Final Report: Waze user churn
Anoosh Moutafian 2024

Problem Statement

Waze Mobile Ltd, released in 2006, doing business as Waze, is a subsidiary company of Google that provides satellite navigation software on smartphones and other computers. Free to download, the Waze app generates revenue through ad sales. Advertisers pay $.002 per impression. As the app relies on crowdsourced information, keeping user numbers up is especially critical. With 140 million monthly active users (as of 2022), moving the needle on churn rates even a few percentage points has large implications.

While user behavior is ultimately out of company control, this investigation aims to identify features associated with churn and suggest steps to reduce user churn by 10%. Of the 14,299 customers, 2536 (18%) churned. The goal is to reduce churn by 10%. Therefore, a successful outcome would be if during a similar period and population, only 16.2%, or 2316 customers churned.

Data Wrangling

A synthetic dataset constructed by Waze with 13 features was used as the basis of the analysis. Of the 14,999 original rows, 700 had to be discarded due to missing churn data. Remaining rows were complete with no nulls or duplicates with reasonable value ranges. Most columns covered the time period of one month. The churn/remain label reflects user behavior during this one-month period.

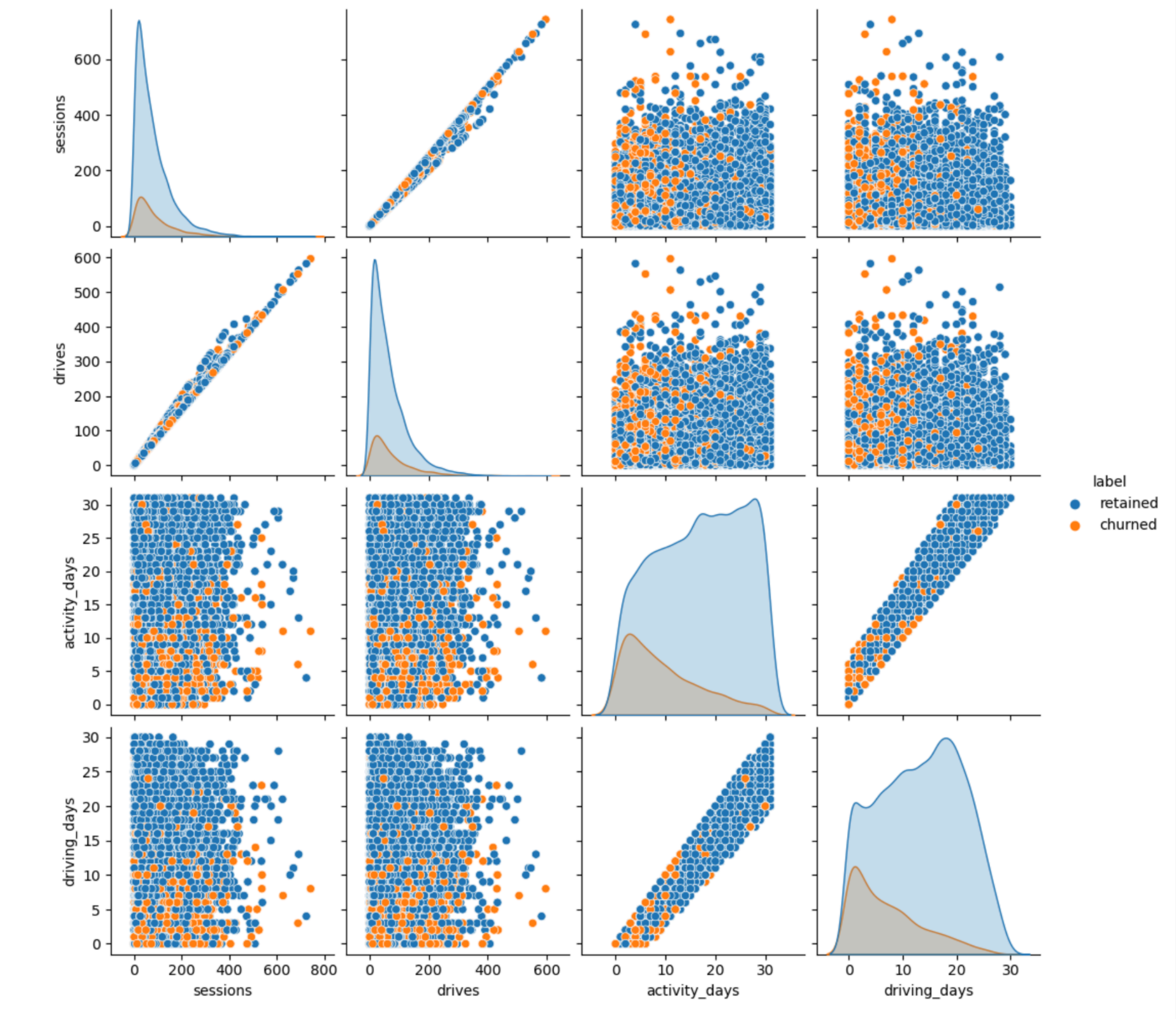
Exploratory Data Analysis

The features with the strongest relationship to the dependent variable seem to be “sessions”, “drives,” “activity_days” and “driving days." There is definitely some multicollinearity, but not sure how to address it.

"sessions" : The number of occurrence of a user opening the app during the month
"drives" : An occurrence of driving at least 1 km during the month

"activity_days" : Number of days the user opens the app during the month
"driving_days" : Number of days the user drives (at least 1 km) during the month

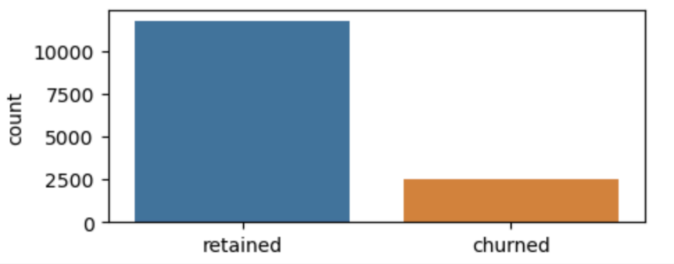
The sessions feature is highly related to the drives feature as they both measure the total number of times the app was used. Activity days and driving days are similarly related. Sessions is also highly related to activity days and drives to driving days as they are both the total use counts vs. days used counts.

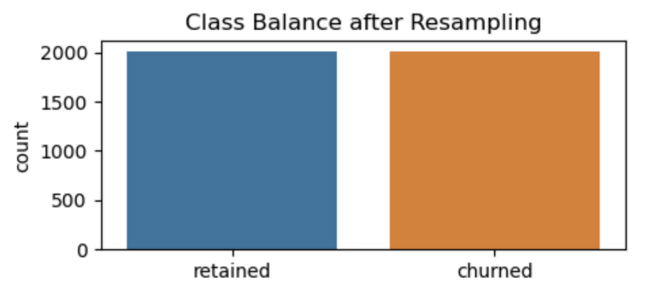


Dataset Limitations: This is not a time series, we have information about what happened during the month, but not when it happened. Is it that light users are more likely to churn, or is it that users who churn don't have a chance to use heavily as they have left the service mid-data collection?

Pre-Processing

The data is unbalanced. The churn/retention ratio is 2536/14299, or 18%. Random undersampling with imblearn was applied to balance the classes to 50% churn rate.





Dummy features were created for the single independent categorical feature ‘device’.

The data was split into testing and training datasets with a test size of 20%.

Numeric features were standardized using Standard Scaler.

Modeling

Logistic Regression, Random Forest, K Nearest Neighbors and Gradient Boost (XGBoost) models were used. Even without optimization, the gradient boost model gives the best results. As our primary business interest is to identify churn, we prioritize sensitivity (recall), the true positive rate.

The random forest model had very slightly higher scores for precision and the related score f1. It might be worthwhile to discuss how the model results will be used with stakeholders. If the planned intervention on customers at risk of churn is expensive or limited, elevating the importance of precision may be warranted.

Feature importance: As expected, driving_days and activity_days were the most important features as identified by both the random forest and gradient boost models. Gradient boost rated driving_days as the most important and random forest rated activity days as more important. Surprisingly, both models rated sessions as low importance. Drives ranked just below driving_days and activity_days by the gradient boost model.

Next steps

The gradient boost model could be optimized by tuning hyperparameters and converting data to DMatrix, an XGBoost unique data structure that can improve performance and efficiency.

Class balancing techniques could also be explored. Random undersampling was used, but resulted in value counts of only around 2,000 for each class.