Paper 6920-2016

A FAST AND SELF-LEARNING OPTIMIZATION ENGINE FOR MULTI-SOURCE ADVERTISEMENT ASSIGNMENT

Huma Zaidi and Chitta Ranjan, eBay Inc.

ABSTRACT

An ad publisher like eBay has multiple ad sources to serve ads from. Specifically, for eBay text ads, there are two ad sources, viz. eBay Commerce Network (ECN) and Google. The problem and solution we have formulated considers two ad sources. However, the solution can be extended for any number of ad sources. A study was done on performance of ECN and Google ad sources with respect to the revenue per mille (RPM, or revenue per thousand impressions) ad queries they bring while serving ads. It was found that ECN performs better for some ad queries and Google performs better for others. Thus, our problem is to optimally allocate ad traffic between the two ad sources to maximize RPM

INTRODUCTION

Internet advertising growth has been remarkable over the last decade. According to Internet Advertising Bureau (IAB) report based on survey by PricewaterhouseCoopers (PwC), the revenue from internet advertising in the United States totaled $42.8 billion for year 2013 (Coopers, 2013). The advertising revenues have been increasing every year and, importantly, there is a consistent increase in year-over-year revenue growths. The revenue grew by 15.2% from 2011 to 2012, 17% from 2012 to 2013 and Q1 of 2014 increased by more than 19% compared to Q1 of 2013. Figure 1 shows this trend (Coopers, 2013). This market is expected to increase at same pace in coming years. Tapping this market efficiently provides a potential for high gains.

There are several ad formats in internet advertising, for example, search, display, mobile, classifieds, etc. As shown in Figure 2, *search* ads have been major part of all internet advertising (Coopers, 2013).

Search ads are typically shown based on a query. Several work has been done over the years to optimize ads served based on a query, and their results have been proved to be effective (Radlinski, et al., 2008) (Cui, Bai, Gao, & Liu, 2015). However, to the best of our knowledge, very little work has been done to optimize ad serve when there are multiple ad inventory sources. For example, consider the case of advertising by eBay: eBay has an in-house ad inventory, as well as, it can draw ads from other sources, for instance, Google. In such a setup, to serve ads to a query, eBay has to decide between its in-house and other the available ad sources to draw ads from. There are several publishers who have in-house and external ad sources. Amazon and eBay are two major examples of them.

The need of using optimization techniques in such cases is quite imperative because, for a given query, one ad source can be better placed (or capable) for serving than other. For example, eBay may be better equipped for serving ads for a query for “glass chessboard” than Google, while the latter can be better placed to serve ads for a “red-blue yoga mat”. Showing ads from a more capable ad source from a pool of several ad sources increases the chance of a click and ad revenue. Moreover, if an in-house ad inventory source is competing with external ad source then for similar capability, in-house ad serve will bring more revenue. This is because a publisher is usually in an ad revenue sharing agreement with external ad sources. Using an optimization model, we can make optimal decisions on assigning ads to the right ad source to give us higher revenue and/or traffic.

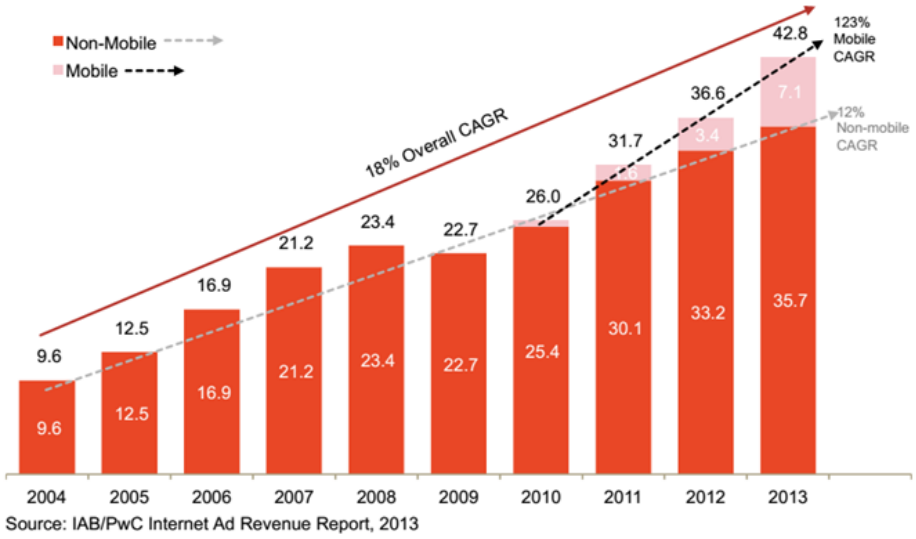

Figure 1. Year over year growth of internet advertising revenue (Coopers, 2013)

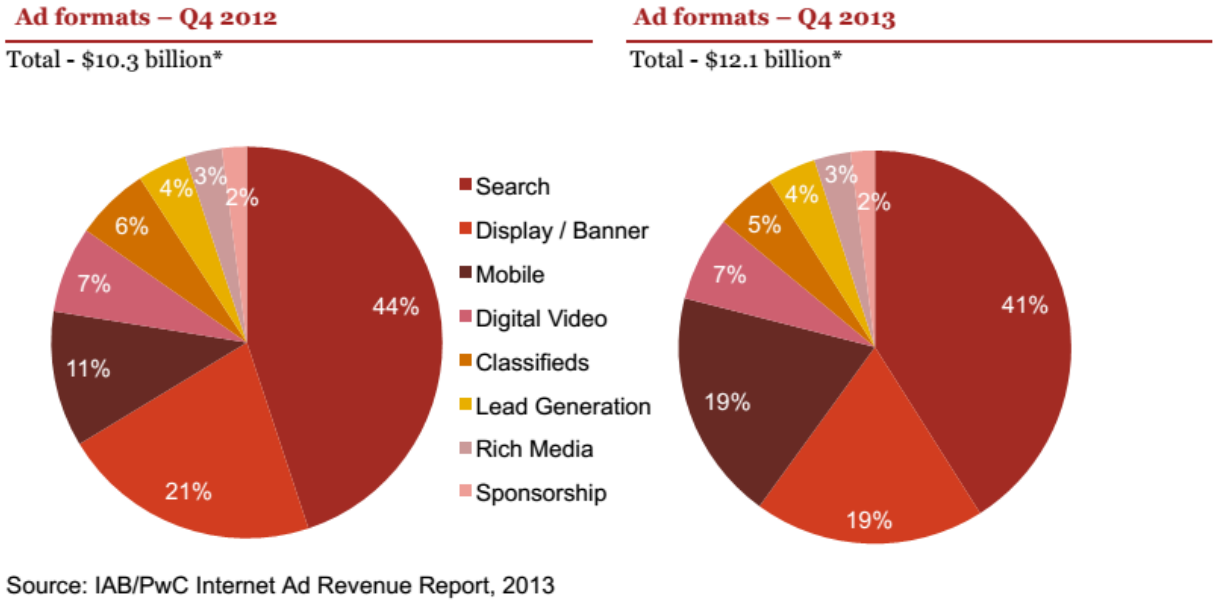

Figure 2. Share of different ad formats in internet advertising revenue. \emph{Search} ads is predominant among all (Coopers, 2013).

Most of the research work previously done models the ad assignment as a linear programming model with an objective to maximize the revenue, subject to applicable constraints related to budget etc. A practically implementable solution to an online advertising problem should be able to allocate the arriving impressions to advertisers almost immediately. Based on a query a decision should be made in real time. An accepted benchmark for an ad server to respond is five milliseconds (Gupta & Mateen, 2014). However, a linear programming model can have limited scalability due to its computational intensiveness.

Another area of research focuses on optimal assignment of display ads. However, display ads assignment models are not applicable on search ads because they both follow different mechanisms. Moreover, display ads are generally shown based on the user profile and historical data (or arbitrarily in a non-optimized system), while, search ads are based on user query making it possible to show more relevant and optimal ads.

Availability of more than one ad source for serving search ads gives the potential to choose among the sources for better outcome, in terms of clicks, revenue, etc. To the best of our knowledge, there is no

existing method that address this problem, is scalable and computationally feasible (to implement for real time decision).

Therefore, in this paper, we propose a methodology for optimally assigning search ads to an ad source for maximizing RPM and/or some other objectives mentioned later. The methodology is scalable for any scenario with any number of external ad source and an in-house ad source. We approach the allocation optimization problem differently, which takes away the main computation intensive component from ad server. This computation gives a threshold for ad assignment rule periodically for real time decision making. The real time decision system, called decision engine, then becomes as simple as assigning based on simple rules and, therefore, is fast. Moreover, the process is self-learning and can run on an ad network without any human intervention. In this paper, we show a model for a scenario with one external and an in-house ad source. We will perform case study with a real data from eBay. The case study result shows that the proposed model brings significant improvement in revenue and/or increase in in-house ad serving.

The paper is organized as follows. We first explain the methodology, which includes the data collection strategy, optimization objectives and modeling, in Optimal Search Ad Traffic . Thereafter, we show our case study and results from eBay data in Case study. In Model extension to multi-dimension, we extend the methodology for a general assignment problem with several ad sources. Finally, we conclude in Conclusion.

OPTIMAL SEARCH AD TRAFFIC ASSIGNMENT

BACKGROUND

For an ad publisher with more than one ad source allocating impressions to 'best' source can bring higher clicks or revenues. We performed our study on eBay, which has primarily two source for search ads, viz. eBay commerce network (ECN) and Google. Based on a study at eBay, it was found that one of the ad source performs better for some queries and vice versa. For example, one source performs better for “women clothing” while other performs better for “Sports” category. To capitalize on this knowledge, sharing traffic (or traffic allocation) optimally among different ad sources becomes imperative. In our work, we develop a model, which instead of using a linear programming model uses a simple rule based model for traffic allocation. The model is developed and demonstrated as per eBay's ad network system, i.e., with two ad sources as mentioned before. However, it can be extended to an advertising system with several ad sources.

ARCHITECTURE

A high level model flowchart is shown in Figure 3. The optimization model is triggered with arrival of a query. A query is scored based on the match between the query content and ad inventory listing on an ad source. Higher score indicates the corresponding ad source has “better” ad inventory to serve for the query. The score is compared with the threshold, Λ, provided by an optimization engine to make a rule-based ad source assignment. Λ denotes a set of thresholds according to the problem. For a setup like at eBay with two ad sources, Λ will be a vector $\boldsymbol{\lambda}$, with each vector element representing an ad category (explained further in following sections). After ads are served, data containing clicks, impressions, revenue, etc. is recorded. The most critical part of this model is finding the optimal threshold, Λ, in the *optimization engine* block. It uses historical data to obtain an optimal threshold, λ, for each ad category. This optimization is performed at periodic intervals, recommending the new optimal Λ for each interval. Data is collected during this interval, added to the historical data and new Λ is computed. All this process happens in real time, however, only scoring and rule based assignment is performed on the ad server. There are several off-the-shelf scoring techniques that are fast; moreover, the rule based comparison takes negligible time. This makes the model practically viable. As mentioned before, the whole process is completely automated, self-learning and can work on an ad network without any human intervention.

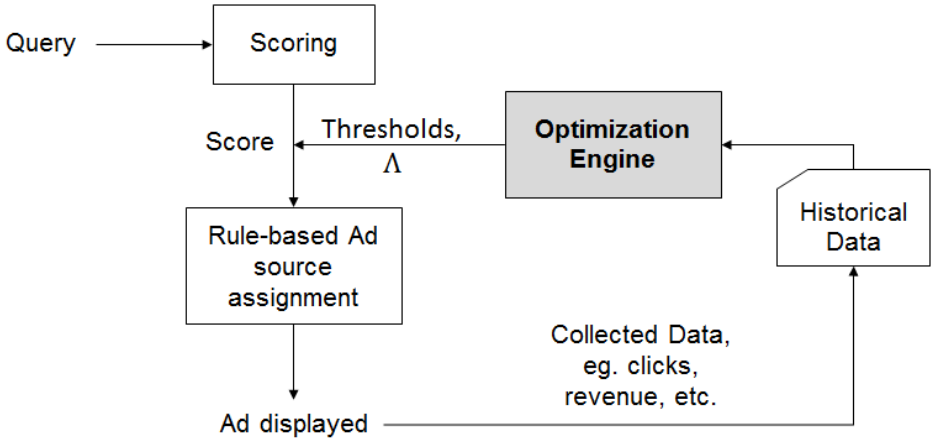

Figure 3. A high level model flowchart.

MODEL

When a query is received, it is sent to the ad server. The ad server gives a score to the query with respect to the ad sources. The query score corresponding to an ad source will be high if the ad source has plentiful relevant ad inventory to serve the query impression and vice-versa. The score falls between [0,1]. As also mentioned before, there are several off-the-shelf fast methods for doing such scoring based on matching an inventory list with query content. Any of those methods can be used for scoring, and it does not effect our model. EBay uses its proprietary technique for the scoring.

In this section, we will present the developed optimization engine for a rule-based ad source assignment. First, we explain the experimental setup and data collection approach in Experimental setup and Data collection. We then present the optimization objectives and propose a corresponding modeling approach in **Error! Reference source not found.**.

Experimental setup

In the advertising realm, the market behavior is quite fluid and dynamic over time. Therefore, we should continuously re-train our model. To obtain right data for such continuous learning we require an experimental setup for our ad network. Suppose we have K ad sources then we need $K + 1$ buckets of ad traffic. Among these buckets, traffic will be assigned to each of K ad sources separately, which means all ad traffic belonging to those buckets will be served only by the k^{th} source, where $k = 1, \dots, K$. These buckets are kept small in the tune of one percent of all traffic. The remaining major part of the traffic is put into the last bucket. This bucket has a mixture of ad assignments from all ad sources. This caters to the main traffic, of approximately 90 percent or higher, where we aim to optimally assign the ads for higher revenue. We will refer to this as hybrid traffic in the paper.

Data collection

Data is collected for all these traffic buckets at the most granular category possible. Specific to eBay's scenario, we have data at country/product-category/user-type/page-type level, called as a *category* in this paper. The daily revenue per mille (RPM) for this category is computed for the buckets, which is used as a data point. As shown in the Figure 4 for two ad source scenario at eBay, we have a scatter plot of the data points with x-axis being the in-house traffic share while the y-axis is the observed RPM. The left side of the plot has 0% in-house traffic share, i.e. it represents the bucket with externally sourced Google served ads, the right side has 100% in-house traffic share and the middle part represents a hybrid mixture traffic where the in-house traffic share can range between 0-100% and the remaining will come from Google. Besides, it will be explained later that each data point is assigned a weight. The size of the dot on the scatter plot denotes its weight, bigger the dot higher the weight. The higher weight data points are the more recent.

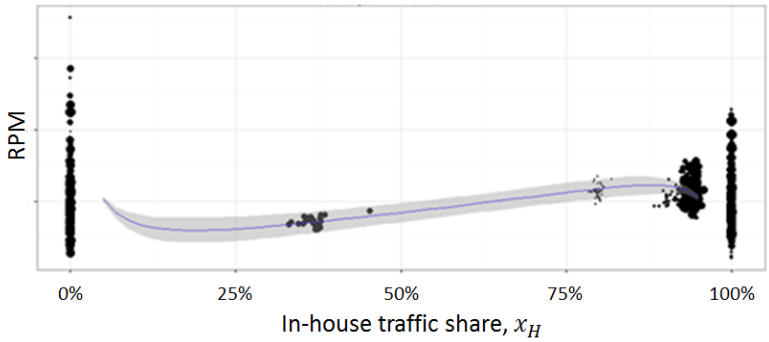

Figure 4. A sample of data for a specific ad category. The right side of the plot has 100% in-house traffic share, left side has 100% external ad source (Google), and the middle region has a mixture traffic, also called hybrid. The size of data point shows its weight, which is higher for recent data.

Optimization Engine

Suppose we denote the in-house traffic share as x_H . The developed optimization engine can determine the optimal in-house traffic share, x_H^*, based on certain objectives mentioned below. Moreover, the threshold parameter Λ regulates the in-house traffic share, with an available one-to-one mapping function between them, meaning if the optimal x_H^* is known the corresponding value for Λ can be determined. We, therefore, estimate the optimal in-house traffic share according to the following objectives,

1. Maximize RPM: Determine the optimal in-house traffic share mixture that maximizes the RPM. An example is shown in Figure 5, where we should select an optimal in-house traffic share of 77% ($x_H^* = 0.77$), as it corresponds to the maximum RPM.
2. Maximize in-house ad inventory traffic share: This objective is to boost the in-house traffic share. This is done by increasing x_H to the limit till the expected RPM is at least as high as that of a benchmark. At eBay, the benchmark is considered equal to pure Google traffic's RPM. For example, in Figure 6, a 100% in-house traffic share ($x_H^* = 1.0$) gives almost same RPM as that of pure Google traffic, hence, we should choose it as our optimal point under this objective.
3. Knowledge discovery: The system is evolving with possible addition of categories (for example, by adding a new product type). Also, the behavior may change spatially and temporally. For continuous learning and model update, we choose less or unexplored traffic share region. See for example, Figure 7, the right side of the traffic share region is not explored. We assign traffic share in that region to garner knowledge. This objective is pivotal for accurate modeling and any cold start problem.

To find solutions for objectives (1) and (2) we fit a polynomial in our data. Since recent data has more importance, we will give higher weights to them. The weights are distributed exponentially, with a weight $\boldsymbol{wi}=\exp(-\boldsymbol{\gamma d})$ Eq.(1). At weight parameter, $\gamma=0.01$, the weights are almost linearly decreasing, while at $\gamma = 0.1$, the go close to zero after about a month old (see Figure 8). This also gives us flexibility to change the effect of data on the model.

$$w_i = \exp(-\gamma d) \qquad \textbf{Eq.(1)}$$

where w_i is the weight for data point i which is d days away from current day.

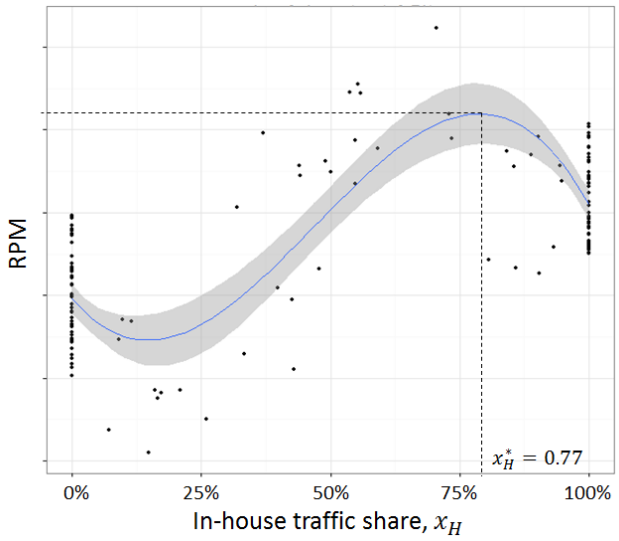

Figure 5. An example serving Objective (1). The optimal in-house traffic share of 77 percent ($x_H^* = 0.77$) gives the maximum RPM

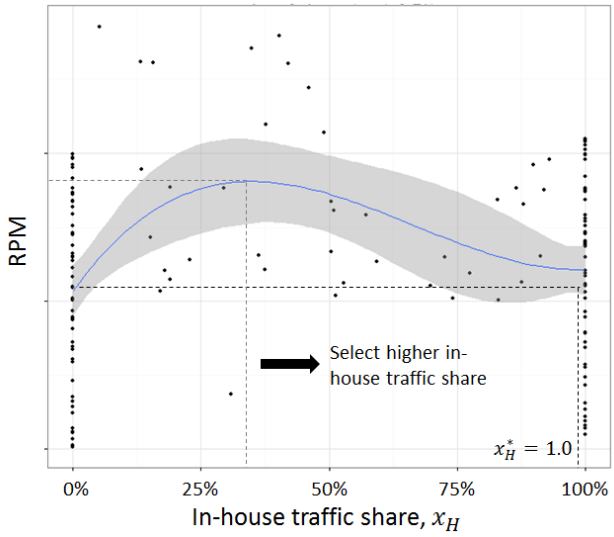

Figure 6. An example for illustrating Objective (2). Although the in-house traffic share corresponding to maximum RPM is about 30%, according to this objective we will select higher in-house traffic share that is at least as good a benchmark RPM. In above example, we can select a 100% in-house traffic share ($x_H^* = 1.0$), as it gives RPM as good as pure Google RPM, the benchmark.

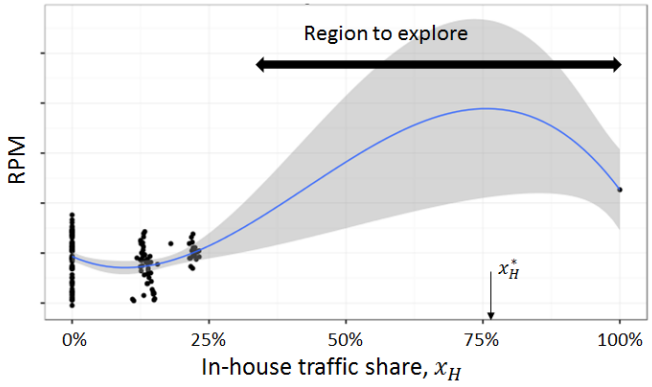

Figure 7. A case illustrating Objective (3), where we want to explore traffic share regions, which has less or no data. In this example, there is no data on the right side of the traffic share, resulting in a poor model fitting. We would assign a traffic share in that region to collect more information, for instance, $x_H^* = 0.75$.

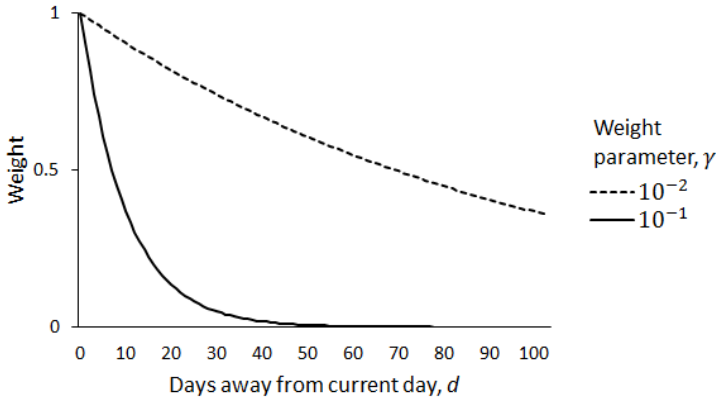

Figure 8. Weights of data with respect to its closeness to the current day. The x-axis shows the number of days the data is away from the current day and y-axis is the corresponding weight based on a exponentially function with parameter γ. For small γ, the weights decrease almost linearly, and for higher γ, it reaches zero after the data is a month old.

To determine the optimal x_H, we need to fit a curve on the observed data. In our problem, we use a polynomial function up to a degree of three to, a) avoid overfitting if higher order is used, b) obtain unique optimal solutions, as shown in examples in Figure 9-Figure 10, and c) ease of interpretation. Besides, our modeling approach will automatically address all possible situations arising from the data, as mentioned below.

- If the data model contains a maxima and a minima point then a cubic polynomial model is found (see Figure 9). Note that we constrain the modeling to have a maximum of cubic degree, this prevents multiple maxima/minima to avoid overfitting.
- If the data model has either of a maxima or a minima then a quadratic polynomial model is found (see Figure 10).
- Else, if the data model has a linear trend then a linear polynomial is found.

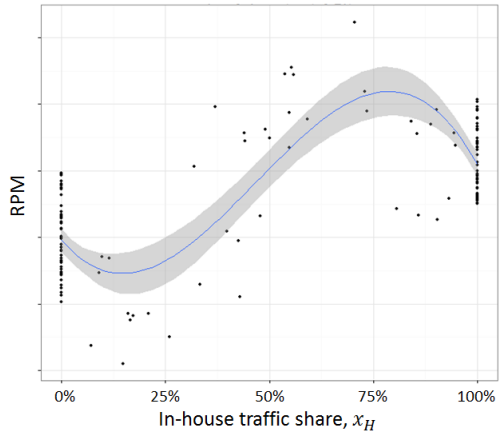

Figure 9. An example of data having a maxima and a minima point. In such case, we would usually assign the traffic share corresponding to the maxima to get higher RPM.

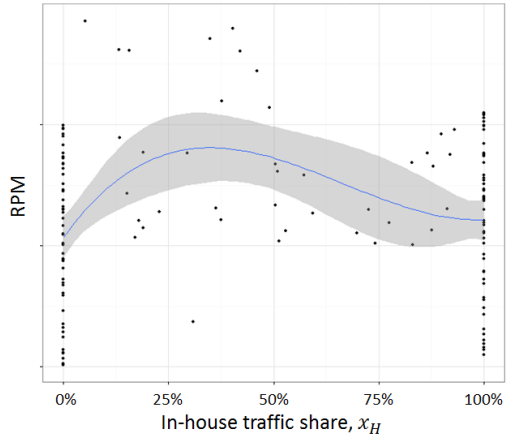

Figure 10. An example of data having a maxima point. A quadratic function is fitted to find the point and the corresponding traffic share is recommended.

The polynomial fitting is mathematically a linear regression approach with the predictor variable being in-house traffic share and the response variable is the RPM, denoted by y. The in-house traffic share ranges between $(0,1)$, while the regression model assumes it to be between $(-\infty, +\infty)$. This becomes a problem when we try to find the optimal traffic shares, when we occasionally get the traffic shares beyond the $(0,1)$ range. To avoid this problem, we use a $logit$ transformation on the predictor variable. The transformed variable denoted by x can be found using the following expression for $logit$ transformation$\boldsymbol{w_i = \exp(-\gamma d)}$ Eq.(1).

$$\boldsymbol{x = \log \frac{x_H}{1-x_H}} \qquad \textbf{Eq.(2)}$$

The inverse logit function, denoted by $logit^{-1}$, will be used later to get the in-house traffic share from x, given by

$$\boldsymbol{x_H = \frac{e^x}{e^x+1}} \qquad \textbf{Eq.(3)}$$

Suppose our fitted polynomial is given by

$$y = f(x) = ax^3 + bx^2 + cx + d$$

Where the coefficients a, b, c and d takes values depending on the depending the data.

To find traffic share for objective (1), say $x_H^{(1)}$, we find the traffic share corresponding to the maximum point of the polynomial. Using calculus, we find the optimal point.

$$\frac{dy}{dx} = f'(x) = 3ax^2 + 2bx + c$$

Setting $\frac{dy}{dx} = 0$, we can find the roots denoted by x^- and x^+ (given below).

$$x^- = \frac{-b - \sqrt{b^2 - 3ac}}{3a}$$

$$x^+ = \frac{-b + \sqrt{b^2 - 3ac}}{3a}$$

Thus, using above results, we can get optimal point, x^*, corresponding to the maximum point on the polynomial using following equation.

$$x^* = \arg_{x \in \{x^-, x^+\}} f''(x) < 0$$

where $f''(x) = \frac{d^2y}{dx^2} = 6ax + 2b$.

Thus,

$$x_H^{(1)} = logit^{-1}(x^*)$$

To find traffic share for objective (2), say $x_H^{(2)}$, we find the maximum possible in-house traffic share that gives RPM at par with a benchmark, the RPM from pure Google traffic in the case of eBay ad network. To obtain it, we solve,

$$ax^3 + bx^2 + cx + d = y_G$$

where y_G is the expected RPM for pure Google traffic. Suppose the maximum of the real roots of the above equation be denoted by x^{**}. Then,

$$x_H^{(2)} = logit^{-1}(x^{**})$$

To obtain the traffic share for objective (3), say $x_H^{(3)}$, we segment the traffic share range of $(0,1)$ into five equal segments, viz., $0.0 - 0.2, 0.2 - 0.4, \ldots, 0.8 - 1.0$. The sum of weight of the data points in each segment is computed. The segment with highest sum will have most 'relevant' data points (with consideration of their recent-ness) and vice-versa. A probability inversely proportional to this sum is assigned to each segment. Thus, the segment with highest probability will have least data points. Based on these probabilities a segment is randomly selected and a traffic share point, $x_H^{(3)}$, is randomly selected within that segment. $x_H^{(3)}$ gives us a traffic share point which we should explore for model learning.

Suppose the segments are denoted by $S \in \{1, \ldots, 5\}$. The probability for selecting any segment, $s \in S$, is given by

$$Pr(s|data) = \frac{1/\sum_{i \in s} w_i}{\sum_{s' \in S}(1/\sum_{i \in s'} w_i)}$$

After randomly selecting a segment based on probabilities in above equation, say s^*, we randomly take a traffic share point, $x_H^{(3)} \in s^*$.

With our multi-objectives we can have multiple scenarios as shown in Figure 11 below. A decision corresponding to each and combination of all scenario is presented in the figure.

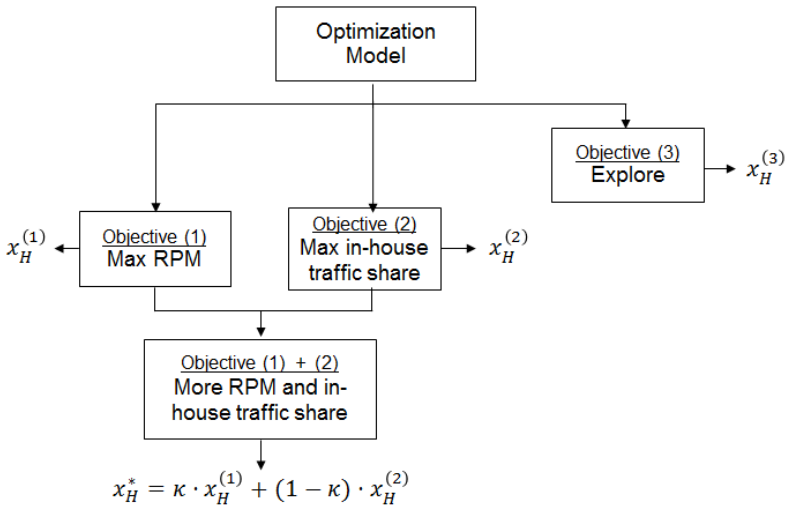

Figure 11. A decision tree for making choices between different optimization objectives.

As mentioned before, $x_H^{(1)}$ and $x_H^{(2)}$ will be used for achieving objectives (1) and (2), respectively, while $x_H^{(3)}$ will be used when we do not have enough data points to meet objective (3). Another possibility is using a combination of objective (1) and (2). This can occur when the business requires more RPM as well as more traffic share of a certain ad source for a business vision or obligation. In our case, the business vision is to get higher in-house traffic share with reasonable negative hit on RPM. In such case, we will take a weighted average of $x_H^{(1)}$ and $x_H^{(2)}$ with weight parameter $\kappa \in (0,1)$; higher κ will mean more importance on having higher RPM.

CASE STUDY

We applied our methodology on eBay's ad network. As also mentioned before, eBay has an in-house and Google ad inventory sources. The ads are categorized into product types, for example, baby, books, clothing, etc.

The traditional optimization technique at eBay was offline. Weekly analysis on the data is performed to tweak the in-house traffic share for the next week. The procedure is repeated every week.

We applied objective (1) on certain categories and traffic on the eBay ad network to automatically regulate the in-house traffic share to maximize RPM. We determine the observed lift in RPM from the traditional and proposed online optimization technique. The lift implies the HUMA: Please explain lift and the result in Figure 12.

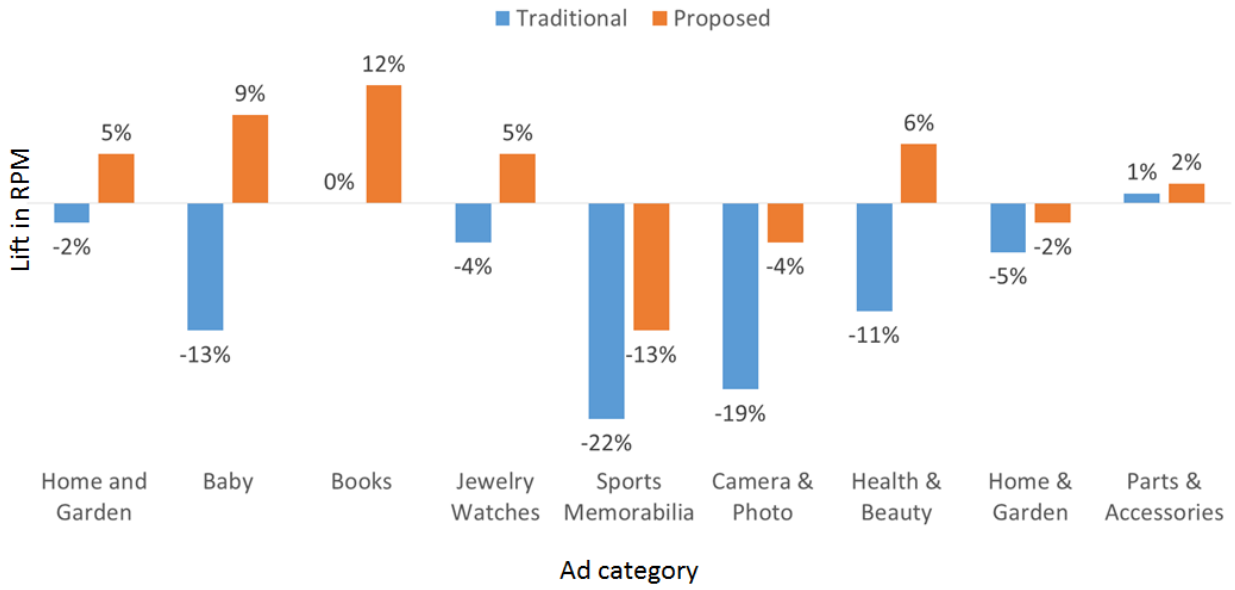

Figure 12. The percentage increase in RPM (lift), with pure in-house as baseline, for the traditional ad allocation approach and proposed optimized ad allocation. The proposed approach shows marked improvement in lift compared to the traditional approach.

Next, we performed the optimization according to the second objective on certain ad network traffic. We maintained the RPM at par with a benchmark and increased the in-house traffic share possible. As shown in Figure 13, the proposed approach could significantly boost the in-house traffic share.

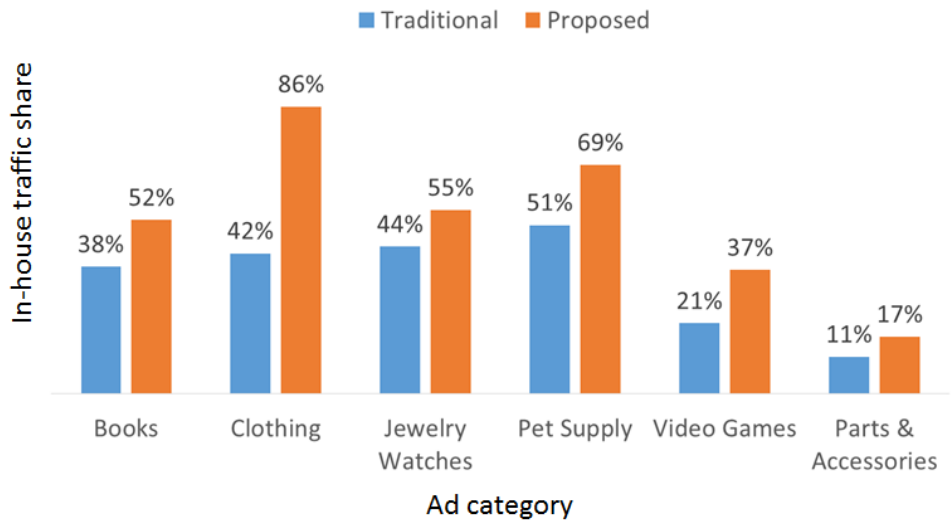

Figure 13. The improvement in percentage of in-house traffic share using the proposed approach, where the objective is to keep the RPM at par with the benchmark of control group (Google, in this case), and push the in-house traffic share to maximum possible level. The traditional approach could not identify this optimal point, hence, the in-house traffic share was set at sub-optimal levels.

Finally, we show real examples of situations in Figure 14 and Figure 15, where automatic explorations was performed for continuous learning of the model. Within each figure, the plot on the left shows the

available data before the automatic exploration imposed by the methodology, while the plot on the right shows the traffic region explored (within green circle) and the updated model. As we can note, the automatic exploration makes the fitted model more accurate, with narrower confidence intervals. This facilitates a better optimization.

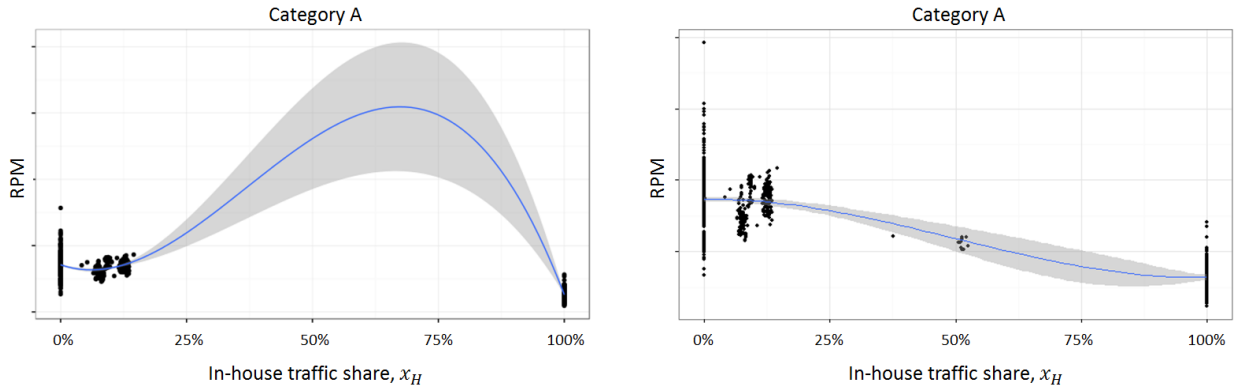

Figure 14. Before and after automatic exploration for Category A.

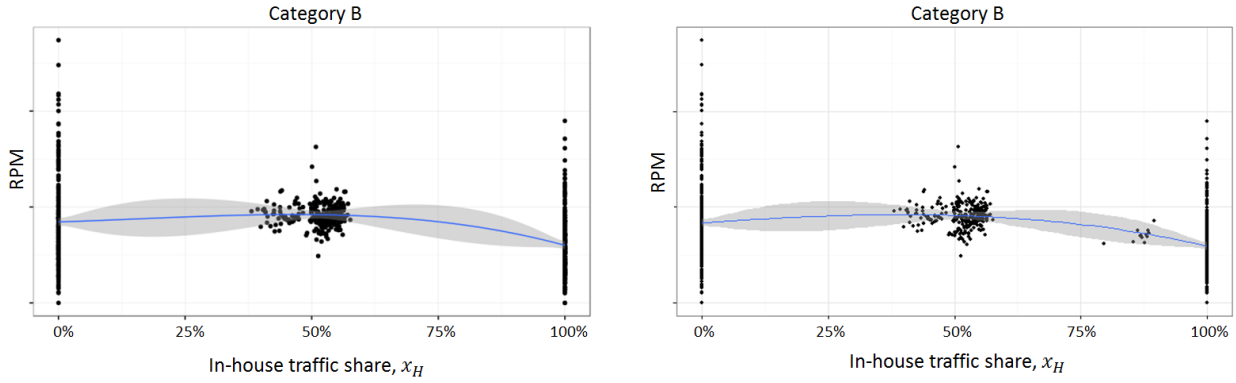

Figure 15. Before and after automatic exploration for Category B.

MODEL EXTENSION TO MULTI-DIMENSION

The proposed methodology can be extended to two kinds of multi-dimensional problem, 1) optimal traffic share assignment with respect to more than one performance metric, and, 2) optimal traffic share assignment in presence of multiple ad sources.

MULTIPLE PERFORMANCE METRIC PROBLEM

Suppose we want to find the optimal traffic share that improves multiple performance metrics, say RPM and CTR (click-through rate), together. To solve this problem we can extend our methodology to perform a multi-dimensional fitting. An illustrative example is shown in Figure 16, where RPM and CTR is plotted on the x and y-axis, respectively, and the in-house traffic share is plotted on the z-axis. The plot shows the distribution found from fitting the observed RPM and CTR for different traffic shares. The peak of the distribution gives the optimal traffic share, x_H^*, that maximizes, both RPM and CTR, together. This optimal traffic share will give us the optimal threshold for our making decision on ad source assignment. Similarly, such optimization can be done if we have more performance metric to optimize together.

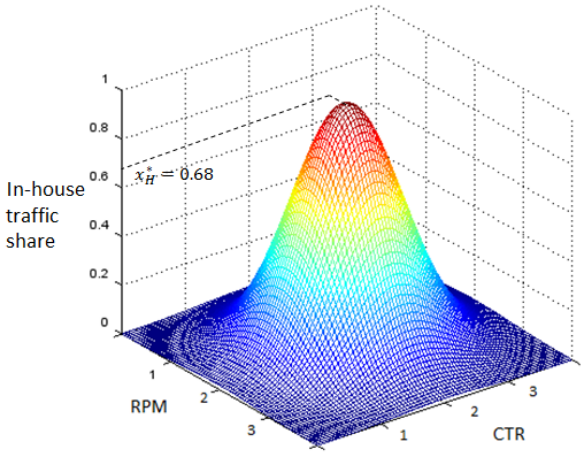

Figure 16. Illustration of multi-dimensional extension of proposed methodology for optimizing against multiple performance metrics, RPM and CTR in the above example. As we can see, maximum RPM and CTR are achieved at an in-house traffic share of 68% ($x_H^* = 0.68$).

MULTIPLE AD SOURCES PROBLEM

There can be problems when we have multiple ad sources at disposal to serve ads from. In such cases, we have to make a decision on the best ad source to fetch ad from. Suppose we are making decision (optimizing) for only one performance metric, RPM. Say we have K number of ad sources. We take the in-house ad source as the base, which means traffic shares are always measured with respect to the in-house. See, for example Figure 17, where we have three ad sources ($K = 3$), comprising of the in-house ad source, and ad sources l and m. We take a similar approach as in the one-dimensional (two ad source) problem explained above, where we took traffic shares as ratio of in-house traffic and the other ad source's (Google) traffic. Thus, we represent traffic share for ad source l as a ratio of in-house traffic (the base) and ad source l's traffic, denoted by x_l, and similarly for ad source m, denoted by x_m.

We observe RPMs for different levels of x_l and x_m. We, then, fit a distribution on it. As shown in the figure, we find the traffic shares, x_l and x_m, corresponding to the peak (maximum RPM) of the distribution, given by x_l^* and x_m^*, respectively. Each of these optimal traffic shares will give thresholds, λ_l and λ_m, respectively, in this case. When a new query arrives, we find its score and perform a stepwise comparison of the score with the threshold. Before performing the stepwise comparison, the external ad sources ($= K - 1$) are ranked according to their average RPM in decreasing order. Suppose in our illustrative example, $RPM_l > RPM_m$, then we start with comparing the query score with λ_l, if $score < \lambda_l$ then we assign the ad to source l, otherwise, we go to the next step and compare the score with λ_m, if $score < \lambda_m$ then we assign the ad to source m, otherwise we assign it to the in-house ad source. A generalization of this stepwise comparison is shown in Figure 18.

This stepwise decision process is a pragmatic approach of solving such multidimensional problem, because otherwise, for collecting data for each combination of ad sources and stepwise comparison for each such combination will grow factorially with more ad sources. For example, if we have five ad sources, $K = 5$, such brute force method will require collecting data and stepwise comparisons for $\binom{K}{2} = \frac{5!}{3!2!} = 10$, cases, as opposed to only $K - 1 = 4$ in the proposed method. This difference increase drammatically with increasing K, implying the superior applicability of the proposed method.

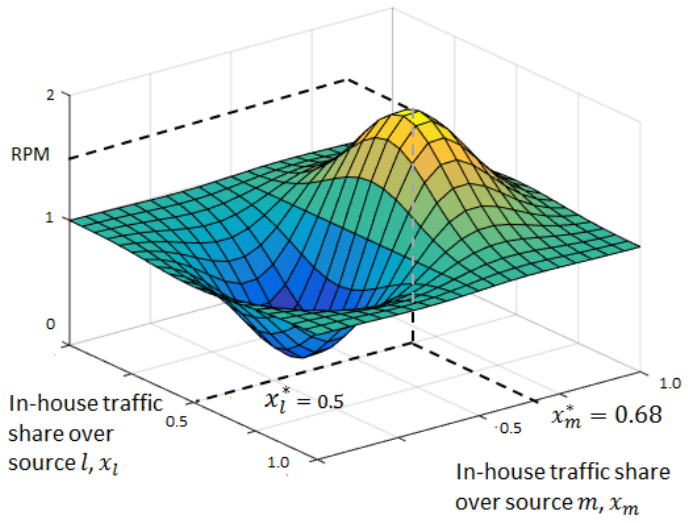

Figure 17. An illustrative example for multi-dimension extension of proposed methodology in presence of multiple ad sources.

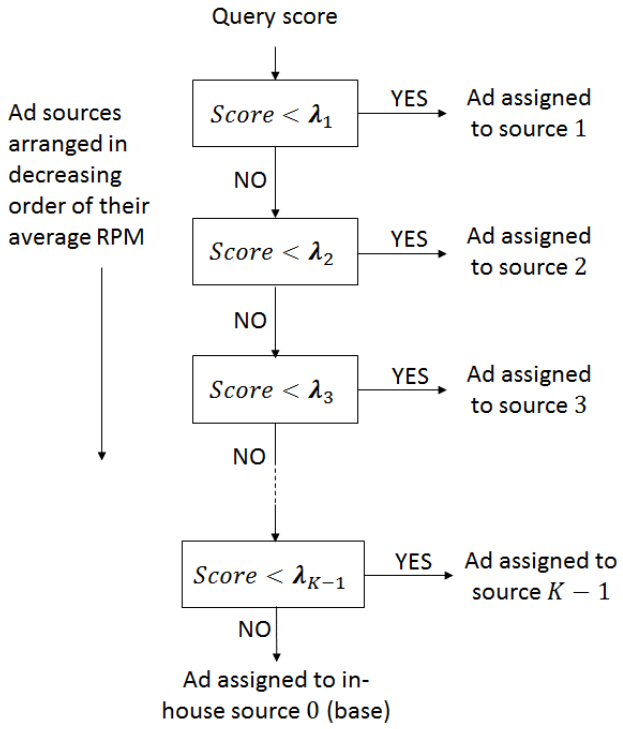

Figure 18. A stepwise comparison of query score with the optimal thresholds corresponding to each ad source. The ad sources are arranged in decreasing order of their average RPM and then comparisons are made step-by-step.

CONCLUSION

Catering to advertisements for search queries from multiple ad sources is a unique problem to certain online industries, for example, eBay and Amazon. Very little work has been done to develop an

optimization algorithm to address such problem. It is important to develop a methodology that is fast in serving an ad impression in a small amount of time (microseconds). We, therefore, propose a methodology that can make optimal decision on fetching ad from the "best" ad source in real time. The proposed methodology focuses around revenue per mille (RPM) metric for the optimization for two ad sources --- an in-house and an external source. However, it can be scaled to a general problem with multiple ad performance metrics, like, click through rate (CTR), cost per click (CPC), etc., to be optimized together. Besides, we also show the methodology's extension to a problem with any number of ad sources. The methodology works on three primary objectives, maximize revenue metric, increase in-house traffic share, or perform knowledge discovery by exploring sparse data regions. We applied the methodology for each objective on eBay ad network, where it significantly outperformed the traditional approach in increasing the revenue metric and in-house traffic. Also, we showed real examples of knowledge discovery with eBay's data.

In conclusion, we developed a methodology that can solve a critical problem of real time optimization of ad assignment in presence of multiple ad sources. In future, we aim to apply the extension of the methodology on a ad network. Besides, we aim to make methodological improvement in terms of creating a small separate traffic for continuous and fast learning for the **(HUMA: Please add to this. I forgot our plan of the future extension of having a small experimental traffic.)**

REFERENCES

Coopers, P. W. (2013). *IAB Internet Advertising Revenue Report: 2013 First Six Months' Results.* New York, États-Unis, Sondage industriel: PwC PricewaterhouseCoopers et IAB Interactive Advertising Bureau.

Cui, Q., Bai, F. S., Gao, B., & Liu, T. Y. (2015). Global Optimization for Advertisement Selection in Sponsored Search. *Journal of Computer Science and Technology*, 30(2), 295-310.

Gupta, A., & Mateen, A. (2014). Exploring the factors affecting sponsored search ad performance. *Marketing Intelligence & Planning*, 32(5), 586-599.

Radlinski, F., Broder, A., Ciccolo, P., Gabrilovich, E., Josifovski, V., & Riedel, L. (2008). Optimizing relevance and revenue in ad search: a query substitution approach. *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, 403-410.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

HUMA ZAIDI
408.439.5994
HUMA.IITB@GMAIL.COM

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.