UNSW
THE UNIVERSITY OF NEW SOUTH WALES

COMP9417 Machine Learning

Project:Elo Merchant Category Recommendation

Group Member and Zid:

Yifei Yue z5392319

Jiahui Xie z5341211

Zhao Zheng z5297877

Xiandong Cheng z5342690

Yuanwei Zhao z5355526

Kaggle link: Elo Merchant Category Recommendation | Kaggle

July 2022

I. Introduction

A. Background

In Brazil, online payment is mainly operated by local banks, and the main channel for online payment is the credit card. Elo Servicos SA is a local brand in Brazil. It was established in 2011 by a joint venture of three major Brazilian banks. There are currently 115 million cards in circulation. It is the largest local online payment brand in Brazil. The company takes credit cards as its core financial product and is also an ”o2o” platform, where the app can recommend services such as local restaurants, hotels, flights, hotels, and other services to users. This competition is jointly organized by Elo and Kaggle. The purpose is to train the model through the historical transaction records of customers and the information data of customers and merchants, and finally predict the loyalty scores of all credit cards in the test set.

B. Objectives and methods

The purpose of this project is to reveal the signals in customer loyalty, identify and provide the most relevant opportunities for individuals through the development of algorithms. The investment of algorithm will improve customers' lives, help ELO reduce unnecessary activities and create the right experience for customers.

For this project, the commonly used algorithms are machine learning algorithms such as Random Forest, LightGBM, XGBosst, etc. or some deep learning algorithms. Then use feature engineering and model fusion to improve the accuracy of the algorithm.

In this project, we use Random Forest, LightGBM, XGBosst methods as the baselines, and then use the method of super parameter optimization to find the optimal parameters. Finally, we use the method of XGB+RF to improve the accuracy. In addition, we also tried the voiting method and the neural network method to try to get better results.

C. Data

In order to ensure privacy and information security, the competition organizer deliberately hides some information, desensitizes and creates new fields, so the data set is not real customer data.

The size of data in this competition is relatively large: there are 7 files in total, which can be divided into 3 categories. The first category contains the Data Dictionary.xlsx which provides only provides the basic information of all data tables including all features and the relevant explanations. The second category contains the necessary training set and test set for machine learning. The last category contains the supplementary datasets. The historical_transactions.csv and new_merchant_transactions.csv contains the history and latest transaction records of credit cards.

D. Task Analysis

After studying the official instructions on the forum and related discussions, we found that the purpose of the competition is to provide customized and personalized suggestions for different users. To achieve this, it is necessary to first estimate the loyalty of each card holder.

E. Evaluation indicators

This problem uses root mean square error (RMSE) to evaluate the performance of the prediction of customer loyalty. The specific calculation method is as follows:

$$\mathrm{RMSE} = \frac{1}{\mathrm{n}}\sqrt{\sum_{\mathrm{i}=1}^{\mathrm{n}} (\mathrm{y}_i - \mathrm{y}_{\mathrm{i}}^{\wedge})^2} \tag{1}$$

y_i is the predicted loyalty score for each credit card, and y_i is the actual loyalty score for the same credit card Id.

II. Exploratory Data Analysis (EDA)

Data exploration not only helps to understand the problem more clearly but also provides a better understanding of the dataset. It will be of great help to the subsequent feature engineering and modeling.

A. Field Category Meaning

First, researching on features is carried out which is mainly about the analysis of each feature and its possible data types types. The explanations of all features can be found in the appendix.

B. Data Quality Analysis

The data correctness is mainly used to confirm that the data itself satisfy the basic logic.

First, as a unique identifier of the analyzed object, we need to verify that every credit card Id is unique and there is no duplication of credit card Id in the training set and the test set. Finally, it is found that the credit card id in the training set and the test set are unique, and there is no duplicate Id in the two data sets.

Second, We need to check if there is any missing values in the data set. By checking each feature, it can be found that the data set has nearly no missing value. And the only missing value in the test set can be filled in many ways. In general, a missing value will not have much impact on the overall modelling.

TABLE I

CHECKING NULL VALUE IN TRAINING SET

Feature	Number of null value
first_active_month	0
card_id	0
feature_1	0
feature_2	0
feature_3	0
target	0

TABLE II

CHECKING NULL VALUE IN TEST SET

Feature	Number of null value
first_active_month	1
card_id	0
feature_1	0
feature_2	0
feature_3	0

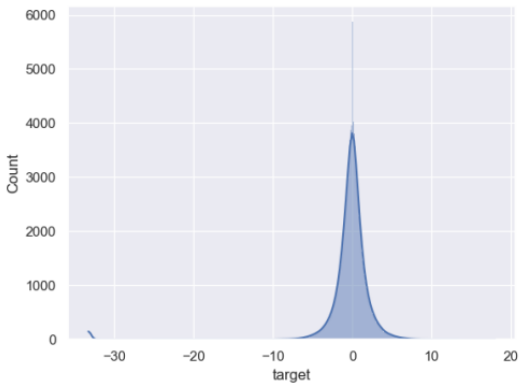
The outlier test is the next stage. We start by scanning the label columns for outliers because we haven't already preprocessed the dataset's features. To start, we may use describe() to see the column's fundamental statistics:

TABLE III

STATISTIC INFORMATION

count	201917.000000
mean	-0.393636
std	3.850500
min	-33.219281
25%	-0.883110
50%	-0.023437
75%	0.765453
max	17.965068

Since this column is a continuous variable, we can use a probability density histogram to visualize the distribution:



The only thing to take into account is that certain outliers are below -30, which necessitates extra care in the analysis that follows. Through the analysis of the data, we found there is around 1.0930% of people have tags with

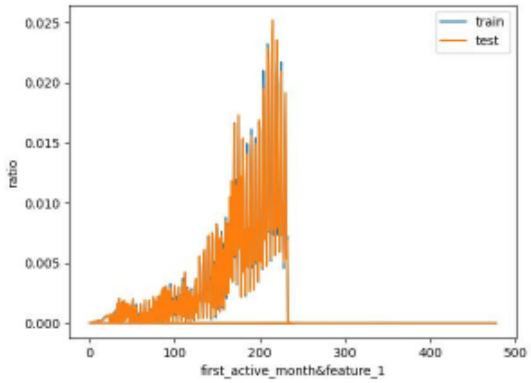
values under 30.

It should be highlighted that outlier identification is done here with labels in mind. In this instance, labels are manually derived using some formula rather than being the outcome of natural numerical measurement or statistics (such as consumption quantity, height and weight, etc). Such anomalies are probably indicative of a certain kind of user. Outliers should be treated as a special category instead of being dealt with as such, and in the subsequent modeling analysis, their feature extraction and modeling analysis should be done separately.

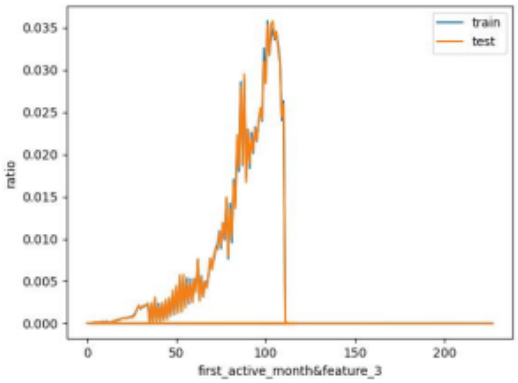
C. Consistency analysis

A straightforward comparison of the distribution of the feature data from the training set and test set is what is meant by the term ”regularity consistency.” The majority of the data sources, in our opinion, follow a particular statistical distribution. Although training and test data may come from different sections of the exact total due to data selection and other factors, if the difference is too substantial, it will significantly influence the ensuing model training. So, determining if there is regular consistency between two data sets is the goal of regular consistency.

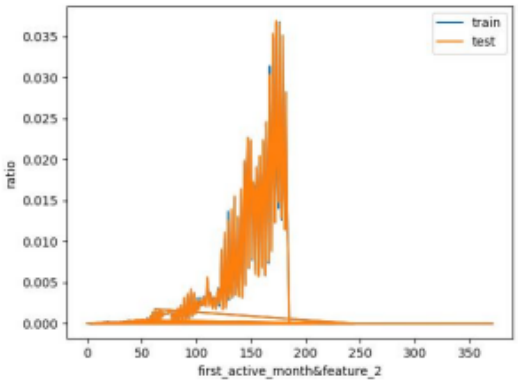
The four variables in the data set may all be discrete. Thus by computing the relative proportion, we can compare the probability distribution of the data. In addition, we can further study the distribution of joint variables in more detail. Joint probability distribution refers to the distribution of the relative proportions of the new variable after pairing discrete variables.



(a)



(b)



(c)

The percentage distribution of each joint variable is discovered to be essentially the same. It can be essentially established that the training set and test set are drawn from the same sample population because of the excellent overall quality of the data set.

D. Data Preprocessing

The previous data exploration revealed that there were three different forms of data in the data: discrete, continuous, and time. As a result, we must preprocess the data, which involves changing the format, substituting temporal data with continuous values, processing text data, and other tasks.

Preprocessing of train and testing set:

First, based on the previous data analysis, it is discovered that there is a missing value in the data set. However, in most cases, having just one missing value has minimal effect, and because this field is a character type, it has to be encoded. Lexicographic sorting can be used to encode it as it naturally has a sequential connection. The time

and date from the original 'first active month' have been replaced with integers.

However, in most cases, having just one missing value has minimal effect, and because this field is a character type, it has to be encoded. Lexicographic sorting can be used to encode it as it naturally has a sequential connection. The time and date data from the original 'first active month' has been replaced with integers.

First active month	First active month(integer data)
2017-06	67
2017-01	62
2016-08	57
2017-09	70
2017-11	72
...	...
2017-09	70
2015-10	47
2017-08	69
2016-07	56
2017-07	68

Preprocessing for merchants data set:

Firstly, discrete field and continuous field are divided.

category_cols:	['merchant_id', 'merchant_group_id', 'merchant_category_id', 'subsector_id', 'category_1', 'most_recent_sales_range'...]
numeric_cols:	['numerical_1', 'numerical_2', 'avg_sales_lag3', 'avg_purchases_lag3', 'active_months_lag3', 'avg_sales_lag6'...]

Treatment of discrete type:

Lexicographical sort encoding is carried out for the discrete fields of character type:

category_1	...	category_4
0	...	0
0	...	0
0	...	0
1	...	1
1	...	1
...	...	...

The missing discrete fields are filled with -1, and then optimized if necessary. Removes duplicate columns from the trade record table and duplicate records for merchant_id.

merchant_id	0
merchant_group_id	0
merchant_category_id	0
subsector_id	0
numerical_1	0
numerical_2	0
category_1	0
most_recent_sales_range	0
most_recent_purchases_range	0
...	...

Treatment of continuous type:

For infinite values, replace INF with the maximum value of the corresponding column.

Missing value processing: use the mean value to fill in the processing.

statistic	avg_purchases_lag12
mean	inf
std	NaN
max	inf

statistic	avg_purchases_lag12
mean	2.633572
std	205.206198
max	61851.333

Preprocessing for New Merchant Transactions.csv:

Separate discrete fields, discrete fields, and time fields.

numeric_cols:	['installments', 'month_lag', 'purchase_amount']
category_cols:	['authorized_flag', 'card_id', 'city_id', 'category_1', 'category_3', 'merchant_category_id'...]
time_cols:	['purchase_date']

Missing values are padded with -1.

authorized_flag	0
card_id	0
city_id	0
category_1	0
category_3	0
merchant_category_id	0
merchant_id	0
category_2	0
state_id	0
subsector_id	0

Time field is divided into categorical variables, extract the month, week, and time period information, and then the new generated month field dictionary encoding sort.

purchase_month	purchase_hour_section	purchase_day
12	2	1
12	3	0
13	2	0
12	1	0
12	3	0
...	...	...

After processing merchant information and submitting record information, the form is merged and the field is reclassified.

III. Feature Engineering

Feature engineering is the process of extracting features from the original data, and the performance of the model built using the features can be optimal in unknown data. The key to the feature engineering of this question is the portrait of the user's transaction behaviour, that is, the quantification of shopping behaviour in various dimensions. Mining the relationship between various transaction behaviours of users and the target column and creating features that may positively impact the model results as much as possible. In order to ensure the stable and efficient operation of the model, feature selection is also required.

A. Derived features

1) Combined features: The card_id is used as the first-level key value, the feature field is used as the second-level key value for feature derivation, and the purchased quantity and consumption amount of the user under each value of each category field is obtained. The specific operation is to combine discrete fields with different values to generate new features with continuous fields and group and sum them according to different card_ids under the

new features. Around each card_id, provide more dimensional information to supplement and display, and then bring it into the model for modelling.

Each categorical variable has many classification levels and is spliced with different continuous fields. The features will become very large. Moreover, there will be a lot of null values and values with a value of 0. When this happens, it needs to be filled in later.

2) Statistical features: First, group card_id, and then calculate various statistics in different id groups, such as extreme value, mean, etc., to generate new features. This extracts more information from the transaction and puts it into train and test.

3) Text Features: There are many id-features in the dataset, and the frequency of their occurrence strongly correlates with the actual transaction behaviour of users. They have been coded only briefly before, and they are only considered logos. If the id column is regarded as text, TF-IDF vector features based on CountVector and NLP fields can also be extracted to understand the popularity of stores and whether users' preferences are shared.

Word frequency statistics can show which words are more important and then analyze which user is more inclined to choose for all merchants. If some merchants appear more frequently, the user prefers this merchant more.

TF-IDF conducts deeper mining based on word frequency statistics. If a merchant frequently appears in many users' lists, then the merchant must be favoured by users. If a user particularly loves a specific merchant, but this merchant does not frequently appear in other people's consumption records, it means that the user's taste is different from others. After the TF-IDF process, more dimensions of calculation and evaluation can be added to the user's preference.

B. Feature selection

When creating features, the idea is that the more the better, the data set becomes very sparse. Although the decision tree model has a default feature selection mechanism, the method that can minimize label impurity will be preferentially selected for growth. If redundant features are substituted for modeling, it will have little effect on the tree model and the integrated model of the tree model. . However, too many features were created before, resulting in too many redundant features, which will affect the modeling efficiency. It is unknown which features are valid for a particular learning algorithm. Therefore, it is necessary to select relevant features that are beneficial to the learning algorithm from all the features. Moreover , the improvement of the model is limited by substituting too many irrelevant features . If only some of the features are selected to build a model, the difficulty of learning tasks can be reduced, the efficiency of the model can be improved, the generalization ability of the model can be enhanced, and overfitting can be reduced. Commonly used methods are Filter and Wrapper.

1) The filter method: In this task, we apply filter method to do feature selection. The correlation coefficient of each feature is calculated. The correlation between the feature and the label is judged, and we only keep the features with high correlation.

2) The wrapper method: The wrapper method is a feature selection method by checking the performance of a pre-trained model. This method is more computationally intensive but at the same time more accurate. The model will calculate the importance of each feature and generate a ranking after training. Unlike filtering feature selection, the wrapper method directly uses the performance of the model as the evaluation criterion for feature selection. The purpose is to select a subset of features that are most beneficial for the performance of the model. Next, according to the importance of each feature, the more essential features are selected and substituted into the subsequent hyperparameter optimization and cross-validation process.

IV. Model training

A. Random Forest

The random forest algorithm randomly samples different subsets from the provided data to build multiple different decision trees and integrates the results of a single decision tree according to the rules of Bagging (regression means average, classification means minority obeys majority). The random forest has powerful learning ability, can be applied to high-dimensional data, and can judge the importance of features and the interaction between different features. The training speed is relatively fast and has a specific ability to resist overfitting. It is a fundamentally superior algorithm to a single decision tree. We combine the random forest algorithm with the filter and wrapper feature selection methods to train the models.

B. LightGBM

Gradient Boosting Decision Tree (GBDT) is a representative boosting method algorithm. It is also one of the industry's most widely used machine learning algorithms and has the most stable performance in practical scenarios. However, GBDT must traverse the entire training data multiple times in each iteration. If the entire training data is loaded into the memory, the size of the training data will be limited; if it is not loaded into the memory, repeatedly reading and writing the training data will consume a massive amount of time. Based on GBDT, lightGBM solves the problems encountered by GBDT in the face of massive data. At the same time, it also draws on some ideas of XGBoost and inherits some of XGBoost 's achievements and data processing processes. The parameter setting method is similar to the XGBoost training method. The training method is performed by first defining the parameters and then calling the function directly. At the same time, the parameters are also set directly in the form of a dictionary. We combine the lightGBM algorithm with the filter and wrapper feature selection methods to train the models.

C. XGBoost

XGBoost is a new-generation boosting algorithm based on a comprehensive upgrade of the gradient boosting tree GBDT. XGBoost is an algorithm system centred on boosting trees, which can implement various types of gradient boosting trees with unparalleled flexibility. XGBoost has made many critical improvements on GDBT to balance accuracy and complexity, significantly reduce model complexity, improve superior operating efficiency, and make the algorithm more suitable for big data algorithms. We combine the XGBoost algorithm with the filter and wrapper feature selection methods to train the models.

D. Neural Network

A mathematical model or computer model that imitates the form and operation of a biological neural network is known as an artificial neural network, also known as a neural network. A neural network is made up of a lot of artificial neurons joined together to process data. Artificial neural networks, which are adaptable systems, may frequently modify their internal structure in response to external input. Modern neural networks are nonlinear statistical data modeling tools that are frequently used to investigate patterns in data or represent complicated interactions between inputs and outputs.

Regression analysis (RSME) is used as the assessment criterion to train the best model to predict the ”target” value in the test data set using the ”feature” data set and the ”target” data set. The input item for a single neuron is the quantity of ”features,” the hidden data layer is set to 100, and the predicted value is the output item. Due of the size of the number of data sets and the existence of outliers in the ”target” variable with values less than -30, SGD is chosen as the loose function. As a result, GD may become stranded at saddle points if utilized as a loose function, leading to the discovery of a local optimum solution.

E. XHBoostRF regressor

XHBoostRF regressor is basically a new version of XGBoost regressor that train random forest instead of gradient boosting. We combine the XHBoostRF algorithm with the filter feature selection method to train the model.

V. Parameter Tuning

A. Two ways of parameter Tuning

1) Grid search with cross validation: Grid search is a parameter tuning method for exhaustive search. Before the search starts, we need to manually list the candidate values of each hyperparameter and eventually form a parameter space. The grid search algorithm will bring all parameter combinations in this parameter space into the model for training and finally select the combination with the strongest generalization ability as the final hyperparameter of the model. For grid search, if a certain point in the parameter space points to the actual minimum value of the loss function, the minimum value and corresponding parameters must be captured when enumerating grid search. At the same time, the goal of parameter optimization is to find the combination that maximizes the generalization ability of the model, so a 3 fold cross-validation is applied to show the generalization ability of the model.

2) TPE hyperparameter optimization: Grid search is a method in large parameter space to verify all points as much as possible and then return the optimal loss function value. This kind of method has inevitable defects in calculation amount and calculation time. Although sklearn is optimized to improve grid search efficiency, it still cannot achieve a win-win in efficiency and accuracy. The Hyperopt optimizer is one of the most common Bayesian optimizers at present. It supports various efficiency improvement tools and is the most commonly used optimizer to implement the TPE method. In practical use, compared to grid search, Gaussian mixture model-based TPE achieves better results with higher efficiency in most cases.

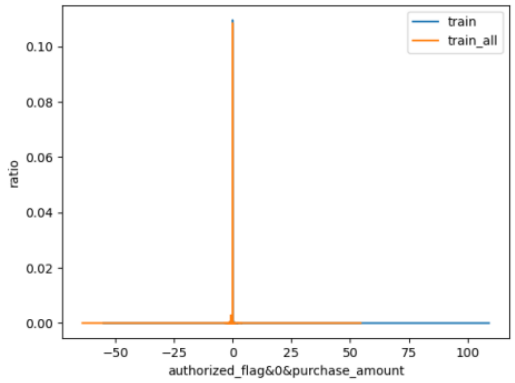
B. Parameter selection

The degree of mutual influence between the various parameters of random forest is very small. Therefore, through our testing experience, we selected the features which have a relatively large impact on the performance of the model to do grid search or TPE hyperparameter optimization to find the best set of parameters.

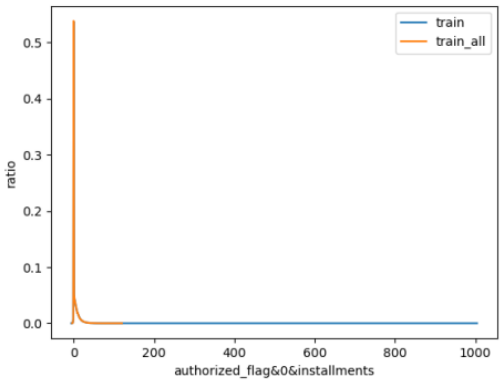
C. Tuning parameters on a small data set

Because there are many features produced through feature engineering. It will take a lot of time and memory to use the entire data set to find the optimum parameters using grid search or TPE hyperparameter optimization. Suppose a subset has the same statistical distribution as the entire data set. In that case, it may be argued that the subset can represent the whole data set, and the best parameter produced from the subset is also representative of the complete data set.

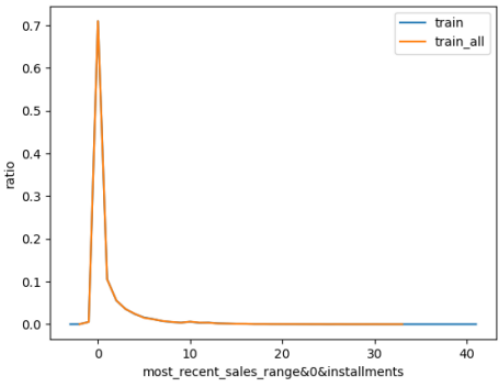
By visualizing the feature distribution of the subset and the original data set, we can find that the distribution is relatively similar, showing that all features are drawn from the same pool. The subset and whole data set exhibit strong consistency in the law. Therefore, we can use this subset to find the best parameters for each model.



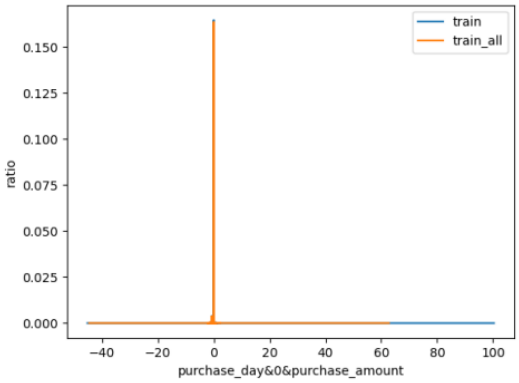
(d)



(e)

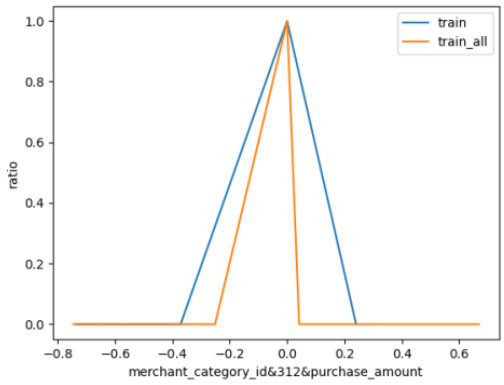


(f)

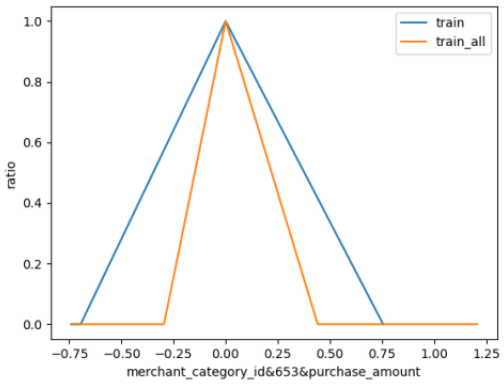


(g)

It should be noted that during the feature engineering process, certain ID class ”features” are also combined. However, the ”feature” of ID class has no significance in the rule consistency analysis because the analysis’s goal is to determine whether the chosen data are consistent with the data’s overall distribution. In the rule consistency analysis, we assume that the entire ”feature” selected is consistent with a particular statistical distribution. It is required to disregard the ID class ”feature” since it was artificially provided in accordance with certain requirements and does not follow the statistical distribution. Additionally, a few unusual ”features” imply that all features produced by feature engineering can serve as data features. As a result, we must select ”features” in the initial model training stage.



(h)



(i)

D. Analysis of parameter tuning results

A total of three kinds of baselines and XGBoostRF have been hyperparametric optimized. From the data in the table below, when we test on our own test set, the scores of the optimal parameter model after grid search have been improved about 0.1 on average (The lower the score, the better performance). Additionally, in the following table, top300 means that the most important 300 features are selected for training through filter or wrapper methods, default means that the default parameters are used, and best means that the optimal parameters after parameter tunning are used.

TABLE IV
PARAMETER TUNING COMPARISON

Model	Our testing dataset	Model	Our testing dataset
Randomforest+top300filter(default)	3.893287582	Randomforest+top300filter(best)	3.760519767
Randomforest+top300wrapper(default)	3.890086979	Randomforest+top300wrapper(best)	3.756524475
LightGBM+top300filter(default)	3.753217633	LightGBM+top300filter(best)	3.751682896
LightGBM+top300wrapper(default)	3.752603983	LightGBM+top300wrapper(best)	3.749994228
XGBoost+top300filter(default)	3.8028438	XGBoost+top300filter(best)	3.76762590
XGBoost+top300wrapper(default)	3.788632496	XGBoost+top300wrapper(best)	3.758084139
XGBoostRF+top300filter(default)	3.779497863	XGBoostRF+top300filter(best)	3.773320191

VI. MODEL FUSION

Model fusion is a method of combining multiple heterogeneous individual learners to obtain a model with stronger generalization ability than a single learner, which belongs to the category of integrated learning. Train multiple models, and then integrate them according to certain methods. It should be easy to understand, simple to implement,

and with good results. It is widely used in industry. The voting method in ensemble learning refers to selecting multiple models for integration, and processing the prediction results of multiple models, taking the average (regression problem), or taking the result with the largest probability (the most times) (classification problem), to reduce the variance of the model and improve the robustness of the model.

In this task, we mainly adopt the voting method to do model fusion based on two standard. The first one takes the average of the results of three basic models, that is, each model accounts for one third of the weight. The second one gives half the weight to the best performing model, the other two methods each account for 25% of the weight. And the models used to do voting model fusion are the top 3 best models after tuning parameters in the table above, which are randomforst+top300wrapper(best), lightGBM+top300wrapper(best) and XGBoost+top300wrapper(best).

VII. CONCLUSION

On the basis of the three baselines, we optimized the xgboost method and adopted other deep learning methods. Among them, the Voiting method performs best, The performance of the RSNE_loss curve method is poor.The following are the results of these additional methods.

TABLE V

PERFORMENCE ON DIFFERENT MODELS AND FILTER

Model	Our testing dataset	Model	Our testing dataset
XGBoostRF+top300filter(default)	3.779497863	XGBoostRF+top300filter(best)	3.773320191
voting_arr	3.748376172	voting_weight	3.747964727
RSME_loss curve(original_train_data)	3.8507	RSME_loss value(train_data_after_feature_engineering)	3.8453

After completing the training of three basic models (Random forest, LightGBM, and XGBoost) and three optimization models (Voting, XGBoostRF, and RSME_loss curve), we selected the best ones and tested the leaderboard private score and leaderboard public score respectively.The private leaderboard is calculated with approximately 70% of the test data and the public leaderboard is calculated with approximately 30% of the test data. The final results will be based on the other 70%, so the final standings may be different.

TABLE VI

LEADERBOARD PRIVATE SCORE AND LEADERBOARD PUBLIC SCORE

	leaderboard private score	leaderboard public score
Randomforest+top300wrapper(best)	3.73359	3.8427
LightGBM+top300wrapper(best)	3.72905	3.83776
XGBoost+top300wrapper(best)	3.74163	3.85638
voting_arr	3.7281	3.8385
voting_weight	3.72753	3.83745

From the result, for the Single model LightGBM has achieved the best result, for the fusion model, voting_ weight has got the best score.

VIII. FUTURE WORK

Comparing the top-ranked solutions to the analysis, there are still some areas that can be improved. For example, the overall quality of the dataset can be further improved by further analyzing the connections within the data and constructing more useful features, and more effective models and fusion methods can be used to improve the overall modeling effect.

The existing dataset is a dataset that records user transaction behavior, which is not large enough. But the supplementary dataset contains transaction data of each credit card in the past 5 years, which can provide a lot of information. Therefore, we can consider extracting valid information from the transaction information and stitching it into the training set for modeling.

The ability to exploit the features of user behavior can greatly affect the user loyalty score. Previously, the features were generated in batch by engineering methods, and there was no particular determination of which features were user behavior features and whether they could be used to derive more features, which should be used as an entry point for feature optimization.

Text features, time series features are the breakthrough point for feature derivation. Analyzing what behavioral characteristics each person will have in different time periods will achieve better results through more fine-grained feature mining. Usually the closer the user behavior is to the current point in time the more valuable it is, so this question should focus on the recent transaction information of users within a certain time span, derive more features around the time series, count the behavioral features of users during this time and substitute them into the modeling.

Since user behavior has a greater impact on the model, second-order cross-derived features can be created for user behavior data. There are outliers in the labels of the training set, which are clearly not the result of natural statistics and are most likely the sign of a particular class of users. Much of the error comes from the fact that the outliers are not accurately predicted, so it is necessary to analyze and handle the outliers. A two-stage modeling can be executed, where the first stage first screens which users are anomalous, and if it is judged to be anomalous, gives it the outliers directly. If not, substitute into the next model for modeling, focusing on prediction around non-anomalous credit cards. Two predictions are made on a dataset to offset the impact of outliers on the modeling.

In terms of model selection, you can choose the efficient CatBoost model, which has made many improvements in the processing of category features in the framework of GBDT algorithm, and can automatically use a hybrid strategy of unique hot coding and average coding to process category features, applying a new gradient boosting mechanism, with accuracy comparable to LightGBM and faster training speed than XGBoost. It can obtain good results without adjusting the parameters and reduce the chance of overfitting, which is more versatile.

In the model fusion part, the Stacking fusion is better than the weighted fusion because of the regression problem

and outliers in this competition. Therefore, the results of Stacking fusion should be better.

REFERENCES

[1] James Bergstra and Yoshua Bengio. Random search for hyper-parameter optimization. *J. Mach. Learn. Res.*, 13(null):281–305, feb 2012.

[2] Weijie Bian, Kailun Wu, Lejian Ren, Qi Pi, Yujing Zhang, Can Xiao, Xiang-Rong Sheng, Yong-Nan Zhu, Zhangming Chan, Na Mou, Xinchen Luo, Shiming Xiang, Guorui Zhou, Xiaoqiang Zhu, and Hongbo Deng. Can: Feature co-action for click-through rate prediction, 2020.

[3] Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 785–794, New York, NY, USA, 2016. Association for Computing Machinery.

[4] Antonio Criminisi, Ender Konukoglu, and Jamie Shotton. Decision forests for classification, regression, density estimation, manifold learning and semi-supervised learning. Technical Report MSR-TR-2011-114, October 2011.

[5] Alex Davies and Zoubin Ghahramani. The random forest kernel and other kernels for big data from random partitions, 2014.

[6] Zhifang Fan, Dan Ou, Yulong Gu, Bairan Fu, Xiang Li, Wentian Bao, Xin-Yu Dai, Xiaoyi Zeng, Tao Zhuang, and Qingwen Liu. Modeling users' contextualized page-wise feedback for click-through rate prediction in e-commerce search. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*. ACM, feb 2022.

[7] Jerome H. Friedman. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29:1189–1232, 2001.

[8] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[9] Erxue Min, Yu Rong, Tingyang Xu, Yatao Bian, Da Luo, Kangyi Lin, Junzhou Huang, Sophia Ananiadou, and Peilin Zhao. Neighbour interaction based click-through rate prediction via graph-masked transformer. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '22, page 353–362, New York, NY, USA, 2022. Association for Computing Machinery.

[10] Fangye Wang, Yingxu Wang, Dongsheng Li, Hansu Gu, Tun Lu, Peng Zhang, and Ning Gu. Enhancing ctr prediction with context-aware feature representation learning, 2022.

[11] Xuesi Wang, Guangda Huzhang, Qianying Lin, and Qing Da. Learning-to-ensemble by contextual rank aggregation in e-commerce, 2021.

IX. APPENDIX

A. *Field Category Meaning*

train.csv and test.csv	
	Mean
card_id	Credit card id.
first_active_month	Months when you first use your credit card for purchases.
feature_1/2/3	Discrete characteristics of anonymous credit cards 1/2/3.
target	Loyalty score target column.

historical_transactions.csv and new_merchant_transaction.csv		
	Mean	Data Type
authorized_flag	Whether to authorize, Y/N	discrete,object
card_id	Credit card id	discrete,object
city_id	City ID (anonymized)	discrete,int64
category_1	Anonymous category feature 1, Y/N	discrete,object
installments	Number of items purchased	continuous,int64
category_3	Anonymous category features 3,A...E	discrete,object
merchant_category_id	Product type id (anonymized)	discrete,int64
merchant_id	Product id (anonymized)	discrete,object
month_lag	Month from the reference date	continuous,int64
purchase_amount	Standardized shopping amount	continuous,float64
purchase_date	Shopping date (time)	Time column, object
category_2	Anonymous category features 2	discrete,float64
state_id	State id (after anonymization)	discrete,int64
subsector_id	Product category group id (anonymized)	discrete,int64

merchant.csv		
	Mean	Data type
merchant_id	Merchant id	discrete,object
merchant_group_id	Commodity group (anonymized)	discrete,int64
merchant_category_id	Product type id (anonymized)	discrete,int64
subsector_id	Product category group id (anonymized)	discrete,int64
numerical_1/2	Anonymous numerical characteristics 1/2	continuous,float64
category_1	Anonymous category features 1	discrete,object
most_recent_sales_range	Sales grade in the most recent active month	discrete,object
most_recent_purchases_range	Number of transactions in the most recent active month Rating	discrete,object
avg_sales_lag3/6/12	Average monthly income for the past 3, 6, and 12 months divided by the income for the previous active month	continuous,float64
avg_purchases_lag3/6/12	Average monthly trading volume for the last 3, 6 and 12 months divided by the volume of the previous active month	Continuous,int64
category_4	Anonymous category features 4	discrete,object
city_id	City id (anonymized)	discrete,int64
state_id	State id	discrete,int64
category_2	Anonymous category features 2	discrete, float64

B. PARAMETER TUNING

Explanation of specific parameters

Random Forest parameter tuning	
	parameter
n_estimators	The number of decision trees, affect overall learning ability. If n estimators is too small, it is easy to underfit, and if n estimators is too large, the model cannot be significantly improved, so n estimators choose a moderate value.
Criterion	Regression tree is an indicator of branch quality, supports mean square error MSE, Feldman mean square error,and absolute mean error MAE.
max_features	The number of features considered when finding the best split point, and the features that exceed the limit will be discarded, affecting the randomness of the model. As this value increases, the attributes considered by each tree increase, and the model generalizes better. But increasing this value will make the algorithm run slower, so a balance needs to be found.
min_samples_split	The minimum sample size required for branching. Usually used in conjunction with max depth to make the model smoother. Setting the number of this parameter too small will cause overfitting, and setting it too large will prevent the model from learning the data.
min_samples_leaf	Restricting a node must contain at least min samples split training samples, this node is allowed to be branched,otherwise the branch will not occur.
max_depth	The maximum depth allowed, all branches exceeding the set depth are cut off . When the decision tree grows one more layer, the demand for the sample size will double, so limiting the depth of the tree can effectively limit overfitting.

XGBoost parameter tuning	
	parameter
Learning rate	The model weights generated by each iteration, which fluctuate between 0.05 and 0.3, are usually set to 0.1 at first.
max_depth	The maximum depth of the tree, used to control overfitting. The larger the max depth, the more 16 specific the model learns.
min_child_weight	The minimum value of the sum of the sample weights in the child nodes.
gamma	Specifies the minimum loss function drop value required for node splitting. The larger the value of this parameter, the more conservative the algorithm will be.
subsample	Controls the random sampling ratio of the tree. Decrease the value of this parameter to avoid overfitting.
colsample_bytree	Determines the proportion of the number of columns randomly sampled for each tree.
Lambda	L2 regularization term for weights. Determines the regularization part of the model, adjusting this parameter to reduce overfitting.
Alpha	L1 regularization term for weights. In high dimensions, make the algorithm run faster.

LGBM parameters tuning		
	Parameters	Properties
Iterative Process	n_estimators, learning_rate, loss, alpha, init	loss_, init_, estimators_
Weak evaluator structure	criterion, max_depth, min_samples_split, min_samples_leaf, min_weight_fraction_leaf, max_leaf_nodes, min_impurity_decrease	
Stop early	validation_fraction, n_iter_no_change, tol	n_estimators_
Training data for weak evaluators	subsample, max_features, random_state	oob_improvement, train_score_
Other	ccp_alpha, warm_start	