Recommendation engine

Data Science group

April 2, 2016

Abstract

In this report we describe a recommendation engine developed using data from Centro historical campaigns and a few third party companies' data. The recommendation engine also includes a tool for an optimal budget allocation. The engine is supposed to be applied in a campaign preparation stage, as well as for running campaigns to improve their performance.

1 Introduction

Main goal of this project is recommending a set of sites which would fit best to the advertiser's objectives, for example, to maximize an audience reach, a total number of clicks, viewability rate of served impressions, and so on. The campaign may also have a mixture of a few goals. In the latter case we should maximize a number of effectively "good" impressions ("engagement rate").

Provided a set of advertiser's criteria for a given campaign (e.g. business vertical, demographics, platform, ad class, ad frequency, etc), an output of the recommendation engine should be a set of sites (publishers) ordered by some total score. This score should be calculated using a campaign key performance indicator (KPI) and similarities (probabilities to fit) to the chosen advertiser's criteria. In other words, it should build association rules between the advertiser's request and sites, and order the sites by a strength of their association to this request. These recommendation would be used during a campaign planning process, and, potentially, later during the campaign run to make tunings "on the fly".

The second goal of the recommendation engine and is optimizing a budget allocation among the sites. The latter can be done using standard (in general, non-linear) minimization tools with linear boundary conditions. These conditions may include, for example, a necessity to satisfy a total budget, a minimal budget per site, or a maximal ad frequency per a site's visitor.

Sections below provide a description of all main ingredients of the recommendation engine.

2 Data selection

To build the recommendation engine, we used the following main data sources.

• *MMS data*:
We used data collected since January 1, 2012 to reflect most recent advertising tendencies, on the one hand, and to have enough statistics to train/build the model, on the other hand. The Python script used for the data retrieval from a data base can be found by this link
`https://stash.centro.net/projects/CENDS/repos/recommendationengine/browse/Scores/MMS/mms_data_pull.py`.
We use data from monthly fact tables, so all the metrics (e.g. the number of impressions, clicks) for a given campaign are calculated per month. This data is mainly used for a click prediction model, vertical-to-sites similarity scores, and delivery rate probability scores (see their definitions below).

• *Comscore data*:
It is a source of demographic data. Using a dedicated API[1] we can download Comscore data for most of Centro's sites for last 17 months. This code calculates and applies the Comscore scores as a component of the recommendation engine:
`https://stash.centro.net/projects/CENDS/repos/recommendationengine/browse/RecommendAndOptimize/apply_cs_score.py`.

• *Kantar data*:
It is a source of site preferences within each industry/vertical. Currently, it is obtained by a direct dump from Kantar[2] of data for half a year (January-July, 2015). All Kantar verticals are mapped to Google verticals. This code calculates the Kantar scores:
`https://stash.centro.net/projects/CENDS/repos/recommendationengine/browse/Scores/VerticalSite_Kantar/kantar_scores.R`
and this one applies them inside the recommendation engine:
`https://stash.centro.net/projects/CENDS/repos/recommendationengine/browse/RecommendAndOptimize/apply_kantar_score.py`
The Kantar data is now available on
`p-drive:/DataScience/DmitryBandurin/Kantar/HUGE_pull_3863394_export.csv.gz`.

• *MOAT data*:
It is a source of an ad viewability and a user interaction rate. The MOAT data is downloaded using a direct user interface for all Centro's campaigns from April 1, 2013, and available on `p-drive:/DataScience/DmitryBandurin/MOAT/MoatExport20130401-20160302.csv`. This code applies the related viewability score inside the engine:
`https://stash.centro.net/projects/CENDS/repos/recommendationengine/browse/RecommendAndOptimize/apply_mt_score.py`.

• *Sizmek data*:
This data are used for a development of a model for an audience overlapping among sites as well as additional studies of the audience activity during a campaign, the number of clicks versus campaign flight days, ad frequencies, and so on.

[1] Special thanks to Naveed Kakal who made it working.
[2] done by Kristin Shamberg

3 Scoring

3.1 Global score

A central part in a site recommendation is defining site scores according to some criteria. Here we assume that main KPIs of an advertising campaign are related with these four: a number of clicks, gross rating point (GRP)[3], a revenue delivery rate (DR), and an ad viewability. Since a campaign may have a few goals, we define a total KPI score in a most general way:

$$KPI\ score = Clicks^{w_{cl}} \times GRP^{w_{grp}} \times DR^{w_{dr}} \times V^{w_v}. \tag{1}$$

If a campaign goal is a maximization of a number of clicks only, then $w_{cl} = 1$, and all other weights are zeros, and so on. However, we may also consider a mixed KPI, where a maximization of GRP as a main goal and the viewability as a subgoal, and take $w_{grp} = 0.7$ and $w_v = 0.3$. In our code we allow for any values of weights in Eq. 1; they are later normalzied

$$w_i' = w_i / \sum w_i, \text{ where } i = cl, grp, dr, v \tag{2}$$

to have a total sum of weights unity.

A site's global score should be proportional to its KPI score:

$$Global\ score = KPI\ score \times Sim\_score. \tag{3}$$

A proportionality factor Sim_score is a degree of similarity (similarity score) of a given site to the campaign requests. Let's call them a campaign input feature vector. Along with KPI, this input vectir typically also includes following data: advertiser name, business vertical, demographic/audience data, geographic area, platform (Desktop, Mobile, Cross-platform), an ad frequency cap (min & max frequencies), and a total budget. Using these features, and data that we have, we define the similarity score as:

$$Sim\_score = CS\_score \times Vertical\_score \times Channel\_association. \tag{4}$$

Here CS_score stands for a Comscore score, and is responsible for matching in demographic data, and $Vertical\_score$ scores the site within a given business vertical. If in a campaign planning, we decide to provide some initial input site(s) which, for some reasons, would fit best the advertiser's needs, $Channel\_association$ can be used to choose/filter other sites working in the same or similar channels as the input site(s).

Another possible application of the recommendation engine is when we are interested in recommending a set of sites which would correspond best to a set of input sites (chosen, for example, based on some prior experience). In this case, we define the global score as

$$Global\ score = Site{-}site\ similarity \times Channel\_association \times DR. \tag{5}$$

All the scores introduced here are considered below in more details.

[3]GRP = %Reach × Ad-Serving-Frequency = #Served impressions / #UVs in a target population.

3.2 Advertiser-publisher (delivery rate) score

The main goal of the advertiser-publisher score is to grade publishers for a given advertiser by their potential delivery rate. The latter is defined as a probability to deliver at least 90% of an ordered revenue, and calculated using MMS data with Centro's historical campaings. If a given advertiser never run a campaign on a given site, then we estimate its DR score using a Collaborative Filtering (CF) method with a bias correction [1]. Figure 1 shows results of DR scores for or Automotive vertical, as an example. One can see that typical score are quite high (> 0.9), while some sites demonstrate low scores (< 0.4). The low scores for the two sites are shown separately in Figure 2. We see that irregardless to adverdisers, these sites really demonstrated low DR in Centro's historical campaigns (orange bars). Also shown here are predictions for DR scores for other advertisers. Figure 3 shows examples of (most often) high DR scores.

We tried to vary the 90% threshold to 99% and 80%, and found that while it results in varying probabilities within each vertical by 3-7%, it almost does not affect the relative publsihers' raiting.

For a case of new advertisers, which did not participate in Centro's campaigns before, we calculate a vertical-publisher scoring matrix and assign to a new advertiser an average rate for this vertical calculated by a similar CF method.

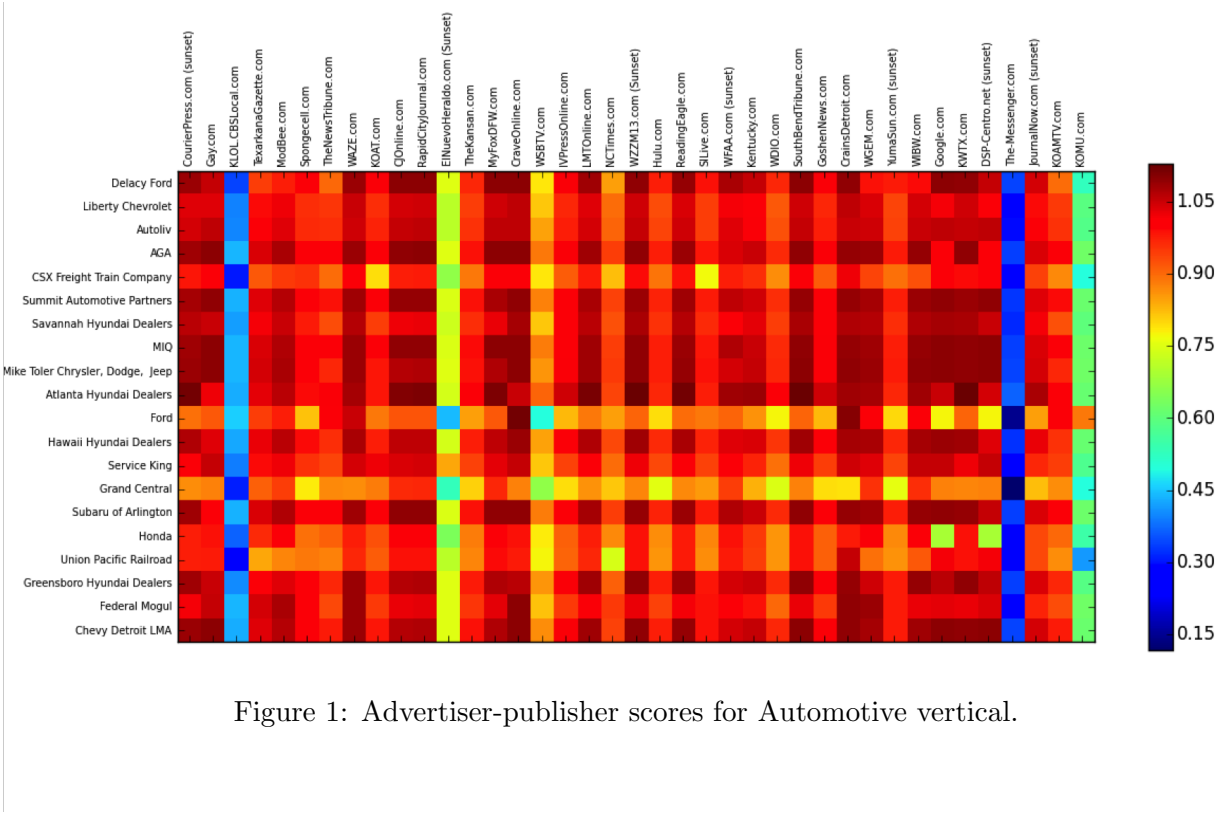

Figure 1: Advertiser-publisher scores for Automotive vertical.

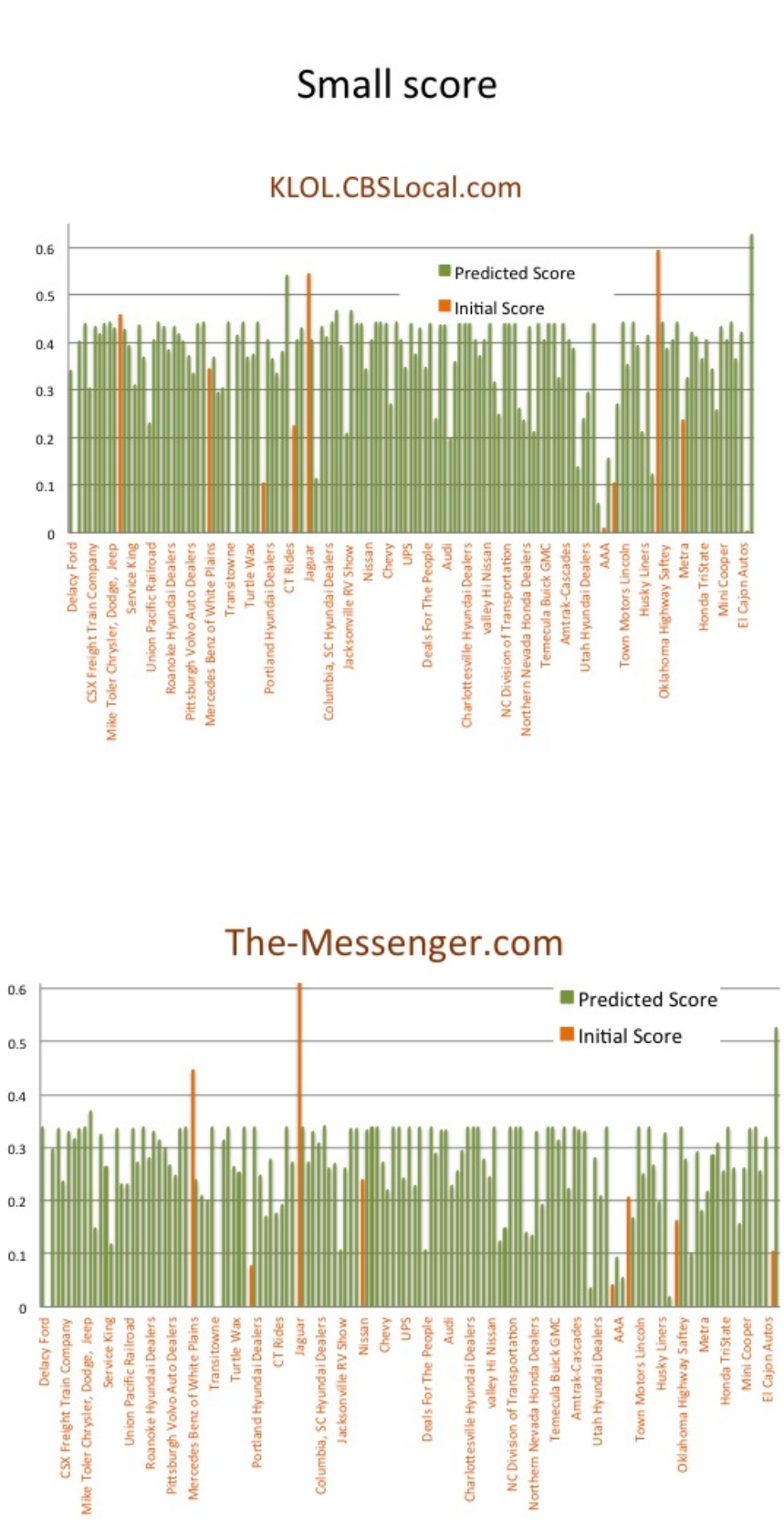

Figure 2: Example of low scores determined using historical data (marked as Initial Score) and predicted scores using the CF method.

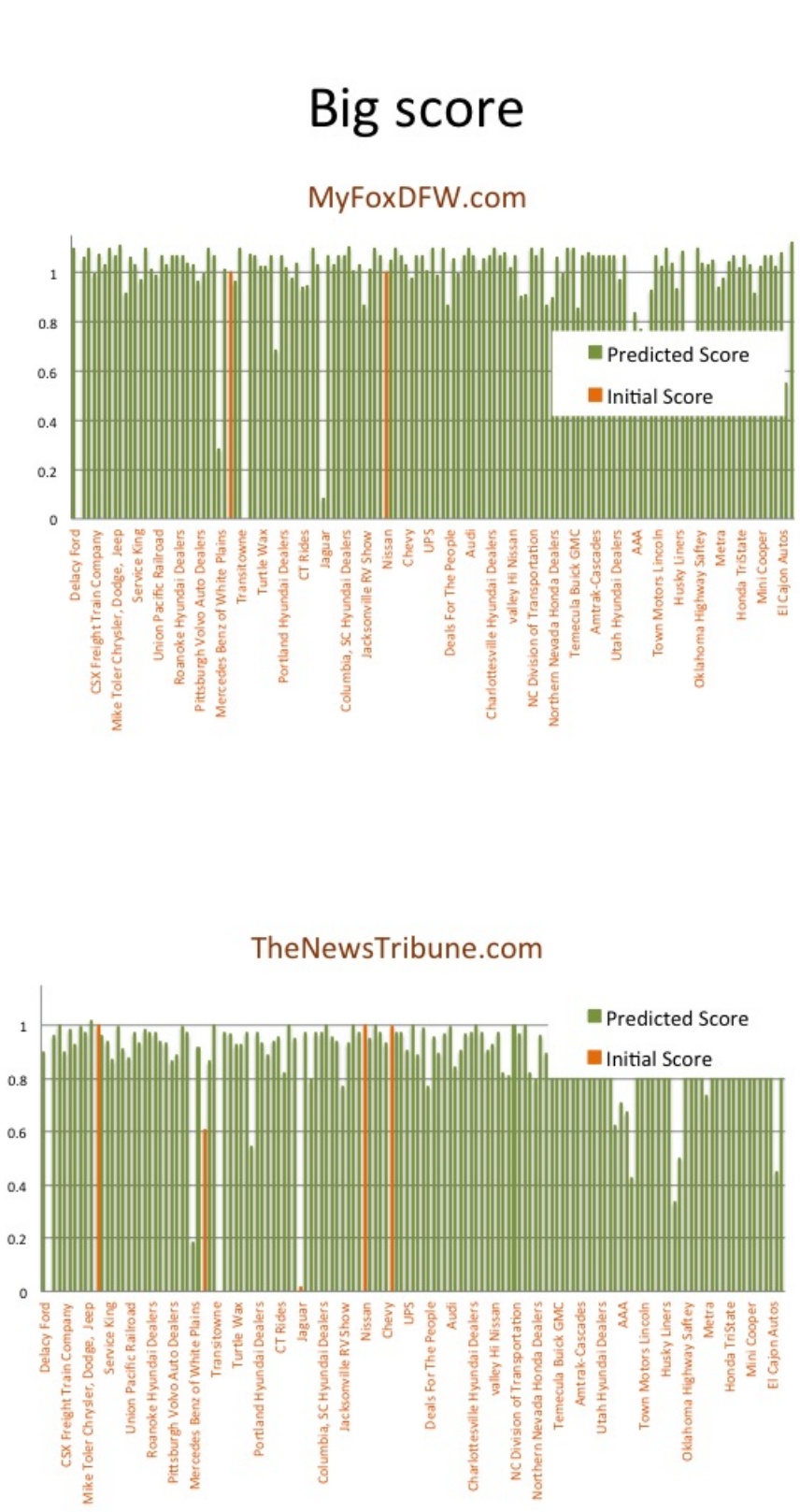

Figure 3: Example of high scores determined using historical data (marked as Initial Score) and predicted scores using the CF method.

The score prediction for new advertiser-publisher intersections, based on the CF method, has some bias. The bias is caused by similarities to other publishers' scores, defined as Pearson's correlations in their scores, and used as weights inside the CF method. As a consequence, it typically overestimates true small scores (since other similar scores are typically higher) and underestimates true high scores (since other similar scores are typically smaller) A relationship between true and estimated scores is shown in Fig. 4 for two verticals, as an example. To deal with such a bias, we estimate a size of the average bias correction needed to bring a predicted score back to the true level, please see Fig. 5.

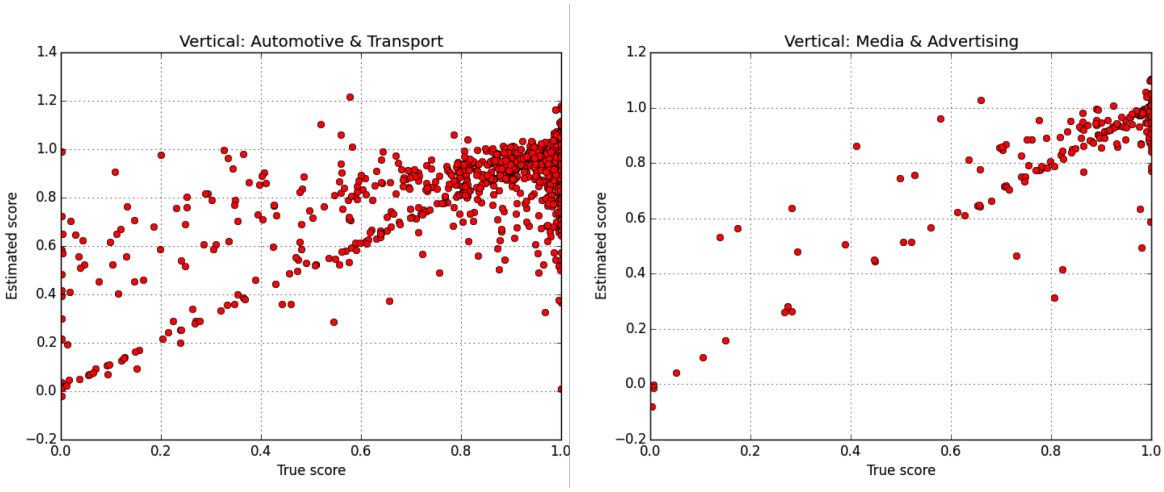

Figure 4: Bias in the predicted score caused by the CF method shown for two verticals, as an example.

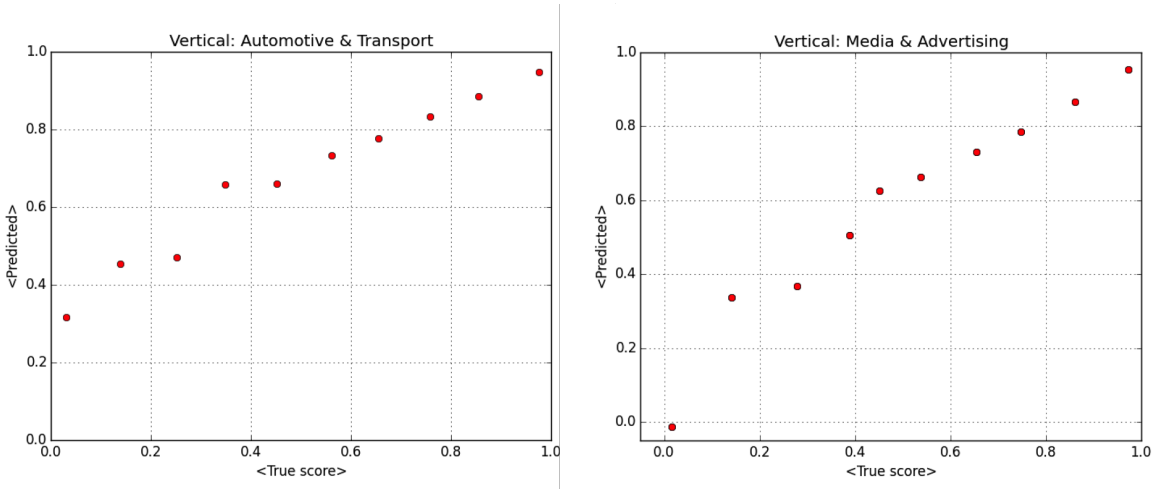

Figure 5: Estimated average bias correction for the predicted scores using CF method.

3.3 Vertical-site score

Kantar [2] provides a service by estimating a dollar amount spent by advertisers and a number of impressions shown on various sites within 57 advertiser industries, 540 sub-industries. We use this

data to rate the site preferences within each industry where a preferenc score is proportional to either a dollar amount (used as a default in our recommendation system) or a number of impressions (an alternative possible choice). The Kantar site score is defined from a normalization to the maximum values within a given vertical, as a quantile (1–100) divided by 100. So, the resulting score varies within 0–1.

Two problems we had to solve on this route is (a) to convert advertiser verticals in Kantar's definition to the Google verticals (used on the Centro platform as a default), and (b) to predict Kantar's score for the sites present in MMS but absent in Kantar data. Regarding (a), the Kantar-to-Google mapping has been done and is available here [3]. Regarding (b), for each industry, we use total site revenues from MMS, and normalize them to be distributed within a range of Kantar dollar amounts. Then for a new MMS site i, its Kantar score is estimated as a weighted average over known Kantar scores, where the weight is a square of the distance between revenues of site i and other Kantar sites. The whole procedure of reading the data, tranformation to Google vertical and a prediction of the Kantar score is implemented in one code mentioned in Section 2.

Figure 6 shows a prediction of Kantar score for new sites (not present in Kantar). This is done using a test sample with known Kantar scores. We found that a fraction of events with $|\text{True} - \text{Predicted}| < 0.10$ is about 92%.

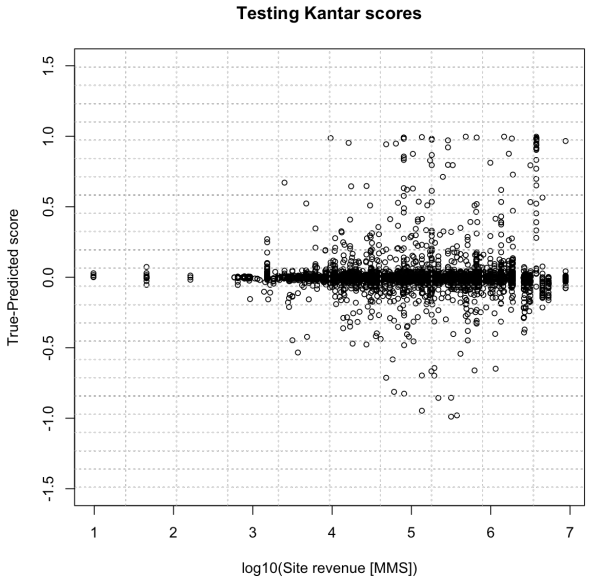

Figure 6: Prediction of Kantar score for new sites using MMS data. Plot shows true-predicted score vs a a size of overall site revenue.

Along with Kantar score, we can also calculate similar score using MMS data only. In this case, we would order sites within a given Google vertical by their overall revenue accumulated during, for example, last three years. Advantage is that we don't need to predict scores, disadvantage is that this score is based on the Centro's campaigns only, and does not take into account a global trend with industries. Nevertheless, this alternative choice is also implemented in our recommendation engine [5].

3.4 Demographic score

The audience scoring is done using multiplatform data (Desktop, Mobile and Total digital) using API to Comscore data, see Section 2. The Comscore score is defined as

$$Comscore\ score = \%Reach^{w_1} \times CIUV^{w_2} \times CIPV^{w_3} \times CIVV^{w_4} \times (PV/UV)^{w_5} \times (Time/UV)^{w_6}, \quad (6)$$

where %Reach, CIUV, etc are standard Comscore measures of audience related activities [4] (UV, PV stand for unique visitors and page views, CI is a composition index, and Time is a time spent by users during a month). They are defined per each platform and thus can be used to calculate the score for each platform separately. The weights w_i in Eq. 6 are introduced to give a preference to some particular factor(s), most important for a given campaign. The resulting score is represented as a quantile on a scale 1–100.

An important ingredient of the audience analysis is a study of audience overlapping between sites. Since it is slightly off from our description of main components of the scoring procedure, we placed it to Appendix (section 6).

3.5 Viewability score

Viewability of ads served during an advertising campaign is one of important components of a campaign planning stage, and also serves as an important indicator of a campaign performance. It is typically used as a standalone KPI, or as a component of a mixed KPI (second KPI). Three main factors characterize an ad viewability: in-view-rate (or in-view-rate $\times$ measure rate), IVR; a user interaction rate (percent of impressions where a user entered a frame of ad and remained active for >0.5 sec), UIR; and user interaction time (average length of time the user interacted with an ad), UIT. They represent different levels of the user interaction and an intensity of interaction with an ad.

As it was shown in [6], the viewability measures depend on a number of factors (where CPM cost, Browser, flight days, vertical, ad size are most important). Overall viewability score is defined in the engine as

$$Viewability\ score = IVR^{w_{ivr}} \times UIR^{w_{uir}} \times UIT^{w_{uit}}, \quad (7)$$

where, similarly to other scores, the weights give preferences for different measures. The individual measures of scores (IVR, UIR, and UIT) are calculated as probabilities to produce at a given site a value greater than a mean value for all sites. So, being probabilistic, they naturally lead to a total viewability score to be within [0,1].

3.6 Site-site similarities

In case we have a few preselected input sites, and would like to recommend other similar sites, we need to estimate a degree of a relationship of the input set with all other possible sites. For this purpose, we use historical campaigns and for each pair of sites i and j we calculate a total number of common campaigns between the two sets, i.e.

$$M^{ij}_{sim} = Set^i \cap Set^j, \quad (8)$$

where M^{ij}_{sim} is an element of a symmetrical site-site similarity matrix, $N \times N$, with N being a total number of sites in MMS. To make this matrix stochastic (probabilistic), we normalize each element

by a total sum in each row. Then to recommend a set of sites V_{recom} for a given input site vector characterized by a vector V_{input}, we need to calculate

$$V_{recom} = M_{sim} \times V_{input}. \tag{9}$$

Here V_{input} is a vector of weights which show a relative importance of the initial input sites.

The probability matrix M is calculated using Centro's historical campaigns stored in MMS. Table 1 shows an extract from the site-to-site similarity matrix where we used 5 sites from the Chicago area. Note that we made all diagonal elements zero to not recommend the input sites again. Total sum in each row is unity.

Table 1: Example of site-to-site similarity matrix.

	abc7chicago.com	chicagotribune.com	suntimes.com	nbcchicago.com	chicago.cbslocal.com
abc7chicago.com	0.000	0.079	0.060	0.026	0.037
chicagotribune.com	0.015	0.000	0.060	0.012	0.012
suntimes.com	0.014	0.072	0.000	0.013	0.006
nbcchicago.com	0.021	0.048	0.045	0.000	0.024
chicago.cbslocal.com	0.034	0.054	0.023	0.027	0.000

3.7 Channel-site association probability

While finding similar sites, as described in Section 3.6, sometimes we may prefer to see only the sites operating either within same, i.e. matching channel (e.g. News, Entertainment, Health, etc), or operating within associative channels. To estimate the association strength (probability), we calculate a fraction of common campaigns used historically for a pair of two channels, and, like in Section 3.6, define a transition probability matrix between the channels, P_{ca}. Then a probability that a given site with a vector of channels names match to the input vector of channels names is calculated as a total ORed probability

$$Channel-site\ association\ probability = 1 - \prod_{ij}(1 - P_{ca}^{ij}), \tag{10}$$

where P_{ca}^{ij} corresponds to association probabilities between the site channel i and input channel j, and the product runs over all possible combinations of i and j. Table 2 shows an extract from $N \times N$ channel-to-channel similarity matrix (where $N = 81$ is a number of unique site channel names).

Table 2: Example of channel-to-channel similarity matrix.

	Automotive	Education	Entertainment	Music	News	Real Estate	Sports
Automotive	0.149	0.000	0.052	0.019	0.260	0.039	0.032
Education	0.000	0.157	0.108	0.036	0.108	0.000	0.000
Entertainment	0.004	0.004	0.123	0.030	0.244	0.004	0.069
Music	0.005	0.005	0.107	0.119	0.172	0.012	0.019
News	0.006	0.001	0.075	0.015	0.552	0.009	0.025
Real Estate	0.024	0.000	0.032	0.028	0.247	0.417	0.012
Sports	0.010	0.000	0.275	0.021	0.319	0.006	0.010

3.8 Recommendation examples

In this section, we present some examples of recommendation sets. Table 3 shows a possible campaign input in terms of the actual recommendation code [9]. Right column contains comments on possible values of the input variables. Table 4 shows 10 top sites ordered by their GlobalScore (just some columns are shown, others are cut off for the space reason). Here 'vertical_score' is based on Kantar data, and 'cs_score' stands for Comscore (audience) score. Since we use chicagotribune.com as an input site and ask for sites in associated (to News) categories (inp_categ = 'Associated'), we may see some sites with associated channel names, e.g. Political and Real Estate sites. Please also note that since e.g. huffingtonpost.com runs in both, News and Political categories/channels, a total category association probability (0.720) is more than for just News sites (0.552).

Table 3: Example of a campaign input.

```
inp_KPI = 'Clicks'                                          #['Clicks','GRP','Delivery Rate','Mixed']
inp_adv = 'Arkansas Natural Gas Providers'                  #empty or a particular name
inp_vert = 'Energy & Utilities'                             #one of 54 Google verticals
inp_demo = 'A18_24_M_I25_40'                                #demographic bin, <Age>_<Gender>_<HHI>
inp_geo = ['Chicago-Naperville-Joliet, IL-IN-WI','National'] #['All', <Local geo>, 'National']
inp_platform ='All'                                         #'All', 'Desktop', 'Mobile'
inp_categ = 'Associated'                                    # 'All', 'Associated', 'Matched'
inp_ad_class = 'Display'                                    #'Display','Video'
inp_cost_type = 'CPM'                                       #'CPM', 'CPC'
inp_cost_subtype = 'CPM'                                    #'CPM','RTB CPM','CPC','RTB CPC'
inp_fdays = 30                                              #flight days
inp_sites = ['chicagotribune.com']                          #a list of input sites, can be empty
```

Along with this, the recommendation engine allows to also choose a set of bottom thresholds. For example, for the demographics variables (first four are in %, and last tow are absolute values):

```
comscore_thr = {'preach':1.,'ciuv':50.,'cipv':50.,'civv':0.,'av_pv_uv':0.,'av_time_uv':0.}
```

and for the viewability data:

```
moat_thr = {'ivr':0.5,'uir':0.25,'uit':0.25}
```

Table 4: Example of a recommendation set for KPI=Clicks and the other campaign conditions above (see Table 3).

site	GlobalScore	vertical_score	cs_score	categ_ass_prob	sim_score	categ
huffingtonpost.com	100.0	0.673	0.662	0.720	0.321	[News, Political]
foxnews.com	92.3	0.832	0.357	0.552	0.164	[News]
eenews.net	88.5	0.455	0.592	0.552	0.149	[News]
wsj.com	84.6	0.653	0.685	0.552	0.247	[News]
politico.com	80.8	0.792	0.575	0.552	0.251	[News]
pandora.com	73.1	0.931	0.799	0.172	0.128	[Music]
nbcnews.com	69.2	0.851	0.421	0.552	0.198	[News]
trulia.com (sunset)	65.4	0.663	0.597	0.247	0.098	[Real Estate]
careerbuilder.com	61.5	0.950	0.744	0.099	0.070	[Recruitment]
totallyher.com	53.8	0.871	0.691	0.130	0.078	[Women's Living]

In Table 5 we show a recommendation set for same campaign conditions as in Table 3, but with KPI=GRP. We also slightly change a format of output columns to show other relevant variables.

Table 5: Example of a recommendation set for KPI=GRP and the other campaign conditions above (see Table 3).

site	GlobalScore	GRP	UV	sim_score	CPM
huffingtonpost.com	100.0	3.550	1060.0	0.321	12.92
nbcnews.com	92.3	4.418	291.5	0.198	10.38
foxnews.com	88.5	5.211	290.8	0.164	8.80
wsj.com	84.6	1.524	235.1	0.247	30.10
pandora.com	80.8	4.992	1054.8	0.128	9.19
politico.com	73.1	1.241	40.5	0.251	36.98
careerbuilder.com	69.2	13.778	159.5	0.070	3.33
totallyher.com	65.4	9.381	544.4	0.078	4.89
aol.com	61.5	4.957	1646.5	0.107	9.25
trulia.com (sunset)	53.8	5.171	242.5	0.098	8.87

4 Optimization of budget allocation

A second part of the recommendation engine allows us to allocate a budget among the sites chosen from a recommendation set. Typical optimization problem that should be solved can be presented in this way:

$$\begin{aligned} \text{Maximize } \mathbf{C^T x} \\ \text{subject to : } \mathbf{Ax} \geq B_1 \text{ and } \mathbf{Ax} \leq B_2 \\ (\text{or just } \mathbf{Ax} = B), \\ x_i \geq 0, \\ \sum_i^N x_i = 1. \end{aligned} \tag{11}$$

Here $\mathbf{C}$ is a utility function (may correspond to clicks, GRP, etc, on individual sites) which should be maximized, $\mathbf{x}$ is a set of the individual site budgets, B is a total budget, and $\mathbf{A}$ is a set of N optimal coefficients which maximize the utility function. For example, a solution for a simple linear budget optimization problem for two sites, where a number of clicks can be expressed as just CTR $\times$ a number of impressions [4], and we want to maximize the number of clicks, with upper and bottom budget limits, is shown on Figure 7.

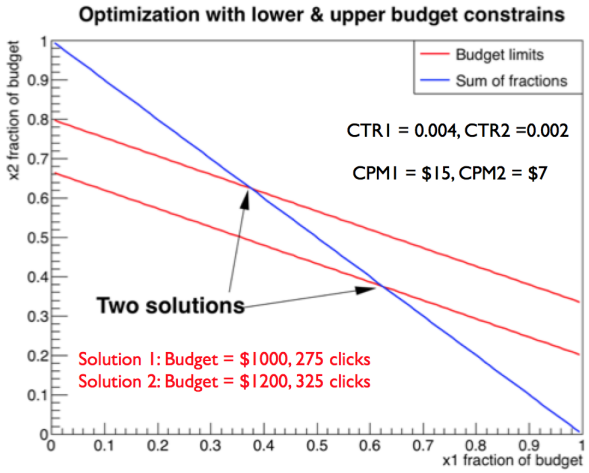

Figure 7: A solution for the budget optimization problem shown on the plot and represented by Eq. (12). A corridor between red lines corresponds to upper and bottom budget limits. A sum of two fractions is shown by blue line and is a unity always.

In general, we have to optimize a budget allocation among a larger amount of sites, the optimization problem may be non-linear, and we have to take into account a number of other factors, like similarities with respect to the advertiser's requests, mentioned above.

For a simplicity, to demonstrate the optimization engine, we choose all 10 sites from Table 4 (see the previous section) as an input for the budget allocation problem. The goal is to maximize a utility function of the selected KPI. Since in our example of Table 3 we have chosen KPI=Clicks, the optimization goal will be a maximization of the total number of effective clicks, i.e. the number of clicks times overall similarity score. As we can see from Table 3, other possible choices of the utility functions are GRP, Delivery Rate, Mixed, where KPI Mixed is a most general case presented in Eq. 1.

[4] In general, it is not quite true, since CTR itself depends on a number of impressions (see [7]), and we have to deal with a non-linear optimization problem.

In general, technically, the budget allocation problem is solved as a result of non-linear optimization problem (e.g. $clicks = \alpha\, impressions^{\beta}$ with $\beta < 1$) with linear boundary conditions caused by a linear sum of separate budget fractions, minimal budget, and ad serving frequencies. Effectively, it orders the sites by derivatives of a utility function (first go the sites with the maximum utility per dollar spent, then with the next-to-the maximum utility/dollar, and so on).

To allocate a budget among the sites, we have to specify CPM costs (otherwise, an average CPM from historical data is taken as a default value), total budget, minimum budget per site, and minimum and maximum ad frequencies. Table 6 shows a set of input variables used to optimize a budget allocation among the 10 sites above. Table 7 presents result of solving the budget allocation problem. Here Budget_frac is an optimal budget fraction for a given site, Objective is total number of effective clicks, Impr is a total number of impressions (in thousands), sim_score is the overall similarity score and Freq is an expected average ad frequency provided the number of impressions and the audience (unique visitors) on a given site. We can see that only 8 sites out of initial 10 are chosen since others are less optimal solutions, or below budget_min, and are excluded by the optimization code. They give about 2125 effective clicks total.

Figure 8 displays the solution for this problem as the objective function (effective clicks) versus the budget spent. One can see an indication for some saturation effect at a high budget.

Table 6: Input variables to optimize a budget allocation.

```
budget = 150000.      # Total budget
budget_min = 500.     # min budget per site
Freq_min = 0.0        # min ad frequency (Ads/UVs)
Freq_max = 3.0        # min ad frequency
```

Table 7: Example of an optimized set for KPI=Clicks.

site	Budget_frac	Objective	Impr	sim_score	Freq
eenews.net	0.017	57.1	85.5	0.149	3.00
huffingtonpost.com	0.299	847.8	3181.6	0.321	0.96
foxnews.com	0.182	476.4	2839.7	0.164	2.75
pandora.com	0.112	238.9	1679.9	0.128	0.28
totallyher.com	0.058	95.4	1620.5	0.078	0.42
nbcnews.com	0.109	150.1	1447.7	0.198	2.89
trulia.com (sunset)	0.092	123.8	1433.2	0.098	2.74
wsj.com	0.131	135.8	598.1	0.247	1.32
TOTAL	1.000	2125.3	12886.2	--	--

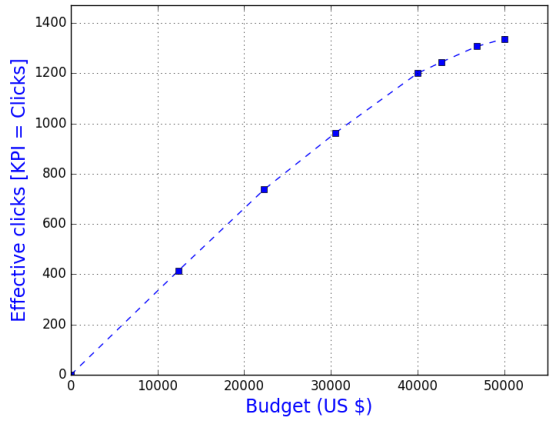

Figure 8: Total number of effective clicks vs budget (see the text in Section 4).

Similarly, if we want to maximize GRP and use a set of sites from Table 5 as an input, we get results shown in Table 8. As compared with Table 7, we have slightly modified a format to show some other features and the columns more relevant for this problem. Here Objective coincides with an effective GRP. The later is obtained from GRP as GRP×sim_score. UV_excl is amount of exclusive UVs as estimated using our model discribed in Section 6 in Appendix. The last column obj_deriv shows a derivate of the objective function (in this case, it is $\text{GRP}_i \times \text{sim\_score}_i/\text{budget}_i$ for site i). The sites are shown in the order of decreasing obj_deriv.

Figure 9 displays the solution for this problem as the objective function (effective GRP) versus the budget spent. Like in Figure 8, one can see an indication for some saturation effect at a high budget resulting from decreasing derivatives from our solver (Table 8).

Table 8: Example of an optimized set for KPI=GRP.

site	Budget_frac	Objective	Impr	sim_score	UV_excl	GRP	UV	Freq	obj_deriv
foxnews.com	0.079	17.6	1348.4	0.285	128.1	61.85	290.8	3.00	0.00149
huffingtonpost.com	0.566	114.0	6573.8	0.378	828.3	301.56	1060.0	3.00	0.00134
nbcnews.com	0.088	16.5	1267.3	0.284	120.0	58.14	291.5	3.00	0.00126
aol.com	0.267	33.0	4329.2	0.166	1200.8	198.59	1646.5	0.60	0.00082
TOTAL	1.000	181.1	13518.7	--	2277.2	620.14	3394.8	--	--

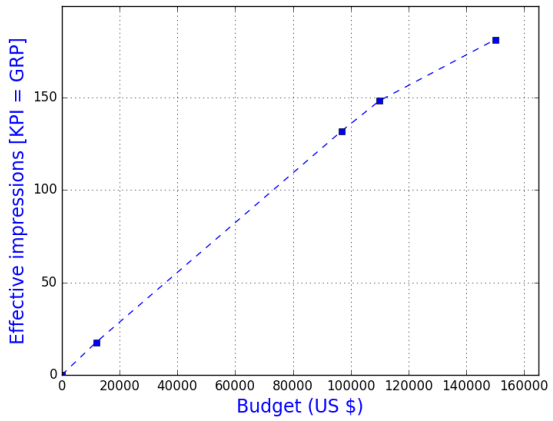

Figure 9: Total effective GRP vs budget (see the text in Section 4).

5 Discussion

In this report we presented a recommendation and optimization tool for advertising campaigns. It has been developed using mainly Centro historical data, but also external third-party data from Comscore, Kantar, Moat and Sizmek. Currently it supports main KPIs and can be extended to other possible KPIs using same approach. The code is available on Stash: a recommendation of sites from initial input sites [8], a general recommendation and a budget optimization tool [9].

As one can see from Section 3, the recommendation engine includes a number of free parameters (weights and thresholds) which are expected to be chosen by a user. While it might be intuitively clear how to choose them for different campaigns, ideally they have to be set automatically to maximize a selected utility function (KPI). Mathematically, it is just another optimization problem provided that we have enough data with the campaigns we run via the recommendation engine with different choices of those free parameters.

Figure 10 shows a data flow for processing an advertising campaign from a choice of objectives (upper left corner) to scoring, ranking and running test campaigns, tuning the parameters during a campaign running period (or after that for future campaigns), as well as a campaign data base that would contain optimized parameters for a given campaign type, and preselected recommendation and optimization sets of sites for some campaigns. These sets can be easily retrieved by a user (e.g. a campaign manager) for an analysis and following tunings during a campaign planning stage.

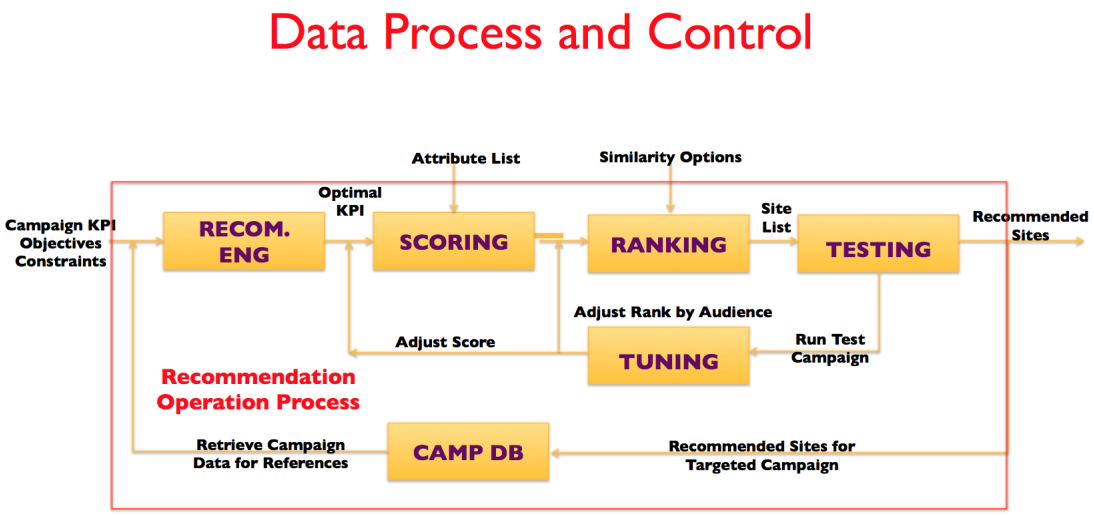

Figure 10: Schema for a campaign data processing and control through the recommendation engine and a dedicated data base.

References

[1] https://en.wikipedia.org/wiki/Collaborative_filtering

[2] http://www.kantarmedia.com

[3] p-drive:/DataScience/DmitryBandurin/Kantar/

[4] http://www.comscore.com

[5] https://stash.centro.net/projects/CENDS/repos/recommendationengine/browse/Scores/VerticalSite_MMS

[6] "Viewability study and prediction model", Data Science group, report, https://stash.centro.net/projects/CENDS/repos/viewability/browse/report/view.pdf

[7] "Click prediction model", Data Science group, report, https://stash.centro.net/projects/CENDS/repos/clickprediction/browse/reports/clicks.pdf

[8] Recommendation code that recommends similar sites from a set of input sites: https://stash.centro.net/projects/CENDS/repos/recommendationengine/browse/RecommendFromSites

[9] Recommendation and optimization code that recommends sites per advertiser's request and optimizes a budget allocation: https://stash.centro.net/projects/CENDS/repos/recommendationengine/browse/RecommendAndOptimize

APPENDIX

6 Studies and modeling of audience overlapping

Unique visitors (UV) on one site may also attend other sites. It leads to an overlap of the UVs in two or more sites, see Figure 11. If a campaign goal is to maximize a total number of *true UVs (or true GRP)*, then we have to take into account the overlap effect.

Comscore provides data on the audience overlap between any two sites in their data base. After some studies, we have found that a size of the two site overlap can be expressed as a ratio of their %Reach, see Figure 12. On these plots, "%Overlap" is normalized to the site 1 audience, i.e. it shows what percentage of additional UVs we can get from site 2. Low overlap is usually associated with large sites (i.e. with large number of UV) and as %Reach of site 1 increases relative to site 2, we see its overlap declines as well. The opposite explanation can be used for site 2.

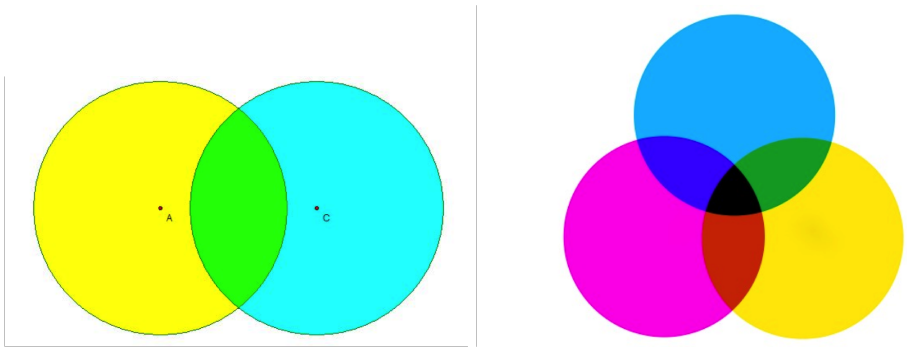

Figure 11: Audience overlap in two sites (left plot) and three sites (right plot).

Using Comscore data, we have also studied a dependence of the overlap on age, income, gender and business vertical. By looking at few Automotive and News subverticals, we did not find a noticeable dependence on income or gender, but we did find a dependence on age and some dependence on vertical. Figure 13 shows a fraction of the overlap for three age groups for Automotive vertical. We see that a younger audience (18-34) is more diverse (average overlap is 0.105) than the older audience (55+, with average overlap of 0.058).

We have built a predictive model that takes %Reach 1, %Reach 1/%Reach 2 as numerical variables and age, vertical as categorical variables to predict the overlap size (RandomForest is used as a predictive tool). A comparison of true vs. predicted values is shown in Figure 14 for two classes.

Left plot of Figure 15 shows what would be a total amount of true UVs on three sites versus the overlap between 1st and 2nd sites (o_2). Here we also assumed that the third site contributes proportionally to $o_3 = o_2^2$. Two cases are considered: same audiences (three unities), and when each subsequent site has twice smaller audience. The first point (at overlap $o_2 = 0$) corresponds to three truly independent sites, while the last point (overlap = 1) shows fully overlapping sites. With large audience overlap between sites, total unique audience grows very slowly with adding more sites. The right plot shows a possible bias in our estimates of true total audience if we don't take into account (e.g. because we don't know) an overlap with 3rd site. We see that the potential bias in the estimates can be large, and is especially large for the sites of a similar size. To minimize uncertainties, we should include only sites with the overlap $o_2 < 0.2 - 0.3$.

Sizmek data provides some information on overlaps of two, three and four sites. A treatment of this data is not straightforward and some assumptions should be made with an increasing bias for a larger amount of sites. However, we can still use them to estimate an approximate size of the effect. Figure 16 shows a size of overlap of two additional sites with the first one, being normalized to the total UVs of the first site (o_3)

versus a size of overlap of one additional site (o_2). We see that $o_3 \approx 0.25 - 0.3o_2$. About similar relationship is expected for o_4 versus o_3.

We can use all this knowledge to develop our overlap model and use it to optimize a budget allocation when KPI is related to a maximization of total amount of truly UVs or GRP (see Section 4).

Right now, mainly due to limitations in the data we have, we suggest using a parametarization for the audience overlap in terms of %Reach of the two (or three) sites. However, a direct data on the site overlaps would be more preferable (e.g. by means of an extended version of API to Comscore). As soon as we get it, one needs to develop a model that uses individual overlap between any two sites directly.

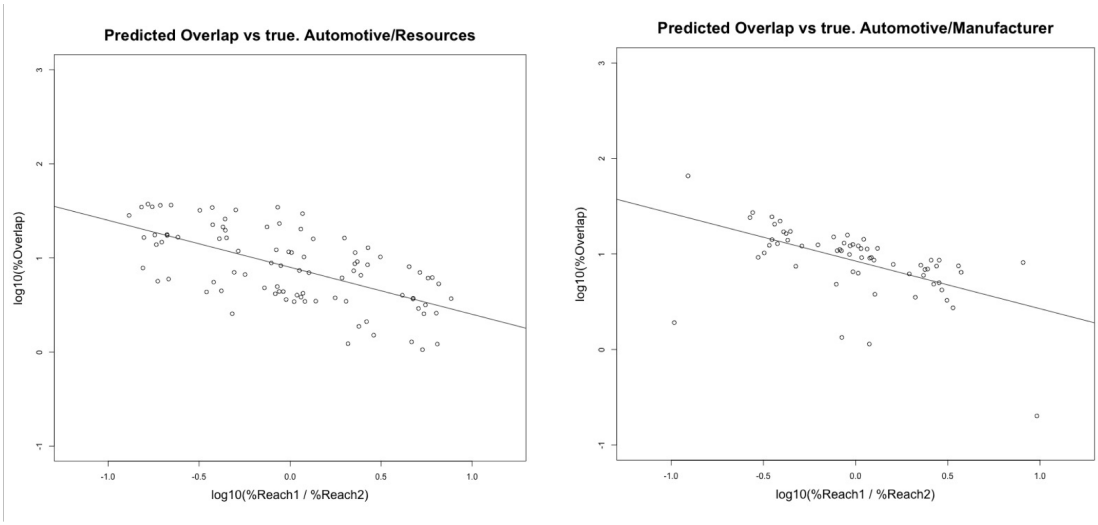

Figure 12: Dependence of audience overlap (normalized to site 1) on the ratio of %Reach of two sites.

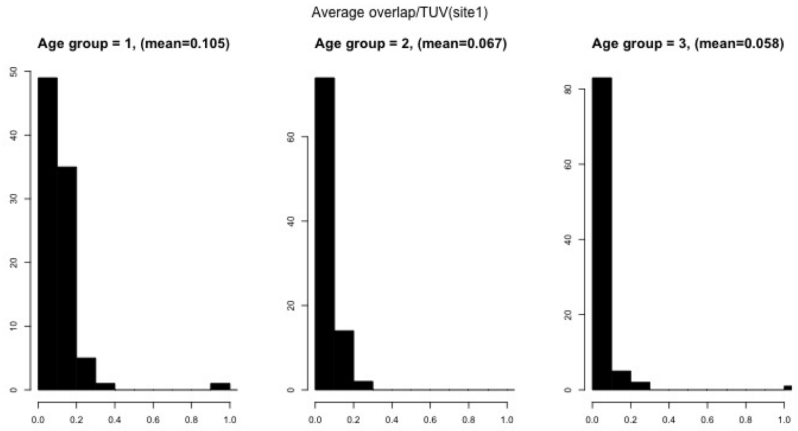

Figure 13: Audience overlap for three age groups.

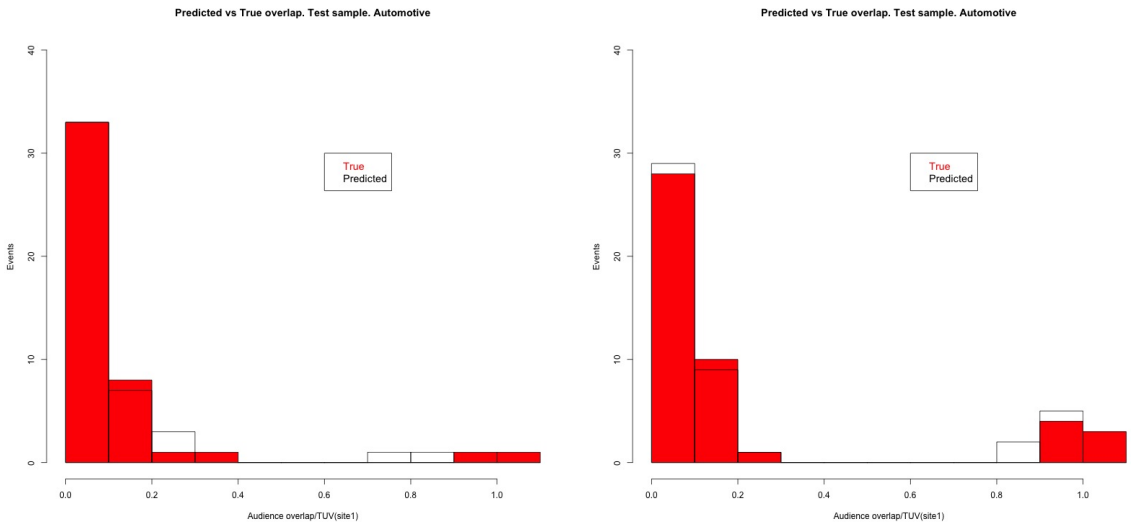

Figure 14: True versus predicted overlap values in two subverticals of Automotive (Resources and Manufacturers).

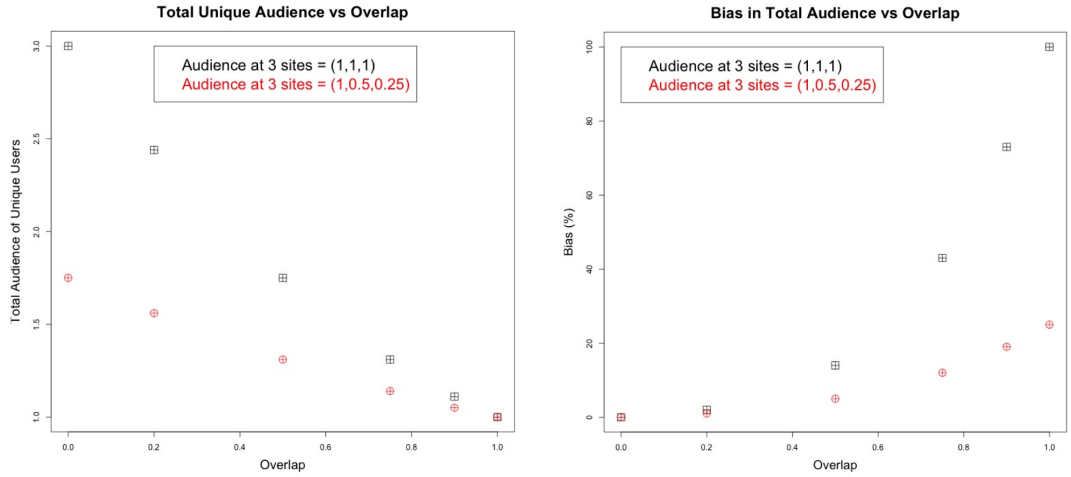

Figure 15: Total amount of true UVs on three sites versus the overlap between 1st and 2nd sites (left plot). A possible bias in our estimates of true total audience if we don't take into account an overlap with 3rd site (right plot).

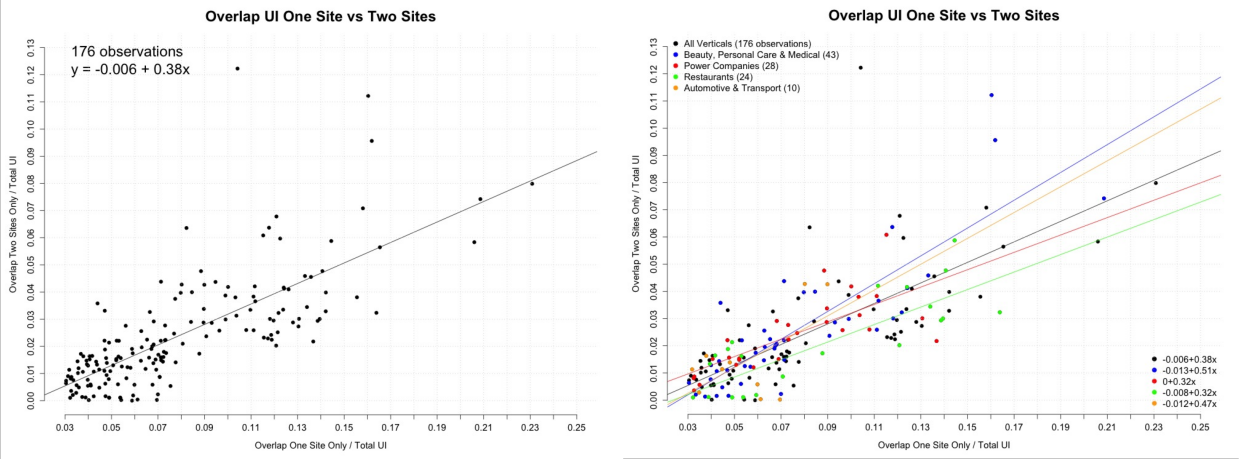

Figure 16: Size of overlap of two additional sites with the first one, being normalized to the total UVs of the first site, versus a size of overlap of one additional site (from Sizmek data).