**Instacart Market Basket Analysis: Predicting Product Reorders Using Machine Learning**

***Abstract* --** This project aims to predict customer reorder behaviors using machine learning models based on the Instacart Market Basket dataset. By applying Logistic Regression, Random Forest, Gradient Boosting (XGBoost), and Deep Learning, we identify products likely to be reordered, optimize inventory, and enhance customer satisfaction through personalized recommendations. Data preprocessing, feature engineering, and cross-validation were used to ensure model robustness and accuracy. Gradient Boosting and Random Forest provided the most balanced results, indicating their suitability for retail environments. Future work will focus on improving model efficiency and exploring real-time deployment for enhanced user experiences.

**1. Introduction**

**1.1 Project Overview**

The Instacart Market Basket Analysis aims to predict whether a product will be reordered by a customer in their future shopping trips. This project focuses on understanding customer purchasing patterns, which can help improve Instacart's recommendation system and ultimately boost customer satisfaction and retention. By using various machine learning models, we attempt to predict the likelihood of products being reordered, enabling personalized product recommendations.

**1.2 Objectives**

The primary objectives of this project are:

* To explore customer purchasing behavior and product reorder patterns.
* To develop machine learning models that can accurately predict product reorders.
* To compare the performances of different models and provide insights into their strengths and limitations.
* To generate actionable recommendations for Instacart based on the analysis results.

**2. Literature Review**

In recent years, machine learning has become an essential tool for predicting customer behavior in the retail sector. The use of machine learning models such as logistic regression, random forest, and gradient boosting has demonstrated significant success in predicting customer preferences, product recommendations, and shopping trends. These models are well-suited for handling the complex nature of retail data, which often involves large volumes of information with high dimensionality.

Logistic regression is often used as a baseline model for classification tasks due to its simplicity and interpretability. Random forest, on the other hand, is an ensemble technique that builds multiple decision trees and averages their results to improve accuracy and reduce overfitting. Gradient boosting further enhances prediction performance by iteratively correcting the errors of weak learners, making it a powerful tool for tasks like product recommendation.

In addition to traditional machine learning models, recent advancements in deep learning have enabled the development of more sophisticated models for retail prediction. Deep learning models, such as neural networks, are capable of capturing complex, non-linear relationships within data. These models have been particularly successful in applications like personalized product recommendations, where they can analyze user behavior and provide highly customized suggestions.

Machine learning techniques offer high accuracy and flexibility, making them ideal for addressing the complex nature of customer purchase behavior. By leveraging historical purchase data and user interactions, these models can provide insights that help retailers optimize their marketing strategies, inventory management, and customer experience. In this study, we utilize a range of machine learning models to predict product reorders, aiming to improve Instacart's recommendation system and enhance overall customer satisfaction.

**3. Data Overview and Preprocessing**

**3.1 Dataset Description**

The Instacart Market Basket dataset used in this study consists of over 3 million grocery orders from more than 200,000 unique users. This dataset includes multiple components: order history, product details, user-specific information, and product-specific attributes. The dataset is comprised of several CSV files, each containing different types of information, such as:

**Orders Dataset**: This dataset includes information on user orders, including order IDs, user IDs, order sequence numbers, the day of the week, and the time of day the order was placed. Additionally, it provides data on the days since the previous order, which is instrumental in understanding purchasing cycles.

**Products Dataset**: Contains product details like product IDs, product names, aisles, and departments. This information helps identify which types of products are being reordered frequently.

**Order-Products Dataset**: Split into prior and train files, this dataset contains the details of each order, including product IDs, add-to-cart order, and whether the product was reordered or not.

**Aisles and Departments Dataset**: These datasets classify products into different aisles and departments, which helps in grouping products for further analysis of purchasing trends.

**3.2 Data Cleaning and Preprocessing Steps**

The preprocessing steps involved multiple stages to ensure data quality and the extraction of meaningful features:

* **Data Cleaning**: To handle missing values, the 'days\_since\_prior\_order' column was filled with zeros for first-time orders. Critical data points such as product IDs and user IDs were also verified for completeness to ensure no records with essential missing data points were used in model training.
* **Dataset Merging**: The various datasets (orders, products, aisles, departments, order\_products\_\_prior, and order\_products\_\_train) were merged to create a unified dataset. This merging process was crucial for linking user purchase history, product information, and user-specific ordering behavior into a comprehensive dataset, allowing deeper analysis of trends and reorder behaviors.
* **Bias Consideration**: During the preprocessing phase, we ensured that frequent users and popular products did not dominate the dataset, thereby introducing bias. A balanced approach was adopted to ensure that both frequent and infrequent users and products were adequately represented, allowing the models to generalize well across different user segments.

**3.3 Feature Engineering & Scaling**

**Feature Engineering**: Feature engineering played a significant role in improving model accuracy. We engineered several new features to better capture user-product interactions:

* **Order Frequency Features**: Mean order frequency and days since the last order were calculated for each user to capture their purchasing habits. These features help determine how frequently a customer buys specific products and are indicative of their likelihood to reorder.
* **Reorder Rate**: The reorder rate was computed to quantify the tendency of users to reorder items. This feature helped the models identify products that were essential or habitual purchases for users.
* **Time-Based Features**: Features such as the average hour of ordering and day of the week were added to capture temporal patterns in purchasing behavior. These features are valuable for understanding when customers are most likely to place orders, helping to predict reorders more accurately.
* **Product Popularity**: A product's popularity was derived based on the total number of times it was purchased across all users. This feature helped the model understand which products were more likely to be reordered on a general level.
* **User-Product Interaction**: Interaction features between users and products, such as the number of times a user had previously ordered a product, were generated to provide personalized information for the model. These features are instrumental in creating a tailored recommendation system that accounts for individual user preferences.
* **Cart Position**: The position of a product in the cart was included as a feature, as it provides insight into the importance of an item to the customer. Items added early in the cart are often staples, while those added later may be impulse buys.

**Feature Scaling:** Numerical features were scaled using StandardScaler to bring them to a common range, which helps in optimizing the performance of machine learning models. Models like logistic regression and gradient boosting benefit from feature scaling, as it leads to faster convergence and improved accuracy. For tree-based models, such as random forest and XGBoost, scaling is not strictly necessary but was applied to maintain consistency across all models.

**4. Exploratory Data Analysis (EDA)**

Exploratory data analysis was conducted to gain insights into customer behavior and identify important trends. Key findings from the EDA include:

* **Most Frequently Reordered Products:** Products such as organic bananas and avocados were among the most frequently reordered, indicating their popularity and staple nature in users' shopping habits. A bar chart of the top 10 most reordered products was generated.

A graph of blue bars

Description automatically generated with medium confidence

* **Time-Based Reordering Patterns:** Most orders were placed on Sundays, with a peak during the early afternoon (between 12 PM and 3 PM). This suggests that customers often do their grocery shopping during weekends and around lunchtime. Promotions could be optimized for these times to increase sales.

A graph of distribution of orders

Description automatically generated

* **Customer Reordering Behavior:** Users with higher reorder rates tend to exhibit consistent purchasing habits. Products added early to the cart were often high-priority items, suggesting that personalized recommendations should prioritize these products.

A green line graph with numbers

Description automatically generated

**5. Machine Learning Models**

**5.1 Logistic Regression Model**

The logistic regression model was implemented to establish a baseline for predicting product reorders. This model estimates the likelihood of a product being reordered based on input features such as order frequency, reorder rate, and user-specific purchase patterns. Its simplicity and interpretability made it an effective choice for understanding relationships between features and reorder behavior.

The data was split into training and testing sets with a 70-30 ratio, ensuring the target variable was stratified to maintain class balance. Before training, the features were standardized using a scaler to improve convergence and model stability. The logistic regression model was trained using scikit-learn with a maximum iteration limit of 1000 to ensure proper convergence.

Evaluation on the test set yielded an accuracy of 77%. The confusion matrix revealed that while the model performed well in predicting products that were not reordered, it struggled with identifying products that were reordered. This limitation was reflected in the recall score of 10% for the reordered class, indicating a high rate of false negatives. Despite this, the precision for the reordered class was 56%, suggesting the model was relatively conservative in its predictions.

The classification report highlighted an imbalance in performance between the two classes, with significantly higher metrics for the non-reordered class compared to the reordered class. This disparity points to the challenge of capturing the complexity of reorder behavior using a linear model.

The ROC curve analysis provided an AUC score of 0.71, indicating moderate discriminatory ability between the reordered and non-reordered classes. This reinforces the conclusion that while logistic regression is a useful starting point for understanding feature significance and baseline prediction, it has limited capacity to handle the non-linear interactions and complexities of the data.

A graph of a logistic regression

Description automatically generatedA chart with yellow and blue squares

Description automatically generated

**5.2 Random Forest Model**

The random forest model was utilized to address the limitations of logistic regression by leveraging its ability to handle non-linear feature interactions and higher-dimensional data. Random forests, being an ensemble learning method based on decision trees, are particularly effective in capturing complex patterns within the data while reducing the risk of overfitting due to the averaging of multiple trees.

The model was trained using 100 estimators with a maximum depth of 10, and the minimum samples required for a split was set to 2. These parameters were chosen to balance computational efficiency and model performance. Feature importance metrics generated by the random forest provided valuable insights into which variables most significantly contributed to predicting reorder likelihood.

The random forest model achieved an accuracy of 78% on the test set, an improvement over logistic regression. The classification report revealed enhanced recall for the reordered class at 16%, indicating that the model was more effective at identifying reordered products. Precision for the reordered class improved to 69%, showing a better balance between false positives and true positives. The F1-score for the reordered class increased, reflecting overall better performance in handling the minority class.

Feature importance analysis revealed that variables such as reorder rate, mean days since prior order, and user-specific purchase frequency were among the most significant predictors of reorder likelihood.

**5.3 Gradient Boosting Model (basic)**

Gradient boosting was employed to further improve predictive performance by sequentially building models that corrected errors made by previous iterations. This approach allows for capturing complex, non-linear relationships in the data while maintaining a high level of flexibility.

The model was trained with 100 estimators, a learning rate of 0.1, and a maximum depth of 3 for each tree. These hyperparameters were chosen to balance the trade-off between model complexity and overfitting. Gradient boosting achieved an accuracy of 87% on the test set, significantly outperforming logistic regression and random forest in terms of overall predictive power.

The classification report showed a precision of 95% and a recall of 49% for the reordered class, resulting in an F1-score of 64%. This indicated a notable improvement in the model’s ability to identify reordered products while maintaining a high level of precision. The increased recall reflects the model’s ability to capture a broader range of reordered instances compared to the previous models.

The ROC curve analysis produced an AUC score of 0.91, demonstrating the model’s strong discriminatory ability between reordered and non-reordered classes. This improvement highlights the effectiveness of gradient boosting in handling complex feature interactions and improving performance across both classes.

The results indicated that gradient boosting was particularly effective in improving recall for the reordered class, making it a suitable choice for scenarios where capturing reordered products is critical. However, further hyperparameter tuning and addressing class imbalance could enhance performance even further.

**5.4 Gradient Boosting Model (XGBoost)**

XGBoost, a high-performance implementation of gradient boosting, was employed to further enhance predictive accuracy and computational efficiency. Its regularization capabilities and optimization techniques make it particularly well-suited for handling large datasets and mitigating overfitting.

The model was trained with 100 estimators, a learning rate of 0.1, and default regularization parameters. Cross-validation was applied during training to evaluate the model’s performance across different data splits, yielding a mean accuracy of 87.8%. On the test set, XGBoost achieved an accuracy of 88%, slightly outperforming the standard gradient boosting model.

The classification report indicated a precision of 92% and a recall of 52% for the reordered class, resulting in an F1-score of 66%. The improvement in recall over gradient boosting highlights XGBoost’s ability to capture even more reordered instances, making it particularly effective for imbalanced datasets.

The ROC curve analysis for XGBoost produced an AUC score of 0.90, underscoring its strong ability to distinguish between reordered and non-reordered classes. Additionally, feature importance analysis identified reorder rate, user-specific purchase patterns, and mean days since prior order as key predictors, providing actionable insights for improving the model and its applications.

XGBoost demonstrated superior predictive performance compared to previous models, particularly in its handling of imbalanced data and capturing complex patterns. Its computational efficiency and flexibility make it an excellent choice for large-scale predictive tasks, though further fine-tuning could potentially enhance its performance even further

A chart of a confusion matrix

Description automatically generated

**5.5 Deep Learning Model**

A deep learning model was implemented to explore its potential in capturing complex, non-linear relationships within the data. A feedforward neural network architecture was chosen, consisting of multiple fully connected layers with ReLU activation functions. The final output layer used a sigmoid activation function to provide probabilities for the binary classification task.

The model architecture included three hidden layers with 256, 128, and 64 neurons, respectively. The network was trained for 20 epochs using the Adam optimizer and binary cross-entropy as the loss function. A batch size of 128 was used to optimize the training process. Accuracy and recall were monitored as metrics during training to assess performance.

The deep learning model achieved an accuracy of 90.4% on the test set, outperforming all previously implemented models. The classification report highlighted a recall of 63% for the reordered class, indicating the model’s superior ability to identify reordered products compared to gradient boosting and XGBoost. The precision for the reordered class was 61%, resulting in an F1-score of 62%, demonstrating a balanced trade-off between precision and recall.

The ROC curve analysis showed an AUC score of 0.91, confirming the model’s strong ability to discriminate between reordered and non-reordered classes. Over the course of training, both training and validation accuracy consistently improved, while the loss decreased, indicating that the model effectively learned from the data without significant overfitting.

Despite its superior performance, the deep learning model required significantly more computational resources and training time compared to traditional machine learning models. While it excelled at capturing complex patterns, the results suggest that further improvements could be achieved by incorporating advanced techniques such as dropout for regularization, hyperparameter tuning, or utilizing a larger dataset.

Deep learning proved to be the most effective model in this study, providing the highest accuracy and recall for the reordered class. This performance makes it particularly well-suited for large-scale applications where predictive accuracy is critical.

**A graph of a graph

Description automatically generated** **A graph showing the difference between training and validation accuracy

Description automatically generated**

**6. Results and Discussion**

**6.1 Summary of Model Performances and Feature Importance**

The performance of each model varied significantly, highlighting their strengths and limitations. Logistic regression served as a baseline, providing interpretability but struggling to capture complex relationships in the data. Random Forest improved upon this by effectively handling feature interactions and ranking feature importance, identifying variables such as reorder rate and user-specific purchase patterns as critical. Gradient Boosting and XGBoost demonstrated even greater predictive power, with XGBoost achieving the highest accuracy among traditional machine learning models. The deep learning model outperformed all others, excelling in recall and overall discriminatory ability.

Key feature importance rankings across models consistently pointed to variables such as reorder rate, mean days since prior order, and user-specific purchase frequency as the most impactful predictors of reorder likelihood.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model** | **Accuracy** | **Precision** | **Recall** | **F1-Score** |
| Logistic Regression | 0.77 | 0.56 | 0.10 | 0.17 |
| Random Forest | 0.78 | 0.69 | 0.16 | 0.26 |
| Gradient Boosting | 0.87 | 0.95 | 0.49 | 0.64 |
| XGBoost | 0.88 | 0.92 | 0.52 | 0.66 |
| Deep Learning | 0.90 | 0.61 | 0.63 | 0.62 |

**6.2 Accuracy, Precision, Recall, and F1 Score Comparisons**

The models displayed distinct trade-offs in terms of accuracy, precision, recall, and F1 score. Logistic regression achieved an accuracy of 77%, with strong precision for the non-reordered class but a poor recall for reordered products. Random Forest improved recall for the reordered class to 16% while maintaining an accuracy of 78%. Gradient Boosting significantly boosted recall to 49%, with an accuracy of 87%, while XGBoost achieved the highest accuracy among traditional models at 88% and further improved recall to 52%.

The deep learning model outperformed all traditional methods, achieving an accuracy of 90.4%, a recall of 63% for the reordered class, and an F1-score of 62%. This marked improvement in recall highlights its ability to capture complex patterns, making it particularly effective for addressing the challenges posed by imbalanced datasets.

A graph of different colored lines

Description automatically generated

**6.3 Strengths and Limitations of Each Model**

* **Logistic Regression**: The primary strength of logistic regression lies in its interpretability and simplicity, making it an excellent baseline model. However, it struggled to capture non-linear relationships, leading to poor performance for the reordered class.
* **Random Forest:** Random Forest effectively captured non-linear feature interactions and provided valuable insights through feature importance rankings. However, its performance gains were limited compared to boosting methods, particularly in recall.
* **Gradient Boosting:** Gradient Boosting demonstrated significant improvements in recall and overall accuracy, making it effective for predicting reordered products. Its iterative learning approach allowed it to correct errors from previous iterations, but it required careful hyperparameter tuning to avoid overfitting.
* **XGBoost:** XGBoost offered the best balance of performance and computational efficiency among traditional models. Its regularization capabilities helped prevent overfitting, while its scalability made it suitable for large datasets. However, like Gradient Boosting, it required extensive tuning for optimal performance.
* **Deep Learning:** The deep learning model exhibited the highest recall and overall accuracy, showcasing its ability to capture complex patterns in the data. However, it required significantly more computational resources and was more prone to overfitting without careful regularization and monitoring.

**7. Conclusion and Recommendations**

The analysis of multiple machine learning models for predicting product reorders in Instacart’s dataset has demonstrated the varying strengths and weaknesses of these approaches. The logistic regression model provided a simple and interpretable baseline, while advanced models like Gradient Boosting, XGBoost, and Deep Learning significantly outperformed it in predictive accuracy and recall, particularly for the reordered class.

**Recommendations for Instacart**

1. Adopt Gradient Boosting or XGBoost for Production:

These models offer an excellent balance of accuracy, recall, and precision, making them reliable for real-world deployment. XGBoost provides high computational efficiency and scalability, which is crucial for handling Instacart’s large-scale dataset.

1. Focus on Feature Engineering:

Features such as reorder rate, user-specific purchase frequency, and mean days since the prior order were identified as highly influential. Additional features, like product price sensitivity, seasonal trends, or customer segmentation, could further improve predictive accuracy.

1. Deploy the Deep Learning Model for High-Stakes Use Cases:

The deep learning model showed the highest recall and overall performance, making it suitable for high-stakes applications where predicting reordered items with minimal false negatives is critical. However, its computational demands may make it less feasible for all operations.

**Future Work**

* **Explore Ensemble Models:** Combining the strengths of multiple models, such as Gradient Boosting and Deep Learning, could yield even better predictive performance and robustness.
* **Real-Time Prediction Systems:** Develop and deploy systems that make real-time predictions during customer browsing sessions, enhancing the relevance of product recommendations.
* **Hyperparameter Optimization:** While initial models were tuned, deeper exploration of hyperparameters using techniques like Bayesian optimization or automated machine learning (AutoML) could yield further gains in performance.