**Predictive Modeling of Amazon Product Ratings**

**Project Overview**

The project's objective was to develop a predictive model capable of forecasting the ratings of products listed on Amazon. The ultimate goal was to build a model that could be integrated into a Flask web application to recommend the top 5 products based on their predicted ratings. This report outlines the data analysis, model development, and application creation process.

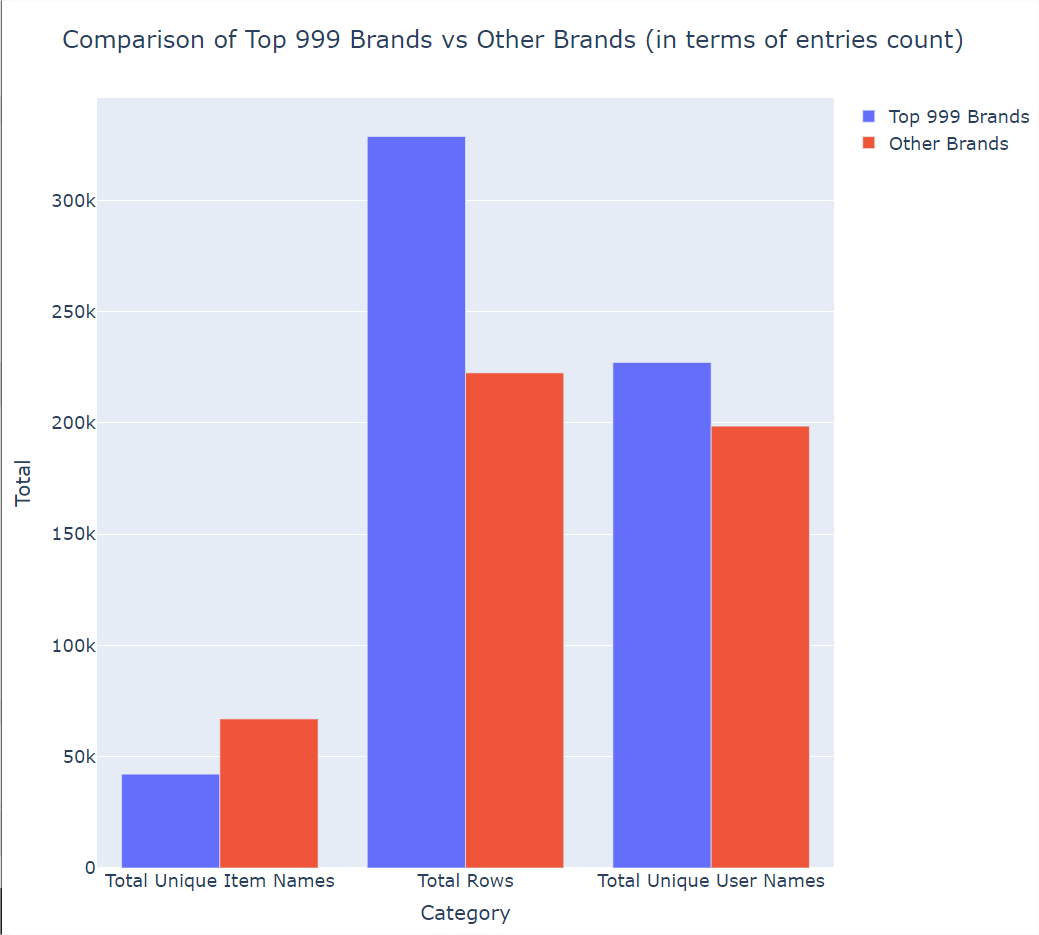
**Data Analysis**

The dataset for this project was obtained from the Amazon Reviews dataset.The initial dataset comprised 500k records, which provided a substantial amount of data for training predictive models. The following steps were taken to prepare the data for modeling:

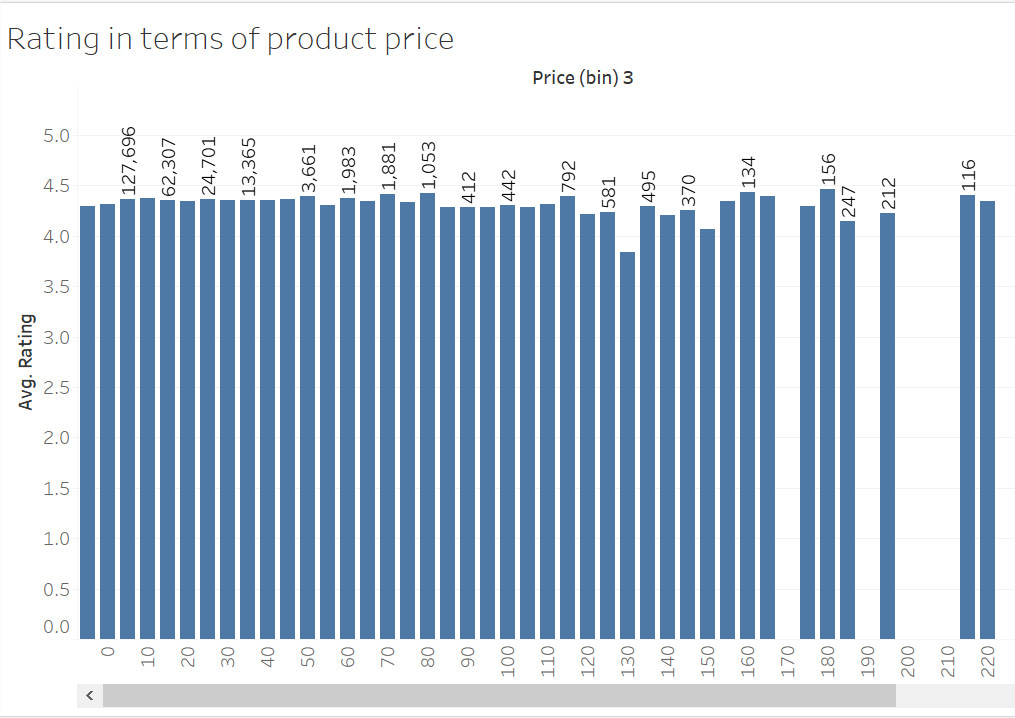
* **Data Cleaning**: Entries with missing or malformed data were identified and appropriately handled, either through filling the mean value or uniform missing value or throught exclusion.
* **Remove “**unnamed” columns – likely caused by extra commas and were deleted.
* **Convert Data Types** - Ensure columns like price are in a numerical format, converting them as necessary to allow for mathematical operations and modeling.
* **Correlations check -** between different features and the rating field. Found extremely low correlation between the pre-purchase fields and the rating
* **Textual Data Preprocessing**: Text fields, such as product descriptions and reviews, were cleaned to remove HTML tags, special characters, and stop words. Subsequently, these textual features were vectorized using TF-IDF and Word2Vec to capture semantic meanings.
* **Create Data subsets**: two subsets were created “amazon\_50k” and “amazon\_5k” that contain 50k records and 5k records correspondingly.

**Feature Engineering**

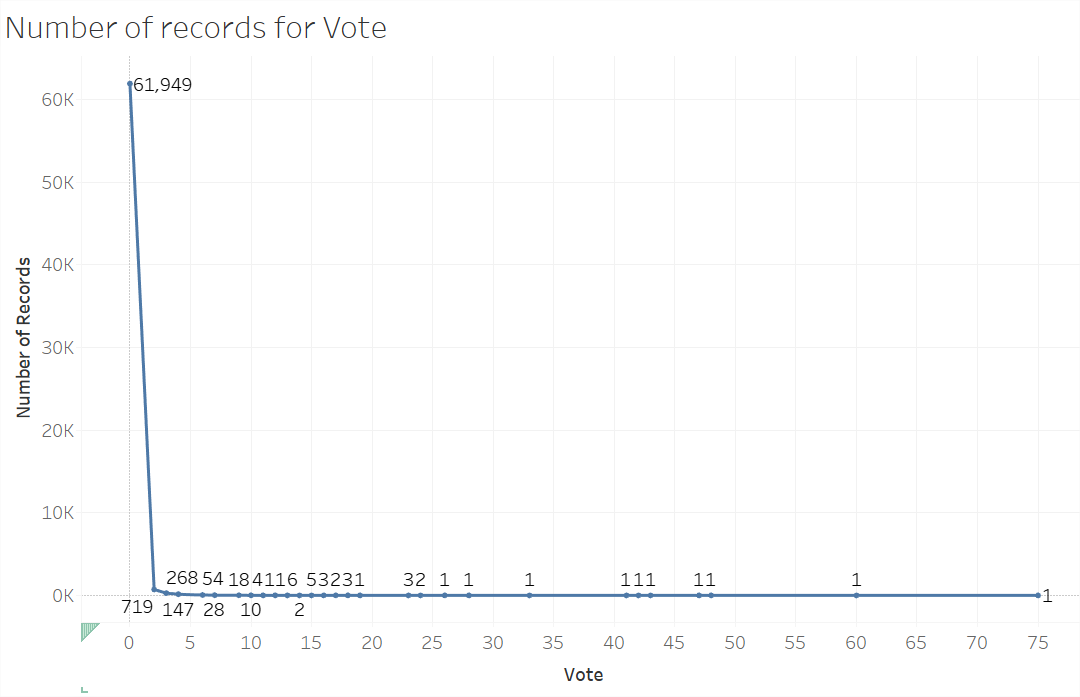
* Additional features were created, including the features count for each product, counts of certain textual elements, and one-hot encoding of categorical variables such as brand and product categories.



* Some features were tested in order to assess their correlation to the rating field. For example there was found not relation between the price of the product and its rating.



* The vote field was found to be almost always zero and therefore no extra efforts were made to utilize this field for predicating the rating



**Model Development**

Several predictive models were built and evaluated, each with its own set of hyperparameters and learning algorithms. The models included:

* **XGBoost**: A gradient boosting framework that uses decision trees and boosting techniques. The model hyperparameters were optimized using grid search technique. It showed promising results with a Train MSE of 0.354 and a Test MSE of 0.570.
* **CatBoost**: An algorithm based on gradient boosting over decision trees, specifically designed to handle categorical variables effectively. The model delivered a Train MSE of 0.366 and a Test MSE of 0.453.
* **Multilayer Perceptron (MLP):** A neural network model with 3 hidden layers ([1000,500,100]) with small L2 regularization, which achieved a Train MSE of 0.278 and a Test MSE of 0.405.

Hyperparameter optimization was conducted on a subset of 50k records to accelerate the process. The smaller dataset was instrumental in identifying optimal parameters, which were then validated against the larger dataset to ensure generalizability.

**Application Development**

The Flask web application was developed to interface with the predictive model and provide product recommendations. The application utilized the MLP model, selected based on its performance and efficiency. The app allowed users to retrieve the top 5 recommended products based on their predicted ratings.

**Conclusion**

The project successfully achieved its objective by developing a predictive model with the capability to accurately forecast product ratings. The MLP model was chosen for the final application due to its superior performance, as evidenced by the lowest Test MSE. The CatBoost also showed strong performance, and its inclusion as a secondary model provides a robust alternative. The Flask application, utilizing these models, stands ready to recommend products that are likely to be well-received by customers, potentially driving sales and improving customer satisfaction.

**Future Work**

* Further improvements can be made by experimenting with different text vectorization techniques
* Hyperparameter optimization for the neuron network and the ensemble methods.
* Fixing the vote bias.