Machine Learning Engineer Nanodegree

Starbucks Capstone Project Proposal

Deepa Balakrishnan

January 13th, 2022

Table of Contents

I. Domain Background

Starbucks is one of the most well-known companies in the world: a coffeehouse chain with more than 30 thousand stores all over the world. It strives to give their customers always the best service and the best experience. As a side feature, Starbucks offers their free app to make orders online, predict the waiting time and receive special offers. This app also offers promotions for bonus points to these users. The promotional offer can be merely an advertisement for a drink or an actual offer such as a discount or BOGO (buy one get one free). This project is focused on tailoring the customer behavior and responses while using the Starbucks mobile app. To avoid churn rate and trigger users to buy Starbucks products, it is important to know which offer should be sent to specific users.

To study about application of machine learning to predict customer churn, I used the reference: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3835039.

II. Problem Statement

The problem we are trying to solve here is, given a promotional offer and the users demographics, how likely is it that the user will be prompted by the offer. Hence this is a classification problem. We will be predicting the probability with which the customer will take up an offer or not. Hence it will help the Starbucks to minimize churn rate.

III. Dataset and Inputs

There are 3 available data sources as mentioned below:

1. The first one is **portfolio**: it contains list of all available offers to propose to the customer. Each offer can be a *discount*, a *BOGO (Buy One Get One)* or *Informational* (no real offer), and we've got the details about discount, reward, and duration of the offer.
2. The next data source is **profile**, the list of all customers that interacted with the app. For each profile, the dataset contains some personal information like gender, age, and income.
3. Finally, there is the **transcript** dataset: it has the list of all actions on the app relative to special offers, plus all the customer's transactions. For each record, we've got a dictionary of metadata, like offer_id and amount spent.

Here is the schema and explanation of each variable in the files:

portfolio.json: Offers sent during 30-day test period (10 offers x 6 fields)

- reward: (numeric) money awarded for the amount spent
- channels: (list) web, email, mobile, social
- difficulty: (numeric) money required to be spent to receive reward
- duration: (numeric) time for offer to be open, in days
- offer_type: (string) bogo, discount, informational
- id: (string/hash)

profile.json: Rewards program users (17000 users x 5 fields)

- gender: (categorical) M, F, O, or null
- age: (numeric) missing value encoded as 118
- id: (string/hash)
- became_member_on: (date) format YYYYMMDD
- income: (numeric)

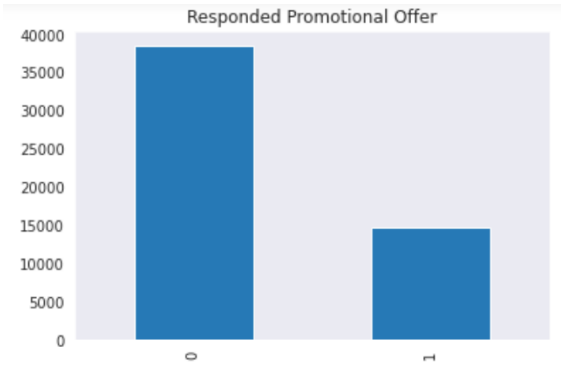
transcript.json: Event log (306648 events x 4 fields)

- person: (string/hash)
- event: (string) offer received, offer viewed, transaction, offer completed
- value: (dictionary) different values depending on event type
    - offer id: (string/hash) not associated with any "transaction"
    - amount: (numeric) money spent in "transaction"
    - reward: (numeric) money gained from "offer completed"
- time: (numeric) hours after start of test

The portfolio.json contains offer_type column, which describes the types of offers that Starbucks is looking to send its customers:

1. BOGO (Buy-One-Get-One): A user needs to spend a certain amount to get a reward equal to that threshold amount.
2. Informational: There is no reward, but neither is there a requisite amount that the user is expected to spend.
3. Discount: A user gains a reward equal to a fraction of the amount spent.

Offers can be delivered via multiple channels.



From the visualization above, it is possible to notice that we are dealing with an **imbalanced dataset**, since approximately 28% of the dataset responded to the promotional offer.

IV. Solution Statement

As mentioned earlier, since this is a classification problem and a more complex one, we will be using advanced algorithm like **XGBoost.** But the challenge here is that such models can adjust very well to the training data, thus leading to overfitting. We should be careful to prevent data redundancy since the same offer could be offered to a customer more than once. So, the same data point could appear in both sets, leading to an overestimate of final performance. Therefore, we will consider the pair offer-person when splitting. This process will help us to avoid overfitting.

V. Benchmark Model

Conversion rate is defined as the percentage of users who have completed an offer. We obtained a baseline conversion rate as 27.65%. Here our goal is to build a machine learning model that will predict the probability of a user taking up an offer or not. This will help us to increase the conversion rate.

Additionally, we also obtained the conversion rate for each offer:

- `0b1e1539f2cc45b7b9fa7c272da2e1d7: 14.87%`
- `2298d6c36e964ae4a3e7e9706d1fb8c2: 36.80%`
- `2906b810c7d4411798c6938adc9daaa5: 20.51%`
- `3f207df678b143eea3cee63160fa8bed: 27.53%`
- `4d5c57ea9a6940dd891ad53e9dbe8da0: 21.27%`
- `5a8bc65990b245e5a138643cd4eb9837: 44.36%`
- `9b98b8c7a33c4b65b9aebfe6a799e6d9: 19.67%`
- `ae264e3637204a6fb9bb56bc8210ddfd: 23.11%`
- `f19421c1d4aa40978ebb69ca19b0e20d: 27.12%`
- `fafdcd668e3743c1bb461111dcafc2a4: 41.44%`

VI. Evaluation Metrics

To evaluate the quality of approach and determine the model that produces best results, I will use classification metrics such as accuracy, recall, precision, and ROC AUC. The metrics will help to identify the success of a targeted offer campaign which is being sent to a selected group of customers. This will also help to analyze the conversion rate of the respective offers.

VII. Project Design

For executing this project, I will be following the below mentioned approach:

1. A Jupyter Notebook with AWS SageMaker will be used as the workspace.
2. After analyzing the three datasets, following are the findings:
    - **portfolio** dataset: We notice that there are no missing values. Since *channels* is a list of categories, we could expand it to have a one-hot encoded feature for each channel. Since the *offer_type* is having enumerated values, it is converted to categorical type to manage data inconsistencies that may occur
    - **profile** dataset: Here I am converting the *became_member_on* to a datetime field. Also, *gender* field being enumerated, it is converted to categorical type.
    - **transcript** dataset: Noticed that there are no missing values. In the available dataset, the *time* variable is expressed in hours, hence, to compare against duration of promotional offer, we can calculate this *time* variable in days.
3. After analyzing the numerical features, we came to know that there are not many outliers except for the *amount* and *time-in-days* fields.
4. Then the most critical categorical datatype features are analyzed and plotted.
5. As mentioned earlier, since the dataset is imbalanced, we will address this problem during model development by leveraging strategies as mentioned below. Used https://www.analyticsvidhya.com/blog/2020/07/10-techniques-to-deal-with-class-imbalance-in-machine-learning/ reference to study about these techniques.
    - Synthetic Minority Oversampling Technique (SMOTE) to create synthetic example of positive class.
    - Adding different weights according to class in loss function
6. Since the same offer could be offered to a customer more than once, data redundancy must be taken care which may cause an overestimate of the final performance. Therefore, we will consider the pair offer-person when splitting the dataset into train and test. This process will help us to avoid overfitting.
7. Once the final dataset is prepared, we leverage the AWS Sagemaker for designing and training XGBoost model. As a refinement technique, hyperparameters will be tuned according to the validation set. To obtain final performance metrics, testing will be done on previously unseen dataset like test set. Once the model artifacts are saved, we can deploy an endpoint. This deployed endpoint could be used for testing inference with JSON as input.
8. The project work and main findings will be encapsulated in a detailed project report.