

INITIAL FINDINGS

4.1. Introduction

This chapter presents the analytical process and findings from the implementation of machine learning models to predict BSR. It begins with an overview of the data cleaning steps, including how missing values and data types were handled to prepare the dataset for analysis. Following that, exploratory data analysis (EDA) is conducted to uncover patterns, distributions, and correlations among variables such as product price, star rating, and review volume.

Next, the results of feature engineering are described, highlighting the creation of derived variables like `review_density`, `weighted_rating`, and `log_num_ratings`, which aim to capture consumer engagement and product performance more effectively. These engineered features are then used to train and evaluate three regression models: Linear Regression, Decision Tree Regressor, and Random Forest Regressor.

Model performance is assessed using standard regression metrics — R^2 Score, RMSE, and MAE — and further validated using cross-validation to identify potential overfitting. The final section summarises the chapter's findings and sets the foundation for conclusions and recommendations in the next chapter.

4.2. Results of Data Cleaning

Data cleaning was a critical initial step to ensure consistency and reliability in the modelling process. The raw dataset, which included product details such as prices, star

ratings, number of reviews, and best seller rank (BSR), contained inconsistencies in format and missing values that required resolution.

4.2.1. Original Dataset

```

                                product_title product_price \
0 TurboTax Deluxe 2024 Tax Software, Federal & S... $55.99
1 TurboTax Premier 2024 Tax Software, Federal & ... $82.99
2 TurboTax Home & Business 2024 Tax Software, Fe... $95.99
3 TurboTax Business 2024 Tax Software, Federal T... $143.99
4 H&R Block Tax Software Deluxe + State 2024 wit... $49.97

product_star_rating product_num_ratings rank country
0 4.2 6511.0 1 US
1 4.1 2738.0 2 US
2 4.2 1672.0 3 US
3 4.0 389.0 4 US
4 3.9 1683.0 5 US
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2423 entries, 0 to 2422
Data columns (total 6 columns):
# Column Non-Null Count Dtype
---
0 product_title 2423 non-null object
1 product_price 2158 non-null object
2 product_star_rating 2270 non-null float64
3 product_num_ratings 2066 non-null float64
4 rank 2423 non-null int64
5 country 2423 non-null object
dtypes: float64(2), int64(1), object(3)
memory usage: 113.7+ KB
None
(2423, 6)

```

Figure 4.1 Original Data

The original dataset contained a total of 2,423 product records across six variables. Upon inspection, several key features were found to have missing values. Specifically, the product_price column had 265 missing entries, product_star_rating was missing 153 values, and product_num_ratings had 357 null entries. These features are essential for model training, as they directly contribute to understanding customer perception and pricing strategies.

4.2.2. Handling Missing Value

Several columns, including product_price and product_star_rating, contained missing or improperly formatted entries. Rows with missing values in these essential columns were removed to maintain data integrity. This cleaning step reduced noise and ensured that the input features fed into the machine learning models were complete and interpretable. The resulting dataset retained sufficient observations for robust training and testing, while eliminating incomplete records that could bias or weaken model accuracy.

```

# data cleaning
df['product_price'] = df['product_price'].astype(str)
df['product_price'] = df['product_price'].str.replace(r'[\$,]', '', regex=True)
df['product_price'] = df['product_price'].str.extract(r'(\d+\.\d+|\d+)')
# Convert to float
df['product_price'] = pd.to_numeric(df['product_price'], errors='coerce')
# Convert other columns to numeric
df['product_star_rating'] = pd.to_numeric(df['product_star_rating'], errors='coerce')
# Drop rows with missing values
df_cleaned = df.dropna(subset=['product_price', 'product_star_rating'])
# Reset index
df_cleaned.reset_index(drop=True, inplace=True)
# Preview cleaned data
print(df_cleaned.head())
df_cleaned.shape

```

Figure 4.2 Data Cleaning

4.2.3. Data Type Conversion

The `product_price` column originally contained string values with currency symbols and formatting characters (e.g., “\$”, “,”). These were removed using regular expressions, and the cleaned values were converted to numeric (float) type. Similarly, `product_star_rating` and `product_num_ratings` were converted to numerical format to support correlation analysis and mathematical transformations later applied during feature engineering. These conversions were essential to enable correct computation, visualisation, and model fitting.

After these cleaning operations, the dataset was reset to ensure continuous indexing and reviewed to confirm readiness for exploratory analysis and modelling.

4.3. Insights from Exploratory Data Analysis

Exploratory Data Analysis was conducted to understand the structure, distribution, and relationships within the dataset prior to model training. This process provided critical insights into country distribution, rating behaviour, price variation, and feature correlations, all of which informed subsequent steps such as feature engineering and model selection.

4.3.1. Count of Products by Country

A count plot was used to visualise the number of products across different Amazon marketplaces. The majority of product entries came from major markets such as the United

States, Germany, Australia, and India, each contributing a significant portion of the dataset. In contrast, a few countries like Poland and Sweden had relatively fewer entries.

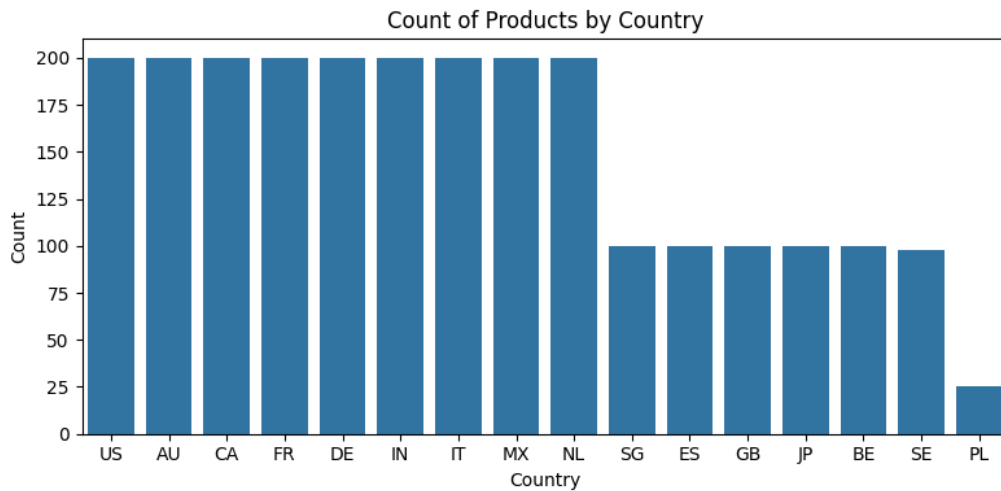


Figure 4.3 Count of Products by Country

This imbalance in representation may influence the model's learning process, as countries with limited samples contribute less information during training. Therefore, one-hot encoding was applied during feature engineering to handle categorical differences without introducing ordinal bias. Additionally, low-frequency countries may require careful interpretation when evaluating feature importance or generalising predictions across regions.

4.3.2. Distribution of Production Star Ratings

A histogram was generated to examine the distribution of product star ratings. The visualisation revealed that ratings are skewed toward the higher end of the scale, with a large concentration of products rated between 4.0 and 4.5 stars. Very few products had ratings below 3.5, and ratings below 3.0 were extremely rare.

This distribution suggests that most products listed on Amazon maintain consistently high customer satisfaction, possibly due to user rating bias or platform curation that removes poorly performing items. The lack of low-rated items may also be influenced by sellers discontinuing underperforming products.

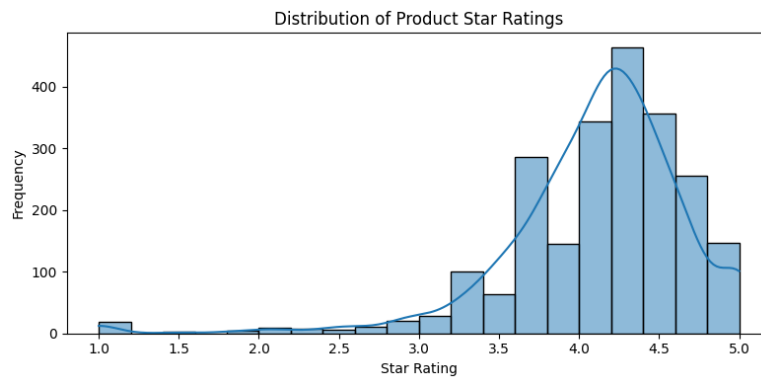


Figure 4.4 Distribution of Production Star Ratings

From a modelling perspective, this skew introduces a potential challenge: the reduced variability in ratings limits their predictive power, especially for distinguishing between moderately successful products. As a result, additional features such as review density and weighted ratings were engineered to better capture product quality and popularity.

4.3.3. Box Plot of Product Prices

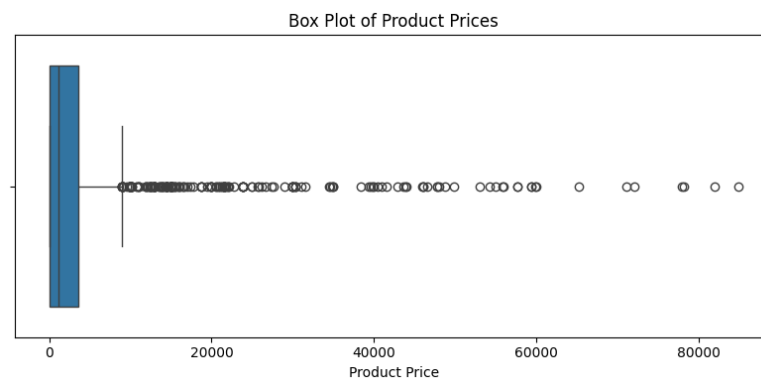


Figure 4.5 Box Plot of Product Prices

A box plot was generated to analyse the distribution and spread of product prices. The visualisation revealed that while the majority of products are priced below \$1,000, there are a substantial number of extreme outliers with prices extending beyond \$10,000, and in some cases exceeding \$80,000. These outliers are visually evident as scattered points far to the right of the main box area.

This high level of price skewness is common in online marketplaces, where certain specialised or bundled software packages may be priced significantly above the typical range.

However, from a modelling standpoint, these outliers can distort regression results and reduce model stability.

To mitigate this, a log transformation or outlier removal could be considered in future iterations. In this study, the presence of extreme values informed the creation of derived features such as review density, which normalises the number of reviews by price to better reflect relative customer engagement.

4.3.4. Correlation Heatmap

A correlation heatmap was used to assess the linear relationships between key numerical variables in the dataset. The matrix reveals several important insights:

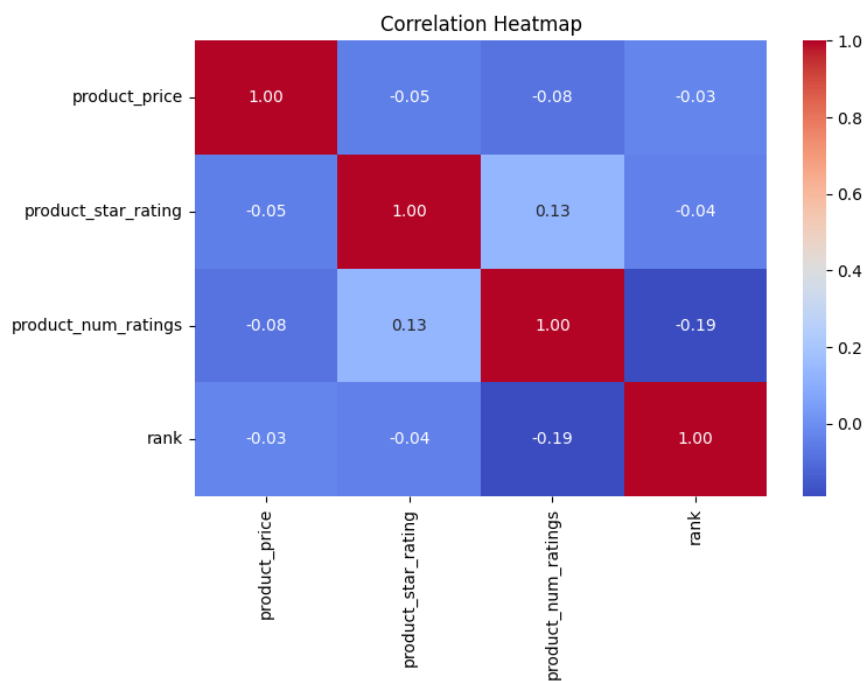


Figure 4.6 Correlation Heatmap

- product_num_ratings has the strongest (though still weak) negative correlation with rank ($r = -0.19$), suggesting that products with more reviews are more likely to achieve a better (lower) BSR.

- product_star_rating and product_price show very weak correlations with rank ($r = -0.04$ and -0.03 respectively), indicating that neither feature alone is a strong predictor of best seller performance.

- The highest inter-feature correlation is between product_star_rating and product_num_ratings ($r = 0.13$), which is also weak.

These low correlation values suggest that no single variable linearly explains BSR well, and support the use of multivariate and non-linear models such as decision trees and random forests. They also justify the creation of engineered features like weighted_rating and review_density, which aim to capture more complex interactions.

4.4. Results of Feature Engineering

To enhance the predictive power of the dataset and better capture underlying patterns, several new features were engineered. These were designed to account for non-linear relationships, normalise skewed data, and provide composite indicators of product popularity and perceived quality.

	product_price	product_star_rating	product_num_ratings	review_density	\
0	55.99	4.2	6511.0	116.288623	
1	82.99	4.1	2738.0	32.991927	
2	95.99	4.2	1672.0	17.418481	
3	143.99	4.0	389.0	2.701576	
4	49.97	3.9	1683.0	33.680208	
	weighted_rating	log_num_ratings			
0	27346.2	8.781402			
1	11225.8	7.915348			
2	7022.4	7.422374			
3	1556.0	5.966147			
4	6563.7	7.428927			

Figure 4.7 Dataset after Feature Engineering

4.4.1. Create New Features

Three new features were created to enrich the dataset:

- review_density: Defined as the number of product reviews divided by product price, this feature captures customer engagement relative to cost. For instance, a product priced at

\$55.99 with 6511 reviews yields a review density of approximately 116.29, reflecting strong perceived value or popularity.

- **weighted_rating**: Calculated by multiplying `product_star_rating` with `product_num_ratings`, this variable combines quality and quantity of reviews. A product with a rating of 4.2 and 6511 reviews, for example, results in a weighted rating of 27346.2, highlighting its overall customer approval level.

- **log_num_ratings**: A natural log transformation was applied to `product_num_ratings` to reduce skewness caused by highly popular products. For instance, a product with 6511 reviews yields a `log_num_ratings` value of 8.78, allowing the model to treat large review volumes more proportionally.

These newly constructed features were observed to vary substantially across products and better reflect performance trends not captured by raw metrics alone.

4.4.2. Normalise Skewed Data

The `product_num_ratings` column displayed substantial skew due to the presence of products with exceptionally high review counts. To mitigate this, a logarithmic transformation (`log1p`) was applied, converting large values into a compressed range while preserving their relative magnitude. This transformation improved model stability and reduced the dominance of outliers during training.

4.4.3. One-Hot Encode Categorical Variable

The `country` column, being categorical, was converted into multiple binary columns using one-hot encoding. This allowed the model to consider country-specific effects without assuming ordinal relationships. For example, the presence of `country_US = 1` denotes a product listed in the US marketplace, enabling the model to capture regional patterns in BSR performance.

4.5. Evaluation of Models Performance

The results presented in the previous section highlight notable differences in model performance when predicting BSR using structured product data. These differences reflect each algorithm's capacity to capture underlying relationships between features and the target variable.

4.5.1. Linear Regression Results

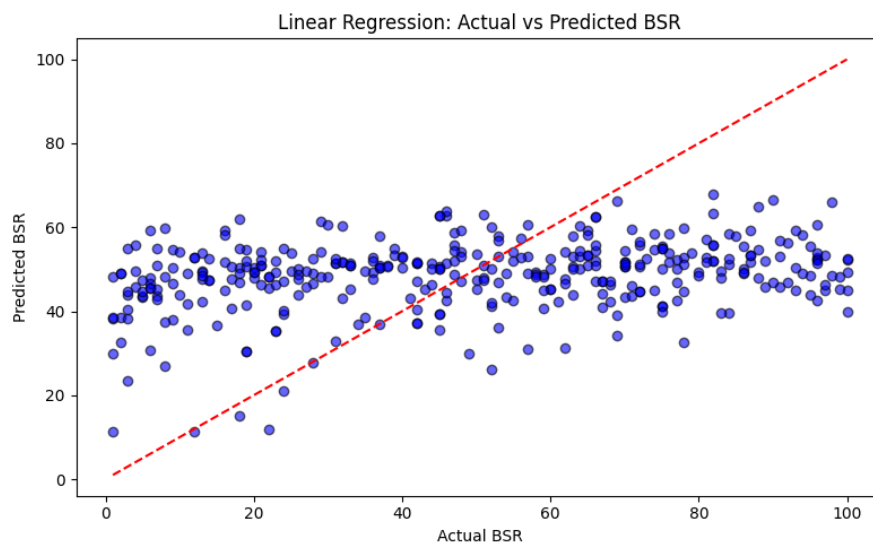
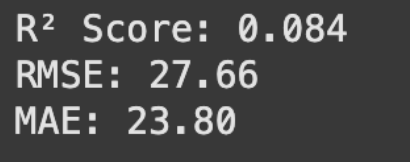


Figure 4.8 Linear Regression

The first model applied to the dataset was Linear Regression, chosen for its simplicity and interpretability. The model was trained using the selected features, including engineered variables like `review_density`, `weighted_rating`, and `log_num_ratings`, as well as one-hot encoded country indicators.

After fitting the model, predictions were made on the test set, and the following evaluation metrics were obtained:



R^2 Score:	0.084
RMSE:	27.66
MAE:	23.80

Figure 4.9 Evaluation Metrics

These results indicate that the Linear Regression model performed poorly in capturing the variability in the Best Seller Rank (BSR). An R^2 score of only 0.084 suggests that the model explains less than 10% of the variance in the target variable. The relatively high RMSE and MAE values further support this conclusion, showing that prediction errors were substantial.

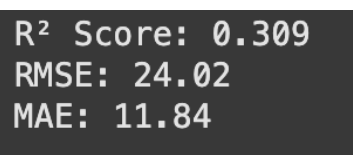
A scatter plot comparing actual and predicted BSR values revealed a wide spread around the ideal prediction line, further confirming the model's lack of accuracy. The poor performance can likely be attributed to the non-linear and complex relationships between features and BSR, which a linear model is not capable of capturing effectively.

As a result, more flexible models—such as decision trees and ensemble methods—were explored in the following sections.

4.5.2. Decision Tree Results

A Decision Tree Regressor was trained to improve upon the shortcomings of the linear regression model by capturing non-linear relationships within the data. The model was trained using the same feature set, including engineered variables and one-hot encoded country data.

After training and predicting on the test set, the performance was evaluated using the following metrics:



R^2 Score:	0.309
RMSE:	24.02
MAE:	11.84

Figure 4.10 Evaluation Metrics

Compared to the Linear Regression model, the Decision Tree achieved a notable improvement in all three metrics. The R^2 score increased from 0.084 to 0.309, indicating a higher proportion of variance explained by the model. Additionally, the RMSE and MAE both decreased, showing that the predictions were closer to the actual BSR values.

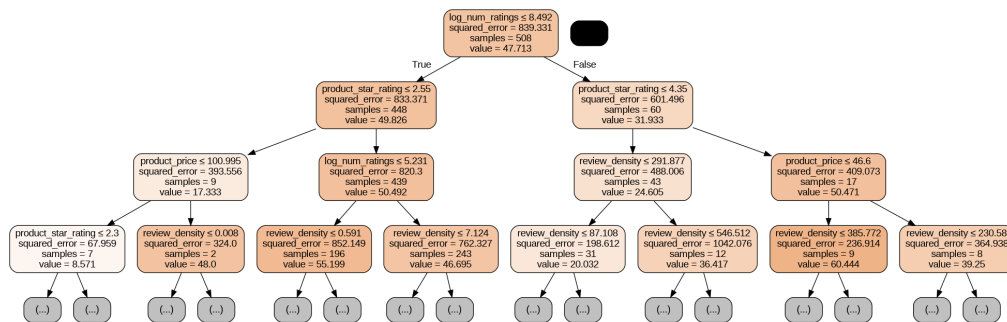


Figure 4.11 Decision Tree

Despite these improvements, the Decision Tree model is still prone to overfitting, especially when trained without pruning or regularisation. This risk was visualised by exporting and plotting the structure of the decision tree (limited to a depth of 3 for interpretability), which showed that splits were strongly influenced by engineered features such as `log_num_ratings` and `review_density`.

While performance was better than the linear model, the Decision Tree's lack of robustness across varied data led to the adoption of ensemble methods—specifically the Random Forest Regressor—as a next step.

4.5.3. Random Forest Results

The Random Forest Regressor was employed as an ensemble-based model to address the overfitting tendency of individual decision trees and to better capture complex feature

interactions. Using 100 estimators and default hyperparameters, the model was trained on the full feature set, including engineered and one-hot encoded variables.

Upon evaluation, the Random Forest model achieved the following results on the test set:

```
R2 Score: 0.575
RMSE: 18.85
MAE: 13.89
```

Figure 4.12 Evaluation Metrics

These results demonstrate a substantial improvement over both the Linear Regression and Decision Tree models. The R^2 score indicates that over 57% of the variance in BSR was explained by the model, while the reduction in RMSE and MAE confirms that prediction errors were considerably lower.

```
Cross-validated R2 scores: [0.68885691 0.0551448 0.76193422]
Average R2: 0.502
```

Figure 4.13 Cross-validated R^2 scores

To validate generalisation, 3-fold cross-validation was conducted, producing an average R^2 score of approximately 0.502, closely aligned with the test performance. This consistency indicates that the model generalises well and is not overfitting.

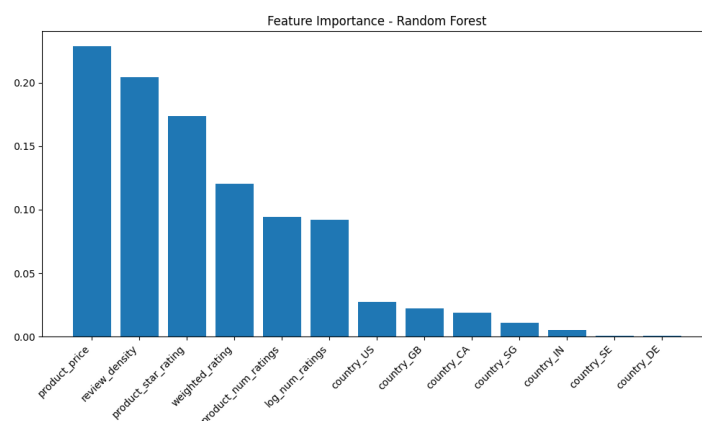


Figure 4.14 Feature Importance

A feature importance plot revealed that variables such as `product_price`, `review_density`, and `product_star_rating` were among the most influential in predicting BSR. This further supports the value of feature engineering in improving model performance.

In summary, Random Forest outperformed all previous models and was considered the most reliable baseline for BSR prediction in this study.

4.6. Summary

This chapter presented the modelling results and evaluation of three machine learning approaches applied to predict Amazon BSR. The workflow progressed from data cleaning and exploratory analysis to feature engineering and supervised regression modelling.

Feature engineering played a crucial role in enhancing model effectiveness. Derived features such as `review_density`, `weighted_rating`, and `log_num_ratings` were found to be important predictors, as confirmed through feature importance analysis.

Initial experiments with Linear Regression revealed that simple linear models were insufficient for capturing the complexity of the data, as indicated by low predictive accuracy and high error values. The Decision Tree Regressor improved upon this by modelling non-linear relationships, though it still showed limitations in generalisation. The most promising results were achieved with the Random Forest Regressor, which demonstrated the best overall performance with an R^2 score of 0.575 and significantly reduced prediction error.

Cross-validation results further supported the generalisability of the Random Forest model, making it a suitable baseline for future refinement or deployment. The chapter highlights that ensemble learning and thoughtful feature transformation are essential for predicting BSR in a complex, non-linear e-commerce environment.