USC University of Southern California

Home Credit Competition GLYZ Group Presentation

2024/04/25

Yuhong Liu
Chenjunjie Wang
Haoming Gao
Xueyan Zhang

University of Southern California

USC

Content

1. **Introduction**
2. **Key accomplishment/Contribution**
3. **Description of dataset**
4. **Description of the method focusing and data wrangling**
5. **Description of experiment**
   - 5.1. Machine learning model and hyperparameter tuning
   - 5.2. Ablation studies
6. **Future plan and conclusion**

USC

1. Introduction

USC

Introduction

Background: This Kaggle competition is trying to identify which customers are most likely to experience loan defaults. Long-term stable solutions will be given preference in the evaluation.

This report delves into our strategy and achievements in a Kaggle competition, focusing on a complex credit scoring task presented by Home Credit. We analyzed the dataset, compared the AUC score, and depict the graph to help us visualize. Our methodology harnesses a Voting Model to blend insights from three potent machine learning models, culminating in our highest Kaggle submission score.

2. Key accomplishments

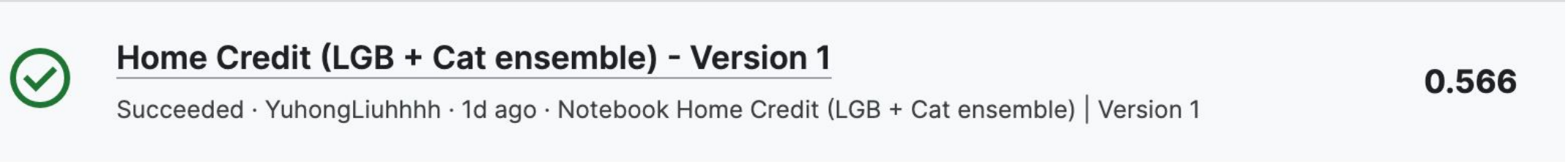
USC

Accomplishment

- For this competition, we have some innovative idea about data wrangling techniques and machine learning methodologies, which significantly enhanced model performance.

- The pinnacle of our achievement was the implementation of a Voting Model that amalgamated the strengths of CatBoost, LightGBM, and XGBoost models.

- This strategic blend led to a remarkable improvement in prediction accuracy, reflected in our best Kaggle submission score.

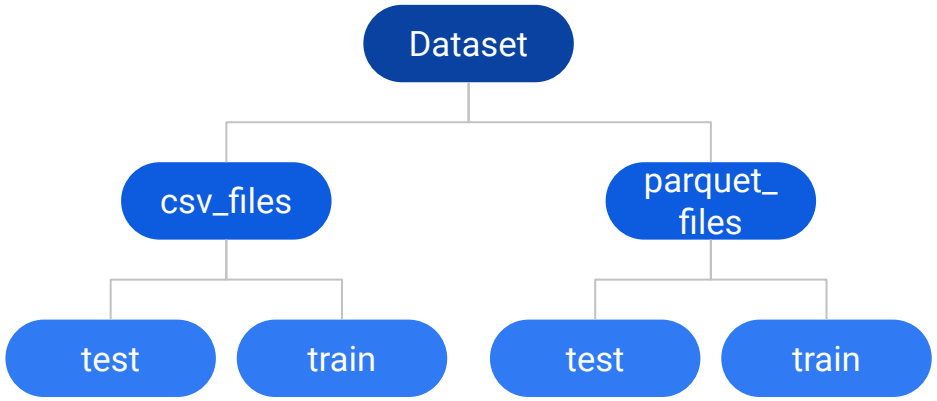


USC

3. Dataset

USC

Description of the Kaggle dataset



01 **Depth**
- depth0
- depth1
- depth2

02 **Internal source**
- Data collected from within the organization, such as records of past interactions with clients, internal credit scores, etc.

03 **External source**
- Data obtained from external entities, like tax registries or other credit bureaus that provide additional information about the clients' creditworthiness.

- **depth0**: Static features directly tied to a specific 'case_id'. These are singular, non-time-varying attributes of a case.
- **depth1**: Indicates that there are historical records for each 'case_id', indexed by 'num_group1'. These could be, for example, previous loan applications or credit checks.
- **depth2**: A more detailed historical record for each 'case_id', indexed by both 'num_group1' and 'num_group2'. This might represent an even more fine-grained history, such as transaction-level data.

USC

4. Data Wrangling

USC

Analysis of the Kaggle dataset

[6]:

	level_0	size_train	path_train	file_name	size_test	path_test
17	0.0	2.9G	/kaggle/input/home-credit-credit-risk-model-st...	credit_bureau_a_2_5	4.0K	/kaggle/input/home-credit-credit-risk-model-st...
16	1.0	2.4G	/kaggle/input/home-credit-credit-risk-model-st...	credit_bureau_a_2_4	4.0K	/kaggle/input/home-credit-credit-risk-model-st...
15	2.0	2.3G	/kaggle/input/home-credit-credit-risk-model-st...	credit_bureau_a_2_3	4.0K	/kaggle/input/home-credit-credit-risk-model-st...
18	3.0	2.2G	/kaggle/input/home-credit-credit-risk-model-st...	credit_bureau_a_2_6	4.0K	/kaggle/input/home-credit-credit-risk-model-st...
21	4.0	1.6G	/kaggle/input/home-credit-credit-risk-model-st...	credit_bureau_a_2_9	4.0K	/kaggle/input/home-credit-credit-risk-model-st...
14	5.0	1.6G	/kaggle/input/home-credit-credit-risk-model-st...	credit_bureau_a_2_2	4.0K	/kaggle/input/home-credit-credit-risk-model-st...
6	6.0	1.5G	/kaggle/input/home-credit-credit-risk-model-st...	credit_bureau_a_1_1	8.0K	/kaggle/input/home-credit-credit-risk-model-st...
20	7.0	1.2G	/kaggle/input/home-credit-credit-risk-model-st...	credit_bureau_a_2_8	4.0K	/kaggle/input/home-credit-credit-risk-model-st...
7	8.0	949M	/kaggle/input/home-credit-credit-risk-model-st...	credit_bureau_a_1_2	8.0K	/kaggle/input/home-credit-credit-risk-model-st...
5	9.0	837M	/kaggle/input/home-credit-credit-risk-model-st...	credit_bureau_a_1_0	8.0K	/kaggle/input/home-credit-credit-risk-model-st...
0	10.0	799M	/kaggle/input/home-credit-credit-risk-model-st...	applprev_1_0	4.0K	/kaggle/input/home-credit-credit-risk-model-st...
19	11.0	706M	/kaggle/input/home-credit-credit-risk-model-st...	credit_bureau_a_2_7	4.0K	/kaggle/input/home-credit-credit-risk-model-st...
11	12.0	671M	/kaggle/input/home-credit-credit-risk-model-st...	credit_bureau_a_2_1	4.0K	/kaggle/input/home-credit-credit-risk-model-st...
29	13.0	645M	/kaggle/input/home-credit-credit-risk-model-st...	static_0_0	12K	/kaggle/input/home-credit-credit-risk-model-st...
1	14.0	544M	/kaggle/input/home-credit-credit-risk-model-st...	applprev_1_1	4.0K	/kaggle/input/home-credit-credit-risk-model-st...
8	15.0	538M	/kaggle/input/home-credit-credit-risk-model-st...	credit_bureau_a_1_3	8.0K	/kaggle/input/home-credit-credit-risk-model-st...
27	16.0	493M	/kaggle/input/home-credit-credit-risk-model-st...	person_1	4.0K	/kaggle/input/home-credit-credit-risk-model-st...
3	17.0	451M	/kaggle/input/home-credit-credit-risk-model-st...	applprev_2	4.0K	/kaggle/input/home-credit-credit-risk-model-st...
10	18.0	448M	/kaggle/input/home-credit-credit-risk-model-st...	credit_bureau_a_2_0	4.0K	/kaggle/input/home-credit-credit-risk-model-st...
12	19.0	384M	/kaggle/input/home-credit-credit-risk-model-st...	credit_bureau_a_2_10	4.0K	/kaggle/input/home-credit-credit-risk-model-st...
30	20.0	366M	/kaggle/input/home-credit-credit-risk-model-st...	static_0_1	12K	/kaggle/input/home-credit-credit-risk-model-st...
32	21.0	240M	/kaggle/input/home-credit-credit-risk-model-st...	static_cb_0	4.0K	/kaggle/input/home-credit-credit-risk-model-st...
35	22.0	119M	/kaggle/input/home-credit-credit-risk-model-st...	tax_registry_c_1	4.0K	/kaggle/input/home-credit-credit-risk-model-st...
33	23.0	116M	/kaggle/input/home-credit-credit-risk-model-st...	tax_registry_a_1	4.0K	/kaggle/input/home-credit-credit-risk-model-st...
28	24.0	95M	/kaggle/input/home-credit-credit-risk-model-st...	person_2	4.0K	/kaggle/input/home-credit-credit-risk-model-st...
4	25.0	45M	/kaggle/input/home-credit-credit-risk-model-st...	base	4.0K	/kaggle/input/home-credit-credit-risk-model-st...
34	26.0	40M	/kaggle/input/home-credit-credit-risk-model-st...	tax_registry_b_1	4.0K	/kaggle/input/home-credit-credit-risk-model-st...
23	27.0	39M	/kaggle/input/home-credit-credit-risk-model-st...	credit_bureau_b_2	4.0K	/kaggle/input/home-credit-credit-risk-model-st...
22	28.0	20M	/kaggle/input/home-credit-credit-risk-model-st...	credit_bureau_b_1	4.0K	/kaggle/input/home-credit-credit-risk-model-st...
25	29.0	4.6M	/kaggle/input/home-credit-credit-risk-model-st...	deposit_1	4.0K	/kaggle/input/home-credit-credit-risk-model-st...
24	30.0	3.6M	/kaggle/input/home-credit-credit-risk-model-st...	debitcard_1	4.0K	/kaggle/input/home-credit-credit-risk-model-st...
26	31.0	1.8M	/kaggle/input/home-credit-credit-risk-model-st...	other_1	4.0K	/kaggle/input/home-credit-credit-risk-model-st...
2	NaN	NaN	NaN	applprev_1_2	4.0K	/kaggle/input/home-credit-credit-risk-model-st...
9	NaN	NaN	NaN	credit_bureau_a_1_4	8.0K	/kaggle/input/home-credit-credit-risk-model-st...
13	NaN	NaN	NaN	credit_bureau_a_2_11	4.0K	/kaggle/input/home-credit-credit-risk-model-st...
31	NaN	NaN	NaN	static_0_2	12K	/kaggle/input/home-credit-credit-risk-model-st...

- First, from the plot, we can observe that static_0_2, credit_bureau_a_1_4, credit_bureau_a_2_11, applprev_1_2 are missing in the train folder

Influence:

- Feature completeness
- Model Bias
- Consistency
- Model Evaluation
- Model Imputation and Augmentation

USC

Analysis of the Kaggle dataset - base table

```
The shape of the train dataset is: (1526659, 5)
The shape of the test dataset is: (10, 4)
```

Display the first 3 rows of the train dataset:

	case_id	date_decision	MONTH	WEEK_NUM	target
0	0	2019-01-03	201901	0	0
1	1	2019-01-03	201901	0	0
2	2	2019-01-04	201901	0	0

Display the first 3 rows of the test dataset:

	case_id	date_decision	MONTH	WEEK_NUM
0	57543	2021-05-14	202201	100
1	57549	2022-01-17	202201	100
2	57551	2020-11-27	202201	100

```
Display the data types of the train dataset:
case_id           int64
date_decision    object
MONTH             int64
WEEK_NUM          int64
target            int64
dtype: object
```

It is essential to focus on the base table since it contains the main entities such as case_id, which can be used to further join other tables. Moreover, it contains the target variable which is the outcome we are trying to predict.

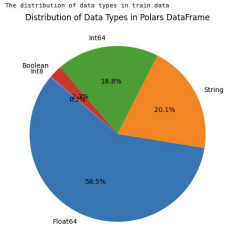
USC

Analysis of Combining Data

After load the dataset and join all different depth files into one overall train and test dataframe:

```
The shape of train data(1526659, 861)
The shape of test data(10, 860)
```

→ Shape of df: train: 1526659, test: 10, feature: 860



→ String type in features occupies more

USC

Analysis of Combining Data

After load the dataset and join all different depth files into one overall train and test dataframe:

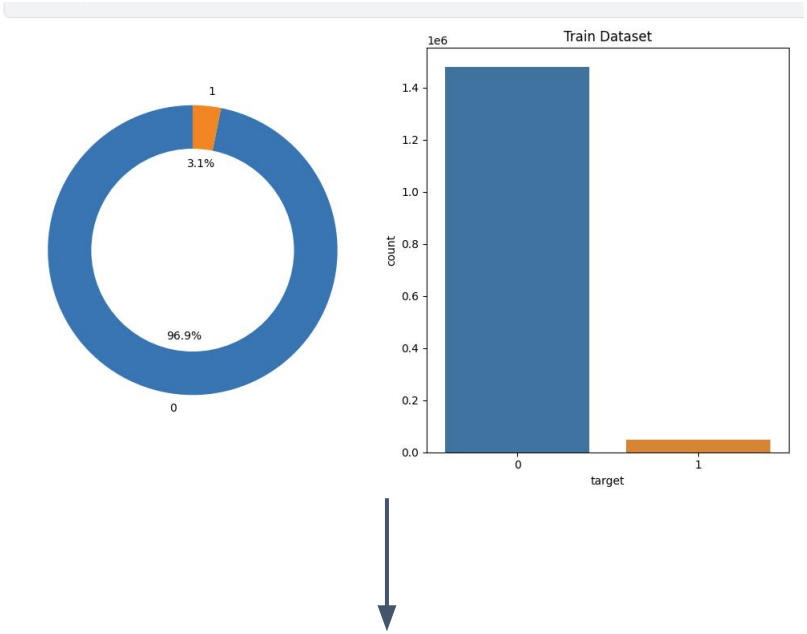
The summary statistics for train data
shape: (9, 862)

statistic --- str	case_id --- f64	WEEK_NUM --- f64	target --- f64	…	max_num_group1_15 --- f64	max_num_group2_15 --- f64	last_num_group1_15 --- f64	last_num_group2_15 --- f64
count	1.526659e6	1.526659e6	1.526659e6	…	1.435105e6	1.435105e6	1.435105e6	1.435105e6
null_count	0.0	0.0	0.0	…	91554.0	91554.0	91554.0	91554.0
mean	1.2861e6	40.769036	0.031437	…	0.08993	0.050694	0.08993	0.050475
std	718946.592285	23.797981	0.174496	…	0.29528	0.465445	0.29528	0.464869
min	0.0	0.0	0.0	…	0.0	0.0	0.0	0.0
25%	766198.0	23.0	0.0	…	0.0	0.0	0.0	0.0
50%	1.357358e6	40.0	0.0	…	0.0	0.0	0.0	0.0
75%	1.739023e6	55.0	0.0	…	0.0	0.0	0.0	0.0
max	2.703454e6	91.0	1.0	…	4.0	31.0	4.0	31.0

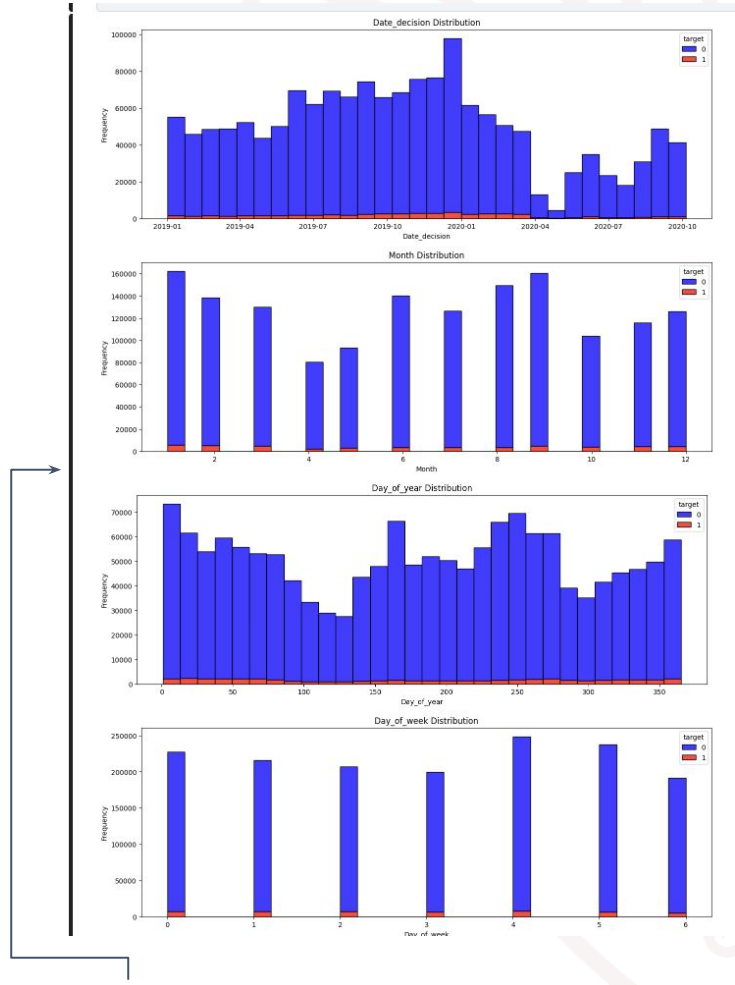
→ statistic for numeric cols

USC

Analysis of Combining Data



This is the class distribution in the training set, we can observe the imbalance here.



Temporal Distribution Analysis of the Target Variable

The dataset is extremely imbalanced, which is expectable for such domain. It seems covid-19 impacted the number of observations greatly.Interesting fact, the bank makes decision even on the weekends. Clearly, this process is automated and runs on a predefined schedule.

USC

How Kaggle dataset related

Description: this dataset is both extensive and intricate, comprising a division into the training set and the testing set. Mainly, we use “case_id” to combine those tables.

Core: the base tables are the central: train_base.csv and test_base.csv, which present core details about the applicants in the training and testing datasets.

Expand the dataset: basing on the core files, the dataset includes the applicants’ credit histories, other institutions, and detailed transaction records. They are be distinguished by either “internal” or “external” and also by the “depth”, which are both mentioned before.

- Within these depths, num_group1 and num_group2 operate as indexing mechanisms that facilitate the integration and synthesis of data across different levels. They are crucial in linking multiple datasets, allowing for a comprehensive aggregation and join tables to analysis applicant information.

Transform the raw data into the input to ML algorithm

USC

5. Experiment - ML model & Hyperparameter Tuning

USC

Experiment

- Description of your experiments, including
  - Machine learning models and hyperparameter tuning
    - Describe and compare the performance of various machine learning algorithms
    - Describe and compare the performance of different parameters
  - Ablation Study

Experimental evaluation criteria

Since the final stability metric, which is the Kaggle score, combines the average Gini score, penalties for the rate of decline, and penalties for variability, the stability metric can be expressed as:

$$\text{stability metric} = \text{mean}(2 \times \text{AUC} - 1) + 88.0 \times \min(0, a) - 0.5 \times \text{std}(residuals)$$

Therefore, by repeating the training five times, we can infer the approximate performance from the mean and standard deviation of the AUC scores. Then, we can compare this performance to compare different models and parameter combinations.

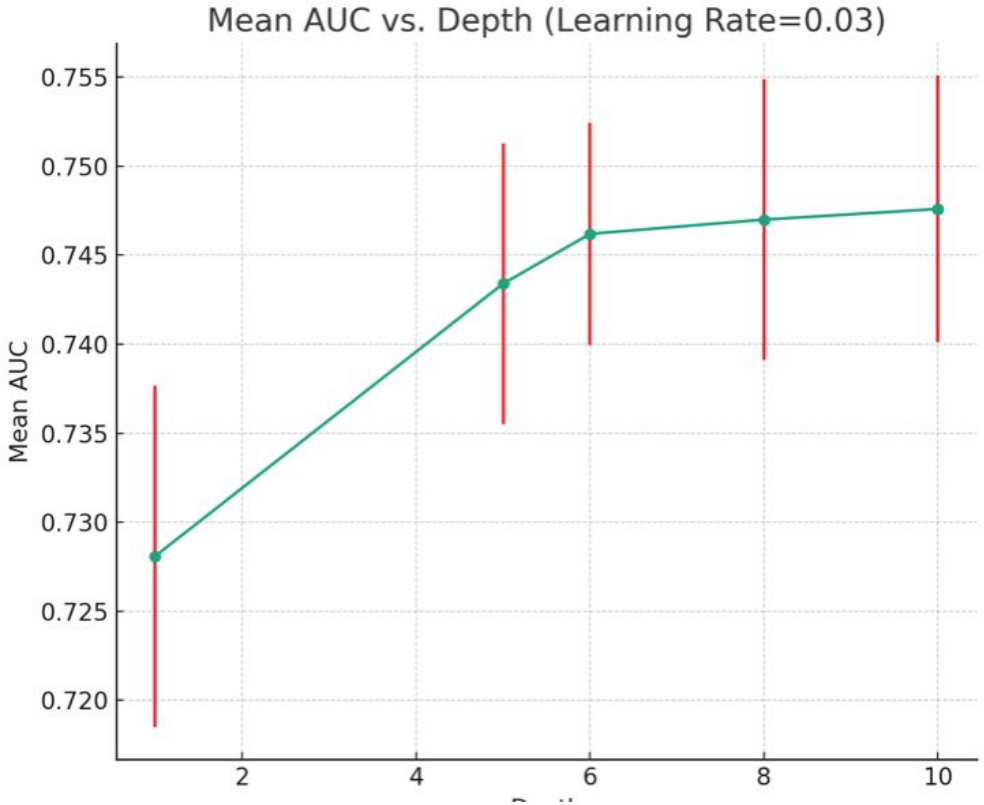
CAT

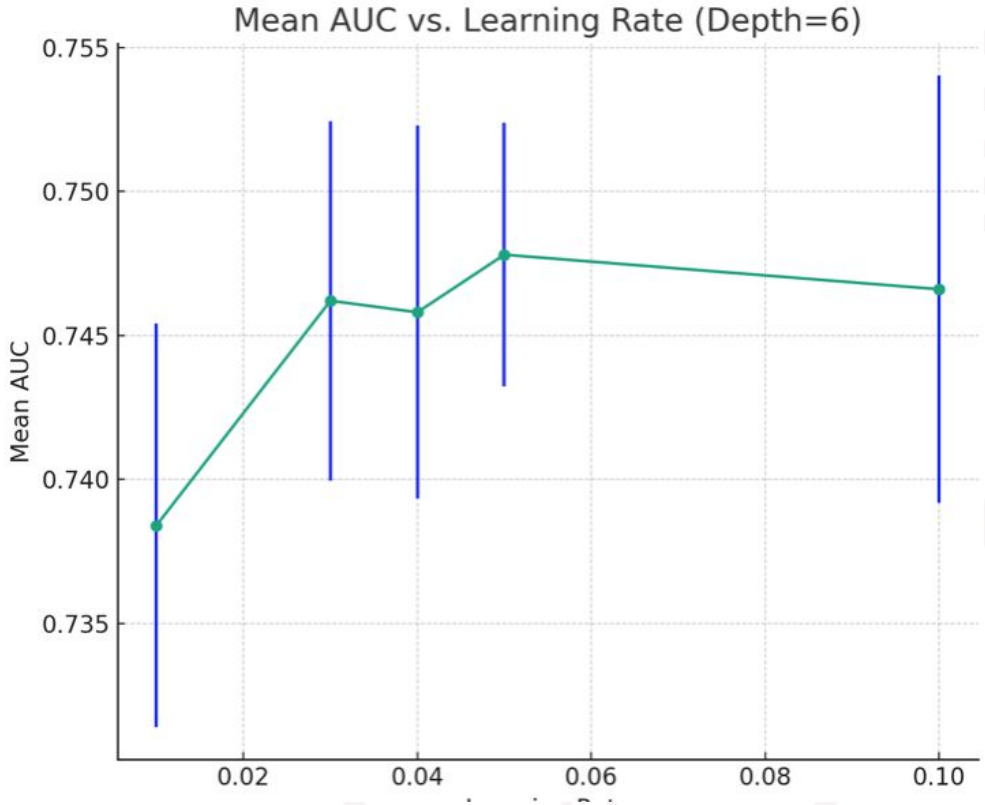
```
clf = CatBoostClassi
eval_metric='AUC',
task_type='GPU',
learning_rate=0.03,
depth=6,
iterations=n_est)
```

Learning_rate : depth	0.03 : 6	0.01 : 6	0.05 : 6		0.04 : 6
AUC mean	0.7462	0.7384	0.7478		0.7458
AUC variance	0.000039	0.000049	0.000021		0.000042
Learning_rate : depth	0.1 : 6	0.03 : 5	0.03 : 10	0.03 : 1	0.03 : 8
AUC mean	0.7466	0.7434	0.7476	0.7281	0.7470
AUC variance	0.000055	0.000062	0.000056	0.000092	0.000622

USC

CAT





Best parameter combination: lr=0.05, depth=6

USC

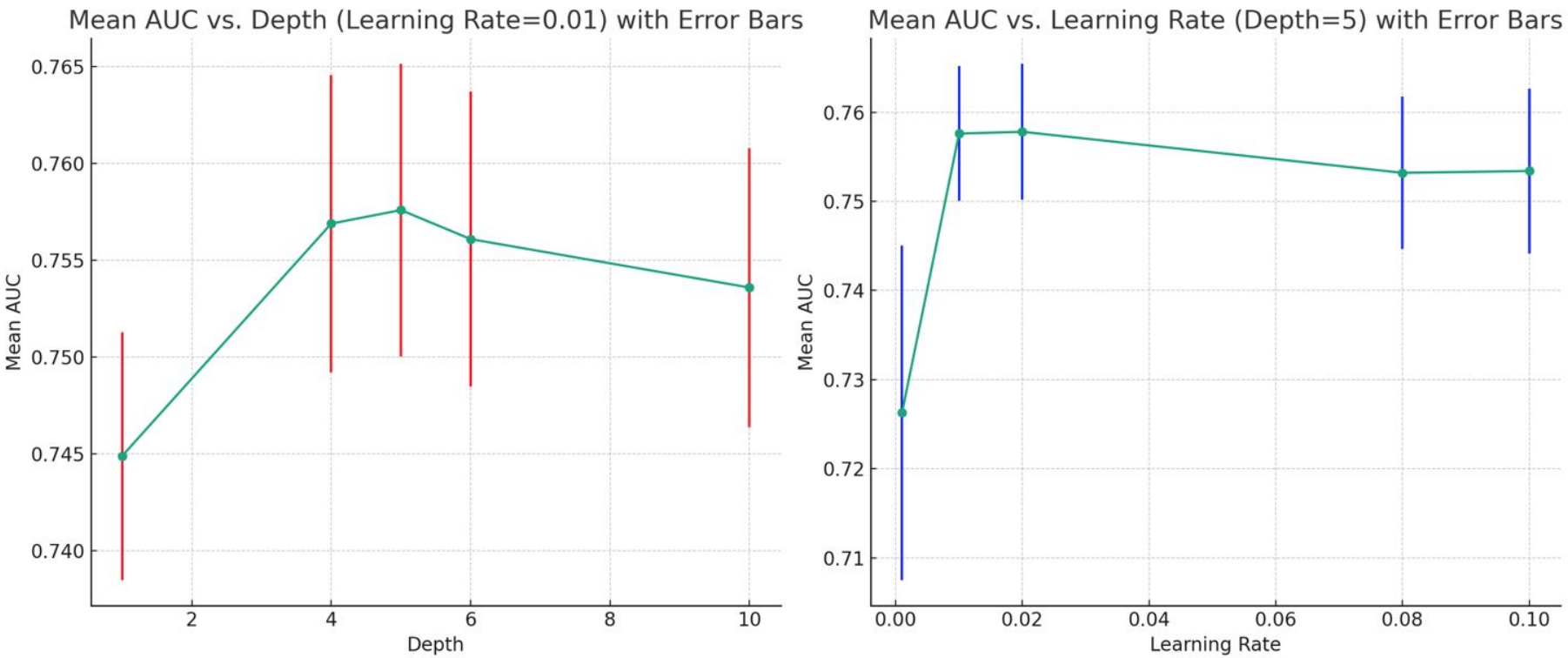
LGB

```
params = {
    "boosting_type": "gbdt",
    "objective": "binary",
    "metric": "auc",
    "max_depth": 10,
    "learning_rate": 0.05,
    "n_estimators": 2000,
    "colsample_bytree": 0.8,
    "colsample_bynode": 0.8,
    "verbose": -1,
    "random_state": 42,
    "reg_alpha": 0.1,
    "reg_lambda": 10,
    "extra_trees":True,
    'num_leaves':64,
    "device": device,
    "verbose": -1,
}
```

Learning_rate : depth	0.05 : 10	0.01 : 5	0.01 : 10	0.01 : 4	0.01 : 6
AUC mean	0.7537	0.7576	0.7536	0.7569	0.7561
AUC variance	0.000075	0.000057	0.000052	0.000059	0.000058
Learning_rate : depth	0.02 : 5	0.1 : 5	0.001 : 5	0.01 : 1	0.08 : 5
AUC mean	0.7578	0.7534	0.7263	0.7449	0.7532
AUC variance	0.000058	0.000085	0.000352	0.000041	0.000732

USC

LGB



Best parameter combination: lr=0.01, depth=5

USC

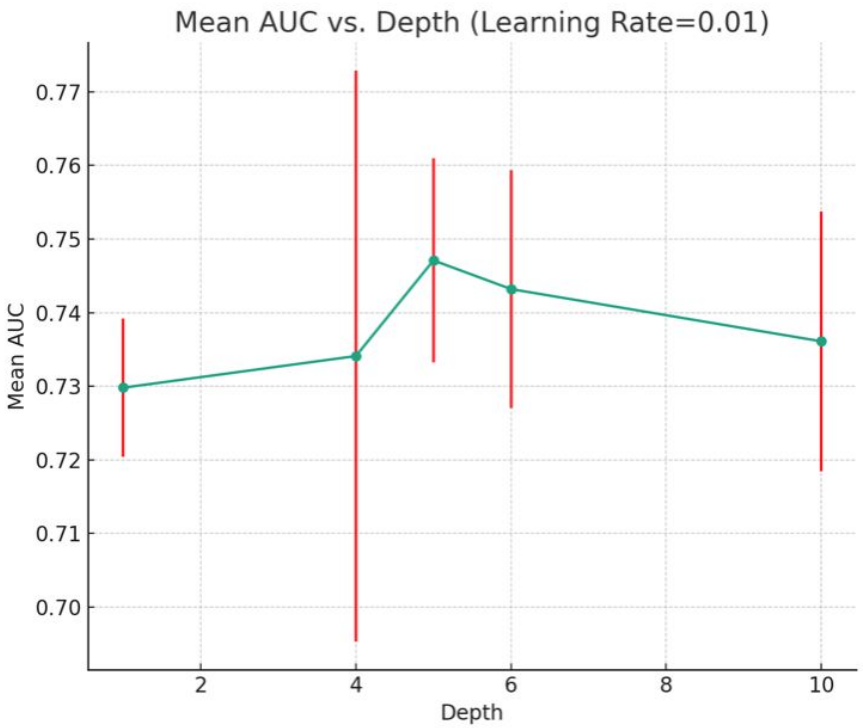
XGB

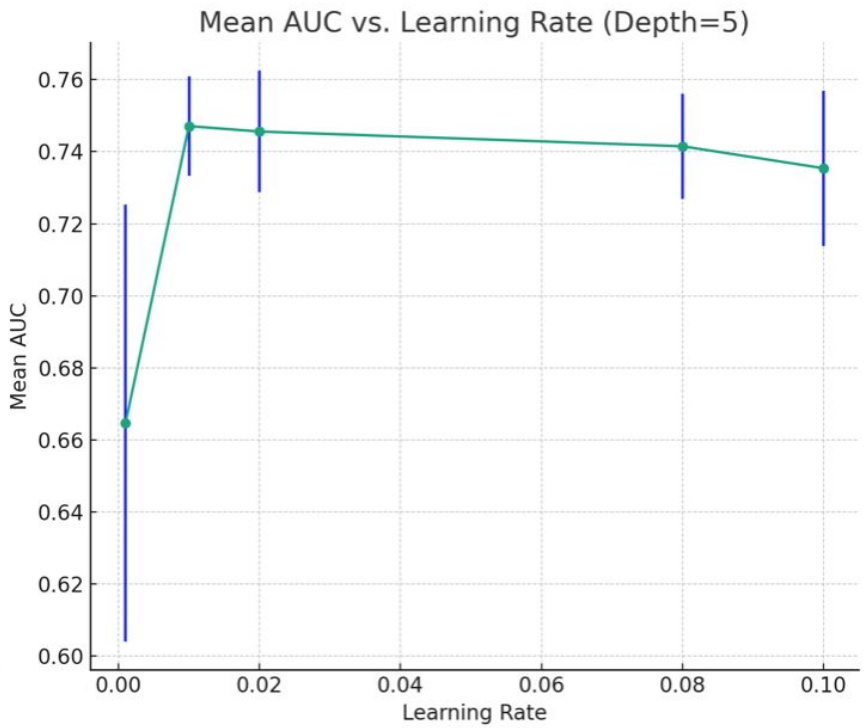
```
params2 = {
    "booster": "gbtree",
    "objective": "binary:logistic"
    "eval_metric": "auc",
    "max_depth": 10,
    "learning_rate": 0.05,
    "n_estimators": 1000,
    "colsample_bytree": 0.8,
    "colsample_bynode": 0.8,
    "alpha": 0.1,
    "lambda": 10,
    "tree_method": 'gpu_hist'
    if device == 'gpu' else 'auto'
    "random_state": 42,
    "verbosity": 0,
    "enable_categorical":True,
```

Learning_rate : depth	0.05 : 10	0.01 : 5	0.01 : 10	0.01 : 4	0.01 : 6
AUC mean	Mean: 0.7279,	Mean: 0.7471,	Mean: 0.7361,	Mean: 0.7341	Mean: 0.7432
AUC variance	0.000281	0.000192	0.000311	0.001503	0.000262
Learning_rate : depth	0.02 : 5	0.1 : 5	0.001 : 5	0.01 : 1	0.08 : 5
AUC mean	0.7456	0.7354	0.6647	0.7298	0.7415
AUC variance	0.000284	0.000463	0.003672	0.000089	0.0002135

USC

XGB





Best parameter combination: lr=0.01, depth=5

USC

Model Comparing

When we compare the best parameter for each model, we can find that LGB model can perform best when dealing with the Kaggle problem because its AUC score has the maximum mean and the smallest standard deviation. Hence we use the LGB parameters for the further hyperparameters tuning comparison

USC

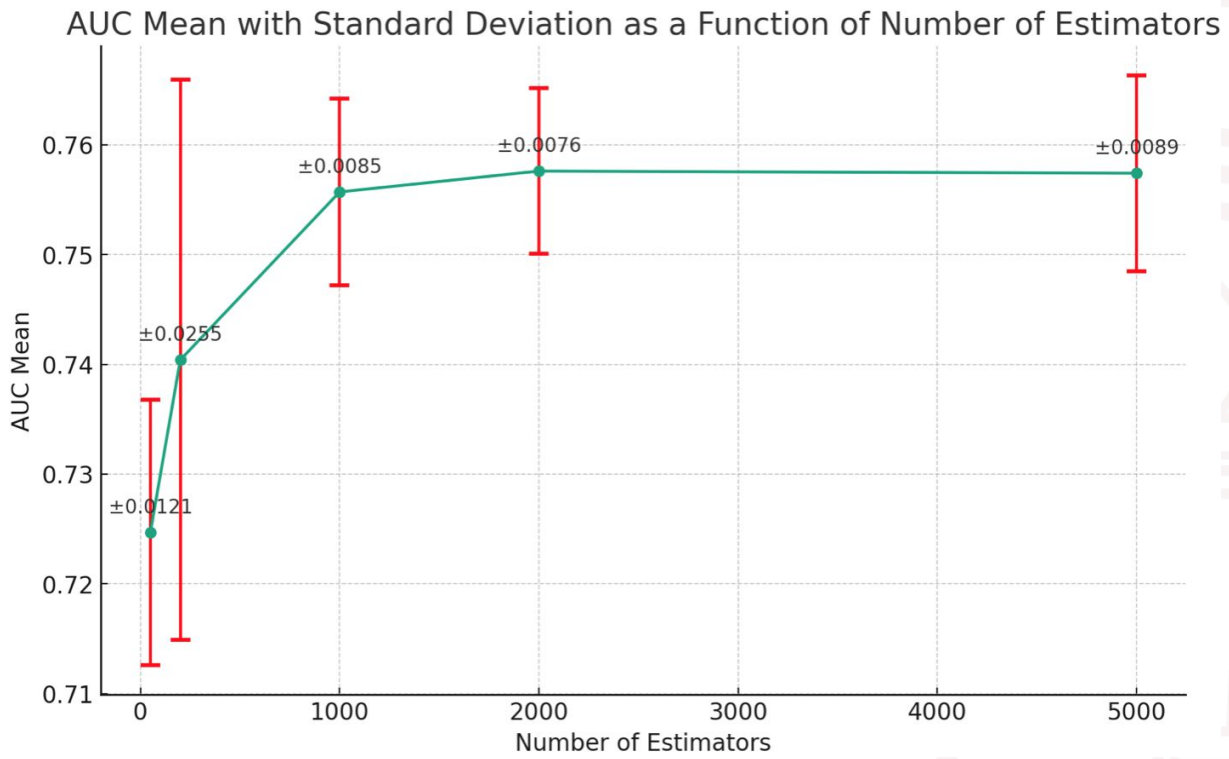
Hyperparameter Tuning - LightGBM

- Hyperparameters:
  - max_depth
  - learning_rate
  - n_estimators
  - num_leaves
  - L2 regularization parameter
    - reg_alpha
    - reg_lamb

num_estimators

n_estimators	50	200	1000	2000	5000
AUC mean	0.7247	0.7404	0.7557	0.7576	0.7574
AUC std	0.0121	0.0255	0.0085	0.00755	0.0089

num_estimators



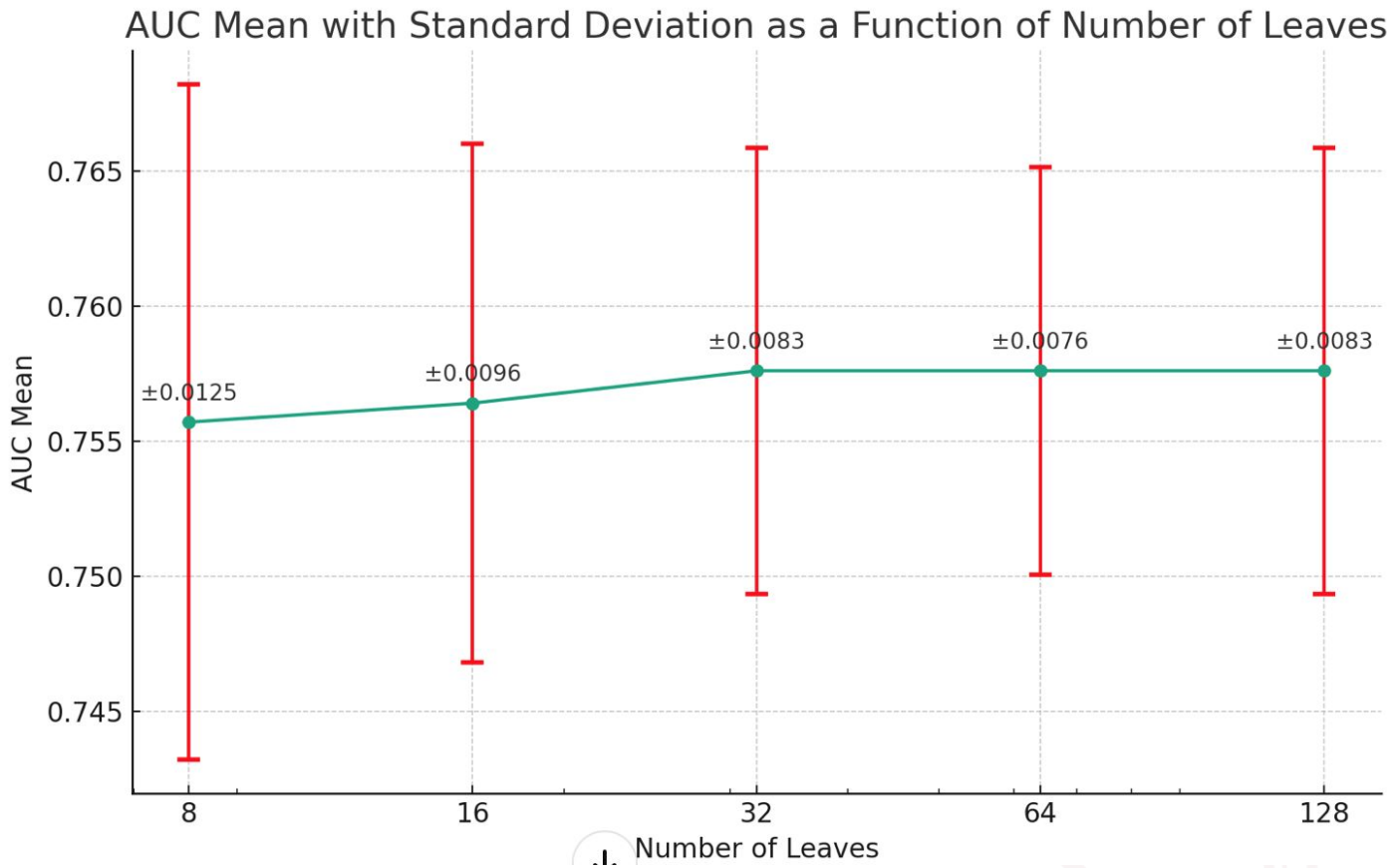
From the graph, we can see that n_estimator = 2000 has the best performance with the largest mean and the smallest std

USC

num_leaves

num_leaves	8	16	32	64	128
AUC mean	0.7557	0.7564	0.7576	0.7576	0.7576
AUC std	0.0125	0.00896	0.00825	0.00755	0.00825

num_leaves



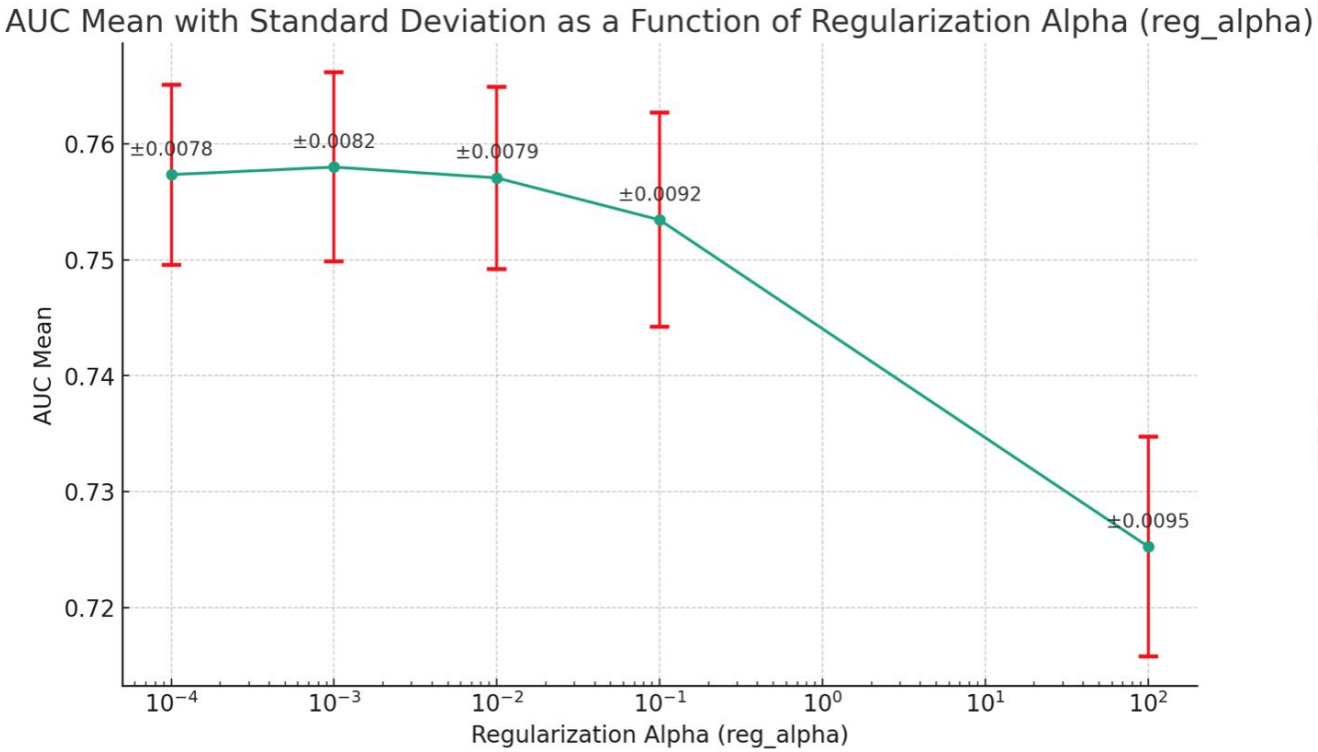
From the graph, we can see that num_leaves = 64 has the best performance with the largest mean and the smallest std

USC

reg_alpha

reg_alpha	0.0001	0.001	0.01	0.1	100
AUC mean	0.75734	0.75799	0.75706	0.75344	0.72526
AUC std	0.00776	0.00816	0.00787	0.00924	0.00950

reg_alpha



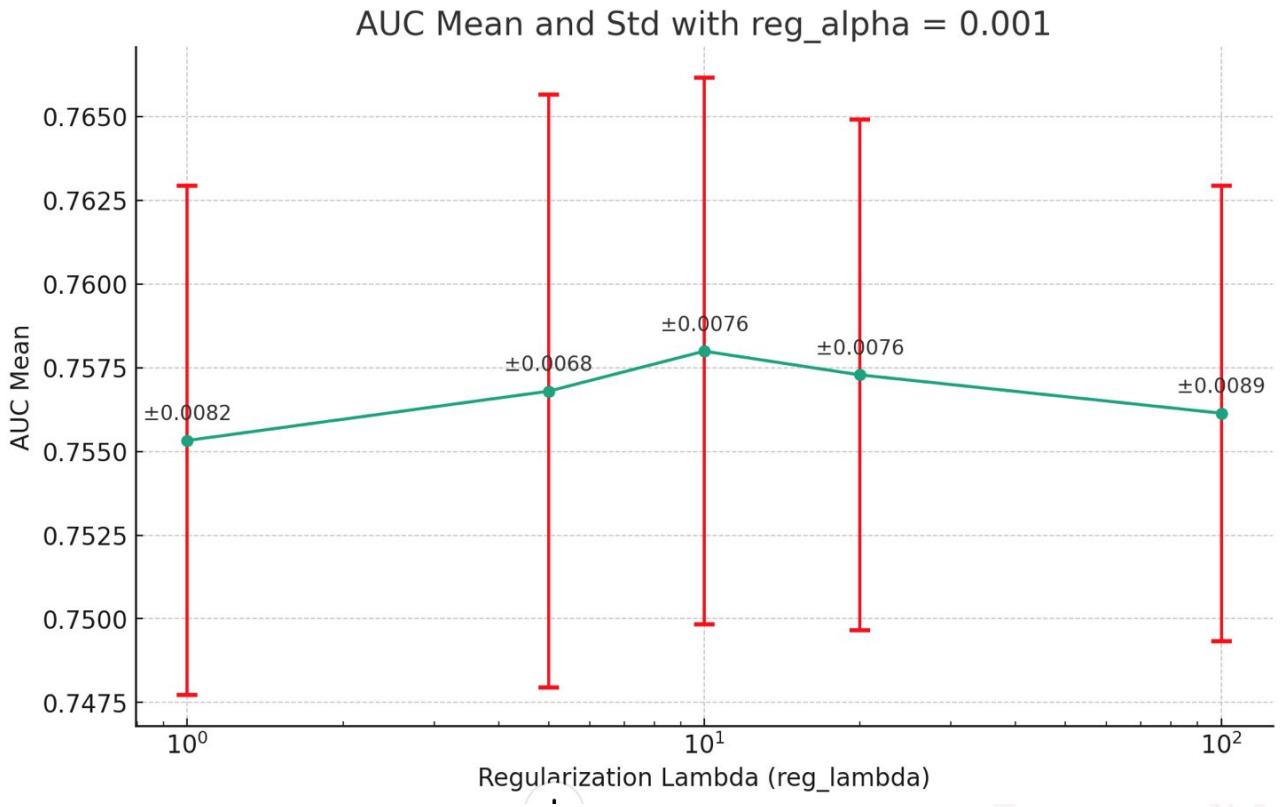
From the graph, we can see that reg_alpha = 0.001 has the best performance with the largest mean and the smallest std

USC

reg_lamb

reg_lamb	1	5	10	20	100
AUC mean	0.75532	0.75679	0.75799	0.75728	0.75614
AUC std	0.00761	0.00885	0.00815	0.00761	0.00680

reg_lamb



From the graph, we can see that reg_lamb = 10 has the best performance with the largest mean and the smallest std

USC

Kaggle evaluation criteria

Kaggle evaluation criteria is shown below:

$$\text{stability metric} = \text{mean}(gini) + 88.0 \times \min(0, a) - 0.5 \times \text{std}(residuals)$$

AUC and stability index have positive correlation to some degree. First, Gini index relies on AUC based on the formula: Gini = 2*AUC -1; hence, it is correlated with AUC. Also, the mean Gini can reflect the stability of the model along with time. However, the stability index also considers two other factors, which are falling rate and the standard deviation of residual. Hence, although the AUC and stability index are positively related, the stability index also includes the stability of the model over time, which means the stability index is maximized only if the model maintains a stable high AUC throughout the evaluation cycle.

Dealing with Kaggle evaluation criteria

- First, we need to deal with the **data preparation** and the **features**. For the data preparation, we have the data reading and preprocessing, and then we use the pandas and polars to do the data clean and data transform. For the features, we expand the feature space by adding features, aggregating data, deleting the irrelative and sparse features.

- Then, we deal with the **model training** and **evaluation**. We used the CatBoost, Light GBM and XGBoost, etc to improve the robustness and accuracy for prediction, and then utilized cross validation strategy to make sure the model validation is comprehensive.

- Lastly, we a**ggregate the predictions of multiple models** through the **voting mechanism** of the “Voting Model.” After having the cross validation and aggregation method, we can deal with the stability to ensure that the performance will work well on different time or data segments. Having this procedure, we can have the high accuracy and robustness, and adaptability which can better deal with the Kaggle evaluation.

Ablation Study - Contribution of various data wrangling steps to the performance

1. Memory Optimization: Clearing “data_store” and Optimizing data types is key for enhancing computation speed and reducing memory footprint.

2. Missing Value Patterns: Grouping columns by the number of missing values can inform better data imputation and cleaning strategies.

3. Feature Selection: Removing columns with over 70% missing values & Choosing high feature correlation and highly unique, non-redundant features ensures the model captures the most discriminatory information.

Contribution of various machine learning modeling steps to the performance

1. A common strategy for model optimization involves ensemble methods, such as the **voting model** used here, which combines multiple models to improve overall predictive performance

2. Model fusion can balance these shortcomings and often achieve higher accuracy, and helps understand how each model performs in different aspects and provides a basis for the final model

3. Subsequently, voting mechanisms or more complex fusion strategies (such as stacking or blending) can be used to combine the predictions of these models. In this process, not only the model predictions are used to train the fusion model, but also specific features of the models or the uncertainty of their predictions can be included.

6. Future Plan

USC

Conclusion and future work

- Our approach in this Kaggle competition has demonstrated that careful data wrangling, coupled with sophisticated model ensembling, can lead to high-performing predictive models.

- The Voting Model's success lies in its capacity to capture diverse data patterns and reduce model-specific biases.

- As we advance, our plan is to deepen our exploration of feature engineering, augmenting the dataset with more predictive features while maintaining a keen eye on model complexity. We will extend our hyperparameter tuning, employing advanced techniques like Bayesian optimization to fine-tune our models further. Moreover, we will explore ensemble methods beyond voting, like stacking, to evaluate if these can yield even better results.

USC

Thanks for listening !

Q&A

USC