Estimating Conversion Rate in Display Advertising from Past Performance Data

Kuang-chih Lee, Burkay Orten, Ali Dasdan, Wentong Li
Turn Inc., 835 Main Street, Redwood City, CA 94063
{klee, borten, adasdan, wli}@turn.com

ABSTRACT

In targeted display advertising, the goal is to identify the *best* opportunities to display a banner ad to an online user who is most likely to take a desired action such as purchasing a product or signing up for a newsletter. Finding the best ad impression, i.e., the opportunity to show an ad to a user, requires the ability to estimate the probability that the user who sees the ad on his or her browser will take an action, i.e., the user will *convert*. However, conversion probability estimation is a challenging task since there is extreme data sparsity across different data dimensions and the conversion event occurs rarely. In this paper, we present our approach to conversion rate estimation which relies on utilizing past performance observations along *user*, *publisher* and *advertiser* data hierarchies. More specifically, we model the conversion event at different *select* hierarchical levels with separate binomial distributions and estimate the distribution parameters individually. Then we demonstrate how we can combine these individual estimators using logistic regression to accurately identify conversion events. In our presentation, we also discuss main practical considerations such as data imbalance, missing data, and output probability calibration, which render this estimation problem more difficult but yet need solving for a real-world implementation of the approach. We provide results from real advertising campaigns to demonstrate the effectiveness of our proposed approach.

Categories and Subject Descriptors

H.3.5 [**Information Storage and Retrieval**]: Online Information Services; G.3 [**Probability and Statistics**]: Correlation and regression analysis; I.2.6 [**Artificial Intelligence**]: Learning

General Terms

Algorithms, Experimentation, Theory

KDD'12, August 12–16, 2012, Beijing, China.

Keywords

Action rate estimation, algorithmic advertising, computational advertising, logistic regression

1. INTRODUCTION

In online display advertising world, advertisers try to market their product to many users by embedding their graphical advertisement within the content in publisher web pages, e.g., the pages of news portals. The advertiser's main goal is to reach the most receptive online audience in the right context, which will then engage with their displayed ad and eventually take a desired action, identified by the type of the campaign, e.g., brand advertising or direct product marketing. The complexity of realizing this goal is so high that advertisers need specialized technology solutions called demand-side platforms (DSP).

DSPs help manage such display advertising campaigns for many different advertisers simultaneously across multiple direct buy markets or ad exchanges where ad impressions can be acquired through real-time auctions or bidding. In a direct buy market, the impression price is decided in advance according to the business negotiations between publishers and advertisers directly. On the other hand, in a real-time ad exchange, a bid has to be submitted for each impression (submitted via ad calls) by DSPs and the impression is sold to the highest bidder in a public auction. DSPs are the platforms where all the information about users, pages, ads, and campaign constraints come together to make the best decision for advertisers (see § 2.1 for an overview).

Advertisers seek the optimal price to bid for each ad call to maximize their campaign performance. The optimal bid price of an ad impression depends on the *value* of that impression to a given advertiser. This value is provided to DSPs as a campaign performance parameter in the form of cost-per-click (CPC) or cost-per-action (CPA) goals. If a CPC or CPA goal is set up, then the optimal bid price can be determined from the *expected cost-per-impression*, which is equal to the click-through-rate (CTR) for this impression multiplied by the CPC goal, or the conversion rate (CVR) multiplied by the CPA goal [8].

In this scenario, campaign performance directly depends on how well the CTR or CVR can be estimated and the performance optimization can be considered as the problem of accurately estimating CTR or CVR. If these quantities are overestimated, bid prices will always be higher than what they should be, the advertiser will waste campaign budget on useless impressions; on the other hand, if these quantities are underestimated, the advertiser will miss high-value

impressions that may have led to actions and the campaign will under deliver.

The CTR and the CVR are directly related to the intention of the user interacting with the ad in a given context and they are fundamentally difficult to directly model and predict. In practice, CVR is even harder to estimate than CTR since conversion events are far rarer than click events. Also, the view-through conversions have longer delays in the logging process (sometimes up to a week), which makes the off-line modeling more difficult. Finally, the ad serving system needs to perform in real-time, requiring the CVR or CTR estimation, optimal bid price calculation, and bid submission to exchanges to be completed in a few milliseconds.

In this paper, we present a simple but effective approach to estimating CVR of ad impressions, to be used in a DSP. Our conversion rate estimation method models the conversion event at different *select* levels of the cross-product of *user*, *publisher* and *advertiser* data hierarchies with separate binomial distributions and estimate the distribution parameters individually. These individual estimators are than combined using logistic regression to obtain a final estimate that can more accurately predict the conversion outcome for each impression.

The rest of the paper is as follows. In § 2, we review the ad call flow, formulate our problem, and discuss previous related work in the literature. In § 3, we describe the CVR estimators using the past performance observations across data hierarchies and also describe the use of logistic regression as a means of combining these individual rate estimates for improved predictive performance. Various practical issues encountered during CVR modeling in a DSP and the proposed solutions are discussed in § 4. Thorough experimental results are presented in § 5, and § 6 lists some concluding remarks and possible future work.

2. BACKGROUND AND RELATED WORK

In this section, we first review the ad call flow. We then formulate the problem of estimating the conversion rate of an event given an ad impression. Finally, we explain the user, publisher and advertiser data hierarchies and also discuss some previous work in the literature related to the estimation of rare events.

2.1 Overview of Ad Call Flow

To put the proposed approach in perspective, let us use the simplified ad call flow in Fig. 1. Thinking of the rectangular bounding box around this figure, the right side is the supply or publisher side. The supply side provides the pages on which there are places that ads can go. The left side is the demand or advertiser side. The demand side provides the ads and constraints such as targeting and budget constraints.

Given a user with a profile, a page with some context, and campaigns from various advertisers (where a campaign include some constraints and ads), the goal (sometimes called the fundamental problem of computational advertising) is to find the best ad under constraints for the user and the page. The mechanics of how this is done is explained next with the help of numbered steps in Fig. 1.

The call is initiated when a user with id <uid> starts to open a web page at URL <url>. The call first reaches the ad exchange the publisher is integrated with. The ad exchange appends <uid> and <url> to a request that it sends to its demand side platform (DSP) partners to ask for an ad and a bid. The request usually contains other information such as geographical location for targeting purposes. The partner DSPs may or may not respond to the request; those that respond (bidding DSPs) need to use <uid> and <url> to gather more information about the user and the page, respectively. (Information about users and pages are usually provided via data management platforms.) The bidding DSPs also search through their respository of previously created campaigns to find a set of candidate ads that can be targeted to the user and the page. The contribution in this paper is at the hearth of the decision engine that selects the best ad from the candidate ads and computes the corresponding bid.

The bidding DSPs return back their proposed ad and bid pair back to the ad exchange. Once the ad exchange collects all the bids, it runs an auction (usually a second price auction) and determines the winning bid and the corresponding ad. It then passes the winning ad and the location of its creative back to the browser. The browser then collects the creative and finally returns the page to the user.

Note that this entire flow needs to happen within one-tenth of a second so that the user can see the page with ads as soon as she opens the page on her web browser. Moreover, the top DSPs can get such ad calls as many as 500,000 times per second as 100s of millions of users open pages on their browsers at any given instant. This latency and throughput constraints put extreme time constraints on each bidding DSP and their decision engines. The proposed approach works under such time constraints and performs well.

2.2 Problem Setup and Formulation

Per § 2.1, an ad call request is sent along with the user and publisher information. Let $\mathcal{U} = \{u_1, u_2, ..., u_l\}$, $\mathcal{P} = \{p_1, p_2, ..., p_m\}$ and $\mathcal{A} = \{a_1, a_2, ..., a_n\}$ represent all the users, publisher web pages and ads (with creatives) in a DSP, respectively. For simplicity, an ad call request to a DSP can be considered to arrive in the form $adrequest = \{user : u_i, page : p_j\}$. The goal is to find the ad among all n ads that has the highest probability of conversion. Mathematically, this can be written as:

$$ad^* = \underset{k=1,...,n}{\arg\max}\, p(conversion|user = u_i, page = p_j, ad = a_k) \quad (1)$$

where ad^* denotes the optimal ad, i.e., the ad with the highest CVR. Note that at a single user level, an event has only two possible outcomes: conversion or no conversion. Then we can see that a conversion event can be modeled as a *Bernoulli* random variable. Let $Y_{ijk} \in \{0, 1\}$ represent the random outcome of the event, when the user u_i is shown ad a_k on site p_j. Then, we can compactly write:

$$Y_{ijk} \sim Bernoulli(p_{ijk}),\ p_{ijk} = p(Y = 1|u_i, p_j, a_k). \quad (2)$$

and with this new representation, equation (1) becomes:

$$ad^* = \underset{k=1,...,n}{\arg\max}\, p(Y = 1|u_i, p_j, a_k) \quad (3)$$

One way of modeling the conversion event outcome is to represent the user, publisher and advertisement using a set of explicit features and build a classification model. Examples of this approach in the sponsored search advertising context can be found [10, 15]. Since low-level data features are poorly correlated with a user's direct intention of taking

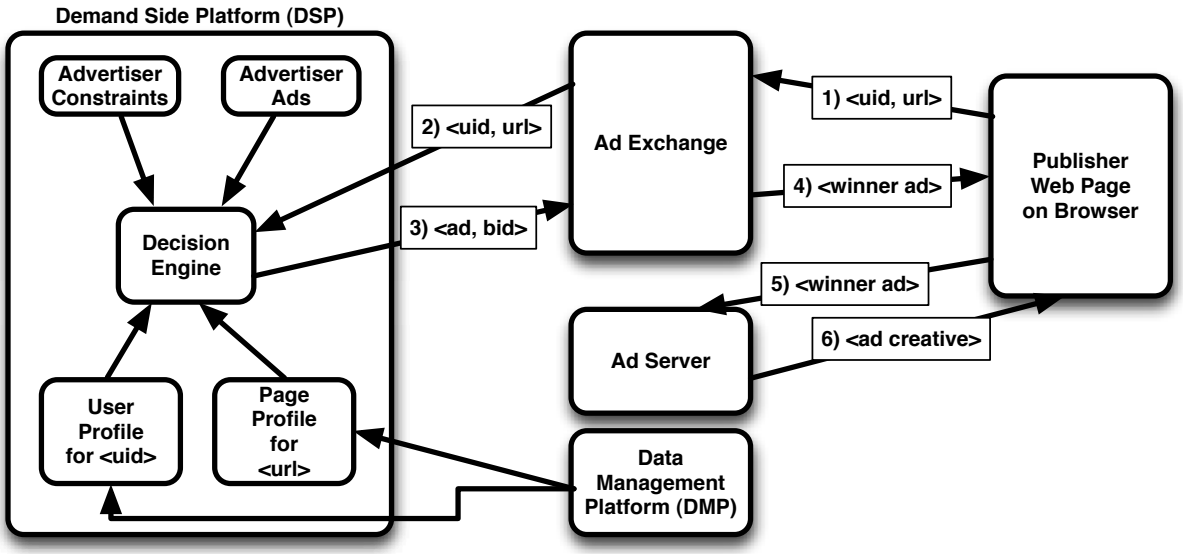

Figure 1: Ad call flow between a web browser and a demand side platform, the place where all information is collected to make the optimal decision. A data management platform aggregates user data. The ad exchange is used to run the auction to match the winning ad to the page and the price of the match.

an action on a display ad, models that are based on these features perform poorly in our context. An alternative idea is to compute the number of times this ad was displayed to "similar" users on this website and observe how many of these impressions result in a conversion. Then, the conversion rate for this user can simply be estimated as the total number of conversions among all similar users divided by the total number of impressions. Note that this is nothing but the *maximum likelihood estimate* (MLE) of the conversion rate and can be written as follows:

$$\begin{aligned} &\tilde{Y}_{ijk} \sim Binomial(T_{ijk}, \tilde{p}_{ijk}), \\ &\tilde{p}_{ijk} = p(Y = 1 | u \in UsrClust_i, p_j, a_k), \\ &\tilde{p}_{ijk}^{\text{MLE}} = \begin{cases} \frac{S_{ijk}}{T_{ijk}}, & \text{if } T_{ijk} \neq 0 \\ \text{unknown}, & \text{if } T_{ijk} = 0, \end{cases} \end{aligned} \tag{4}$$

where T_{ijk} denotes the number of trials (i.e., impressions) and S_{ijk} denotes the number of successes (i.e., conversions) when ad a_k was shown to users from $u \in UsrClust_i$ on website p_j. It is important to point out that in equation (4), i does not index the particular user whose conversion rate we are trying to estimate, but it indexes a cluster of users whose conversion rates are assumed to be similar to this user. Subsequently, $\tilde{Y}_{ijk}$ represents the conversion event of all the users in cluster $UsrClust_i$ (hence Binomial) whereas Y_{ijk} represents only the event of user i converting (i.e., Bernoulli event). This kind of user grouping can either be achieved by explicit clustering based on some similarity metric or it can be implicitly done by using data hierarchies as we will explain in the next section.

2.3 Data Hierarchies

In the display advertising context user, publisher and advertiser related data can be considered as adhering to some hierarchical structure. For example, every ad in the DSP can be considered as belonging to an advertising campaign which in turn belongs to an advertiser (e.g., Advertiser: 'Acme Cars' → Campaign: '2011 Year End Sales' → Ad: 'Incredible Year End Sales Event!'). Similarly, a website on which an ad will be displayed is under a top level domain (TLD) which is owned by a publisher and the publisher itself might belong to some category based on its main content (e.g., Publisher Type: 'News' → Publisher: 'Acme City Times' → Page: 'Auto News'). An example taxonomy demonstrating this hierarchical structure for the user, publisher and advertiser data is shown in Fig. 2.2. Note that this figure does not represent Turn Inc.'s data taxonomy and it is mainly provided for illustrative purposes and clarity of presentation.

The data hierarchies enable us to define explicit or implicit 'user clusters'. Explicit user clustering, as one might guess, is based on representing each user by a set of features (e.g., demographic information, geographic features, income level, type of websites they frequently visit, activity level etc.) and clustering them based on some similarity metric such as the Euclidean distance. On the other hand, implicit clustering is based on using data hierarchies rather than the user features. For example, the group of users who visit websites in a certain category, such as sports, can be considered as an implicit cluster. We can represent this grouping by the Cartesian product {Users × Publisher Type}. As another example, we can also consider all the users who were shown an ad from ad-campaign (such as 'Acme Cars Year End Sales') on a particular site (e.g., Acme Times Autos), which can be represented by the Cartesian product {Users × Publisher Type × Campaign}. If we assume that user, publisher and advertiser data have ℓ_u, ℓ_p and ℓ_a levels in their respective data hierarchies, there are $\ell_u \times \ell_p \times \ell_a$ such possible implicit user clusters. Then, given that $adrequest = \{user : u_i, page : p_j\}$, we can identify the suitable explicit and implicit user clusters from the data hierarchies and use past count data (i.e., number of impressions and number of conversions) of each level to get different estimates of

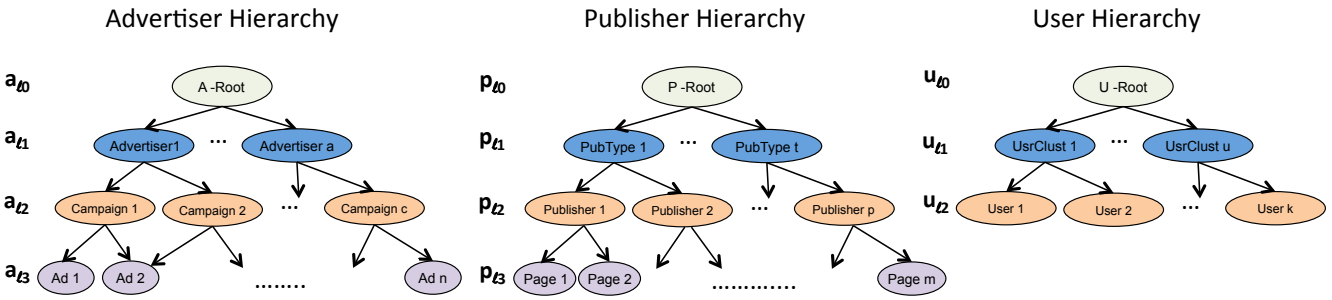

Figure 2: Sample user, publisher and advertiser taxonomy representing the hierarchical structure of the data. The subscript ℓ_i designates the i-th level in each component's data hierarchy.

$p_{ijk} = p(Y = 1|u_i, p_j, a_k)$. This is explained in more detail in § 3.1.

2.4 Related Work

Among the few attempts aiming to utilize the data hierarchies to estimate the CVR/CTR [2, 1, 19, 14], Agarwal et al. [1, 19] relax the dependency between user information and (advertiser, publisher) pairs in the following:

$$p(Y = 1|u_i, p_j, a_k) \propto p(Y = 1|u_i)p(Y = 1|p_j, a_k). \quad (5)$$

They propose to estimate $p(Y = 1|u_i)$ by regression methods on user features, such as demographic, geographic, and temporal (recency/frequency) information, and $p(Y = 1|p_j, a_k)$ by some probabilistic inference on pre-defined strict hierarchies of (publisher, advertiser) pairs. If the pair (publisher, advertiser) does not have enough number of successes, the CVR/CTR estimation will be based on its parent node. Menon et al. utilizes the same hierarchical constraints to estimate CVR/CTR by using collaborative filtering [14]. We argue that those approaches are hard to be carried out in our setting. First, the ordering of hierarchies is not straightforward in some cases. For example, it is hard to judge that tier (site quality) information or inventory source (exchanger id) should be the parent node given a publisher site. Furthermore, our proposed hierarchies contain user information and the structure becomes more complicated. As the level of hierarchies becomes deeper, the probabilistic inference proposed in [2, 1] will be hard to satisfy the real-time ad serving constraint.

On the contrary, there are also some studies that try to cluster users based on their behaviors and interests without considering data hierarchy. Cerrato et al. [7] propose a rule-based approach to learn a set of DNF rules that maximize the coverage of conversion users. Ahmed et al. [3] present an unsupervised user clustering framework based on Latent Dirichlet Allocation [5], where each user cluster captures some user interests directly. They demonstrate that this framework can be applied for audience selection in display advertising.

Richardson et al. [15] apply logistic regression to estimate CTR using relative term CTR, ad quality, and other features for search engine advertising. Our work bears some resemblance to their method in the sense that both methods combine a set of individual CVR/CTR estimators in different context using logistic regression to obtain a final estimate. However in this paper, we consider our individual estimators come from different levels of domains ($user, publisher, advertiser$) in data hierarchies, which capture versatile information compared to term CTR used in [15]. Furthermore, since conversion events are typically sparser than click events, our implementation needs to handle many practical issues, such as data skewness and missing features, and our proposed solutions are discussed in § 4.

3. CONVERSION RATE ESTIMATION

In this section, we provide details of our conversion rate (CVR) estimation method. We first revisit data hierarchies and explain how past observations from different levels in the hierarchies yield 'weak' estimators of the desired CVR for a given ad call. Afterwards, we discuss how we can combine these weak estimators using logistic regression to get a more accurate estimator.

3.1 Past Performance at Different Hierarchical Levels

Recall from equation (1) that given an ad request parameterized by $\{user : u_i, page : p_j\}$, we would like to find the ad, a_k, which has the highest CVR, p^*_{ijk}. As we explained in the last section, if we can identify a group of users whose CVR is similar to that of user u_i's, then we can make the following approximation:

$$p_{ijk} = p(Y = 1|u_i, p_j, a_k) \approx \tilde{p}_{ijk} = p(\tilde{Y} = 1|u \in \mathcal{C}_{u_i}, p_j, a_k) \quad (6)$$

where $\mathcal{C}_{u_i}$ denotes a cluster of users in which u_i belongs. Note that this approximation relies on an explicit user clustering and observes all the users in $\mathcal{C}_{u_i}$ who were shown ad a_k when they visited website p_j. Then, the simplest possible estimator for p_{ijk} is given by:

$$\hat{p}_{ijk} = \hat{\tilde{p}}^{\text{MLE}}_{ijk} = \begin{cases} \frac{S_{ijk}}{T_{ijk}}, & \text{if } T_{ijk} \neq 0 \\ \text{unknown}, & \text{if } T_{ijk} = 0, \end{cases} \quad (7)$$

this simple estimator has the advantage that if all the users in cluster $\mathcal{C}_{u_i}$ have the same CVR and if we have increasingly more observations T_{ijk}, $\hat{p}_{ijk}$ will converge to the true value p_{ijk} (i.e., estimator is consistent). The first challenge is to identify a group users who truly have the same or very similar CVRs. This is related to identifying the users' conversion/purchase intentions and inherently a difficult prob-

lem as we pointed out earlier. Even if we can find such clusters that are reasonably *homogeneous* in terms of users' behaviors (and hence their CVRs), the main challenge is to collect enough data at the granularity of ad a_k and website p_j. In other words, we generally will not have enough observations at (ad, website) level. Mathematically, this means that T_{ijk} will usually be slow. Also recall that conversion is a very rare event. Therefore, for a true conversion rate that is on the order of 10^{-5}, we need many impressions (on the order of millions, e.g., see the formulas in [4]) at the (ad, website) level before we can get a reliable maximum likelihood estimate. This data sparsity problem can be alleviated by going up in the user, publisher and advertiser data hierarchies since there will certainly be more observations at higher levels than there are at lower levels. We can then generalize equation (6) as follows:

$$p_{ijk} \approx \tilde{p}_{ijk} = p(\tilde{Y} = 1 | u \in \mathcal{C}_{u_i}, p \in \mathcal{C}_{p_j}, a \in \mathcal{C}_{a_k}). \quad (8)$$

In equation (8), $\mathcal{C}_{p_j}$ and $\mathcal{C}_{a_k}$ define grouping of the webpages and advertisers, respectively. These groupings are dictated by the respective data hierarchies as shown in Fig. 2.2. For example, at level $\ell_a = \ell_2$, we group together all ads that belong to the same campaign and at level $\ell_p = \ell_1$ we group together all webpages of all publishers that are of the same type (e.g., automotive related publishers). Then we can identify the groups at levels ℓ_u, ℓ_p and ℓ_a as $\mathcal{C}_{u_i} = \{\mathcal{C}_u : u_i \in \mathcal{C}_u^{\ell_u}\}$, $\mathcal{C}_{p_j} = \{\mathcal{C}_p : p_j \in \mathcal{C}_p^{\ell_p}\}$ and $\mathcal{C}_{a_k} = \{\mathcal{C}_a : a_k \in \mathcal{C}_a^{\ell_a}\}$, respectively. In this compact representation, $\mathcal{C}_u^{\ell_u}$ identifies all the user groups at the ℓ_u-th level in the user hierarchy. $\mathcal{C}_p^{\ell_p}$ and $\mathcal{C}_a^{\ell_a}$ similarly identify the webpage and ad groupings at the ℓ_p-th and ℓ_a-th levels.

As we identified in § 2.3, if the user, publisher and advertiser data have ℓ_u, ℓ_p and ℓ_a levels in their respective hierarchies, there are $\ell_u \times \ell_p \times \ell_a$ possible ways of combining count data from different levels. Let $\hat{\tilde{p}}^1_{ijk}, ..., \hat{\tilde{p}}^M_{ijk}$ denote maximum likelihood estimates of the CVRs at M different combinations of (user, publisher, advertiser) hierarchical levels. For example, $\tilde{p}^1_{ijk}$ can be $p(Y = 1 | u \in \mathcal{C}^{\ell_2}_{u_i}, p \in \mathcal{C}^{\ell_3}_{p_j}, a \in \mathcal{C}^{\ell_3}_{a_k},)$ and $\tilde{p}^M_{ijk}$ can be $p(Y = 1 | u \in \mathcal{C}^{\ell_1}_{u_i}, p \in \mathcal{C}^{\ell_1}_{p_j}, a \in \mathcal{C}^{\ell_1}_{a_k},)$

While going up along the data hierarchies solves the problem of data sparsity, it introduces bias in the CVR estimation. This is not really surprising since grouping together more distinct type of users at higher levels violates the assumption that the users in the same group have similar CVRs. Consequently, many of the M estimators are quite 'weak' in the sense that their CVR values, $\tilde{p}_{ijk}$, are poorly correlated with the true p_{ijk}. Therefore, many of these weak estimators are discarded and we use only a *select* few set of estimators that are highly correlated with the true CVR. In the next section, we discuss how we can combine these weak estimators to obtain a final estimator that has better predictive performance.

3.2 Combining Estimators using Logistic Regression

Following the notation of § 3.1, let $\hat{\tilde{p}}^1_{ijk}, ..., \hat{\tilde{p}}^M_{ijk}$ denote maximum likelihood estimates of the CVRs of events at M different levels. Recall that each of these levels correspond to a distinct combination of the user, publisher and advertiser hierarchies and, as such, it is not always clear which one will yield the best estimate of the true event CTR, p_{ijk}. Rather than trying to pick the best one among all M estimators, we seek to optimally combine these estimators. In other words, we would like to obtain:

$$p_{ijk} = f(\hat{\tilde{p}}^1_{ijk}, ..., \hat{\tilde{p}}^M_{ijk}; \boldsymbol{\beta}) \quad (9)$$

for some function $f(\cdot) : [0,1]^M \rightarrow [0,1]$ which has a set of parameters denoted by $\boldsymbol{\beta}$. Note that for every impression, the outcome of all M estimators can be computed and logged in a database. Additionally, in the 'ad-exploration' stage, we can serve impressions with different ads and observe whether the impression resulted in a conversion or not. Assume that for a running campaign we collect such training data for N different impressions. Let $y_s \in \{0, 1\}$ encode the conversion outcome of the s-th impression and $\hat{\tilde{p}}^{s1}, ..., \hat{\tilde{p}}^{sM}$ denote the associated maximum value estimates of the CVR value for the same impression, where $s = 1, ..., N$. Note that for clarity of presentation we dropped the subscripts i, j and k and compactly represent the impression by the subscript s. However, it should be clear that every impression is served to a different user on a new website with a different ad and the dependence of the CVR p^s on these data components is always implied. The log-likelihood of this training data can be written as follows:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\beta}) &= \sum_{s=1}^{N} y_s \log p^s(\boldsymbol{\beta}) + (1 - y_s) \log(1 - p^s(\boldsymbol{\beta})) \\ &= \sum_{s=1}^{N} y_s \log f(\hat{\tilde{p}}^{s1}, ..., \hat{\tilde{p}}^{sM}; \boldsymbol{\beta}) + \\ &\qquad (1 - y_s) \log(1 - f(\hat{\tilde{p}}^{s1}, ..., \hat{\tilde{p}}^{sM}; \boldsymbol{\beta})) \quad (10)\end{aligned}$$

We can use log-likelihood as a goodness-of-fit measure and choose the set of parameters $\boldsymbol{\beta}$ such that they will maximize the $\mathcal{L}(\boldsymbol{\beta})$ over the training set. In our system we are using the sigmoid function $f(\mathbf{x}; \boldsymbol{\beta}) = \frac{1}{1+e^{-\boldsymbol{\beta}^T \mathbf{x}}}$ to combine different CVR estimates since it always produces estimates between 0 and 1 and the resulting model can be interpreted easily by comparing their linear combination coefficients. With this choice of $f(\cdot)$, the optimum set of parameters can be found by solving the following optimization problem:

$$\begin{aligned}\boldsymbol{\beta}^* &= \arg\max_{\boldsymbol{\beta}} \mathcal{L}(\boldsymbol{\beta}) \\ &= \arg\max_{\boldsymbol{\beta}} \sum_{s=1}^{N} y_s [\boldsymbol{\beta}^T \hat{\tilde{\mathbf{p}}}^s - \log(1 + \boldsymbol{\beta}^T \hat{\tilde{\mathbf{p}}}^s)] \quad (11)\end{aligned}$$

where $\hat{\tilde{\mathbf{p}}}^s = [\hat{\tilde{p}}^{s1}\ \hat{\tilde{p}}^{s2}\ ...\ \hat{\tilde{p}}^{sM}]^T$. Another way to interpret this formulation is to think of the M individual estimators as factors (or features) in a classification model and the optimization process as finding the optimal linear combination coefficients that will classify the training data (i.e., the conversion impressions and no conversion impressions) as accurately as possible. However, we should point out that since we are not actually interested in classifying an impression but rather in estimating its probability of conversion, we only need the probabilistic scores assigned by logistic regression and we never have to select a classification threshold.

In order to measure how well the estimated p^s values explain the training data, we can compute the data likelihood using the fitted $\boldsymbol{\beta}^*$ as shown in equation (10). We can also measure the performance by treating the logistic regression output as classification scores and computing the *area under the receiver operating characteristics curve*. We will refer to this as the AUC metric. This metric explains how well the

logistic regression scores of the conversion and no conversion impressions separate.

4. PRACTICAL ISSUES IN CONVERSION RATE ESTIMATION

In this section, we discuss several practical issues encountered in implementing our proposed CVR estimation method in our DSP and present our current solutions.

4.1 Data Imbalance

When we consider the CVR estimation as a classification problem (see § 3.2), we ideally would like to have sufficient amount of training examples from both the conversion and no-conversion event classes. However, typical conversion rates in a display advertising campaign ranges anywhere from 10^{-3} to 10^{-6}. In other words, only 1 out of a million impressions may result in a conversion event on average. Hence, any training data we collect will be highly skewed towards the no-conversion class. In the learning context there are different scenarios that might result in skewed training data [16, 17], but in our problem the two main reasons for having data imbalance can be identified as follows:

1. The average conversion rate of an advertising campaign is inherently very low and we do not get to observe sufficient amount of conversion events.

2. The ratio of the number of no-conversion events to that of conversion events is very large, but we still have reasonable amount of data from the minority (i.e., conversion) class.

In the first scenario, we do not have enough number of representative examples from the minority class and we can not reliably train a CVR estimation model. Then, we simply use a different manually-crafted, rule-based model that combines the base estimators $\hat{p}^{s1}, ..., \hat{p}^{sM}$ and never train a logistic regression model.

In the second case we are able to train a logistic regression model using the skewed data. Performance of a model when trained on a highly skewed data set has to be carefully analyzed and as Visa et al. [16] and Weiss [17] suggested, we use AUC as a metric since it is more robust in imbalanced data scenarios. In our experiments, we observed that data imbalance does not necessarily effect the AUC performance of the logistic regression model as long as we have sufficient amount of examples from the minority class and this observation is also verified by Maloof [12]. While building a CVR estimation model for a specific campaign, we try to keep all examples from the minority class to ensure sufficient representation but only a subset of the majority class due to memory constraints.

4.2 Output Calibration

For training data collection, we use stratified sampling strategy and only retain a small subset of the examples from the no-conversion class. This strategy results in a training data set where the proportion of the examples from different classes does not match the true data proportions (i.e., the inherent conversion rate for the campaign). Subsequently, the output scores of the logistic regression model optimized on such training data do not represent the actual scale of the CVR values for this advertising campaign. Since the bidding process requires accurate estimates of the CVR values, we need to calibrate the logistic regression output scores to take into account the true proportions of the event classes. The goal of the calibration process can be simply expressed as finding the true CVR value for the s-th impression given the logistic regression score:

$$\hat{p}^s = p(Y^s = 1 | f(\hat{\vec{\mathbf{p}}}^s; \boldsymbol{\beta}^*)) \tag{12}$$

where $f(\hat{\vec{\mathbf{p}}}^s; \boldsymbol{\beta}^*)$ represent the logistic regression score given the M weak estimator values and $\hat{p}^s$ represent the *final* CVR estimate for this impression after the calibration. One way of obtaining this mapping from the logistic regression score to the calibrated output is to assume a parametric form for the density in equation (12) and compute its parameters using the training examples. However, this procedure requires some information or intuition about what the true density might look like. Instead, we follow an empirical approach and make the following approximation:

$$\hat{p}^s \approx p(Y^s = 1 | v_1 \leq f(\hat{\vec{\mathbf{p}}}^s; \boldsymbol{\beta}^*) < v_2) \tag{13}$$

for some score values within the range v_1 and v_2. Specifically, we group the logistic regression output scores into n equal sized bins, where $0 \leq v_1 < v_2 < \ldots, v_{n+1} \leq 1$ define the range of scores in each bin. Then we can approximate the calibration output $\hat{p}^s$ for the i-th bin using the maximum value estimation as follows:

$$\begin{aligned} \hat{p}(i) &\approx p(Y^s = 1 | v_i \leq f(\hat{\vec{\mathbf{p}}}^s; \boldsymbol{\beta}^*) < v_{i+1}) \\ &\approx \frac{\#\text{ Conversion examples with } f(\hat{\vec{\mathbf{p}}}^s; \boldsymbol{\beta}^*) \in [v_i, v_{i+1})}{\#\text{ All examples with } f(\hat{\vec{\mathbf{p}}}^s; \boldsymbol{\beta}^*) \in [v_i, v_{i+1})} \end{aligned} \tag{14}$$

In other words, calibration procedure becomes a simple conversion rate estimation in each score bin. As a result, we obtain the calibrated CVR for every bin $\hat{p}(1), ..., \hat{p}(n)$. At runtime, when we need to estimate the CVR for a new impression, we first compute the logistic regression score and convert it to the final estimate by:

$$\begin{aligned} \hat{p}^{Test} &\approx p(Y^{Test} = 1 | v_i \leq f(\hat{\vec{\mathbf{p}}}^{Test}; \boldsymbol{\beta}^*) < v_{i+1}) \\ &= \alpha \hat{p}(v_i) + (1 - \alpha) \hat{p}(v_{i+1}) \end{aligned} \tag{15}$$

assuming the logistic regression score falls within $[v_i, v_{i+1})$, where $\alpha = \frac{v_{i+1} - f(\hat{\vec{\mathbf{p}}}^{Test}; \boldsymbol{\beta}^*)}{v_{i+1} - v_i}$. In practice, even the bin conversion rates, $\hat{p}(v_i)$, may be not accurately estimated in case there is only a small number of observations in a score bin. For this reason we employ some post-processing to smoothen the $\hat{p}(v_i)$ values. Specifically, we enforce that $\hat{p}(v_i)$'s are monotonically increasing by applying Pool Adjacent Violators Algorithm (PAVA) [9]. The whole conversion curve can be further smoothed by fitting an exponential function to it.

4.3 Missing Features

Even though we would like to combine all the weak estimator outputs $\hat{\vec{p}}^1, ..., \hat{\vec{p}}^M$ using logistic regression, some of these values may not be available for certain impressions. For example, if user IDs are not found in the user profile servers, or the publisher's webpage does not match any of the known categories in the publisher taxonomy, the estimators using these information sources simply can not be computed. Moreover, there may not be sufficient number of conversion events in one of the hierarchical levels to calculate a reliable estimator output using the past performance observations.

Little et al. [11] classify the nature of missing data into missing completely at random (MCAR), missing at random (MAR), and not missing at random (NMAR). For simplicity, we assume that our impression data follows the MAR pattern, meaning that whether some portion of the data is missing or not only depends on the data being observed. Under the MAR assumption, one simple but effective approach widely used in the data mining community to handle missing data is called *imputation*, which attempts to complete the missing attributes by filling them with specific values.

Here, we present a probabilistic framework for missing value imputation, which subsumes the strategy that we ended up using. Assume that the past performance estimators of a given impression s are partitioned into two groups $\hat{\tilde{\mathbf{p}}}^{s} = [\hat{\tilde{\mathbf{p}}}^{so_s}, \hat{\tilde{\mathbf{p}}}^{sm_s}]^T$, where $\hat{\tilde{\mathbf{p}}}^{so_s}$ and $\hat{\tilde{\mathbf{p}}}^{sm_s}$ represent the sub-vectors for the observed and missing past performance estimators, respectively. Furthermore, assume that $\hat{\tilde{\mathbf{p}}}^{so_s}$ and $\hat{\tilde{\mathbf{p}}}^{sm_s}$ are jointly Gaussian, so that

$$\begin{aligned} \hat{\tilde{\mathbf{p}}}^{s} &\sim \mathcal{N}(\boldsymbol{\mu}, \Sigma), \\ \hat{\tilde{\mathbf{p}}}^{so_s} &\sim \mathcal{N}(\boldsymbol{\mu}_{o_s}, \Sigma_{o_s}), \\ \hat{\tilde{\mathbf{p}}}^{sm_s} &\sim \mathcal{N}(\boldsymbol{\mu}_{m_s}, \Sigma_{m_s}) \end{aligned} \tag{16}$$

where $\boldsymbol{\mu} = [\boldsymbol{\mu}_{m_s}^T, \boldsymbol{\mu}_{o_s}^T]^T$, and $\Sigma = \begin{pmatrix} \Sigma_{m_s} & \Sigma_{m_s o_s} \\ \Sigma_{m_s o_s}^T & \Sigma_{o_s} \end{pmatrix}$.

One can show that $\hat{\tilde{\mathbf{p}}}^{sm_s} | \hat{\tilde{\mathbf{p}}}^{so_s}$ is also normally distributed:

$$\begin{aligned} \hat{\tilde{\mathbf{p}}}^{sm_s} | \hat{\tilde{\mathbf{p}}}^{so_s} \sim \mathcal{N}\Big(& \boldsymbol{\mu}_{m_s} + \Sigma_{m_s o_s} (\Sigma_{o_s})^{-1} (\hat{\tilde{\mathbf{p}}}^{so_s} - \boldsymbol{\mu}_{o_s}) \, , \\ & \Sigma_{m_s} - \Sigma_{m_s o_s} (\Sigma_{o_s})^{-1} \Sigma_{m_s o_s}^T \Big). \end{aligned} \tag{17}$$

Then, it is easy to see that the optimal estimate of $\hat{\tilde{\mathbf{p}}}^{sm_s}$ (in the least squares sense) is the mean of the conditional distribution in equation (17), which we refer to as the *Bayesian Least Squares Estimator* (BLSE). We can adopt this framework for missing value imputation by replacing the Gaussian parameters in equation (17) with their empirical estimates. Specifically, we can replace $\boldsymbol{\mu}$ by its sample mean estimate, and Σ by the sample covariance matrix, which is computed using the pair-wise complete observation scheme. Then, the BLSE-based imputation method estimates the missing attribute values using:

$$\hat{\tilde{\mathbf{p}}}_{BLSE}^{sm_s} = \hat{\boldsymbol{\mu}}_{m_s} + \hat{\Sigma}_{m_s o_s} (\hat{\Sigma}_{o_s})^{-1} (\hat{\tilde{\mathbf{p}}}^{so_s} - \hat{\boldsymbol{\mu}}_{o_s}) \tag{18}$$

Other common imputation strategies, such as unconditional mean (or median) imputation, and conditional mean (or median) imputation, can be treated as simple variations or special cases of this probabilistic framework. For example, mean (or median) imputation does not use the second term in equation (18) and simply estimates the missing attribute values using the columnwise means (or medians). There are other imputation methods in the literature that are worth mentioning, such as imputation by singular value decomposition (SVD) [6], Gaussian mixture modeling (GMM) [18] and Collaborative Filtering [13]. However, those approaches come with the cost of increased computational complexity with little or no effect on the final CVR estimation.

After experimenting with different strategies (see § 5.2), we decided to use unconditional median imputation, which simply fills in all the missing values for an attribute by the median of all the non-missing values for the same attribute among the training examples. Median imputation is an attractive choice since its computation time is very small (satisfying the runtime constraints) and its performance is on par with the other computationally expensive approaches.

4.4 Feature Selection

The final practical step we would like to mention is a simple 'feature selection step'. Considering the past performance estimators $\hat{\tilde{p}}^1, ..., \hat{\tilde{p}}^M$ as features for the logistic regression algorithm, we first analyze the data to observe its attribute statistics. If many of the training examples have a certain attribute missing, we do not want to impute that attribute from a very limited set of examples as it will result in a poor model fit with a poor predictive performance. In our modeling approach, we discard such attributes if more than %65 of the training examples have that particular attribute missing. Finally, we discard those attributes whose variance is below a certain threshold. We set that threshold to 10^{-8}. After these simple preprocessing steps, we impute the remaining attributes' missing values and use the resulting data for model fitting.

5. EXPERIMENTAL RESULTS AND DISCUSSION

The proposed framework has been implemented and deployed in Turn, a leading DSP in the online advertising industry. In this section, we present several results demonstrating the performance of our algorithms on 5 different campaigns, Camp1 to Camp5, that are on flight in our platform as of this writing. All of these campaigns serve large number of daily impressions (in millions), which makes model training more robust, and also each belongs a different advertising category, e.g., autos and gifts. In our experiments, we have used impression log data from January of 2012. We split the data into two parts according to the impression date and use the first week's observations for campaign specific model training, and the second week's observations for model testing.

We denote the imbalance ratio (IR) as the number of impressions without conversion events divided by the number of impressions with conversion events. The imbalance ratio among these campaigns ranges through multiple 100s, which shows that the actual (i.e., full) datasets are heavily imbalanced. As discussed in § 4.1, we try to retain all impressions resulting in conversions to ensure sufficient representation during model training, but only a subset of the impressions without conversions due to memory constraints. After splitting the conversion impressions among the train and test datasets, we subsampled the no-conversion impressions for each campaign to keep the imbalance ratio around $2 - 4$.

5.1 Data Imbalance and Score Calibration

In this part, we explore the influence of dataset imbalance on the model performance for each campaign. As we have already pointed out, we use the area under the ROC curve (AUC) as a performance metric. For each campaign we subsampled the training data before modeling to simulate the balanced, moderately skewed, and highly skewed data scenarios by choosing $IR = \{1, 3, 10\}$, respectively. Performance results on the test datasets are shown in Table 1.
We can see from these results that the data imbalance does not have any significant influence on model performance. Note that even with an IR of 10 there was sufficient amount of examples from the conversion class in the training data.

IR	Camp1	Camp2	Camp3	Camp4	Camp5
1	0.744	0.865	0.738	0.66	0.84
3	0.743	0.881	0.741	0.661	0.837
10	0.740	0.885	0.745	0.646	0.833

Table 1: Test AUCs for different imbalance ratios.

Even though there was no significant performance change due to data imbalance, the scale of the logistic regression output scores are quite different for different imbalance scenarios, as illustrated in Fig. 5.1. This plot shows the histogram of the logistic regression output scores for campaign 5 on the left hand side, for different values of IR (other campaign score distributions have similar patterns and are omitted). Since our goal is to accurately estimate the CVR for an impression, we need to calibrate the logistic regression output to obtain scores that represent the actual scale of the event probabilities. After the calibration (see § 4.2 for details), the scores of different data skew scenarios are much closer to each other and their scale coincides with the actual campaign conversion rates (the RHS of Fig. 5.1).

5.2 Missing Value Imputation

Missing data is another very common practical problem we encounter in our system and in this subsection we test the performance of the two imputation methods: median imputation and Bayesian least squares estimation-based (BLSE) imputation. For this experiment, we have imputed the missing attribute values (i.e., the missing base estimator values) and train logistic regression models for each campaign using the imputed data. The performance results on the test data are shown in Table 2.

Imp. Method	Camp1	Camp2	Camp3	Camp4	Camp5
median	0.741	0.876	0.742	0.663	0.836
BLSE	0.741	0.875	0.746	0.662	0.834

Table 2: Test AUCs for the two imputation methods.

These results indicate that there is no significant difference in the model's predictive performance among the two imputation method. Due to its simplicity and computational advantages, we use median imputation before model training in our platform.

5.3 Baseline Estimators vs Logistic Regression

Baseline estimators using past performance observations from distinct hierarchical levels have different CVR estimation performance. Here, we compare the CVR estimation performance of logistic regression to that of two different baseline estimators to demonstrate the performance lift we can achieve by using information from finer hierarchical levels and also the lift we observe by optimally combining these baseline estimators. The first baseline estimator uses a generic user cluster and campaign level count data to estimate the performance. That is, it estimates the CVR, p_{ijk}, of the user u_i (when served with the ad a_k on page p_j) as:

$$p_{ijk} \approx \hat{\hat{p}}^1 = p^{MLE}(Y = 1 | u_i \in \mathcal{C}^G_{u_i}, a_k \in Campaign_{a_k}) \tag{19}$$

where $\mathcal{C}^G_{u_i}$ represents the group of users similar to user u_i according to some generic (i.e., not campaign specific) features such as demographics, past browsing patterns, etc. The second baseline estimator we compare against utilizes past performance data in a finer user cluster, which was generated using campaign specific clustering/targeting criteria. Also, rather than using observations on all the ads in the campaign, it uses only observations for the ad a_k. This estimator can be written as:

$$p_{ijk} \approx \hat{\hat{p}}^2 = p^{MLE}(Y = 1 | u_i \in \mathcal{C}^A_{u_i}, a_k) \tag{20}$$

where $\mathcal{C}^A_{u_i}$ represents the users similar to u_i according to campaign specific features. Performance of all these estimators for each campaign on the test data is given in Table 3. The first thing to note in these results is the

Est.	Camp1	Camp2	Camp3	Camp4	Camp5
$\hat{\hat{p}}^1$	0.583	0.618	0.578	0.522	0.714
$\hat{\hat{p}}^2$	0.710	0.851	0.639	0.653	0.799
LR	0.741	0.876	0.742	0.663	0.836

Table 3: AUC of different conversion rate estimators.

relative performance increase by using $\hat{\hat{p}}^2$ over $\hat{\hat{p}}^1$, which uses ad (rather than campaign) level and campaign specific user cluster (rather than generic user cluster) level observations. This is somewhat expected since $\hat{\hat{p}}^2$ estimates the CVR among users which are more homogeneous from a campaign goal perspective. Logistic regression, which combines several baseline estimators, performs even better than $\hat{\hat{p}}^2$. The average performance lift achieved by logistic regression over the two baseline estimators are %28.2 and %5.92 respectively. Hence, using as many quality baseline estimators as possible and combining their outcomes optimally using logistic regression will yield a more accurate CVR estimate.

6. CONCLUSIONS

We have presented a flexible and principled approach to estimate conversion rates for serving display advertisements in real-time. We have shown that our algorithm provides consistent improvement in conversion rate estimation over some baseline estimators. The presented approach also provides simple but effective recipes for handling practical issues persistent in a real DSP, such as missing data and dataset imbalance. The offline training for the proposed approach over a large number of campaigns runs fast thanks to parallelization on a distributed computing cluster using R, Pig, and Java on Hadoop.

As future work, we also would like to incorporate more user and publisher information obtained from third party media providers into data hierarchies to improve model prediction.

Acknowledgments

We would like to thank Goutham Kurra, Santanu Kolay, and Andrey Svirsky for many insightful discussions and for their feedback.

7. REFERENCES

[1] D. Agarwal, R. Agrawal, and R. Khanna. Estimating rates of rare events with multiple hierarchies through scalable log-linear models. *ACM SIGKDD Conf. on Knowledge Discovery and Data Mining*, 2010.

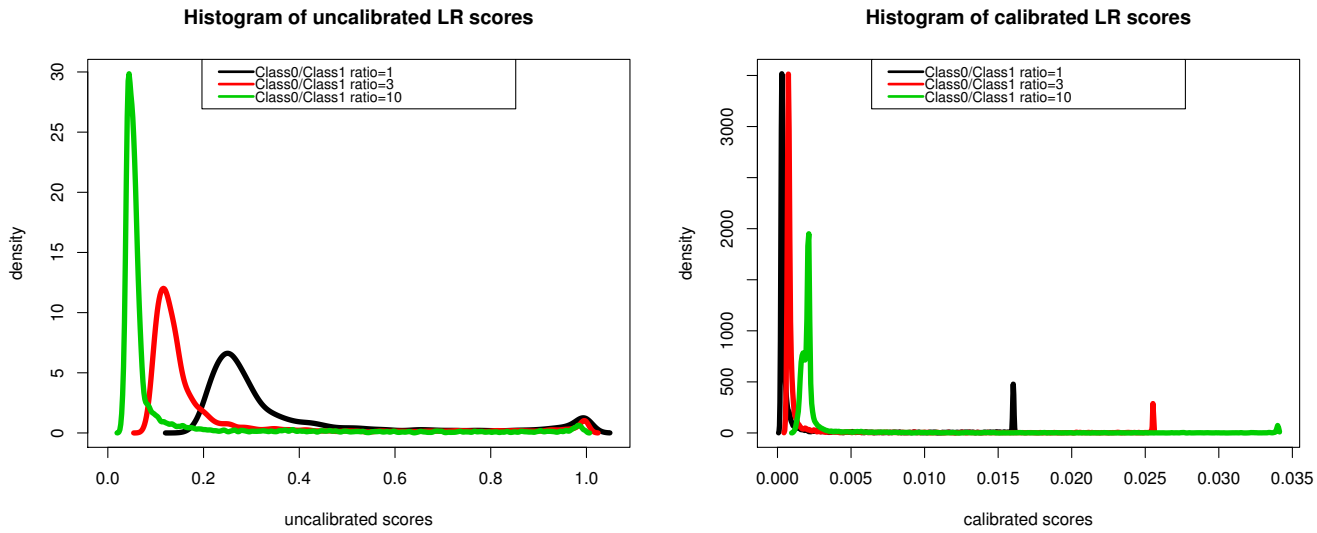

Figure 3: Score calibration.

[2] D. Agarwal, A. Broder, D. Chakrabarti, D. Diklic, V. Josifovski, and M. Sayyadian. Estimating rates of rare events at multiple resolutions. *ACM SIGKDD Conf. on Knowledge Discovery and Data Mining*, 2007.

[3] A. Ahmed, Y. Low, M. Aly, V. Josifovski, and A. J. Smola. Scalable distributed inference of dynamic user interests for behavioral targeting. *ACM SIGKDD Conf. on Knowledge Discovery and Data Mining*, 2011.

[4] E. Bax, A. Kuratti, P. Mcafee, and J. Romero. Comparing predicted prices in auctions for online advertising. *Int. J. of Industrial Organization*, 30:80–88, 2012.

[5] D. Blei, A. Ng, and M. Jordan. Latent dirichlet allocation. *J. of Machine Learning Research*, 3:993–1022, 2003.

[6] J.-F. Cai, E. J. Candes, and Z. Shen. A singular value thresholding algorithm for matrix completion. *SIAM J. on Optimization*, 20:1956–1982, 2008.

[7] D. Cerrato, R. Jones, and A. Gupta. Classification of proxy labeled examples for marketing segment generation. *ACM SIGKDD Conf. on Knowledge Discovery and Data Mining*, 2011.

[8] Y. Chen, P. Berkhin, B. Anderson, and N. R. Devanur. Real-time bidding algorithms for performance-based display ad allocation. *ACM SIGKDD Conf. on Knowledge Discovery and Data Mining*, 2011.

[9] J. de Leeuw, K. Hornik, and P. Mair. Isotone optimization in r: Pool-adjacent-violators algorithm (pava) and active set methods. *J. of Statistical Software*, 32(5):1–24, 2009.

[10] T. Graepel, J. Q. Candela, T. Borchert, and R. Herbrich. Web-scale bayesian click-through rate prediction for sponsored search advertising in microsoft's bing search engine. *International Conf. on Machine Learning*, 2010.

[11] R. J. A. Little and D. B. Rubin. *Statistical Analysis with Missing Data*. John Wiley and Sons, 1987.

[12] M. A. Maloof. Learning when data sets are imbalanced and when costs are unequal and unknown. *ICML Workshop on Learning from Imbalanced Datasets II*, 2003.

[13] B. M. Marlin and R. S. Zemel. Collaborative prediction and ranking with non-random missing data. *3rd ACM Conf. on Recommender Systems*, 2009.

[14] A. Menon, K. Chitrapura, S. Garg, D. Agarwal, and N. Kota. Response prediction using collaborative filtering with hierarchies and side-information. *ACM SIGKDD Conf. on Knowledge Discovery and Data Mining*, 2011.

[15] M. Richardson, E. Dominowska, and R. Ragno. Predicting clicks: estimating the click-through rate for new ads. *WWW*, pages 521–530, 2007.

[16] S. Visa and A. Ralescu. Issues in mining imbalanced data sets - a review paper. *Proc. of the 16th Midwest AI and Cognitive Science Conf.*, pages 67–73, 2005.

[17] G. M. Weiss. Mining with rarity: A unifying framework. *ACM SIGKDD Explorations*, 6:7–19, 2004.

[18] D. Williams, X. Liao, Y. Xue, L. Carin, and B. Krishnapuram. On classification with incomplete data. *IEEE Trans. On Pattern Analysis And Machine Intelligence*, 29, 2007.

[19] L. Zhang and D. Agarwal. Fast computation of posterior mode in multi-level hierarchical models. *Neural Information Processing Systems Foundation*, 2008.