analyzing visual cues in games to enhance micro-transactions and conducting in-depth studies of industry leaders like Tencent Games to glean insights that could benefit other developers.

Methodology

Acquisition

The dataset was obtained from a CSV file named 'googleplaystore.csv'.
The pandas library in Python was used to read the CSV file and load the data into a DataFrame.

Preparation

Column Removal: Irrelevant columns such as 'Current Ver', 'Android Ver', 'Size', 'Genres', and 'Last Updated' were dropped from the dataset as they were not necessary for the project's objectives.
Handling Missing Values: Rows containing missing values in relevant columns such as 'Rating', 'Content Rating', and 'Type' were removed from the dataset to ensure data integrity for subsequent analysis.
Type Conversion: The 'Type' column was converted to numerical values, where 'Free' was represented as 0 and 'Paid' as 1, facilitating future analysis and modeling tasks.
Categorical Data Encoding: The 'Category', 'Content Rating', and 'Installs' columns were converted to categorical data types for efficient processing. One-hot encoding was applied to categorical columns to convert them into a numerical format suitable for machine learning algorithms.

Machine Learning Modeling:

Random Forest Classification: A Random Forest Classifier model was trained using the preprocessed data. The model's accuracy was evaluated on the testing set, achieving a certain accuracy score. Additionally, a classification report was generated to provide detailed performance metrics such as precision, recall, and F1-score.
Random Forest Regression: Utilizing K-Fold cross-validation, a Random Forest Regressor model was trained and evaluated on multiple folds of the dataset. Key regression metrics including Mean Squared Error (MSE), Mean Absolute Error (MAE), and R^2 Score were calculated and averaged across the folds to assess model performance.
Hyperparameter Tuning: Grid Search CV was employed to fine-tune the hyperparameters of the Random Forest Regressor model. The best parameters and corresponding score (MSE) were identified to optimize model performance.
K-Nearest Neighbors (KNN) Classification: A KNN Classifier model was trained and evaluated on the dataset. Model accuracy was assessed on the testing set, and a classification report was generated to provide detailed performance metrics.
Support Vector Regression (SVR): A Support Vector Regression (SVR) model was trained using the preprocessed data. Regression metrics including Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and R^2 Score were computed to evaluate model performance.

Results and Evaluation

RandomForestClassifier
Accuracy: 60.62%
Precision & Recall: Precision similar to accuracy; recall (macro average) low at 0.46.
Analysis: The model's modest accuracy could result from overfitting due to a small test size (0.2). The model is likely unsuitable for predicting the 'installs' column without adjustments, such as changing the test size or implementing cross-validation.

RandomForestRegressor with K-Fold
Metrics: MSE: 0.2428, MAE: 0.2940, R^2: 0.0883.
Interpretation: While MSE and MAE suggest close predictions to actual values, the low R^2 indicates poor variance explanation, suggesting the model might not effectively capture the underlying patterns.
Grid Search CV
Optimized Parameters: 200 trees, max depth of 20, min samples split of 10, min samples leaf of 4.
Best Score: -0.2292 (negative MSE).
Comment: Optimization via Grid Search CV improved the model's MSE performance.

K-Nearest Neighbors
Accuracy: 39.97%
Overall Performance: Low precision, recall, and F1-score across classes, indicating poor prediction capability. Significant tuning required; however, RandomForestClassifier outperforms KNN by about 50%.

Support Vector Regression
Metrics: MSE: 0.2198, RMSE: 0.4688, MAE: 0.3232, R^2: 0.1084.
Assessment: SVR shows average performance with potential for improvement through parameter tuning, feature engineering, or alternative kernel functions.

Impacts on Game Industry
Forecasting Installs: These models, especially SVR, can help game companies forecast installs to refine launch strategies and marketing budgets, leading to optimized resource allocation and enhanced profitability.
Strategic Development: Data-driven decisions from these models can guide the development of new games aligned with market demands, potentially increasing a game's success rate.
Competitive Strategies: Insights from predictive models enable companies to strategically position their games against competitors, optimizing release timing and market focus.