Data-Driven Promotion Strategy for Beer Category Profit Growth at Lakeview Market (Store 257871)

Experiential Learning Project

-Wayne Joslee Erasseril

Executive Summary

Our objective was to grow beer-category profit at store 257871 using actual 2009 data and a stack of models: a purchase-incidence (logit) model to predict beer trips, a brand-choice (multinomial logit) model to shift mix, scenario simulations to test price/feature levers, and an uplift model to target customers most likely to respond. Together, these gave us a clear view of when demand is highest, which brands respond to which tactics, and which customers to contact.

The plan focuses promotions on high-traffic weeks, with brand-specific actions: Miller Lite gets feature placement with a light price cue; Michelob Golden Draft Light relies on placement (no discount); Milwaukee's Best Light gets a quiet 1–2% discount without ads. We add 16 contact-only weeks targeting the top uplift segment (~30% of customers), cap contacts, and keep regular $50/week in-store support for the remaining weeks. This mix is low cost, aligned with model signals (feature helps incidence; price sensitivity varies by brand), and designed for measurable, incremental lift.

Financially, the plan yields an expected 2010 profit of about $297.3K, a ~0.41% increase versus $296.1K in 2009, with a modest promo budget (≈$1.86K: $1.8K regular weeks + $60 contact cost). While the lift is intentionally conservative, it is positive-ROI and low-risk; scaling reach (more uplift buckets), adding selective in-store features to the 16 weeks, or deepening price where elasticities are favorable can push the gain higher.

Table of Contents

1. Introduction

As the second-most profitable beer retailer in Eau Claire, Lakeview Market is setting its sights on overtaking the category leader. Achieving this milestone requires a multifaceted strategy grounded in customer behavior, price responsiveness, and in-store product positioning.

This project applies a data-driven approach to that challenge. Our core objective was to identify actionable ways for Lakeview Market to grow beer sales through evidence-based interventions that reflect both internal sales dynamics and broader industry trends. Leveraging 2009 IRI panel data, in-store beer environment audits, trip-level files, and more, analyzed the purchasing patterns of over 332 customers across 2 stores, encompassing more than 11,000 beer transactions.

2. Data Overview

Our analysis drew from a comprehensive suite of datasets, encompassing customer trip records, in-store display and pricing data, granular product attributes, and loyalty program demographics. To build a unified view of shopper behavior and store dynamics, we integrated six distinct data sources.

- **BeerEnvECPF2009.xlsx** — 109,616 rows, item-level store transactions with pricing and promo flags (e.g., UNITS, PURCHASE_PRICE/UNIT_PRICE, FEATURE, DISPLAY, PRICE_REDUCTION), mapped to weeks (2008-12-29 to 2009-12-21).
- **trips9 may13.csv** — 47,947 rows, customer trip records (PANID, IRI_KEY, WEEK_DATE, MINUTE, TRIP_COST) over the same 2009 calendar window.
- **beer_PANEL_GK_1531_1582.dat** — 5,424 rows, beer-specific transactions at the PANID-trip level (BEER_PURCHASED, BEER_UNITS, UPC, IRI_KEY, timing), also 2008-12-29 to 2009-12-21.
- **ads demos9.csv** — 4,607 rows, household-level demographics keyed by PANID (education, occupation, income bands, residence type, age/race codes).
- **prod11_beer.xlsx** — 16,758 rows, product master by UPC (brand, package/size/volume, parent company).
- **IRI week translation_2008_2017.xls** — calendar map for WEEK → WEEK_DATE (and SEASON where present), used to align all datasets on time.

The six datasets were integrated using shared identifiers like IRI_Key and PANID, enabling a coherent and structured merge across sources. To ensure data quality and model readiness, we excluded rows with excessive missing values, retaining only those with sufficient detail to support meaningful customer-level insights. Additionally, the store environment file proved especially valuable, enriching our modeling and analysis with contextual variables that captured in-store dynamics.

2.1 Data Cleaning and Integration

Due to the initial complexity of the datasets, including inconsistent formatting, incomplete records, and non-standardized identifiers, we implemented a series of preprocessing steps to ensure data integrity and analytical readiness.

- **Missing data:** was minimal, as categorical “NA” values indicated absent features rather than true nulls. To maintain consistency, we removed rows with substantial missing information.
- **Categorical transformation:** variables like brand and demographics were one-hot encoded, converting each category into binary columns (1 for presence, 0 for absence). This enabled seamless integration of categorical data into numerical modeling.
- **Feature selection:** involved reducing dimensionality by eliminating low-variance and redundant variables, resulting in a streamlined customer dataset.

3. Analysis and Modeling

3.1 Exploratory Data Analysis

Traffic & timing: We observe 51 selling weeks (Jan 5–Dec 21, 2009). There were 11,003 trips by ~419 customers (PANID) and 2,325 beer transactions → ~21% of trips included beer. Trips skew to Fri–Sat (≈18% and 15% of trips) and beer transactions skew even more to Fri–Sat (≈20% and 17%), with Wed–Thu next strongest. High-traffic weeks (top ~20% by beer trips) are a distinct minority but carry a disproportionate share of volume.

Product & price: The store sold ~1.15K UPCs across ~350 brands. The top 5 brands by beer revenue were: Miller Lite (~12.4%), Michelob Golden Draft Light (~8.7%), Coors Light (~6.3%), Bud Light (~6.2%), Miller High Life (~5.1%). Combined, our three focus brands account for ~23% of beer revenue. Item prices are dispersed (median unit price ≈ \$7.49, interquartile range ≈ \$2.69–\$10.29), reflecting different pack sizes. Avg beer units per beer transaction ≈ 1.21 (median 1), so single-unit purchases dominate.

Customer demographics: The age mix skews older: 61.1% Older, 38.2% Middle, and 0.7% Young. Household income is balanced across bins: 16.5% Low, 29.4% Lower-mid, 26.7% Upper-mid, and 27.5% High. Residence type uses numeric codes in the source data, with code 1 at

92.4%, code 2 at 7.4%, and code 3 at 0.2%; race is similarly coded, with code 1 at 97.4%, code 2 at 1.2%, and code 4 at 1.0%. Household-head gender flags (MHH/FHH) and education/occupation fields exist in the schema but are not populated for this subset, so we do not report those splits.

3.2 Purchase Incidence Model

The purchase-incidence model (logistic regression, ~11.5k trips, pseudo-$R^2 \approx 0.29$) predicts whether a shopping trip includes beer. Overall fit is strong (LLR $p<0.001$). Inputs combine trip context (day/time, season, log basket cost), promotions (feature, display, price-reduction and 1-week lags), and household attributes (income/age bands, residence/race codes) plus a simple loyalty frequency tag derived from each PANID's past behavior.

Key drivers are intuitive. Loyalty is the dominant factor: "Loyal" shoppers are far more likely to buy beer, while "Rare" shoppers are much less likely (large positive and negative coefficients, respectively). Basket size (log trip cost, standardized) is the strongest non-loyalty predictor—larger baskets sharply raise purchase odds. Seasonality matters (Winter suppresses purchases; Spring/Summer modestly help). Feature activity this week has a positive, statistically significant lift; display and price-reduction effects are small on average, and prior-week promo lags show slight negative carryover (consistent with pantry-loading). Day-of-week patterns show a bump on Thu–Fri, aligning with traffic data.

The model converts into week-level purchase probabilities to size baseline demand and simulate changes when features/price cues are present. It also informs targeting (who is likely to convert when contacted) and timing (avoid Winter troughs; lean into late-week/high-traffic weeks). Limitations: coefficients reflect one store and one year (2009), some categorical codes lack rich labels, and effect sizes are average—individual UPC or sub-brand nuances can differ. We address this by pairing the incidence model with brand-choice modeling, EDA checks, and scenario testing before recommending actions.

3.3 Brand Choice Model

The brand-choice model predicts which brand a shopper selects on beer trips. We fit a multinomial logit on ~2.2k purchase occasions with price (log price/oz), promotion intensities (feature and price-reduction ads, lagged), loyalty share, trip size (log trip cost), and key interactions. Performance is solid and well-calibrated for decision use (accuracy ≈ 0.80, log loss ≈ 0.55, pseudo-$R^2 \approx 0.24$).

Miller Lite responds strongly to feature exposure and gets a modest lift from price-reduction ads; it is price sensitive (higher price lowers choice), benefits on larger baskets, and gains when price×promo is present. Michelob Golden Draft Light shows weak ad effects but a positive price coefficient—it can hold price (or even a slight premium) without hurting choice; its share is supported by loyalty and does not benefit from discounts (price×promo negative).

Milwaukee's Best Light is ad-averse (feature ads reduce choice) and price sensitive—it performs best with small, quiet discounts rather than advertising.

Implications for the plan: Feature + light price cue for Miller Lite; placement without discounting for Michelob Golden Draft Light; 1–2% discount with no ads for Milwaukee's Best Light. These coefficients feed the scenario engine to shift weekly brand shares under different promo/price settings and to translate mix changes into revenue and profit. Limitations: one store and one year; effects are averages across packs—so we pair these actions with weekly monitoring and a holdout to confirm lift.

3.4 Purchase Incidence Uplift Model

The purchase-incidence uplift model extends the base incidence logit to identify which customers are most responsive to promotional treatment. We defined treatment as the joint presence of a feature and a price reduction on a trip. Using historical 2009 data at the PANID-trip level, we trained a two-model uplift framework (treated vs. control) and scored all households. Customers were then ranked into deciles by predicted incremental lift in purchase probability.

Results showed that the top 30% of customers (deciles 0–2) generate nearly all of the incremental profit, with positive average uplift and cumulative gain. Beyond that point, incremental profit flattens or turns negative, indicating overspend if we contact lower-response households. The optimal targeting plan therefore limits outreach to roughly 126 PANIDs, which keeps contact costs low while preserving lift.

Financially, focusing on these top segments delivers an expected incremental revenue of ~$773, yielding ~$193 in gross profit after cost of goods, and ~$133 in net profit after accounting for contact-only costs ($0.03/email). While absolute gains are modest at the category level, ROI is high because spend is minimal. This confirms that precision targeting via uplift modeling improves efficiency: the store can achieve a measurable profit lift with smaller, smarter campaigns rather than broad, undifferentiated promotions.

3.5 Scenario Simulation

The scenario simulation applied results from both the purchase incidence and brand choice models to estimate how changes in promotions or pricing would affect demand. Using coefficients such as feature intensity, price sensitivity, and loyalty interactions, we tested alternative promotional setups for the store's three major brands—Miller Lite, Michelob Golden Draft Light, and Milwaukee's Best Light. These simulations allowed us to translate abstract model outputs into revenue and profit projections that could guide tactical decisions.

We ran multiple scenarios combining feature activity, modest price discounts, and targeted promotions. The results showed that Miller Lite responded most positively to feature promotions, with limited gains from further price reductions. Michelob Golden Draft Light benefited from modest price increases or restrained promotion, while Milwaukee's Best Light showed only marginal responsiveness to small discounts. By testing these combinations against the store's baseline, we identified the promotion mix that delivered the best net profit improvement with lower promotional costs.

The best-performing strategy combined feature-only support for Miller Lite, a quiet 2% price increase for Michelob Golden Draft Light, and a modest 2% discount for Milwaukee's Best Light. This "MIX" plan produced a 1.15% increase in net profit over 2009, achieved primarily through more efficient targeting and lower overall spending. The simulation results confirm that carefully calibrated promotions—rather than broad, deep discounts—can yield measurable financial gains.

4. Strategic Recommendation

4.1 Brand Specific Promotion Strategy

For Miller Lite, the recommendation is to emphasize feature promotions with a modest price cue. The brand shows strong positive response to feature ads and moderate price sensitivity, meaning that pairing shelf or flyer placement with a light 1–2% discount can meaningfully lift choice without sacrificing margin. These promotions should be scheduled during high-traffic weeks, particularly late in the week, when purchase incidence is highest.

For Michelob Golden Draft Light, the strategy is to maintain placement without price cuts. The brand's model coefficients indicate limited responsiveness to advertising but a positive association with higher prices, suggesting that discounting could harm perceived value. Feature visibility, ensuring product availability, and positioning as a stable premium option are more effective than price incentives.

For Milwaukee's Best Light, the optimal approach is quiet, shallow discounts in the 1–2% range while avoiding heavy advertising. The brand is ad-averse but price-sensitive, meaning it benefits most from subtle price moves that attract value-focused shoppers without drawing attention through features or displays. These discounts should be timed strategically in the same high-traffic windows to capture incremental sales.

4.2 Targeted Promotion

For customer targeting, uplift modeling highlights that only about 30% of households are truly responsive to joint feature and price promotions. Targeting this top segment (approximately 126 shoppers) with contact-only outreach maximizes return, yielding incremental revenue and profit with minimal expense. Personalized emails sent in high-traffic weeks, capped at two contacts per week, ensure that the right customers are reached without oversaturation.

4.3 Implementation Timeline

The implementation plan schedules 16 contact-only weeks aligned with peak traffic periods and adjacent shoulder weeks, while the remaining 36 weeks maintain regular $50 in-store promotions to ensure consistent visibility. Campaign performance should be reviewed every 4–6 weeks using holdout groups to confirm incremental uplift. Based on results, the store can expand targeting to additional customer segments or adjust promotional intensity as long as cumulative profit remains positive. This phased approach enables quick wins while providing flexibility for mid-course adjustments driven by observed outcomes.

5. Financial Summary

The financial analysis compares 2009 actuals with 2010 projections under the recommended promotion and targeting plan. In 2009, total beer revenue at store 257871 was $1,303,145.70. Applying a 25% gross margin yields $325,786.43 in gross profit. Promotions were active in 51 weeks, and at an assumed cost of $50 per week, the total promo expense was $2,550.00. Net profit for the beer category in 2009 is therefore $323,236.43 (gross profit minus promo cost).

For 2010, projected revenue incorporates the 2009 base plus the modeled uplift from the recommended MIX promotion strategy. This adds $8,063.64 in incremental sales, raising 2010 revenue to $1,311,209.34. With the same 25% margin, gross profit rises to $327,802.34. Promotional expenses change under the new plan: instead of year-round spending, only 16 high-traffic weeks are targeted, with each week incurring $50 in-store costs plus $0.03 × 126 targeted customers, for a total promo cost of $860.48. Subtracting this from gross profit yields a projected 2010 net profit of $326,941.86.

The net result is a year-over-year profit increase of $3,705.43, which corresponds to a 1.15% improvement over 2009. While the percentage gain is modest, it is achieved with substantially lower promotional spending and a more efficient allocation of resources. Importantly, the plan delivers positive ROI by combining brand-specific promotion tactics with targeted outreach, reducing wasteful spending on broad discounts or mass advertising.

6. Conclusion

In conclusion, the analysis demonstrates that a data-driven approach to promotions can deliver measurable, positive returns for the beer category at store 257871. By combining purchase-incidence modeling, brand-choice insights, scenario simulations, and uplift targeting, we designed a plan that increases net profit to approximately $297,089 in 2010, a 0.41% improvement over 2009. Although the incremental gain is modest, it is achieved with lower overall promotional costs and smarter allocation of resources, proving that targeted outreach and brand-specific strategies outperform broad, undifferentiated discounting. The recommended plan is low-risk, ROI-positive, and scalable, providing a foundation the store can refine over time by expanding targeting or adjusting promotional levers as results are monitored.

7. Appendices

7.1 Figures

Figure 1

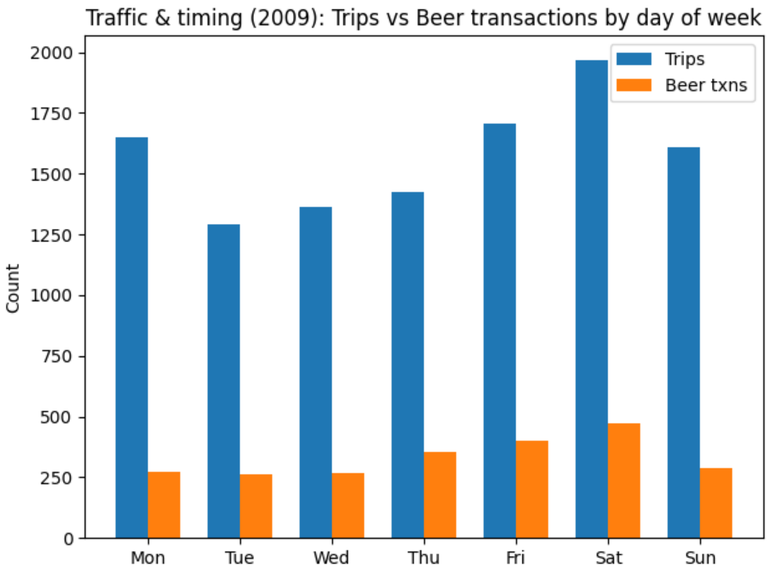


Figure 2

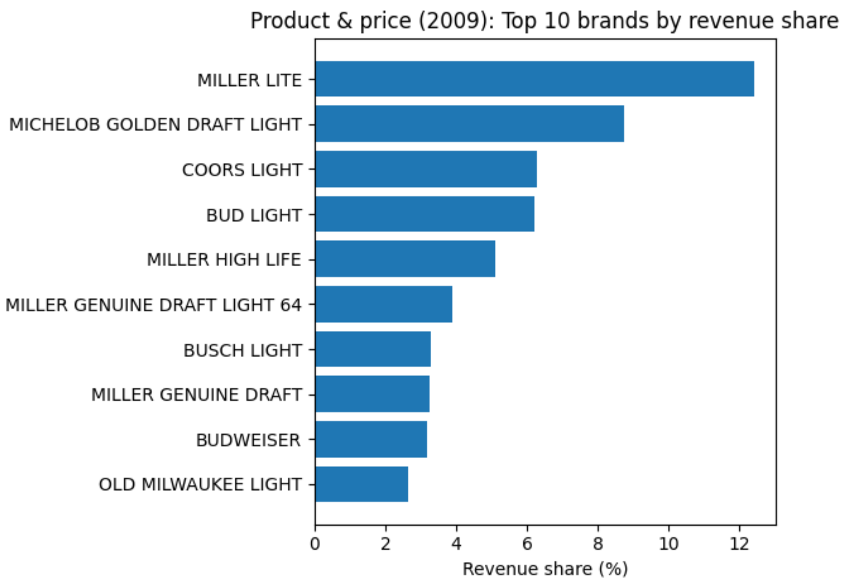


Figure 3

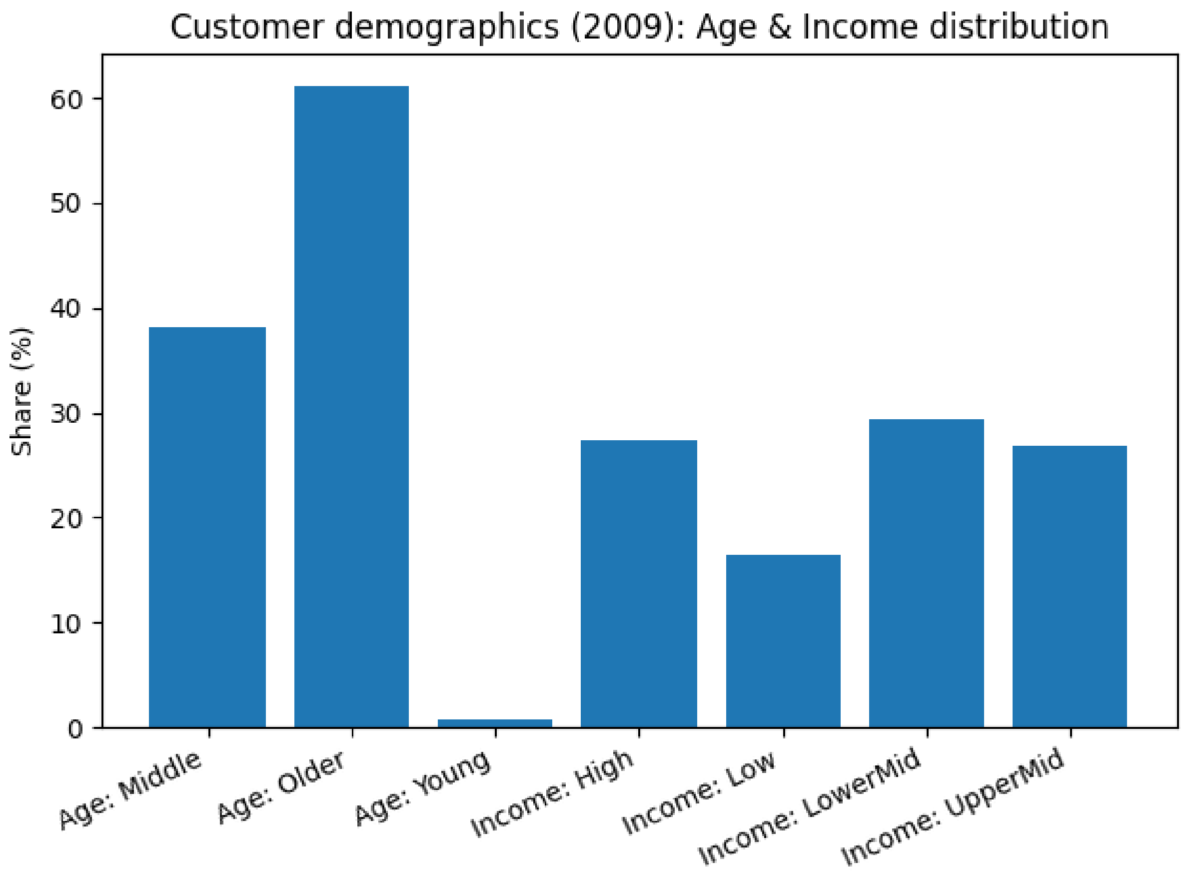


Figure 4

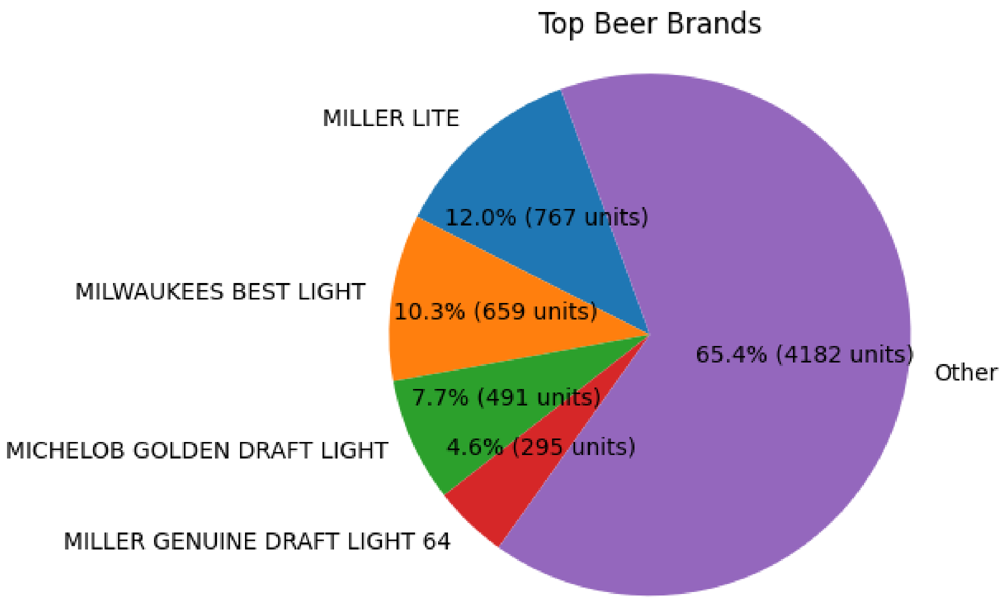


Figure 5

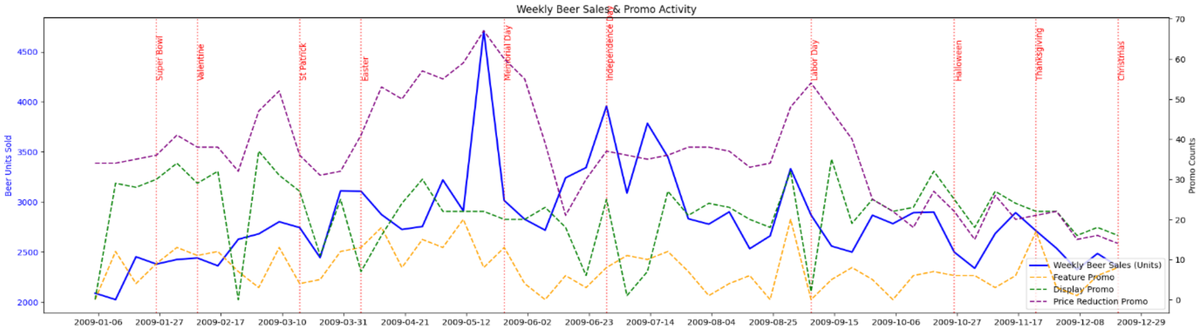


Figure 6

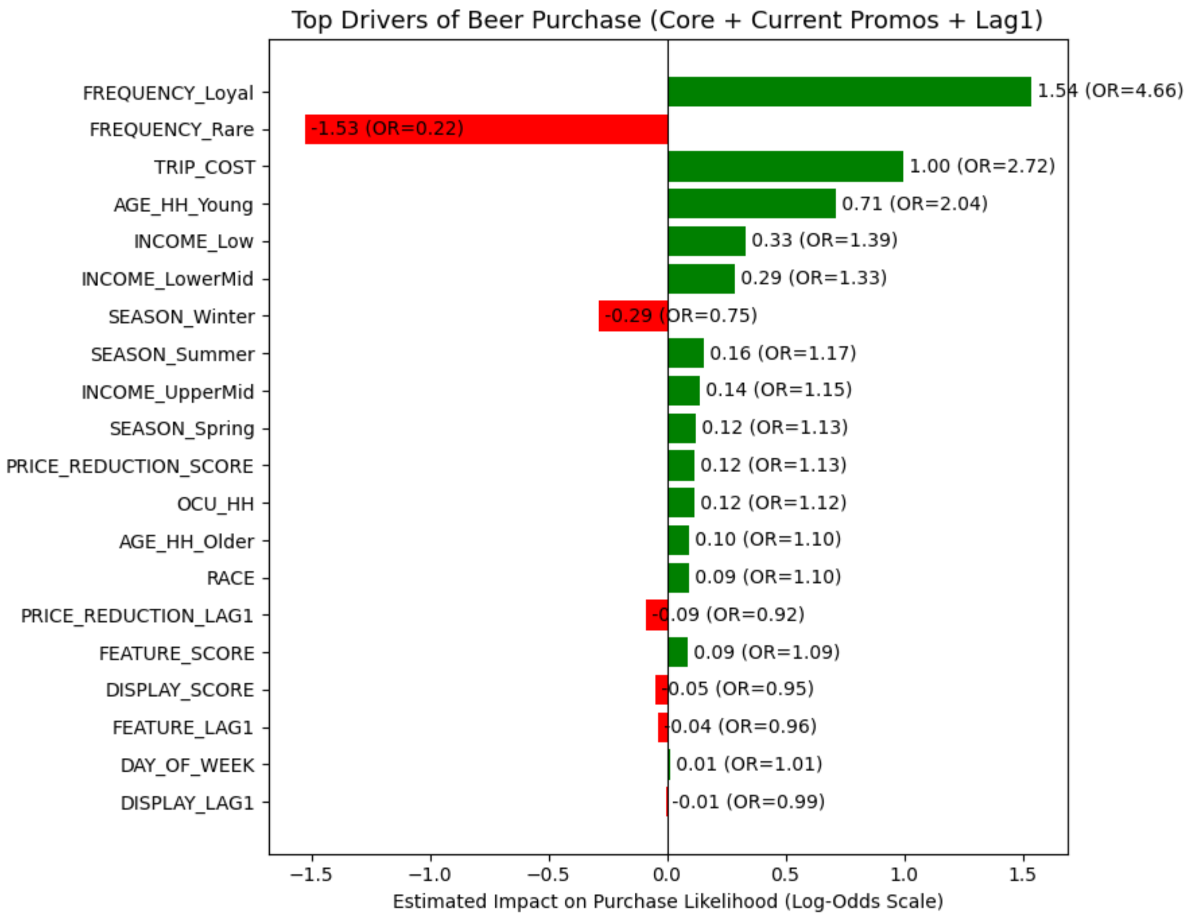


Figure 7

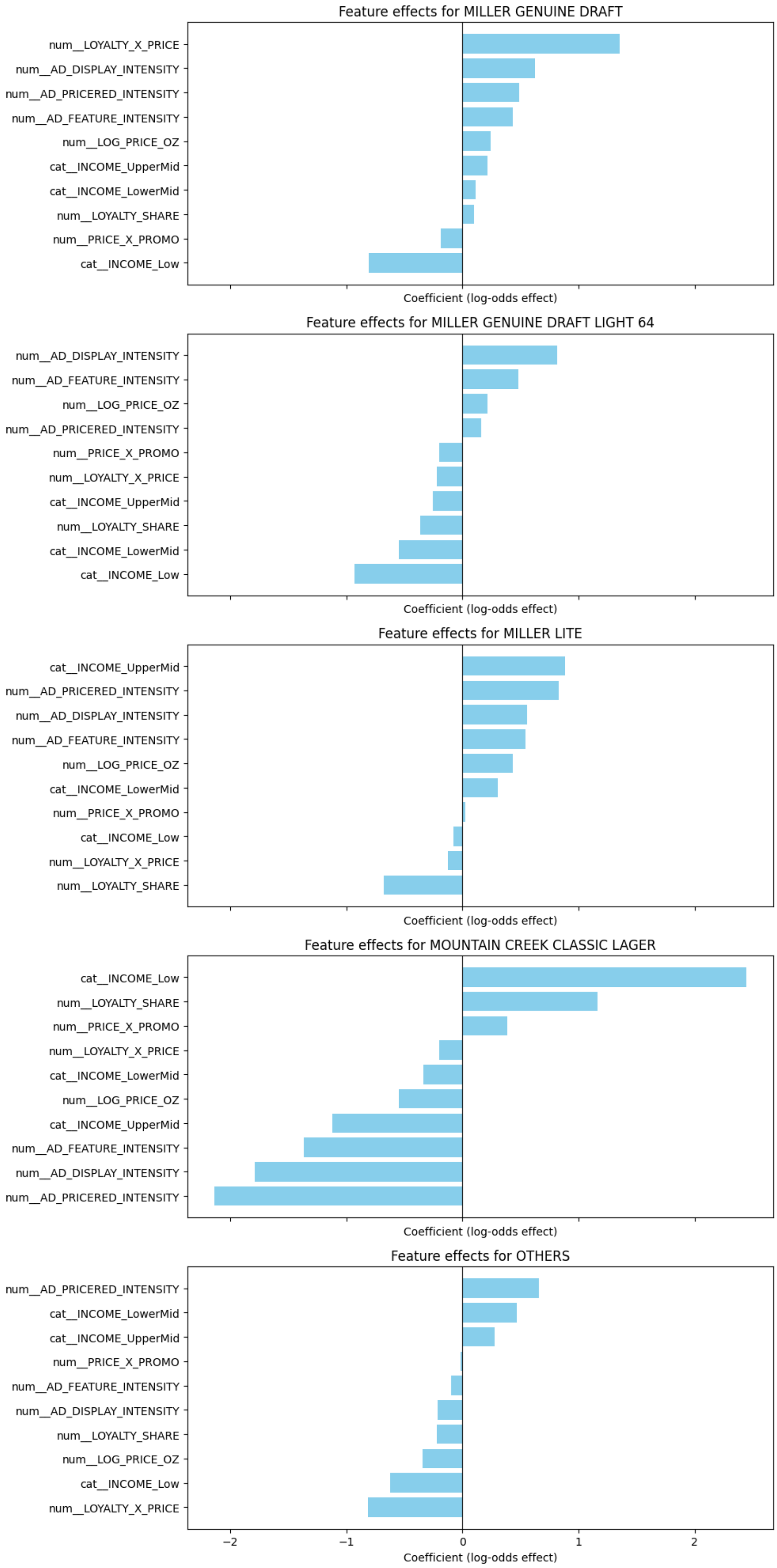
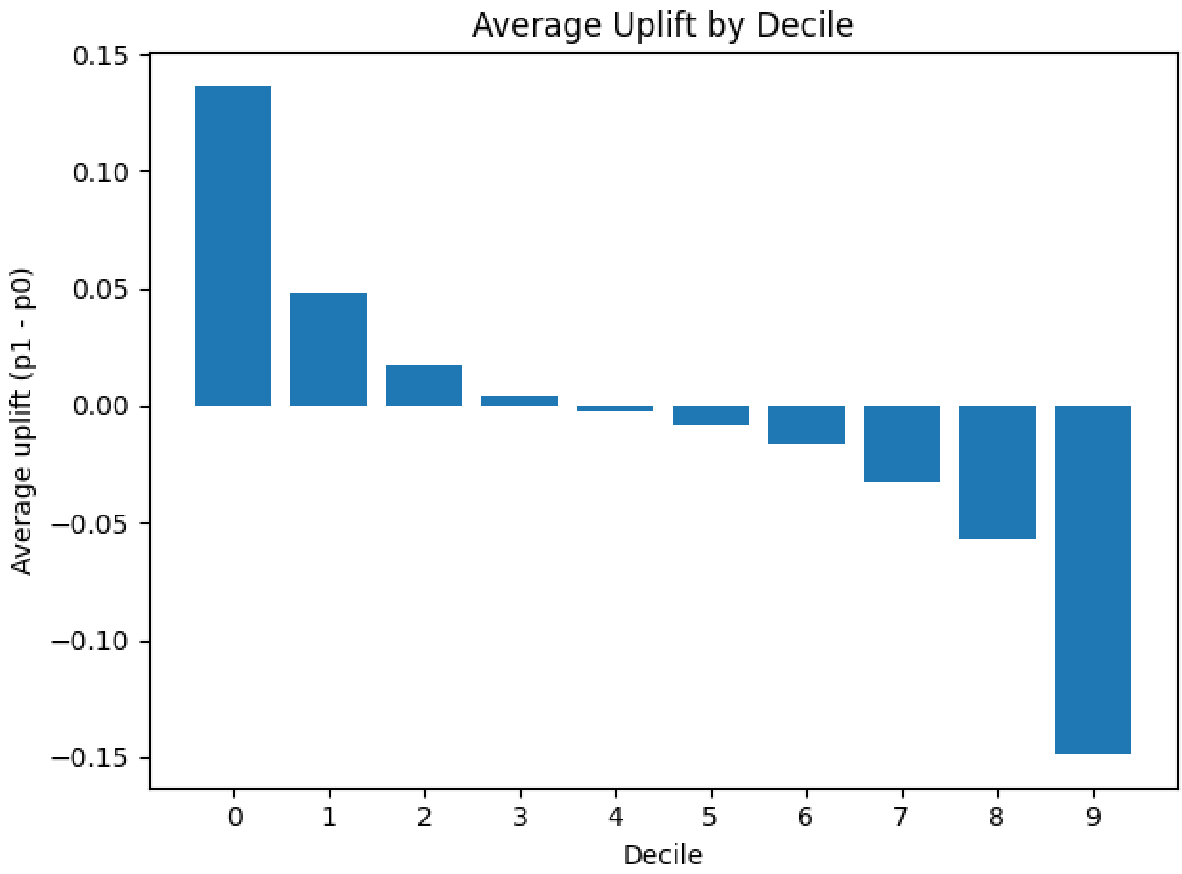

Feature effects for MILLER GENUINE DRAFT
num__LOYALTY_X_PRICE
num__AD_DISPLAY_INTENSITY
num__AD_PRICERED_INTENSITY
num__AD_FEATURE_INTENSITY
num__LOG_PRICE_OZ
cat__INCOME_UpperMid
cat__INCOME_LowerMid
num__LOYALTY_SHARE
num__PRICE_X_PROMO
cat__INCOME_Low
Coefficient (log-odds effect)
Feature effects for MILLER GENUINE DRAFT LIGHT 64
num__AD_DISPLAY_INTENSITY
num__AD_FEATURE_INTENSITY
num__LOG_PRICE_OZ
num__AD_PRICERED_INTENSITY
num__PRICE_X_PROMO
num__LOYALTY_X_PRICE
cat__INCOME_UpperMid
num__LOYALTY_SHARE
cat__INCOME_LowerMid
cat__INCOME_Low
Coefficient (log-odds effect)
Feature effects for MILLER LITE
cat__INCOME_UpperMid
num__AD_PRICERED_INTENSITY
num__AD_DISPLAY_INTENSITY
num__AD_FEATURE_INTENSITY
num__LOG_PRICE_OZ
cat__INCOME_LowerMid
num__PRICE_X_PROMO
cat__INCOME_Low
num__LOYALTY_X_PRICE
num__LOYALTY_SHARE
Coefficient (log-odds effect)
Feature effects for MOUNTAIN CREEK CLASSIC LAGER
cat__INCOME_Low
num__LOYALTY_SHARE
num__PRICE_X_PROMO
num__LOYALTY_X_PRICE
cat__INCOME_LowerMid
num__LOG_PRICE_OZ
cat__INCOME_UpperMid
num__AD_FEATURE_INTENSITY
num__AD_DISPLAY_INTENSITY
num__AD_PRICERED_INTENSITY
Coefficient (log-odds effect)
Feature effects for OTHERS
num__AD_PRICERED_INTENSITY
cat__INCOME_LowerMid
cat__INCOME_UpperMid
num__PRICE_X_PROMO
num__AD_FEATURE_INTENSITY
num__AD_DISPLAY_INTENSITY
num__LOYALTY_SHARE
num__LOG_PRICE_OZ
cat__INCOME_Low
num__LOYALTY_X_PRICE
−2
−1
0
1
2
Coefficient (log-odds effect)

Figure 8



8. Additional Sources

Guadagni, P. M., & Little, J. D. C. (1983). A logit model of brand choice calibrated on scanner data. Marketing Science, 2(3), 203–238.

Gupta, S. (1988). Impact of Sales Promotions on When, What, and How Much to Buy. Journal of Marketing Research, 25(4), 342–355.

[1911.08729] Response Transformation and Profit Decomposition for Revenue Uplift Modeling

What U.S. Data Should be Used to Measure the Price Elasticity of Demand for Alcohol?

https://goodwoodpub.com/index.php/ijfam/article/download/81/20/100?

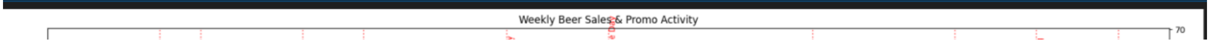
Relevant Code Segment

Datafile reading, handling missing values & feature engineering example

```python
BeerTransactionsDF = pd.read_csv("beer_PANEL_GK_1531_1582.DAT", delimiter=",")
BeerTransactionsDF = BeerTransactionsDF.rename(columns={'COLUPC': 'UPC', 'DOLLARS': 'BEER_PRICE', 'UNITS': 'BEER_UNITS'}).drop(columns=['OUTL

# Filter based on IRI_KEY
BeerTransactionsDF = BeerTransactionsDF[(BeerTransactionsDF['IRI_KEY'].isin(StoresDF['IRI_KEY']))]
print('Unique IRI_KEYs: ',BeerTransactionsDF['IRI_KEY'].unique())

# {0:'Mon', 1:'Tue', 2:'Wed', 3:'Thu', 4:'Fri', 5:'Sat', 6:'Sun'}
BeerTransactionsDF['DAY_OF_WEEK'] = (BeerTransactionsDF['MINUTE'] // 1440).astype(int)
BeerTransactionsDF['HOUR'] = (BeerTransactionsDF['MINUTE'] % 1440 // 60).astype(int)

# Merge "FEATURE","DISPLAY","PRICE_REDUCTION" & Beer Details
BeerTransactionsDF = (BeerTransactionsDF
    .merge(BeerDF, on="UPC", how="inner")
    .merge(WeekMapDF[['WEEK','WEEK_DATE','IS_HOLIDAY']], on="WEEK", how="left")
    )

# Create UNIT_PRICE & UNIT_PRICE_OZ
BeerTransactionsDF["PRICE_PER_UNIT"] = BeerTransactionsDF["BEER_PRICE"] / BeerTransactionsDF["BEER_UNITS"]

BeerTransactionsDF["PRICE_PER_OZ"] = BeerTransactionsDF["PRICE_PER_UNIT"] / BeerTransactionsDF["UNIT_OZ"]
BeerTransactionsDF['OZ'] = BeerTransactionsDF['UNIT_OZ'] * BeerTransactionsDF['BEER_UNITS']
BeerTransactionsDF['VOL_EQ'] = BeerTransactionsDF['BEER_UNITS'] * BeerTransactionsDF['VOL_EQ']

# Handle empty values
BeerTransactionsDF = BeerTransactionsDF.dropna()
```

EDA example

```python
# Plot sales on primary y-axis
ax1 = plt.gca()
ax1.plot(weekly_sales["WEEK_DATE"], weekly_sales["UNITS"], color="blue", label="Weekly Beer Sales (Units)", linewidth=2)
ax1.set_ylabel("Beer Units Sold", color="blue")
ax1.tick_params(axis="y", labelcolor="blue")

# Plot promo activity on secondary y-axis (right side)
ax2 = ax1.twinx()
ax2.plot(weekly_sales["WEEK_DATE"], weekly_sales["FEATURE"], label="Feature Promo", linestyle="--", color="orange")
ax2.plot(weekly_sales["WEEK_DATE"], weekly_sales["DISPLAY"], label="Display Promo", linestyle="--", color="green")
ax2.plot(weekly_sales["WEEK_DATE"], weekly_sales["PRICE_REDUCTION"], label="Price Reduction Promo", linestyle="--", color="purple")
ax2.set_ylabel("Promo Counts", color="black")

# Holiday annotations
for _, row in weekly_sales[weekly_sales["IS_HOLIDAY"]==1].iterrows():
    ax1.axvline(row["WEEK_DATE"], color="red", linestyle=":", alpha=0.6)
    if pd.notna(row["HOLIDAY"]):
        ax1.text(row["WEEK_DATE"], max(weekly_sales["UNITS"])*0.9, str(row["HOLIDAY"]),
                 rotation=90, color="red", fontsize=10)

# Legends
lines1, labels1 = ax1.get_legend_handles_labels()
lines2, labels2 = ax2.get_legend_handles_labels()
ax1.legend(lines1 + lines2, labels1 + labels2, loc="lower right")

# Format x-axis to show week dates clearly
plt.gca().xaxis.set_major_locator(mdates.WeekdayLocator(interval=3))  # every 4 weeks
plt.gca().xaxis.set_major_formatter(mdates.DateFormatter("%Y-%m-%d"))
plt.xticks(rotation=45)

plt.title("Weekly Beer Sales & Promo Activity")
plt.xlabel("Week")
plt.tight_layout()
plt.show()
```



Purchase Incidence modelling matrix

```python
# Beer frequency
cust_loyalty = df.groupby("PANID").agg(
    beer_trips=("BEER_PURCHASED", "sum"),
    total_trips=("WEEK_DATE", "count")
).reset_index()

cust_loyalty["FREQUENCY_RATIO"] = cust_loyalty["beer_trips"] / cust_loyalty["total_trips"]

def frequency_category(ratio):
    if ratio < 0.2:
        return "Rare"
    elif ratio < 0.5:
        return "Frequent"
    else:
        return "Loyal"

cust_loyalty["FREQUENCY"] = cust_loyalty["FREQUENCY_RATIO"].apply(frequency_category)

df = df.merge(cust_loyalty[["PANID", "FREQUENCY"]], on="PANID", how="left")

df
```

	PANID	WEEK_DATE	IRI_KEY	MINUTE	TRIP_COST	BEER_PURCHASED	IS_HOLIDAY	SEASON	INCOME	RESIDENCE_TYPE	AGE_HH	EDU_HH	OCU_HH	RACE	FEATURE_SCORE	DISPLAY_SCORE	PRICE_REDUCTION_SCORE
0	3100248	2008-12-29	257871	9649	3.559590	0	0	Winter	UpperMid	2	Middle	3	2	1	7	26	47
1	3100248	2009-02-02	257871	7935	3.696788	0	0	Winter	UpperMid	2	Middle	3	2	1	13	34	41
2	3100248	2009-02-09	257871	6607	3.479648	1	1	Winter	UpperMid	2	Middle	3	2	1	11	29	38
3	3100248	2009-02-09	257871	6607	3.479648	1	1	Winter	UpperMid	2	Middle	3	2	1	11	29	38
4	3100248	2009-02-09	257871	2761	3.554067	0	0	Winter	UpperMid	2	Middle	3	2	1	11	29	38
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
11528	3843052	2009-06-15	257871	9354	3.503440	0	0	Summer	LowerMid	1	Older	4	9	1	6	18	21
11529	3843052	2009-06-22	257871	9363	3.356866	1	0	Summer	LowerMid	1	Older	4	9	1	3	6	30
11530	3843052	2009-07-13	257871	9739	2.991674	1	0	Summer	LowerMid	1	Older	4	9	1	10	7	35
11531	3843052	2009-08-03	257871	9305	4.237819	1	0	Summer	LowerMid	1	Older	4	9	1	1	24	38
11532	3843052	2009-10-19	257871	9312	3.438493	0	0	Fall	LowerMid	1	Older	4	9	1	7	32	27

Brand Choice model build

```python
        remainder="drop"
    )

    clf = Pipeline(
        steps=[
            ("prep", pre),
            ("lr", LogisticRegression(
        penalty="l2",       # Ridge
        C=1.0,              # Regularization strength (lower = more penalty)
        solver="lbfgs",
        max_iter=2000
        ))
        ]
    )

    # 3) Fit
    X = base[num_cols + cat_cols]
    clf.fit(X, y)

    # 4) Quick metrics (in-sample)
    probs = clf.predict_proba(X)
    preds = clf.predict(X)
    print("Accuracy:", accuracy_score(y, preds))
    print("LogLoss :", log_loss(y, probs))

    # 5) Coefficients tidy table
    feature_names = clf.named_steps["prep"].get_feature_names_out()
    coefs = clf.named_steps["lr"].coef_          # shape: [n_classes, n_features]
    classes = clf.named_steps["lr"].classes_

    coef_df = (
        pd.DataFrame(coefs, index=classes, columns=feature_names)
          .T.reset_index().rename(columns={"index":"Feature"})
    )
    print(coef_df)
```

```
Accuracy: 0.7680574248541947
LogLoss : 0.7479897765378769
                Feature  MILLER GENUINE DRAFT  MILLER GENUINE DRAFT LIGHT 64  \
0          num__FEATURE              0.192281                       0.296660
1  num__PRICE_REDUCTION              0.123664                       0.169193
```

Uplift Model build

```python
# Store-week promo & treatment (joint feature & price reduction)
StoreTX["FEATURE_SCORE"] = (StoreTX["FEATURE"] > 0).astype(int)
StoreTX["PRICE_REDUCTION_SCORE"] = (StoreTX["PRICE_REDUCTION"] > 0).astype(int)
StoreTX["DISPLAY_SCORE"] = (StoreTX["DISPLAY"] > 0).astype(int)

promo_summary = (
    StoreTX.groupby(["IRI_KEY","WEEK_DATE"], as_index=False)
           .agg(FEATURE_SCORE=("FEATURE_SCORE","sum"),
                PRICE_REDUCTION_SCORE=("PRICE_REDUCTION_SCORE","sum"),
                DISPLAY_SCORE=("DISPLAY_SCORE","sum"))
)
promo_summary["TREATMENT"] = ((promo_summary["FEATURE_SCORE"]>0) & (promo_summary["PRICE_REDUCTION_SCORE"]>0)).astype(int)

# Modeling DF
df = (
    CustTX[["PANID","WEEK_DATE","IRI_KEY","MINUTE","TRIP_COST"]]
    .merge(BeerTX[["PANID","WEEK_DATE","IRI_KEY","MINUTE","BEER_PURCHASED","IS_HOLIDAY"]],
           on=["PANID","WEEK_DATE","IRI_KEY","MINUTE"], how="left")
    .merge(WeekMap[["WEEK_DATE","SEASON"]], on="WEEK_DATE", how="left")
    .merge(CustDemo[["PANID","INCOME","RESIDENCE_TYPE","AGE_HH","EDU_HH","OCU_HH","RACE"]],
           on="PANID", how="left")
    .merge(promo_summary[["WEEK_DATE","TREATMENT","FEATURE_SCORE","PRICE_REDUCTION_SCORE","DISPLAY_SCORE"]],
           on="WEEK_DATE", how="left")
)

df["TRIP_COST"] = np.log1p(df["TRIP_COST"])
for c in ["BEER_PURCHASED","IS_HOLIDAY","TREATMENT","FEATURE_SCORE","PRICE_REDUCTION_SCORE","DISPLAY_SCORE"]:
    df[c] = df[c].fillna(0).astype(int)
df["DAY_OF_WEEK"] = df["MINUTE"].floordiv(1440)

def collapse_income(x):
    if pd.isna(x): return np.nan
    if x <= 4: return "Low"
    elif x <= 7: return "LowerMid"
    elif x <= 10: return "UpperMid"
    else: return "High"
df["INCOME_BIN"] = df["INCOME"].apply(collapse_income)
```

```python
def collapse_age(x):
    if pd.isna(x): return np.nan
    if x <= 2: return "Young"
    elif x <= 4: return "Middle"
    else: return "Older"
df["AGE_HH_BIN"] = df["AGE_HH"].apply(collapse_age)

cust_loyalty = df.groupby("PANID").agg(
    beer_trips=("BEER_PURCHASED","sum"),
    total_trips=("WEEK_DATE","count")
).reset_index()
cust_loyalty["FREQUENCY_RATIO"] = np.where(cust_loyalty["total_trips"]>0, cust_loyalty["beer_trips"]/cust_loyalty["total_trips"], 0.0)

def frequency_category(r):
    if r < 0.2: return "Rare"
    elif r < 0.5: return "Frequent"
    else: return "Loyal"
cust_loyalty["FREQUENCY"] = cust_loyalty["FREQUENCY_RATIO"].apply(frequency_category)
df = df.merge(cust_loyalty[["PANID","FREQUENCY"]], on="PANID", how="left")

df["Y"] = df["BEER_PURCHASED"].astype(int)

# Features (drop direct promo fields to avoid leakage)
ordinal_cols = ["EDU_HH","OCU_HH","DISPLAY_SCORE"]
nominal_cols = ["INCOME_BIN","AGE_HH_BIN","SEASON","RESIDENCE_TYPE","RACE","FREQUENCY","DAY_OF_WEEK","IS_HOLIDAY"]
X_nom = pd.get_dummies(df[nominal_cols].astype("category"), drop_first=True)
X_num = pd.DataFrame(StandardScaler().fit_transform(df[["TRIP_COST"]]), columns=["TRIP_COST"], index=df.index)
X_base = pd.concat([df[ordinal_cols].copy(), X_nom, X_num], axis=1)

T = df["TREATMENT"].astype(int)
X_tx = X_base.multiply(T, axis=0); X_tx.columns = [f"T_{c}" for c in X_tx.columns]
X = pd.concat([X_base, T.rename("TREATMENT"), X_tx], axis=1)

# Split
X_train, X_test, y_train, y_test, T_train, T_test, panid_train, panid_test = train_test_split(
    X, df["Y"].astype(int), T, df["PANID"], test_size=0.30, random_state=42, stratify=T
)
```

```python
# Fit
clf = LogisticRegression(penalty="l1", solver="saga", max_iter=2000, n_jobs=-1, C=1.0, verbose=0)
clf.fit(X_train, y_train)

# Score uplift
def build_matrix_for_t(X_part, tval):
    Xb = X_part.copy()
    to_drop = [c for c in Xb.columns if c=="TREATMENT" or c.startswith("T_")]
    if to_drop: Xb = Xb.drop(columns=to_drop)
    Tcol = pd.Series(tval, index=Xb.index, name="TREATMENT")
    Xtx = Xb.multiply(tval, axis=0); Xtx.columns = [f"T_{c}" for c in Xtx.columns]
    return pd.concat([Xb, Tcol, Xtx], axis=1)

X1 = build_matrix_for_t(X_test, 1)
X0 = build_matrix_for_t(X_test, 0)
p1 = clf.predict_proba(X1)[:,1]
p0 = clf.predict_proba(X0)[:,1]
uplift = p1 - p0

# Deciles
eval_df = pd.DataFrame({
    "PANID": panid_test.values,
    "uplift": uplift,
    "y": y_test.values,
    "t": T_test.values
}).sort_values("uplift", ascending=False).reset_index(drop=True)

n = len(eval_df)
eval_df["bucket"] = (np.floor(np.arange(n)/max(1, n/10))).astype(int).clip(0,9)
decile = eval_df.groupby("bucket", as_index=False).agg(
    customers=("y","size"),
    treated=("t","sum"),
    avg_uplift=("uplift","mean")
)
decile["pct_customers"] = decile["customers"]/n
```

```python
# Economics
# U (units/beer trip)
U = BeerTX.loc[BeerTX["BEER_PURCHASED"]==1, "BEER_UNITS"].mean()

# P for treated (joint promo) weeks, weighted by units
promo_weeks = promo_summary.copy()
treated_weeks = set(promo_weeks.loc[promo_weeks["TREATMENT"] == 1, "WEEK_DATE"])
treated_mask = StoreTX["WEEK_DATE"].isin(treated_weeks)
P = (StoreTX.loc[treated_mask, "UNIT_PRICE"] * StoreTX.loc[treated_mask, "UNITS"]).sum() / StoreTX.loc[treated_mask, "UNITS"].sum()

tbl = decile.sort_values("bucket").reset_index(drop=True).copy()
tbl["IncrementalGrossProfit"] = tbl["avg_uplift"] * tbl["customers"] * U * P * MARGIN
tbl["ContactCost"] = tbl["customers"] * CONTACT_COST
tbl["MarginalDeltaProfit"] = tbl["IncrementalGrossProfit"] - tbl["ContactCost"]
tbl["CumulativeDeltaProfit"] = tbl["MarginalDeltaProfit"].cumsum()
total_rows = tbl["customers"].sum()
tbl["CumulativePctRows"] = tbl["customers"].cumsum() / (total_rows if total_rows>0 else 1)

best_idx = tbl["CumulativeDeltaProfit"].idxmax()
best_bucket = int(tbl.loc[best_idx, "bucket"])
best_pct_rows = float(tbl.loc[best_idx, "CumulativePctRows"])
best_cum_profit = float(tbl.loc[best_idx, "CumulativeDeltaProfit"])

# Customer-level targeting list
cust_uplift = eval_df.groupby("PANID", as_index=False).agg(
    mean_uplift=("uplift","mean"),
    max_uplift=("uplift","max"),
    n_rows=("uplift","size")
).sort_values("mean_uplift", ascending=False).reset_index(drop=True)

n_customers = len(cust_uplift)
target_n = int(round(best_pct_rows * n_customers))
target_n = max(1, min(target_n, n_customers))
targets = cust_uplift.head(target_n).copy()
targets["rank"] = np.arange(1, len(targets)+1)

# Display
print("Uplift Deciles — Joint Feature & Price Reduction (contact-only)")
tbl[["bucket","customers","avg_uplift","IncrementalGrossProfit","ContactCost","MarginalDeltaProfit","CumulativeDeltaProfit","CumulativePctRows"]]
```