价格弹性的因果计算
——Market Basket Analysis

张浩怡

2025 年 12 月 18 日

Outline

1 Introduction

2 Data Cleaning & EDA

3 Modeling
- Binning OLS
- Poisson and Ridge Regression
- DML

Outline

1 Introduction

2 Data Cleaning & EDA

3 Modeling
- Binning OLS
- Poisson and Ridge Regression
- DML

Implication of Price Elasticity

- 价格需求弹性反映了需求量对价格变动的敏感程度.

Calculate the price elasticity θ

虽然价格通过复杂的决策变量间接影响需求，但在数学上可简化为以下比率：
设 Q 为需求量，P 为价格，则弹性 θ 为：

$$\theta = \frac{\text{需求量变动百分比}}{\text{价格变动百分比}} = \frac{\mathrm{d}Q/Q}{\mathrm{d}P/P} \implies \log(Q) \sim \theta \log(P) \tag{1}$$

也即 θ 可以看作价格量变动对需求量变动的因果效应.

- 进一步，厂商 (Firm) 可依据弹性值优化定价策略：
  - 缺乏弹性 ($|\theta| < 1$) → 提价可增加收入
  - 富有弹性 ($|\theta| > 1$) → 降价可增加收入

Problem Solving Approach

1. 虽然 A/B 测试是评估价格弹性的理想手段，但在同一时期对不同用户展示差异化价格，会严重损害用户体验与品牌信誉，因此往往不可行.
2. 替代方案是基于历史观测数据进行因果推断. 然而，如何有效剥离季节性、产品质量变化等混杂因素的影响，是估计真实因果效应的核心难点.
3. 在使用普通 OLS 回归，Poisson 回归与 Ridge 回归的基础上，本研究进一步使用双重机器学习框架，主要解决以下两个问题:
   - **高维变量筛选**: 通过正则化技术，从大量特征中自动筛选出重要的控制变量;
   - **非线性拟合**: 相比传统线性回归，DML 引入非参数模型，能更准确地捕捉复杂的非线性关系.

Inference Process: Double Machine Learning

1. **样本分割与交叉拟合**: 将数据分为训练集和估计集. 利用随机森林模型从高维混杂变量 X 中学习非线性关系:
   - 估计处理变量（倾向性得分）: $g(X) = \mathbb{E}[P \mid X]$
   - 估计结果变量（基线需求）: $m(X) = \mathbb{E}[Q \mid X]$
2. **计算正交残差**:

$$\widetilde{P} = P - g(X), \quad \widetilde{Q} = Q - m(X) \tag{2}$$

3. **部分线性回归**: 基于残差建立回归方程, 消除偏差后得到的系数 θ 即为无偏估计量:

$$\ln\left(\widetilde{Q}\right) = \theta \cdot \ln\left(\widetilde{P}\right) + \varepsilon \tag{3}$$

Outline

1 Introduction

2 Data Cleaning & EDA

3 Modeling
- Binning OLS
- Poisson and Ridge Regression
- DML

Dataset Description

1. 本研究使用 Kaggle 公开数据集 "Association Rules and Market Basket Analysis", 共 541909 条数据.
2. 原始数据旨在进行购物篮关联分析, 记录了每一笔交易的详细清单 (Transaction Log), 见图 1.
3. 为了计算价格弹性, 我们需要对原始交易数据进行聚合处理, 将粒度从"单次交易"转换为"商品-时间"维度的销量与价格数据.
4. 此外, 还需要挖掘更多的协变量信息以剥离季节性、产品质量变化等混杂因素 (Confounders) 的影响.

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
443883	574714	22406	MONEY BOX KINGS CHOICE DESIGN	1	2011-11-06 14:29:00	1.25	15427.0	United Kingdom
432385	573871	84347	ROTATING SILVER ANGELS T-LIGHT HLDR	24	2011-11-01 12:34:00	2.55	14426.0	United Kingdom
36610	539451	22083	PAPER CHAIN KIT RETROSPOT	1	2010-12-17 16:59:00	5.91	NaN	United Kingdom
320462	564974	23322	LARGE WHITE HEART OF WICKER	5	2011-08-31 15:32:00	2.95	NaN	United Kingdom
248120	558777	84920	PINK FLOWER FABRIC PONY	2	2011-07-04 10:23:00	3.29	NaN	United Kingdom

图: Data Sample

Data Overview: Temporal Distribution

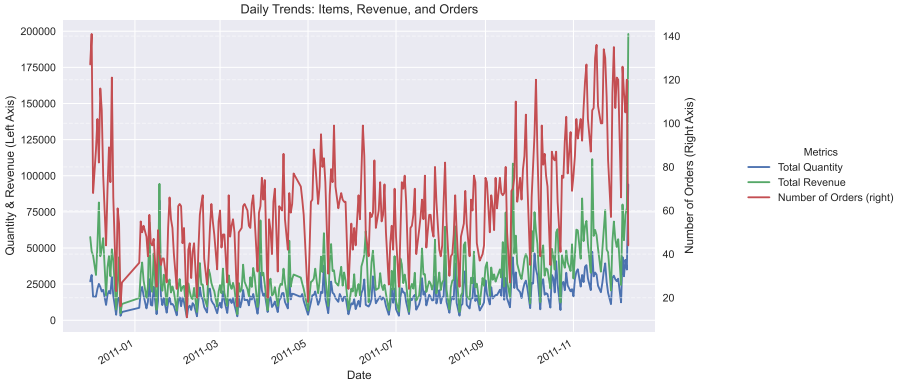

图: Daily Fluctuations: Items Sold vs. Transaction Count. 可以看到数据存在明显的波动.

Data Preprocessing

```python
# 剔除异常产品数据
df = df[~df.StockCode.isin(
    ['POST', 'DOT', 'M', 'AMAZONFEE', 'BANK CHARGES', 'C2', 'S']
    )] # 上述代码代表的是服务、罚款、运营成本或纠错记录

# 清洗控制变量
df['InvoiceDate'] = pd.to_datetime(df.InvoiceDate)
df['Date'] = pd.to_datetime(df.InvoiceDate.dt.date)
df['revenue'] = df.Quantity * df.UnitPrice

# 剔除异常偏差值，超出范围的波动属于数据噪音
df = (
    df.assign( # 当前订单的价格是" 标准价格" 的多少倍
    dNormalPrice=lambda d: d.UnitPrice / d.groupby('StockCode')
    .UnitPrice.transform('median')
    ).pipe(lambda d: d[(d['dNormalPrice'] > 1./3) &(d['dNormalPrice'] < 3.)] # 正常的商业调整
    ).drop(columns=['dNormalPrice'])
)

# 聚合处理，计算加权平均价格
df = df.groupby(['Date', 'StockCode', 'Country'], as_index=False).agg({
    'Description': 'first', 'Quantity': 'sum', 'revenue': 'sum'
})
df['Description'] = df.groupby('StockCode').Description.transform('first')
df['UnitPrice'] = df['revenue'] / df['Quantity']
```

Feature Engineering

为了解决因果推断中的内生性问题，构造了以下多维度的控制变量：

- Temporal Features：月份、日期、周几，用于剥离时间趋势的影响.
- Item Features：商品在架时长、历史中位数价格（锚点价）.

```
df = df.assign(
    # --- 时间混杂因素 (Time Confounders) ---
    month = lambda d: d.Date.dt.month,       # 季节性
    DoM   = lambda d: d.Date.dt.day,         # 月度周期
    DoW   = lambda d: d.Date.dt.weekday,     # 周度周期

    # --- 商品特征 (Product Characteristics) ---
    stock_age_days = lambda d: (
        d.Date - d.groupby("StockCode").Date.transform("min") # 产品在架时长
    ).dt.days, # 以防将清仓甩卖的高销量误认为是低价带来的正常弹性

    sku_avg_p = lambda d: d.groupby("StockCode").UnitPrice.transform(
        "median"                             # 该商品在历史所有时间段内的中位数价格
    )
) # 控制了商品异质性

```

Outline

1 Introduction

2 Data Cleaning & EDA

3 Modeling
- Binning OLS
- Poisson and Ridge Regression
- DML

Binning and Smoothing

在将单价和数量取 log，原始交易数据存在大量噪音，直接绘图难以观察趋势. 我们采用 Binscatter 方法:

1. **分箱**: 将自变量 X (如价格) 按分位数划分为 N 个等深区间.
2. **降噪**: 计算每个区间内的平均价格 $\overline{P}$ 和平均需求 $\overline{Q}$.
3. **拟合**: 基于均值点进行 OLS 回归.

```python
def binned_ols(df, x, y, n_bins=15):
    # 1. Binning (Quantile Cut)
    x_bin = x + '_bin'
    df[x_bin] = pd.qcut(df[x], n_bins)

    # 2. De-noising (Mean per bin)
    tmp = df.groupby(x_bin).agg({
        x: 'mean', y: 'mean'
    })

    # 3. Regression on Binned Data
    mdl = sm.OLS(
        tmp[y], sm.add_constant(tmp[x])
    )
    return mdl.fit()
```

Visualizing the Demand Curve

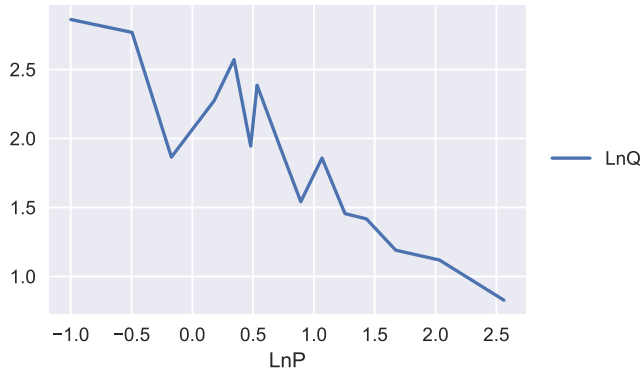

图: Observe messy relationship between $\ln(P)$ and $\ln(Q)$: Binning MSE: 0.064, 注意, 这仅说明价格与需求之间存在非常稳健的线性结构关系.

OLS Regression Result

表: OLS Regression Results

Dep. Variable:	LnQ	R-squared:	0.843
Model:	OLS	Adj. R-squared:	0.831
Method:	Least Squares	F-statistic:	70.05
Date:	Tue, 16 Dec 2025	Prob (F-statistic):	1.36e-06
Time:	10:17:36	Log-Likelihood:	0.39183
No. Observations:	15	AIC:	3.216
Df Residuals:	13	BIC:	4.632
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P> |t|	[0.025	0.975]
const	2.3252	0.085	27.321	0.000	2.141	2.509
LnP	-0.5949	0.071	-8.370	0.000	-0.748	-0.441

Omnibus:	1.908	Durbin-Watson:	2.520
Prob(Omnibus):	0.385	Jarque-Bera (JB):	0.381
Skew:	-0.233	Prob(JB):	0.826
Kurtosis:	3.626	Cond. No.	2.19

Incorporating Multivariate Covariates

```
feature_generator_full = ColumnTransformer([
        # 1. 商品固定效应：给每个 StockCode 一个独立截距，捕捉每个商品特有的基准销量.
        ("StockCode", OneHotEncoder(handle_unknown='ignore'), ["StockCode"]),
        # 2. 时间固定效应：对月份、日期、周几进行独热编码，捕捉非线性的季节性和周期性规律.
        ("Date", OneHotEncoder(handle_unknown='ignore'), ["month", "DoM", "DoW"]),
        # 3. 商品属性特征：从描述文本中提取 n-gram (1-3 词组)，捕捉细粒度属性对需求的影响.
        (
            "Description",
            CountVectorizer(min_df=0.0025, ngram_range=(1, 3)),
            "Description",
        ),
        # 4. 区域固定效应：捕捉不同国家的消费习惯差异.
        ("Country", OneHotEncoder(handle_unknown='ignore'), ["Country"]),
        # 5. 连续控制变量：标准化处理，防止数值较大的特征在正则化中占据过大权重.
        (
            "numeric_feats",
            StandardScaler(),
            ["stock_age_days", "sku_avg_p"],
        ),
        # 6. 处理变量：保留对数价格，作为回归的核心自变量，用于计算弹性系数 theta.
        ("LnP", "passthrough", ["LnP"]),
    ])
```

Estimation Results: Poisson & Ridge

表: Comparison of Estimated Price Elasticities ($\widehat{\theta}$) by Model Specification

Model Strategy	Elasticity (θ)	RMSE
Poisson (Count)	-2.963772	141.9903 (件)
Ridge (Log-Log)	-1.927997	146.3423 (件)

- Poisson 回归在 RMSE 指标上优于 Ridge 回归，说明了针对销量的离散计数特性建模比简单的对数线性转换更为准确.
- 尽管 Poisson 表现更优，但绝对误差 (RMSE ≈ 142) 仍处于较高水平. 这反映了微观层面的单品销量存在较大的随机波动，未来可考虑引入更多细粒度特征以提升预测精度.

DML: Model Specification

在双重机器学习框架下，我们针对价格和销量的不同数据分布特性，分别构建了第一阶段的 Nuisance Models:

1. **价格模型**: $P \sim X$
   - 算法: 随机森林
   - 理由: 价格受多种市场因素（季节、竞品、库存）的非线性影响，RF 能有效捕捉高维特征中的复杂交互关系.
2. **需求模型**: $Q \sim X$
   - 算法: Poisson 回归
   - 理由: 销量 (Q) 本质上是非负整数的计数数据，相比传统线性模型，Poisson 回归能更准确地拟合长尾分布，避免预测负值.

DML: Code

```python
# 定义用于建模数量的管道 model_q
model_q = Pipeline([
    # 第一阶段: 特征处理
    # 使用 feature_generator_full 对原始输入数据进行特征转换
    ('feat_proc', feature_generator_full),
    # 第二阶段: Poisson 回归模型, 适用于计数型 (非负整数) 目标变量
    ('model_q',
    linear_model.PoissonRegressor(
        alpha=1e-6,           # L2 正则化强度, 值越小正则化越弱.
        fit_intercept=False,  # 不拟合截距项 (通常在特征已中心化或包含常数列时使用)
        max_iter=100_000,     # 最大迭代次数, 设置较高以确保收敛
    )),
])
# 定义用于建模价格或其他连续目标的管道 model_p
model_p = Pipeline([
    # 第一阶段: 特征处理 (与 model_q 共享相同的特征工程流程)
    ('feat_proc', feature_generator_full),
    # 第二阶段: 随机森林回归器, 适用于非线性关系和特征交互
    ('model_p',
    RandomForestRegressor(
        n_estimators=50,       # 决策树的数量
        min_samples_leaf=3,    # 每个叶节点至少包含 3 个样本, 用于控制过拟合
    ))
])

```

DML: Progressive De-confounding

为了直观展示 DML 剔除混杂因素的效果，我们对比了三个不同处理阶段的数据，并分别进行分箱回归：

1. **原始数据**：$\ln P, \ln Q$：包含所有噪音和混杂因素，反映原始的市场相关性.
2. **去均值化数据**：$\mathrm{d}\ln(P), \mathrm{d}\ln(Q)$：剔除了商品层面的固定效应，仅保留组内变异.
3. **DML 正交残差**：$\mathrm{d}\ln\left(\widetilde{P}\right), \mathrm{d}\ln\left(\widetilde{Q}\right)$：利用机器学习剔除了所有观测到的混杂因素 (X)，反映纯粹的价格与需求因果关系.

	LnP	LnQ	dLnP	dLnQ	dLnP_res	dLnQ_res
117465	0.223144	2.484907	0.010178	-0.011537	-0.186340	-2.282933
233036	-0.198451	4.430817	-0.091114	1.175205	0.102530	-2.875243
208735	1.423108	3.332205	-0.016607	1.290633	-1.455620	-1.052630
75134	-0.867501	2.302585	-0.142779	-0.027194	0.666210	-1.517855
122280	0.500775	1.386294	-0.202359	-0.855506	-0.703135	-2.181317

图: Data Sample of Three Processing Stages

DML: Visualizing the Demand Curve

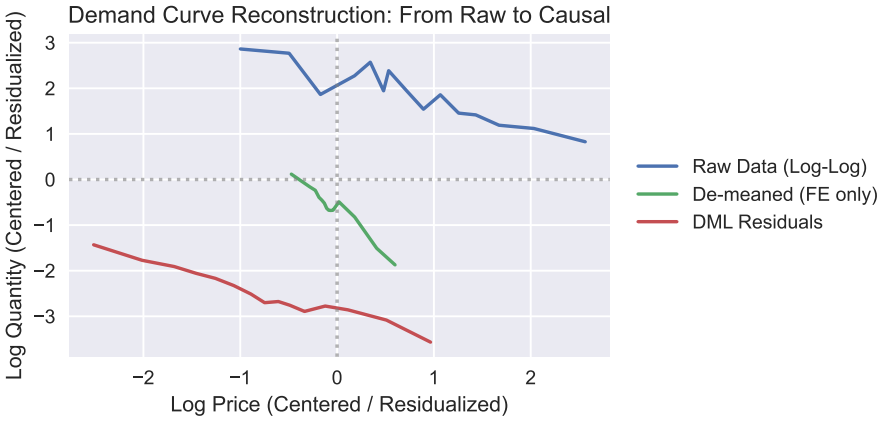

图: Binned Scatter Plots: From Raw Data to Orthogonalized Residuals

DML: Model Diagnostics

表: Evolution of Elasticity ($\widehat{\theta}$) and Goodness-of-Fit Across Stages

Stage	Elasticity ($\widehat{\theta}$)	Binned MSE	Binned RMSE
Raw Data	-0.5949	0.0641	0.2532
De-meaned	-1.8033	0.0138	0.1173
DML	-0.5812	0.0117	0.1080

- RMSE 从 Raw 的 0.25 降至 DML 的 0.108 (-57%). 这意味着 DML 成功剥离了大量非线性噪音, 分箱点最紧密地围绕回归线分布, 线性关系最强.

Robustness Strategy: Refined DML Estimation

1. 观察到 DML 结果与简单的去均值结果存在显著差异，表明单纯控制固定效应不足以消除时变混杂因素，DML 的引入是必要的.
2. 这是因为当价格残差 $\widetilde{P}$ 接近于零时（即价格变化完全被协变量解释），会导致估算不稳定.
3. 为了确保弹性估计的稳健性与无偏性，我们在标准 DML 基础上采用 Chernozhukov 提出的改进型 DML 估计量. 相比于传统的残差回归，该方法分母使用原始价格 P，对第一阶段的估计误差更具鲁棒性:

$$\widehat{\theta}_{\text{OLS}} = \frac{\widetilde{P}^{\top}\widetilde{Q}}{\widetilde{P}^{\top}\widetilde{P}} \xrightarrow{\text{Improved}} \widehat{\theta}_{\text{DML}} = \frac{\widetilde{P}^{\top}\widetilde{Q}}{\widetilde{P}^{\top}P}$$

4. 进一步采用 2-Fold 样本分割策略: 利用样本 A 训练辅助模型并预测样本 B 的残差（反之亦然），从而彻底消除过拟合带来的自身偏差.

DML Diagnostics: Cross-Fitting Results

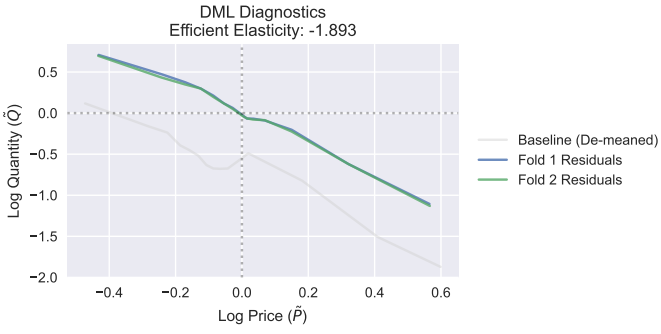

图: Diagnostic Plot: Binned Residuals vs. Fitted Line: DML Elasticity: -1.89, Binned RMSE: 0.047

Outlook: Towards Smart Pricing

1. 异质性分析 (From ATE to CATE): 当前模型计算的是全局平均弹性 (Global ATE), 掩盖了不同品类的差异. 是否利用 Causal Forest 或分层模型, 细化评估不同品类 (Category-level) 甚至单品 (SKU-level) 的价格敏感度差异?
2. 动态环境模拟 (Dynamic Simulation): 能否引入状态转移方程 (State Transition) 构建马尔可夫环境, 以捕捉跨期效应? 如何模拟用户囤货、需求透支等行为, 评估长期累积收益, 而非仅看单日销量?
3. 复杂策略优化 (Policy Optimization): 如何将弹性估算转化为具体的厂商决策支持. 探索多步决策问题, 如确定最佳促销频率或评估连续降价的综合效益.

SHANGHAI UNIVERSITY OF FINANCE & ECONOMICS
1917