Research on E-commerce Order Discount Type Prediction Based on Hybrid Model: Case of JD.com

Jimmy Qiu [a], Annabelle Zhu [b], Durai Sundaramoorthi [c], and Salih Tutun [d]

[a] Master of Science in Quantitative Finance, Olin Business School, Washington University in St. Louis, Clayton, Missouri, 63124

[b] Master of Science in Business Analytics, Olin Business School, Washington University in St. Louis, Clayton, Missouri, 63124

[c] Department of Data Analytics, Olin Business School, Washington University in St. Louis, Clayton, Missouri, 63124

[d] Department of Data Analytics, Olin Business School, Washington University in St. Louis, Clayton, Missouri, 63124

* Corresponding author.

Contact: tianxingqiu@wustl.edu

Abstract

Keywords: E-Commence, Discount Strategy, Resampling, Ensemble Learning, Naïve Bayes Classifier, Hybrid Model.

1. INTRODUCTION

1.1 Overview

1.1.1 Importance of E-commerce to life

With the widespread popularity of the Internet and smart phones, e-commerce has gradually become an indispensable part of modern life. Nowadays, when people have purchase demands, they usually browse e-commerce websites or software through smart devices, find and compare the goods and services they want to buy, make payment through electronic payment platforms, and receive the purchased goods through express logistics. Different from physical stores, e-commerce will not restrict consumers because of the difference in geography and time. E-commerce tends to give consumers more shopping choices and preferential discount policies, so that consumers are more inclined to make purchases on e-commerce platforms. At present, e-commerce has become an irreplaceable part of business in the world. Represented by Amazon, eBay in the United States, JD.com, Taobao, PDD and other large e-commerce platforms in China, e-commerce will bring huge profits to the economy, thus promoting its own development.

JD.com stands out in China's e-commerce industry for its high-quality goods and logistics services. JD.com's products are often divided into JD.com self-operated and third-party merchants, and merchants selected and certified by JD.com 's platform can always win the trust of consumers. JD.com 's huge and efficient logistics warehouse and express delivery network, which took several years to build, provide the fastest e-commerce delivery service in China. JD.com will also hold large-scale promotional activities like 618 Shopping Festival to provide diversified and targeted preferential discount policies for consumers, which can promote commodity sales and consumer consumption on the platform.

1.1.2 How do people use E-commerce

For ordinary people, using mobile phones, computers and other functional devices to browse the e-commerce platform is the most common way. People often search for the items they need to buy on the web pages and software launched by e-commerce platforms, or on the small programs on social media, while e-commerce platforms also recommend items that their customers may be interested in and buy. When choosing goods, people choose them according to their personal preference, budget, time of delivery and other factors. At the same time, some consumers will actively try to combine different goods in one order, so that their own coupons can finally give

them the maximum discount value. Moreover, consumers are likely to buy VIP services on the platform, so that they can get more discounts and better services.

1.1.3 What strategies do E-commerce companies use

An important question about e-commerce is how e-commerce platforms can sell goods to consumers as much as possible to maximize the operating profit of e-commerce platforms.

On the one hand, e-commerce needs to provide high-quality goods to gain the reputation and trust of consumers, which makes consumers believe that they can buy the same good goods online as they can buy offline. On the other hand, e-commerce platforms need to adopt various promotion strategies to urge consumers to buy their products. As the quality of goods is related to merchants, e-commerce cannot fully control it, but the promotion and preferential strategy is one of the business decisions of e-commerce platforms. Different e-commerce platforms will launch a variety of marketing strategies to compete with each other.

Generally, the common promotion strategies of e-commerce platforms are as follows:

- Directly give consumers discounts in the specified category or brand of goods, including direct discount coupons, coupons can be enjoyed with different commodity combinations, there is a total order reached a certain price to enjoy the discount, buy one get one free, and so on.
- Promotions are held at certain times of the year, such as holidays. Famous shopping festivals include Black Friday, Christmas, Thanksgiving, National Day, Chinese Spring Festival, National Day, Double 11 shopping Festival, 618 shopping Festival and so on. During these special periods, almost all categories of goods will have different discounts, thus prompting consumers to buy.
- For every purchase, the e-commerce platform will record points for consumers. When the points reach a certain amount, the order amount can be deducted, so as to achieve the discount effect.
- Consumption on special platforms will have special services. For example, individual e-commerce platform shopping on mobile phone will have more coupons than web page. The implicit purpose is that e-commerce platform would like consumers to download their own software, so that they can spend more time browsing on the software.
- For consumers who buy VIP services on e-commerce platforms, e-commerce platforms will also offer exclusive discounts in different forms.

1.2 Problem Description

The committee provided several questions for our participants to study. Among the original several questions, there are those about sku attributes (Q1-2), brand promotion for users at different levels (Q3), discount effect analysis (Q4), promotion of differentiated services for PLUS members (Q5) and optimization of supply chain logistics (Q6-7).

The original seven questions of course provide us with a lot of research perspectives and ideas. What's different, however, is that we finally chose a question we wanted to study, namely how to predict the types of discounts that will occur when a user purchases a product from four kinds of information: ***User***, ***Sku***, ***Click*** and ***Order***.

This is a typical multi-classification problem because the type of discount in each user's order may contain a combination of multiple discounts, which makes it difficult to predict the type of discount in an order. The deep reason why we choose to study this problem is that we want to explore what factors have obvious influence or predictive ability on order types through training the model of predicting order types, so as to have a positive impact on the company and users. In other words, the research on this issue can not only promote the company to make reasonable order discounts and guide more users to place orders for consumption, but also enable users to know more about the company's discount pricing mechanism, so as to have more opportunities to get more discounts and obtain economic benefits in online shopping. It's a win-win situation for both.

1.2.1 Data Description

In the introduction article provided by the committee, the datasets provided has been introduced and described in detail. Here, we will not describe the data again for each dataset. Instead, we will make some additional explanations on four datasets involved in our research problem, so as to make our paper have a better summary of the data. Description tables for four of the data sets will be referenced.

（1） Sku

Field	Data type	Description	Sample value
sku_ID	string	Unique identifier of a product	b4822497a5
type	int	1P or 3P SKU	1
brand_ID	string	Brand unique identification code	c840ce7809
attribute1	int	First key attribute of the category	3
attribute2	int	Second key attribute of the category	60
activate_date	string	The date at which the SKU is first introduced	2018-03-01
deactivate_date	string	The date at which the SKU is terminated	2018-03-01

- ***sku_ID*** will be used for subsequent dataset joining.
- ***brand_ID*** has thousands of values, which is very difficult to convert. It can be removed later.
- Both ***attribute1*** and ***attribute2*** have a large number of missing values, which need to be processed in the preprocessing part of the data.
- Both ***activate_date*** and ***deactivate_date*** are not helpful for the problem we are exploring. They can be removed later to reduce redundant data.

（2） User

Field	Data type	Description	Sample value
user_ID	string	User unique identification code	000000f736
user_level	int	User level	10
first_order_month	string	First month in which the customer placed an order on JD.com (format: yyyy-mm)	2017-07
plus	int	If user is with a PLUS membership	0
gender	string	User gender (estimated)	F
age	string	User age range (estimated)	26–35
marital_status	string	User marital status (estimated)	M
education	int	User education level (estimated)	3
purchase_power	int	User purchase power (estimated)	2
city_level	int	City level of user address	1

- ***user_ID*** will be used for subsequent dataset joining.
- ***user_level***, ***plus***, ***education***, ***purchase_power***, and ***city_level*** can be used directly for standardization and model training.
- ***gender***, ***age***, and ***marital_status*** will be converted into multiple categorical variables using One-Hot Encoder before entering the model. ***first_order_month*** will be converted to a numeric variable which ban be calculated against other converted date variable.

（3） Click

Field	Data type	Description	Sample value
sku_ID	string	SKU unique identification code	b4822497a5
user_ID	string	User unique identification code	94ff800585
request_time	string	The time at which the customer clicks the SKU item page (format: yyyy-mm-dd HH:MM:SS)	2018-03-01 23:57:53
channel	string	The click channel	wechat

- ***sku_ID***, ***user_ID*** will be used for subsequent dataset joining.
- ***request_time*** will be converted to numeric variables like *first_order_month* in *User* dataset.

- ***channel*** needs to be transformed using the One-Hot Encoder, turning it into multiple categorical variables and then entering the model.

（4） Order

Field	Data type	Description	Sample value
order_ID	string	Order unique identification code	3b76bfcd3b
user_ID	string	User unique identification code	3cde601074
sku_ID	string	SKU unique identification code	443fd601f0
order_date	string	Order date (format: yyyy-mm-dd)	2018-03-01
order_time	string	Specific time at which the order gets placed (format: yyyy-mm-dd HH:MM:SS)	2018-03-01 11:10:40.0
quantity	int	Number of units ordered	1
type	int	1P or 3P orders	1
promise	int	Expected delivery time (in days)	2
original_unit_price	float	Original list price	99.9
final_unit_price	float	Final purchase price	53.9
direct_discount_per_unit	float	Discount due to SKU direct discount	5.0
quantity_discount_per_unit	float	Discount due to purchase quantity	41.0
bundle_discount_per_unit	float	Discount due to “bundle promotion”	0.0
coupon_discount_per_unit	float	Discount due to customer coupon	0.0
gift_item	int	If the SKU is with gift promotion	0
dc_ori	int	Distribution center ID where the order is shipped from	29
dc_des	int	Destination address where the order is shipped to (represented by the closest distribution center ID)	29

- ***order_ID***, ***sku_ID***, and ***user_ID*** will be used for subsequent dataset joining.
- ***order_date***, ***order_time*** will be converted to a numeric variable, and the new information is calculated after joining with other datasets before entering the model.
- The three attributes ***promise***, ***dc_ori*** and ***dc_des***, are not helpful for the problem we explored. They can be removed later to reduce redundant data. ***type*** is the same as that of *Sku* dataset, so it can be removed, too.
- ***original_unit_price*** and ***final_unit_price*** only require one of the input model training.
- ***direct_discount_per_unit***, ***quantity_discount_per_unit***, ***bundle_discount_per_unit***, ***coupon_discount_per_unit*** and ***gift_item*** are the five target variables of model training, which need to be converted into binary variables in the follow-up to participate in model training as labels of classification model.

1.2.2 What we hope to do in this paper

Based on the relevant datasets provided by JD.com, this paper will focus on the relationship between the discount types of consumers' orders and the sku attributes, user attributes and user click attributes.

For the original data, after the preliminary cleaning and processing, the important features are transformed and selected. The SMOTE resampling technique is used to expand the sample capacity and adjust the sample balance.

Then, common models, ensemble learning models and naïve bayes models are used to train and optimize model parameters according to training results and establish the relationship between order discount strategy and other dimensions of high importance.

Finally, the model with the best performance is selected and combined into a hybrid model, which is expected to be an effective tool and reference not only for the company to make a favorable combination of shopping orders for specific consumers but also for consumers to learn more about the decision mechanism of order discounts.

1.3 Main Contributions

The main contributions of this work can be summarized as follows:

- We creatively chose to explore the prediction problem of order discount type, transformed a multi-classification problem into several dichotomy problems, and optimized each dichotomy problem separately to achieve excellent prediction effect.
- In the modeling analysis, we used the SMOTE oversampling method to solve the imbalance problem of the partial type of discount data sample, which brought positive effect to the following model training.
- We used up to 15 machine learning classification models for training, including traditional models, ensemble learning models, and naïve bayes models. In the aspect of model selection, a wide range of parameter grids are used for model optimization, which greatly improves the accuracy of the model.
- Hybrid model is adopted instead of single model as the final output. For each type of discount, we filter and optimize the corresponding model. Finally, five single optimization models are combined to form a hybrid model that makes full use of and combines the advantages of different models. It has significant advantages over a single model and is more flexible.

- The final model explains the decision mechanism of order discount type to a certain extent by ranking the importance of predictors. The more reasonable the discount is established, the better the company can promote users to place orders, and the better users can get the final price discount of the order.

1.4 Paper Outline

In the remaining part of this paper, we will first summarize the relevant researches and papers on e-commerce problems.

Secondly, we will introduce the three types of machine learning models, SMOTE resampling method, and evaluation metrics of classification results involved in the experiment.

Next, we will do a complete data modeling process, including data preprocessing, feature engineering, sampling balance, model training and model validation.

In the last part of the paper, we will summarize our results, discuss and think about the problems in the whole process, and summarize the parts that can be expanded in the future.

2. RELATED WORK

2.1 Overview

This competition focuses on the discussion of issues related to E-commerce platform, so we need to enumerate and describe the problems that often occur in the operation of e-commerce, at the same time, summarize the excellent papers and academic achievements in this field, and try to find out the theories and methods worthy of reference and learning. In fact, the study of e-commerce has been going on for a long time, with the expansion of the electronic commerce platform in recent years, competition between different platforms is becoming increasingly fierce. In order to gain a competitive advantage in the market and more share, businesses need to have a more accurate way of pricing strategy, better promotion strategy, more rapid and efficient supply chain and logistics system. We will also carry out a literature review on these issues.

2.2 E-commence

(Liu et al., 2018) studies the information-sharing strategy for a retail platform on which multiple competing sellers distribute their products. The study developed a game-theoretic model where multiple sellers compete on a retail platform by selling substitutable products and the platform charges a commission fee for each transaction. The platform owns superior demand information and can control the accuracy level when sharing the information with the sellers. They found that the platform always has incentives to share the information, and such sharing benefits both the platform and all sellers. When there is no fairness constraint, the optimal strategy for the platform is to select a subgroup of sellers and truthfully share information with them. Under the fairness constraint, the platform must share the same information with all sellers and thus has an incentive to reduce the accuracy of the shared information. Moreover, the study identified a simple pricing mechanism that can achieve the optimal information-sharing outcome.

(Zhang et al., 2019) studied the value of short-lived and experientially oriented pop-up stores, a popular type of omnichannel retail strategy, on both retailers that participate in pop-up store events and retailing platforms that host these retailers. They conduct a large-scale, randomized field experiment with Alibaba Group involving approximately 800,000 customers. The study found that pop-up store visits substantially increased customers' subsequent expenditure at participating retailers' Tmall stores. In addition, from a platform perspective they showed that pop-

up store visits increased customers' purchases at retailers that sell related products on Tmall but did not participate in the pop-up store event.

(Feldman et al., 2019) compared the performance of two approaches for finding the optimal set of products to display to customers landing on Alibaba's two online marketplaces, Tmall and Taobao. The first approach we test is Alibaba's current practice. This procedure embeds thousands of product and customer features within a sophisticated machine learning algorithm that is used to estimate the purchase probabilities of each product for the customer at hand. The second approach uses a featured multinomial logit (MNL) model to predict purchase probabilities for each arriving customer. Experiments showed that despite the lower prediction power of our MNL-based approach, it generates significantly higher revenue per visit compared to the current machine learning algorithm with the same set of features. The study also conducted various heterogeneous-treatment-effect analyses to demonstrate that the current MNL approach performs best for sellers whose customers generally only make a single purchase.

(Aouad et al., 2019) introduced the click-based MNL choice model, a novel framework for capturing customer purchasing decisions in e-commerce settings. The main modeling idea is to assume that the click behavior within product recommendation or search results pages provides an exact signal regarding the alternatives considered by each customer. They study the resulting assortment optimization problem, where the objective is to select a subset of products, made available for purchase, to maximize the expected revenue. The study identified a simple greedy heuristic, which can be implemented at large scale, while also achieving near-optimal revenue performance in our experiments.

2.3 Supply Chain

From the perspective of supply and demand, E-commerce platform is an intermediary platform connecting commodities and customers. Even though some E-commerce platforms have their own products, it has to be admitted that most of the products on E-commerce platforms still come from many third-party brands in the market. Therefore, to build an efficient and fast supply chain and logistics system is an important issue for E-commerce platforms to solve. There are a lot of excellent research results in the academic circle, which are worth summarizing and discussing.

Large E-commerce platforms often have multilevel supply chain systems. How to solve the problem of information sharing in supply chain systems? For example, Many companies have embarked on initiatives that enable more demand information sharing between retailers and their upstream suppliers. (Lee et al., 2000) used analytical models to address questions for a simple two-level supply chain with nonstationary end demands. Their analysis suggests that the value of demand information sharing can be quite high, especially when demands are significantly correlated over time. (Cui et al., 2015) provide an empirical and theoretical assessment of the value of information sharing in a two-stage supply chain and they proved that the value of downstream sales information to the upstream firm stems from improving upstream order fulfillment forecast accuracy.

Also, researchers will consider the location of the inventory. (Shen et al., 2003) consider a joint location-inventory problem involving a single supplier and multiple retailers. The key problem is to determine which retailers should serve as distribution centers and how to allocate the other retailers to the distribution centers. Their study formulated this problem as a nonlinear integer-programming model and showed that this pricing problem can (theoretically) be solved efficiently. (Daskin et al., 2002) introduce a distribution center (DC) location model that incorporates working inventory and safety stock inventory costs at the distribution centers. The model is formulated as a non-linear integer-programming problem. They proposed Lagrangian relaxation solution algorithm and discussed the sensitivity of the results to changes in key parameters including the fixed cost of placing orders. Finally, they proved that significant reductions in these costs might be expected from E-commerce technologies. (Zheng et al., 2020) analyzed the existing distribution modes adopted by China's E-commerce enterprises. Based on the empirical analysis of the electronic mall at JD.com, this paper compared and investigated the different logistics distribution modes faced by E-commerce enterprises embracing the new features, new challenges, and new advantages of big data. The Analytic Hierarchy Process (AHP) method and entropy value are applied to investigate the e-commerce enterprise distribution choice mode and the Technique for Order Preference by Similarity to an Ideal Solution (TOPSIS) method is used to verify the model.

Moreover, (Bray et al., 2019) estimated the effect of supply chain proximity on product quality. They found that quality improves more slowly across geographically dispersed supply chains and supply chain distance is more detrimental to quality when automakers produce early-

generation models or high-end products, when they buy components with more complex configurations, or when they source from suppliers who invest relatively little in research and development.

2.4 Pricing Strategy

Compared with physical stores, E-commerce platforms often have a certain price advantage. Accurate pricing of commodities is a big problem for e-commerce platforms to solve, because there are many factors affecting pricing strategies.

In this regard, (Allon & Zeevi, 2011) tried to address the simultaneous determination of pricing, production, and capacity investment decisions by a monopolistic firm in a multi-period setting under demand uncertainty. The study analyzed the optimal decision with particular emphasis on the relationship between price and capacity.

(Cohen et al., 2020) considered the problem faced by a firm that receives highly differentiated products in an online fashion. The firm needs to price these products to sell them to its customer base. Products are described by vectors of features and the market value of each product is linear in the values of the features. The firm does not initially know the values of the different features but can learn the values of the features based on whether products were sold at the posted prices in the past. The study proposed a modification of the prior algorithm where uncertainty sets are replaced by their Löwner-John ellipsoids, showed how to adapt their algorithm to the case where valuations are noisy. Finally, they presented computational experiments to illustrate the performance of the algorithm.

2.5 Promotion Strategy

(Zhang et al., 2020) studied how a promotion strategy that offering customers a discount for products in their shopping cart affects customer behavior in the short and long term on a retailing platform. The study conducted a randomized field experiment involving more than 100 million customers and 11,000 retailers with Alibaba Group, the world's largest retailing platform. They randomly assigned eligible customers to either receive promotions for products in their shopping cart (treatment group) or not (control group). In the short term, the promotion program doubled the sales of promoted products and it boosted customer engagement, increasing the daily number of products customers viewed and their purchase incidence on the platform. Importantly, long-

term effects of price promotions on consumer engagement and strategic behavior spilled over to sellers that did not previously offer promotions to customers. This research documents the causal effects of dynamic pricing through price promotions on consumer behavior on a retailing platform, which have important implications for platforms and retailers.

Besides, (Liu et al., 2015) focused on Chinese online purchaser segmentation based on large volume of real transaction data on Taobao.com. The study firstly extracted and investigated Chinese online purchaser behavior indicators and classified them into six types by cluster analysis, these six categories are: economical purchasers, active-star purchasers, direct purchasers, high-loyalty purchasers, risk-averse purchasers and credibility-first purchasers. then it built an empirical model to estimate the sensitivity of each type of online purchasers to three mainstream promotion strategies (discount, advertising and word-of-mouth), and found that economical purchasers are the most sensitive to discount promotion; direct purchasers are the most sensitive to advertising promotion; active-star purchasers are the most sensitive to word-of-mouth promotion. Finally, the implications of online purchaser classification for marketing strategies were discussed.

(Liao et al., 2009) investigated factors of marketing communications and consumer characteristics that induce reminder impulse buying behavior. Both sales promotion strategy and its interaction effects with product appeal are found to have significant influences on reminder impulse buying. Specifically, an instant-reward promotion promotes stronger reminder impulse buying than a delayed-reward promotion. Furthermore, both a utilitarian product appeal with a price discount promotion and a hedonic product appeal with a premium promotion can encourage greater reminder impulse buying.

(Jiang et al., 2015) demonstrated that online price promotion and product recommendations should be jointly considered and optimally determined. Through attractive price discounts, E-sellers can motivate customers to purchase the promoted product; and by way of online recommendation systems, E-sellers can encourage customers to buy non-discounted items. The best promotional effect can be achieved by concurrently optimizing price promotion and product recommendation, because the loss from discount can be compensated for by the gains from the regular items. To maximize the profit, the study proposed an analytical model to help E-sellers exploit the potential of online promotion, and to maximize the influence of product recommendation on E-seller's sales and profits.

3. RESEARCH METHODOLOGY

3.1 Main Framework

The main framework of this paper revolves around building a model that can predict the type of e-commerce order discounts. The input of the model is the data related to the four data sets of *User*, *Sku*, *Click* and *Order*, and the output is the discount type of the corresponding order. Among them, there are five kinds of discounts for orders, respectively:

① *Direct* ②*Quantity* ③*Bundle* ④*Coupon* ⑤*Gift*

The main processes (**Figure 1**) are as follows:

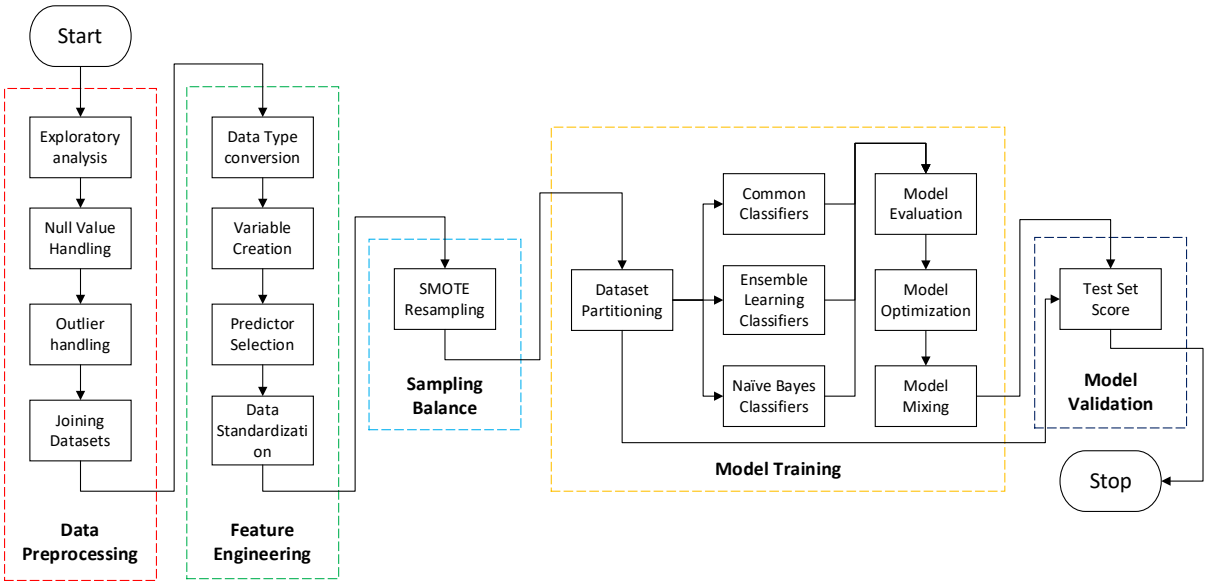


Figure 1. Main Process

(1) Data preprocessing. In this part, we will first conduct exploratory analysis of the original data set, including the analysis of the meaning of variables and the distribution of values. After that we will deal with null value and outliers in the datasets. Finally, we will join these datasets on different fields, as a whole dataset for subsequent processing.

(2) Feature engineering. In this step, for the category variables, we will convert them into numerical variables so as to conduct model training in the later stage. At the same time, some variables that may contribute to the later training will be generated and added to participate in the training as variables not included in the original datasets. Then we will screen predictors for training. Finally, we will standardize some continuous numerical variables.

(3) Sampling Balance. In this step, we will take the processed dataset and the features contained in it as samples and do *SMOTE* resampling according to different discount types. After

resampling, the ratio of samples with the certain discount type and without the certain discount type will be adjusted, in this way, the problem of overfitting caused by too few samples can be avoided, thus producing a positive promotion effect on model training.

(4) Model Training. The resampled dataset will be split into training set and test set. The training set is input into different models, and the samples of each different discount type will be trained to find the model with the best classification effect of each discount type. Then the corresponding model parameters will be further optimized as preparation for the mixed model. Encapsulate the best models for each discount type classification as a hybrid model.

(5) Model Validation. The test set will be input into the hybrid model to verify the validity of the model.

3.2 Resampling Method

In the process of machine learning modeling, the problem of sample imbalance often occurs, that is, the number of samples belonging to one category is too small or the proportion of samples in the whole sample set is too low compared with the samples of other categories.

For example, in the sample of a patient's physical examination, the number of patients with cancer is much lower than the number of patients without cancer. In this way, the imbalance of samples will occur, and the over-fitting phenomenon is very easy to occur in model training. When the proportion of positive and negative sample data in our sample data is extremely unbalanced, the effect of the model will be biased towards the results of most classes

In order to solve this problem, we can consider using resampling method. Resampling means eliminating the problem of unbalanced samples by increasing or decreasing too few of them. Resampling is divided into **(1) Over-sampling** and **(2) Under-sampling**.

Over-sampling means to achieve sample balance by increasing the number of minority samples in the classification. The most direct method is to simply copy the minority samples to form multiple records. The disadvantage of this method is that if the sample features are few, it may lead to the problem of overfitting. The improved oversampler method can produce a new synthetic sample by adding random noise and interfering data in a few classes or by some rules. The typical oversampling algorithms include **RandomOverSampler**, **ADASYN** and **SMOTE**.

Under-sampling is to achieve sample equilibrium by reducing the number of samples of most classes in the classification. The most direct method is to remove some samples of most classes

randomly to reduce the size of most classes. The disadvantage is that some important information in most classes samples will be lost the typical undersampling methods include **RandomUnderSampler**, **NearMiss** and **ClusterCentroids**

In general, over-sampling and under-sampling are more suitable for the unbalanced distribution of big data, especially the first method (over-sampling) is more widely applied.

It should be noted that resampling is required only when the proportion of the two categories in the dichotomy is very unbalanced (such as the large ratio of 100:1). If the proportion difference between the two is not large and the number of training sets is sufficient, resampling still may cause the training time to be extended without reason and reduce the efficiency of training.

3.2.1 SMOTE

SMOTE (Synthetic Minority Oversampling Technique) is an improved scheme based on random oversampling algorithm. Because random oversampling adopts the strategy of simply copying samples to increase a small number of samples, it is easy to cause the problem of model overfitting. Even if the information learned by the model is obtained, the information is too specific to be general.

The basic idea of SMOTE is to analyze the minority sample and add the new sample into the data set artificially according to the minority sample. For minority sample a, randomly select a nearest neighbor sample b, and then randomly select a point c from the line between a and b as the new minority sample. The specific algorithm flow is shown below.

(1) For each sample x_i in the minority class, the distance from it to all samples in the minority class sample set is calculated using Euclidean distance as the standard, and its K nearest neighbor is obtained.

(2) A sampling ratio was set according to the sample imbalance ratio to determine the sampling multiplier N. For each minority sample x_i, a number of samples were randomly selected from its K neighbors, assuming that the selected neighbor was $\widetilde{x_i}$.

(3) For each randomly selected nearest neighbor $\widetilde{x_i}$, the new sample is constructed with the original sample respectively according to the following Equation (1) and **Figure 2**.

$$x_{new} = x_i + rand(0,1) \times (\widetilde{x_i} - x_i) \quad (1)$$

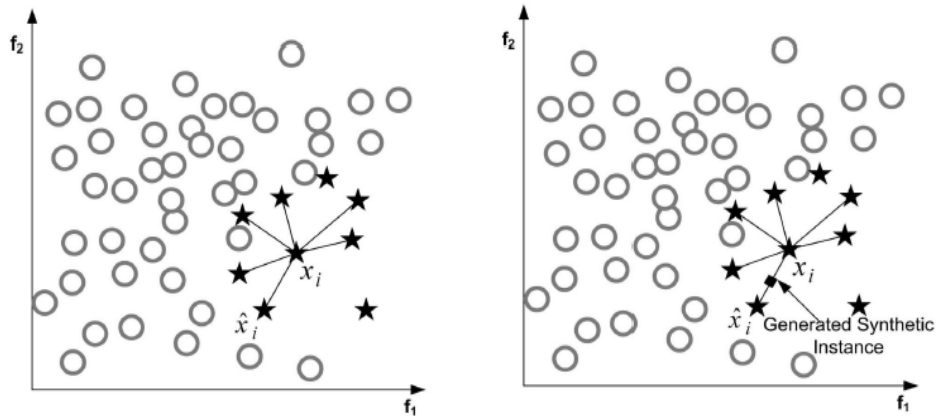


Figure 2. Generated Synthetic Instance in SMOTE

There are two main problems in this algorithm.

(1) First, there is some blindness in the selection of nearest neighbor. As can be seen from the algorithm process above, in the process of algorithm execution, K value needs to be determined, that is, how many neighbor samples need to be selected, which needs to be solved by users themselves. As can be seen from the definition of K value, the lower limit of K value is M value (M value is the number of neighbor samples randomly selected from K neighbors, and there are M<K), the size of M can be determined according to the number of negative class samples, the number of positive class samples and the final equilibrium rate to be reached in the data set. However, there is no way to determine the upper limit of K value, which can only be tested repeatedly according to the specific data set. So, it's not known how to determine the value of K in order for the algorithm to be optimal.

(2) In addition, the algorithm is unable to overcome the problem of data distribution of unbalanced datasets, and it is easy to generate the problem of distribution marginalization. Because of the negative samples distribution determines the choice of neighbors, if a negative class samples were in negative edge of the distribution of sample set, the resulting negative class sample and adjacent sample “artificial” sample will be at the edge of and will be more and more marginalized, which blurs the positive samples and negative samples border, and the boundary is becoming more and more blurred. This boundary fuzziness improves the balance of data set but increases the difficulty of classification algorithm.

Algorithm 1 is the pseudocode for SMOTE:

Algorithm 1: *SMOTE* (T, N, k)

Input: Number of minority class samples *T*; Amount of SMOTE *N%*; Number of nearest neighbors *k*
Output: *(N/100) * T* synthetic minority class samples
Procedure:

```
1: (* If N is less than 100%, randomize the minority class samples as only a random percent of them
   will be SMOTEd. *)
2: if N < 100
3:  then Randomize the T minority class samples
4:      T = (N / 100) * T
5:      N = 100
6: endif
7: N = (int)(N/100) (* The amount of SMOTE is assumed to be in integral multiples of 100. *)
8: k = Number of nearest neighbors
9: numattrs = Number of attributes
10: Sample [] []: array for original minority class samples
11: newindex: keeps a count of number of synthetic samples generated, initialized to 0
12: Synthetic [] []: array for synthetic samples
      (* Compute k nearest neighbors for each minority class sample only. *)
13: for i :1 to T
14:       Compute k nearest neighbors for z, and save the indices in the nnarray
15:       Populate (N, i, nnarray)
16: endfor

     Populate (N, i, nnarray) (* Function to generate the synthetic samples. *)
17: while N ≠ 0
18:      Choose a random number between 1 and k, call it nn. This step chooses one of the k nearest
neighbors
              of i.
19:     for attr: 1 to numattrs
20:         Compute: dif = Sample[nnarray[nn]] [attr] – Sample[i][attr]
21:         Compute: gap = random number between 0 and 1
22:         Synthetic[newindex][attr] = Sample[i][attr] + gap * dif
23:      endfor
24:      newindex++
25:      N = N – 1
26: endwhile
27: return (* End of Populate. *)
End of Pseudo-Code
```

3.3 Classification Model

As the core element of machine learning classification, classification model plays a decisive role in the final classification effect. A good classification model can effectively learn the rules in the training set, so as to achieve a good classification effect in the prediction of classification. At the same time, different types of classification models can achieve different effects when solving different problems. For example, linear classifiers such as logistic regression classifiers, LDA, support vector machines, etc. tend to achieve good results when dealing with linear

separable problems. When dealing with linear indivisible problems, nonlinear classifiers such as decision tree, random forest and neural network are better.

In the modeling and analysis module of this paper, we will use different classification models to fit the training set, make predictions on the test set, find out the optimal classification model corresponding to different target variables, and finally combine into a hybrid model. The models to be used in the fourth part of this paper are classified into three categories, namely Common Classifier, Ensemble Learning Classifier and Naive Bayesian Classifier. The following is an overview of these models.

3.3.1 Common Classifier

Common classifiers include **Logistic Regression, KNN, LDA, QDA, SVM, Neural Network,** and **Decision Tree**. As these models are the most common classifier in machine learning modeling, they have a long history of development and the principle of classification is well known, so this paper will not give too much overview of their underlying algorithms. However, for these common models, the following problems should be paid attention to during the optimization process:

- In **Logistic Regression** training, regularization is often needed to solve the problem of model overfitting. Among them, whether to choose L1 regularization or L2 regularization is a common optimization problem. Similarly, the regularization parameter, the penalty term of the error function, also needs to be optimized to achieve the best effect.
- The most common optimization parameter of **KNN** classifier is the selection of K value. In addition to the need to choose the appropriate K value, how to define the distance between the data is also a problem to be considered. The parameters can also be optimized when choosing Euclidean distance or Manhattan distance, weighted distance or unweighted distance.
- For dichotomies, **LDA** is aimed at: data are subject to Gaussian distribution, with different mean values and the same variance. **QDA** is for: the data obey gaussian distribution, and the mean value is different, the variance is different. For LDA classifier, the optimization methods include Singular value decomposition (SVD), Least square method (LSQR) and Eigenvalue decomposition (Eigen), which correspond to three kinds of optimizers and need to be optimized as model parameters. Compared with LDA, QDA requires more parameters to be estimated. When the data volume is large enough, variance will not be a major problem, and QDA will perform better.

- The kernel function of **SVM** is the first parameter to be optimized. Secondly, the parameter Gamma of partial kernel function and the penalty parameter C of error function should be optimized.
- The **Multilayer-Perceptron Neural Network (MLP-NN)**, as the most complex model, has many hyperparameters that can be optimized. Because the Python program library *Sklearn* adopted in the experiment does not support GPU operation, and the sample number of the original data set is more than 400,000, not all hyperparameters can be optimized. The hidden layer parameter and regularization parameter are two important super parameters that can be optimized first.
- **Decision Tree** is a tree structure in which each internal node represents a judgment on an attribute, each branch represents the output of a judgment result, and finally each leaf node represents a classification result. Decision Tree generation algorithms are ID3, C4.5 and C5.0, etc. For the Decision Tree, the *"criterion"* to be used such as Gini or Entropy, the maximum depth of the tree - *"max_depth"*, the policy to specify when splitting nodes - *"splitter"*, the number of features to be considered for optimal partitioning - *"max_features"*, and the maximum number of leaf nodes - *"max_leaf_nodes"* are hyperparameters often used for optimization.

3.3.2 Ensemble Learning Classifier

The main idea of integrated learning is to train several weak classifiers at the same time, and then combine these weak classifiers to make prediction. Its core idea is how to train multiple weak classifiers and how to combine these weak classifiers. The implementation idea is shown in the **Figure 3**.

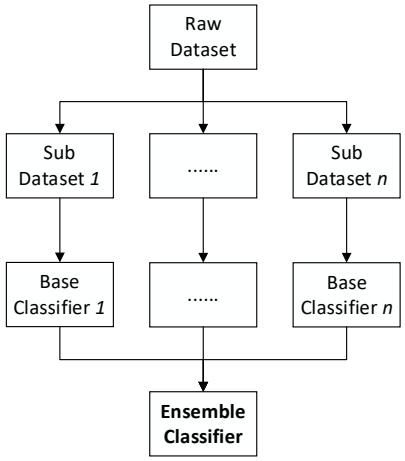


Figure 3. The Implementation Idea of Ensemble Learning

Ensemble learning has a high accuracy rate in machine learning models, but its disadvantage is that the training process of the model may be complicated, and the efficiency is not very high. Currently, ensemble learning is mainly based on two algorithms: **Bagging** and **Boosting**. The representative algorithms of the former are **Bagging Tree**, **Random Forest** and **Extra Tree**, while the representative algorithms of the latter are mainly **AdaBoost Tree**, **GBDT** and **XgBoost Tree**. In terms of reducing the generalization error of the model, Bagging reduces the variance, Boosting the model bias.

In ensemble learning, the performance of a combined classifier is usually better than that of a single classifier, but two conditions need to be met: (1) A base classifier should be independent of each other, or at least have low correlation. (2) A base classifier should be better than a random classifier, that is, it should have an accuracy rate higher than 0.5 in dichotomy.

However, because the base classifiers are trained to solve the same problem, it is obviously difficult to be independent of each other. Therefore, according to the generation mode of a single classifier, ensemble learning can be divided into two categories:

1. **Base classifiers do not have strong dependencies on each other and generate sequences in parallel, such as Bagging.**

Bagging, also known as Bootstrap Aggregating, acquires different data sets by repeated sampling from the training set with uniform probability. Bagging improves the generalization error by reducing the variance of the base classifier. It is important to note that bagging helps reduce the error caused by random fluctuations in the training data if the base classifier is unstable. If the base classifier is stable, that is, the slight change in the training data set has little influence on it, then the error of the combined classifier is mainly caused by the base classifier migration. In this case, Bagging may not improve the base classifier significantly and may even reduce the performance of the classifier.

Bagging Tree is formed by combining Bagging and Decision Tree.

Based on Bagging Tree, **Random Forest** is derived. Random Forest improves the establishment of Decision Tree. For ordinary Decision Tree, we will select an optimal feature among all N sample features on the node to make the division of left and right subtrees of the decision tree. But the random forest selected M sample features at random nodes ($M<N$). Then, an optimal feature is selected from the randomly selected M sample features to make the division of

the left and right subtrees of the decision tree. This further enhances the generalization ability of the model.

Extra Tree is a derivative of Random Forest, and its principle is almost the same as the Random Forest, except that: (1) For the training set of each Decision Tree, the Random Forest adopts random sampling bootstrap to select the sampling set as the training set of each decision Tree. While Extra Tree generally does not adopt random sampling, that is, each Decision Tree adopts the original training set. (2) After the partition features are selected, the decision tree of Random Forest will choose an optimal partition point based on Gini, Entropy and other criteria, which is the same as the traditional decision tree. But Extra Tree is more random. It will randomly choose a criterion to divide the Decision Tree.

Because the partition points are randomly selected, rather than the optimal partition point, the size of the generated decision tree will generally be larger than the Random Forest. The variance of Extra Tree is reduced compared with Random Forest; the bias is increased. Therefore, in some cases, Extra Tree has a better generalization ability than Random Forest.

2. Base classifiers have strong dependencies on each other and generate sequences sequentially, such as Boosting.

Boosting's working principle is to train the weak learner L_i based on the initial training set with weight, update the weight of the sample based on learning error rate, boosting the weight of the classified sample, making them pay more attention to the data with high misclassification later. The weak learner L_{i+1} is trained based on the updated training set, so the cycle training is done. If the number of iterations is set to T, then the training ends when $n = T$. Finally, a strong learner is formed by combining certain strategies. Boosting algorithms are the two most famous algorithms, AdaBoost and Gradient Boost.

The AdaBoost combined with the Decision Tree constitutes the **AdaBoost Tree**.

Gradient Boost combined with the Decision Tree forms the classic model **GBDT**.

XgBoost, whose full name is eXtreme Gradient Boosting, is ensemble with the decision Tree to make up the **XgBoost Tree**. Compared with the traditional GBDT, XgBoost Tree performs the second-order Taylor expansion of the cost function, using both the first and second derivatives in the optimization phase, while GBDT only uses the information of the first derivative in the optimization phase. XgBoost Tree, on the other hand, adds regular terms to the loss function and

uses them to control the complexity of the model. Variance of the model is reduced to prevent overfitting.

3.3.3 Naïve Bayesian Classifier

Naïve Bayes classifier is a simple probabilistic classifier based on Bayes' theorem. It assumes independence between features. The ideological basis of Naïve Bayes is as follows: for a given item to be classified, the probability of the occurrence of each category under the condition of such occurrence is solved, which is the largest is considered to be in which category the item to be classified belongs. The advantages of this algorithm lie in its simplicity and high learning efficiency, and it can be compared with Decision Tree and Neural Network in some classification problems. However, the accuracy of the algorithm is affected to some extent because the algorithm assumes the independence between independent variables (conditional feature independence) and the normality of continuous variables.

To understand the naive Bayes classifier, we must first understand **Bayes' theorem**. The latter is actually the formula for calculating the conditional probability.

According to **Bayes' theorem** (Equation 2):

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{2}$$

Suppose an individual has n characteristics, F_1, F_2..., F_n. There are m categories, namely C_1, C_2...,C_m. The Bayesian classifier calculates the category with the highest probability, which is the maximum value of the following formula (Equation 3):

$$P(C|F_1, F_2 \ldots, F_n) = \frac{P(F_1F_2 \ldots F_n|C)P(C)}{P(F_1F_2 \ldots F_n)} \tag{3}$$

Because $P(F_1F_2 \ldots F_n)$ is the same for all the categories, it can be omitted. The problem becomes to find $P(F_1F_2 \ldots F_n|C)P(C)$.

Naive Bayes Classifier assumes that all features are independent of each other, therefore we can get Equation 4:

$$P(F_1F_2 \ldots F_n|C)P(C) = P(F_1|C)P(F_2|C) \ldots P(F_n|C)P(C) \tag{4}$$

Each term on the right side of the Equation 4 can be obtained from the statistical data, from which we can calculate the corresponding probability of each category, so as to find the class with the highest probability. Although the assumption of all features are independent of each other is

unlikely to be true in reality, it can greatly simplify the calculation, and studies have shown that it has little impact on the accuracy of classification results.

3.4 Evaluation Metric

When a classifier fits the training set, it can make classification prediction on the test set, and then the prediction result of the classifier on the test set will be produced. At the same time, the original classification results of the test set will be labeled and compared with the predicted results of the classifier. From this, many evaluation criteria are derived to evaluate the classification effect of a classifier.

3.4.1 Confusion Matrix

Confusion matrix is a situation analysis table that summarizes the prediction results of classification models in machine learning. In the form of matrix, records in the dataset are summarized according to two criteria of real classification and classification predicted by classification models.

Where the rows of the matrix represent the true value and the columns of the matrix represent the predicted value. Take dichotomy as an example to look at the representation as **Table 1**.

Confusion Matrix	**Actually Positive**	**Actually Negative**
Predicted Positive	**True Positive (TP)**	**False Positive (FP)**
Predicted Negative	**False Negative (FN)**	**True Negative (TN)**

Table 1. Confusion Matrix

Variables in the confusion matrix are defined as follows:

- **TP** (True Positive):

  Predict the Positive class to the Positive class. True class is 1, and the prediction is 1.

- **FN** (False Negative):

  Predicting the Positive class to the Negative class. True class is 1, and the prediction is 0.

- **FP** (False Positive):

  Predicting the Negative class as a Positive class. True class is 0, and the prediction is 1.

- **TN** (True Negative):

  Predicting the Negative class to be Negative class. True class is 0, and the prediction is 0.

Based on the values of the 4 variables in the confusion matrix, many evaluation criteria can be derived to measure the classification effect of the classifier, among which **Accuracy, Precision, Recall** and **F1-score** are most commonly used.

3.4.2 Accuracy, Precision, Recall, F1-Score

According to the number of TP, FN, FP and TN in the existing confusion matrix, we can calculate the four criteria used to evaluate the performance of the classifier. The following are the calculation formulas and meanings of the four classification evaluation criteria.

- **Accuracy** (ACC):

$$\boldsymbol{Accuracy} = \frac{(TP + TN)}{(\ TP + TN + FP + \ FN)} \tag{2}$$

The formula of **Accuracy** is shown in Equation (2). Accuracy is the most common evaluation criterion. It represents the number of correct samples divided by the number of samples. Generally speaking, the higher the accuracy, the better the classifier.

The accuracy is limited. In the case of the imbalance of positive and negative samples, the evaluation index of accuracy has great defects. For example, in Internet advertisements, the number of clicks is very small, generally only a few thousandths. If acc is used, even if all are predicted to be negative (no click) ACC has more than 99%, which is meaningless.

- **Precision** (Positive Prediction Value):

$$\boldsymbol{Precision} = \frac{TP}{(TP\ +\ FP)} \tag{3}$$

The formula of **Precision** is shown in Equation (3). It represents the proportion of positive examples that are actually positive examples in an example divided into positive examples.

- **Recall** (Sensitivity; Hit Rate; True Positive Rate):

$$\boldsymbol{Recall} = \frac{TP}{(TP\ +\ FN)} \tag{4}$$

The formula of **Recall** is shown in Equation (4). It represents the proportion of correct classification among all positive examples and measures the classifier's ability to recognize positive examples.

- **F1-score**

Sometimes there are contradictions between Precision and Recall, so they need to be considered comprehensively. **F-measure** (also known as **F-score**) is the weighted harmonic average of **Precision** and **Recall**, which is shown in Equation (5)**:**

$$\boldsymbol{F\ score} = \frac{(\alpha^2 + 1)Precision \times Recall}{\alpha^2(Precision + Recall)} \quad (5)$$

The most common formula is when the $\alpha = 1$. The formula becomes Equation (6).

$$\boldsymbol{F1\ score} = \frac{2 \times Precision \times Recall}{(Precision + Recall)} = \frac{2TP}{(2TP\ +\ FP\ + FN)} \quad (6)$$

3.4.3 TPR, FPR, ROC & AUC

In addition to the above 4 commonly used classification evaluation criteria. Moreover, we can draw the ROC curve based on the predicted results of the classifier and the classification threshold, and then obtain the AUC value. We can evaluate the classifier through these two evaluation indicators.

The classifier usually sets a threshold. If a sample has a prediction greater than this value, it is classified as positive, and if it is less than this value, it is classified as negative. If we reduce this threshold, more samples will be identified as positive, and fewer samples will be identified as negative. This improves the recognition rate of positive classes, but also causes more negative classes to be misidentified as positive classes. To reflect this change, the ROC curve was introduced, which can be used to evaluate a classifier.

ROC (Receiver Operating Characteristic) curve is a curve with false positive rate (FPR) and true rate (TPR) as the axis. In the ROC curve, the X-coordinate of each point is FPR, and the Y-coordinate is TPR. TPR stands for the probability of classifying positive examples into correct ones, while FPR stands for the probability of misclassifying negative examples into positive ones. This also represents the trade-off between TP (true rate) and FP (false positive rate) of the classifier. A typical ROC curve is shown in **Figure 4**.

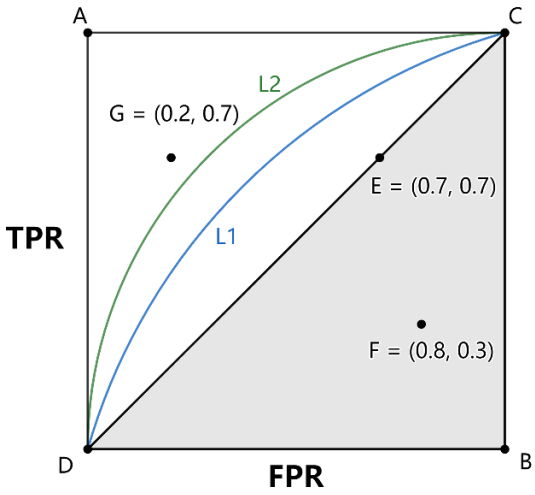


Figure 4. ROC Curve Sample

In **Figure 4**:

(1) The larger the area bounded by the curve and the FPR axis, the better the performance, that is, the performance corresponding to the L2 curve in the figure is better than that corresponding to the L1 curve. In other words, the closer the curve is to point A (upper left), the better the performance of the classifier. The closer the curve is to Point B (lower right), the worse the performance of the classifier. Point A has the most perfect classification effect, while point B has the most poor performance.

(2) The points on the C-D line indicate that the algorithm performance and the classification effect of the random classifier are the same. Such as points C, D, E. Being above C-D (that is, the curve is inside the white triangle) indicates that the algorithm performs better than the random classifier, such as the point G. Being below C-D (that is, the curve is inside the grey triangle) indicates that the algorithm's performance is worse than that of the random classifier, such as point F.

The Area Under the ROC Curve is called the **AUC**. AUC is defined as the **Area Under the Curve**. Obviously, the value of this Area will not be greater than 1. Generally speaking, the higher the AUC value is, the higher the accuracy will be. The value of AUC corresponds to the following situations:

(1) **AUC = 1**, the classifier is a perfect classifier. When using this prediction model, perfect prediction can be obtained no matter what threshold is set. For the vast majority of predictions, there is no perfect classifier.

(2) **0.5 < AUC < 1**, the classifier is better than random classifier. If the classifier sets a reasonable threshold, it will have predictive value.

(3) **AUC = 0.5**, the classifier is the same as the random classifier (such as coin toss). The model has no predictive value.

(4) **AUC < 0.5**, the classifier is worse than the random classifier. But as long as the results are always taken contrary to the prediction, it will be better than the random classifier.

4. MODELING ANALYSIS

According to the main framework mentioned in the third part, the modeling analysis is divided into 5 parts: (1)**Data preprocessing** (2)**Feature engineering** (3)**Sampling Balance** (4)**Model training** (5)**Model mixing**.

4.1 Data preprocessing

In the process of data preprocessing, we first change the string format of some data in the original dataset into numerical data which is acceptable to the model. Next, the meaningless missing values in the data set are removed.

It is worth noting that in the **SKU** dataset, due to the two attributes, ***Attribute1*** and ***Attribute2***, there are too many missing values. If the row with the missing values is removed directly, the subsequent data table connection will result in a large number of missing data from other data tables, resulting in insufficient training samples. Hence, in this step, we need to fill in the missing value, and finally we choose to use the mode of each of the two attributes to fill in the missing value.

At the same time, different from the general modeling process, in this step, we do not standardize the data for the time being, because we have not screened the features and need to retain the original information of the data.

4.2 Feature engineering

In the process of machine learning modeling, after obtaining original data and data preprocessing, feature engineering is required to input data into the model for training.

In this step, we divide the feature project into three parts: variable creation, attribute transformation and factor selection to make full use of the information in the data set.

4.2.1 Variable Creation

In the variable creation section, there are the following steps:

(1) First, we connect the ORDER data set with SKU and USER data set through the ID of the user and sku to get the joined dataset.

(2) In the ORDER data set, The times of purchase of each user and times of purchase of each sku are counted and incorporated into the previous data set. The fields used for connection are ***user_ID*** and ***sku_ID*** respectively.

(3) In the CLICK dataset, the number of clicks per user on a certain sku and the total number of clicks, which is also known as the "***User click index***". Divide these two variables to get

the user's "***Click ratio***" for each item -- a ratio that represents how interested the user is in different items.

At the same time, the total number of clicks on each item is calculated as the "***Sku click index***". Then, the earliest and latest time for each user to click on a sku is subtracted to obtain the "***Click duration***" of each user for a sku, which can represent the user's attention cycle for a SKU.

Next, count the number of different SKUs clicked by each user (that is, how many SKUs a user has clicked on) and the number of channels used by the clicks (that is, how many channels the user has clicked on Skus).

Finally, connect all of these new variables to the previous dataset. The two fields used for the connection are the ***user_ID*** and the ***sku_ ID***. Note that the joined dataset needs to match the user and sku at the same time.

(4) After obtaining the connected data set, we can calculate the time interval between a user's last click on a certain sku and the final purchase of the sku, which can represent the user's urgency to purchase a sku.

(5) At this time, we have obtained a large dataset joining four datasets including ORDER, CLICK, SKU and USER and related information derived from them.

4.2.2 Attribute Transformation

In this part, we mainly standardized part of numerical variables and converted categorical variables into numerical variables. For categorical variables, Label Encoder and One-Hot Encoder can be used to process them as numeric variables, respectively.

For variables with degree relationship (such as user's age, etc.), we use Label Encoder to convert. For variables without degree relationship (such as user's gender, marital status, etc.), we use One-Hot Encoder to convert.

Specially, in SKU dataset there is a categorical variable, ***brand_ID***, has nearly 2000 kinds of values. Even we used One-Hot Encoder and PCA dimension reduction, still there will be hundreds of extra variables. If this attribute is retained, there is a very high time cost for model training. Therefore, in this part, we do not transform this attribute, but choose to directly remove it from the predictors of model training in the next part.

4.2.3 Predictor Selection

After the above two parts of processing, in this step we will select the predictors that will eventually be used to input into the model. First, we delete name variables such as ***user_ID*** because they have no practical significance. Second, we remove duplicates or collinear variables. Finally, after all processing, the final data set contains the following factors, which is shown in **Table 2**:

Order	User	Sku	Click
• original_unit_price • sku_buy_index • user_buy_index	• user_level • plus • education • city_level • purchase_power • age_num • gender_F • gender_M • gender_U • marital_status_M • marital_status_S • marital_status_U	• type • attribute1 • attribute2	• user_click_on_single_sku • user_click_index • ratio_single_sku_click_to_all_sku_click • diff_sku • sku_click_index • channel_app • channel_mobile • channel_others • channel_pc • channel_wechat • diff_channel • click_duration • cilck_order_duration

Table 2. Final Predictors

4.3 Sampling Balance

After the above processing, we have obtained a final dataset with **32 predictors and 426287 rows**. At the same time, we also have five target datasets as labels, which are: ***Direct, Quantity, Bundle, Coupon, Gift***.

First, the original dataset was randomly divided into training set and test set according to a ratio of 8:2. In the process of cross validation, different training sets and validation sets are further divided according to the number of K-Folds (in our training, K is 3), which are used to optimize the hyperparameters of the model. The test set will be used to verify the validation of the model and will not participate in the training and optimization process of the model. The split process is shown in **Figure 5**.

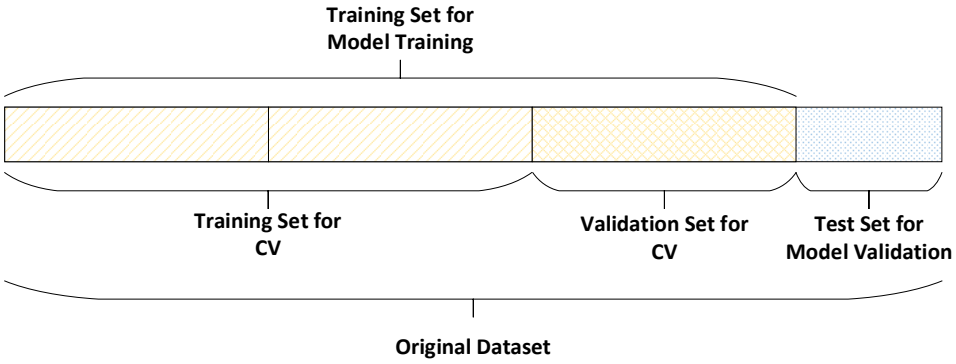


Figure 5. Original Dataset Split

Next, what we need to think about is sampling balance. As the method explained in the third part of the article, for the training set with sampling imbalance problem, use SMOTE resampling method to adjust the ratio of two kinds of samples in the training set to a proper level (at least equal to **1:10**), and then put into different machine learning model for training.

Note that, the subset used to do SMOTE resampling is the training set for model training after partitioning, which means the training set can be larger than before. However, the test set for model validation is the subset used to test the classification effect of the model outside of the training set, which means it will remain unchanged. If we do SMOTE resampling on test set, the classification effect will be overestimated because some samples are generated but not original. This will cause the results of the classifier to look very good, but this is not the real effect of classification. The Change of training set after SMOTE is shown in **Table 3**.

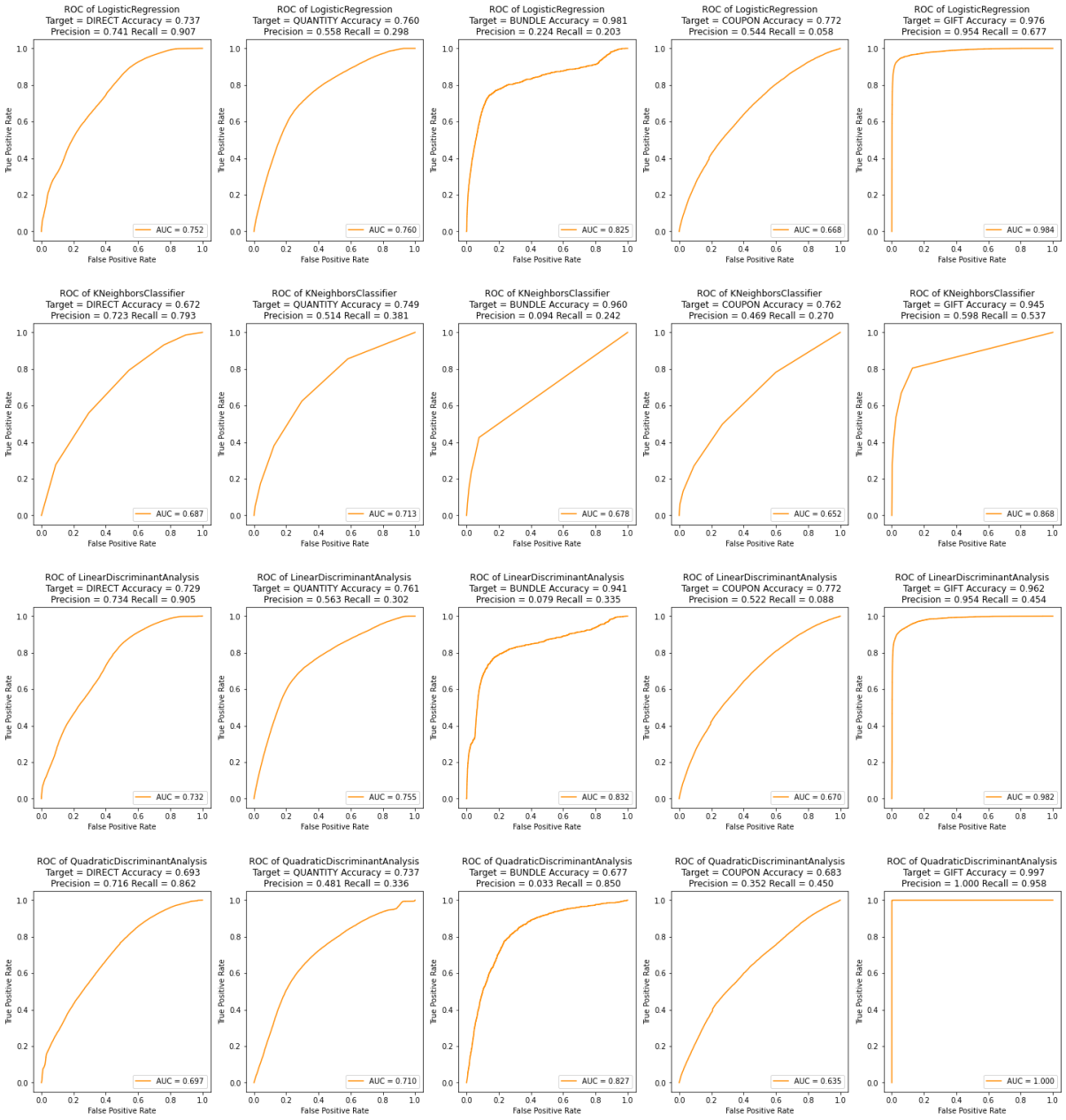
Class / Target		DIRECT	QUANTITY	BUNDLE	COUPON	GIFT
Before SMOTE	**1**	219107	253825	336522	261830	318375
	0	121922	87204	4507	79199	22654
After SMOTE	**1**	Not Changed	Not Changed	336522	Not Changed	318375
	0			**33652**		**31837**

Table 3. Change of Training Set Class After SMOTE Resampling

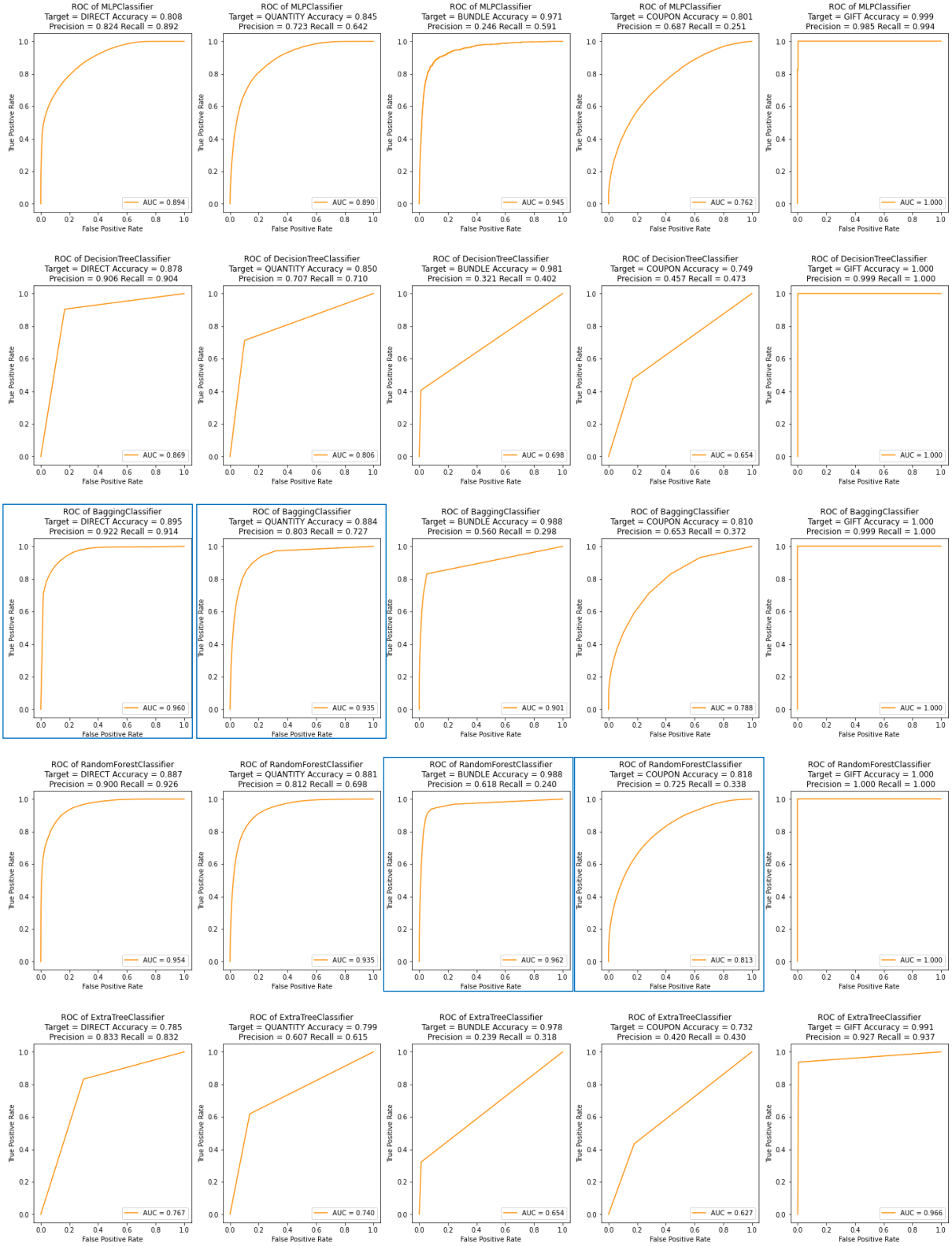
4.4 Model Training

(1) First, the target variable is selected. Since there are 5 types of target variables, we will find the model with the best training prediction effect for each target variable one by one, and finally combine them into a hybrid model.

(2) After the selection of target variables, 15 models of 3 kinds will be used for training, respectively. For each model, first training will depend on the default hyperparameters. K-Fold cross validation will be used to remove the training result deviation caused by the randomness of training set partition. Results are shown in **Figure 6** and **Table 4**.



(ROC Curve for **SVM** is unavailable due to the lengthy training process)


ROC of MLPClassifier
Target = DIRECT Accuracy = 0.808
Precision = 0.824 Recall = 0.892
AUC = 0.894
ROC of MLPClassifier
Target = QUANTITY Accuracy = 0.845
Precision = 0.723 Recall = 0.642
AUC = 0.890
ROC of MLPClassifier
Target = BUNDLE Accuracy = 0.971
Precision = 0.246 Recall = 0.591
AUC = 0.945
ROC of MLPClassifier
Target = COUPON Accuracy = 0.801
Precision = 0.687 Recall = 0.251
AUC = 0.762
ROC of MLPClassifier
Target = GIFT Accuracy = 0.999
Precision = 0.985 Recall = 0.994
AUC = 1.000
ROC of DecisionTreeClassifier
Target = DIRECT Accuracy = 0.878
Precision = 0.906 Recall = 0.904
AUC = 0.869
ROC of DecisionTreeClassifier
Target = QUANTITY Accuracy = 0.850
Precision = 0.707 Recall = 0.710
AUC = 0.806
ROC of DecisionTreeClassifier
Target = BUNDLE Accuracy = 0.981
Precision = 0.321 Recall = 0.402
AUC = 0.698
ROC of DecisionTreeClassifier
Target = COUPON Accuracy = 0.749
Precision = 0.457 Recall = 0.473
AUC = 0.654
ROC of DecisionTreeClassifier
Target = GIFT Accuracy = 1.000
Precision = 0.999 Recall = 1.000
AUC = 1.000
ROC of BaggingClassifier
Target = DIRECT Accuracy = 0.895
Precision = 0.922 Recall = 0.914
AUC = 0.960
ROC of BaggingClassifier
Target = QUANTITY Accuracy = 0.884
Precision = 0.803 Recall = 0.727
AUC = 0.935
ROC of BaggingClassifier
Target = BUNDLE Accuracy = 0.988
Precision = 0.560 Recall = 0.298
AUC = 0.901
ROC of BaggingClassifier
Target = COUPON Accuracy = 0.810
Precision = 0.653 Recall = 0.372
AUC = 0.788
ROC of BaggingClassifier
Target = GIFT Accuracy = 1.000
Precision = 0.999 Recall = 1.000
AUC = 1.000
ROC of RandomForestClassifier
Target = DIRECT Accuracy = 0.887
Precision = 0.900 Recall = 0.926
AUC = 0.954
ROC of RandomForestClassifier
Target = QUANTITY Accuracy = 0.881
Precision = 0.812 Recall = 0.698
AUC = 0.935
ROC of RandomForestClassifier
Target = BUNDLE Accuracy = 0.988
Precision = 0.618 Recall = 0.240
AUC = 0.962
ROC of RandomForestClassifier
Target = COUPON Accuracy = 0.818
Precision = 0.725 Recall = 0.338
AUC = 0.813
ROC of RandomForestClassifier
Target = GIFT Accuracy = 1.000
Precision = 1.000 Recall = 1.000
AUC = 1.000
ROC of ExtraTreeClassifier
Target = DIRECT Accuracy = 0.785
Precision = 0.833 Recall = 0.832
AUC = 0.767
ROC of ExtraTreeClassifier
Target = QUANTITY Accuracy = 0.799
Precision = 0.607 Recall = 0.615
AUC = 0.740
ROC of ExtraTreeClassifier
Target = BUNDLE Accuracy = 0.978
Precision = 0.239 Recall = 0.318
AUC = 0.654
ROC of ExtraTreeClassifier
Target = COUPON Accuracy = 0.732
Precision = 0.420 Recall = 0.430
AUC = 0.627
ROC of ExtraTreeClassifier
Target = GIFT Accuracy = 0.991
Precision = 0.927 Recall = 0.937
AUC = 0.966
True Positive Rate
False Positive Rate

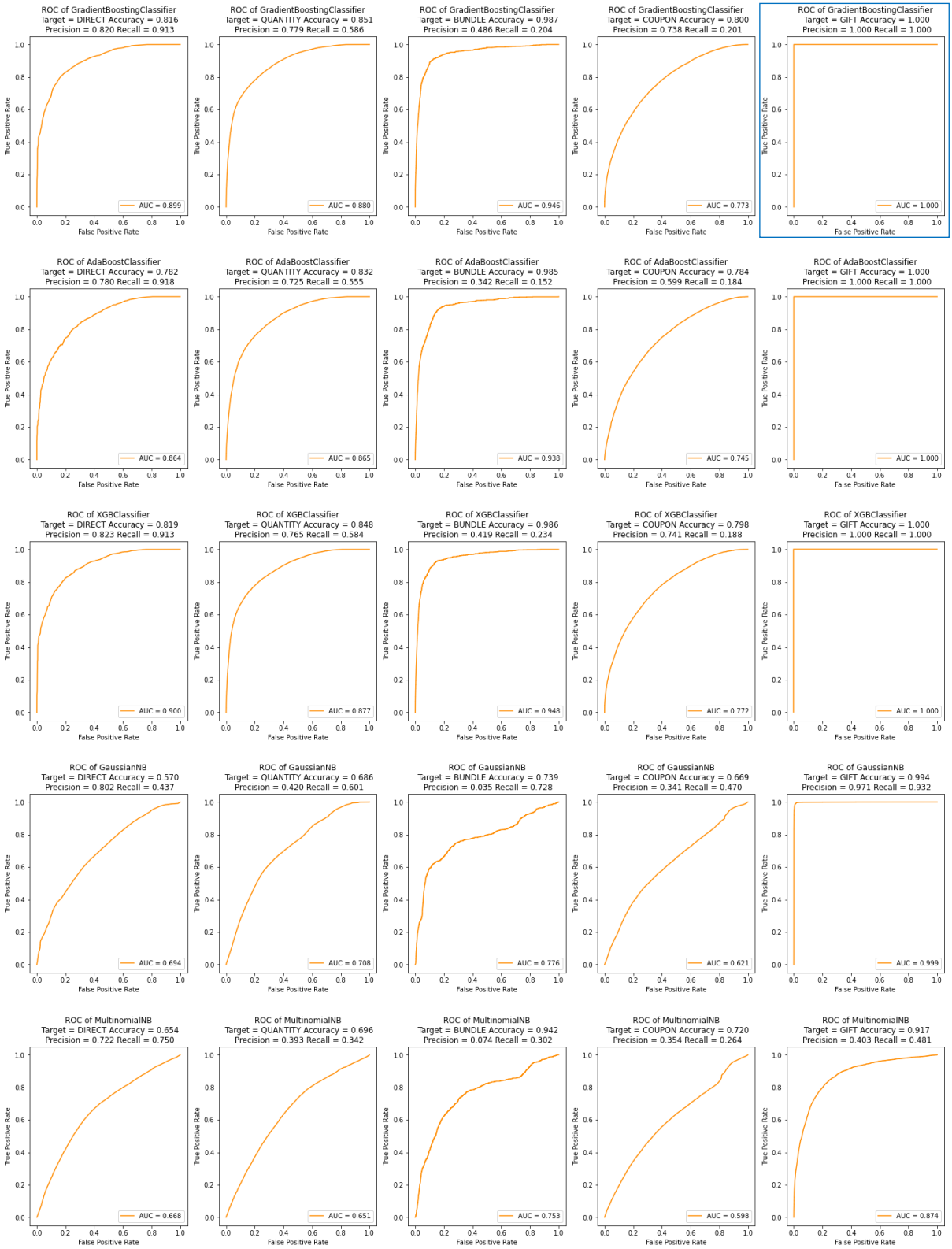


Figure 6. Evaluation Metrics Value of Different Classifiers with Default Parameters

Classifier / Target	DIRECT	QUANTITY	BUNDLE	COUPON	GIFT
Logistic Regression	0.75189	0.76048	0.82460	0.66798	0.98409
KNN	0.68730	0.71274	0.67773	0.65249	0.86789
LDA	0.73172	0.75532	0.83153	0.66987	0.98193
QDA	0.69718	0.70965	0.82712	0.63544	0.99997
SVM	0.70345	0.73346	0.87808	0.66966	0.96975
Neural Network	0.89351	0.88990	0.94491	0.76237	0.99994
Decision Tree	0.86885	0.80567	0.69774	0.65394	0.99996
Bagging Tree	**0.95965**	**0.93536**	0.90076	0.78835	**1.00000**
Random Forest	0.95422	0.93475	**0.96213**	**0.81306**	**1.00000**
Extra Tree	0.76705	0.74036	0.65437	0.62749	0.96587
GBDT	0.89896	0.88001	0.94599	0.77314	**0.99999**
Adaboost Tree	0.86391	0.86542	0.93758	0.74490	**1.00000**
Xgboost Tree	0.89968	0.87697	0.94815	0.77232	**1.00000**
Gaussian Naive Bayes	0.69383	0.70780	0.77559	0.62101	0.99888
Multinomial Naive Bayes	0.66757	0.65082	0.75305	0.59795	0.87354

Table 4. Test Score of Different Classifiers with Default Parameters

It can be seen from the values of various evaluation metrics and test set scores that the 5 target variables correspond to the 3 kind of best models with default hyperparameters, respectively. Among them, DIRECT and QUANTITY are most suitable for sorting **Bagging Tree**. **Random Forest** perform best when BUNDLES and COUPONS are target variables.

GIFT is special. Except for a few models whose test scores are less than 0.9, most models achieve a very high level of classification of this target variable, and even have perfect ROC curve and full score of test set, which is very rare. Because in normal machine learning modeling, test set scores can't be this high, and the ROC curve can't be this perfect. In this case, it is possible that the classification of GIFT variables is so easy and simple that the model fitting and test results are so perfect. To be prudent, however, we chose **GBDT**, whose test set score is second only to full score, as the most appropriate classification model for GIFT variables.

(3) For each target variable, the model with the best prediction results on the test set is selected and optimized. Results of optimization are shown in **Table 5** and **Figure 7**. After optimization, the best models for each target variable are combined into a hybrid model. In the next step, the hybrid model is validated and evaluated.

Target	**Optimized Model**	**Hyper Parameter (Part)**	**Test Score**
DIRECT	**Bagging Tree**	max_features: 1 max_samples: 0.5 n_estimators: 200	0.97066
QUANTITY		max_features: 1 max_samples: 0.5 n_estimators: 200	0.95104
BUNDLE	**Random Forest**	max_features: 'log2' n_estimators: 200	0.96722
COUPON		max_features: None n_estimators: 200	0.82430
GIFT	**GBDT**	max_features: None n_estimators: 100	0.99999

Table 5. Result of Optimization

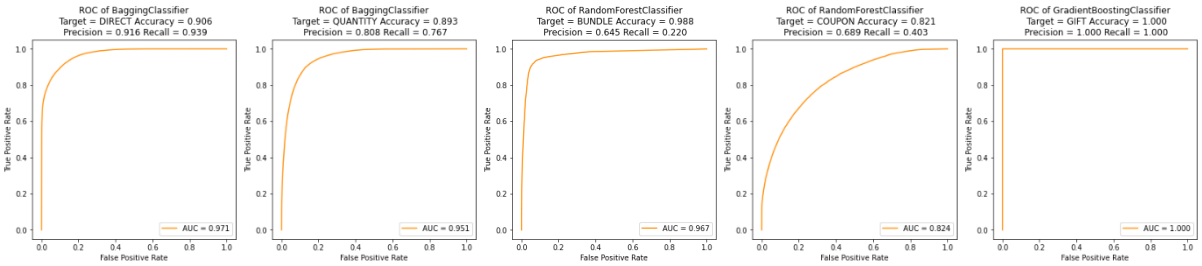


Figure 7. Evaluation Metrics Value of Different Classifiers with Optimized Parameters

4.5 Model Validation

In order to verify the validation of the hybrid model, we randomly select a certain number of samples from the dataset before partitioning with different random seeds and input them into the hybrid model for prediction. For each discount type, calculate the relevant classification evaluation metrics and draw the ROC curve. The validation results are shown in **Figure 8** and **Table 6-10**. **Figure 9** shows the line chart of evaluation metrics of validation test for different targets.


ROC of BaggingClassifier
Target = DIRECT Accuracy = 0.969
Precision = 0.971 Recall = 0.980
AUC = 0.995
ROC of BaggingClassifier
Target = QUANTITY Accuracy = 0.960
Precision = 0.934 Recall = 0.911
AUC = 0.990
ROC of RandomForestClassifier
Target = BUNDLE Accuracy = 0.998
Precision = 0.968 Recall = 0.849
AUC = 0.998
ROC of RandomForestClassifier
Target = COUPON Accuracy = 0.962
Precision = 0.959 Recall = 0.876
AUC = 0.984
ROC of GradientBoostingClassifier
Target = GIFT Accuracy = 1.000
Precision = 1.000 Recall = 1.000
AUC = 1.000
ROC of BaggingClassifier
Target = DIRECT Accuracy = 0.969
Precision = 0.971 Recall = 0.981
AUC = 0.995
ROC of BaggingClassifier
Target = QUANTITY Accuracy = 0.960
Precision = 0.933 Recall = 0.908
AUC = 0.990
ROC of RandomForestClassifier
Target = BUNDLE Accuracy = 0.997
Precision = 0.962 Recall = 0.827
AUC = 0.996
ROC of RandomForestClassifier
Target = COUPON Accuracy = 0.962
Precision = 0.959 Recall = 0.875
AUC = 0.984
ROC of GradientBoostingClassifier
Target = GIFT Accuracy = 1.000
Precision = 1.000 Recall = 1.000
AUC = 1.000
ROC of BaggingClassifier
Target = DIRECT Accuracy = 0.968
Precision = 0.970 Recall = 0.981
AUC = 0.995
ROC of BaggingClassifier
Target = QUANTITY Accuracy = 0.960
Precision = 0.930 Recall = 0.912
AUC = 0.990
ROC of RandomForestClassifier
Target = BUNDLE Accuracy = 0.998
Precision = 0.971 Recall = 0.853
AUC = 0.996
ROC of RandomForestClassifier
Target = COUPON Accuracy = 0.963
Precision = 0.958 Recall = 0.879
AUC = 0.985
ROC of GradientBoostingClassifier
Target = GIFT Accuracy = 1.000
Precision = 1.000 Recall = 1.000
AUC = 1.000
ROC of BaggingClassifier
Target = DIRECT Accuracy = 0.968
Precision = 0.970 Recall = 0.980
AUC = 0.995
ROC of BaggingClassifier
Target = QUANTITY Accuracy = 0.960
Precision = 0.934 Recall = 0.910
AUC = 0.990
ROC of RandomForestClassifier
Target = BUNDLE Accuracy = 0.998
Precision = 0.965 Recall = 0.843
AUC = 0.996
ROC of RandomForestClassifier
Target = COUPON Accuracy = 0.962
Precision = 0.959 Recall = 0.875
AUC = 0.984
ROC of GradientBoostingClassifier
Target = GIFT Accuracy = 1.000
Precision = 1.000 Recall = 1.000
AUC = 1.000
ROC of BaggingClassifier
Target = DIRECT Accuracy = 0.969
Precision = 0.971 Recall = 0.982
AUC = 0.995
ROC of BaggingClassifier
Target = QUANTITY Accuracy = 0.961
Precision = 0.934 Recall = 0.911
AUC = 0.991
ROC of RandomForestClassifier
Target = BUNDLE Accuracy = 0.998
Precision = 0.971 Recall = 0.841
AUC = 0.997
ROC of RandomForestClassifier
Target = COUPON Accuracy = 0.963
Precision = 0.959 Recall = 0.877
AUC = 0.984
ROC of GradientBoostingClassifier
Target = GIFT Accuracy = 1.000
Precision = 1.000 Recall = 1.000
AUC = 1.000
True Positive Rate
False Positive Rate

Figure 8. Evaluation Metrics Value of Different Classifiers with Optimized Parameters Under Random Sample 1-10

Target	Random State	Accuracy	Precision	Recall	F1 Score	AUC
DIRECT	1	0.969	0.971	0.980	0.975	0.995
	2	0.969	0.971	0.981	0.976	0.995
	3	0.968	0.970	0.981	0.975	0.995
	4	0.968	0.970	0.980	0.975	0.995
	5	0.969	0.971	0.982	0.976	0.995
	6	0.969	0.971	0.981	0.976	0.995
	7	0.969	0.971	0.981	0.976	0.995
	8	0.969	0.971	0.981	0.976	0.995
	9	0.968	0.971	0.980	0.975	0.995
	10	0.968	0.969	0.980	0.974	0.994

Table 6. Result of Validation Test for Target Variable *DIRECT*

Target	Random State	Accuracy	Precision	Recall	F1 Score	AUC
QUANTITY	1	0.960	0.934	0.911	0.922	0.990
	2	0.960	0.933	0.908	0.920	0.990
	3	0.960	0.930	0.912	0.921	0.990
	4	0.960	0.934	0.910	0.922	0.990
	5	0.961	0.934	0.911	0.922	0.991
	6	0.961	0.933	0.910	0.921	0.991
	7	0.960	0.934	0.909	0.921	0.990
	8	0.959	0.932	0.908	0.920	0.991
	9	0.961	0.936	0.910	0.923	0.990
	10	0.960	0.935	0.907	0.921	0.990

Table 7. Result of Validation Test for Target Variable *QUANTITY*

Target	Random State	Accuracy	Precision	Recall	F1 Score	AUC
BUNDLE	1	0.998	0.968	0.849	0.905	0.998
	2	0.997	0.962	0.827	0.889	0.996
	3	0.998	0.971	0.853	0.908	0.996
	4	0.998	0.965	0.843	0.900	0.996
	5	0.998	0.971	0.841	0.901	0.997
	6	0.998	0.970	0.848	0.905	0.996
	7	0.998	0.971	0.857	0.910	0.995
	8	0.998	0.978	0.841	0.904	0.997
	9	0.998	0.975	0.852	0.909	0.996
	10	0.997	0.980	0.842	0.906	0.997

Table 8. Result of Validation Test for Target Variable *BUNDLE*

Target	Random State	Accuracy	Precision	Recall	F1 Score	AUC
COUPON	1	0.962	0.959	0.876	0.916	0.984
	2	0.962	0.959	0.875	0.915	0.984
	3	0.963	0.958	0.879	0.917	0.985
	4	0.962	0.959	0.875	0.915	0.984
	5	0.963	0.959	0.877	0.916	0.984
	6	0.963	0.960	0.879	0.918	0.985
	7	0.962	0.957	0.876	0.915	0.984
	8	0.963	0.959	0.877	0.916	0.985
	9	0.964	0.959	0.880	0.918	0.985
	10	0.963	0.957	0.880	0.917	0.985

Table 9. Result of Validation Test for Target Variable *COUPON*

Target	Random State	Accuracy	Precision	Recall	F1 Score	AUC
GIFT	1	1.000	1.000	1.000	1.000	1.000
	2	1.000	1.000	1.000	1.000	1.000
	3	1.000	1.000	1.000	1.000	1.000
	4	1.000	1.000	1.000	1.000	1.000
	5	1.000	1.000	1.000	1.000	1.000
	6	1.000	1.000	1.000	1.000	1.000
	7	1.000	1.000	1.000	1.000	1.000
	8	1.000	1.000	1.000	1.000	1.000
	9	1.000	1.000	1.000	1.000	1.000
	10	1.000	1.000	1.000	1.000	1.000

Table 10. Result of Validation Test for Target Variable *GIFT*

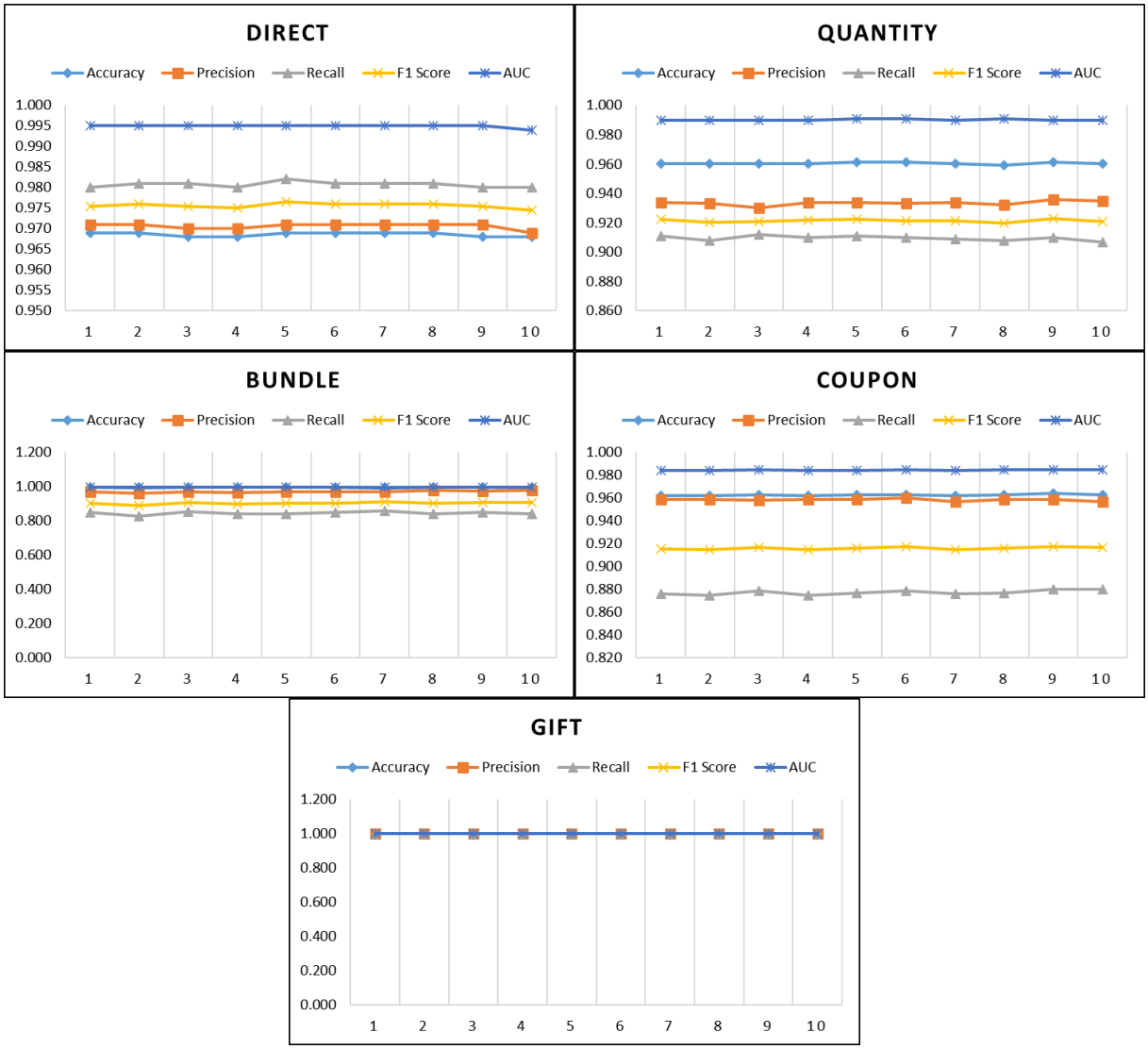


Figure 9. Line Chart for Evaluation Metrics of Validation Test

It can be seen that no matter under which random state, the Accuracy, Precision, Recall, F1 Score and AUC, all reach a high value. The final classification effect of the hybrid model reached a very high level.

The final result of modeling analysis fully illustrates the advantages of hybrid model. Compared with the single model and multi-classification problem, the mixed model has corresponding models for different target variables to classify, which greatly improves the final classification effect and finally passes the validity verification.

REFERENCE

[1] Allon, G., & Zeevi, A. (2011). A note on the relationship among capacity, pricing, and inventory in a make-to-stock system. *Production and Operations Management*.

[2] Aouad, A., Feldman, J., Segev, D., & Zhang, D. (2019). Click-Based MNL: Algorithmic Frameworks for Modeling Click Data in Assortment Optimization. *SSRN Electronic Journal*.

[3] Bray, R. L., Serpa, J. C., & Colak, A. (2019). Supply chain proximity and product quality. *Management Science*.

[4] Cohen, M. C., Lobel, I., & Paes Leme, R. (2020). Feature-Based Dynamic Pricing. *Management Science*.

[5] Cui, R., Allon, G., Bassamboo, A., & Van Mieghem, J. A. (2015). Information sharing in supply chains: An empirical and theoretical valuation. *Management Science*.

[6] Daskin, M. S., Coullard, C. R., & Shen, Z. J. M. (2002). An inventory-location model: Formulation, solution algorithm and computational results. *Annals of Operations Research*.

[7] Feldman, J., Zhang, D., Liu, X., & Zhang, N. (2019). Customer choice models versus machine learning: Finding optimal product displays on {Alibaba}. *Available at SSRN 3232059*.

[8] Lee, H. L., So, K. C., & Tang, C. S. (2000). Value of information sharing in a two-level supply chain. *Management Science*.

[9] Liao, S. L., Shen, Y. C., & Chu, C. H. (2009). The effects of sales promotion strategy, product appeal and consumer traits on reminder impulse buying behaviour. *International Journal of Consumer Studies*.

[10] Liu, Y., Li, H., Peng, G., Lv, B., & Zhang, C. (2015). Online purchaser segmentation and promotion strategy selection: evidence from Chinese E-commerce market. *Annals of Operations Research*.

[11] Shen, Z. J. M., Coullard, C., & Daskin, M. S. (2003). A joint location-inventory model. *Transportation Science*.

[12] Zhang, D. J., Dai, H., Dong, L., Qi, F., Zhang, N., Liu, X., Liu, Z., & Yang, J. (2020). The Long-term and Spillover Effects of Price Promotions on Retailing Platforms: Evidence from a Large Randomized Experiment on Alibaba. *Management Science*.

[13] Zhang, D. J., Dai, H., Dong, L., Wu, Q., Guo, L., & Liu, X. (2019). The value of pop-up stores on retailing platforms: Evidence from a field experiment with Alibaba. *Management Science*.

[14] Zheng, K., Zhang, Z., & Song, B. (2020). E-commerce logistics distribution mode in big-data context: A case analysis of JD.COM. *Industrial Marketing Management*.