Predictive programmatic re-targeting to improve website conversion rates

ABHIRAM PRASAD\(^1\), SNEHA CHOKSHI\(^1\) and SAHIL KHAN\(^1\)

\(^1\)Research and Development, MiQ Digital India, BANGALORE, 560001, KA, INDIA

E-mail: abhiram.prasad@miqdigital.com (ABHIRAM PRASAD)
E-mail: sneha@miqdigital.com (SNEHA CHOKSHI)
E-mail: sahilkhan@miqdigital.com (SAHIL KHAN)

Abstract. In the era of programmatic advertising, the advertisers have huge amount of first party data to leverage on enabling them to do highly granular re-targeting. Programmatic re-targeting is the ability to use data to show an ad to a user who has demonstrated an interest in your product offerings before. Re-targeting ads are a powerful conversion optimization tool and are typically known to outperform conventional targeting in terms of performance. As per 99 Firms, 41% of marketing allocation in 2018 to paid display spend was on re-targeting and for most of the websites, only 2% of web-traffic converts on the first time visit. In this paper, a conversion is referred as a purchase made and a converting user is one who made the purchase on the website. The question that arises is - "should we be re-targeting all the users who have landed on the site?". In ad campaigns which has low budgets or in campaigns where the conversion rate is really low even though a huge volume of users visit the site, it may not make complete sense to simply re-target all those users, instead we would want to re-target those who are clearly showing an intent to make a purchase either through their on-site browsing behaviors or their past conversion patterns. Through this paper we present the use of first party privacy preserving data to do predictive programmatic re-targeting of users who are going to make a conversion in the next few days given their past site-browsing and conversion behavior using a structured data science and advanced ML based framework. Additionally, this project allows to tie the model results to real time programmatic activation by the creation of user segments depending on whether the user is going to make a conversion for the first time, or is converting again. The final outputs are these user segments, which are going to be used by in house ad-traders who would be able to bid deferentially for a specified period of time against each of the segments on a demand side platform. We have successfully tested this model on 2 advertising clients and were able to capture 80-85% of the actual converts happening over the next few days of them landing.

Keywords: Programmatic advertising, Re-targeting, Display Ads, Re-marketing, Advertising technology, Real time bidding

1. Introduction

Programmatic advertising’s ability to quickly, accurately and cost-effectively target a huge audience is rapidly reshaping the media business. Re-targeting can be a powerful digital marketing tactic, allowing one to track past website visitors and their on-site behavior. It is an important complimenting strategy to other advertising strategies like content marketing, paid search, and prospecting strategy. \([1]\) According to Criteo, a global-technology company that serves online display advertisements, website visitors who
are re-targeted are more likely to convert by 43%. Additionally, according to Wishpond- a company that allows the creation of marketing tools, the click through rate (CTR) of a re-targeted ad is 10x higher than the CTR of a typical display ad. Furthermore, according to Chango a programmatic advertising company 68% of marketing agencies and 49% of brands have a dedicated budget for re targeting. There are more than enough reasons for any brand/company to consider programmatic re-targeting seriously and use it as a conversion optimization tool.

There are multiple ways to do a programmatic re-targeting. One of the most popular re-targeting strategies is to re-target all the visitors of the website. But in this approach we run into the risk of spending more than what is required for a very little conversion rate. Not all the visitors of a site have an intent to make a purchase. Apart from that we also run into the risk of re-targeting all the users the same way irrespective of what stage they were in the marketing funnel. There are other rule based re-targeting approaches like for example: re-target all the users who were present on the site in the last 2 hours and who came from this particular geography. Here we have rigid rules and this has to be updated manually as and when the features of interest change. There are other rule based re-targeting approaches like for example: re-target all the users who were present on the site in the last 2 hours and who came from this particular geography. Here we have rigid rules and this has to be updated manually as and when the features of interest change. There are other rule based re-targeting approaches like for example: re-target all the users who were present on the site in the last 2 hours and who came from this particular geography. Here we have rigid rules and this has to be updated manually as and when the features of interest change. There are some novel approaches of generating those rules from data using constrained Sparse CPA [2]. The paper "Optimizing targeting effectiveness based on survey responses" suggests the use of survey data from the audience to determine which ads would be more suitable for that set of audience [3]. However, this faces the risk of scale because very few users would be filling the survey and it would also be difficult to remove inherent biases in the responses.

Through this paper we devise a re-targeting approach that uses the privacy friendly first party cookie data from the advertiser’s/brand’s website and build user profiles based on their on-site behavior and their previous purchase patterns using an advanced data science and machine learning based framework. The machine learning model also sets a time to live period for each of the cookies that determines until what period should the cookie be re-targeted allowing the in-house traders to automatically set the same on a DSP. We further tie the results of the model into real-time programmatic activation by re-targeting various user segments that are created by the models deferentially.

2. Methods
2.1. Problem Formulation
2.1.1. Scoping the Problem Statement Currently targeting strategies are more helpful against specific end goals and KPI’s, but there’s one tactic that will always add to the overall performance of your digital strategy, and that’s re targeting [4]. It becomes important to come up with an approach that is highly data driven, dynamic, targeting at a user level but still privacy friendly, an approach that is robust across multiple advertisers, while having a good enough scale for the in-house ad-traders to target on. A major question that arises is of how long should the cookies that are likely to make a conversion be re-targeted? Furthermore, should all the cookies that are likely to make a conversion be re-targeted with similar budgets and pace? While keeping all the above questions in mind it is also important that we simply don’t end up re-targeting all the users that have landed on the site as this can prove costly for advertising campaigns that have extremely low conversion rates even though the volume of users who are landing on the site is huge and also when the budgets of the ad-campaigns are tight.

2.1.2. Proposed Approach We through this paper make an attempt to solve the problem of programmatic re-targeting keeping all the above questions in mind. This approach is going to be highly data driven while being privacy friendly as we shall be using the first party cookie data that are captured by placing pixels on the sites of advertisers to predict the cookies that are likely to make a conversion in the next few days using an advanced data science framework. These cookies would be re-targeted for a specific period of time. Since it is going to be an end to end framework from the point of taking inputs from the traders to preparing the features, to training the advertiser specific models to pushing the cookies to the segments for re-targeting this approach is robust across multiple advertisers and dynamic.
This approach also answers the question of how long the cookies need to be re-targeted as we have designed our data, ML model in such a way that it captures that information as well. In this approach we are giving the freedom of making a decision of which cookies to re-target, to an ML model and hence making a shift from the conventional rule based re-targeting.

End to End Project Framework

Stage-1: Trader Inputs-The pipeline begins with taking inputs from the ad-traders in terms of which advertiser should we run the pipeline for, what are the pixel ids placed in the site of that advertiser. We would majorly be needing the id of the pixel that is placed in the final confirmation page of the advertiser’s website and the id of the all-site pixel which is a constant pixel placed across all the pages of the website. We also take some additional inputs like the DSP segment ids which are the ids of the segments where the re-targeting cookies would be pushed and the traders would be able to associate these segments into their Line items of the ad-campaign they are running for that advertiser. Apart from this we would also be taking some keywords that occur in the URL structure of the product, checkout or cart pages of the advertiser’s website.

Stage-2: Model Training Pipeline-Once we receive the required inputs from the traders we run the training pipeline which has multiple components embedded within it. This pipeline would run once in 2 weeks. One of the first steps involve preparing the data. We pull user-level interaction data that is obtained from the pixel feed (data obtained when a pixel that is placed across all the pages of the site is triggered). We pick one particular date which is at least 15 days behind the date of running the training pipeline as our date of interest. Meaning we would be profiling the behaviors of all the users who had landed on the site on that particular day (date of interest) by taking the historical data (both landing and conversions data) of past 25 days from the date of interest. We define our target variable in this way: 1: If the user is going to convert in the next 15 days from the date of interest. 0: If the user is not going to convert in the next 15 days from the date of interest.

Therefore, we have a case of binary classification and to assign labels for the data set we would be checking if the users who had landed on the site on date of interest have converted in the next 15 days. Hence, the business problem of re-targeting when converted to a data problem would look something like this:

"Given the historical browsing and conversion behavior of all the users who had landed on the site on a particular day, predict which of those users are likely to make a conversion in the next 15 days."

Stage-3: Model Testing Pipeline-This pipeline is going to run daily at the end of the day. The goal of this pipeline would be to predict the set of users (from the pool of users who have landed on the site on that day) who are likely to make a conversion in the next 15 days. These cookies are pushed to a storage location along with a TTL (time to live) of next 15 days from the time of pushing. These cookies along with TTL after being pushed to segments are used by traders who associate them to a re-targeting line item of that advertiser’s campaign. The users for whom the model has predicted as likely to make a conversion fall under different segments based on whether they are going to convert for the first time or have already made a conversion earlier in the look back period of 25 days. This allows the ad-traders to bid variably for each of those segments. The segments are updated on a daily basis after the testing pipeline predicts the set of cookies from that day to re-target on.

Stage-4: Activation-It is not just important to predict the set of users who are likely to make a conversion in the next few days, since the primary objective is to drive business value out of the predictions. The cookies along with TTL and their segment ids after being pushed to a storage location, should now be pushed to DSP audience targeting segments, so that the in-house traders can associate them with the Line items of the ad-campaigns. This is done through a back end API developed by the in-house engineering team which takes cookie ids, segment ids and TTL as arguments and forms segments of cookie ids. This pipeline runs on a daily basis just after the job of testing pipeline.

End to End Flow:
2.2. Data Understanding

a) Pixel feed data: This data is formed as and when a user visits the site. This is obtained when a pixel-a HTML tag that can be an image of 1x1 px in dimension and CSS transparent is placed across all the pages of the site. The objective of this pixel would be to send dynamic data from the web page and fire it to the internal server. The pixel is placed in the page so that the tag fires when it is triggered. (a trigger could be a page load, button-click etc.) This table gets populated as and when a user visits the site. The pixel feed data obtained on the date of interest gives us the set of users who landed on the site on that day, and we use the same table with a date ranging from the last 25 days to the date of interest to form the browsing behavioral features. The raw pixel feed has fields like date_time, user_id, http_referer, dayserial numeric. A snapshot of the sample pixel feed data is shown in figure 2:

![Figure 2: Sample pixel feed data.](image)

We primarily use the columns "dt", "user_id_64" and "http_referer" to generate browsing features like number of sessions, avg session time, product page counts, cart page counts etc. on a user level. The data source logic to generate browsing behavioral features from the pixel feed is shown in figure 3.

b) Pixel server feed: This table gets loaded as and when a user makes a conversion on the site. Here the conversion mostly means a sales conversion. This is obtained by placing a pixel other than the all-site pixel in the final confirmation page of the website. This table consists of order/purchase transactional details like ‘dt’, ‘uid’, ‘host name’, ‘category’, ‘product_name’, ‘revenue’, ‘quantity’ etc. This feed allows us to perform two things. First, to prepare conversion behavioral features like time since last conversion, number of conversions in the past, avg duration between conversions, avg session duration before and after the most recent conversion etc. Second, this feed also allows us to create target variable (whether the user converted in the next few days or not). A snapshot of the sample of pixel server feed is shown in figure 4.

We would be using the past 25 days of pixel server feed from the date of interest and join it with the pixel feed data just on date of interest (D.O.I) to form the conversion behavioral features. When this pixel data on D.O.I is joined with the conversion/pixel server feed data with the date ranging from D.O.I to the next 15 days we would be getting our target labels whether the user has converted in the next 15 days.
Figure 3. Data source logic to generate browsing behavioral features using the pixel feed.

| dt         | id        | hostname         | referer                     | category               | quantity | product_name | revenue     |
|------------|-----------|------------------|-----------------------------|------------------------|----------|--------------|-------------|
| 2020-04-30 | 00:02:18  | 18.70.75.315     | https://checkout.adb2.ca/   | Voyeur_en_CA/Voyeur_en_CA | 2.1      | 1,03,17,00,046 | 1,99,22.99  |
| 2020-04-30 | 00:24:45  | 18.70.129.5.125  | https://checkout.adb2.ca/pay  | Voyeur_en_CA           | 1        | 297362 1     | 135.99      |
| 2020-04-30 | 00:29:00  | 17.542.245.59.18 | https://checkout.adb2.ca/confirmation | Voyeur_en_CA         | 1        | 2973896      | 110.99      |
| 2020-04-30 | 00:03:46  | 18.70.101.76.62  | https://checkout.adb2.ca/    | Voyeur_en_CA           | 1        | 1848886      | 110.99      |

Figure 4. Sample data of pixel server feed.

or not. The data source logic to generate conversion behavioral features with pixel feed and pixel server feed is shown in the figure 5.

Figure 5. Data source logic to generate conversion behavioral features using the pixel server feed.

The target variable for training the model is generated by joining the data source-1(which is the pixel feed data on the D.O.I) with the conversion data/pixel server feed from the date ranging for the next 15 days from the date of interest. This allows us to see which of the users who have landed on the site on the D.O.I have converted in the next 15 days.

The data source logic to generate the labels is shown in figure 6:
2.3. Initial Exploratory Results

As a step of the data science framework, exploratory data analysis (EDA) was carried out to understand data better. Some of the insights that were obtained from the EDA helped to build a stronger narration, helped in choosing a right look ahead and look back period and threw some light on the scale of conversions, on-site visitors etc. The following key insights were found during EDA for 2 different advertisers for multiple dates of interest. The graphs shown in each of the figures from figure 7 to figure 11 are for one of those advertisers.

- **Number of unique users visiting the site** - This gives an idea about the scale/volume of users on multiple days in the advertiser’s site. When this volume is high it may not make complete sense to simply re-target all the users landing on the site. The figure-7 shows # of unique users landing on the site for different dates of interest (various days of week so as to not create any bias). On an average there were 119048 unique users visiting the site each day.

- **% of users who have converted in the look back period** - This gives an idea about what % of users who have landed on the site that day have converted in the look back period. This allows us...
to decide if we need to consider conversion behavioral features like number of conversions by the
users in the past, time since last conversion, average duration between successive conversions etc.
for the prediction of target variable. On an average 4.66% of the total users had converted in the
look back period. The figure-8 shows % of users(of the total volume of users that day) who have
converted in the past look back period.

Figure 8. % users(of the total volume of users that day) who have converted in the look back period.

• % of users who are converting in the look ahead period-This data gives us an idea on what % of
users(who have landed on the site that day) are converting in the next few days/look ahead period.
All the users who are converting in the look ahead period belong to class 1.It hence gives us an
idea on the degree of class imbalance.On an average 1.91% of the total users are converting in the
look ahead period. This means on an average 98% belong to class 0 and 2% belong to class 1. The
figure-9 shows the % of users(from multiple dates) who are converting in the look ahead period.

Figure 9. % of users (from multiple dates) who are converting in the look ahead period.
• **% of converters (who are converting in the look ahead) who have already converted in the look back period**: This gives us an idea on what % of users those who are converting in the look ahead period have also converted in the look back period. This further allows us to segment the converters/class 1 group into those who are converting for the first time vs those who had already converted (in look back period) and are converting again (in look ahead period) hence allowing differential bidding. We found that on an average 37% of the converters were repeat converters and rest 63% were first time converters. The figure-10 shows % of converters who have converted already in the look back period.

![Graph showing % of converters who had already converted in the lookback period.](image)

**Figure 10.** % of converters who had already converted in the lookback period.

• **% of users who are landing on the site for first time (have at least not landed on the site in the last 25 days)** - This data gives us an idea on what % of users who have landed on the site that day are landing on the site for the first time (at least have not visited the site in the last 25 days). It was observed that for one of the advertisers around 72% of the visitors were visiting the site for the first time. The figure 11 shows the % of visitors who are visiting the site for the first time on that day.

Apart from the above insights there were a couple of insights which helped us in building a stronger narration and also decide the look ahead and look back periods.

• It was seen that for most of the E-commerce clients users who convert generally convert in not more than 15 days of landing, hence allowing us to set a look ahead period of 15 days.

• It is ideal to have a large look back window as it allows us to understand the behavior of the users better but the only constraint that we have is the cookie expiry period. It was observed that a cookie id associated with a particular user would roughly last for somewhere around 37-45 days. Hence, as a result we had set a look back period of 25 days, therefore a total period-look back+ look ahead- 25+15 days which is almost equal to the cookie expiry period.

• It was observed that for most of the E-commerce clients people do not convert on the same day of first time landing. For one of the clients the average duration between first landing and first conversion was 3-4 days. Since we had already observed that on an average 70-75% of the visitors were landing on the site for the first time it is highly unlikely that these set of users convert on the
same day, a part of those visitors who are going to convert are going to take at least 3-4 days for making a conversion. Hence, our approach of taking a look ahead period makes sense because we are not predicting the set of users who are going to convert on the same day or Day 0 conversions.

2.4. Data Preparation
This step involves preparing the features on the basis of which our target labels are predicted. It also involved missing and outliers values treatment. We broadly divide our input set of features into 2 categories:

- **Browsing features:**
  - Time since first landing: This feature tells us how long it has been since the user first landed on the site (in the period of look back window). This feature is prepared solely using the pixel feed with date ranging from the past 25 days.
  
  - Total number of sessions: This feature lets us know the total number of sessions the user got into. Here a user would get into a new session if there is at least a gap of 30 minutes from the previous visit on the site. This feature is also formed solely using the pixel feed (time stamp column -dt).

  - Number of sessions, total session time before last or the most recent conversion: These features gives us an idea of browsing behavioral features before the recent conversion. This feature is formed using a combination of pixel feed and pixel server feed (conversions feed).

  - Number of sessions, total session time after last or the most recent conversion: These features gives us an idea of browsing behavioral features after the recent conversion. They kind of tell us if the user is still interested in purchasing some product although they have made a conversion. This feature is formed using a combination of pixel feed and pixel server feed (conversions feed).

  - Homepage, product page, search page, cart/checkout page counts: These page counts allow us to know if the user is genuinely interested in buying some product. This feature is formed using the pixel feed.

- **Conversion features:**
  - Number of conversions: This feature tells us about number of conversions made by the user in the past look back period of 25 days.

  - converted at least once: It tells us if the user had converted at least once in the past (look back period). This feature also allows us to form multiple segments once the model predicts the set of users likely to
convert in the next 15 days. They can be segmented on the basis of whether they had converted at least once in the past or not.

c) Time since last conversion: This feature tells us how long it has been since the most recent conversion made in the look back period. This feature is formed using the pixel server feed.

d) Average conversion period: This feature gives us the min average duration between the successive conversions of the users. For those who have not converted in the look back period this feature is set to 26 days. This feature is formed using the pixel server feed.

**Target variable:**
The target label is formed using the next 15 days of pixel server feed. The user is assigned class-1 if he has converted in the next 15 days or else is assigned a label 0. At the end of missing value treatment, outlier treatment and correlation based feature selection. Feature selection has been shown to have great benefits in improving the classification performance in machine learning. [5]. A good feature set contains features that are highly correlated with the class yet uncorrelated to each other. [6] We were finally left with features like converted at least once, time since first landing, total number of sessions, home, product, search and cart page counts.

### 2.5. Modeling and Training

One of the first steps before model selection and training comes choosing the right evaluation metric. Evaluation metric plays a critical role in achieving the optimal classifier during the classification training. [7]. Choosing a right evaluation metric depends on multiple factors like the business objective and class imbalance. Performance metrics for binary classification are designed to capture trade-offs between four fundamental population quantities: true positives, false positives, true negatives and false negatives. [8]. There are a wide range of evaluation metrics to choose from for binary classifiers. Some of the most common binary classification metrics are: threshold metrics [9] like accuracy, precision, binary recall, F1, F-beta scores, false positive rate and rank metrics [9] like ROC curve, precision-recall curve. However, for highly imbalanced data sets class imbalance problem is encountered and results in classifier’s sub optimal performance. [10] Evaluation metrics like accuracy cannot be considered because it is heavily dependent on the class skewness. The business objective of this project has more room for false positive kind of miss classification and has very little room for false negative kind of miss classification. This essentially means we would rather classify a non convert as a convert and re-target him than miss classifying a convert as non convert. Hence, the most suitable metric was binary recall as this penalizes false negatives kind of miss classification. However, it is important to find a balance between binary recall and the % positive predictions from the volume. The model would have a very high binary recall of may be 100% if it predicts the entire volume as converts. This essentially would mean re-targeting all the users who have landed on the site, however we do not want that. We want to keep the % of positive predictions on the lower side meaning we would want to re-target only those who genuinely have chance of conversion. Hence, we need to find a sweet spot and find a compromise between binary recall and % positive predictions.

Now when the examples across the classes are imbalanced, many machine learning algorithms fail and metrics used to evaluate those models, such as classification accuracy, become dangerously misleading. [11] To deal with the problem of class imbalance, there are multiple ways of doing so. Some of the common methods include:

- **Stratification:** Stratification is the technique to sample evenly based on the sample classes so that the training and validation sets have similar ratio of classes. It is essential to ensure your training and validation sets share approximately the same ratio of examples from each class, so that you can achieve consistent predictive performance scores in both sets. Scikit-learn has an implementation...
of stratification called StratifiedKFold that ensures both the training and cross validation sets have the same class ratio.

- Re-sampling to get more balanced data: The most straightforward technique to handle the class imbalance is to balance the data by re-sampling. Down-sampling the majority class, up-sampling the minority class or synthetic sampling techniques like SMOTE are some of the methods to balance the data.

- Class-weighted/Cost sensitive learning: Without re-sampling the data, one can also make the classifier aware of the imbalanced data by incorporating the weights of the classes into the cost function. Intuitively, we want to give higher weight to minority class and lower weight to majority class. Cross entropy is a common choice for cost function for many binary classification algorithms and is given by:

Cross-Entropy = \(-y \log(p) - (1-y) \log(1-p)\)

where \(y\) is the class binary indicator (0 or 1) and \(p\) is the predicted probability for instance belonging to class 1. To incorporate the weights of two classes (0 and 1) into the cross entropy, one can define a weighted cross entropy:

Weighted Cross-entropy = \(-w_0 y \log(p) - w_1 (1-y) \log(1-p)\)

in which \(w_0\) and \(w_1\) are the weights for class 1 and 0, respectively. Cost Sensitive learning can also be achieved by setting the parameters like `class_weight` and `scale_pos_weight` in many scikit-learn classifiers such as support vector machine (SVM) and Random Forest, XGBClassifier. In class-weighted/Cost sensitive version of XGBoost as a part of tuning and selecting the model for training, we had used 2 models for comparison namely- XGBoost and CatBoostClassifier. However, since XGBoost was performing slightly better in terms of binary recall and % positive predictions, XGBoost was the chosen model. The interpretation of miss classification errors may differ across classes. This is referred to as cost sensitivity of miss classification errors and is a foundational challenge of imbalanced classification [11]. The hyper-parameter `scale_pos_weight` was explored and played around with extensively in this project. It is designed to tune the behavior of the algorithm for imbalanced classification problems. By default, the `scale_pos_weight` hyper parameter is set to the value of 1 and has the effect of weighing the balance of positive examples, relative to negative examples when boosting decision trees. [12] For an imbalanced binary classification data set, the negative class refers to the majority class (class 0) and the positive class refers to the minority class (class 1). XGBoost is trained to minimize a loss function and the “gradient” in gradient boosting refers to the steepness of this loss function, e.g. the amount of error. A small gradient mean a small error and, in turn, a small change to the model to correct the error. A large error gradient during training in turn results in a large correction. The `scale_pos_weight` value is used to scale the gradient for the positive class. This has the effect of scaling errors made by the model during training on the positive class and encourages the model to over-correct them. In turn, this can help the model achieve better performance when making predictions on the positive class. As such, the `scale_pos_weight` can be used to train a class-weighted or cost-sensitive version of XGBoost for imbalanced classification. We have tried to compare the performance of the model with three methods broadly:

a) Probability Threshold Moving: This method of threshold-moving method that a posteriori changes the decision threshold of a model to counteract the imbalance, thus has a potential to adapt to the performance measure of interest [13], in this case the binary recall. Below in figure-12 is a snapshot of model performance with cross-validation folds when we just used probability thresholding to handle imbalance data. This method was however not chosen for final training because

b) Synthetic sampling with threshold moving: In this method we have used SMOTE for synthetic sampling of the data and the used the thresholding. The sampling was done just on the training folds and
the cross validation folds had the same real distribution. This was done by performing SMOTE fold wise while doing the Stratified cross validation. This ensures that there is no data leakage to the validation folds and hence preventing over optimistic results. The following figure 13 shows the model performance on validation folds with the method of synthetic sampling and thresholding.

![Figure 12. Thresolding.](image1)

![Figure 13. SMOTE and thresholding.](image2)

c) Cost sensitive learning with threshold moving: In this method we had used the concept of cost sensitive learning discussed earlier along with probability threshold moving. This method gave some best results balancing out the binary recall and % positive predictions. The following figure 14 shows the model performance with cost sensitive version of XGBClassifier with thresholding:

![Figure 14. Cost sensitive learning and thresholding](image3)

Finally, since we were getting a good balance between binary recall and % positive predictions from cost sensitive learning and thresholding method, this method was adopted for the final testing. This method was giving a binary recall somewhere around 80% with around 33% of the population predicted as positives. This means that the model was able to rightly capture 80% of the actual converts who are actually going to convert in the next 15 days while around 33% of the volume of users that day were predicted as converts which is a totally acceptable volume (compared to re-targeting all the users who are landing on the site).

3. Results
We have tested the approach for 2 advertisers for multiple dates of interest and the model was able to capture roughly around 80-84% of the actual converts for one of the advertisers and around 77-81% for the other advertiser. The % of positive predictions from the total volume lied between 32-37%. This essentially means the traders would not have to re-target all the users landing on site but somewhere around 30-35% but still capture 80% of the actual converts that are going to happen over the next few days. The following figures 15 and 16 show the results obtained for the advertisers carried over multiple dates of interest as test data set.

Once we receive the predicted positive cookies from the model we divide them into majorly two tables based on one of the derived input feature which tells us if the user had converted earlier in the look-back period. Those who are converting for the first time are placed in a different segment than those who are making a repeated conversion.
4. Activation Strategy
Once we obtain the tables, they are uploaded into a DSP segment along with a TTL period of 15 days which is equal to our look-ahead period. This essentially means that the cookies will be re-targeted for the next 15 days from the time of their upload. The 2 segments created are updated on a daily basis as and when the cookies are predicted and sent into tables at the end of each day. If the predicted cookies that day are already present in the segment their TTL would now get updated to 15 days starting from the latest time of upload. The traders can then associate these segments to the re-targeting strategy Line item and bid deferentially based on the type of the segment.

Once the trader associates the segments for the first time, after a buffer period of 15 days a validation
pipeline which checks how many of the users from that day that actually converted over 15 days did the model predict right. This pipeline would run on a daily basis to check the overlap between the actual converts and the predicted positive set of cookies.

5. Conclusion

As the world is moving towards programmatic media and as marketers strive to compete with more brands vying for customers’ attention, a comprehensive re-targeting strategy is necessary to deliver results and pave the way for a deeper understanding of customer intent. Before re-targeting, many marketers would purchase ad space on sites that they thought their readers frequently visited. From there, marketers would use demographic data to target ads based on whom they thought their audience was. This data focused on things like a prospect’s age, gender, location, occupation, and interests. While this could be effective for some campaigns, it still relied on a marketer’s inherent assumptions about their ideal customer. So what is the solution? We through this project presented in the paper have used a predictive model harnessing the power of privacy preserving first party cookie data to solve the problem of re-targeting. In the programmatic ecosystem, real time bidding within DSP’s have made re-targeting campaigns particularly powerful. This approach is highly data centric and data driven which in turn means we are not relying on any inherent assumptions. We have tested this approach for 2 advertisers over multiple dates and were successfully able to capture around 80% of the actual conversions taking place while also ensuring we do not re-target everybody blindly. As the next steps we are in the process of completely automating the pipeline end to end which can make the project highly scalable.

References

[1] Braux P D 2019 12 Statistics to Make You Consider Retargeting URL https://www.spiralytics.com/blog/retargeting-statistics/

[2] Grbovic M and Vucetic S 2014 Generating ad targeting rules using sparse principal component analysis with constraints WWW WWW ’14 Companion pp 283–84 URL https://doi.org/10.1145/2567948.2577351

[3] Farahat A and Mallon K 2013 Optimizing targeting effectiveness based on survey responses URL https://patents.google.com/patent/US20130046613A1/en

[4] Marouchos C 2020 Programmatic Advertising 101: Retargeting Ads

[5] Braytee A, Liu W, Catchpoole D R and Kennedy P J 2017 Multi-Label Feature Selection using Correlation Information Conference on Information and Knowledge Management pp 1649–56 URL https://doi.org/10.1145/3132847.3132858

[6] Hall M A 1999 Correlation-based Feature Selection for Machine Learning URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.455.4521&rep=rep1&type=pdf

[7] M H and and S M 2015 International Journal of Data Mining & Knowledge Management Process 5 01–11

[8] Natarajan N, Koyejo O O, Ravikumar P K and Dhillon I S 2014 Consistent Binary Classification with Generalized Performance Metrics (NIPS’14 vol 2) (MIT Press) pp 2744–52

[9] Jeni L A, Cohn J F and Torre F D L 2013 Facing Imbalanced Data—Recommendations for the Use of Performance Metrics International Conference on Affective Computing and Intelligent Interaction vol 2013 pp 245–51
[10] Wasikowski M and Chen X 2010 *IEEE Transactions on Knowledge and Data Engineering* **22** 1388–1400

[11] Brownlee J 2020 *Imbalanced Classification with Python: Choose Better Metrics, Balance Skewed Classes and Apply cost-sensitive learning* (Machine Learning Mastery)

[12] Brownee J 2020 How to Configure XGBoost for Imbalanced Classification

[13] Collell G, Prelec D and Patil K R 2016 *CoRR* arXiv preprint arXiv:1606.08698