Starbucks Project Proposal

Domain Background

- This program is about sending Starbucks Rewards promotional offers to selected customers. There are various offers to be sent to huge amount of customers. To save cost and achieve higher profit, it’s critical to send (correct) offers to appropriate audience.
- Each person in the simulation has some hidden traits that influence their purchasing patterns and are associated with their observable traits, and which is indicated from the collected data. Given the size / complexity of the dataset, data analysis and machine learning can help a lot.

Problem Statement

- The task is to identify which groups of people are most responsive to each type of offer and how to effectively present each type of offer. To turn this into a feasible project, I will predict whether a customer will respond to an offer based on input of customer and offer information.

Solution Statement

- Rather than segmenting customer and offer groups and analyzing each case individually, I opted to develop a XGBoost model, largely used for supervised learning problems, such as regression and classification. Specifically, I’ll create a XGBoost binary classifier that takes tabular data containing customer / offer features as input and outputs a 1 or 0 to indicate whether the promotional offer was successful or not. I will develop and deploy the model on AWS so that it can be used to predict future cases.

Datasets and Inputs

- This data set contains simulated data that mimics customer behavior on the Starbucks rewards mobile app, provided in the Capstone Workspace. There are 3 files:

    - portfolio.json - containing offer ids and meta data about each offer (duration, type, channel, etc.)
    - profile.json - demographic data for each customer (age, gender, income, etc.)
    - transcript.json - records for transactions, offers received, offers viewed, and offers completed, including the timestamp of purchase and the amount of money spent on a purchase. Note that it’s a time series dataset, and the order is import.
- The raw files are not ready for model training, and a lot of data processing work is needed. I’ll first do some EDA on three datasets separately, then I’ll transform transcript file to a tabular dataset, and each row will be one offer send to a customer, merging portfolio and profile columns, and add one column marking the customer is influenced by the offer or not.

Benchmark Model

- Since I'm working on a binary classifier, which is a type of machine learning model used to classify data into two categories, Logistic Regression is a great choice. This classic model is easy to implement and popular in the industry because it can produce reliable results with relatively little computational power.

Evaluation Metrics

- In order to assess the performance of my binary classification model, I’ll be utilizing the AUC score, also known as the area under the ROC curve. This metric takes into account the trade-off between the true positive rate and the false positive rate for different classification thresholds. By using the AUC score, I can gain a more comprehensive understanding of how well the model is able to discriminate between positive and negative instances. Furthermore, this metric is particularly useful when dealing with imbalanced datasets, as it is less sensitive to changes in the class distribution compared to other evaluation measures such as accuracy or precision.

Project Design

1. **Data Exploration:** Check the attributes of the provided data (distribution, missing value, anomalies) and decide on an approach for preprocessing it before modeling.

2. **Data Processing:** This will be the most time-consuming part, as I need to design an algorithm to filter successful offer, and convert transactions data to trainable dataset. A lot of other feature engineering tasks will be done, such as create new features, remove noise data (for example, customers complete offer without view / receive it).
3. **Model Implementation:** I'll first try with Logistic Regression as a baseline. Then I'll work with AWS sagemaker to train the XGBoost classifier (including hyperparameter tuning). I'll also analyze the training report to understand model performance better, by utilizing sagemaker Debugger. After that, I need to deploy it and invoke it to make predictions.
4. **Conclusion:** It'll be the summarization of my project solution, and some reflections on how I could improve further.