Targeting Converters for New Campaigns Through Factor Models

Deepak Agarwal
Yahoo! Research
dagarwal@yahoo-inc.com

Sandeep Pandey
Yahoo! Research
spandey@yahoo-inc.com

Vanja Josifovski
Yahoo! Research
vanjaj@yahoo-inc.com

ABSTRACT

In performance based display advertising, campaign effectiveness is often measured in terms of *conversions* that represent some desired user actions like purchases and product information requests on advertisers' website. Hence, identifying and targeting potential converters is of vital importance to boost campaign performance. This is often accomplished by marketers who define the user base of campaigns based on behavioral, demographic, search, social, purchase, and other characteristics. Such a process is manual and subjective, it often fails to utilize the full potential of targeting. In this paper we show that by using past converted users of campaigns and campaign meta-data (e.g., ad creatives, landing pages), we can combine disparate user information in a principled way to effectively and automatically target converters for new/existing campaigns. At the heart of our approach is a factor model that estimates the affinity of each user *feature* to a campaign using historical conversion data. In fact, our approach allows building a conversion model for a brand new campaign through campaign meta-data alone, and hence targets potential converters even before the campaign is run. Through extensive experiments, we show the superiority of our factor model approach relative to several other baselines. Moreover, we show that the performance of our approach at the beginning of a campaign's life is typically better than the other models even when they are trained using all conversion data after the campaign has completed. This clearly shows the importance and value of using historical campaign data in constructing an effective audience selection strategy for display advertising.

Categories and Subject Descriptors

H.4.m [**Information Systems**]: Miscellaneous

General Terms

Algorithms, Performance, Experimentation

Keywords

conversions, factor, targeting

1. INTRODUCTION

Users engage in online activities like search, consuming content on websites, interacting with friends and colleagues on social media, buying products online, and so on. This provides opportunities for businesses to advertise their products online and attract user attention in different contexts, creating a lucrative and rapidly growing online advertising industry. Variety of ads are served to the users such as those including text, logo, pictures, rich media (a combination of text, audio and video), and interactive content for active user participation. Such online advertising is heavily focused on *targeting* users that are of "high value," and is often tracked and measured in terms of *conversions* that represent some desired user actions like purchases, form fillings, product information requests. Targeting converters ensures good advertiser ROI and publisher revenue, however this involves pinpointing appropriate users amid the chaos and huge volumes of online user activity data.

Matching the "right" users to campaigns is a complex process that is critical to the performance of an advertising campaign. At a fundamental level it requires effective inference of user interests and campaign requirements. Not surprisingly, this has attracted wide attention from the research community and sophisticated models have been built for addressing this problem [6, 8, 23]. These models work by learning from the past users targeted for a campaign, to identify potential future converters. However, they do not apply for new campaigns for which no prior targeting information is available. One can imagine dealing with this by launching a new campaign on random users and waiting until enough conversions are obtained for the above techniques to kick in. However, this delay can result in significant monetary loss to the advertiser and the publisher for several reasons. First, conversions are very rare events. Very few users click on the ad and even fewer convert. Second, learning models to target users is known to be a challenging and high-dimensional problem that requires large number of conversions for reliable estimation. Third, campaigns have short lifetimes and are monitored closely. Below par performance at the beginning can force the advertiser to stop the campaign altogether.

In practice, optimization for new campaigns typically involves marketers working with advertisers to understand the goals and nature of their ad campaigns, and matching them to users based on information like demographics, behavioral, social, geographical, and others. Often this is a trial and error process whereby marketers target specific user dimensions. Although statistical and visualization tools adept at summarizing user information in an effective fashion are at the disposal of marketers, the decision of what to target is mostly subjective and depends on the expertise of the marketers involved. Hypotheses are tested by running trial

WWW 2012, April 16–20, 2012, Lyon, France.
ACM 978-1-4503-1229-5/12/04.

campaigns, new hypotheses are generated, tested, and the process repeats. Such an unsupervised process not aimed at optimizing a readily measurable proxy may fail to extract optimal value for both advertisers and publishers.

In this paper we provide methods that learn user affinity to campaigns from combining (a) user profile data based on behavior, demographics, etc., (b) campaign meta-data in terms of associated ad creatives, landing pages, and (c) the user-campaign targeting interactions collected on historical campaigns. We show such a combination is potent and can significantly enhance targeting performance. Our method allows us to build targeting model for a new campaign solely using the campaign meta-data, even before running the campaign. Moreover, as the campaign starts running and additional conversion data is procured, we are able to adapt the initial model over time.

Our approach can have significant implications for display advertising. For instance, recent advances in technology (such as real-time bidder, ad exchanges like DoubleClick/RightMedia [2]) have revolutionized display advertising and given rise to a complex ecosystem of intermediaries like demand side platforms, ad-networks, and others. Several entities in this complex advertising ecosystem have access to both user behavioral and past campaign performance data. Hence, any of these entities can potentially use our approach to enhance targeting for new/existing campaigns. For instance, a publisher like Yahoo! can combine user conversion information on campaigns obtained through RightMedia with user behavioral data based on online activities like visits to various Yahoo! pages, web searches and vertical searches. Similarly, an ad-network can buy user behavioral data through data exchanges like BlueKai [1] and combine it with partial transactional data available through real-time bidder and advertiser-side conversion logs.

Our Approach. Campaign performance can be measured either through ad clicks or conversions. While clicks serve as a proxy for user's interest in the advertised product, they can often be misleading, e.g., click fraud, bounce clicks [17]. Hence, in this paper we use conversions as our performance metric as it is closer to the advertisers' real goals.

Based on data from historical campaigns that were run for targeting conversions, we extract information and build models to predict conversion propensity as a function of user profile and campaign meta-data information. The modeling strategy is such that given a user profile and a new campaign with meta-data, we are able to predict conversion propensity even before running the campaign, as shown in Figure 1(a). The campaign metadata helps in understanding what the advertising campaign is about and thus identifying the potential targeting set. As we run the campaign and collect additional transactional data, our conversion model adapts and becomes more accurate in predicting conversion propensity. This supervised targeting approach that combines user profile data to optimally predict conversion rates is markedly different from the usual unsupervised targeting approach practiced by marketers.

Although the supervised approach looks promising at first blush, it is extremely challenging for several reasons such as low conversion rates, high dimensional user profiles, user cookie churn, variability in user interests and other temporal effects. In fact, even the conversion definition across campaigns is different and depends on the advertiser and campaign type. We deal with these challenges and make the following contributions in this paper.

Contributions. We provide a novel modeling solution called FACTOR that pools data across campaigns to mitigate the effect of sparsity. In fact, FACTOR is able to predict user-affinity to a new campaign by only using the campaign meta-data to begin with; the predictions improve as the campaign runs and obtains more conversion information. The main idea is to tie the campaign specific user feature coefficients in multiple logistic regression classifiers (one per campaign) through a factor model. Our model not only learns separate factors for each user feature and campaign, it also simultaneously learns a function that predicts campaign factors based on campaign meta-data. This allows generalization to brand new campaigns. We run experiments on more than 100 real-world ad campaigns and report impressive gains relative to other strong baselines. Our approach is amenable to distributed computing in a map-reduce paradigm and scales gracefully to large advertising applications.

2. PROBLEM SETUP AND CHALLENGES

Our problem setup is as follows. We are given the targeting data for historical campaigns that were run in the past, we call them the *train* campaigns. For each train campaign, we have a list of users who were targeted and the ones who converted (i.e., user-campaign interaction data). Also, we are given a representation of the users and the campaigns whereby (see Figure 1(a)):

- **User:** A user is represented in terms of her past online activities. These activities are tracked by the advertisers, publishers and third parties through browser cookies that uniquely identify the user. This includes the history of page visits, ad views, and search queries. Based on the content of these events, a user profile is composed in terms of a feature vector representing a user for modeling purposes (more details in Section 5).
- **Campaign:** A campaign is characterized in terms of the meta-data associated with it such as *ad creatives* and *landing pages*. An ad creative is an image or text snippet that is displayed to the user. Upon a click on the ad, the user is taken to a web page associated with this creative, called the landing page. We construct a campaign feature vector using the creative and landing page content. The metadata helps in understanding what the advertising campaign is about and thus identifying the potential targeting set.

Given this training data (user-campaign interactions, user profiles and the campaign meta-data), our goal is to learn a model that can be used to estimate conversion propensity of a user for a brand new campaign (called *test* campaign) by merely using the campaign meta-data. As the new campaign is run and additional conversion data flows in, the initial model should adapt and get better (see Figure 1(a)). Before proceeding to our learning strategy, we first describe the technical challenges that we face.

2.1 Challenges

Learning models for targeting converters is challenging for several reasons. First, the number of conversions for each campaign is small relative to the number of user features (**rarity**). We have hundreds of thousands of user features per campaign but the number of conversions range from few hundreds to few thousands. (Our dataset is described in details in Section 5.) Also, as shown in Figure 1(b), a large fraction of user features in a campaign occurs only a few times, making the data sparser. Hence it is difficult to use the campaign-specific data for identifying converters for a new campaign. Pooling data across campaigns in a proper way is an attractive approach to mitigate such sparsity. Such data pooling is technically challenging though.

Multi-tasking has been widely used in the literature for pooling data across multiple learning tasks. However, most of the existing work assumes the same user features occur across different campaigns, while this is not true in our case as seen in Figure 1(c) where we plot the distribution of user feature occurrences across campaigns (**sparsity**). Around 15% of features occur in a few campaigns followed by a long tail distribution. This presents additional challenges which we shall address. Another important aspect of our problem is that when a new campaign arrives, we want to be able to build its targeting model using its meta-data. Moreover, we want to ensure that with the availability of additional conversion data procured once the campaign starts running, we are able to adapt the initial model over time.

While pooling data across campaigns, we must realize that campaigns come from different advertisers and have different characteristics (**heterogeneity**). They span a diverse set of domains like travel, sports goods, movie rentals, online shopping, etc. The number of conversions across campaigns also show wide variation. In fact, the definition of conversion itself is different across campaigns. In some campaigns for instance, filling a form may be a conversion, while for some others the user may have to subscribe to a service, and so on. Hence, any approach that pools data across campaigns has to be learned in an unbiased fashion after adjusting for such heterogeneity. For instance, campaigns with large number of conversions should not exert too much influence on the modeling framework, in such cases the modeling technique will not work well on new campaigns that are of different kind. Such unbiased estimation has to be conducted carefully.

3. OUR APPROACH

Our approach works by pooling data across campaigns using the three sources of information: user-campaign interactions, user profiles and the campaign metadata, as shown in Figure 1(a). First, we describe how, for a single campaign, we use the past user-campaign interactions and user profiles to identify its potential future converters. We show that this works reasonably well (in our experiments) but suffers from rarity of conversions and sparsity of features. Then, in Section 3.2 we describe how we generalize this by pooling data across campaigns using the campaign metadata and Hierarchical Bayesian framework.

Notations:. We introduce some notations first. Let $\boldsymbol{z}_j$ denote the feature vector for campaign j. We note that campaign feature vector is the only information available for a new campaign. Let $\mathcal{U}_j$ denote the set of users that are exposed to campaign j. Let $((\mathcal{I}_{ij}))$ be a feature incidence matrix where rows form the user feature and columns are the campaign features, a 1 in the $(i,j)^{th}$ cell indicates the i^{th} user feature is present in campaign j, while 0 indicates feature absence. We denote by $\boldsymbol{X}_{k,j}$ the feature vector associated with an arbitrary user $k \in \mathcal{U}_j$, the feature ids occurring in $\boldsymbol{X}_{k,j}$s obviously correspond to the 1s in the j^{th} column of the feature incidence matrix $\mathcal{I}$.

3.1 Learning from a Single Campaign

We begin by describing our conversion modeling for a single campaign j. This can be framed as a binary classification task where each user profile makes an instance. The converted users are the positive instances, while the rest make the negatives. A widely used classifier for such binary classification tasks is logistic regression [20]. Indeed, comparison to other classifiers like SVM and Naive Bayes on our data showed that SVM had similar performance, while Naive Bayes performed poorly. Classifiers like decision trees and random forests are not suited to our task due to sparse occurrence frequencies of features in campaigns as shown in Figure 1(b). Hence we use logistic regression as our base model to develop data-pooling solutions across campaigns.

Mathematically speaking, let y_{kj} denote the binary conversion indicator for user $k \in \mathcal{U}_j$ on campaign j. Thus, $y_{kj} = 1$ implies user k converted on campaign j while $y_{kj} = 0$ means she did not. As mentioned earlier, $\boldsymbol{X}_{k,j}$ denotes the corresponding user feature vector. We will sometimes use $\boldsymbol{Y}_j$ to denote the entire response vector for the j^{th} campaign. The logistic regression model for campaign j in the training data is given as follows.

$$y_{kj} \sim \text{Bernoulli}(p_{kj}); k \in \mathcal{U}_j$$
$$\log \tfrac{p_{kj}}{1-p_{kj}} = \boldsymbol{X}^{'}_{k,j}\boldsymbol{\beta}_j$$

The unknown coefficient vector $\boldsymbol{\beta}_j$ is estimated by using a maximum likelihood (MLE) approach using the data from past users exposed to j, i.e., for users in $\mathcal{U}_j$. Let N_j denote the total number of users exposed to campaign j, i.e., $N_j = |\mathcal{U}_j|$, M_j denote the total number of unique features, i.e., $M_j = \text{length}(X_{k,j})$, and n_j denote the total number of converters, i.e., $n_j = \sum_k y_{kj}$. The log-likelihood ℓ_j under a Bernoulli model is

$$\ell_j = \sum_k (y_{kj} \log \frac{p_{kj}}{1-p_{kj}} + \log(1-p_{kj})),$$

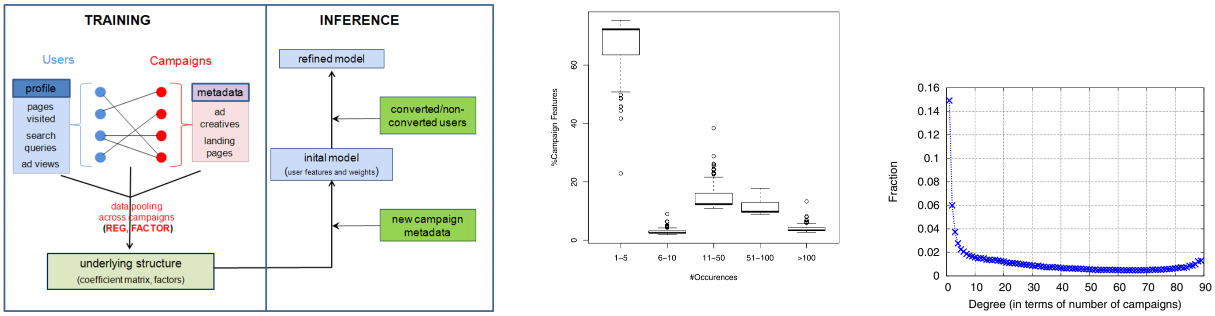
and plugging-in the expression of p_{kj} as a function of unknown $\boldsymbol{\beta}_j$, we get

$$\ell_j = \sum_k (y_{kj}\boldsymbol{X}^{'}_{k,j}\boldsymbol{\beta}_j - log(1 + exp(\boldsymbol{X}^{'}_{k,j}\boldsymbol{\beta}_j))).$$

The unknown $\boldsymbol{\beta}_j$ is obtained by maximizing the function ℓ_j, this problem has a rich literature dating back to the 70s and is referred to as logistic regression [18].

In our application, it is routine to observe small number of converters but large number of features, i.e., $n_j \ll M_j$. For instance, in our data n_j's range from a few hundreds to few thousands, while M_j's are in hundreds of thousands. With such extreme imbalance in the number of positives relative to the dimension of unknown coefficient vector $\boldsymbol{\beta}_j$, the estimates are unstable and have high variance.

To get a sense of the difficulty involved, consider one binary user feature. The MLE then has a closed form solution given by the log-odds ratio between the binary variables y_{kj} and the only binary feature X_{kj}. It is clear that the estimate diverges to the boundaries $\{-\infty, +\infty\}$ when either y's and X's do not co-occur together or the X's occur only when y's



(a) Approach overview

(b) Per-campaign feature distribution

(c) Across campaigns feature distribution

Figure 1: (a) Flow of our approach. (b) Feature occurrence distribution within a campaign (for 89 campaigns). (c) Distribution of feature occurrence across 89 campaigns.

occur. In other words, we need feature *overlap* with both label types, i.e., X's have to co-occur with both kinds of labels. More precisely, if the *cone* spanned by the columns of feature rows corresponding to converters have no overlap with the corresponding cone for non-converters, MLE for logistic regression does not exist! For a complete technical characterization of this issue, we refer the reader to [14].

To ensure better behavior in high dimensional setting such as ours, constraints (regularization) on the unknown coefficients $\boldsymbol{\beta}_j$ or borrowing information across campaigns (data pooling) is important. Proper ways to borrow information across campaigns is the crux of our problem, as described next.

3.2 Pooling Data Across Campaigns

We now discuss various data pooling strategies. Recall that our goal is not to merely fit logistic regression reliably to training campaigns, we want to estimate the conversion propensity of users to a new campaign using the campaign specific meta-data, before even running the campaign. Such a process can simultaneously utilize all rich sources of data and provide more fine grained user targeting compared to the human supervised procedures whereby marketers design targeting strategy through broad segment behavior.

We perform data pooling by working in a hierarchical Bayesian modeling framework. Appealing again to the feature incidence matrix $\mathcal{I}$, let β_{ij} denote the coefficient associated with feature id i in campaign j corresponding to pairs (i, j) for which $\mathcal{I}_{ij} = 1$. Our Bayesian framework assumes independent Gaussian priors on β_{ij}s, i.e.,

$$\beta_{ij} \sim N(\beta_{ij0}, \sigma_{ij}^2) \qquad (1)$$

We combine the prior in Equation 1 with the logistic log-likelihood $\sum_j \ell_j$ to obtain the posterior distribution of βs. The posterior mean provides estimated values of the coefficients β_{ij}s. Note that for a new campaign j^*, we use the prior mean β_{ij^*0} as the initial targeting model, since no conversion data is available on campaign j^* to begin with. Hence, modeling the prior means β_{ij0}s properly is crucial for an effective generalization to new campaigns.

Since σ_{ij}^2s are the prior variance parameters and generally harder to estimate than the mean, we choose two parameterizations: (a) *Per-campaign variance* component model, i.e., $\sigma_{ij}^2 = \sigma_j^2$ for all i, and (b) *global variance* component model, i.e., $\sigma_{ij}^2 = \sigma^2$ for all (i, j). To give some intuition on the scale of σ^2s, note that the interval $(\beta_{ij0} - 3\sigma_{ij}, \beta_{ij0} + 3\sigma_{ij})$ represents the range within which we restrict the magnitude of β_{ij} with high probability a-priori. With hundreds of thousands of features and small number of conversions, it is important to keep this range to be small. Based on extensive experiments, we found restricting σ^2s in the range $[10^{-3}, 10^{-1}]$ works well in our problem settings. Larger values are too flexible and lead to high posterior variance and unreliable posterior mean estimates, while smaller values tend to restrict posterior means of β_{ij}s too close to β_{ij0}s and restricts the scope of learning from the available user-campaign interaction data.

3.2.1 *Modeling Prior Mean*

As evident, the main component of our data pooling approach in a hierarchical Bayesian framework is in estimating the prior distribution; most importantly the prior means β_{ij0}s. We discuss three possibilities.

ZEROMEAN. This assumes $\beta_{ij0} = 0$ and is commonly used to impose regularization when fitting a single ill-conditioned logistic regression. The prior variance restricts the range of coefficients and the posterior mean does not diverge even when feature cones have low overlap. As evident, this does not generalize to brand new campaigns. However, once a campaign is launched and it obtains some conversions, we can use the zero prior to combine with log-likelihood and obtain posterior mean for coefficients. This is commonly used in the current conversion-based advertising systems [6] and serves as the baseline in our framework (we shall refer to this as ZEROMEAN).

Linear Regression Prior. A natural approach is to model the prior mean as a function of campaign features $\boldsymbol{z}_j$. In other words, we assume $\beta_{ij0} = g_i(\boldsymbol{z}_j)$, where g_i is an unknown function for user feature i. Although one can use different kinds of non-linear functions, we confine ourselves to a linear function $\boldsymbol{g}_i^{'}\boldsymbol{z}_j$, where the coefficient vectors $\boldsymbol{g}_i$s are unknown. Such a model generalize to new campaigns if we can estimate the unknown coefficients $\boldsymbol{g}_i$ for each user feature i from the training data. To avoid over-fitting, we constrain the $\boldsymbol{g}_i$s by imposing an L_2 penalty term. We perform such an estimation through an EM algorithm as described in section 4. For the sake of easy reference, we shall refer to this model as REG; the parameter vectors $\boldsymbol{g}_i$ across different features will be denoted by a matrix $\boldsymbol{G}$ whose i^{th} row is $\boldsymbol{g}_i$.

Factor Model. Training a global function $\boldsymbol{g}_i^{'}\boldsymbol{z}_j$ for each user feature in REG involves estimating a large number of parameters $\boldsymbol{G}$. For instance, if there are 100K user features and 30 campaign features, we have to estimate 3M parameters to predict the prior mean β_{ij0}s! This could be daunting and may require a large number of historical campaigns that may not be always available.

To reduce the number of unknown parameters, we take recourse to a factor model that is used in collaborative filtering applications to model user affinity to items. However, in our scenario, we do not apply it to the original user-campaign interaction data, instead we use it to model the prior mean of a coefficient matrix of user features for different campaigns. The factor model assumes

$$\beta_{ij0} = \boldsymbol{u}_i^{'}\boldsymbol{v}_j$$

where $\boldsymbol{u}_i$ and $\boldsymbol{v}_j$ are r-dimensional unknown latent factors for user feature i and campaign j respectively.

We note that if the coefficient matrix was complete, this is equivalent to assuming a probabilistic PCA [21] prior on the coefficients β_{ij}s. But since the matrix is not complete, such an interpretation is not precise, thinking as a matrix completion problem is more germane. Two issues remain. To estimate the factors reliably when the matrix is incomplete, some constraints need to be imposed on the factors. Also the campaign factors $\boldsymbol{v}_j$ should be computable for a new campaign. We address both issues by putting appropriate priors on the feature and campaign factors. More precisely,

$$\begin{aligned} \boldsymbol{u}_i &\sim MVN(\boldsymbol{0}, a_u) \\ \boldsymbol{v}_j &\sim MVN(\boldsymbol{D}\boldsymbol{z}_j, a_v) \end{aligned} \tag{2}$$

where $\boldsymbol{D}$ is an unknown matrix with r rows, each row corresponds to coefficients of a linear regression that predicts the corresponding campaign factor; scalars a_u and a_v are variance components that determine the range within which we should constrain the factors. All unknown quantities in Equation 2 are estimated from data as we discuss in Section 4. For easy reference, we shall refer to this model as FACTOR. We note that the total number of unknown parameters to estimate prior means β_{ij0} is now r times the total number of campaign features. If $r = 5$ and we have 30 campaign features, this reduces to estimating 150 parameters as opposed to 3M with REG.

Some Remarks on FACTOR**.** It is worthwhile to summarize some important facts about FACTOR.

- *Fewer Unknown Parameters*: We assume the coefficient matrix can be well approximated a-priori through a low-rank matrix decomposition. A linear regression is then used to estimate the campaign factors as opposed to predicting each user feature coefficient. This is key in reducing the number of unknown parameters required to estimate the prior mean.

- *Dealing with Campaign Size Variation*: Assuming β_{ij}s to be centered around $\boldsymbol{u}_i^{'}\boldsymbol{v}_j$ with some prior variance is a crucial relaxation that allows the posterior mean of some β_{ij}s to deviate away from the estimated prior mean $\boldsymbol{u}_i^{'}\boldsymbol{v}_j$. This prevents campaigns with large number of conversions from having excessive influence on the factor estimates.

- *Building and Adapting Models for New Campaigns*: For a new campaign j^*, one uses the campaign features $\boldsymbol{z}_{j^*}$ to predict the campaign factor as $\boldsymbol{D}\boldsymbol{z}_{j^*}$. Since the factors for user features are already known from the training phase, we can compute the prior mean β_{ij^*0} efficiently as $\boldsymbol{u}_i^{'}\boldsymbol{D}\boldsymbol{z}_{j^*}$. This serves as the initial model for the campaign. Also, due to the aforementioned relaxation via prior variance, the model can be adapted over time as the number of conversions obtained on the campaign increases. Without this relaxation, adaptation would not be possible.

4. MODEL FITTING

We now describe our model fitting procedures for REG and FACTOR. For both, we use an EM algorithm [9] to estimate the unknown parameters in the prior of β_{ij}s.

Parameters for EM:. Denoting the unknown parameters in the prior distribution by $\boldsymbol{\Theta}$, our goal is to maximize the marginal log-posterior of observed data. The marginalization (integration) is performed with respect to latent variables (unobserved random variables) $\boldsymbol{\Delta}$. In the case of REG for instance, $\boldsymbol{\Delta} = \{\beta_{ij}\}_{\forall(i,j)}$, the set of all unknown coefficients corresponding to pairs (i, j) that are present in the incidence matrix $\mathcal{I}$, and $\boldsymbol{\Theta} = (\boldsymbol{G}, \{\sigma_j^2\}_{\forall j})$ where i runs over all user feature ids observed in the training data, and j runs over campaigns. For a global variance components model, $\boldsymbol{\Theta} = (\{\boldsymbol{G}, \sigma^2)$.

For FACTOR, the latent variables are $\boldsymbol{\Delta} = (\{\beta_{ij}\}, \{\boldsymbol{u}_i\}, \{\boldsymbol{v}_j\})_{\forall(i,j)}$; $\boldsymbol{\Theta} = (\{\sigma_j^2\}_{\forall j}, \boldsymbol{D}, a_u, a_v)$ and $(\sigma^2, \boldsymbol{D}, a_u, a_v)$ for the campaign-specific and global variance component models respectively. A crucial parameter here is $\boldsymbol{D}$, the regression coefficient matrix, that helps predicting factors $\boldsymbol{v}_j$ for new campaigns.

We note that the marginal distribution involves computing the integral of product of likelihood $exp(\sum_k \ell_k)$ and prior $\prod_{ij} N(\beta_{ij}; \beta_{ij0}, \sigma_{ij}^2)$ with respect to $\boldsymbol{\Delta}$; this cannot be computed in closed form, it is also difficult to compute this using numerical integration techniques due to the high dimensionality of the integral. Hence we use the EM algorithm by working with the log-posterior of complete data $(\{y_{kj}\}, \boldsymbol{\Delta})$ conditional on $\boldsymbol{\Theta}$.

4.1 EM Algorithm

Starting from some initial estimate $\boldsymbol{\Theta}_{init}$, the EM algorithm maximizes the marginal log-posterior by iterating through the expectation (E) and maximization (M) steps until the solution converges. Each sweep of the E and M steps are guaranteed not to reduce the marginal log-posterior. Let $\boldsymbol{\Theta}_{curr}$ be the current estimate of $\boldsymbol{\Theta}$ in the iterative process. In the E-step, we compute the expected log-posterior of complete data with respect to the conditional distribution of $[\boldsymbol{\Delta}|\{y_{ij}\}, \boldsymbol{\Theta}_{curr}]$. In the M-step, we maximize the expected log-posterior (from E-step) with respect to $\boldsymbol{\Theta}$ and obtain a new estimate. To ensure convergence, it is sufficient to use any *hill climbing* method in the M-step that provides a new improved value of $\boldsymbol{\Theta}$; such variants are called generalized EM (GEM) [15].

Monte Carlo E-step:. The E-step in our model cannot be computed in closed form for both REG and FACTOR since the posterior distribution $[\boldsymbol{\Delta}|\{y_{ij}\}_{\forall(i,j)}, \boldsymbol{\Theta}]$ is non-standard. However, it is possible to draw samples from this high dimensional distribution by using modern sampling methods based on Markov Chain Monte Carlo (MCMC) [22]. We use MCMC sampling procedures to approximate the expected complete log-posterior by using a Monte-Carlo mean computed from samples drawn from $[\boldsymbol{\Delta}|\{y_{ij}\}, \boldsymbol{\Theta}_{curr}]$. In the M-step, we maximize the Monte-Carlo mean for obtaining a new value $\boldsymbol{\Theta}$. This is called MCEM algorithm in the literature and have been widely used in various applications (see [7] and references therein.)

Next we provide details of sampling algorithms and M-step computations for both REG and FACTOR.

4.2 Estimating the REG MODEL

We describe the sampling procedure first.

Sampling for REG**:.** We use a Gibbs sampler [10] to sample from $[\boldsymbol{\Delta} = \{\beta_{ij}\}_{\forall(i,j)}|\{y_{kj}\}_{\forall(i,j)}, \boldsymbol{\Theta}_{curr}]$. Each sweep in a Gibbs sampler cycles through the individual co-ordinates (blocks of co-ordinates) in $\boldsymbol{\Delta}$ sequentially drawing a sample from lower dimensional full conditional distributions. This idea of taming the curse of dimensionality by sampling sequentially through lower-dimensional distribution to obtain samples from very high-dimensional distribution makes Gibbs sampling a powerful computing tool. In this case, the full conditional distributions being sampled are univariate: $[\beta_{ij}|Rest, \{y_{ij}\}, \boldsymbol{\Theta}_{curr}]$, where *Rest* denotes all co-ordinates except the one being sampled that are fixed at their latest sampled values. The process is repeated several times, the samples obtained are realizations from a Markov chain with stationary distribution $[\boldsymbol{\Delta} = \{\beta_{ij}\}_{\forall(i,j)}|\{y_{ij}\}_{\forall(i,j)}, \boldsymbol{\Theta}_{curr}]$. We discard the first few samples (called burnin), and take the rest as our samples to compute the Monte Carlo mean. Each univariate distribution is a one dimensional density that is non-standard but log-concave, it is sampled by using an adaptive rejection sampling algorithm as described in [11].

Scalability:. We note that the coefficient vectors of different campaigns are independent of each other a-posteriori, i.e., $[\boldsymbol{\Delta} = \{\beta_{ij}\}|\{y_{kj}\}, \boldsymbol{\Theta}_{curr}] = \prod_j [\Delta_j = \boldsymbol{\beta}_j|\boldsymbol{Y}_j, \boldsymbol{\Theta}_{curr}]$. This has important implications. It means sampling of latent variables can be done separately for each campaign, i.e., we can run *independent* Gibbs sampler for each campaign and decouple the sampling in the E-step across campaigns. We exploit this structure to achieve scalability by parallelizing the E-step across campaigns in a map-reduce framework, the mapper splits the data by campaigns and the reducer runs the Gibbs sampler for each campaign using adaptive rejection sampling.

M-step for REG**:.** In the M-step, we estimate $\boldsymbol{\Theta}$ by minimizing

$$E \sum_{(i,j):\mathcal{I}_{ij}=1} ((\beta_{ij} - \boldsymbol{g}_i^{'}\boldsymbol{z}_j)^2/\sigma_j^2 + log(\sigma_j^2))$$

where the expectation is with respect to the posterior of latent variables at the latest $\boldsymbol{\Theta}$ value. If $\bar{\beta}_{ij}$ and τ_{ij}^2 denote the mean and variance computed from the monte-carlo samples drawn in the E-step, this reduces to minimizing

$$\sum_{(i,j):\mathcal{I}_{ij}=1} (((\bar{\beta}_{ij} - \boldsymbol{g}_i^{'}\boldsymbol{z}_j)^2 + \tau_{ij}^2)/\sigma_j^2 + log(\sigma_j^2))$$

For a given user feature i, we estimate $\boldsymbol{g}_i$ as $\hat{\boldsymbol{g}}_i$ through a linear regression of $\bar{\beta}_{ij}$s on $\boldsymbol{z}_j$. Denoting by RSS_j the residual sum-of-squares defined as $\sum_{i:\mathcal{I}_{ij}=1}(\bar{\beta}_{ij} - \hat{\boldsymbol{g}}_i^{'}\boldsymbol{z}_j)^2$, the estimated σ_j^2 is given as

$$\hat{\sigma_j^2} = (RSS_j + \sum_{i:\mathcal{I}_{ij}=1} \tau_{ij}^2)/|i : \mathcal{I}_{ij} = 1|$$

For the global variance components model that assumes $\sigma_j^2 = \sigma^2$, the estimated σ^2 is given by

$$\hat{\sigma^2} = (\sum_j RSS_j + \sum_{(i,j):\mathcal{I}_{ij}=1} \tau_{ij}^2)/|(i,j) : \mathcal{I}_{ij} = 1|$$

All above expressions can be derived through simple calculus.

4.3 Estimating the FACTOR Model

Next we describe the exact sampling procedure for FACTOR, followed by a more scalable but approximate estimation method.

Sampling for FACTOR**.** For this model, the β_{ij}s are augmented with latent factors $\boldsymbol{u}_i$ and $\boldsymbol{v}_j$. Since the latent factors are shared across campaigns, the posterior independence that helped us decouple the sampling across campaigns is no longer valid. Each sweep of the Gibbs sampler would sample the β_{ij}s conditional on the latent variables and then sample the latent variables conditional on the sampled β_{ij}s. It is easy to show that conditional distributions of $\boldsymbol{u}_i$ given *Rest* and that of $\boldsymbol{v}_j$ given *Rest* are Gaussian [3]. In fact, the conditional Gaussian distributions for $\boldsymbol{u}_i$s are independent across user features and hence can be sampled in parallel. The same logic applies to $\boldsymbol{v}_j$s.

Each sweep of the Gibbs sampler can be parallelized as follows: sample β_{ij}s given the latent factors by sampling blocks $\boldsymbol{\beta}_j$s across campaigns in parallel, then sample $\boldsymbol{u}_i$s across user features in parallel followed by a parallel sampling of $\boldsymbol{v}_j$s.

Approximate E-step and M-step for FACTOR**:.** The parallelized sampling scheme described above is not amenable to parallelization in a map-reduce framework using commonly available software framework like Hadoop, since running a separate map-reduce job per Gibbs iteration is time consuming and inefficient. Hence we make the following approximation: we assume the latest latent factors and $\boldsymbol{D}$ are fixed parameters that are estimated as part of the M-step, i.e., as part of $\boldsymbol{\Theta}$, and we sample the β_{ij}s in a map-reduce framework as in REG. We then fit the latent factor model described below in Equation 3 to the Monte-Carlo mean $\bar{\beta}_{ij}$ computed from MCMC samples by using the algorithm described in [3]. The latent factor model we fit in the M-step is as follows.

$$\begin{array}{rcl} \bar{\beta}_{ij} & \sim & N(\boldsymbol{u}_i^{'}\boldsymbol{v}_j, \sigma_{ij}^2) \\ \boldsymbol{u}_i & \sim & MVN(\boldsymbol{0}, a_u) \\ \boldsymbol{v}_j & \sim & MVN(D\boldsymbol{z}_j, a_v) \end{array} \quad (3)$$

We note that from the model fit in Equation 3, we only use the fitted $\boldsymbol{u}_i$s, $\boldsymbol{v}_j$s, and $\boldsymbol{D}$, and regard them as estimates of the fixed parameters. The approximation is a valid generalized EM algorithm if the solution of $(\boldsymbol{u}_i, \boldsymbol{v}_j, \boldsymbol{D})$ obtained by fitting the model in Equation 3 improves the marginal log-posterior. This is indeed true if a_u and a_v are fixed and do not change over iterations since we are then performing constrained optimization with the same constraints over EM iterations. In our fitting process, we found these parameters to vary little across iterations.

We adopt this approximation that works in a map-reduce framework, since most modern advertising systems store their data on the cloud and map-reduce framework is an attractive way to perform distributed computing in such scenarios.

The σ_j^2s and σ^2 are estimated as in the M-step for REG by using estimated $\boldsymbol{u}_i^{'}\boldsymbol{v}_j$ as fitted prior mean in the RSS_j computations, all other computations are the same.

5. EXPERIMENTS

We begin by describing our dataset.

5.1 Data

We constructed a dataset consisting of 114 display advertising campaigns registered on a large US advertising network. All selected campaigns are performance-based, i.e., advertisers pay for actual conversions. The definition of conversion differs across campaigns. For each campaign we collected a sample of 30,000 users who were targeted during the four weeks period from mid March to mid April, 2011. Conversion information for the users targeted during this period was collected by looking ahead if necessary, due to the lag in obtaining data for certain kinds of conversions. Overall, we ended up with more than 3 million users to conduct our experiments.

Each campaign serves as a dataset for our scenario whereby each user targeted for the campaign is an instance. Users who converted are the positive instances, while the rest are negative examples. Our goal is to learn targeting models that can identify potential converters for a campaign, i.e., distinguish between positive and negative instances.

Next we describe how the users and campaigns are represented in our data.

User Representation. For each user we construct a profile based on her online activities preceding the time she was impressed with an ad. Any potentially personally identifiable was removed and all the data was anonymized. The user activities include past page visits, search queries, ad views and clicks. These activities have associated textual content that were used to construct the features for the user profile, e.g., the text of issued search queries, the content of pages viewed, etc. The weight for each feature was set to be binary in our experiments. Note that while predicting for a user during the evaluation, say on day t, we allow the prediction models to access user history up to day $t-1$. Hence, the prediction method is not using any future information.

Campaign Representation. For the campaign we have two sources of data: (a) past users who have been targeted for the campaign and of those who converted, (b) meta-data in the form of *ad creatives* (details below). If the campaign is brand new, then the former information is not available.

An ad creative is an image or text snippet for the ad that is displayed to the user. Upon a click on the ad, the user is taken to a web page associated with this creative, also called a *landing page*. The creative and the landing page give a succinct characterization of the ad campaign, and they can be useful to infer the domain of the campaign. For our experiments we crawled each landing page, parsed, and attributed the extracted content to its corresponding campaign. Then we constructed feature vectors for each campaign based on word unigrams from the associated content. To reduce noise when using these feature vectors in our models, we projected them into a 30-dimensional linear subspace using PCA.

Simulating past, new and existing campaigns. We randomly partition the 114 campaigns into a **training set** of 89 campaigns and a **test set** with the remaining 25 campaigns. The training set simulates the past campaigns for which the advertising network has served ads and is aware of the converters and non-converters. The test set simulates the new/existing campaigns. We use the campaigns in the training set to learn our models, and then build targeting models for the campaigns in the test set (see Figure 1(a)). To evaluate the performance on the test set, we compute the evaluation metric (described later) from 2-fold cross validation. In other words, we randomly create three folds for each campaign in the test set. All simulations are done on two folds and the evaluation metric is computed on the third fold. Sometimes, we shall refer to the two folds of a test campaign as training data for the test campaign.

To simulate a test campaign in different stages of its life, we make a sample of its training data (from two folds) available to the modeling approach for training. For example, when the sampling percentage, say P, is 0, no information about converted/non-converted users for this campaign is available while model training. This simulates a brand new campaign and our approach would build model for this campaign using its meta-data only. As the campaign gets older, some converters/non-converters for it become known to the advertising network. We simulate this by gradually increasing the value of P and allowing our approach to use converted/non-converted user information in conjunction with the meta-data.

Evaluation Metric. For evaluating the models produced for test campaigns, we use the area under the ROC curve. The AUC gives the probability with which the targeting method assigns a higher score to a random positive example than a random negative example (i.e., probability of concordance) [12]. So, a purely random method will have an area under the curve of exactly 0.5.

An alternative metric could be to measure precision/recall at a certain rank in the list. However, different campaigns may have different requirements in terms of precision and recall

Next we evaluate our approaches on this dataset. We start by describing the different methods that are compared.

5.2 Baseline and Our Approaches

The current state-of-the art for targeting converters is to learn a targeting model for each campaign using the past data for the campaign [6]. This is an instance of our proposed approach (in Section 3) when the prior mean for each feature is set to 0, i.e., ZEROMEAN. This would serve as baseline for our experiments. We compare it against the other two instantiations of our approach, REG and FACTOR. The parameters for each method are tuned using

2-fold cross validation and the performance numbers on the third fold from the best setting are reported, unless stated otherwise.

5.3 Performance Evaluation of ZEROMEAN

We start by investigating model ZEROMEAN which is the predominant approach used in current advertising systems [6, 8, 23]. It learns a targeting model for each campaign using the past campaign data. The approach has several shortcomings. First, clearly, this approach is unable to produce models for brand new campaigns which do not have any past data. Second, for campaigns with small amount of historical data either due to their scope, scale or age, ZEROMEAN struggles due to lack of positive examples.

We illustrate this by conducting experiments on the 25 test campaigns. We simulate the different stages of a campaign and compute our evaluation metric as discussed before. The performance results for ZEROMEAN are shown in Figure 2(a). We tuned the value of prior variance and found 0.001 provides the most stable and best results. On the x-axis we plot the number of positives examples available for training (on the log scale with base 10), and on the y-axis we show the AUC value for each of the 25 test campaigns. Also, we plot curves denoting for a given number of positives, the average AUC and the weighted average AUC over test campaigns (weighted by the number of conversions).

Both the unweighted and weighted average show the same trend: when the number of positive examples, P, is small, the models produced by ZEROMEAN are not good (resulting in AUC close to 0.55). In fact, some campaigns get an AUC of 0.5 (or below) which is what the RANDOM approach would do. This relates to the point we made earlier in Section 3 that learning conversion models for targeting is very challenging due to the sparsity of features and lack of positives. Until 100 or more conversions are obtained for training, the performance stays below 0.6. However, obtaining 100 conversions in a real world setting with several advertisers competing for the same user segments can take a significant amount of time (several weeks for small campaigns). Advertisers often desire fast turnaround and good ROI. Hence it is important to deal with the cold-start issue in an effective manner, as demonstrated by our proposed approach in the next section.

It is worth noting that the unweighted average is slightly higher than the weighted average in Figure 2(a). On further investigation, we found that small-scale campaigns are more targeted in terms of their desired users. Hence, they are relatively easier to model and perform better, which makes the unweighted average better than weighted average. Given that the trends exhibited by both the metrics are similar, we only report the weighted average in the subsequent experiments to avoid clutter in plots.

5.4 Using Past Campaigns to Build Models for New Campaigns

In this section we describe how our approach from Section 3 can be used to build models for campaigns which are brand new or have little historical data. Then we compare the ZEROMEAN approach against REG and FACTOR on test campaigns.

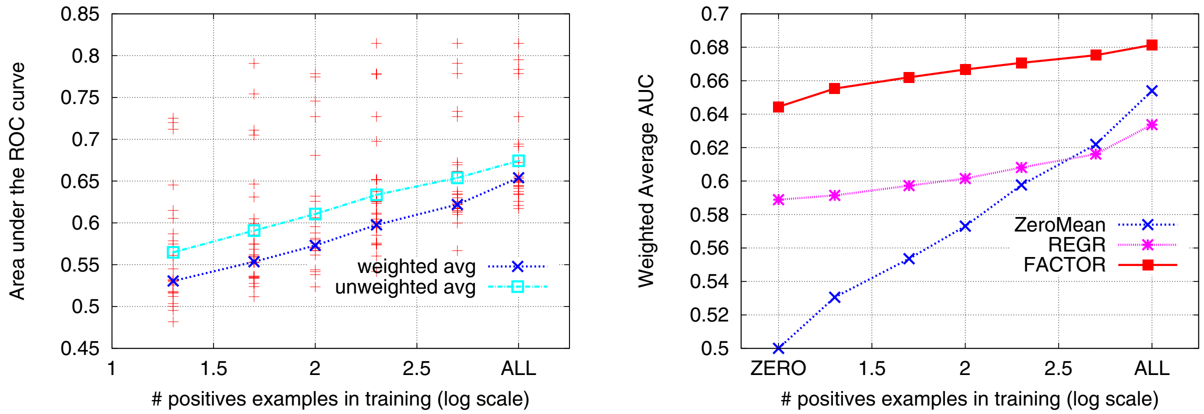
TRAIN. We first train REG and FACTOR over 89 train campaigns to estimate the $\boldsymbol{G}$ matrix used by REG and the $\boldsymbol{D}$ matrix and user factors $\{\boldsymbol{u}_i\}_{\forall i}$ used in FACTOR using the EM framework described in Section 3. We use map-reduce to scale the training procedure, as described before. After training, we perform inference to build models for the campaigns in the test set, described below.

INFERENCE. In the inference task we are given the meta-data of a campaign, say c, in the test set. Using this metadata and the estimated parameters (e.g., $\boldsymbol{G}$, $\boldsymbol{D}$), we compute the prior mean (β_{ij0}) for the user feature weights. With this prior and the available training data for campaign c, we compute the posterior model. To simulate the test campaign c at different stages of its life, we vary the number of positive examples (P) available for training data in the test phase as described before. When the campaign is brand new, P is 0, and as it gets older, the value of P is increased.

RESULTS. In Figure 2(b) we show the performance of the three methods (ZEROMEAN, REG and FACTOR). We ran FACTOR with 5 factors for this experiment (we study the effect of using different factors later in this section). On the x-axis we vary the number of positive examples and on the y-axis we denote the weighted average AUC over the 25 test campaigns. (When P is 0, the logarithm is not defined and we refer to this point as ZERO in the figures.) As expected, all approaches improve their performance as more positive examples trickle in.

Note that the average AUC for ZEROMEAN is 0.5 when $P = 0$ since the approach is unable to produce any model for a brand new campaign. More importantly, we see that REG and FACTOR provide good AUC numbers even without any training data for these new campaigns. In fact, FACTOR gives AUC of about 0.65 without any training data which is higher than the performance of ZEROMEAN with all training data. This is a great news since it means that using FACTOR we can bootstrap a new campaign with a good model and start targeting the right users at the outset. Moreover, as more users convert, we incorporate this knowledge to refine the model (through the posterior) and improve the AUC further.

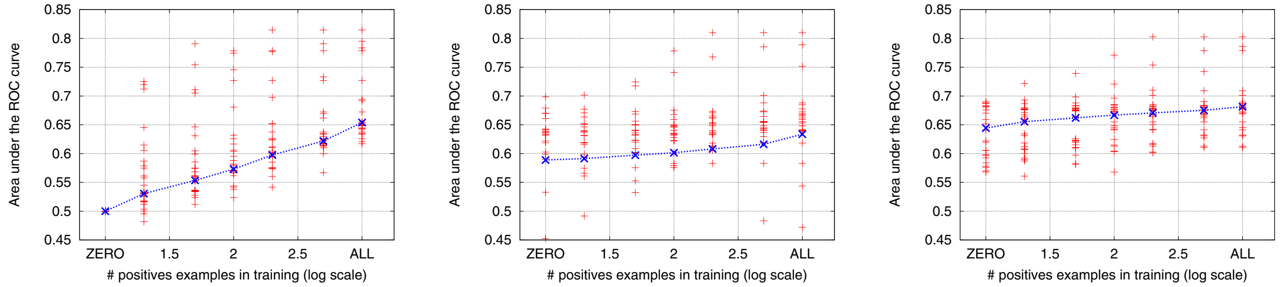
In order to investigate this further, in Figure 3 we show the performance over individual campaigns (along with the weighted average). It is worth noting that the variation in performance across campaigns is reduced using REG and FACTOR. For example, we see that ZEROMEAN suffers when there is little training data and so it has many campaigns below 0.6 AUC in the left region. On the other hand, REG performs quite well in this region by leveraging the knowledge extracted from existing campaigns. But due to high-dimensionality of $\boldsymbol{G}$ and campaign heterogeneity, REG can produce inaccurate models for some campaigns (even when all training data is made available, as shown towards the right in the figure). In theory, this problem can be avoided by increasing the prior variance and allowing the data to dominate rather than the prior mean. However, such fine tuning over individual models can be expensive and difficult to perform in a large advertising system. Our final approach, FACTOR, performs well for any amount of training data (P), it produces models which perform well uniformly across campaigns. This is encouraging since it shows that the improvement of FACTOR in average AUC is not coming at the cost of hurting some campaigns for the benefit of others, instead it is by uniformly improving the model for each campaign.



(a) Performance analysis of ZEROMEAN.

(b) Comparison of the three approaches (in terms of the weighted average AUC).

Figure 2: Performance comparison of our proposed approaches on the test campaigns for different number of positives made available for training. ZERO refers to the setting when the campaign is brand new.



(a) ZEROMEAN

(b) REG

(c) FACTOR

Figure 3: Performance variations over individual campaigns.

5.5 Effect of Parameters on FACTOR

In this section we investigate the effect of several parameters involved in training/testing of our FACTOR approach.

Number of EM Iterations. First we study the convergence behavior of the EM approach proposed for FACTOR in Section 4. In Figure 4 we show the performance of FACTOR on the test campaigns after different number of EM iterations (for $P = 0$). Each EM iteration takes about 1 hour of runtime for training over 89 campaigns (and 2 million users), with the help of the approximation and the map-reduce framework described before. As expected, the performance goes up with more iterations. However, we note that most of the improvement is achieved within the first 10-15 iterations. This is quite attractive since it implies that the model can be learned without much delay, in practice, and also updated frequently as the underlying data changes.

Effect of the number of factors. In the previous sections we had shown the results for FACTOR for 5 factors. In Figure 4 we vary the number of factors and plot the average weighted AUC over test campaigns after 5 iterations of the EM algorithm. We note that the performance of our approach is not very sensitive to the number of factors.

Effect of Initialization. We initialize the EM algorithm for FACTOR approach with zero mean and a constant value of variance, say σ^2, for each campaign. Then in each iteration of the EM algorithm we update the per-campaign variance and the global variance estimates, as described in Section 3. Figure 4 shows that FACTOR is not sensitive to the initial variance value. In other words, the approach does not need too much fine tuning and works well for a large range of settings. We also observed that results with per-campaign and global variance components are similar for both REG and FACTOR.

6. RELATED WORK

Our multi-tasking model FACTOR is related to a rich literature on multi-task learning [16] that learns multiple related tasks simultaneously to take advantage of similarities across tasks. In particular, we are related to approaches that uncover the common (latent) features that can benefit each individual task/domain [5, 13, 19].

More recently [4] proposed a method that learnt multiple logistic regressions across different articles in a news recommendation system by using a low-rank projection, i.e., $\boldsymbol{\beta}_j = U\boldsymbol{v}_j$, where the matrix U is shared across all regressions. Although this formulation is closest to our FACTOR, it has several differences. First, both this work and most others described above assume the presence of same features across all tasks. This is not true in our scenario where a feature id may only occur in a few tasks. But the most

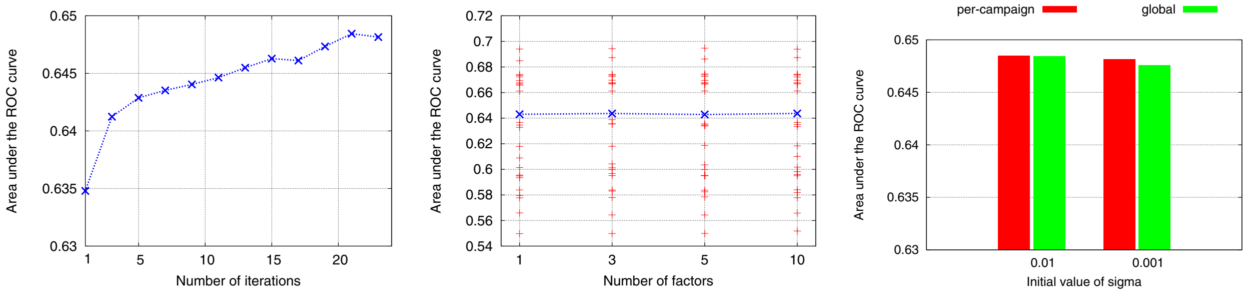


Figure 4: Effect of various parameters.

important difference is that FACTOR assumes β_{ij}s are centered around $\boldsymbol{u}_i^{'}\boldsymbol{v}_j$ and not equal to it, this relaxation is important to obtain good performance in both cold-start and warm start scenarios (campaigns with large number of conversions); the model proposed in [4] does not have this property.

As mentioned before, the factor model we use to model the prior is borrowed from the collaborative filtering literature [3] where it is used to directly model the data. Instead, we model the original data through logistic regression and push the factor model one level up to estimate the prior means of the feature weights.

7. DISCUSSION

Ad exchanges and recent technologies like real-time bidder that enable cherry-picking of impressions by ad-intermediaries have revolutionized display advertising. It provides an ecosystem where owners of rich user data can target valued user to ensure good revenue for publishers and better ROI for advertisers. The real challenge is in building statistical models that can laser in on appropriate users: we have shown combining historical campaign data with rich user profile information and campaign metadata through our new model FACTOR is a promising step in this direction. We showed promising offline results in this paper, and are in the process of "productizing" and testing the method on a live advertising system.

8. REFERENCES

[1] Bluekai. `www.bluekai.com/`.

[2] Rightmedia. `http://rightmedia.com/`.

[3] D. Agarwal and B.-C. Chen. Regression-based latent factor models. In *KDD*, KDD '09, pages 19–28, New York, NY, USA, 2009. ACM.

[4] D. Agarwal, B.-C. Chen, and P. Elango. Fast online learning through offline initialization for time-sensitive recommendation. In *KDD*, pages 703–712, 2010.

[5] A. Argyriou, C. Micchelli, M. Pontil, and Y. Ying. A spectral regularization framework for multi-task structure learning. *NIPS*, 2008.

[6] A. Bagherjeiran, A. O. Hatch, A. Ratnaparkhi, and R. Parekh. Large-scale customized models for advertisers. In *ICDM Workshops*, 2010.

[7] J. Booth and J. Hobert. Maximizing generalized linear mixed model likelihoods with an automated monte carlo EM algorithm. *J.R.Statist. Soc. B*, 1999.

[8] Y. Chen, D. Pavlov, and J. Canny. Large-scale behavioral targeting. In *KDD*, pages 209–218, 2009.

[9] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *J. of the Royal Statistical Society, Series B*, 1977.

[10] A. E. Gelfand. Gibbs sampling. *Journal of the American Statistical Association*, 95:1300–1304, 2000.

[11] W. Gilks and P.Wild. Adaptive rejection sampling for gibbs sampling. 41:337–348, 1992.

[12] M. Greiner, D. Pfeiffer, and R. D. Smith. Receiver operating characteristic (roc) curves. *Preventive Veterinary Medicine*, 45:23–41, 2000.

[13] S. Lee, V. Chatalbashev, D. Vickrey, and D. Koller. Learning a meta-level prior for feature relevance from multiple related tasks. In *ICML*, 2007.

[14] M.J.Silvapulle. On the existence of maximum likelihood estimates for the binomial response models. *Journal of the Royal Statistical Society, Series B*, 43:310–313, 1981.

[15] R. Neal and G. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In *Learning in Graphical Models*, 1998.

[16] S. J. Pan and Q. Yang. A survey on transfer learning. Technical Report HKUST-CS08-08, Hong Kong University of Science and Technology, 2008.

[17] Y. Peng, L. Zhang, M. Chang, and Y. Guan. An effective method for combating malicious scripts clickbots. In *ESORICS*, 2009.

[18] S. Press and S. Wilson. Choosing between logistic regression and discriminant analysis. *Journal of the American Statistical Association*, 73:699–705, 1978.

[19] A. P. Singh and G. J. Gordon. Relational learning via collective matrix factorization. In *KDD*, 2008.

[20] T.Hastie, R.Tibshirani, and J.Friedman. *The Elements of Statistical Learning*. Springer, 2009.

[21] M. E. Tipping and C. M. Bishop. Probabilistic principal components analysis. *Journal of the Royal Statistical Socienty, Series B*, 61:611–622, 1999.

[22] W.R.Gilks, S.Richardson, and D.J.Spiegelhalter. *Markov Chain Monte Carlo in Practice*. Chapman and Hall, 1996.

[23] J. Yan, N. Liu, G. Wang, W. Zhang, Y. Jiang, and Z. Chen. How much can behavioral targeting help online advertising? In *WWW*, pages 261–270, 2009.