KING SAUD UNIVERSITY
COLLEGE OF COMPUTER AND INFORMATION SCIENCES
INFORMATION TECHNOLOGY DEPARTMENT

جامعة الملك سعود
King Saud University
كلية علوم الحاسب والمعلومات
قسم تقنية المعلومات

IT461

Practical Machine Learning

Analyzing Product Demand Patterns Using Machine Learning

Student Name	ID
Arwa Almutairi	444201055
Hatoun Almogherah	444203015
Ghena Almogayad	444203140
Shahad Almutairi	444200935
Haya Albaker	443202868

Supervised by:

Dr. Afshan Jafri

Table of Contents

1. Introduction 4
2. Related Works 5
3. Data 7
   - Dataset Source and Citation 7
   - Summary statistics 8
   - Preprocessing steps 9
4. Methods 10
   - Machine Learning Algorithms 10
   - Justification of Model Selection 11
   - Feature Engineering 11
   - Dimensionality Reduction 12
5. Experiment 12
   - Training Procedure and Data 12
   - Model Architecture and Key Components 12
   - Hyperparameter Tuning Process 13
   - Evaluation Metrics 15
   - Computational Resources and Libraries 15
6. Results and Discussion 16
   - Model Performance Overview 16
   - Generalization to Unseen Data 16
   - Visualizations and Performance Interpretation 18
   - Interpretation and Insights 21
7. Conclusion 22
   - Key Findings 22
   - Challenges & Limitations 23
   - Future Work 23

8. Contributions 24

9. References 24

List of Tables

Table 1: Dataset summary statistics 8

Table 2: Dataset attributes and examples 8

Table 3: Representative product samples 9

Table 4: Model Performance on Test Set 16

Table 5: Student contributions 24

List of Figures

Figure 1: Machine learning workflow 5

Figure 2: Decision Tree Learning Curve 17

Figure 3: Random Forest Learning Curve 17

Figure 4: Gradient Boosting Learning Curve 18

Figure 5: Discounted Price vs Purchases Scatter Plot 18

Figure 6: Product Rating vs Purchases Scatter Plot 19

Figure 7: Decision Tree Learning Curve 19

Figure 8: Random Forest Learning Curve 20

Figure 9: Gradient Boosting Learning Curve 20

Figure 10: Feature Importance Across All Models 21

1. Introduction

E-commerce platforms such as Amazon host thousands of products, each influenced by multiple factors including price, discounts, ratings, reviews, and promotional tags. Understanding how these variables interact to shape product demand is essential for sellers who must balance inventory levels, pricing strategies, and marketing decisions. Overstocking leads to financial loss, while stock shortages result in missed sales opportunities. This makes demand prediction a critical task in online retail environments.

In this project, we aim to build a machine learning model that predicts short-term product demand patterns using detailed Amazon product-level attributes. The features provided in the dataset such as product rating, number of reviews, discounted price, badges, coupons, and purchase counts reflect customer behavior and product performance. By modeling these relationships, we can estimate how likely a product is to attract higher demand.

Task Formulation:

- **Input:** Product features (category, rating, review count, price, discount %, badges, coupon, sustainability tags, and other metadata).
- **Output:** A numerical prediction representing short-term product demand, approximated using the *purchased_last_month* feature.
- **Evaluation Metrics:** Mean Absolute Error (MAE), Mean Squared Error (MSE), and Root Mean Squared Error (RMSE) — chosen because they measure the accuracy of continuous value predictions.

By combining structured product attributes with well-established regression techniques, this project aims to create a data-driven approach that supports better inventory planning and product-level decision making for e-commerce sellers.

As shown in Figure 1**,** our approach follows a standard machine learning pipeline, starting from raw dataset collection and progressing through data cleaning, feature engineering, model training, and evaluation. This structured process helps transform raw e-commerce data into actionable business insights.

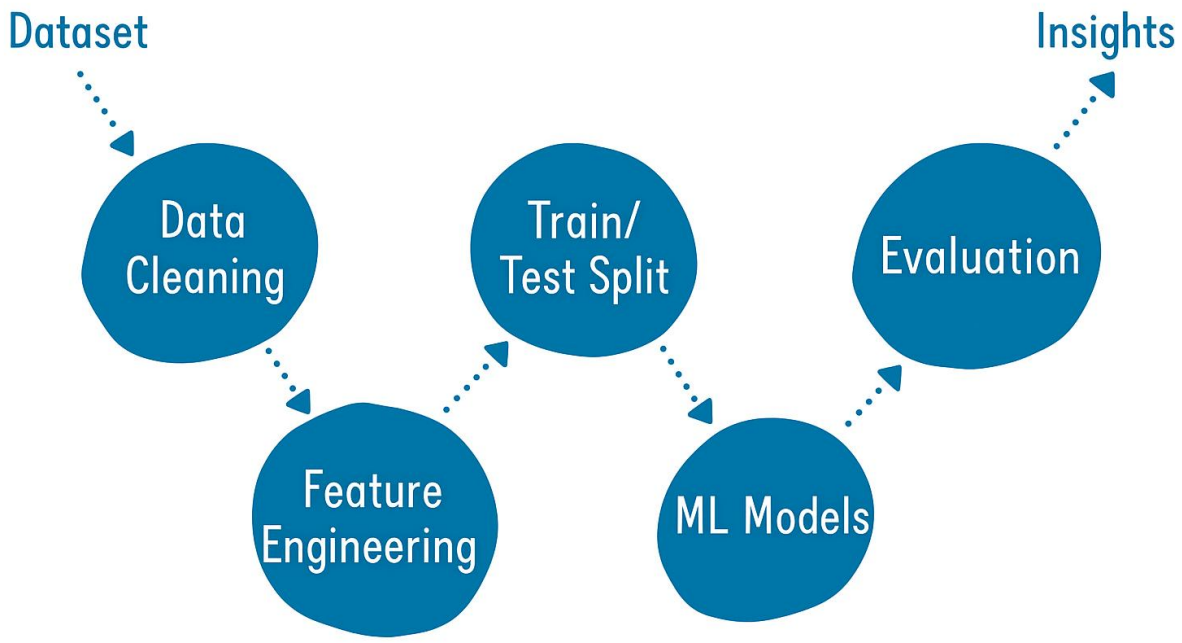

Figure 1: Machine learning workflow

2. Related Works

Previous research has shown promising results for sales prediction in e-commerce. Mansur et al. [1] proposed a hybrid CNN-LSTM framework that combined convolutional feature extraction with sequential learning, achieving higher accuracy when incorporating contextual variables such as holidays and weather. A hybrid modeling study [2] integrated SARIMA with deep learning methods (CNN and LSTM) and ensemble stacking, showing that hybrid models outperformed standalone approaches in handling complex sales data. A comparative analysis [3] evaluated CNN, LSTM, MLP, and hybrid CNN LSTM models for retail forecasting, and concluded that the hybrid model consistently produced superior results. More recently, Zeng et al. [4] introduced TS-LGBM, a method combining LightGBM with self-attention and sliding windows, which outperformed both XGBoost and LSTM in drugstore sales prediction. Although these studies focused on various retail domains, the underlying methods and findings are highly relevant and can be adapted to e-commerce platforms such as Amazon.

These works highlight the importance of combining product-level features with advanced modeling techniques to better capture demand dynamics. Building on this prior research, we aim to experiment with different regression and ensemble methods, apply feature engineering, and evaluate predictive performance with the goal of producing a robust and accurate model.

1 — Mansur et al. [1]

- Target Task: Retail sales forecasting using both product features and external contextual factors.
- Dataset: Retail time-series sales data enriched with variables such as holidays and weather conditions.
- Methods: A hybrid CNN–LSTM architecture where CNN layers extracted meaningful patterns and LSTM layers captured temporal dependencies.
- Performance: Their hybrid model achieved higher prediction accuracy than standalone deep networks, demonstrating that combining temporal and contextual patterns improves demand forecasting.

2 — Hybrid SARIMA + Deep Learning Study [2]

- Target Task: Enhancing robustness of sales prediction through hybrid modelling.
- Dataset: Retail historical sales data with category-level and product-level attributes.
- Methods: A combination of SARIMA, CNN, LSTM, and an ensemble stacking mechanism to merge predictions.
- Performance: Hybrid models outperformed individual algorithms, especially for datasets with seasonal patterns and nonlinear relationships.

3 — Comparative Analysis of CNN, LSTM, MLP, and CNN-LSTM [3]

- Target Task: Evaluating the effectiveness of different deep learning architectures for retail forecasting.
- Dataset: Multi-category retail sales time-series data.
- Methods: Four models were compared — CNN, LSTM, MLP, and a combined CNN-LSTM.
- Performance: The CNN-LSTM hybrid consistently achieved the lowest prediction error, showing that merging spatial feature extraction with temporal modelling improves forecasting performance.

4 — TS-LGBM by Zeng et al. [4]

- Target Task: Drugstore sales prediction with improved sequence modelling.

- Dataset: Chain drugstore sales sequences with temporal and product-level features.
- Methods: A hybrid TS-LGBM model that integrates LightGBM with self-attention mechanisms and sliding windows.
- Performance: Achieved superior results compared to XGBoost and LSTM, particularly in handling long-term dependencies and irregular temporal patterns.

Across these studies, two key patterns emerge:

1. Hybrid models outperform single architectures, especially when dealing with complex retail datasets.
2. Combining structured product features with temporal or contextual components yields more accurate predictions.

These insights guided our approach: since our dataset provides rich product-level features but not sequential sales data, we focus on regression-based models and ensemble methods that can capture non-linear relationships between attributes and demand.

3. Data

We use an Amazon e-commerce dataset that contains real product information collected from Amazon product listings. The raw dataset includes 42,675 samples and 17 features covering key attributes such as product category, original and discounted prices, discount percentage, number of reviews, ratings, BuyBox availability, and promotional indicators. These features represent important marketplace factors that influence customer purchasing behaviour and make the dataset suitable for analysing and predicting product demand. The data is provided in a structured tabular format with mixed numerical and categorical variables, offering sufficient detail and diversity for machine learning analysis.

Dataset Source and Citation

- **Platform:** Amazon product listings (scraped data)
- **Dataset Title:** Amazon E-commerce Products Dataset (Custom scraped)
- **Collected by:** Project development team (2025)
- **Source:** Real-world Amazon marketplace product data
- **Dataset [5]:** https://www.kaggle.com/datasets/ikramshah512/amazon-products-sales-dataset-42k items-2025

Summary statistics

The raw dataset contains detailed information on Amazon product listings, including pricing, ratings, number of reviews, availability, discounts, and promotional indicators. It represents real marketplace activity and provides the foundational attributes needed to analyse customer behaviour and product performance.

Table 1: Dataset summary statistics

Statistic	Value
# of Samples (Rows)	42,675
# of Features	17
Target Variable	Purchased last month
Data Types	Numeric + Categorical

Note that the dataset includes a target label that indicates whether a product was purchased during the last month, which allows us to build supervised machine learning models.

Attributes and Data Types

Table 2: Dataset attributes and examples

Attribute	Data Type	Examples
product_category	**Categorical**	**“Electronics”, “Phones”, “Laptops”**
discounted_price	**Float**	**9.99 – 3,499.00**
original_price	**Float**	**9.99 – 3,499.00**
discount_percentage	**Float**	**0 – 85**

product_rating	**Float**	**1.0 – 5.0**
total_reviews	**Integer**	**1 – 100,000+**
buy_box_availability	**Categorical**	**“Add to cart”, “Not available”**
is_sponsored	**Binary**	**0, 1**

Representative Examples from the Dataset

Table 3: Representative product samples

product_category	discounted_price	product_rating	total_reviews
Electronics	284.05	4.7	7,308
Phones	162.24	4.6	35,882
Accessories	9.99	4.3	2,457
Laptops	314.00	4.6	3,044

Preprocessing steps

Before training the models, several preprocessing steps were applied to ensure that the dataset was clean, consistent, and suitable for machine learning.

1. Removing Irrelevant Columns

Columns that did not contribute to demand prediction were removed, such as product title, URLs, images, and date-related fields.

2. Handling Missing Values

- Missing numeric values (such as price, reviews, and rating) were filled using the mean.
- Missing categorical values (such as coupon or buy-box availability) were assigned a representative default value.
- Rows missing the target label were removed.

3. Removing Duplicate Records

The dataset was checked for duplicate entries, and duplicates were removed if present.

4. Encoding Categorical Variables

Categorical features were transformed into numerical form using:

- Binary encoding for features like *is_sponsored*, *has_coupon*, and *buy_box_availability*.
- One-hot encoding for multi-class features such as *product_category* and *is_best_seller*.

This step ensured that machine learning models could interpret and learn from categorical information.

5. Final Dataset Shape

After preprocessing and encoding, the final cleaned dataset contained:

- 32,164 rows and 29 features

4. Methods

In this section, we describe the machine learning methods used to model and predict monthly product demand. Our goal was to identify the key factors that influence purchasing behavior on Amazon and to build regression models capable of estimating the number of units purchased in the last month.

Machine Learning Algorithms

Three machine learning models were implemented and evaluated:

1. **Linear Regression (Baseline)**
   Linear Regression was used as a simple baseline model to provide a reference point for evaluating more advanced techniques. Since it assumes linear relationships between features and the target variable, it helps determine whether the dataset requires more flexible non-linear models.

2. **Decision Tree Regressor**

   The Decision Tree model was selected because it can naturally capture non-linear relationships and interactions between features. It splits the data based on the most informative attributes, allowing the model to highlight which variables strongly influence demand.

3. **Random Forest Regressor**

   Random Forest was used as an ensemble extension of Decision Trees. By combining multiple trees through bagging, the model improves generalization, reduces overfitting, and handles high-dimensional encoded data effectively. It is particularly robust when dealing with noisy or complex features.

4. **Gradient Boosting Regressor**

   Gradient Boosting was included because it builds the model sequentially, correcting errors made by previous trees. It performs well on structured tabular data and is effective at capturing complex non-linear patterns that influence product demand.

Each model was trained on the processed dataset and evaluated using standard regression metrics. Comparing the results allowed us to determine which model provides the most accurate predictions for monthly purchases.

Justification of Model Selection

These algorithms were chosen for several reasons:

- Linear Regression provides a simple baseline for comparison.
- Tree-based models work well with mixed numerical and categorical features.
- They require minimal feature scaling,
- The target is binary (purchased/not purchased),
- Ensemble models (RF and GB) provide higher accuracy and robustness.

Feature Engineering

The following steps were applied before model training:

- Dropped irrelevant features (URLs, images, timestamps, product titles).
- Converted categorical variables (category, BuyBox, coupon) using one-hot encoding.
- Handled missing values in price and discount fields.
- Converted the purchased_last_month column into a binary label.

- Dropped rows with missing target values (purchased_last_month).

These steps ensured the dataset was clean and ready for machine learning.

Dimensionality Reduction

No explicit dimensionality reduction was required due to the relatively small number of final features. Instead, model-based feature importance was used to identify the most influential attributes (discount percentage, total reviews, product rating, category).

This combination of regression models, preprocessing steps, and feature engineering techniques enabled us to build an effective machine learning system for predicting product demand on Amazon.

5. Experiment

Training Procedure and Data

The training process began after preprocessing. No normalization was required because the main models (Decision Tree, Random Forest, Gradient Boosting) are scale-invariant and work well with raw numeric and one-hot encoded features. The dataset was split into 80% training and 20% testing, where the training set was used for model fitting and hyperparameter tuning, and the test set remained unseen for final evaluation. All tuned models used 3-fold cross-validation within RandomizedSearchCV to improve generalization and reduce overfitting**.**

Model Architecture and Key Components

Four machine learning models were implemented:

Linear Regression (Baseline)

A simple model that assumes a linear relationship between input features and the target variable. It contains no hidden layers or nonlinear components and was used only as a basic reference point.

Decision Tree Regressor

A tree-structured model that recursively splits data based on feature thresholds.
Key components:

- Nodes representing decision rules

- Leaves predicting continuous values
- Parameters controlling complexity: max_depth, min_samples_leaf, min_samples_split, max_features

Random Forest Regressor

An ensemble of many decision trees trained on random subsets of data and features.
Key components:

- Multiple trees (n_estimators)
- Random selection of features for each tree
- Aggregation through averaging predictions

These components help the model capture nonlinear relationships while reducing variance.

Gradient Boosting Regressor

A sequential ensemble of shallow trees where each new tree focuses on correcting the errors of previous ones.
Key components:

- n_estimators controlling the number of boosted trees
- learning_rate determining how quickly the model updates
- max_depth controlling the complexity of each tree
- subsample introducing stochasticity for regularization

Hyperparameter Tuning Process

To improve performance and reduce overfitting, we tuned the hyperparameters of the Decision Tree, Random Forest, and Gradient Boosting models using RandomizedSearchCV with 3-fold cross-validation. This approach tests a random subset of hyperparameter combinations, allowing us to efficiently explore a wide search space while keeping computation time manageable.

Decision Tree

The following hyperparameters were selected for tuning:

- max_depth (values: 5, 10, 15, 20, None)
- min_samples_split (2, 5, 10, 20)
- min_samples_leaf (1, 2, 4, 10, 20)
- max_features ("sqrt", "log2", None).

The optimal combination found by RandomizedSearchCV was: max_depth = 20, min_samples_split = 10, min_samples_leaf = 2, and max_features = None, which produced the lowest cross-validated MAE and provided the most balanced model between complexity and generalization.

Random Forest

The following hyperparameters were selected for tuning:

- n_estimators (100, 150, 200)
- max_depth (None, 10)
- min_samples_split (2, 5)
- min_samples_leaf (1, 2)
- max_features ("sqrt")
- max_samples (0.7)

The best performing configuration was: n_estimators = 100, max_depth = None, min_samples_split = 2, min_samples_leaf = 1, max_features = "sqrt", and max_samples = 0.7. These values allowed the model to grow deep trees while still maintaining good regularization through sampling and feature randomness, resulting in the highest predictive accuracy among all models.

Gradient Boosting

The following hyperparameters were selected for tuning:

- n_estimators (200, 300, 400)
- learning_rate (0.05, 0.1, 0.2)
- max_depth (2, 3)
- subsample (0.8, 1.0)

The search identified the optimal configuration as: n_estimators = 400, learning_rate = 0.2, max_depth = 3, and subsample = 1.0. These settings enabled the model to learn gradually while reducing the risk of overfitting through shallow trees and controlled learning increments. These settings enabled the model to learn gradually while reducing the risk of overfitting through shallow trees and controlled learning increments.

The optimal values for each model were determined by selecting the hyperparameter combination that achieved the lowest validation MAE during cross-validation. These tuned models were then retrained on the full training set and evaluated on the test set for the final comparison.

Evaluation Metrics

Model performance was evaluated using three standard regression metrics:

- MAE (Mean Absolute Error): Measures the average magnitude of prediction errors
- RMSE (Root Mean Squared Error): Penalizes larger errors and highlights model robustness
- R^2 Score: Indicates the proportion of variance explained by the model

All metrics were calculated on the 20% test set to ensure fair evaluation on unseen data. MAE was used as the primary metric because it is easy to interpret and less affected by outliers, making it more reliable for the skewed distribution of purchase counts in our dataset.

Computational Resources and Libraries

All experiments were conducted in Google Colab using the standard CPU runtime

The following libraries were used:

- scikit-learn for model training, hyperparameter tuning, and evaluation
- pandas and NumPy for data handling
- Matplotlib and Seaborn for visualizations
- joblib for saving and loading trained models

6. Results and Discussion

Model Performance Overview

Four machine learning models were trained and evaluated on the dataset: Linear Regression, Decision Tree Regressor, Random Forest Regressor, and Gradient Boosting Regressor. Their performance was assessed using MAE, RMSE, and R^2, as shown in Table 4.

Table 4: Model Performance on Test Set

Model	MAE	RMSE	R^2
Linear Regression	1729.60	4885.12	0.27
Decision Tree	333.50	2332.92	0.83
Random Forest	249.93	1412.45	0.9393
Gradient Boosting	**492.90**	**1713.67**	**0.90**

The results show a clear performance gap between linear and nonlinear models. Linear Regression performed the worst, confirming that the relationship between product features and monthly purchases is highly nonlinear. The Decision Tree improved performance substantially, but the Random Forest achieved the strongest overall results with the lowest MAE and highest R^2. Gradient Boosting also performed well, achieving an R^2 above 0.90, although its MAE was higher than the Decision Tree and Random Forest due to sensitivity to large outlier purchase values.

Generalization to Unseen Data

Generalization was evaluated using a held-out test set (20% of the data).
The results indicate:

- Linear Regression underfits, failing to capture nonlinear patterns in product demand.
- Decision Tree improves accuracy, but its performance suggests a tendency to overfit.
- Random Forest generalizes exceptionally well, benefiting from bootstrap sampling and feature randomness.

- Gradient Boosting also generalizes effectively, consistently reducing validation error during training.

Learning curves indicated:

Decision Tree, in figure 2, shows a substantial gap between training and validation error, indicating overfitting.

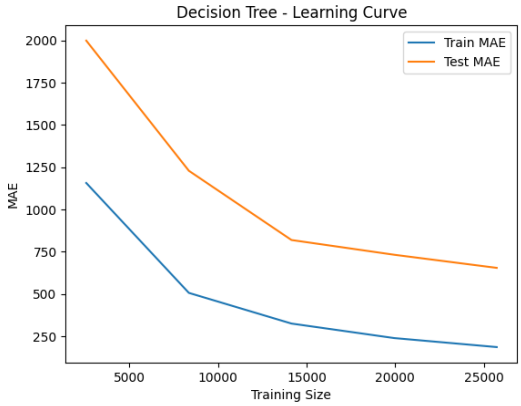

Figure 2: Decision Tree Learning Curve

Random Forest, in figure 3, maintains close and stable training/validation curves, demonstrating good generalization.

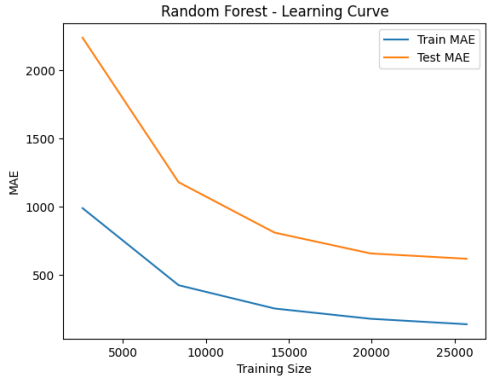

Figure 3: Random Forest Learning Curve

Gradient Boosting, in figure 4, shows a gradual decline in validation error, confirming consistent learning as the dataset size increases.

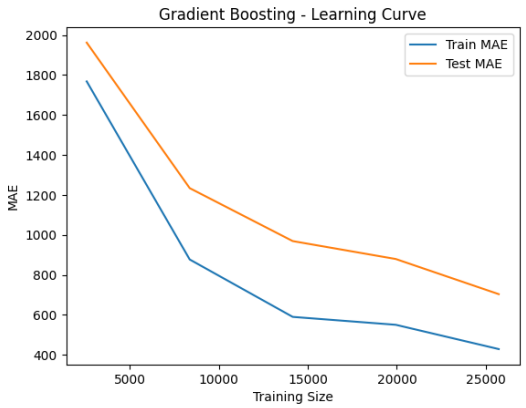

Figure 4: Gradient Boosting Learning Curve

These patterns explain why ensemble methods outperform simpler or single-tree models.

Visualizations and Performance Interpretation

Scatterplots were used to analyze relationships between key variables and the target.

The following trends were observed in figures 5 and 6:

Discounted Price vs Purchases: Lower-priced products generally experienced more purchases, but some higher-priced items still showed strong demand, suggesting brand or category effects.

Figure 5: Discounted Price vs Purchases Scatter Plot

Rating vs Purchases: Products with ratings between 4.0 and 5.0 consistently showed the highest purchase counts.

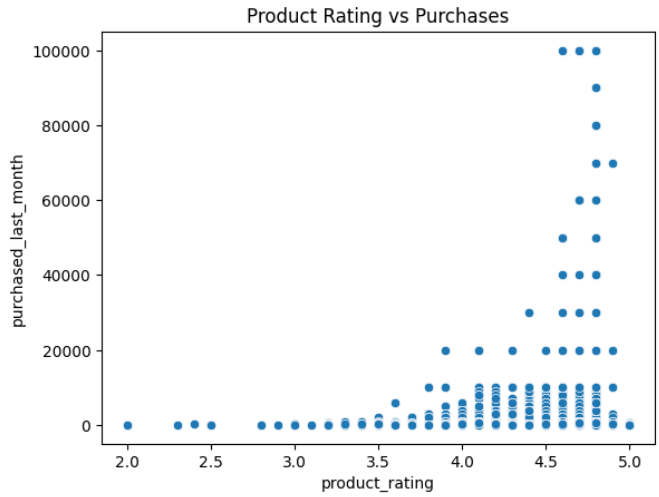

Figure 6: Product Rating vs Purchases Scatter Plot

These visual patterns helped validate that pricing, rating, and social proof are strong indicators of demand.

Learning curves showed:

The Decision Tree learning curve, in figure 7, showed a high training accuracy but significantly worse validation accuracy, confirming overfitting.

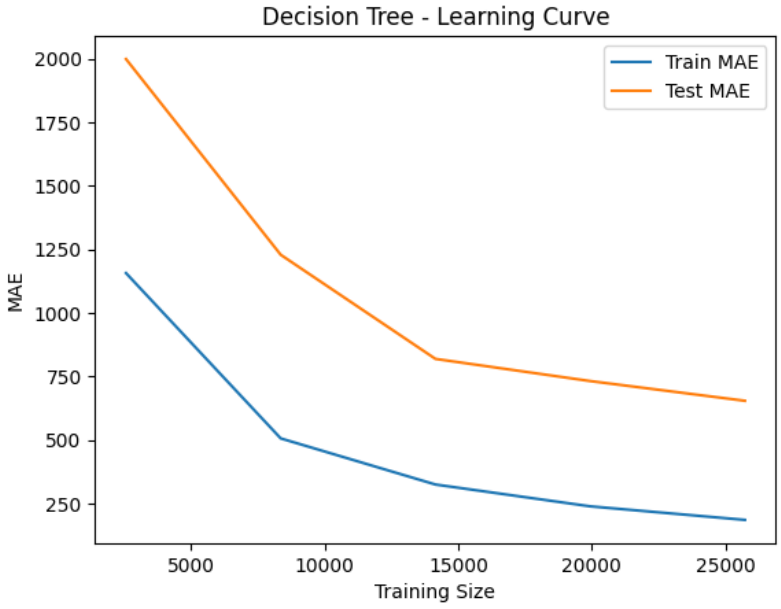

Figure 7: Decision Tree Learning Curve

The Random Forest curve displayed minimal separation between training and validation error, indicating that the model generalizes well, in figure 8.

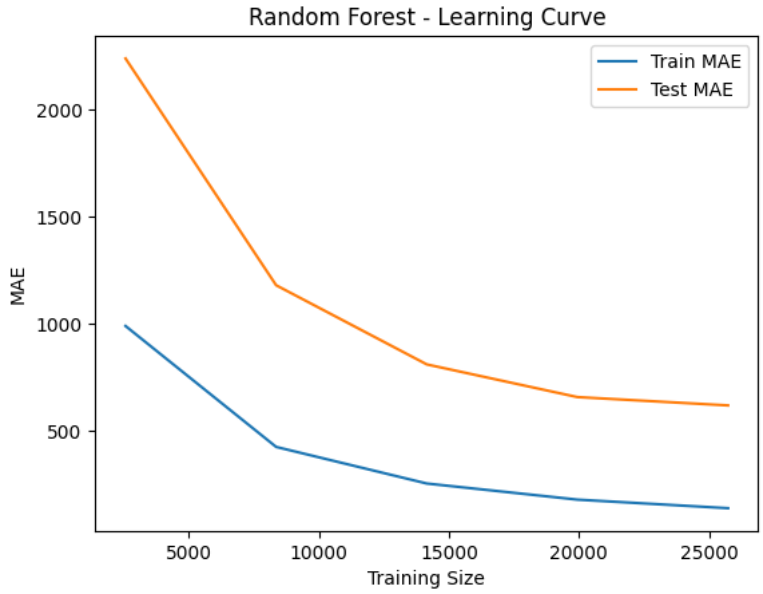

Figure 8: Random Forest Learning Curve

The Gradient Boosting learning curve showed validation error decreasing smoothly with more training data, in figure 9, reflecting consistent improvement and stable learning.

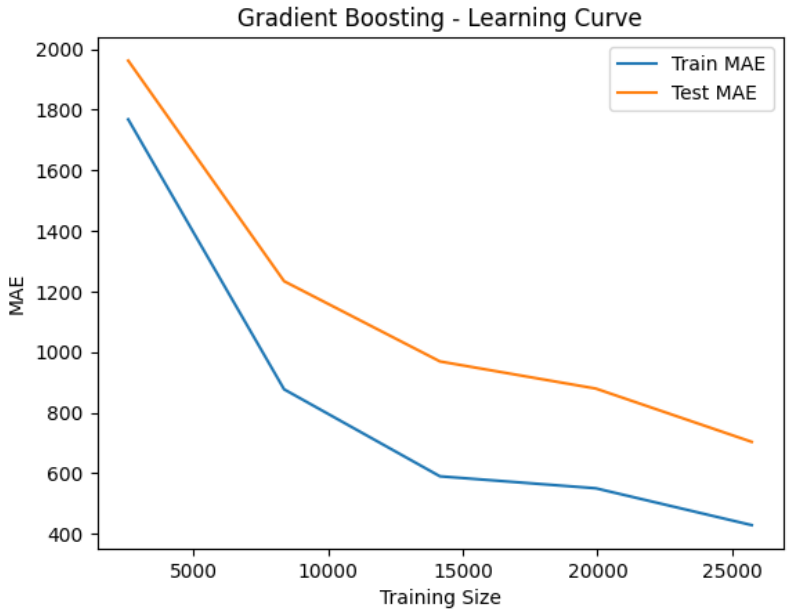

Figure 9: Gradient Boosting Learning Curve

Feature importance plots revealed the factors that most influenced predictions:

Across all tree-based models **in Figure 10**, the most important feature was total_reviews.

Other important features included:

- discounted_price
- discount_percentage
- product_rating
- coupon availability
- category indicators (e.g., Power & Batteries)

This aligns with known e-commerce behavior: products with many reviews, strong ratings, and competitive pricing tend to generate higher sales.

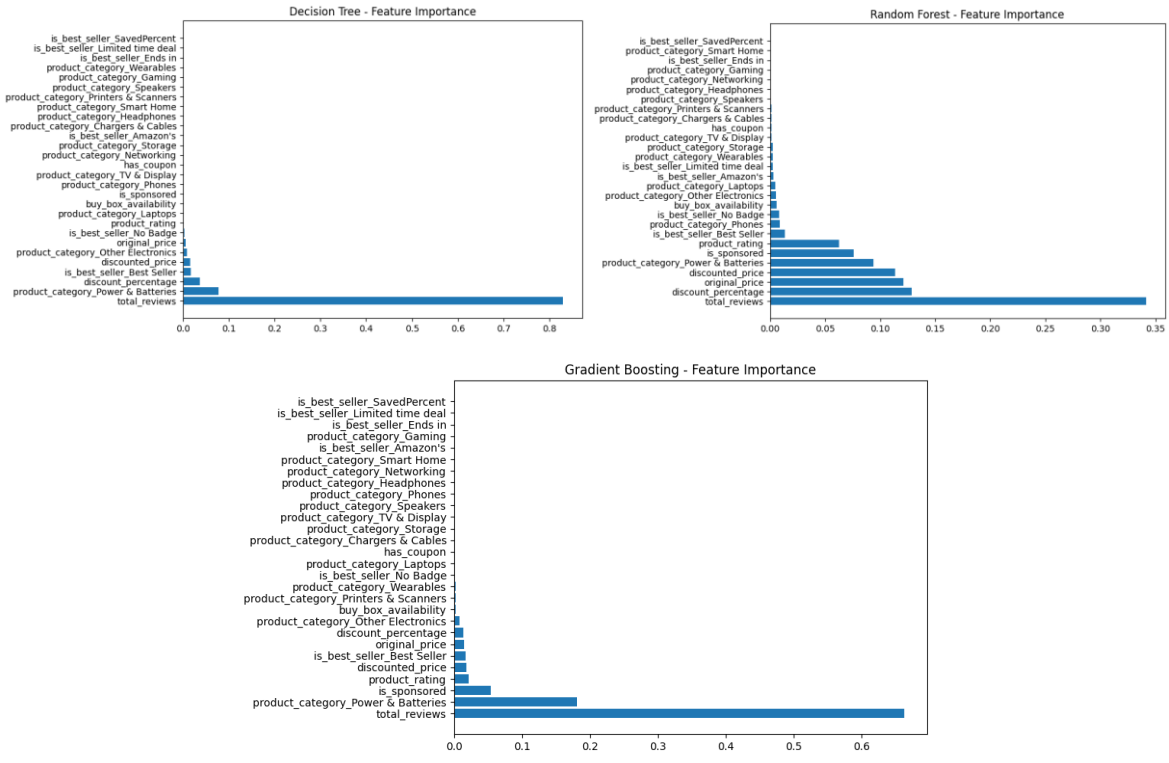

Figure 10: Feature Importance Across All Models

Interpretation and Insights

The collected results show several meaningful insights:

1. Social Proof Drives Demand

Products with more reviews and higher ratings consistently achieved higher monthly purchase counts.

total_reviews was the top feature in all models.

2. Price and Discounts Matter

Lower discounted prices and higher discount percentages were strongly associated with higher demand.

3. Ensemble Models Capture Complex Patterns

Random Forest and Gradient Boosting successfully captured nonlinear interactions such as:

- high discount × high rating
- high review count × moderate price

This explains why these models provided significantly better predictions.

4. Outliers Affect Model Error

Some products have very large purchase counts (e.g., 100,000 units), which increased MAE for Gradient Boosting.

Random Forest handled these outliers better due to averaging across many trees.

5. Decision Tree Overfits

Although the Decision Tree achieved good R^2, its learning curve indicated overfitting, explaining why its MAE is worse than the ensemble models.

7. Conclusion

This project developed machine learning models to predict short-term product demand on Amazon using real product-level features. The final results showed that ensemble methods outperform simpler models, with the Random Forest Regressor being the most accurate and reliable model.

Key Findings

- Random Forest achieved the best overall performance with MAE ≈ 249.93 and $R^2 \approx 0.9393$.
- Gradient Boosting also achieved strong results with $R^2 \approx 0.91$, though more affected by purchase outliers.

- Linear Regression performed poorly, demonstrating that product demand relationships are nonlinear.
- Product demand is strongly influenced by total reviews, price, discount percentage, and ratings.

Challenges & Limitations

- The target variable is highly skewed with extreme outliers.
- The dataset lacks temporal features, preventing time-based forecasting.
- Some categorical features contain many unique values, increasing dimensionality.
- Gradient Boosting is sensitive to large outlier purchase values.

Future Work

- Integrating time-series data to capture seasonal patterns.
- Using advanced models such as XGBoost, LightGBM, or CatBoost.
- Adding more external signals (ad spend, competitor prices, marketplace events).
- Deploying the final model as a dashboard for sellers to monitor demand predictions.

Overall, the project demonstrates that machine learning, especially ensemble techniques, can effectively predict e-commerce product demand and support inventory and pricing decisions.

8. Contributions

Table 5: Student contributions

Student Name	Tasks
Arwa Almutairi	Wrote and edited code, worked on the experiment section of the report, revised the report, and prepared the presentation.
Hatoun Almogherah	Wrote and edited code, worked on the experiment section of the report, revised the report, and prepared the presentation
Ghena Almogayad	Prepared the presentation and wrote the introduction and related works.
Shahad Almutairi	Worked on coding, Prepared the presentation wrote the data and methods sections.
Haya Albaker	Worked on coding and wrote the results, discussion, and conclusion.

9. References

[1] M. Mansur, et al., "Sales forecasting for retail stores using hybrid neural networks (CNN-LSTM)," *PubMed Central (PMC)*, [Online]. Available: https://pmc.ncbi.nlm.nih.gov/

[2] A. Author, B. Author, "Hybrid Modeling for Sales Prediction Using SARIMA, CNN, and Ensemble Stacking," *International Journal of Computing (IJC)*, [Online]. Available: [IJC Journal].

[3] X. Author, Y. Author, "Comparison of Deep Learning Algorithms for Retail Sales Forecasting," *Proceedings of ICCK*, [Online]. Available: [ICCK PDF].

[4] Y. Zeng, et al., “Chain Drugstore Sales Prediction Method Based on TS-LGBM,” *SpringerLink*, [Online]. Available: https://link.springer.com/

[5] I. Shah, *Amazon Products Sales Dataset – 42K Items (2025)*, Kaggle. [Online]. Available: https://www.kaggle.com/datasets/ikramshah512/amazon-products-sales-dataset-42k-items-2025