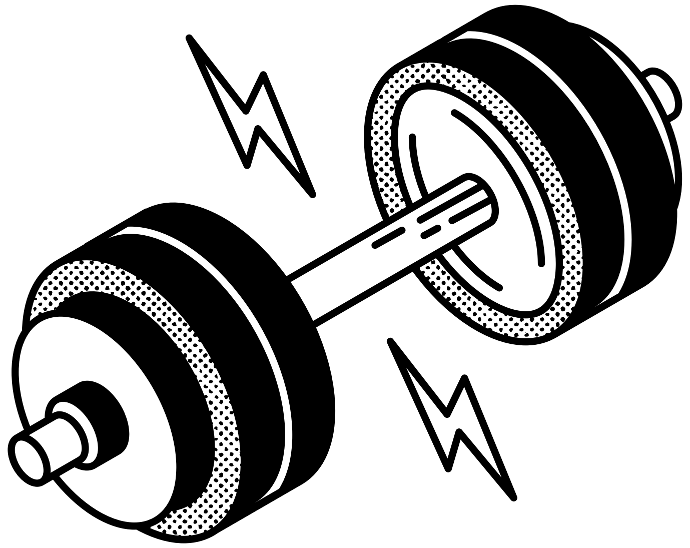
Gympass: churn prediction



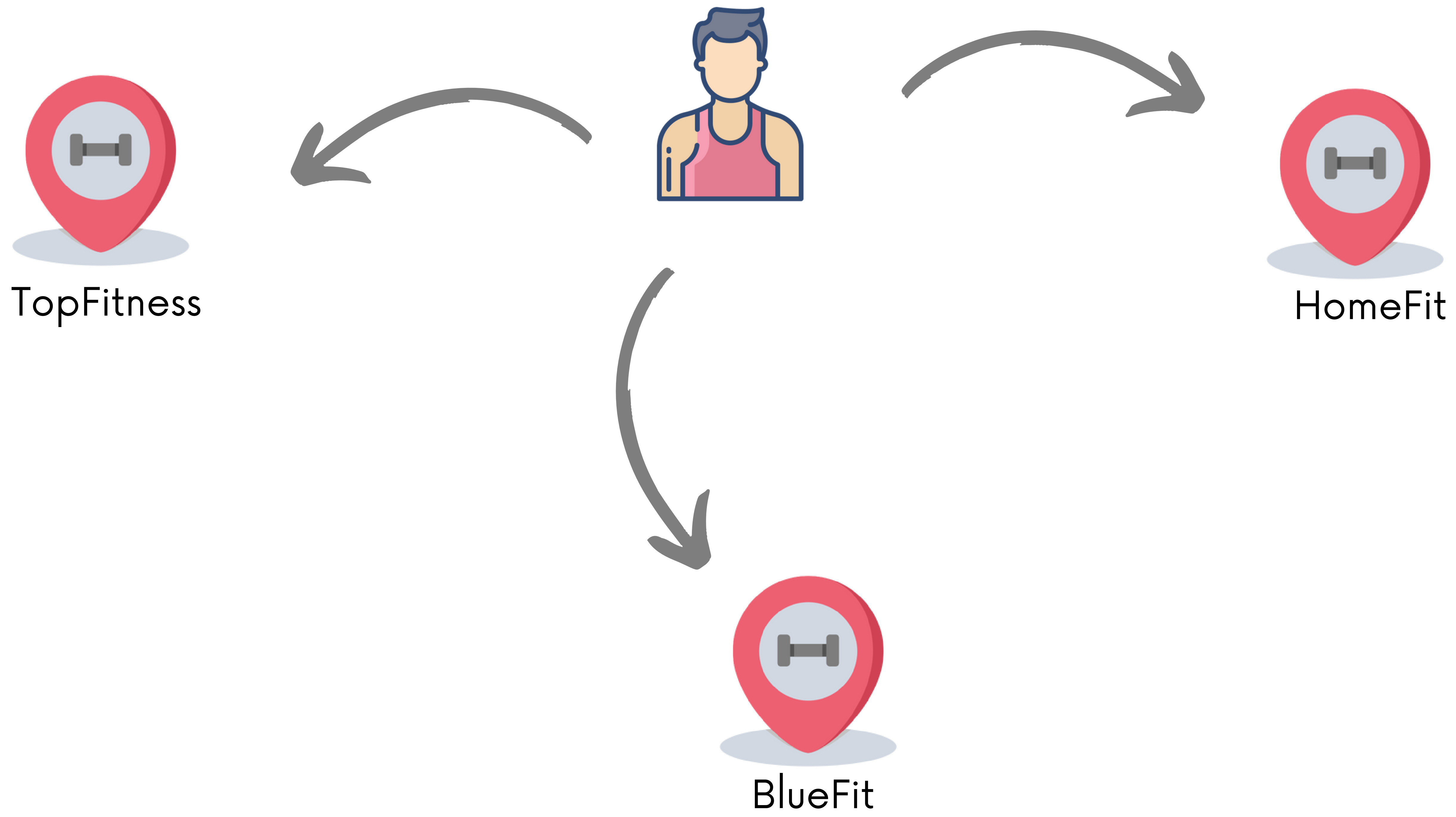
Kevin Takano

Introducing Ricardo

Ricardo

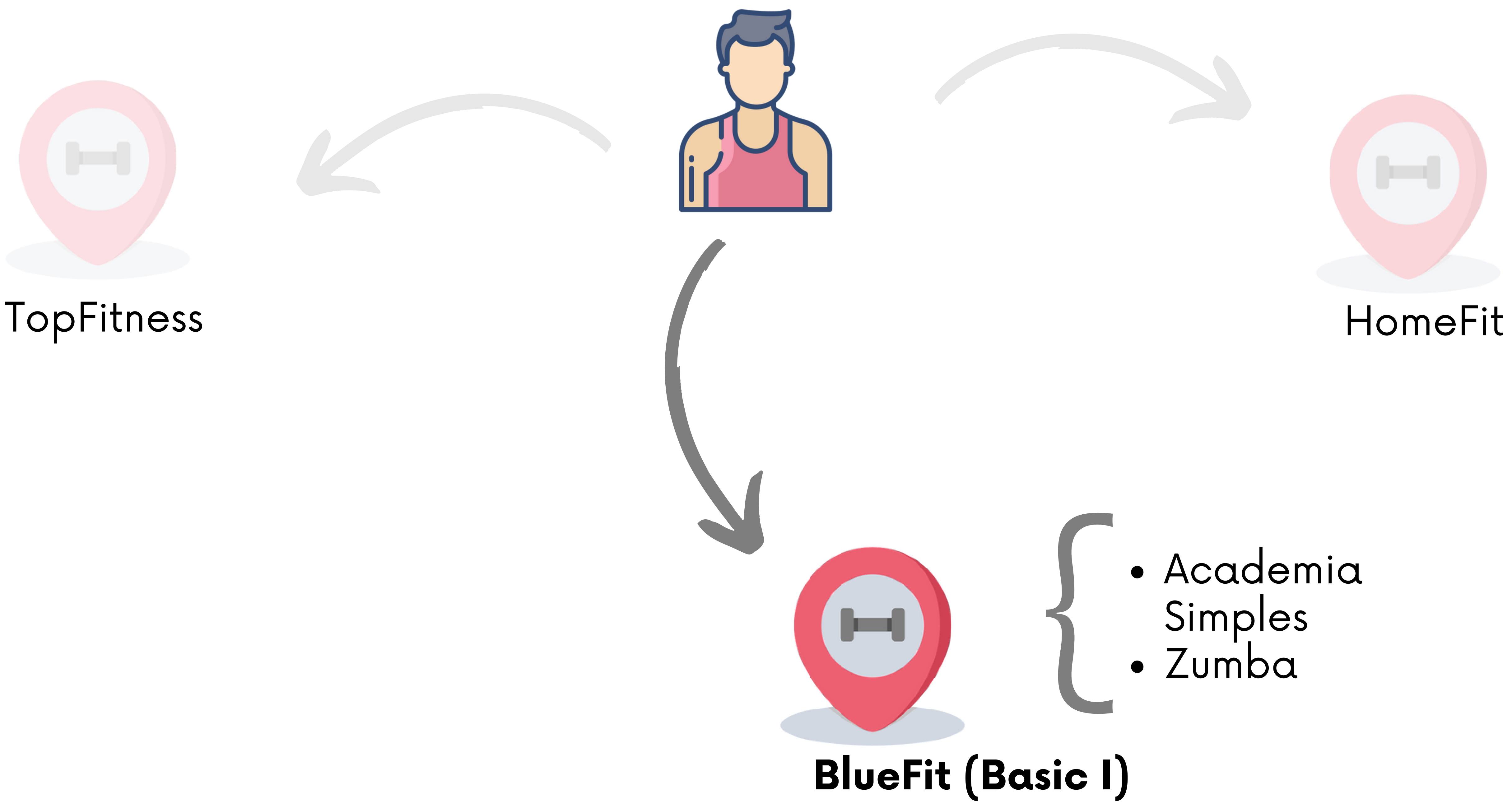
- He wants to start a fitness lifestyle.
- Your company has a contract with **Gympass.**

What is gympass?



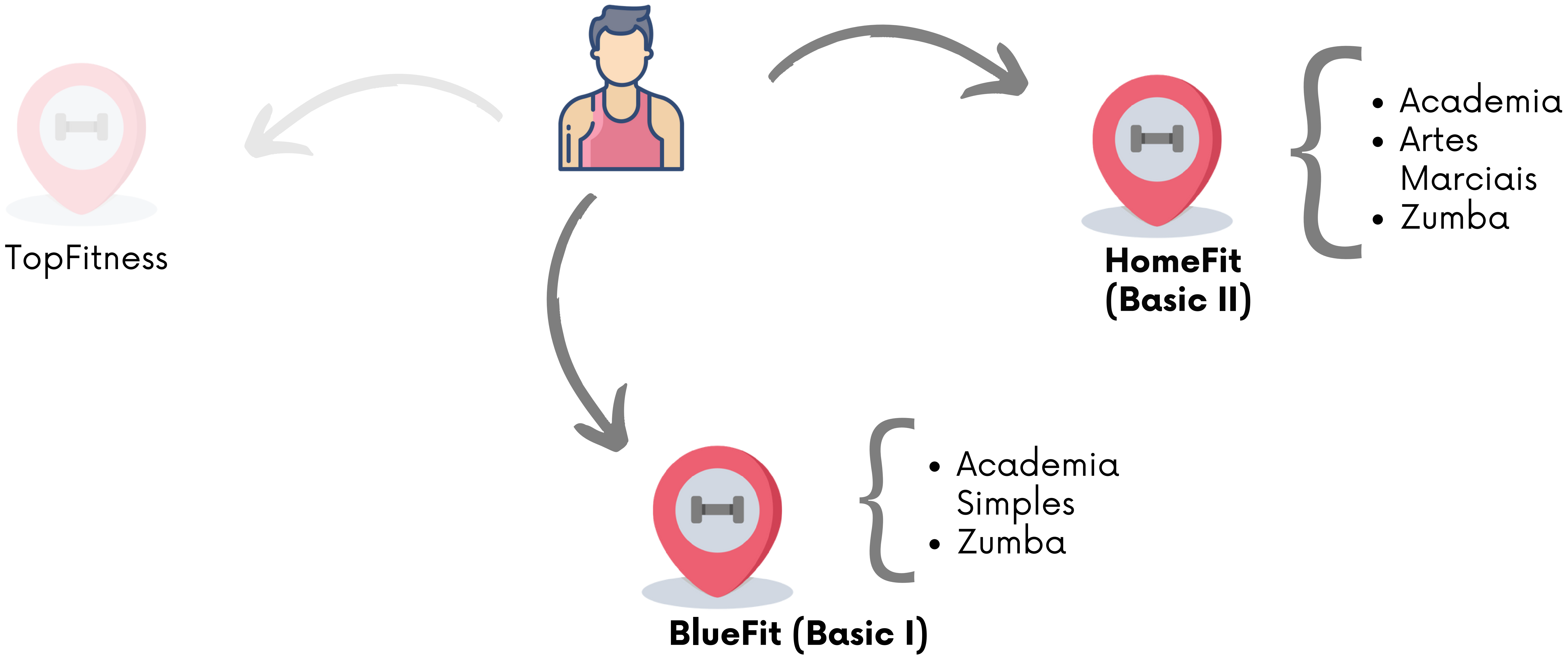
Gympass Plans

Basic I



Gympass Plans

Basic II



Gympass Plans

Silver

TopFitness (Silver)

- Academia
- Artes Marciais
- Yoga
- Pilates
- Zumba

HomeFit (Basic II)

- Academia
- Artes Marciais
- Zumba

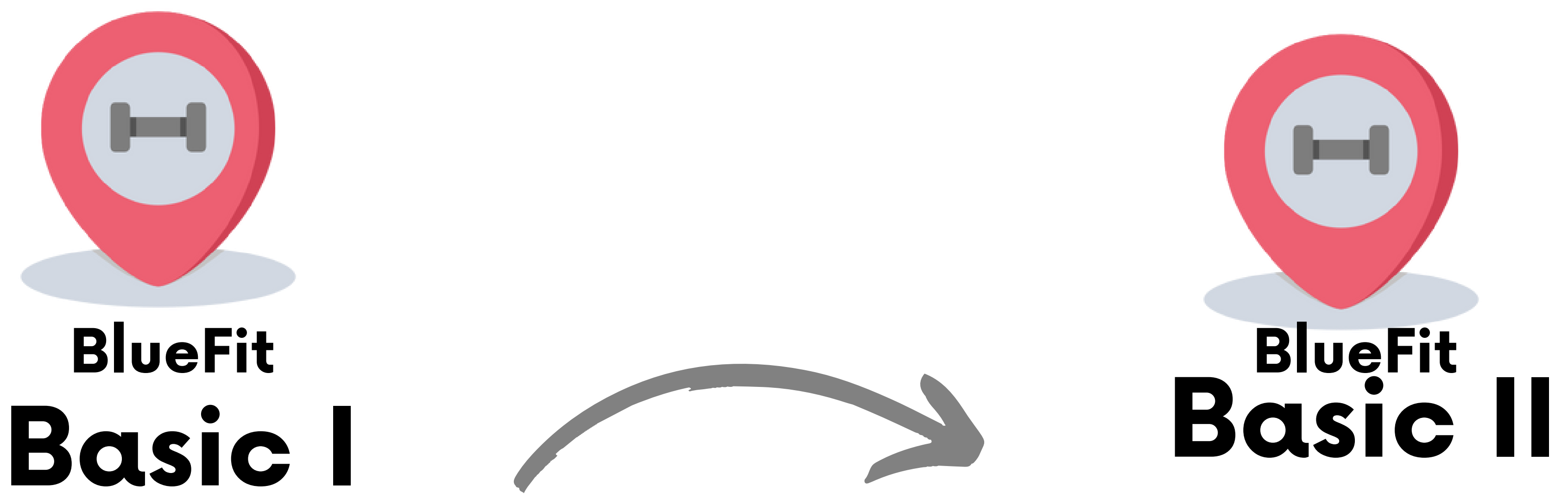
BlueFit (Basic I)

- Academia Simples
- Zumba

Gympass Plans

- Basic I: 39.90
- Basic II: 59.90
- Silver: 99.90
- Silver+: 149.90

Business problem: Sometimes a Gym wants to **upgrade** an plan



However, it's not so easy

After an gym uptier:

Keep the plan

Maybe he did not use the upgraded gym.
Or prefers to go to another gym

After an gym uptier:

Keep the plan

Maybe he did not use the upgraded gym.
Or prefers to go to another gym

Upgrade the plan

He is a loyal gymuser

After an gym uptier:

Keep the plan

Maybe he did not use the upgraded gym.
Or prefers to go to another gym

Upgrade the plan

He is a loyal gymuser

Churn the plan

He can thinks the plan is too expensive / maybe he can pay directly in the gym

In this task, our goal is to predict which **gyms are suitable** for an upgrade

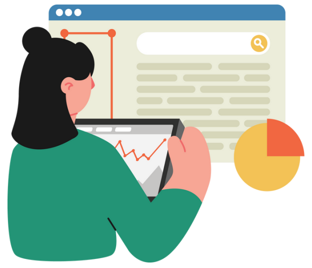
- We need to **predict user churn.**

Strategy

The main ideia was to **classify each user** as **churn or non-churn user,** and **after** calculate the **total churn users by gym**

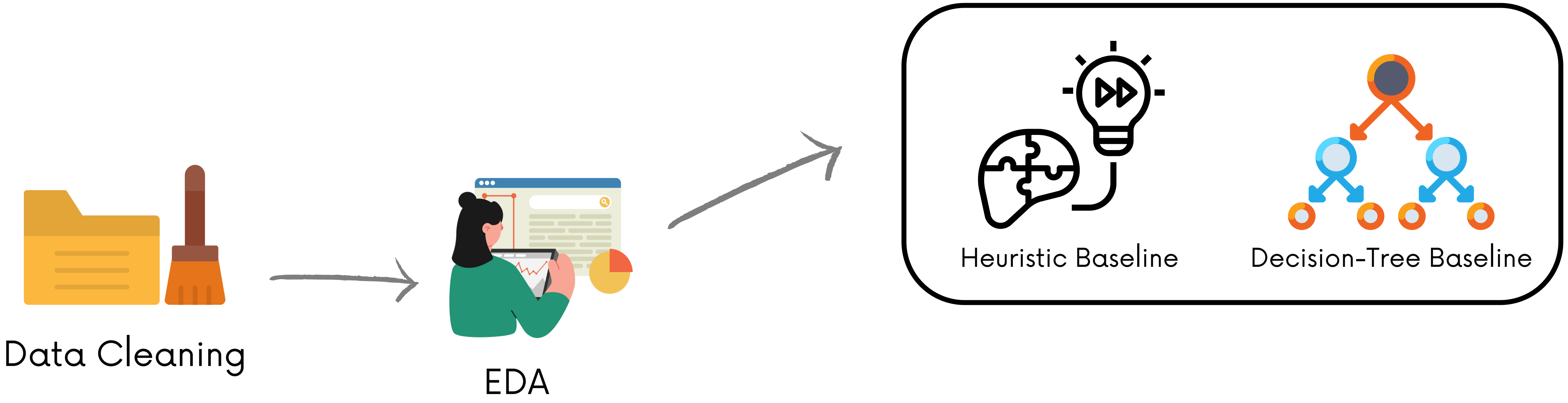
Overall Strategy

Data Cleaning

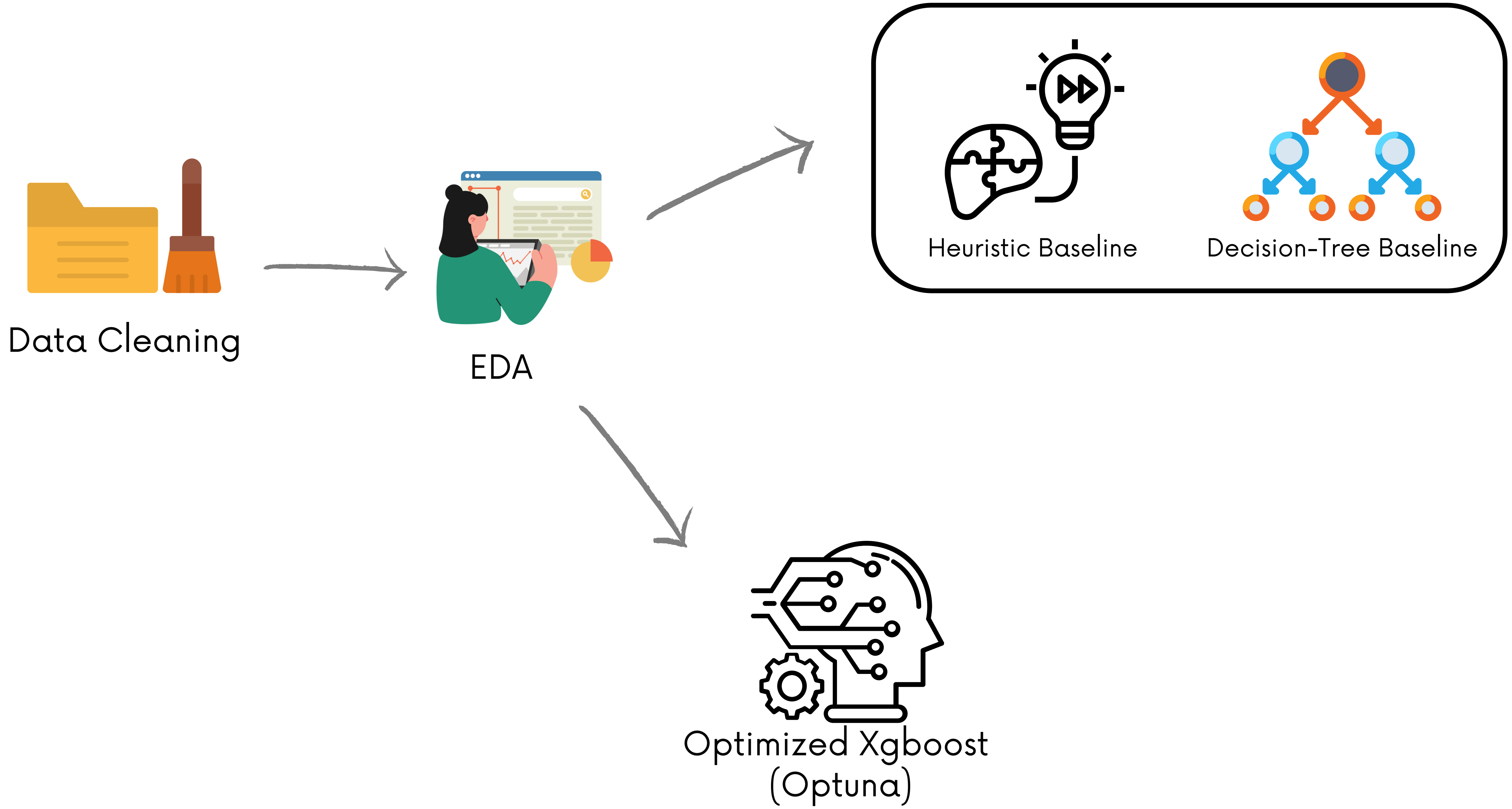


EDA

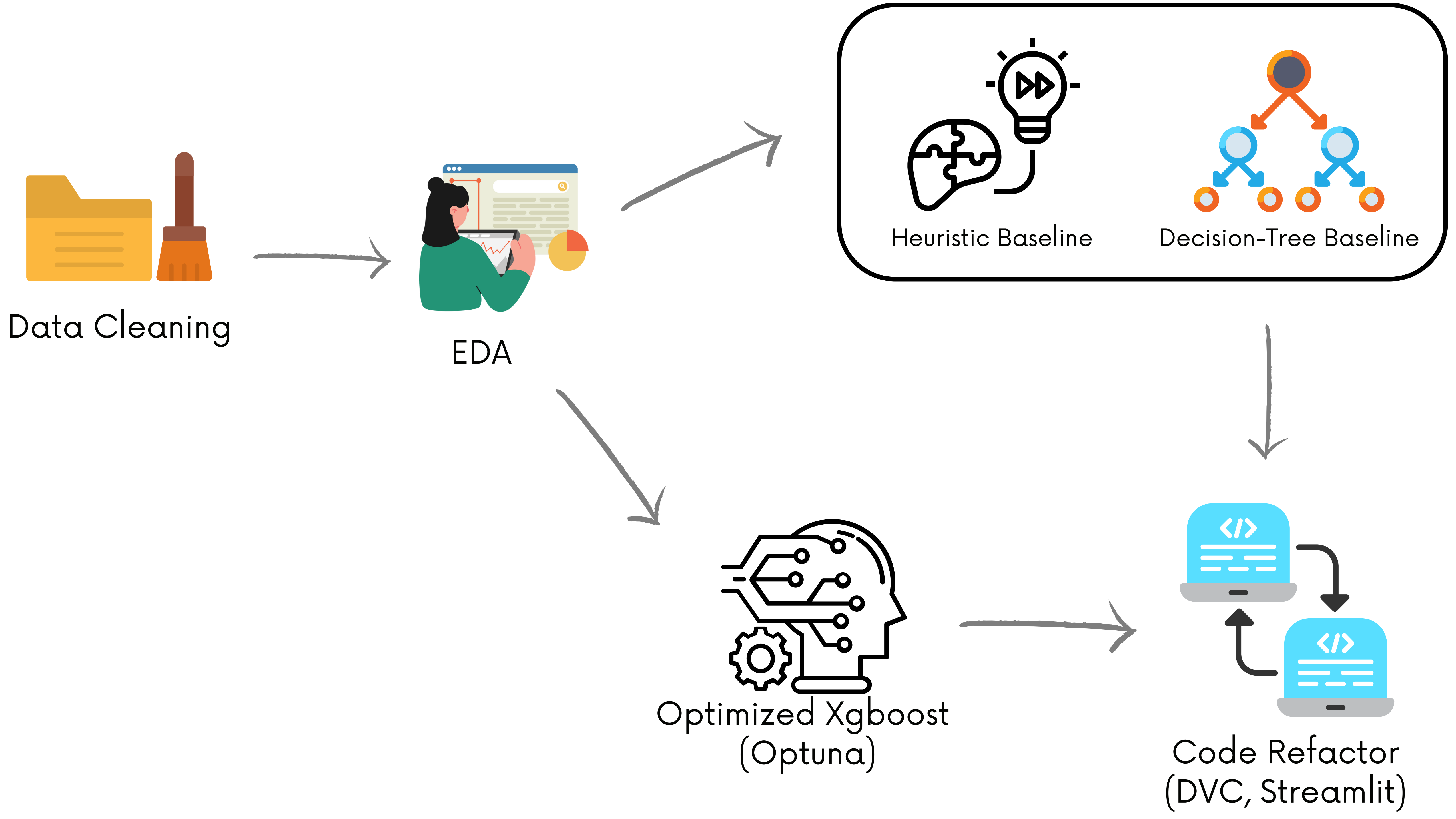
Overall Strategy



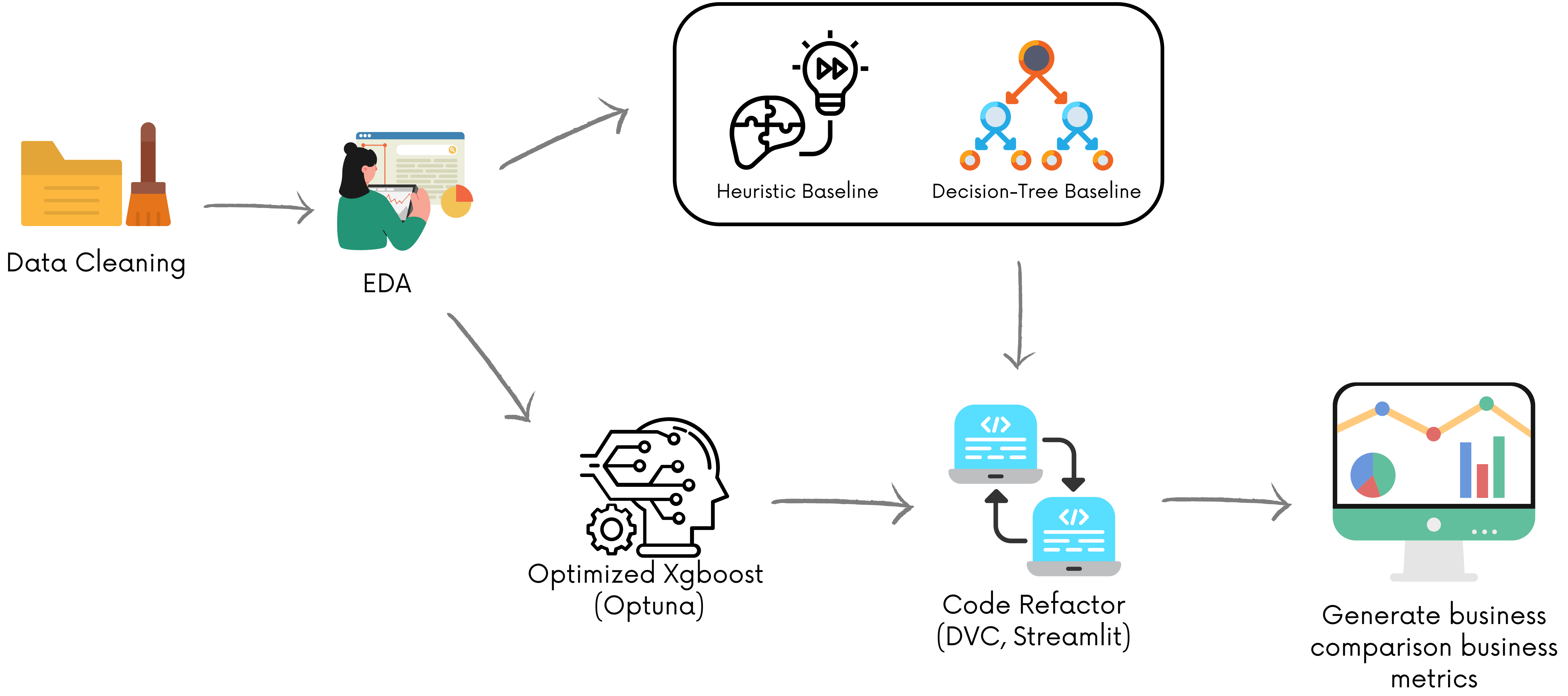
Overall Strategy



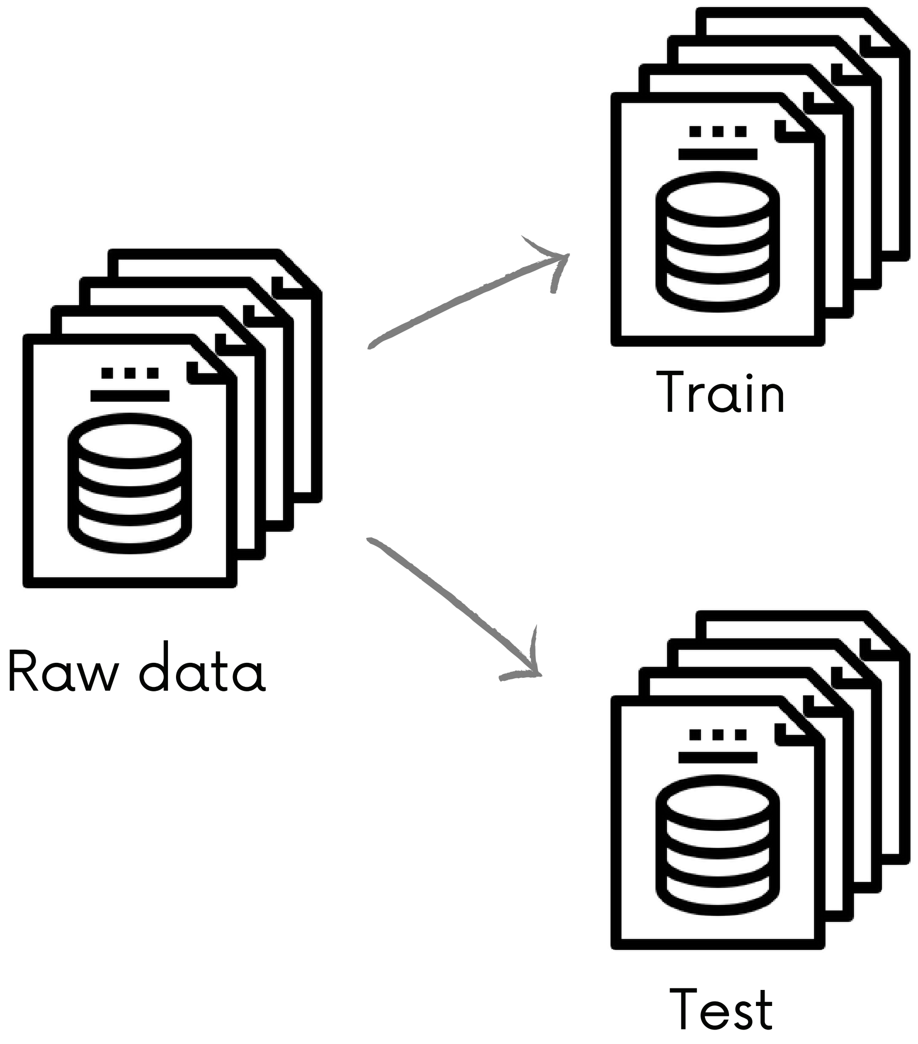
Overall Strategy



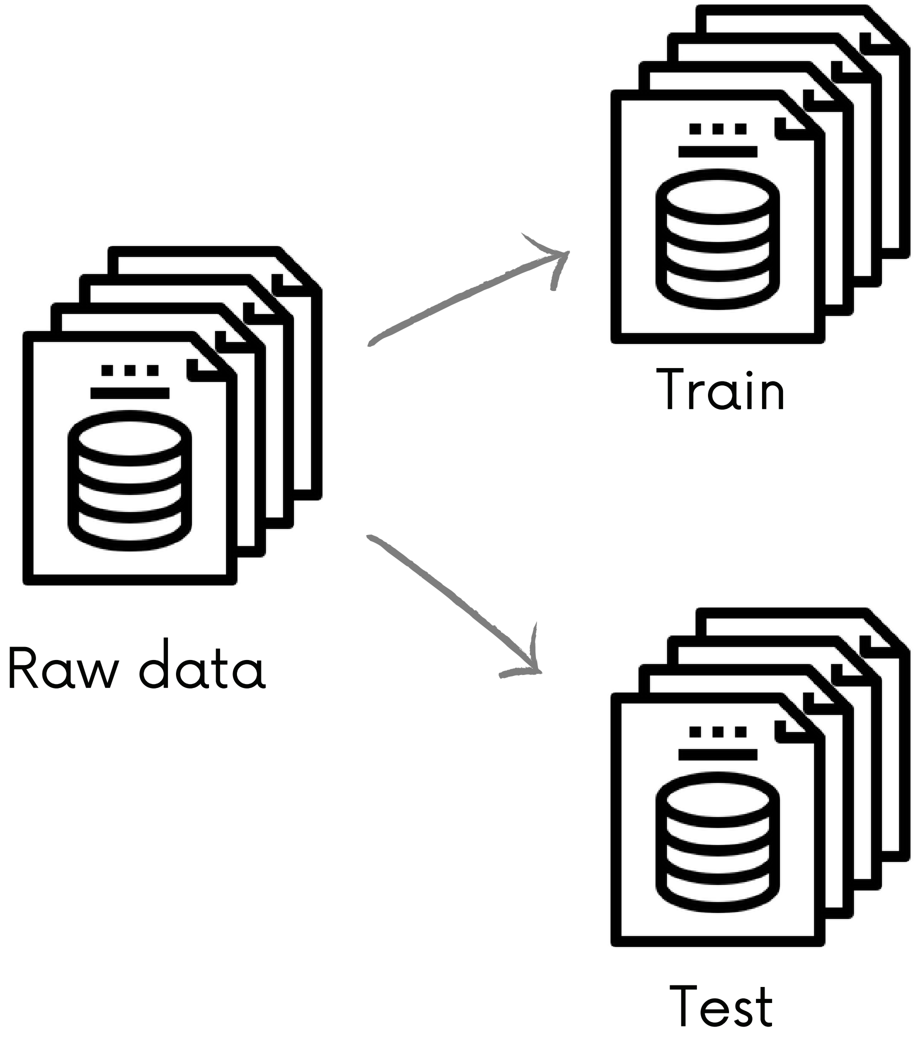
Overall Strategy



Data Split and Validation

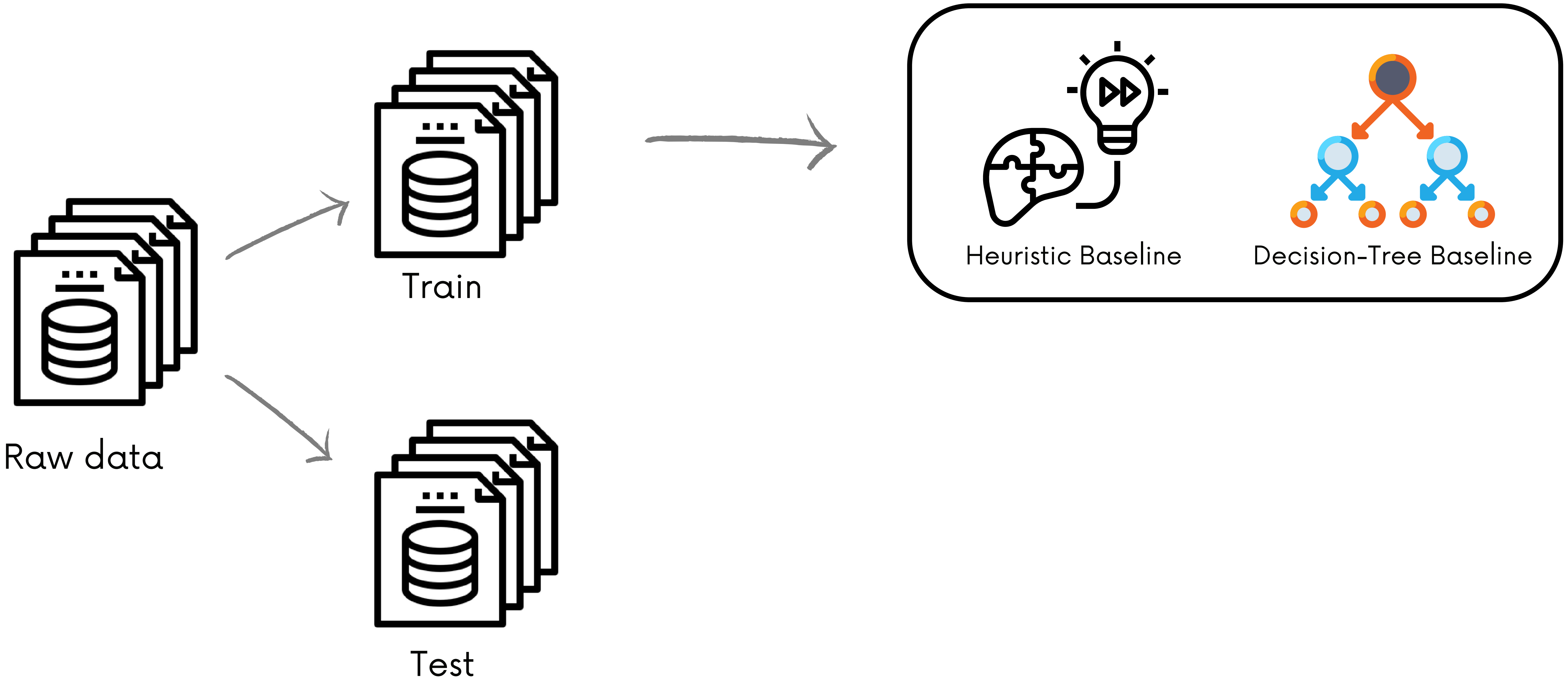


Data Split and Validation



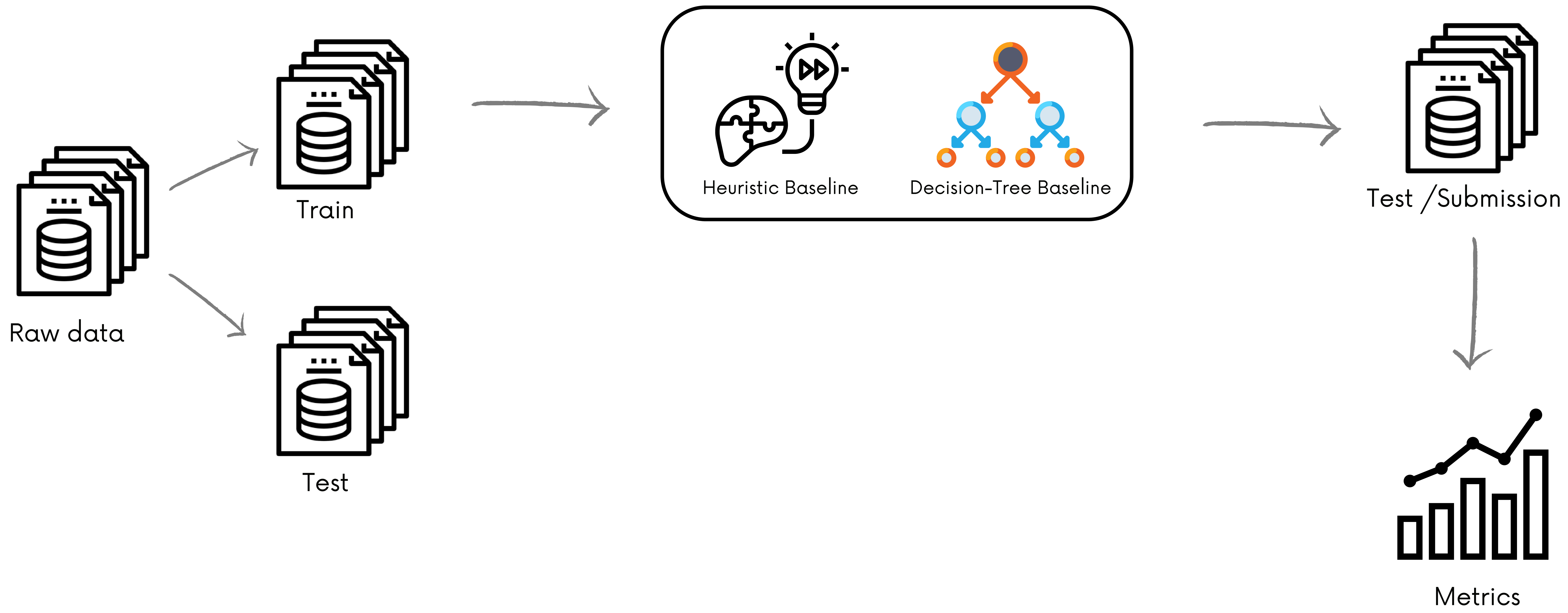
* I splitted considering gym indexes, not users

Data Split and Validation



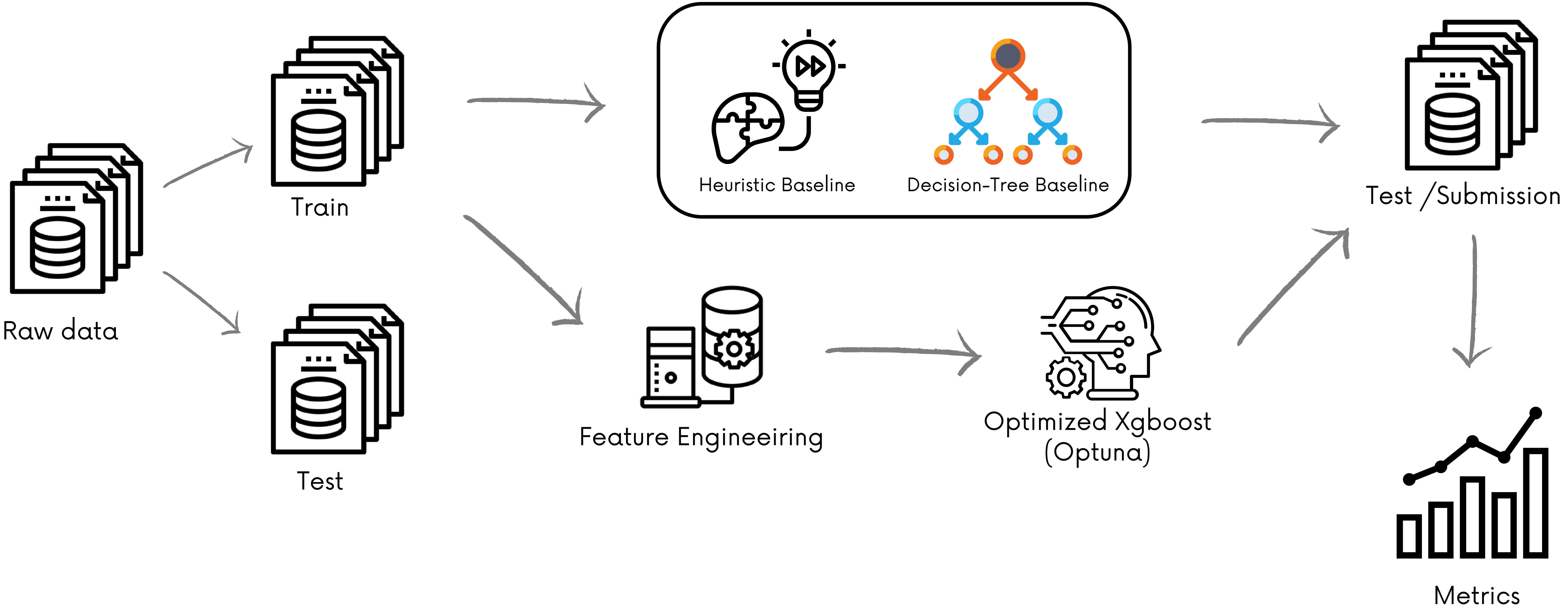
* I splitted considering gym indexes, not users

Data Split and Validation



* I splitted considering gym indexes, not users

Data Split and Validation

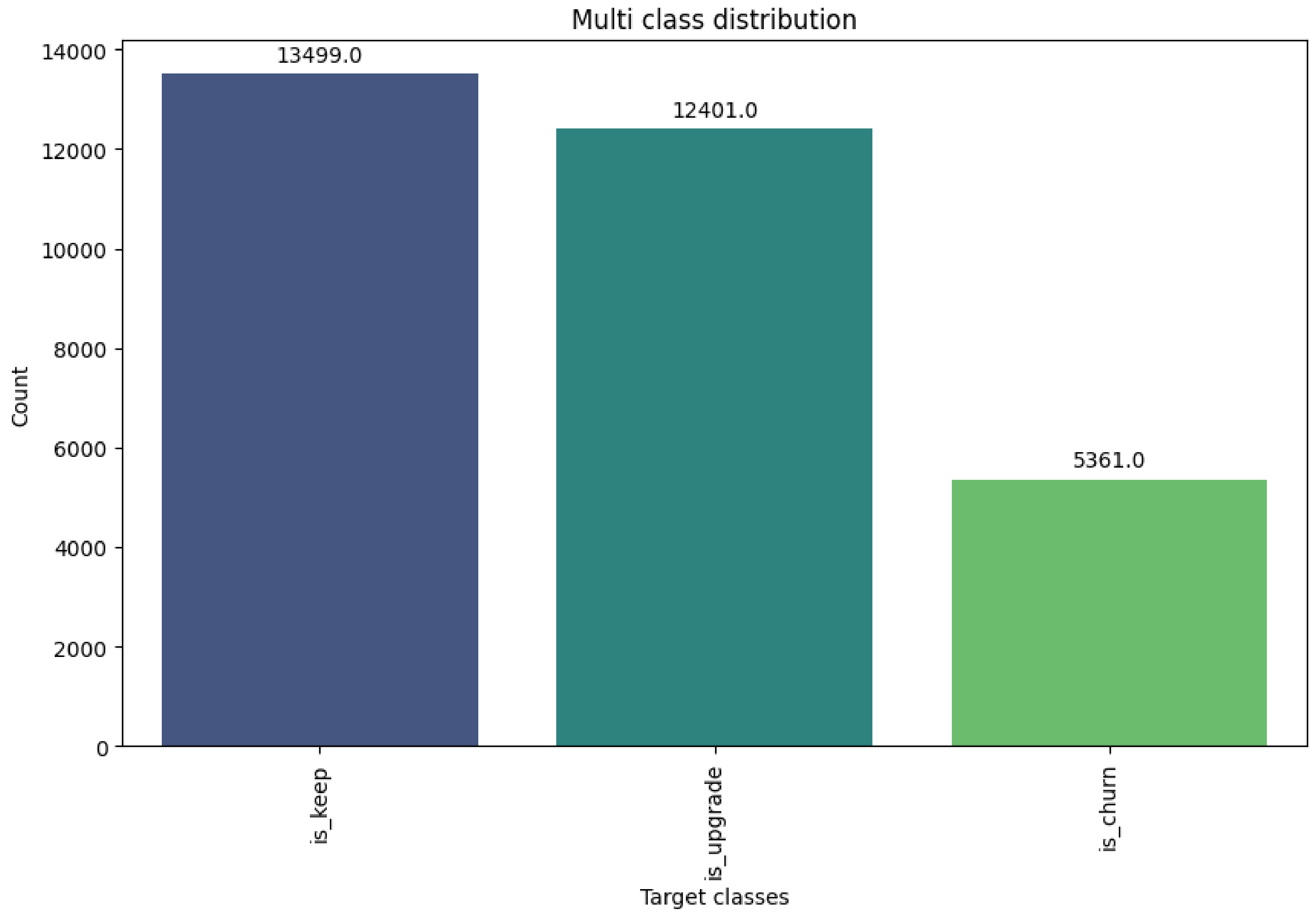


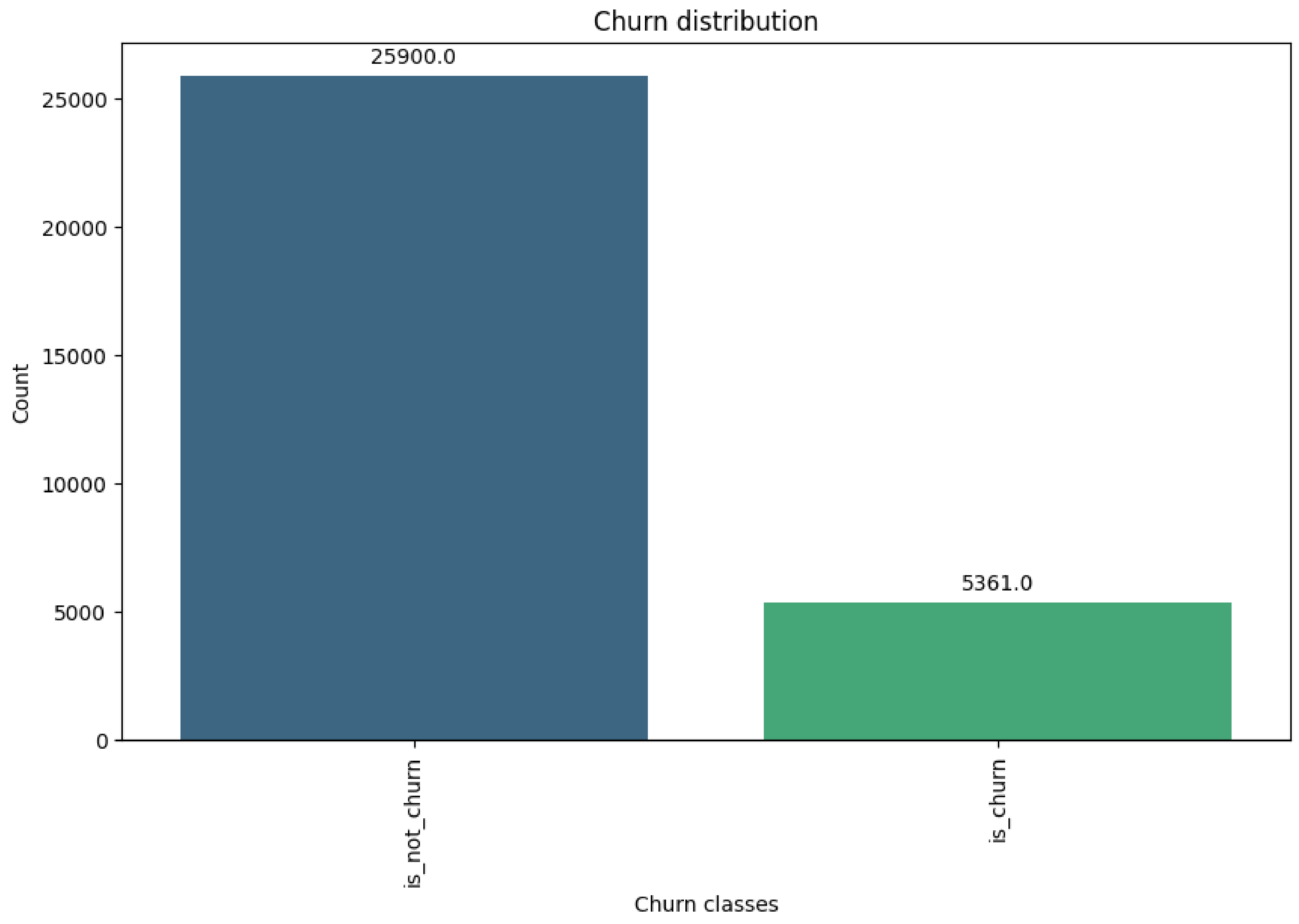
* I splitted considering gym indexes, not users

Exploratory Data Analysis

The most found important patterns

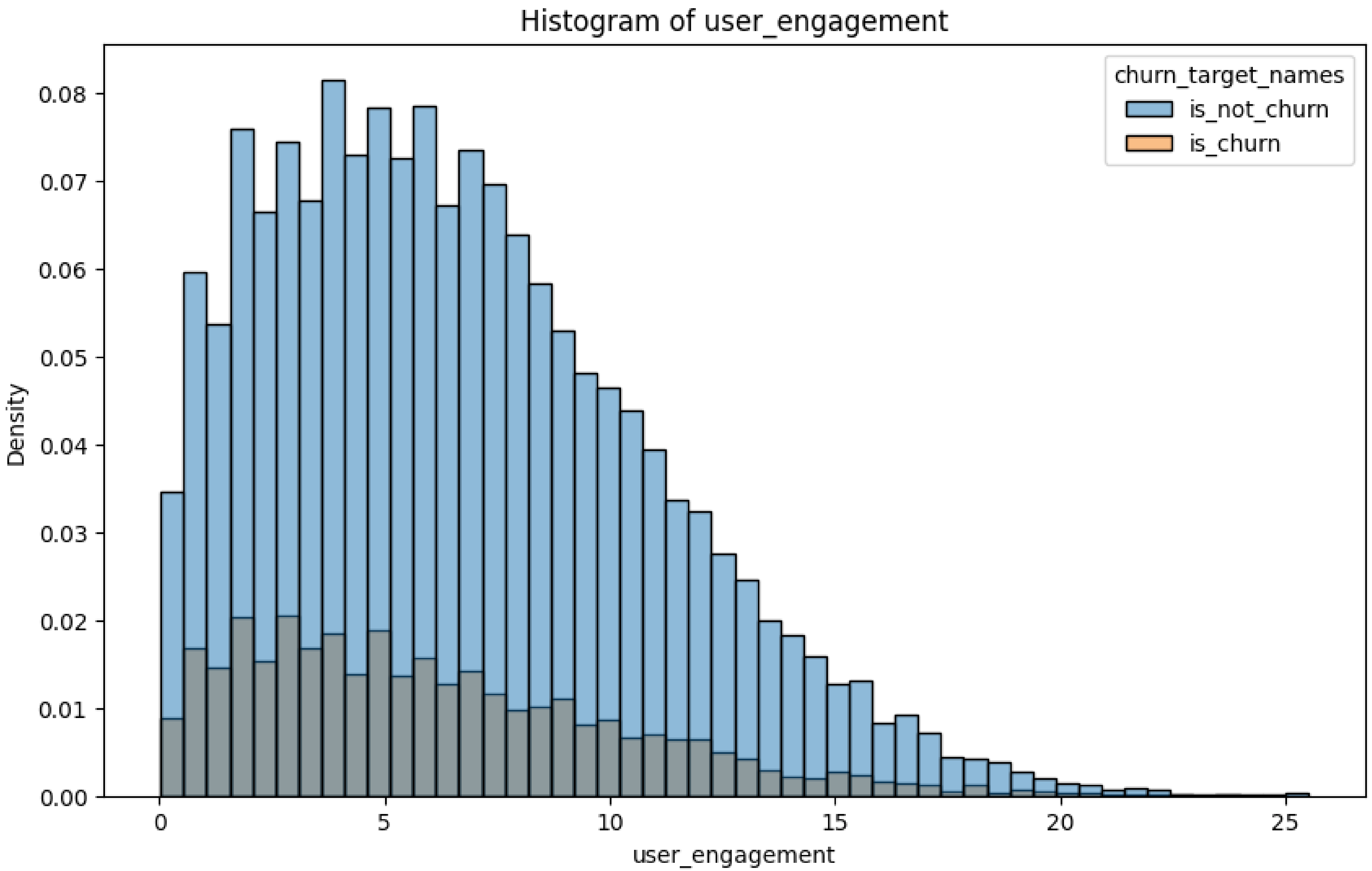
Target Distribution





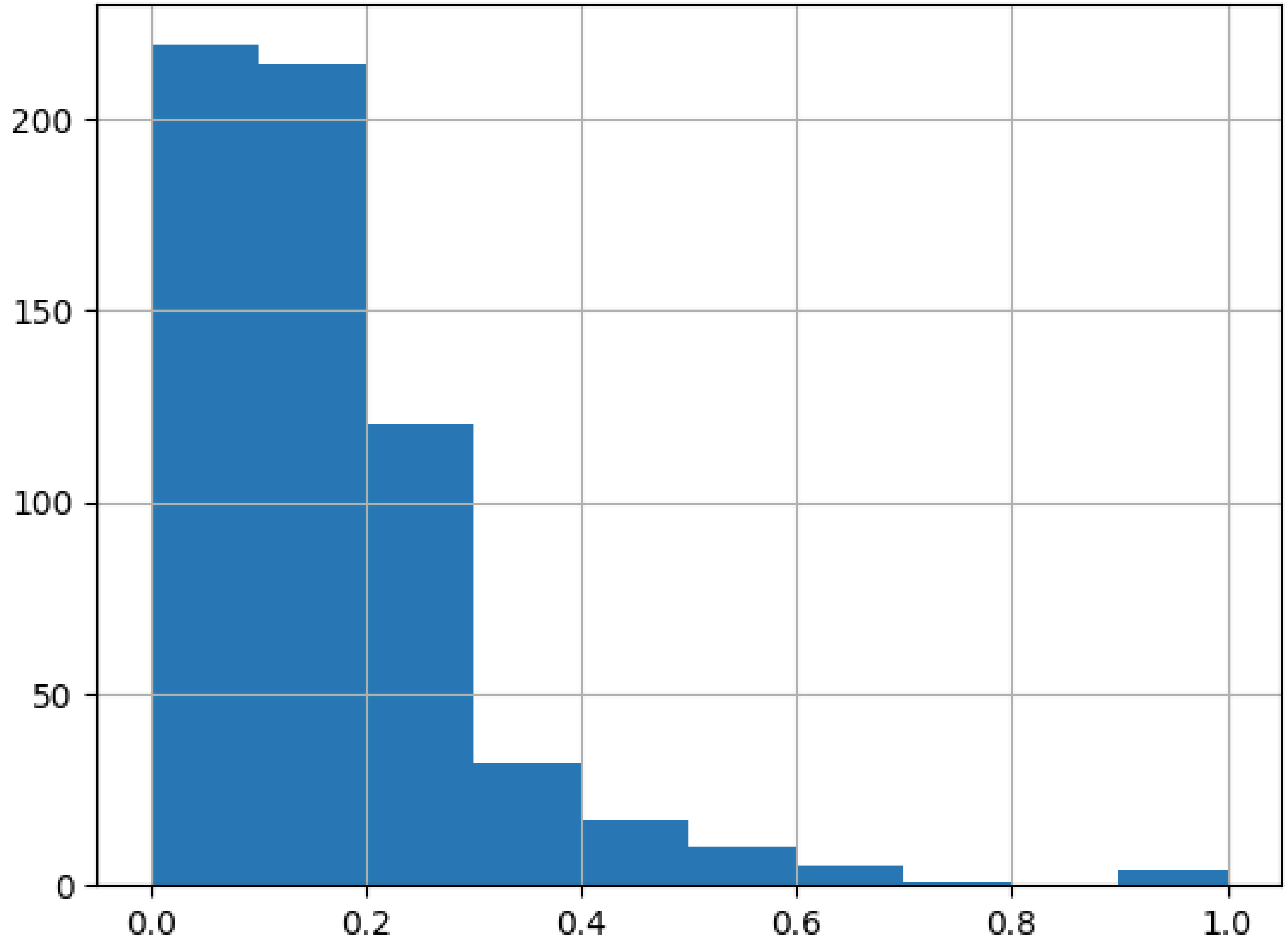
User Engagement Distribution

user_engagement = user_lifetime_visits / user_billings



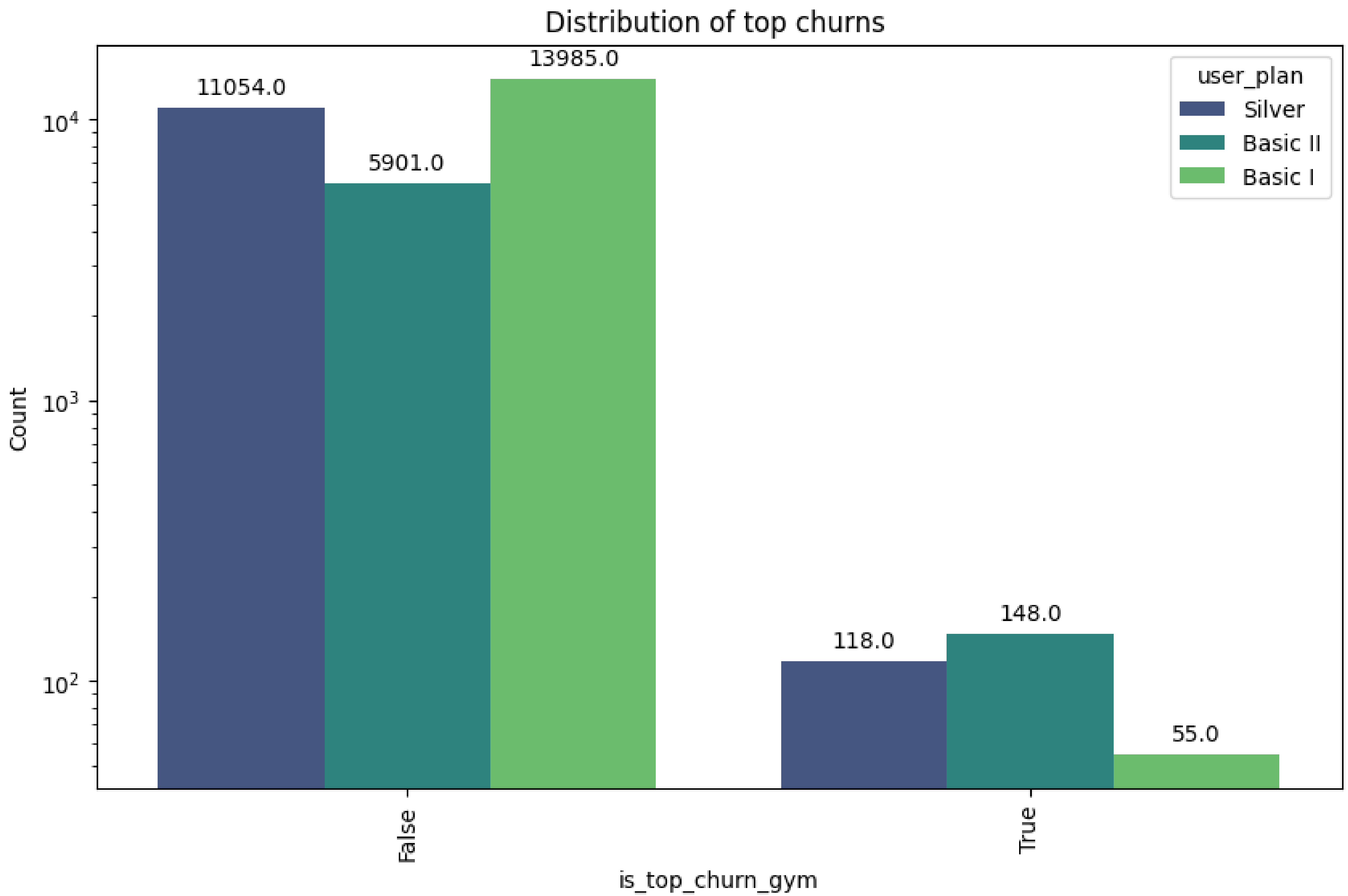
Churn rate gym distribution

churn_rate = rate of lost users



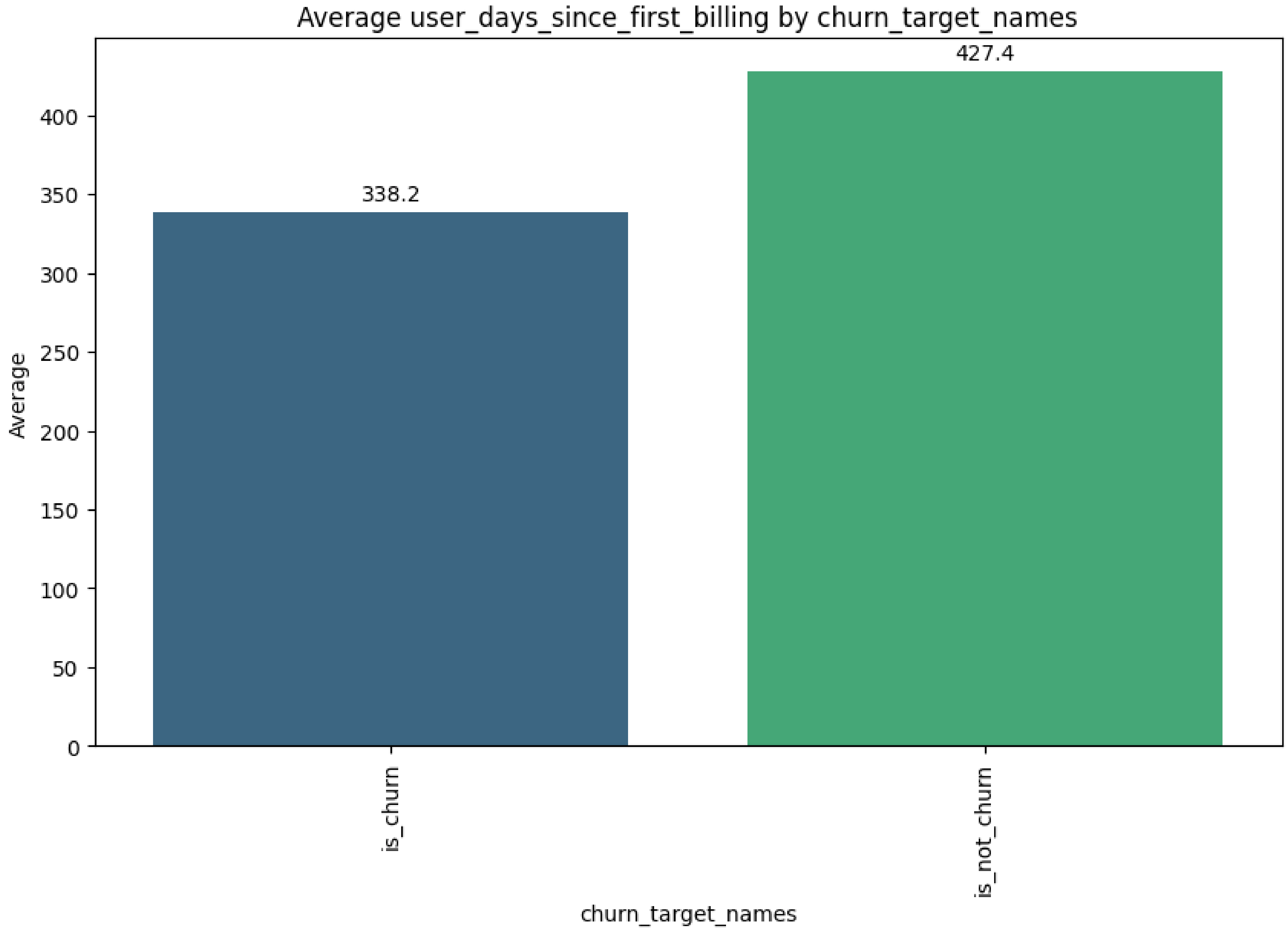
Top churn gym user plan distribution

Top churn gym is a gym that has churn_rate > 0.5

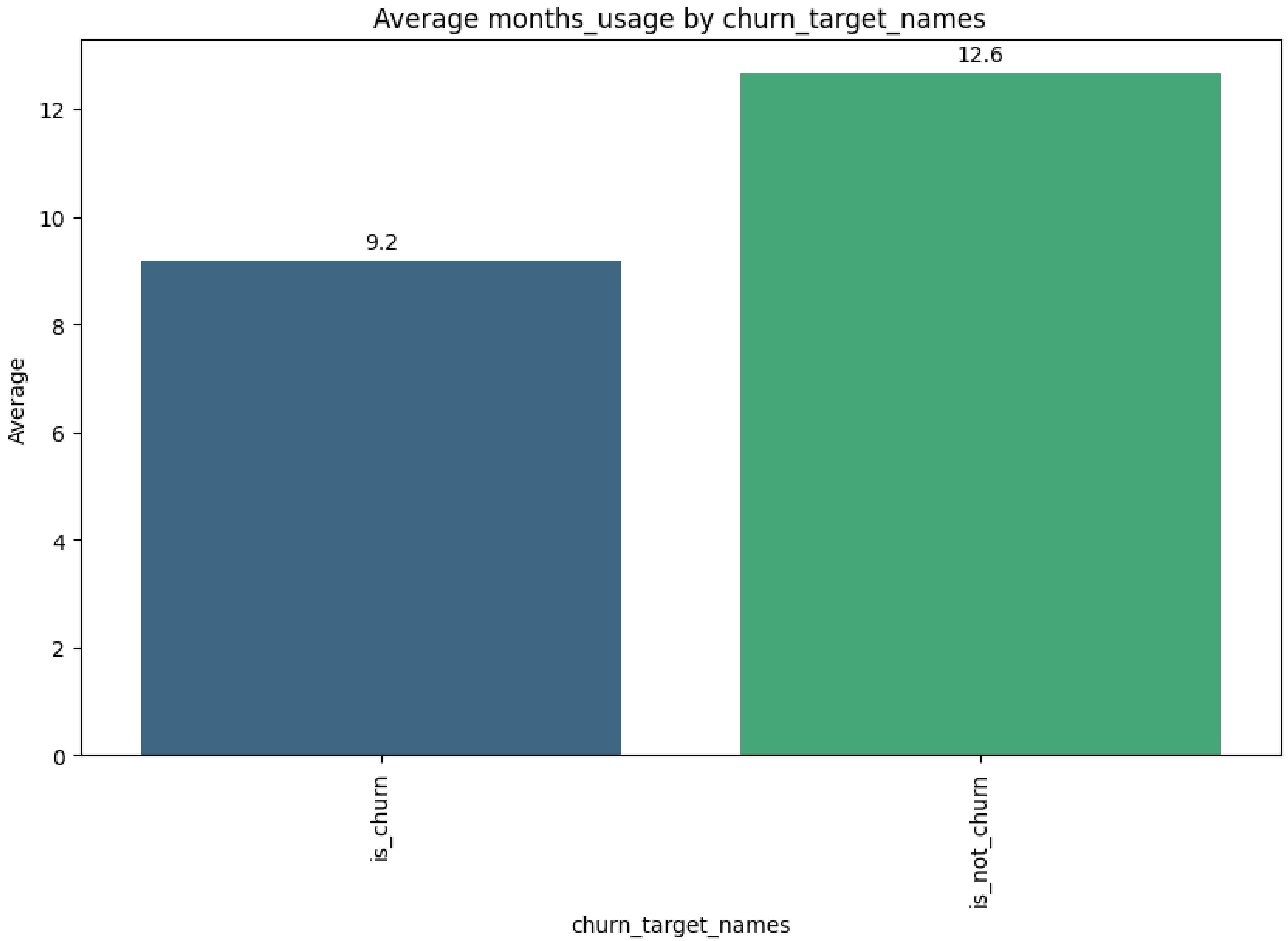


Average user_days_since_first_billing by target

Frequency and Recency affects churn

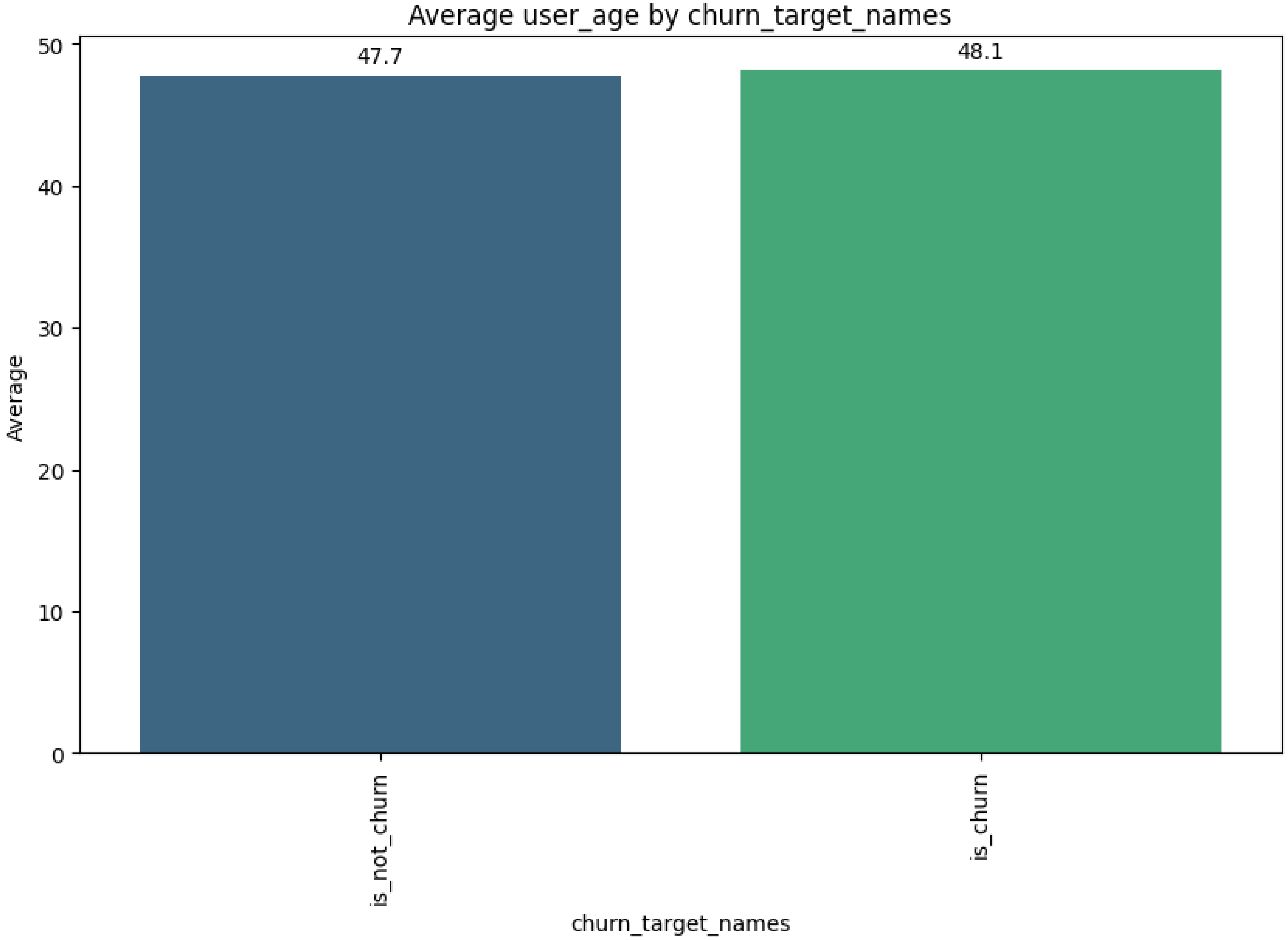


Number of user_billings (months_usage) affects churn

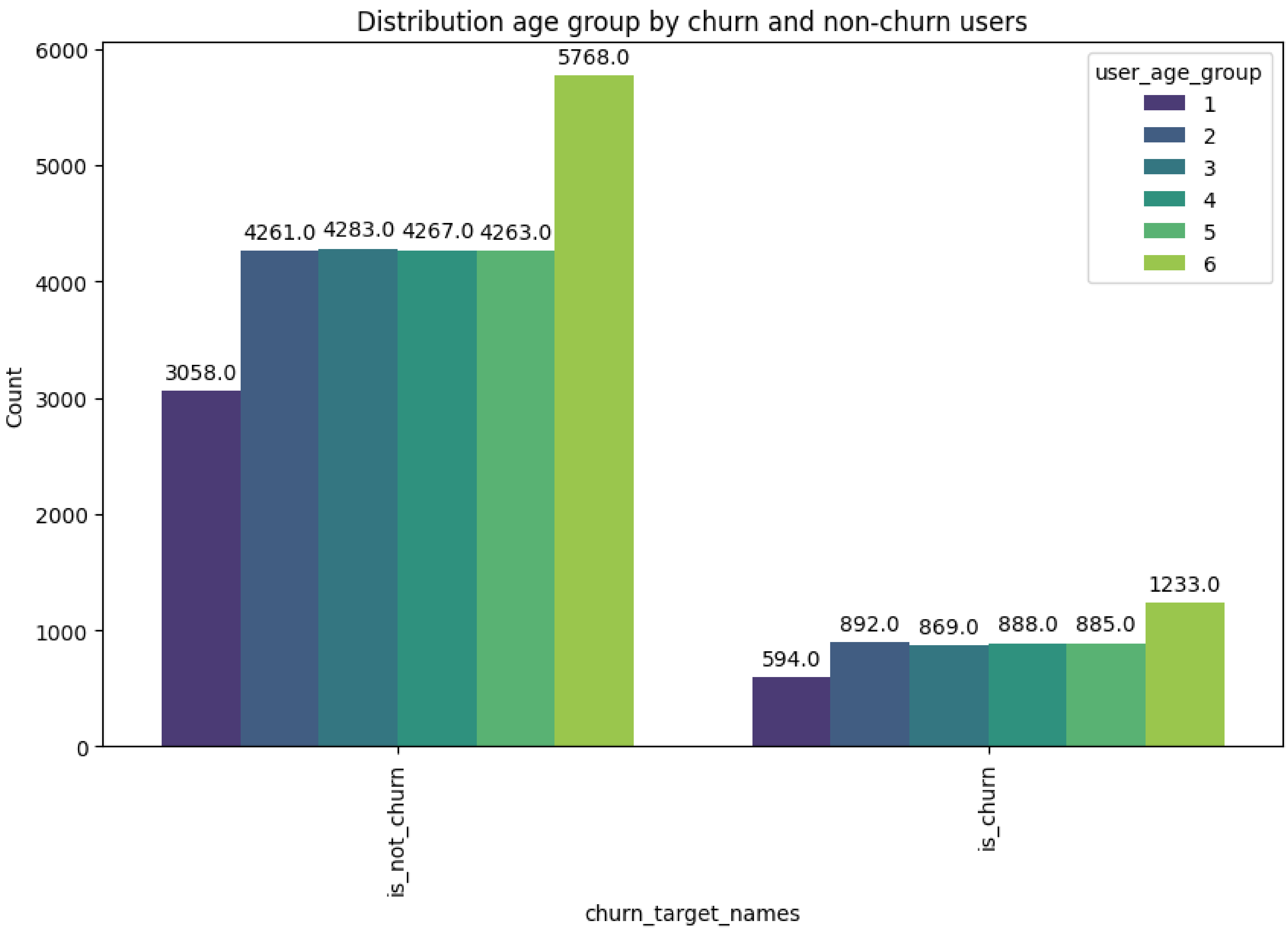


Average user_age by target

Looks like not help in churn information



Average user_age_group by target



Most important assumptions for modelling

1 I assumed the "**applications**" file contains information of **all users** of each gym

2 Loyalty affects churn

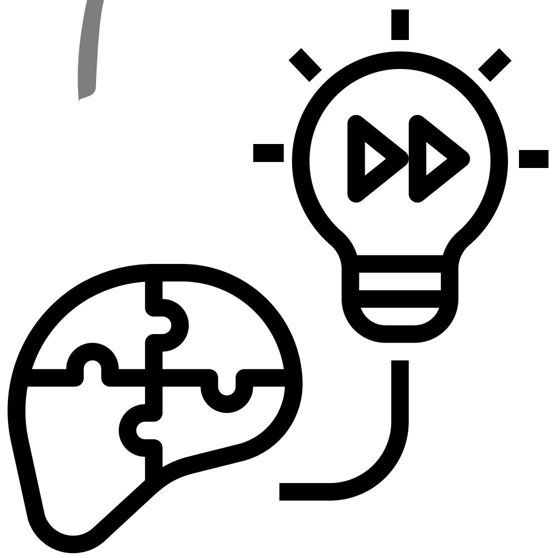
3 Recency and Frequency affects churn

4 User characteristics affects (e.g user age) churns

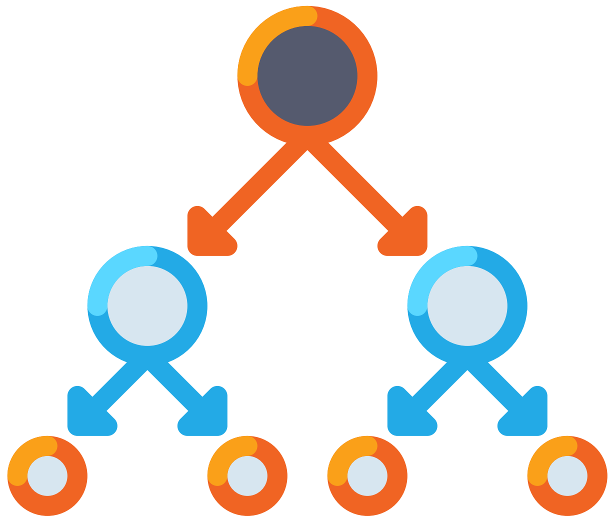
Baselines

Baselines

Think in model
business



Heuristic Baseline

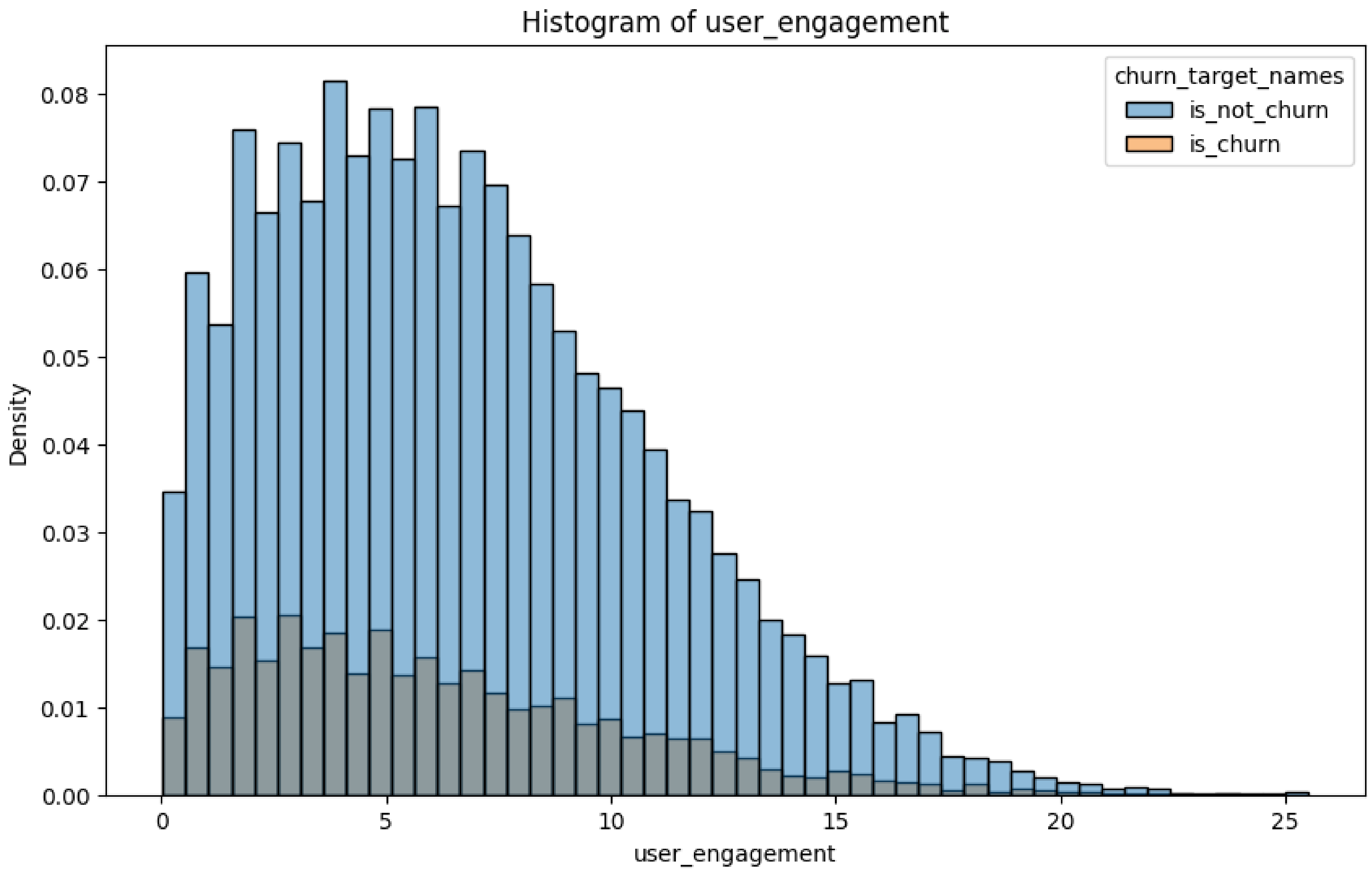


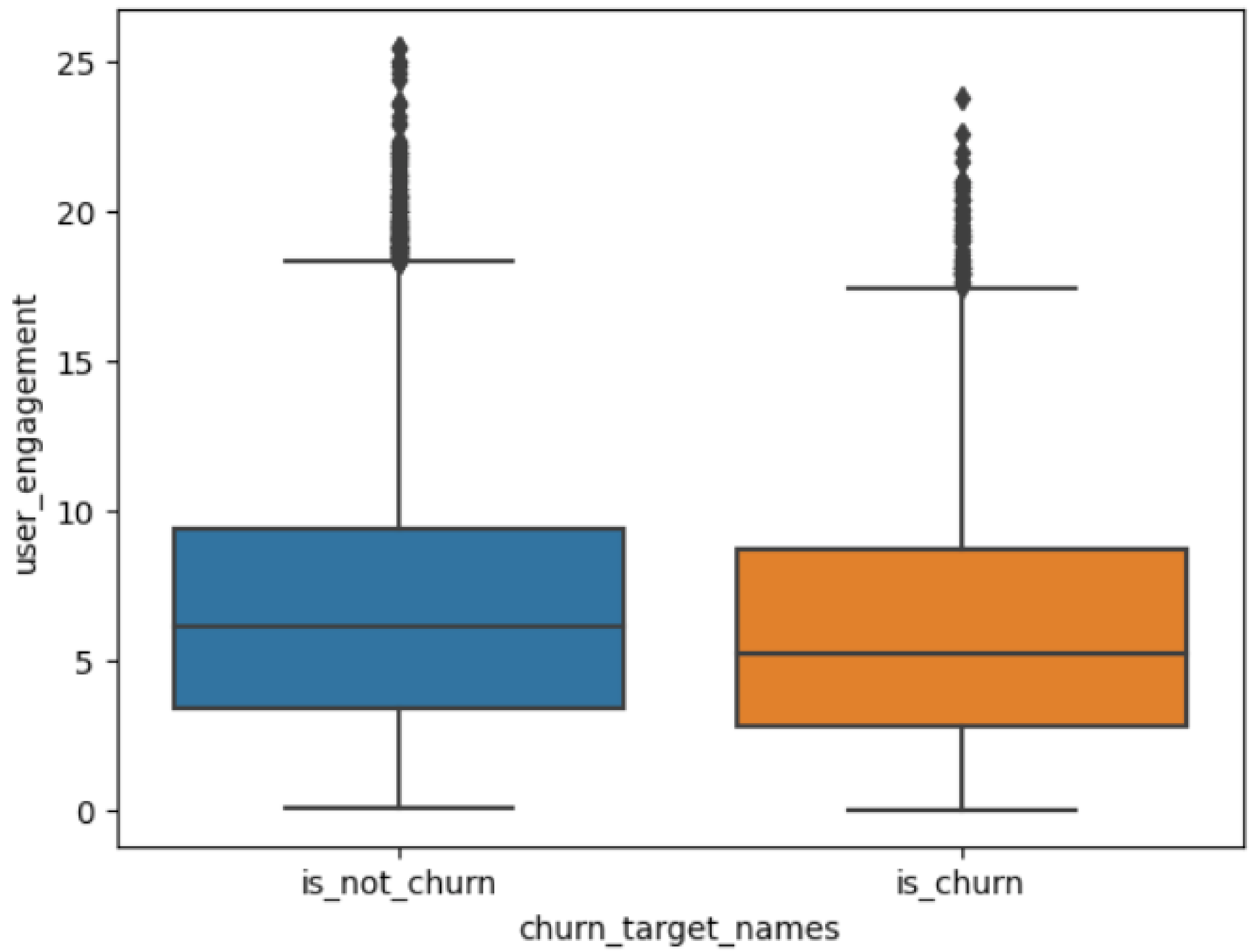
Decision-Tree Baseline

User raw-features

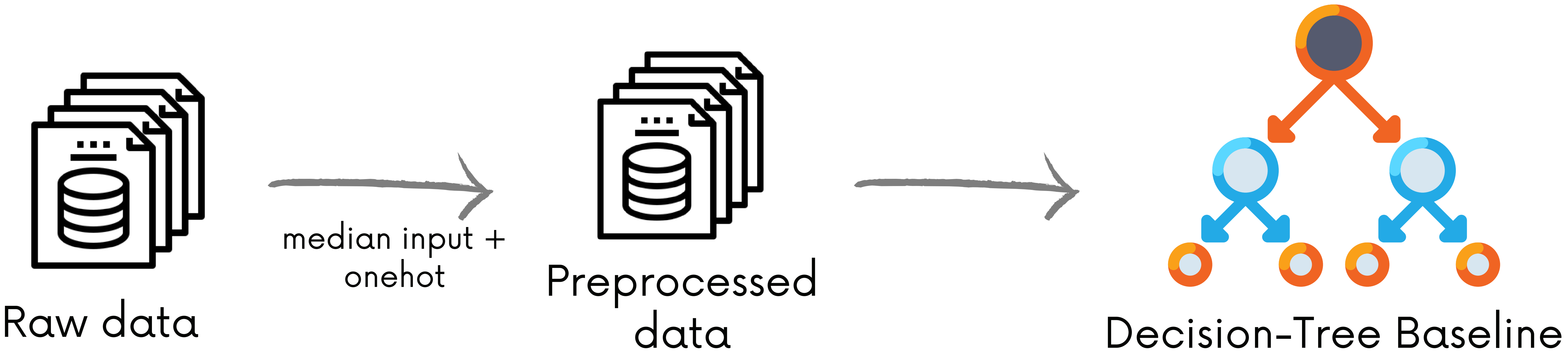
Heuristic Baseline

user_engagement.quantile < 20, selected threshold to maintain a similar distribution





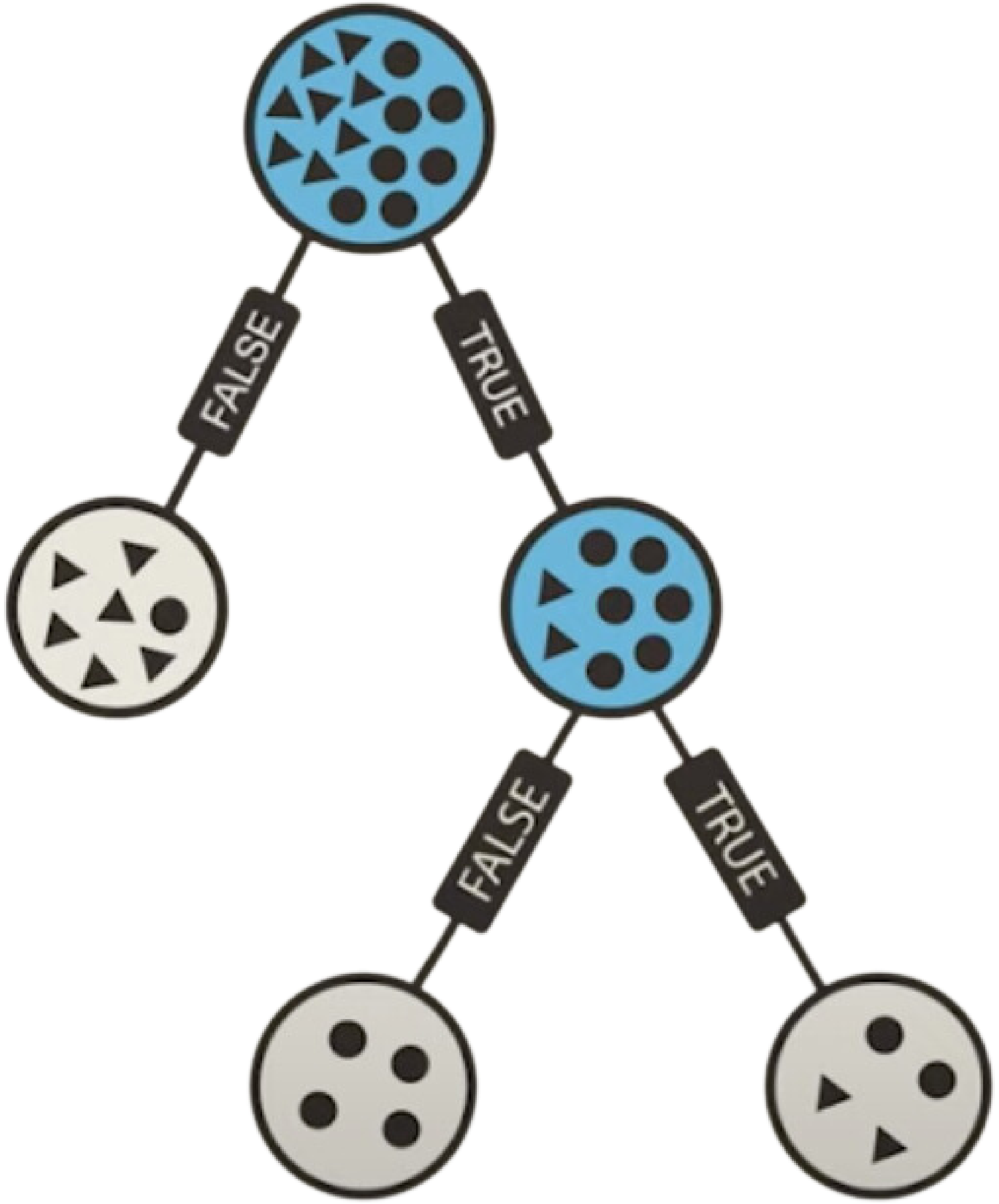
Decision-Tree



How decision trees works?

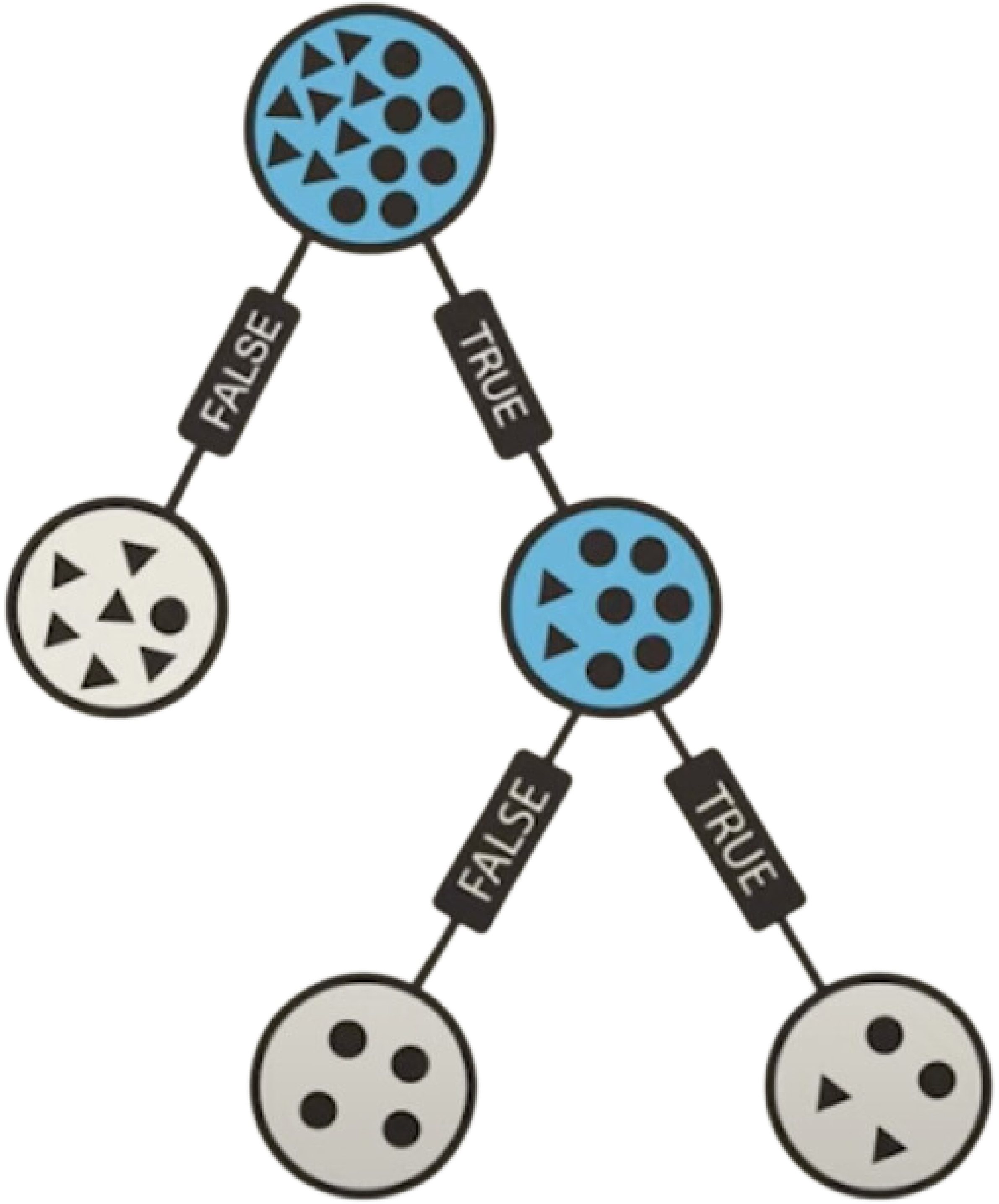
overall concept

How decision trees works?



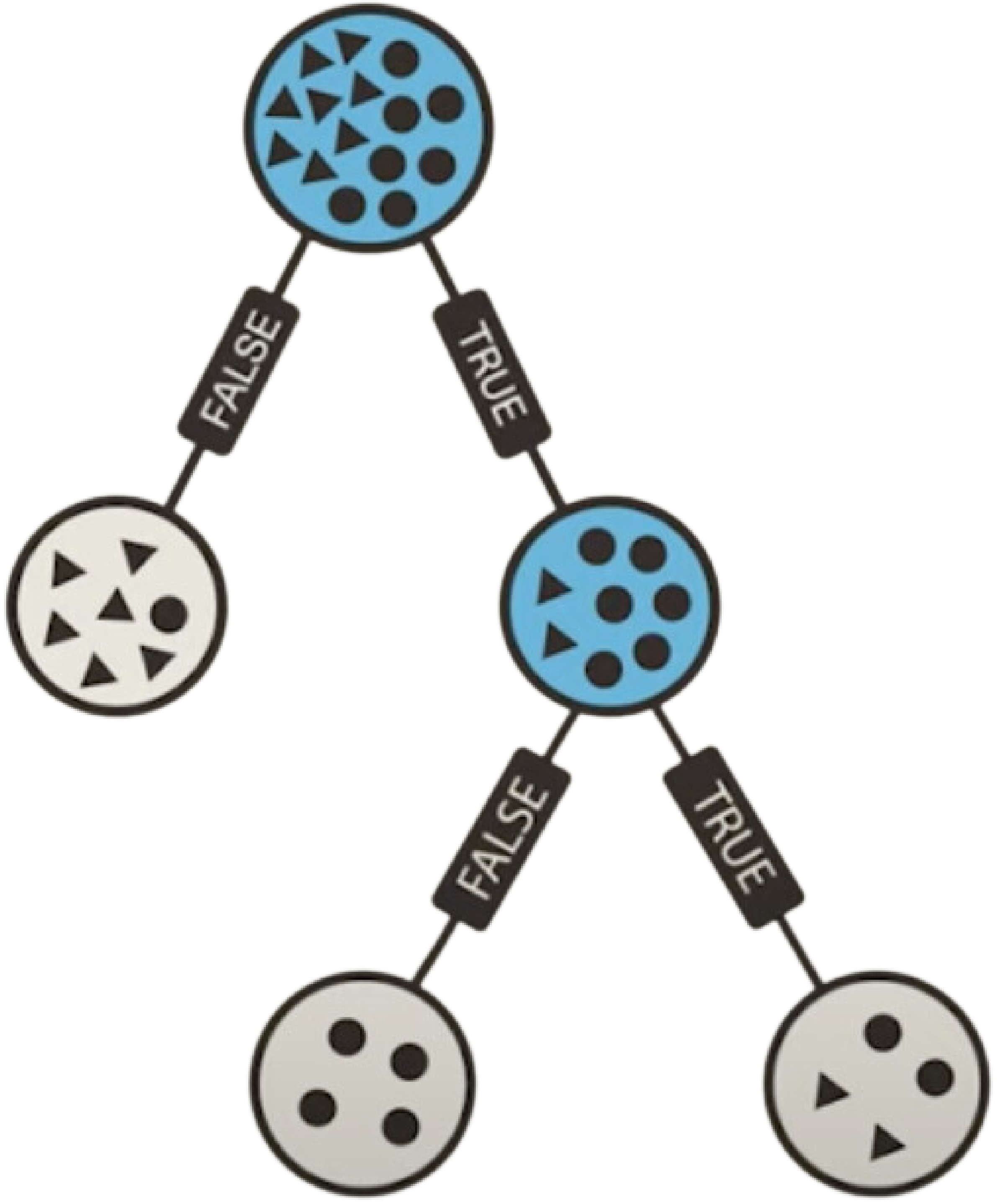
1. Represents decision as the branch of each node.

How decision trees works?



1. Represents decision as the branch of each node.
2. Each node is calculated thinking in the amount of information gain.

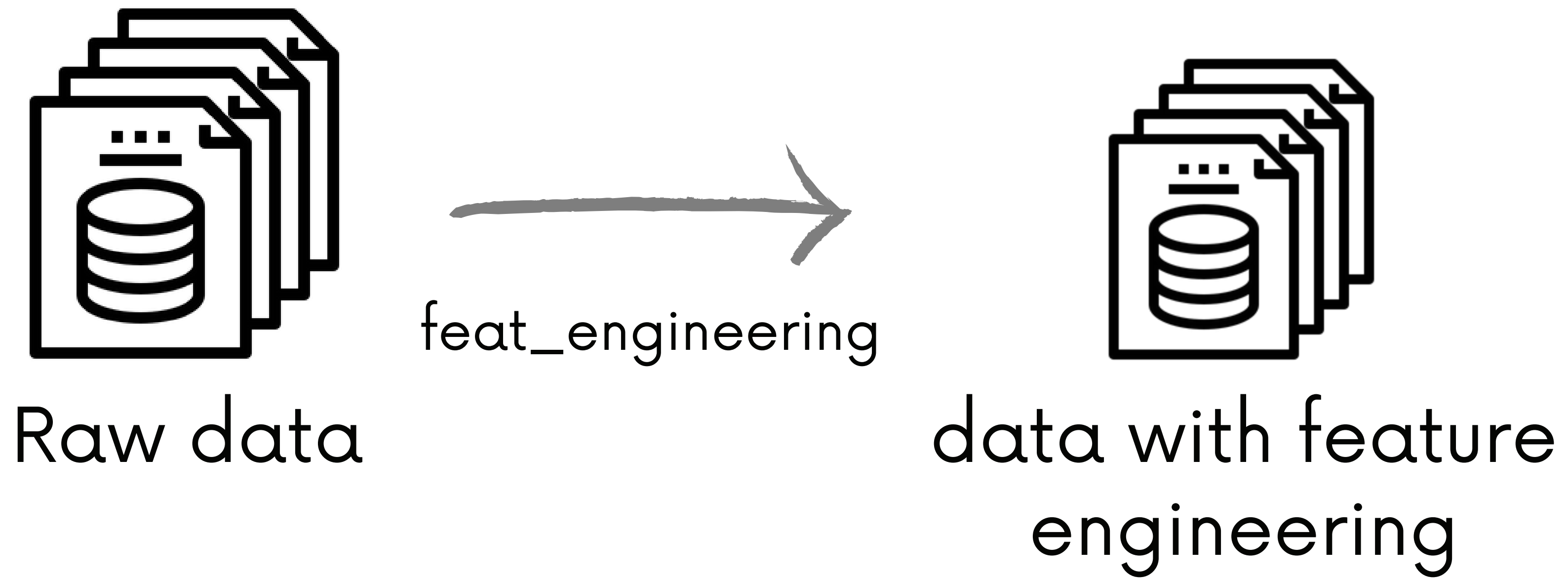
How decision trees works?



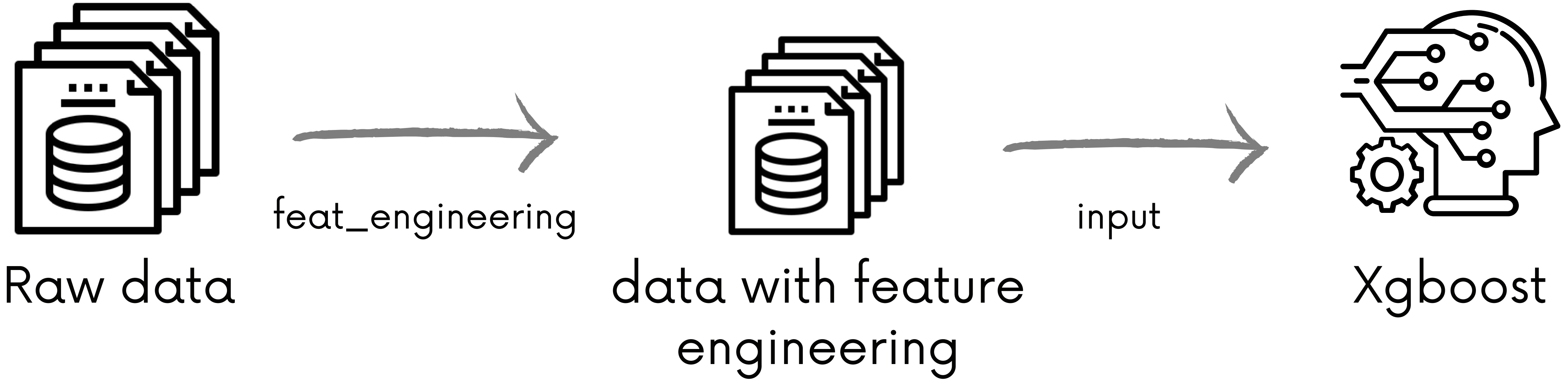
1. Represents decision as the branch of each node.
2. Each node is calculated thinking in the amount of information gain.
    a. The info gain is calculated at the impurity level of the child nodes
    b. We have multiple formulas for info gain: gini, entropy, etc.

Modeling Phase

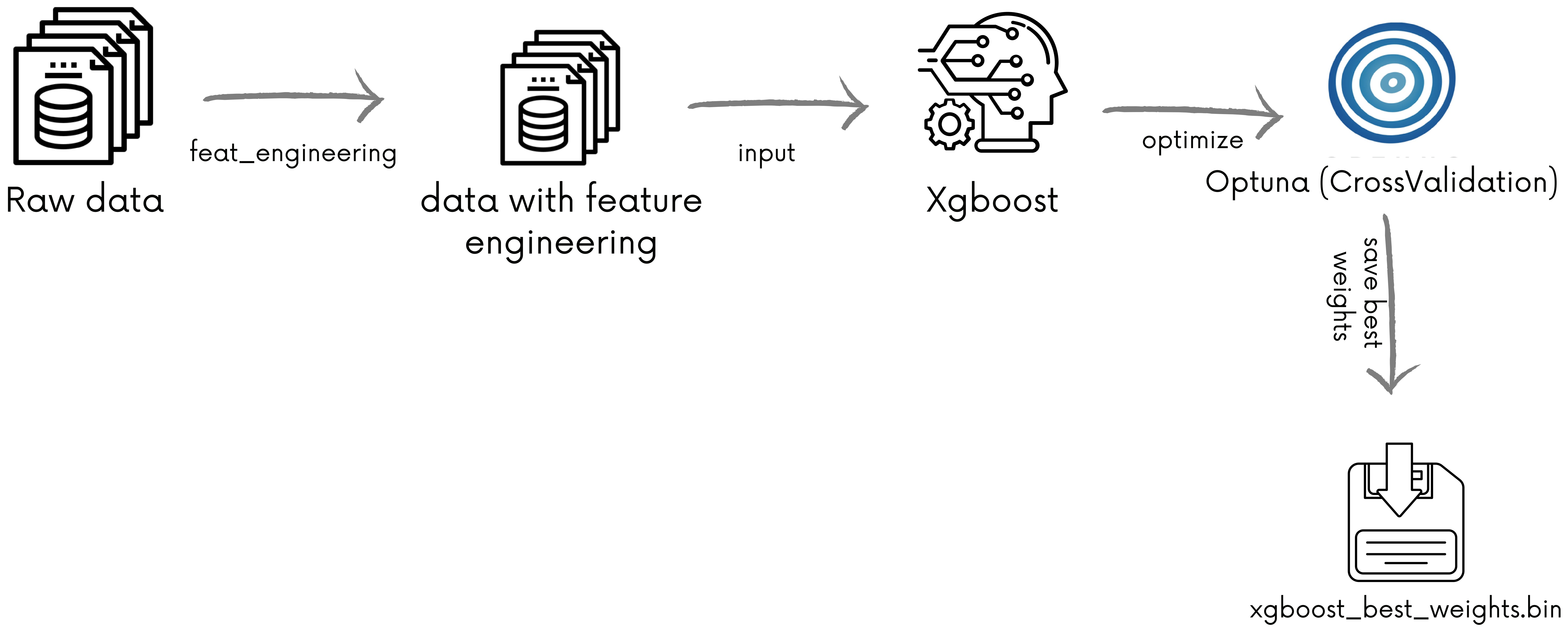
Modeling Phase



Modeling Phase



Modeling Phase



Feature Engineering

Feature Engineering

1 Created most of features thinking about how to calculate how loyal the user is.

Feature Engineering

2 Use TargetEncoder for categorical features

Feature Engineering

3 Tried to create "gym features". I just aggregated features with stats metrics for each gym, example:

- Average, Standard Deviation, Skew, Kurtosis of user_life_time

Feature Engineering

4 Transformed features using log to solve skewing.

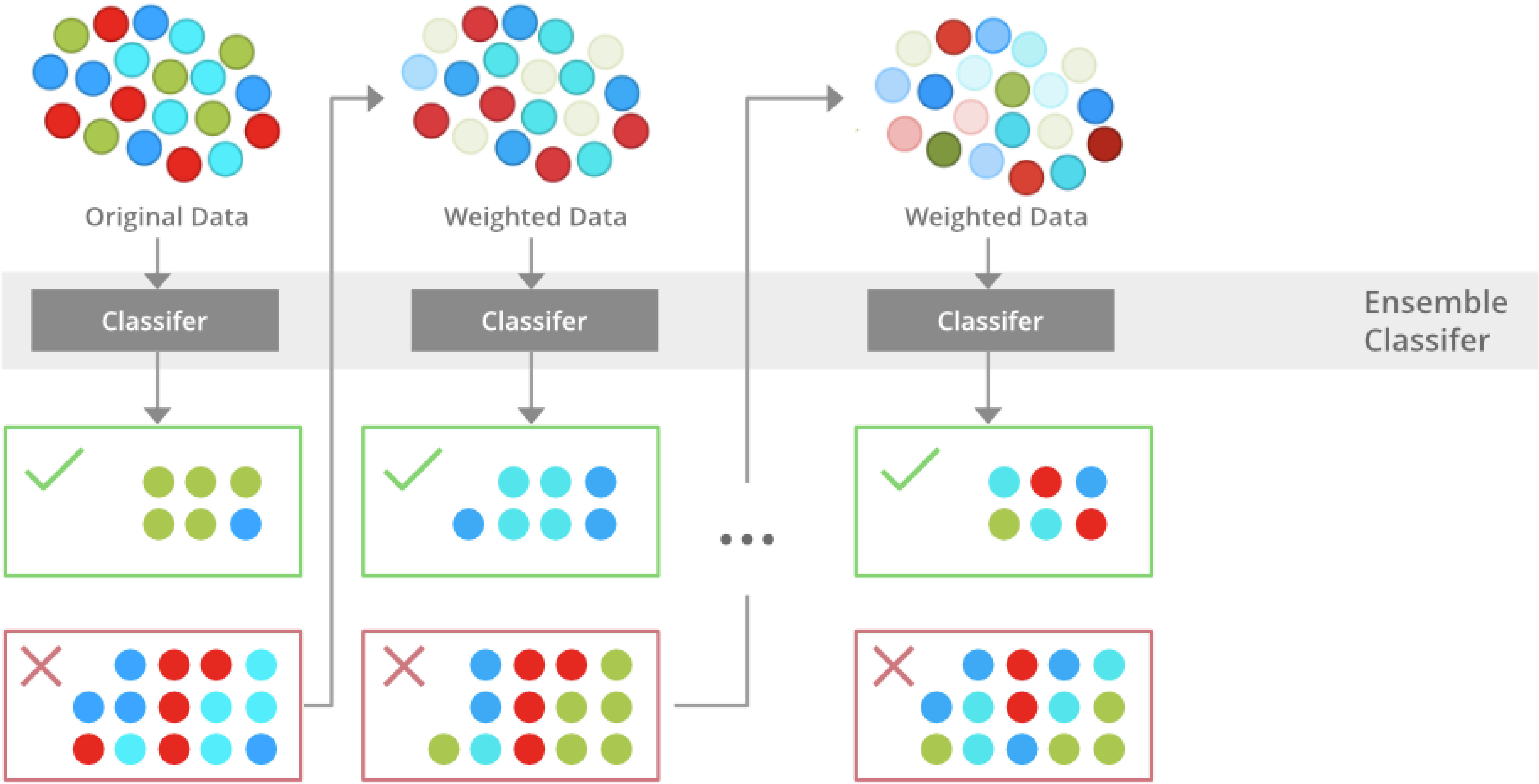
Feature Engineering

5 Transformed the 1% of gym categories to "Other"

How Xgboost works?

overall concept

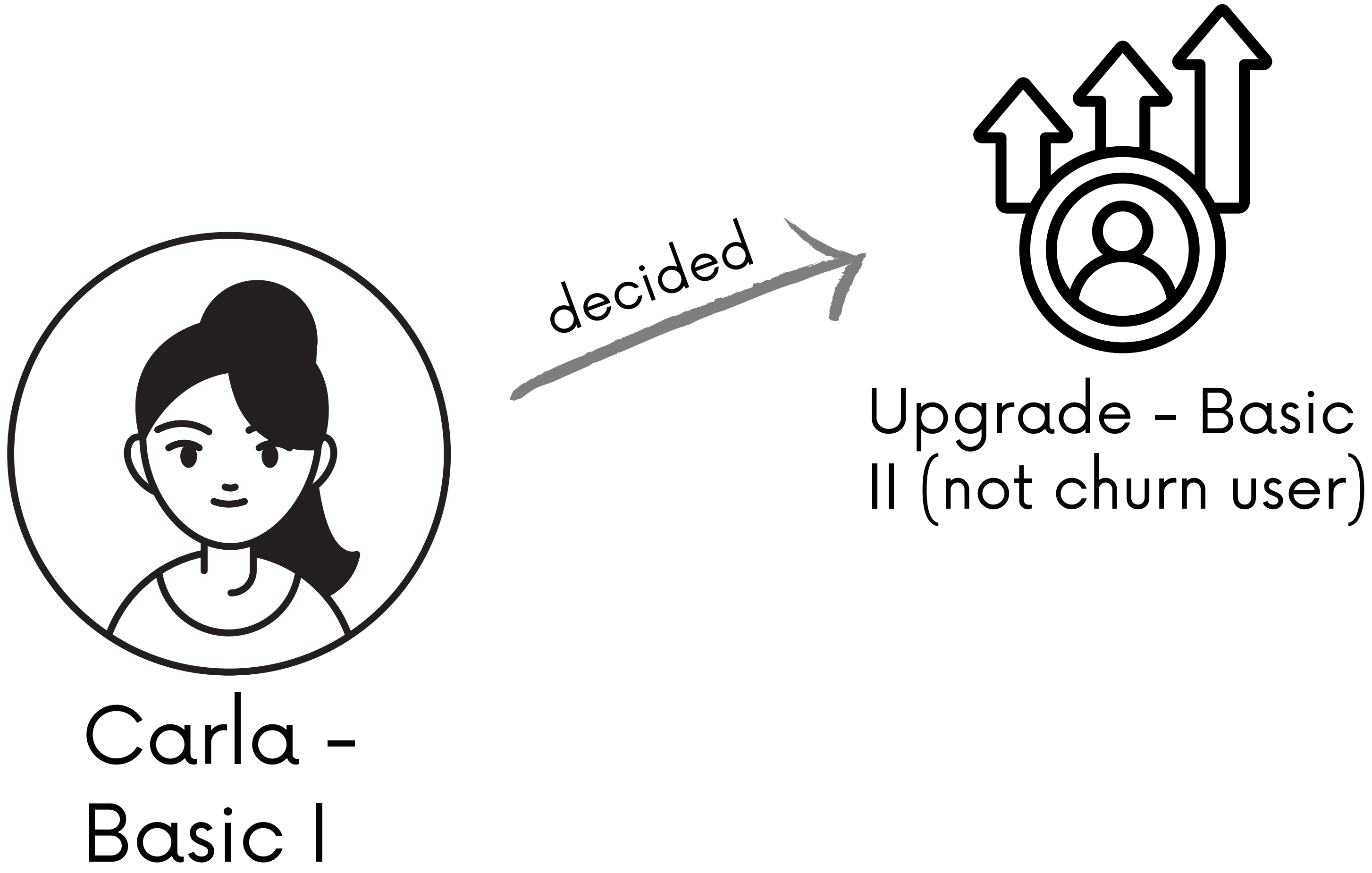
How Xgboost works?



Model Output and metrics

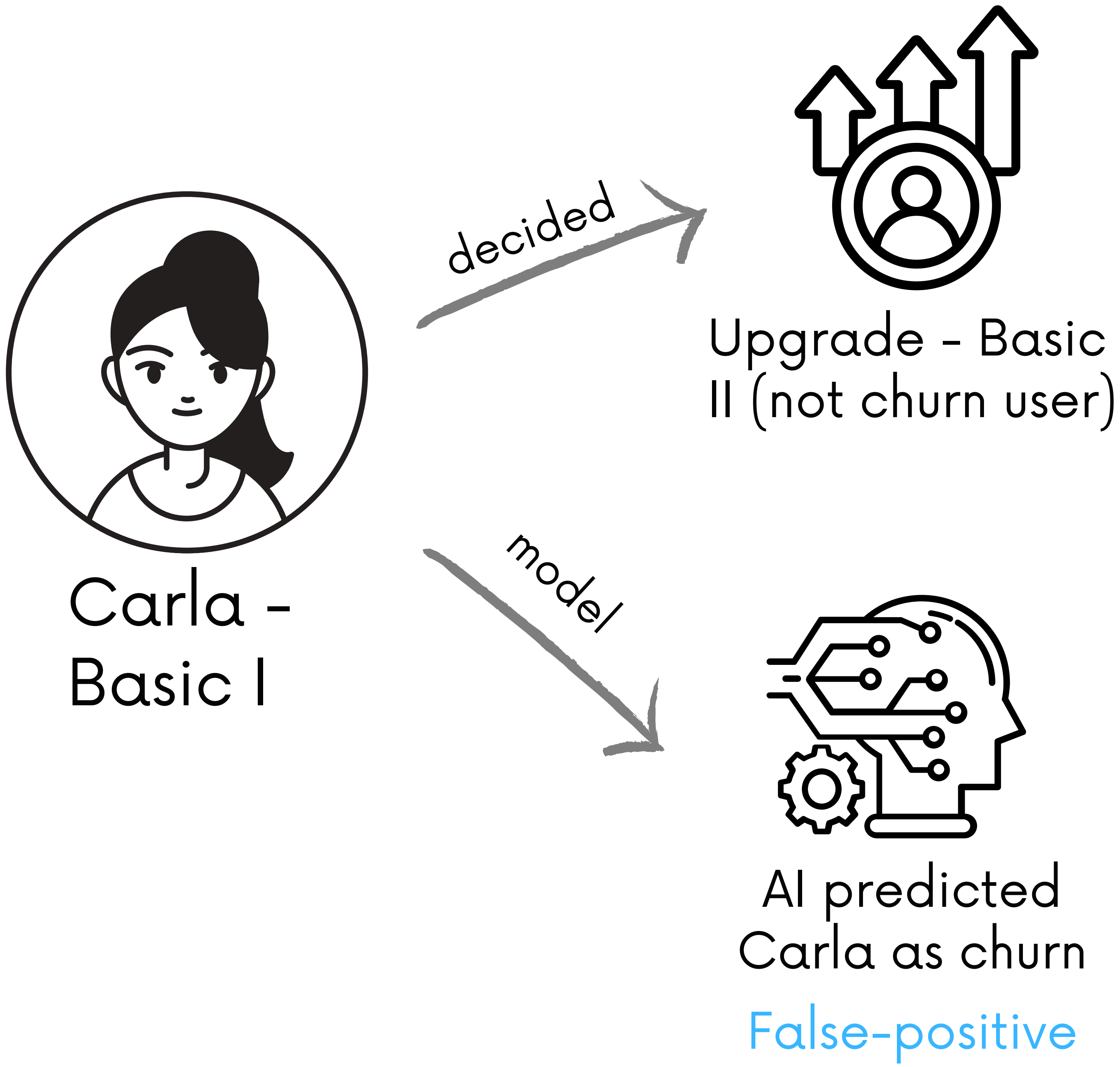
What is more important to churn?

Gym uptier occurs..



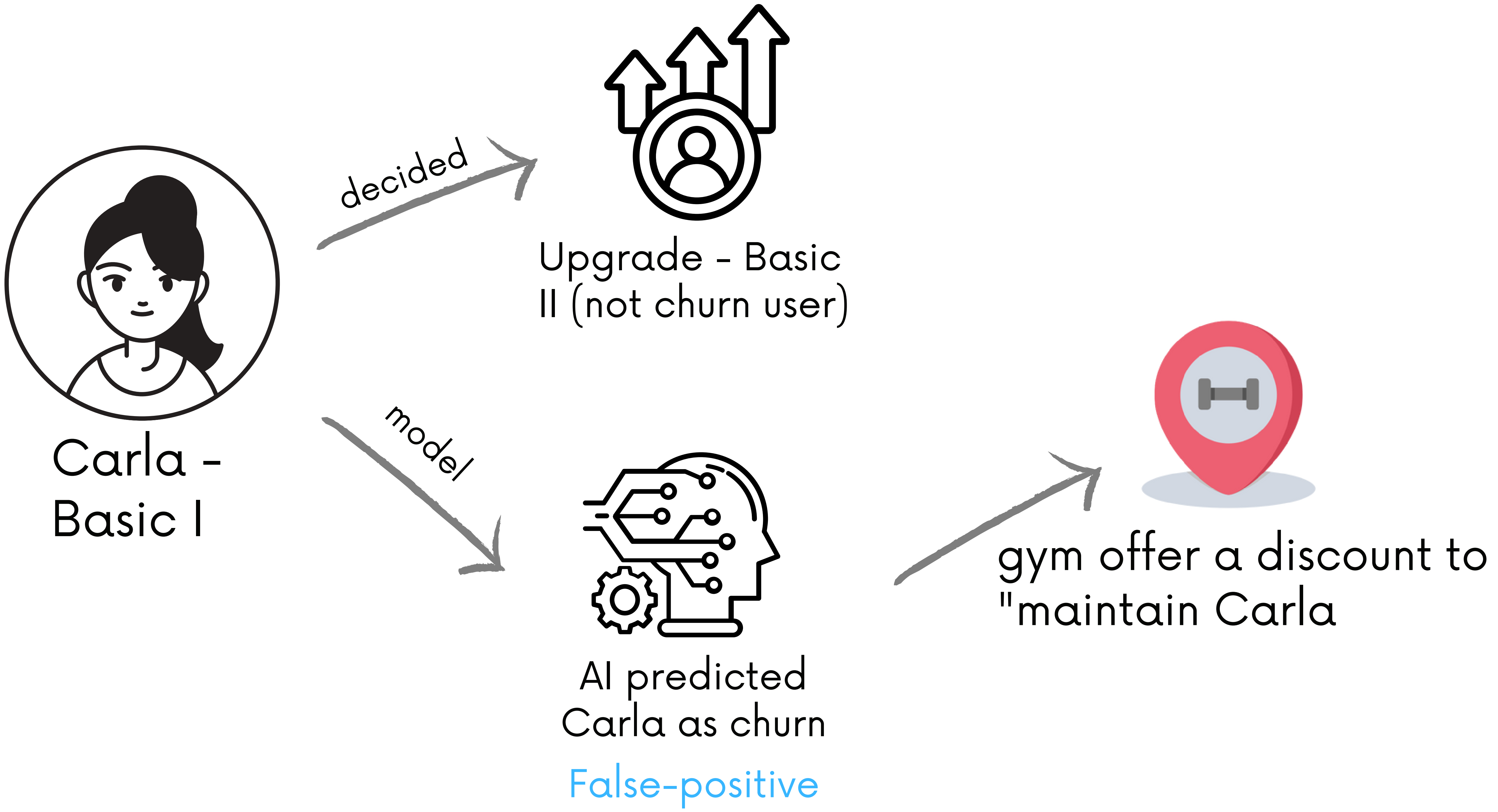
What is more important to churn?

Gym uptier occurs..



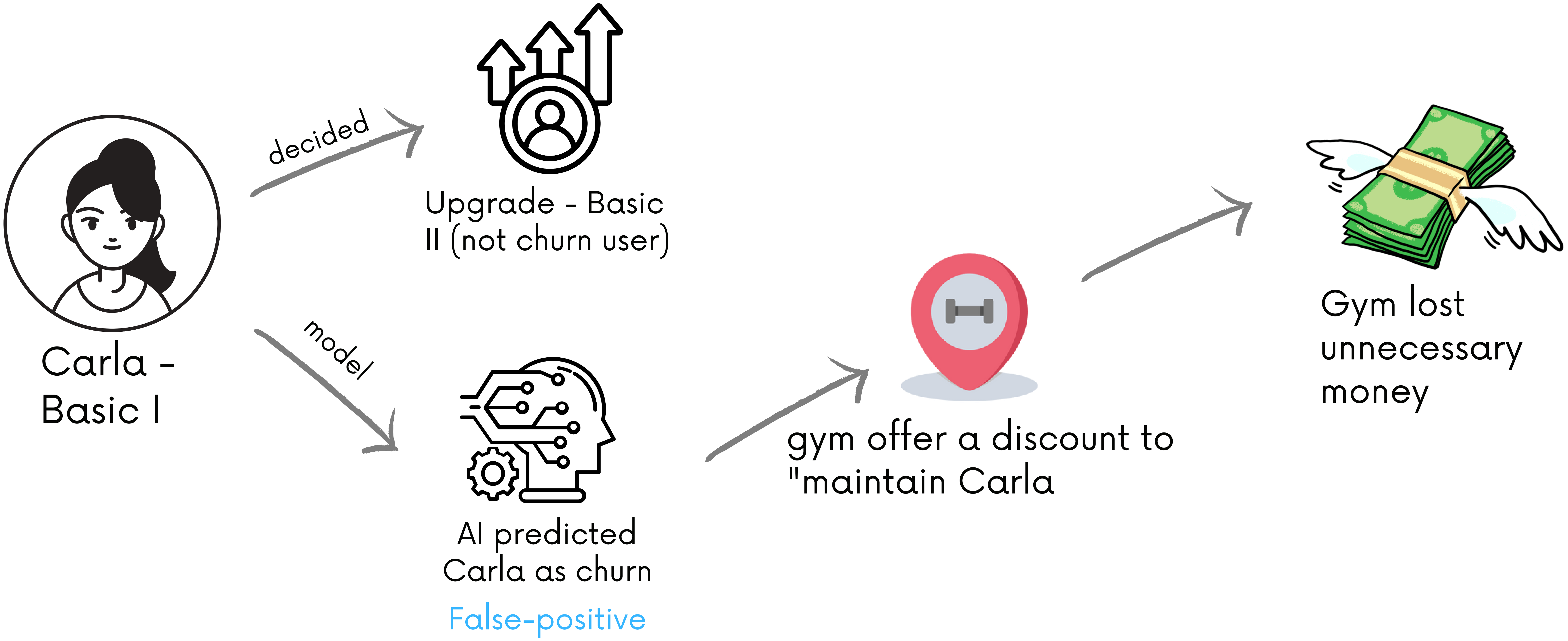
What is more important to churn?

Gym uptier occurs..



What is more important to churn?

Gym uptier occurs..

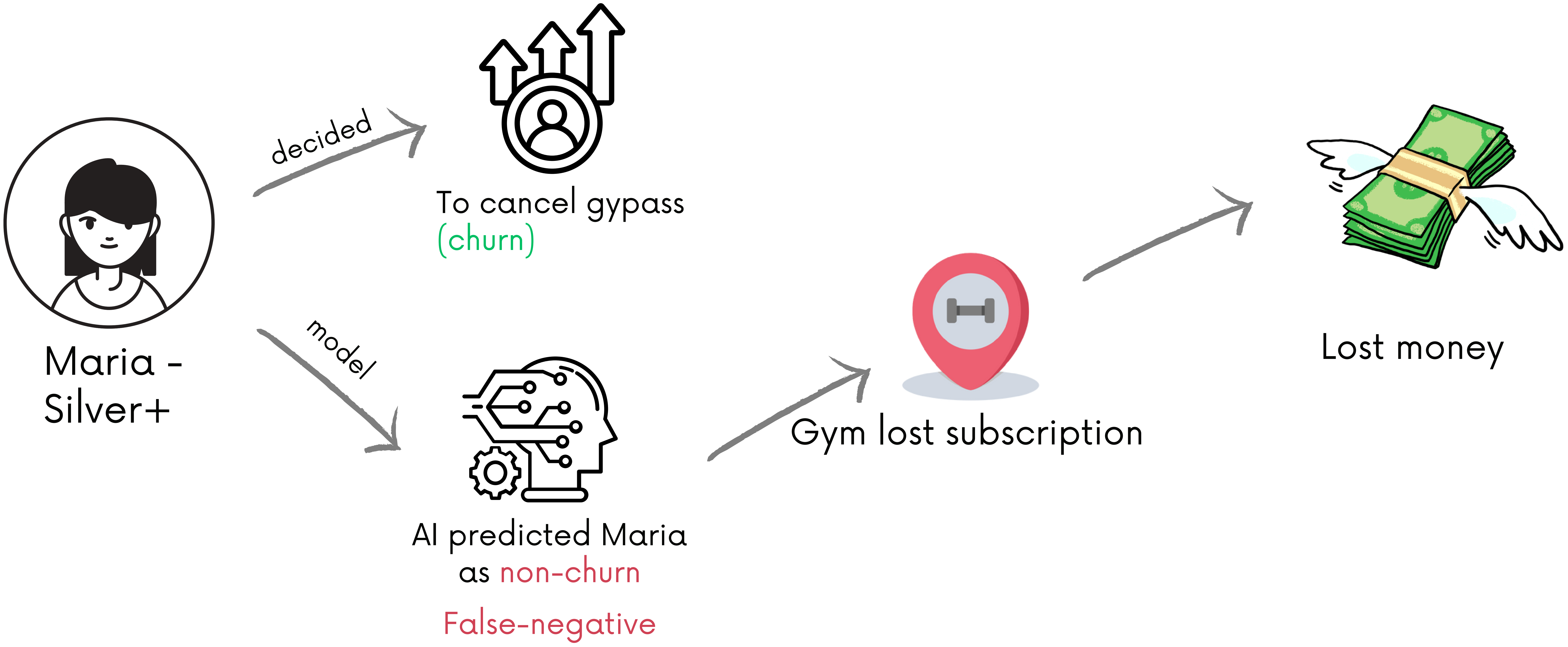


What is more important to churn?

False-positives
affects precision

What is more important to churn?

Gym uptier occurs..

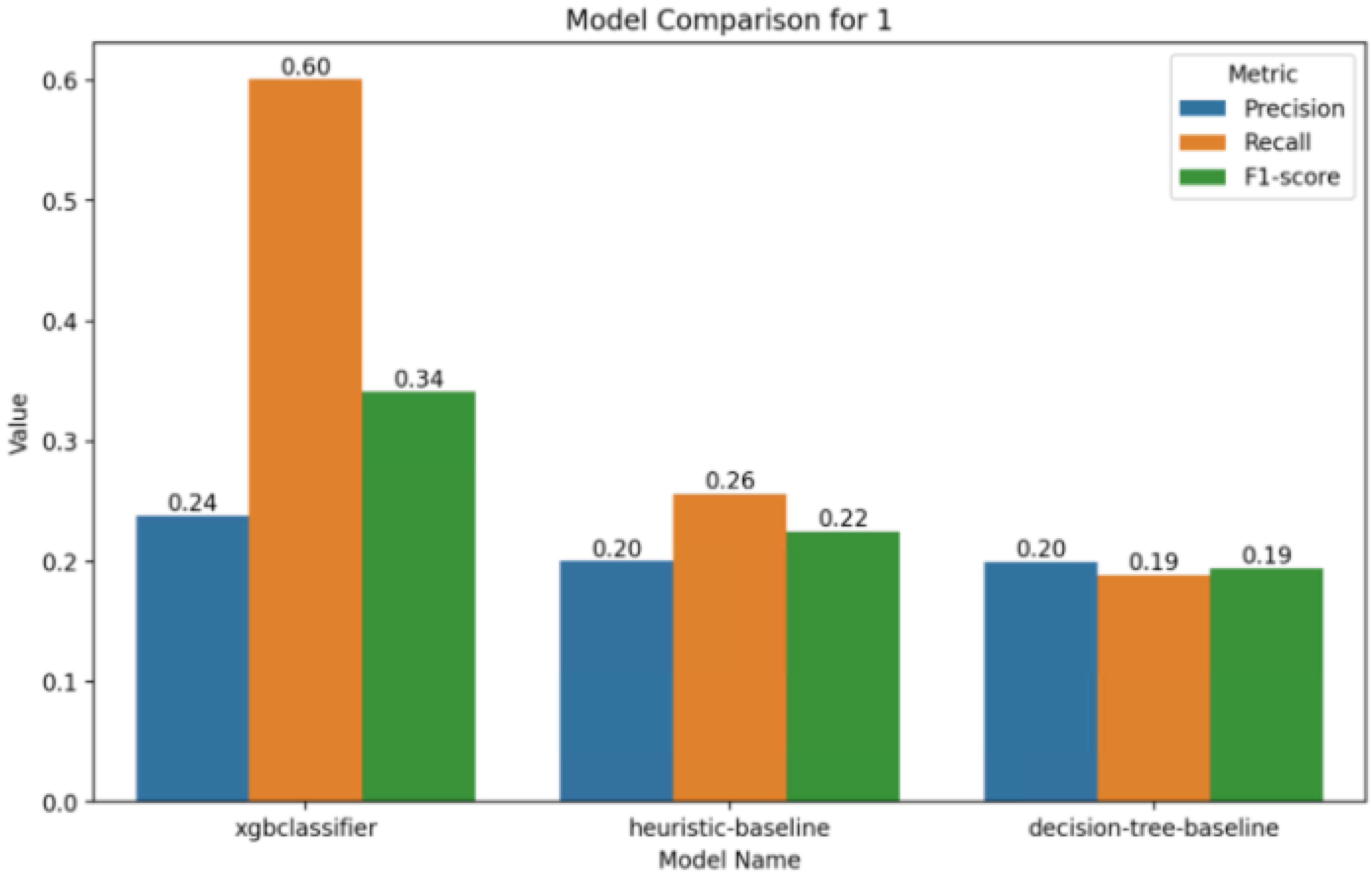


What is more important to churn?

False-negatives
affects recall

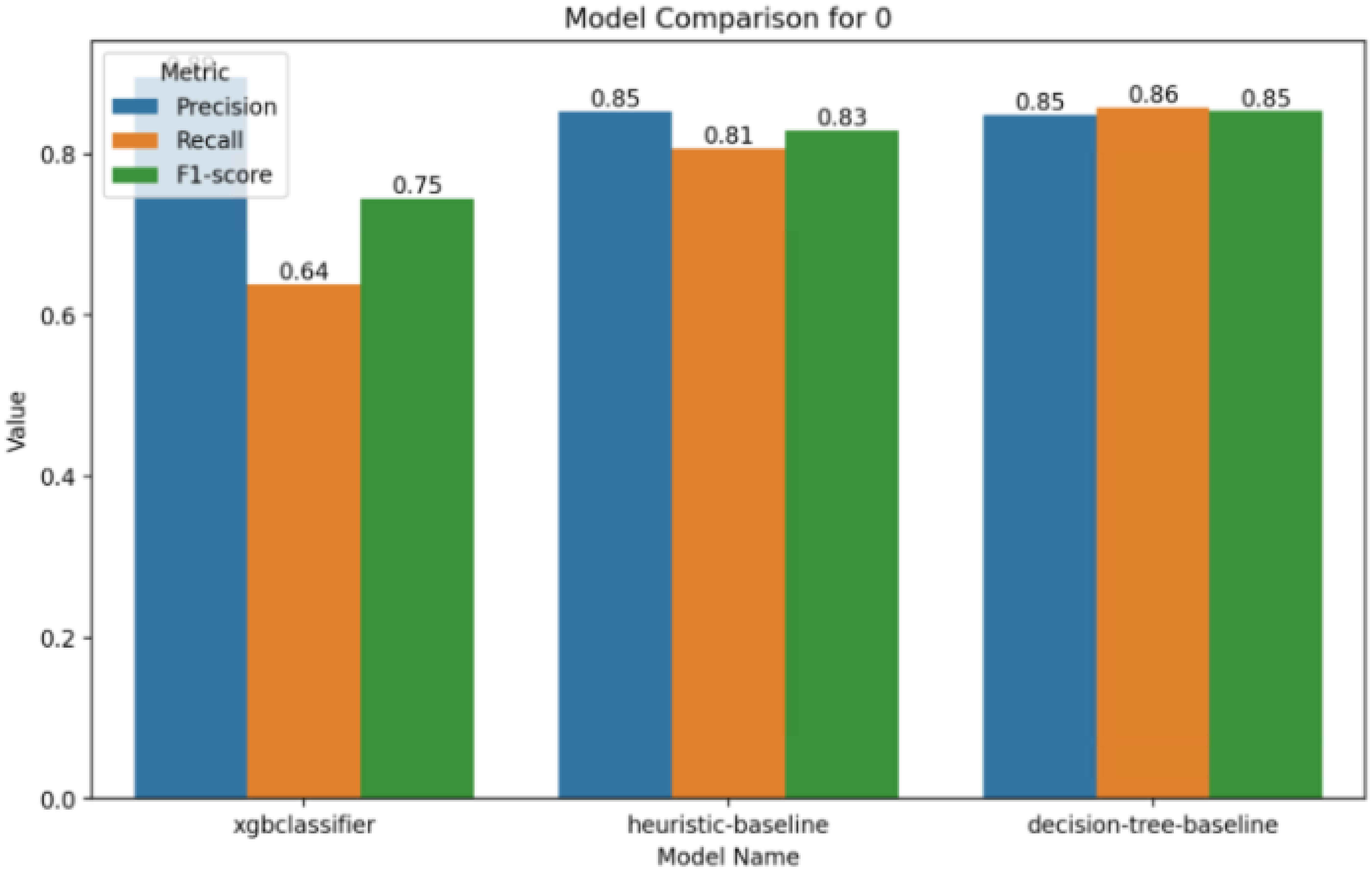
Model Results

Churn class

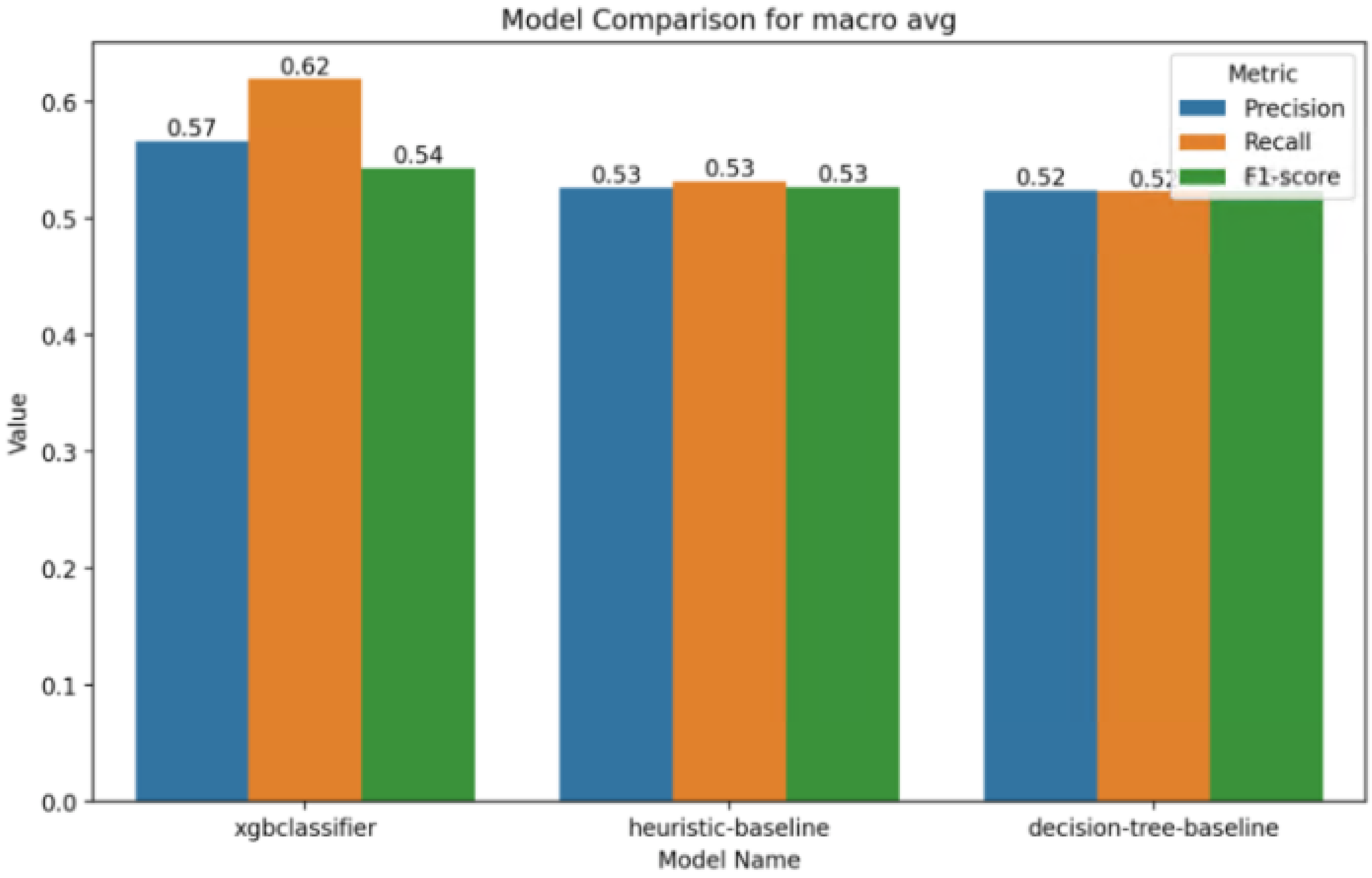


Model Comparison for 1
Metric
Precision
Recall
F1-score
0.60
0.34
0.24
0.26
0.22
0.20
0.20
0.19
0.19
Value
0.0
0.1
0.2
0.3
0.4
0.5
0.6
xgbclassifier
heuristic-baseline
decision-tree-baseline
Model Name

Non-churn class

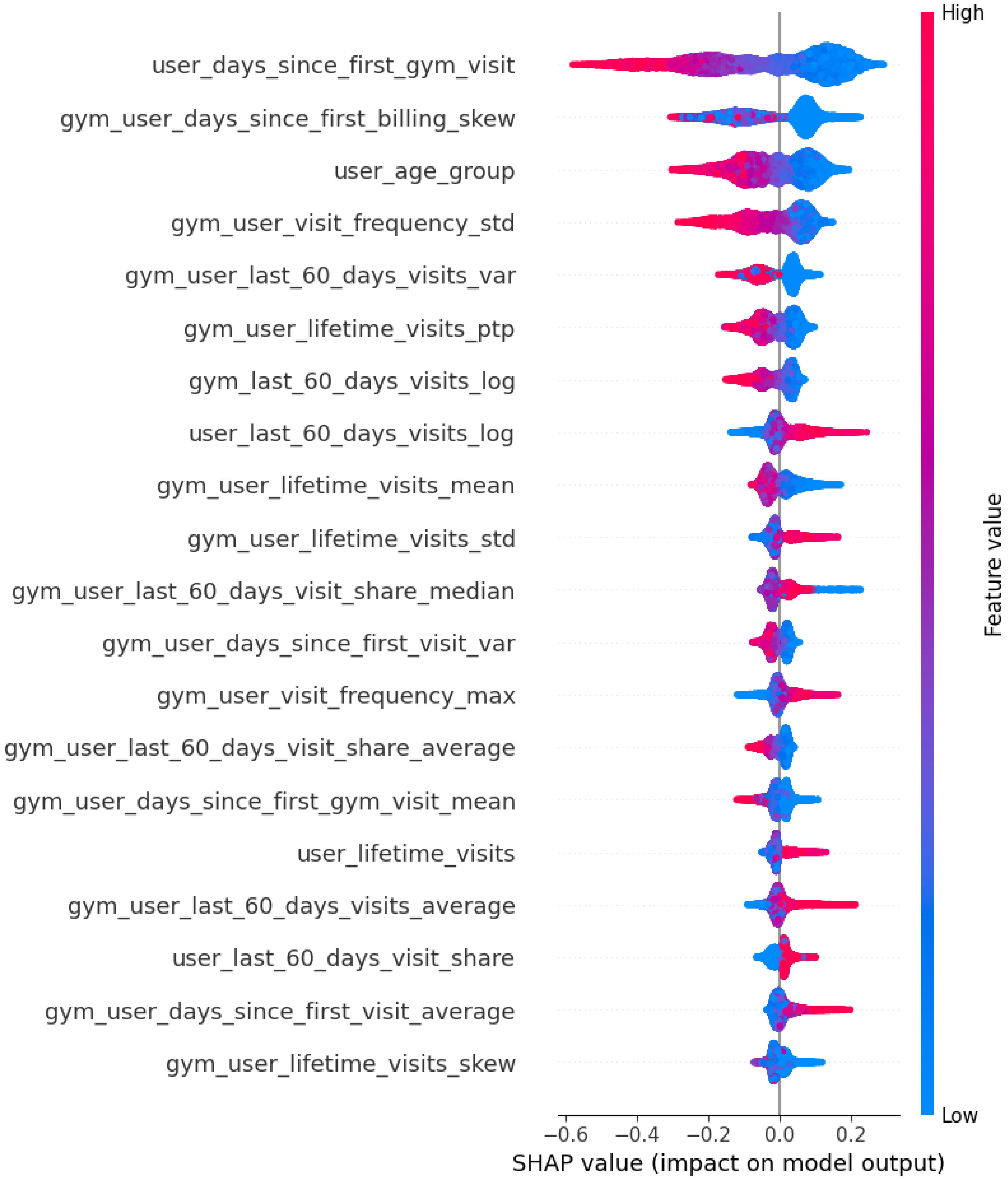


Both class



Model Comparison for macro avg
Metric
Precision
Recall
F1-score
0.57
0.62
0.54
0.53
0.53
0.53
0.52
0.0
0.1
0.2
0.3
0.4
0.5
0.6
Value
xgbclassifier
heuristic-baseline
decision-tree-baseline
Model Name

Model Interpretation (SHAP)

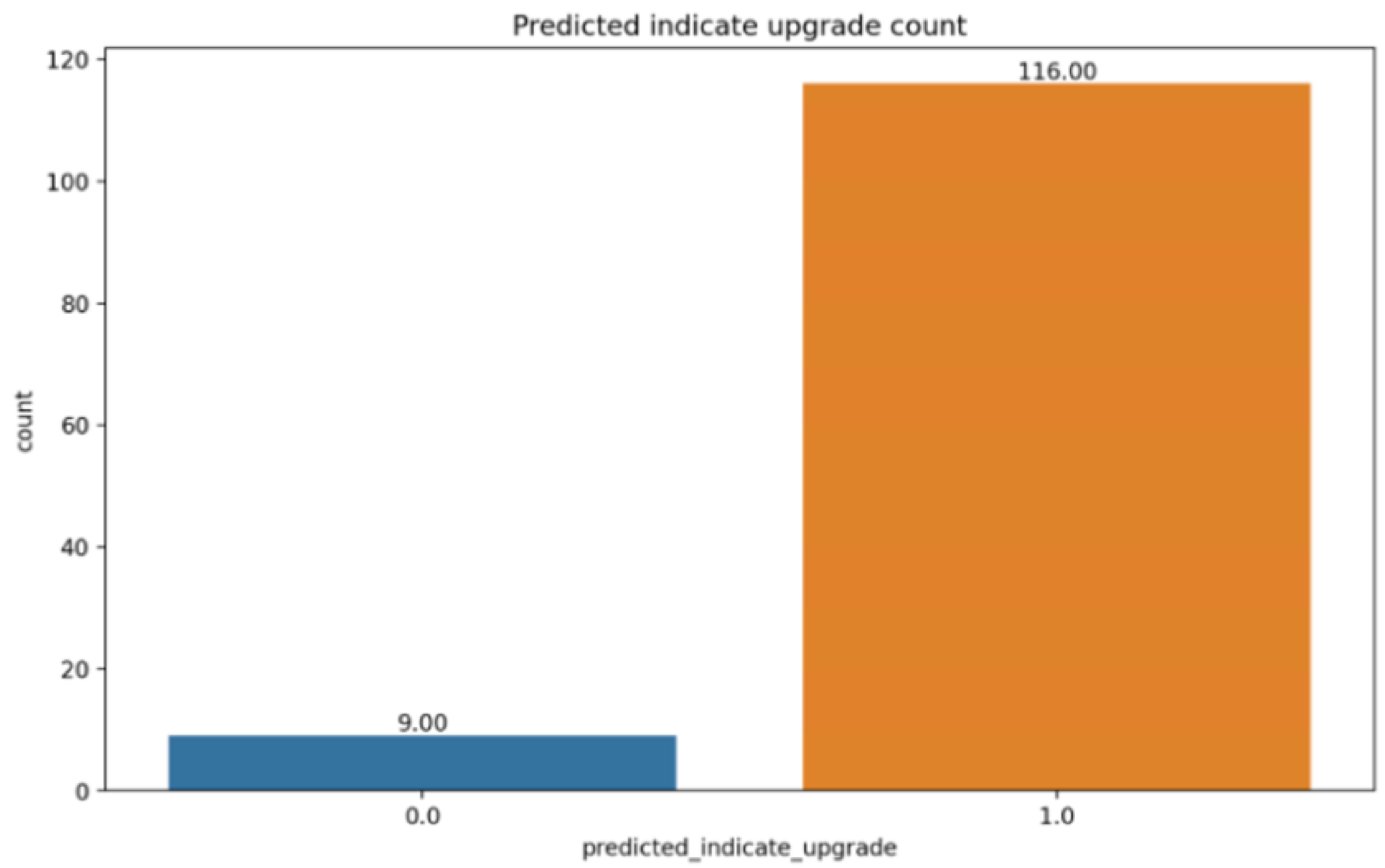


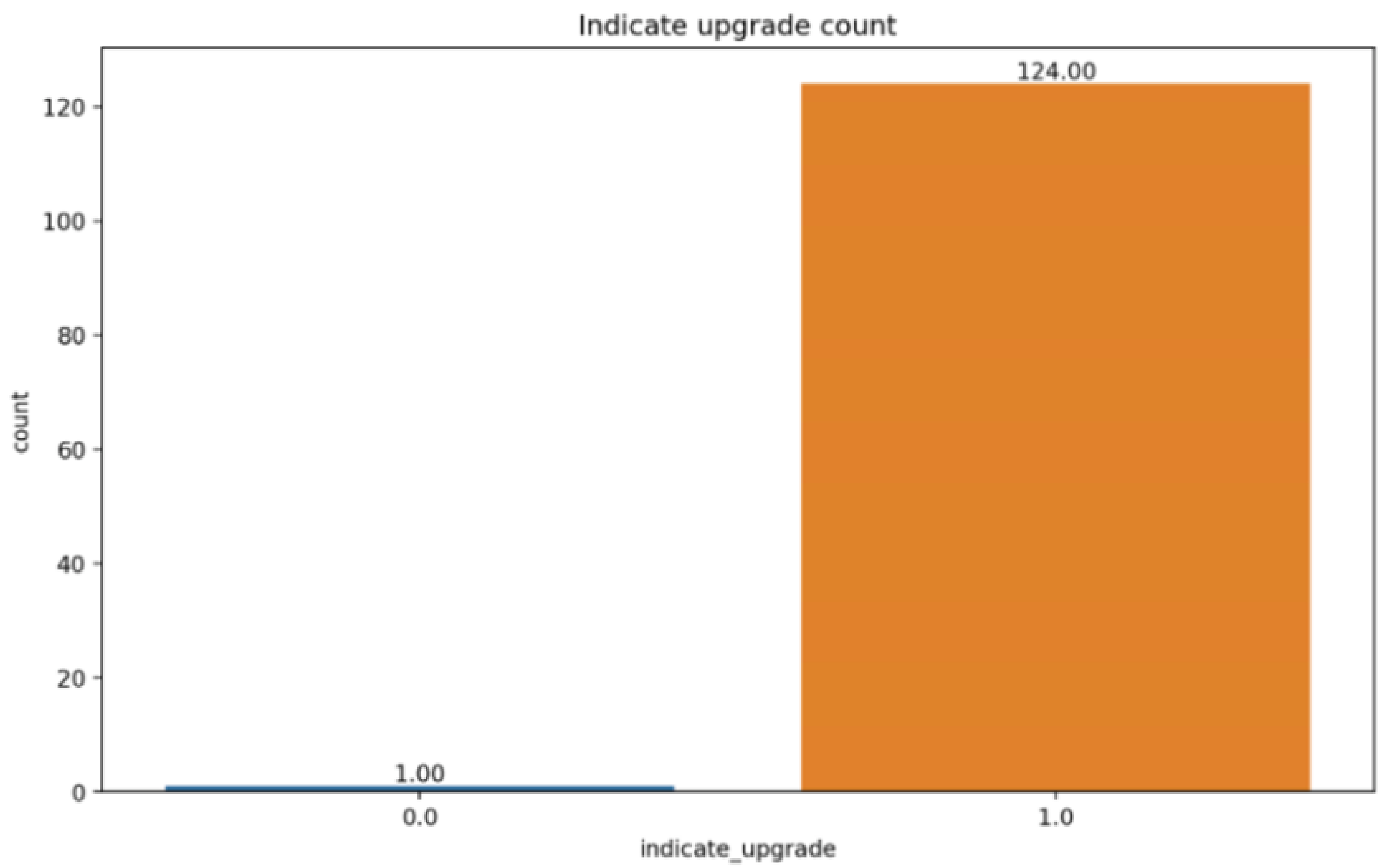
user_days_since_first_gym_visit
gym_user_days_since_first_billing_skew
user_age_group
gym_user_visit_frequency_std
gym_user_last_60_days_visits_var
gym_user_lifetime_visits_ptp
gym_last_60_days_visits_log
user_last_60_days_visits_log
gym_user_lifetime_visits_mean
gym_user_lifetime_visits_std
gym_user_last_60_days_visit_share_median
gym_user_days_since_first_visit_var
gym_user_visit_frequency_max
gym_user_last_60_days_visit_share_average
gym_user_days_since_first_gym_visit_mean
user_lifetime_visits
gym_user_last_60_days_visits_average
user_last_60_days_visit_share
gym_user_days_since_first_visit_average
gym_user_lifetime_visits_skew
−0.6
−0.4
−0.2
0.0
0.2
SHAP value (impact on model output)
High
Low
Feature value

Gyms to indicate upgrade

In test dataset

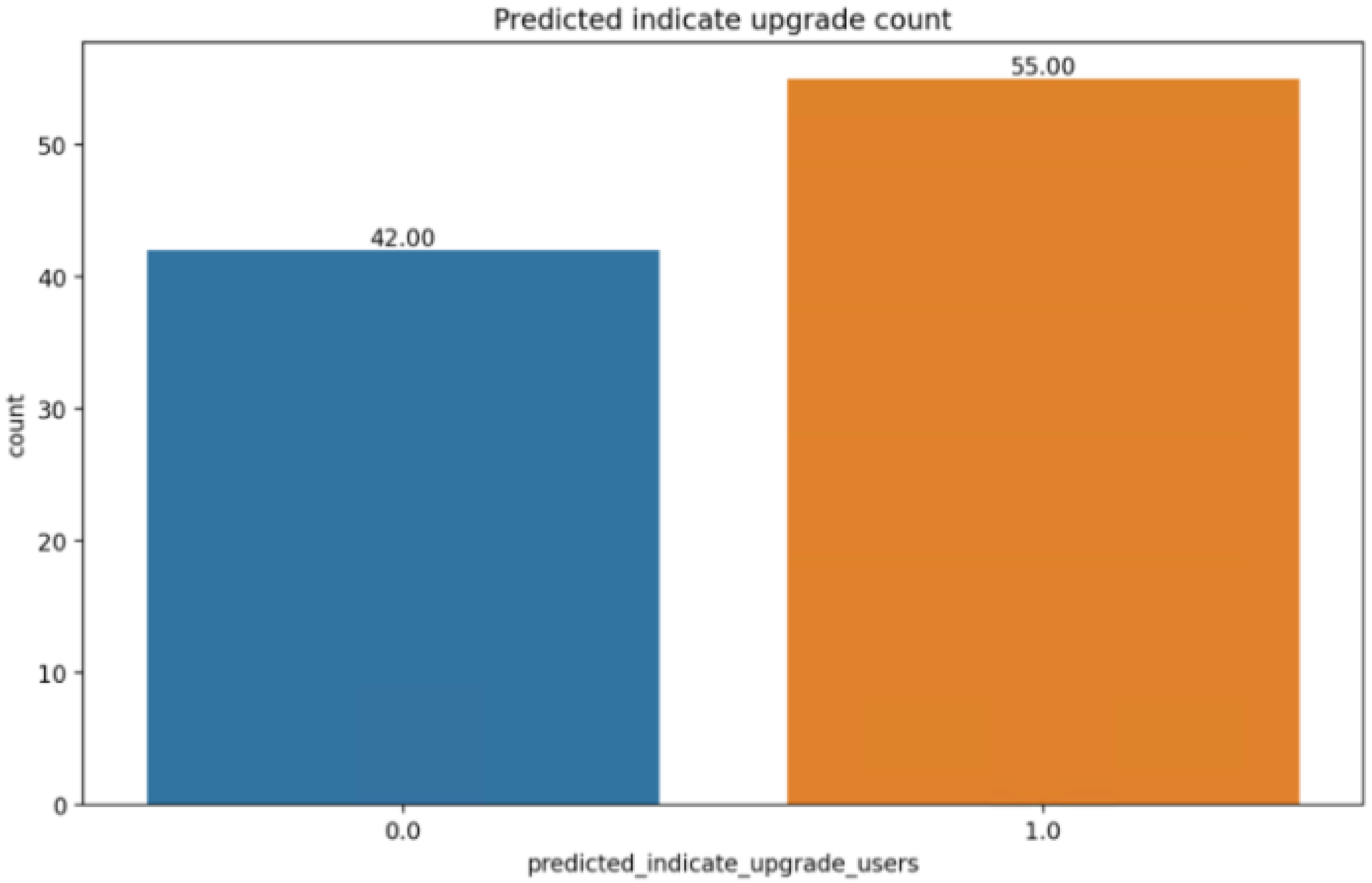
- show streamlit (to show the decision threshold vs profit)

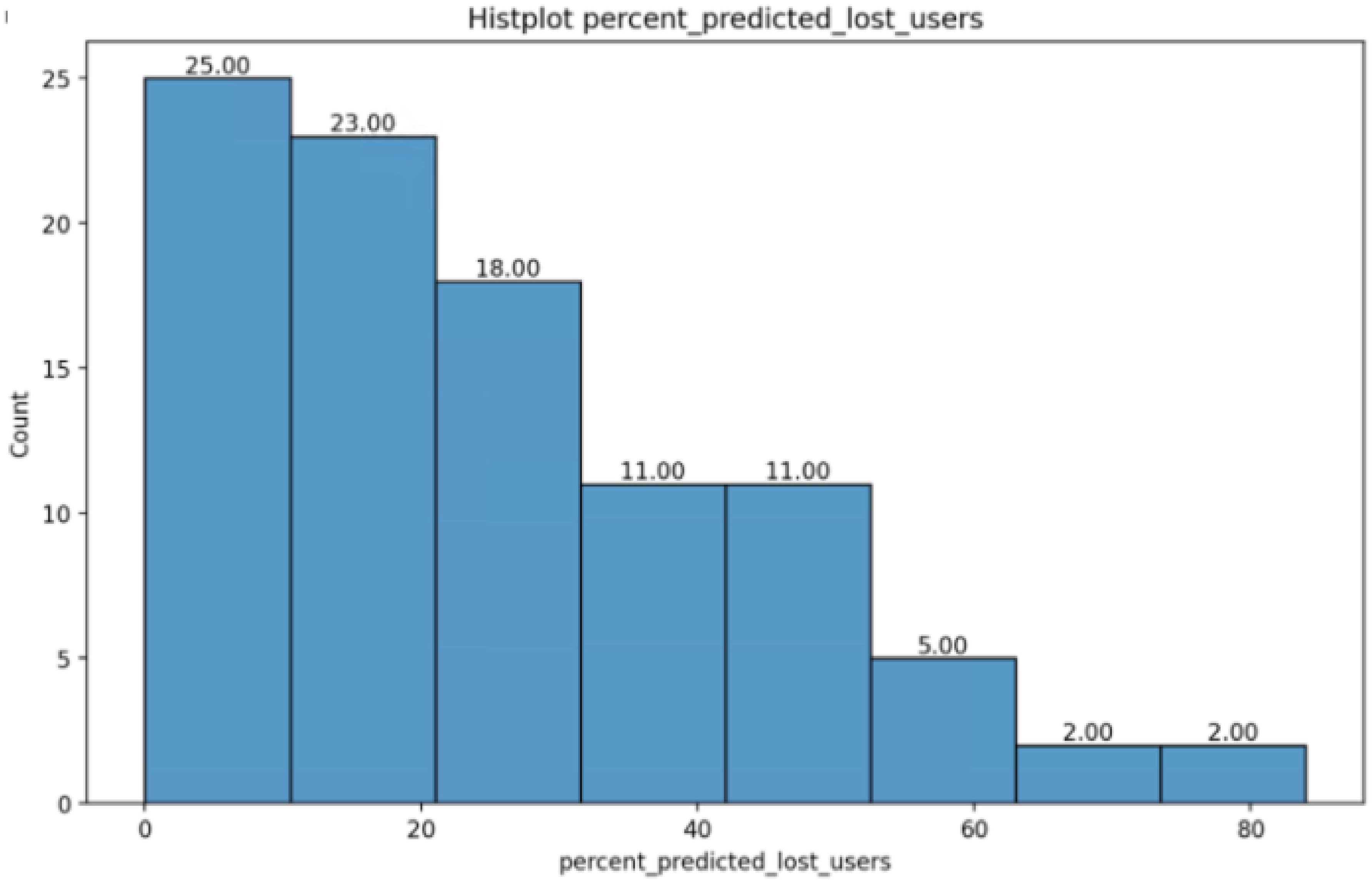




In submission dataset

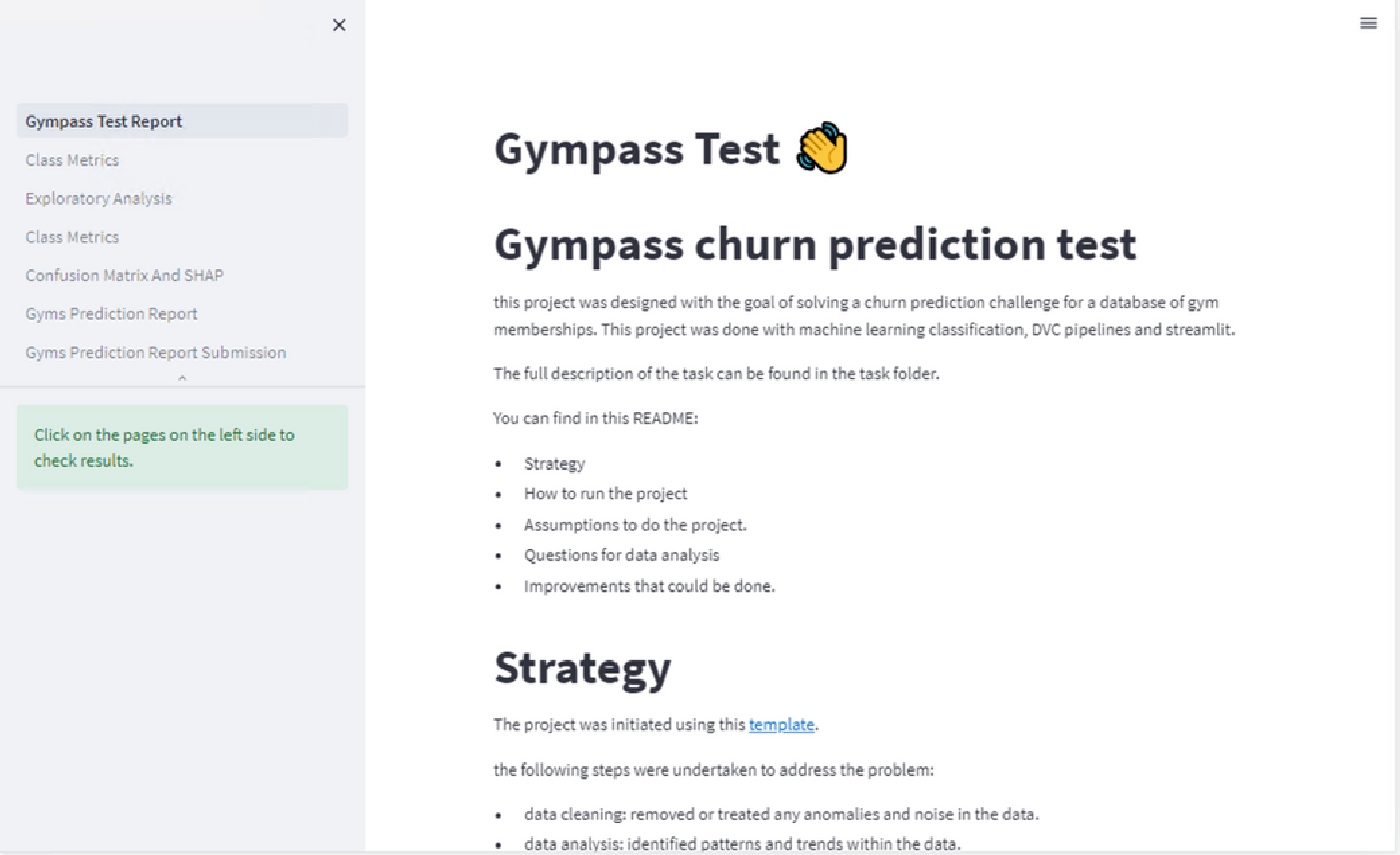
- show streamlit (to show the decision threshold)





Code Refactor

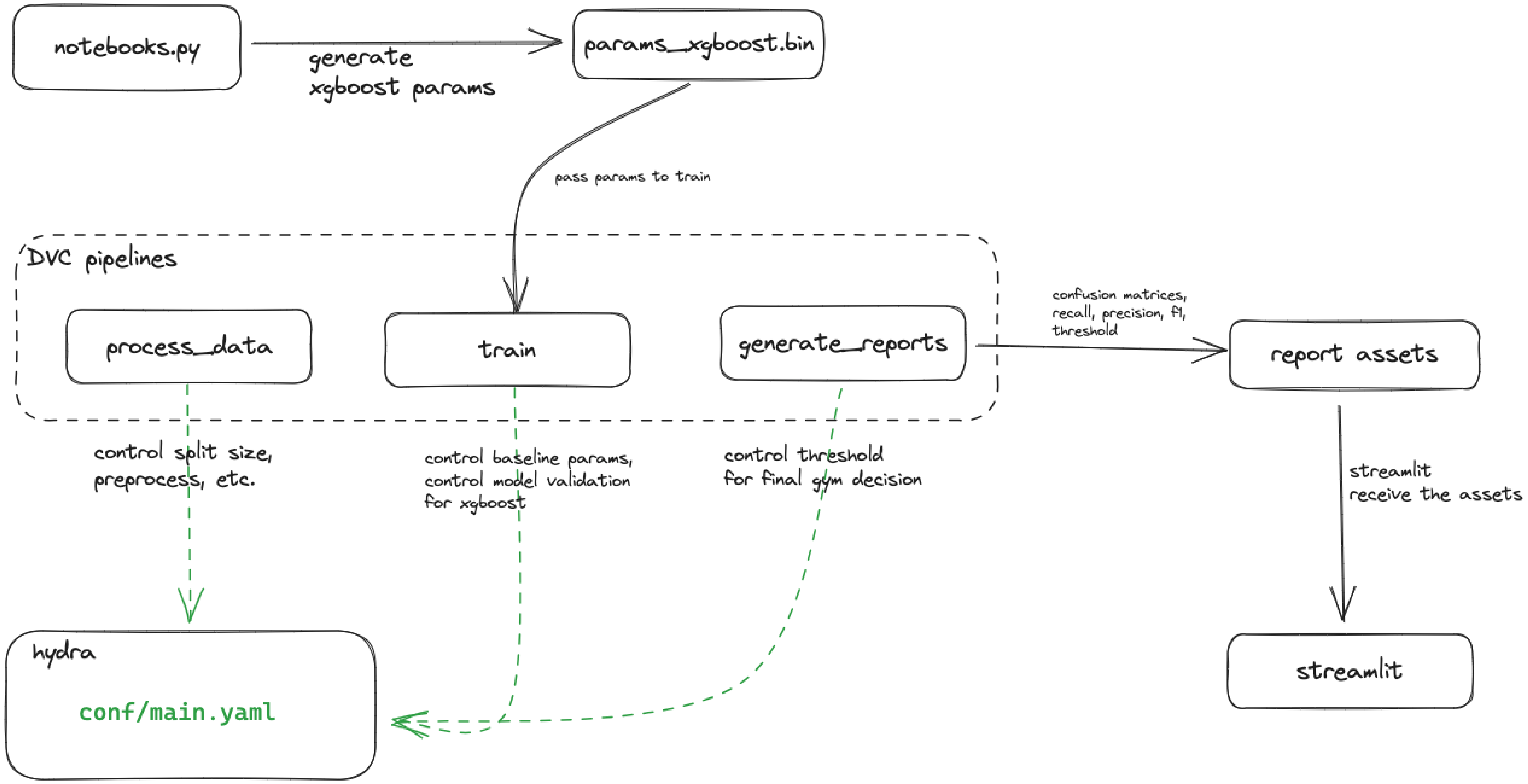
Code Refactor



For code refactoring we used:

- Hydra to use static params in yaml
- DVC to create pipelines
- Streamlit for report

Code Refator



Improvements

Try **aggregated** data by gyms instead of focusing in users → Transform in regression problem

Improvements

Use the features by **different windows** instead of just 60. To do this I would need the timestamp of each visit (or other interaction) to the gym

Improvements

Fix the confusion matrix (maybe the level is reversed)

Improvements

Use user **app interactions**: user search tokens, time using the app, time using other gym pass partnership apps (zenklub, etc)

Improvements

Use **gym location**, address, state, city, region. Maybe try to join **with public data** (ex: the financial health of the location, if its local is dangerous, number of stars in google maps)

Improvements

Get **RFM** and other **loyalty metrics (CLV, Customer Score, etc)** for each customer

Improvements

Use the distance of how far the visited gym is from user's home

Improvements

Use TVAE to synthesize churn data or other techniques (imbalance problem)

Improvements

Use number of upgrades/cancel/downgrades of each user in past.

Improvements

Model SHAP interpretation by sample cases

Improvements

Retrain in all database before predict to submission data

Improvements

Ensembles