# Data Mining Project Report for an E-commerce Platform

Ahmet Selim Bayram
*Artificial Intelligence and Data Engineering Department*
*Istanbul Technical University*
Istanbul, Turkıye
bayramah20@itu.edu.tr

Emre Günel
*Artificial Intelligence and Data Engineering Department*
*Istanbul Technical University*
Istanbul, Turkıye
gunele20@itu.edu.tr

***Abstract*—This project focuses on developing a predictive analytics framework aimed at enhancing the customer experience on a e-commerce platform. The objective is to provide timely recommendations for replenishing essential everyday items frequently purchased by customers. A data-driven system has been developed to predict the specific week when the purchase is expected to occur.**

**As part of this project, a comprehensive review of the literature on predictive analytics was conducted to establish a theoretical foundation. Based on insights gained, predictive models were proposed and tested, like Long Short-Term Memory (LSTM) networks and Extreme Gradient Boosting (XGBoost). However, initial experiments with these models did not yield satisfactory results, indicating limitations in their ability to capture the complexities of customer replenishment behavior in the given dataset. This highlighted the need for further exploration and refinement of model architectures to improve prediction accuracy.**

***Index Terms*—predictive analytics, e-commerce, data mining, customer recommendations, sales forecasting**

## I. Introduction

A leading global e-commerce platform offers a wide range of household and consumable products. The platform aims to enhance customer experience through predictive analytics. One of the major challenges faced by the platform is ensuring customers receive timely reminders to restock frequently purchased items, such as personal care products, cleaning supplies, and pet food. This project proposes innovative data mining solutions to address this problem by leveraging historical transactional data, product attributes, and category hierarchies.

Four primary datasets are utilized to predict customer behavior and product purchasing cycles, each described below:

- **Transactions Dataset:** This dataset contains historical transaction data representing individual customer purchases. Each row records:
  - `customer_id`: A unique identifier for each customer.
  - `product_id`: A unique identifier for each purchased product.
  - `purchase_date`: The date of purchase.
  - `quantity`: The number of units purchased in the transaction.

  This dataset is critical for analyzing purchasing frequency and quantities across customers and products.
- **Product Catalog Dataset:** This dataset provides detailed information about products, linking them to categorical and manufacturer-specific attributes. It includes:
  - `product_id`: A unique product identifier.
  - `manufacturer_id`: The manufacturer associated with the product.
  - `attribute_1`–`attribute_5`: Categorical attributes describing specific features of the product, such as product group or functional category.
  - `categories`: A list of associated categories providing additional context.

  This dataset is instrumental in incorporating product-level features into the predictive models.
- **Product Category Map Dataset:** This dataset outlines the hierarchical structure of product categories. It includes:
  - `category_id`: A unique identifier for each product category.
  - `parent_category_id`: The parent category to which a specific category belongs.

  This hierarchical structure is crucial for grouping products and understanding relationships between similar items.
- **Test Dataset:** This dataset serves as the input for prediction and includes:
  - `id`: A row index.
  - `customer_id`: A unique identifier for the customer.
  - `product_id`: A unique identifier for the product.
  - `prediction`: A column to be populated with predicted values representing the week of replenishment.

  It provides the structure and format required for submitting predictions.

The integration and processing of these datasets aim to develop predictive models capable of determining both the occurrence and timing of customer product replenishment. This approach supports the platform's goal of optimizing inventory management, enhancing customer satisfaction, and driving personalized marketing strategies.

## II. Literature Review

Predictive analytics techniques are widely used in e-commerce to enhance customer experiences and maximize business outcomes. Machine learning models such as Random Forest, Gradient Boosting (XGBoost, LightGBM), and Logistic Regression are effective in predicting customer purchases based on their characteristics and past behaviors. XGBoost, developed by Chen and Guestrin (2016), is particularly preferred due to its high performance on large datasets [1]. These models utilize past purchase data and product features to predict purchasing timing and recurrence probability.

The application of predictive analytics in e-commerce has been shown to significantly improve inventory management, customer behavior analysis, and sales forecasting. By leveraging machine learning algorithms such as decision trees and neural networks, businesses can forecast product demand and optimize their stock levels, reducing excess inventory and preventing stockouts [5]. This approach enables businesses to maintain optimal stock levels and lower storage costs.

For analyzing purchasing cycles and timing, traditional statistical models like ARIMA (Autoregressive Integrated Moving Average) continue to be effective in forecasting future purchase events. These models are especially useful for time-dependent data analyses and have proven successful in predicting customer behavior based on historical sales data [2].

As traditional statistical models face limitations in handling large-scale time-dependent data, deep learning models such as LSTM (Long Short-Term Memory) and GRU (Gated Recurrent Unit) have gained popularity. LSTM, developed by Hochreiter and Schmidhuber (1997), stands out for its capacity to learn long-term dependencies, making it suitable for predicting customer purchasing behavior over extended periods [3]. These models can identify patterns in customers' purchasing history, which is crucial for making accurate predictions.

In addition to these approaches, personalized recommendation systems play a significant role in predicting customer behavior. Collaborative filtering techniques, as demonstrated by Koren et al. (2009), successfully provide suitable product recommendations to users based on their past purchase and rating data [4].

Recent studies have shown that deep learning models like LSTM outperform traditional models such as SARIMA in short-term sales forecasting tasks, especially for large datasets in e-commerce contexts. In a comparative analysis, LSTM-based approaches were found to offer more accurate predictions compared to traditional methods like SARIMA, demonstrating the strength of deep learning models in capturing complex patterns in time series data [6]. For example, Ecevit et al. (2023) demonstrated that LSTM-based models outperformed both Prophet and SARIMA in short-term sales forecasting tasks, indicating the superiority of LSTM in capturing hourly sales variations [6]. Similarly, Rahman & Zaman (2024) developed a hybrid CNN-BiLSTM model that effectively captured temporal dependencies and outperformed traditional models in retail sales forecasting [7].

## III. Data Preprocessing and Feature Engineering

To ensure the datasets were ready for predictive modeling, extensive preprocessing and feature engineering steps were performed. The raw data was transformed into a structured format suitable for both LSTM and XGBoost models, with tailored preprocessing to address the specific requirements of each model.

### A. LSTM Model

*1) Data Cleaning and Integration:* To prepare the data for the LSTM model, multiple preprocessing steps were applied:

- **Transaction Data:** The `purchase_date` field was converted to a datetime format, and the data was sorted by `customer_id`, `purchase_date`, and `product_id`. Weekly aggregation was performed to create a time series for each customer-product pair, with weeks as columns and quantities as values.
- **Product Catalog:** The `categories` column was exploded to create one-to-many mappings between products and categories. Each product was enriched with its associated parent categories, hierarchical information, and specific attributes such as manufacturer ID.
- **Customer Features:** Features such as total purchases, unique products, purchase days, and the most frequent product were aggregated at the customer level. Additionally, the top five categories for each customer were identified based on purchased quantities.

*2) Feature Aggregation:* Three main data representations were created:

1) **Time Series Data:** Weekly purchase quantities for each customer-product pair were structured into sequences of eight historical weeks.
2) **Customer Features:** Aggregated customer-level numerical features were integrated into the sequences.
3) **Product Features:** Each product's categorical attributes, parent categories, and manufacturer information were included to enhance model inputs.

*3) Sequence Creation and Tensor Conversion:* Input sequences were generated to prepare the data for the LSTM model. The process involved:

- Extracting sliding windows of eight weeks from the time series data as input (`X`).
- Generating multiclass target labels (`y`), where each label represents the first week in the four-week prediction horizon with a repurchase. A label of 0 indicates no repurchase occurred during this period.
- Combining customer and product features into separate arrays to ensure compatibility with tensor-based models.

This approach structured the data into a format suitable for sequential modeling, incorporating historical trends and entity-specific characteristics for enhanced predictive accuracy.

*4) Embedding Layers:* Embedding layers were utilized to encode categorical features into low-dimensional vectors:

- **Customer Embeddings:** The `customer_id` was embedded into a vector space, with the output dimension

calculated as the nearest power of two greater than or equal to the square root of the number of unique customers.
- **Product Embeddings:** Each categorical feature (e.g., product ID, manufacturer ID) was embedded separately and combined into a unified representation.

This approach captures latent relationships between entities (e.g., customers and products), improving model performance.

*5) Time Series Modeling with LSTM:* The LSTM model architecture included:

- Two stacked LSTM layers with 32 and 16 units, respectively, to capture temporal dependencies in sequential data.
- Dense layers with ReLU activation to process the LSTM output.
- Dropout layers to mitigate overfitting during training.
- A final softmax layer for multiclass classification, predicting the week of replenishment.

*6) Scaling and Normalization:* Numerical features were normalized using MinMaxScaler to ensure compatibility with the LSTM model. Separate scalers were applied to customer features and time series data, and the scalers were saved for reproducibility.

*7) Model Training and Evaluation:* The final data was split into training, validation, and test sets using a 70-20-10 ratio. The model was trained using the Adam optimizer with a learning rate of 0.001 and sparse categorical cross-entropy loss. Class weights were computed to handle imbalanced target labels, ensuring robust performance across all classes.

### B. XGBoost Model

*1) Data Transformation and Aggregation:* The preprocessing pipeline for the XGBoost model began with converting the `purchase_date` column to a datetime format for accurate temporal calculations. The dataset was grouped by `customer_id` and `product_id` to compute essential features such as:

- First and last purchase dates,
- Total purchase count, and
- Second last purchase date.

A custom function was applied to derive the second last purchase date, ensuring it only considered customer-product pairs with two or more transactions.

*2) Calculating Purchase Patterns:* To analyze purchase patterns, several derived features were computed:

- **Average Days Between Purchases:** Calculated as the difference between the first and last purchase dates divided by the total purchases minus one.
- **Week Code:** Represented the number of weeks between the last purchase date and a reference date (2020-12-25).
- **Product Popularity:** Computed based on purchase frequency and categorized into deciles, labeled from 1 (least popular) to 10 (most popular).

*3) Filtering and Enriching the Data:* To ensure meaningful patterns were retained, the dataset was filtered and enriched as follows:

- Customer-product pairs with fewer than two purchases were excluded.
- Logical filters were applied to include:
  - Pairs with total purchases greater than two and a positive `week_code`, or
  - Pairs with total purchases greater than one and a non-positive `week_code`.
- A second week code, `week_code_2`, was introduced to represent the number of weeks between the second last purchase date and the reference date.

*4) Final Steps and Output:* The final preprocessing steps prepared the dataset for the XGBoost model:

- Binary and multiclass outputs were created to determine both the occurrence of a purchase and the specific week of replenishment.
- Unnecessary columns were removed, and features were renamed for clarity.
- The processed dataset was saved as `preprocessed_train_set.csv` for training.

## IV. Proposed Models

Given the structure of the dataset, which includes transactional data, product attributes, and category hierarchies, we evaluated different modeling approaches to predict customer replenishment behavior. This section provides details of the Long Short-Term Memory (LSTM) model and the XGBoost-based two-stage framework.

### A. LSTM Model

The Long Short-Term Memory (LSTM) model was employed to capture sequential patterns in customer transactions. This model is particularly well-suited for time-series data and was designed to predict both whether a product would be repurchased and the most likely week for replenishment.

*1) Input Data Preparation:* Input sequences were generated by aggregating weekly purchase quantities for each customer-product pair. Historical data spanning eight weeks was used to predict purchasing behavior for the subsequent four weeks. The features included:

- **Weekly Purchase Quantities:** Time-series input representing purchase behavior over eight weeks.
- **Customer-Level Features:** Aggregated numerical features such as total purchases, average purchase quantity, and the top five product categories for each customer.
- **Product-Level Features:** Categorical features such as product attributes, parent categories, and manufacturer details, embedded into low-dimensional vectors for training.

*2) Model Architecture:* The architecture of the LSTM model consisted of the following components:

- **LSTM Layers:** Two stacked LSTM layers with 32 and 16 units, respectively, to capture temporal dependencies in the time-series data.

- **Dense Layers:** Fully connected layers with ReLU activation to process the LSTM output.
- **Dropout Layers:** Dropout layers were used to mitigate overfitting during training.
- **Embedding Layers:** Embedding layers encoded categorical features such as `customer_id` and `product_id` into low-dimensional spaces, capturing latent relationships between entities.
- **Softmax Output:** A final softmax layer predicted the week of replenishment as a multiclass classification problem.

*3) Training and Evaluation:* The model was compiled using the Adam optimizer and sparse categorical cross-entropy loss. To address class imbalances, class weights were computed and applied during training. The dataset was split into training, validation, and test sets with a 70-20-10 ratio. The model was trained for 10 epochs with a batch size of 1024, and validation performance was monitored to prevent overfitting.

### B. XGBoost Model

The XGBoost model was implemented as part of a two-stage framework to predict customer replenishment behavior. This framework consisted of:

1) A binary classification model to determine whether a repurchase was needed.
2) A multi-class classification model to predict the specific replenishment week for eligible cases.

*1) Binary Classification Model:* The binary classification model, implemented using XGBoost, identified whether a repurchase was needed:

- **Imbalanced Data Handling:** The `scale_pos_weight` parameter addressed the imbalance between "no repurchase" (label 0) and "repurchase needed" (labels 1–4).
- **Feature Engineering:** Features such as average days between purchases, product popularity scores, and week codes were utilized.
- **Optimization:** Hyperparameters such as tree depth and learning rate were tuned using cross-validation.
- **Evaluation Metrics:** Precision, recall, and F1-score were computed, with a focus on minimizing false negatives to ensure accurate detection of repurchase needs.

*2) Multi-Class Classification Model:* For cases flagged by the binary classification model, a multi-class classification model predicted the specific replenishment week:

- **Model Choice:** Random Forest was used for its ability to handle categorical and numerical features effectively.
- **Features:** Key features included average purchase intervals, popularity scores, and week codes.
- **Hyperparameter Tuning:** Parameters such as the number of trees (`n_estimators`) and maximum depth were fine-tuned using cross-validation.
- **Overfitting Mitigation:** The model focused on relevant data subsets to improve prediction accuracy and reduce overfitting.

*3) Combined Framework and Evaluation:* The two models were integrated into a pipeline:

- **Step 1:** The binary classification model determined whether a repurchase was required.
- **Step 2:** If a repurchase was predicted, the multi-class model predicted the replenishment week.

Performance was evaluated using precision, recall, and F1-score metrics, with results aligned to the competition's scoring system. This modular framework effectively handled dataset complexity and imbalances, achieving accurate and interpretable predictions.

## V. Discussion and Evaluation

This section evaluates the performance of the proposed models, using relevant metrics such as accuracy, precision, recall, and F1-score.

### A. LSTM Model Performance

TABLE I
LSTM Model Classification Report

| Class | Precision | Recall | F1-Score |
|---|---|---|---|
| 0 | 0.88 | 0.25 | 0.40 |
| 1 | 0.03 | 0.02 | 0.03 |
| 2 | 0.03 | 0.07 | 0.05 |
| 3 | 0.00 | 0.00 | 0.00 |
| 4 | 0.03 | 0.67 | 0.06 |
| **Accuracy** | 0.25 | | |

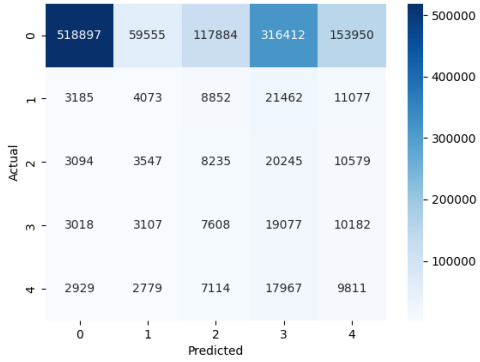


Fig. 1. Confusion Matrix for LSTM Predictions.

### B. XGBoost Model Performance

*1) Binary Classification Model:* The binary classification model, implemented using XGBoost, achieved an overall accuracy of 92%. Its precision, recall, and F1-scores for class 0 (no replenishment needed) were 0.84, 0.70, and 0.77, respectively, as shown in Table II. For class 1 (replenishment needed), these scores were 0.93, 0.97, and 0.95, respectively, demonstrating the model's ability to minimize false negatives and identify cases requiring replenishment effectively.

TABLE II
BINARY CLASSIFICATION MODEL RESULTS

| Class | Precision | Recall | F1-score | Support |
|---|---|---|---|---|
| 0 | 0.84 | 0.70 | 0.77 | 3442 |
| 1 | 0.93 | 0.97 | 0.95 | 14374 |
| Accuracy | 0.92 | | | |
| Macro Avg | 0.89 | 0.84 | 0.86 | 17816 |
| Weighted Avg | 0.91 | 0.92 | 0.91 | 17816 |

*2) Multi-Class Classification Model:* The multi-class classification model, implemented using Random Forest, achieved an overall accuracy of 32%, as presented in Table III. The class-specific precision, recall, and F1-scores ranged from 0.00 to 0.41, 0.00 to 0.66, and 0.00 to 0.51, respectively. While the model performed well for class 4 (longer replenishment intervals), it struggled significantly with intermediate classes (1–3), indicating a need for improvement in capturing subtle patterns in these intervals.

TABLE III
MULTI-CLASS CLASSIFICATION MODEL RESULTS

| Class | Precision | Recall | F1-score | Support |
|---|---|---|---|---|
| 0 | 0.00 | 0.00 | 0.00 | 445 |
| 1 | 0.20 | 0.18 | 0.19 | 399 |
| 2 | 0.22 | 0.24 | 0.23 | 522 |
| 3 | 0.21 | 0.16 | 0.18 | 529 |
| 4 | 0.41 | 0.66 | 0.51 | 965 |
| Accuracy | 0.32 | | | |
| Macro Avg | 0.21 | 0.25 | 0.22 | 2860 |
| Weighted Avg | 0.25 | 0.32 | 0.27 | 2860 |

*3) Combined Model Results:* The combined framework achieved an overall accuracy of 83%, effectively leveraging the strengths of both the binary and multi-class models. As shown in Table IV, the binary classification model maintained high precision (0.93) and recall (0.97) for class 0, while the multi-class model performed moderately well for class 4 (precision: 0.41, recall: 0.54). However, intermediate classes (1–3) exhibited lower scores, highlighting areas for improvement in feature engineering and model tuning.

TABLE IV
COMBINED MODEL RESULTS

| Class | Precision | Recall | F1-score | Support |
|---|---|---|---|---|
| 0 | 0.93 | 0.97 | 0.95 | 14374 |
| 1 | 0.20 | 0.10 | 0.13 | 745 |
| 2 | 0.22 | 0.16 | 0.19 | 787 |
| 3 | 0.21 | 0.11 | 0.15 | 724 |
| 4 | 0.41 | 0.54 | 0.47 | 1186 |
| Accuracy | 0.83 | | | |
| Macro Avg | 0.39 | 0.37 | 0.39 | 17816 |
| Weighted Avg | 0.81 | 0.83 | 0.82 | 17816 |

## VI. CONCLUSION

This study developed a predictive analytics framework for a global e-commerce platform to enhance customer experience by providing timely replenishment recommendations for frequently purchased items. By integrating transactional data, product attributes, and categorical hierarchies, a robust dataset was created to effectively model customer purchasing behavior.

We implemented and evaluated two predictive modeling approaches: Long Short-Term Memory (LSTM) networks and an XGBoost-based two-stage framework.

The LSTM model demonstrated its ability to capture sequential patterns in purchasing behavior. However, the dataset exhibited a significant class imbalance, with the majority of records belonging to class 0 (no replenishment required). This imbalance posed challenges for the model in correctly identifying minority classes, especially intermediate replenishment intervals. To address this issue, class weights were applied during training to penalize misclassification of underrepresented classes and improve the model's ability to generalize. Despite these efforts, the LSTM model achieved an overall accuracy of 25%, with relatively high precision for class 0 (0.88) but poor performance across minority classes, particularly for classes 1, 2, and 3. These results underscore the need for advanced strategies to handle imbalanced data, such as over-sampling, under-sampling, or using generative methods like SMOTE.

The XGBoost framework, combining binary and multi-class classification models, achieved better overall performance. The binary model excelled in identifying whether replenishment was required, with an accuracy of 92% and high precision and recall for both replenishment and non-replenishment cases. The multi-class model performed moderately well for longer replenishment intervals (class 4) but struggled with intermediate classes, resulting in an overall accuracy of 31%. The combined framework leveraged the strengths of both models, achieving an accuracy of 83%, effectively addressing the dataset's imbalanced nature for binary predictions but facing challenges in the multi-class setting.

Our findings indicate that a hybrid approach integrating LSTM and XGBoost could further optimize performance by combining the temporal strengths of LSTM with the structured data processing efficiency of XGBoost. Such an ensemble strategy could better handle the challenges posed by varying replenishment patterns and the inherent class imbalance.

Future work should focus on integrating additional data sources, such as real-time customer engagement metrics, weather, and promotions, to enhance predictive accuracy. Advanced feature engineering and experimentation with other deep learning architectures, such as attention mechanisms, could further improve model performance. Additionally, exploring ensemble methods, hybrid architectures, and advanced class balancing techniques may offer scalable and accurate solutions to meet the demands of a dynamic e-commerce environment.

In conclusion, this project demonstrates the potential of predictive analytics to transform e-commerce operations, providing actionable insights for inventory management, personalized marketing, and customer satisfaction. While the models showcased promising results, addressing class imbalance and optimizing predictions for intermediate replenishment intervals remain key areas for future research.

## VII. Additional Details

**-GitHub Repository Link:**

-Kaggle Team Name: 150230725_150230726

For any inquiries, you can reach out to the team members: -Ahmet Selim Bayram (bayramah20@itu.edu.tr) -Emre Günel (gunele20@itu.edu.tr).

## References


[1] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 785-794. DOI: 10.1145/2939672.2939785.

[2] P. J. Brockwell and R. A. Davis, *Introduction to Time Series and Forecasting*. Springer, 2016. DOI: 10.1007/978-3-319-29854-2.

[3] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735-1780, 1997. DOI: 10.1162/neco.1997.9.8.1735.

[4] Y. Koren, R. Bell, and C. Volinsky, "Matrix Factorization Techniques for Recommender Systems," *IEEE Computer*, vol. 42, no. 8, pp. 30-37, 2009. DOI: 10.1109/MC.2009.263.

[5] A. R. Jakkula, "Predictive Analytics in E-Commerce: Maximizing Business Outcomes," *Journal of Marketing & Supply Chain Management*, vol. 2, no. 2, pp. 1-3, 2023. DOI: 10.47363/JMSCM/2023(2)158.

[6] A. Ecevit, İ. Öztürk, M. Dağ, and T. Özcan, "Short-term sales forecasting using LSTM and Prophet based models in e-commerce," *Acta Infologica*, vol. 7, no. 1, pp. 59-70, 2023. DOI: 10.26650/acin.1259067.

[7] S. Rahman and M. S. U. Zaman, "Time series sales forecasting: A hybrid deep learning regularization approach," in *2024 3rd International Conference on Advancement in Electrical and Electronic Engineering (ICAEEE)*, 2024. DOI: 10.1109/ICAEEE62219.2024.10561831.

[8] J. Allard, "Customer Sequence Modeling," *Medium*, Aug. 18, 2023. [Online]. Available: https://medium.com/@jeffrey.m.allard/customer-sequence-modeling-4894786b6189. [Accessed: Jan. 12, 2025].