CS583A: Santander Customer Transaction Prediction

YIFENG LIU

May 20, 2019

1 Summary

[Problem descriptions:] I participate an active competition of Santander Customer Transaction Prediction. [Methodology:] The final model we choose is LightGBM, a gradient boosting framework that uses tree based learning algorithms. It is designed to have some advantages: faster training speed and higher efficiency, lower memory usage, better accuracy, support of parallel and GPU learning, capable of handling large-scale data. I also use Neural Network and XGBoost. [Implementation:] I implement the neural network using Keras and run the code on Colab a Google online Python Notebook tool with GPU, implement the XGBClassifier from package xgboost, implement the lightGBM from package lightgbm. [Evaluation metric:] Performance is evaluated on the roc_auc_score from package Sklearn. [Score and ranking:] In the public leaderboard, our score is 0.90115; we rank 886 among the 8802 teams. In the private leaderboard, our score is 0.89975; we rank 2567 among the 8802 teams.

2 Problem Description

Problem. This competition purpose is predicting their customer's future decision of making money transaction or not. This is a binary classification problem. They want to use the machine learning model to predict the customer's behavior, and then provide the better service for customers. The competition is at `https://www.kaggle.com/c/santander-customer-transaction-prediction`.

Data. The dataset separate into train data and test data, the test data separate into private test data and public test data. The two datasets are $20,0000 \times 200$ dataframes. The number of training samples is $n = 20,000$. The number of classes is 2. The training set is imbalanced: $n_1 = 10\%$ which means the customers will make a transaction and $n_0 = 90\%$ means will not. For every feature, the data is normalized, they are all in good Gaussian distribution, and doesn't have outlier data.

Challenges. In order to safe the customers' information, the data doesn't have features' name, I cannot know the real means of these features, that make it is difficult to normalized data. The data already be regularization and normalized, it's hard to explore the detail information behind the features. The target is imbalanced, it's hard for model to detect the relationship between the features and target. The feature and feature, the feature and target have very low correlation, it is hard to find the relationship between features and target.

3 Solution

Model. Finally I choose to ensemble my three model LightGBM, XGBoosting, and NN. A description of LightGBM is online: `https://github.com/microsoft/LightGBM`. A description of XGBoosting is online: `https://github.com/dmlc/xgboost`, I use the XGBClassifier to make the binary classification. The NN use the Dense layers, BatchNormalization layers, and Dropout layer.

Implementation. I build my NN by Keras, it has 12 layers, and the final activation is the sigmod. The result is a $1\times20{,}000$ numpy array. I implement LightGBM from the package lightgbm, and result is a $1\times20{,}000$ numpy array. I implement XGBoosting from package xgboost. My code is available at `https://github.com/pallasathena92/NN/blob/master/final.ipynb`. I run the code on Colab, the cloud GPU make the speed become faster, and it takes more than 10 hours to train the model, and I spend 8 hours on running adjust hyper-parameter method.

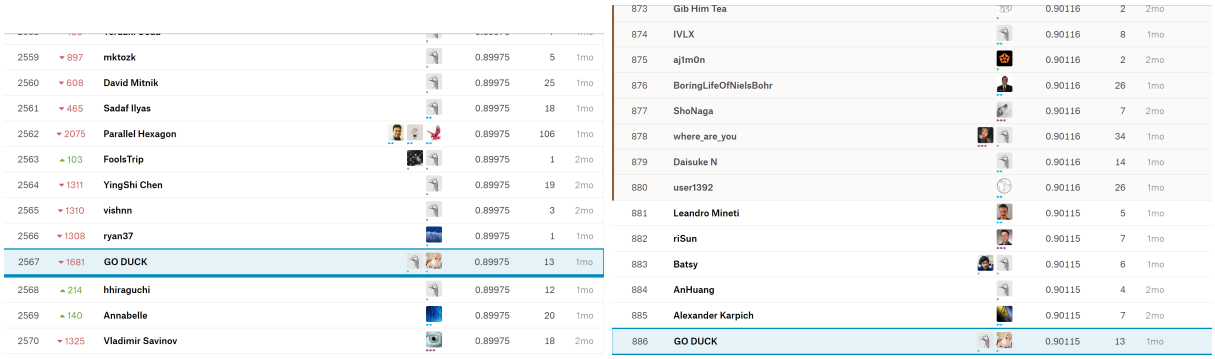
Settings. In the NN model, the loss function is binary cross-entropy. The opitmizer is RMSprop, learning rate is $2E-4$, batch_size is 32 and epoches is 50. And I use regularization method are dropout and batchnormalization layers to avoid the overfitting. In XGBoosting model, the learning rate is 0.27003, the LightGBM's learning rate is 0.0083.

Advanced tricks. In XGBoosting model I use the BayesSearchCV to find the best combination of parameters, which almost spend 8 hours. In NN model, because the weak lineal relationship between features and target, I try to use the polynormial features method to add the linear relationship between features. In order to deal with the imbalanced data, I make the data augment at the first, randomly choose the features which belongs to target 1 to ensemble a new train sample which target is 1, after data augment, I get 20% train data belong to target 1. And because after adjust hyper-parameter, use the different model, the outcome doesn't have a big improvement. After analyze the data, I find there is some fake data in the test data, and all the train data is true. And then add the count of every feature' value, the LightGBM get a better outcome, because the LightGBM is just sensitive to the horizontal information, when I add the count of feature value that provide the vertical information of data to the model.

Cross-validation. I tune the parameters using 5-fold cross-validation in LightBGM model and add the early stop rounds is 4000. In XGboosting model, I use the StratifiedFold to split train data and cross-validation data and add the early stop rounds is 10, that is a reason why the XGBoosting is faster than LightGBM in this project. In NN model, I split 90% train data and 10% validation data by shuffling data index.

4 Compared Methods

Because use the different adjust hyper-parameter method, I can't compare these three models' speed. The CV score of LightGBM is 0.92279, the CV score of XGBoosting is 0.59, the CV score of NN is 0.8997, so when I ensemble these three model, I give the high wight to the LightGBM model is 0.4 and the same weight to NN model, and the XGBoosting model's weight is 0.2. I use the StratifiedFold in XGBoosting, which make the speed of XGBoosting is faster than LightGBM use KFold with 11 folds.



2559	▼897	mktozk	0.89975	5	1mo
2560	▼608	David Mitnik	0.89975	25	1mo
2561	▼465	Sadaf Ilyas	0.89975	18	1mo
2562	▼2075	Parallel Hexagon	0.89975	106	1mo
2563	▲103	FoolsTrip	0.89975	1	2mo
2564	▼1311	YingShi Chen	0.89975	19	2mo
2565	▼1310	vishnn	0.89975	3	2mo
2566	▼1308	ryan37	0.89975	1	1mo
2567	▼1681	GO DUCK	0.89975	13	1mo
2568	▲214	hhiraguchi	0.89975	12	1mo
2569	▲140	Annabelle	0.89975	20	1mo
2570	▼1325	Vladimir Savinov	0.89975	18	2mo

(a) Private leaderboard.

873	Gib Him Tea		0.90116	2	2mo
874	IVLX		0.90116	8	1mo
875	aj1m0n		0.90116	2	2mo
876	BoringLifeOfNielsBohr		0.90116	26	1mo
877	ShoNaga		0.90116	7	2mo
878	where_are_you		0.90116	34	1mo
879	Daisuke N		0.90116	14	1mo
880	user1392		0.90116	26	1mo
881	Leandro Mineti		0.90115	5	1mo
882	riSun		0.90115	7	1mo
883	Batsy		0.90115	6	1mo
884	AnHuang		0.90115	4	2mo
885	Alexander Karpich		0.90115	7	2mo
886	GO DUCK		0.90115	13	1mo

(b) Public leaderboard.

Figure 1: Our rankings in the leaderboard.

5 Outcome

I participated in an active competition. Our score is 0.90115 in the public leaderboard and 0.89975 in the private leaderboard. I rank 886/8802 in the public leaderboard and 2567/8802 in the private leaderboard. The screenshots are in Figure 1.

6 feature work

I want to use the same adjust hyper-parameter method and cross-validation method to compare these models' speed. And I want to find out why the performance of XGBoosting is bad in this project.