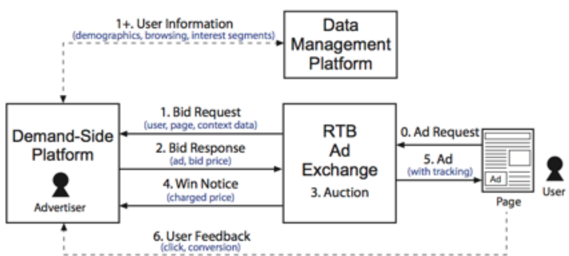
COMPGW02 Web Economics Individual Report (Group 05)

Kah Siong Tan
University College London
School of Computer Science
kah.tan.16@ucl.ac.uk

1. INTRODUCTION

Online advertising allows advertisers to bid and place their advertisements onto the web pages that are being visited by users. The process of user landing on a web page to the advertiser bidding and successfully displays their advertisements to the users takes place within a fraction of a second. Such real-time bidding based advertisements requires bidding strategies that would bring in the maximum profit over costs. Figure 1 shows an illustration of the interaction described above. This report proposed a bidding system consisting of two components, Factorisation Machines (FM) Click-Through-Rate(CTR) estimation model and a Threshold-Sigmoid (ST) bidding strategy. The former attempt to learn the chances of a click based on features collected from bid requests, while the latter formulate an approach to offer the best bid that capture essence of limited budget, capitalising on higher clicks chances, base bidding and bid scaling in non-linear manner. The strategy was implemented and evaluated against a linear bidding strategy. The results showed that FM CTR model performed better than the linear CTR model. The ST bidding strategy achieved a higher CTR with lower costs compared to linear bidding strategy.

Figure 1: Real-time advertisements bidding process



The report also discussed class imbalance and appropriate evaluation measures that were taken for the model during CTR estimation and optimal bidding. All source codes can be found in [9] committed under handle 'jax79sg' .

2. LITERATURE REVIEW

Prediction of CTR, or CTR estimation is a widely studied subject. There have been different models and variants developed over it. Some notable examples are Linear Regression (LR) [3], Recurrent Neural Networks (RNN) [5], Boosted Trees [16] and Factorisation Machines [12]. The most popular technique would LR for its simplicity and efficiency. However neural networks tend to provide surprisingly good results despite their difficulty in interpretation. Boosted trees provide very clear interpretation compared to neural networks and can determine its own feature importance. Factorisation Machines perform well on sparse data. It has been previously shown that higher order feature interactions could improve CTR estimation [3]. However, LR models are unable to capture feature interactions with an order higher than one without delicate feature engineering and a resulting explosion of the number of features. Factorisation Machines, first proposed by [13], have shown its ability to model higher order feature interactions with linear complexity. There has been prior research in CTR estimation using Factorisation Machines and its variants [15][8]. These models have been shown to outperform well known LR models. Several Factorisation Machines implementation have since surfaced. The prominent ones are libFM [14], FastFM [2] and Polylearn [11].

Class imbalance has been an inherent characteristic of online advertising [1] in that the ratio of user clicks is extremely small. Several novel approaches [6][10] have been proposed but under-sampling or over-sampling the dataset remains the popular approaches to this issue. Over-sampling would be useful if the number of the minority class in the dataset is too small for effective machine learning. Two commonly used algorithms for over-sampling are SMOTE [4] and AYSNA [7].

3. APPROACH AND RESULTS

3.1 Data Exploration

3.1.1 *Dataset interpretation*

Training, validation and test datasets are made available to the students. An analysis revealed the content of the datasets to be a mashup of logs from bid requests, bid responses, win notices and user feedbacks data exchanges in and out of DSP bidding agent indicated in Figure 1. Essentially, all three datasets referred to the same content type, except for test set which contains only information that is available prior to bidding. Another observation is that almost all the attributes in the datasets are categorical in nature. Due to space constraints, only certain attributes are summarised in Table 1.

3.1.2 *Data pre-processing*

With a better understanding of the datasets, a series of

Table 1: Interpretation of attributes

Attribute	Interpretation	Remarks
Useragent (Bid request)	String identifies OS and browser, separated by underscore.	Categorial. The OS and Browser will be separated into 2 different columns after data preprocessing.
Usertags (Bid request)	Integers represent segment of users, separated by commas.	Categorial. Only available in DSPâĂŹs proprietary database. Data pre-processing is required to split
Bidprice (Bid response)	Integer represent price bidded by DSP bidding agent.	Not available in test set.
Payprice (Win notice)	Integer represent market price, or second highest bid made in auction (Second price auction).	Not available in test set.
Click (Win notice)	Integer 0 or 1 representing a click or no click.	Categorial. Not available in test set.

sanity checks was performed on the datasets. Table 2 lists problems found and the recommended solutions.

Table 2: Problems on datasets and solutions

Check item	Problems	Solutions
Nulls in columns	Nulls detected in domain, keypage, adexchange, usertags.	Nulls treated as an additional categorical data for domain, keypage and usertag.
Bidprice < Payprice	3691 and 33579 such records found in validation and training set respectively.	Records donâĂŹt fit notion of a second price auction and were removed.
Discrepancies between training and validation sets.	useragent, region, city, adexchange, slotvisibility, slotformat and usertag columns have values thatâĂŹs not in both validation and test datasets. This will cause problems in prediction.	To ensure consistency, training and validation sets were appended and unique values of affected columns consolidated and used for training and predictions.

3.1.3 *Class imbalance*

Class imbalance were common occurrences in online advertisements. The datasets indeed showed significant imbalance of non-clicks against clicks. Out of more than 2.5 million impressions, only 0.07% of impressions were clicked. The low empirical figure of 1986 clicks could also posed a problem in learning the effective clicks. For most classification algorithms to perform, the number of samples of each class should preferably be around the same. The literature review mentioned several options to mitigate this issue. This issue will be revisited in later sections.

3.1.4 *Display advertising statistics*

3.1.4.1 *CTR, CPM, CPC.*

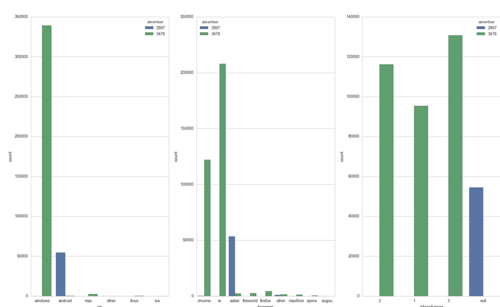
This section elaborate on metrics related to the display advertising. The CTR refers to the probability of clicks among all the impressions. The Cost-Per-Mile (CPM) refers to the approximate cost incurred for each 1000 impressions. The Cost-Per-Click (CPC) refers to the approximate incurred cost for each click. The computation draws from the dataset after the actions taken in Sections 3.1.1.

Table 3: : Display advertising statistics from training set

Advertiser	Clicks	Imp	Cost	CTR	CPM	CPC
1458	451	540293	37231239	0.0008	68909	82552
2259	45	146778	13649026	0.0003	92990	303311
2261	37	120619	10789152	0.0003	89448	291598
2821	144	231416	20625766	0.0006	89128	143234
2997	251	54487	3413227	0.004	62642	13598
3358	204	289982	24517382	0.0007	84547	120183
3386	358	498554	38341028	0.0007	76904	107097
3427	323	439787	33297891	0.0007	75713	103089
3476	173	342243	26328601	0.0005	76929	152188

From Table 3, it was obvious that advertiser 2997 has a leading edge in the CTR. Investigating further, advertiser 2997 was compared with an average performer 3476 in Figure 2.

Figure 2: Comparison of advertiser 2997 and 3476 in terms of OS, browser and ad-exchange



Advertiser 2297 focused entirely on android OS and safari browsers, it had also received bids from an exchange different from 3476. Although anyone of these could have resulted in the high CTR but it would likely be the android OS since its touch based and may have contributed to unintentional clicks.

3.1.4.2 *Bid, pay, reserved prices.*

Table 4: Statistics of pay price, bid price and reserved prices

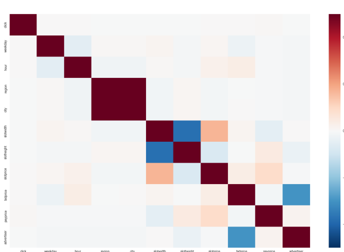
	Minimum	Mean	Maximum
Bid price	227	272.98	300
Pay price	0	78.14	300
Reserved price	0	26.73	300

Table 4 noted that there are reserved prices at 0 Fen, this means that if one do not wish to bid for an impression, a bid value of less than zero has to be submitted. Another interesting finding was that there were winning bids that paid nothing for it (pay price is zero). Reserved price typically start at around 26.73 on average but the bid price starts at a minimum of 227. This was surprisingly much higher than the average reserved price. Although the reason for this is unclear, this information could prove useful in the bidding strategy.

3.1.5 Features correlation

Finally, an analysis was performed on attributes to discover relationships between them. Figure 3 shows the following pairs of attributes inversely correlated at varying strengths, bid price-advertiser, slot height-slot price, slot width-pay price. The following displayed at positive correlation at varying strength, slot width-slot height, slot width-slot price. Second order interaction features could be useful to produce better CTR prediction models.

Figure 3: Correlation between attributes



3.2 Bidding Strategy

The bidding system is built on two components. CTR estimation to predict the probability of click for each impression, and optimal bidding to derive bid price based on the probability of click and other bid related parameters. Sections 3.2.1 to 3.2.3 discuss the approach on CTR estimation while sections 3.2.4 discuss on the optimal bidding strategy.

3.2.1 Model: Factorisation Machine (FM)

The FM model was explored as a non-linear model for CTR estimation. A second-order FM model was used meaning only second-order interactions between variables will be modelled. To explain the intuition of FM, one must understand Matrix Factorisation. In real number factorisation, it means decomposing the number into more numbers such that the product of these numbers equals the original number. In matrix factorisation, the same intuition applies. For example, a matrix of shape UxI can be decomposed into a UxK and a IxK matrix where $K << U$ and $K << I$. The interaction between the row and the column of the original matrix can be approximated by the dot product of a pair of vectors $row_i.col_j$. The equation for a multivariate linear regression is stated as follows.

$$y = w_0 + \sum_{i=1}^{n} w_i x_i \quad (1)$$

where y is the label, x_i are the features, w_0 and w_i are the weights to be trained. Now consider the polynomial situation where pairwise feature interactions are captured, the equation would be as follows.

$$y = w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} w_{ij} x_i x_j \quad (2)$$

where $x_i x_j$ refers to the interaction between features x_i and x_i, and w_0, w_i and w_{ij} refers to the weights to be trained respectively. The problem is now a non-linear problem and would be computationally too expensive for a sparse set of features. To solve this, FM imitate the workings of matrix factorisation and use the dot product of features interactions in modelling the feature interactions. The equation would be as follows.

$$y = w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} < v_i, v_j > x_i x_j \quad (3)$$

where $< v_i, v_j >$ is a vector that model the feature interaction. Equation 3 model a two-way interaction but can be extended to higher order interactions. Recall the K parameter in Matrix Factorisation, this would capture K hidden features for each feature interaction vector.

There are several implementations of Factorisation Machines. LibFM from the author of [13] was the defacto implementation to go to for FM. However, it was only available in C. Polylearn follows scikit-learn conventions closely and offered a wide range of tools from scikit-learn to be used, it was however plagued with long training times on my training set. FastFM claimed to follow scikit-learn interfaces but proved not to be so during my implementation. It was however clearly documented which allowed me to override methods to fit the evaluation architecture of our framework. FastFM support a variety of optimisers such as Alternating Least Squares (ALS), Markov Chain Monte Carlo (MCMC) and Stochastic Gradient Descent (SGD) to solve for the optimal values of $< v_i, v_j >$ vectors for each pair of features x_i and x_j. For this assignment, FastFM is chosen to be used as the baseline for developing my codes upon. The predict and predict_proba methods were overwritten while the fit method was retained. The SGD solver was chosen mainly for my familiarity with its tuning. Wrapper classes were also written specifically to support the evaluator classes which the team has written.

3.2.2 Data

3.2.2.1 Features.

Inherently, FM do not have the capability to compute importance of each attribute as would Boosted Tree algorithms. Yet, learning interactions between features is the main draw of the algorithm. Unfortunately, this implied that the successful performance of FM depends heavily on careful feature selection. As an example, it can probably learn that the probability of click will be very high when the impression came from a bid request where region is 276 and city is 282. But if one of these features were not considered in training, the opportunity to bank on this information would be lost. After consideration, only the following features were removed from the datasets.

Table 5: Features removed from training

Feature unused	Reason
logtype	Only consists of 1s, making it useless.
bidprice	Not available in test set
payprice	Not available in test set
click	Label to predict

3.2.2.2 Class Imbalance.

Data sets from online advertising tend to be class imbalanced. In the dataset, it was shown that the percentage of clicks to the total number of impression is 0.07%. There were several approaches including selecting evaluation metrics that are less impacted by class imbalance, or by under-sampling or over-sampling data. With over two million impressions, simply down-sampling would have been a perfect choice for its simplicity if not for the extreme class imbalance of only 1986 impressions with clicks. To mitigate the issue, the non-clicks in the dataset was first randomly down-sampled to a ratio of 8:2 to the all clicks impression. Thereafter an oversampling SMOTE algorithm was applied to generate synthetic impressions such that the ratio between clicks and non-clicks are balanced out.

3.2.3 Training

3.2.3.1 Metrics.

Accuracy, precision, recall and F1 are considered, but their values varies across thresholds and metrics such as accuracy do not hold well in extreme class imbalance scenarios. For example, if 99.9% of impressions are non-clicks, and the model predict non-clicks even for actual clicks, the accuracy will still be high. One class imbalance insensitive metric is Area Under the Curve (AUC) of Receiver Operating Characteristic (ROC). The AUC is also independent of the threshold that decides if a prediction is a positive or negative. Finally, AUC works for any binary classification models that produce the relevant probabilities. In this assignment, the unique requirements of computational advertising may mean that different thresholds may be set for different models. With above in mind, the AUC was used as the main metric for CTR estimation evaluation.

3.2.3.2 Training parameters.

There are a few hyper-parameters that needs to be set. The most important parameter here would be the rank. Effectively, higher rank means more 'hidden' features can be learnt and it should contribute to a better AUC. Experimentation on higher ranks resulted in slightly longer training time though. Interestingly the AUC dropped after a rank of 32 and above was set, this is because of the larger number of features. Generally, less iterations is possible with bigger step size. The optimal set of parameters were chosen via a searchgrid optimising for highest AUC.

Table 6: Training parameters

Type	Hyper-parameter	Remarks
Model	Rank (16)	The K in the matrix factorisation
	L2 regularisation linear components (0.005)	Regularisation on weight of w_i
	L2 regularisation pairwise components (0.005)	A regularisation on weight of $< V_i, V_j >$
SGD*	Max Iterations (200000)	
	Step size (0.01)	

* Focused on SGD, thus ALS MCMC not indicated in report.

3.2.3.3 Thresholding.

The problem of computational advertising adds a domain specific consideration in that clicks could be considered as one of the contributing factors to the Return Of Investment of the bidding process. The goal was to be able to bid for a lucrative slot and from there generate profits in the form of follow up purchases by the user, which in turn be generated into revenue for the slot bidder. This means that it may be necessary to capture all the clicks and bid for it than to miss it. As such, the threshold for determining a potential click could be shifted. This understanding would be useful in determining the optimal bidding strategy later.

3.2.4 Optimal Bidding: Sigmoid-Threshold (ST)

For a successful bidding, a bid must be calculated such that it would be above the market price (2nd price). A linear bidding strategy was proposed in the assignment brief.

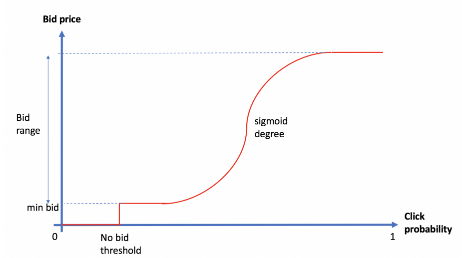
$$bid = basebid * (pCTR/avgCTR) \tag{4}$$

where the tuning parameter *basebid* is the bidprice for the average CTR cases. With the $pCTR$ computed in the CTR estimation model, the next step is to find the optimal basebid that generates a bid for every impression that optimise the CTR and click metrics given the limited budget. It was noted in data exploration that the minimum bid for all the impressions was 227. Although the analysis did not capture the reason, this information was strongly considered for the bidding strategy. A variation of linear bidding would be to capture the essence of limited budget, capitalising on higher chances of clicks (Thresholding), bidding from a base and scaling the bid along the probabilities of clicks in a non-linear manner. This means that lower confidence bid requests are ignored, and a cap on the highest bid per request to ensure high confidence bids are treated similarly and does not exhaust the budget quickly. Figure 4 exhibits the idea, which will be termed as Sigmoid-Threshold (ST) bidding strategy in this report. The bid function would look like

$$Bid = (1/(1 + e^{(\theta*(Pr(click=1)) + (-0.2-T))})) * R + m \tag{5}$$

where, θ is a hyper-parameter, $Pr(click = 1)$ is the probability of a click, T is the threshold of $Pr(click = 1)$ to start

Figure 4: Visualisation of Sigmoid-Threshold bidding strategy



bidding, R is the maximum bidding range and m is the minimum bid. Note that the equation doesn't consider a zero bid for below T. This would be handled in the software implementation. θ, R, T and m are optimised by iteratively running the algorithm with different combinations of the parameters (Grid Search). The target of optimisation was to score against number of clicks and CTR.

3.2.5 *Evaluation*

The FM and Logistic Regression (LR) CTR estimation models are trained on my training set and then evaluated based on the validation set. The two models are then compared by AUC. Figure 5 show that FM performed better in terms of AUC. Moreover, an analysis of predicted probabilities for click and non-clicks of each model showed that FM showed a better distinction between a click and a non-click. This would also offer stronger outcomes if threshold is performed.

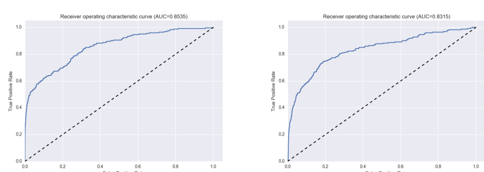


Figure 5: Left: AUC for FM, Right AUC for LR

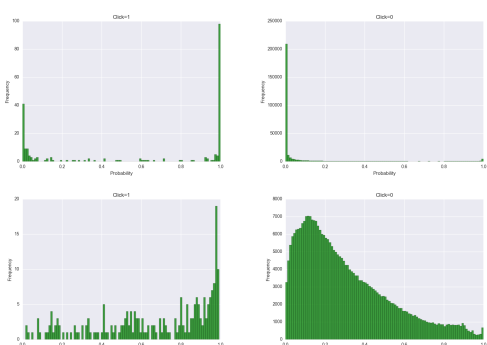


Figure 6: Upper left: FM predicted probabilities for clicks. Lower left: LR predicted probabilities for clicks. Upper right: FM predicted probabilities for non-clicks. Lower right: LR predicted probabilities for non-clicks

Both the Linear(based on average CTR of 0.07% on validation set) and ST bidding strategies were optimised for their parameters for best CTR and number of clicks on the FM CTR estimator and compared.

Table 7: Comparing advertising metrics between Sigmoid-Threshold (ST) and Linear (L) bidding strategies.

Model	CTR	Spend	Clicks	CPM	CPC
ST	0.0036	3088940	140	79458.26	22063.86
L	0.0015	6249815	138	66192.34	45288.51

Table 7 show ST bidding strategy achieved a higher CTR rate with a significantly lower spend. In terms of clicks, there wasn't much difference. The lower cost is attributed to the ST bidding function which places a limit on the maximum confident bid on each bid request, making all high confident bids to bid a same price. The linear bidding exhausted its budget before the campaign ends, in contrast the ST bidding sustained throughout the campaign. This is important when budget is constrained and the model is to be used online without knowing how many impressions will arrive.

4. CONCLUSION

The FM model has been shown to be an effective solution to binary classification problems despite being less easy to interpret. Further improvement can be done by exploring higher level of feature interactions. The ST bidding strategy worked as envisioned, but more can be done to explore more complex strategies.

4.1 Roles

Table 8: Member mainly involved in item.

	KS	CS	Min
Data Read Write		X	
Feature extraction	X	X	X
Imbalance Resampling	X	X	
Constant Bidding		X	
Random Bidding (Gaussian)		X	
Random Bidding (Uniform)			X
Logistic Regression	X	X	X
XGBoost		X	
CNN			X
Factorisation Machine variants	X		
Bidding Strategies	X	X	X
Ensemble (CTR models)	X	X	X
Evaluator class for Bids	X	X	
Evaluator class for Clicks		X	X

The team composed of Kah Siong (KS), Chun Siong (CS) and Min Ying (Min). We worked by picking up items to work on based on our progress and workload. Table 8 is based on member ownership in specific aspects of the listed items, however other members do enhance existing codes to add new features or or enhance performance. The group report was collectively written, with some aspects of it being ported from the individual reports.

References

[1] Josh Attenberg and Foster Provost. 2010. Why Label when you can Search? Alternatives to Active Learning for Applying Human Resources to Build Classification Models Under Extreme Class Imbalance. In *KDD '10 Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, Vol. 17. 1–5.

[2] Immanuel Bayer. 2016. fastFM: A Library for Factorization Machines. In *Journal of Machine Learning Research*, Vol. 17. 1–5.

[3] O. Chapelle, E. Manavoglu, and R. Rosales. 2014. Simple and scalable response prediction for display advertising. In *ACM Trans. Intell. Syst. Technol*, Vol. 5. 61–1–61–34.

[4] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. 2002. SMOTE: Synthetic Minority Over-sampling Technique. In *Journal of Artificial Intelligence Research*, Vol. 16. 321–357.

[5] Qiao-Hong Chen, Shi-Min Yu, Zi-Xuan Guo, and Yu-Bo Jia. 2016. Estimating AdsâĂŹ Click through Rate with Recurrent Neural Network. In *ITM Web of Conferences*, Vol. 17. 1–5.

[6] Xinjian Guo, Yilong Yin, Cailing Dong, Gongping Yang, and Guangtong Zhou. 2008. On the Class Imbalance Problem. In *Fourth International Conference on Natural Computation.*

[7] Haibo He, Yang Bai, Edwardo A. Garcia, and Shutao Li. 2008. ADASYN: Adaptive synthetic sampling approach for imbalanced learning. In *Neural Networks, 2008. IJCNN 2008. (IEEE World Congress on Computational Intelligence). IEEE International Joint Conference on*, Vol. 17. 1–5.

[8] Yuchin Juan, Yong Zhuang, Wei-Sheng Chin, and Chih-Jen Lin. 2016. Field-aware Factorization Machines for CTR Prediction. In *RecSys '16 Proceedings of the 10th ACM Conference on Recommender Systems.* 43–50.

[9] Min Ying Ong Kah Siong Tan, Chun Siong Poh. 2017. Web Economics assignment Group 5. (Apr 2017). https://github.com/jax79sg/webecon2017

[10] Bartosz Krawczyk. 2016. Learning from imbalanced data: open challenges and future directions. *Springer Link* 5, 4 (Nov 2016).

[11] Vlad Niculae. 2016. A library for factorization machines and polynomial networks for classification and regression in Python. (2016). Retrieved March 6, 2017 from https://github.com/scikit-learn-contrib/polylearn/

[12] Zhen Pan, Enhong Chen, Qi Liu, Tong Xu, Haiping Ma, and Hongjie Lin. 2016. Sparse Factorization Machines for Click-through Rate Prediction. In *2016 IEEE 16th International Conference on Data Mining.*

[13] Steffen Rendle. 2010. Factorization Machines. In *Data Mining (ICDM), 2010 IEEE 10th International Conference.* 995–1000.

[14] STEFFEN RENDLE. 2012. Factorization Machines with libFM. In *ACM Trans. Intell. Syst. Technol*, Vol. 3.

[15] Anh-Phuong TA. 2015. Factorization Machines with Follow-The-Regularized-Leader for CTR prediction in Display Advertising. In *Big Data (Big Data), 2015 IEEE International Conference on.*

[16] Ilya Trofimov, Anna Kornetova, and Valery Topinskiy. 17. *Using boosted trees for click-through rate prediction for sponsored search.* Technical Report. Yandex.