Referral Propensity Project

Introduction

What is a propensity model you may ask? Propensity modelling is the application of mathematical models to **predict whether users will perform a particular action**. At Setel, propensity models could help marketers identify who among the audience is most likely to make a fuel or store purchase, top-up, refer a friend (RaF), etc. Marketers could then focus efforts and resources such as personalized products, messages and offers to give the identified users the nudge they need to pull the trigger. For this project, we will be focusing on **referral** action.

Problem Statement

The number of referrers and **referees** that originated from recent registration months has **decreased**. In other words, users from younger registration cohorts **refer lesser** than users of older registration cohorts.

After some investigations, we found out that younger cohorts had **lower retention rates** which attributed to a drop in referral count. The majority of users from younger cohorts are also attributed to *OGA* and *KPOGA*, which itself has lower retention rates after the first month.

To increase referrals of younger cohorts, we need to first increase the retention rate of younger cohorts. We believe this can be done through diversifying acquisition away from *KPOGA* and *OGA* to other channels with better retention rates such as **RaF**.

Proposed Solution

To increase RaF acquisitions, we can build a machine learning model that **predicts whether a user is likely to perform a referral action**. Each user will be assigned a **probability score**, making this a binary classification problem.

Objective

We aim to **increase RaF acquisitions** which would then **increase Monthly Engaged Users** (MEU).

Gathering Data

Sample of users

Firstly, we determine the userid(s) of the 2 groups of Setel customers:

1. Customers who have made **at least one** referral
2. Customers who have **never** referred

with the condition that these customers must have made **at least 3 fuel or store purchases of any amount** because we want to avoid picking users who have just recently registered but haven't gotten a chance to refer.

User attributes

After having the list of customers, we brainstorm on the characteristics of these users that might affect whether they refer or not.

Data	Attributes
Demographics	Gender, age, tenure, state, preferred station, fuel grade, vehicle type, payment method, etc.
Transactional	First and latest transaction date, frequency of fuel purchase and top-up, ticket size ($) for fuel and top-up, fuel volume purchased, etc.
Marketing	Acquisition channels

Feature Engineering

Feature engineering is a process of transforming the given data into a form that is easier to interpret by creating new input features from the existing ones.

For example, attributes `min_transaction_date_gmt8` and `max_transaction_date_gmt8` don't seem too useful when used independently. We can instead define an additional attribute `active` which indicates the number of days between the first and last transaction date, which brings more value. The table below shows other features which have been engineered.

Attribute	Derived from
`active`	`max_transaction_date_gmt8` - `min_transaction_date_gmt8`
`avg_days_btw_purchases`	`active` / `fuel_purchase_freq`
`avg_days_btw_topups`	`active` / `topup_freq`
`avg_fuel_volume`	`total_fuel_volume` / `fuel_purchase_freq`
`avg_fuel_spent`	`total_fuel_spent` / `fuel_purchase_freq`
`avg_topup_spent`	`total_topup_spent` / `topup_freq`

Exploratory Data Analysis (EDA)

a. Missing values

We noticed that a hefty 35% of records do not have information on `age` and `gender` but we cannot afford to drop these rows therefore we have to impute them. Age is imputed with the **average** age of the users from that preferred station. Besides, we also introduced a new class for `gender` - "**Undefined"** to replace records with missing `gender` information.

After imputing `age` and `gender`, we **drop the remaining records with missing values** as it is only ~2% of the dataset. Despite dropping those records, we still have a balanced dataset for classification.

b. Binning

To prevent overfitting, we attempt to reduce the number of unique values for categorical features especially for `stationid`, `channel`, `vehicle_type`.

Unfortunately, `stationid` consists of too many unique values thus we will be dropping the feature itself.

For model improvement, we can try clustering the stations based on latitude and longitude.

For `channel`, we will group channels with fairly similar operations. We will also be combining the minority channels into one and name it ***Others***. The remaining channels will remain as standalone channels.

For `vehicle_type`, we group *luxury cars* and *regular cars* as **cars**; *premium bikes* and *regular bikes* as **bikes**; *pick-up trucks*, *light commercial vehicles*, and *heavy trucks* are classified as **commercial vehicles**.

Description of Features

This is the final dataset that we'll be using for the training and testing of machine learning models.

Feature	Data Type	Description
`refer`	integer	Binary indication of whether the user has ever referred
`age`	float	User's age

gender	object	User's gender
tenure	integer	Number of days elapsed since registration date
channel	object	Acquisition channel attributed to the user
state	object	State that the user lives in (mode)
fuel_grade	object	User's preferred fuel grade (mode)
vehicle_type	object	User's predicted vehicle type (mode)
paymentmethod	object	User's preferred payment method (mode)
avg_days_btw_purchases	float	Average number of days between purchases
avg_days_btw_topups	float	Average number of days between top-ups
avg_fuel_volume	float	Average fuel volume per transaction
avg_fuel_spent	float	Average RM spent on purchasing fuel
avg_topup_spent	float	Average RM spent on top-up

Train-Test Split

The dataset is to be divided into 2 subsets, with the first subset being used to fit the machine learning model (**train dataset**) and the second subset being used as input to the model in which the predictions made are compared to the expected values (**test dataset**).

For this project, we will use a split percentage of **70%-30%** (70% train, 30% test). This resulted in train and test datasets of sizes *180,185* and *77,223* respectively.

Data Pre-processing

After obtaining our train dataset, we will transform both numerical features and categorical features from the train dataset in order to better fit some of the machine learning models.

Numerical features will be **standardized** (rescaled to ensure *mean* and *standard deviation* to be *0* and *1* respectively) using *scikit-learn*'s `StandardScaler()` function.

Categorical features will be One-Hot Encoded which each category value is converted into a new column and assigned a 0 or 1 (notation of true/false) using *scikit-learn*'s `OneHotEncoder()` function.

Model Fitting

We're finally ready to fit the machine learning models. We'll experiment with multiple baseline machine learning algorithms below:

- Logistic Regression
- Support Vector Machines
- k-Nearest Neighbours
- Naive Bayes
- Extreme Gradient Boosting (*XGBoost*)
- Categorical Gradient Boosting (*CatBoost*)

The *CatBoost* model is the only model that was trained using the **non-pre-processed** train dataset.

Model Performance

Model	Precision	Recall	f1-score	Training Time
Logistic Regression	0.66	0.66	0.66	~ 1 minute
Support Vector Machines	0.69	0.69	0.69	~ 1 hour
k-Nearest Neighbours	0.65	0.65	0.65	~ 1 second
Naive Bayes	0.62	0.62	0.61	~ 1 second
Extreme Gradient Boosting	0.69	0.69	0.69	~ 1 minute
Categorical Gradient Boosting	0.71	0.70	0.70	~ 5 minutes

Model Evaluation

Categorical Gradient Boosting (CatBoost) seems like the best performing model with a reasonable model training time. CatBoost also allows us to skip data pre-processing as there is no need to transform categorical features into numeric features (one-hot-encoding). Thus, we will be able to use the dataset in its original format, which is simpler and more conducive for deployment.

	Predicted: 0	**Predicted 1**
Actual: 0	25,200 (65%)	13,393 (35%)
Actual: 1	9,463 (24%)	29,167 (76%)

Sensitivity - 29,167 / (29,167 + 9,463) = 0.75 (75%)

Specificity - 29,167 / (29,167 + 13,393) = 0.69 (69%)

In conclusion, we are able to engage with **75%** of customers who **will make a referral** but missing out 25% for sure. Out of all the customers who are predicted to refer, **31%** are **incorrectly** predicted to refer. If the goal is to send out special incentives such as credits or loyalty points to encourage referral, it's fine even to engage with those who are unlikely to refer as it does not cause any negative problem.