Predicting User Churn

Anoosh Moutafian 2024

Waze provides satellite navigation software on smartphones

- Revenue generated via ad sales
- App relies on crowdsourced information
- 150 million monthly active users

Dataset: Synthetic dataset constructed by Waze

- 14,299 complete rows
- 13 features (most cover one-month time period)
- Data labeled churn/retain (reflects user behavior behavior during one-month period)

https://github.com/amoutafian/waze_capstone/blob/main/data/waze_dataset.csv

Problem Statement

User churn during dataset month

- 18%
- 2536 users churn /14299 total

Goal: Reduce user churn by 10%

- 18% $\longrightarrow$ 16.2%
- 2536 $\longrightarrow$ 2316 users churn
- 220 users retained/14299 total

Questions:

How can we predict user churn?

Which features contribute most strongly?

Results of successful retentions:

- Increased ad impressions at $.002 each
- Increased app quality with additional crowdsourced info

Data Cleaning, Exploration, Wrangling and Preprocessing

Of the original 14,999 observations (each representing a Waze user), 700 were removed due to missing churn information

Features were explored individually and in relationship to other features

The imbalanced churn/retain feature was balanced using random oversampling
18% → 33% churn rate

The data was split into 80% train and 20% test sets

Dummies were made from the single independent categorical feature

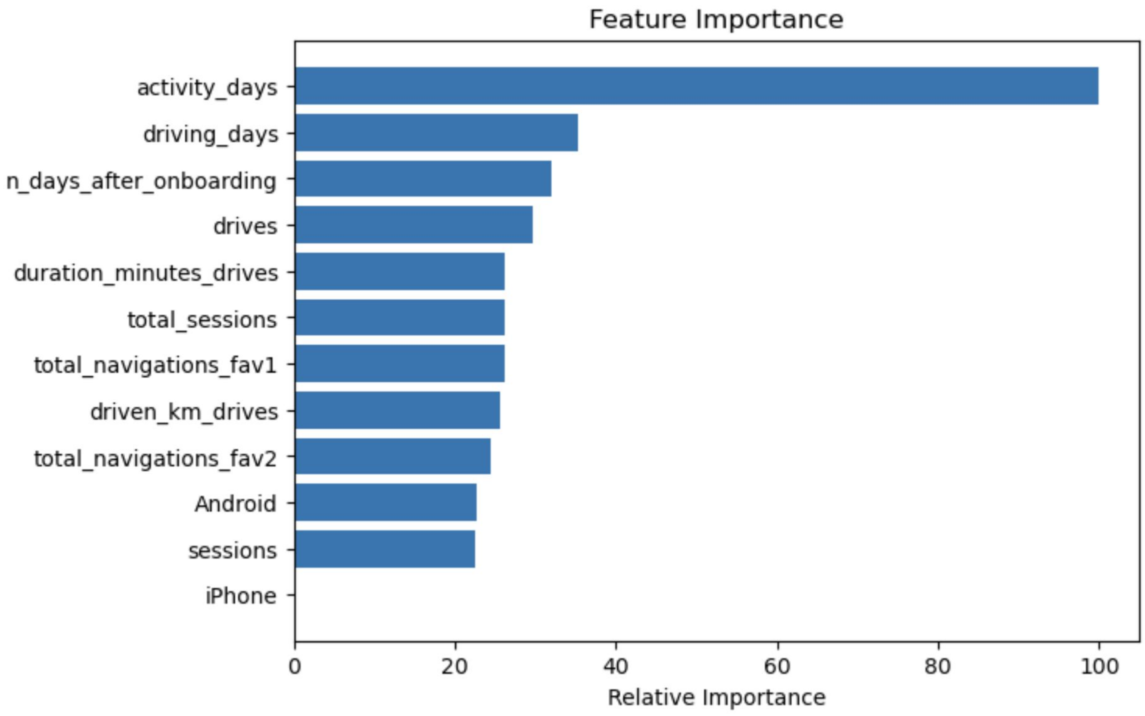
Numerical features were standardized

Relative feature importance as identified by XGB

activity_days: Number of days the user opens the app during the month

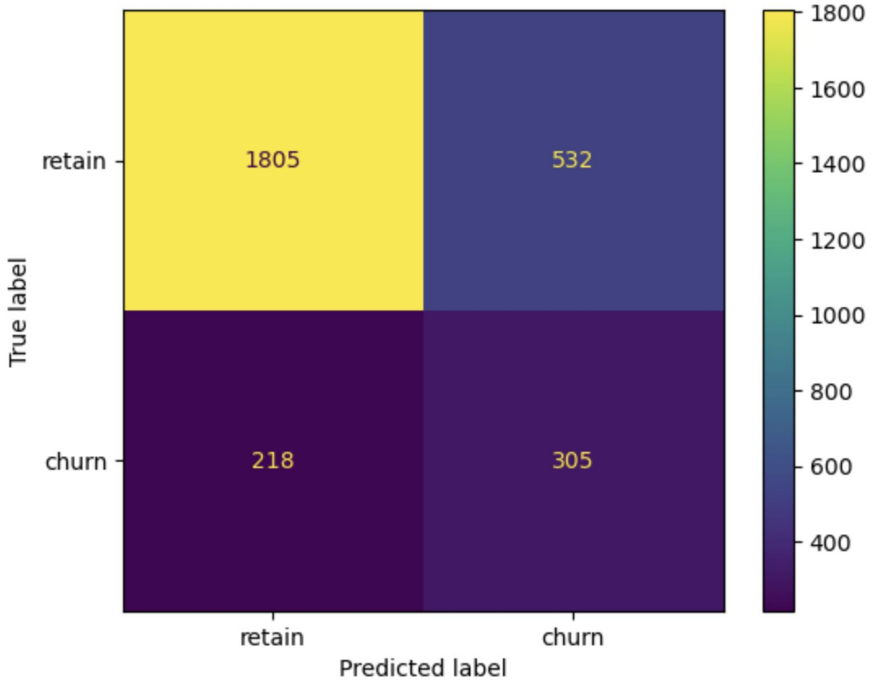
driving_days: Number of days the user drives >= 1 km during the month

n_days_after_onboarding: Number of days since user onboarding



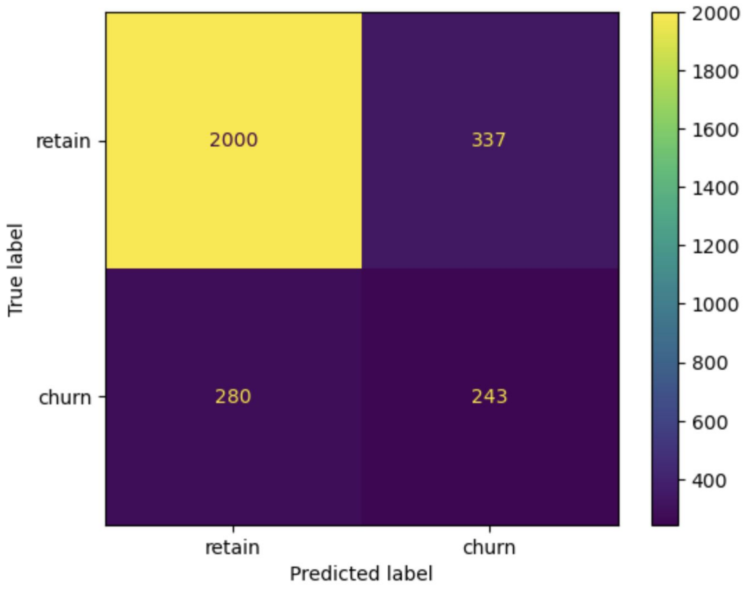
Random Forest had high accuracy (81%), but low recall (19%) meaning it wasn't great at predicting which users would churn

	precision	recall	f1-score	support
0	0.84	0.95	0.89	2337
1	0.48	0.19	0.27	523
accuracy			0.81	2860
macro avg	0.66	0.57	0.58	2860
weighted avg	0.77	0.81	0.78	2860



Logistic Regression had 74% accuracy and 58% recall, making the model of choice for this use case

	precision	recall	f1-score	support
0	0.89	0.77	0.83	2337
1	0.36	0.58	0.45	523
accuracy			0.74	2860
macro avg	0.63	0.68	0.64	2860
weighted avg	0.80	0.74	0.76	2860



Next Steps:

Tune gradient boost model hyperparameters

Convert data to DMatrix (XGBoost proprietary data structure)

Experiment with class balancing techniques