2021

Machine Learning Capstone Project Proposal

Optimize Starbucks App Promotion Offering Strategy

Elaine Liu

I. Domain Background

Starbucks App Success Story

Since its initial debut in 2011 and complete country-wide rollout in 2015, the Starbucks app has been a success story of how mobile apps are able to drive customer loyalty and customer-brand interactions.

According to a survey by Numerator in 2019, around 60% of Starbucks guests used the company's app, with the majority of those using it to order ahead or pay in store. The survey also suggests that the frequency of Starbucks visits appeared to be correlated with app usage - app users were roughly 2 times more likely to visit multiple times per week, 5 times more likely to visit daily, and, astonishingly, 10 times more likely to visit multiple times per day.[1]

During 2020, when physical store sales were significantly impacted by the outbreak of COVID-19, almost a quarter of all transactions at Starbucks stores in the US were mobile orders through the company's app.[2]

The Starbucks App helps the company better position itself for evolving consumer preferences

One of the challenges Starbucks faces is maintaining and further developing its ability to retain and convert customers with evolving preferences and tastes in an industry where new products are constantly introduced to the market.

The Starbucks App provides valuable data associated with customers' individual preferences and purchasing behaviors, through which the business is able to craft personalized product offerings and promotions.

II. Problem Statement

Starbucks sends out promotion offers to users through the Starbucks App once every few days. A promotion can be an advertisement for a drink or an actual offer such as a discount or BOGO (buy one get one free). Some users may not receive any offers during certain weeks. Not all users receive the same offer. Every offer has a validity period before it expires.

There are varying customer profiles and behaviors. Some customers might only be receptive to certain types of promotions but not the others. Some customers might make a purchase regardless of whether they received the offer.

---

[1] Yee, Chris. "Mobile Mastery: Insights into the Starbucks App." Numerator, 30 Oct. 2019, www.numerator.com/resources/blog/mobile-mastery-insights-starbucks-app.

[2] "2020-Starbucks-Annual-Report.pdf" https://s22.q4cdn.com/869488222/files/doc_financials/2020/ar/ 2020-Starbucks-Annual-Report.pdf

Because there are costs associated with each promotion, the goal of this project is to devise an optimized strategy using machine learning techniques that drives the highest return on each offer sent by identifying the offers with which customers are most likely to engage.

III. Datasets and Inputs

This project utilizes three types of data.

Portfolio

Portfolio data contains offer IDs and metadata about each offer (e.g. duration, type).

- ID: offer ID
- Offer Type: type of offer (i.e. BOGO, discount, informational)
- Difficulty: minimum required spend to complete an offer
- Reward: reward given for completing an offer
- Duration: time for offer to be open, in days
- Channels: whether the offer is sent through web email, mobile, or social channels

Profile

Profile data contains demographic data for each customer.

- Age: age of the customer
- Became member on: date when customer created an app account
- Gender: gender of the customer
- ID: customer ID
- Income: customer's income

Transcript

Transcript data contains records for transactions, offers received, offers viewed, and offers completed.

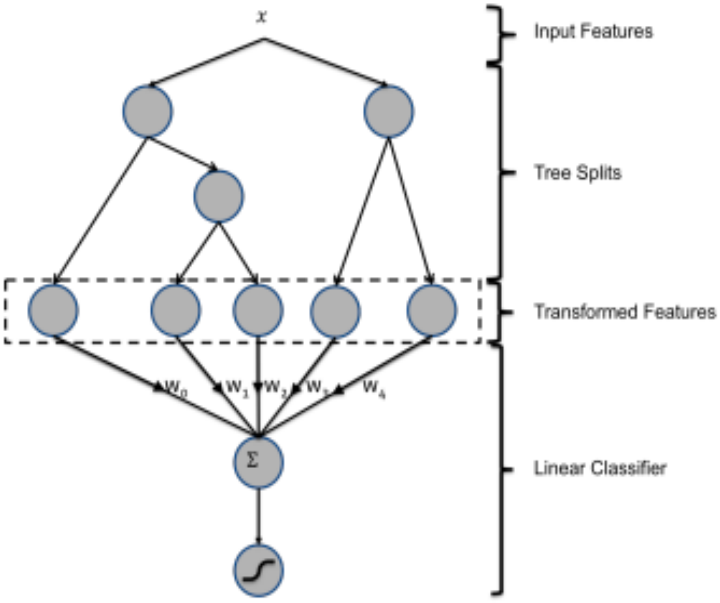
- Event: record description (e.g. transaction, offer received, offer viewed)
- Person: customer ID
- Time: time in hours since the start of the test. The data begins at time t=0
- Value: either an offer ID or transaction amount depending on the record

IV. Solution Statement

To drive the highest return for each promotion offer, it is essential to predict whether a customer would click into or view a promotion offer or not. Our problem then is essentially a problem of predicting the view rate, a binary classification problem that categorizes data into two buckets: “view” or “not view.”

This project aims to explore the implementation of a hybrid model that combines gradient boost decision trees (GBDT) with logistic regression (LR), as proposed by the Facebook research paper ("Practical Lessons from Predicting Clicks on Ads at Facebook")[3] to predict whether a customer would view a promotion offer.

Hybrid GBDT & LR Model Architecture



Benchmark Model

This project would compare the hybrid model performance against the performance of using the LR model alone and using the GBDT model alone.

Evaluation Metrics

The evaluation metrics would be log-loss, recall, precision, and F1-score.

The aforementioned Facebook paper used Normalized Cross Entropy (equivalent to the average log loss per impression divided by what the average log loss per impression would be if a model predicted the background click through rate (CTR) for every impression) as one of its evaluation metrics. For the simplicity of this project, log-loss would be used as the evaluation metric, along with other traditional classification metrics such as recall, precision, and F1-score.

Note that although there is a cost associated with every offer sent, the business would not want to miss any customers who are likely to act on the offer after receiving it, even if it means there

[3] He, Xinran, Stuart Bowers, Joaquin Quiñonero Candela, Junfeng Pan, Ou Jin, Tianbing Xu, Bo Liu, et al. "Practical Lessons from Predicting Clicks on Ads at Facebook." Proceedings of 20th ACM SIGKDD Conference on Knowledge Discovery and Data Mining - ADKDD'14, 2014. https://doi.org/10.1145/2648584.2648589.

would be a higher chance of sending offers to customers who would not be influenced by the offer. In other words, we would prioritize minimizing false negatives. Thus, metrics invariant to classification-threshold like AUC would not be optimal for our use case.[4]

V. Project Design

Data Exploration and Visualization

Load data and normalize tables to establish relationships between Portfolio, Profile, and Transcript data tables.

Perform data visualization on each individual data table as well as the normalized table to uncover general characteristics of customers, offers, transactions, and the interactions among them.

Data preprocessing

Clean data - remove transactions where customers made a purchase without receiving an offer, as those are irrelevant for our project.

Create target label - assign 1 if the offer was received and viewed, and 0 otherwise.

Handle missing values and perform feature scaling, standardization and normalization, if required.

Split data into training and testing sets based on time

Since transaction data is collected over time, instead of randomly splitting the data into training and testing sets, the data would be split based on time. Before a certain time threshold, the data would be included in the training set, and beyond that, the data would be included in the testing set. This way the model would be trained on historical data and avoid making predictions on past data. The threshold would be set so as to ensure that the amount of data in training vs testing is roughly 3:1.

In addition, data splitting would be performed before feature engineering because some features would be calculated based on transaction data over time (see explanation in Feature Engineering section below), and using testing data to calculate these features would lead to data leakage.

Feature Engineering

Create additional features related to offers and customers derived from transaction data that serve to uncover underlying characteristics of offers and customers that are not represented in

[4] "Classification: ROC Curve and AUC; Machine Learning Crash Course." Google. Google. Accessed March 2, 2021. https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc.

the Portfolio and Profile. For example, average time for the offer to be viewed across all customers, average view rate of offers, average time for the customer to view an offer, average transaction amount of a customer.

These features would be calculated based on training (historical) data only to avoid data leakage - the model would only use past information to make predictions on future data.

Lastly, these features would be directly combined to the testing data so that training and testing have the same set of features.

VI. Project Design

Hybrid GBDT & LR model

Step 1: Split training data into training and validation
Step 2: Train GBDT (LightGBM) using Bayesian Optimization
Step 3: Use the trained tree to transform the original features to categorical features (each individual tree is a categorical feature that takes as value the index of the leaf an instance ends up falling in[5]) for both training and testing data
Step 4: Perform one-hot encoding on features created by GBDT on training and testing data
Step 5: Combine the one-hot encoded features created in Step 4 with the original features
Step 6: Train and tune LR
Step 7: Use the trained LR to make predictions on testing data
Step 8: Evaluate model performance

Benchmark model (GBDT-only; LR-only)

Step 1: Train and tune the benchmark model (i.e. GBDT, LR) using original features
Step 2: Use the trained benchmark model to make predictions on testing data
Step 3: Evaluate model performance

Model Performance Evaluation

Compare the performance of the hybrid model, GBDT-only model, and LR-only model based on evaluation metrics.

[5] He, Xinran, Stuart Bowers, Joaquin Quiñonero Candela, Junfeng Pan, Ou Jin, Tianbing Xu, Bo Liu, et al. "Practical Lessons from Predicting Clicks on Ads at Facebook." Proceedings of 20th ACM SIGKDD Conference on Knowledge Discovery and Data Mining - ADKDD'14, 2014. https://doi.org/10.1145/2648584.2648589.