# Retail Pricing Strategy Optimization

**Table of Contents**

## 1. Overview

This project addresses the challenge of optimizing product pricing in the retail industry using machine learning. By analyzing historical sales and product data, a predictive model was built to estimate how different pricing strategies affect sales quantity and revenue. The model is deployed in a Streamlit web application to support real-time pricing simulation.

## 2. Introduction

In a highly competitive retail environment, data-driven pricing is essential to maximize profitability and customer satisfaction. Traditional pricing relies on static rules or guesswork. This project aims to build a complete AI/ML solution that enables businesses to simulate pricing scenarios, forecast revenue, and evaluate demand elasticity.

## 3. Problem Statement

Ineffective pricing strategies result in suboptimal sales, excess inventory, and missed revenue opportunities. The objective is to leverage historical sales data to create a machine learning model that predicts the impact of price changes on sales volume and revenue, enabling optimal pricing decisions.

## 4. Business Value

- Maximize revenue and profit margin
- Support pricing teams with data-driven recommendations
- Reduce underpricing or overpricing risks
- Understand price sensitivity and customer behavior
- Improve stock management by forecasting demand

# 5. Dataset Description

| Feature Name | Description |
|---|---|
| product_category_name | Product category name |
| unit_price | Price per unit |
| qty | Quantity sold |
| total_price | unit_price × qty |
| freight_price | Shipping cost |
| product_score | Customer rating |
| product_weight_g | Product weight in grams |
| comp_1 | Competitor price |
| weekday/weekend/holiday | Temporal features |
| volume | Size-based product metric |

# 6. Methodology

## 6.1 Data Preprocessing

- Removed nulls and duplicates

- Converted date strings to datetime

- Label-encoded product categories

- Scaled numerical columns

- Handled outliers using IQR

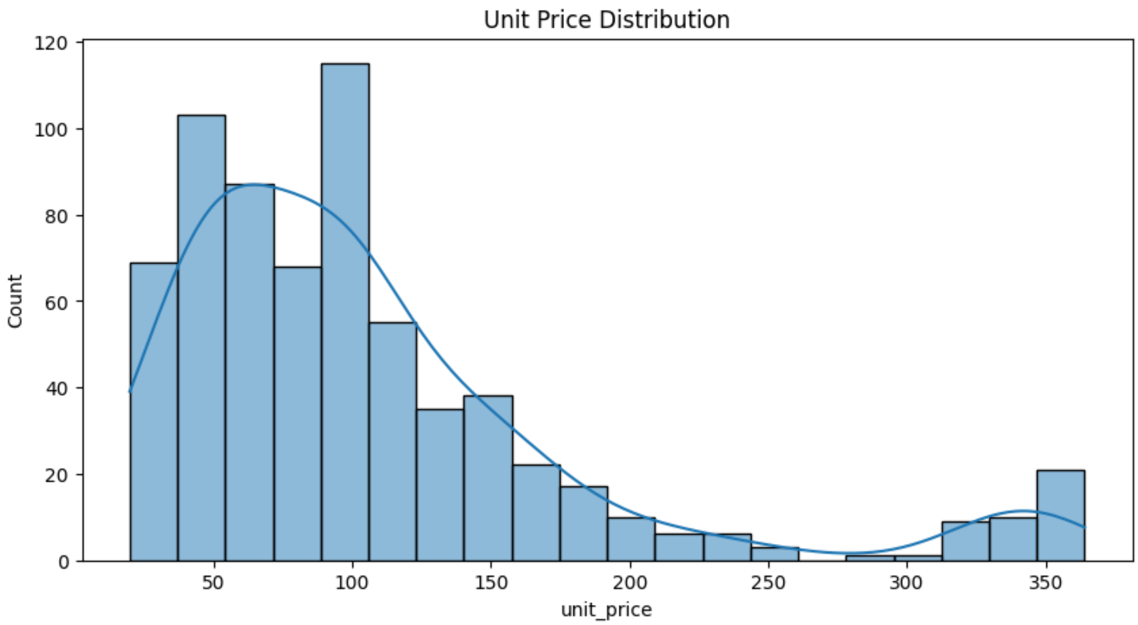
## 6.2 Exploratory Data Analysis (EDA)

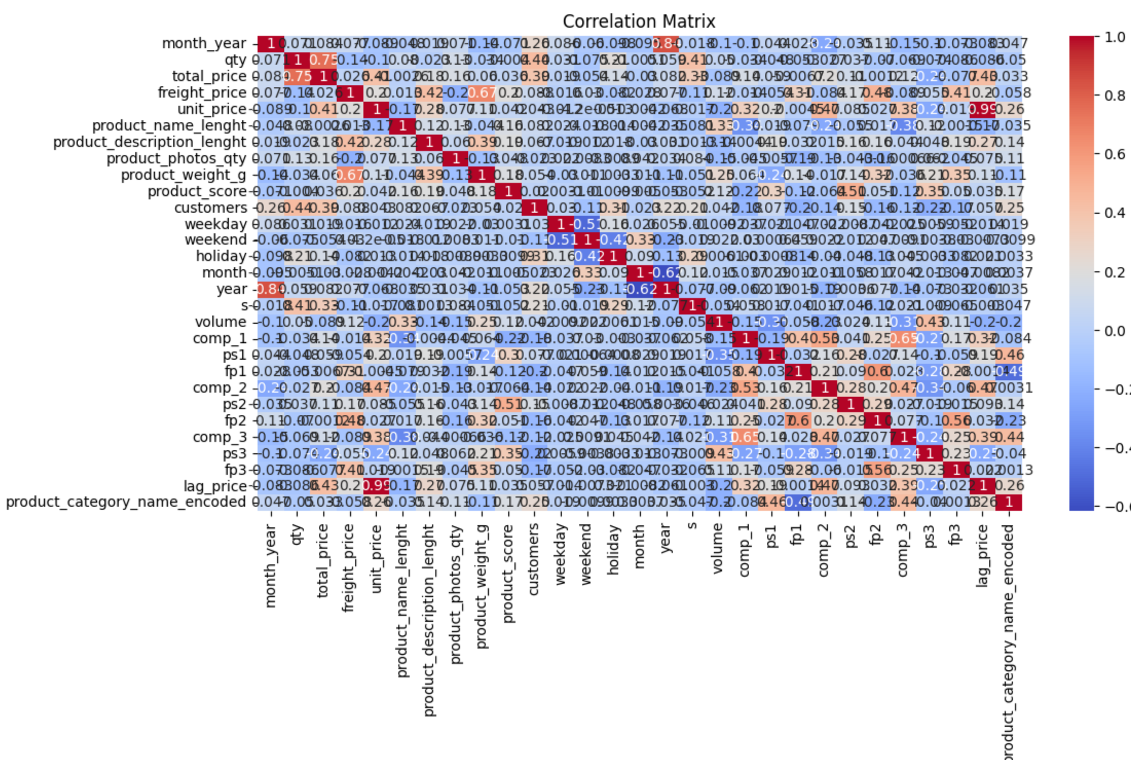
- Price vs. quantity distribution across categories

- Correlation heatmaps

- Demand trends over time

- Visual insights using Seaborn/Matplotlib


Unit Price Distribution
Count
unit_price
0
20
40
60
80
100
120
50
100
150
200
250
300
350


Correlation Matrix
month_year
qty
total_price
freight_price
unit_price
product_name_lenght
product_description_lenght
product_photos_qty
product_weight_g
product_score
customers
weekday
weekend
holiday
month
year
s
volume
comp_1
ps1
fp1
comp_2
ps2
fp2
comp_3
ps3
fp3
lag_price
product_category_name_encoded
1.0
0.8
0.6
0.4
0.2
0.0
-0.2
-0.4
-0.6

## 6.3 Model Building

- Target: unit_price or qty

- Models: Linear Regression, Random Forest

- Feature set: volume, product_score, comp_1, category, etc.

## 6.4 Model Evaluation

- Metrics: MSE, RMSE, $R^2$

- Random Forest outperformed Linear Regression

- Feature importance: volume > score > comp_1

### Evaluation

```python
def evaluate_model(name, y_true, y_pred):
    print(f"\n{name} Evaluation:")
    print("MSE:", mean_squared_error(y_true, y_pred))
    print("RMSE:", np.sqrt(mean_squared_error(y_true, y_pred)))
    print("R2 Score:", r2_score(y_true, y_pred))
```

[22] Python

```python
evaluate_model("Linear Regression", y_test, y_pred_lr)
evaluate_model("Random Forest", y_test, y_pred_rf)
```

[23] Python

```
...
Linear Regression Evaluation:
MSE: 66.23691974472548
RMSE: 8.138606744690733
R2 Score: 0.987739112487981

Random Forest Evaluation:
MSE: 43.96479326807407
RMSE: 6.630595242365052
R2 Score: 0.9918618289191814
```

## 6.5 Price Elasticity Analysis

- Formula: Elasticity = % change in quantity / % change in price

- Elastic categories respond significantly to price changes

- Inelastic categories are price stable

## Price Elasticity of Demand

```python
df['price_change_pct'] = df['unit_price'].pct_change()
df['qty_change_pct'] = df['qty'].pct_change()
df['elasticity'] = df['qty_change_pct'] / df['price_change_pct']
print("\nElasticity Sample:")
display(df[['unit_price', 'qty', 'price_change_pct', 'qty_change_pct', 'elasticity']].head(10))
```

[24] Python

```
Elasticity Sample:
```

| | unit_price | qty | price_change_pct | qty_change_pct | elasticity |
|---|---|---|---|---|---|
| 0 | 45.950000 | 1 | NaN | NaN | NaN |
| 1 | 45.950000 | 3 | 0.000000 | 2.000000 | inf |
| 2 | 45.950000 | 6 | 0.000000 | 1.000000 | inf |
| 3 | 45.950000 | 4 | 0.000000 | -0.333333 | -inf |
| 4 | 45.950000 | 2 | 0.000000 | -0.500000 | -inf |
| 5 | 45.950000 | 3 | 0.000000 | 0.500000 | inf |
| 6 | 40.531818 | 11 | -0.117915 | 2.666667 | -22.615213 |
| 7 | 39.990000 | 6 | -0.013368 | -0.454545 | 34.003203 |
| 8 | 39.990000 | 19 | 0.000000 | 2.166667 | inf |
| 9 | 39.990000 | 18 | 0.000000 | -0.052632 | -inf |

# 7. Deployment

- Model saved using joblib

- Interactive Streamlit dashboard for simulation

- Features: dropdown, slider, prediction output, visualization, CSV download

# 8. Results and Observations

- RF achieved lowest RMSE

- Elasticity and EDA aligned

- Certain categories showed strong price sensitivity

- Streamlit app allows easy simulation

## 9. Conclusion

This project successfully implemented an end-to-end ML solution for dynamic retail pricing. It provides actionable insights for business users and supports interactive decision-making via the Streamlit dashboard. Future improvements could include time-series forecasting and competitor scraping.

## 10. Tools and Technologies Used

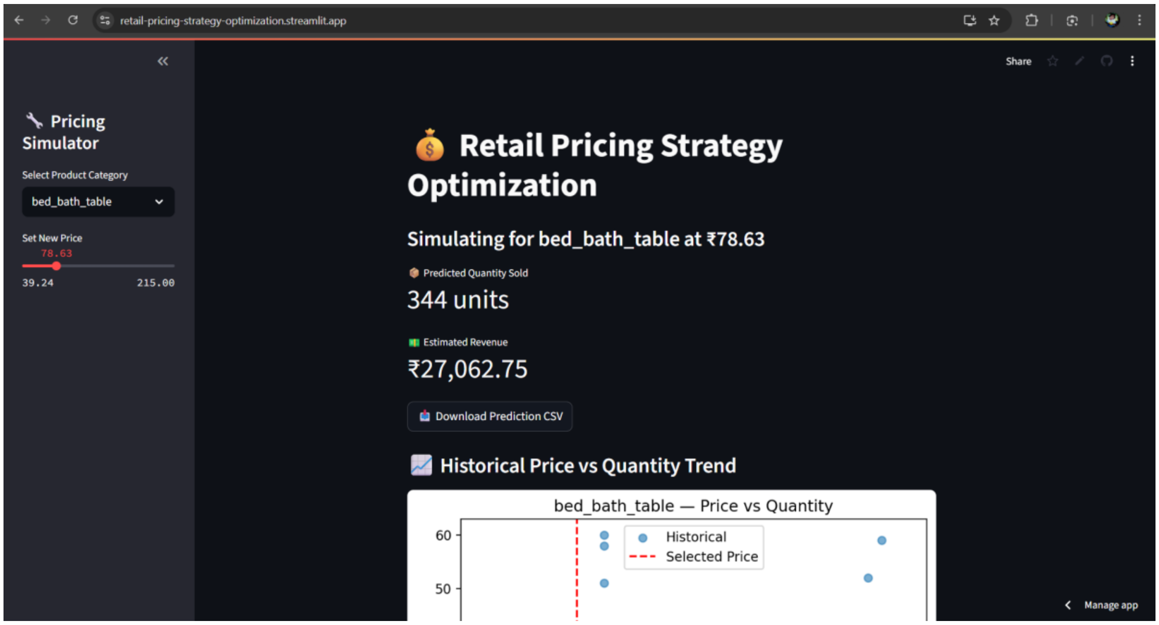
- Python
- pandas, numpy, matplotlib, seaborn
- scikit-learn, joblib
- Streamlit
- Jupyter Notebook, VS Code

## 11. References

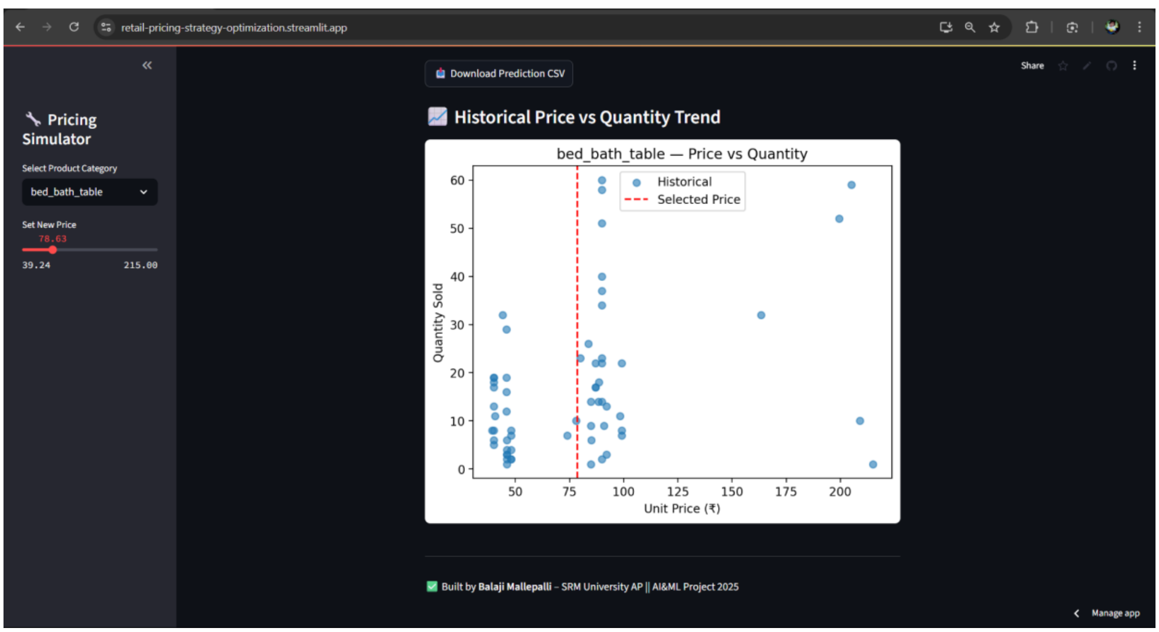

- https://scikit-learn.org
- https://docs.streamlit.io
- Kaggle pricing datasets
- Industry blogs on retail optimization


## 12. Appendix

- Streamlit App Link : https://retail-pricing-strategy-optimization.streamlit.app/




- Price vs Quantity Plot



## 13. Submitted By

Balaji Mallepalli

B.Tech CSE (AI/ML)

SRM University – AP

Project Title: Retail Pricing Strategy Optimization

Submission Date: 5th July 2025