COMPGW02: Web Economics Coursework Project

Group Report

Lynray Barends
SN: 17032368
ucakljm@ucl.ac.uk

Alexander Cowen-Rivers
SN: 17036378
ucakaia@ucl.ac.uk

Tim Warr
SN: 13026783
ucabtw2@ucl.ac.uk

ABSTRACT

In this report, we explore the problem of real-time bidding in display advertising. We use the iPinYou dataset to train and evaluate multiple CTR predictors, bid-landscape forecasters and bid optimisation strategies. Our best is an ensemble of two optimisation strategies; one based on a leading paper in the bid optimisation field and the other a variation of linear bidding strategy, where the CTR predictor architecture is a neural model.

Keywords

Real-time auction, CTR prediction, bid-landscape forecasting, bid optimisation, multi-agent reinforcement learning

1. INTRODUCTION

The code [1] for this project can be found on GitHub.

1.1 Problem Context

When a user visits a web site, they will reveal information such as their *web browser*, *operating system* and *IP address*, this is combined with general information such as *date and time* as well as facts about the ad space available (*slot height, width, visibility* to form a feature vector representing the unique visit of a given user at a particular time. We refer to this feature vector as an *impression.*

Real-time bidding is an approach in display advertising that allows impressions to be auctioned off individually, in the time it takes for a web page to load. This is opposed to the (perhaps better known) *keyword advertising*, in which an advertiser pays to have their listing show to any user that, for example, uses a specific search term on a search engine. One of the main benefits of real-time bidding is therefore that it allows an advertiser to gain greater flexibility over who sees its adverts and when.

Most commonly, the auctions carried out in real-time bidding are second price auctions, which means that the highest bidder will win the auction and pay the second highest bidder's price. Studies from auction theory have shown that second-price auctions promote truthful bidding [5], however once budget constraints of each participant in the auction is taken into consideration, this might not be as clear [12].

The problem we consider in this report is the one of developing machine learning algorithms that are able to (a) predict whether a given impression is likely to result in a click and (b) bid successfully on these impressions, when taking into account budget constraints and market considerations.

[1]`https://github.com/uclwe/rtb`

1.2 Evaluation Metrics

To compare the success of bidding strategies, we define the following performance metrics (the unit for those related to prices is *Chinese fen*):

- **Clicks** - the number of auctions won by a strategy that resulted in a click-through (our principle objective is to maximise the number of clicks for a fixed budget).
- **CTR** - click-through rate:
$$\frac{\#clicks}{\#\ auctions\ won}$$
- **Cost** - sum of the price payed across all actions won
- **aCPM** - average cost *per mile* (per 1000 impressions)
- **aCPC** - average Cost Per Click.
- **pCTR** - predicted probability a given impression will result in a click-through

The budget for this task is fixed at 6250 CNY, which corresponds to 6,250,000 Chinese Fen.

1.3 Approach

The report is structured as follows: in Section 2 we briefly explore the related work in the field. In Section 3, we first benchmark some basic bidding strategies including bidding a constant amount for each impression and bidding a random amount. We then describe and optimise a linear bidding strategy using various CTR predicting algorithms. Furthermore, we develop our 'best bidding strategy' i.e. the strategy for which we obtained the highest number of clicks on the validation set. We conclude with an analysis of the performance across all strategies and summarise some of the key areas we think are important to focus on for future development of RTB strategies.

2. RELATED WORK

In [13], the authors provide a basic statistical analysis of the full iPinYou dataset (note that in this project we only use a smaller subset - due to computational complexity and resource limitations). They also provide some benchmarking algorithms for CTR-prediction (logistic regression & gradient boosted decision trees) and bid optimisation (constant, random, linear, bidding below max eCPC).

The authors of [7], introduce an ensembled decision tree with logistic regression model to predict clicks on Facebook adverts. They highlight the importance of good feature selection and describe how to sensibly down-sample and re-calibrate a high dimensional dataset into one that is less computationally intensive to perform inference over.

Other papers focus on the problem of bid-landscape forecasting, which involves predicting the market price distribution of a give impression. Specifically [4] forecasts the bid for each sample using GBDT regression and then fits a GMM to generate a campaign level bid distribution.

Non-linear bid optimisation strategies are described in papers such as [11], [12]. More specifically, in [12], the authors describe a strategy which takes into account the budget and the campaign lifetime, wanting to bid on more impressions instead of a subset of high valued ones. This strategy is cost effective as it bids more frequently on lower valued impressions, meaning that the advertisers save money and the chances of winning are higher. In [11], the authors challenge the idea of using value based bidding, as they suggest that this strategy favours users with an already high action rate (e.g. click, or conversion). Instead, they suggest a lift-based bidding strategy that is based on the action rate lift i.e. focusing on users that would be more influenced by the ad specifically.

The authors of [3], [10] formulate the real-time bidding problem as a Markov Decision Process (MDP) and introduced model-based and model-free (respectively) reinforcement learning algorithms to develop bidding strategies that are able to additionally consider budget constraints.

The work of [8] is an extension to [3] and formulates the real-time bidding problem with competition as a multi-agent reinforcement learning problem, which they using the actor-critic framework to solve.

3. APPROACH & RESULTS

3.1 Data Exploration

PLEASE REFER TO INDIVIDUAL REPORTS.

3.2 Basic Bidding Strategies

In order to form a benchmark, by which to compare later approaches, we first implemented two simple strategies - which are described below.

3.2.1 Constant Bidding Strategy

This strategy involves bidding a constant value across all impressions. The only parameter to tune is therefore c - the value of this constant bid. One has to find a value for c that is sufficiently high, so as to win as many actions as possible, but low enough to have a realistic chance of placing bids for many actions (otherwise the fixed budget of 625 0CNY will be exhausted early on).

We performed a statistical analysis of the training data (please refer to individual reports) and initially trialled values for c based off our findings. The preliminary tests included bidding low (2nd quartile), high (3rd quartile), the median, and the mean market price. The mean market price ($\mu = 78$) produced the best result, resulting in 67 clicks (0.0046 CTR) on the validation set.

We then implemented a grid search that evaluated the click through performance across a range of constant bid parameters. Our best constant bids were between 77 and 79, producing 68 clicks (0.00046 CTR) on the validation set. We note that this corresponds with the mean market price of the training set (please see above).

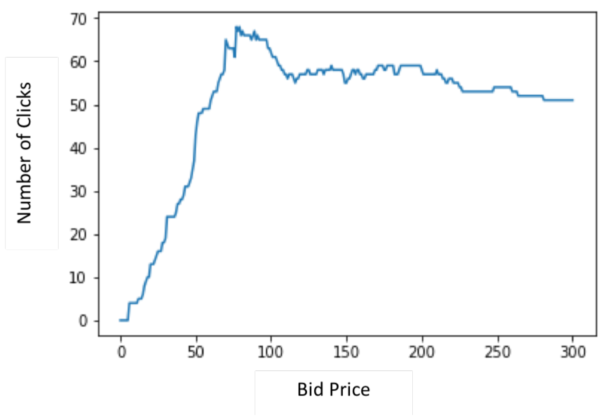


Figure 1: Constant Bidding Grid Search

One reason why constant bidding could be considered sub-optimal is that the strategy is easy for your competitors to exploit.

3.2.2 Random Bidding

Our second basic strategy was one of random bidding, where bids would be selected randomly according to a distribution. We propose that the market price of each impression is noisy, and so a simple stochastic strategy may have some advantages over the fixed bid strategy, described previously. We chose to explore two types of random bidding strategies:

- **Random Uniform:** This bidding strategy bids randomly between an upper and lower bound, each having an equal chance of being selected.

- **Random Normal:** This bidding strategy bids randomly according to a normal distribution, where mean and standard deviation are inserted as parameters.

As with the constant bidding strategy, we performed a couple of single tests based on our market price values taken from the training data. In search of the best parameters for our strategies, we performed a grid search, and used our validation set to get the parameters that produced the highest number of clicks. As these strategies were random, the performance varied slightly every time the test was run, so we performed multiple trials in order to get a more accurate reading for the most optimal parameters.

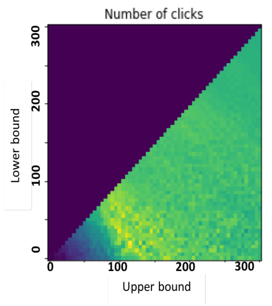


Figure 2: Random Uniform optimisation

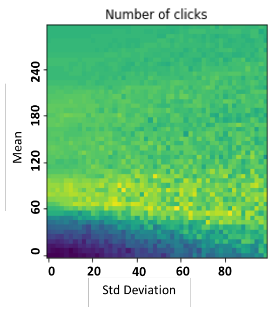


Figure 3: Random Normal optimisation

Table 1: Comparison of bidding strategies

Approach	Parameter(s)	Cost	Clicks	CTR	CPC
Constant	$c = 77$	100%	68	0.000460	91.9
Random (uniform)	$c_1 = 15, c_2 = 120$	100%	80	0.000523	78.12
Random (normal)	$\mu = 61.2, \sigma = 59.2$	100%	73	0.000534	85.64
Linear (GBDT)	$base\_bid = 52.7$	98%	163	0.001063	37.76
Linear (MLP)	5 Layers	100%	156	0.001302	39.87
MARL DRLB w Linear Approx	-	100%	72	0.000413	84.74

We note (Figure 2), that the most optimal upper bound for the random uniform bidding strategy is close to 100 - this is interesting, since in the training data, the bid prices ranged from 250 to 300. For the random normal bidding strategy, the best parameters for the mean were close to 60 (see Figure 3). This is close to the median market price value in the training data.

While these strategies provided satisfactory results for a first pass, they do not take the features of the impression and user into account. This information could be important in informing us about the value of the impression.

3.3 Linear Bidding Strategy

Strategy

A drawback of the *basic bidding strategies* mentioned previously, is that the bid price is independent of any information that could be inferred from the impression. One simple way to incorporate impression-specific information is the linear bidding strategy [13]. It is defined as:

$$bid = base\_bid \times pCTR/avgCTR \tag{1}$$

In other words, this function defines a bidding strategy that increases linearly based on the probability of a click.

CTR Prediction

The data was formatted in the way that we describe in Section 6.1. We developed CTR predictors using logistic regression (due to their popular use in rare event identification - [1],[9],[6]) and gradient boosted decision trees. The models' parameters were initially tuned based on what we could find in literature and then fine-tuned using a combination of grid search and course to fine search. Evaluation metrics we considered for the models included log-loss, ROCAUC and precision & recall.

Optimisation

Optimisation was performed for the linear bidding strategy (using both CTR predictors) on the validation set. Since the term $avgCTR$ is a constant (defined as the average click through rate on the training set), we actually performed optimisation for the $base\_bid/avgCTR$. In order to ensure robustness of the CTR optimisation procedure, we performed five fold cross validation [2]. We also performed coarse to fine parameter optimisation over our $base\_bid/avgCTR$, by first doing a wide random search over a range of 10^8 values, then performing a high density narrow random search over the areas of highest performance (a range of 10^2 values).

Our tests found that GBDT was more successful than LR (and better than an ensemble between the two). The optimal parameter for $base\_bid/avgCTR$ was found to be 75275, this is equivalent to a base bid of 52.7 CNY. Results of this can be seen in Table 1 and a representation of the base bid optimisation is shown in Figure 4.

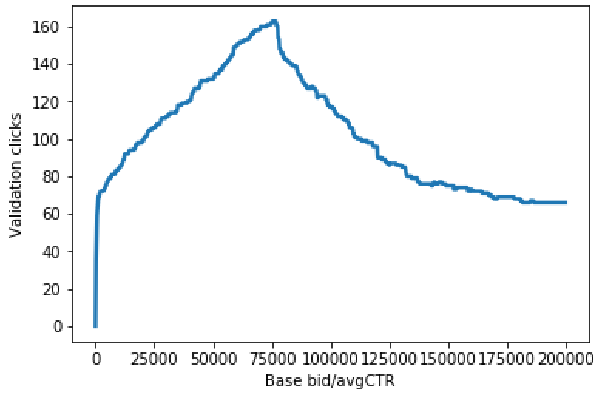


Figure 4: Optimising linear bidding strategy

Analysis

The performance of the linear bidding strategy on our GBDT CTR model was extremely strong. As we see in Table 1, the strategy was able to obtain 163 clicks on the validation set with a CPC of 37.76. Figure 5, shows a basic plot of the distribution of this strategies' bids. Whilst the majority of bids were reasonable (min, median, mean 5.4, 48.6, 112.8 respectively), we note that the maximum bid was 147120.8. In the single-agent setting, the objective is to beat the market price (i.e. the second highest bid) - so even with such a high maximum bid, the highest one would pay for an impression is under 300.

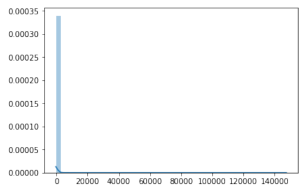


Figure 5: Distribution of linear bids

In a multi-agent setting however, the idea of having a maximum bid of nearly 150,000 (2%) of the budget is extremely risky. If two linear bidding strategies were to go up against one another, then they could exhaust their budgets very quickly. We explicitly state that strategies used in the multi-agent setting are best when trained as such, but at least a maximum cap should be put on bid price to avoid early budget exhaustion.

3.4 Best Bidding Strategy

PLEASE REFER TO INDIVIDUAL REPORTS.

3.5 Group Bidding Strategy

In this subsection we combined the best aspects from our individual bidding strategies. To summarise; ACR focused on building a neural CTR-predictor and implemented a reinforcement learning agent that could consider budget constraints; LB implemented and developed upon the ORTB [12] bid optimisation strategy; additionally TW focused on predicting the market price distribution for a given impression using GBDT and a Gaussian mixture model for the log market price.

3.5.1 Combined bidding strategy

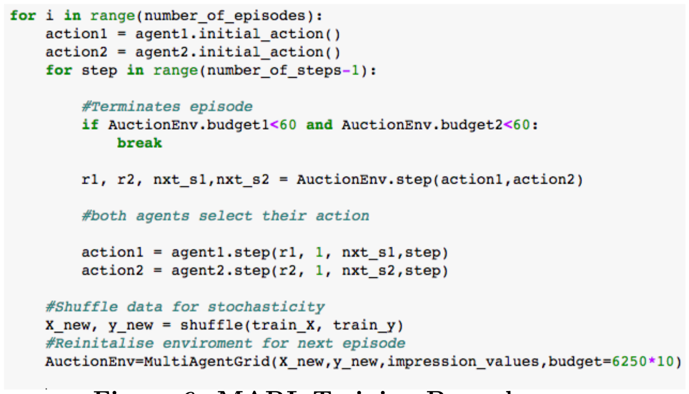
The first consideration was to develop a stronger CTR predictor. We tuned ensembles of the LR, GBDT and MLP models and found that both precision and recall could be improved. Producing an ensemble of multiple predictors is a common approach taken in machine learning, since the bias of one predictor can be offset by others and was something that we found used in the CTR prediction literature [7]. This improved CTR predictor was then used in the ORTB model, where we obtained 169 clicks (CTR = 0.001344) on the validation set, and 178 (CTR = 0.00141) on the test set.

Additionally, we ensembled the bid optimisation strategies at the bid level. This meant that we took a vector of bids from each model and combined them. We applied two approaches to combining the bids from strategies S_1, S_2 where λ is the interpolation parameter: (1) take a linear combination of each strategies' bids $bid = \lambda S_1(bid) + (1-\lambda)S_2(bid)$ and (2) randomly sample bids from S_1 with probability λ and from S_2 otherwise.

The linear combination of bidding strategies where S_1 was ORTB with a GBDT CTR predictor and S_2 was a linear bidding strategy with an MLP CTR predictor and $\lambda = 0.7$ produced the strongest results on the validation set out of any other strategy that we tried **mention number of clicks=? and CTR=?**.We submitted our strongest strategies to the test set API and at the time of writing (with less than 48 hours left of the competition), our bidding strategy is producing the second highest number of clicks within the class.

3.5.2 Multi-agent reinforcement learning

It was decided to create a multi-agent method with two agents. We created the single agent learning environment named FeatureGrid, in which we trained ACR's variant reinforcement learning agent; a modified version of DynaQ with linear approximation, to adjust its linear bidding strategy through a finite number of adjustments e.g. 50% bidding increase, rather than let the agent choose the bidding value directly. This was similar to the model Deep Reinforcement Learning to Bid (DRLB) [10], however in this model they had only considered the single-agent case using non-linear approximation, whereas we expanded on their work to the multi-agent scenario. Due to computation resources, the only major difference between our model and the original DRLB was we used linear approximation functions in place of their non-linear neural approximation functions, which later turned out to be detrimental to the low performance we observed, which is to be expected which replacing multiple highly non-linear functions with a function with exponentially smaller modelling capabilities. We also had to adapt ACR's single agent training procedure so that an epsisode terminates once both agents have processed the reduced impression matrix, or both agents have a budget of lower than 60 CNY, as shown (3.5.2). Please refer to ACR's report for a detailed description of the reinforcement learning algorithm used by each agent.



```
for i in range(number_of_episodes):
    action1 = agent1.initial_action()
    action2 = agent2.initial_action()
    for step in range(number_of_steps-1):

        #Terminates episode
        if AuctionEnv.budget1<60 and AuctionEnv.budget2<60:
            break

        r1, r2, nxt_s1,nxt_s2 = AuctionEnv.step(action1,action2)

        #both agents select their action

        action1 = agent1.step(r1, 1, nxt_s1,step)
        action2 = agent2.step(r2, 1, nxt_s2,step)

    #Shuffle data for stochasticity
    X_new, y_new = shuffle(train_X, train_y)
    #Reinitalise enviroment for next episode
    AuctionEnv=MultiAgentGrid(X_new,y_new,impression_values,budget=6250*10)
```

Figure 6: MARL Training Procedure

To convert our auction environment into a multi-agent one, we must first adapt FeatureGrid, into our new environment: MultiAgentGrid, in which each step from the environment will return a reward, next state for both agents. In our modelling we only chose two agents. However, it wouldn't be hard to add in further if felt deemed necessary. Please refer to ACR's report for a detailed description of the core reinforcement learning algorithm used by both agent, as well as environment construction. Agent1 and Agent2 used ACR's implementation of DRLB algorithm, however both had differing initial parameters as well as different action sets.

We kept Agent1 as the same settings as defined in ACR's report. We let Agent2 have a slightly different initial lambda $\lambda_0 = \frac{1}{80000}$ as well as a higher variance of adjustments available to it 2.

$$actions = [-0.75, -0.15, -0.08, 0, 0.08, 0.15, 0.75] \qquad (2)$$

.

We provide the training graph below for the multi-agent (3.5.2). We show the normalised reward, total wins and total clicks collected per episode.

We trained the multi-agents for 100 episodes of a dataset of size 200K randomly chosen examples. With a learning rate/ step size of 0.1. From inspecting the top left graph (3.5.2), we see that Agent1 had the highest normalised total reward throughout training, which we believe directly corresponded to its initial λ_0^{agent1} being closer to the optimal $\lambda*$, whereas λ_0^{agent2} had a deviation from the near optimal $\lambda*$ value of 10%, with a difference of their mean normalised total reward of 20%

In general, we observed when using the DRLB algorithm with linear function approximation instead of non-linear function approximation; our model's overall performance is highly determined by its initial lambda value. When pulling out the final action counts, interesting enough both agents converged to near identical action selection distributions. Both agents would choose action 1, which always reduced their bid by the most significant amount possible, with a probability of 0.225 and then select actions 2-7 uniformly with a probability of 0.129. Another potential obstacle for our multi-agent algorithm was that our adaptive learning rate

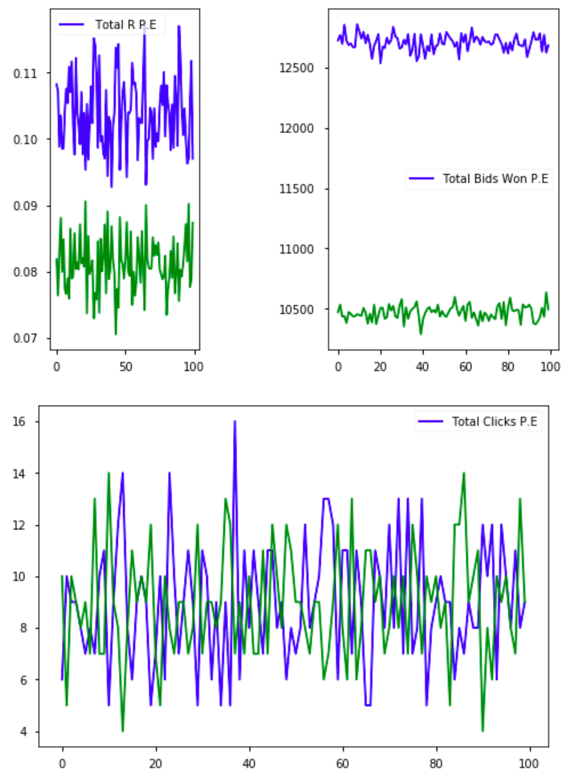


Figure 7: Multi Agent Learning Curves

hyper-parameter r was not set low enough, to allow sufficient exploration of the action state space.

4. CONCLUSION

4.1 Summary

In conclusion we have shown a variety of bid optimisation algorithms that were successful in maximising clicks across an unseen dataset with a fixed budget. We developed a strong baseline (the linear bidding strategy, which is commonly used in industry) and were able to exceed its performance through a variety of non-linear methods.

Furthermore, we explored the literature on reinforcement learning and multi-agent reinforcement learning approaches to RTB optimisation and were able lay a strong foundation for further development of these agents.

4.2 Future Work

In future we would like to focus solely on the multi-agent RL approaches to RTB optimisation. More specifically, we outline a few of the key components that we would consider next: (1) Optimise the adaptive epsilon greedy annealing rate to search for better convergence results; (2) Implement training process in TensorFlow with more efficient batching (to utilise GPUs and for faster training) (3) Extension of multi-agent learning to a much higher number of agents; (4) Deeper analysis on the behaviour of each model on a per-bid basis.

4.3 Contribution

All team members contributed equally towards the group aspects of this project.

ACKNOLEDGEMENTS

We would like to thank Ying Wen for his development of the test set API, which was very useful for when we wanted to assess the performance of our top-performing bid optimisation methods.

5. REFERENCES

[1] D. Agarwal, R. Agrawal, R. Khanna, and N. Kota. Estimating rates of rare events with multiple hierarchies through scalable log-linear models. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '10, pages 213–222, New York, NY, USA, 2010. ACM.

[2] Y. Bengio and Y. Grandvalet. Bias in estimating the variance of k-fold cross-validation. *Statistical Modeling and Analysis for Complex Data Problems*, page 75âĂŞ95, Apr 2004.

[3] H. Cai, K. Ren, W. Zhang, K. Malialis, J. Wang, Y. Yu, and D. 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.

[4] Y. Cui, R. Zhang, W. Li, and J. Mao. Bid landscape forecasting in online ad exchange marketplace. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 265–273. ACM, 2011.

[5] B. Edelman, M. Ostrovsky, and M. Schwarz. Internet advertising and the generalized second-price auction: Selling billions of dollars worth of keywords. *American economic review*, 97(1):242–259, 2007.

[6] M. V. D. Eeckhaut, T. Vanwalleghem, J. Poesen, G. Govers, G. Verstraeten, and L. Vandekerckhove. Prediction of landslide susceptibility using rare events logistic regression: A case-study in the flemish ardennes (belgium). *Geomorphology*, 76(3):392 – 410, 2006.

[7] X. He, J. Pan, O. Jin, T. Xu, B. Liu, T. Xu, Y. Shi, A. Atallah, R. Herbrich, S. 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*, pages 1–9. ACM, 2014.

[8] J. Jin, C. Song, H. Li, K. Gai, J. Wang, and W. Zhang. Real-time bidding with multi-agent reinforcement learning in display advertising. *arXiv preprint arXiv:1802.09756*, 2018.

[9] G. King and L. Zeng. Logistic regression in rare events data. *Political Analysis*, 9(2):137âĂŞ163, 2001.

[10] D. Wu, X. Chen, X. Yang, H. Wang, Q. Tan, X. Zhang, and K. Gai. Budget constrained bidding by model-free reinforcement learning in display advertising. *arXiv preprint arXiv:1802.08365*, 2018.

[11] J. Xu, X. Shao, J. Ma, K.-c. Lee, H. Qi, and Q. Lu. Lift-based bidding in ad selection. In *AAAI*, pages 651–657, 2016.

[12] W. Zhang, S. Yuan, and J. 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.

[13] W. Zhang, S. Yuan, J. Wang, and X. Shen. Real-time bidding benchmarking with ipinyou dataset. *arXiv preprint arXiv:1407.7073*, 2014.

6. APPENDIX

6.1 Feature Selection

- The following features were dropped (since they were unique, or near-unique across the training set) *"bidid", "userid", "IP", "url"*
- *"urlid"* was dropped, since it only took 1 possible value in the training set
- *"usertag"* was converted from strings of usertags, to 1-hot-encodings for each of the 69 possible user tags
- The following features were one-hot-encoded *"useragent", "region", "city", "adexchange", "domain", "slotid", "slotvisibility", "slotformat", "creative", "keypage", "usertag", "advertiser"*
- The following features were left unchanged: *weekday, hour, slotwidth, slotheight, slotprice*