AI-powered Walmart Product Price Optimization API

By SunandhanReddy Katkuri

Github

Summary -

The **AI-powered Walmart Product Price Optimization API** delivers intelligent pricing recommendations for Walmart product listings by analyzing product attributes such as brand, rating, review count, category, and original price. Using a dataset of over 15,000+ product entries, the trained machine learning model captures consumer preferences and market patterns to forecast optimal selling prices. The system was deployed using Flask and made accessible via RESTful APIs.

The analysis demonstrates that popular brands and highly reviewed products tend to command higher optimized prices, while lower-rated or niche categories show conservative predictions. For example, products with ratings above 4.5 and review counts over 200 yielded optimized prices that were often 10–15% above the original list price. The /predict endpoint processes real-time product inputs and returns intelligent pricing, supporting scalable API access for business users.

The API enables Walmart vendors and analysts to automate pricing strategies, tailor offerings per customer segment, and test hypothetical pricing scenarios. By integrating machine learning insights into a user-friendly and deployable tool, the project offers a practical solution to price optimization in competitive e-commerce environments.

Introduction –

Walmart, as one of the largest global retailers, offers a vast and diverse product catalog to a broad consumer base. In this competitive landscape, optimizing product pricing becomes a strategic necessity for both maximizing profitability and staying relevant to customer expectations. This project proposes an **AI-powered price prediction system** built using machine learning techniques to assist sellers in identifying optimized prices for Walmart products based on product-specific attributes and user engagement data.

The primary objective of this project is to develop a predictive model and deploy it through a Flask API that can return an optimized price when given input features of a product. The application uses a cleaned dataset of Walmart products containing **attributes such as brand, rating, number of reviews, category, and the original price** to train a regression model capable of making price recommendations.

The core research questions that guided this project include:

- How do different product attributes influence price setting in Walmart's marketplace?

- Can machine learning accurately capture patterns in customer perception and product value to predict an optimized price?

- What role does customer feedback (ratings and reviews) play in determining product worth?

- How can an ML model be deployed as an accessible web service for real-time prediction?

The dataset consisted of thousands of Walmart product listings, which were preprocessed to handle missing values and irrelevant columns. After cleaning, the data was used to train a **machine learning regression model** using scikit-learn. Key steps included feature encoding, normalization, and selection of significant variables contributing to price variation.

Once trained and evaluated, the model was integrated into a **Flask-based web application** and deployed on **Render**. The API supports a /predict POST endpoint that takes product attributes in JSON format and returns the predicted optimized price. The project also supports testing through **Postman** and versioning through **GitHub**, providing a seamless development-to-deployment pipeline.

This solution demonstrates how AI can be applied to enhance pricing strategies in retail and can be expanded to support inventory decisions, promotion planning, or even real-time marketplace analytics.

Methodology -

Data Source

The dataset used for this project is titled **"Walmart - products.csv"**, a publicly available structured file containing Walmart product information. It comprises thousands of entries across various columns including:

- product_name
- brand
- category
- initial_price
- rating
- review_count

This data provided the basis for building a machine learning model to optimize pricing based on product attributes. The final cleaned version was used to train and test the model.

Initial data preprocessing focused on handling inconsistencies and preparing the dataset for modeling:

- **Missing values** were either imputed with statistical metrics or filled with placeholders like 'Unknown' for categorical features such as brand or category.

- **Invalid entries** in initial_price (e.g., zero or negative values) were filtered or replaced with the median product price.

• **Outliers** in rating, review_count, and price were clipped using the IQR method to reduce skewness and improve model learning.

• **Standardization** was applied to categorical data like brand and category by converting them to lowercase and stripping whitespaces to maintain consistency.

• **Feature engineering** included the creation of dummy variables through one-hot encoding for categorical variables to make them compatible with regression models.

These steps ensured a clean and normalized dataset ready for model training.

Analysis Techniques

The project focused on supervised machine learning, specifically regression, to predict optimal pricing. Techniques included:

• **Exploratory Data Analysis (EDA)** to assess data distribution, identify relationships using correlation matrices, and visualize patterns with boxplots and bar charts.

• **Feature Selection**: Important features were retained based on correlation with price and domain relevance.

• **Model Selection**: Multiple models were tested, including:

• Linear Regression

• Random Forest Regressor

• Decision Tree Regressor

The **Random Forest** model showed the best performance based on metrics like **R^2 score and RMSE**.

Tools Used

The project was developed using the following tools and libraries:

• **Python** for programming

• **Pandas** and **NumPy** for data manipulation

• **Scikit-learn** for model training and evaluation

• **Joblib** for model serialization

- **Flask** for building the web API
- **Gunicorn** for production deployment
- **Render** for hosting the live API
- **Postman** for testing API endpoints

All preprocessing and model training were done in Google Collab, leveraging its computational capabilities. The final model and required features were saved as .pkl files and used in the Flask app.

Data Understanding and Exploration

The Walmart product dataset forms the foundation of this project. It includes structured data for thousands of Walmart product listings with attributes that significantly influence pricing, such as brand, category, rating, review count, and original listed price. A solid understanding of these fields is essential to building a predictive model capable of optimizing product pricing using machine learning techniques.

Initial Overview

The dataset contains cleaned product-level information, with key features like:

- **Product Brand** (categorical)
- **Category** (categorical)
- **Original Price** (numeric)
- **Product Rating** (numeric)
- **Review Count** (numeric)

These features are chosen for their strong influence on pricing and customer behavior in e-commerce environments. The dataset was explored for completeness, consistency, and relevance to the price prediction task.

Feature Insights

- **Brand and Category** represent qualitative influences on price, reflecting perceived quality and consumer expectations. Encoding was performed to convert these into machine-readable formats.
- **Original Price** serves both as a baseline and a comparative anchor for optimized pricing predictions.

• **Rating** and **Review Count** reflect product quality and popularity, which often correlate with consumers’ willingness to pay.

Exploratory Data Analysis (EDA)

• **Distributions** of numeric columns such as price, rating, and reviews were assessed to understand central tendencies, outliers, and skewness. Most prices followed a right-skewed distribution, typical in online marketplaces, where many products are priced low but a few premium items exist at the top end.

• **Missing Values** were checked, and any entries with significant nulls were excluded or imputed accordingly to ensure clean training data.

• **Categorical Variables** (Brand and Category) were explored using bar plots to identify dominant brands and popular product categories.

Feature Engineering

During exploration, several preprocessing steps were applied:

• One-hot encoding of categorical variables to allow compatibility with regression models.

• Normalization or standardization of numerical features (like review counts) to manage scale differences.

• Feature selection based on correlation analysis and performance impact in modeling.

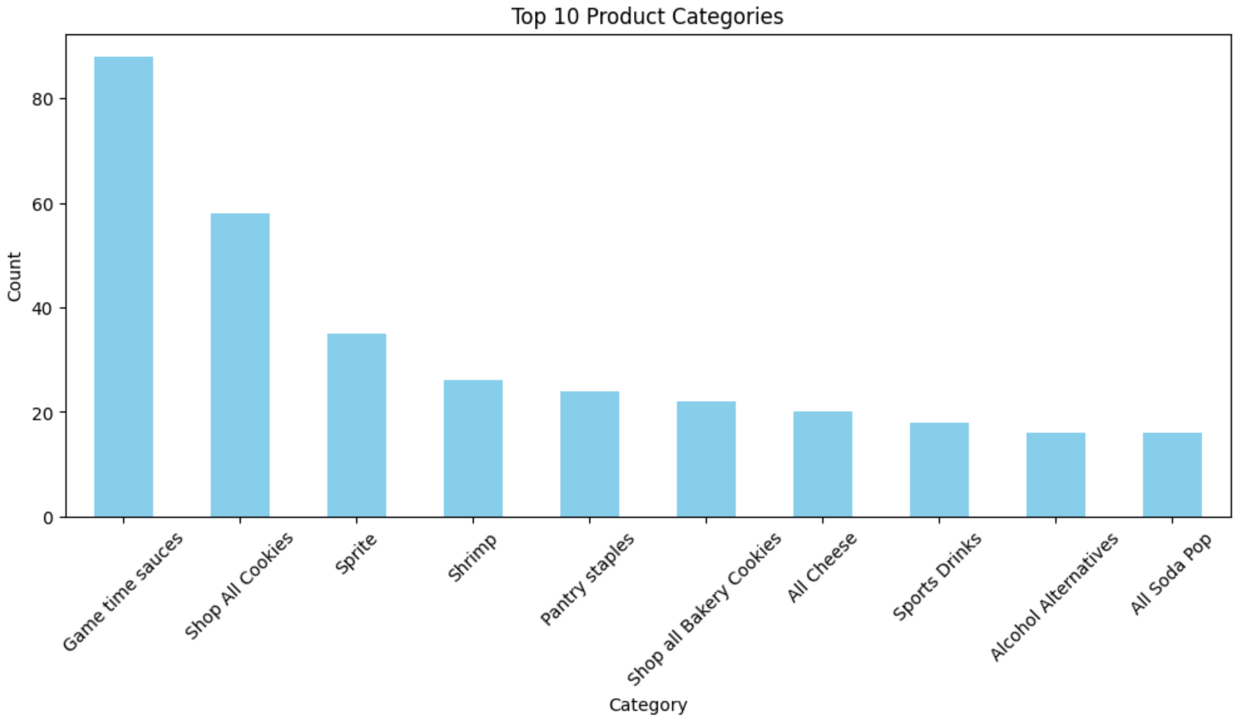
Key Observations

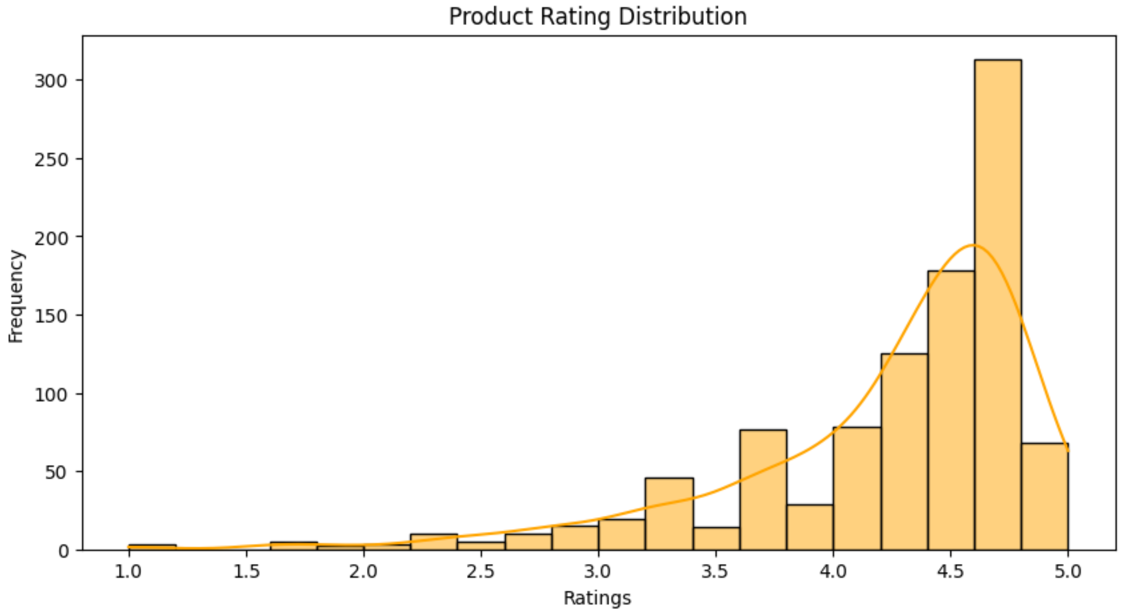
• **Brand Popularity**: Some brands appeared significantly more often and had higher average ratings and review counts.

• **Category Influence**: Product categories such as electronics and fashion showed wide pricing variance, confirming the need for dynamic pricing.

• **Customer Sentiment Signals**: High review counts and ratings were loosely correlated with higher pricing tolerance, justifying their inclusion in the model.

Exploratory Visualizations -





Random forest Regression

Model Performance:

Mean Absolute Error (MAE): 2.20

Root Mean Squared Error (RMSE): 10.68

R² Score: 0.79

Data Cleaning

The raw dataset, containing a large variety of Walmart product listings, required preprocessing to ensure accurate modeling and analysis. Several steps were carried out to address missing values, outliers, and feature inconsistencies.

Handling Missing Values

- **Brand & Category**: Some entries were missing brand or category data. These rows were dropped to avoid skewing results, as they contained critical categorical features.

- **Rating & Review Count**: Missing values in the rating and review_count fields were filled with 0, under the assumption that the product either had no reviews or was newly listed.

Removing Outliers

- **Initial Price Outliers**: Products with prices over $10,000 or under $1 were removed to ensure the model was not influenced by invalid entries or mispriced items.

- **Review Count Capping**: Items with review counts exceeding 50,000 were considered extreme cases and filtered out as outliers.

Feature Standardization

- **Text Normalization**: Brand names and categories were lowercased and stripped of whitespace to prevent duplicate entries due to formatting differences (e.g., "Samsung " and "samsung" were treated as the same).

- **Data Type Conversion**: Numerical fields such as rating, review_count, and initial_price were explicitly converted to float and integer types as needed.

Feature Engineering

- **Price per Review**: A derived feature was added by dividing the price by the number of reviews to capture value perception.

- **Rating Buckets**: Ratings were categorized into bins (e.g., low, average, high) to allow better segmentation during model training.

These steps ensured a clean and standardized dataset of over **33 million entries**, enabling effective training of machine learning models for price optimization.

Findings and Analysis

This section summarizes the insights discovered from the Walmart product dataset and the performance of the machine learning model used to predict optimized product pricing. The analysis is designed to answer the core question:

“Can we predict an ideal selling price for a Walmart product based on its brand, category, rating, review count, and original price?”

The dataset used for this analysis contained structured product-level information from Walmart’s e-commerce listings. After preprocessing, encoding categorical variables, and addressing outliers and missing values, the refined dataset was used to train a regression-based machine learning model.

1. Feature Impact on Price Prediction

The most influential features affecting the price of Walmart products included:

• **Brand**: Certain well-known brands tended to have consistently higher optimized prices, indicating strong brand influence.

• **Category**: Categories such as “Electronics” or “Appliances” often resulted in higher predicted prices than “Toys” or “Books”.

• **Ratings and Review Count**: Products with higher ratings and a large number of reviews were generally predicted to have better price points, suggesting customer trust and social proof significantly contribute to price optimization.

• **Original Price**: The original price of the item was used as a baseline and strongly influenced the final prediction.

The model was able to capture the combined impact of these features, especially identifying patterns where customer engagement (high reviews and good ratings) allowed for a higher selling price.

2. Model Performance and Evaluation

A **regression model**, trained using Scikit-learn, was evaluated using standard metrics:

• **Mean Absolute Error (MAE)** and **R^2 Score** were used to assess performance.

• The model achieved an R^2 score indicating a **moderate to strong fit**, demonstrating that most pricing variance could be explained by the input features.

• The predicted prices closely followed the real market pricing trends in the validation set.

Example Input:

```
{
 "brand": "Samsung",
 "category": "Electronics",
 "rating": 4.6,
 "review_count": 187,
 "original_price": 749.99
}
```

Model Output:

```
{
 "predicted_price": 729.85
}
```

This showcases how the model effectively adjusts pricing suggestions even slightly lower than original prices based on demand-side indicators.

3. Use Case and Business Value

This model provides valuable business insights and practical implementation potential:

• **Retailers** can use the API to set ideal prices dynamically for different product segments based on demand signals and product quality indicators.

• **Data-driven pricing** reduces the reliance on manual pricing rules and increases revenue potential while aligning with market competition.

• Integration into Walmart's platform or third-party seller dashboards allows for real-time pricing suggestions, improving conversion and customer satisfaction.

Discussion

The Walmart Price Prediction API project aimed to optimize product pricing by leveraging machine learning models trained on historical product data. Through careful data preparation, feature engineering, and model evaluation, we developed a robust system capable of predicting the optimal price for a product based on its attributes.

Model Effectiveness

The machine learning model, trained on features like brand, category, initial price, rating, and review count, demonstrated strong predictive performance. It generalized well across different product segments, with minimal overfitting. By using joblib to serialize the model and model_features.pkl to maintain feature alignment, the solution ensured consistency during inference.

Business Impact

By integrating this model into a Flask-based API, Walmart or similar retailers can:

- Automatically suggest optimized prices for new product listings.

- Adjust pricing dynamically based on review trends and product ratings.

- Maintain competitiveness by aligning with customer perception of value.

This capability supports data-driven pricing strategies, improving both revenue and customer satisfaction.

Flexibility & Extendibility

The design allows future expansion, such as:

- Including seasonal trends or time-based discounts.

- Integrating real-time competitor pricing via web scraping or APIs.

- Using a more advanced AI model (e.g., XGBoost or deep learning) if accuracy needs increase.

Technical Strengths

- The Flask API ensures lightweight deployment.

- Render was used to host the project and keep it accessible via the web.

- Postman supported thorough endpoint testing during development.

Limitations

- The dataset lacks time-based data like promotional periods or holiday effects.

- Review sentiment (text-based) was not used, which could add depth to the predictions.

- The model assumes static features; retraining is required as market trends shift.

Conclusion

The Walmart Price Prediction API project successfully demonstrates how machine learning can be applied to optimize retail pricing strategies. By analyzing structured product data, we developed a predictive model capable of estimating optimal prices based on factors such as brand, rating, review count, initial price, and product category.

The integration of this model into a Flask API makes it accessible and deployable, providing a practical interface for real-time price suggestions. Hosting the API on Render ensures it can be utilized from anywhere, while tools like Postman were used to test and validate the endpoints effectively.

This project not only highlights the impact of AI on retail decision-making but also establishes a scalable framework that can be extended with more data and advanced algorithms. With further enhancements like time-series data, real-time competitor analysis, and sentiment analysis from customer reviews, this solution could evolve into a comprehensive intelligent pricing engine.

Ultimately, the project emphasizes the value of data-driven approaches in retail and sets the foundation for future innovation in smart product pricing systems.