# Delayed Feedback Model with Negative Binomial Regression for Multiple Conversions

Youngmin Choi, Mugeun Kwon, Younjin Park, Jinsoo Oh, Suyoung Kim
LINE Plus
{youngmin.choi,mugeun.kwon,younjin.park,jinsoo.oh,sediah}@linecorp.com

## ABSTRACT

In the display advertising market, one of the most popular advertisers' goals is acquiring conversions such as app installs and purchases, and an important technology that enables the advertising platform to support this campaign goal is to predict conversion rate (CVR). There are two major difficulties in predicting CVR: one is that conversions often don't happen immediately after a click, and the other is that some advertising products have to accept multiple conversions. In this paper, we introduce a new model - jointly trained Negative Binomial and Order Statistics - to tackle the multiple conversions and a series of conversion delays, simultaneously. Our proposed model shows the significant improvement in the real traffic data.

## CCS CONCEPTS

• **Information systems → Computational advertising**; • **Mathematics of computing → Probabilistic algorithms**; • **Theory of computation → Theory and algorithms for application domains**.

## KEYWORDS

Display Advertising, Conversion Prediction, Machine Learning, Count Prediction

**ACM Reference Format:**
Youngmin Choi, Mugeun Kwon, Younjin Park, Jinsoo Oh, Suyoung Kim. 2020. Delayed Feedback Model with Negative Binomial Regression for Multiple Conversions. In *AdKDD '20, August 23, 2020, San Diego, California.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145xxx

## 1 INTRODUCTION

In the display advertising market, advertisers register their ads on Demand-side platform (DSP) to show their ads on publishers' sites and the ads participate in Real-time bidding (RTB) through DSP. In this course, the advertisers pay using various payment options offered by DSP, such as cost-per-impression (CPM), cost-per-click (CPC) or cost-per-conversion (CPA).

CPA option is preferred in the real advertising market because it allows the advertisers to manage the price of their campaign goal directly. In LINE Ads Platform, the revenue of CPA bid products accounts for more than 70% of the total. To provide the CPA option, this paper deals with the problem to predict CVR that is the conversion rate after the preceding ad click event. When solving CVR prediction problem, we have to take two additional characteristics of conversion data into account.

The first intrinsic characteristic of conversion data is the *delay* which is defined as the interval between click and conversion. For example, after a user clicks an ad, it may take up to 4 weeks to complete an action such as an install or a purchase, which is finally counted as a conversion. That is, even if no conversion has occurred so far, there is still a possibility of conversion in the future. Therefore, CVR prediction has to consider the possibility that conversion will occur after building the training set.

The second one is *multiple occurrences*. Some types of conversions are regarded as only one conversion through the process of deduplication even if they occur multiple times. Typical examples of the conversion types in this kind are *app open* and *registration*. In contrast, a user may purchase the advertiser's products several times after clicking an ad and the advertiser want to accept all the *multiple* conversions, and is willing to pay the cost for each conversion. For this reason, it has to be allowed to consider all the multiple conversion events as valid for some types of conversions. This requires CVR prediction model to handle *count* data as well as *binary*.

There are many attempts to solve two problems above. In the *delay* issue, Delayed Feedback Model (DFM)[5] introduced some distributions for the delay and joined the delay distribution with logistic regression model so that he could build an excellent and very practical CVR prediction model. There are some additional studies[11, 17] to improve prediction for the delay. In case of multiple occurrence, the generalized linear models (GLMs) such as Poisson and Negative Binomial regression models are the best known models for modeling count data [3, 4].

We had to launch products that is able to take into account delays and multiple occurrences of conversions together. However, to the best of our knowledge, there is no study to deal with this problem. Thus, we construct a combined model of extended Delayed Feedback Model based on Order Statistics and Negative Binomial regression model. This study can cover not only the advertising but also many areas with delayed multiple actions, such as e-commerce and communication services.

The rest of this paper is organized as follows: We describe some related works in Section 2 and introduce the characteristics of conversions with our real data in Section 3. In Section 4, we propose our model in detail. Our implemented system is described in Section 5. Finally, we present experiment results in Section 6 and Section 7 concludes this paper.


*AdKDD '20, August 23, 2020, San Diego, California*

ACM ISBN 978-1-4503-XXXX-X/18/06…$xx
https://doi.org/10.1145xxx

## 2 RELATED WORK

Recently, many studies have been actively conducted in online advertising domain, such as CTR/CVR prediction [9, 14]. [6, 8, 15] presented various methods of feature representation used in CTR and CVR predictions. In [6, 8], deep learning has been applied for the improvements of interactions between features. DFM [5] optimized the joint distribution of conversion and delay to improve the accuracy of CVR prediction. After this work, there have been various attempts to fit the distribution of the delay time, more exactly. [11] proposed DFM with Weibull distribution, and [17] used non-parametric distribution as delay distribution.

The other related domain is the prediction of count data which means non-negative integer. GLMs such as Poisson and Negative Binomial regression models are the best known models for modeling count data [3, 4]. Although Poisson regression is suitable for rare and large-count data and has an easy and simple structure to fit the model[12], it has some weaknesses because of the assumption that the variance and mean of the model are equal. In the real-world data, overdispersion, the variance of the model is greater than the mean, often occurs[10]. Negative Binomial regression does not have the assumption. Thus, it is more flexible than Poisson regression and proper for count data with overdispersion.

Our model is an extended Delayed Feedback Model replacing Logistic regression with Negative Binomial regression for predicting multiple conversions. To model each delays of conversions, we apply the idea of Order Statistics as well.

## 3 CHARACTERISTICS OF CONVERSIONS

Among many ads with various objectives and payment methods, we should select the best ads which are expected to maximize our revenue, based on estimated revenue per 1000 impressions, that is known as eCPM, the de facto standard of RTB. Equation (1) clearly shows the relation between revenue and prediction when advertiser choose CPA payment option.

$$\text{eCPM} = \text{CPA} \times \text{pCVR} \times \text{pCTR} \times 1000 \tag{1}$$

where CPA is the price the advertiser is willing to pay for a conversion, pCVR and pCTR are predicted conversion rate and click-through rate, respectively. Since pCVR is directly used in calculating the estimated cost, inaccurate prediction may prevent advertisers from achieving their goal, decrease in media's revenue and provide user with negative experience.

In the rest of this section, we describe some characteristics and statistics of the conversion with LINE Ads Platform's data.

### 3.1 Multiple Conversions

Generally, as described in DFM, one click can have multiple conversions and many recent products need to consider those conversions as mentioned before.

To quantify this issue, let *Non-binary ratio* be the proportion of the number of clicks having at least two conversions to the number of clicks having conversions. This indicator represents how much data cannot be predicted correctly with the binary model.

Table 1 shows *Non-binary ratio* of our 4 conversion types, which are sorted in descending order by the number of events in recent 3 months. More than 25% of clicks in Type A and Type D have multiple conversions. Notice that, in case of Type C, *Non-binary ratio* is zero, because the deduplication policy is applied to meet the requirements of the product.

**Table 1: *Non-binary ratio* by LINE's conversion type.**

| Conversion Type | Type A | Type B | Type C | Type D |
|---|---|---|---|---|
| *Non-binary ratio* (%) | 26.4 | 1.1 | 0.0 | 59.3 |

### 3.2 Conversion Delay

DFM takes only one delay into consideration for a single click when predicting conversion probability. However, we have to cope with the case of multiple delays coming from each of conversions. Figure 1 shows the delay distribution of one of recent campaigns with multiple delays. The plot shows the probability distribution function (PDF) according to the conversion order. As shown in the PDF, the peak of distribution is moving to the right as the order of conversion increases. Thus, we take this observation into account in our model with the idea of Order Statistics.

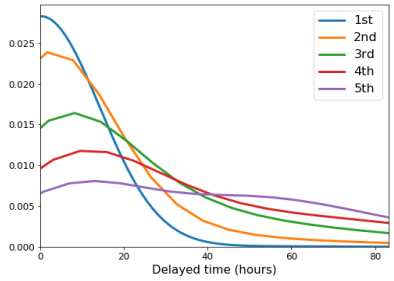


**Figure 1: Estimated PDF of $i$-th conversion delay for a recent campaign. Those functions are tamed by using kernel density estimation using Gaussian kernel.**

## 4 PROPOSED MODEL

This section explains our model and learning algorithm in detail. Before going deeper, we define some notations of variables.

**Table 2: Observed data corresponding to Y**

| Y | Observed Data |
|---|---|
| $Y = 0$ | $X = \mathbf{x}, E = e$ |
| $Y = 1$ | $X = \mathbf{x}, E = e, D_1 = d_1$ |
| $\dots$ | $\dots$ |
| $Y = a$ | $X = \mathbf{x}, E = e, D_1 = d_1, D_2 = d_2, \dots, D_a = d_a$ |

We use $X \in \mathbb{R}^m$ to denote the feature vector composed of user, ad and site information, where $m$ is the size of the feature vector. $Y \in \{0, 1, \dots\}$ denotes the number of observed conversions when we train the model and $C \in \{0, 1, \dots\}$ is a hidden variable, representing the number of conversions which will be obtained eventually. $E$ indicates the elapsed time since the click at the time the model is

trained. $D_i \in [0, \infty)$ is the $i$−th delay between the click and $i$−th conversion and whether $D_i$ is observed or not depends on the value of $Y$, and see Table 2 for easy understanding. As in Figure 2, if

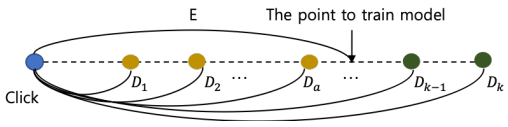


**Figure 2: Visualization of a event with count conversions when $Y = a$ and $C = k$.**

the number of observed conversions ($Y$) is $a$, then that of eventual conversions ($C$) is greater than or equal to $a$ and the following relations are established:

1. When $a = 0$,

$$Y = 0 \Leftrightarrow C = 0 \text{ or } (C = k \text{ and } E < D_1 < \cdots < D_k \text{ for all } k > 0) \tag{2}$$

2. When $a > 0$ and $D_1 < \cdots < D_a < E$,

$$Y = a \Leftrightarrow C = a \text{ or } (C = k \text{ and } E < D_{a+1} < \cdots < D_k \text{ for all } k > a). \tag{3}$$

## 4.1 Initial Approach

The model we first approached is an extended version by repeating DFM[5], as shown in Figure 3. In this model, two types of probability model, Logistic regression and Exponential distribution, in DFM are repeated to predict every $i$-th conversion and its delay. First, the model to predict multiple conversions is expressed as a

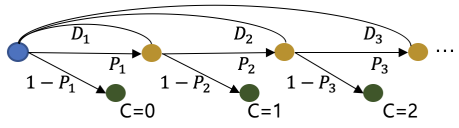


**Figure 3: The model we first considered, which is extended from Delayed Feedback Model.**

multiplication of consecutive binary logistic models given $X$:

$$Pr(C = k|X = \mathbf{x}) = p_1(\mathbf{x})p_2(\mathbf{x}) \cdots p_k(\mathbf{x})(1 - p_k(\mathbf{x})) \tag{4}$$

where $p_i(\mathbf{x}) = (1+\exp(-\mathbf{w}_{ci} \cdot \mathbf{x}))^{-1}$, $k$ is the number of conversions, $p_i(\mathbf{x})$ is the probability of $i$-th conversion given $(i-1)$-th conversion is occurred and $w_{ci}$ is the parameter of $p_i(\mathbf{x})$.

Second, the delay of each $i$-th conversion is predicted by an Exponential distribution with PDF $f(z|X = \mathbf{x}) = \lambda(\mathbf{x}) \exp(-\lambda(\mathbf{x})z)$ given $X$:

$$Pr(D_i = d_i|X = \mathbf{x}) = \lambda_i(\mathbf{x}) \exp(-\lambda_i(\mathbf{x})d_i) \tag{5}$$

where $\lambda_i(\mathbf{x}) = \exp(\mathbf{w}_{di} \cdot \mathbf{x})$ and $w_{di}$ is the parameter of $\lambda_i(\mathbf{x})$.

However, there are several limitations to apply this model to production system. The number of parameters in Equation (4), (5) can increase to infinity theoretically, which results in infeasible complexity for prediction. To apply this model on production system, we should limit the number of parameters under some reasonable size.

## 4.2 New Model

We propose a feasible model that can be applied by substituting (4), (5) with (6), (7) using some assumptions: The first assumption is that $p_i(\mathbf{x})$, the probability of $i$-th conversion, is the same $p(\mathbf{x})$, $\forall i$. Then, we can get the following equation:

$$Pr(C = k|X = \mathbf{x}) = p(\mathbf{x})^k(1 - p(\mathbf{x})) \tag{6}$$

where $p(\mathbf{x}) = (1 + \exp(-\mathbf{w}_c \cdot \mathbf{x}))^{-1}$, $k$ is the number of conversions and $\mathbf{w}_c$ is a weight vector. Note that, this is the well-known Negative Binomial regression.

The second is that the time delays between click and multiple conversions, $D_1, \ldots, D_k$, follow an Order Statistics[2] of i.i.d. random variables following an Exponential distribution with PDF, $f(z|X = \mathbf{x}) = \lambda(\mathbf{x}) \exp(-\lambda(\mathbf{x})z)$ given $X$. Thus,

$$\begin{aligned} &Pr(D_1 = d_1, \ldots, D_k = d_k|X = \mathbf{x}, C = k) \\ &= k! \prod_{i=1}^{k} f(d_i|X = \mathbf{x}) = k!\lambda(\mathbf{x})^k \exp(-\lambda(\mathbf{x}) \sum_{i=1}^{k} d_i) \end{aligned} \tag{7}$$

where $\lambda(\mathbf{x}) = \exp(\mathbf{w}_d \cdot \mathbf{x})$ , $0 < d_1 < d_2 < \cdots < d_k$ and $\mathbf{w}_d$ is a weight vector.

The Exponential distribution could be replaced with Weibull, Log-normal distributions. In our model, we combined Negative Binomial regression and Exponential distribution which gives a powerful computational advantage that will be explained soon.

Before combining the two models, we need an another notable relation between variables. The number of observed conversions $Y$ depends on the time the model is trained. However, the fact could not affect the number of final conversions and the model prediction. Thus, the elapsed time $E$ and $C$ are independent given $X$,

$$Pr(C|X, E) = Pr(C|X). \tag{8}$$

From (3), we get the following equation.

$$\begin{aligned} &Pr(Y = a, D_1 = d_1, \ldots, D_a = d_a|C = k, X = \mathbf{x}, E = e) \\ &= Pr(D_1 = d_1, \ldots, D_a = d_a, E < D_{a+1} < \cdots < D_k|C = k, X = \mathbf{x}, E = e) \\ &= k!\lambda(\mathbf{x})^a \exp(-\lambda(\mathbf{x}) \sum_{i=1}^{a} d_i) \int_e^{\infty} \cdots \int_{d_{k-1}}^{\infty} \lambda(\mathbf{x}) \exp(-\lambda(\mathbf{x})d_k) d(d_k) \cdots d(d_{a+1}) \\ &= k!\lambda(\mathbf{x})^a \exp(-\lambda(\mathbf{x}) \sum_{i=1}^{a} d_i) \frac{1}{(k-a)!} \exp(-(k-a)\lambda(\mathbf{x})e) \\ &= a!\lambda(\mathbf{x})^a \exp(-\lambda(\mathbf{x}) \sum_{i=1}^{a} d_i) \binom{k}{k-a} \exp(-(k-a)\lambda(\mathbf{x})e) \end{aligned} \tag{9}$$

Note that, since (9) does not hold when $a = 0$, this case can be written from (2) as:

$$Pr(Y = 0|C = k, X = \mathbf{x}, E = e) = \exp(-k\lambda(\mathbf{x})e) \tag{10}$$

By the law of total probability, the probability of a conversion event $(Y = a, D_1 = d_1, \ldots, D_a = d_a)$ is as follows.

$$\begin{aligned} &Pr(Y = a, D_1 = d_1, \ldots, D_a = d_a|X = \mathbf{x}, E = e) \\ &= \sum_{k=0}^{\infty} Pr(Y = a, D_1 = d_1, \ldots, D_a = d_a|C = k, X = \mathbf{x}, E = e) Pr(C = k|X = \mathbf{x}) \end{aligned} \tag{11}$$

In (11), it is clear that $Pr(Y = 0|C = 0, X = \mathbf{x}, E = e) = 1$ and $Pr(Y = a|C = k, X = \mathbf{x}, E = e) = 0$ when $a > k$. This is because the number of observed conversions never be greater than the number of conversions we eventually get.

By plugging (9) and (10) into (11), (11) is derived as:
1. When the number of observed conversions, $Y$ , is 0 (i.e. $a = 0$),

$$
\begin{aligned}
&Pr(Y=0|X=\mathbf{x},E=e) \qquad (12)\\
&=\sum_{k=0}^{\infty} Pr(Y=0|C=k,X=\mathbf{x},E=e)Pr(C=k|X=\mathbf{x},E=e)\\
&=Pr(Y=0|C=0,X=\mathbf{x},E=e)Pr(C=0|X=\mathbf{x})\\
&\qquad +\sum_{k=1}^{\infty} Pr(Y=0|C=k,X=\mathbf{x},E=e)Pr(C=k|X=\mathbf{x})\\
&=(1-p(\mathbf{x}))+\sum_{k=1}^{\infty}\exp(-k\lambda(\mathbf{x})e)p(\mathbf{x})^k(1-p(\mathbf{x}))\\
&=(1-p(\mathbf{x}))+(1-p(\mathbf{x}))\sum_{k=1}^{\infty}(\exp(-\lambda(\mathbf{x})e)p(\mathbf{x}))^k\\
&=(1-p(\mathbf{x}))+p(\mathbf{x})(1-p(\mathbf{x}))(\exp(\lambda(\mathbf{x})e)-p(\mathbf{x}))^{-1}
\end{aligned}
$$

2. When one or more conversions are observed (i.e. $a \geq 1$),

$$
\begin{aligned}
&Pr(Y=a,D_1=d_1,\ldots,D_a=d_a|X=\mathbf{x},E=e) \qquad (13)\\
&=\sum_{k=a}^{\infty} Pr(Y=a|C=k,X=\mathbf{x},E=e)Pr(C=k|X=\mathbf{x},E=e)\\
&=\sum_{k=a}^{\infty} a!p(\mathbf{x})^k(1-p(\mathbf{x}))\lambda(\mathbf{x})^a\exp(-\lambda(\mathbf{x})\sum_{i=1}^{a}d_i)\binom{k}{k-a}\exp(-(k-a)\lambda(\mathbf{x})e)\\
&=a!(p(\mathbf{x})\lambda(\mathbf{x}))^a(1-p(\mathbf{x}))\exp(-\lambda(\mathbf{x})\sum_{i=1}^{a}d_i)\\
&\qquad \times\sum_{k=a}^{\infty}\binom{k}{k-a}(p(\mathbf{x})\exp(-\lambda(\mathbf{x})e))^{k-a}\\
&=a!(p(\mathbf{x})\lambda(\mathbf{x}))^a(1-p(\mathbf{x}))\\
&\qquad \times\exp(-\lambda(\mathbf{x})(\sum_{i=1}^{a}d_i))\sum_{j=0}^{\infty}\binom{j+(a+1)-1}{j}(p(\mathbf{x})\exp(-\lambda(\mathbf{x})e))^j\\
&=a!(p(\mathbf{x})\lambda(\mathbf{x}))^a(1-p(\mathbf{x}))\exp(-\lambda(\mathbf{x})(\sum_{i=1}^{a}d_i))(1-p(\mathbf{x})\exp(-\lambda(\mathbf{x})e))^{-(a+1)}.
\end{aligned}
$$

The above infinite sum, a part of equation (13), is simplified by using Negative Binomial series, reducing the complexity for training and prediction.

## 4.3 Optimization

In this section, we propose how we learn our model on the basis of joint optimization. Suppose we observe $n$ samples, denoted by $\{\mathbf{x}_j, a_j, e_j, d_{j1}, \ldots, d_{ja_j}\}_{j=1}^{n}$ , then the joint negative log likelihood is obtained from (12), (13):

$$
\begin{aligned}
&L(\mathbf{w}_c,\mathbf{w}_d) \qquad (14)\\
&=-\sum_{j=1}^{n}\log(Pr(Y_j=a_j,D_{j1}=d_{j1},\ldots,D_{ja_j}=d_{ja_j}|X_j=\mathbf{x}_j,E_j=e_j))\\
&=-\sum_{j=1,y_j=0}^{n}\log((1-p(\mathbf{x}_j))(1+p(\mathbf{x}_j)(\exp(\lambda(\mathbf{x}_j)e_j)-p(\mathbf{x}_j))^{-1})\\
&\quad -\sum_{j=1,y_j\geq 1}^{n}\{\log(a_j!)+a_j\log(p(\mathbf{x}_j)\lambda(\mathbf{x}_j))+\log(1-p(\mathbf{x}_j))\\
&\qquad -\lambda(\mathbf{x}_j)\sum_{i=1}^{a_j}d_{ji}-(a_j+1)\log(1-p(\mathbf{x}_j)\exp(-\lambda(\mathbf{x}_j)e_j))\}.
\end{aligned}
$$

Our objective function (14) is non-convex due to the similar reason illustrated in [5] because there are several options to minimize the objective function. Specifically, there are at least two directions to minimize the objective function when the most of observed clicks are not converted. One direction is lowering conversion rate and increasing delay, and the other is raising conversion rate and decreasing delay.

The parameters in our model are $\mathbf{w}_c, \mathbf{w}_d$ which are obtained by minimizing the negative log likelihood (14) using the L-BFGS optimizer[13], [1]. In order to use L-BFGS optimizer, we need to calculate the gradients of the parameter vectors. By the chain rule, the gradients with respect to $\mathbf{w}_c, \mathbf{w}_d$ can be derived as follows:
First, gradient with respect to $\mathbf{w}_c$ is calculated as:

$$
\begin{aligned}
&\frac{\partial L(\mathbf{w}_c,\mathbf{w}_d)}{\partial \mathbf{w}_c} \qquad (15)\\
&=\sum_{j=1,y_j=0}^{n}\frac{1-(1-2p(\mathbf{x}_j))A(\mathbf{x}_j)^{-1}-(p(\mathbf{x}_j)-p(\mathbf{x}_j)^2)A(\mathbf{x}_j)^{-2}}{(1-p(\mathbf{x}_j))+(p(\mathbf{x}_j)-p(\mathbf{x}_j)^2)A(\mathbf{x}_j)^{-1}}\frac{\partial p(\mathbf{x}_j)}{\partial \mathbf{w}_c}\\
&+\sum_{j=1,y_j\geq 1}^{n}\{\frac{-a_j}{p(\mathbf{x}_j)}+\frac{1}{1-p(\mathbf{x}_j)}-(a_j+1)\frac{exp(-\lambda(\mathbf{x}_j)e_j)}{1-B(\mathbf{x}_j)}\}\frac{\partial P(\mathbf{x}_j)}{\partial \mathbf{w}_c}
\end{aligned}
$$

where $A(\mathbf{x}_j)=\exp(\lambda(\mathbf{x}_j)e_j)-p(\mathbf{x}_j)$ and $B(\mathbf{x}_j)=p(\mathbf{x}_j)\times\exp(-\lambda(\mathbf{x}_j)e_j)$.
Second, gradient with respect to $\mathbf{w}_d$ is calculated as:

$$
\begin{aligned}
&\frac{\partial L(\mathbf{w}_c,\mathbf{w}_d)}{\partial \mathbf{w}_d} \qquad (16)\\
&=\sum_{j=1,y_j=0}^{n}\frac{(p(\mathbf{x}_j)-p(\mathbf{x}_j)^2)A(x_i)^{-2}\exp(-\lambda(\mathbf{x}_j)e_j)e_j}{(1-p(\mathbf{x}_j))+(p(\mathbf{x}_j)-p(\mathbf{x}_j)^2)A(x_i)^{-1}}\frac{\partial\lambda(\mathbf{x}_j)}{\partial \mathbf{w}_d}\\
&+\sum_{j=1,y_j\geq 1}^{n}\{-\frac{a_j}{\lambda(\mathbf{x}_j)}+\sum_{i=1}^{a_j}d_{ji}+(a_j+1)\frac{B(\mathbf{x}_j)e_j}{1-B(\mathbf{x}_j)}\}\frac{\partial\lambda(\mathbf{x}_j)}{\partial \mathbf{w}_d}
\end{aligned}
$$

where $A(\mathbf{x}_j)=\exp(\lambda(\mathbf{x}_j)e_j)-p(\mathbf{x}_j)$ and $B(\mathbf{x}_j)=p(\mathbf{x}_j)\times\exp(-\lambda(\mathbf{x}_j)e_j)$.

## 4.4 Prediction

Unlike other models [5, 17], we should predict the expected number of conversions, not the probability of a conversion because we have to cover multiple conversions. After we get estimated parameter vectors $\mathbf{w}_c$ and $\mathbf{w}_d$, the predicted value given $X$ can be calculated as:

$$E(C|X=\mathbf{x})=\frac{p(\mathbf{x})}{1-p(\mathbf{x})} \qquad (17)$$

where $p(\mathbf{x})=(1+\exp(-\mathbf{w}_c\cdot\mathbf{x}))^{-1}$.

# 5 IMPLEMENTATION

To deploy our model into the production system, we implemented 3 parts of components, called *Data Pipeline*, *Model Training* and *Model Serving*, as shown in Figure 4.

## 5.1 Data Pipeline

Gathering the credible data always matters. It is hard to collect conversions directly because they occur on the advertisers' sites or mobile applications. For this, we first track every possible user activity data from advertisers' web and mobile applications. If clicks from our system contribute to activities, then our *Conversion Worker*

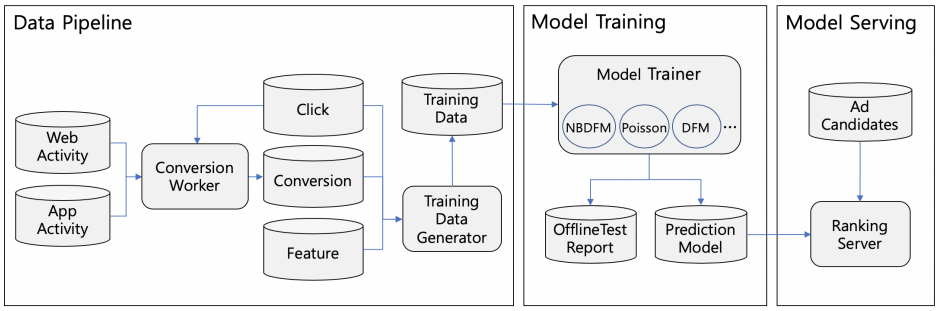


**Figure 4: Overview of Implemented System**

treats those activities as conversions. There are many studies [7, 18] about how to assign the conversion credit to the clicks. We use *last-click attribution model* by default. In case of click and feature data, they are easily obtained in our internal system.

We periodically generate training data with feature, click and conversion data. Here, feature data includes information such as demographics, historical behaviors, device model, ad slots, etc. To avoid the skew between the production and offline experiment result, we keep sharing the single data pipeline in both of them.

## 5.2 Model Training

Once the entire data is ready, we train our model. Like the *Data Pipeline*, in order to avoid the skew between the production and offline system, we use the same code for them and this enables fast deployment in production as well.

The details of how to train our model is as follows: Since some additional conversion events may occur after the training as shown in Figure 2, we loads conversion and click data in memory and dynamically join them just before training. In the process of joining, the data is labeled with the number of conversions and a series of delays which makes our proposed model to handle multiple events. We repeatedly train our models with a short period of time to reflect newly occurred events to the model.

## 5.3 Model Serving

*Ranking Server* loads trained models that are distributed from *Model Trainer* as soon as the training is done. For each request, we compute eCPMs of all thousands of possible ad candidates that are filtered from hundreds of thousands valid ads by some business constraints. Then, the best ads that maximize revenue are selected. As mentioned in the bottom of Section 4.2, our model is computational efficient so that we complete the entire serving process on the order of 10 ms including retrieving features.

# 6 EXPERIMENT

## 6.1 Dataset & Preprocessing

There are well-known datasets for binary conversion problem on which various experiments are performed [5, 16]. To evaluate the impact of the models that predict count data, it is not suitable to use those binary target data.

Instead, our experiment is conducted with the real-traffic conversion logs collected from our various services. Among them, we choose conversion Type *A* at Table 1 which is in service by our model because it has sufficient traffic volumes and contains many multiple conversions.

We create 7 experiment datasets with consecutive periods to check the consistency of the results, and each dataset setting is as follows: 3 weeks of training data and the following day for the test data. For training data, we only count the conversions that occurred during the training period. However, for the test data, we consider all conversions which occurred eventually, after the test days within the maximum delay of conversions.

## 6.2 Evaluation Metrics

In this experiment, Mean Squared Error (MSE) and Calibration are used as our main evaluation metrics. Our data contains non-binary target values, it is unable to use Logloss as an evaluation metric. Instead, we choose MSE as our key metric.

*Mean Squared Error.* MSE is the average of squared sum of deviance from the actual. The smaller value, the better model.

*Calibration.* Calibration measures the average ratio of actual number of conversions to predicted number of conversions.

## 6.3 Competing Models

We compare our proposed *Negative Binomial Delayed Feedback Model* (NBDFM) with the following baselines:

*Delayed Feedback Model.* DFM treats conversions as a binary variable, i.e. all but the first conversion are ignored[5].

*Generalized Linear Model.* When the target events are count data, *Poisson* regression and *Negative Binomial* regression are popular modeling methods described in Section 2. We also include the classical *Logistic* regression that is widely used for binary classification problems.

*Delayed Feedback Model+Poisson Regression.* This model, an simple additive model of DFM and Poisson regression, is our first heuristic deployed model for multiple conversions. In (18), the expected number of conversions can be divided into two parts. The first part means the probability that one or more conversions occur and is modeled by DFM considering a delay of the first conversion. In the second part, Poisson regression predicts the number of conversions after the first conversion, not considering any delays of

them.

$$E(C) = \sum_{i=1}^{\infty} iP(C=i) = P(C \geq 1) + \sum_{i=2}^{\infty}(i-1)P(C=i) \quad (18)$$

*Oracle Generalized Linear Model.* The only but the big difference from GLMs described above is that *Oracle* model can look into the future, so that the model can train with entire conversions which are basically impossible to observe at the given training time. Therefore, the performance of this model could be considered as an upper bound in this experiment.

## 6.4 Parameter Settings

As described in Section 4, our models are trained by L-BFGS optimizer. For reproducibility, we provide the detail settings of our optimizer here. 5 correction pairs are used to approximate the Hessian matrix. The termination conditions are as follows: (1) gradient tolerance: $10^{-5}$, (2) function tolerance: $10^{-8}$, (3) max iterations: 300. For fair evaluation, all settings are the same for each model.

**Table 3: Overall weighted metric of different models on 7 test days. The column 'Diff' shows the difference of MSE between the given model and DFM.**

| | MSE | Diff | Calibration(%) |
|---|---|---|---|
| DFM | 0.09219 | - | 141.77 |
| Logistic | 0.09231 | -0.00012 | 146.60 |
| Poisson | 0.08681 | 0.00537 | 108.85 |
| Negative Binomial | 0.08682 | 0.00536 | 108.19 |
| DFM+Poisson | 0.08723 | 0.00496 | 106.25 |
| **NBDFM** | **0.08454** | **0.00764** | **101.12** |
| *Oracle* Logistic | 0.09223 | - | 140.82 |
| *Oracle* Poisson | 0.08298 | - | 100.34 |
| *Oracle* Negative Binomial | 0.08248 | - | 99.29 |

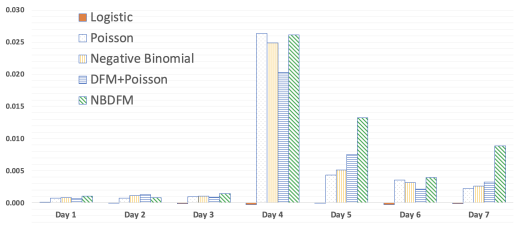


**Figure 5: 'Diff's on each test day except for *Oracle* models.**

## 6.5 Results

Table 3 shows that our proposed NBDFM is the best in terms of both MSE and Calibration except *Oracle* models which are considered as the upper bound. In case of MSE, our model achieved 0.08454 which is the lowest value among other competing models, and the calibration is the most closest to 100%. See also Figure 5 for daily results.

Since DFM and Logistic regression only consider the binary data, they under-predict a lot in terms of the calibration. Naive GLMs and our first deployed model DFM+Poisson are not good enough because they do not consider all the delays of conversions.

# 7 CONCLUSION

We solve the real-world problem of predicting number of conversions that are intrinsically count data, taking into account conversion delays. Our novel model is the extended DFM with combining Negative Binomial regression and Order Statistics. As a result, we achieved the notable model accuracy compared to the several baselines, and this model has been deployed to our production system.

# REFERENCES

[1] Galen Andrew and Jianfeng Gao. 2007. Scalable training of L 1-regularized log-linear models. In *Proceedings of the 24th international conference on Machine learning*. 33–40.
[2] Barry C Arnold, Narayanaswamy Balakrishnan, and Haikady Navada Nagaraja. 1992. *A first course in order statistics*. Vol. 54. Siam.
[3] A Alexander Beaujean and Morgan B Grant. 2016. Tutorial on using regression models with count outcomes using R. *Practical Assessment, Research, and Evaluation* 21, 1 (2016), 2.
[4] A Colin Cameron and Pravin K Trivedi. 2013. *Regression analysis of count data*. Vol. 53. Cambridge university press.
[5] Olivier Chapelle. 2014. Modeling delayed feedback in display advertising. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. 1097–1105.
[6] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & deep learning for recommender systems. In *Proceedings of the 1st workshop on deep learning for recommender systems*. 7–10.
[7] Brian Dalessandro, Claudia Perlich, Ori Stitelman, and Foster Provost. 2012. Causally motivated attribution for online advertising. In *Proceedings of the Sixth International Workshop on Data Mining for Online Advertising and Internet Economy*. 1–9.
[8] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: a factorization-machine based neural network for CTR prediction. *arXiv preprint arXiv:1703.04247* (2017).
[9] Xinran He, Junfeng Pan, Ou Jin, Tianbing Xu, Bo Liu, Tao Xu, Yanxin Shi, Antoine Atallah, Ralf Herbrich, Stuart Bowers, et al. 2014. Practical lessons from predicting clicks on ads at facebook. In *Proceedings of the Eighth International Workshop on Data Mining for Online Advertising*. 1–9.
[10] Joseph M Hilbe. 2011. *Negative binomial regression*. Cambridge University Press.
[11] Wendi Ji, Xiaoling Wang, and Feida Zhu. 2017. Time-aware conversion prediction. *Frontiers of Computer Science* 11, 4 (2017), 702–716.
[12] Michael H Kutner, Christopher J Nachtsheim, John Neter, William Li, et al. 2005. *Applied linear statistical models*. Vol. 5. McGraw-Hill Irwin New York.
[13] Robert Malouf. 2002. A comparison of algorithms for maximum entropy parameter estimation. In *proceedings of the 6th conference on Natural language learning-Volume 20*. Association for Computational Linguistics, 1–7.
[14] H Brendan McMahan, Gary Holt, David Sculley, Michael Young, Dietmar Ebner, Julian Grady, Lan Nie, Todd Phillips, Eugene Davydov, Daniel Golovin, et al. 2013. Ad click prediction: a view from the trenches. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. 1222–1230.
[15] Steffen Rendle. 2010. Factorization machines. In *2010 IEEE International Conference on Data Mining*. IEEE, 995–1000.
[16] Marcelo Tallis and Pranjul Yadav. 2018. Reacting to Variations in Product Demand: An Application for Conversion Rate (CR) Prediction in Sponsored Search. *arXiv preprint arXiv:1806.08211* (2018).
[17] Yuya Yoshikawa and Yusaku Imai. 2018. A Nonparametric Delayed Feedback Model for Conversion Rate Prediction. *arXiv preprint arXiv:1802.00255* (2018).
[18] Ya Zhang, Yi Wei, and Jianbiao Ren. 2014. Multi-touch attribution in online advertising with survival theory. In *2014 IEEE International Conference on Data Mining*. IEEE, 687–696.