A Hybrid approach for Users Retention Rate Prediction

How to name our team

Haocheng Xu†
University of Electronic Science and Technology of China
Chengdu, China
smile.xuhc@gmail.com

Siyi Liu
University of Electronic Science and Technology of China
Chengdu, China
ssui.Liu1022@gmail.com

Jiaxin Wu
University of Electronic Science and Technology of China
Chengdu, China
wjx990812@gmail.com

ABSTRACT

The WSDM Cup's Retention Rate of Baidu Hao Kan APP Users Challenge refers to the task of estimating whether the new user will use Baidu Hao Kan App the next day. Baidu Hao Kan APP is one of the most popular platforms that provides users with massive high-quality short video content.

In this paper, we present a hybrid approach to this user retention rate issue in short video platform. The proposed method includes building two kinds of commonly-used classifiers as our basic models for retention rate prediction, i.e. Gradient Boosting Decision Trees (GBDT) models and deep learning models. After that, ensemble methods are applied to all models, whose results significantly improve upon the best single model.

In the competition, we eventually won the third place with an ROC-AUC score of 0.7646. The code of our solution is available at Github[1].

CCS CONCEPTS

•**Computing methodologies**→Classification and regression trees; Neural networks;

KEYWORDS

Feature Engineering, GBDT, Deep Learning, Stacking Technique

1 INTRODCTION

The challenge provides the real-world user data collected from Baidu Hao Kan APP, which includes user portrait data (gender, age, education, geography, interest, etc.), user watching behavior, APP usage data, installation source, etc. The goal of this competition is estimate whether a new user will use Baidu Hao Kan APP the next day. The users who will use the APP again to watch videos the next day are called retained users.

In order to justify the performance of the models, this competition uses AUC (Area under the Curve of ROC) as the evaluation metric, which is a criterion for judging the pros and cons of a two-class prediction model. The ROC curve (receiver operating characteristic curve) is a commonly used measure that reflects the continuous variables of sensitivity and specificity. Each point on the ROC curve reflects the sensitivity to the same signal stimulus.

Our proposed approach to tackle this challenge incorporates the Gradient Boosting Decision Trees (GBDT) models as our main models and deep learning models as base models. As to our GBDT and Fully Connected Neural Network models, the input of these models are extracted from raw data through feature engineering. As to our deep learning models (except Fully Connected Neural Network), we apply embedding, one-hot encoder and label encoder in data preprocessing. Final model is further boosted by the application of ensemble methods to our base models. The main model based on GBDT plays a major role in the competition. Although the deep learning models did not perform as well as it, they still made much contribution after ensemble.

The rest of the paper is organized as follows: In Section 2, we describe the details of the training and testing datasets; In Section 3, we discuss the components of our approach, including gradient boosted decision tree models, deep learning models and model ensemble; Section 4 reports the model performance; Finally, we conclude our analysis and the future work of the task.

2 THE DATASET DESCRIPTION

The dataset that was provided in the Challenge consists of user portrait data (gender, age, education, geography, interest, etc.), user watching behaviors, APP usage data, installation data, etc. From the datasets we could discard the territory code feature because it only contains one class. The data contains a 16-day users' behavior log which starts from Oct. 16, 2018 and ends at Oct. 31, 2018. Table 1 describes the information about the user and log numbers in training and testing datasets.

Table 1: Information about the Dataset

	training	testing
user number	1351424	500951
record number	97480100	36073183

Table 2 shows the number of unique value of each categorical feature in training and testing datasets.

[1] https://github.com/Smilexuhc/2019WSDMCUp-baidu-user-retention-rate-prediction

Table 2 Information about the Number of Unique Value of Categorical Features

	Gender	Age	Education	Installation channel	Video ID	Video type	Video tag	Video creator	Show	Click/Comment/Like/share	Rec type	Tag number
training	3	8	4	853	1212133	229	901701	85823	2	3	3	232336
testing	3	8	4	770	841560	218	641534	74467	2	3	3	184661
total	3	8	4	870	1340563	229	990180	89325	2	3	3	247157

Each user contains one or more logs. Detailed information about the length of each log is summarized in Table 2.

Table 3: Information about the Length of Each User Records

	training	testing
maximum	19954	18396
minimum	1	1
mean	72	72
25%	16	16
50%	28	28
75%	71	71
92%	200	199

We can observe that about 92% of the log lengths are less than 200.

3 APPROACH

In this section, we will introduce our method to extract features from raw data and different models. Our main models are GBDT models, including Xgboost[1], LightGBM[2] and Catboost[3]. Considering that the task is similar to a recommender system problem, we also build deep learning models, including Bi-directional Recurrent Neural Network, Bi-directional Recurrent Neural Network with self-Attention, Multi-Head Attention Mechanism and TextCNN[4]. We will detail the method to extract features for GBDT models and Fully Connected Neural Network models and preprocessing of other deep learning models. The ensemble method of all our models to further improve accuracy will be illustrated lastly.

3.1 GBDT and Fully Connected Neural Network Model

3.1.1 Feature Extraction. According to the business scenario, there are two kind of features to extract. First, we generate user profile. Then, users' interest about the categories and tags of videos can be caught by exploring users' click, like, share and comment behaviors, and we compare them with the interest caught by APP's recommender system. A bad recommender system may bore users. Our features are stated below:

1. Users' basic personal information: gender, age, degree of education and etc.
2. Users' action information: By respectively computing all the actions' happening time, we can get count features. Then, we can get time-interaction features by computing all the action's statistic information like maximum, mean, minimum, variance, skewness, kurtosis and etc.
3. Users' video-watching features: We compute statistic information on videos' duration, upload time, watching duration, watching rate (use video's length to divide video's watching duration) and intervals (time-video's upload time).
4. Users' video-interaction features: These features are built to describe users' interest. First, we compute the classifications of user's most watched 5 videos' and 500 most used tags, of which videos are clicked, liked, shared and commented. We find that the most important features are ones that describe a user's previous interaction with the video itself and other similar videos[8]. We further explores whether users has watched the same video and clicked the same video creator more than once. Then we can count the times of users' watching a certain video and or the videos from its' creator.
5. Interest-match features: We can explore all the videos that were recommended as well as users' interest caught by the app's recommender system. Then we compare it with users' real interest concluded from the clicking, liking, sharing and commenting action.
6. Nonlinear features: Some numerical features are calculated by root and square operation in order to increase the nonlinearity of data.
7. Interaction features: Some categorical features are interacted with others.

3.1.2 Feature Selection. In total, we created more than 1200 features from the raw data. The huge number of features greatly increase the time of training and the risk of overfitting. In addition, some features have negative effect on the model's performance. Therefore, we employ a simple strategy in feature selection. We respectively compute LightGBM, Xgboost and Catboost models' feature importance and take the intersection of their 300 most important features. The left features are divided into groups based on the way that were built. Then we re-train the model with these features added in groups. Features which increased accuracy in offline validation process are added to the test model and otherwise are discarded. Some features which exhibit an increase in model accuracy but were found to be redundant based on its high correlation with others are also manually discarded.

3.2 Deep Learning Model

In recent years, deep learning has achieved excellent performance in many sequence-related tasks. From the dataset we can see that

each user's behavior in one day is sequential, so we can use those models which are widely used in Natural Language Processing and Recommender System tasks in our task.

3.2.1 Data Preprocessing. Deep learning requires a different form of data from the models we mentioned above. Deep learning models cannot handle category features directly, so as to the category feature, we apply embedding, one-hot encoder and label encoder in data preprocessing. As to the features with a large number of categories (installation_channel, video_id, video_type, video_tag, video_creator), we use embedding to reduce the dimension of each feature, others features are preprocessed through one-hot encoder or label encoder.

Because of different users have different record lengths, we unify the user's sequence length to 200. From Table 2 we can see that about 92% of users have a record length of less than 200, so we choose 200 to maximize the integrity of the information and prevent users who are highly active from affecting the loss during training [8]. If a user's record length is less than 200, the vacancy is filled with 0, otherwise truncated.

3.2.2 Model Selection. We build four different deep learning models for this task.

1. Bi-directional Recurrent Neural Network
   Recurrent neural network are widely used in modeling sequential data. Gated Recurrent Unit (GRU) networks[5] are a type of RNNs that have been shown to optimization issues that plague vanilla-type RNNs. In this task, we designed a Bi-directional GRU model with the hidden state of 512 to tackle this challenge.
2. Bi-directional Recurrent Neural Network with self-Attention
   Attention mechanism have become an integral part of compelling sequence modeling[6]. Based on the RNN model we mentioned above, we also applied self-attention mechanism in it. Self-attention mechanism could allow model to automatically (soft-)search for parts of user's records that are relevant to predicting users retention.
3. Multi-Head Attention Mechanism
   Ashish et al. (2017) used a new simple network architecture based solely on attention mechanism for machine translation tasks[7]. Inspired by this, we designed a model based solely on attention mechanism for this challenge.
4. TextCNN
   TextCNN network is a Convolutional Neural Network proposed for text classification in 2014. Due to its simple structure and sound effect, it is widely used in natural language processing field. So we applied it for this classification tasks.

3.3 Model Ensemble

To make our model predict more accurately, we performed a three-stage strategy which combined stacking and linear combination. The detailed process is illustrated in Figure 1.

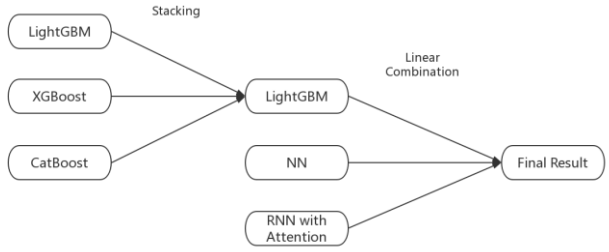

Figure 1: Ensemble Figure

Step 1: Features are fed into three GBDT models, LightGBM, Xgboost and Catboost. For the stacking strategy, we use a 5-fold stacking technique. We split the data into five folds and train five models using the leave-one-out strategy. The trained models are applied to the test data and generate one prediction for each model. In this way, on the training data, we obtain three different scores, generated respectively from LightGBM, Xgboost and Catboost models. In order to perform prediction on the test data, we average these 5 groups of predicted scores from each model.

Step 2: The outputs from the first stage will then be regarded as features and added into features set for the second stage learning, forming a stacking model learning process. The prediction scores are treated as new training data and the average of the prediction on the test data from the first stage are used as new test data. The second stage's results are generated using the new set of training and test with the LightGBM model.

Step 3: Based on the performance of deep learning models and stacking model, we try different kinds of model combination strategies and generate our final submission by a linear combination of these models, which can be represented as:

$$Score = 0.7 \times stacked\ \ GBDT\ \ model + 0.2 \times nn + 0.1 \times RNN\ \ with\ Attention$$

The weights are carefully tuned based on the performance of each model on leaderboard and offline validation set.

4 EVALUATION RESULTS

We build a solid offline evaluation system because of the limited chance to evaluate our model online. The scores on offline validation set are highly close to the scores on online leaderboard, whose difference is less than 0.001.

4.1 GBDT and Fully Connected Neural Network Model Results

Note that the proposed learning process contains a few techniques such as parameter optimization and feature selection. We may apply different settings to obtain the final results. In Table 4, we show different settings and their performance.

Table 4: GBDT and Fully Connected Neural Network Models and Their Scores on the Leaderboard

Model	AUC on Leaderboard
LGB single model with all features	0.7626
NN single model	0.758
LGB single model with feature selection validation	0.763
LGB single model with feature selection and parameter optimized validation	0.7636

4.2 Deep Model Results

Due to lack of time and computing resources, we did not try enough model parameters to achieve good results. In table 5, we show performances of different models.

Table 5: Deep Models and Their Scores on the Leaderboard

Model	AUC on Leaderboard
Bi-directional Recurrent Neural Network	0.7268
Bi-directional Recurrent Neural Network with self-Attention	0.7412
Multi-Head Attention Mechanism	0.7105
TextCNN	0.7014

We can observe that Bi-directional Recurrent Neural Network with self-Attention performs well on this task compared to other models.

4.3 Ensemble Model Results

Limited by the computing resources, we only perform some experiments. We apply ensemble methods in different models with different weight for linear combination. The results are shown in Table 6. Finally, we get the ensemble strategy mentioned above.

Table 6: Ensemble Models and Their Scores on the Leaderboard

Model	AUC on Leaderboard
GBDT and NN stacking model	0.7642
0.7 * stacking + 0.2 * NN + 0.1*RNN with attention (Final model)	0.7646
0.7 * stacking + 0.15 * NN + 0.15 * RNN with attention	0.7635
0.7 * stacking + 0.2 * NN + 0.075 RNN with attention + 0.025 * RNN	0.7640
0.4 * RNN with attention + 0.3 * RNN + 0.2 * multi-head attention + 0.1 * TextCNN	0.735
0.7 * stacking + 0.2 * NN + 0.1 * linear combination of DL model	0.7641

5 CONCLUSION

In this paper, we describe our approach to WSDM Cup's Retention Rate of Baidu Hao Kan APP Users Challenge. We apply ensemble method to five different deep learning models and Gradient Boosting Decision Trees. According to the results of our experiments, the most effective method to increase the accuracy of model is feature engineering, which increase the score of a single LightGBM model from 0.71 to 0.763 on the leaderboard. Although the performance of the deep learning models is not satisfactory, their scores are increased after the application of ensemble methods. As to our final result with an ensemble of two different type of models, we got an AUC score 0.7646 on the leaderboard.

There are still some methods we can try in the future to improve the performance of model. The hyper-parameters of GBDT models can optimized using bayes optimization. We can continuously analyze user behavior to extract more effective features that better represent the information of the raw user log data and perform well-designed feature selection method. Hundreds of models can be built by AutoML tool and added to our ensemble method, though this method consumes much computing resources. In addition, our deep learning models still has considerable room for improvement. So we will try to add additional feature engineering to deep learning models and adjust our models to make it more appropriate for the task.

REFERENCES

[1] Chen, T., & Guestrin, C. (2016, August). Xgboost: A scalable tree boosting system. In Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining (pp. 785-794). ACM.

[2] Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., ... & Liu, T. Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree. In Advances in Neural Information Processing Systems (pp. 3146-3154).

[3] Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V., & Gulin, A. (2018). CatBoost: unbiased boosting with categorical features. In Advances in Neural Information Processing Systems (pp. 6639-6649).

[4] Kim, Y. (2014). Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882*.

[5] Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

[6] Bahdanau, D., Cho, K., & Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

[7] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems* (pp. 5998-6008).

[8] Covington, P., Adams, J., & Sargin, E. (2016, September). Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems* (pp. 191-198). ACM.