

RETAIN THE LEGENDS: PREDICTING & PREVENTING CHURN - APEX LEGENDS S15

BY: VICTORIA BRIGOLA



BEHIND THE ECLIPSE: WHAT DRIVES THIS STUDY

INSPIRATION

**My gamer &
game-art roots
sparked a deep dive
into why Season 15
Eclipse players
drop off.**

SCOPE

**Season 15
'Eclipse' isolates a
single meta,
avoiding
cross-patch noise.**

OBJECTIVE

**Goal: a 7-day churn
early-warning
system for design,
live-ops, and
monetization.**

THE CHURN CHALLENGE

- **Churn rate:** ~70.9 % of Season 15 player match observations go silent (≥ 7 days idle)
- **Key question:** Can match-level stats, legend choices, and play-cadence forecast churn one week ahead?
- **Process:** Data wrangling, exploratory analysis & insight generation, feature engineering, model training/evaluation & interpretation via feature importances.

ECLIPSE ALLIES: STAKEHOLDER LINEUP

GAME DESIGN (RESPAWN LEADS):

Use churn insights to rotate legends and tweak balance.

**MARKETING
(STRATEGISTS):**
Send targeted outreach when play frequency declines.

LIVE-OPS (PRODUCT MANAGERS):

Trigger events/reminders before 7-day inactivity, when predicted risk crosses threshold.

**MONETIZATION
(TEAMS):**
Time battle pass & bundles around churn-risk peaks.



PIPELINE GAME PLAN

- **Dataset:** Apex Legends Season 15 (Ranked) · target 7-day churn
- **Workflow:** Prep → split → compare models → tune → test → set decision threshold
- **Models:** Logistic Regression, Random Forest, XGBoost
- **Tools:** Python (pandas, NumPy, scikit-learn, imbalanced-learn/SMOTE), XGBoost, matplotlib · Jupyter



SEASON 15 DATA LENS:

[HTTPS://WWW.KAGGLE.COM/DATASETS/D8TARY/APEX-LEGENDS-SEASON-15-RANKED-DATASET-RAW](https://www.kaggle.com/d8tary/apex-legends-season-15-ranked-dataset-raw)

**Key Features (36 columns incl. 2 empty → 34 usable;
Nov 3, 2022–Jan 15, 2023):**

- **Combat:** my_kills, my_damage, my_assists
- **Legends & Squads:** my_legend, legend_diversity (engineered), teammate_count (proxy for squad size)
- **Cadence:** session_frequency (7-day, engineered), days_since_last_match (engineered)
- **Performance:** squad_placed, my_revives, my_knocks

0	date	499	non-null	datetime64[ns]
1	game	499	non-null	int64
2	map	499	non-null	object
3	match_type	499	non-null	object
4	my_duration	265	non-null	float64
5	my_rank	498	non-null	object
6	rp_earned	487	non-null	float64
7	premade_squad	497	non-null	object
8	voice_chat	497	non-null	object
9	squad_placed	475	non-null	float64
10	teammate_count	314	non-null	float64
11	my_quit	317	non-null	float64
12	teammate_quit_count	306	non-null	float64
13	my_legend	287	non-null	object
14	teammate_1_legend	277	non-null	object
15	teammate_2_legend	269	non-null	object
16	my_damage	284	non-null	float64
17	teammate_1_damage	275	non-null	float64
18	teammate_2_damage	270	non-null	float64
19	my_kills	283	non-null	float64
20	teammate_1_kills	278	non-null	float64
21	teammate_2_kills	274	non-null	float64
22	my_assists	281	non-null	float64
23	teammate_1_assists	277	non-null	float64
24	teammate_2_assists	273	non-null	float64
25	my_knocks	281	non-null	float64
26	teammate_1_knocks	278	non-null	float64
27	teammate_2_knocks	273	non-null	float64
28	my_revives	283	non-null	float64
29	teammate_1_revives	277	non-null	float64
30	teammate_2_revives	274	non-null	float64
31	my_respawns	283	non-null	float64
32	teammate_1_respawns	276	non-null	float64
33	teammate_2_respawns	275	non-null	float64
34	Unnamed: 34	0	non-null	float64
35	Unnamed: 35	0	non-null	float64

DATA RECON: INITIAL INSPECTION

- `df.shape: (499, 36)`
- `df.head()`
- `df.describe()`
- `df.columns`
- `df.isna().sum()`
- `df.duplicated().sum()`

```
df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 499 entries, 0 to 498
Data columns (total 36 columns):
#   Column                Non-Null Count  Dtype  
---  -
0   date                  499 non-null   datetime64[ns]
1   game                  499 non-null   int64   
2   map                   499 non-null   object  
3   match_type           499 non-null   object  
4   my_duration           265 non-null   float64  
5   my_rank               498 non-null   object  
6   rp_earned             487 non-null   float64  
7   premade_squad         497 non-null   object  
8   voice_chat           497 non-null   object  
9   squad_placed          475 non-null   float64  
10  teammate_count        314 non-null   float64  
11  my_quit               317 non-null   float64  
12  teammate_quit_count    306 non-null   float64  
13  my_legend             287 non-null   object  
14  teammate_1_legend     277 non-null   object  
15  teammate_2_legend     269 non-null   object  
16  my_damage             284 non-null   float64  
17  teammate_1_damage     275 non-null   float64  
18  teammate_2_damage     270 non-null   float64  
19  my_kills              283 non-null   float64  
20  teammate_1_kills      278 non-null   float64  
21  teammate_2_kills      274 non-null   float64  
22  my_assists            281 non-null   float64  
23  teammate_1_assists    277 non-null   float64  
24  teammate_2_assists    273 non-null   float64  
25  my_knocks             281 non-null   float64  
26  teammate_1_knocks     278 non-null   float64  
27  teammate_2_knocks     273 non-null   float64  
28  my_revives            283 non-null   float64  
29  teammate_1_revives    277 non-null   float64  
30  teammate_2_revives    274 non-null   float64  
31  my_respawns           283 non-null   float64  
32  teammate_1_respawns   276 non-null   float64  
33  teammate_2_respawns   275 non-null   float64  
34  Unnamed: 34           0 non-null     float64  
35  Unnamed: 35           0 non-null     float64  
dtypes: datetime64[ns](1), float64(26), int64(1), object(8)
memory usage: 140.5+ KB
```

```
df.dtypes

date                  datetime64[ns]
game                  int64
map                   object
match_type           object
my_duration           float64
my_rank              object
premade_squad         object
voice_chat           object
squad_placed          float64
teammate_count        float64
teammate_quit_count   float64
my_legend            object
teammate_1_legend     object
teammate_2_legend     object
my_damage            float64
teammate_1_damage     float64
teammate_2_damage     float64
my_kills             int64
teammate_1_kills      int64
teammate_2_kills      int64
my_assists           int64
teammate_1_assists    int64
teammate_2_assists    int64
my_knocks            int64
teammate_1_knocks     int64
teammate_2_knocks     int64
my_revives           int64
teammate_1_revives    int64
teammate_2_revives    int64
my_respawns          int64
teammate_1_respawns   int64
teammate_2_respawns   int64
dtype: object
```

FORGE THE DATA - DATA CLEANING

- *Check duplicate rows*

df.duplicated().sum() → none dropped

- *Remove columns*

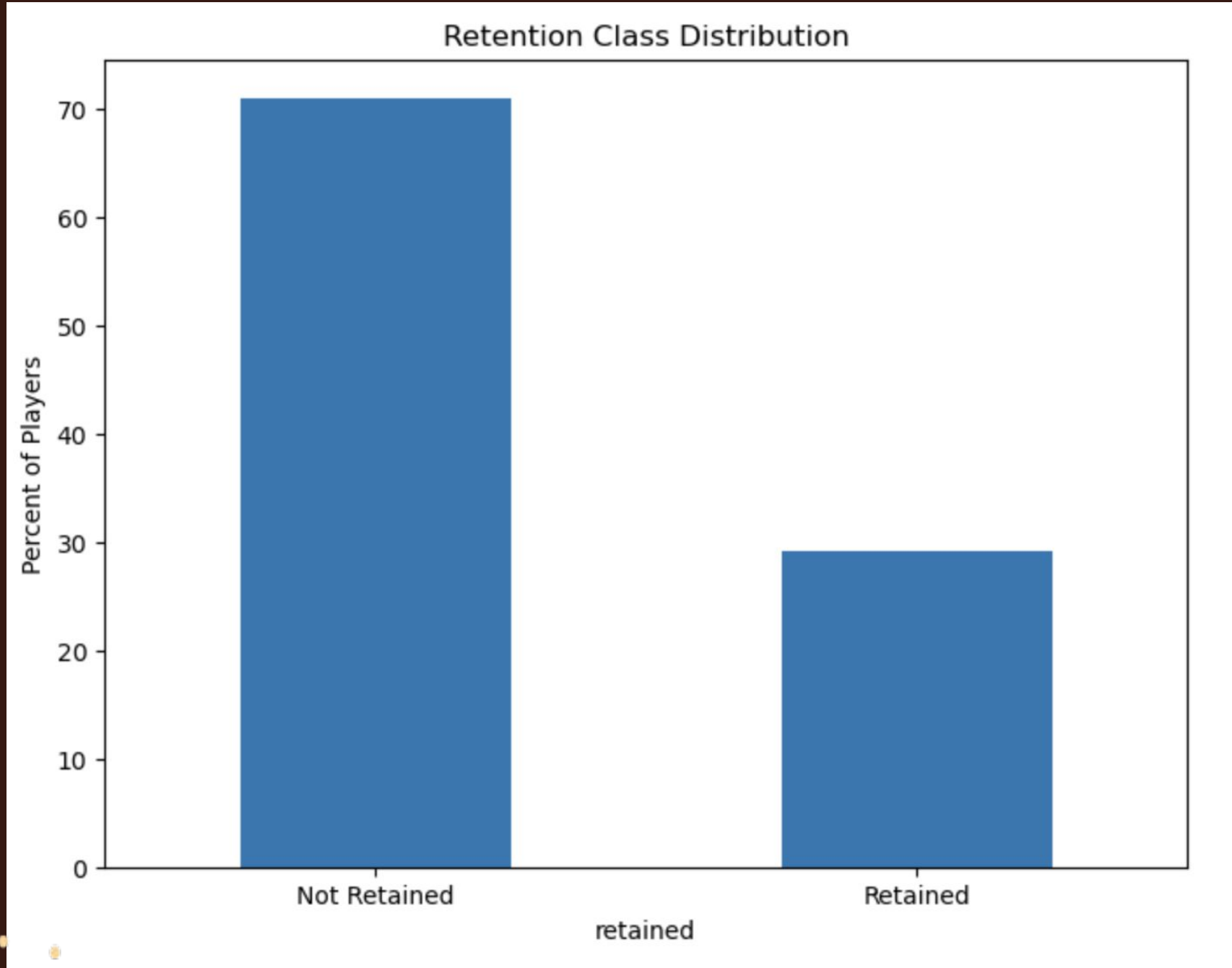
Unnamed: 34, Unnamed: 35 (placeholders); rp_earned, my_quit (non-model)

- *Impute missing values*

**my_*/teammate_* counts → 0 ; numeric → median ; categorical columns → mode;
cast zero-filled counts to int**

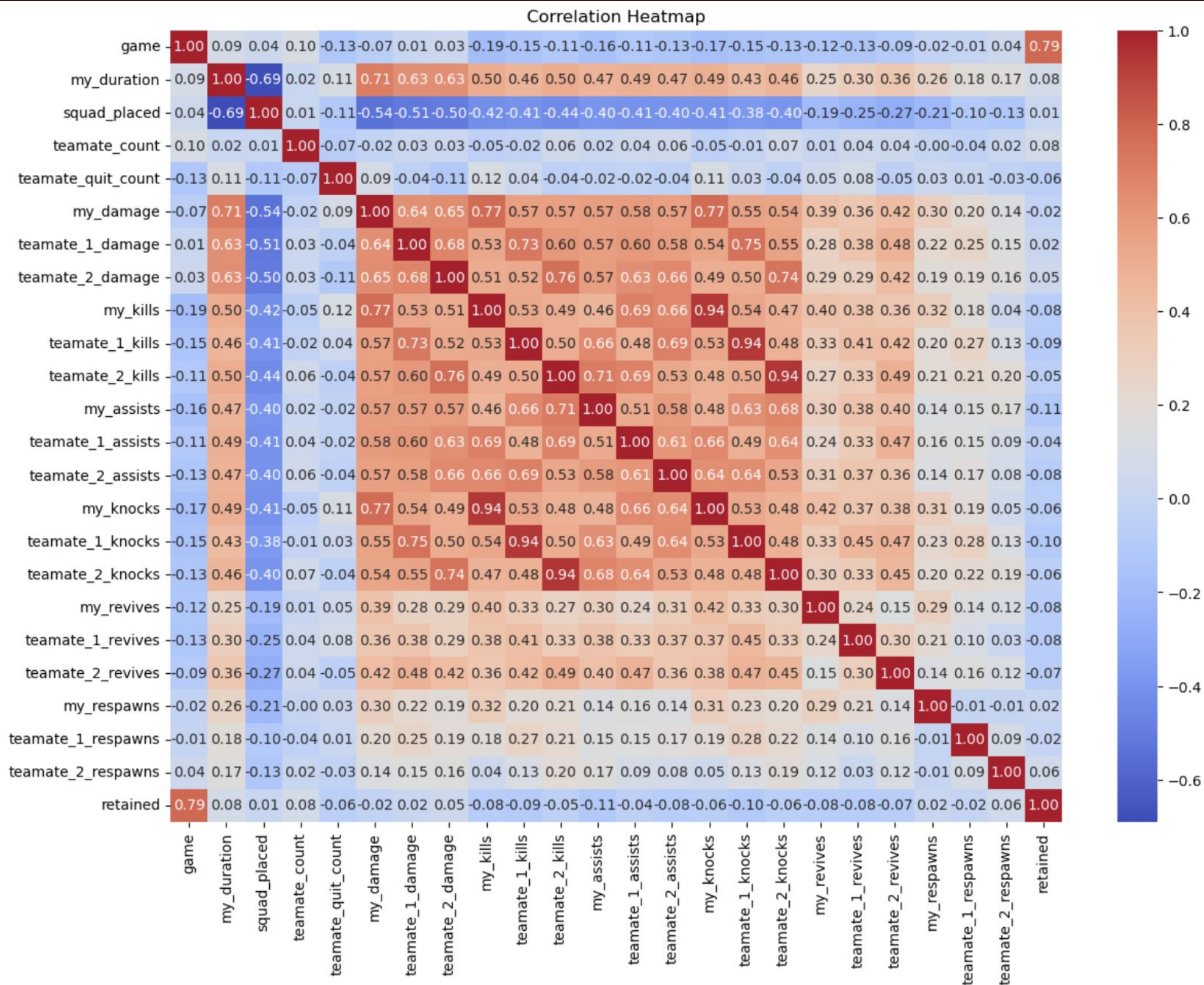
- *df.shape*
(499, 35)

CHURN DISTRIBUTION (BY MATCH RECORDS): EDA



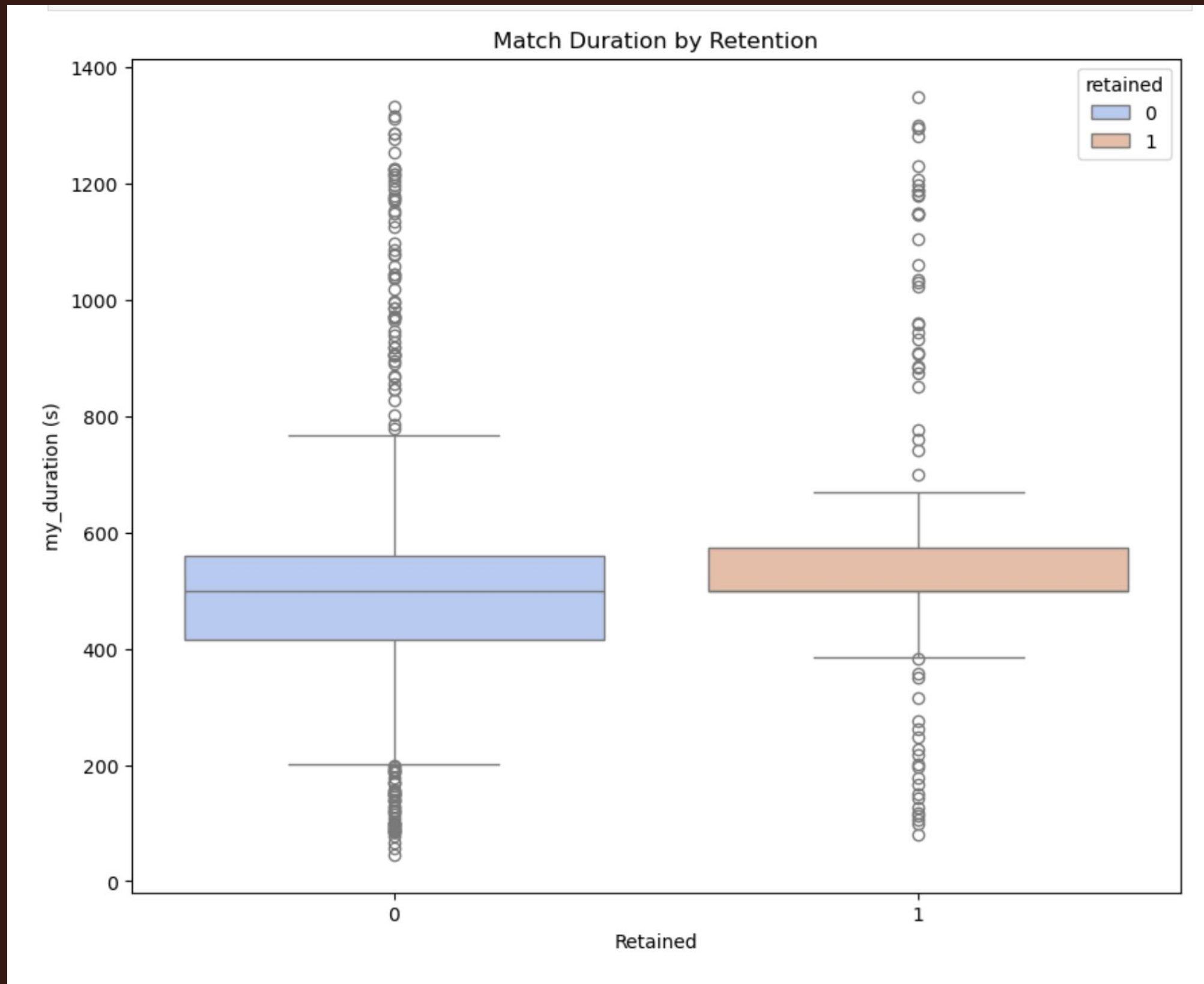
- **RETAINED:** 145 MATCHES (29%)
- **CHURNED:** 354 MATCHES (71%)

CORRELATION HEATMAP: EDA



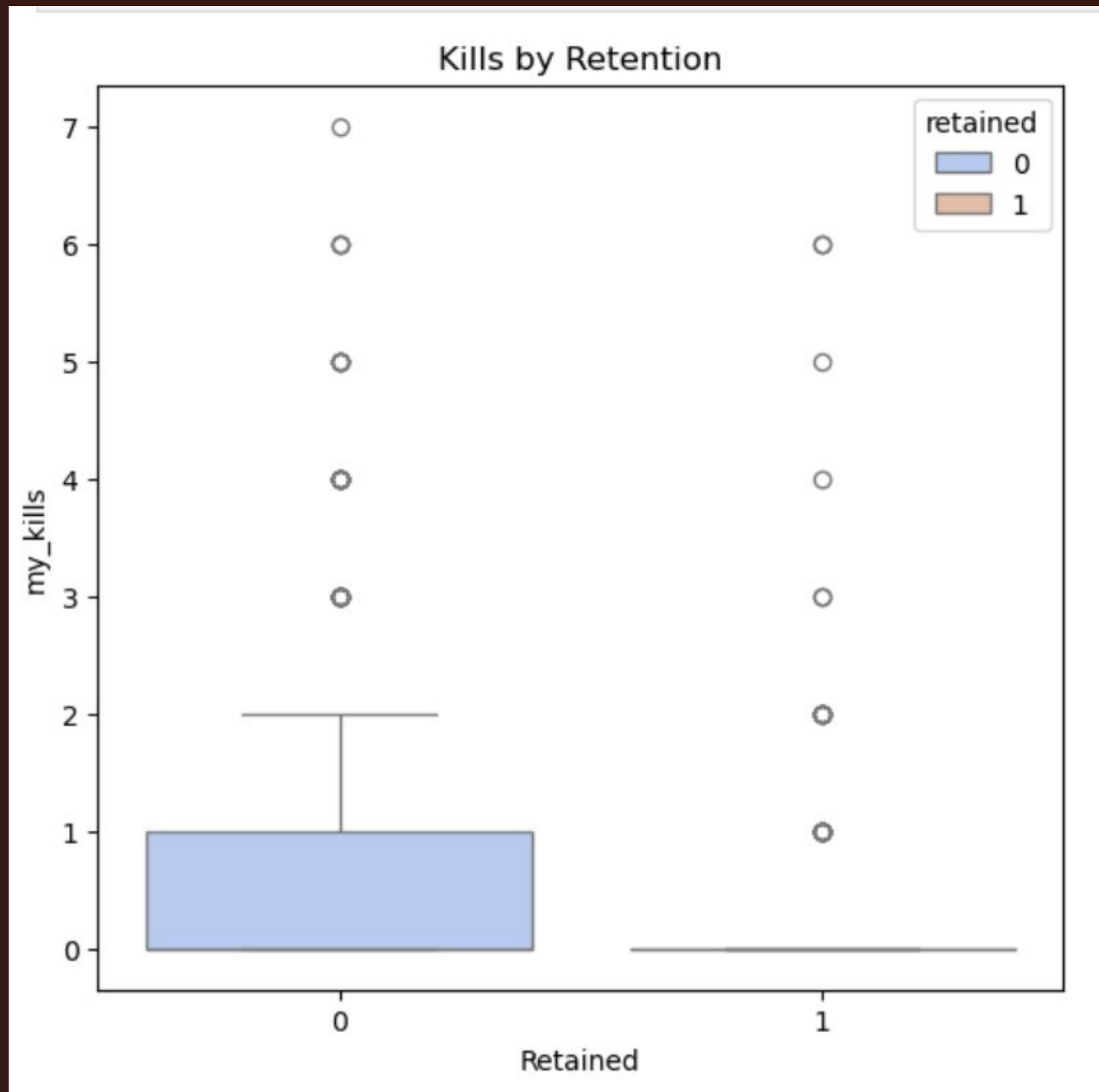
- Key features: my_duration, my_damage, my_kills
- Top pair: my_kills vs my_knocks - $r \approx 0.94$
- Supporting: my_duration vs my_damage $\rightarrow r \approx 0.71$;
my_revives vs my_knocks $\rightarrow r \approx 0.42$

MATCH DURATION BY RETENTION: EDA



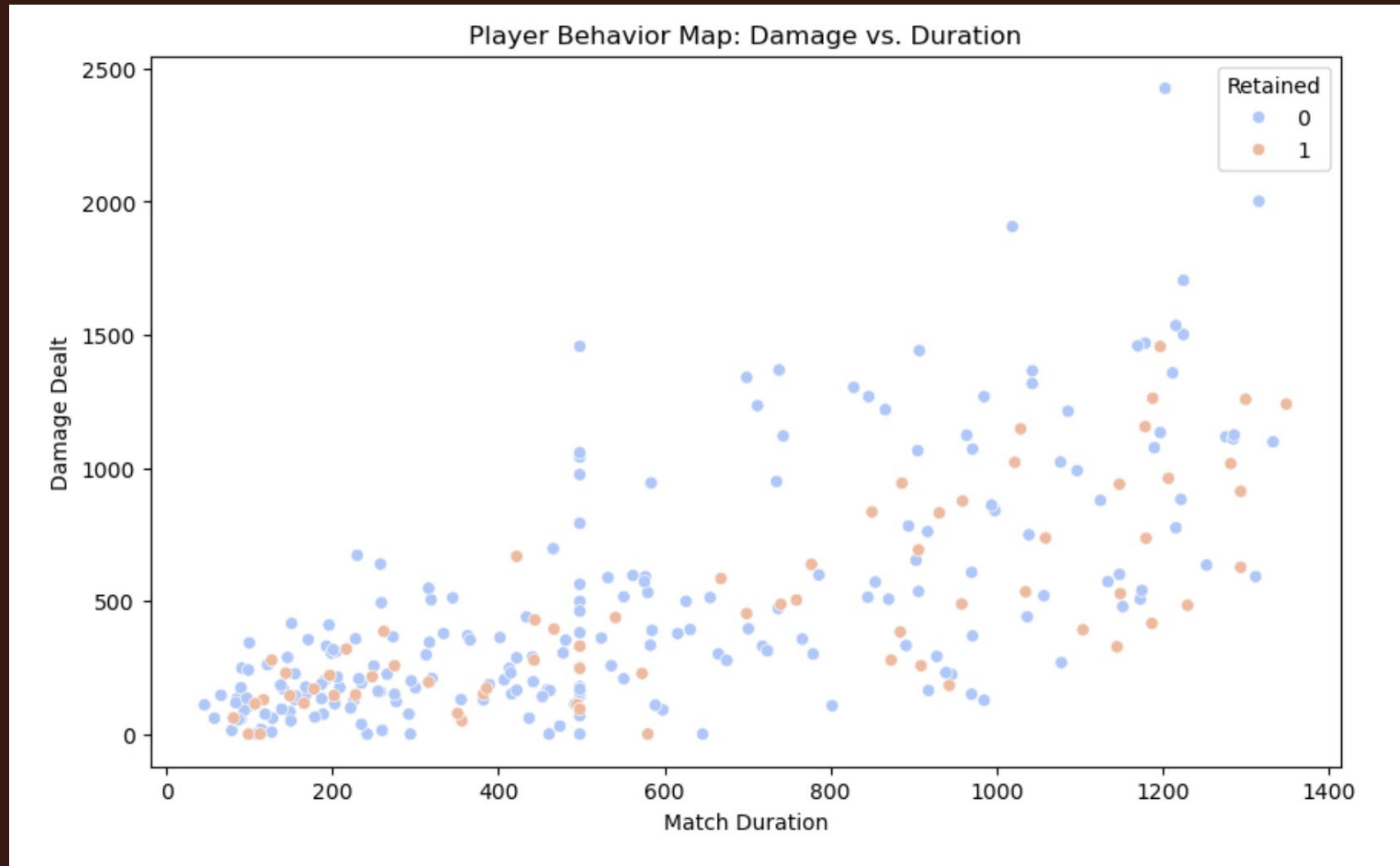
- **Retained (1): higher mean session length; median ≈ 498 s**
- **Churners (0): distribution shifted shorter; fewer long sessions**
- **Model input: my_duration (s), standardized with other numeric features**

KILLS BY RETENTION: EDA



- **Medians:** both groups = 0
- **Means:** churners (0) **0.61** > retained (1) **0.40**
- **Model input:** my_kills (per match), standardized

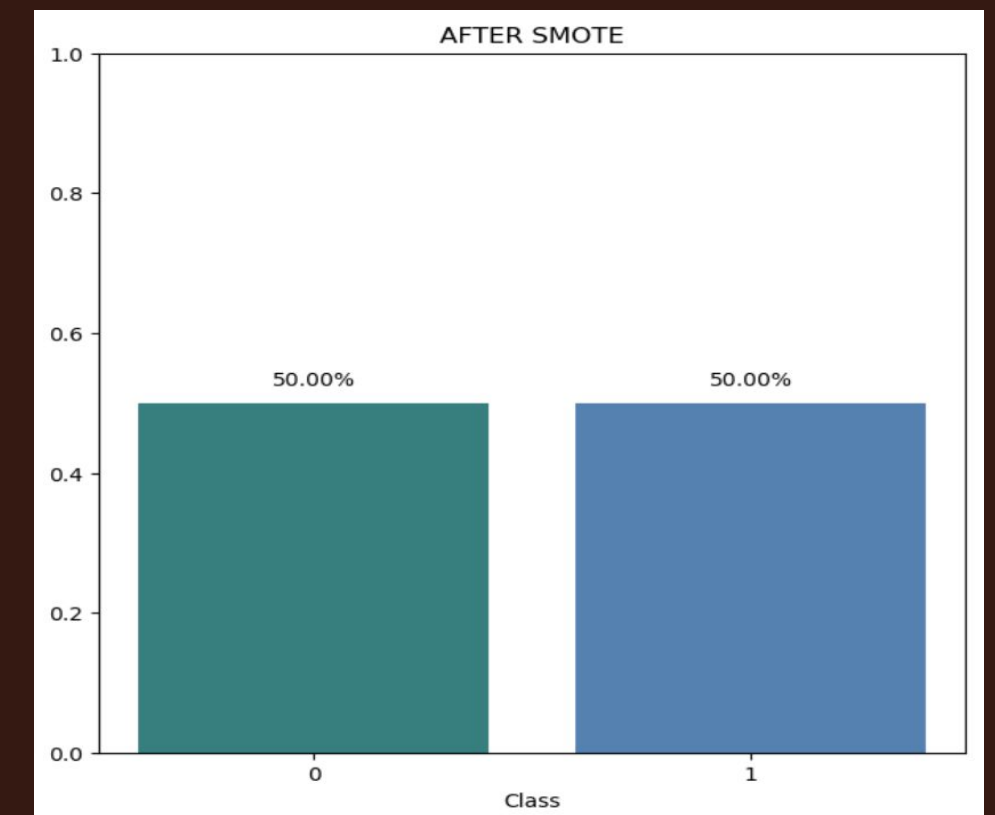
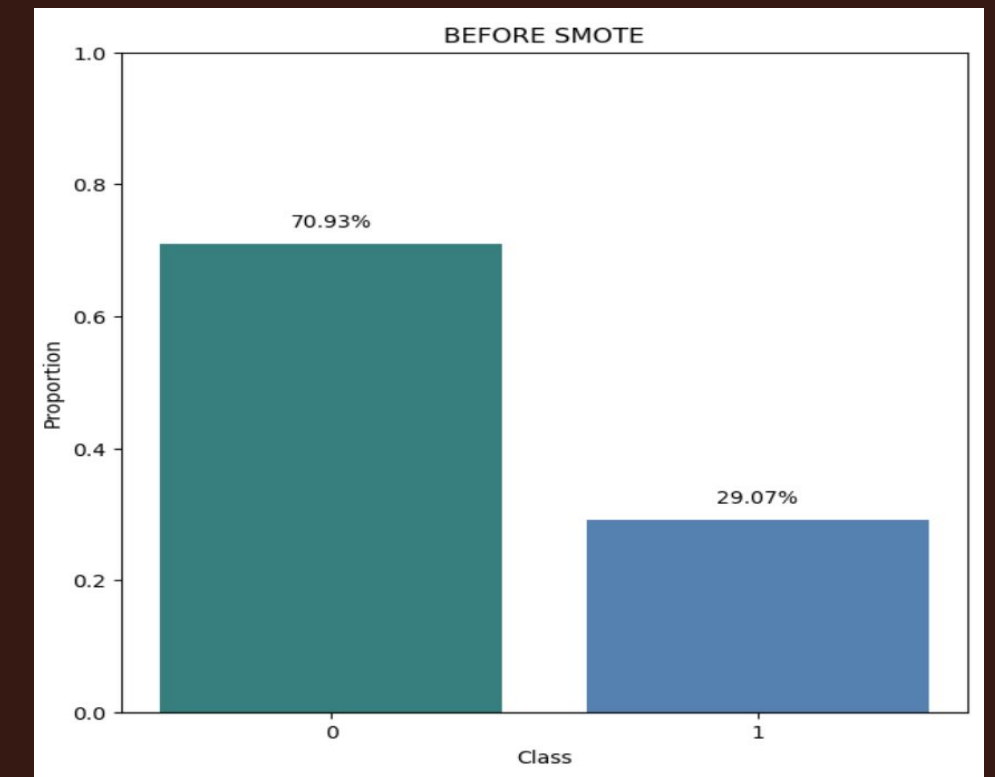
DAMAGE VS DURATION: EDA



- **Features: my_duration (s), my_damage**
- **Relationship: strong positive correlation ($r \approx 0.71$)**
- **Pattern: longer matches → more damage; labels overlap (no single-threshold rule)**

PRE-MATCH TUNE-UP: PREPROCESSING

- **Train/Test:** stratified 80/20 (X_train 399, X_test 100)
- **Encode:** one-hot categoricals (drop_first), align test to train schema
- **Scale:** StandardScaler on numeric columns (post-encoding)
- **SMOTE (LR-only, in-CV):** retained (class 1) lifted ~29% → ~50% in training folds; base train/test unchanged
- **Final matrix:** X_train (399, 114) · X_test (100, 114)



DIALED IN: TUNING & SIGNALS

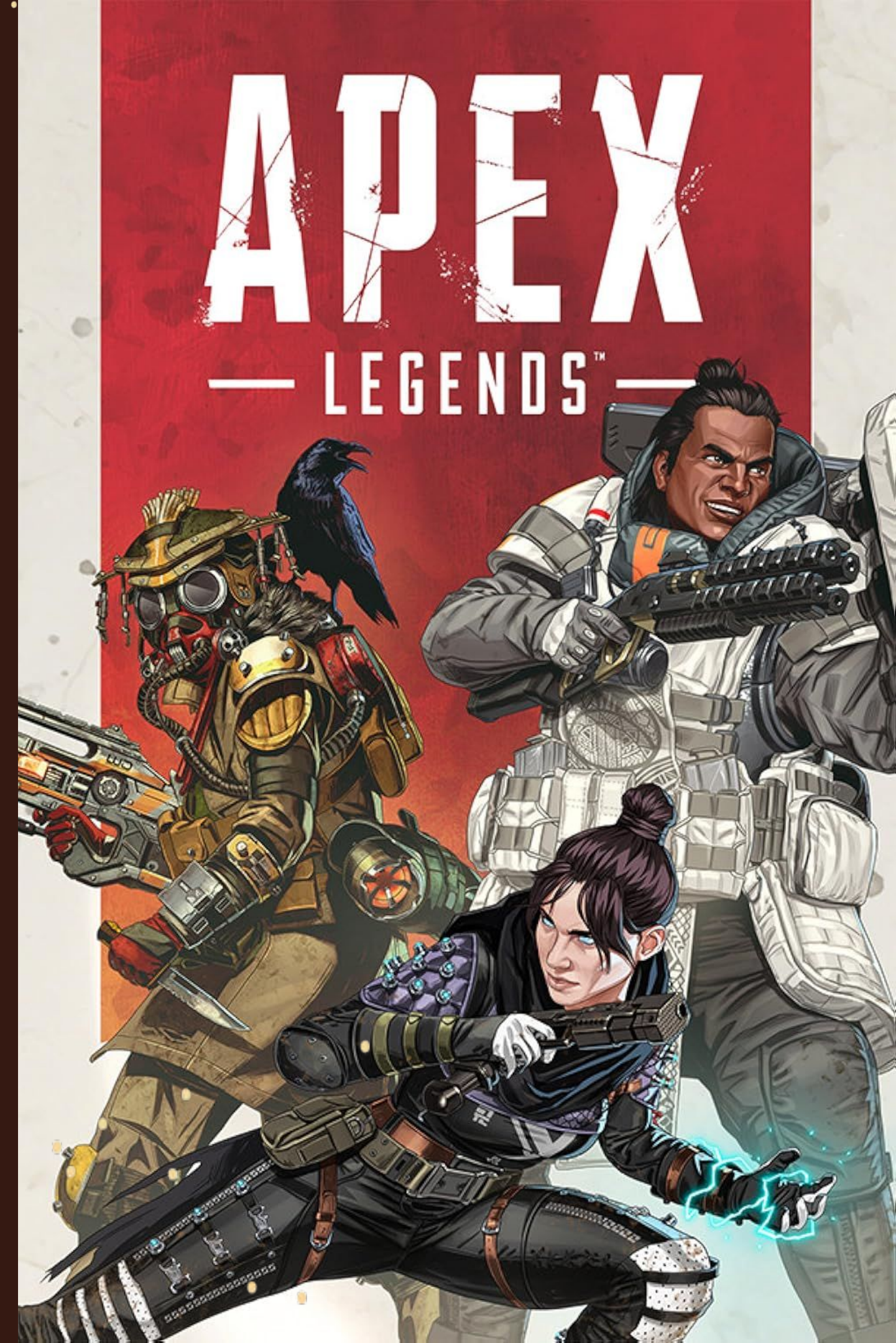
- **Tuning:** 5× CV + SMOTE (LR-only, in-CV); optimize F1
- **LR tuned config:** L1, C=0.1, liblinear, class_weight=None
- **Top signals (effect on churn risk):**
 - Monday play pattern — increases risk (*coef +0.127*)
 - Teammate legend: Horizon — increases risk (*coef +0.076*)
 - Revives — increases risk (*coef +0.070*)
 - Assists — decreases risk (*coef -0.133*)
 - Teammate knocks — decreases risk (*coef -0.099*)

FINAL MODEL SHOOT-OUT

Model	F1	PR AUC	ROC AUC	Precision	Recall	Accuracy
Logistic Regression — tuned	0.471	0.304	0.516	0.311	0.966	0.370
Logistic Regression — baseline	0.315	0.285	0.397	0.233	0.483	0.390
XGBoost (current)	0.269	0.288	0.492	0.304	0.241	0.620
Random Forest (current)	0.237	0.255	0.413	0.233	0.241	0.550
XGBoost — simple baseline	0.157	0.260	0.441	0.182	0.138	0.570
Random Forest — simple baseline	0.000	0.283	0.465	0.000	0.000	0.660

EARLY RADAR: CAN CHURN BE PREDICTED A WEEK OUT?

- **Winner:** Logistic Regression (L1), tuned
- **Why it wins:** Recall-first early warning; interpretable; simple to ship.
- **Hold-out results:** Recall 0.97 (*F1 0.47*) on the 7-day churn label.
- **What it enables:** Nightly risk list for Live-Ops; coefficients make nudges explainable.



THE PLAYBOOK:

FLAG → NUDGE → RETAIN

- **Targeting:** Threshold from precision–recall (PR) curve; prioritize lower assists/knocks and a Monday spike.
- **Timing:** Nudge at 5–6 idle days (just before the 7-day label).
- **Tactics:** Reminders, squad-up/assist challenges, LTM surfacing, rank-placement prompts.
- **Guardrails:** Outreach cap, cooldowns, max touches/week.
- **Learning loop:** Log actions→outcomes; weekly precision/recall review; monitor drift, refit each season, A/B test nudges.

