Starbucks Promotional Response Prediction Using Machine Learning

1. Domain Background

Digital loyalty programs have become a central component of modern marketing strategy, enabling companies to personalize promotions and measure customer engagement at scale. Starbucks' Rewards Program is one of the most successful implementations of such systems. As documented by Dholakia in the *Harvard Business Review*, the program's structure—combining mobile integration, personalized offers, and gamified incentives—has significantly strengthened customer loyalty and increased purchase frequency [1]. In these systems, users receive targeted promotions such as BOGO deals, discounts, or informational messages delivered through channels like the mobile app, email, web, or social media.

Understanding which customers will respond positively to which promotional offers is a key challenge in marketing analytics. Research shows that personalized marketing increases engagement and conversion, while poorly targeted offers can reduce customer satisfaction and erode profit margins. The dataset used in this project originates from the Udacity Starbucks Capstone Challenge, which provides a simulated but realistic environment for studying promotional effectiveness [2]. The simulation mirrors real consumer behavior, including customers who respond strongly to certain offers, customers who ignore or negatively respond to promotions, and customers who transact independently of any offer. It also includes complex scenarios such as offer completion without viewing, time-limited offer windows, and multi-step behavioral sequences (receive → view → transact → complete), which reflect real-world attribution challenges.

Academic research in marketing analytics consistently shows that customer responsiveness to promotional interventions varies widely across demographic groups, behavioral segments, and offer characteristics. Machine-learning-based personalization methods have been shown to improve targeting efficiency and reduce wasted marketing spend by identifying high-value responders and customers who prefer not to be contacted [3], [4]. This project aims to apply these principles to model Starbucks' promotional response patterns and enable more effective marketing decision-making.

References

[1] U. Dholakia, "How Starbucks' Rewards Program Drives Customer Loyalty," *Harvard Business Review*, Feb. 2016. Available: https://hbr.org/2016/02/how-starbuckss-rewards-program-drives-customer-loyalty

[2] Udacity, “Starbucks Capstone Challenge Dataset,” Data Scientist Nanodegree Program, Udacity, 2019. Available: https://www.udacity.com/

[3] S. A. Neslin, S. Gupta, W. Kamakura, J. Lu, and C. H. Mason, “Defection detection: Measuring and understanding the predictive accuracy of customer churn models,” *Journal of Marketing Research*, vol. 43, no. 2, pp. 204–211, 2006.

[4] J. Burez and D. Van den Poel, “Handling class imbalance in customer churn prediction,” *Expert Systems with Applications*, vol. 36, no. 3, pp. 4626–4636, 2009.

---

2. Problem Statement

Customers vary widely in how they respond to offers. Sending the wrong offer can waste marketing budget, reduce margins, or annoy customers—including those who prefer **not** to receive any promotions.

Goal:
Predict whether a customer will **respond positively** to a given promotional offer.

A “response” is defined as:
Offer viewed, and
Offer completed,
Within the offer’s validity period

This definition reflects true marketing influence rather than simple event co-occurrence.

The problem is:

- measurable (timestamped events),
- actionable (send / do not send offer),
- and realistic in its causal structure.

---

3. Datasets and Inputs

The project uses the three provided Starbucks JSON files:

1. portfolio.json — Offer Metadata

- `id`: offer ID
- `offer_type`: bogo / discount / informational
- `difficulty`: minimum spend required
- `reward`: incentive provided
- `duration`: validity window
- `channels`: list of delivery channels

2. profile.json — Customer Demographics

- `id`: customer ID
- `age`, `gender`, `income`
- `became_member_on`: membership start date

3. transcript.json — Event Log

Records include:

- `offer received`
- `offer viewed`
- `offer completed`
- `transaction`

Each entry contains:

- `person` = customer ID
- `time` = hours since test start

- `value` = offer_id, amount spent, or reward

This dataset enables:

- complete reconstruction of offer timelines
- causal labeling based on viewing + completing
- detection of customers who complete without being influenced
- identification of customers who prefer no offers

---

4. Solution Statement

The goal of this project is to develop a supervised machine learning system that predicts whether a customer will positively respond to a promotional Starbucks offer. The full solution will be implemented in AWS SageMaker, using an end-to-end workflow that includes data preprocessing, feature engineering, model training, evaluation, and inference.

The modeling pipeline consists of the following major components:

Preprocessing and Data Preparation

To ensure reproducibility and consistent data handling, all preprocessing steps are explicitly defined and executed within SageMaker JupyterLab. These steps include:

- Loading JSON data into structured Pandas DataFrames
- Flattening the transcript event logs and expanding the nested `value` field
- Constructing offer-level timelines (offer start → expiration)
- Defining the response label (viewed AND completed within offer window)
- Cleaning demographic fields (e.g., fixing missing ages, inconsistent gender labels)
- Handling missing values using median/mode imputation
- Encoding categorical variables (one-hot encoding for channels, offer type, gender)

- Creating engineered features such as membership tenure, difficulty, reward, duration, and behavioral patterns

Documenting these steps ensures that the model can be re-trained or audited without ambiguity, prevents accidental changes in data preparation, and avoids data leakage.

Model Development and Training

Training is performed in the SageMaker environment using:

- **Logistic Regression** (benchmark baseline)
- **Random Forest** (tree-based model for non-linear interactions)
- **AutoGluon Tabular** with bagging and stacking (the final selected model)

AutoGluon automatically trains multiple architectures (LightGBM, XGBoost, CatBoost, neural networks) and builds a **Weighted Ensemble** that achieved the best performance.

Model Selection and Evaluation

The final model is chosen using validation ROC-AUC, with additional metrics including precision, recall, and F1-score. This ensures the model not only predicts responders well but also minimizes false positives—an essential requirement in promotional targeting to avoid unnecessary marketing costs.

Deployment and Inference

The trained model is saved from SageMaker to S3 for long-term storage and can also be:

- Loaded locally for ad-hoc inference
- Deployed as a SageMaker Endpoint (if real-time inference is desired)
- Used with SageMaker Batch Transform (if batch scoring becomes necessary later)

In this project, inference is demonstrated inside the SageMaker Notebook by loading the saved AutoGluon predictor and scoring customer–offer pairs from the test dataset.

---

5. Benchmark Model

The benchmark is **Logistic Regression**, chosen because:

- It is widely used in traditional marketing response modeling
- It provides interpretable coefficients
- It establishes a performance baseline

Observed benchmark performance:
Accuracy: ~0.75
ROC-AUC: ~0.839

This serves as the threshold that more sophisticated models must beat.

---

6. Evaluation Metrics

Primary Metric — ROC-AUC

- Measures ranking quality
- Works well for imbalanced classes
- Ideal for marketing response prediction

Secondary Metrics — Precision, Recall, F1

These help evaluate:

- How many true responders the model captures
- How many false responders the model incorrectly predicts
- Trade-offs between marketing reach and wasted promotions

---

7. Project Design

Step 1 — EDA & Data Cleaning

- Examine demographic distributions
- Expand transcript `"value"` field
- Investigate event frequencies
- Handle missing ages and genders
- Identify unrealistic ages (e.g., age 118 → missing)

Step 2 — Offer Labeling Logic

For each offer instance:

- Determine offer window based on `duration * 24 hours`
- Check if offer was **viewed** within window
- Check if offer was **completed** within window
- Determine:
  - positive responders
  - negative responders (viewed but not completed)
  - uninfluenced completers (completed without viewing)
  - offer-irrelevant purchasers

Step 3 — Feature Engineering

- One-hot encode channels
- Encode offer types

- Membership tenure
- Transaction behavior
- Difficulty, reward, and duration

Step 4 — Modeling

- Train/Test/Validation split
- Logistic Regression
- Random Forest
- AutoGluon with bagging & stacking
- Select the best model (AutoGluon WeightedEnsemble_L3)

Step 5 — Deployment Artifacts

- Save the trained AutoGluon predictor
- Optionally upload to S3
- Demonstrate inference using:
    - local notebook prediction
    - model reload from disk

Step 6 — Insight Extraction

Interpret model outputs to identify:

- which demographic groups respond most
- which offers work best for which customers
- segments for which *no offer is optimal*