ISTANBUL TECHNICAL UNIVERSITY

MANAGEMENT FACULTY

MANAGEMENT ENGINEERING DEPARTMENT

ISL 439E Introduction to Machine Learning in Business Applications

(CRN: 22947)

Mustafa Ercengiz- 070210255

INSTRUCTOR: PROF. DR. Tolga Kaya

Table of contents

1. Introduction – Research Problem, Aim, and Scope

Churn is a significant issue in the gaming world, as an enormous percentage of new players leave almost as soon as they have installed. In mobile gaming, it's common to lose close to 90% of new users within the first week. This project is for player churn prediction for a mobile crossword puzzle game. Data.description has also defined churn.as an inactive user who never plays the game after the first week. If the game developers can precisely identify the players who will churn, they can focus retention efforts (e.g., marketing campaigns.or rewards. within the game. to drive player engagement. and lifetime value.

Objective: The purpose of this project is to build a machine learning model that will predict whether or not a new player will churn after the first week based on first-week play data. The model will assist the team in knowing the cause of the majority of churn and enable the team to act early on players who are at risk.

2. Data Overview

The 49,999 players' training data includes all the 23 features with Churn labels to train the model. The test data includes the same features without Churn labels for final evaluation(As shown in fig 2.1). We verified the data for missing values or any anomalies prior to analysis.

```python
import pandas as pd
import matplotlib.pyplot as plt
# Load datasets
train_df = pd.read_excel('train_data.xlsx')
test_df = pd.read_excel('test_data.xlsx')

# Basic info
print("Train shape:", train_df.shape)
print("Test shape:", test_df.shape)
train_df.head(5)
```

```
Train shape: (49999, 24)
Test shape: (5931, 24)
```

Figure 2.1 - Data Shapes

```
country                          0
device_brand                  1254
device_model                  1254
re_install                       0
os                               0
attribution_event_timestamp      0
ecpi                           528
lang                             0
current_gold                     0
totalPowerUp                     0
bonus_cnt                        0
duration                         0
hint1_cnt                        0
hint2_cnt                        0
hint3_cnt                        0
lvl_no                           0
repeat_cnt                       0
banner_impr                      0
inter_impr                       0
rewarded_impr                    0
user_id                          0
campaignid                       0
partnerid                        0
churn                            0
dtype: int64
```

Figure 2.1 - Missing Data

-Device brand/model: ~1,254 entries (≈2.5%) have missing device_brand and device_model (likely cases where device info wasn't captured).

-ECPI: 528 entries (~1.1%) have missing ecpi (acquisition cost).

-Country: (In the test set, 1 missing country is observed; none missing in train).

3. Exploratory Data Analysis (EDA) and Visualizations

An extensive EDA performed to uncover game play behaviors and their correlation to churn. Key actions included building summary statistics and graphing distributions of features, typically dividing churned and retained players. Discussion always ensued for each ofthese graphs to provide explanations of findings. A number of interesting observations of the EDA include:

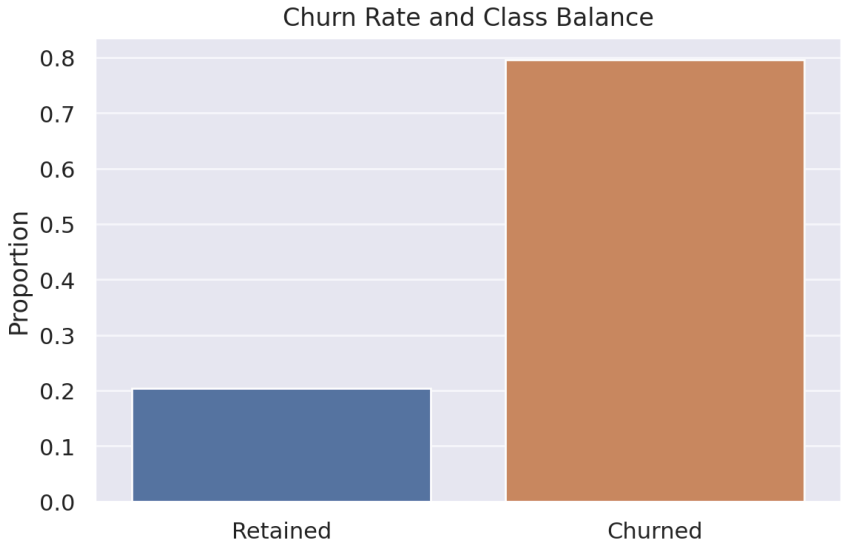


Fig 3.1 Churn Rate and Retained User proportions

Dataset Imbalance: As shown in *Fig 3.1* The rate of retention is approximately 80% so the aimed dependent variable in the dataset is not balanced

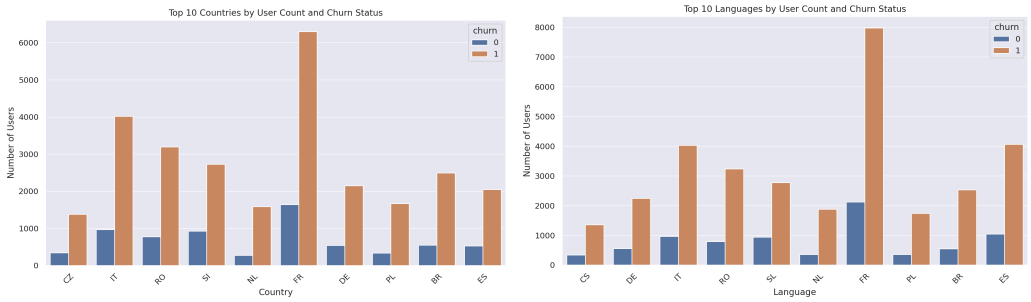


Fig 3.2.1 Churn Rate relative to Language and Country

country	Percentage	Count	Cumulative
FR	15.88	7938	15.88
IT	9.97	4985	25.85
RO	7.94	3969	33.79
SI	7.30	3650	41.09
BR	6.09	3045	47.18
DE	5.39	2694	52.57
ES	5.15	2577	57.72
PL	4.00	2002	61.72
NL	3.72	1861	65.44
CZ	3.44	1722	68.88
ID	3.35	1675	72.23
BG	3.23	1615	75.46
CA	2.31	1157	77.77
US	2.23	1116	80.00
BE	2.01	1003	82.01
AZ	1.95	976	83.96
GR	1.89	943	85.85
SK	1.67	833	87.52
PT	1.55	773	89.07
VE	1.49	745	90.56

lang	Percentage	Count	Cumulative
FR	20.18	10089	20.18
ES	10.21	5104	30.39
IT	9.98	4991	40.37
RO	8.05	4027	48.42
SL	7.43	3713	55.85
BR	6.16	3079	62.01
DE	5.61	2805	67.62
NL	4.47	2233	72.09
PL	4.19	2097	76.28
CS	3.39	1696	79.67
BU	3.36	1682	83.03
EN	3.31	1656	86.34
ID	3.24	1620	89.58
GR	1.85	926	91.43
AZ	1.75	877	93.18
SK	1.61	805	94.79
PT	1.57	786	96.36
HU	1.28	641	97.64
UK	0.55	273	98.19
RU	0.48	239	98.67

Fig 3.2.2 Country Distribution in Dataset

As shown in *Fig 3.2.1 and Fig 3.2.2* FR, IT, RO, SI and DE are countries with the most players and most the lang preferences are correlated with countries. Churn distribution is unique: some countries have significantly different churn rates.

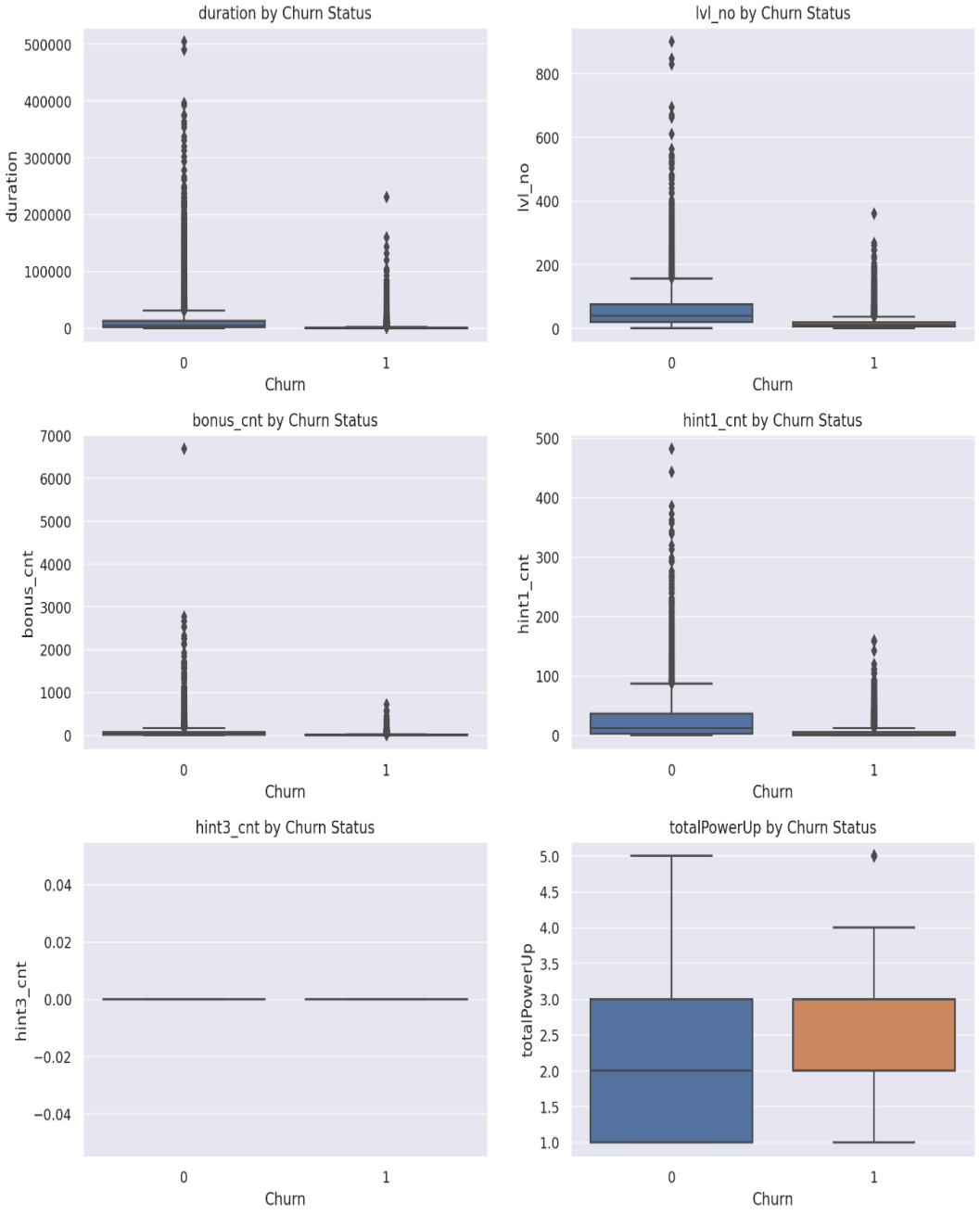


Fig 3.3 Churn Rates relative to gameplay behavior patterns

As mentioned in Fig 3.3, Churned players show decreased intensity of interaction with core gameplay features, meaning less interaction. While intensity of feature usage can be utilized as highly predictive features for churn prediction, features Hint2, Hint3, Os are empty/ineffective features will be removed further.

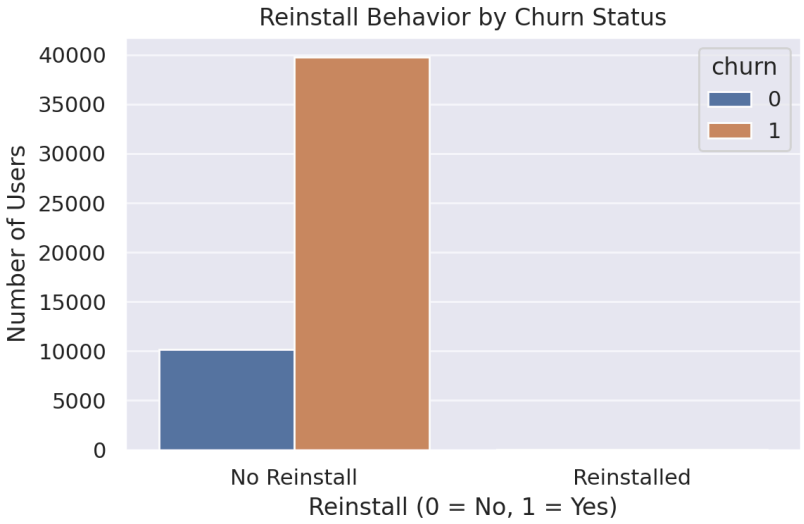


Fig 3.4 Reinstall Behavior patterns

The Fig 3.4 illustrates how churn is related to whether the user re-installed the application. A very small proportion of users re-installed the application. Of those who re-installed, most did not churn.

device_brand	Percentage	Count	Cumulative
samsung	51.84	25267	51.84
Xiaomi	17.91	8728	69.75
motorola	5.55	2703	75.30
HUAWEI	4.75	2313	80.05
OPPO	4.20	2049	84.25
HONOR	2.64	1288	86.89
realme	1.99	972	88.88
LENOVO	1.96	955	90.84
vivo	1.25	608	92.09
Google	0.86	421	92.95
TECNO	0.68	332	93.63
TCL	0.65	318	94.28
INFINIX	0.64	311	94.92
LGE	0.43	211	95.35
OnePlus	0.41	202	95.76
ZTE	0.38	183	96.14
HMD Global	0.36	174	96.50
INFINIX MOBILITY LIM	0.24	115	96.74
Blackview	0.22	107	96.96
Lenovo	0.22	106	97.18

Fig 3.5 Device Brand Distrubution

The Fig 3.5 shows that some of the brands like “samsung” and “Xiaomi” dominates the distribution

campaignid	Percentage	Count	Cumulative
25	33.13	16565	33.13
23	12.67	6335	45.80
34	5.93	2965	51.73
27	4.79	2396	56.52
33	4.52	2260	61.04
19	4.08	2040	65.12
24	3.63	1814	68.75
28	3.32	1662	72.07
30	3.08	1540	75.15
10	3.01	1507	78.16
26	2.56	1282	80.72
12	2.48	1238	83.20
13	2.24	1118	85.44
15	2.01	1006	87.45
29	1.49	745	88.94
4	0.98	488	89.92
7	0.97	486	90.89
11	0.95	477	91.84
16	0.95	475	92.79
17	0.90	451	93.69
22	0.85	423	94.54
14	0.78	391	95.32
5	0.77	384	96.09
9	0.73	363	96.82
3	0.70	350	97.52
2	0.68	338	98.20
8	0.61	307	98.81
18	0.40	201	99.21
6	0.39	196	99.60
32	0.23	114	99.83

Fig 3.6 Campaign Distribution

As shown in 3.6 after campaign ID 14 the others only cantains 5 percent of the remaining ones so similar to device brand it is a feature that highly dominated by few variables.

partnerid	Percentage	Count	Cumulative
4	76.31	38154	76.31
1	23.63	11816	99.94
2	0.06	28	100.00
3	0.00	1	100.00

Fig 3.7 Partner Distribution

As shown in 3.7 near the whole dataset having 2 partner id. Only 1 out of 49999 is 3. the 2nd and 3rd partners might one hot encoded as others

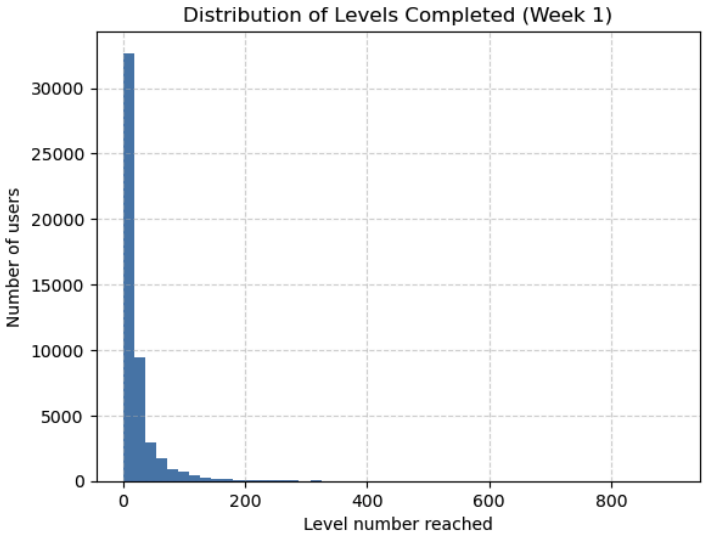


Fig 3.8 Distribution of Completed levels Levels

As shown in 3.8 Levels Completed (lvl_no): Highest level completed within a week can range from 1 to 900. Median is 13 and mean ~24, and it indicates a skew (a couple of players went very high). Most users are just at low levels, and a few reach hundreds of levels.

This is natural from a casual game – many new players quit early on, but a few get highly addicted.

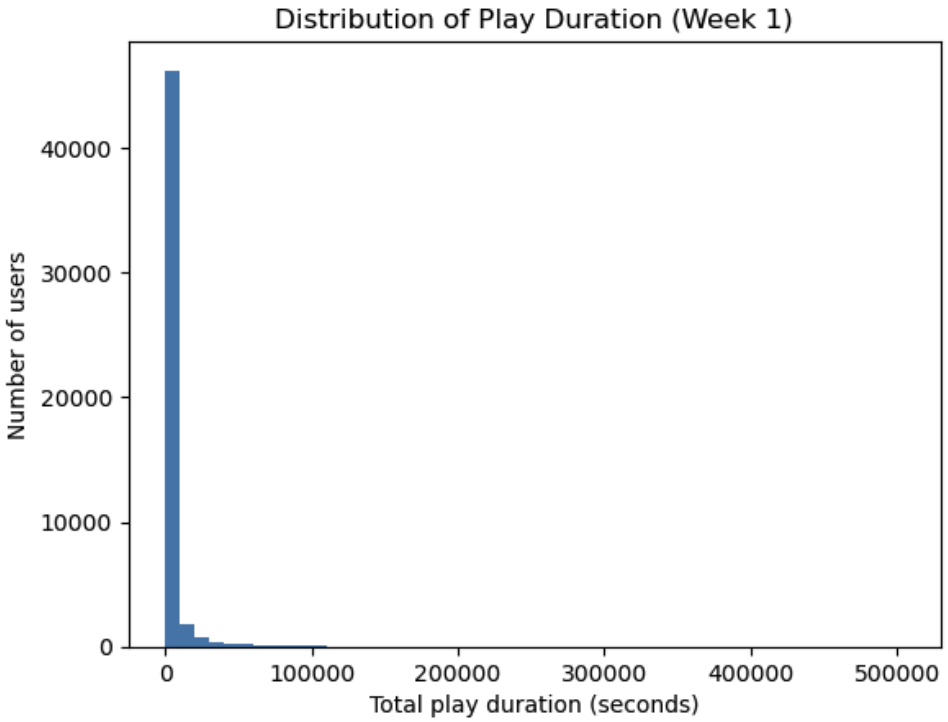


Fig 3.9 Distribution of Play Duration

As shown in fig 3.9 large spike at the low end indicates a lot of churned users early (short overall time), and this is likely the case for churners. Few users have extremely high playtime, indicating high engagement (and likely they did not churn).

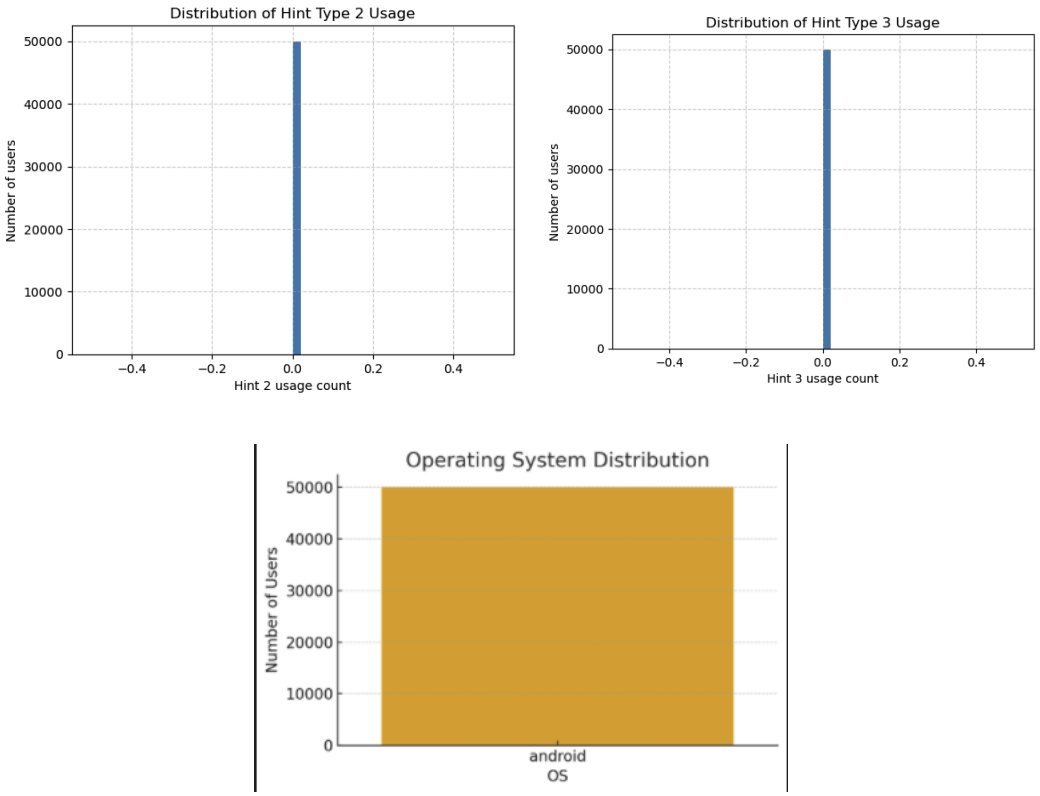


Fig 3.10 Garbage Features

Those are the features that won't have any effect on the statistical models because they only have 1 variable. Wise decision is dropping them

4. Feature Engineering

In order to understand the relative importance of time from datetime columns 3 separate columns are created. After the creation of those columns one way ANOVA test applied to check if its any significant relationship between churn and time in terms of day/week/hour.

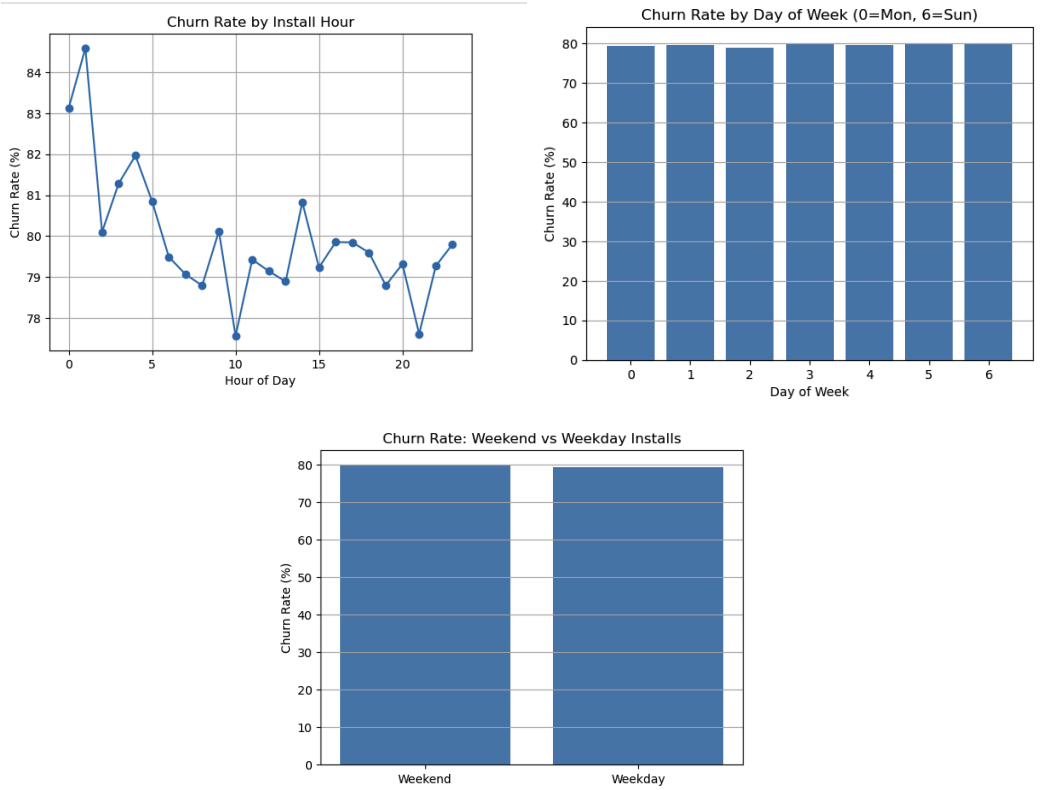


Figure 4.1

After the results of ANOVA test only the hour column has a significant relationship with being churn therefore churn varies by hour of day.

In addition to another variable created by dividing levels to duration to understand if passing levels in shorter time effects being churn or not but in the model testing it had worse result so it does not used

Level-Duration Ratio

In addition to standard changes, another feature, lvl_duration_ratio, was created to capture user progress speed. It was a ratio of levels achieved to overall play time in the initial week. The rationale for this variable was that players who moved faster through levels would be more engaged and, thus, lower churn rate.

In order to deal with possible opportunities of skewness and lesseningheteroscedasticity, a log-transformed variant of the variable of interest—log_lvl_duration_ratio—was also established by utilizing the log1p function.

We incorporated both elements into an extended modeling workflowwith a Random Forest classifier and evaluated them with 5-fold cross-validation. While the two elements were quite predictive (5th and 6th in feature importances, respectively), they did not result in any significantdifference in model performance. Cross-validated mean accuracy (81.1%), recall (83.0%), and AUC (87.2%) were not significantly worse than the baseline model.

Due to their low incremental value, they were left out of the final resulting model. They are still reported, however, as they denote the feature experimentation phase for the purpose of illustrating a systematic approach of hypothesis-driven feature engineering.

5. Missing Data and Regression Imputation

As shown in the figure 2.1 there was - ~1,254 entries (≈2.5%) have missing device_brand and device_model and 528 entries (~1.1%) have missing ecpi (acquisition cost). In order to fill the gap, Ecpi's ‘Regression Imputation’ is used. This method leverages relationships with existing features—such as campaign id, partner id, and country—to produce more contextually accurate estimates for missing values. The imputation was applied to the training and test data.

Preprocessing Pipeline

A. Data Loading and Initial Cleaning

1. Loaded training dataset from train_data.xlsx.
2. Dropped non-informative or high-cardinality columns:
    - ['os', 'hint2_cnt', 'hint3_cnt', 'user_id', 'device_model', 'repeat_cnt']

B. Time Feature Extraction

1. Created new features from attribution_event_timestamp:
    - install_hour: Hour of the day the event occurred.
    - install_dayofweek: Day of the week (0 = Monday).
    - install_weekday: Binary flag indicating if the event was on a weekday.

C. Splitting Dataset

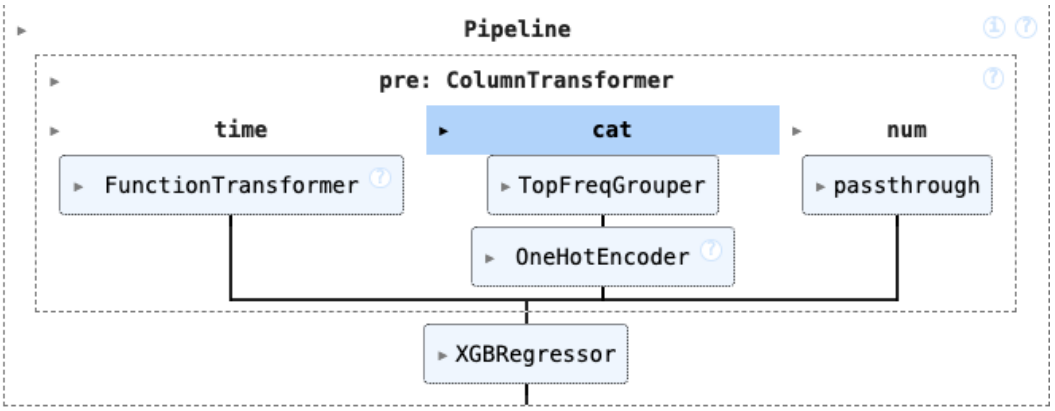
1. Kept only rows where ecpi was not null for model training.
2. Target (y) = ecpi
3. Features (X) = All columns except ecpi and churn
4. Split into training and validation sets:
    - 90% training / 10% validation
    - random_state=42 for reproducibility

D. Custom Categorical Preprocessing

1. Created a custom transformer: TopFreqGrouper
    - Groups infrequent categories into an "OTHER" label if frequency < 1%.
2. Applied OneHotEncoder on grouped categories.
3. Passed numeric columns without transformation.

E. ColumnTransformer for Pipeline Integration

1. Time features extracted via FunctionTransformer.
2. Categorical features processed through the cat_pipe (TopFreqGrouper + OneHotEncoder).
3. Numeric features used as-is via 'passthrough'.
4. After all transformation 87 resulted with columns



Model Selection

Having completed preprocessing (i.e., feature engineering, encoding, scaling), we tried various regression models to determine the optimal method for predicting the ECPI (Event Cost Per Install) metric. The performance is measured using Root Mean Squared Error (RMSE) on cross-validation (5-fold) and validation sets where lower values indicate higher accuracy.

1. Linear Regression

Approach: Models linear relationships between features and targets.
Results:
CV RMSE: 0.15208 | RMSE value: 0.15605
Interpretability: Provided a detailed equation with coefficients for all features (e.g., country, device_brand, totalPowerUp).

The equation shown as

Figure 5.1

2. Lasso Regression

Approach: Linear regression with L1 regularization to reduce overfitting by shrinking coefficients.
Results:
CV RMSE: 0.23732 | RMSE value: 0.23738
The equation shown as

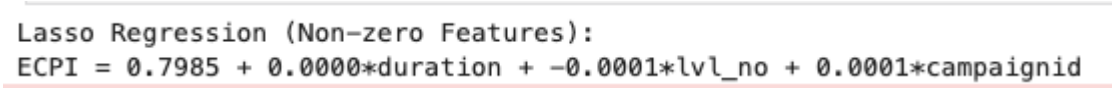


Figure 5.1

3. Decision Tree

Approach: Non-linear model that splits data hierarchically based on feature thresholds.
Results:
CV RMSE: 0.08581 | RMSE value: 0.07482

4. Random Forest

Approach: Ensemble of decision trees with bagging to reduce variance. Number of estimator is 100
Results:
CV RMSE: 0.06469 | RMSE value: 0.05319

Advantage: Robust to noise and overfitting, with the best validation performance.
5. XGBoost

Approach: Gradient-boosted trees optimized for speed and performance. n_estimators are 400 and learning rate is 0.05, max depth 6
Results:
CV RMSE: 0.06619 | RMSE value: 0.05597
Strength: Competitive performance, slightly behind Random Forest but still strong.

```
Model RMSE (lower is better):
              CV_RMSE  Val_RMSE
Linear        0.15208   0.15605
Lasso         0.23732   0.23738
DecisionTree  0.08581   0.07482
RandomForest  0.06469   0.05319
XGBoost       0.06619   0.05597
```

Figure 5.1 Overall Evaluation

Even though the best performance was seen in the Random Forest Regressor, intuitively the XGBoost Model because generally it performs better with more data and RMSE difference was not significant. Therefore XGB model trained with 100% train data used for filling empty values in test and train data.

6. Model Development for “Churn” and Inferences

Build a robust classification model to predict user churn from preprocessed data, optimizing for recall (identifying actual churners) while maintaining good precision and accuracy.

Data Preprocessing

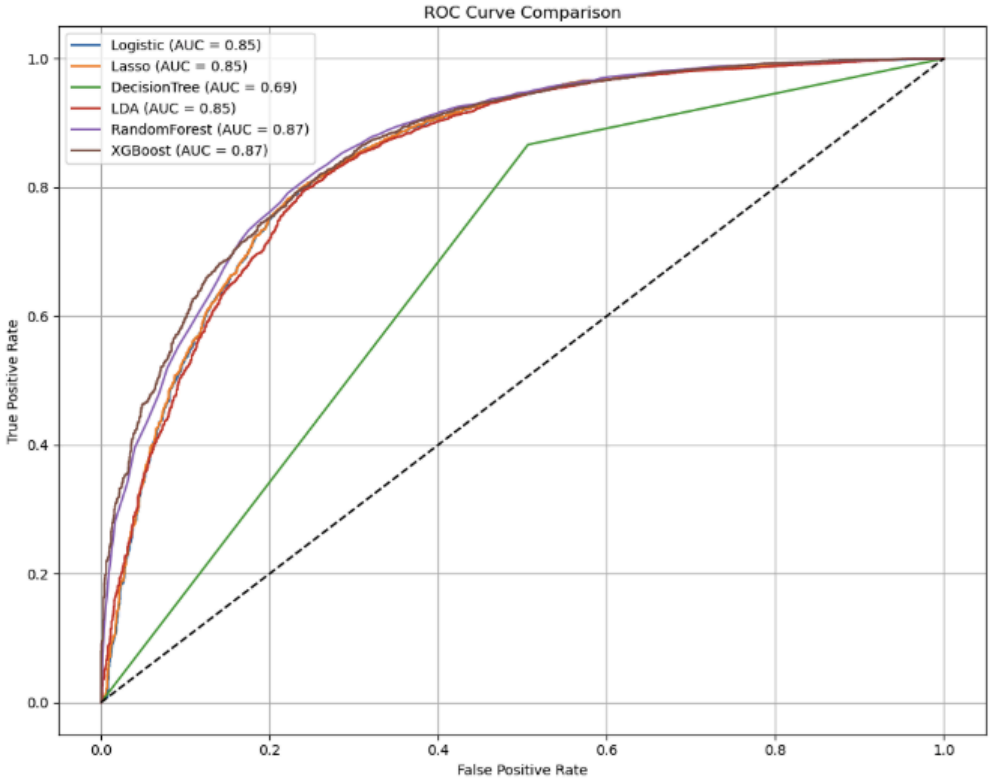
Main 2 different data preprocessing method used.

Aspect	Version 1: Selective One-Hot Encoding	Version 2: Full One-Hot Encoding via pd.get_dummies
Categorical Columns Encoded	Only key columns (country, device_brand, re_install, lang, campaignid, partnerid)	All categorical columns including week, hour, day, etc.
Rare Category Handling	Rare values (<1%) grouped as 'other'	Not explicitly handled – all categories expanded
Encoding Method	sklearn.OneHotEncoder	pd.get_dummies() with drop_first=True
Final Feature Count	~87 columns	482 columns
Pros	Simpler, more interpretable, lower risk of overfitting	Captures more granular variation and interactions
Cons	May miss useful interactions	Higher dimensionality increases risk of overfitting and longer training time

Model Evaluation Summary

Encoding Version	Model	Accuracy	Precision	Recall	F1 Score	ROC AUC
Selective OHE (~87 cols)	Logistic Regression	0. 8520	0.8660	0.9637	0.9122	0.8136
	Poly Logistic (d=2)	0.7978	0.8890	0.8531	0.8707	0.7229
	LDA	0.8508	0.8563	0.9769	0.9126	0.8434
	Decision Tree	0.7964	0.8750	0.8689	0.8719	0.6896
	Random Forest	0.8584	0.8841	0.9466	0.9143	0.8588
	XGBoost	0.8570	0.8832	0.9459	0.9134	0.8597
Full OHE (~482 cols)	Logistic Regression	0.8543	0.8654	0.9676	0.9137	0.8479

	Lasso (L1)	0.8544	0.8654	0.9677	0.9137	0.8491
	LDA	0.8466	0.8521	0.9769	0.9102	0.8460
	Decision Tree	0.7929	0.8722	0.8669	0.8696	0.6851
	Random Forest	0.8579	0.8796	0.9518	0.9143	0.8664
	XGBoost	0.8539	0.8780	0.9484	0.9118	0.8656



Inferences from Feature Importance Analysis

To understand what drives churn in our dataset, we analyzed feature importances from three different model families: **Random Forest**, **XGBoost**, and **Logistic Regression**. Each provides unique interpretability:

Random Forest — Gini Importance

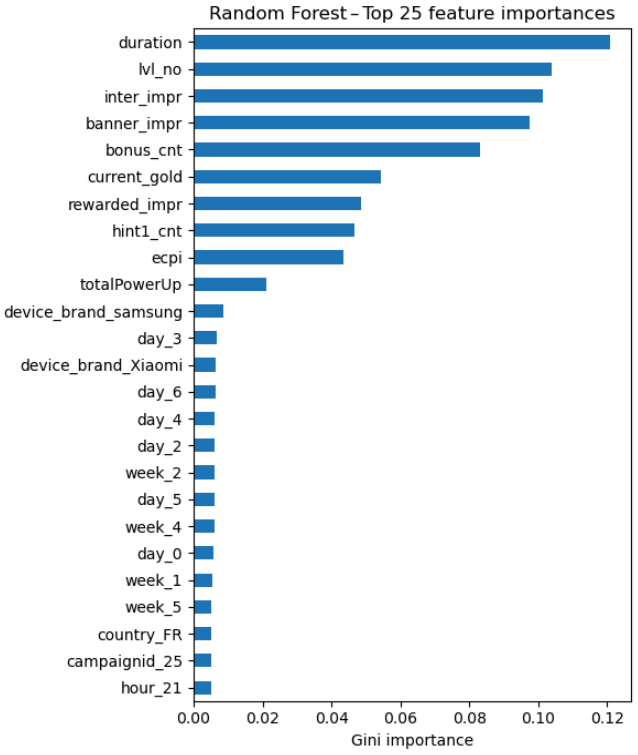
The most significant feature was duration (session length), followed by lvl_no, inter_impr, and banner_impr.

Features like bonus_cnt, rewarded_impr, and hint1_cnt for in-game behavior were also highly ranked.

Some engineered time features like day_3, day_6, and week_2 made it to the top 25, and what that suggests is that the weekday/hour when the user plays impacts churn.

Marketing identifiers (campaignid_25, partnerid_4) were relatively lower in significance but did contribute.

Interpretation: Session duration and level number are stronger retention indicators. Low ad interaction (inter_impr, banner_impr) is a sign of churn.

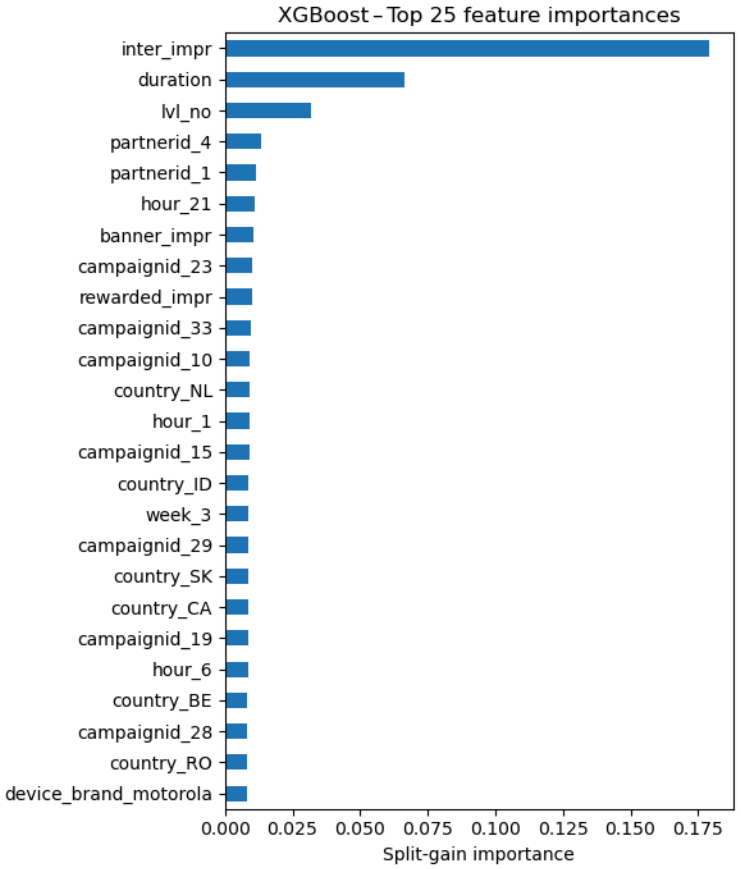


XGBoost — Gain-Based Importance

After training inter_impr was the best predictor of churn (gain = 0.179) followed by duration and lvl_no.
Some of the most frequently occurring features were partnerid_4, campaignid_23, campaignid_33, and country_ID/NL, which suggests the ability of XGBoost to detect nonlinearity between campaign and geography.

Time-of-day activity (e.g., hour_21, hour_6, week_3) also had gain value, suggesting user patterns of behavior are significant.

Interpretation: XGBoost captured more advanced behavioral-marketing interactions (e.g., partnerid_4 at late-night) than Random Forest.



Logistic Regression Coefficients

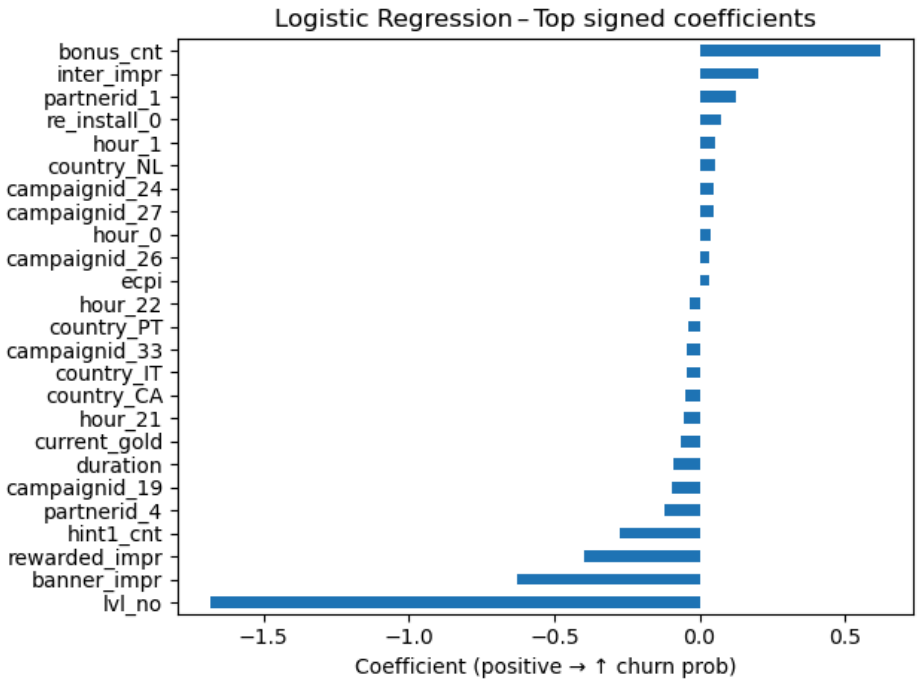
Largest negative coefficients (estimating lower churn):

lvl_no (−1.68), banner_impr (−0.63), rewarded_impr (−0.40), hint1_cnt (−0.27)

Largest positive coefficients (estimating higher churn):

bonus_cnt (+0.62), inter_impr (+0.20), partnerid_1 (+0.13)

Interpretation: The logistic regression confirms player engagement metrics as significant, along with indicating some acquisition channels (partnerid_1, campaignid_19) as problematic.



---

7. Conclusion

The goal in this project was to build a churn prediction model that would detect mobile game users most likely to churn in their first week. Early churn is rampant in mobile gaming, and being able to predict it accurately enables timely, targeted intervention to improve retention and long-term value.
It began with extensive EDA and data cleaning. Missing ECPI values were imputed using a regression model (XGBoost) fit on a selectively encoded feature set (~87 features). This selective encoding — instead of the full one-hot expansion (~482 features) — was used to reduce dimensionality and improve interpretability for imputation. Time-related features such as install hour and weekday were extracted and found to be somewhat predictive of churn (especially install hour, $p < 0.001$ via ANOVA).

Several classification models were compared on a stratified train-validation split. Validation metrics were computed over key indicators like accuracy, precision, recall, F1 score, specificity, and ROC AUC. The best performers were Random Forest and XGBoost, both trained on a full encoding (~482 features). They performed very similarly on validation:

Random Forest:
Accuracy: 0.8584
Precision: 0.8841
Recall: 0.9466
F1 Score: 0.9143
ROC AUC: 0.8588

XGBoost:
Accuracy: 0.8570
Precision: 0.8832
Recall: 0.9459
F1 Score: 0.9134
ROC AUC: 0.8597

To finalize the model selection, predictions were compared on the held-out test set. Random Forest and XGBoost had a disagreement rate of only 3.25%, confirming both were well-calibrated. Despite the closeness, XGBoost was chosen as the final model due to its slightly superior AUC and general robustness on big tabular datasets.

Feature importance between models consistently implicated core behavioral signals — most notably inter_impr, duration, lvl_no, and banner_impr — as top predictors of churn. One engineered feature, lvl_duration_ratio, was tried but failed to improve model performance and was excluded from the final model.