

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.**

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 (35 total from November 2, 2022 to January 14, 2023):

- **Combat:** kills, damage, assists
- **Legends & Squads:** legend_choice, legend_diversity, squad_size
- **Cadence:** session_frequency, days_since_last_match
- **Performance:** match_placement, revives, accuracy

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

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 stops.

LIVE-OPS (PRODUCT MANAGERS):

Trigger events/reminders at 7-day inactivity.

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



THE CHURN CHALLENGE

- **Churn rate:** ~25 % of Season 15 players 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

DATA RECON: INITIAL INSPECTION

- **df.shape:**
(499, 35)
- **df.head()**
- **df.describe()**

```
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

- *Drop duplicate player-match entries*

```
df.drop_duplicates(subset=['player_id','match_id'], inplace=True)
```

- *Remove columns*

```
to_drop = ['spectator_count','streamer_flag','session_id',  
'rp_bin','rp_earned','rp_delta','rp_change','my_quit','teammate_quit_count', 'game_id']
```

```
df.drop(columns=to_drop, inplace=True)
```

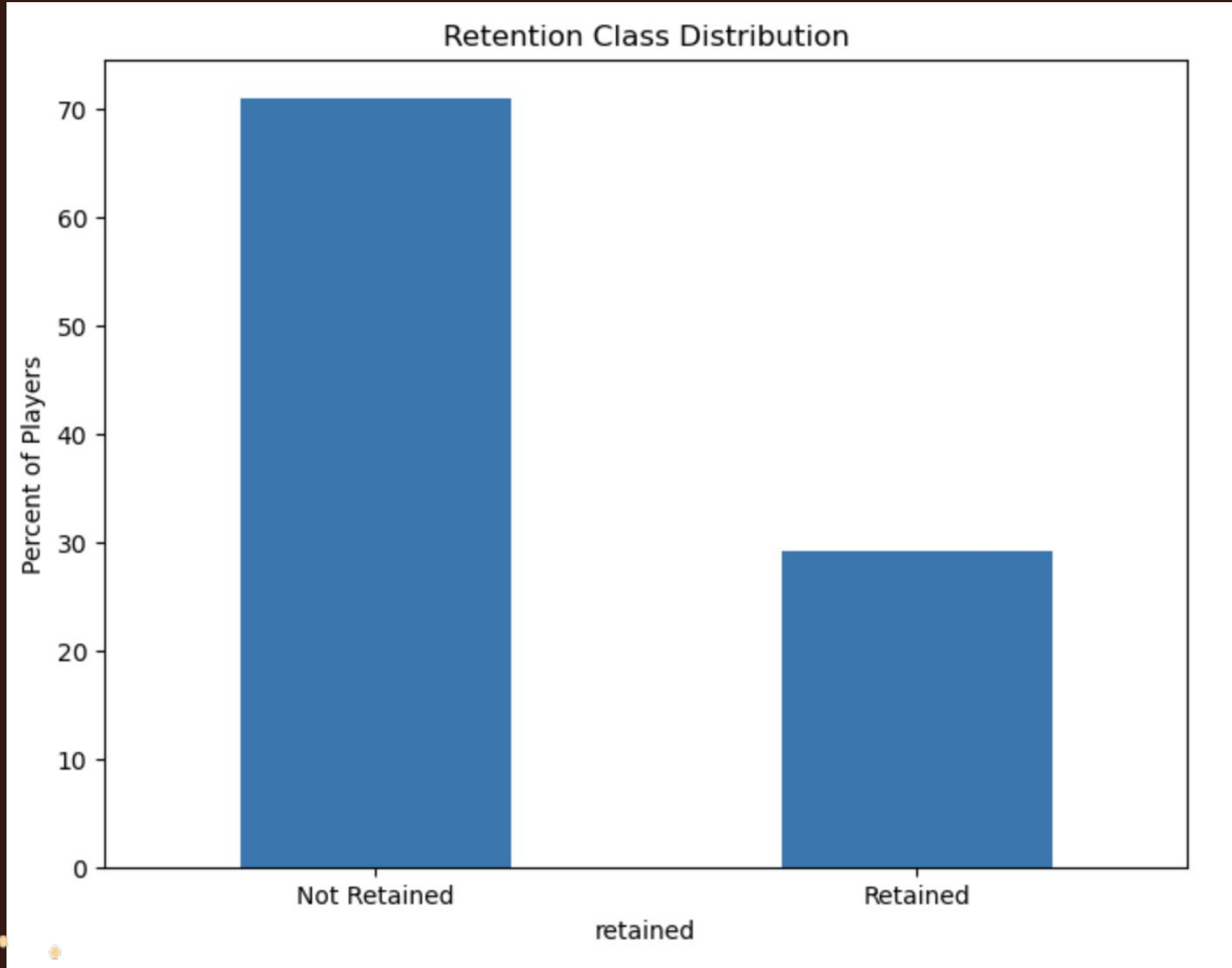
- *Impute missing match_duration with map-level median*

```
df['match_duration'] = (df.groupby('map_id')['match_duration'].transform(lambda x:  
x.fillna(x.median())))
```

```
df.shape
```

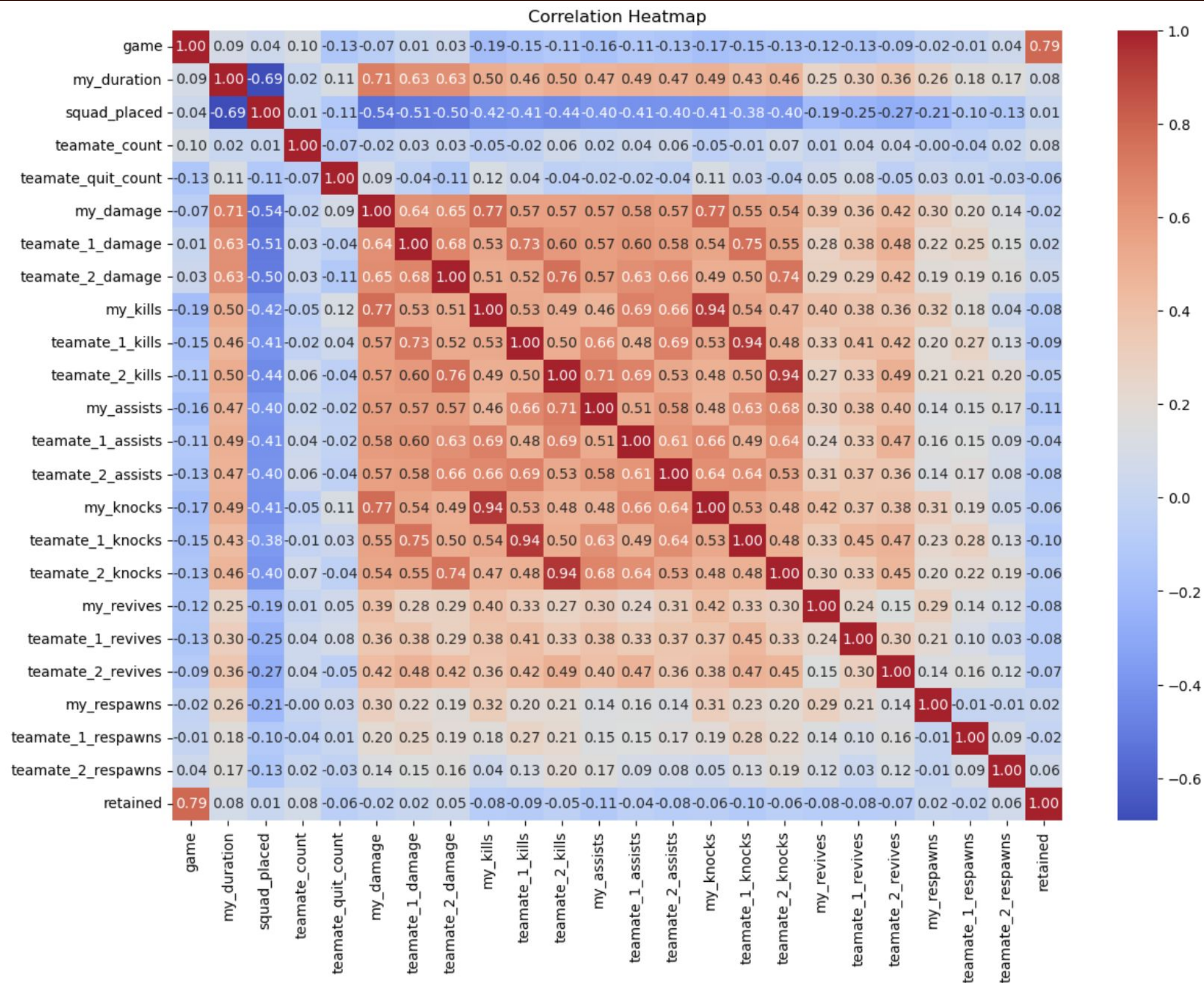
```
(499, 33)
```

CHURN DISTRIBUTION: EDA



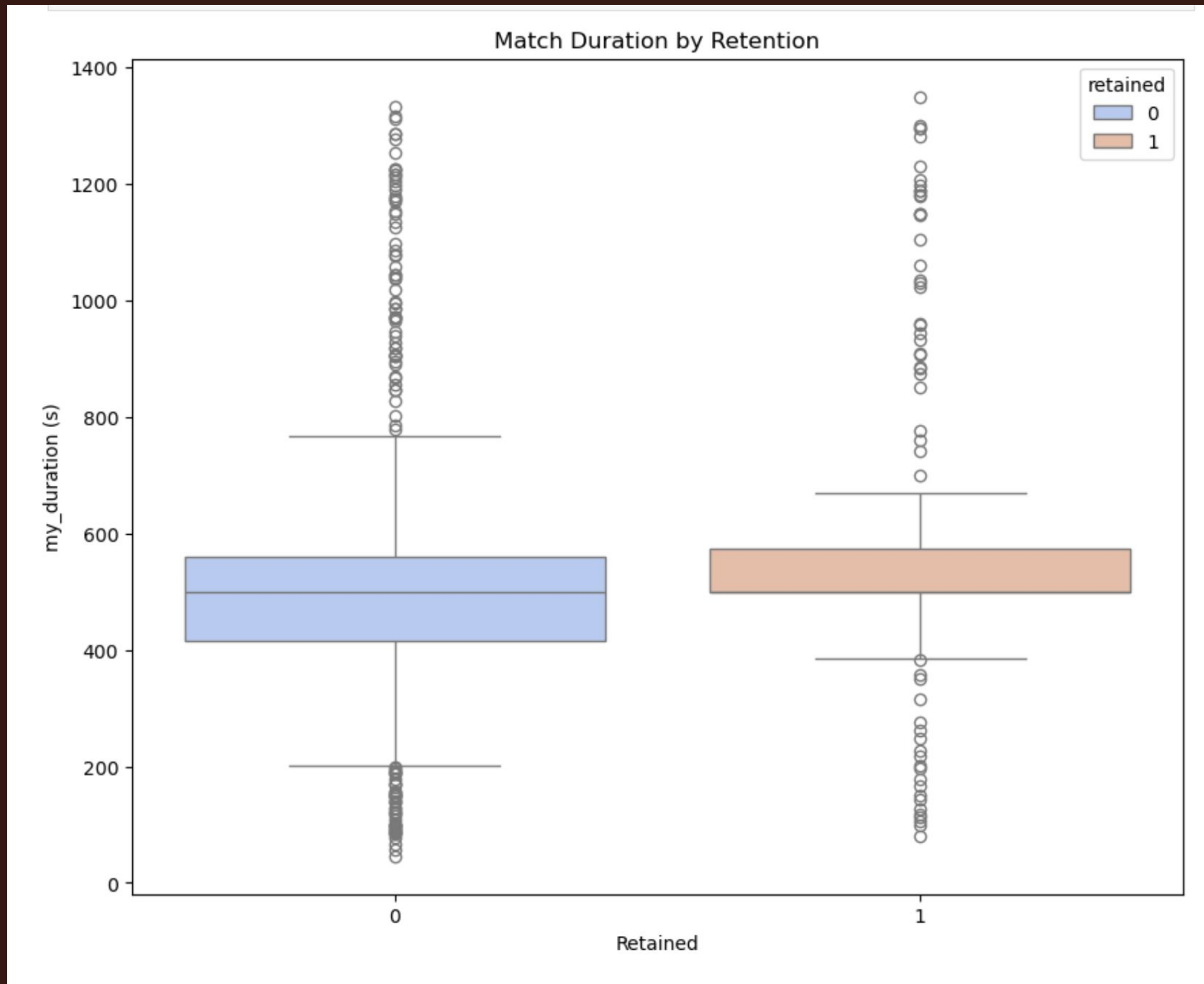
- **RETAINED:** 374 MATCHES (75 %)
- **CHURNED:** 125 MATCHES (25 %)

CORRELATION HEATMAP: EDA



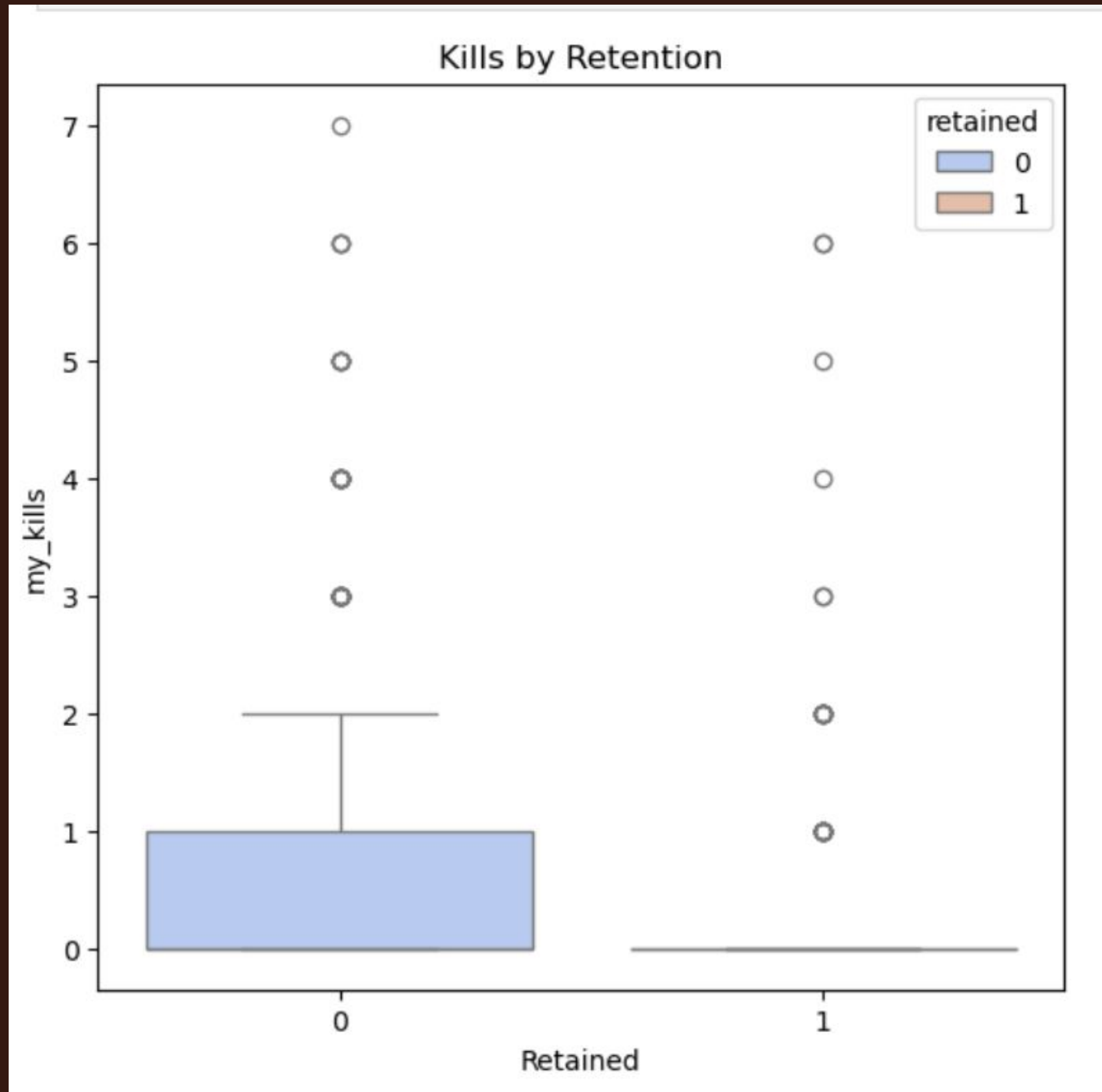
- Key features: match_duration, avg_damage, avg_kills, days_since_last_match
- match_duration vs avg_damage: $r \approx 0.71$
- my_revives correlations: $r \approx 0.42$ with my_knocks and $r \approx 0.25$ with match_duration

MATCH DURATION BY RETENTION: EDA



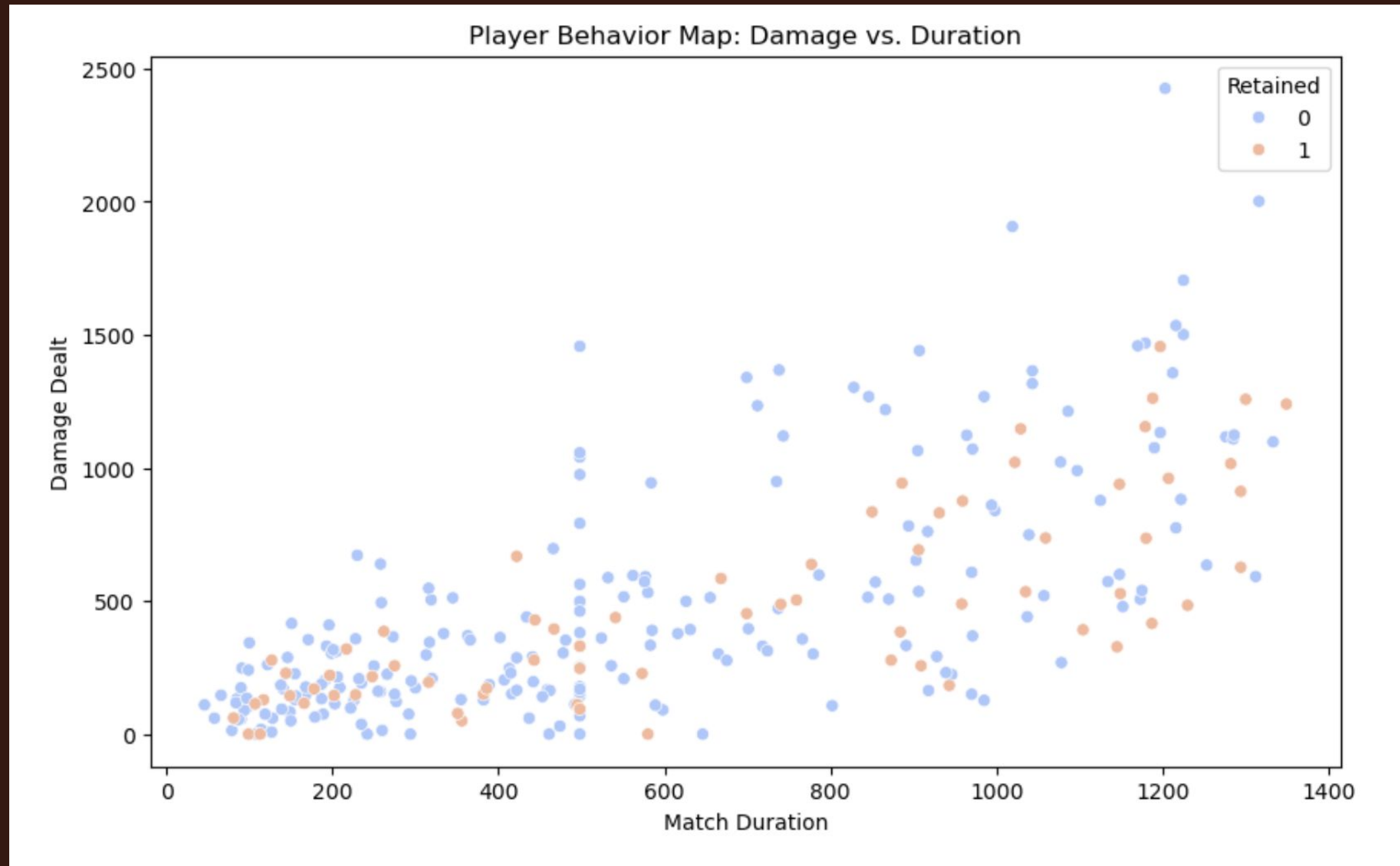
- **Retained (1): Higher median session lengths**
- **Churners (0): Concentrated at shorter durations**
- **Model input:** match_duration standardized as a core predictor

KILLS BY RETENTION: EDA



- **Higher median kills:
retained ≈ 1 vs. churners 0**
- **Model input: avg_kills
standardized as a core
predictor**

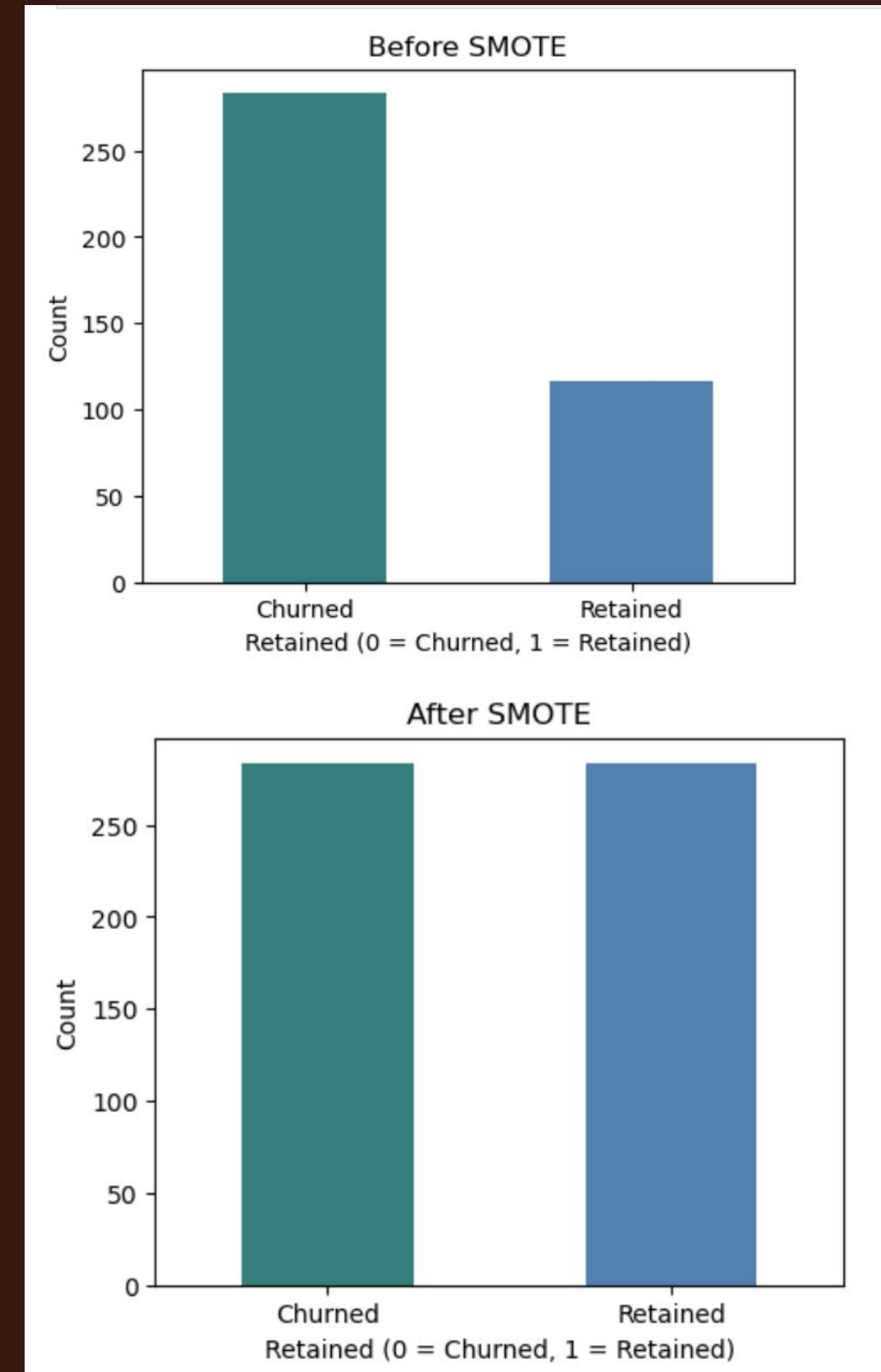
DAMAGE VS DURATION: EDA



- **Clusters: high-damage vs long-survival playstyles**
- **Features: avg_damage & match_duration**

PRE-MATCH TUNE-UP: PREPROCESSING

- **Train/Test Split:** stratified 80 / 20
(preserves churn ratio)
- **One-Hot Encoding:** 118 dummy columns
(map & legend)
- **Standard Scaling:** 4 numeric features \rightarrow 0 mean, 1 σ
- SMOTE lifted churners from $\sim 25\%$ to $\sim 45\%$
- **Final Matrix:** 566 train \times 114 features, 100 test \times 114



FINAL MODEL SHOOT-OUT

Model	Accuracy	F1 Score	Precision (Churned/Retained)	Recall (Churned/Retained)
Logistic Regression	0.83	0.73	0.91 / 0.68	0.85 / 0.79
Random Forest	0.90	0.83	0.93 / 0.83	0.93 / 0.83
XGBoost	0.91	0.84	0.93 / 0.86	0.94 / 0.83

Winner → XGBoost: Acc 0.91 | F1 0.84 | Recall (churn) 0.94

MISSION ACCOMPLISHED:

A CHURN-PREDICTION MODEL BUILT TO SCALE

- **Feature Matrix:** 114 columns after one-hot encoding & leakage checks
- **Models Tested:** Logistic Regression → Random Forest → **XGBoost**
- **Eval Split:** stratified 80 / 20 (566 train · 100 test)
- **Chosen Model:** **XGBoost** (default params, random_state = 42)
- **Test Metrics:** Acc 0.91 | F1 0.84 | Recall-churn 0.94

EARLY RADAR: CAN WE PREDICT CHURN A WEEK OUT?

- **Forecast achieved:** XGBoost flags churn 1 week ahead with 0.94 recall and 0.93 precision
- **Lead indicator pattern:** 5–6 idle-day gap signals high risk before the 7-day cut-off
- **Top drivers:** days_since_last_match, session_frequency, match_duration, avg_kills
- **Outcome:** reliable early-warning system EA can utilize the same pipeline for any new seasons logs.

