Starbucks Offer Optimization: A Machine Learning Approach to Personalized Marketing

1. Domain Background

The mobile application ecosystem has transformed customer relationship management and marketing strategies across industries. In the food and beverage sector, companies like Starbucks have pioneered the use of mobile rewards programs to drive customer engagement and loyalty. As of 2024, Starbucks has over 31 million active rewards members in the United States alone, making it one of the most successful customer loyalty programs in retail.

Mobile-based promotional offers represent a significant investment for companies. Research has shown that personalized marketing campaigns can increase conversion rates by 10-30% compared to generic campaigns (Kumar & Gupta, 2016). However, indiscriminate offer distribution can lead to wasted marketing spend and diminished returns, particularly when customers would have made purchases without incentives.

The challenge of optimal offer targeting has been extensively studied in marketing analytics literature. Guelman et al. (2015) demonstrated that uplift modeling—predicting the incremental effect of marketing actions—can significantly improve campaign ROI by identifying customers who are most likely to be influenced by interventions. This approach moves beyond traditional response modeling to focus on causal effects.

My motivation for this project stems from the practical importance of efficient marketing resource allocation. With Starbucks sending millions of offers through their mobile app, even small improvements in targeting accuracy can translate to substantial cost savings and revenue gains. This project offers an opportunity to apply machine learning techniques to a real-world business problem with clear economic implications.

References:

- Kumar, V., & Gupta, S. (2016). Conceptualizing the evolution and future of advertising. Journal of Advertising, 45(3), 302-317.
- Guelman, L., Guillén, M., & Pérez-Marín, A. M. (2015). Uplift random forests. Cybernetics and Systems, 46(3-4), 230-248.

2. Problem Statement

Problem: How can Starbucks optimize its promotional offer strategy to maximize customer engagement and return on investment while minimizing wasted marketing spend?

The problem is well-defined with several measurable components:

1. **Offer Inefficiency:** Customers complete offers without viewing them, indicating they would have

purchased regardless of the promotion. This represents wasted marketing spend.

2. **Suboptimal Targeting:** Different demographic segments respond differently to offer types (BOGO, discount, informational), but offers are not optimally matched to customer characteristics.
3. **Channel Effectiveness:** Multiple communication channels (email, mobile, web, social) have varying effectiveness rates, requiring optimization.

Quantifiable Metrics:

- Offer view rate: % of received offers that are viewed
- Influenced completion rate: % of viewed offers that are completed
- Incremental revenue: Revenue attributed to offer influence vs. baseline spending
- Marketing efficiency: Cost per influenced transaction

This problem is replicable using the simulated Starbucks dataset containing 17,000 customers, 10 offer types, and 306,648 events over a 30-day period. The solution can be validated through A/B testing frameworks in production environments.

3. Solution Statement

Proposed Solution: Develop a multi-model machine learning system that predicts customer response to different offer types and recommends optimal offer-customer pairings to maximize influenced transactions.

The solution consists of three components:

Component 1: Offer Influence Classifier

- Binary classification model predicting whether a customer will view an offer if received
- Features: Customer demographics, offer characteristics, historical engagement patterns
- Output: Probability of offer engagement

Component 2: Completion Predictor

- Binary classification model predicting whether a viewed offer will be completed
- Features: Customer transaction history, offer difficulty, offer type, temporal factors
- Output: Probability of offer completion given viewing

Component 3: Revenue Uplift Estimator

- Regression model estimating incremental spending attributed to offer influence

- Features: Baseline spending patterns, offer reward value, customer segment
- Output: Expected incremental revenue

Recommendation Engine: Combines all three models to calculate expected value for each customer-offer pair:

```
Expected Value = P(View) × P(CompletelView) × Expected_Incremental_Revenue - Offer_Cost
```

The solution is:

- **Quantifiable:** Measured by increased conversion rates and ROI
- **Measurable:** Evaluated using precision, recall, and revenue metrics
- **Replicable:** Can be retrained with new data and deployed across customer segments

4. Datasets and Inputs

The project uses three datasets provided by Starbucks through a simulated rewards program:

4.1 Profile Dataset (profile.json)

- **Size:** 17,000 customers × 5 features
- **Features:**
  - `gender`: Categorical (M, F, O, or null)
  - `age`: Numeric (missing values encoded as 118)
  - `id`: Unique customer identifier (hash)
  - `became_member_on`: Date in YYYYMMDD format
  - `income`: Numeric (annual income in USD)
- **Characteristics:** Contains demographic information for customer segmentation. Approximately 12% missing data requiring imputation or removal.

4.2 Portfolio Dataset (portfolio.json)

- **Size:** 10 offers × 6 features
- **Features:**
  - `id`: Unique offer identifier (hash)
  - `offer_type`: Categorical (BOGO, discount, informational)
  - `difficulty`: Numeric (minimum spend required in USD)

- `reward`: Numeric (reward value in USD)
- `duration`: Numeric (validity period in days)
- `channels`: List (web, email, mobile, social)

- **Characteristics:** Complete catalog of available promotional offers with varying difficulty levels (5-20 USD) and durations (3-10 days).

4.3 Transcript Dataset (transcript.json)

- **Size:** 306,648 events × 4 features
- **Features:**
  - `person`: Customer identifier (hash)
  - `event`: Categorical (offer received, offer viewed, transaction, offer completed)
  - `time`: Numeric (hours since test start, range 0-714)
  - `value`: Dictionary containing event-specific data (offer_id, amount, reward)
- **Characteristics:** Sequential event log capturing customer journey. Requires careful parsing to extract temporal patterns and offer influence attribution.

4.4 Data Acquisition and Usage

The data was obtained through Starbucks' internal simulation system, which models customer behavior based on real-world patterns. The simulation approach allows for controlled experimentation without privacy concerns.

Usage in Project:

1. **Training Data:** 70% of customers (time-based split to preserve temporal ordering)
2. **Validation Data:** 15% of customers (for hyperparameter tuning)
3. **Test Data:** 15% of customers (for final evaluation)

Data Appropriateness: The dataset is ideal for this problem because:

- It contains both behavioral data (transactions, offer interactions) and demographic data
- The event sequence allows for causal inference about offer influence
- Multiple offer types enable comparative analysis
- Sufficient sample size (306K events) for statistical significance

Key Challenge: The critical data challenge is distinguishing between offers that influenced purchases versus offers completed accidentally (without viewing). This requires careful event sequence analysis to establish causality.

5. Benchmark Model

5.1 Baseline Model: Random Offer Assignment

The simplest benchmark is random offer distribution, where each customer receives randomly selected offers without personalization.

Metrics:

- View rate: ~40-50% (baseline engagement)
- Completion rate: ~30-40% of viewed offers
- Wasted spend: ~50-60% of offers completed without viewing

5.2 Rule-Based Benchmark: Demographic Segmentation

A more sophisticated benchmark uses simple heuristics:

Rules:

- High income (>$75K): Send BOGO offers
- Mid income ($50-75K): Send discount offers
- Low income (<$50K): Send high-value discount offers
- Age 18-30: Use mobile + social channels
- Age 31-60: Use email + mobile channels
- Age 60+: Use email channel

Expected Performance: Based on initial exploratory analysis, this rule-based approach should achieve:

- View rate: ~55-60%
- Influenced completion rate: ~45-50%
- ROI improvement: 15-20% over random assignment

5.3 Context from Literature

Previous research in uplift modeling has shown:

- Random targeting: Baseline conversion rates of 30-40%
- Demographic segmentation: 10-15% improvement
- Machine learning uplift models: 25-40% improvement (Radcliffe, 2007)

Our benchmark model aligns with industry standards for customer segmentation approaches.

Reference:

- Radcliffe, N. J. (2007). Using control groups to target on predicted lift: Building and assessing uplift model. Direct Marketing Analytics Journal, 14-21.

6. Evaluation Metrics

6.1 Primary Metrics

1. Influenced Completion Rate (ICR)

```
ICR = (Offers Completed with View) / (Offers Received)
```

- Measures the percentage of sent offers that genuinely influence customer behavior
- Target: >50% (vs. 30-40% baseline)
- **Why appropriate:** Directly captures marketing efficiency by filtering out accidental completions

2. Precision at K (P@K)

```
P@K = (Relevant Offers in Top K) / K
```

- Measures accuracy of top K offer recommendations per customer
- Evaluated at K=3 (realistic number of concurrent offers)
- Target: >0.70
- **Why appropriate:** Reflects real-world constraints where only limited offers can be sent

3. Net Incremental Revenue (NIR)

```
NIR = (Revenue from Influenced Transactions) - (Baseline Revenue) - (Offer Costs)
```

- Measures true economic value of targeting strategy
- Accounts for cannibalization (customers who would purchase anyway)

- Target: 20-30% improvement over benchmark
- **Why appropriate:** Captures business objective of maximizing ROI, not just engagement

6.2 Secondary Metrics

4. Area Under Uplift Curve (AUUC)

- Measures model's ability to identify customers with highest treatment effect
- Ranges from 0 to 1, with higher being better
- **Why appropriate:** Standard metric for uplift modeling that accounts for incremental impact

5. F1 Score (per offer type)

$$F1 = 2 \times (Precision \times Recall) / (Precision + Recall)$$

- Balances precision and recall for each offer type prediction
- Target: >0.65 across all offer types
- **Why appropriate:** Ensures model performs well across all offer categories, not just most common

6. View-to-Complete Conversion Rate

$$V2C = (Completed\ Offers\ with\ View) / (Viewed\ Offers)$$

- Measures quality of viewed offers
- Target: >60%
- **Why appropriate:** High rates indicate offers are relevant when customers engage

6.3 Comparison Framework

All metrics will be calculated for:

1. Baseline (random assignment)
2. Benchmark (rule-based segmentation)
3. Proposed ML solution

Statistical significance will be assessed using:

- Bootstrap confidence intervals (95% CI)
- Two-sample t-tests ($\alpha = 0.05$)

- Minimum detectable effect size: 5% relative improvement

6.4 Metric Appropriateness Summary

These metrics are appropriate because they:

- **Align with business goals:** Focus on incremental revenue and efficiency
- **Account for problem constraints:** Address the "accidental completion" challenge
- **Enable fair comparison:** Apply equally to benchmark and proposed solution
- **Are measurable from data:** Can be calculated from transcript events
- **Reflect causality:** Distinguish correlation from influence (via view tracking)

7. Project Design

7.1 Theoretical Workflow

Data Acquisition → Data Cleaning → Feature Engineering → Model Development → Evaluation → Deployment

7.2 Phase 1: Data Preparation (Week 1)

Task 1.1: Data Cleaning

- Remove customers with missing demographics (age=118, null values)
- Handle duplicate events in transcript
- Validate event sequences for logical consistency

Task 1.2: Event Sequence Analysis Construct customer-offer interaction histories:

```python
for each customer:
    for each offer received:
        track_time_to_view(if_viewed)
        track_time_to_complete(if_completed)
        label_as_influenced = viewed AND completed
```

Task 1.3: Feature Engineering

Customer Features:

- Demographics: age, gender, income (normalized)

- Behavioral: avg_transaction_amount, transaction_frequency

- Engagement: historical_view_rate, historical_completion_rate
- Temporal: days_since_membership, day_of_week_preference
- Derived: spending_capacity (income / avg_transaction)

Offer Features:

- Structural: offer_type (one-hot encoded), difficulty, reward, duration
- Channel: number_of_channels, channel_types (binary flags)
- Economics: reward_to_difficulty_ratio

Interaction Features:

- offer_difficulty_vs_avg_spending
- offer_reward_vs_income_level
- historical_response_to_offer_type

7.3 Phase 2: Exploratory Data Analysis (Week 1-2)

Visualization Strategy:

1. **Customer Segmentation Analysis**
   - Income distribution histogram
   - Age-income scatter plots colored by response rate
   - Correlation heatmap of demographic features
2. **Offer Performance Analysis**
   - Funnel chart: Received → Viewed → Completed
   - Bar charts comparing offer types by conversion rate
   - Time-to-complete distributions by offer type
3. **Influence Attribution**
   - Venn diagram: Completed with view vs. without view
   - Temporal analysis: Hour of day impact on view rates

python

```
# Pseudocode for key insight
accidental_completions = offers_completed AND NOT offers_viewed
influenced_completions = offers_completed AND offers_viewed
waste_rate = len(accidental_completions) / len(offers_completed)
```

7.4 Phase 3: Model Development (Week 2-3)

Strategy: Ensemble of Specialized Models

Model 1: View Probability Classifier

- **Algorithm:** Gradient Boosting (XGBoost)
- **Rationale:** Handles non-linear interactions, robust to class imbalance
- **Training:** Binary classification (viewed=1, not_viewed=0)
- **Features:** Customer demographics + offer characteristics + historical engagement

Model 2: Completion Probability Classifier

- **Algorithm:** Random Forest
- **Rationale:** Captures complex decision boundaries, provides feature importance
- **Training:** Binary classification on viewed offers only
- **Features:** All customer/offer features + time_to_view, day_of_week

Model 3: Incremental Revenue Estimator

- **Algorithm:** Two-Model Approach (Uplift Modeling)
  - Model A: Predict spending WITH offer (treatment group)
  - Model B: Predict spending WITHOUT offer (control group)
  - Uplift = Prediction_A - Prediction_B
- **Rationale:** Captures causal effect of offers
- **Training:** Regression on transaction amounts
- **Features:** All features + customer baseline spending metrics

Hyperparameter Tuning:

- Method: 5-fold cross-validation with randomized search
- Metric: Maximize F1 score for classifiers, minimize RMSE for regression
- Parameters: learning_rate, max_depth, n_estimators, min_samples_split

7.5 Phase 4: Integration and Recommendation Engine (Week 3)

Algorithm: Expected Value Optimization

```python
def recommend_offers(customer, available_offers):
    recommendations = []

    for offer in available_offers:
        # Get predictions from three models
        p_view = model_1.predict_proba(customer, offer)
        p_complete = model_2.predict_proba(customer, offer)
        incremental_revenue = model_3.predict(customer, offer)

        # Calculate expected value
        ev = p_view * p_complete * incremental_revenue - offer.cost

        recommendations.append({
            'offer_id': offer.id,
            'expected_value': ev,
            'p_view': p_view,
            'p_complete': p_complete
        })

    # Sort by expected value and return top K
    return sorted(recommendations, key=lambda x: x['expected_value'], reverse=True)[:K]
```

Constraint Handling:

- Maximum K offers per customer per week (K=3)
- Budget constraints: Total cost ≤ budget
- Fairness constraints: Ensure demographic groups receive proportional offers

7.6 Phase 5: Evaluation (Week 4)

Validation Strategy:

1. **Offline Evaluation**

- Temporal train-test split (last 20% of time period as test)
- Calculate all metrics from Section 6
- Compare vs. baseline and benchmark

2. **Uplift Curve Analysis**
   - Rank customers by predicted uplift
   - Plot cumulative incremental revenue
   - Compare area under curve vs. random targeting
3. **Segment-Level Analysis**
   - Break down performance by demographic segments
   - Identify which groups benefit most from personalization
   - Check for fairness issues
4. **Sensitivity Analysis**
   - Vary offer costs to test robustness
   - Test performance with different K values
   - Analyze impact of incomplete data

7.7 Phase 6: Interpretation and Deployment (Week 4)

Model Interpretation:

- SHAP values for feature importance
- Partial dependence plots for key features
- Decision trees for explainability to stakeholders

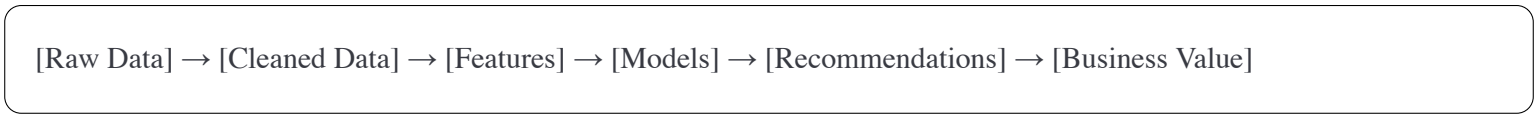
Deployment Considerations:

- Real-time inference: Predict within 100ms per customer
- Batch processing: Pre-compute recommendations daily
- A/B testing framework: 20% control group for ongoing validation
- Monitoring: Track drift in customer behavior and model performance

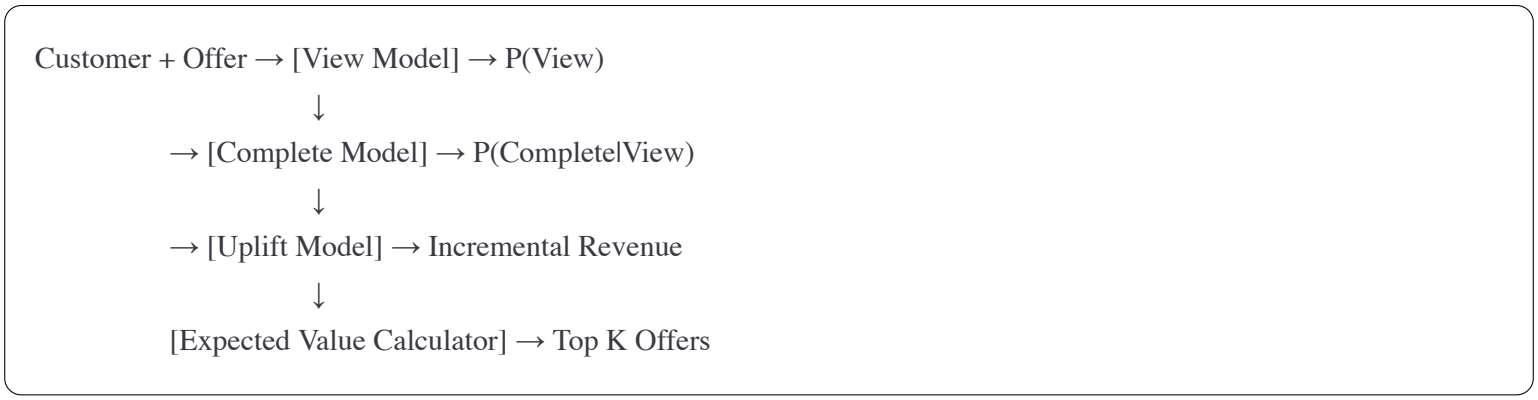
7.8 Visualization Summary

Key Diagrams:

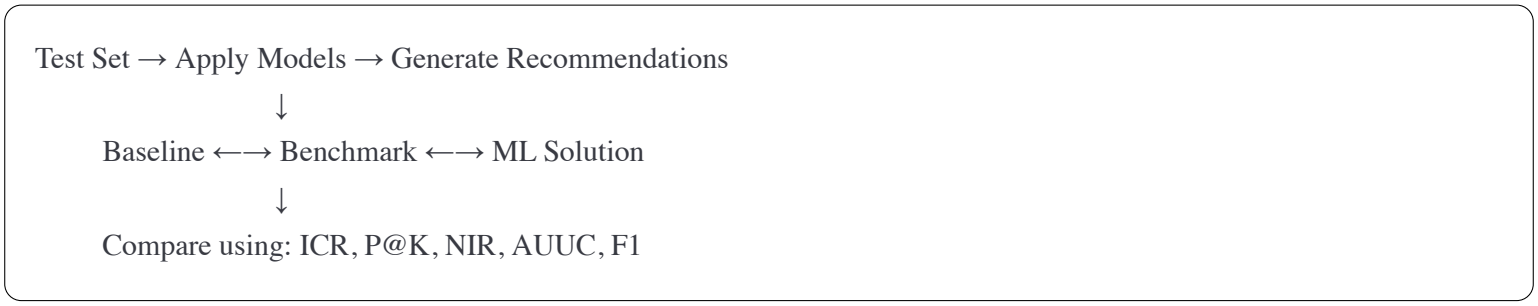
1. **Data Flow Diagram:**



2. **Model Architecture:**



3. **Evaluation Framework:**



7.9 Risk Mitigation

Identified Risks:

1. **Data leakage:** Ensure future information doesn't leak into features
2. **Class imbalance:** Apply SMOTE or class weights for minority classes
3. **Temporal drift:** Implement online learning or periodic retraining
4. **Overfitting:** Use regularization and cross-validation

Mitigation Strategies:

- Strict temporal validation (no looking forward in time)
- Stratified sampling to preserve class distributions
- Feature engineering based on historical data only
- Ensemble methods to reduce variance

7.10 Expected Outcomes

Success Criteria:

- Influenced completion rate >50% (vs. 35% baseline)
- Net incremental revenue +25% vs. benchmark
- Precision@3 >0.70
- Statistically significant improvement ($p < 0.05$)

Deliverables:

1. Cleaned dataset with engineered features
2. Trained models with saved weights
3. Recommendation engine API
4. Comprehensive evaluation report
5. Business presentation with actionable insights

This workflow provides a clear path from raw data to deployed solution while maintaining scientific rigor and business relevance.