COMPGW02 Individual Report

Parham Oghabi
parham.oghabi.17@ucl.ac.uk

1. INTRODUCTION

Online advertising is a growing industry which generates a large percentage of revenues for companies such as Google and Facebook. In this assignment, I have worked on an online advertising problem to devise a bidding strategy for advertisers to compute a bid for each impression in a real-time bidding environment in order to maximize their KPI, which is the total number of clicks. To develop this bidding strategy, the CTR model was trained using contextual and user behavioral features from the iPinYou dataset, which contained real logs of won ad impressions and their respective clicks. My optimal bidding strategy was found to be a combination of a non-linear and concave strategy with a linear bidding strategy and various evaluation metrics such as clicks, CTR, and CPC were computed to support the results. In addition to the bidding strategy, an exploratory analysis of the iPinYou dataset was also performed. Even though the data is a subset of the original data, the analysis matches the results in [1]. All codes related to the analysis, CTR model, and bidding strategies can be found on my Github: https://github.com/oghabi/realtimebidding.git

2. RELATED WORK

Research in real-time bidding in the context of display advertising has been limited compared to other areas of advertising but has started gaining attention in recent years. In order to compute the value of a bid on an impression, its CTR (click through rate) should be estimated based on the bid request. Various non-linear deep models have been researched by authors in [2, 3] such as using convolution and max-pool layers to automatically extract features from creative images and also using multiple dense layers to learn non-linear contextual and behavioral features to predict the CTR of an impression. By using both the bid request features and the automatically extracted ad image features, they were able to suggest creatives which had the highest CTR. Non-deep models such as factorization machines in [4] and Xgboost have also been explored and deemed to be very effective due to its scalability on large datasets. Linear models cannot extract nonlinear correlations between the hand-crafted bid request features but factorization machines use generic latent factor models which solve these shortcomings.

Traditional bidding strategies such as constant bids and linear models have been mostly used in DSPs. However, researchers have come up with a variety of better strategies such as non-linear optimal strategies such as ORTB in [5] which bids higher values for lower predicted CTR compared to a linear strategy. The previous strategies are a static optimization problem and computes bids on each impression independently of other impressions. There has also been work done on feedback controller based bidding strategies in [6] which dynamically adjusts the bids to control the number of clicks KPI. [7] also formulates the problem as a reinforcement learning problem to computer dynamic real-time bids. In this paper, I have chosen to combine the strategy in [5] and a linear strategy to come up with my optimal strategy.

3. Data Exploration

3.1 Basic Statistics

The dataset was a subset of the original released iPinYou log. It only consisted of won impressions. Each row contained the bid request and a variety of features, the bid price, the pay price and whether the impression was clicked or not. Table 1 outlines the basic statistical information of the training dataset. The bid, pay, and floor price were all adjusted and divided by 1000.

Table 1. Basic Statistics of Train Data

Number of Impressions	2,430,981
Number of Clicks	1793
Cost (Pay Price)	189,984.6
CTR	0.074%
Average CPM	78.15
eCPC	105.95

Table 2. Basic Statistics of Each Advertiser in Train Data

Advertiser ID	Clicks	Bid price	Pay Price	Impressions	CTR	CPM	eCPC
1458	385	147,705	33,968.7	492,353	0. 0782%	68.99	88.23
2259	43	38,508	12,428.23	133,673	0.0322%	92.97	289.03
2261	36	31,701	9,873.78	110,122	0.0327%	89.66	274.27
2821	131	61,343	18,828.04	211,366	0.0620%	89.07	143.73
2997	217	13,802	3,129.27	49,829	0.4355%	62.80	14.42
3358	202	61,729	22,447.231	264,956	0.0762%	84.72	111.12
3386	320	136,512	34,931.82	455,041	0.0703%	76.77	109.16
3427	272	95,218	30,458.71	402,806	0.0675%	75.62	111.98
3476	187	77,042	23,918.78	310,835	0.0602%	76.95	127.90

Each row or won impression in the training data belongs to a certain advertiser which specialized in a certain industry. There were a total of 9 advertisers and Table 2 above describes the basic statistics for each of the advertisers. We can see that even though we are using a small smaller dataset, our results match the benchmarks set in [1]. From the above statistics on both the whole training data and the advertiser specific dataset, we can see that (i) most of the CTRs are around 0.08%, which matches the average CTR (0.1%) for desktop display advertising; (ii) the expected cost per click for each advertiser varies a lot due to the fact that the market of each advertising campaign is different and they are targeting different bid request features (ie. such as location, user demographic, and time). In the following sections, I will discuss the effects of various bid request features on the CTR, pay price, and the eCPC, specifically for advertisers 1458, 3358. These were chosen in order to verify my results with the analysis done in [1].

3.2 User Feedback

Figure 1 describes the effect of various bid request features on the CTR for advertisers 1458 and 3358. Features such as weekday, hour of day, OS of user, browser, and the ad exchange were used.

Figure 1: Effect of various bid request features on the CTR of advertisers 1458 and 3358.

Same features have different effects on the CTR of different advertisers:

i) On Wednesdays, the CTR for advertiser 3358 drops significantly. Whereas on Thursdays, it has the highest CTR. This indicates that advertiser 3358 might want to focus on bid requests on Thursdays and win as much impressions on that day in order to have a higher CTR and possible generate higher revenues.
ii) Advertiser 1458 has an extremely high CTR from iOS mobile users. On the other hand, Linux and Mac users tend to click more on ads from advertiser 3358. A similar analysis can be done for the browsers. Advertiser 3358 gets most clicks from the browser Safari.
iii) The hour of day also has an impact on the CTR. In the mornings, the CTR is low on average compared to the evening. Advertiser 3358 has a spike in its CTR at around 10 pm.

Other features such as the user segmentation tags from DMPs also have a varying effect on the CTR and can help advertisers refine their targeting rules on different user segments.

3.3 Bidding Behaviour

If an advertiser wins an auction, they have to pay the pay price, not their bid price. This is the second highest bid and is defined as the market price of the impression. Figure 2 displays the effects of the previously mentioned features on the pay price. We can see that the won impressions of advertiser 3358 cost a lot more than the ones for 1458 on all days of the week and on most hours of the day. Furthermore, we can also see that ad exchange 1 is a more competitive environment for advertiser 3358 due to the high pay price in that exchange. These analyses are useful for advertisers because they can then optimize their bidding strategies and CTR prediction models to maximize their KPI.

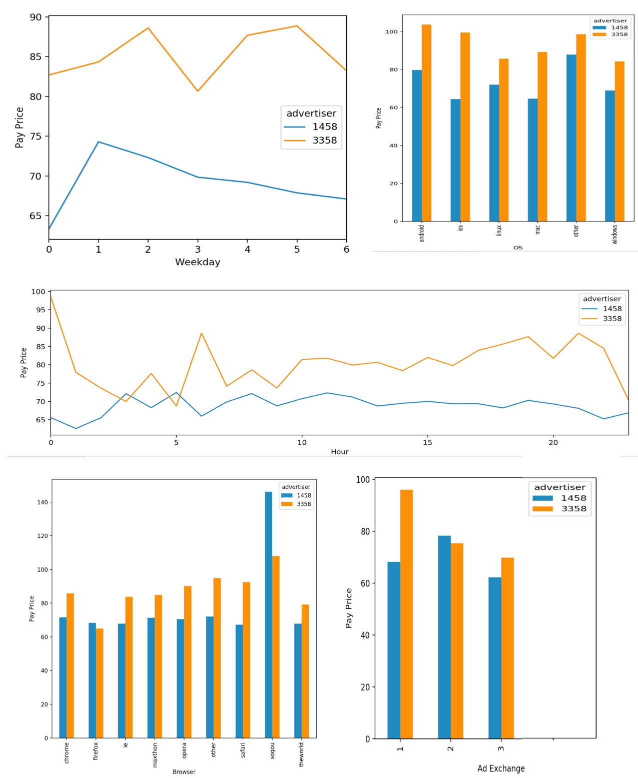

Figure 2: Effect of various bid request features on the Pay Price of advertisers 1458 and 3358.

3.4 eCPC

Finally, Figure 3 shows the effects of the same features on the expected cost per click but only for advertiser 3358. The goal of advertisers is to maximize the number of clicks but also to reduce their expected cost per click in order to spend less to achieve a click. We can see from Figure 3 that the eCPC for advertiser 3358 is extremely high on Wednesdays. This can be explained from the previous two figures. Advertiser 3358 has the lowest CTR on Wednesdays and its highest pay prices are also on Wednesdays which explains the high eCPC on the same day. The same reasoning can be done for the high eCPC on ad exchange 2, it has a low CTR and high market price on that exchange. Advertiser 3358 is more cost effective in advertising to Linux and Mac users as opposed to Windows users. Advertisers can use the eCPC to re-allocate their budget to optimize their strategy.

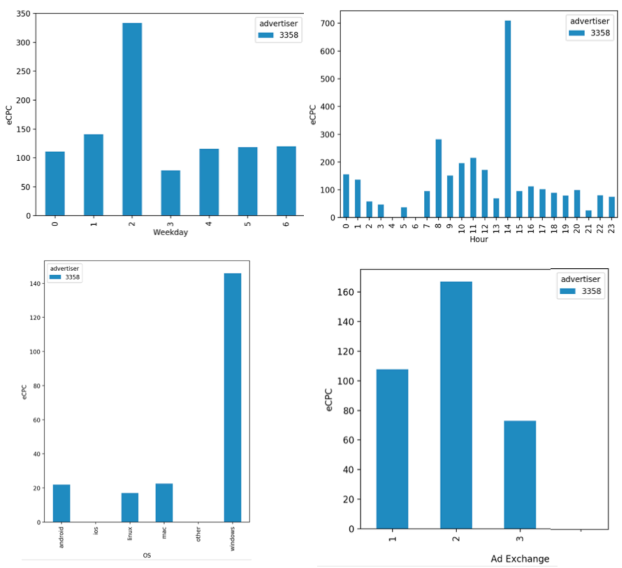

Figure 3: Effect of various bid request features on the eCPC of advertiser 3358.

4. CTR Estimation

4.1 Feature Engineering

In order to train the CTR prediction model, the following features were used from the bid request: weekday, hour, OS, browser, region, slot width and height, slot format, slot visibility, slot floor price bucket, and user tags.

The user agent column is parsed to obtain the operating system and the browser. Each user tag is also considered as a separate feature. The slot floor price is split into one of the five possible buckets:0, [1, 10], [11,50], [51, 100], [101, 1000). Higher order combination of features was not computed and all features were binary which results in one-hot encoding input features. There was a total of 198 features which represented each single incoming bid request.

4.2 CTR Prediction Model

From the statistical analysis done on the advertiser specific and the whole dataset, it is clear that the dataset is very imbalanced with only 0.073% of the won impressions being clicked. Therefore, the majority class in the dataset was downsampled to 2.5% of the original size, which resulted in 62,522 impression logs. 2.5% was chosen as the optimal value based on the results by Facebook researchers in [8]. Negative downsampling decreased the training time and improved the results.

From Figures 1 to 3, we saw that different features have a varying effect on the CTR, bids, and eCPC for various advertisers. Therefore, it is reasonable to have different CTR prediction models for different advertisers and to train each model independently. In fact, some DSPs do just that and allocate a small budget at first to bid randomly for each advertiser when they sign up to generate some training data for its model. In addition to the nine independent advertiser-specific logistic regression models, the following models were also implemented: single Logistic Regression, Xgboost (boosted decision tree), Factorization Machine. The AUC values of the models on the validation data are shown in Tables 3 and 4.

Hyper-parameter optimization, such as the L2 regularizer value and the depth and number of trees in Xgboost, was performed for all the models using scikit-learn's grid search on the validation data. L2 regularization was also applied to avoid overfitting and the loss is the cross entropy between the predicted probability and whether a click actually occurred or not. Class weights were also applied to models which allowed to do so in order to penalize the model if it predicted the minority sample class incorrectly. This was done in addition to the downsampling to give better results. All models were trained on the downsampled training data.

Table 3. AUC scores of various CTR prediction models

Model	AUC	Confusion Matrix	
Single Logistic Regression (L2 = 0.01)	85.11	252,191	51,532
		58	144
Xgboost (# trees = 300, depth = 3, learning rate = 0.1)	87.14	303,169	554
		135	67
Factorization Machines	83.45	303,060	663
		144	58

Table 4. AUC scores of 9 independently trained advertiser specific Logistic Regression on both the original and downsampled dataset

Avertiser ID	AUC (with downsampling)	AUC (without downsampling)
1458	94.88	90.67
2259	20.8	24.4
2261	60.58	42.5
2821	60.12	63.3
2997	66.7	60.9
3358	91.1	90.6
3386	68.1	69.5
3427	90	90
3476	78.26	79.1

Logistic Regression is a linear model which can scale to billions of samples and parameters but is incapable of extracting complex and non-linear features from the bid request features without manually inputting higher order combination of features. Therefore, we can see from the confusion matrix in Table 3 that even though Logistic Regression predicted the most number of clicks correctly, it also over-predicted the non-clicks and this can result in us spending the budget wastefully. Therefore, models such as Xgboost and Factorization Machines tend to be more useful. Table 4 also shows the nine logistic regression models which were trained independently on each advertiser's won impressions. The AUC values are fairly better when trained on the downsampled dataset compared to when the data is not downsampled. Since downsampling was necessary, I decided to proceed with the Xgboost model since it had the highest AUC score.

Another important point to note is that since downsampling was performed, we have to take into account that the mean of the predicted CTRs of the models will be much higher than the average CTR of the original training data. Therefore, all the model predicted CTRs have to be calibrated using the equation in [8]: p/(p+(1-p)/w) where p is the model predicted CTR and w is the negative downsampling rate. The mean of the calibrated pCTR of Xgboost also performed better than the single Logistic Regression model as shown in Table 5 below and matches the average CTR of the training data in Table 1.

Table 5. Mean of the calibrated and non-calibrated predicted CTRs

	Mean of Calibrated pCTR	Mean of Non-Calibrated pCTR
Logistic Regression	1.9%	34.11%
Xgboost	0.071%	2.04%

5. Bidding Strategy

5.1 My Best Bidding Strategy

Initially, the non-linear optimal real-time bidding strategies in [5] were used. Compared to a linear strategy, ORTB [5] is a concave bidding function which allocates more budget and bids higher for cases with a low predicted CTR and bids lower for cases with a high predicted CTR. Figure 4 shows the winning function, bidding function, and the graph of the bid price against the predicted CTR for the two versions of ORTB. The bidding functions were tested on the validation set. The winning function is a function of the bid price which depends on the predicted CTR by Xgboost using the contextual and user behavioural features.

$$w(b(\theta)) = \frac{b(\theta)}{c + b(\theta)}$$

$$b_{\mathrm{ORTB1}}(\theta) = \sqrt{\frac{c}{\lambda}\theta + c^2} - c$$

$$w(b(\theta)) = \frac{b^2(\theta)}{c^2 + b^2(\theta)}$$

$$b_{\mathrm{ORTB2}}(\theta) = c \cdot \left[\left(\frac{\theta + \sqrt{c^2\lambda^2 + \theta^2}}{c\lambda}\right)^{\frac{1}{3}} - \left(\frac{c\lambda}{\theta + \sqrt{c^2\lambda^2 + \theta^2}}\right)^{\frac{1}{3}}\right]$$

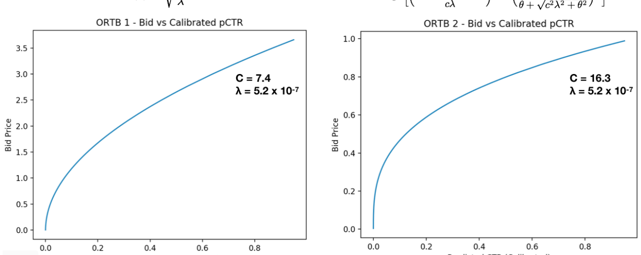

Figure 4: The graphs of the bid price vs the calibrated pCTR for the two ORTB bidding functions on the validation set.

Due to the results and reasons mentioned in the following evaluation subsection, a mixture of ORTB 1 and a linear bidding strategy was used as my best strategy instead of only using a concave non-linear strategy. Figure 5 shows the intersection of my optimal linear bidding function and ORTB 1 strategy to the left and my best bidding strategy to the right. It essentially uses the ORTB 1 bidding strategy until a certain threshold and then follows the linear strategy once the calibrated predicted CTR exceeds that threshold.

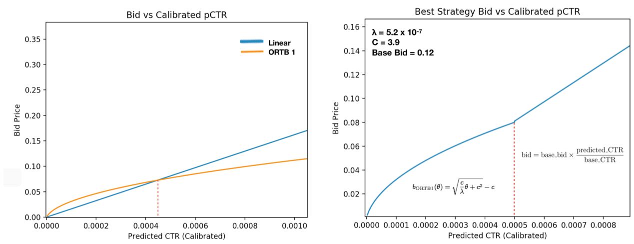

Figure 5: The left plot is the intersection of the linear and ORTB 1 bidding strategy against the calibrated pCTR. The right plot is my best strategy which is a combination of the linear and ORTB 1 functions

The parameters C, λ, the threshold, and base bid value were tuned using grid search on the validation set in order to meet the budget constraint and to maximize the number of clicks and deplete the budget. The base CTR was chosen as the average training set CTR mentioned in Table 1. Setting the threshold value close to the intersection of ORTB 1 and the linear function yielded a good number of clicks on the validation set. Figure 6 below shows the process of finding the optimal value of C for ORTB 1. The number of clicks increase until that optimal value and then drops slightly.

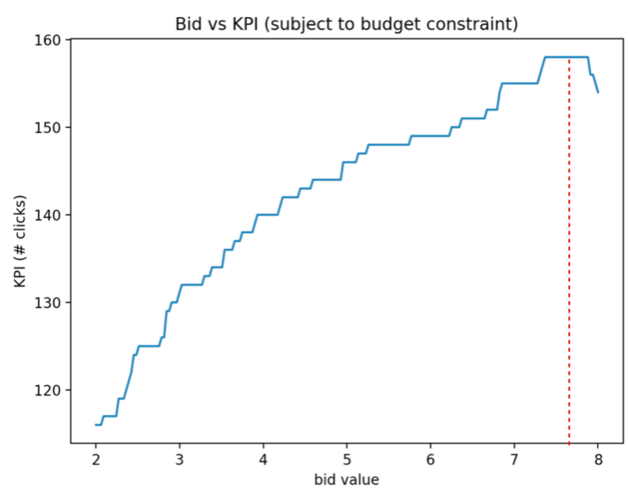

Figure 6: Plot showing the hyper-parameter tuning process of ORTB 1. The optimal value of C is shown by the red dashed line.

The following subsection will discuss the evaluation metrics for the three models and why my best strategy performed the best.

5.2 Evaluation

Table 5 outlines the various evaluation metrics of the three different strategies on the validation set with the optimally tuned parameters as stated in Figures 4 and 5. Clearly, my optimal bidding strategy outperforms ORTB 1 and 2 in terms of maximizing the KPI (number of clicks). The validation set had a total of 202 clicks and we can see that my strategy not only achieves the highest number of clicks but also achieves the highest CTR. As a result, the eCPC is very low which is very beneficial for advertisers.

Table 5. Evaluation Metrics of the three different bidding strategies showing that my best bidding strategy performs better than the others on the validation set.

	ORTB 1	ORTB 2	ORTB 1 + Linear
Clicks	158	147	163
Won Impressions	151,763	157,603	141,939
CTR	0.104%	0.09%	0.115%
Budget Spent	6249.97	6249	6249
Avg CPM	41.18	39.65	44.03
Avg CPC	39.55	42.5	38.34

ORTB 2 achieves the lowest number of clicks because as shown in Figure 4, the strategy has a sudden increase in its bid prices in the beginning which causes it to deplete the budget quickly on the validation set. Therefore, even though it won the most number of auctions, it only did so because it bid very high on impressions with a low pCTR. ORTB 2 is useful for competitive impressions with a high market price.

Another positive point about my strategy is that it can achieve its maximum number of clicks (163 in Table 5) without depleting the budget. Table 6 below shows the minimum budget required to achieve their maximum possible clicks. Even though the number of won impressions drops for all strategies except ORTB 2, that is not very important since the main goal for advertisers is to maximize the number of clicks and CTR and to minimize the eCPC. ORTB 2 cannot save money due to its aggressive bidding behavior but my strategy spends much less money than ORTB 1, yet achieves more clicks.

Table 6. The minimum budget required by each bidding strategy to achieve its maximum number of clicks possible.

	ORTB 1	ORTB 2	ORTB 1 + Linear
Clicks	158	147	163
Minimum Budget Required for max clicks	5928	6249	5733
Won Impressions	147,179	157,603	130,435

6. Conclusions

In this assignment, I performed a statistical analysis of the iPinYou won impression logs. I also designed a Xgboost CTR prediction model which achieved an AUC score of 87 on the validation set. Using this model to predict the impression CTR, I combined a non-linear and linear bidding strategy which outperformed the concave ORTB bidding strategies on the validation set. The strategy achieved its goal of maximizing the KPI (number of clicks) by not only meeting the budget constraint but also being able to save money.

For questions 2 to 5, we each separately attempted to come up with the best hyper-parameters (for constant, random, and linear bidding) and the best bidding strategies (for questions 4 and 5) which resulted in the highest number of clicks on the validation set. We then compared our results. My hyper-parameters for the constant, random, and linear strategies were the best so we used my results. For question 5, we combined our CTR prediction models to build a stacked ensemble prediction model and we picked my individual bidding strategy since it performed best on the validation set. This allowed us to get a few more clicks on the actual test set when we used the feedback API.

In the future, it would be interesting to have a real-time multi-agent feedback REST API for the coursework in order to submit our python script which can dynamically change the bid sequentially for each impression based on the API feedback (won or lost and also was it clicked if won) instead of submitting a list of pre-set bids. This would allow us to implement reinforcement learning bidding strategies in [7] as it is able to get real-time feedback on each single impression sequentially and adjust its strategy over time on the test set. In the current setting for criterion 2's group competition, if some auctions are lost in the test set then that would result in not utilizing the full budget since it will not be able to be re-invested whereas a real-time multi-agent feedback API will resolve that issue.

7. REFERENCES

[1] Weinan Zhang, Shuai Yuan, Jun Wang, and Xuehua Shen. *Real-time bidding benchmarking with ipinyou dataset*. arXiv preprint arXiv:1407.7073, 2014.

[2] Qiang Liu, Feng Yu, Shu Wu, and Liang Wang. *A Convolutional Click Prediction Model.* CIKM 2015.

[3] Junxuan Chen et al. *Deep CTR Prediction in Display Advertising*. MM 2016.

[4] Richard J. Oentaryo et al. *Predicting Response in Mobile Advertising with Hierarchical Importance-Aware Factorization Machine.* WSDM 2014.

[5] Weinan Zhang, Shuai Yuan, and Jun Wang. *Optimal real-time bidding for display advertising.* In Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, pages 1077–1086. ACM, 2014.

[6] Weinan Zhang et al. *Feedback Control of Real-Time Display Advertising*. WSDM 2016.

[7] Han Cai, Kan Ren, Weinan Zhang, Kleanthis Malialis, Jun Wang, Yong Yu, and Defeng Guo. *Real-time bidding by reinforcement learning in display advertising*. In Proceedings of the Tenth ACM International Conference on Web Search and Data Mining, pages 661–670. ACM, 2017.

[8] Xinran He et al. *Practical Lessons from Predicting Clicks on Ads at Facebook.* ADKDD 2014.