

Price Elasticity of Demand and Dynamic Pricing in Online Retail: A Machine Learning Approach to Revenue Optimization

Harighanesh A and D. Senthilkumar*
 sri venkateswara college of engineering

Abstract—Pricing is among the most powerful levers a retailer controls, yet many online retailers still set prices with fixed mark-ups. This paper measures how sensitive demand is to price and tests whether that sensitivity can guide better prices. Using weekly product-level transactions of a UK-based online retailer, price elasticity is estimated three ways: a log-log fixed-effects regression, a gradient-boosting demand model with counterfactual price perturbation, and double machine learning that adjusts for seasonality and product effects. Elasticity is then compared across price tiers and seasons, and a guard-railed dynamic pricing policy, limited to $\pm 20\%$ changes inside each product's historical price range, is simulated on a chronological hold-out period. Results: the pooled elasticity is -3.24 (95% CI -3.74 to -2.73), so demand is strongly price-elastic, although volume discounts to wholesale customers probably exaggerate its size; double machine learning agrees (-3.19) but the machine learning counterfactual does not (positive sign); gradient boosting lowers forecast error (WAPE) by 16.8% against a historical-mean baseline; and the simulated pricing policy shows revenue gains of 77%, which are implausibly large and are read as artefacts of the estimated elasticity and the model, not as achievable gains. Implications, limits of observational price data and the need for controlled price experiments are discussed.

Index Terms—Dynamic pricing, price elasticity of demand, revenue optimization, demand forecasting, gradient boosting, double machine learning, retail analytics.

NOMENCLATURE

i, t	Product and week indices.
Q_{it}, P_{it}	Units sold and realised average price.
ε	Price elasticity of demand.
α_i, β	Product effect and pooled elasticity in the log-log model.
$\mathbf{x}_{it}$	Calendar and trend controls (month, week of year, peak flag, time index).
$\hat{Q}_i(P)$	Units predicted by the machine learning model at price P .
e_{it}	Counterfactual elasticity from the machine learning model.
θ	Double machine learning elasticity.
$p_{\min}^i, p_{\max}^i$	Price multipliers and the set of candidate multipliers. Lowest and highest training price of product i .
c_i, κ	Assumed unit cost and its ratio to median price.
R_{it}, Π_{it}	Revenue and profit of product i in week t .

Results come from the authors' run of the accompanying script on Online Retail II in Google Colab. Manuscript received October 23, 2026. The authors are with the Department of Artificial Intelligence and Data Science, Sri Venkateswara College of Engineering, Sriperumbudur, Chennai, India (e-mail: acm@svce.ac.in).

I. INTRODUCTION

Online retail makes prices visible, comparable and easy to change. Two ideas sit at the centre of data-driven pricing: *price elasticity of demand*, the percentage change in quantity demanded for a one percent change in price, and *dynamic pricing*, the adjustment of prices over time in response to demand, season, inventory or competition [1], [2]. When demand is elastic, a price cut raises revenue; when it is inelastic, a price rise does. Knowing which regime a product is in is therefore the foundation of any pricing decision.

Many retailers hold years of transaction history but have never run pricing experiments. Estimating elasticity from such observational data is difficult because prices are not set at random: they move with seasons, promotions and stock levels, which also move demand, so a naive regression of quantity on price can be biased and even give a positive elasticity. Machine learning models, in contrast, forecast demand well but do not by themselves identify a trustworthy price response [3], [4].

Related work in electricity retail has shown the value of learning price response from data and then optimising prices under explicit limits. Ruan *et al.* [5] learn appliance-level consumption behaviour with graph attention networks and search for customised retail prices whose changes are bounded to hedge risk for both consumer and retailer. This paper follows a similar philosophy, learning price response first and optimising second under stated guard-rails, but in product retail, where data are transactions rather than smart-meter readings and where elasticity can be estimated with econometric and causal machine learning tools.

The research questions are:

- RQ1.** How elastic is demand for an online retailer's products, and how much does it vary across products?
- RQ2.** Does elasticity differ by price tier and by season?
- RQ3.** Does a machine learning demand model forecast weekly

demand better than a baseline and a linear log-log model?

- RQ4.** Can a guard-railed pricing policy raise predicted revenue or profit relative to observed prices?

The contributions are fourfold. (1) A reproducible pipeline from raw transactions to pricing recommendations on open data.

(2) Triangulation of elasticity with fixed-effects regression, machine learning counterfactuals and double machine learning. (3) Segment-level evidence by price tier and season. (4) A pricing simulation with explicit constraints, cost sensitivity and an

honest account of its limits.

The rest of the paper is organised as follows. Section II gives preliminaries. Section III describes the data and Section IV the estimation framework. Section V defines the evaluation framework, and Section VI the guard-railed pricing model. Section VII presents numerical studies, and Section VIII concludes.

II. PRELIMINARIES

A. Price Elasticity of Demand

For a product with price P and quantity Q , the own-price elasticity is

$$\varepsilon = \frac{\Delta Q/Q}{\Delta P/P} \approx \frac{\partial \ln Q}{\partial \ln P} \quad (1)$$

so a log-log demand model delivers ε directly as a slope. Meta-analyses of econometric studies report average elasticities well below -1 and substantial variation by product type, data level and model [6], [7], which motivates product- and segment-level estimates rather than one industry average.

Since revenue is $R = PQ$, its sensitivity to price is

$$\frac{\partial R}{\partial P} = Q(1 + \varepsilon), \quad (2)$$

which is negative when $\varepsilon < -1$ (cutting price raises revenue) and positive when $-1 < \varepsilon < 0$ (raising price raises revenue).

B. Dynamic Pricing and Revenue Management

Dynamic pricing was formalised as a control problem under inventory and time limits [8] and consolidated in standard texts [1], [2]; surveys stress the trade-off between exploiting current demand estimates and experimenting to learn them [9]. With constant elasticity $\varepsilon < -1$ and unit cost c , the profit-maximising price follows the classical mark-up rule

$$P^* = \frac{c\varepsilon}{1 + \varepsilon}, \quad (3)$$

which shows why accurate elasticities matter: errors in ε translate directly into mispricing. Machine learning has been combined with price optimisation for an online flash-sales retailer [3], and algorithmic pricing is already widespread on large marketplaces, raising transparency questions [10].

III. DATA AND PROBLEM SETTING

A. Dataset

The *Online Retail II* dataset [11] contains transactions of a UK-based online retailer between December 2009 and December 2011. The firm mainly sells unique all-occasion gift-ware, and many customers are wholesalers. Each record holds an invoice number, stock code, description, quantity, invoice date, unit price (GBP), customer identifier and country.

B. Cleaning and Aggregation

Cancelled invoices (numbers starting with “C”), rows with non-positive quantity or price, and non-product codes such as postage, manual adjustments and bank charges are removed. Transactions are then aggregated to the product-week level (Table I). The

realised price is revenue divided by units. Products are kept if they have at least 40 weeks of sales and a price coefficient of variation of at least 0.03, since elasticity cannot be identified for a product whose price never changes; the 150 highest-revenue products among these form the panel. Weeks without sales have no logarithm, so the analysis is conditional on positive sales.

TABLE I
VARIABLES OF THE PRODUCT-WEEK PANEL

Variable	Definition
Q_{it}	Units of product i sold in week t
P_{it}	Revenue divided by units (GBP)
$\ln Q_{it}, \ln P_{it}$	Natural logarithms
Relative price	P_{it} over the product’s median training price
Month, week of year	Calendar position of week t
Peak flag	1 for October–December, else 0
Time index	Weeks since the first observation
Price tier	Tercile of the product’s median price

C. Chronological Split

The first 80% of weeks form the training set and the last 20% the test set; only products present in training are scored. Product base prices are computed from training weeks only to avoid look-ahead.

IV. PROPOSED ELASTICITY ESTIMATION FRAMEWORK

Fig. 1 summarises the pipeline. Three estimators, labelled A, B and C in the subsections below and in the tables, are used because each fails in a different way, and agreement between them is more credible than any single estimate.

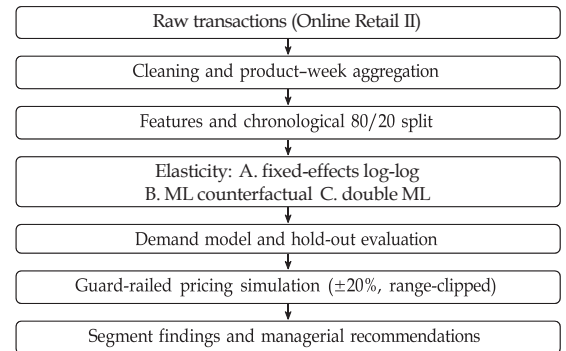


Fig. 1. Framework of the proposed analysis pipeline.

A. Log-Log Fixed-Effects Regression

$$\ln Q_{it} = \alpha_i + \beta \ln P_{it} + \theta_{m(t)} + \delta t + \varepsilon_{it}, \quad (4)$$

where α_i absorbs each product’s baseline popularity and price level, $\theta_{m(t)}$ are month effects capturing seasonality, δ is a linear trend and β is the pooled elasticity. Standard errors are clustered by product. Per-product regressions

$$\ln Q_{it} = a + b \ln P_{it} + c \text{peak}_t + d t + u_{it} \quad (5)$$

give the distribution of product-level elasticities (median, interquartile range and share significantly negative at the 5% level).

B. Machine Learning Counterfactual Elasticity

A histogram-based gradient boosting regressor [12], [13] predicts $\ln Q_{it}$ from product identity, $\ln P_{it}$, relative price, month, week of year, peak flag and time index. Let $\hat{Q}_{it}(P)$ be the predicted units at price P . The counterfactual elasticity re-predicts each test row with the price raised by one percent:

$$e_{it} = \frac{\ln \hat{Q}_{it}(1.01 P_{it}) - \ln \hat{Q}_{it}(P_{it})}{\ln 1.01}, \quad (6)$$

and is averaged by product and overall.

C. Double Machine Learning

Following [14], consider the partially linear model

$$\ln Q_{it} = \theta \ln P_{it} + g(\mathbf{z}_{it}) + u_{it}, \quad \ln P_{it} = m(\mathbf{z}_{it}) + v_{it}, \quad (7)$$

where $\mathbf{z}_{it}$ holds the controls (product mean $\ln Q$ and $\ln P$, month,

week of year, peak flag, time index). Random forests estimate g and m with 5-fold cross-fitting, giving residuals q_{it} and v_{it} . Then

$$\hat{\theta} = \frac{\sum \tilde{v}_{it} \tilde{q}_{it}}{\sum \tilde{v}_{it}^2}, \quad \hat{\sigma}^2 = \frac{\sum \tilde{v}_{it}^2 - \frac{(\sum \tilde{v}_{it} \tilde{q}_{it})^2}{\sum \tilde{v}_{it}^2}}{n-1}, \quad (8)$$

with $\tilde{u}_{it} = \tilde{q}_{it} - \hat{\theta} \tilde{v}_{it}$. The controls cannot remove confounding by unobserved promotions or stock-outs, so $\hat{\theta}$ is an adjusted, not randomised, estimate.

D. Segment Analysis

Products are split into terciles of median price (low, mid, high) and weeks into peak (October–December) and off-peak. Estimator A is re-estimated within each segment.

V. EVALUATION FRAMEWORK

Forecast accuracy on the hold-out weeks uses, on the unit scale, MAE

$$MAE = \frac{1}{n} \sum |Q - \hat{Q}|, \quad RMSE = \sqrt{\frac{1}{n} \sum (Q - \hat{Q})^2}, \quad (9)$$

$$WAPE = \frac{\sum |Q - \hat{Q}|}{\sum Q} \quad (10)$$

and on the log scale

$$R_{\ln}^2 = 1 - \frac{\sum (\ln Q - \ln \hat{Q})^2}{\sum (\ln Q - \ln \bar{Q})^2}. \quad (11)$$

WAPE is preferred to percentage errors because weekly units of slow products are small and near-zero denominators make percentage errors unstable. Benchmarks are each product's historical mean of $\ln Q$ in the training period and the fixed-effects regression. Elasticity estimates are also checked for sign, plausible magnitude and agreement across estimators.

VI. GUARD-RAILED DYNAMIC PRICING MODEL

A. Formulation

For a hold-out product-week (i, t) and multiplier m applied to the observed price, the candidate price is

$$P_{it}^*(m) = \min \{ \max \{ m P_{it}, P_{it}^{\min} \}, P_{it}^{\max} \}, \quad (12)$$

Algorithm 1 Guard-Railed Grid-Search Pricing

Require: Trained model $\hat{Q}$, hold-out rows, M , $[P_{it}^{\min}, P_{it}^{\max}]$, κ , objective J
Ensure: Multipliers m^* and simulated changes 1:
 2: **for** each hold-out row (i, t) **do**
 3: $J^* \leftarrow J_{it}(1)$, $m^* \leftarrow 1$ $\triangleright$ observed price is a candidate
 4: **for** each $m \in M \setminus \{1\}$ **do**
 $P_{it}^* \leftarrow \min \{ \max \{ m P_{it}, P_{it}^{\min} \}, P_{it}^{\max} \}$
 5: predict $\hat{Q}_{it}(P^*)$; compute $R_{it}(m)$, $\Pi_{it}(m)$
 6: **if** $J_{it}(m) > J^*$ **then**
 7: $J^* \leftarrow J_{it}(m)$, $m^* \leftarrow P^*/P_{it}$
 8: **end if**
 9: **end for**
 10: **end for**
 11: **return** m_{it}^* , $\Delta R\%$, $\Delta \Pi\%$

which keeps it inside the range of prices seen in training. Predicted revenue and profit are

$$R_{it}(m) = P_{it}^*(m) \hat{Q}_{it}(P_{it}^*(m)), \quad (13)$$

$$\Pi_{it}(m) = P_{it}^*(m) - c_i \hat{Q}_{it}(P_{it}^*(m)), \quad (14)$$

with the assumed unit cost $c_i = \kappa P_{it}^-$, where P_{it}^- is the median training price and $\kappa \in \{0.5, 0.6, 0.7\}$ because the data contain no cost information. The pricing model is

$$m_{it}^* = \arg \max_{m \in M} J_{it}(m), \quad M = \{0.80, 0.85, \dots, 1.20\}, \quad (15)$$

where J_{it} is either R_{it} or Π_{it} . The $\pm 20\%$ cap is a deliberate guard-rail, comparable in spirit to bounding price changes to limit consumer and retailer risk in [5]. Simulated gains are measured against the model's own prediction at the observed price, so the comparison isolates the policy from model error:

$$\Delta R\% = 100 \frac{\sum R_{it}(m_{it}^*)}{\sum R_{it}(1)} - 1, \quad (16)$$

and $\Delta \Pi\%$ analogously.

B. Algorithm

Because $\hat{Q}$ comes from a tree ensemble, the problem has no closed form, and a grid search over the small set M is exact and fast (Algorithm 1).

VII. NUMERICAL STUDIES

A. Set Up

All code is in Python with pandas, statsmodels, scikit-learn and matplotlib [15], [16]; the random seed is 42. The gradient boosting model uses learning rate 0.05, 400 iterations, maximum depth 6 and minimum leaf size 20; the double-ML forests use 150 trees with minimum leaf size 5. The panel contains 150 products and 13,885 product-weeks from 30 November 2009 to 5 December 2011, with mean weekly units 212.8, mean realised price 4.20 GBP and a mean within-product price coefficient of variation of 0.19. Experiments ran on a Google Colab cloud runtime (Python 3.13).

B. Elasticity Estimates (RQ1)

Table II reports the pooled estimates. The fixed-effects elasticity is -3.24 (95% CI -3.74 to -2.73), so a 10% price increase is

associated with a fall of about 27% in weekly units. The double-ML estimate is -3.19 , and the machine learning counterfactual averages $+1.15$. The fixed-effects and double-ML estimates agree closely and their intervals almost coincide, which supports the sign and rough size of the effect after adjusting for product, season and trend. The machine learning counterfactual disagrees and has the wrong sign: tree ensembles are piecewise constant, so a one percent price change rarely crosses a split, and price also acts as a proxy for product and season. The gradient boosting model is therefore used for forecasting and not as an elasticity estimator. At product level the median elasticity is -4.17 (interquartile range -5.39 to -3.12), and 92% of products are significantly negative (Fig. 2 and Fig. 3). These magnitudes are larger than the averages typically reported in meta-analyses [6], [7]. A likely reason is that many customers are wholesalers who receive volume discounts, so weeks with very large orders have low average prices, which mechanically links low price to high quantity. The estimates are best read as upper bounds on price sensitivity.

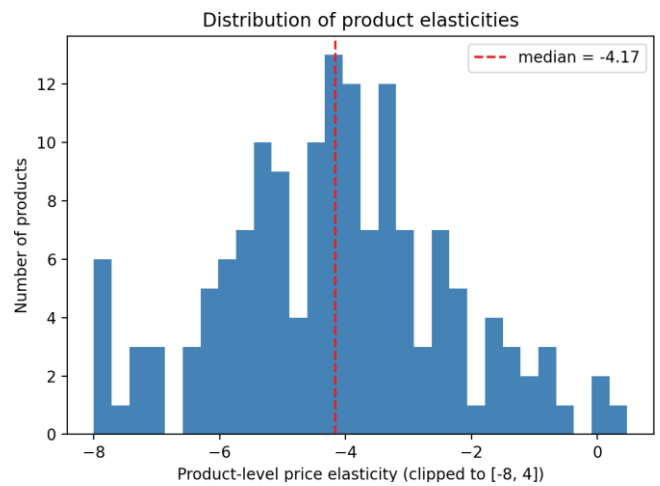


Fig. 3. Distribution of product-level elasticities (median marked).

TABLE II
 POOLED PRICE ELASTICITY ESTIMATES

Method	Elasticity	95% CI	n
A. Fixed-effects log-log	-3.24	$[-3.74, -2.73]$	13,885
C. Double ML	-3.19	$[-3.55, -2.84]$	13,885
B. ML counterfactual	$+1.15$	n/a	3,033

C. Segments (RQ2)

Table III compares price tiers and seasons. Point estimates are most elastic for mid-priced products (-3.55) and least elastic for low-priced ones (-2.56), and peak-season demand (-3.01) is slightly less elastic than off-peak demand (-3.24), which would fit gift buyers being less price-sensitive before Christmas.

However, the confidence intervals overlap heavily (for example peak -3.56 to -2.46 against off-peak -3.87 to -2.61), so the data do not show that price sensitivity differs significantly across tiers or seasons. The pattern is at most a hypothesis for future work.

TABLE III
 ELASTICITY BY PRICE TIER AND SEASON

Segment	Elasticity	SE	n
Price tier: low	-2.56	0.45	4,900
Price tier: mid	-3.55	0.28	4,350
Price tier: high	-3.31	0.57	4,635
Season: peak (Oct-Dec)	-3.01	0.28	3,601
Season: off-peak	-3.24	0.32	10,284

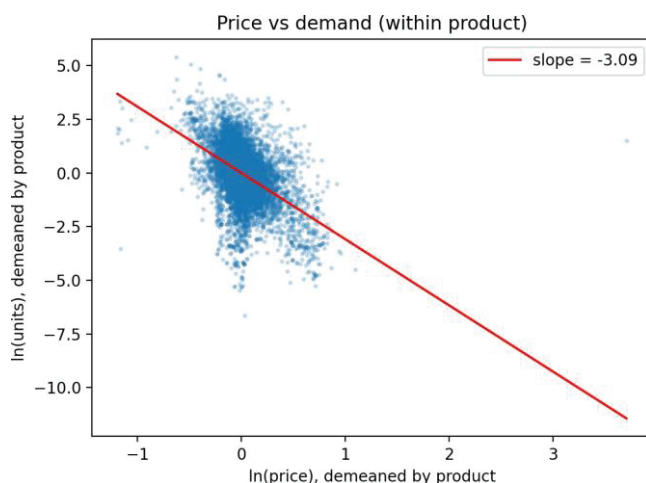


Fig. 2. Relationship between $\ln P$ and $\ln Q$ after removing product means (simple slope -3.09 ; one extreme price observation stretches the axis).

D. Forecast Accuracy (RQ3)

Table IV gives hold-out accuracy. Gradient boosting lowers WAPE from 0.593 to 0.493 relative to the historical-mean base-line (a 16.8% reduction) and is the best model on every metric ($\text{MAE } 108.6, \text{RMSE } 258.5, R^2_{\ln} 0.667$). The fixed-effects log-log model reaches a higher log-scale R than the baseline (0.508 against 0.385) but a higher unit-scale error (WAPE 0.631), which may reflect back-transforming log predictions. Permutation importance (Fig. 4) shows that product identity matters most, followed by the two price features, relative price and log price, and then week of year; month, the peak flag and the time index add almost nothing once week of year is included. Because the two price features are correlated, the importance of each understates the joint effect of price.

TABLE IV
 HOLD-OUT FORECAST PERFORMANCE

Model	MAE	RMSE	WAPE	R^2_{in}
Product historical mean	130.5	292.0	0.593	0.385
Fixed-effects log-log	139.0	338.0	0.631	0.508
Gradient boosting	108.6	258.5	0.493	0.667

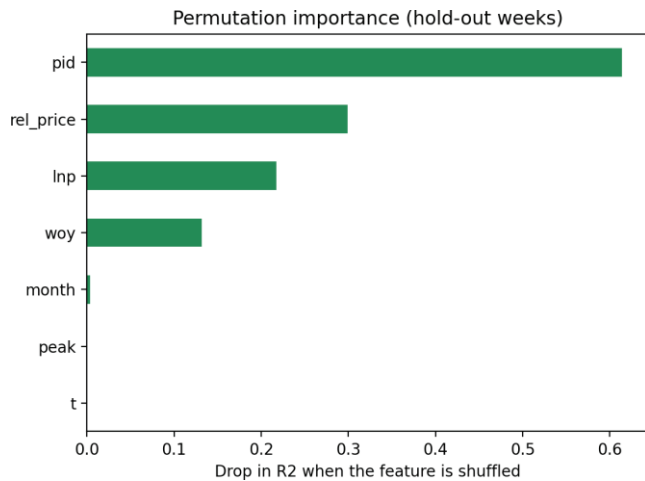


Fig. 4. Permutation importance on the hold-out weeks.

E. Pricing Simulation (RQ4)

Table V reports simulated outcomes. The revenue-maximising policy changes predicted revenue by +76.7% and lowers prices in 89.5% of weeks and raises them in 5.5%. Profit-maximising policies change predicted profit by +40.4% to +56.2% across $\kappa \in \{0.5, 0.6, 0.7\}$. The direction of the policy, mostly price cuts with an average multiplier below one, is the same for every cost ratio, while the size of the gain depends on κ . These gains are not credible as achievable results. They follow from an elasticity near -3 taken at face value: with $\varepsilon < -1$ every price cut raises revenue, so the optimiser pushes prices toward the lower guard-rail, and for most product-weeks the simulated revenue change exceeds 40% (Fig. 5). The elasticity is probably overstated (Section VII-B), the tree model is noisy, and choosing the best of several candidate prices from a noisy model overstates gains. The figures are an upper bound that illustrates the method and not a forecast; only a controlled price experiment can show what a price cut would really do.

TABLE V
 SIMULATED PRICING POLICY OUTCOMES

Objective	κ	$\Delta R\%$	$\Delta \Pi\%$	Avg. m	Down/Up
Revenue	0.6	+76.7	+42.4	0.88	89.5/5.5
Profit	0.5	+73.5	+56.2	0.90	81.0/11.9
Profit	0.6	+68.0	+48.8	0.92	75.1/16.9
Profit	0.7	+57.3	+40.4	0.95	64.9/26.2

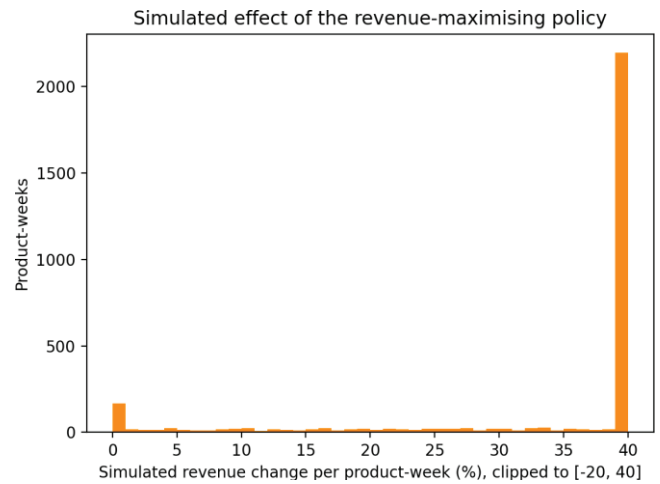


Fig. 5. Distribution of simulated revenue change per product-week, clipped to $[-20, 40]\%$; most product-weeks sit at the upper limit.

F. Discussion

Managerial implications. At face value the estimates say that most products are price-elastic, so targeted price cuts and pro-motions are worth testing, while the few inelastic products are candidates for margin protection. Because the estimated elasticities are probably inflated by volume discounts, cuts should be trialled on a small set of products first, and A/B price tests should confirm any gain before roll-out. Week-to-week changes should be capped, cost floors respected and prices kept inside the range the model has seen. Elasticities should be re-estimated periodically as assortment, competition and customer mix drift. **Limitations.** (i) The data are observational; unobserved promotions, stock-outs and competitor prices can bias estimates even with controls. (ii) Cost, promotion, competitor and inventory data are absent, so κ is an assumption. (iii) Weeks without sales are excluded, so estimates are conditional on positive sales. (iv) Many customers are wholesalers, which limits transfer to consumer retail and, through volume discounts, probably inflates the estimated elasticities. (v) Weekly prices are revenue-weighted averages that can mix discounts and bulk deals, so price and quantity are mechanically linked. (vi) Gains are simulated, and back-transforming log predictions slightly understates unit forecasts. (vii) A single temporal split is used; rolling-origin evaluation would be more reliable. (viii) The machine learning counterfactual elasticity has the wrong sign, so the pricing simulation, which relies on the same model, is unreliable in size. **Ethics.** Dynamic pricing raises concerns about fairness, transparency and customer trust and may be regulated by consumer-protection law [10]. The policy here prices products over time and does not personalise prices to individuals.

VIII. CONCLUSION

This paper built a reproducible pipeline that estimates price elasticity with three complementary methods, compares demand forecasting models and simulates a guard-railed dynamic pricing policy on open retail data. Demand for the retailer's products is

strongly price-elastic (pooled elasticity about -3.2 , with fixed-effects regression and double machine learning agreeing), although volume discounting probably exaggerates the magnitude. Gradient boosting forecasts weekly demand 16.8% better than a historical-mean baseline in WAPE, but its counterfactual price re-sponse has the wrong sign, so forecasting accuracy and elasticity estimation should be treated as separate tasks. Differences across price tiers and seasons are not statistically distinguishable. The simulated pricing gains are very large and should be read as an upper bound produced by the elasticity estimate, not as achievable improvements. Future work includes randomised price experiments, richer controls for promotions and competition, hierarchical Bayesian elasticity models that share information across products, explainability with SHAP [17], stock-aware optimisation, and bandit-based pricing that learns while it earns.

ACKNOWLEDGMENT

Harighanesh A thanks D. Senthilkumar, Department of Artificial Intelligence and Data Science, Sri Venkateswara College of Engineering, for his guidance, supervision and feedback throughout this work. The authors also thank the donor and maintainers of the Online Retail II dataset and the UCI Machine Learning Repository for making the data publicly available.

Data and code availability. The Online Retail II dataset is publicly available from the UCI Machine Learning Repository [11]. The analysis script used to produce all tables and figures is available from the authors on request.

Use of AI tools. Anthropic's Claude was used to assist with drafting and editing the manuscript and with writing the analysis code and L^AT_EX source. The authors executed the analysis on the public dataset and take full responsibility for the content of the paper.

REFERENCES

- [1] K. T. Talluri and G. J. van Ryzin, *The Theory and Practice of Revenue Management*. New York, NY, USA: Springer, 2004.
- [2] R. L. Phillips, *Pricing and Revenue Optimization*. Stanford, CA, USA: Stanford Univ. Press, 2005.
- [3] K. J. Ferreira, B. H. A. Lee, and D. Simchi-Levi, "Analytics for an online retailer: Demand forecasting and price optimization," *Manuf. Serv. Oper. Manage.*, vol. 18, no. 1, pp. 69–88, 2016.
- [4] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32,

2001.

- [5] J. Ruan *et al.*, "Graph deep-learning-based retail dynamic pricing for demand response," *IEEE Trans. Smart Grid*, vol. 14, no. 6, pp. 4385–4397, Nov. 2023, doi: 10.1109/TSG.2023.3258605.
- [6] G. J. Tellis, "The price elasticity of selective demand: A meta-analysis of econometric models of sales," *J. Marketing Res.*, vol. 25, no. 4, pp. 331–341, 1988.
- [7] T. H. A. Bijmolt, H. J. van Heerde, and R. G. M. Pieters, "New empirical generalizations on the determinants of price elasticity," *J. Marketing Res.*, vol. 42, no. 2, pp. 141–156, 2005.
- [8] G. Gallego and G. van Ryzin, "Optimal dynamic pricing of inventories with stochastic demand over finite horizons," *Manage. Sci.*, vol. 40, no. 8, pp. 999–1020, 1994.
- [9] A. V. den Boer, "Dynamic pricing and learning: Historical origins, current research, and new directions," *Surv. Oper. Res. Manage. Sci.*, vol. 20, no. 1, pp. 1–18, 2015.
- [10] L. Chen, A. Mislove, and C. Wilson, "An empirical analysis of algorithmic pricing on Amazon Marketplace," in *Proc. 25th Int. Conf. World Wide Web*, 2016, pp. 1339–1349.
- [11] D. Chen, "Online Retail II," UCI Machine Learning Repository, 2019. [Dataset].
- [12] G. Ke *et al.*, "LightGBM: A highly efficient gradient boosting decision tree," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017.
- [13] F. Pedregosa *et al.*, "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, 2011.
- [14] V. Chernozhukov *et al.*, "Double/debiased machine learning for treatment and structural parameters," *Econometrics J.*, vol. 21, no. 1, pp. C1–C68, 2018.
- [15] S. Seabold and J. Perktold, "Statsmodels: Econometric and statistical modeling with Python," in *Proc. 9th Python Sci. Conf.*, 2010.
- [16] W. McKinney, "Data structures for statistical computing in Python," in *Proc. 9th Python Sci. Conf.*, 2010.
- [17] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017.

Harighanesh A is a final-year B.Tech. student in Artificial Intelligence and Data Science at Sri Venkateswara College of Engineering, Sriperumbudur, Chennai, India. He has completed internships as a Data Analysis intern at Mitochondrion Info Tech Pvt Ltd (July–August 2026) and as an AI intern at Codework AI, where his projects included retail demand forecasting with seasonal feature engineering and Random Forest models, credit card fraud detection, sales prediction and sentiment analysis. He is also a Business Analyst intern at MD Smart Advisory (August–November 2026). His research interests include business analytics, pricing and demand forecasting.