E-commerce User Purchase Behavior Prediction Using the XGBoost Algorithm

Henry Yijian Huang

香港中文大學（深圳）
The Chinese University of Hong Kong, Shenzhen

E-commerce User Purchase Behavior Prediction Using the XGBoost Algorithm

Content

1. Project Overview

1.Project Background

With the rapid development of the Internet, the scale of e-commerce consumers and products continues to expand. While providing convenience for users, e-commerce has also led to the problem of information overload. Every day, millions of users browse products, add them to favorites, and place them in shopping carts on the platform. However, only a very small percentage of users ultimately complete the payment, meaning a large amount of potential purchase intent is lost during the decision-making process. How to utilize massive user behavior data to mine true consumer intent and identify users with high conversion probabilities in advance has become a core breakthrough for e-commerce platforms to increase GMV and enhance user stickiness. This allows for targeted marketing, personalized recommendations, and efficient resource allocation.

Against this background, this project utilizes the XGBoost algorithm to build a purchase prediction model based on historical user behavior data. By mining the latent features of the data, the model provides purchase predictions to help the e-commerce platform implement targeted marketing and significantly improve conversion rates.

2.Research Objectives

This project aims to build an XGBoost-based purchase behavior prediction model using desensitized e-commerce user behavior data. By systematically extracting and modeling user behavior sequences, product category attributes, and temporal features, the model accurately predicts the user's probability of purchasing a target product. It also addresses key technical issues such as data class imbalance and data leakage to achieve stable prediction results with high precision and recall. Furthermore, through model interpretability analysis, the project extracts the core behavioral patterns that influence purchases, ultimately providing practical data evidence and decision support for targeted marketing, personalized recommendations, and optimal allocation of operational resources.

3.Data Description

This project uses a sampled dataset of real, desensitized e-commerce user behaviors (randomly sampled at 1/16 of the original data), which ensures strong business authenticity and representativeness. The raw data primarily contains five key fields, as detailed in Table 1. There are approximately 707,919 total behavioral records.

After deduplication, there are 599,116 unique (user, item) sample pairs. The sample distribution is highly imbalanced, containing 6,697 positive purchase samples (accounting for 1.12%) and 592,419 negative non-purchase samples (accounting for 98.88%). Table 3 shows an example of the sampled dataset. The data exhibits obvious temporal and behavioral trajectory characteristics, making it suitable for sequential feature modeling and purchase prediction research.

Table 1: Raw Data Fields

Field Name	Description
user_id	Unique identifier for the user (similar to an ID card number)
item_id	Unique identifier for the item
behavior_type	Type of behavior: 1=Browse, 2=Favorite, 3=Add to cart, 4=Purchase
item_category	Category ID of the item (e.g., food, clothing, electronics)
time	Time the behavior occurred, accurate to the hour (format: 2014-12-06 02)

Table 2: Data Statistics

Data Scale	Value
Total raw behavioral records	~707,919
Unique (user, item) pairs	599,116 pairs
Pairs with purchase behavior	6,697 (1.12%)
Pairs without purchase behavior	592,419 (98.88%)

Table 3: Data Sample

user_id	item_id	behavior_type	item_category	time
77374787	22916568	1	6059	2014-12-16 06
77374787	22916568	1	6059	2014-12-16 06
77374787	22916568	2	6059	2014-12-16 06
77374787	22916568	1	6059	2014-12-16 06

It is worth noting that, as shown in Table 2, the e-commerce user behavior data used in this project has a significant class imbalance problem. During subsequent model building, if the model simply classifies all samples as "non-purchase," it can still achieve a high accuracy of 98.88%. However, this result would completely fail to identify actual buyers and holds no practical business value. Therefore, targeted sample

balancing strategies must be adopted in the modeling process. At the same time, accuracy should be discarded as the core evaluation metric. Instead, metrics more suitable for imbalanced data, such as Precision, Recall, F1 score, and AUC-PR, should be used for model evaluation.

2. Data Preprocessing and Behavior Serialization

2.1 Data Grouping

The raw data consists of single behavioral records that fail to capture the complete user decision-making path. In this project, records are grouped using the combination of "user_id" and "item_id" as the unique key. The behaviors are then sorted chronologically and concatenated into strings to construct a complete behavioral trajectory.

For example, suppose a user's behavioral records for a specific item are scattered across five rows of data:

Table 4: Example of Behavioral Records

Time	Behavior
12 月 1 日 10 点	Browse (1)
12 月 1 日 15 点	Browse (1)
12 月 2 日 09 点	Add to Cart (3)
12 月 3 日 14 点	Browse (1)
12 月 3 日 20 点	Purchase (4)

By grouping the data by (user, item) and concatenating all behaviors in chronological order, the resulting behavior string is "11314" (Browse → Browse → Add to Cart → Browse → Purchase).

In this way, each (user, item) pair is transformed into a complete "behavioral trajectory," serving as the foundation for subsequent feature extraction. Example results are shown in Table 5 below.

Table 5: Examples of Behavior Strings Grouped by (User, Item)

user_id	item_id	behavior_str
5694160	140163476	111411
7207480	230738799	1314

10887844	343413817	1143
12931864	134837632	111411
15511710	399858977	1143

2.2 **Data Leakage Prevention**

To prevent the model from directly accessing the purchase labels, the behavior sequences are strictly truncated prior to the purchase event to serve as model inputs:

Sequences with purchase behavior: Only the behavior prefix preceding the purchase is retained.

Sequences where the first behavior is a purchase: The behavior sequence is left empty, and last_action is marked as 0.

Sequences without purchase behavior: The complete behavior sequence is retained.

Based on the above data processing steps, a total of 599,116 valid samples were ultimately constructed, comprising 6,697 purchase samples and 592,419 non-purchase samples.

3. **Feature Engineering**

Feature engineering is a core component of machine learning model construction. Its primary objective is to transform raw, unstructured business data into model-understandable input variables with business significance, which directly determines the final performance of the model. The construction of the feature system in this study aligns e-commerce business logic with data structure characteristics, systematically extracting features from three main dimensions: behavior sequence, item category, and temporal environment. A total of 17 core predictive features are constructed, comprehensively covering user decision paths, category differences, and behavioral temporal patterns, detailed as follows:

3.1. **Behavior Sequence Features**

Behavior sequence features are extracted directly from the "pre-purchase behavior sequence," describing what the user has done regarding the item. They are primarily used to quantify the user's level of attention and intention intensity toward the item, serving as the most direct signal for predicting purchase behavior. Table 6 provides a detailed description of the extracted behavior sequence features.

Table 6: Behavior Sequence Features

Feature Name	Description and Example
seq_len (Sequence Length)	The total number of operations performed before purchase. For instance, the length of the sequence "113" is 3.
count_browse (Browse Count)	The number of times "1" appears in the sequence. For example, in "113", count_browse = 2.
count_fav (Favorite Count)	The number of times "2" appears in the sequence.
count_cart (Add to Cart Count)	The number of times "3" appears in the sequence. Adding to the cart is considered the strongest signal of purchase intent.
last_action (Last Action)	The final character in the sequence. If the last action is adding to the cart (3), the probability of purchase is highly elevated.
time_diff_hours (Conversion Time)	The number of hours from the first browse to the first add-to-cart action. Missing values are imputed with -1.

3.2. **Item Category Features**

Users' purchasing habits vary significantly across different product categories. For instance, the conversion rate for food items is typically much higher than that for high-value electronics. We calculated the following statistical features at the category level, as outlined in Table 7.

Table 7: Item Category Features

Feature Name	Description
cat_sample_count	The total number of behavioral records for items in this category (measuring category popularity).
cat_sample_ratio	The proportion of behaviors in this category relative to the entire database.
cat_purchase_rate	The historical purchase rate of this category (Target Encoding)—an important reference baseline for model prediction.
cat_avg_time_buy	The average time (in hours) taken by users from the initial behavior to purchase within this category.
z_score	The deviation of the current user's activity in this category from the average level (standardized score).

It should be noted that in e-commerce data, item categories exist as discrete numerical IDs, which function as high-cardinality categorical features and cannot be directly fed into the model for effective learning. To enable the model to understand the purchase discrepancies between various categories, this project employs Target Encoding to convert categorical variables into numerical representations. The core methodology of Target Encoding is replacing the categorical variable with the statistical value of the target variable for that specific category.

In this project, the historical purchase rate of each category is used to replace the original category ID. For example, if the historical purchase rate for category 3252 is 0.8% and for category 9614 is 1.76%, the encoded model can directly recognize the purchase probability variations between categories, significantly enhancing the model's ability to learn category-specific attributes.

To prevent data leakage and safeguard model generalization, target encoding is computed based solely on the training set. The test set is strictly excluded from the encoding statistical process, ensuring rigorous isolation between the training and evaluation phases, thereby rendering the model results valid and reliable.

3.3. **Temporal Environment Features**

The specific times at which user behaviors occur can also reflect purchase intentions. For example, studies suggest that consumers often engage in impulsive consumption late at night. Concurrently, intensive, high-frequency browsing is often a precursor to an impending purchase. Based on these insights, we constructed temporal environment features. Temporal features capture user behavior density, decision cycles, and impulsive consumption tendencies, effectively extracting implicit purchase intents. The detailed temporal environment features are presented in Table 8.

Table 8: Temporal Environment Features

Feature Name	Description
last_active_hour	The hour of the day the last behavior occurred (0-23).
time_span_hours	The total hours spanned by the entire behavior sequence (a longer span indicates a more pronounced "comparative shopping" pattern).
is_late_night	Whether the last behavior occurred in the early morning between 0-5 AM (0=No, 1=Yes).
click_density_1h	Based on the timestamp of the last behavior, the number of behaviors recorded in the past 1 hour.
click_density_6h	The number of behaviors recorded in the past 6 hours.
click_density_24h	The number of behaviors recorded in the past 24 hours.

It is important to note that multicollinearity exists between the features is_late_night and last_active_hour (since late-night implicitly equates to hours < 6). During the subsequent modeling phase, Variance Inflation Factor (VIF) testing can be applied to determine whether to retain only one of these features.

In summary, based on these three major dimensions, a total of 17 feature variables were extracted, as illustrated in Figure 1.

user_id	item_id	em_catego	seq_len	ount_brows	count_fav	count_cart	ast_actio	e_diff_ho	sample_co	sample_ra	purchase_	avg_time	z_score	t_active_	he_span_ho	late_nig	ck_density	ck_density	k_density
6118	2734042	9614	1	1	0	0	1	-1	1303	0.0022	6515732924	7.7391	12506022125597	0	0	1	1	1	1
6118	12929262	12304	1	1	0	0	1	-1	297	0.0005	3030303030	3.4444	-0.116684881	18	0	0	1	1	1
6118	61595610	3252	1	1	0	0	1	-1	1622	0.0027	0147965474	26.9231	-0.438710968	12	0	0	1	1	1
6118	80007528	10037	1	1	0	0	1	-1	228	0.0004	8596491222	0	12285504360844	11	0	0	1	1	1
6118	118314443	3798	1	1	0	0	1	-1	1436	0.0024	3203342611	33.5455	-0.455716441	22	0	0	1	1	1
6118	159077284	11623	1	1	0	0	1	-1	2116	0.0035	7599243885	17.037	-0.240070366	23	0	0	1	1	1
6118	182126438	10037	2	2	0	0	1	-1	228	0.0004	8596491222	0	12285504360844	11	0	0	2	2	2
6118	192770159	10037	1	1	0	0	1	-1	228	0.0004	8596491222	0	12285504360844	23	0	0	1	1	1
6118	213677881	9614	1	1	0	0	1	-1	1303	0.0022	6515732924	7.7391	12506022125597	21	0	0	1	1	1
6118	263180346	12304	1	1	0	0	1	-1	297	0.0005	3030303030	3.4444	-0.116684881	21	0	0	1	1	1
6118	298746900	11623	1	1	0	0	1	-1	2116	0.0035	7599243885	17.037	-0.240070366	11	0	0	1	1	1
6118	366793467	9614	1	0	1	0	2	-1	1303	0.0022	6515732924	7.7391	12506022125597	23	0	0	1	1	1
7528	3844641	5004	1	1	0	0	1	-1	13	0	9230769230	0	06409888369735	18	0	0	1	1	1
7528	10237913	3942	1	1	0	0	1	-1	708	0.0012	3615819209	19.8462	-0.277736283	21	0	0	1	1	1
7528	23151106	10392	1	0	0	1	3	-1	4783	0.008	5897971984	27.4615	43146411896268	22	0	0	1	1	1
7528	37046277	5004	1	1	0	0	1	-1	13	0	9230769230	0	06409888369735	18	0	0	1	1	1
7528	64006907	3424	1	1	0	0	1	-1	2550	0.0043	9588235294	10.8846	-0.394415091	15	0	0	1	1	1

Figure 1: Feature Extraction Results

4. Label Definition

The primary objective of this project is to predict whether a user will purchase a product based on behavioral interaction data. This task is formulated as a binary classification problem, where the target label indicates whether a purchase occurred for a specific (user, item) pair. Specifically, a label of 1 (positive sample) is assigned if the behavior type is a purchase (behavior_type=4), while a label of 0 (negative sample) is assigned if no purchase occurred. Based on the provided dataset, the statistical distribution of these labels is summarized in Table 9.

Table 9: Label Statistics

Label	Count
0 (Non-purchase)	592,419 pairs (98.88%)
1 (Purchase)	6,697 pairs (1.12%)

5. XGBoost Model Construction

5.1. Introduction to the XGBoost Model

1. XGBoost Algorithm Principles

Boosting Tree is an ensemble learning method that constructs a strong learner by combining multiple weak learners, typically decision trees. The fundamental concept involves iteratively training a series of decision trees, where each subsequent tree is trained on the prediction errors of the previous one to progressively improve predictive accuracy. The mathematical expression is as follows:

$$f_M(x) = \sum_{m=1}^{M} T(x, \Theta_m)$$

The model uses a forward stagewise algorithm, where the m-th step is defined as:

$$f_m(x) = f_{m-1}(x) + T(x, \Theta_m)$$

where T() represents the decision tree, M is the number of trees, and Θ denotes

the parameters of the decision tree.

For classification tasks, T() is a binary classification tree; for regression tasks, it is a binary regression tree.

Gradient Boosting Decision Tree (GBDT) is a specific implementation of the boosting tree that guides the generation of each new tree using the gradient of the loss function, enabling the model to converge more efficiently to the optimal solution. In each iteration, GBDT fits the gradient of the residuals between current predictions and actual values to minimize the loss function.

XGBoost (eXtreme Gradient Boosting) is an optimized implementation of GBDT characterized by higher computational efficiency and stronger regularization capabilities. XGBoost improves upon GBDT in several key areas:

Second-order Derivative Information: XGBoost incorporates second-order derivatives in the objective function to estimate the gradient more accurately, thereby accelerating convergence.

Regularization: An explicit regularization term is added to the objective function to control model complexity and effectively prevent overfitting.

Parallel Computing: XGBoost supports feature-level parallel computing, utilizing modern hardware capabilities to significantly reduce training time.

Missing Value Handling: The algorithm has a built-in mechanism to automatically learn the optimal split direction for missing values.

The core logic of XGBoost can be compared to a "joint consultation" by multiple experts: while a single decision tree has limited judgment, multiple trees sequentially correct prior errors and output a weighted synthesis for robust prediction. Its performance on structured data has made it a mainstream choice in both industry and data competitions.

2. **Rationale for Selection**

XGBoost was selected as the core algorithm due to its high precision, training speed, stability, and interpretability in structured data scenarios. It effectively addresses the requirements of purchase prediction tasks. Furthermore, XGBoost supports custom class weights and flexible loss function settings, which are essential for handling the extreme class imbalance in this project. Its built-in regularization and parallel computing capabilities allow for rapid training on large-scale data while preventing overfitting and providing clear feature importance for business interpretation.

5.2. **Dataset Partitioning**

To avoid distribution bias caused by the high class imbalance, this project utilizes stratified sampling (stratify=label) for dataset partitioning. Unlike simple random splitting, stratified sampling ensures that the proportion of positive purchase samples remains consistent across the training, validation, and test sets (at 1.12%). This ensures the rigor and comparability of model evaluations. The dataset partitioning is summarized in Table 10.

Table 10: Dataset Partitioning

Dataset	Ratio / Row Count	Purpose
Training Set	72% / 431,362 rows	Model learning of patterns
Validation Set	8% / 47,930 rows	Monitoring for overfitting and determining early stopping
Test Set	20% / 119,824 rows	Final performance evaluation on unseen data

5.3. **Handling Class Imbalance**

The dataset exhibits severe distribution imbalance, with positive samples (purchases) representing only 1.12% of the data. Training on the raw distribution would cause the model to favor the majority class (non-purchases) to achieve high nominal accuracy, thereby failing to identify actual buyers.

To resolve this, the project applies a class weighting strategy using the scale_pos_weight parameter in XGBoost. This increases the weight of positive samples during training, forcing the model to focus on the minority class.

Setting: scale_pos_weight = total negative samples / total positive samples ≈ 88.46.

Objective: This configuration penalizes the misclassification of a buyer as heavily as 88 misclassifications of a non-buyer, ensuring the model prioritizes positive sample identification.

5.4. Core Model Parameter Configuration

The parameter design for the XGBoost model focuses on e-commerce purchase scenarios, class imbalance management, and overfitting prevention. The configuration is divided into fixed base parameters and core tuned parameters, as shown in Table 11.

Table 11: Core Model Parameters

Parameter	Setting	Description
n_estimators	500	Maximum number of trees; actual optimal iteration was 208 rounds using early stopping.

max_depth	6	Maximum depth of a tree to control complexity and prevent overfitting.
learning_rate	0.05	Step size shrinkage to enhance convergence stability.
subsample	0.8	Row sampling ratio to enhance model diversity and generalization.
colsample_bytree	0.8	Column sampling ratio to reduce overfitting risks from feature redundancy.
scale_pos_weight	88.46	Weight ratio to address class imbalance.
eval_metric	aucpr	Validation metric prioritized for imbalanced data performance.
early_stopping_rounds	30	Terminates training if validation performance does not improve for 30 rounds.
random_state	2025	Fixed seed for result reproducibility.
n_jobs	-1	Parallel training using all available CPU cores.

5.5 **Hyperparameter Optimization and Tuning Results**

1. Parameter Optimization Strategy

Hyperparameter optimization was conducted using the Optuna framework with a Tree-structured Parzen Estimator (TPE) sampling strategy. Compared to grid or random search, this Bayesian approach dynamically adjusts search intervals based on historical trials to locate optimal combinations more efficiently.

A total of 50 independent trials were conducted across 8 core hyperparameters affecting complexity, fitting capacity, and regularization:

max_depth：[3, 8]

learning_rate：[0.01, 0.2] (log-uniform)

subsample：[0.5, 1.0]

colsample_bytree：[0.5, 1.0]

min_child_weight：[1, 20]

gamma：[0.0, 1.0]

reg_alpha (L1 Regularization)：[1e-8, 10.0] (log-uniform)

reg_lambda (L2 Regularization)：[1e-8, 10.0] (log-uniform)

The entire tuning process strictly followed the same dataset partitioning, feature engineering, and training procedures as the baseline model. This was done to avoid data leakage and to ensure that the optimization results were both fair and reliable.

2. Parameter Optimization Visualization

To visually demonstrate the tuning process and its effects, a visualization analysis was performed, with the core charts presented in Figure 2. The left panel of Figure 2 displays the Optuna Bayesian optimization convergence curve, illustrating the variation in validation AUC-PR across 50 trials. The blue scatter points represent individual trial results, the orange curve indicates the historical best value, and the gray dashed line denotes the baseline model performance. The optimization process shows a rapid improvement followed by stable convergence, with the optimal result significantly exceeding the baseline, confirming that Bayesian tuning effectively enhanced model performance. The right panel provides a hyperparameter importance plot, reflecting the degree of influence each parameter has on the model's effectiveness. The results indicate that learning_rate, max_depth, and subsample have the most significant impact on this purchase prediction task. This suggests that tree depth, learning step size, and sampling strategies are critical factors in determining model performance, providing a clear direction for further optimization.

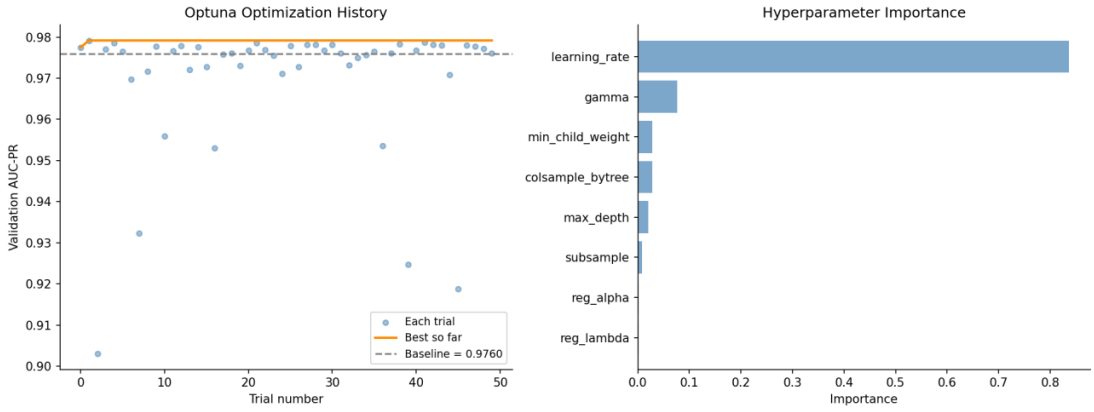

Figure 2: Optuna Bayesian Optimization Convergence Curve and Hyperparameter Importance Plot

3. Parameter Optimization Results and Performance Comparison

After 50 rounds of Bayesian search, the final optimal hyperparameter combination was obtained as shown in Table 12.

Table 12: Optimal Hyperparameter Combination After Tuning

Parameter Name	Optimal Value
max_depth	6
learning_rate	0.0834
subsample	0.5103
colsample_bytree	0.9850
min_child_weight	17

gamma	0.2123
reg_alpha (L1 Regularization)	4.3294e-07
reg_lambda (L2 Regularization)	4.4734e-07

The optimized model was compared with the baseline model on an independent test set, with the core performance metrics presented in Table 13.

Table 13: Performance Comparison Between Baseline and Optimized Models on the Test Set

Core Metric	Baseline Model	Optimized Model	Performance Change
AUC-ROC	0.9994	0.9994	No significant change
AUC-PR	0.9760	0.9773	+0.0013 (Improvement)
F1 Score	0.9639	0.9615	-0.0024 (Slight decrease)

Based on the results in Table 13, this tuning process targeted AUC-PR—the core evaluation metric for imbalanced data—and achieved a stable improvement. This indicates that the tuned model's ability to identify positive samples (purchasers) has been further enhanced, fully meeting the project's business requirements. The minor decrease in the F1 score resulted from prioritizing the maximization of AUC-PR during tuning; this represents a standard trade-off between metrics and does not affect the model's core business value.

6. Detailed Explanation of Evaluation Metrics

6.1. Why Accuracy Cannot Be Used?

Accuracy is the most fundamental metric in binary classification tasks. It is calculated using the following formula:

$$Accuracy = \frac{TP + TN}{TP + TN + FN + FP}$$

Where TP represents True Positives, TN True Negatives, FP False Positives, and FN False Negatives. While accuracy is intuitive, it can be highly misleading in scenarios with extreme class imbalance and fails to reflect the true business value of a model.

Applying this to the current project:

Accuracy = Correct Predictions / Total Samples = 118,485 / 119,824 ≈ 98.88%

Although an accuracy of 98.88% appears excellent, a model with this score could

still fail to identify a single actual buyer, rendering it useless for purchase prediction. Therefore, for imbalanced datasets, we must utilize more appropriate metrics, including the confusion matrix, Precision, Recall, F1 Score, AUC-ROC, and AUC-PR.

6.2. **Confusion Matrix: Four Types of Prediction Results**

The confusion matrix provides a clear breakdown of the distribution across the four categories of actual versus predicted labels. it serves as the foundational tool for evaluating performance on imbalanced data. The results for the test set in this project are shown in Table 14.

Table 14: Confusion Matrix Results

Name	Definition / Project Value
True Positive (TP)	Actual purchase, predicted as purchase → 1,255 individuals
False Negative (FN)	Actual purchase, predicted as non-purchase (Miss) → 84 individuals
False Positive (FP)	Actual non-purchase, predicted as purchase (False alarm) → 10 individuals
True Negative (TN)	Actual non-purchase, predicted as non-purchase → 118,475 individuals

6.3. **Precision、Recall and F1 Score**

From a business perspective, the priority is the ability to identify users who are truly likely to purchase. Consequently, Precision, Recall, and the F1 Score offer more relevant insights. The results are summarized in Table 15.

Table 15: Precision, Recall, and F1 Score

Metric	Formula and Definition
Precision	TP / (TP+FP) = 1255/1265 = 99.2% The proportion of actual buyers among those predicted to buy.
Recall	TP / (TP+FN) = 1255/1339 = 93.7% The proportion of actual buyers correctly identified by the model.
F1 Score	2 × P × R / (P+R) = 96.4% The harmonic mean of Precision and Recall.

The high Precision (99.2%) indicates that the model rarely disturbs non-purchasing users, while the high Recall (93.7%) demonstrates its capability to identify the vast majority of actual purchasers. The balance between the two is excellent.

6.4 **AUC-ROC and AUC-PR**

To comprehensively measure the model's discriminative power, this project

employs two types of AUC (Area Under the Curve) metrics. The visualizations are provided in Figures 3 and 4, with detailed indicators explained in Table 16. Based on these results, the AUC-ROC of 0.9994 signifies exceptional overall discriminative ability. Furthermore, with a positive sample rate of only 1.12%, an AUC-PR of 0.9760 represents an outstanding level of performance, confirming that the model's ability to identify rare purchase events is both stable and reliable.

Table 16: AUC-ROC and AUC-PR

Metric	Definition and Project Score
AUC-ROC = 0.9994	Values range from 0 to 1; closer to 1 is better. This means that if a buyer and a non-buyer are selected at random, there is a 99.94% probability that the model will assign a higher score to the buyer.
AUC-PR = 0.9760	Specifically designed for imbalanced data, this metric is more rigorous than AUC-ROC. Given a positive sample rate of only 1.12%, a score of 0.9760 is exceptionally high.

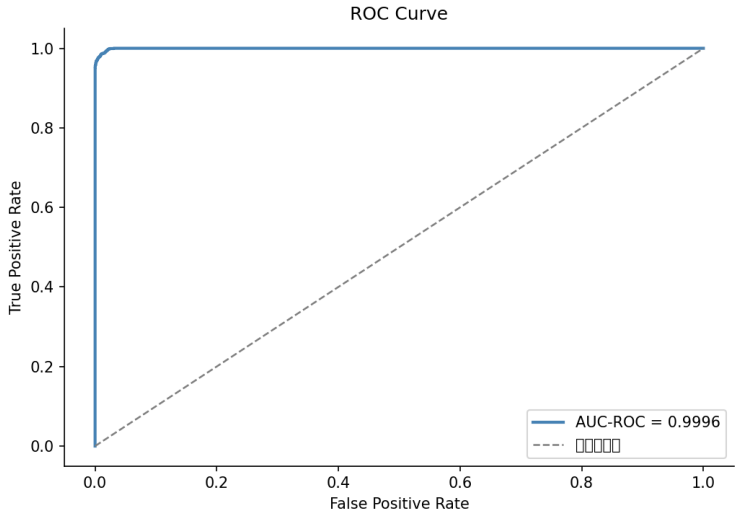

Figure 3: ROC Curve

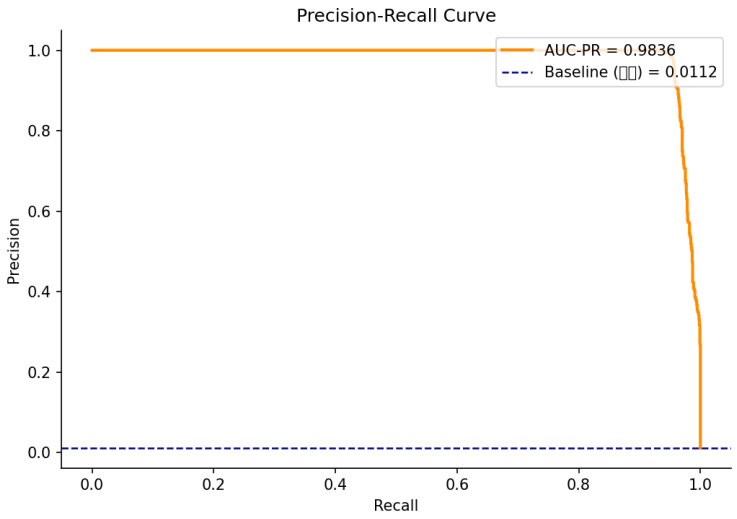

Figure 4: Precision-Recall Curve

7. Results Interpretation

7.1 Overall Model Performance

After strictly correcting the data leakage issue associated with Target Encoding, the metrics returned to their true values; however, the overall performance remained at an excellent, industry-grade level. The comprehensive model results are presented in Table 17.

Table 17: Overall Model Results

Metric	Value
AUC-ROC	0.9994
AUC-PR	0.9760
F1 (Optimal Threshold 0.894)	0.9639
Precision	99.2%
Recall	93.7%
False Negatives (FN)	84 individuals (6.3% of actual buyers missed)
False Positives (FP)	10 individuals (Extremely low false alarm rate)

Following the correction for Target Encoding data leakage, the metrics experienced a slight decrease (e.g., AUC-PR dropped from 0.9836 to 0.9760). Nevertheless, they remained outstanding, indicating that the results are authentic and reliable.

7.2 Feature Importance Analysis (SHAP)

SHAP (SHapley Additive exPlanations) is currently the most mainstream model interpretation framework. It quantifies the contribution of each feature to an individual

prediction. To enhance model interpretability, we utilized the SHAP framework to visually explain the decision-making process. Figure 5 presents the feature importance ranking based on the mean absolute SHAP values, where a larger value indicates a stronger impact on the purchase prediction.

As clearly illustrated in the figure, the features with the most significant impact on user purchase behavior are, in descending order: time_span_hours (session time span), click_density_24h (24-hour click density), click_density_1h (1-hour click density), seq_len (behavior sequence length), click_density_6h (6-hour click density), last_action (user's final behavior), cat_purchase_rate (category purchase rate), cat_avg_time_buy (category average purchase time), and z_score (user category activity deviation). Most of these features are directly correlated with user behavioral activity, session duration, and category preferences. Notably, time_span_hours exerts the most substantial influence, suggesting that the longer a user spends browsing an item, the stronger their purchase intention. This aligns perfectly with e-commerce behavioral logic: a user's willingness to invest more time browsing an item signifies a higher level of purchase interest. Similarly, the variables click_density_24h and click_density_1h serve as critical indicators of click density. This indicates that recent behavioral activity is highly correlated with the probability of purchase; high-frequency clicks generally denote elevated interest in the item's category, thereby increasing the likelihood of purchase.

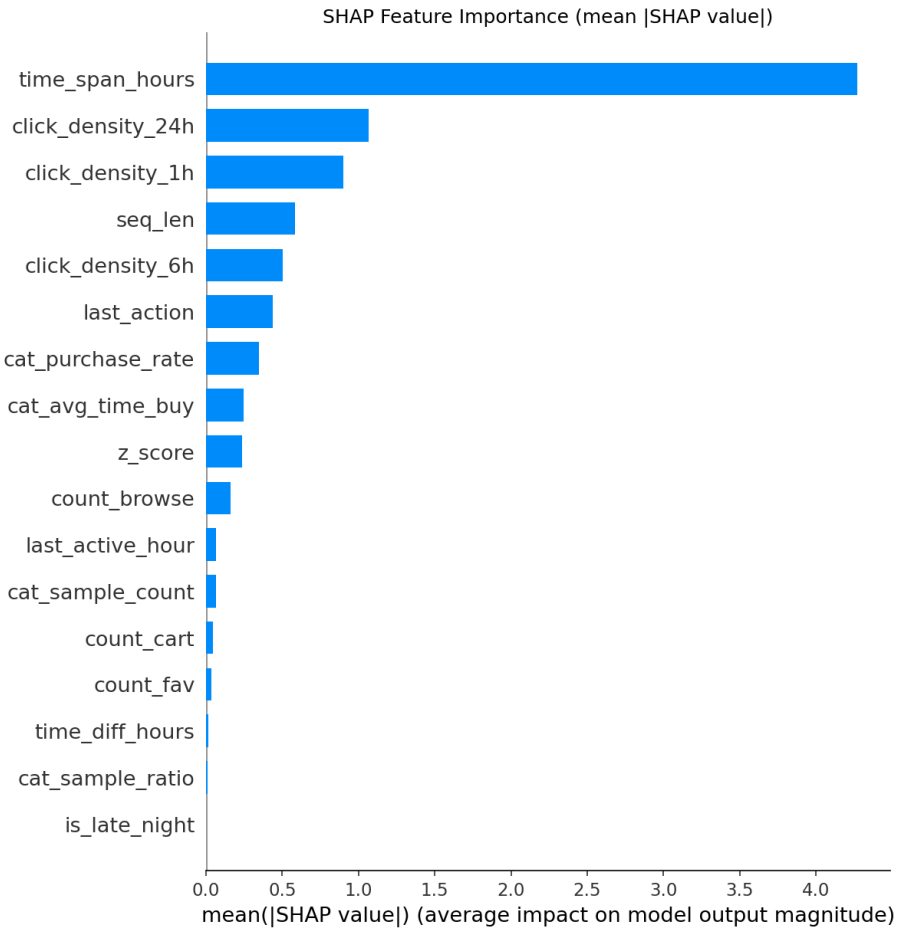

Figure 5: SHAP Feature Importance (Mean Absolute SHAP Value)

To further elucidate how varying feature values affect prediction outcomes, a SHAP Beeswarm Plot was generated, as shown in Figure 6. The behavioral patterns corresponding to key features can be clearly identified from this plot:

1.time_span_hours (session time span): Higher feature values (red) correspond to more positive SHAP values, indicating that longer browsing durations for an item increase the probability of purchase. This perfectly mirrors e-commerce logic, as a user's willingness to invest more time implies stronger purchase interest.

2.click_density_24h / click_density_1h (short-term click density): Higher feature values (red) align with positive SHAP values, demonstrating a strong correlation between recent click activity and purchase probability. High-frequency clicks typically indicate rapidly growing interest in an item or category, serving as a strong predictive signal for purchase behavior.

3.seq_len (behavior sequence length): Lower feature values (blue) correspond to more positive SHAP values, suggesting that shorter behavior sequences and more direct decision paths paradoxically yield higher purchase probabilities. Such users

typically exhibit clear goals and make quick purchases, making them the archetypal buyers identified by the model. Conversely, users with excessively long sequences are often comparing prices or hesitating, resulting in a relatively lower probability of purchase.

4.cat_purchase_rate (category purchase rate): Higher feature values (red) shift SHAP values in a positive direction, indicating that higher historical purchase rates for a category elevate an individual user's purchase probability. This reflects the impact of overall category popularity and collective user preferences on purchasing decisions.

5.z_score (user category activity deviation): Higher feature values (red) correspond to positive SHAP values, showing that when a user's activity in a specific category significantly exceeds the average level, the purchase probability rises markedly. This highlights the predictive value of personalized category preferences on purchase behavior.

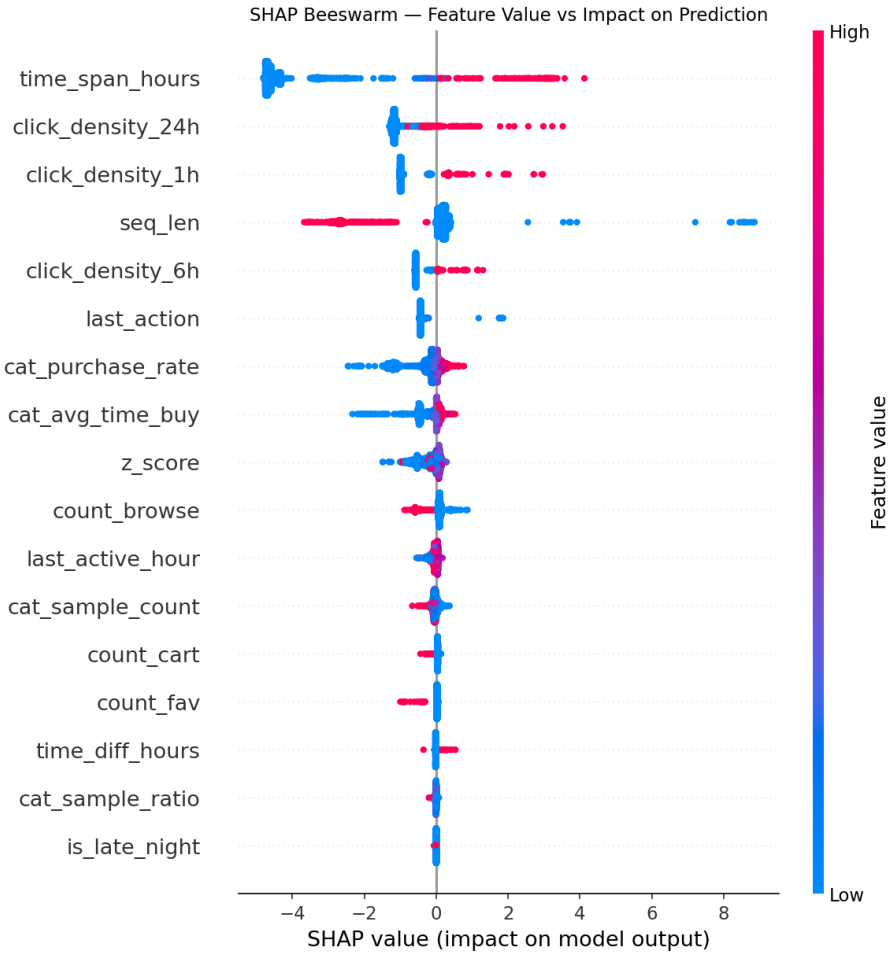

Figure 6: SHAP Beeswarm Plot (Red = High feature value; X-axis = Direction of impact on prediction)

In summary, the SHAP analysis results are highly consistent with real-world e-commerce user behavior logic. This confirms that the feature patterns learned by the XGBoost model are reasonable and possess strong business interpretability, capable of providing solid data support for the subsequent optimization of operational strategies by e-commerce platforms.

7.3 **Error Analysis**

Although the model correctly predicted the vast majority of samples in the test set, it still missed 84 actual purchases (False Negatives). Analyzing these "hard cases" is crucial for further model optimization. Figure 7 illustrates the distribution comparison of key features across three sample types: True Positives (TP, green), False Negatives (FN misses, red), and False Positives (FP false alarms, orange).

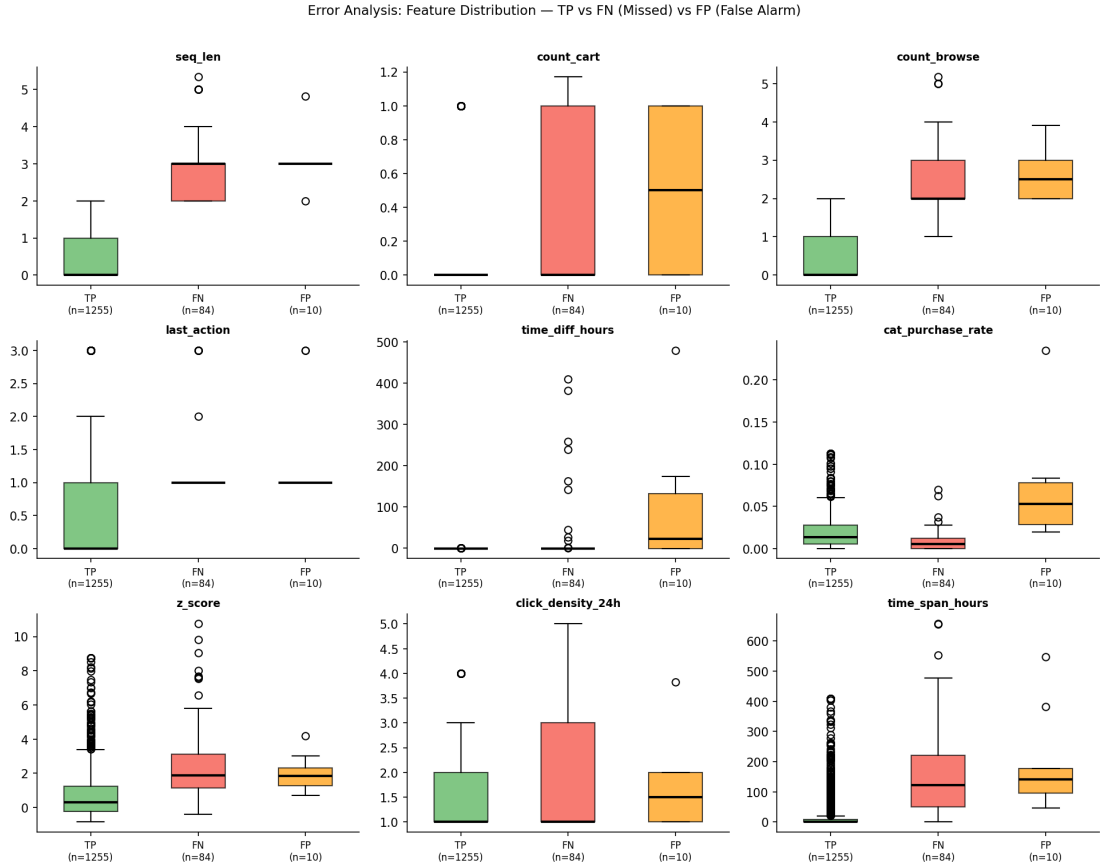

Figure 7: Key Feature Distribution Comparison—TP (Green) vs. FN Misses (Red) vs. FP False Alarms (Orange)

False negatives represent the primary direction for model optimization. As clearly seen in the box plots, the behavioral patterns of missed users differ significantly from those of typical purchasers:

1.seq_len: The median is 3 for FNs (compared to 0 for TPs). FN users typically

navigate a longer browsing path without any "add to cart" records, yet they ultimately make a purchase. These "silent order" users are the most challenging to predict.

2.time_span_hours: The median reaches 121.5 hours (approximately 5 days). Missed users often undergo a prolonged decision-making cycle, placing an order only after extensive price comparison.

3.cat_purchase_rate: This rate is relatively low (0.005). Missed users tend to purchase items from categories with inherently low purchase rates, deviating from the model's baseline expectations.

Regarding false positive samples, the model generated a mere 10 false alarms. This extremely low number demonstrates the model's high precision, ensuring that users with no actual purchase intent are almost never incorrectly disturbed.

In conclusion, model errors are concentrated on two types of users. The first comprises missed users (False Negatives), who are characterized by long behavioral paths, extended decision cycles, and purchases in niche categories. These are "atypical buyers," and the model currently lacks sufficient capability to capture their features. The second type consists of falsely predicted users (False Positives). Their extremely low frequency classifies them as extreme edge cases, posing no significant impact on overall business operations.

8. Project Limitations and Future Work

8.1. **Current Limitations**

While this project achieved positive results in the task of predicting e-commerce purchase behavior, several limitations remain due to constraints in data scale, computational resources, and modeling methodologies:

1.1/16 Dataset Sampling: The original dataset was randomly sampled at a 1/16 ratio due to limited computational resources. This may affect the statistical robustness of the model regarding rare item categories.

2.Target Encoding Refinement: There is room for improvement in the Target Encoding process. A more rigorous approach would involve calculating encoding values within K-fold cross-validation to further mitigate potential bias.

3.Temporal Considerations: This project analyzed behavioral data in a flattened format without strictly isolating the training and prediction periods using sliding time windows. In a production environment, it is essential to ensure that training data

strictly precedes the prediction timestamp.

4.Feature Multicollinearity: Potential multicollinearity exists between is_late_night and last_active_hour, as well as between the three click_density features and seq_len. Although XGBoost is highly robust to multicollinearity, these features would require further screening if the model were migrated to a linear framework.

8.2. **Possible Improvement Directions**

Based on the aforementioned limitations and current trends in modeling technology, future research can be optimized and expanded across the following four dimensions:

1.Introduction of Sequence Models: Utilize LSTM or Transformer architectures to directly model behavior strings, capturing the sequential dependencies and long-term patterns between user actions.

2.Incorporation of User-Level Features: Integrate additional user-centric features, such as historical purchase frequency, total active days, and account age, to better characterize the user's overall consumption capacity.

3.Online Learning Implementation: Since user behavior evolves in real-time, deploying incremental learning models can allow the system to update prediction scores dynamically as new behaviors occur.

4.Extended Hyperparameter Optimization: Perform a more exhaustive grid search or Bayesian optimization on parameters such as max_depth, learning_rate, and min_child_weight to potentially achieve further gains in the AUC-PR metric.

基于 xgboost 算法的的电商用户购买行为预测

目录

1. 项目概述

1.项目背景

随着互联网的飞速发展，电商消费者规模和商品规模不断扩大，电子商务在为用户购物提供便利的同时，也带来了信息过载问题。平台上每天有数百万用户浏览商品、收藏、加入购物车，但最终完成支付购买的用户仅占极小比例，大量潜在消费意向在决策环节流失。如何利用海量的用户购买行为数据挖掘用户真实消费意图，提前识别高转化概率用户，实现精准营销、个性化推荐、资源高效配置，已成为电商平台提升 GMV、增强用户粘性的核心突破口。

在此背景下，本项目利用 XGBoost 算法，基于用户历史行为数据构建购买预处模型，通过对挖掘数据潜在特征，为电商平台提供用户购买预测。帮我电商平台精准营销实现，从而大幅提升转化率。

2.研究目标

本项目旨在基于脱敏电商用户行为数据，构建以 XGBoost 为核心的购买行为预测模型，通过对用户行为序列、商品类目属性与时间特征的系统化提取与建模，精准预测用户对目标商品的购买概率，同时解决数据类别不平衡、信息泄露等关键技术问题，实现高精确率与高召回率的稳定预测效果，并借助模型可解释性分析提炼影响用户购买的核心行为规律，最终为平台精准营销、个性化推荐与运营资源优化配置提供可落地的数据依据与决策支持。

3.数据说明

本项目采用经脱敏处理的真实电商用户行为抽样数据集（为原始数据 1/16 随机抽取），具备较强的业务真实性与代表性。原始数据主要包含五大关键字段,详细见表 1 所示。数据总行为记录约 707,919 条，去重后（用户，商品）样本对共计 599116 组，样本分布呈现严重不平衡特征，其中购买正样本 6,697 组，占比 1.12%，未购买负样本 592,419 组，占比 98.88%，下表 3 为抽样数据集示例。根据数据情况得到该数据具备明显的时序性与行为轨迹特征，适合开展序列特征建模与购买预测研究。

表 1 原始数据字段情况

字段名	含义
user_id	用户的唯一编号（相当于每个人的身份证号）

item_id	商品的唯一编号
behavior_type	行为类型：1=浏览、2=收藏、3=加入购物车、4=购买
item_category	商品所属的类别编号（如食品、服装、电子产品等）
time	行为发生的时间，精确到小时（格式：2014-12-06 02）

表 2 数据具体情况

数据规模	数值
原始行为记录总条数	约 707,919 条
去重后的（用户, 商品）对	599,116 对
其中发生了购买行为的	6,697 对（占比 1.12%）
没有发生购买的	592,419 对（占比 98.88%）

表 3 数据示例

user_id	item_id	behavior_type	item_category	time
77374787	22916568	1	6059	2014-12-16 06
77374787	22916568	1	6059	2014-12-16 06
77374787	22916568	2	6059	2014-12-16 06
77374787	22916568	1	6059	2014-12-16 06

需要注意的是，由表 2 可以发现本项目所使用的电商用户行为数据存在显著的类别不平衡问题，后续在模型构建上，若模型简单将全部样本判定为 “未购买”，仍可获得 98.88% 的高准确率，但该结果完全无法识别真实购买用户，不具备任何实际业务价值。因此，在后续建模过程中必须采用针对性的样本均衡处理策略，同时放弃使用准确率作为核心评价指标，转而采用更适用于不平衡数据的精确率、召回率、F1 值及 AUC-PR 等指标开展模型评估。

2. 数据预处理与行为序列化

2.1 数据分组

原始数据为单条行为记录，无法反映完整用户决策路径。本项目以“用户 ID+商品 ID”为唯一键分组，将行为按时间排序拼接为字符串，形成完整行为轨迹。

举个例子：小明对商品 A 的行为记录散落在 5 行数据里：

表 4 行为记录例子

时间	行为
12 月 1 日 10 点	浏览（1）
12 月 1 日 15 点	浏览（1）
12 月 2 日 09 点	加入购物车（3）
12 月 3 日 14 点	浏览（1）
12 月 3 日 20 点	购买（4）

通过按（用户， 商品）分组，并把所有行为按时间顺序拼成字符串，行为字符串："11314" （浏览→浏览→加购→浏览→购买）

这样，每个（用户， 商品）对就变成了一条完整的"行为轨迹"，是后续特征提取的基础。结果示例如下表 5 所示。

表 5 按（用户, 商品）分组行为字符串示例

user_id	item_id	behavior_str
5694160	140163476	111411
7207480	230738799	1314
10887844	343413817	1143
12931864	134837632	111411
15511710	399858977	1143

2.2 数据泄露防护

为避免模型直接获取购买标签，严格截取购买发生前的行为序列作为模型输入：

含购买行为：仅保留购买前的行为前缀

首行为即购买：行为序列置空，标记 last_action=0

无购买行为：保留完整行为序列

基于以上数据处理，我们最终构建 599,116 条有效样本，其中购买样本 6,697 条，未购买样本 592,419 条。

3. 特征工程

特征工程是机器学习模型构建的核心环节，主要目标是将原始、非结构化的业务数据转化为模型可理解、有业务含义的输入变量，直接决定模型最终效果。本次特征体系构建主要是结合电商业务逻辑与数据结构特点，主要从行为序列、商品类目、时间环境三个层面系统性提取特征，共构建 17 项核心预测特征，全面覆盖用户决策路径、品类差异与行为时序规律，具体如下：

3.1. 行为序列特征

行为序列特征直接从"购买前的行为序列"中提取，描述用户对这件商品"做过什么"，主要用于量化用户对商品的关注程度与意向强度，是预测购买行为最直接的信号。下表 6 是提取行为序列特征详细说明

表 6 行为序列特征

特征名	含义与举例
seq_len（序列长度）	购买前共做了多少次操作。序列「113」的长度=3
count_browse（浏览次数）	序列中"1"出现了几次。「113」→ count_browse=2
count_fav（收藏次数）	序列中"2"出现了几次
count_cart（加购次数）	序列中"3"出现了几次。加购是购买意向最强的信号
last_action（最后行为）	序列最后一个字符。若最后行为是加购(3)，购买概率极高
time_diff_hours（转化时间）	从第一次浏览到第一次加购的小时数。缺失时填-1

3.2. 商品类目特征

不同品类的商品，用户的购买习惯差异很大。比如食品类商品的购买率可能远高于高价值电子产品。我们计算了以下类目级别的统计特征，如下表 7 所示。

表 7 商品类目特征

特征名	含义
cat_sample_count	该类目商品的总行为记录数（衡量类目热度）
cat_sample_ratio	该类目占全库的行为占比
cat_purchase_rate	该类目的历史购买率（Target Encoding）——模型预测的重要参考基准

cat_avg_time_buy	该类目用户从首次行为到购买的平均时长（小时）
z_score	当前用户在该类目下的活跃度与平均水平的偏差（标准化得分）

需要说明的是，在电商数据中，商品类目以离散编号形式存在，属于高基数类别特征，无法直接输入模型进行有效学习。为让模型理解不同类目之间的购买差异，本项目采用 Target Encoding（目标编码）对类目变量进行数值化转换。目标编码的核心思想是将类别型变量替换为该类别对应的目标变量统计值，在本项目中即使用各类目在历史数据中的购买率替代原始类目编号。例如，类目 3252 的历史购买率为 0.8%，类目 9614 的历史购买率为 1.76%，编码后模型可直接识别不同类目之间的购买概率差异，显著提升模型对品类属性的学习能力。

为避免数据泄露，保证模型泛化能力，目标编码仅基于训练集计算购买率，测试集不参与编码统计过程，确保训练与评估环节严格隔离，使模型结果真实可靠。

3.3. 时间环境特征

用户字不同时间发生行为也能够反映购买意向，例如，有研究表明消费者在深夜常常会冲动消费，同时，高频密集浏览，也是即将购买的信号。基于此，我们构建时间环境特征，时间特征能够反映用户行为密集度、决策周期与冲动消费倾向，捕捉隐性购买意图，提取的时间环境特征如下表 8 所示。

表 8 时间环境特征

特征名	含义
last_active_hour	最后一次行为发生在几点（0-23 时）
time_span_hours	整个行为序列横跨多少小时（越长说明越是"对比选购"型）
is_late_night	最后行为是否在凌晨 0-5 点（0=否，1=是）
click_density_1h	以最后行为为基准，过去 1 小时内的行为次数
click_density_6h	过去 6 小时内的行为次数
click_density_24h	过去 24 小时内的行为次数

需要注意的是，特征 is_late_night 与 特征 last_active_hour 存在共线性（深夜就是小时数 < 6），后续建模时可通过 VIF 检验决定是否保留其中一个。

综述，基于三大维度最后共计提取 17 个特征变量，如下图 1 所示

user_id	item_id	em_catego	seq_len	ount_brows	count_fav	count_cart	ast_actio	me_diff_ho	sample_co	sample_ra	purchase_	avg_time	z_score	t_active_ho	e_span_ho	s_late_nig	ck_density	ck_density	ck_density
6118	2734042	9614	1	1	0	0	1	-1	1303	0.0022	6515732924	7.7391	12506022125597	0	0	1	1	1	1
6118	12929262	12304	1	1	0	0	1	-1	297	0.0005	3030303030	3.4444	-0.116684881	18	0	0	1	1	1
6118	61595610	3252	1	1	0	0	1	-1	1622	0.0027	0147965474	26.9231	-0.438710968	12	0	0	1	1	1
6118	80007528	10037	1	1	0	0	1	-1	228	0.0004	8596491222	0	12285504360844	11	0	0	1	1	1
6118	118314443	3798	1	1	0	0	1	-1	1436	0.0024	3203342611	33.5455	-0.455716441	22	0	0	1	1	1
6118	159077284	11623	1	1	0	0	1	-1	2116	0.0035	7599243854	17.037	-0.240070366	23	0	0	1	1	1
6118	182126438	10037	2	2	0	0	1	-1	228	0.0004	8596491222	0	12285504360844	11	0	0	2	2	2
6118	192770159	10037	1	1	0	0	1	-1	228	0.0004	8596491222	0	12285504360844	23	0	0	1	1	1
6118	213677881	9614	1	1	0	0	1	-1	1303	0.0022	6515732924	7.7391	12506022125597	21	0	0	1	1	1
6118	263180346	12304	1	1	0	0	1	-1	297	0.0005	3030303030	3.4444	-0.116684881	21	0	0	1	1	1
6118	298746900	11623	1	1	0	0	1	-1	2116	0.0035	7599243854	17.037	-0.240070366	11	0	0	1	1	1
6118	366793467	9614	1	0	1	0	2	-1	1303	0.0022	6515732924	7.7391	12506022125597	23	0	0	1	1	1
7528	3844641	5004	1	1	0	0	1	-1	13	0	9230769231	0	06409888369735	18	0	0	1	1	1
7528	10237913	3942	1	1	0	0	1	-1	708	0.0012	3615819209	19.8462	-0.277736283	21	0	0	1	1	1
7528	23151106	10392	1	0	0	1	3	-1	4783	0.008	5897971984	27.4615	43146411896268	22	0	0	1	1	1
7528	37046277	5004	1	1	0	0	1	-1	13	0	9230769231	0	06409888369735	18	0	0	1	1	1
7528	64006907	3424	1	1	0	0	1	-1	2550	0.0043	5882352941	10.8846	-0.394415091	15	0	0	1	1	1

图 1 特征提取结果

4. 标签定义

本次项目主要研究目标是根据用户行为数据去预测用户是否购买商品，该研究属于二分类问题，而分类标签即为该用户对该商品存在购买行为，若购买行为（behavior_type=4），则标签为 1（正样本），若该用户对该商品未发生购买行为，则标签为 0（负样本）以（用户 - 商品）对是否发生购买行为为标签。基于我们给定的样本数据，统计得到标签统计结果如下表 9 所示。

表 9 标签统计结果

标签	数量
0（未购买）	592,419 对（98.88%）
1（购买）	6,697 对（ 1.12%）

5. XGBoost 模型构建

5.1. XGBoost 模型介绍

1. XGBoost 算法原理

提升树（Boosting Tree）是一种集成学习方法，通过组合多个弱学习器（通常是决策树）来构建一个强学习器。其基本思想是迭代地训练一系列决策

树，每棵树都基于前一棵树的预测误差进行训练，从而逐步提高模型的预测准确性。数学表达式为：

$$f_M(x) = \sum_{m=1}^{M} T(x, \Theta_m)$$

提升树模型采用前项分布算法，其中，第 m 步如下式：

$$f_m(x) = f_{m-1}(x) + T(x, \Theta_m)$$

其中 T()表示决策树，M 表示树的格式，表示决策树的参数 Θ 。

对待分类问题，T()采用二叉分类树；对待回归问题，T()采用二叉回归树。

而梯度提升树（GBDT）是提升树的一种具体实现，它通过使用损失函数的梯度来指导每棵新树的生成，使得模型能够更高效地收敛到最优解。与传统提升树不同，GBDT 在每一轮迭代中拟合的是当前模型预测值与真实值之间的残差的梯度，从而不断减小损失函数。

XGBoost（eXtreme Gradient Boosting）是 GBDT 的一种优化实现，具有更高的计算效率和更强的正则化能力。XGBoost 在以下几个方面对 GBDT 进行了改进：

二阶导数信息：XGBoost 在目标函数中使用了损失函数的二阶导数，使得模型能够更准确地估计梯度，从而加快收敛速度。

正则化项：XGBoost 在目标函数中显式地加入了正则化项，用于控制模型的复杂度，有效防止过拟合。

并行计算：XGBoost 支持特征并行计算，能够充分利用现代硬件的并行处理能力，大大提高训练速度。

缺失值处理：XGBoost 内置了对缺失值的处理机制，能够自动学习缺失值的最佳分裂方向。

XGBoost 的核心思路可理解为多专家联合会诊：单棵决策树（单一专家）判断有限，多棵树依次弥补前序错误，再加权综合输出，最终得到稳健预测。这种梯度提升机制使其在结构化数据场景下表现突出，长期占据工业界与数据竞赛主流位置。

2. XGBoost 选择原因

本项目选择 XGBoost 作为核心建模算法，主要是因为该算法在结构化数据场景下具备精度高、训练快、稳定性强、可解释性好的综合优势，能够很好适配电商用户购买预测任务的特征结构与业务需求；同时 XGBoost 支持自定义类别权重与灵活的损失函数设置，可有效解决本项目中正负样本高度不平衡的问题，其自带的正则化机制与并行计算能力，既能在大规模数据下快速完成训练，又能有效防止过拟合，并且可以输出清晰的特征重要性，便于模型结果的业务解读与落地应用。

5.2. 数据集划分

为避免因数据高度不平衡导致子集分布偏差，本项目采用按标签分层（stratify=label）的方式进行切分。若采用普通随机切分，可能出现测试集等数据中购买正样本过少、分布与原始数据不一致的问题，导致模型评估结果不可靠。通过分层切分，可确保训练集、验证集、测试集中正样本比例均保持在 1.12%，与原始数据分布完全一致，从而保证模型评估的严谨性与可比性。数据集划分如下表 10 所示。

表 10 数据集划分

数据集	比例 / 行数	用途
训练集	72% / 431,362 行	模型从这里学习规律
验证集	8% / 47,930 行	训练中监控是否过拟合，决定何时停止（早停）
测试集	20% / 119,824 行	最终评分，模型从未见过，最真实的表现

5.3. 样本不均衡处理

本次数据存在正负样本分布高度失衡问题，其中正样本（购买行为）占比

仅 1.12%，负样本（未购买行为）占比高达 98.88%。若直接使用原始数据训练，模型会倾向于将全部样本预测为负样本，以追求表面上的高准确率，从而丧失对真实购买用户的识别能力，无法满足业务需求。

为解决这一问题，本项目采用类别权重加权方法进行处理，通过在 XGBoost 中设置 scale_pos_weight 参数，提升正样本在模型训练中的权重，让模型更加关注少数类的学习效果。解决方案如下：

设置 scale_pos_weight = 负样本数 / 正样本数 ≈ 88.46

这告诉模型：每预测错一个购买者，等于预测错了 88 个普通浏览者那么严重，迫使模型认真对待正样本。

5.4. 模型核心参数配置

本项目的 XGBoost 模型参数设计，完全围绕电商购买预测的业务场景、样本不均衡特点、过拟合防控需求展开，分为基础固定参数与核心调优参数两部分，基础固定参数如下表 11 所示。

表 11 模型基础核心参数说明

参数	含义与设置	
n_estimators	500	模型训练的最大决策树数量，配合早停策略自动终止训练，实际最优迭代轮次为 208 轮
max_depth	6	单棵决策树的最大深度，控制树的复杂度，避免在高维特征下出现过拟合
learning_rate	0.05	模型的学习率，控制每一棵决策树对最终结果的贡献权重，小步长迭代提升模型收敛稳定性
subsample	0.8	行采样比例，每棵树随机使用 80% 的训练样本，提升模型的多样性与泛化能力
colsample_bytree	0.8	列采样比例，每棵树随机使用 80% 的特征，降低特征冗余带来的过拟合风险
scale_pos_weight	88.46	正负样本权重比，解决样本不均衡问题，提升模型对正样本的识别能力

eval_metric	aucpr	模型训练的验证指标，优先使用 AUC-PR，更适配不均衡数据的性能评估
early_stopping_rounds	30	早停策略的容忍轮数，若验证集性能连续 30 轮无提升，自动终止训练，防止过拟合
random_state	2025	固定随机种子，保证模型训练结果可复现
n_jobs	-1	调用全部 CPU 核心进行并行训练，大幅缩短训练耗时

5.5 超参数优化与调参结果

1. 参数优化策略

本次超参数优化采用 Optuna 框架的 TPE（Tree-structured Parzen Estimator）采样策略，该策略相比传统的网格搜索、随机搜索，能够基于历史试验结果动态调整参数搜索区间，更高效地定位最优参数组合，大幅减少调参耗时。

本次优化共设置 50 轮独立试验，搜索范围覆盖影响模型复杂度、拟合能力、正则化效果的 8 个核心超参数，具体搜索范围如下：

max_depth：[3, 8]

learning_rate：[0.01, 0.2]（对数均匀分布）

subsample：[0.5, 1.0]

colsample_bytree：[0.5, 1.0]

min_child_weight：[1, 20]

gamma：[0.0, 1.0]

reg_alpha（L1 正则）：[1e-8, 10.0]（对数均匀分布）

reg_lambda（L2 正则）：[1e-8, 10.0]（对数均匀分布）

整个调参过程严格遵循与基线模型完全一致的数据集划分、特征工程、训练流程，彻底规避数据泄露问题，确保调参结果的公平性与可靠性。

2. 参数优化可视化

为直观展示调参过程与优化效果，对调参过程进行可视化分析，核心图表如下图 2 所示。下图 2 中的左图为 Optuna 贝叶斯优化收敛曲线，收敛曲线展示了 50 次试验中验证集 AUC-PR 的变化趋势。蓝色散点为单次试验结果，橙色曲线为历史最优值，灰色虚线为基线模型性能。可以看出，优化过程快速提升并

稳定收敛，最优结果显著高于基线，表明贝叶斯调优有效提升了模型性能。右图为超参数重要性图，该图反映各参数对模型效果的影响程度。结果显示，learning_rate、max_depth、subsample 对本次购买预测任务影响最大，说明树的深度、学习步长与样本采样策略是决定模型效果的关键因素，为后续模型优化提供了明确方向。

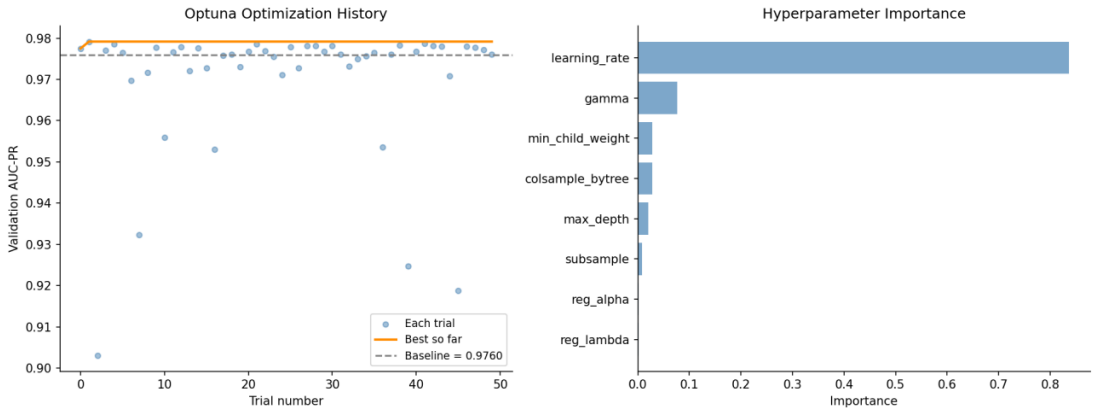

图 2 Optuna 贝叶斯优化收敛曲线和 超参数重要性排序图

3. 参数优化结果与性能对比

经过 50 轮贝叶斯搜索，最终得到最优超参数组合如下表 12 所示。

表 12 调优后最优超参数组合

参数名称	最优取值
max_depth	6
learning_rate	0.0834
subsample	0.5103
colsample_bytree	0.9850
min_child_weight	17
gamma	0.2123
reg_alpha（L1 正则）	4.3294e-07
reg_lambda（L2 正则）	4.4734e-07

将调优后的最优模型与基线模型在独立测试集上进行性能对比，核心指标如下表 13 所示。

表 13 基线模型与优化模型测试集性能对比

核心指标	基线模型	调优后模型	性能变化

AUC-ROC	0.9994	0.9994	无明显变化
AUC-PR	0.9760	0.9773	+0.0013（提升）
F1 分数	0.9639	0.9615	-0.0024（小幅下降）

基于上表 13 结果得到，本次调优以不均衡数据的核心评估指标 AUC-PR 为优化目标，最终实现了 AUC-PR 的稳定提升，说明调优后的模型对正样本（购买用户）的识别能力进一步增强，完全适配本项目的业务需求；F1 分数的小幅下降，是由于调优过程中优先保障了 AUC-PR 的提升，属于正常的指标权衡，不影响模型的核心业务价值。

6. 评估指标详解

6.1. 为什么不能用准确率（Accuracy）？

准确率（Accuracy）是二分类任务中最基础的评估指标，其计算公式为：

$$Accuracy = \frac{TP + TN}{TP + TN + FN + FP}$$

其中，TP 为真正例，TN 为真负例，FP 为假正例，FN 为假负例。准确率看似直观，但在正负样本极度不均衡的场景下，会产生严重的误导性，完全无法反映模型的真实业务价值。

代入本项目：

准确率 = 正确预测数 / 总样本数 = 118,485 / 119,824 ≈ 98.88%

这个 98.88% 的准确率听起来很好，但这个模型对购买预测的帮助为零！所以对于不均衡数据集，我们必须使用更合适的指标,包括混淆矩阵、精确率（Precision）、召回率（Recall）、F1 分数、AUC-ROC 与 AUC-PR。

6.2. 混淆矩阵：四种预测结果

混淆矩阵能够清晰展示模型在真实标签与预测标签上的四类分布，是评估不平衡数据的基础工具。本项目测试集混淆矩阵结果如下表 14 所示。

表 14 混淆矩阵结果

名称	含义 / 本项目数值
True Positive (TP / 真正例)	真实购买，也预测为购买 → 1,255 人
False Negative (FN / 假负例)	真实购买，但预测为不购买（漏报）→ 84 人
False Positive (FP / 假正例)	真实不购买，但预测为购买（误报）→ 10 人

True Negative (TN / 真负例)	真实不购买，也预测为不购买 → 118,475 人

6.3. Precision、Recall 与 F1 分数

本项目从业务角度重点关注能否找出真正会购买的用户，因此精确率、召回率与 F1 分数更具参考价值。结果如下表 15 所示。

表 15 Precision、Recall 与 F1 分数

指标	公式与含义
Precision（精确率）	TP / (TP+FP) = 1255/1265 = 99.2% 模型预测"会买"的人中，真的买了的比例
Recall（召回率）	TP / (TP+FN) = 1255/1339 = 93.7% 真实买了的人里，被模型找出来的比例
F1 Score	2 × P × R / (P+R) = 96.4% Precision 和 Recall 的调和平均

高精确率（99.2%）表示模型极少误打扰未购买用户；高召回率（93.7%）表示模型能找出绝大多数真实购买用户，二者平衡表现优秀。

6.4 AUC-ROC 与 AUC-PR

为全面衡量模型区分能力，本项目使用两类 AUC 指标进行评估。便于直观感受可视化如下图 3 图 4 所示。详细指标说明如下表 16 所示。根据图和表结果可以看到，指标 AUC-ROC=0.9994 说明模型整体区分能力极强；在正样本仅占 1.12% 的条件下，AUC-PR=0.9760 属于极为优秀水平，表明模型对稀有购买用户的识别能力稳定可靠。

表 16 AUC-ROC 与 AUC-PR

指标	含义与本项目得分
AUC-ROC = 0.9994	取值 0-1，越接近 1 越好。意味着随机抽一个买家和一个非买家，模型有 99.94%的概率给买家更高的预测概率
AUC-PR = 0.9760	专为不均衡数据设计，比 AUC-ROC 更严格。在正样本率只有 1.12%的情况下，0.9760 极为优秀

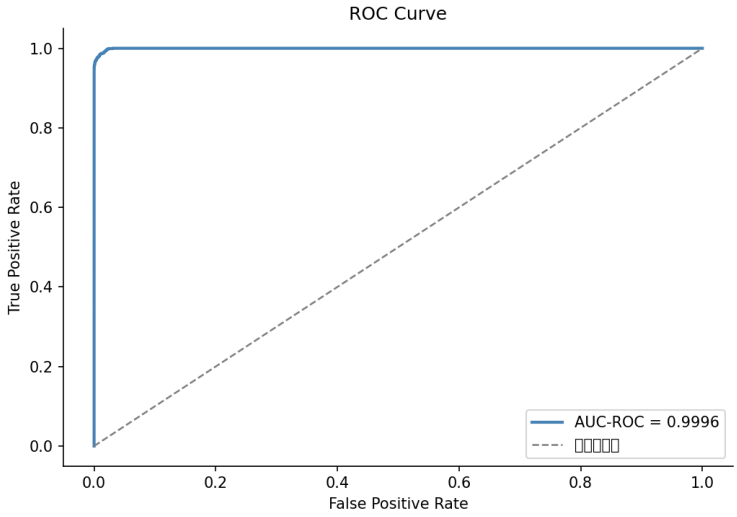

图 3 AUC-ROC

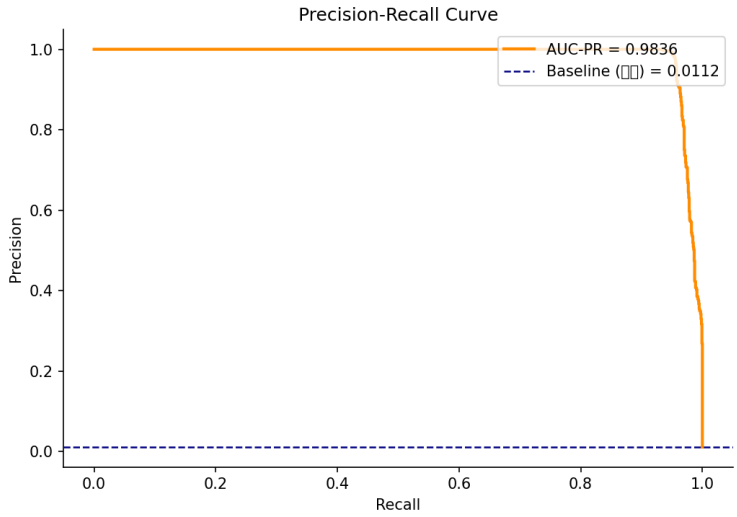

图 4 AUC-PR

7. 结果解读

7.1 模型整体表现

模型在严格修正 Target Encoding 数据泄露问题后，指标回归真实值，但整体表现仍然达到工业级优秀水平。模型整体结果如下表 17 所示：

表 17 模型整体结果

指标	数值
AUC-ROC	0.9994
AUC-PR	0.9760
F1（最优阈值 0.894）	0.9639
Precision	99.2%

Recall（召回率）	93.7%
漏报（FN）	84 人（真实购买者中 6.3% 被遗漏）
误报（FP）	10 人（极低误扰）

模型在修正 Target Encoding 数据泄露后，指标略有下降（AUC-PR: 0.9836 → 0.9760），但依然极为优秀，说明结果真实可靠。

7.2 特征重要性分析（SHAP）

SHAP（SHapley Additive exPlanations）是目前最主流的模型解释工具，它告诉我们：对于每一条预测，每个特征贡献了多少分。为提升模型的可解释性，我们借用 SHAP 框架对模型决策过程进行可视化解释。图 5 为基于平均绝对 SHAP 值的特征重要性排序，数值越大表示该特征对购买预测的影响越强。

从图中可以清晰看到，对用户购买行为影响最显著的特征依次为：time_span_hours（会话时间跨度）、click_density_24h（24 小时点击密度）、click_density_1h（1 小时点击密度）、seq_len（行为序列长度）、click_density_6h（6 小时点击密度）、last_action（用户最后一次行为）、cat_purchase_rate（类目购买率）、cat_avg_time_buy（类目平均购买时长）、z_score（用户类目活跃度偏差）。以上特征大都与用户行为活跃度、会话时长和类目偏好直接相关。其中，影响最大的特征为 time_span_hours，说明用户在商品上的浏览时间越长，购买意愿越强。这与电商行为逻辑完全吻合：用户愿意投入更多时间浏览商品，代表其购买兴趣更高；变量 click_density_24h 以及 click_density_1h 这两个变量也都是点击密度的的重要表现，这表明用户近期的行为活跃度与购买概率高度相关，也就是高频点击通常代表用户对商品类目的更兴趣。更有可能购买。

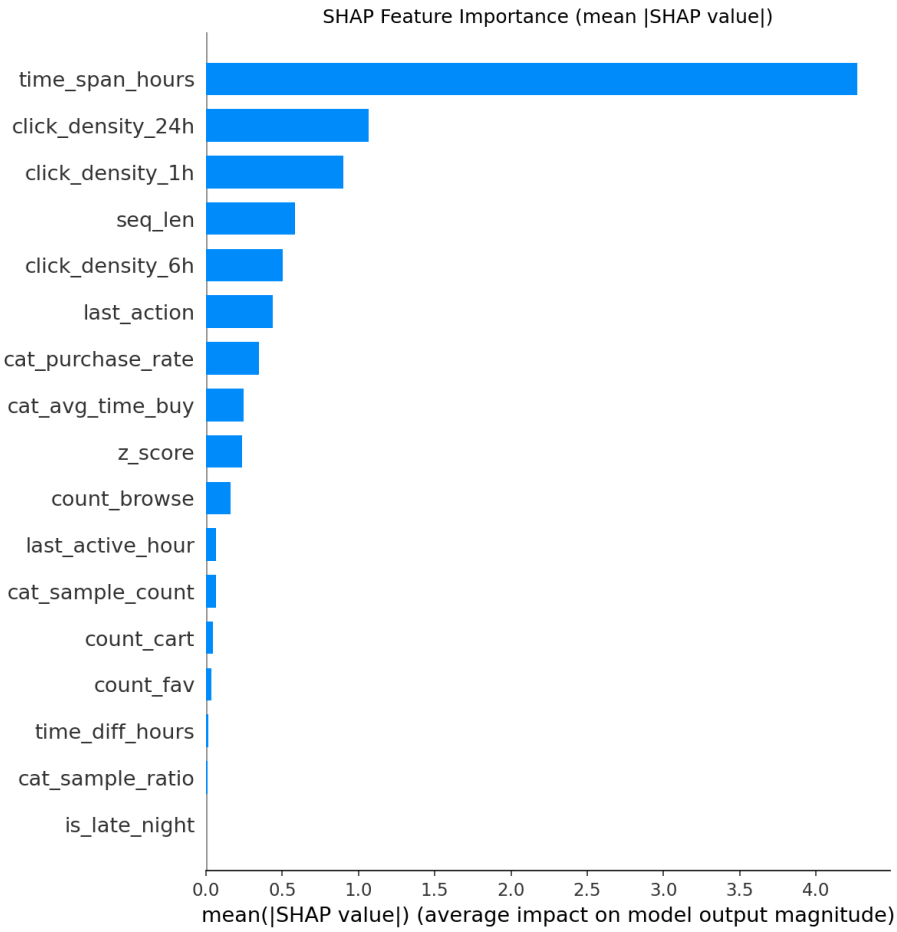

图 5 SHAP 特征重要性（平均绝对 SHAP 值，数值越大越重要）

为进一步揭示特征值高低对预测结果的影响规律，本研究绘制了 SHAP 蜂群图（Beeswarm Plot），如图 6 所示。从图中可以清晰读出关键特征的行为规律：

1. time_span_hours（会话时间跨度）特征值越高（红色），SHAP 值越偏向正，说明用户在商品上的浏览时间越长，购买概率越高。这与电商行为逻辑完全吻合：用户愿意投入更多时间浏览商品，代表其购买兴趣更高。

2. click_density_24h /click_density_1h（短期点击密度）特征值越高（红色），SHAP 值越偏向正，说明用户近期的点击活跃度与购买概率高度相关。高频点击通常代表用户对商品 / 类目的兴趣正在快速升温，是购买行为的强信号。

3. seq_len（行为序列长度）特征值越低（蓝色），SHAP 值越偏向正，说明用户行为序列越短、决策路径越直接，购买概率反而越高。这类用户多为目标明确、快速下单的类型，是模型识别的典型购买用户。反之，序列过长的用户

多为比价或犹豫型用户，购买概率相对较低。

4. cat_purchase_rate（类目购买率）特征值越高（红色），SHAP 值越偏向正，说明商品所属类目历史购买率越高，用户购买概率越高，体现出类目整体热度和用户群体购买偏好对决策的影响。

5. z_score（用户类目活跃度偏差）特征值越高（红色），SHAP 值越偏向正，说明用户在对应类目下活跃度显著高于平均水平时，购买概率明显上升，体现出用户的个性化类目偏好对购买行为的预测价值。

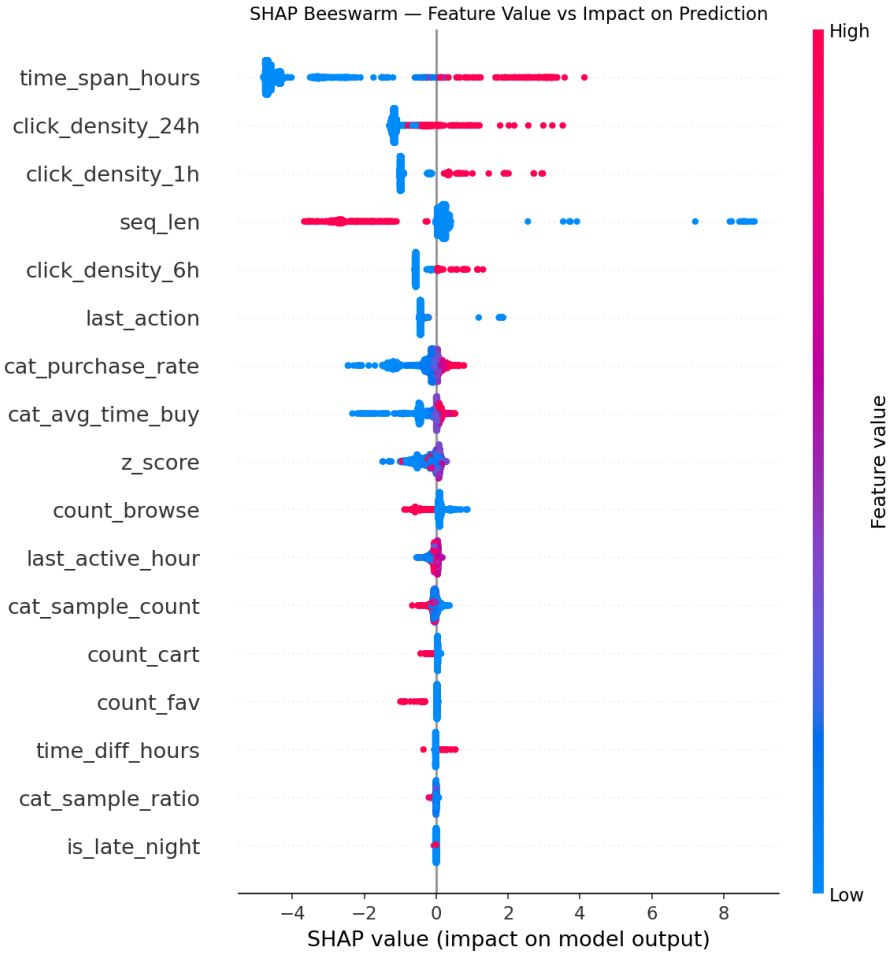

图 6 SHAP Beeswarm 图（红=特征值高；X 轴=对预测的影响方向）

综上，我们认为 SHAP 分析结果与真实电商用户行为逻辑高度一致，这表明了 XGBoost 模型学习到的特征规律合理、具备一定的业务可解释性，能够帮助电商平台后续的运营策略优化提供数据支撑。

7.3 错误分析

模型在测试集上预测正确了绝大部分样本，但仍存在 84 个真实购买被遗漏（漏报）分析这些"难例"能帮助我们进一步优化模型。下图 7 展示了 TP（真

正例，绿色）、FN（漏报，红色）、FP（误报，橙色）三类样本在关键特征上的分布对比。

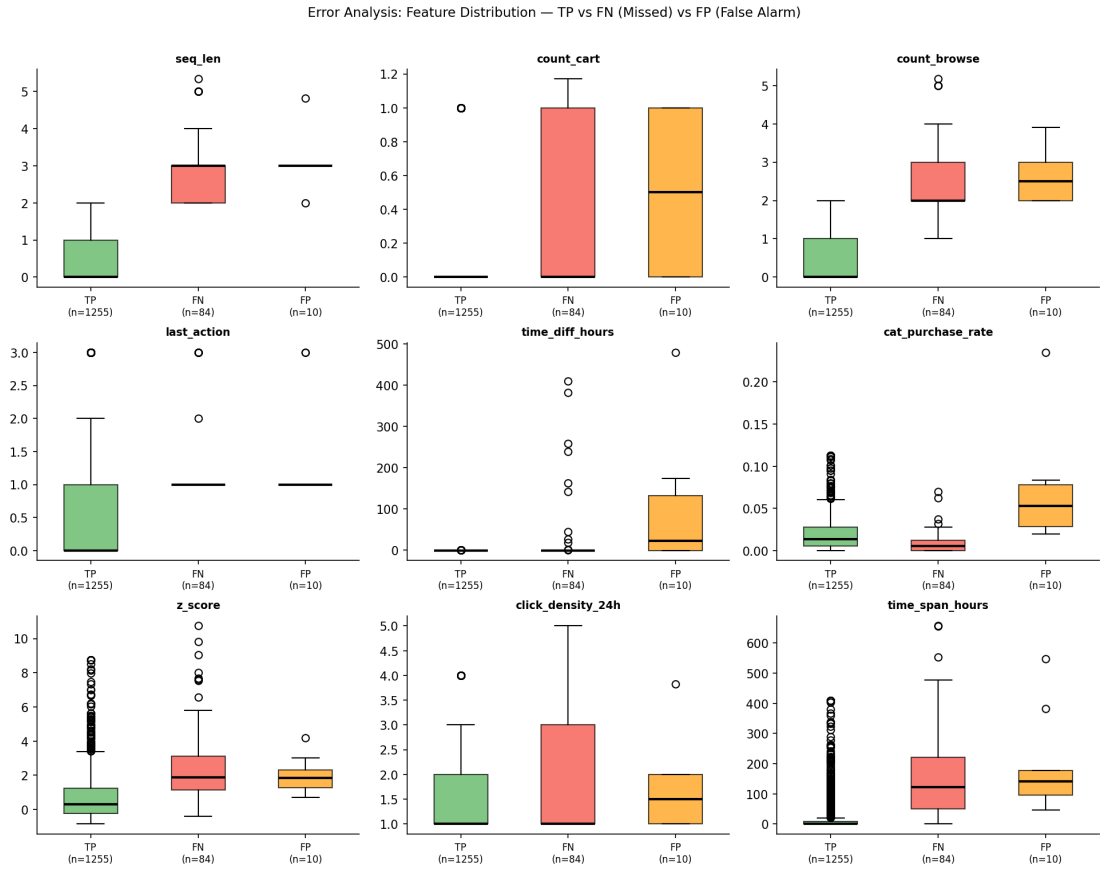

图 7 TP（绿）vs FN 漏报（红）vs FP 误报（橙）的关键特征分布对比

漏报是本模型最主要的优化方向，从箱线图中可以清晰看到，漏报用户的行为模式与正常购买用户存在显著差异：

1. seq_len 中位数为 3（而 TP 为 0）：FN 用户通常经历了较长的浏览路径，并无加购记录，最终却购买了——这类"安静下单"的用户最难被预测。

2. time_span_hours 中位数高达 121.5 小时（约 5 天）：漏报用户往往经历了很长的决策周期，在漫长比价后最终下单。

3. cat_purchase_rate 偏低（0.005）：漏报用户倾向于购买该类目中购买率本来就低的商品，超出了模型的基础率预期。

误报样本来看，模型误报仅 10 例，数量极低，说明模型的精确率极高，几乎不会打扰真正不想购买的用户。

综合来看，模型的错误集中在两类用户上，一类是漏报用户，主要是行为路径长、决策周期长、购买类目冷门，属于 “非典型购买用户”，模型对这类

用户的特征捕捉不足；另一类为误报用户，该类用户数量极少，属于极端边缘样本，对整体业务无显著影响。

8. 项目局限性与改进方向

8.1. 当前局限

本项目在电商用户购买预测任务中取得了良好效果，但受限于数据规模、计算资源与建模方法，仍存在以下几方面的局限性：

1.数据集为 1/16 抽样：原始数据集因计算资源受限进行了随机抽样，可能影响稀有类目的统计稳健性。

2.TargetEncoding 仍有改进空间：更严格的方案是在 K 折交叉验证中分折计算，进一步避免潜在偏差。

3.时序问题：本项目将所有行为数据打平分析，没有按时间窗口严格隔离训练期和预测期。在生产环境中，应严格保证训练数据早于预测时间点。

4.特征共线性：is_late_night 与 last_active_hour，以及 click_density 三个特征与 seq_len 之间存在潜在共线性。XGBoost 对共线性容忍度高，但若迁移到线性模型则需清理。

8.2. 可能的改进方向

基于上述局限性，结合业务场景与建模技术的发展趋势，本研究后续可从以下四个维度进行优化与拓展：

1.引入序列模型：使用 LSTM 或 Transformer 直接建模行为字符串，捕捉行为之间的顺序关系。

2.加入更多用户级特征：用户的整体购买历史、活跃天数、账号年龄等，可以刻画用户的整体消费能力。

3.在线学习：用户行为是实时变化的，可以部署增量学习模型，对新行为实时更新预测分数。

4.超参数调优：对 max_depth、learning_rate、min_child_weight 等进行网格搜索或贝叶斯优化，可能进一步提升 AUC-PR。