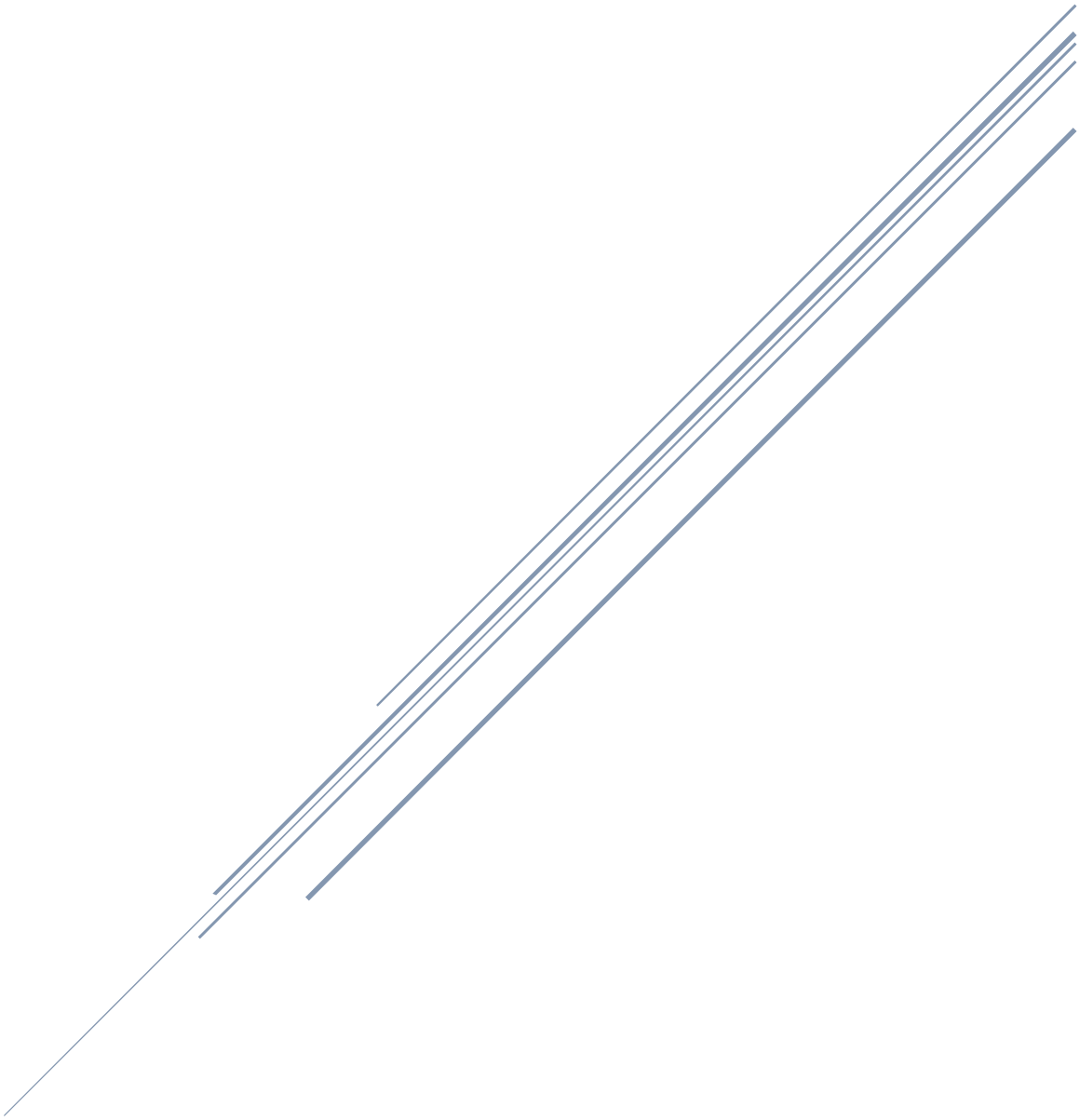
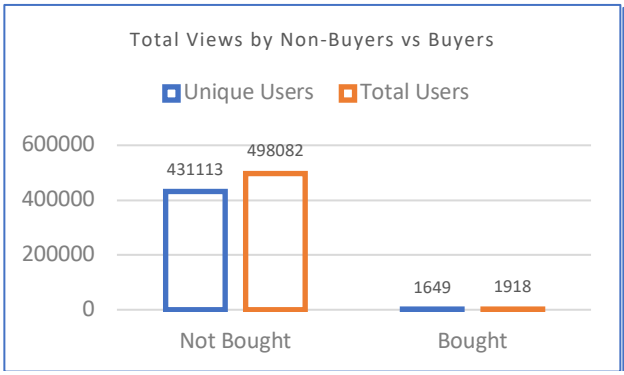
# LEAD GENERATION FOR EUREKA FORBES

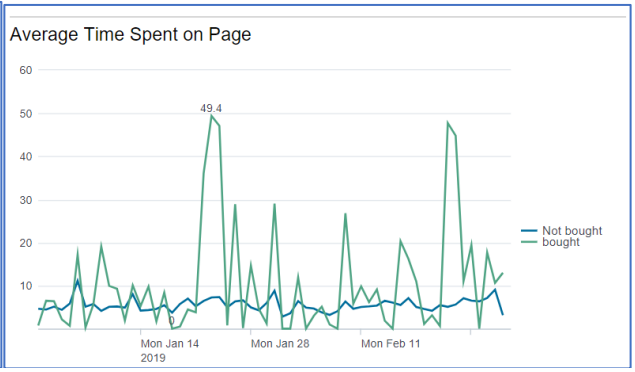


## BUSINESS PROBLEM

Eureka Forbes is operating as a market leader in the water purification industry in India, where a huge population of the country is still using untreated water or traditional water treatment methods such as boiling and so, there is still huge room for penetration in the market. Their business model is significantly reliant on their extensive network of direct sales but chasing every possible lead, door to door, is certainly not going to allow it to maintain its position for too long.

Moreover, other digital marketing initiatives by the firm without proper customer segregation and targeting have also not been as successful as one might hope. In any business, customer is the key and companies spend millions of dollars every year in trying to understand customer behavior and demographics. Similarly, to remain relevant in this $2 billion industry and sustain their market position, Eureka Forbes has to streamline its marketing efforts to target its potential leads more smartly in order to optimize its promotional efforts and spending. In summary, Eureka Forbes noticed significant number of digital consumers visiting the website (generating immense customer data) but was not able to turn these visitors into prospective consumers (see below).




### Opportunity

Currently, there is great potential market in the household segments as majority of Indian households, especially in rural parts, are still using unprocessed or traditionally treated water, and so an increase in the prospective consumer conversion rate could greatly help Eureka Forbes enlarge its market share and increase revenue. In previous marketing campaigns, Eureka Forbes did not leverage the digital footprints data captured by the website.

The digital traffic generated by their website generates huge amounts of customer data, which could be leveraged to really improve the ROI on their marketing and promotional efforts. In fact, these customer level data can provide meaningful insights into consumer online behaviours and purchase intent, and it can help Eureka Forbes identify the customer cluster that is most likely to make a purchase and increase the efficiency of their marketing campaigns at lower costs.

### Initiative

Eureka Forbes has been accumulating tremendous amount of customer data generated through their website, which is not being leveraged by the organization. Kashif and his team have been tasked with the responsibility to leverage this wealth of data and come up with appropriate models to predict which leads are most likely to be converted to sales and also understand what

type of customer behaviors and demographics are most desirable to better target them for door to door sales or other digital promotional offers. This opportunity will not only help Eureka Forbes to concentrate their marketing efforts on the most important and valuable clients but also will create provision for personalized pricing in a market where a huge part of the population is price sensitive.

## ANALYTICAL PROBLEM

In order to come up with a more efficient means of customer segmentation and targeting, Eureka Forbes has to use the available customer data to investigate whether a particular lead (website visitor) will ultimately make a purchase of any product or not and hence, this type of predictive modelling falls under the compass of classification problem. More precisely, it is a binomial classification problem whereby the aim is to try to predict a yes or no relative to each customer's purchase decision. Now, while it is a supervised learning problem where the analytics team can and will leverage huge amounts of existing customer data to train and test the model being developed, one particular reservation that needs to be considered is that the data is highly imbalanced.

### Analytical Solution

Find consumers who exhibit desired behavior and apply machine learning to model which other customers are most likely to have the same behavior. The end goal is to convert website visitors into prospective consumers either by making a purchase or sales lead generation actions (e.g. filling interest form or sharing contact details for an in-home demo).

### Supervised Learning Problem

It is going to be a supervised learning problem because there is data for converted consumer (2830 out of 709327 consumers converted) and we will use these to train our model and target the desired consumer clusters. We will use classification model because this algorithm will help identify consumer clusters and predict in which cluster they fall for new values. The x – variables will be the features from the dataset, and the y variables will be whether the visitor is converted within 7 days (Data: converted_in_7days).

## PRELIMINARY DATA ANALYSIS

An initial analysis of the customer data reveals that the variable of interest for Eureka Forbes is the "converted_in_7_days" column and Kashif and his team need to come up with a supervised machine learning, binomial classification model to predict the outcome of this column based on the numerous other variables/features available to them.

### Data Type

It is also worth mentioning here that the Eureka Forbes data set contains many variables/columns other than the target variable and will certainly require certain dimension reduction techniques (e.g. PCA) to avoid over-fitting the models. In the provided dataset (see appendix 1), we have 61 features (5 categorical, 56 numeric), 1 ID (client ID), and 1 target variable (converted_in_7days).

**Logistic Regression, Decision Trees & Random Forests**

Several classification models could be used to approach the Eureka Forbes case from logistic regression, decision trees to random forests and so on. While logistic regression may be the most basic method to utilize considering the simplicity of the model and its ability to be easily interpreted, due to the high number of variables in the dataset, models such as decision trees and random forests could give a higher accuracy and better set of predictions. The ultimate choice of a model will depend on the performance of the models on the test datasets for cross-validation.

## IMBALANCED DATA

The converted customers account for only 0.004% of the total dataset and this imbalanced class distribution will introduce bias and measurement errors as minority class tend to be treated as noise and ignored in classification algorithms like Decision Tree and Logistic Regression. Since the data is highly imbalanced with only 2,830 customers converted to sales within 7 days out of a total of 709,328 website visitors, which is less than 0.01%, training a supervised model based on this data may be difficult and requires some additional steps.

**Solution**

This imbalanced classification issue can be addressed by adopting resampling techniques such as increasing or decreasing the frequency of the minority class; in this case, Eureka Forbes can reduce the number of majority class used as data input. Up-sampling and down-sampling are both, techniques used to remedy imbalanced data; however, they are prone to omitting important information. There are several other approaches for handling imbalanced data that range from cluster-based over sampling, bagging-based techniques, boosting-based techniques, gradient tree boosting techniques and so on.

## PRE-PROCESSING DATA

First, we factorized all categorical and binary variables for the regression tree. Data was processed and imputed for missing values using multiple custom R functions. Below is a glimpse of the structure:

```
> str(Eureka)
'data.frame':   30000 obs. of  60 variables:
 $ DemoReqPg_CallClicks_evt_count            : int  0 0 0 0 0 0 0 0 0 0 ...
 $ air_purifier_page_top                     : int  0 0 0 0 0 0 20 0 0 0 ...
 $ bounces                                   : int  0 0 0 1 1 0 0 0 2 2 ...
 $ bounces_hist                              : int  1 NA 0 0 NA NA NA NA NA 0 ...
 $ checkout_page_top                         : int  0 0 0 0 0 0 0 0 0 0 ...
 $ contactus_top                             : int  0 0 0 0 0 0 0 0 0 0 ...
 $ converted_in_7days                        : Factor w/ 3 levels "0","1","2": 1 1 1 1 1 1 1 1 1 1 ...
 $ country                                   : Factor w/ 2 levels "d","i": 2 1 2 1 1 1 1 1 1 1 ...
 $ customer_service_amc_login_top            : int  0 0 0 0 0 0 0 0 0 0 ...
 $ customer_service_request_login_top        : int  0 0 0 0 0 0 51 0 0 0 ...
 $ demo_page_top                             : int  0 7 0 0 0 9 43 12 10 206 ...
 $ device                                    : Factor w/ 3 levels "desktop","mobile",..: 2 2 2 2 2 2 2 2 2 2 ...
 $ dsls                                      : int  0 6 1 0 0 0 56 23 2 5 ...
 $ fired_DemoReqPg_CallClicks_evt            : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
 $ fired_help_me_buy_evt                     : Factor w/ 2 levels "0","1": 1 1 1 2 1 1 2 1 1 1 ...
 $ fired_phone_clicks_evt                    : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
 $ goal4Completions                          : int  0 0 0 0 0 0 0 0 0 0 ...
 $ help_me_buy_evt_count                     : int  0 0 0 1 0 0 1 0 0 0 ...
 $ help_me_buy_evt_count_hist                : int  0 NA 0 1 NA NA NA NA NA 0 ...
 $ newUser                                   : Factor w/ 1 level "0": 1 1 1 1 1 1 1 1 1 1 ...
```

Rare categories were identified and combined with a custom function in R:

```
> table(Eureka$converted_in_7days)# check for rare categories

    0     1     2
29840   159     1
```

After processing our dataset, we used a 70/30 split to create Train and Test datasets.

```
# creating test data set
set.seed(77850)
inTrain <- createDataPartition(Eureka$converted_in_7days, p = .7,
                               list = FALSE)
training <- Eureka[ inTrain,]
testing <- Eureka[ -inTrain,]
```

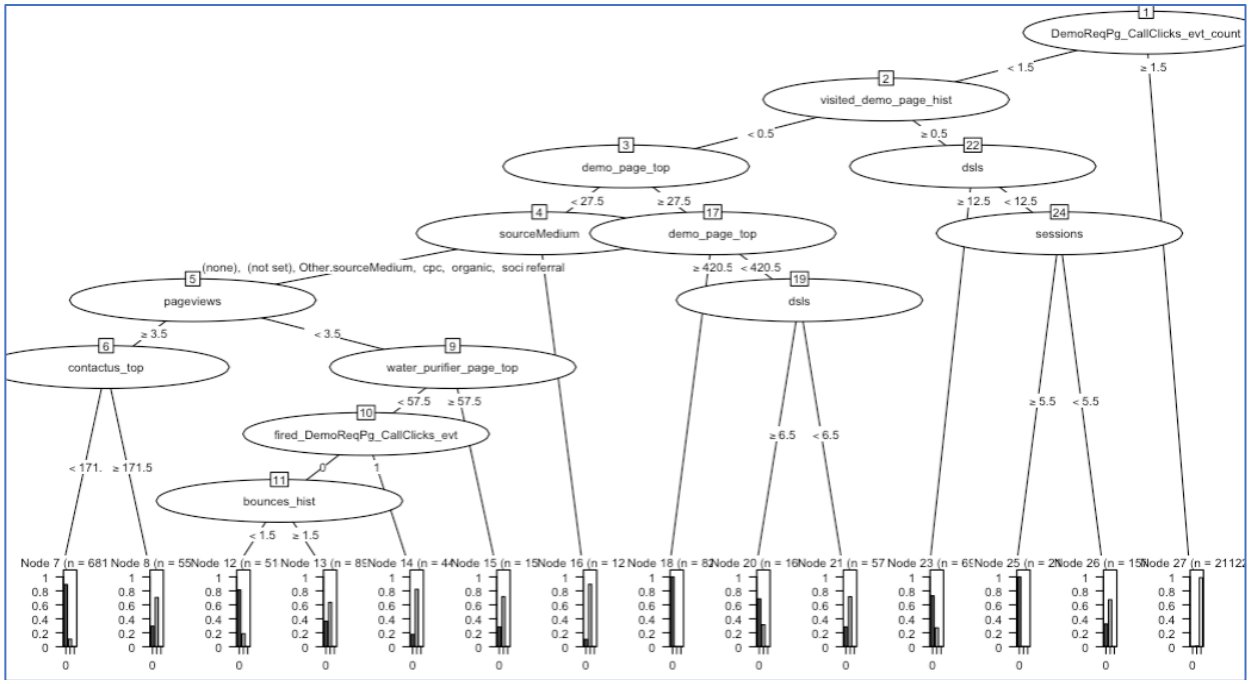
In order to incorporate imbalanced data issue, we also used down-sample and up-sample function in R on our training dataset using groupdata2 package. Both methods were used to assess which one performs better for our models.

## DECISION TREE MODEL

Decision trees split data on various categorical/continuous variables and selects the "split" to minimize entropy. Entropy is the measure of how much information we can obtain from the ambiguity in our dataset. The first split in our data is the feature with the highest informational gain. All in all, the tree helps us understand customer behavior and make predictions.

With the aim to split up this business decision into smaller clusters of similar features, we used both recursive partitioning (rpart) and conditional inference (ctree) on our training data set to develop our model. The rpart method recursively splits our data until preset termination criteria (e.g. complexity parameter (cp)) is achieved whereas, ctree method is a non-parametric version used to select variables.

Using **rpart** (appendix 3) on our "up sample" training data generated below decision tree after setting cp to 0.005. This value of cp was selected after performing cross validation. Navigating down to Node 27, for example, tells us conversion rate is high when DemoReqPg_CallClicks_evt_count is more than or equal to 1.5.




Using **ctree** (appendix 4), a few more variables become significant, and this can be used to further gain business insight. For example, navigating down to Node 29 tells us the following:

- When DemoReqPg_CallClicks_evt_count (which is also our most significant variable) is less than or equal to 0,
- Goal4Completions is less than or equal to 0,
- The visited_demo_page_hist is greater than 1
- Our customer retention rate is low

## RANDOM FOREST MODEL

Random forest (RF) model builds on the above model and utilizes a randomness technique called Bootstrapping to train multiple trees on random subsets of the data set. This method runs a combination of machine learning functions and computes an aggregate prediction. RF models are created by combining multiple decision trees (mentioned above), generating a decision model on each subset and averaging all predictions for final result.

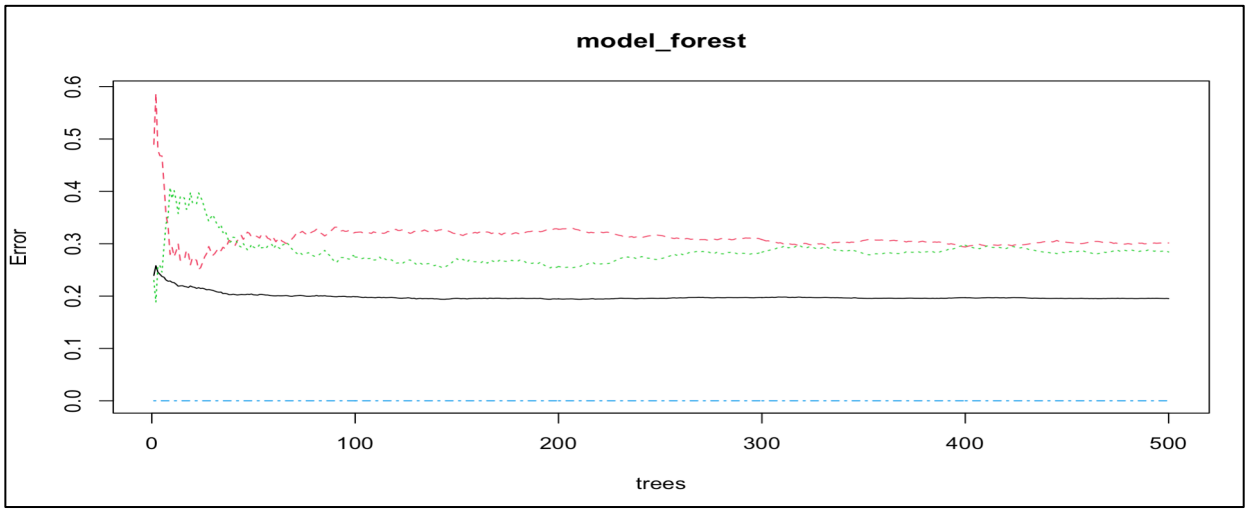
We build RF model for Eureka Forbes using randomForest package in R after preprocessing data (discussed above). This model generates many decisions trees with new samples, and we were able to observe how hyperparameter tuning adjusted how well the model performed. Hyperparameters essentially determine how the model will train itself with the given data.

The RF model was also used with our "up sample" training data and the following parameters were used to alter predictive power of the model:

- ntree – number of trees in forest
- ntry – number of random columns included
- nodesize – minimum number of datapoints in leaf node
- maxnodes – maximum number of leafs in a tree

```
# rf model
model_forest <- randomForest(converted_in_7days~ .,
                             data=train_up,
                             type="classification",
                             importance=TRUE,
                             ntree = 500,
                             mtry = 10,
                             nodesize = 10,
                             maxnodes = 10,
)
```

As you can observe below, after number of trees hits 300, the variance in error is no different and therefore, we selected 500 to ensure highest possible accuracy.



Using RF model reduces variances significantly as it trains multiple subsets and selects the "split" feature more randomly (lower bias). We used vapImpPlot in R to asses **feature importance** (see appendix 5) which tells us following variables are most important to our model based on mean decrease in accuracy and Gini Index (probability of variable being wrongly classified):

- demo_page_top
- demoReqPg_CallClicks_evt_count
- fired_DemoReqPg_CallClicks_evt

## DECISION TREE VS RANDOM FOREST MODEL

Decision tree models are very reliable when handling missing values and outliers within the data. Additionally, this model makes it relatively easier to visualize our model which is very informative in understanding how the model makes its ideal choice.

The one drawback of using decision tree models to predict data is that they are prone to overfitting data. However, there are "pruning" methods available in R to counter this by altering hyperparameters in the model. Tree depth is a hyperparameter in the rpart function that determines how many questions are asked before each predicted classification occurs. Moreover, adjusting "max_depth" (maximum depth of the tree) in rpart reduces overfitting but might develop a more biased model.

Random forest models build on the decision tree model, hence, are very powerful models that can handle multiple feature types and are able to run multiple trees in parallel. This model also has tendency to overfit, if hyperparameters are not used, but that is still significantly less than decision tree models. The randomness component in RF models also reduces the chance of developing uncorrelated trees.

Although, RF model work well with different types of data (including missing data), it is not as easy to interpret as the decision tree model. Many other methods are required in parallel to assess how the model performs such as plotting a ROC curve or evaluating feature importance (discussed in detail below).

## MODEL PERFORMANCE

**In the decision tree model,** we generated multiple rpart predictions using ANOVA and CART_cp methods and modifying hyperparameters (for ANOVA) and complexity parameter (cp) value (for CART_cp). The CP is used to determine the size of decision tree and the "root node error" values tells us the percentage of correctly sampled data in our tree nodes. We generated a few models using rpart and ctree but were not able to obtain a very high root node error (quite low value of 0.005247 for model below).

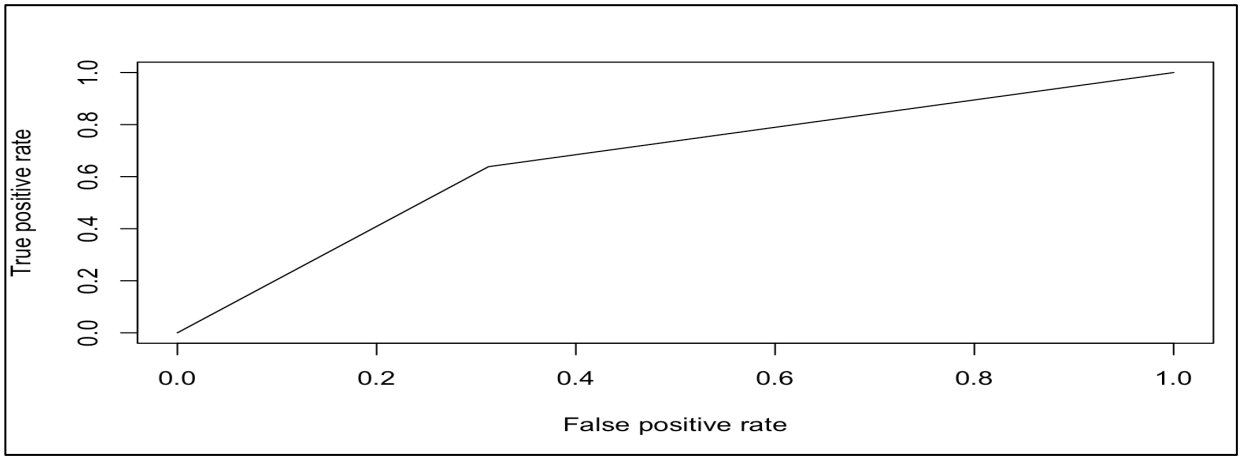
```
Classification tree:
rpart(formula = converted_in_7days ~ ., data = training, method = "class",
    control = CART_cp)

Variables actually used in tree construction:
[1] bounces_hist               goal4Completions           visited_checkout_page_hist
[4] visited_demo_page_hist     visited_storelocator_hist

Root node error: 551/105012 = 0.005247

n= 105012

          CP nsplit rel error xerror    xstd
1 0.00072595      0   1.00000      1 0.04249
2 0.00050000      5   0.99637      1 0.04249
```

After comparing multiple models with decision trees, we concluded **the random forest model performed better**. A confusion matrix assists compiling statistics in our model that will aid in understanding all possible outcomes. The confusion matrix below depicts our RF model does have false positive/negative values; however, its overall accuracy is quite decent (accuracy = 0.7044). This implies our model is making predictions accurately for 70% of the time.

```
Confusion Matrix and Statistics

                          Reference
Prediction                     0     1 Other.converted_in_7days
  0                        10517    29                        0
  1                         4403    49                        1
  Other.converted_in_7days     0     0                        0

Overall Statistics

               Accuracy : 0.7044
                 95% CI : (0.6971, 0.7117)
    No Information Rate : 0.9947
    P-Value [Acc > NIR] : 1

                  Kappa : 0.0117

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: 0 Class: 1 Class: Other.converted_in_7days
Sensitivity           0.70489 0.628205                        0.000e+00
Specificity           0.63291 0.704846                        1.000e+00
Pos Pred Value        0.99725 0.011004                              NaN
Neg Pred Value        0.01123 0.997250                        9.999e-01
Prevalence            0.99473 0.005200                        6.667e-05
Detection Rate        0.70118 0.003267                        0.000e+00
Detection Prevalence  0.70311 0.296886                        0.000e+00
Balanced Accuracy     0.66890 0.666525                        5.000e-01
```

ROC demonstrates a linear relationship and somewhat of a balance between model sensitivity (true positive rate) and specificity (1 minus false positive rate). The ROC has an area under curve (AUC) equal to 0.6630386:

```
> forest_AUC
[1] 0.6630386
```



## BUSINESS IMPLICATIONS

Based on the results for all models, we are able to notice that the amount of calls fired for demo requests (demoReqPg_CallClicks_evt_count) are significant in assessing whether visitors were converted in 7 days (lead generation). Moreover, it is worth considering promotional techniques to get more customers requesting demos. Both our decision tree and random forest model also give importance to demo_page_top variable which also supports our conclusion above.

The business implications of this are far reaching as Eureka Forbes now has a very good idea about where they should focus their limited resources of their door to door sales and distribution network to maximize potential sales. It is certainly not possible for the organization to send employees to every website visitors' doors for a demo, but seems arguably rational to send a salesman down to a site visitor's address who has spent considerable time on the 'DemoReq' page, or 'ContactUs' page whether they ultimately requested a demo or not.

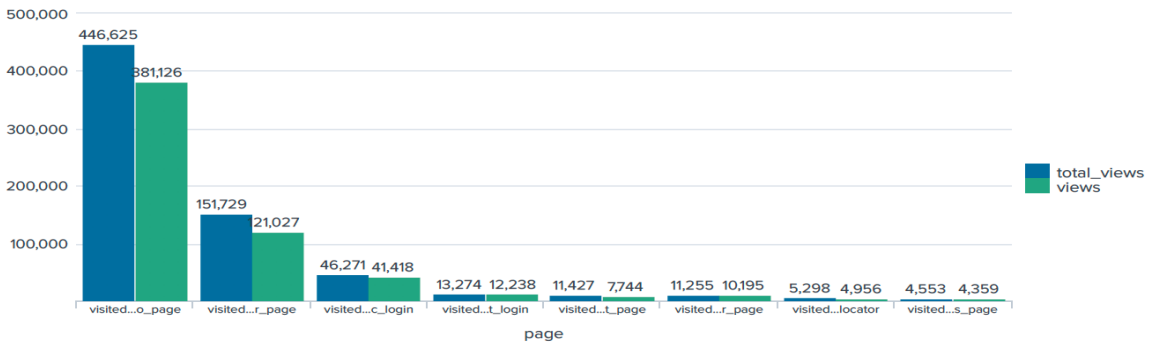
Furthermore, Eureka Forbes could come up with digital promotions and campaigns directed at this particular segment of the market considering their interest is relatively higher than the other customers and has a higher probability of being converted than the others. Focusing their digital efforts on this 2-3% of all the site visitors will not only allow them to reduce costs of their advertising budget but will also allow them to personalized pricing whereby they can offer additional discounts to customers who request for a demo.

## APPENDIX

### 1. DATA TYPES

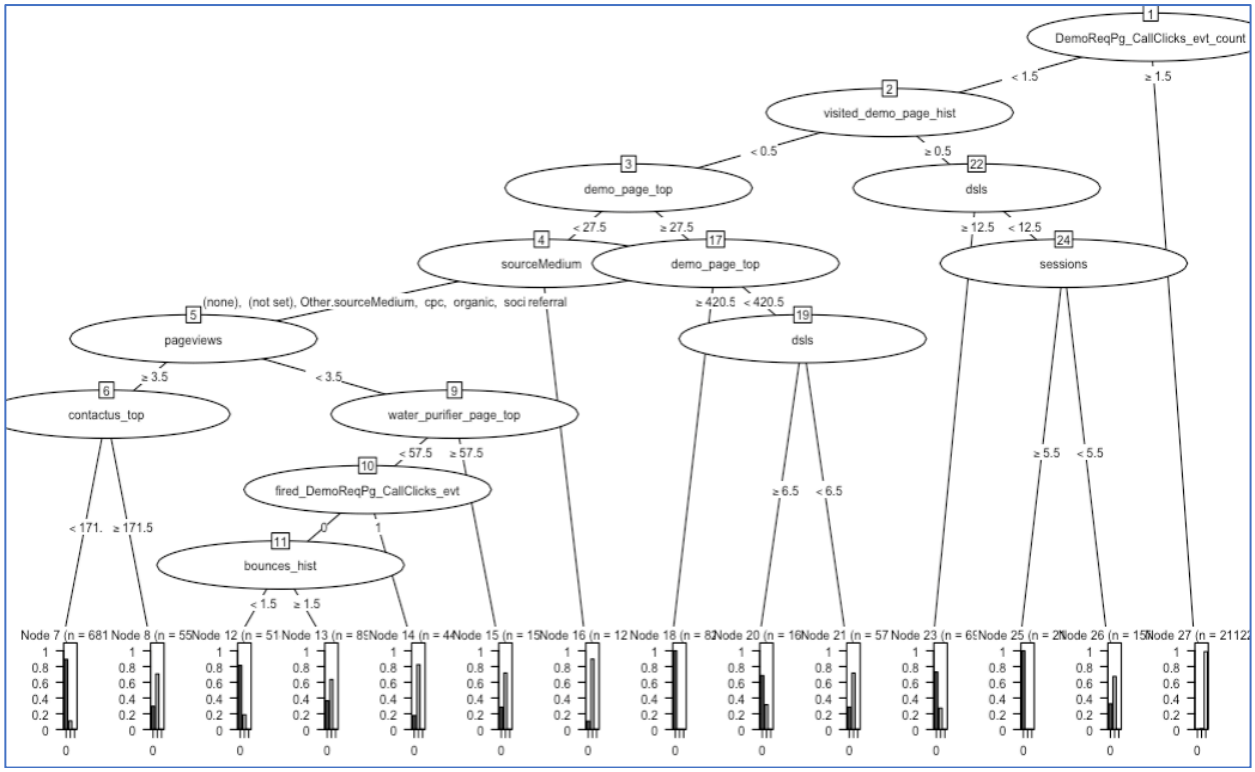
| Geo-demographic | Time Data | Channel Data | Visitor Action Data | History Data |
|---|---|---|---|---|
| ▪ country<br>▪ region | ▪ date<br>▪ air_purifier_page_top<br>▪ checkout_page_top<br>▪ contactus_top<br>▪ customer_service_amc_login_top<br>▪ customer_service_request_login_top<br>▪ demo_page_top<br>▪ Dsls<br>▪ Offer_page_top<br>▪ Security_solutions_page_top<br>▪ sessionDuration<br>▪ storelocator_top<br>▪ successbookdemo_top<br>▪ vacuum_cleaner_page_top<br>▪ water_purifier_page_top<br>▪ Dow | ▪ device<br>▪ sourceMedium<br>▪ Fired_DemoReqPg_CallClicks_evt<br>▪ Fired_help_me_buy_evt<br>▪ Fired_phone_clicks_evt<br>▪ Phone_clicks_evt_count | ▪ Pageviews<br>▪ Sessions<br>▪ Bounces<br>▪ Goal4Completions<br>▪ Help_me_buy_evt_count<br>▪ newUser<br>▪ paid<br>▪ visited_air_purifier_page<br>▪ visited_checkout_page<br>▪ visited_contactus<br>▪ visited_customer_service_amc_login<br>▪ visited_customer_service_request_login<br>▪ visited_demo_page<br>▪ visited_offer_page<br>▪ visited_security_solutions_page<br>▪ visited_storelocator<br>▪ visited_vacuun_cleaner_page<br>▪ visited_water_purifer_page | ▪ pageviews_hist<br>▪ sessions_hist<br>▪ bounces_hist<br>▪ paid_hist<br>▪ sessionDuration_hist<br>▪ visited_air_purifier_page_hist<br>▪ visited_checkout_page_hist<br>▪ visited_contactus_hist<br>▪ visited_customer_service_amc_login_hist<br>▪ visited_customer_service_request_login_hist<br>▪ visited_demo_page_hist<br>▪ visited_offer_page_hist<br>▪ visited_security_solutions_page_hist<br>▪ visited_storelocator_hist<br>▪ visited_vacuum_cleaner_page_hist<br>▪ visited_water_purifier_page_hist<br>▪ phone_clicks_evt_count_hist<br>▪ help_me_buy_evt_count_hist |

## 2. TOTAL PAGE VIEWS AND UNIQUE PAGE VIEWS

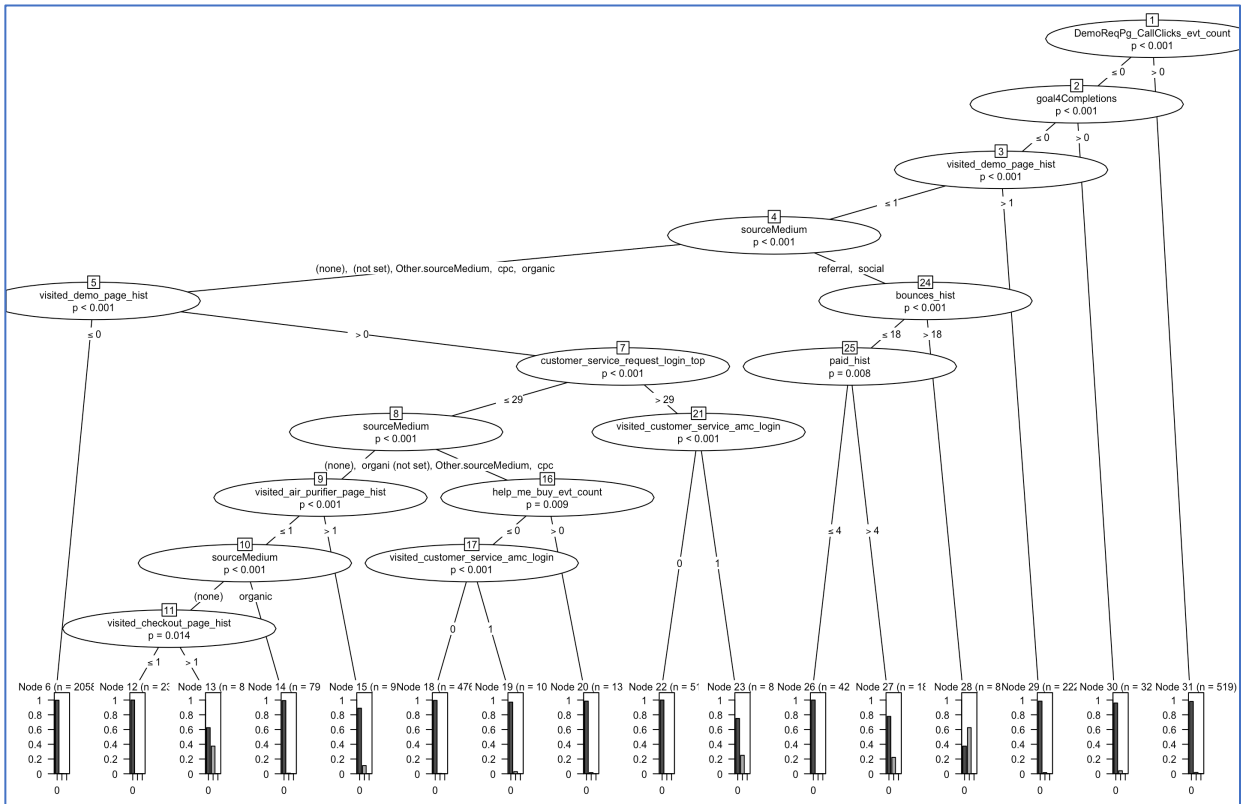



| page | total_views | views |
|---|---|---|
| visited_demo_page | 446625 | 381126 |
| visited_water_purifier_page | 151729 | 121027 |
| visited_customer_service_amc_login | 46271 | 41418 |
| visited_customer_service_request_login | 13274 | 12238 |
| visited_checkout_page | 11427 | 7744 |
| visited_air_purifier_page | 11255 | 10195 |
| visited_storelocator | 5298 | 4956 |
| visited_security_solutions_page | 4553 | 4359 |

## 3. RPART TREE



## 4. CTREE TREE




## 5. VARIABLE IMPORTANCE (RF MODEL)