COURSEWORK 1

Churn Prediction for FoodCorp Report

Machine Learning and Predictive Analytics, 2022/23

University of Nottingham, Nottingham, United Kingdom

Student ID: 20494689

Module Code: BUSI4373

Executive Summary

This executive summary presents the findings of a churn prediction system using time-series customer transactional data of FoodCorp. Before building a churn prediction model, a churn definition is proposed that defines when a customer is considered churned or not churned. Based on this churn definition, a churn prediction system was built using Gradient Boosting (XGBoost) machine that combines multiple weak predictors to create a strong predictor. Upon initial investigation, it was found that there is a huge difference between the number of active customers and churned customers which is common in a churn prediction task. This was addressed by using imbalanced class techniques that balance them out. Moreover, cross-validation and hyper-parameter tuning techniques were used to generalise well to new data as well as optimize the performance of the model.

Based on the business case, it is assumed that the cost of losing a churning customer is higher than investing in customers who are not going to churn. Therefore, the churn prediction system is prioritised to increase the accuracy of predicting customers that are going to churn more than that of predicting non-churning customers. Finally, a pen portrait of churning and non-churning customers is drawn to aid with future marketing strategies.

1. Current Level of Churn

Customer churn is the loss of customers over a given time. In the case of FoodCorp, churn is defined as the period of inactivity in a customer's purchase behaviour. Therefore, the churn is formally defined as:

$$\boldsymbol{If}\ \text{date\_today} - \text{date\_of\_last\_purchase} > \text{defined\_period\_of\_inactivity}\ (\beta)\ \textbf{then}\ \text{churn}\ (1)\ \textbf{otherwise},\ \text{not churn}\ (0)$$

The current churn level indicates that out of 8,997 customers, there are 2611 (29%) active customers and 6386 (71%) churned customers.

In this business case, β is defined as 39 days because it captures 59.98% of customers visiting within this period and targets 13.07% of the active customers. This is defined by assuming that generally in a retail environment, customers tend to purchase weekly or monthly. In this case, 1 week of inactivity captures only 25.72% of the total customers but 39 days of inactivity captures more than 50% of total customers. Moreover, choosing a very high inactivity period also has its caveats that is the churn prediction system might ignore the seasonality nature of customer purchase behaviour and overshadow them resulting in an inconsistent performance of the churn prediction model.

2. Churn Prediction System

Feature Engineering:

After performing exploratory Data Analysis and deciding on the churn definition, an intuitive feature engineering was carried out in two stages, resulting in a single temporal data frame with 11 features describing customer purchase behaviour. Feature engineering completely done using PostgreSQL and was done in two stages.

In the first stage of feature engineering, a non-temporal data frame ("engineered_ml_data") was constructed from the "receipts" and "receipt_lines" table that captures the static customer purchase behaviour. During this process, the date format of "purchased_at" was changed to integer format taking the oldest date as 1. This was done because machine learning models cannot process data type format. Since "purchased_at" only contains the dates when a purchase was made, it was then processed to contain all dates between the oldest date and the latest date in the dataset. This was done for future aid in creating a temporal data frame.

Feature name	Description	Data type
customer_id	Unique ID of a customer	Integer
purchased_day	Day of record	Integer [1-609]
day_of_week	Day of the week	Integer [0-6] 0 = "Sunday" 6 = "Saturday"
day_of_month	Day of the month	Integer [1-31]
month_of_year	Month of the year	Integer [1-12] 1 = "January"

		12 = “December”
visits	Total visits on the day of record	Integer
quantity	Total quantity on the day of record	Float
value	Total value on the day of record	Float

Table 2.1: Non-temporal feature description

Using the non-temporal data, total visits per day were calculated to identify the time-series sales activity of FoodCorp. It was noticed that the sales during the first 85 days were lower than the median sales activity per day. Therefore, these days were omitted from the training of the machine learning model as it does not represent the sales activity after 85 days.

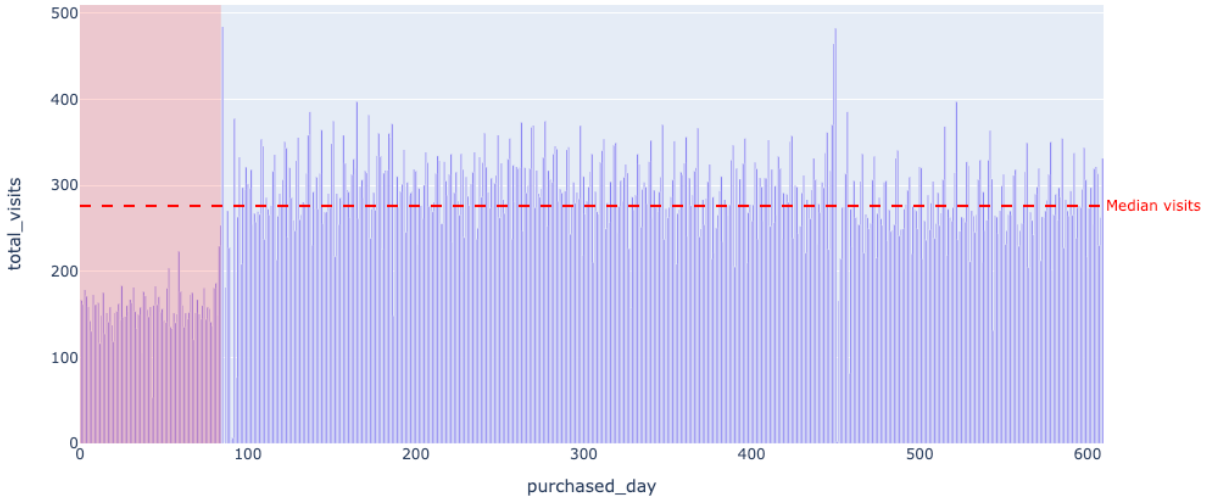

Fig 2.1: shopping activity of FoodCorp

In the final stage of feature engineering, a temporal data frame (“temporal_ml_data”) with input and output labels is constructed from the non-temporal data frame constructed in stage one. This will be the training data for the churn prediction system.

Feature name	Description	Data type
customer_id	Unique ID of a customer	Integer
Ref_day	Assumed present day for an instance	Integer [145-602]
day_of_week	Day of the week	Integer [0-6] 0 = “Sunday” 6 = “Saturday”
day_of_month	Day of the month	Integer [1-31]
month_of_year	Month of the year	Integer [1-12] 1 = “January” 12 = “December”
recent_purchased_day	The recency of a purchase	Integer
total_visits	Frequency of purchases	Integer
customer_age	Days since the last purchase made by a customer	Integer

purchase_since_2m	the customer spent in the last 2 months from ref_day	Float
monetary_value	Average money spent by customer till ref_day	Float
churn	Whether the customer churned on ref_day or not	Binary integer 0 = “active” 1 = “churn”

Table 2.2: Temporal training data feature description for churn prediction system

These features are constructed based on Recency, Frequency and Monetary Value (RFM) methodology. RFM analysis is a marketing method used to segment customers based on their transactional data and analyse customer behaviour. The period since a customer's previous transaction is referred to as recency. Monetary value is the average amount of money spent by a client on purchases, while frequency is the total number of transactions a customer has made during a specific period. Monetary value does not reset when a customer is churned and then later returns as storing them would indicate the value of a returning customer and also the effectiveness of past marketing campaigns.

Moreover, the day of the week, the day of the month and the month of the year is constructed to incorporate the seasonality of customers' purchase behaviours and finally the purchase value since the last 2 months is calculated as a feature to know whether customers would come back within 2 months.

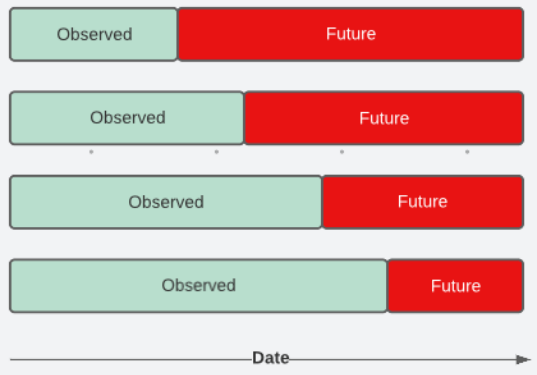

Fig 2.2: Observed and Future Sliding Windows

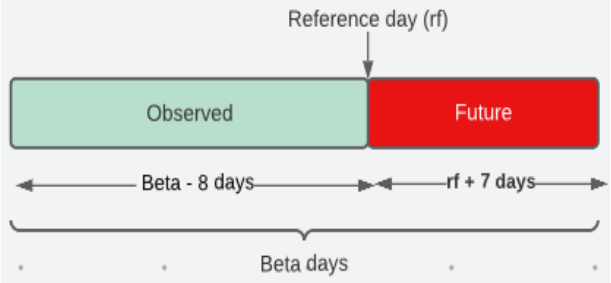

Fig 2.3: Churn implementation

The way features in Table 2.2 are created is by keeping the end of the observed day as the reference day and then using the historic data available in the observed region of “engineered_ml_data” to calculate the input features. The output is calculated by observing if there has been any purchase in the last 32 days and the next 7 days so that the period of calculation aligns with the churn definition of 39 days.

This process is performed recursively by sliding the reference day across the dataset while calculating their corresponding features. Therefore, at any instance time, a machine learning model can be trained using observed data and make predictions on future data. In this churn prediction model, the churn is predicted every week so the prediction data will always be 7 days ahead of the training data.

For each reference day (rf) till $present\ day - 7$:

- Compute input features for all customers on that reference day using the historic data $(start\ day -> rf)$.
- Compute churn label based on the churn definition (β) that is if a purchase was made within the last $\beta - 8$ days from rf to $rf + 7$ days or not.
- Slide reference day $(rf = rf + 1)$ by 1.

Finally, a complete dataset with each day as a reference day and their corresponding input and output features are constructed.

After the construction of the temporal data, it was observed that for a few days, all the features have a value of 0, which means that the customer did not start purchasing in FoodCorp, Therefore, these records were removed from the temporal dataset as leaving them unattended might skew churn prediction model.

Prediction approach:

A binary classification machine learning model is used to build the churn prediction system where churn is defined as 1 and not churn is defined as 0 (Table 1.2). Before training any binary classifiers, a baseline model is defined using a decision tree classifier. Any model that is chosen for churn prediction must perform better than the baseline model to qualify for further investigation. Two ensemble models are evaluated for churn prediction that is Random Forest classifier and the Xtreme Gradient boosting classifier (XGBoost) which is a regularized version of the gradient boosting algorithm.

Random Forest classifier is a bagging algorithm that aggregates the prediction power of multiple decision trees to make a more robust and generalised machine learning model.

On the other hand, XGBoost is a boosting algorithm that combines multiple weaker models, typically decision trees to make a more robust and accurate prediction model. Gradient boosting algorithms try to improve a weaker model by iteratively adding decision trees to reduce the error of the previous tree. Both have their caveats like both models are prone to overfitting so their hyper-parameter must be tuned accordingly. Like most ensemble models, Random Forest and XGBoost are black box models that can be difficult to interpret.

The prediction approach is subdivided into smaller steps that involve an imbalanced class problem, base model evaluation, cross-validation, hyper-parameter tuning, final model evaluation and model interpretation.

From the initial investigation, it was evident that the ratio of active customers and churned customers is very high. This is bad for a machine learning model as it introduces bias by developing more prediction accuracy towards the majority class and as a result, the model is not generalised and might perform poorly on new data. To handle this issue a resampling technique called SMOTE (Synthetic Minority Over-sampling Technique) is used to over-sample the minority class. This is only performed on the training data after it is split into train, validation and test set to build the model otherwise the resampled data might get included in the validation and test set. After resampling, the base Decision tree, Random Forest and XGBoost were evaluated and XGboost performed the best, closely followed by Random

Forest and Decision tree. The best-performing base model was chosen for further evaluation and model deployment.

Cross-validation was performed using the XGBoost classifier with multiple training-validation sets and a single test hold-out set. Training-validation set was built in such a way that the validation set is always 7 days ahead in time compared to the training set rather than a traditional random split since the goal is to always predict in time. While training, the size of the training set is 100 days of historic data.

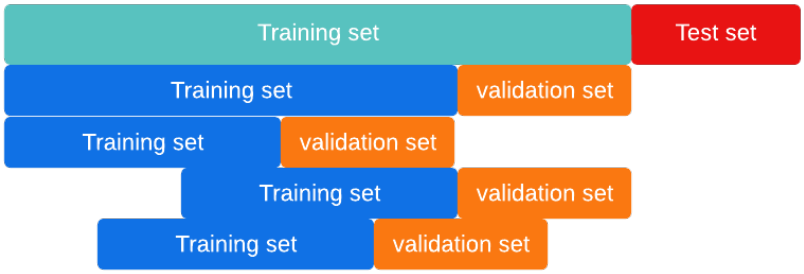

Fig 2.3: Training validation and test split

Hyper-parameter tuning is performed using the Python Optuna framework because of its smart parameter value selection and pruning capabilities which reduces the tuning time significantly. The hyper-parameter of each training validation set is tuned so that each cross-validation delivers the best-performing model for that training validation set. Parameters of the best-performing model are then chosen for further testing and deployment.

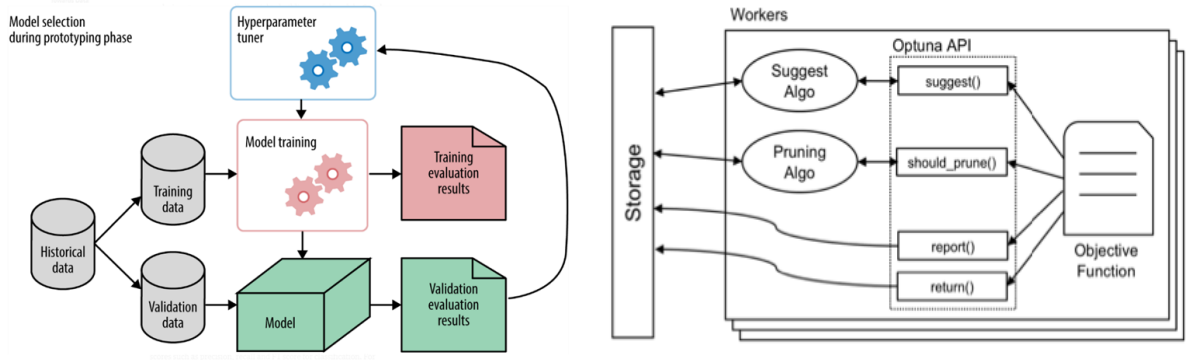

Fig 2.4: Hyper-parameter tuning process (S, 2020) and Optuna workflow (Akiba et al., 2019)

Hyper-parameter	Description
n_estimators	Number of boosting rounds
learning_rate	Boosting learning rate (xgb's "eta")
max_depth	Maximum tree depth for base learners.
min_child_weight	Minimum sum of instance weight(hessian) needed in a child.
gamma	Minimum loss reduction required to make a further partition on a leaf node of the tree.
subsample	Subsample ratio of the training instance.
colsample_bytree	Subsample ratio of columns when constructing each tree.
reg_lambda	L2 regularization term on weights (xgb's lambda).

scale_pos_weight	Balancing of positive and negative weights.

Table 2.3: XGBoost tuned hyper-parameters

The best-performing model is chosen and trained with the complete training data and evaluated against the hold-out test data. The Performance is evaluated based on the model's capability to correctly identify churned customers because misclassifying a churned customer is a costly act for FoodCorp might lose misclassified customers. Whereas misclassification of active customers is not as costly because there is less chance that such customers will be lost. Therefore, Recall is the most important metric to evaluate the performance of the model.

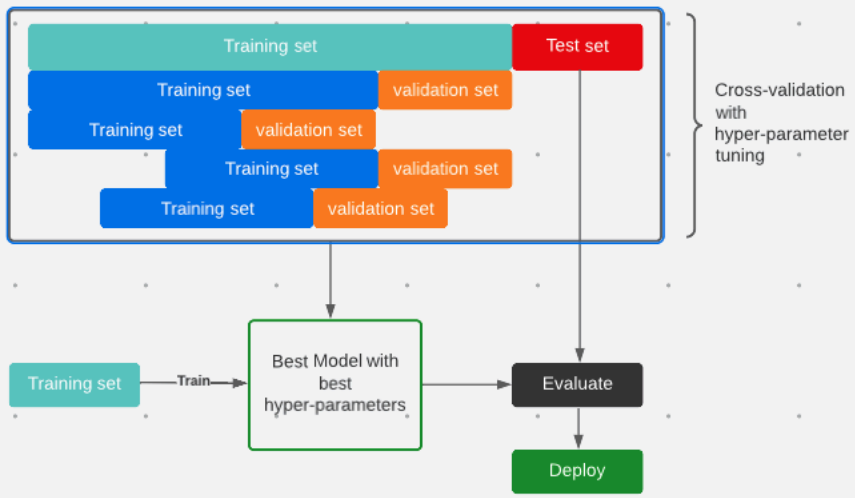

Fig 2.5: Model development workflow

3. Churn vs non-churn customers

In this comparative study of churn and non-churn customers, a few characteristics were used such as total average visits, total average monetary value, total average customer age, store activity per day and favourite products. The goal of these metrics is to provide an overview of churn and non-churn customer behaviours that can help improve future marketing strategies.

Upon examining the total average visits per day, it was found that non-churn customers visit the stores way more than the churn customers. It might indicate that the non-churn customers are frequent and loyal and the non-churn customers might be casual shoppers. There is a frequent change in the total average monetary value per day of churn and non-churn customers indicating that there are times when FoodCorp is losing valuable customers who are willing to spend more on the products, but they could also be seasonal shoppers who buy products only during a certain time of the year. Therefore, a survey of churn customers might help identify the reason behind such events.

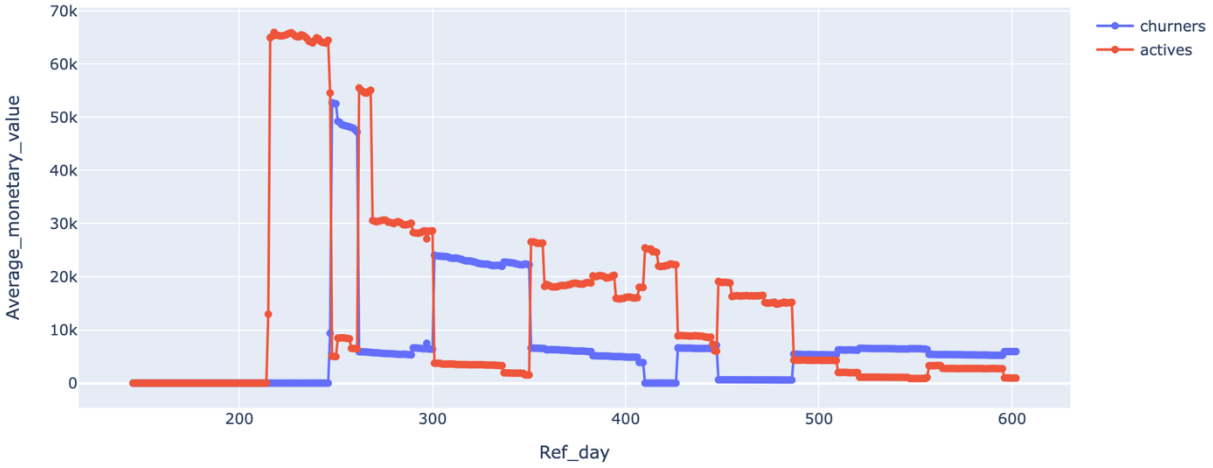

Fig 3.1: Total average monetary value

The total average customer age suggests that non-churn customers tend to do longer business in a single stint as compared to churn customers. While examining the store activity per day, it was found that recently there has been a significant decrease in the Nottingham and Birmingham store, which indicates that most customers are churning from these stores. Therefore, an investigation into recent changes in these stores could be beneficial. Both churn and non-churn customers have similar product demand, which could be further investigated to why customers with similar demands churn.

Churn customer	Non-churn customer
Tend to be less active and make fewer purchases as compared to non-churn customers.	Tend to be more active and make frequent purchases as compared to churn customers.
High monetary value customers, might indicate high value purchases.	Also has high monetary value customers who might be regular customers.
Many churn customers originated from Nottingham and Birmingham stores indicating a potential service dissatisfaction.	Similar product demands ensuring high quality products.

Table 3.1: Churn and non-churn customers pen portrait

The technicality of these metrics:

These metrics were produced using the engineered machine learning training data as well as the receipts and receipt_lines table. These insights were mainly derived from exploratory data analysis and domain knowledge. All these insights were plot using Python Plotly for an interactive interface.

References:

1. S, S. (2020). Hyperparameter Tuning using Optuna. [online] Analytics Vidhya. Available at: https://www.analyticsvidhya.com/blog/2020/11/hyperparameter-tuning-using-optuna/.
2. Akiba, T., Sano, S., Yanase, T., Ohta, T. and Koyama, M. (2019). Optuna: A Next-generation Hyperparameter Optimization Framework. [online] Available at: https://arxiv.org/pdf/1907.10902.pdf.