國立臺灣大學電機資訊學院電機工程學研究所

博士論文

Graduate Institute of Electrical Engineering

College of Electrical Engineering and Computer Science

National Taiwan University

Doctoral Dissertation

在即時競標中使用設限資料預測可獲勝的價格

Predicting Winning Price in Real Time Bidding with Censored Data

吳齊軒

Wush Chi-Hsuan Wu

指導教授：陳銘憲教授

Advisor: Ming-Syan Chen, Professor

中華民國 108 年 1 月

Jan, 2019

doi:10.6342/NTU201900270

doi:10.6342/NTU201900270

誌謝

首先感謝我的指導老師陳銘憲教授以及葉彌妍老師。老師們的指導使我得以找到自己研究的方向與興趣，並且有機會嘗試自己找到的研究方向。而不論實驗的成功與失敗，老師也都不會生氣，只是繼續督促學生進行研究。也感謝老師們的協助，讓我有多次機會出國開會交流，了解世界的深與大，以及學術工作的美好之處。

接著我要感謝實驗室的夥伴們。兆殷和玉芬學姊在實驗室中都幫過我不少忙，我是非常感謝的。立言則常常與我交流過研究上的想法，給了我很多幫助。許多和冠叡與寓鈞聊天的時候，也讓我學到與領悟不少研究與人生的看法。

再來我想要感謝在宇匯知識科技認識的培林、偉平、昌祥。沒有你們的協助，我沒辦法找到這個有趣的研究題材。也感謝你們不吝於分享網路廣告業的知識與研究心得。

最後我要感謝我的家人。謝謝親愛的妻子，幫我分擔許多照顧兒子的責任，讓我在喜獲麟兒之後仍能完成博士學位。謝謝爸爸媽媽的支持，讓我在攻讀博士的過程中，從來沒有後顧之憂。謝謝可愛又天使兒子，每天都早睡晚起才能讓我找到時間做研究。

doi:10.6342/NTU201900270

doi:10.6342/NTU201900270

摘要

在網路廣告產業，近年來業界普遍地使用即時競標的方式來交易廣告的播放機會。在這篇論文中，我們站在買方的立場，研究如何使用機器學習與統計方法來從過去的交易紀錄中預測每個競標的獲勝價格。買方在這個問題的主要挑戰是，若競標落敗後，將無法得知獲勝價格。因此在參考倖存分析與經濟學中常用的受限迴歸模型後，我們設計出針對落敗資料的損失函數。我們更進一步發現即時競標的資料並不符合受限迴歸模型的假設。因此我們運用獲勝機率來設計一般迴歸與受限迴歸的混和模型，以降低因不符假設而帶來的誤差。我們更進一步地運用深度學習技術與多種統計分布來推廣原本的模型，並且仍然讓模型能同時從獲勝與落敗的歷史紀錄中學習。我們研究那些在點擊率預測中獲得成功的模型是否也能改善獲勝價格的預測。實驗結果顯示，如果只看獲勝的資料，深度學習的模型確實改善了預測的精準度。而深度學習模型在落敗資料上的表現，也可以透過從落敗的資料上學習來改善。最後，我們再研究將混和模型的技術運用在深度學習的模型上。

doi:10.6342/NTU201900270

doi:10.6342/NTU201900270

Abstract

Real-Time Bidding is currently the most popular ad auction process for online advertising. In this study, we study how to predict the winning price of each bid from the aspect of a bidder by leveraging the machine learning and statistical methods on the bidding history. A major challenge is that the real winning price is not observed by the bidder after losing. We propose to utilize the idea from censored regression model, which is widely used in the survival analysis and econometrics, to derive the loss for the losing data. Moreover, the assumption of the censored regression is violated in the real data, so we propose a model which uses the winning rate prediction to mitigate the impact of violation. It is named as the mixture model. Furthermore, We generalize the winning price model to incorporate the deep learning models with different distributions and propose an algorithm to learn from the historical bidding information, where the winning price are either observed or partially observed. We study if the successful deep learning models of the click-through rate can enhance the prediction of the winning price or not. We also study how different distributions of winning price can affect the learning results. Experiment results show that the censored regression usually outperforms the linear regression and the proposed averaged model always outperforms the linear regression. Experiment results also show that the deep learning models indeed boost the prediction quality when they are learned on the historical observed data. In addition, the deep learning models on the unobserved data are improved after learning from the censored data. Finally, we

doi:10.6342/NTU201900270

study the combination of the mixture model and the deep learning model.

doi:10.6342/NTU201900270

Contents

doi:10.6342/NTU201900270

doi:10.6342/NTU201900270

List of Figures

doi:10.6342/NTU201900270

doi:10.6342/NTU201900270

List of Tables

doi:10.6342/NTU201900270

doi:10.6342/NTU201900270

Chapter 1

Introduction

The revenues of the internet advertising in the United States totaled \$88.0 billion for the full year of 2017 [11]. The programmatic advertising aggregates the audiences from disparate sites and targets the relevant end users more effectively. Due to vast traffics of ads, many applications in this field use machine learning algorithm to optimize the Key Performance Indicator (KPI) such as the Click-Through Rate (CTR) and the ConVersion Rate (CVR). It is undoubtedly that the programmatic advertising is a major field of applied machine learning.

To increase the profit for publishers and reach for the advertisers, the market works hard to decrease the term of exchanging ads from months to single event. Finally, the Real-Time Bidding (RTB) becomes one of the major technology of the internet advertising [8]. In RTB display advertising, an impression, which is the opportunity to display the ad, is sold by an auction held by the ad exchange and the advertisers bid the impression with the help from the agents, which are called the Demand Side Platform (DSP). Currently, the major rule of the auction is the second price auction, which is first introduced in [15]. If the DSP bids the highest bidding price, then it wins the impression and pays the second highest bidding price to the ad exchange. Please refer to Yuan et al. [22] or Wang et al. [16] for more details about RTB display advertising.

There are three common paying models for the advertiser: the cost-per-mile (CPM), cost-per-click (CPC) and cost-per-action (CPA). The advertisers pay for every impressions in the CPM model, for every click in the CPC model and for every conversion or prede-

doi:10.6342/NTU201900270

fined actions in the CPA model. Therefore, the DSPs need to predict the expected income of impressions according to the paying model and compute the bidding price accordingly in real time. These are the main challenges of the DSPs.

In this dissertation, we study the prediction of the winning price, which is defined as the lowest price to win, in the second price auction from the aspect of a certain DSP. Note that the winning price is usually the same as the cost of winning the bid, which plays an important role in computing the bidding price. Here is an example of the winning price. Suppose there are four DSPs, A, B, C, and D, competing a specific impression. The bids offered by A, B, C, and D are 50, 100, 150, and 200, respectively. Then, the winning prices of A, B and C are all 200 because the highest price from their competitors is 200, which is offered by D. On the other hand, the winning price of the D is 150 because the highest price from its competitors is 150, which is from C.

Bidding the trustful value of the impression is the dominant strategy to maximize the profit if the budget and time are both unlimited [2]. In practice, the best strategy should consider the practical constraints, so the best strategy needs more information to compute such as the winning rate and the cost of winning. For example, the winning rate and the cost of winning a bid are used in Zhang et al. [25] to derive the optimal bidding strategy. Lin et al. [13] improved the bidding strategy by incorporating the CTR predictor with a winning price predictor.

One can model the winning price as the traditional supervised learning problem. However, Ghost et al. [7] pointed out that there is the *partial observable exchange* for the DSPs. More specifically, the winning price is usually only observed by the winner of the auction. The losers only know that the winning price is higher than their bidding price. This kind of partially observed winning price is called *censored*, and is studied in many fields such as survival analysis [12] and econometrics [14]. We call the censored bidding information as the losing data, and the winning bidding information as the winning data.

In [21], we introduced the idea of using winning probability as the loss function to learn the winning price from the losing data, and the mixture model which consistently outperforms the model from the winning data only. In [20], we extended the model in two

doi:10.6342/NTU201900270

directions. The structure of the expected value can be switched to successful deep learning models in related field such as the model proposed by Cheng et al. [5] and the model proposed by Wang et al. [17]. The distribution of the error which is closely related to the loss function can be switch to non-normal distribution such as log-normal distribution or gumbel distribution.

In this dissertation, we revised the theory of the mixture model proposed by [21]. After the revision, the new mixture model consistently outperforms the original mixture model. Then we conduct the experiments of combining different models from [20].

The dissertation is organized as follows. Chapter 2 introduces the mechanism of the RTB display advertising related to the winning price and the problem definition. The related works are in chapter 3. Models of the winning price and the algorithm to fit the models are introduced in chapter 4 and chapter 5 respectively. Then we present our experiments in chapter 6. The conclusion and the related work are in chapter 7.

doi:10.6342/NTU201900270

doi:10.6342/NTU201900270

Chapter 2

The Winning Price of the RTB Display Advertising

In this chapter, we describe the mechanism of modern RTB and discuss the definition and the availability of the winning price in different cases of bidding result first. These rules show the challenge of modeling winning price. Then we define our problem.

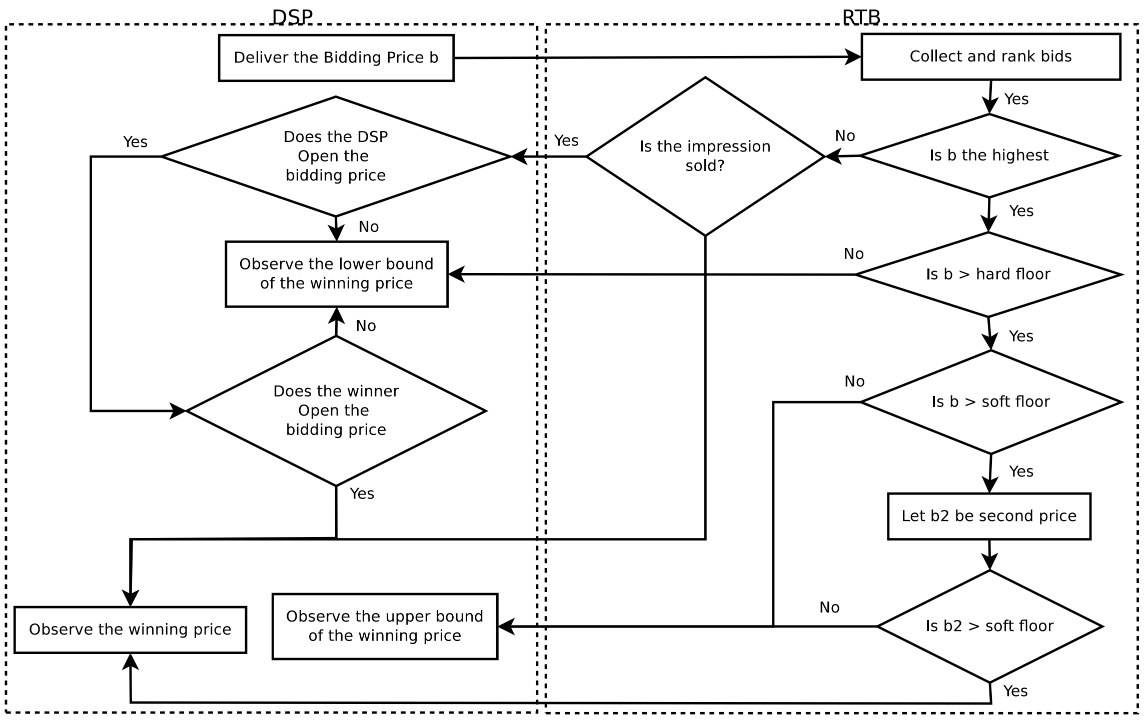
DSP
RTB
Deliver the Bidding Price b
Collect and rank bids
Yes
Is b the highest
No
Is the impression sold?
Yes
Does the DSP Open the bidding price
Yes
No
Observe the lower bound of the winning price
Yes
Is b > hard floor
No
Yes
No
Is b > soft floor
No
Does the winner Open the bidding price
Yes
Yes
Let b2 be second price
Observe the winning price
Observe the upper bound of the winning price
No
Is b2 > soft floor
Yes

Figure 2.1: A flow chart of auction process and winning price censoring.

In the modern display advertising market, the RTBs provide a trading platform to let

doi:10.6342/NTU201900270

publishers and DSPs trade with wider partners. Therefore, the publishers sell the impression more efficiently and the DSPs and advertisers raise their reach.

The auction process is shown in Fig. 2.1. Modern RTBs grant the option that the publishers may set a soft floor price and the hard floor price. Therefore, the auction is not a pure second price auction.

We discuss the cases and the related winning price here. Without loss of generality, suppose we are one of the DSP and our bidding price is b to a specific ad impression. After receiving the bidding price from the DSPs, the RTBs rank these bids according to the bidding price and check whether the ad impression is sold or not. When the first price is smaller than the hard floor price, the ad impression is unsold. In this case, the winning price is the hard floor price because it is the lowest price to win the impression. On the other hand, if the first price exceeds the hard floor price, then the ad impression is sold to the DSP which bid with the first price. If we are the only one whose bidding price exceeds the hard floor price, then the winning price is still the hard floor price. Otherwise, the winning price is the highest bidding price from the competitors. It shows that the winning price is only affected by the competitors and the first floor price.

The cases of the information censoring are discussed here. If we win the ad impression, then the availability of the winning price is related to the soft floor price and the true winning price, named c in this paragraph. If c is higher than the soft floor price, then the RTB charges us c so the winning price is available. If c is smaller, then the RTB charges us b and the winning price is left censored, which represents that we only know the upper bound of the winning price. If we lose the ad impression, the winning price is generally right censored. Some RTBs grant options of making bidding price public to DSPs. If we make our bidding price public and the winner also makes the bidding price public, then the winning price becomes available.

In general, the cases of information censoring in practice are more complicated than pure second price auction. However, the main difference is due to the soft floor price and the option of publicity of bidding price. In our data, the ratio of the case different from second price auction is small.

doi:10.6342/NTU201900270

In this dissertation, we mainly study how to predict the winning price based on the historical bid information the DSP observed. More specifically, we assume that winning price of the losing dataset are all right censored and of the winning dataset are all observed.

doi:10.6342/NTU201900270

doi:10.6342/NTU201900270

Chapter 3

Related Work

We review the previous work of studying winning price in the real-time bidding, mainly at the DSP side, in this chapter. Many of them are mainly studying bidding strategy.

Ghosh et al. proposed adaptive bidding algorithms under a fixed budget with user-specified constraints in [7]. They assumed that the winning price was drawn i.i.d. from a specific pdf which is over simplified.

Cui et al. studied the prediction of the winning price and modeled it with the mixture-of-log-normal distribution on various targeting attributes in [6]. They used the gradient boosted decision tree and log-normal distribution to model the winning price. However, they are on the seller side, so there is no discussion of censored data. In this dissertation, we evaluate the log-normal distribution with our proposed algorithm in the experiments.

Zhang et al. studied the bidding strategy and the cost, which is usually the same as the winning price, is the input in [24]. For simplification, they assumed that the cost is the bidding price.

Wang et al. used the non-parametric distribution to model the winning price in [18]. They used the decision tree to cluster the feature vectors and obtained the non-parametric distribution of the winning price of each clusters. The survival functions were introduced to handle the censoring issue. The main issue of their method is the scalability due to the clustering. On the other hand, the regression based methods are well studied and widely used in the industry for predicting the click-through rate or the conversion rate.

Zhu et al. used the censored linear regression, exponential link function and the gamma

doi:10.6342/NTU201900270

distribution to model the winning price in [26]. They give an empirical analysis to show that the winning price of the iPinYou dataset is more likely to be gamma distributed. Finally, they divide the estimation problem into two sub-problems.

We started to study the censoring of the winning price in the side of the ad impression buyer in [21]. We combined the censored regression model, linear regression model and winning rate mixture model to predict the winning price. Then in [20], we study how to improve the model with and without censored data by the selection of the link structure and the distribution. We show that the link structure linear and the normal distribution is not the best model in our experiments. In this dissertation, we combine the results of [21] and [20]. More specifically, we fit the mixture model according to deep learning models. Furthermore, we revised the mixture model and make them more feasible in practice.

doi:10.6342/NTU201900270

Chapter 4

The Models of the Winning Price

In this chapter, we introduces our proposed models of the winning price.

4.1 Winning Price Model

Without loss of generality, we label the bids with $1, 2, ..., N$. The labels are split into two groups: the labels of winning data $\mathbb{W}$ and the labels of losing data $\mathbb{L}$. We denote the winning price of the i-th bid is w_i.

Ideally, the winning price is the highest bidding price from other competitors. Therefore, the mathematical definition of the winning price is

$$w_i = \max\left\{b_i^2, b_i^3, ..., b_i^M\right\}. \tag{4.1}$$

The b_i^k denotes the bidding price of the i-th bid from the k-th competitor. Note that the floor price discussed in Chapter 2 can be viewed as a competitor who always bids the same price as the floor price.

However, we cannot directly observe the bidding price from the competitors in practice. Therefore, we use a statistical model to approximate the winning price. We denote the vectorized feature of the bid as x_i which will be introduced in the next paragraph. We assume that the conditional distribution of the winning price w_i given x_i is normal distributed and the mean is $g(x_i)$. For simplicity, we assume that the function f is linear.

doi:10.6342/NTU201900270

Therefore, the approximated winning price model is:

$$w_i \sim \mathcal{N}\left(\beta^T x_i, \sigma^2\right). \tag{4.2}$$

This is the linear regression model. The notation $\mathcal{N}$ represents the normal distribution and the two slots which follow the notation $\mathcal{N}$ is the mean and variance σ^2 respectively. The $\beta^T x_i$ is the inner product between the regression coefficient β and the feature vector x_i.

There are two sources of the features. One is from the ad exchange system. In practice, the ad exchange system broadcasts the information related to the publishers such as the domain name, the format of the required ad, and the visibility. The other one is from the DSP. The ad exchange system passes the identity of the audience to the DSPs so that the DSPs can use the browsing history or other characteristic of the audience which are collected by the DSP itself or other collaborated data management platforms. For convenience, we assume that the x_i is the numerical vector extracted from these features. The main difference between the winning price model and the CTR model is that the x_i used by the winning price model does not include the features of the ad contents such as the campaign id and the creative id. The reason is that the competitors do not know the ad we are going to display to the audience. More description of the x_i in RTB will be given in Sec. 6.1.

According to the model of Eq. 4.2, the loss function of the i-th bid is defined as follow. If $i \in \mathbb{W}$, then the loss is the negative log likelihood of the normal distribution $\mathcal{N}(\beta^T x_i, \sigma^2)$ which is the squared loss:

$$\mathcal{L}_w(\beta, \sigma^2 | w_i, x_i) = \frac{(w_i - \beta^T x_i)^2}{2\sigma^2} + \frac{\log(2\pi\sigma^2)}{2}. \tag{4.3}$$

If $i \in \mathbb{L}$, then the w_i is unobserved. However, we use the fact the the winning price is larger than our bidding price to derive the loss. Suppose the b_i is our bidding price, we use the negative log of the probability of $w_i > b_i$ as the loss function:

$$\mathcal{L}_l(\beta, \sigma^2 | b_i, x_i) = -\log\left(1 - \Phi\left(\frac{b_i - \beta^T x_i}{\sigma^2}\right)\right), \tag{4.4}$$

doi:10.6342/NTU201900270

where Φ is the cumulative density function (cdf) of the standard normal distribution.

We further name the models based on the data they learn from. The winning model, which corresponds to the parameters β_w and σ_w^2, is learned from the winning data only. We denote its prediction as $\hat{w}_i^w$. If the $w_i \sim \mathcal{N}\left(\beta_w^T x_i, \sigma_w^2\right)$, then $\beta_w^T x_i$ will minimize the expected mean squared error and the mean absolute error. Therefore, we will use $\hat{w}_i^w = \beta_w^T x_i$ as the point estimation. The full model, which corresponds to the parameters β_f and σ_f^2, is learned from the full data including the winning data and the losing data. Similarly, We denote its prediction as $\hat{w}_i^f = \beta_f^T x_i$. The full model is also called as the *censored regression* model. Note that we cannot learn the model from the losing data only because the solution of the loss Eq. 4.4 will tend to ∞.

4.2 Generalized Winning Price Model

The model in Eq. 4.2 is extendable.

First, the function g that links the features x_i and the expected value of the w_i can be non-linear. More specifically, We let g be a network structure of the deep learning model. In [20], we called it as the *link structure*. We still denote it as g and use β to represent the set of parameters of the network structure. For example, the linear regression model introduced in Sec. 4.1 is a special case that the $g(x_i|\beta)$ is $\beta^T x_i$, which is a single layer network. We will introduce the link structures we used in Sec. 4.2.1.

Second, the loss functions of winning data and losing data are generalized based on the maximal likelihood principle [1]. Suppose $f_\Theta(\cdot|g(x_i|\beta))$ and $\mathcal{F}_\Theta(\cdot|g(x_i|\beta))$ are the probability density function (pdf) and the cumulative density function (cdf) of the winning price w_i given x_i respectively. The loss function for the winning data becomes:

$$\mathcal{L}_w(\beta, \Theta|w_i, x_i) = -\log\left(f_\Theta(w_i|g(x_i|\beta))\right), \tag{4.5}$$

and for the losing data becomes:

$$\mathcal{L}_l(\beta, \Theta|b_i, x_i) = -\log\left(1 - \mathcal{F}_\Theta(b_i|g(x_i|\beta))\right), \tag{4.6}$$

doi:10.6342/NTU201900270

where $g(x_i|\beta)$ is related to the location parameter and Θ is the set of other parameters of the family of the conditional distribution. For example, if the conditional distribution of the winning price w_i given x_i is normal distribution, then the loss functions of winning data is Eq. 4.3 and of losing data is Eq. 4.4, and the $g(x_i|\beta)$ is the mean and Θ is $\{\sigma^2\}$.

In Fig. 4.1, we show the components of the proposed generalized winning price model. We will introduce the studied link structures and distributions in the following paragraph.

4.2.1 Link Structures

The studied link structures in our experiments are linear, wide, deep, cross, $\mathrm{wide\_and\_deep}$ and $\mathrm{cross\_and\_deep}$. The wide, deep and $\mathrm{wide\_and\_deep}$ models are studied by Cheng et al. [5] and the cross, deep and $\mathrm{cross\_and\_deep}$ models are studied by Wang et al. [17] for the CTR prediction.

The linear structure is $g_{linear}(x_i) = x_i^T\beta$.

The wide structure is the same as the linear structure in essence. The difference is that we do cross product transformations before vectorizing the raw features into vector x_i. Given the raw features, we iterate all two combinations of the feature to generate the new 2-level interaction of these features. The value of the generated features is the product of the source features. For example, given $\mathrm{a{:}1,b{:}2,c{:}3,d{:}4}$, then the generated features are $\mathrm{ab{:}2,ac{:}3,ad{:}4,bc{:}6,bd{:}8,cd{:}12}$. After generation, we vectorize the generated feature to obtain x_i and use the same link structure $g_{wide}(x_i) = x_i^T\beta$. The term wide is from [5].

The wide structure is widely used in the field of statistics. The difference between the wide structure and the linear structure in practice is whether the feature affects the $g(x_i)$ in a fixed way. For example, if the first entry of x_i increases 1, the variation of $g_{linear}(x_i)$ is fixed and irrelevant to the other entries. However, the variation of $g_{wide}(x_i)$ depends on other entries. Following the example of wide, if the a increases 1, then many generated features are changed and the changing is affected by other entries. The ab will increase the value of b, and the ac will increase the value of c. Therefore, the variation of $g_{wide}(x_i)$ after changing one entry depends on other entries.

The deep structure is an embedding layer and several layers of dense neuron network.

doi:10.6342/NTU201900270

Given the $x_i \in \mathbb{R}^p$, the parameter of the embedding layer is a matrix $W \in \mathbb{R}^{k \times p}$. k is the length of the embedding vector and p is the dimension of the feature vector. If the j-th entry of x_i, denoted as $x_{i,j}$ is non-zero, then the j-th column of the matrix W, denoted as W_j, is extracted and multiplied by $x_{i,j}$. Then all extracted vectors are concatenated as a long vector, then the long vector is fed into several layers of the dense neuron network.

The wide_and_deep structure concatenates the wide structure and the last layer of the deep structure. More specifically, let the layer L_{wide} is the input layer of the wide structure, which is the last layer before the output, which is a scalar. The layer L_{deep} is the last layer before the output. Then the last layer of wide_and_deep before the output is the concatenation of L_{wide} and L_{deep}.

The cross structure, proposed by Wang et al. [17], generalized the wide structure. For each layer of the cross network, the formula between the input and output is

$$u_{l+1} = u_0 u_l^T w_l + c_l + u_l. \tag{4.7}$$

The first input u_0 is the given feature vector x_i, and the last one u_K is the output of the K depth cross network.

The cross_and_deep structure concatenates the last layer of the cross structure and the last layer of the deep structure similarly as the wide_and_deep structure.

4.2.2 Distributions

Is there a better assumption of the underlying distribution or a better formula of the loss in Fig. 4.1 of the winning price? In this dissertation, we study the normal, log-normal and Gumbel distribution.

The normal distribution is widely used and we use it as an example in the beginning of this section.

Cui et al. [6] studied the winning price and used the log-normal distribution to fit the winning price. Therefore, we study the performance of the log-normal distribution. We replace the w_i and b_i by $\log w_i$ and $\log b_i$, respectively in the formula of the normal

doi:10.6342/NTU201900270

distribution and we denote it by $\mathrm{lognormal}$.

Because the winning price is defined as the maximal bidding price from the competitors in Eq. 4.1, we study one of the limiting distribution of the extreme value theory, the Gumbel distribution which is first studied by Gumbel [9]. The reader might not be familiar with the Gumbel distribution, so we give more description of it here.

According to the extreme value theory, suppose $X_1, X_2, ..., X_n$ are independent and identical distributed random variables. Then $M_n =$ max $(X_1, X_2, ..., X_n)$ is also a random variable. Similar to the central limit theorem, it exists a distribution of M_n as $n \to \infty$. It is called the extreme value distribution. If we assume that the pdf of the distribution of X decreases exponentially when the value tends to infinity, which is called that the distribution of X has exponential tail, then the limiting distribution will be the Gumbel distribution, which is also known as the type I extreme value distribution.

Because the formula of the extreme value theory is closed to the definition of the winning price in Eq. 4.1 after adding following assumptions, we study the performance of the Gumbel distribution in this work. If the bidding price from the competitors are identical and independent distributed, and the number of the competitors are large, then the distribution of the winning price will converge. The Gumbel distribution is one of the three possible limiting distribution, and many widely used distributions such as the normal distribution has exponential tail, so we study the Gumbel distribution in our experiments. We denote it as gumbel. This is also an example of using the generalized winning price model with non-normal distribution.

To derive the loss function of the Gumbel distribution, we need to insert the formula of the cdf and the pdf of the Gumbel distribution into the Eq. 4.5 and Eq. 4.6. The cdf of the Gumbel distribution is:

$$\mathcal{F}_{Gumbel}(x|\mu, \sigma) = e^{-e^{-\frac{x-\mu}{\sigma}}}. \tag{4.8}$$

The pdf of the Gumbel distribution is

$$f_{Gumbel}(x|\mu, \sigma) = \frac{1}{\sigma} e^{-(z+e^{-z})}, \tag{4.9}$$

doi:10.6342/NTU201900270

where $z = \frac{x-\mu}{\sigma}$.

As shown in Fig. 4.2, the parameter μ controls the center of the distribution, so it is the *location parameter*. The parameter σ controls the shape of the distribution, so it is the *shape parameter*.

Suppose $X \sim \mathcal{F}_{Gumbel}(x|\mu, \sigma)$, then the expected value of X, denoted by $E(X)$, is

$$E(X) = \mu + \gamma\sigma, \tag{4.10}$$

and the variance of X, denoted by $Var(X)$, is

$$Var(x) = \frac{\pi^2}{6}\sigma^2, \tag{4.11}$$

where γ is the Euler-Mascheroni constant. Empirically, $\gamma \approx 0.5772$. Both Eq. 4.10 and Eq. 4.11 will be used to derive the initialization formula Eq. 5.1 in Chapter 5.

According to our generalized winning price model, the output of the link structure g should be the expectation of the Gumbel distribution. Note that we let $g(x_i|\beta) = \mu$ in our implementation because the $\gamma\sigma$ will be a constant when we fitting the parameter β. So the gumbel model in our experiments is:

$$w_i \sim \mathcal{F}_{Gumbel}(x|\mu = g(x_i), \sigma). \tag{4.12}$$

4.3 Mixture Model

In the Sec. 6.2, we will show that the expected values of the winning price on the winning data and the losing data are different. Therefore, we should use different model for the winning data and the losing data. For the winning data, we should use the winning model. For the losing data, because we cannot directly learn from the losing data only, we use the full model instead. Therefore, the proposed mixture model in [21] predicts the winning price by the weighted sum of the predictions from the winning model and the full model. We denote its prediction as $\hat{w}_i^m = \hat{w}_i^w * p_i + \hat{w}_i^f * (1 - p_i)$, where $\hat{w}_i^w$ is the prediction

doi:10.6342/NTU201900270

from the winning model and $\hat{w}_i^f$ is the prediction from the full model. In [21], we plug in p_i by the estimated winning rate from a logistic regression model. The prediction of the mixture model will be close to the prediction of the winning model if the x_i is close to winning data. On the other hand, if the feature x_i is similar to the losing data, then the mixture model will use the full model to predict the winning price.

We further revised the mixture model. First, plugging in p_i by the estimated winning rate is based on the assumption that $\hat{w}_i^w$ outperforms $\hat{w}_i^f$ on the winning data and $\hat{w}_i^f$ outperforms $\hat{w}_i^w$ on the losing data. This is not always true.

Furthermore, the idea of using winning rate might imply that the winning price depends on the bidding price. It is well known that higher bidding price has higher winning rate. Therefore, if the mixture model is based on the winning rate, then implicitly it depends on the bidding price. However, according to Eq. 4.1, the winning price does not depend on our bidding price.

In this study, we propose the revised mixture model as follow. We create an indicator $\rho_i \in \{0, 1\}$ of i-th bid to represent whether the winning model is better than the full model. On the winning data, we can directly compare the accuracy of the winning model and the full model, so we let:

$$\rho_i = \begin{cases} 0 & \text{if } \|\hat{w}_i^w - w_i\| > \left\|\hat{w}_i^f - w_i\right\|. \\ 1 & \text{otherwise.} \end{cases} \tag{4.13}$$

On the losing data, we only know the lower bound, so the larger one is the better:

$$\rho_i = \begin{cases} 0 & \text{if } \hat{w}_i^f > \hat{w}_i^w. \\ 1 & \text{otherwise.} \end{cases} \tag{4.14}$$

It might be not trivial to the reader that how we derive the Eq. 4.14. Suppose the relationship between the two estimators $\hat{w}_i^1, \hat{w}_i^2$ and the bidding price b_i on the losing data is $\hat{w}_i^1 < \hat{w}_i^2 < b_i$, then $\hat{w}_i^2$ is better because we know $b_i < w_i$. Similarly, if the relationship is $\hat{w}_i^1 < b_i < \hat{w}_i^2$, then $\hat{w}_i^2$ is still better. Finally, if the relationship is $b_i < \hat{w}_i^1 < \hat{w}_i^2$, then it is impossible to decide which one is more accurate. For convenience, we set $\hat{w}_i^2$ is better,

doi:10.6342/NTU201900270

and derive the Eq. 4.14.

Then, we learn a model to predict the ρ_i based on x_i. This is a problem of classification, so we can use similar link structures introduced in Sec. 4.2.1 with classification losses such as logistic loss to fit the model.

doi:10.6342/NTU201900270

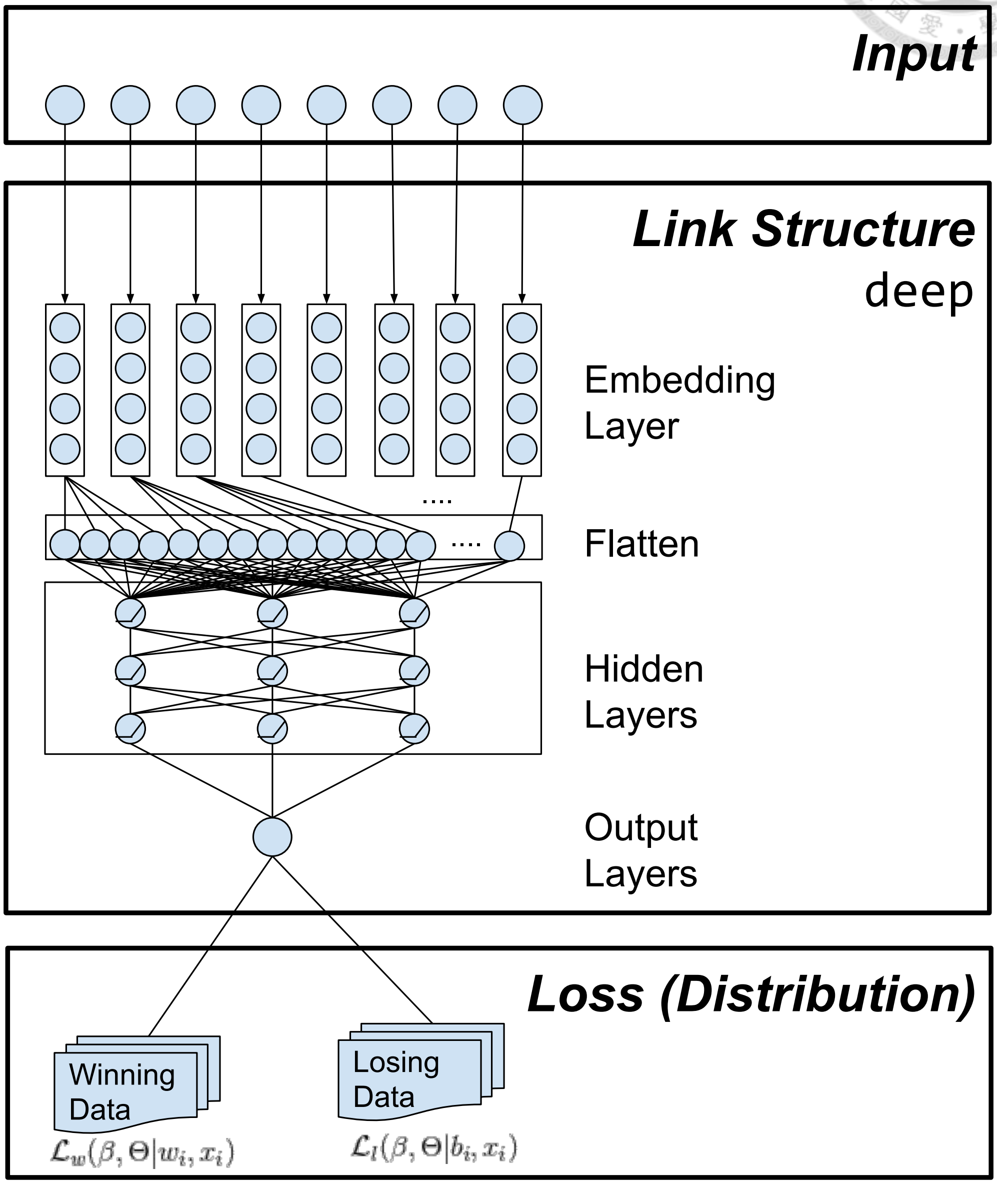

Figure 4.1: The proposed generalized winning price model. The link structure deep can be replaced by different structures. Section 4.2.1 lists the studied link structures. The loss is related to the conditional distribution of the winning price w_i given x_i. For winning data and losing data, we use different loss function based on Eq. 4.5 and Eq. 4.6 respectively.

doi:10.6342/NTU201900270

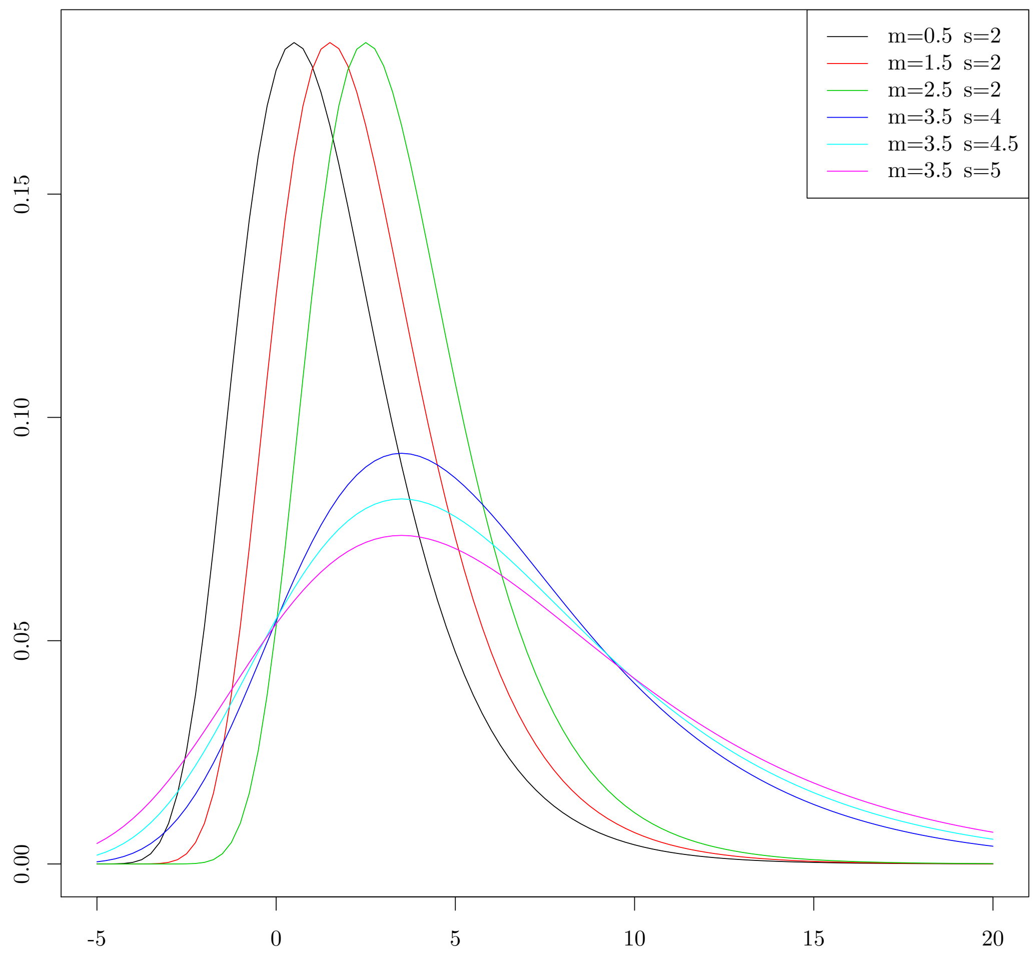

Figure 4.2: The PDF of the Gumbel distribution with different μ's and σ's.

doi:10.6342/NTU201900270

doi:10.6342/NTU201900270

Chapter 5

Algorithm

In this chapter, we describe how to train the corresponding deep learning model with different link structure and data distribution.

There are two groups of parameters to learn. The first group is the parameters β of the link structure $g(x_i|\beta)$. The second group is the parameters Θ of the conditional pdf and cdf. For example, the regression coefficients β of the linear regression are the first group and the variance σ^2 is the second group.

We use the coordinate descent method to learn the two groups. Many existed deep learning frameworks let the users define their own loss function, which is the function of the predictions and the real observations. Therefore, it is straightforward to train the parameters of the link structure via these frameworks directly when we fix Θ. After each epoch, we update Θ while β is fixed. The iterations is stopped when the loss stops improving on the validation dataset which is randomly selected from training dataset and will not be used in the training of both groups. The detailed algorithm is shown in Alg. 5.1.

In our experiments, we use the Adadelta algorithm proposed by Zeiler [23] to fit β and use the L-BFGS-B algorithm proposed by Byrd et al. [3] to optimize Θ. The parameters in β are randomly initialized except the bias term of the output layer, which we will denoted it as β_0 for convenience.

The Θ and β_0 are initialized as follow. First, we assume that $g(x_i|\beta) = \beta_0$. Then, the moments of the winning price become the function of the Θ and β_0. Takes normal distribution as an example, the first moment is β_0, and the second moment is $\beta_0^2 + \sigma^2$. In

doi:10.6342/NTU201900270

Input:
- The loss function $\mathcal{L}_w$ for winning data and $\mathcal{L}_l$ for losing data.
- $D_W = \{(x_i, w_i)|i = 1, 2, ..., n\}$: The winning data which contains the observed winning price and the corresponding vectorized features.
- $D_L = \{(x_i, w_i)|i = 1, 2, ..., n\}$: The losing data which contains the bidding price and the corresponding vectorized features.
- Split D_W and D_L into the training dataset and the validation dataset.

Initialization:

- Setting Θ according to the reduced moment estimators.
- Setting β randomly. The bias term of the output layer should be set by the moment estimator too.

Model Fitting via Coordinate Descent on Training Dataset:

- Fix Θ, learn β.
- Fix β, optimize Θ.
- Monitor the loss on the validation dataset and stop the iteration if the loss stops improving.

Figure 5.1: Coordinate descent algorithm of the generalized winning price model

general, we let $m_k(\Theta, \beta_0)$ be the analytic form of the k-th moment.

On the other hand, we use $\sum_{i \in D_W} w_i^k$ as an empirical estimator of the k-th moment. Therefore, we obtain a set of equations:

$$\begin{cases} \sum_{i \in D_W} w_i = m_1(\Theta, \beta_0) \\ \sum_{i \in D_W} w_i^2 = m_2(\Theta, \beta_0) \\ \dots \\ \sum_{i \in D_W} w_i^K = m_k(\Theta, \beta_0) \end{cases} \tag{5.1}$$

The number K depends on the number of parameters of Θ. We initialize Θ and β_0 by the simultaneous solution of Eq. 5.1.

For example, let the conditional distribution of w_i given x_i is gumbel which is intro-

doi:10.6342/NTU201900270

duced in Sec. 4.2.2. Then we plugin Eq. 4.10 and Eq. 4.11 into Eq. 5.1 to obtain:

$$\begin{cases} \frac{1}{N}\sum_{i\in D_W} w_i = \mu + \gamma\sigma \\ \frac{1}{N}\sum_{i\in D_W} w_i^2 = \frac{\pi^2}{6}\sigma^2 + (\mu+\gamma\sigma)^2 \end{cases} \tag{5.2}$$

The initial value of $\beta_0 = \mu$ and $\Theta = \{\sigma\}$ are set as the simultaneous solution of the Eq. 5.2.

doi:10.6342/NTU201900270

doi:10.6342/NTU201900270

Chapter 6

Experiments

In this chapter, we will introduce our experiments to study the modeling of the winning price. The experiments are based on the opened RTB dataset: the iPinYou dataset. We will introduce the dataset and how we vectorize the features. Then, we compare the distribution of the winning price on the winning data and losing data. The performance of the linear regression models based on winning data and losing data are also evaluated. We want to understand the effects of learning from losing data. The mixture model are also studied, too.

Then, we will directly study the deep learning models and distributions. Different combinations of the link structure and distributions are studied. We want to understand which model predicts the winning price better.

Because the difference between the winning data and losing data remains. Therefore, we apply the technique of mixture model and the revised mixture model to deep learning models and distributions.

6.1 Datasets and Features

The iPinYou dataset [25] contains three seasons of RTB records in the perspective of the DSP. We use the data of season 2 and 3 because the private feature is not provided in season 1. The features we used are shown in Table. 6.1. All numerical features are transformed to categorical by binning. Then all features are converted to binary features via hashing

doi:10.6342/NTU201900270

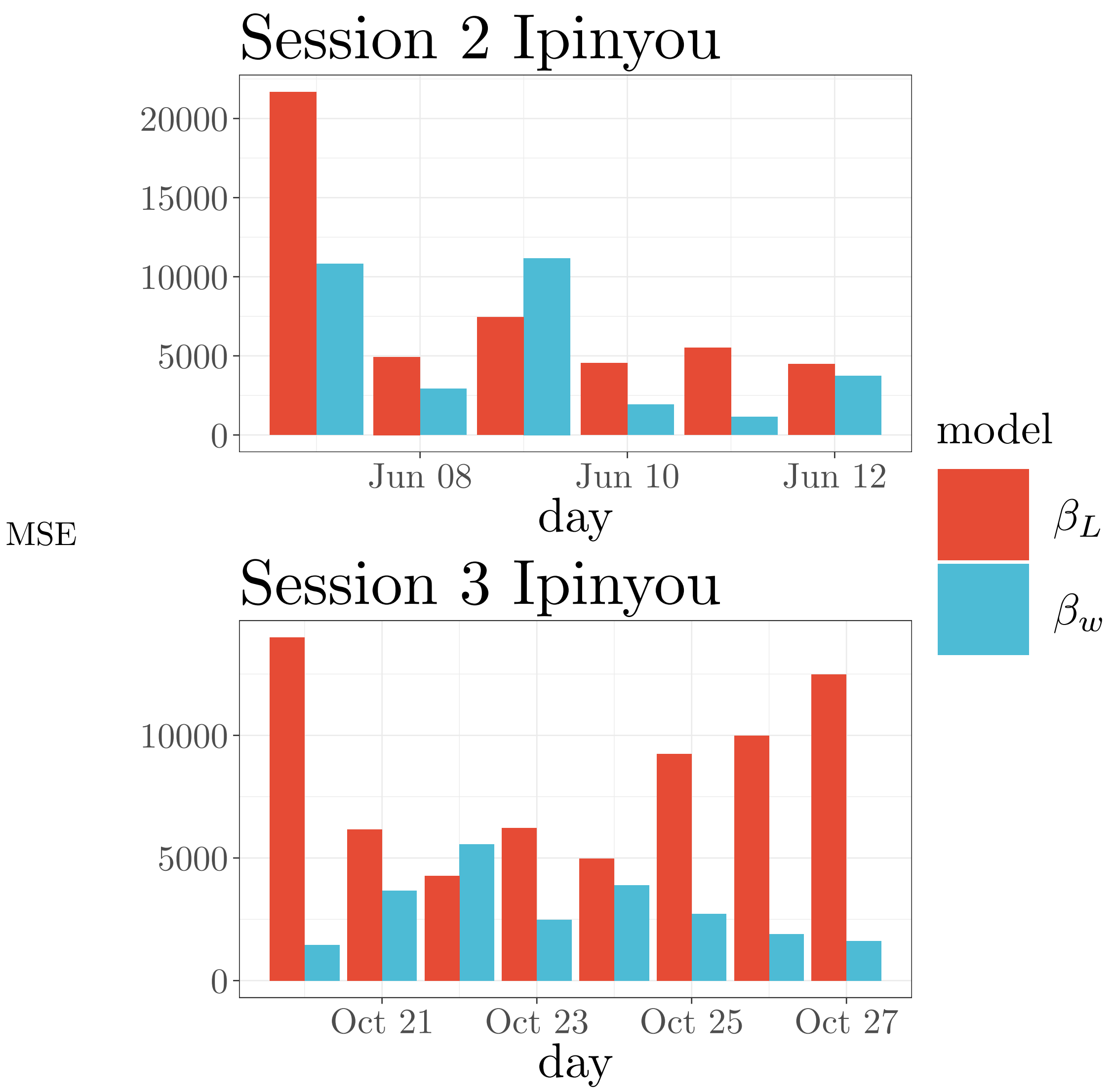

Figure 6.1: The mean squared error of the predictions. β_w is the model learned from winning data. β_L is the model learned from losing data. The results are evaluated at the winning data.

doi:10.6342/NTU201900270

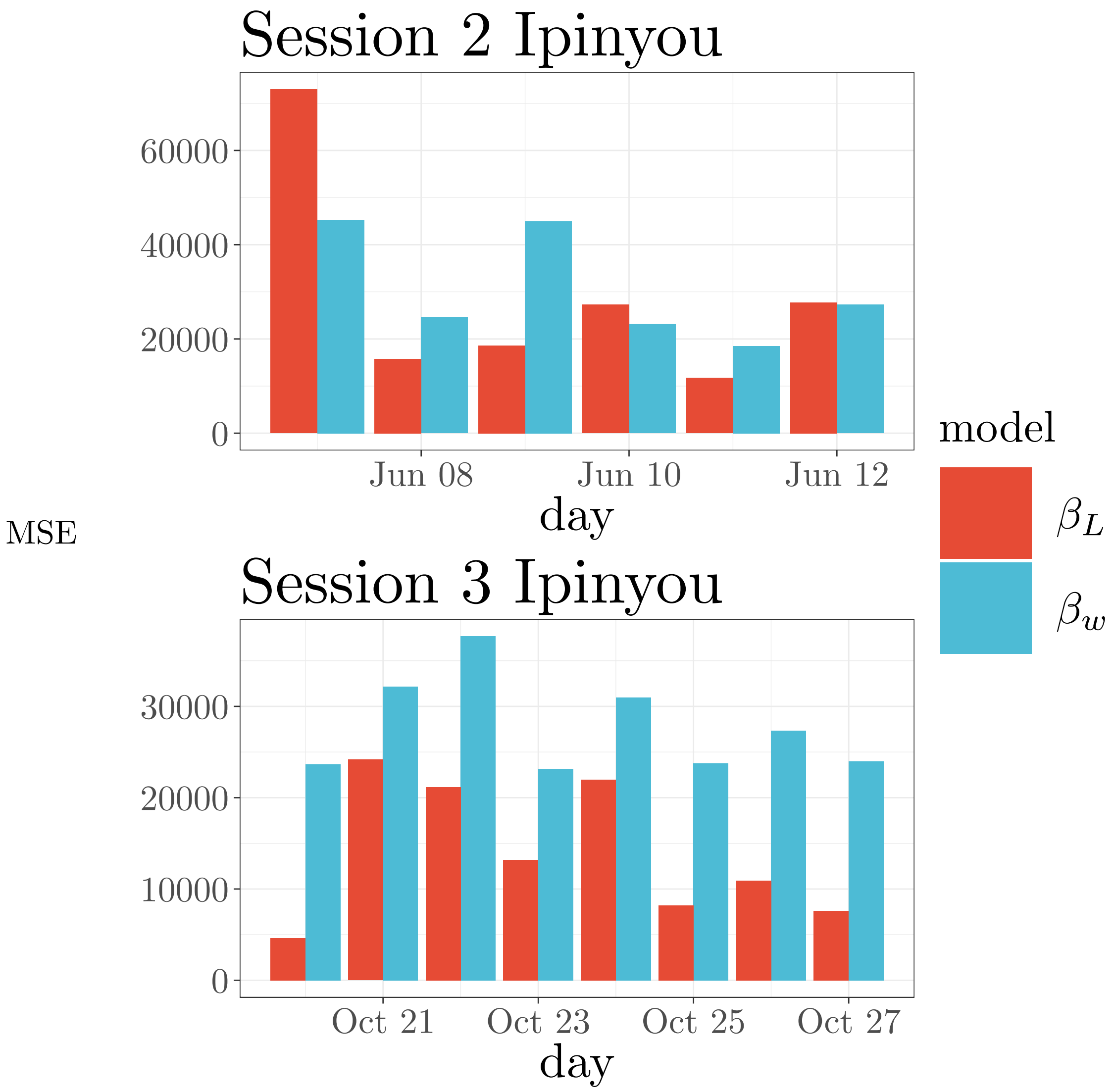

Figure 6.2: The mean squared error of the predictions. β_w is the model learned from winning data. β_L is the model learned from losing data. The results are evaluated at the losing data.

doi:10.6342/NTU201900270

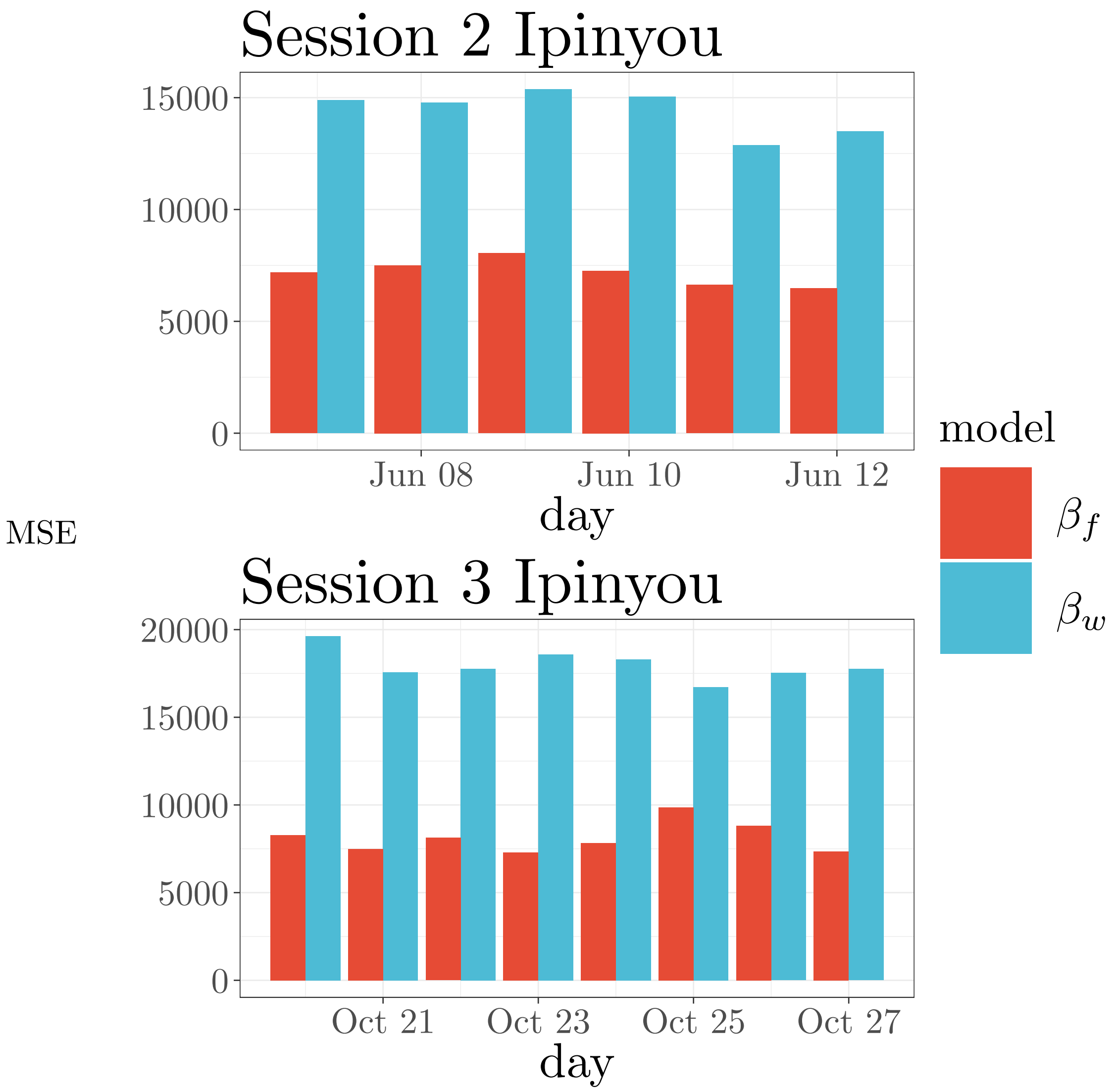

Figure 6.3: The mean squared error of the predictions. β_w is the model learned from winning data. β_f is the model learned from both winning data and losing data. The results are evaluated at the losing data.

doi:10.6342/NTU201900270

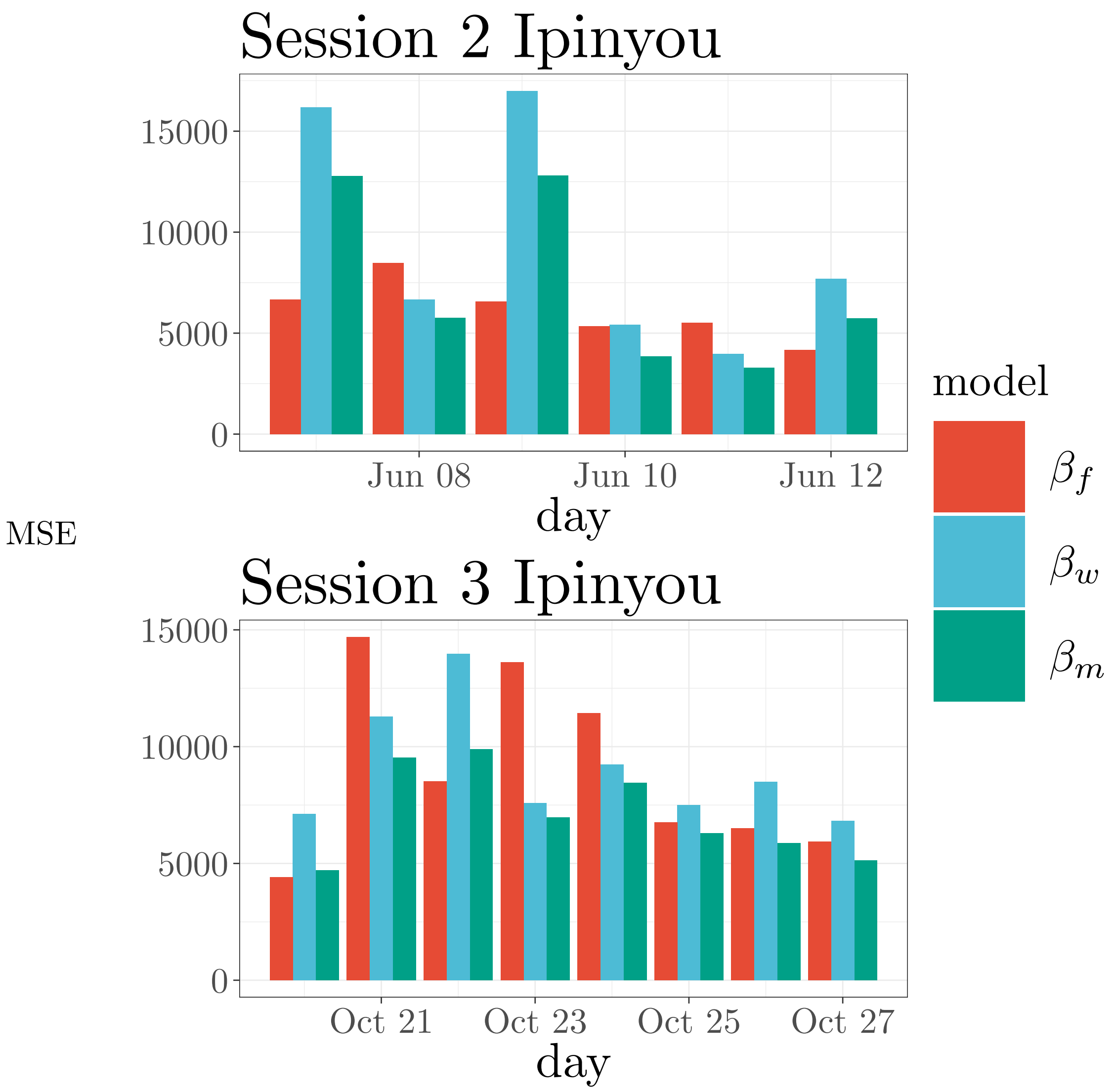

Figure 6.4: The mean squared error of the predictions. β_w is the model learned from winning data. β_f is the model learned from full data. β_m is the mixture model. The results are evaluated at the full data.

doi:10.6342/NTU201900270

Table 6.1: An example of features in iPinYou dataset [25].

Example	Feature Name
183.18.197.*	IP
216	Region
236	City
2	AdExchange
3d68edb4b8f5bba8bea6782e33c8e228	Domain
e63cfda49ec2a36a0cafcd646906227b	URL
3844656199	AdSlotId
250	AdSlotWidth
250	AdSlotHeight
OtherView	AdSlotVisibility
Na	AdSlotFormat
7321	CreativeID
6	weekday
02	hour
2259	adid
10684,10102,10006	usertag

Table 6.2: The statistics and the performance of the winning rate model for the dataset iPinYou Season 2.

	day	bids	win	Avg. WP	Avg. WP at W	Avg. WP at L	WR	EWR	WR AUC	WR logloss
1	2013-06-06	1821479	1514416	74.86	52.47	185.33	0.83	0.83	0.89	0.29
2	2013-06-07	1806062	1524314	72.31	51.12	186.97	0.84	0.85	0.90	0.26
3	2013-06-08	1634967	1352038	81.14	58.49	189.42	0.83	0.83	0.87	0.31
4	2013-06-09	1651630	1366097	81.32	58.96	188.29	0.83	0.83	0.88	0.30
5	2013-06-10	1920576	1603798	79.84	58.91	185.76	0.84	0.83	0.91	0.26
6	2013-06-11	1745905	1461085	79.62	58.92	185.84	0.84	0.86	0.85	0.36
7	2013-06-12	1657578	1378728	80.00	58.80	184.79	0.83	0.85	0.84	0.35

trick proposed by Weinberger et al. [19]. This trick is widely used in the field of the online advertising such as He et al. [10], Zhang et al. [25] and Chapelle [4]. Then we prune out those levels whose rate of appearance is less than 0.01%. Similar pruning is applied to the input of the wide model.

Only the winning price of the winning bids are available in the iPinYou dataset, so we only use these data in our experiments. All bidding price is divided by two and compared to the original winning price, so we obtain a simulated bidding result. The winning data and losing data in this section are all defined by the simulated result instead of the real result. In our settings, the pattern of the winning price is not changed, and we know the real winning price on the simulated losing data. Therefore, we can evaluate the performance of models on simulated losing data.

We show the basic statistics of the iPinYou datasets in the Table. 6.1 and Table. 6.1. For

doi:10.6342/NTU201900270

Table 6.3: The statistics and the performance of the winning rate model for the dataset iPinYou Season 3.

	day	bids	win	Avg. WP	Avg. WP at W	Avg. WP at L	WR	EWR	WR AUC	WR logloss
1	2013-10-19	228133	170739	86.74	50.89	193.36	0.75	0.75	0.85	0.39
2	2013-10-20	214295	159646	87.90	51.95	192.91	0.74	0.74	0.87	0.36
3	2013-10-21	848760	621626	90.09	51.41	195.95	0.73	0.73	0.89	0.34
4	2013-10-22	681700	503108	91.51	52.89	200.31	0.74	0.74	0.87	0.37
5	2013-10-23	226791	170850	89.27	53.45	198.69	0.75	0.75	0.83	0.40
6	2013-10-24	245897	197279	74.19	45.59	190.24	0.80	0.80	0.83	0.34
7	2013-10-25	318240	245671	82.28	49.20	194.26	0.77	0.77	0.85	0.36
8	2013-10-26	268380	198709	89.61	49.19	204.91	0.74	0.74	0.89	0.35
9	2013-10-27	110467	84730	86.01	48.44	209.71	0.77	0.77	0.90	0.31

each row, the statistics of the specific day is shown. The column *Avg. WP* is the averaged winning price. The column *Avg. WP at W* is the averaged winning price at winning data. The column *Avg. WP at L* is the averaged winning price at losing data. The column *WR* is the winning rate of the simulated bidding result.

6.2 The Difference Between the Winning Data and Losing Data

Are the pattern of winning price on the winning data and on the losing data different?

To answer the question, a naive way is to read the Table. 6.1 and Table. 6.1. They show that the averaged winning price at losing data, which is the column *Avg. WP at L*, is consistently higher than the averaged winning price at winning data, which is the column *Avg. WP at W*. Therefore, the patterns of the winning price are different at the observed data and at the censored data.

More specifically, we conduct the following experiments to study the pattern of the winning price conditioning on the features x_i. The estimator β_w is learned from the winning data of a specific day, which is called training day, via minimizing the Eq. 4.3 and the estimator β_L is learned from the losing data of the training day in the same way. We evaluate the performance at both the winning and losing data of the next day of the training day.

Note that the estimator β_L is impractical because the winning price at the losing data is unavailable.

doi:10.6342/NTU201900270

The results at the winning data are shown in Fig. 6.1. The β_w outperforms except two days and the difference of the MSE is more dramatic for cases which the β_w outperforms. Therefore, the estimator β_w generally outperforms.

The results at the losing data are shown in Fig. 6.2. Both datasets show different patterns of the result. It is hard to tell that which estimator is better in the dataset of iPinYou season 2. The estimator β_L consistently outperforms in the dataset of iPinYou season 3. In contrast, the estimator β_w consistently outperforms in the dataset of iPinYou season 2.

As a summary, the results in Fig. 6.1 and Fig. 6.2 show that the pattern of the winning price conditioning on the features are still different at the winning data and losing data. Otherwise, the result of two estimators should be similar.

6.3 The Performance of the Censored Regression

Learning β_L is impossible because we do not observe the winning price on losing data. Therefore, we use the loss Eq. 4.3 for winning data and Eq. 4.4 for losing data to learn a model which is denoted as β_f. The f means that the model is learned from full data.

The results are shown in Fig. 6.3. The β_f outperforms the β_w in both datasets iPinYou season 2 and 3.

The previous results show that learning from losing data is essential to predict the losing data. The Sec. 6.2 shows that the winning data and losing data are different. However, the model cannot directly learn from the losing data, so we use the censored regression introduced in Sec. 4.1 to learn from losing data. The Fig. 6.3 shows that the censored does perform better on the losing data.

6.4 The Performance of the Mixture Model

However, learning from losing data might damage the performance on the winning data because they are different. Therefore, we proposed the mixture model in Sec. 4.1, which will be denoted as β_m, and compare it with the winning model β_w and the full model

doi:10.6342/NTU201900270

β_f based on the evaluation on the full data. All of them are learned from the data of a specific day, which is called training day, via minimizing the objective function. We use the logistic regression to learn the winning rate.

Note that the performance of the logistic regression is shown in Table. 6.1 and Table. 6.1. The column *EWR* is the expected winning rate according to our winning rate model. The column *WR AUC* is the Area Under Curve (AUC) of our winning rate model. The column *WR Logloss* is the the Logarithm Loss (logloss) of our winning rate model.

The results are shown in Fig. 6.4. First, the β_m consistently outperforms the β_w, no matter whether the β_f is better than the β_w or not, and no matter the AUC and logloss of the predicted winning rate. The β_m is the averaged of the β_w and β_f, so it is interestingly that sometimes the performance of the β_m is the best.

One aspect is that the β_m does pick the more appropriate prediction from β_w and β_f. The winning rate provides a good indication of which estimator is better. If the impression is more likely to be won, then the β_w is better as shown in Sec. 6.2. If the impression is more likely to be lost, then the β_f provides the information from other losing data.

6.5 Comparing Different Link Structures on the Winning Data

In the following experiments, we want to focus on the effect of different link structures and the distributions. Therefore, we randomly split the training dataset and testing dataset from the full data of the iPinYou Season 2 and Season 3. There is no daily progressive validation anymore because the daily variation is large according to the experiments in Sec. 6.2, Sec. 6.3 and Sec.6.4.

When we only learn the model from the winning data, as shown in Table 6.4 and Table 6.5, the link structures cross, deep, $\mathrm{wide\_and\_deep}$, and $\mathrm{cross\_and\_deep}$ all outperform the link structure linear and wide. This is an evidence that the deep learning models that are successful in CTR prediction in the literature, such as Cheng et al. [5] and Wang et al. [17], also work better on the winning price problem.

doi:10.6342/NTU201900270

Table 6.4: Prediction results of different link structures with normal distribution and without censored data on the winning data from iPinYou 2nd Season.

	structure	winning_lll	winning_mse	winning_mae
1	cross	-4.4576	435.8893	13.5904
2	cross_and_deep	-4.4328	414.7468	13.2153
3	deep	**-4.4306**	**412.8637**	13.1343
4	linear	-4.5387	512.6252	15.9282
5	wide	-4.4720	448.5964	14.2188
6	wide_and_deep	-4.4312	413.3338	**13.0625**

Table 6.5: Prediction results of different link structures with normal distribution and without censored data on the winning data from iPinYou 3rd Season.

	structure	winning_lll	winning_mse	winning_mae
1	cross	-4.7718	816.9675	21.2496
2	cross_and_deep	-4.7540	787.6714	20.5687
3	deep	-4.7556	790.3033	**20.3882**
4	linear	-4.8335	924.4068	23.6405
5	wide	-4.7815	833.1215	21.7567
6	wide_and_deep	**-4.7530**	**786.5131**	20.4484

In our opinion, one reason is that the input features are similar, so the underlying relationship between the response and the features can be well approximated by similar models. Moreover, the prediction problem on the winning data is a traditional regression problem and does not involve the censored data.

However, the link structures cross, deep, wide_and_deep and cross_and_deep do not always outperform linear and wide in Table 6.5 and Table 6.5. For example, linear is the best link structure on the 2nd season compared by the log-likelihood. Unlike the traditional regression problem, the censored regression problems are not always improved by introducing the sophisticated link structure. In our opinion, the prediction on the losing data based on the censored data is based on the two assumptions and introducing sophisticated link structure might not be helpful. The first is that the winning data and the losing data are generated by the same model, although we already know that this assumption is wrong in [21]. Therefore, the sophisticated link structure might fit the winning data too well and the performance on the losing data is decreased. The second is the correct distribution of the winning price in the model, which is the reason why we extend our generalized winning price model to more distributions.

doi:10.6342/NTU201900270

Table 6.6: Prediction results of different link structures with normal distribution and on the losing data from iPinYou 2nd Season.

	structure	losing_lll_no	losing_lll_yes	losing_mse_no	losing_mse_yes	losing_mae_no	losing_mae_yes
1	cross	-20.8739	-9.0624	14738.6942	9420.6178	108.5623	79.8223
2	cross_and_deep	-21.0636	-9.1919	14001.3381	9196.6989	104.8069	78.8044
3	deep	-21.0732	-8.9986	**13923.0122**	8881.3482	**104.5022**	**76.4858**
4	linear	**-19.6334**	**-8.6330**	16034.5496	13478.5403	115.9354	106.2937
5	wide	-20.6144	-8.9124	14943.2811	11970.2267	109.8618	97.5983
6	wide_and_deep	-21.2974	-9.0653	14102.2706	**8819.0171**	105.2625	76.7369

Table 6.7: Prediction results of different link structures with normal distribution and on the losing data from iPinYou 3rd Season.

	structure	losing_lll_no	losing_lll_yes	losing_mse_no	losing_mse_yes	losing_mae_no	losing_mae_yes
1	cross	-15.7112	-8.6219	18447.7811	16356.7480	127.8237	105.5414
2	cross_and_deep	-16.1634	-7.6145	17936.7242	11244.4442	125.6698	91.9554
3	deep	-16.1079	**-6.9873**	18017.7939	16103.3782	125.8273	119.6627
4	linear	**-15.3388**	-7.0096	20296.4183	14262.8885	135.8718	110.9407
5	wide	-15.9169	-7.0570	19219.6432	12732.2718	131.5473	101.7985
6	wide_and_deep	-15.9669	-7.5330	**17861.6889**	**11072.0938**	**125.1455**	**90.6631**

6.6 Comparing Different Distributions

We compare the performance of different distributions under the link structure deep on the winning data in Table 6.8 and Table 6.9. As one can see, there is no significant winner of the three distributions. The gumbel distribution outperforms other distributions once, the normal distribution twice, and the lognormal distribution three times. Compared to the result in 6.5, we conclude that the effect of link structure is more significant compared to the effect of the distribution on the winning data.

6.7 Overall Comparison of Link Structures and Distributions

We compare the performance of different distributions and link structures on the winning data and losing data in Table 6.10, Table 6.11, Table 6.12, and Table 6.13. There is no specific combination of the distribution and the link structure that wins all comparisons.

On the winning data, deep outperforms all other link structures five times and wide_and_deep once. The distribution lognormal outperforms all other three times, normal twice, and gumbel once. The results still show that the sophisticated link structure do improve the performance on the winning data.

doi:10.6342/NTU201900270

Table 6.8: Prediction results of different distributions with link structure deep and without censored data on the winning data from iPinYou 2nd Season.

	loss	winning_lll	winning_mse	winning_mae
1	gumbel	**-4.3216**	468.8233	14.6015
2	lognormal	-4.5377	458.0666	**12.9183**
3	normal	-4.4306	**412.8637**	13.1343

Table 6.9: Prediction results of different distributions with link structure deep and without censored data on the winning data from iPinYou 3rd Season.

	loss	winning_lll	winning_mse	winning_mae
1	gumbel	-4.6535	843.8549	21.5606
2	lognormal	**-4.6484**	848.8295	**19.9958**
3	normal	-4.7556	**790.3033**	20.3882

However, on the losing data, the link structures linear and wide outperform other sophisticated link structures based on the log-likelihood measurement. The sophisticated link structures outperform linear and wide link structure based on the mean squared error and mean absolute error. And The distribution normal outperforms all other three times, lognormal twice, and gumbel once. In our opinions, there is no significant winner of the link structure and the distribution. It shows that learning from censored data and predicting the losing data is still a great challenge even if we introduce the link structure from deep learning and the non-normal distributions. Note that the distribution lognormal sometimes gives a high value to predict the log of the winning price and the mean squared error and the mean absolute error become large. Therefore, we report NA if the value exceeds 10^5 in the tables.

The proposed algorithm makes more flexibility to model the winning price but it also requires carefully tuning for specific dataset and measurement.

6.8 The Mixture Model of Generalized Winning Price Model

In this section, we compare the mixture mode and the revised mixture model based on the link structure deep with normal distribution.

The result is shown in Table. 6.14. Both mixture models outperform winning model as shown in Sec. 6.4. The revised mixture model performs better than mixture model on

doi:10.6342/NTU201900270

Table 6.10: Prediction results of different distributions and link structures without censored data on the winning data from iPinYou 2nd Season.

	structure	loss	winning_lll	winning_mse	winning_mae
1	cross	gumbel	-4.3432	491.6216	15.0109
2	cross	lognormal	-4.5629	479.0097	13.7080
3	cross	normal	-4.4576	435.8893	13.5904
4	cross_and_deep	gumbel	-4.3274	463.9229	14.4616
5	cross_and_deep	lognormal	-4.5388	469.3962	12.9662
6	cross_and_deep	normal	-4.4328	414.7468	13.2153
7	deep	gumbel	**-4.3216**	468.8233	14.6015
8	deep	lognormal	-4.5377	458.0666	**12.9183**
9	deep	normal	-4.4306	**412.8637**	13.1343
10	linear	gumbel	-4.4346	545.8220	16.5974
11	linear	lognormal	-4.6372	550.2760	15.9430
12	linear	normal	-4.5387	512.6252	15.9282
13	wide	gumbel	-4.3648	492.2358	15.3457
14	wide	lognormal	-4.5751	490.3757	14.1944
15	wide	normal	-4.4720	448.5964	14.2188
16	wide_and_deep	gumbel	-4.3218	464.3983	14.3919
17	wide_and_deep	lognormal	-4.5426	474.6091	13.1613
18	wide_and_deep	normal	-4.4312	413.3338	13.0625

losing data and all data.

In our opinion, the proposed revised mixture model is at least comparable to the original mixture model. Furthermore, it does not have the issue of implicit dependency of the bidding price. Therefore, the revised mixture model should be more feasible in practice.

doi:10.6342/NTU201900270

Table 6.11: Prediction results of different distributions and link structures without censored data on the winning data from iPinYou 3rd Season.

	structure	loss	winning_lll	winning_mse	winning_mae
1	cross	gumbel	-4.6658	867.2767	21.8831
2	cross	lognormal	-4.6687	893.9746	20.8158
3	cross	normal	-4.7718	816.9675	21.2496
4	cross_and_deep	gumbel	-4.6529	846.9736	21.5401
5	cross_and_deep	lognormal	-4.6500	892.0552	20.0674
6	cross_and_deep	normal	-4.7540	787.6714	20.5687
7	deep	gumbel	-4.6535	843.8549	21.5606
8	deep	lognormal	**-4.6484**	848.8295	**19.9958**
9	deep	normal	-4.7556	790.3033	20.3882
10	linear	gumbel	-4.7240	960.0766	23.9640
11	linear	lognormal	-4.7203	950.9485	22.6279
12	linear	normal	-4.8335	924.4068	23.6405
13	wide	gumbel	-4.6762	880.1435	22.5572
14	wide	lognormal	-4.6678	919.9535	21.0478
15	wide	normal	-4.7815	833.1215	21.7567
16	wide_and_deep	gumbel	-4.6566	842.2466	21.7194
17	wide_and_deep	lognormal	-4.6541	918.0869	20.4997
18	wide_and_deep	normal	-4.7530	**786.5131**	20.4484

Table 6.12: Prediction results of different distributions and link structures with censored data on the losing data from iPinYou 2nd Season. We report NA if the value exceeds 10^5.

	structure	loss	losing_likelihood	losing_mse	losing_mae
1	cross	gumbel	-7.3952	10288.4104	88.3192
2	cross	lognormal	-7.6055	NA	897.3822
3	cross	normal	-9.0624	9420.6178	79.8223
4	cross_and_deep	gumbel	-7.4159	10304.0098	87.4528
5	cross_and_deep	lognormal	-7.7986	NA	NA
6	cross_and_deep	normal	-9.1919	9196.6989	78.8044
7	deep	gumbel	-7.3738	10276.9203	87.4072
8	deep	lognormal	-7.1944	NA	264.3231
9	deep	normal	-8.9986	8881.3482	**76.4858**
10	linear	gumbel	-7.6529	15514.4172	122.5010
11	linear	lognormal	**-6.6842**	39241.4479	116.1995
12	linear	normal	-8.6330	13478.5403	106.2937
13	wide	gumbel	-7.5362	11206.8191	93.2920
14	wide	lognormal	-6.8614	NA	159.8261
15	wide	normal	-8.9124	11970.2267	97.5983
16	wide_and_deep	gumbel	-7.8029	10274.7159	87.0145
17	wide_and_deep	lognormal	-7.3441	NA	283.0034
18	wide_and_deep	normal	-9.0653	**8819.0171**	76.7369

doi:10.6342/NTU201900270

Table 6.13: Prediction results of different distributions and link structures with censored data on the losing data from iPinYou 3rd Season. We report NA if the value exceeds 10^5.

	structure	loss	losing_likelihood	losing_mse	losing_mae
1	cross	gumbel	-11.2795	13641.9595	107.5384
2	cross	lognormal	-76.2128	NA	NA
3	cross	normal	-8.6219	16356.7480	105.5414
4	cross_and_deep	gumbel	-8.3036	12384.1147	101.4792
5	cross_and_deep	lognormal	-8.1363	NA	1307.0256
6	cross_and_deep	normal	-7.6145	11244.4442	91.9554
7	deep	gumbel	-7.0293	17544.6791	136.3222
8	deep	lognormal	-6.8771	**6547.0021**	139.0764
9	deep	normal	-6.9873	16103.3782	119.6627
10	linear	gumbel	-6.8253	12752.7522	105.1365
11	linear	lognormal	-6.8407	NA	246.2628
12	linear	normal	-7.0096	14262.8885	110.9407
13	wide	gumbel	**-6.7611**	11825.1855	99.1948
14	wide	lognormal	-7.2980	NA	566.7420
15	wide	normal	-7.0570	12732.2718	101.7985
16	wide_and_deep	gumbel	-6.7755	11087.2502	94.7485
17	wide_and_deep	lognormal	-8.0712	NA	1318.3044
18	wide_and_deep	normal	-7.5330	11072.0938	**90.6631**

Table 6.14: The results of mixture model and revised mixture model.

measurements	winning model	full model	mixture model	revised mixture model
all_mae	46.65	45.42	45.00	**44.75**
all_mse	5033.13	**4588.72**	4976.21	4832.13
winning_mae	**20.55**	26.23	21.91	22.61
winning_mse	**788.73**	1217.55	897.85	943.70
losing_mae	124.21	**102.46**	113.64	110.54
losing_mse	17647.47	**14607.81**	17097.09	16388.55

doi:10.6342/NTU201900270

doi:10.6342/NTU201900270

Chapter 7

Conclusion and Future Work

In this dissertation, we study the problem of predicting the winning price in the real time bidding. We describe the real mechanism of the second price auction in the real time bidding, define the winning price and give a mathematical model of the winning price. Then we propose a statistical approach to model the winning price on the buyer side. Then the linear regression model based on the winning dataset is studied. The importance of learning from the losing dataset are emphasized and the censored regression model are applied. Then we generalize these winning price models with different link structure from the field of deep learning and different distributions. We proposed an algorithm to fit the model from censored data and the algorithm can be easily implemented with the deep learning frameworks. We showed that the performance is improved after applying sophisticated link structure from the deep learning models on the winning data and the performance is improved on the losing data after learning from censored data. Then, the mixture model is proposed and revised. The robustness of the linear winning price models and the generalized models are enhanced after applying the mixture model.

However, the mixture model is not the satisfactory solution of this problem. Both MSE and MAE of losing dataset are still huge compared to the MSE and MAE of the winning dataset. In our opinion, the lack of direct observation of the winning price on the losing dataset makes this problem very difficult. A better learning strategy should study the data collection instead of focus on learning the historical data only. Therefore, applying the reinforcement learning to collect the information efficiently from the market is our future

doi:10.6342/NTU201900270

work.

doi:10.6342/NTU201900270

Bibliography

[1] J. Aldrich. R.a. fisher and the making of maximum likelihood 1912-1922. *Statist. Sci.*, 12(3):162–176, 09 1997.

[2] L. M. Ausubel and P. Milgrom. The lovely but lonely vickrey auction. In *Combinatorial Auctions, chapter 1*. MIT Press, 2006.

[3] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu. A limited memory algorithm for bound constrained optimization. *SIAM J. Sci. Comput.*, 16(5):1190–1208, Sept. 1995.

[4] O. Chapelle. Modeling delayed feedback in display advertising. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, pages 1097–1105, New York, NY, USA, 2014. ACM.

[5] H.-T. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, H. Aradhye, G. Anderson, G. Corrado, W. Chai, M. Ispir, R. Anil, Z. Haque, L. Hong, V. Jain, X. Liu, and H. Shah. Wide & deep learning for recommender systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*, DLRS, pages 7–10, New York, NY, USA, 2016. ACM.

[6] 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*, KDD '11, pages 265–273, New York, NY, USA, 2011. ACM.

doi:10.6342/NTU201900270

[7] A. Ghosh, B. I. Rubinstein, S. Vassilvitskii, and M. Zinkevich. Adaptive bidding for display advertising. In *Proceedings of the 18th International Conference on World Wide Web*, WWW '09, pages 251–260, New York, NY, USA, 2009. ACM.

[8] Google. The arrival of real-time bidding. 2011.

[9] E. J. Gumbel. Statistics of extremes. *Columbia University Press, New York*, 1958.

[10] X. He, J. Pan, O. Jin, T. Xu, B. Liu, T. Xu, Y. Shi, A. Atallah, R. Herbrich, S. Bowers, and J. Q. Candela. Practical lessons from predicting clicks on ads at facebook. In *Proceedings of the Eighth International Workshop on Data Mining for Online Advertising, ADKDD 2014, August 24, 2014, New York City, New York, USA*, pages 1–9, 2014.

[11] Internet Advertising Bureau. IAB Internet Advertising Revenue Report 2013 first six month's results. Technical report, Internet Advertising Bureau, 2013.

[12] J. Klein and M. Moeschberger. *Survival Analysis: Techniques for Censored and Truncated Data*. Statistics for Biology and Health. Springer, 2003.

[13] C.-C. Lin, K.-T. Chuang, W. C.-H. Wu, and M.-S. Chen. Combining powers of two predictors in optimizing real-time bidding strategy under constrained budget. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, CIKM '16, pages 2143–2148, New York, NY, USA, 2016. ACM.

[14] W. Schnedler. Likelihood estimation for censored random vectors. Working Papers 0417, University of Heidelberg, Department of Economics, Feb. 2005.

[15] W. Vickrey. Counterspeculation, auctions, and competitive sealed tenders. *The Journal of Finance*, 16(1):8–37, 1961.

[16] J. Wang, S. Shen, and S. Seljan. Real-time bidding: A new frontier of computational advertising research. Workshop in CIKM, 11 2013.

doi:10.6342/NTU201900270

[17] R. Wang, B. Fu, G. Fu, and M. Wang. Deep & cross network for ad click predictions. In *Proceedings of the ADKDD'17*, ADKDD, pages 12:1–12:7, New York, NY, USA, 2017. ACM.

[18] Y. Wang, K. Ren, W. Zhang, J. Wang, and Y. Yu. Functional bid landscape forecasting for display advertising. In P. Frasconi, N. Landwehr, G. Manco, and J. Vreeken, editors, *ECML/PKDD (1)*, volume 9851 of *Lecture Notes in Computer Science*, pages 115–131. Springer, 2016.

[19] K. Weinberger, A. Dasgupta, J. Langford, A. Smola, and J. Attenberg. Feature hashing for large scale multitask learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 1113–1120, New York, NY, USA, 2009. ACM.

[20] W. Wu, M.-Y. Yeh, and M.-S. Chen. Deep censored learning of the winning price in the real time bidding. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery &#38; Data Mining*, KDD '18, pages 2526–2535, New York, NY, USA, 2018. ACM.

[21] W. C.-H. Wu, M.-Y. Yeh, and M.-S. Chen. Predicting winning price in real time bidding with censored data. In *Proceedings of the 21st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '15, pages 1305–1314, New York, NY, USA, 2015. ACM.

[22] S. Yuan, J. Wang, and X. Zhao. Real-time bidding for online advertising: Measurement and analysis. *CoRR*, abs/1306.6542, 2013.

[23] M. D. Zeiler. Adadelta: An adaptive learning rate method. *CoRR*, abs/1212.5701, 2012.

[24] 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*, KDD '14, pages 1077–1086, New York, NY, USA, 2014. ACM.

doi:10.6342/NTU201900270

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

[26] W. Y. Zhu, W. Y. Shih, Y. H. Lee, W. C. Peng, and J. L. Huang. A gamma-based regression for winning price estimation in real-time bidding advertising. In *2017 IEEE International Conference on Big Data (Big Data)*, pages 1610–1619, Dec 2017.

doi:10.6342/NTU201900270