COMPGW02- WEB ECONOMICS Group Report

Bangwang Liu
University College London
School of Computer Science
uczlbwl@ucl.ac.uk

Jihong Deng
University College London
School of Computer Science
ucabjd3@ucl.ac.uk

Leila Illanes Mayta
University College London
School of Computer Science
uczllil@ucl.ac.uk

1. INTRODUCTION

Real-time bidding, different from the conventional negotiation or key words matching system, provides a new approach for selling and buying online display advertising for each individual impression in real time based on user data. From the position of Demand-Side Platforms (DSPs), equally advertisers, the contextual and user behaviours information are explored to investigate the bidding strategy by computational algorithms, especially with the machine learning training models.

For this study, it was provided three datasets of impressions for training, validation and testing. The data represent winning auctions of a Real-Time Bidding (RTB) environment where 9 advertisers competed for online advertisement. This data was analyzed using basic methods, machine learning and deep learning algorithms. Based on that analysis, the aim of this study is to formulate a bidding strategy to help advertisers to place ads in a real time bidding system, having as constraint a limited budget. The code and report are upload at the Github [5].

2. KEYWORDS

Demand-Side Platforms (DSPs), Real-Time Bidding (RTB), Gradient Boosting Decision Trees (GBDT), XGBoost, Convolutional Neural Network (CNN).

3. BIDDING STRATEGIES

For the following analysis, some definition is listed:

-Total cost = $\sum payprice$

- CTR = $\frac{\text{total number of clicks}}{\text{total number of impressions}}$,

- Cost-Per-Mile (CPM) = $\frac{\text{total cost}}{\text{total number of impressions}} * 1000$,

- Effective cost-per-click(eCPC) = $\frac{total\ cost}{total\ numbers\ of\ clicks}$.

3.1 Basic Bidding Strategies

3.1.1 Constant Bidding

The initial exploration of bidding price estimation is constant bidding which refers to choosing one number as bidding price to compete all the auctions. Since the maximum bidding price in both validation is no more than 300 CNY fen, the pre define bidding price should be ranged 0 to 300. After taking each bid into the all impressions, the bid which wins the largest clicks will be considered as optimal constant bidding. The best bids in training dataset and validation dataset are 25 and 77 , with clicks 134 and 68 respectively.

3.1.2 Random bidding

To further explore the bidding price estimation, random bidding has been applied. It basically presets a proper range, from which the bid will be randomly selected in every auction. Similarly, after a set of loopings which means altering the lower bound and range between two bounds, some proper ranges can be calculated. For instance, in training dataset, bidding price between 15 and 35 performed better than other ranges with average 137 clicks, while one of the competitive bid in validation set is from 60 to 80 with average clicks 77, which perform slightly better than constant bidding.

3.2 Linear Bidding Strategy

The bidding price is estimated in two steps. In the first step, it is estimated the CTR in order to predict the probability of "clicks" in the advertisement since the dataset is composed of impressions. In this part, many models were developed for click prediction such as Logistic regression, xgboost, Gradient Boosting Decision Trees and neural network. Then, the bid price was calculated by applying the linear equation (1). The main input of the equation is the "click" prediction.

$bid = \frac{pCTR}{avgCTR} * base\_bid$ (1).

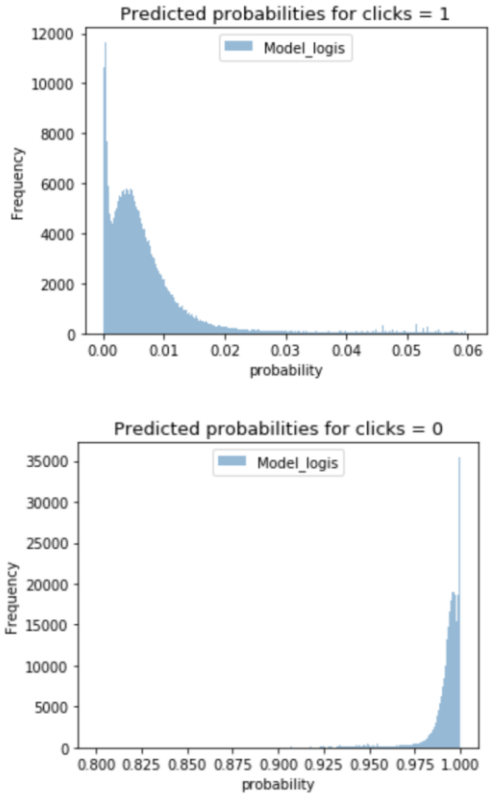
3.2.1 Logistic regression

Logistic regression is a linear approach where the response or dependant variable is between 0 and 1, which can also be interpreted as a binary type. When yi is equal to 1 means success, and yi = 0 means failure, so the response has a binary distribution [1]. In this case, a user "click" is represented by 1.

In this study, features were calculated based on 12 variables of the dataset. On the one hand, some of them were transformed to dummy variables since they represent to different categories and these are: hour, slot visibility, weekday, advertiser, browse, agent, region, slot width and usertag. On the other hand, two variables were considered as integer features (slotheight and city) because the former one represent size, and the latter multiple categories. So, the model had 163 features in total.
With the purpose of balancing the number of "clicks" and "non-clicks", the engineering cleaning process provided a proportion of ones and zeros of 1:100, respectively. Thus, the logistic model was run with 181093 rows.

The accuracy of this model is 0.99879 and the AUC score is 0.80797, which shows the robustness of the prediction. As expected, the prediction results suggest a small proportion of clicks, which vary between the probability of 0.0001 and 0.0493 (shown in figure 1), while the non-clicks predicted probabilities are in the range of 0.9507 - 0.9999, at 95% of confidence. Therefore, it is not convenient to classify success in the conventional way (prob>0.5), but instead use the success probability in the equation (1) for bid price estimation.

Figure 1. Histogram of clicks and non-clicks probabilities, Logistic regression

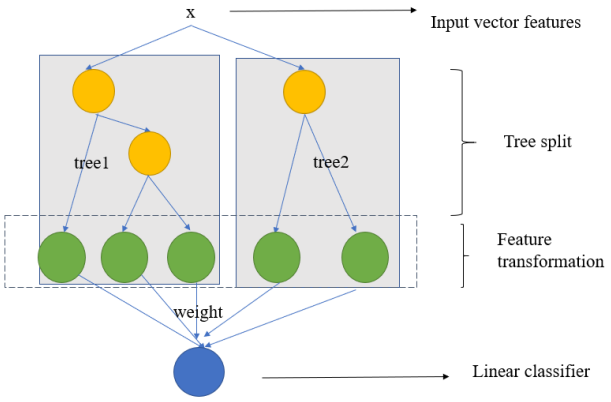


3.2.2 Gradient Boosting Decision Trees (GBDT) with Logistic Regression

Taken into account that the feature combination in the LR could not be solved directly by characteristics cartesian product or needs the considerable pre-analysis feature engineering, the hybrid model combining the Gradient Boosting Decision Trees with Logistic Regression is introduced [3].

This model uses Gradient Boosting Machine with L2- TreeBoost and is trained in a batch manner, implemented by lightbgm package. Here, each categorical feature is regarded as independent tree with index value as different leaf node. As illustrated in figure 2, each iteration creates a new tree to transfer the feature from real-value to binary value by fitting the linear classifier. The features generated by GBDT can be applied directly as the feature of LR, eliminating the links of manual processing. The input characteristics of LR are completely dependent on the characteristics obtained through GBDT. Every leaf node corresponds to a distinguishing feature and characteristic combination, which corresponds to the one-dimensional feature of LR.

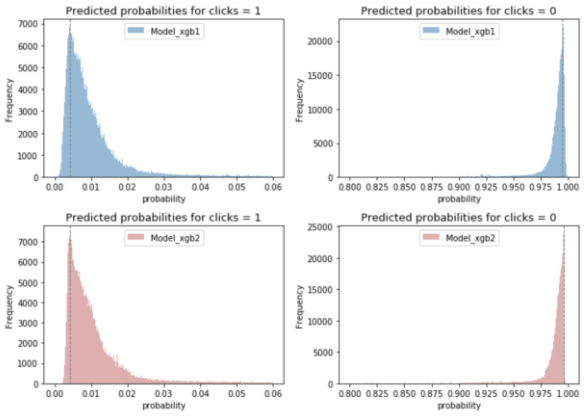
Figure 2. Graphic representation of GBDT combined with LR



3.2.3 XGBoost

The core steps of the XGBoost algorithm are basically the same as Gradient Boosting introduced above, both of them aim at iteratively generating a base learner, and then adds the update learner, but XGBoost is known as an advanced implementation of GB algorithm. This is because the XGBoost has the specific regularization step which could improve overfitting problem, the in-build method to processing missing values, and also the built-in cross validation method. This paper applies XGBoost through the default xgboost package in python, imports sklearn cross-validation to find optimal number of estimators, modifies parameters, and finally performs the highest clicks number in the testing set. The figure 3 illustrates there is a huge impact of tuning parameter on the predict CTR, also as mentioned in logistic regression, the probabilities of clicks 1 tend to be really low, it is therefore important to implement tuning and sample in order to improve model performance.

Figure 3. Histogram of clicks and non-clicks probabilities XGBoost



The correspond sample and tuning process will be illustrated in best bidding strategy part.

3.2.4 Neural Network and XGBoost

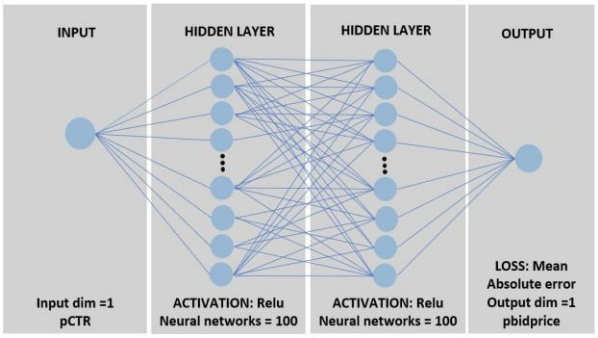
A combined model was developed using xgboost to predict CTR and NN to predict payprice. Regarding that the former model has

the highest accuracy for the linear bidding, we used it as an input of a new model to predict the payprices, which means: payprices = f(pCTR).

The model has two layers, where there are 100 neurons working under relu activation functions. This activation was selected because the input values are positive (see next equation), and also it helps to introduce non linearities to the model.

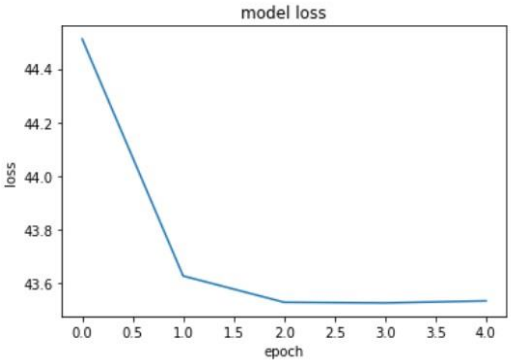
elu activation: f(x) = max{0, x} (2)

Figure 4. Architecture of CNN for price prediction



After running the model for 5 epochs, the results show a decreasing loss function, represented in this case by the mean absolute error (see the figure 5), which means that the model effectively was improving after each epoch.

Figure 5. Mean absolute error, neural network



The results shows 111 clicks (CTR = 0.000365) and 164446 impressions spending all budget (Cost = 6250000). Comparing this performance with other models, it is clear that this model achieved more impressions than the rest, but not enough number of clicks. Therefore, it is not considered as the best bid estimation model.

3.2.5 Performance Evaluation

From the table 1, the XGBoost model outperforms obviously compared with other models, in terms of click number and total cost of the whole bidding process while the CNN model performs worst here.

Table 1. The result performance of different models under linear strategy

	base bid	clicks	impression	CTR	cost	CPM	eCPC
LR	21	153	132419	0.00116	6250000	47.20	3485.78
GBDT+LR	79	93	109772	0.00084	6250000	56.94	3485.78
XGBoost	5	169	136237	0.00121	6249.998	44.00	3698.22
CNN	19	69	103387	0.00067	6250000	60.45	3485.78

3.3 Further Developed Bidding Strategy- Non-linear Bidding Strategy

3.3.1 Optimal Real-bidding Strategy

As dedicated in [3], this non-linear bidding strategy suggests the optimal activities of focusing on lower valued impressions due to its cost efficiency and winning chance. This optimization framework formula matches the impression level feature to the bidding strategy based on some dependencies between predicted Key Performance Indicators (KPIs) such as CTR, budget constraints and auction winning ratio. The functional optimization design proposed in [3] is

$$b(\ )_{ORTB} = \arg max_{b(\ )} N_T \int_\theta \theta w\big(b(\theta)\big) * p_\theta(\theta) d\theta,$$

$$subject\ to\ N_T \int_\theta b(\theta) * w\big(b(\theta)\big) * p_\theta(\theta) d\theta \leq B$$

Simplifying this formula for the progressive relationship x → θ → b → wand concave function assumption of w, the more straightforward result is $b()_{ORTB} = \sqrt{\frac{c}{\lambda} * \theta + c^2} - c$, where λ depends on the KPI (CTR here) and c is a constant.

Table 2: Notations and description

Notation	Description
$\boldsymbol{B}$	The campaign budget
$\boldsymbol{\theta}$	The predicted CTR
$\boldsymbol{b(\theta)}$	The bidding price
$\boldsymbol{w(b(\theta))}$	The winning ratio
$\boldsymbol{N_T}$	The number of bid requests

3.3.2 Model Performance Comparison

Illustrated in table 3, the practice result proves the reduce of eCPC compared with the linear strategy. Since the large number of advertisement flow in the reality market shows the heavy-tailed distribution with low price, only relying on the CTR is not rational in some condition. This ORTB strategy provides a tradeoff of maximally achieving the balance between CTR and winning ratio, causing the expectation of inclining the budget to low value impressions.

The probable reason contains the winning probability setting in the ORTB strategy. For the dataset this strategy produced, the log information covers the winning condition while in this dataset we experiment, all the records are the winning records, indirectly

making impacts on the strategy performance. Meanwhile, the value selection of c and λ is not precise enough even looping in the selected range.

Table 3. The result performance of different models under ORTB strategy

	c	λ	clicks	impression	CTR	cost	CPM	eCPC
LR	1	1.55e-6	136	136920	0.00099	5451321	39.81	3040.34
GBDT+LR	40	2.55e-6	92	156489	0.00059	6250000	37.97	3314.16
XGBoost	1	2.05e-6	150	150982	0.00099	5928489	39.27	3306.46
CNN	20	1.65e-6	67	96681	0.00069	6250000	64.65	3465.78

Table 4. AUC Evaluation of Bid Estimation Model

Bid Estimation Model	AUC
Logistic Regression	0.80797
GBDT+Logistic Regression	0.89857
XGBoost	0.87480
CNN	0.6779

3.4.1 Best Bid Estimation Model.

It can be seen from the above tables, after combining the different models for predict CTR calculation with linear or nonlinear bidding strategy, the final best combined model is XGBoost with linear bidding strategy. Before applying the models, there are few steps need to be done, the whole process will be explained below.

First of all, features engineering, which mentioned before, is the cornerstone of whether the model could be robust. When dummying object format input variables for categories, rather than altering all the object format features into one hot code format, an initial data exploration on the dataset is needed. For example, looking at the dataset, feature 'useragent' actually contains both device (either cell phone or computer) agent and browse type of one IP. In this case, separating the 'useragent' into two features 'agent' and 'browse' may help model improve its performance. Other feature engineering details will be described in individual report.

Table 5. Feature 'Useragent' profiles

IP	USERAGENT
125.37.175.*	windows_ie
171.36.92.*	windows_chrome
59.46.106.*	mac_safari
114.250.226.*	android_opera

After that, a simple overview of clicks num (1793 clicks and 303925 impression) from validation set reminds the necessity of sample, which means abstract a certain ratio of value 0 and 1 from feature 'click', by keep adjusting the ratio and measuring by average predict CTR, a suitable sample size can be generated to build the model.

Entering the built XGBoost model, tuning parameters part, there are basically three types of parameters[2], taking a set of them as examples :

Table 6. XGBoost parameters

Parameter Type	Parameter Name	Definition
General Parameters	booster	Type of model
	silent	Running messages
	nthread	Number of cores
Booster Parameters	Learning rate	Rate of learning on each step
	Min_child_weight	The minimum sum of weights of all observations required in a child
	Max_depth	The maximum depth of a tree
	gamma	The minimum loss reduction required to make a split
Learning Task Parameters	eval_metric	Metric for validation data, rmse, log loss etc.

In this report, the focuses would be the changes of parameters 'n_estimators', 'learning rate', 'max_depth' and 'min_child_weight', by applying cross validation, those parameters will be allocated with optimal values in order to avoid underfitting and overfitting problem and hence make our model robust.

After all the adjustment, the best CTR prediction XGBoost model has been build. Comparing the performance between model with the linear and nonlinear bidding strategy, the best bid estimation model is the combination of XGBoost and linear bidding strategy.

4.CONCLUSIONS

Before modelling, there a process of features engineering and sampling. Since most of the features are categorical values, it was necessary to make a previous analysis before creating binary variables and scale transformations. Also, taking into account the big size of the training dataset, it was essential to sample the most representative data. In addition, the original dataset was unbalanced in terms of number of clicks and non-clicks. Thus, the final sample has a proportion of 1:50 of ones and zeros, respectively.

5. REFERENCES

[1] Bolstad, W.M., 2012. Logistic Regression. In Wiley Series in Computational Statistics. Hoboken, NJ, USA: John Wiley & Sons, Inc., pp. 179–202.

[2]. Chen, T. and 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.

[3] He, Xinran et al., (2014). Practical Lessons from Predicting Clicks on Ads at Facebook. Proceedings of the Eighth International Workshop on data mining for online advertising,

pp.1–9. Available from: https://dl-acm-org.libproxy.ucl.ac.uk/citation.cfm?doid=2648584.2648589 (Accessed: 4th April, 2018).

[4] Zhang, W et al., (2016). Optimal Real-Time Bidding for Display Advertising. Optimal Real-Time Bidding for Display Advertising, pp.Doctoral thesis, UCL (University College London).

[5] Github link: https://github.com/Jihong-Deng/COMPGW002-Web-Economics-Coursework