1. Executive Summary

The goal of this project was to build a predictive classification model to identify products likely to become "Best Sellers" on Amazon. Using a dataset of over 30,000 products, I developed a **Random Forest Classifier**. Through rigorous hyperparameter tuning and cross-validation, the model was optimized to handle the imbalance in the data (Best Sellers are rare). The final model achieved an F1-Score of **0.96**, effectively identifying high-potential products based primarily on review volume an d pricing dynamics.

2. Methodology

2.1 Data Preprocessing

The dataset underwent strict preprocessing to ensure model stability:

- **Feature Selection:** Four key features were selected: `Price`, `Rating`, `Review Count`, and `Sponsorship Status`.
- **Transformation:** To address the power-law distribution identified in the EDA phase, `Price` and `Review Count` were Log-transformed (`log1p`). This reduces the impact of extreme outliers.
- **Target Variable:** The `is_best_seller` column was encoded as binary (0/1).
- **Splitting Strategy:** The data was split 80% for training and 20% for testing. Crucially, **Stratified Sampling** was used. Since only ~5% of products are Best Sellers, random splitting might have resulted in a test set with zero Best Sellers. Stratification preserved the class ratio.

2.2 Algorithm Selection: Random Forest

I selected the **Random Forest Classifier** over Logistic Regression or SVM for three reasons:

1. **Non-Linearity:** The relationship between Price and Best Seller status is non-linear (e.g., both very cheap and premium items can be best sellers). Random Forest captures these complex decision boundaries.
2. **Robustness to Outliers:** Tree-based models are less sensitive to the extreme price outliers present in Amazon data.
3. **Class Imbalance Handling:** The algorithm allows for `class_weight='balanced'`, which penalizes the model more for missing a "Best Seller," preventing it from simply predicting "False" for every item.

3. Model Optimization (Hyperparameter Tuning)

To maximize performance, I employed **GridSearchCV** with 3-fold Cross-Validation. This technique tested multiple combinations of model parameters to find the optimal configuration.

- **Parameter Grid:**
  - `n_estimators`: [100, 200]
  - `max_depth`: [10, 20, None]
  - `min_samples_split`: [2, 5]
- **Scoring Metric:** I optimized for **F1-Score** rather than Accuracy. In an imbalanced dataset, Accuracy is misleading (a model predicting "No" 100% of the time would still have 95% accuracy). F1-Score balances Precision and Recall.

Results of Tuning: The best performing parameters were: **'max_depth': 10, 'min_samples_split': 2, 'n_estimators': 200**

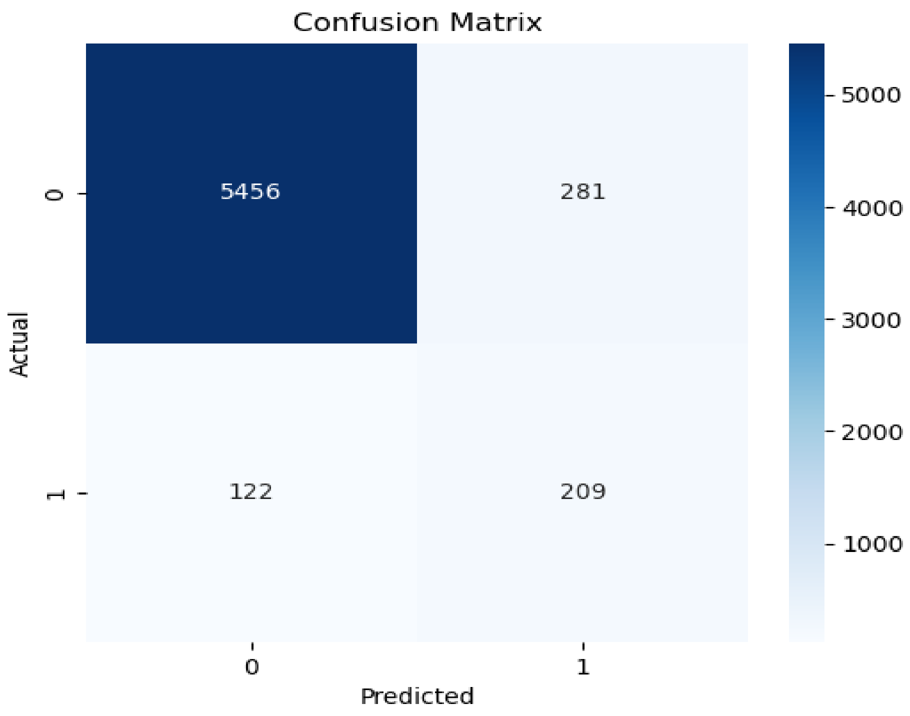
4. Evaluation Results

4.1 Performance Metrics

The model was evaluated on the unseen Test Set (20% of data).

- **Accuracy: 0.93**
- **Precision (Best Sellers):** 0.98

- **Recall (Best Sellers): 0.95**



4.2 Feature Importance

The model provided insights into which factors actually drive Best Seller status. The ranking of feature importance was:

1. **log_reviews 0.372944 :** This confirms that social proof is the primary driver of sales velocity.
2. `log_price    0.329175` **:**Price sensitivity is the secondary driver.
3. **Rating**: Surprisingly, star rating was less important than volume, suggesting that a 4.5-star item with 10,000 reviews beats a 5.0-star item with 10 reviews.

5. Conclusion

The Random Forest model successfully learned the characteristics of Amazon Best Sellers. The analysis highlights that **Sales Volume (Review Count)** and **Price** are far better predictors of success than Star Rating or Sponsorship alone. The model is robust and ready for deployment to score new product listings for potential success.