EFFICIENT CLICK-THROUGH RATE PREDICTION FOR DEVELOPING COUNTRIES VIA TABULAR LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Despite the rapid growth of online advertisement in developing countries, existing highly over-parameterized Click-Through Rate (CTR) prediction models are difficult to be deployed due to the limited computing resources. In this paper, by bridging the relationship between CTR prediction task and tabular learning, we present that tabular learning models are more efficient and effective in CTR prediction than over-parameterized CTR prediction models. Extensive experiments on eight public CTR prediction datasets show that tabular learning models outperform twelve state-of-the-art CTR prediction models. Furthermore, compared to over-parameterized CTR prediction models, tabular learning models can be fast trained without expensive computing resources including high-performance GPUs. Finally, through an A/B test on an *actual* online application, we show that tabular learning models improve not only offline performance but also the CTR of real users.

1 INTRODUCTION

With the spread of mobile devices, the e-commerce market in developing countries is rapidly growing. For example, the Indian e-commerce market is expected to grow to US$ 200 billion by 2026 from US$ 38.5 billion as of 2017[1]. Accordingly, Click-Through Rate (CTR) prediction has become more important in online advertisements in developing countries as well as developed countries. Regardless of industry and academia, highly over-parameterized CTR prediction models (Lian et al., 2018; Song et al., 2019; Cheng et al., 2020) have recently been proposed to improve performance using deep neural networks. These CTR prediction models should train a considerable amount of parameters to deal with millions of input features. Therefore, massive computing resources, including high-performance GPUs, are required to train the over-parameterized CTR prediction models.

However, these CTR prediction models are hard to be applied to real-world applications under limited computing resource scenarios of developing countries. First, it is well known that real-world applications have a *stale problem* that the performance of their CTR prediction models is severely degraded over time by many reasons including change of user/item pools (Nasraoui et al., 2007; Radinsky et al., 2012; Trevisiol et al., 2014) (See Figure 2). Second, hyper-parameter tuning is essential for over-parameterized CTR prediction models because they tend to be sensitive to hyper-parameters (Cheng et al., 2020). To alleviate these issues, solutions using daily training (Koren, 2009; Vartak et al., 2017; Wang et al., 2019), meta-learning (Lee et al., 2019; Chen et al., 2019), and reinforcement learning (Jagerman et al., 2019; Shi et al., 2019) have been suggested. However, since these solutions require sufficient computing resources, they are hard to be applied to real-world applications in developing countries.

In this paper, we bridge the relationship between CTR prediction and tabular learning (Chen et al., 2015; Ke et al., 2017; Dorogush et al., 2018) to find a practical CTR prediction model that can be easily deployed in developing countries. In CTR prediction, conventional tabular learning models (Ke et al., 2017; Dorogush et al., 2018) have been neglected as baselines because input features in CTR prediction are mainly composed of highly sparse categorical features and it is difficult to be trained by the tabular learning models (Ke et al., 2019). Meanwhile, the recently proposed methods (Ke et al., 2017; Dorogush et al., 2018; Ayria, 2020) handling categorical features in tabular learning

[1]Reported by India Brand Equity Foundation (Nov 2020)

have dramatically improved the performance by slightly modifying primitive categorical feature encoders (Fisher, 1958; Micci-Barreca, 2001). Borrowing tabular learning models with these methods to CTR prediction, we demonstrate that tabular learning models outperform existing CTR prediction models accompanying their cost-efficiency.

Our contributions. (1) We explicitly investigate the relationship between CTR prediction task and tabular learning, and suggest that tabular learning models could act as efficient baselines in CTR prediction. (2) Extensive experiments show that tabular learning models outperform current CTR prediction models, accompanying their cost-efficiency. We believe that our extensive experiment results of tabular learning models in CTR prediction can contribute to research communities related to CTR prediction. (3) Finally, our online experiments including the A/B test on an active online application validate the effectiveness of tabular learning methods under limited computing resources.

2 Related Works

2.1 Click-Through Rate Prediction

CTR prediction is to predict the probability of the user u clicking on the item v. The major difference with collaborative filtering (Rendle et al., 2012) is that CTR prediction utilizes additional side information about users and items as input features containing highly sparse categorical features. Factorization Machines (FM) (Rendle, 2010) is the most representative CTR prediction model which considers the first and second-order feature interactions from input features, simultaneously. Recently, various CTR prediction models have been proposed to capture the high-order feature interactions via deep neural networks Cheng et al. (2016); He & Chua (2017); Guo et al. (2017); Lian et al. (2018); Qu et al. (2018); Song et al. (2019); Cheng et al. (2020). However, these CTR prediction models are too over-parameterized to be deployed under limited computing resource scenarios.

2.2 Tabular Learning

Tabular learning refers to a learning methodology handling *tabular heterogeneous data* as input. It is known that gradient boosting models based on decision trees (Chen et al., 2015; Ke et al., 2017; Dorogush et al., 2018) show superior performance in tabular learning (Harasymiv, 2015). There have been many attempts to improve the performance of gradient boosting models using deep networks. However, they only achieve similar performance to gradient boosting models although they utilize a large number of computing resources (Miller et al., 2017; Zhou & Feng, 2017; Ke et al., 2018; Lay et al., 2018; Yang et al., 2018; Feng et al., 2018). Note that we describe the remainder of this paper using gradient boosting models as representative models of tabular learning since they show better performance than deep neural network-based models.

3 Efficient Click-Through Rate Prediction

Although tabular learning and CTR prediction have developed orthogonal to each other so far, they share similar formulation of problem definition.

$$\arg\min_{f} \sum_{i} \mathcal{L}(y^i, f(x^i_1, \cdots, x^i_n)) \tag{1}$$

It is to find the function f where minimizes the sum of difference, which is measured by a loss function $\mathcal{L}$ over all the i-th data instances between actual target y^i and the output value of function f with given input x^i. The major difference between tabular learning and CTR prediction is that each x is often assumed to be the set of heterogeneous *numerical* features in tubular learning while it is assumed to be the set of highly sparse categorical features in CTR prediction. In CTR prediction, for instance, x_k ($k = 1, \cdots, n$) can be the categorical index of the user u or item v, the gender or age group of users, and the item category.

Conventional gradient boosting is not capable of handling categorical features. To take the edge off this problem, several methods have been proposed to convert categorical features into numerical features such as one-hot encoding, Label Encoding (LE), and Target Encoding (TE), but they are not

Table 1: Evaluation results of three tabular learning models and twelve CTR prediction models on eight real-world datasets. Logloss and AUROC with 95% confidence interval of 10-runs is provided.

	Model	KDD12	Criteo	Avazu	Talking Data	Amazon	Movielens	Book Crossing	Frappe
Logloss	XGBoost	0.1588 ± 0.0001	0.4595 ± 0.0050	0.3901 ± 0.0014	0.1327 ± 0.0008	0.4947 ± 0.0003	0.2831 ± 0.0011	0.5141 ± 0.0001	0.2849 ± 0.0019
	LightGBM	0.1602 ± 0.0000	0.4569 ± 0.0003	0.3916 ± 0.0003	0.1319 ± 0.0003	0.5627 ± 0.0000	0.2437 ± 0.0014	0.5191 ± 0.0006	0.1176 ± 0.0022
	CatBoost	**0.1584 ± 0.0001**	**0.4507 ± 0.0002**	**0.3840 ± 0.0001**	**0.1284 ± 0.0001**	**0.2221 ± 0.0004**	**0.1192 ± 0.0003**	**0.4962 ± 0.0001**	**0.0780 ± 0.0005**
	FM	0.1595 ± 0.0001	0.4575 ± 0.0002	0.3912 ± 0.0004	0.1342 ± 0.0008	0.5257 ± 0.0036	0.2783 ± 0.0027	0.5224 ± 0.0009	0.2125 ± 0.0053
	FFM	0.1599 ± 0.0001	0.4522 ± 0.0001	0.3899 ± 0.0002	0.1347 ± 0.0003	0.4780 ± 0.0007	0.2414 ± 0.0040	0.5143 ± 0.0008	0.1651 ± 0.0020
	AFM	0.1607 ± 0.0005	0.4605 ± 0.0005	0.3941 ± 0.0002	0.1389 ± 0.0020	0.5498 ± 0.0063	0.2714 ± 0.0112	0.5229 ± 0.0045	0.2648 ± 0.0051
	DCN	0.1599 ± 0.0001	0.4596 ± 0.0002	0.3926 ± 0.0003	0.1351 ± 0.0005	0.4304 ± 0.0021	0.2897 ± 0.0006	0.5226 ± 0.0011	0.2400 ± 0.0066
	NFM	0.1626 ± 0.0015	0.4568 ± 0.0005	0.3910 ± 0.0005	0.1338 ± 0.0004	0.4922 ± 0.0153	0.2862 ± 0.0243	0.5256 ± 0.0016	0.1418 ± 0.0074
	MLP	0.1593 ± 0.0001	0.4568 ± 0.0003	0.3923 ± 0.0006	0.1334 ± 0.0008	0.4228 ± 0.0018	0.2829 ± 0.0068	0.5233 ± 0.0012	0.2375 ± 0.0152
	Wide & Deep	0.1593 ± 0.0001	0.4567 ± 0.0003	0.3921 ± 0.0005	0.1335 ± 0.0009	0.4273 ± 0.0013	0.2833 ± 0.0044	0.5232 ± 0.0011	0.2387 ± 0.0274
	DeepFM	0.1602 ± 0.0001	0.4586 ± 0.0004	0.3920 ± 0.0004	0.1343 ± 0.0006	0.4317 ± 0.0024	0.2889 ± 0.0020	0.5219 ± 0.0019	0.2244 ± 0.0102
	xDeepFM	0.1603 ± 0.0002	0.4589 ± 0.0004	0.3926 ± 0.0004	0.1344 ± 0.0006	0.4346 ± 0.0017	0.2883 ± 0.0012	0.5223 ± 0.0018	0.2214 ± 0.0142
	PNN	0.1604 ± 0.0001	0.4573 ± 0.0003	0.3927 ± 0.0003	0.1357 ± 0.0005	0.4355 ± 0.0028	0.2902 ± 0.0013	0.5225 ± 0.0007	0.1957 ± 0.0057
	AutoInt	0.1611 ± 0.0001	0.4615 ± 0.0002	0.3953 ± 0.0006	0.1409 ± 0.0014	0.4556 ± 0.0043	0.2906 ± 0.0011	0.5236 ± 0.0014	0.2736 ± 0.0233
	AFN	0.1598 ± 0.0002	0.4552 ± 0.0006	0.3912 ± 0.0009	0.1327 ± 0.0005	0.4274 ± 0.0016	0.2768 ± 0.0036	0.5259 ± 0.0024	0.1853 ± 0.0046
AUROC	XGBoost	0.7646 ± 0.0003	0.7889 ± 0.0061	0.7613 ± 0.0028	0.9719 ± 0.0003	0.7395 ± 0.0005	0.9380 ± 0.0006	0.8019 ± 0.0002	0.9353 ± 0.0007
	LightGBM	0.7572 ± 0.0003	0.7922 ± 0.0003	0.7584 ± 0.0006	0.9724 ± 0.0001	0.4946 ± 0.0009	0.9483 ± 0.0008	0.7951 ± 0.0006	0.9855 ± 0.0005
	CatBoost	**0.7655 ± 0.0003**	**0.7993 ± 0.0002**	**0.7734 ± 0.0002**	**0.9736 ± 0.0001**	**0.9598 ± 0.0001**	**0.9848 ± 0.0001**	**0.8183 ± 0.0001**	**0.9925 ± 0.0002**
	FM	0.7618 ± 0.0006	0.7922 ± 0.0003	0.7600 ± 0.0008	0.9710 ± 0.0003	0.6823 ± 0.0118	0.9375 ± 0.0014	0.7945 ± 0.0009	0.9659 ± 0.0012
	FFM	0.7610 ± 0.0004	0.7979 ± 0.0001	0.7622 ± 0.0005	0.9707 ± 0.0002	0.7592 ± 0.0008	0.9529 ± 0.0015	0.8014 ± 0.0007	0.9752 ± 0.0004
	AFM	0.7571 ± 0.0029	0.7883 ± 0.0006	0.7541 ± 0.0005	0.9689 ± 0.0010	0.6013 ± 0.0334	0.9423 ± 0.0077	0.7938 ± 0.0037	0.9544 ± 0.0014
	DCN	0.7602 ± 0.0006	0.7895 ± 0.0005	0.7569 ± 0.0006	0.9700 ± 0.0004	0.8146 ± 0.0018	0.9348 ± 0.0004	0.7929 ± 0.0008	0.9669 ± 0.0014
	NFM	0.7576 ± 0.0014	0.7929 ± 0.0007	0.7600 ± 0.0006	0.9712 ± 0.0004	0.7402 ± 0.0212	0.9441 ± 0.0047	0.7917 ± 0.0014	0.9821 ± 0.0011
	MLP	0.7627 ± 0.0005	0.7929 ± 0.0005	0.7574 ± 0.0012	0.9717 ± 0.0003	0.8233 ± 0.0013	0.9413 ± 0.0036	0.7926 ± 0.0012	0.9628 ± 0.0055
	Wide & Deep	0.7626 ± 0.0004	0.7930 ± 0.0005	0.7580 ± 0.0005	0.9715 ± 0.0010	0.8184 ± 0.0009	0.9407 ± 0.0033	0.7932 ± 0.0025	0.9648 ± 0.0034
	DeepFM	0.7585 ± 0.0005	0.7908 ± 0.0006	0.7581 ± 0.0006	0.9705 ± 0.0006	0.8139 ± 0.0024	0.9354 ± 0.0018	0.7944 ± 0.0008	0.9630 ± 0.0035
	xDeepFM	0.7580 ± 0.0007	0.7904 ± 0.0005	0.7572 ± 0.0006	0.9705 ± 0.0004	0.8101 ± 0.0025	0.9366 ± 0.0019	0.7941 ± 0.0013	0.9686 ± 0.0038
	PNN	0.7584 ± 0.0007	0.7927 ± 0.0006	0.7569 ± 0.0004	0.9702 ± 0.0002	0.8094 ± 0.0029	0.9361 ± 0.0013	0.7934 ± 0.0008	0.9742 ± 0.0015
	AutoInt	0.7559 ± 0.0018	0.7872 ± 0.0002	0.7516 ± 0.0010	0.9682 ± 0.0006	0.7873 ± 0.0055	0.9361 ± 0.0004	0.7927 ± 0.0009	0.9427 ± 0.0117
	AFN	0.7598 ± 0.0016	0.7947 ± 0.0007	0.7600 ± 0.0015	0.9717 ± 0.0005	0.8173 ± 0.0017	0.9430 ± 0.0016	0.7921 ± 0.0025	0.9760 ± 0.0018

suitable for the CTR prediction task. One-hot encoding is not practical because the input dimension becomes too large when we apply one-hot encoding to highly sparse categorical features (having extremely high cardinality). LE, which converts a categorical feature into an arbitrary number, would show sub-optimal performance since there are few correlations between the target category and its encoded number. Meanwhile, TE changes the categorical feature into an informative number by calculating the mean of target values with each categorical feature (Micci-Barreca, 2001). However, TE causes overfitting by giving excessive information on each categorical feature (Schifferer et al., 2020).

Recently, there have been new attempts to process categorical features in tabular learning. Ke et al. (2017) adopts Fisher (1958) to find the optimal split over categorical features. CatBoost Encoding (Dorogush et al., 2018) is a variant of TE preventing overfitting by random permutation. In addition, K-Fold Target Encoding (Ayria, 2020) is intended to increase generality of TE through the K-fold validation. Schifferer et al. (2020) won a competition by applying the K-fold TE to XGBoost (Chen et al., 2015).

Recent advances in encoding methods of categorical features make it possible for tabular learning to be used for CTR prediction although they do not focus on categorical features with tremendously high cardinality. Nevertheless, most CTR prediction models preclude gradient boosting as a baseline (Qu et al., 2018; Song et al., 2019; Cheng et al., 2020). He et al. (2014); Juan et al. (2016) employ gradient boosting as not a baseline but a feature pre-processing method. Only a few studies (Ke et al., 2019) have compared gradient boosting to their model. However, they do not take advantage of the recent advance in encoding methods to deal with categorical features.

Consequently, we suggest gradient boosting models with the recent advance in handling categorical features as the baselines of the CTR prediction task. In Section 4, we will show that gradient boosting not only can be trained at low cost, but also shows better performance than existing CTR prediction models. These results demonstrate that gradient boosting is suitable for use in developing countries with limited resources. Noticeably, it is known that considering high order interaction between features is important in the CTR prediction task and gradient boosting also has the capability of modeling high order interaction by depth (> 1) of the decision tree.

4 EXPERIMENTS

In this paper, we compare gradient boosting models with existing CTR prediction models. In Section 4.1, we assess the performance of each model on CTR prediction benchmark datasets. In Section 4.2, we conduct experiments to show the cost-efficiency of gradient boosting models. In

Table 2: Ablation study results of three tabular learning models regarding to encoding methods of categorical features. Logloss and AUROC with 95% confidence interval of 10-runs is provided.

	Model	Encoding	KDD12	Criteo	Avazu	Talking Data	Amazon	Movielens	Book Crossing	Frappe
Logloss	XGBoost	LE	0.1624 ± 0.0002	**0.4613 ± 0.0009**	0.3930 ± 0.0007	**0.1332 ± 0.0007**	0.5532 ± 0.0010	0.3125 ± 0.0012	0.5415 ± 0.0033	**0.1805 ± 0.0098**
		TE	0.1723 ± 0.0010	0.4726 ± 0.0008	0.5194 ± 0.0025	0.1543 ± 0.0023	0.6047 ± 0.0004	0.3619 ± 0.0014	0.8306 ± 0.0006	0.3059 ± 0.0011
		-	**0.1588 ± 0.0001**	**0.4595 ± 0.0050**	**0.3901 ± 0.0014**	**0.1327 ± 0.0008**	**0.4947 ± 0.0003**	**0.2831 ± 0.0011**	**0.5141 ± 0.0001**	0.2849 ± 0.0019
	LightGBM	LE	0.1617 ± 0.0004	0.4628 ± 0.0010	**0.3913 ± 0.0005**	**0.1302 ± 0.0002**	**0.5151 ± 0.0006**	0.4609 ± 0.0005	0.5383 ± 0.0049	0.2576 ± 0.0068
		TE	0.1656 ± 0.0001	0.4710 ± 0.0001	0.4260 ± 0.0001	0.1439 ± 0.0003	0.5661 ± 0.0001	0.3226 ± 0.0002	0.6013 ± 0.0004	0.2942 ± 0.0003
		-	**0.1602 ± 0.0000**	**0.4569 ± 0.0003**	**0.3916 ± 0.0003**	0.1319 ± 0.0003	0.5627 ± 0.0000	**0.2437 ± 0.0014**	**0.5191 ± 0.0006**	**0.1176 ± 0.0022**
	CatBoost	LE	0.1622 ± 0.0001	0.4656 ± 0.0004	0.3924 ± 0.0001	0.1303 ± 0.0001	0.5232 ± 0.0003	0.4776 ± 0.0009	0.5507 ± 0.0002	0.2888 ± 0.0014
		TE	0.1683 ± 0.0032	0.4654 ± 0.0005	0.4774 ± 0.0226	0.1447 ± 0.0011	0.5943 ± 0.0024	0.3209 ± 0.0100	0.6252 ± 0.0143	0.2795 ± 0.0021
		-	**0.1584 ± 0.0001**	**0.4507 ± 0.0002**	**0.3840 ± 0.0001**	**0.1284 ± 0.0001**	**0.2221 ± 0.0004**	**0.1192 ± 0.0003**	**0.4962 ± 0.0001**	**0.0780 ± 0.0005**
AUROC	XGBoost	LE	0.7422 ± 0.0013	**0.7869 ± 0.0011**	0.7564 ± 0.0013	**0.9718 ± 0.0003**	0.5833 ± 0.0042	0.9241 ± 0.0006	0.7743 ± 0.0032	**0.9721 ± 0.0025**
		TE	0.7258 ± 0.0012	0.7768 ± 0.0009	0.6960 ± 0.0004	0.9572 ± 0.0021	0.3054 ± 0.0032	0.9043 ± 0.0008	0.7029 ± 0.0010	0.9295 ± 0.0005
		-	**0.7646 ± 0.0003**	**0.7889 ± 0.0061**	**0.7613 ± 0.0028**	**0.9719 ± 0.0003**	**0.7395 ± 0.0005**	**0.9380 ± 0.0006**	**0.8019 ± 0.0002**	0.9353 ± 0.0007
	LightGBM	LE	0.7468 ± 0.0021	0.7852 ± 0.0013	**0.7599 ± 0.0009**	**0.9730 ± 0.0001**	**0.6892 ± 0.0012**	0.8324 ± 0.0005	0.7793 ± 0.0054	0.9514 ± 0.0026
		TE	0.7165 ± 0.0007	0.7767 ± 0.0006	0.6939 ± 0.0003	0.9652 ± 0.0003	0.2908 ± 0.0092	0.9091 ± 0.0001	0.6967 ± 0.0059	0.9290 ± 0.0003
		-	**0.7572 ± 0.0003**	**0.7922 ± 0.0003**	**0.7584 ± 0.0006**	0.9724 ± 0.0001	0.4946 ± 0.0009	**0.9483 ± 0.0008**	**0.7951 ± 0.0006**	**0.9855 ± 0.0005**
	CatBoost	LE	0.7436 ± 0.0003	0.7816 ± 0.0005	0.7579 ± 0.0003	0.9729 ± 0.0001	0.6711 ± 0.0009	0.8164 ± 0.0008	0.7641 ± 0.0004	0.9360 ± 0.0006
		TE	0.7300 ± 0.0065	0.7867 ± 0.0012	0.6822 ± 0.0179	0.9654 ± 0.0005	0.2798 ± 0.0069	0.9086 ± 0.0013	0.7047 ± 0.0064	0.9395 ± 0.0008
		-	**0.7655 ± 0.0003**	**0.7993 ± 0.0002**	**0.7734 ± 0.0002**	**0.9736 ± 0.0001**	**0.9598 ± 0.0001**	**0.9848 ± 0.0001**	**0.8183 ± 0.0001**	**0.9925 ± 0.0002**

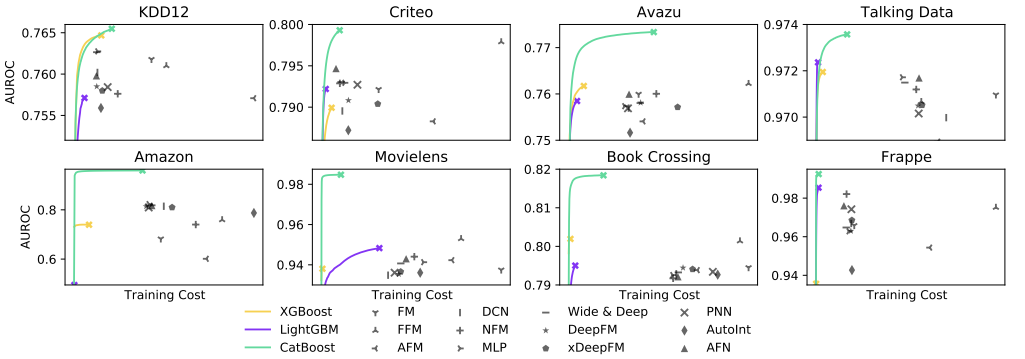

Figure 1: AUROC by training cost estimated on AWS EC2 instances.

Section 4.3, we verify how much recent categorical feature encoding methods contribute to the performance through ablation study. In Section 4.4, we demonstrate the performance of gradient boosting models on online experiments and the possibility to alleviate the stale problem at a low cost.

Datasets. To assess the performance in CTR prediction, we conduct experiments on the following eight public CTR prediction datasets: KDD12, Criteo, Avazu, Data, Movielens (Harper & Konstan, 2015), Book-Crossing (Ziegler et al., 2005), and Frappe. We arbitrarily split each dataset into the train, valid, and test sets in a ratio of 8:1:1[2].

Baseline Models. Three gradient boosting models and twelve CTR prediction models are considered to justify the efficiency and effectiveness of gradient boosting models in CTR prediction. XGBoost (Chen et al., 2015), LightGBM (Ke et al., 2017) and CatBoost (Dorogush et al., 2018) are considered for gradient boosting models. Since the originally proposed XGBoost uses LE, K-fold TE (Ayria, 2020) is applied to XGBoost following Schifferer et al. (2020). Considered CTR prediction models are as follows: FM (Rendle, 2010), FFM (Juan et al., 2016), AFM (Xiao et al., 2017), DCN (Wang et al., 2017), MLP, NFM (He & Chua, 2017), Wide & Deep (Cheng et al., 2016), DeepFM (Guo et al., 2017), xDeepFM (Lian et al., 2018), PNN (Qu et al., 2018), AutoInt (Song et al., 2019), and AFN (Cheng et al., 2020). In developing countries, since it is difficult to perform hyper-parameter tuning every day, we do not newly tune hyper-parameters and do our best to keep originally hyper-parameters reported in each paper.

4.1 Performance Comparison

The evaluation results are summarized in Table 1. CatBoost, which is one of the gradient boosting models, outperforms all the baseline models on all the datasets with a large margin while a slight

[2]For KDD12, Criteo, Avazu, and Talking Data, 10% random sampling is used because they are too large not to be suitable for our extensive experimentation

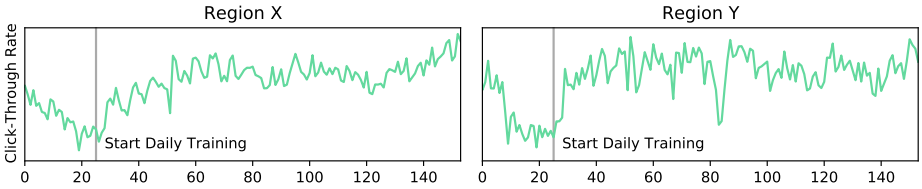

Figure 2: Alleviating stale problem by daily training with CatBoost. CTR over time after first model deployment on two main regions X and Y of our application is plotted.

increase in AUROC or decrease in Logloss at *.001-level* is known as a significant improvement in CTR prediction as pointed out in previous works (Cheng et al., 2016; Guo et al., 2017; Song et al., 2019). In addition, the other gradient boosting methods (XGBoost and LightGBM) achieve better or comparable performance to CTR prediction models over all the benchmark datasets.

4.2 Efficiency Comparison

In each dataset, the efficiency of gradient boosting models is validated by plotting the change of AUROC according to the training cost [3]. Training costs are estimated on AWS EC2 instances. `c5a.4xlarge` is used for gradient boosting models because they do not need GPUs. `p3.2xlarge` is used for CTR prediction models because they require GPUs. For gradient boosting models, training cost and AUROC in every ten epochs are also reported. Boosting models shows dramatic improvement in the performance at the early stages of training.

4.3 Ablation Study

By replacing the latest categorical feature encoding methods with LE and TE, Table 2 shows that how much the leverage of recent advance of the categorical encoding methods contributes to performance improvement. Mostly, recent categorical feature encoding methods show statistically significant better results.

4.4 Online Experiments

Based on the offline test in Section 4.1, CatBoost is adopted for deploying to our online application downloaded more than 10M times. Table 3 shows CTR gain by CatBoost relative to the control group (heuristic algorithm) in online A/B test for seven days on two main regions (Region X and Y). CatBoost outperforms the existing heuristic algorithm with a large margin. Not only that, after we deployed our first model, we experienced a stale problem that performance continued to decline. To solve this problem, we started daily training, and as a result, we were able to solve the stale problem (See Figure 2).

Table 3: CTR gain of CatBoost model in online A/B test.

	CTR Gain
Region X	+ 59.47%
Region Y	+ 84.96%

5 Conclusion

We suggest tabular learning models as CTR prediction for developing countries by explicitly shedding some light on the relationship between tabular learning and CTR prediction task. The state-of-the-art performance on eight public datasets and better results on online A/B test can be achieved at a low cost with tabular learning models (especially gradient boosting) and the recent advance in categorical feature encoding methods. In addition, our study provides room for improvement of applications in developing countries under limited computing resources by showing that the state-of-the-art performance is achieved with tabular learning models with the recent advance in categorical feature encoding methods.

[3] Although we only report AUROC, Logloss shows similar trends.

REFERENCES

Avazu. Avazu dataset. https://www.kaggle.com/c/avazu-ctr-prediction. [Online; accessed 06-March-2020].

Pourya Ayria. K-fold target encoding. https://medium.com/@pouryaayria/k-fold-target-encoding-dfe9a594874b, 2020. [Online; accessed 06-March-2020].

Tianqi Chen, Tong He, Michael Benesty, Vadim Khotilovich, Yuan Tang, Hyunsu Cho, et al. Xgboost: extreme gradient boosting. *R package version 0.4-2*, 1(4), 2015.

Yihong Chen, Bei Chen, Xiangnan He, Chen Gao, Yong Li, Jian-Guang Lou, and Yue Wang. λopt: Learn to regularize recommender models in finer levels. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 978–986, 2019.

Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. Wide & deep learning for recommender systems. In *Proceedings of the 1st workshop on deep learning for recommender systems*, pp. 7–10, 2016.

Weiyu Cheng, Yanyan Shen, and Linpeng Huang. Adaptive factorization network: Learning adaptive-order feature interactions. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, 2020*, pp. 3609–3616. AAAI Press, 2020. URL https://aaai.org/ojs/index.php/AAAI/article/view/5768.

Criteo. Criteo dataset. https://www.kaggle.com/c/criteo-display-ad-challenge. [Online; accessed 06-March-2020].

Talking Data. Talking data dataset. https://www.kaggle.com/c/talkingdata-adtracking-fraud-detection. [Online; accessed 06-March-2020].

Anna Veronika Dorogush, Vasily Ershov, and Andrey Gulin. Catboost: gradient boosting with categorical features support. *arXiv preprint arXiv:1810.11363*, 2018.

Ji Feng, Yang Yu, and Zhi-Hua Zhou. Multi-layered gradient boosting decision trees. *arXiv preprint arXiv:1806.00007*, 2018.

Walter D Fisher. On grouping for maximum homogeneity. *Journal of the American statistical Association*, 53(284):789–798, 1958.

Frappe. Frappe dataset. https://www.baltrunas.info/context-aware. [Online; accessed 06-March-2020].

Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. Deepfm: A factorization-machine based neural network for ctr prediction. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, IJCAI'17, pp. 1725–1731. AAAI Press, 2017. ISBN 9780999241103.

Vasyl Harasymiv. Lessons from 2 million machine learning models on kaggle. https://www.kdnuggets.com/2015/12/harasymiv-lessons-kaggle-machine-learning.html, 2015. [Online; accessed 06-March-2020].

F Maxwell Harper and Joseph A Konstan. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, 5(4):1–19, 2015.

Xiangnan He and Tat-Seng Chua. Neural factorization machines for sparse predictive analytics. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*, pp. 355–364, 2017.

Xinran He, Junfeng Pan, Ou Jin, Tianbing Xu, Bo Liu, Tao Xu, Yanxin Shi, Antoine Atallah, Ralf Herbrich, Stuart Bowers, et al. Practical lessons from predicting clicks on ads at facebook. In *Proceedings of the Eighth International Workshop on Data Mining for Online Advertising*, pp. 1–9, 2014.

Rolf Jagerman, Ilya Markov, and Maarten de Rijke. When people change their mind: Off-policy evaluation in non-stationary recommendation environments. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pp. 447–455, 2019.

Yuchin Juan, Yong Zhuang, Wei-Sheng Chin, and Chih-Jen Lin. Field-aware factorization machines for ctr prediction. In *Proceedings of the 10th ACM Conference on Recommender Systems*, pp. 43–50, 2016.

KDD12. Kdd cup 2012, track 2 dataset. `https://www.kaggle.com/c/kddcup2012-track2`. [Online; accessed 06-March-2020].

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. *Advances in neural information processing systems*, 30:3146–3154, 2017.

Guolin Ke, Jia Zhang, Zhenhui Xu, Jiang Bian, and Tie-Yan Liu. Tabnn: A universal neural network solution for tabular data. 2018.

Guolin Ke, Zhenhui Xu, Jia Zhang, Jiang Bian, and Tie-Yan Liu. Deepgbm: A deep learning framework distilled by gbdt for online prediction tasks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 384–394, 2019.

Yehuda Koren. Collaborative filtering with temporal dynamics. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 447–456, 2009.

Nathan Lay, Adam P Harrison, Sharon Schreiber, Gitesh Dawer, and Adrian Barbu. Random hinge forest for differentiable learning. *arXiv preprint arXiv:1802.03882*, 2018.

Hoyeop Lee, Jinbae Im, Seongwon Jang, Hyunsouk Cho, and Sehee Chung. Melu: meta-learned user preference estimator for cold-start recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1073–1082, 2019.

Jianxun Lian, Xiaohuan Zhou, Fuzheng Zhang, Zhongxia Chen, Xing Xie, and Guangzhong Sun. xdeepfm: Combining explicit and implicit feature interactions for recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1754–1763, 2018.

Daniele Micci-Barreca. A preprocessing scheme for high-cardinality categorical attributes in classification and prediction problems. *ACM SIGKDD Explorations Newsletter*, 3(1):27–32, 2001.

Kevin Miller, Chris Hettinger, Jeffrey Humpherys, Tyler Jarvis, and David Kartchner. Forward thinking: Building deep random forests. *arXiv preprint arXiv:1705.07366*, 2017.

Olfa Nasraoui, Jeff Cerwinske, Carlos Rojas, and Fabio Gonzalez. Performance of recommendation systems in dynamic streaming environments. In *Proceedings of the 2007 SIAM International Conference on Data Mining*, pp. 569–574. SIAM, 2007.

Yanru Qu, Bohui Fang, Weinan Zhang, Ruiming Tang, Minzhe Niu, Huifeng Guo, Yong Yu, and Xiuqiang He. Product-based neural networks for user response prediction over multi-field categorical data. *ACM Transactions on Information Systems (TOIS)*, 37(1):1–35, 2018.

Kira Radinsky, Krysta Svore, Susan Dumais, Jaime Teevan, Alex Bocharov, and Eric Horvitz. Modeling and predicting behavioral dynamics on the web. In *Proceedings of the 21st international conference on World Wide Web*, pp. 599–608, 2012.

Steffen Rendle. Factorization machines. In *2010 IEEE International Conference on Data Mining*, pp. 995–1000. IEEE, 2010.

Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. *arXiv preprint arXiv:1205.2618*, 2012.

Benedikt Schifferer, Gilberto Titericz, Chris Deotte, Christof Henkel, Kazuki Onodera, Jiwei Liu, Bojan Tunguz, Even Oldridge, Gabriel De Souza Pereira Moreira, and Ahmet Erdem. Gpu accelerated feature engineering and training for recommender systems. In *Proceedings of the Recommender Systems Challenge 2020*, pp. 16–23. 2020.

Bichen Shi, Makbule Gulcin Ozsoy, Neil Hurley, Barry Smyth, Elias Z Tragos, James Geraci, and Aonghus Lawlor. Pyrecgym: a reinforcement learning gym for recommender systems. In *Proceedings of the 13th ACM Conference on Recommender Systems*, pp. 491–495, 2019.

Weiping Song, Chence Shi, Zhiping Xiao, Zhijian Duan, Yewen Xu, Ming Zhang, and Jian Tang. Autoint: Automatic feature interaction learning via self-attentive neural networks. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pp. 1161–1170, 2019.

Michele Trevisiol, Luca Maria Aiello, Rossano Schifanella, and Alejandro Jaimes. Cold-start news recommendation with domain-dependent browse graph. In *Proceedings of the 8th ACM Conference on Recommender systems*, pp. 81–88, 2014.

Manasi Vartak, Arvind Thiagarajan, Conrado Miranda, Jeshua Bratman, and Hugo Larochelle. A meta-learning perspective on cold-start recommendations for items. 2017.

Renzhong Wang, Dragomir Yankov, Michael R Evans, Senthil Palanisamy, Siddhartha Arora, and Wei Wu. Predicting user routines with masked dilated convolutions. In *Proceedings of the 13th ACM Conference on Recommender Systems*, pp. 481–485, 2019.

Ruoxi Wang, Bin Fu, Gang Fu, and Mingliang Wang. Deep & cross network for ad click predictions. In *Proceedings of the ADKDD'17*, pp. 1–7. 2017.

Jun Xiao, Hao Ye, Xiangnan He, Hanwang Zhang, Fei Wu, and Tat-Seng Chua. Attentional factorization machines: learning the weight of feature interactions via attention networks. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pp. 3119–3125, 2017.

Yongxin Yang, Irene Garcia Morillo, and Timothy M Hospedales. Deep neural decision trees. *arXiv preprint arXiv:1806.06988*, 2018.

Zhi-Hua Zhou and Ji Feng. Deep forest. *arXiv preprint arXiv:1702.08835*, 2017.

Cai-Nicolas Ziegler, Sean M McNee, Joseph A Konstan, and Georg Lausen. Improving recommendation lists through topic diversification. In *Proceedings of the 14th international conference on World Wide Web*, pp. 22–32, 2005.