Personalized Shopping Recommendations: Predicting Customer Reorder from Instacart Data

Kelly Thackeray; Oumayma Ben Aoun; Phung Tran; Prajwal Parajuli

Introduction

Personalized product recommendations are central to modern e-commerce. Online grocery platforms, in particular, depend on effective recommendation systems to predict customer reorders, optimize basket composition, and improve user satisfaction. Unlike entertainment or retail recommendation domains, grocery shopping often display strong habitual patterns like milk, eggs, bread purchased weekly, or coffee restocked monthly. Predicting which items a customer will reorder in their next shopping trip is valuable for both the business and the customer where customer save time by receiving relevant suggestions, and business benefit from increased sales and stronger customer engagement.

The objective of this project is to develop a hybrid recommendation system that predicts which product a customer will buy in their next order. The system will use several machine learning and statistical methods together, each model serving a specific role.

- Implicit Matrix Factorization with Alternating Least Square (ALS): A collaborative filtering model that decomposes the customer-product interaction matrix into latent factors for users and items. In this project, ALS will be training on prior orders to generate candidate products for each customer, reflecting hidden purchase patterns.
- Item-Based Collaborative Filtering (Item-Item Similarity): A neighborhood based model that measures how often items are purchased together using similarity metrics using cosine or BM25 weighting. Here, it will identify products linked to a customer's recent purchases, such as cereal and milk or bread and butter, and add them to the candidate pool.
- Quadratic Discriminant Analysis (QDA): A statistical classification model that separates classes by estimating different covariance structures for each class. For this project, QDA will use features such as order frequency, time since last purchase, and product popularity to assign reorder probabilities to candidate products.
- Light Gradient Boosting Machine (LightGBM) with LambdaRank Objective: A gradient boosting framework that builds decision trees and optimizes ranking through pairwise comparisons. In this project, LightGBM will combine ALS scores, Item-Item similarities and QDA probabilities, along with engineered features, to generate the final ranked list of Top-N product recommendation.

By combining these approaches, the system will generate Top-N ranked recommendation for each customer. The project will evaluate results using a combination of ranking metrics such as Recall at K, Hit Rate at K, Normalized Discounted Cumulative Gain, and Mean Average Precision; to measure how well the system predicts and orders the correct products. Probability metric, including the Brier Score and Expected Calibration Error, will assess the quality of probability estimates from models such as QDA. In addition, business metrics such as Coverage, Diversity, and Long-tail Share will ensure that recommendation are varied, balanced across the catalog, and practically useful. Together, these metrics provide a balanced view of technical accuracy and business value.

Data

This project uses the **Instacart Online Grocery Basket Analysis Dataset**, a publicly available dataset released through Kaggle for research purposes. The dataset originates from Instacart, a U.S. based online grocery delivery service, and contains anonymized transaction logs of customer purchases. All personally identifiable information was removed before release, and the dataset was structured to support the Kaggle competition on predicting which products a user will reorder.

The dataset is designed to address the problem of reorder prediction given a user's purchase history and product attributes, the goal is to predict which items will appear in the user's next order. This problem directly supports recommender system development, inventory management, and customer personalization.

The dataset consists of six interrelated files:

- **orders.csv**: Contains metadata for all customer orders. Variables include:
  - `order_id`: unique identifier for the order
  - `user_id`: unique identifier for the customer
  - `eval_set`: indicates whether the order belongs to the "prior," "train," or "test" set
  - `order_number`: the sequence number of the order for each user
  - `order_dow`: day of the week the order was placed
  - `order_hour_of_day`: hour of the day the order was placed
  - `days_since_prior_order`: number of days since the previous order
    This file contains about **3.4 million rows** and **7 variables**.
- **order_products___prior.csv**: Records products from users' previous orders. Variables Include:
  - `order_id`: unique identifier of the order
  - `product_id`: unique identifier of the product
  - `add_to_cart_order`: the sequence in which the product was added to the cart
  - `reordered`: binary indicator (1 if product was ordered before, 0 otherwise)
    This file contains about **32.4 million rows** and **4 variables**.
- **order_products___train.csv**: Provides labeled training data for model development.

Contains the same variables as `order_products_prior.csv` but limited to the training set. This file has about **1.3 million rows** and **4 variables**.

- **products.csv**: Contains product-level metadata:
  - `product_id`: product identifier
  - `product_name`: name of the product
  - `aisle_id`: identifier for the aisle
  - `department_id`: identifier for the department
    This file has **49,688 rows** and **4 variables**.
- **aisles.csv** — Defines the grocery aisles. Variables:
  - `aisle_id`: identifier for the aisle
  - `aisle`: aisle name
    This file contains **134 rows**.
- **departments.csv** — Defines the grocery departments. Variables:
  - `department_id`: identifier for the department
  - `department`: department name
    This file contains **21 rows**.

The dataset is large and heterogeneous, containing over **3 million orders** and nearly **50,000 products**. Its structure allows modeling of sequential purchasing behavior, user-level and product-level features, and hierarchical product categories. Because the interactions are implicit (a product is either purchased or not), the dataset is well suited to evaluating algorithms for candidate generation, classification, and ranking in a recommendation system. Utilizing these dataset, the following questions will guide the analysis:

- Does the time gap between consecutive orders (`days_since_prior_order`) significantly influence the likelihood that a customer will reorders items?
- Are frequently purchased products more likely to be reordered than less popular products, and does this pattern hold across different aisles and departments?
- How accurately can reorder probability be predicted for items in the most recent order, based on customer history and product features?
- Which aisles or departments demonstrate higher reorder rates, and can these patterns inform category-level strategies (recommendation)?
- Does incorporating latent factors from ALS improve reorder prediction compared to using only historical frequencies?
- Do advanced ranking models (e.g., LightGBM with LambdaRank) outperform simpler classifiers (e.g., Logistic Regression or QDA) in top-K recommendation quality?
- Are reorder patterns consistent across time-of-day or day-of-week, and do these temporal features enhance prediction accuracy?

Methods

Implicit Matrix Factorization with Alternating Least Squares (ALS)

Alternating Least Square (ALS) is a collaborative filtering method that predicts which products a customer is likely to purchase by learning hidden relationship between customers and products. It begins with a large, sparse interaction matrix $R \in \mathbb{R}^{m \times n}$, where m is the number of customers, n is the number of products, and each entry r_{ui} indicated whether customer u purchased product i. The algorithm factorizes this matrix into two lower dimensional representation $U \in \mathbb{R}^{m \times k}$, which encodes customer latent factors, and $V \in \mathbb{R}^{n \times k}$, which encodes product latent factors. The approximation can be expressed as :

$$R \approx UV^{\top}$$

where, the dot product $U_u^{\top} V_i$ serves as a predicted preference score for customer u and product i.

Because the Instacart dataset reflects implicit feedback (purchases versus non-purchases rather than explicit ratings), the implicit version of ALS is applied. In this case, the algorithm assigns confidence weights to each observation, defined as:

$$c_{ui} = 1 + \alpha r_{ui}$$

where α is a scaling factor that increases the importance of repeated purchases. The optimization objective is then:

$$\min_{U,V} \sum_{u,i} c_{ui}(p_{ui} - U_u^{\top} V_i)^2 + \lambda\Big(\|U\|^2 + \|V\|^2\Big),$$

where $p_{ui} \in \{0, 1\}$ indicates whether customer u purchased product i, and λ is a regularization parameter to prevent over fitting. The algorithm alternates between fixing customer vectors to solve for product vectors, and fixing product vectors to solve for customer vectors, until convergence is achieved.

ALS is an appropriate choice for this project because it can model large-scale, implict interaction data effectively. In the context of Instacart, it uncovers latent purchasing preferences, such as a customer's tendency to reorder staple items or favor particular product categories. Within the project pipeline, ALS serves as the candidate generation step, narrowing the nearly 50,000 available products to a focused set of personalized suggestions that subsequent models can refine and rank.

Item-Item Similarity with k Nearest Neighbors (KNN)

Item-based collaborative filtering is a neighborhood-based model that identifies relationships between products by measuring how often they are purchased together. Using similarity metrics such as cosine similarity or BM25 weighting, we can compute an item–item similarity matrix that captures the strength of association between products. For example, if cereal and milk or bread and butter frequently co-occur in customer baskets, the model will detect these links and add them to the candidate pool for recommendation.

Similarity between two items j and k can be defined as:

$$s_{jk} = \frac{\text{co-occurrence}(j,k)}{\sqrt{\text{freq}(j) \cdot \text{freq}(k)}}$$

where co-occurrence(j,k) is the number of baskets containing both items, and freq(j) is the number of baskets containing item j. This normalization avoids bias toward highly popular products and ensures that similarity reflects meaningful association rather than raw frequency.

Given a target product, the k Nearest Neighbors (KNN) algorithm ranks all other items by similarity score and retrieves the top K nearest neighbors. These outputs serve multiple purposes in the system:

- Cold-history users: Even with limited purchase history, KNN can recommend related items without requiring a full behavioral profile.
- Explainable recommendations: The system can communicate suggestions in a transparent way, such as "Because you bought X, you may like Y," which strengthens user trust.
- Complementary discovery: Beyond substitutes, KNN surfaces complementary items frequently bought together (e.g., coffee and filters), supporting cross-selling opportunities.

From a statistical perspective, similarity scores can be viewed as normalized measures of association strength, providing interpretable insights into catalog structure. High-similarity clusters indicate substitutable products (e.g., different brands of the same staple), while diffuse clusters suggest broader variety (e.g., snack foods). Such findings can also inform merchandising and promotion strategies.

Because KNN does not rely on explicit training, it updates easily as new interactions occur. The method requires a large number of observations to produce stable similarity scores. In smaller datasets, item–item similarity may be noisy or biased toward popular products, but in larger datasets the co-occurrence patterns are strong enough to be reliable.

Quadratic Discriminant Analysis (QDA)

Quadratic Discriminant Analysis (QDA) is a classification method that extends Linear Discriminant Analysis by allowing each class to have its own covariance matrix. This flexibility enables QDA to model quadratic, curved decision boundaries between classes, making it more suitable for problems in which feature interactions lead to nonlinear separations. In contrast, logistic regression assumes linear boundaries, which may be too restrictive for complex purchasing behaviors.

Formally, let $X \in \mathbb{R}^p$ denote the predictor vector and $Y \in \{1, \ldots, K\}$ the class label. For each class k, QDA estimates the class prior π_k, mean vector μ_k, and covariance matrix Σ_k. Using Bayes' theorem, the posterior probability that an observation belongs to class k is:

$$P(Y = k \mid X = x) = \frac{\pi_k N(x \mid \mu_k, \Sigma_k)}{\sum_{l=1}^{K} \pi_l N(x \mid \mu_l, \Sigma_l)},$$

where $N(x \mid \mu_k, \Sigma_k)$ is the multivariate normal density for class k. The decision rule assigns each observation to the class with the highest posterior probability.

QDA is an appropriate choice for this project because the reorder decision in the Instacart dataset is influenced by nonlinear interactions between features such as order recency, purchase frequency, and product popularity. By modeling these interactions through quadratic boundaries, QDA can provide more accurate probability estimates than linear methods. While QDA can be prone to over fitting in high-dimensional settings, the large sample size of the Instacart dataset helps reduce this risk, especially with appropriate feature selection.

In the project pipeline, QDA serves as a probability estimator for classification. The predicted reorder probabilities can then be used as input features for the downstream ranking model, ensuring that both individual-level purchase tendencies and nonlinear feature effects are captured in the overall recommendation system.

Learning-to-Rank with Gradient Boosted Trees (LightGBM / XGBoost)

Gradient Boosted Decision Trees (GBDTs) are a powerful ensemble method that builds decision trees in sequence, where each tree is trained to correct the residual errors of the previous ones. This iterative process results in a strong predictive model capable of capturing complex, non-linear relationships. Popular frameworks such as LightGBM and XGBoost provide efficient, scalable implementations of GBDTs, along with objectives specifically tailored for ranking tasks.

In the Learning-to-Rank (LTR) setting, the objective is not to predict absolute probabilities or ratings, but rather to produce scores that reflect the relative ordering of items. This project leverages two specialized ranking objectives: the pairwise objective in XGBoost and the LambdaRank objective in LightGBM.

Ranking Objectives

- Pairwise Objective (XGBoost): The model learns by comparing pairs of items within the same user's candidate set. If item i is more relevant than item j, the model is trained to assign a higher score to i. The optimization problem can be expressed as:

$$L = \sum_{(i,j)} \ell(y^i - y^j)$$

  where (i, j) are item pairs with different relevance labels, $\hat{\boldsymbol{y}}_i$ is the predicted score for item i, and ℓ is a differentiable loss (e.g., logistic loss).

- LambdaRank Objective (LightGBM): Extends the pairwise framework by weighting training pairs according to their impact on ranking metrics such as NDCG. Pairs that affect the top of the ranking list receive greater emphasis, ensuring that optimization aligns directly with evaluation goals like top-K performance.

This design allows the model to learn nuanced ranking functions that balance collaborative signals, user behavior, item attributes, and contextual factors, while directly optimizing for

metrics most relevant to recommendation quality.

Literature Review

Recommendation systems are central to e-commerce platforms, helping customers discover relevant products and improving overall sales. In online grocery, where purchases are frequent and often repetitive, accurate reorder prediction and ranking are especially important. Prior work on collaborative filtering and classification offers several methods that directly inform this project.

Hu, Koren, and Volinsky (2008) introduced a weighted Alternating Least Squares (ALS) approach designed for implicit feedback, where purchases or clicks stand in for explicit ratings. Their method assigns different confidence weights to observed and unobserved interactions, addressing the uncertainty inherent in missing values. This approach is well suited to grocery reorder prediction, where the signal is primarily purchase history rather than explicit ratings, and frequent reorders can reveal strong latent preferences.

Another influential approach is item–item collaborative filtering, described by Linden, Smith, and York (2003) in their work on Amazon's recommender system. This method identifies associations between items based on co-occurrence in shopping baskets, producing recommendations that are both effective and explainable. In grocery data, item–item similarity naturally captures staple pairings such as bread and butter or cereal and milk, supporting candidate generation for next-basket prediction.

For classification tasks, logistic regression has long been used to estimate the probability of binary outcomes, offering interpretability through feature weights that map directly to reorder likelihood. Quadratic Discriminant Analysis (QDA) provides additional flexibility by allowing class-specific covariance matrices, enabling curved decision boundaries that can capture nonlinear feature interactions such as recency, frequency, and product popularity. These methods are relevant to predicting the likelihood of reorders in the Instacart dataset, where such interactions strongly influence customer behavior.

Finally, Gradient Boosted Decision Trees (GBDTs) have emerged as state-of-the-art models for structured prediction tasks due to their ability to capture complex feature relationships. LightGBM (Ke et al., 2017) provides a scalable implementation, and its LambdaRank objective directly optimizes ranking metrics such as Normalized Discounted Cumulative Gain (NDCG). By emphasizing the quality of top-ranked predictions, this approach is particularly well suited for the final stage of grocery recommendation systems, where producing an ordered list of items most relevant to each customer is the ultimate goal.

Together, these methods demonstrate that collaborative filtering, classification, and ranking models each contribute distinct strengths. This project integrates them into a hybrid pipeline tailored to the Instacart dataset, combining candidate generation with probability estimation and final ranking to address the challenges of reorder prediction in online grocery.

Tentative Schedule and Task Distribution

The responsibilities have been distributed among team members with consideration of the interdependencies between models. Prajwal is assigned to the development of the Alternating Least Squares (ALS) model, which generates user–item latent factors for candidate generation and provides input features for downstream models. Kelly is responsible for implementing the Item–Item Collaborative Filtering (KNN) approach, which computes co-occurrence similarities and contributes additional candidate signals and features for ranking. Phung is tasked with Quadratic Discriminant Analysis (QDA), which depends on the outputs of ALS and KNN, as it incorporates latent factors and similarity scores into its feature set to produce reorder probabilities. Finally, Oumayma is assigned the Gradient Boosted Decision Trees (LightGBM with LambdaRank), which synthesizes the outputs of all preceding models; ALS factors, KNN similarities, and QDA probabilities, to generate the final ranked recommendations.

Each member will initially work independently within a separate Jupyter Notebook to ensure modular development and maintain clarity. Subsequently, the code will be consolidated and standardized as a group effort. Data preparation will be undertaken collaboratively from September 20 to October 5 to ensure a uniform cleaned dataset for all models. Model development will proceed in overlapping phases: ALS and KNN (October 1–10), QDA (October 8–15, following the completion of ALS and KNN), and LightGBM (October 12–18, once QDA outputs are available). The final phase, scheduled between October 18–23, will involve integration, debugging, and refinement of visualizations and the report in preparation for submission.

Reference

Kaggle. (n.d.). *Instacart online grocery basket analysis dataset*. Retrieved from: https://www.kaggle.com/datasets/yasserh/instacart-online-grocery-basket-analysis-dataset

Li, X. (2025). Design of a web content personalized recommendation system based on collaborative filtering improved by combining k-means and LightGBM. *Journal of Web Engineering, 24*(2), 267–290. https://doi.org/10.13052/jwe1540-9589.2425

Bilbao, J., & Bilbao, I. (2024). Application of linear discriminant analysis and k-nearest neighbors techniques to recommendation systems. *WSEAS Transactions on Information Science-and Applications, 21, 160-168*. https://doi.org/10.37394/23209.2024.21.16

Fiarni, C., & Maharani, H. (2019). Product recommendation system design using cosine similarity and content-based filtering methods. *International Journal of Information-Technology and Electrical Engineering, 3*(2), 42–48. https://doi.org/10.22146/ijitee.45538

Jannach, D., Lerche, L., & Zanker, M. (2018). Recommending based on implicit feedback. In P. Brusilovsky & D. He (Eds.), *Social information access* (Lecture Notes in Computer Science, Vol. 10100, pp. 510-569)$. Springer. https://doi.org/10.1007/978-3-319-90092-6_14