TARGETED MARKETING

This is submitted as the final project report of the course CSE602: Machine Learning-I

by

Muhammad Amin Ghias
Uzair Zahidi

Presented to
Dr. Sajjad Haider
Professor
Department of Computer Science
School of Mathematics and Computer Science (SMCS)
Institute of Business Administration (IBA), Karachi

Spring Semester 2022
Institute of Business Administration (IBA), Karachi, Pakistan

<u>TARGETED MARKETING</u>

This is submitted as final project report of course CSE602: Machine Learning-I

by

Muhammad Amin Ghias
(ERP ID: 25366)

Uzair Zahidi
(ERP ID: 25374)

Submitted to:

<u>Dr. Sajjad Haider</u>

<u>Professor</u>

School of Mathematics and Computer Science (SMCS)
Institute of Business Administration (IBA), Karachi

Spring Semester 2022
Institute of Business Administration (IBA), Karachi, Pakistan

Dedication

This project report is dedicated to our parents, whose full support has enabled us to complete this project timely and effectively

Acknowledgement

First of all, thanks to Allah for allowing us to complete this project successfully and build an effective ML model for targeted marketing.

Furthermore, we would like to thank our teacher Dr. Sajjad Haider for extending his knowledge, experience and assistance throughout the course.

Also, we are thankful to our families and friends for their continuous support and encouragement.

Table of Contents

Table of Figures

Abstract

With a large amount of money spend on marketing these days Targeted Marketing is of utmost importance. This project will use the dataset and study the effect of advertisements on customers at individual level. For Instance, how many customers will return to buy a product due to treatment with advertisement, and how many customers are buying without the need of any promotion. Also, the Individuals who have been targeted by advertisements and have not returned that is marketing had no effect on them. Thus, we will build a predictive machine learning model that will predict the class of customers which will ultimately increase the ROI of marketing by Targeting only the persuadable class customers

Keywords:
Targeted, treatment, model, machine learning, predictive, class, persuadable

1. Introduction

In this modern time companies utilize a considerable amount of their revenue around (2% to 10%) on the marketing of their products. This is a big amount of money and efficient utilization of it is necessary for the companies. So, companies need marketing done with maximum output, which means their money is spent well with the best returns.
Companies try from oldest to newest marketing techniques with the main goal of getting efficient results. A considerable portion of the money spent on marketing and advertisement is wasted, the trouble is it is difficult to pinpoint the main reason and reduce the loss.
For this purpose, we need an approach and model of marketing that is efficient with minimum loss.

1.1 Business Problem

The normal marketing and advertisement models target audience, customers on a larger scale (nearly all customers) with this approach the effect of marketing at smaller, individual level cannot be studied and considered. Each person responds differently when exposed to advertisements and hence understanding the response of every single one is required, in this way we can find the reasons and customers which decrease the efficiency of our marketing. Hence a targeted marketing model is necessary for improved ROI.
For this targeted model we need to find customers on which there will be the most positive impact of advertisements. So, knowing about a user is most important and this knowledge will help us differentiate between users and tell which customers should be targeted.

1.2 Main Objective

The main purpose of our project is to decrease the overall cost of marketing and in this way increase the return of investment and efficiency. To achieve this, we will build a model to differentiate users into different classes and point out only which classes of customers should be targeted with advertisement. Ultimately, we will predict which users need targeted marketing treatment and which don't.

1.3 Background

The model is based on uplift modelling principle,

Uplift modelling is a technique that is used to estimate the effect of treatment of user at personal level by using different machine learning models. It's mostly used for targeted marketing and promotions.

It studies analyzes and predicts the outcome with and without the treatment. In order to understand the cause and effect of treatment different experiments and implementations are done.
Customers can be broadly classified into 4 classes as show in figure below
(a) Persuadable
(b) Sure Thing
(c) Do-Not-Disturb
(d) Lost Cause

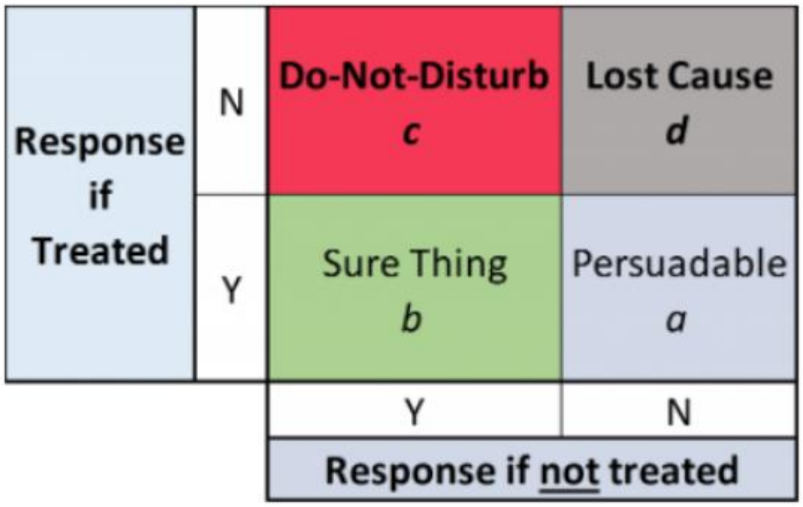


Figure 1-1 Uplift model matrix

(Lendave n.d.):

The users who are in class persuadable should be targeted only in order to get best results. Treatment on the customers in other class will only be a waste of money and time. For the class Do not disturb those customers should not be exposed as treatment will have a negative impact on them

The goal would be to find the persuadable customers and target them

Uplift modelling technique tells the effectiveness of the impact of treatment, while normal predictive modelling only predicts the result.

Thus, using this approach and technique of uplift model we can focus our effort and do targeted marketing only on these persuadable customers.

2. Data acquisition and understanding

The dataset used for this project is acquired from Kaggle:

https://www.kaggle.com/datasets/arashnic/uplift-modeling?select=criteo-uplift-v2.1.csv

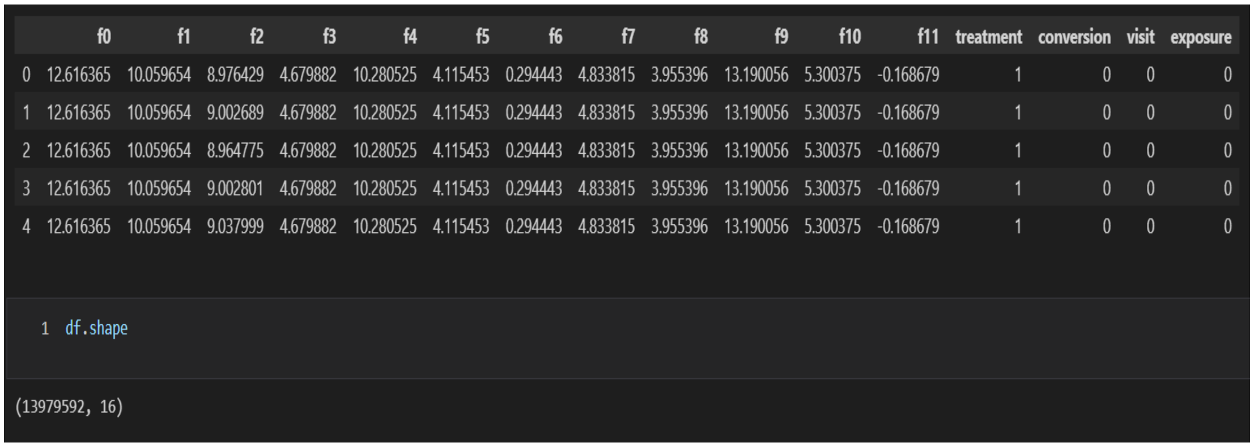
2.1 About the dataset

The data was obtained from Kaggle "Uplift Modeling, Marketing and Campaign Data" provided by AI lab of Criteo (The Criteo AI Lab is pioneering innovations in computational advertising it operates within the spectrum of two main areas: ML Engineering and ML Research) The dataset has 13M rows, each row gives information about 12 features of a user, 4 binary labels (1, 0) visits, conversions, treatment, exposure. If visit is 1 it means the user visited on the advertiser website during the test period (2 weeks). (Möbius-Kaggle n.d.)

For privacy reasons the feature names have been anonymized and their values randomly projected to keep predictive power while making it practically impossible to recover the original features or user context.

2.2 Dataset main characteristics

The data contains 13 million instances from a randomized control trial collected in two weeks. Each instance has 12 features that were anonymized plus a treatment variable and two target variables (visits and conversion). There is another extra variable called "exposure" which indicates whether the user was effectively exposed to the treatment



	f0	f1	f2	f3	f4	f5	f6	f7	f8	f9	f10	f11	treatment	conversion	visit	exposure
0	12.616365	10.059654	8.976429	4.679882	10.280525	4.115453	0.294443	4.833815	3.955396	13.190056	5.300375	-0.168679	1	0	0	0
1	12.616365	10.059654	9.002689	4.679882	10.280525	4.115453	0.294443	4.833815	3.955396	13.190056	5.300375	-0.168679	1	0	0	0
2	12.616365	10.059654	8.964775	4.679882	10.280525	4.115453	0.294443	4.833815	3.955396	13.190056	5.300375	-0.168679	1	0	0	0
3	12.616365	10.059654	9.002801	4.679882	10.280525	4.115453	0.294443	4.833815	3.955396	13.190056	5.300375	-0.168679	1	0	0	0
4	12.616365	10.059654	9.037999	4.679882	10.280525	4.115453	0.294443	4.833815	3.955396	13.190056	5.300375	-0.168679	1	0	0	0

```
df.shape
```

```
(13979592, 16)
```

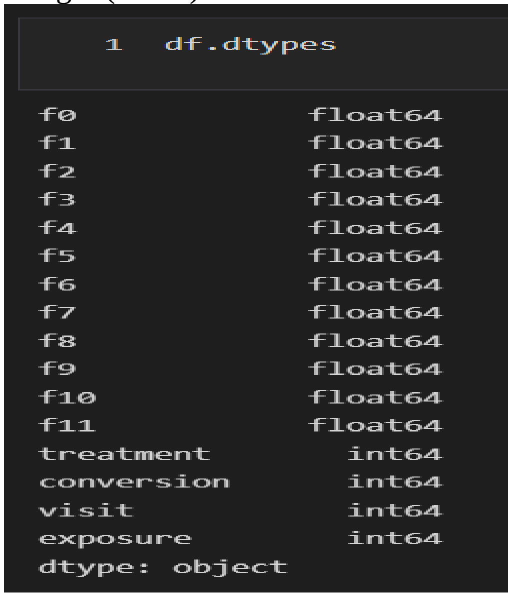
Figure 2-1 Original dataset

2.2.1 Dataset Size

The size of the csv is **3.17 GB**
The no. of rows = 13979592
Number of columns = 16

2.2.2 Data types

The features f0 to f11 are of float type while the rest
Treatment is integer (0 or 1)
Exposure is integer (0 or 1)
Conversion is integer (0 or 1)
Visit is integer (0 or 1)



```
df.dtypes
```

```
f0                float64
f1                float64
f2                float64
f3                float64
f4                float64
f5                float64
f6                float64
f7                float64
f8                float64
f9                float64
f10               float64
f11               float64
treatment           int64
conversion          int64
visit               int64
exposure            int64
dtype: object
```

Figure 2-2 Data type of columns

2.3 Data Exploration

To begin with any analysis, we should first explore the dataset look for any outliers, data distribution, and null values etc.

2.3.1 Outliers identification

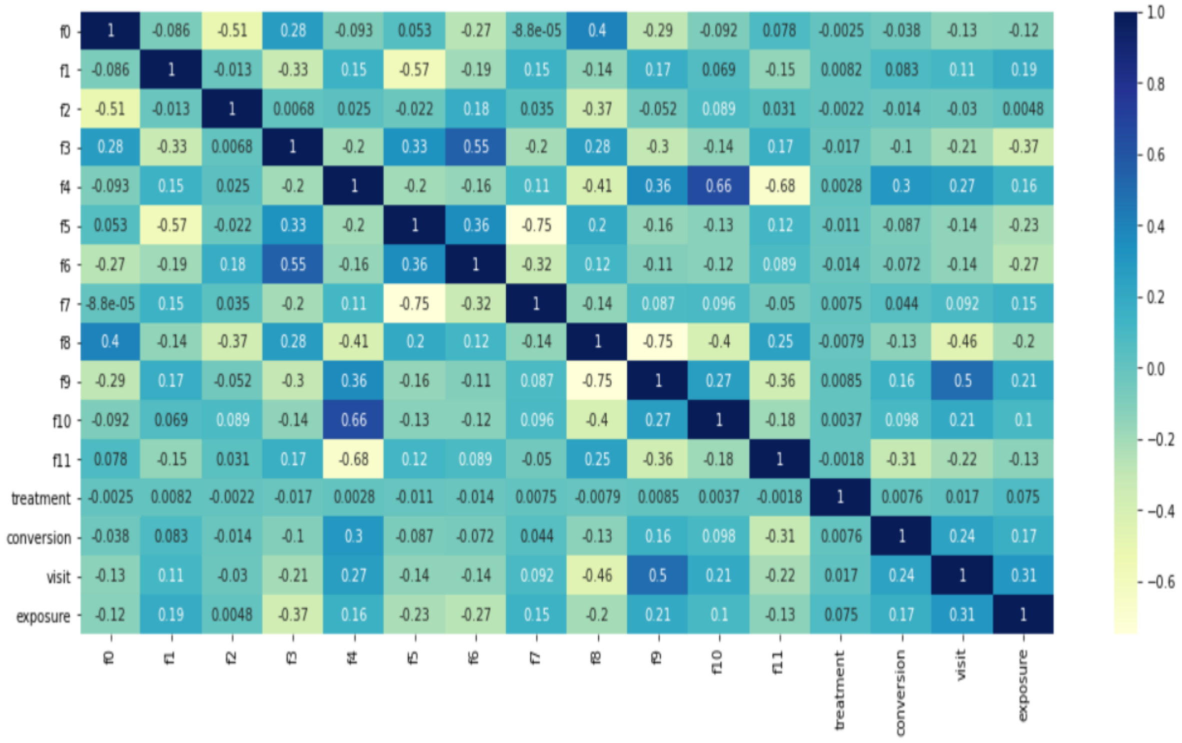
The dataset has 12 features, a histogram of all features from f0 to f12 was created to see the distribution of these features and to identify any outliers. No outlier was found in any feature.

2.3.2 Null values identification

Looked for any null values in all columns in our dataset and didn’t ffindany

2.3.3 Making correlation matrix

Correlation matrix is created to find correlation between features.



In analysis from table based on correlation with visit following features had least correlation <0.1

f2	0.030331
f7	0.091858

2.3.4 Data cleaning and filtering

The dataset didn’t have any outliers and null values, so no cleaning was required. The dataset is filtered based on correlation and variance and correlation, this filtered dataset and unfiltered both were used in our ML model to check If filtering was required or not.

(Note: Later on, it was found out that filtering did not improve results hence and all features were be used)

2.3.5 Exploring target variable

The histogram of the target variable (visit) shows that there is a class imbalance between 0 (customers who did not visit) and 1 (customers who visited) a greater number of 0 as compared to 1

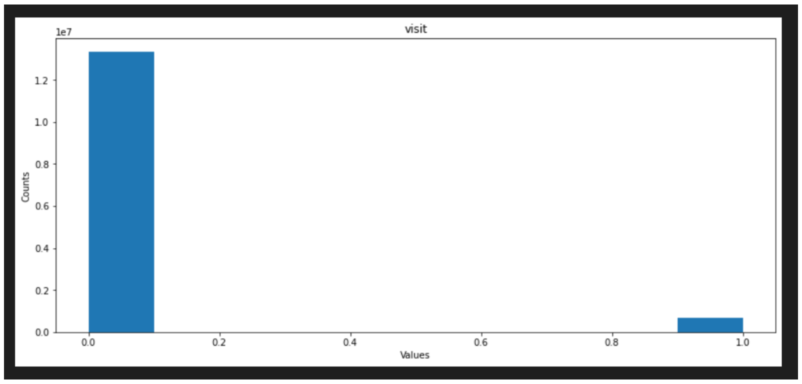
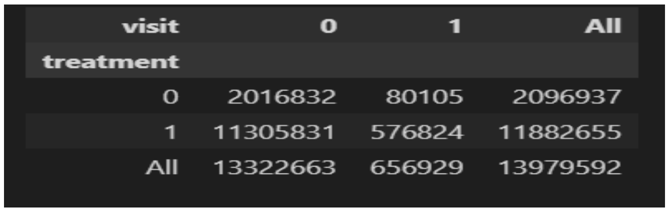


Figure 2-3 Histogram of visits

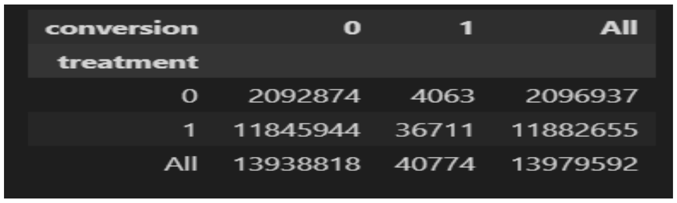
The relation of visit with other classes exposure, treatment and conversion also found



visit	0	1	All
treatment			
0	2016832	80105	2096937
1	11305831	576824	11882655
All	13322663	656929	13979592

Figure 2-4 Visit-Treatment Relationship

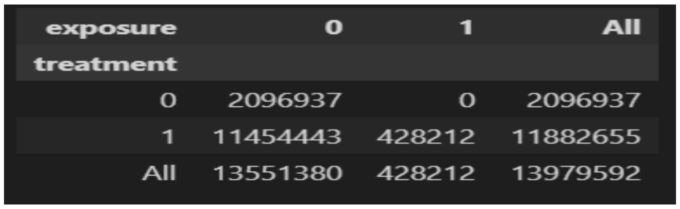
The table above shows the relationship between treatment and visit, it can be seen here that there are users who have visited even with treatment 0



conversion	0	1	All
treatment			
0	2092874	4063	2096937
1	11845944	36711	11882655
All	13938818	40774	13979592

Figure 2-5 conversion-treatment relationship

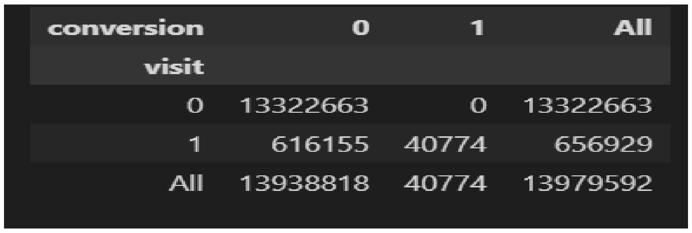
Above table shows relationship between conversion and treatment, it shows that even with 0 treatment some users have conversion 1



exposure	0	1	All
treatment			
0	2096937	0	2096937
1	11454443	428212	11882655
All	13551380	428212	13979592

Figure 2-6 exposure-treatment relationship

The above table shows that when treatment is 0 the exposure is always zero

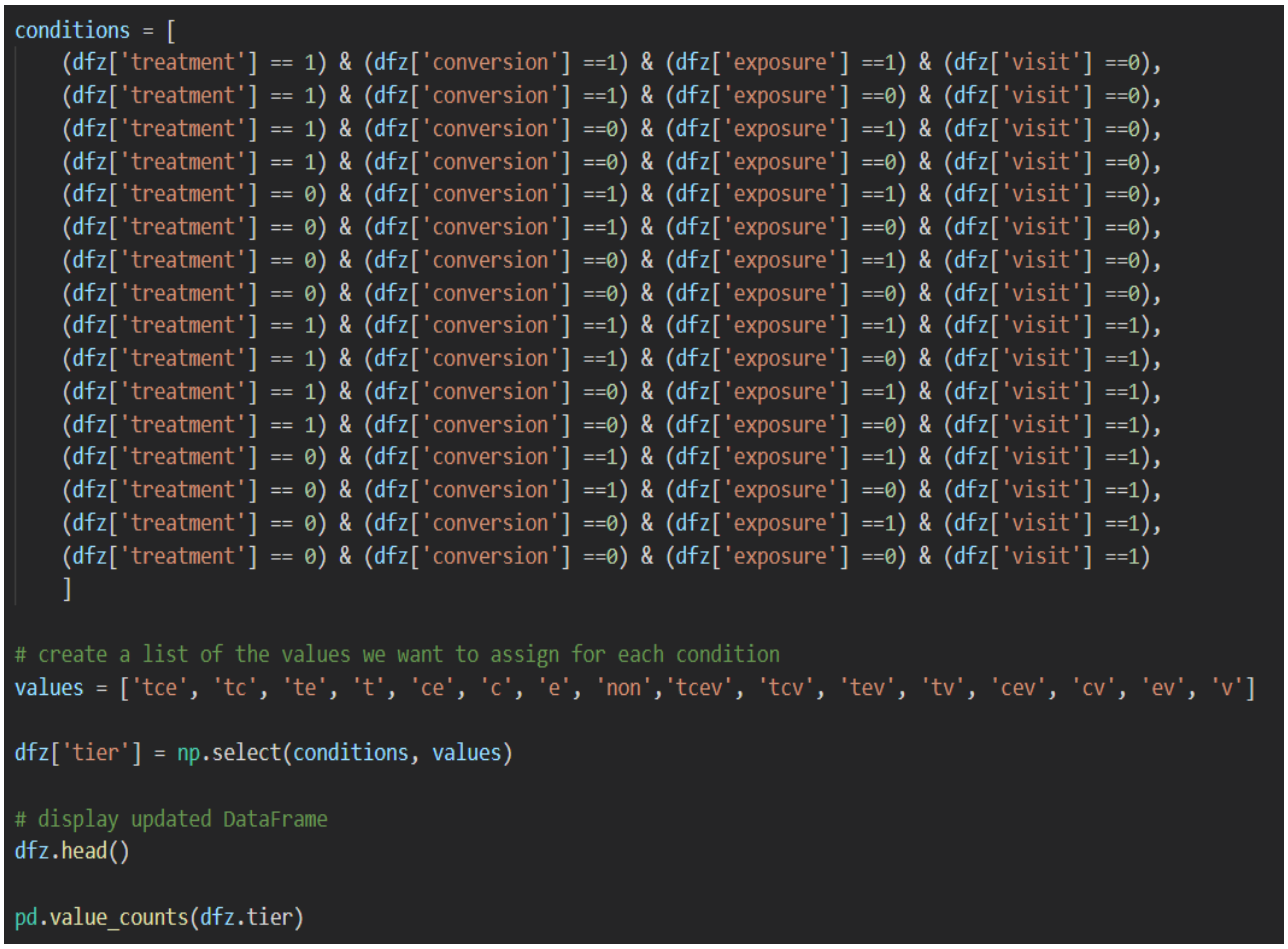


conversion	0	1	All
visit			
0	13322663	0	13322663
1	616155	40774	656929
All	13938818	40774	13979592

Figure 2-7 conversion-visit Relationship

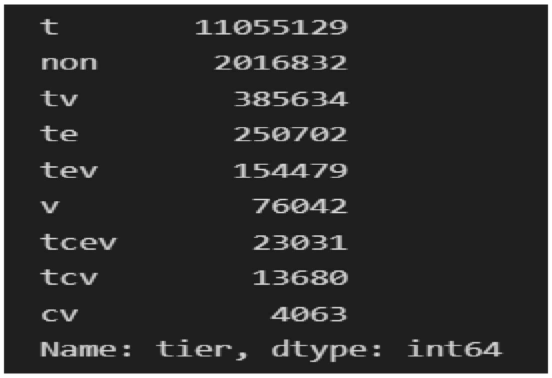
The above table shows when visit is 0 the conversion is also zero

We can further study the relationship between the 4 treatment, visit, conversion and exposure in our dataset to get more knowledge about our data. We have formulated the 16 possible conditions of these 4 and then found the facts, the conditions which are shown below



```
conditions = [
    (dfz['treatment'] == 1) & (dfz['conversion'] ==1) & (dfz['exposure'] ==1) & (dfz['visit'] ==0),
    (dfz['treatment'] == 1) & (dfz['conversion'] ==1) & (dfz['exposure'] ==0) & (dfz['visit'] ==0),
    (dfz['treatment'] == 1) & (dfz['conversion'] ==0) & (dfz['exposure'] ==1) & (dfz['visit'] ==0),
    (dfz['treatment'] == 1) & (dfz['conversion'] ==0) & (dfz['exposure'] ==0) & (dfz['visit'] ==0),
    (dfz['treatment'] == 0) & (dfz['conversion'] ==1) & (dfz['exposure'] ==1) & (dfz['visit'] ==0),
    (dfz['treatment'] == 0) & (dfz['conversion'] ==1) & (dfz['exposure'] ==0) & (dfz['visit'] ==0),
    (dfz['treatment'] == 0) & (dfz['conversion'] ==0) & (dfz['exposure'] ==1) & (dfz['visit'] ==0),
    (dfz['treatment'] == 0) & (dfz['conversion'] ==0) & (dfz['exposure'] ==0) & (dfz['visit'] ==0),
    (dfz['treatment'] == 1) & (dfz['conversion'] ==1) & (dfz['exposure'] ==1) & (dfz['visit'] ==1),
    (dfz['treatment'] == 1) & (dfz['conversion'] ==1) & (dfz['exposure'] ==0) & (dfz['visit'] ==1),
    (dfz['treatment'] == 1) & (dfz['conversion'] ==0) & (dfz['exposure'] ==1) & (dfz['visit'] ==1),
    (dfz['treatment'] == 1) & (dfz['conversion'] ==0) & (dfz['exposure'] ==0) & (dfz['visit'] ==1),
    (dfz['treatment'] == 0) & (dfz['conversion'] ==1) & (dfz['exposure'] ==1) & (dfz['visit'] ==1),
    (dfz['treatment'] == 0) & (dfz['conversion'] ==1) & (dfz['exposure'] ==0) & (dfz['visit'] ==1),
    (dfz['treatment'] == 0) & (dfz['conversion'] ==0) & (dfz['exposure'] ==1) & (dfz['visit'] ==1),
    (dfz['treatment'] == 0) & (dfz['conversion'] ==0) & (dfz['exposure'] ==0) & (dfz['visit'] ==1)
    ]

# create a list of the values we want to assign for each condition
values = ['tce', 'tc', 'te', 't', 'ce', 'c', 'e', 'non','tcev', 'tcv', 'tev', 'tv', 'cev', 'cv', 'ev', 'v']

dfz['tier'] = np.select(conditions, values)

# display updated DataFrame
dfz.head()

pd.value_counts(dfz.tier)
```

Figure 2-8 treatment-conversion-visit-exposure Relationship



```
t         11055129
non        2016832
tv          385634
te          250702
tev         154479
v            76042
tcev         23031
tcv          13680
cv            4063
Name: tier, dtype: int64
```

Figure 2-9 count for Retains of TEVC

Here
t=Treatment is 1 rest 0
non = all 4 variables 0
tv= treatment and visit are 1 rest 0
te = treatment and exposure are 1 rest 0
cv = conversion and visit are 1 rest 0

Following things can be highlighted from figure above:

- Our dataset has nearly 76042 customers who visit without any treatment that is when visit is 1 only.
- There are 385634 users who have treatment and visit 1, that is how many have visited due to treatment
- There are 11055129 users who have been given treatment 1 but have no conversion and visits
- There are 4063 users with visits and conversion 1

From the observations it is visible that there are some customers on which:

- treatment had no effect
- Where customers visit irrespective of treatment
- Some customers who may have visited due to treatment

Thus, we can see that the effect is different with treatment and entirely different effect without treatment.
We will make our approach by dividing the dataset in 2 parts

1. Dataset where treatment is 0
2. Dataset where treatment is 1

2.3.6 Choosing the evaluation matrix

As there is class imbalance in our problem therefore, we will use ROC AUC as the evaluation matrix in this project.

3. Approach for project

After data exploration and analysis in order to achieve our goal, we will use the following:

3.1 Part-1

Step 1. We will break dataset into 2 on basis of treatment

1. Dataset where customers have not been treated that is treatment = 0
2. Dataset where customers have been treated that is treatment =1

3.1.1 Model-1

Step 2. We will train dataset on treatment 0 and predict on dataset of treatment 1

Step 3. Break dataset of treatment 1 on basis of predicted values into 4 smaller datasets of each class for treatment 1 dataset the 4 classes:
Sure thing (Who will visit irrespective of treatment or not)
Don't Disturb (Who will change from visiting to not visiting when treated with advertisement, that is negative effect of treatment)
Lost Cause (Who will not visit irrespective of treatment or not)
Persuadables (Who will visit when they are targeted)

3.1.2 Model-2

Step 4. Now train on treatment 1 dataset and predict on dataset of treatment 0

Step 5. Break dataset of treatment 0 on basis of predicted values into 4 smaller datasets of each class for treatment 0

Step 6. Combine all the datasets with added class feature now our dataset has 4 classes

3.2 Part-2

3.2.1 Final Model

Step 7. Now finally we will train a final-model on the new dataset with class column to predict the class of customer.

4. ML Models used in our project

We used different models to find which model gave the best results on our evaluation metric of ROC AUC

4.1 Logistic Regression

As our final target variable is visit which has two classes (0 and 1) we use logistic regression.
We used LogisticRegression() module of sklearn

4.2 Decision Tree

Descission Tree module of sklearn was used
DecisionTreeClassifier(max_depth=5)
Parameters modified: max_depth

4.3 Random Forrest

Random Forrest module of sklearn was used
RandomForestClassifier(max_depth=5,n_estimators=50, verbose=2)
Parameters modified: max_depth, n_estimators, max_features

4.4 Gradient Boosting

Gradient boosting module of sklearn was used
GradientBoostingClassifier(max_depth=2,n_estimators=50, verbose=2)
Parameters modified: max_depth, n_estimators

4.5 Naive Bayes Gaussian

Naive bayes module of sklearn was used

4.6 KNN

KNN module of sklearn is used
KNeighborsClassifier(n_neighbors=5)
Parameters modified: n_neighbours

4.7 Scaled KNN

Using sklearn module of KNN on scaled dataset
Parameters modified: n_neighbours

4.8 Hist-gradient boosting

As the size of dataset is very large, we have used hist-gradient boosting
HistGradient boosting module of skelarn was used
HistGradientBoostingClassifier(max_depth=2, max_iter=50, verbose=10)
Parameters modified: max_depth, max_iter

5. Implementation of our Approach part-1

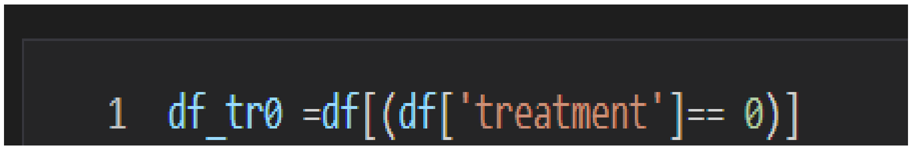
Implementing the steps mentioned above on our dataset

5.1 Splitting dataset into 2

Our dataset is very large and contains 13 million rows so it must be split. As per out data exploration and analysis we will split our dataset based on treatment.

5.1.1 Making dataset for treatment = 0

From our main dataframe we formed a dataset where the value of treatment =0 and made a new dataset



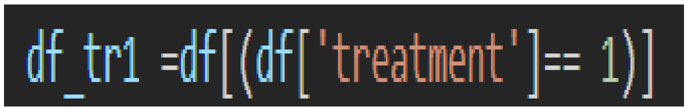
```
df_tr0 =df[(df['treatment']== 0)]
```

Figure 5-1 Making dataset for treatment=0

The DatatFrame has now
Rows = 2096937 (these are all customers with treatment=0)
Columns = 16

5.1.2 Making dataset for treatment = 1

From our main dataframe we formed a dataset where the value of treatment =1 and made a new dataset



```
df_tr1 =df[(df['treatment']== 1)]
```

Figure 5-2 Making dataset for treatment=1

The DatatFrame has now
Rows = 11882655 (these are all customers with treatment=1)
Columns = 16

5.2 Feature Importance

The important features have been found using random forest classifier as show in figure below

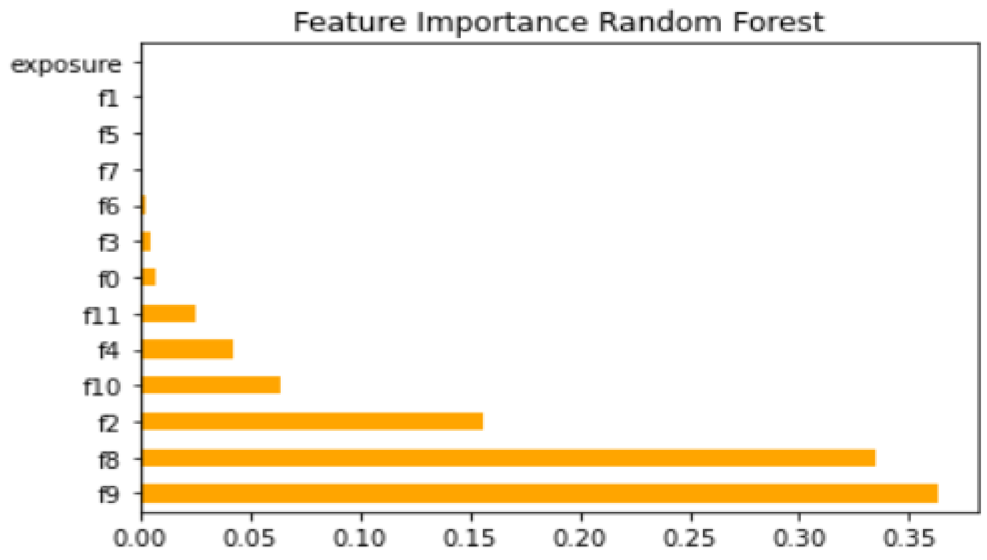


Figure 5-3 Important Features

This shows that feature f9 and f8 are most important features

5.3 Working on dataset for treatment = 0 (Model-1)

As per the steps mentioned in our approach, we will start our ML working with DatatFrame of treatment=0 and apply all our models to find the winner model

Using the ML models on train-test split of treatment=0 DatatFrame, we got the following results of ROC AUC:

Logistic regression :	0.9240255402388866
Decision Tree :	0.9412635366793343
Random Forest :	0.9419635440768807
Graident Boosting :	0.9407031423543095
k-NN :	0.81705752459023
Scaled k-NN :	0.8325238834033942

5.3.1 1st best model

It can be seen that random forest gave the first best result of ROC AUC

5.3.2 2nd best model

Decision tree is the 2nd best model
The parameters are max_depth=5

5.3.3 3rd best model

Gradient boosting is the 3rd best model
Best parameters: max-dept =2, n_estimators=50

5.3.4 Winner model-1 and its parameters

Now in order to find the parameters for random forest we used GridSearchCV partially and experimented/iterated more with different values of parameters

From Gridsearchcv we found the best parameters
{'max_features': 7, 'n_estimators': 300}

Using iterations/experiments we found the best parameters as
max_depth = 10

Best features
RandomForestClassifier(max_depth=10,n_estimators=800,verbose=2, max_features=7)
AUC score
Random Forest: 0.9471275509513373

5.3.5 Training best model-1 on full dataset for treatment =0 and applying on dataset for treatment =1

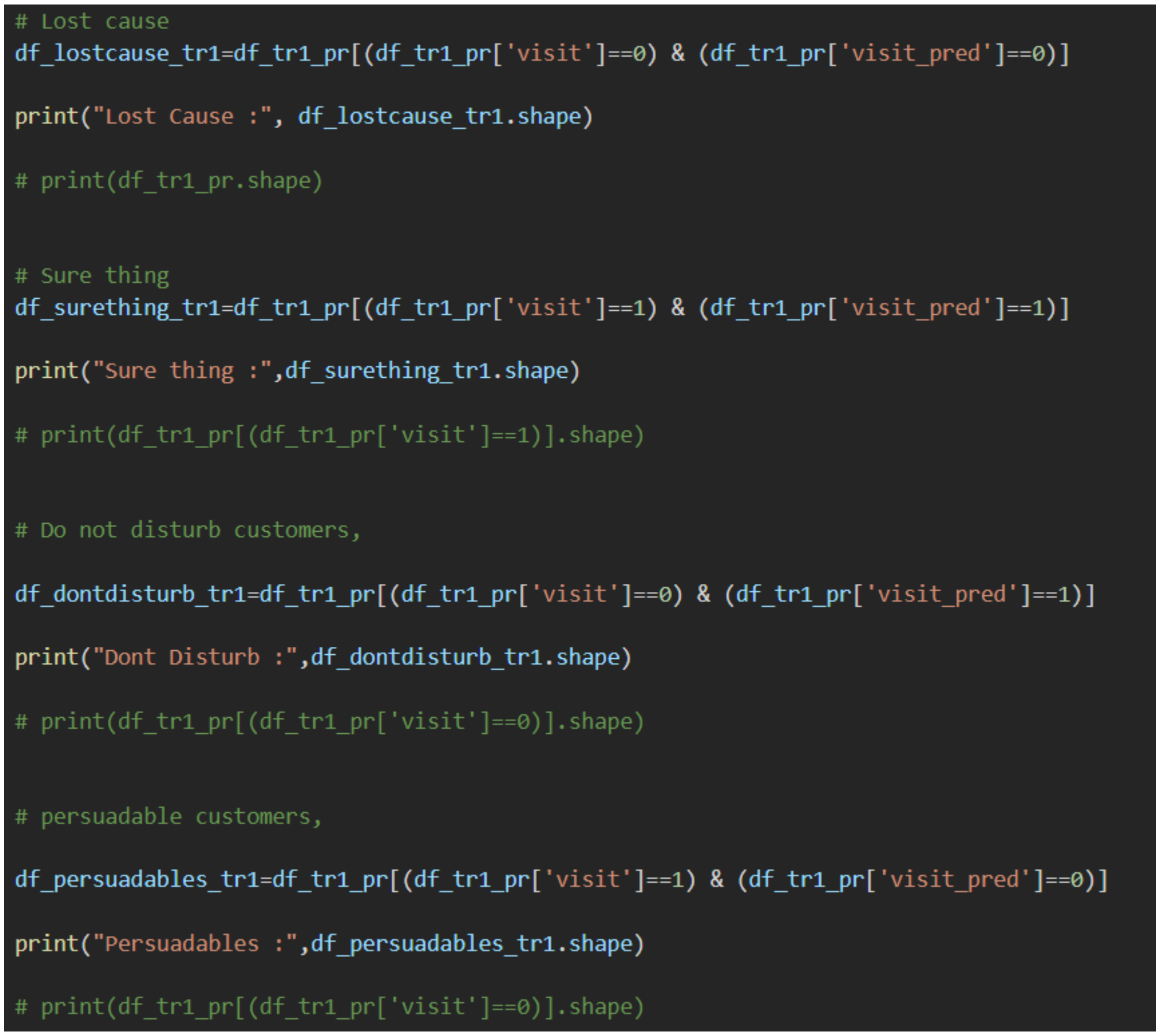
Now we trained the best model-1 on complete dataset of treatment=0, then this fitted model was applied on treatment=1 dataset to predict the visit

On the test data that is treatment=1 dataset Model-1 performed as follows
Model-1
RandomForestClassifier(max_depth=10,n_estimators=800, verbose=2, max_features=7)
with AUC = 0.9448954418150723

In this way we got value of predicted visit, which have been added as a new column of predicted visits is added in dataset of treatment=1
The dataset now has visit and predicted visit columns

5.3.6 Making the 4 datasets of classes for treatment=1 dataset

Now based on certain conditions with visit and predicted visit we have classified users into different class as shown in figure below



```
# Lost cause
df_lostcause_tr1=df_tr1_pr[(df_tr1_pr['visit']==0) & (df_tr1_pr['visit_pred']==0)]

print("Lost Cause :", df_lostcause_tr1.shape)

# print(df_tr1_pr.shape)


# Sure thing
df_surething_tr1=df_tr1_pr[(df_tr1_pr['visit']==1) & (df_tr1_pr['visit_pred']==1)]

print("Sure thing :",df_surething_tr1.shape)

# print(df_tr1_pr[(df_tr1_pr['visit']==1)].shape)


# Do not disturb customers,

df_dontdisturb_tr1=df_tr1_pr[(df_tr1_pr['visit']==0) & (df_tr1_pr['visit_pred']==1)]

print("Dont Disturb :",df_dontdisturb_tr1.shape)

# print(df_tr1_pr[(df_tr1_pr['visit']==0)].shape)


# persuadable customers,

df_persuadables_tr1=df_tr1_pr[(df_tr1_pr['visit']==1) & (df_tr1_pr['visit_pred']==0)]

print("Persuadables :",df_persuadables_tr1.shape)

# print(df_tr1_pr[(df_tr1_pr['visit']==0)].shape)
```

Figure 5-4 Condition for making customer classes on treatment=1 dataset

Note here predicted visit has been trained on treatment=0 dataset, so the result predicted are when treatment is 0 (that is no treatment on users)

Lost cause

If the visit =0 in treatment 1 dataset and predicted visit =0 (treatment 0).
This shows that these users even if they have treatment or not, they will not visit and hence they are a lost cause and should not be treated

Sure Thing

If the visit =1 in treatment 1 dataset and predicted visit =1 (treatment 0).
This shows that these users even if they have treatment or not, they will visit and hence they are a Sure thing and should not be treated

Do not Disturb

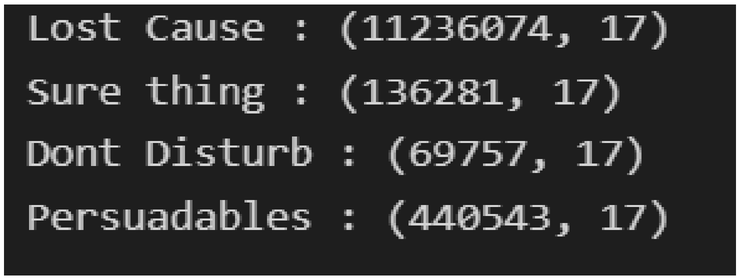
If the visit =0 in treatment 1 dataset and predicted visit =1 (treatment 0)
this shows that these users when they are treated, they will not visit and if they are not treated left alone, they will visit hence treatment has a negative impact on these users and should not be disturbed and should never be treated.

Persuadable

If the visit =1 in treatment 1 dataset and predicted visit =0 (treatment 0).
This shows that these users when they are treated, they will visit and if they are not treated left alone, they will not visit hence treatment has a positive impact on these users and should always be treated.

We must target persuadable only

For treatment=1 dataset the size of the 4 datasets of each user class is shown below



```
Lost Cause : (11236074, 17)
Sure thing : (136281, 17)
Dont Disturb : (69757, 17)
Persuadables : (440543, 17)
```

Figure 5-5 treatment=1 user classes dataset

5.4 Working on dataset for treatment = 1 (Model-2)

Now we will work on dataset with treatment=1 (excluding the Sure thing users)

Trying to find best models for treatment=1 dataset

5.4.1 1st best model

Gradient Boosting is the 1st best model

5.4.2 2nd best model

Random forest is 2nd best model
RandomForestClassifier(max_depth=10,n_estimators=800, verbose=10, max_features=7, n_jobs=-1)
AUC 0.9276242580523582

5.4.3 Winner model-2 and its parameters

The best parameters for gradient boosting were found to be
GradientBoostingClassifier(max_depth=2, n_estimators=200)

5.4.4 Training best model-2 on full dataset for treatment =1 and applying on dataset for treatment =0

Now we trained the best model-2 on complete dataset of treatment=1, then this fitted model was applied on treatment=0 dataset to predict the visit

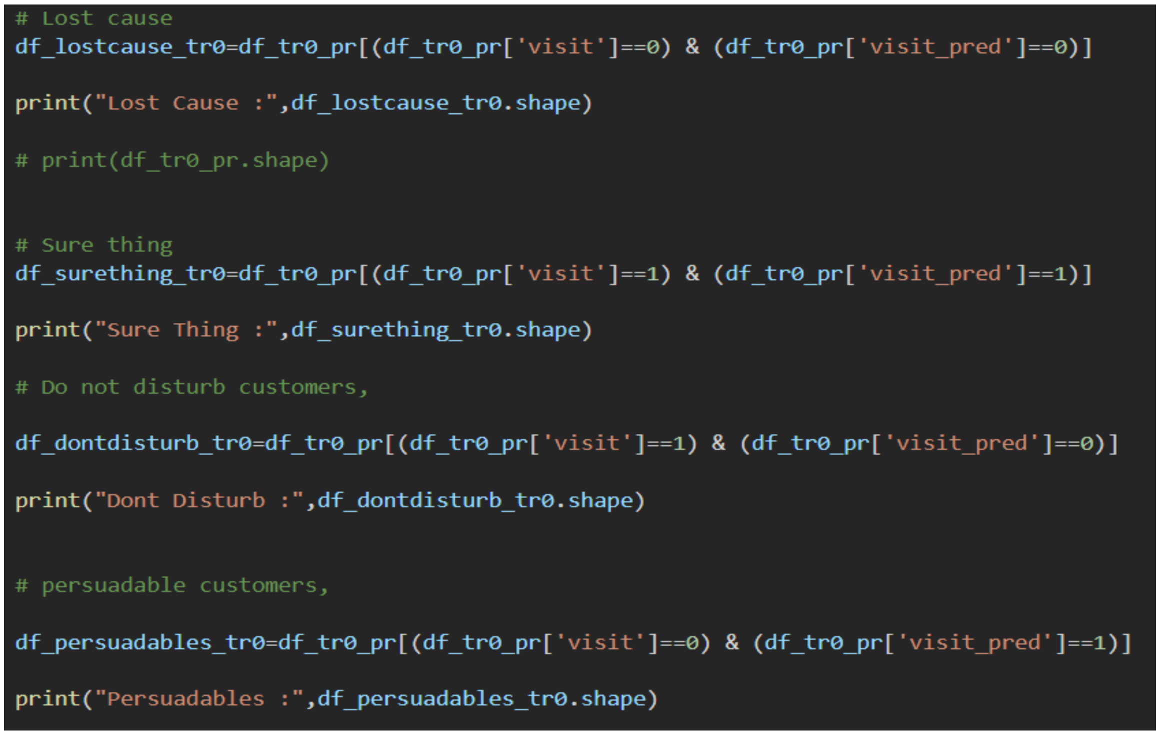
On the test data that is treatment=0 dataset Model-2 performed as follows

Model-2

GradientBoostingClassifier (max_depth=2, n_estimators=200)
AUC 0.9327316640956339

5.4.5 Making the 4 datasets of classes for treatment=0 dataset

Now based on certain conditions with visit and predicted visit we have classified users into different class as shown in figure below



```
# Lost cause
df_lostcause_tr0=df_tr0_pr[(df_tr0_pr['visit']==0) & (df_tr0_pr['visit_pred']==0)]

print("Lost Cause :",df_lostcause_tr0.shape)

# print(df_tr0_pr.shape)


# Sure thing
df_surething_tr0=df_tr0_pr[(df_tr0_pr['visit']==1) & (df_tr0_pr['visit_pred']==1)]

print("Sure Thing :",df_surething_tr0.shape)

# Do not disturb customers,

df_dontdisturb_tr0=df_tr0_pr[(df_tr0_pr['visit']==1) & (df_tr0_pr['visit_pred']==0)]

print("Dont Disturb :",df_dontdisturb_tr0.shape)


# persuadable customers,

df_persuadables_tr0=df_tr0_pr[(df_tr0_pr['visit']==0) & (df_tr0_pr['visit_pred']==1)]

print("Persuadables :",df_persuadables_tr0.shape)
```

Figure 5-6 Condition for making customer classes on treatment=0 dataset

Note here predicted visit has been trained on treatment=1 dataset, so the result predicted are when treatment is 1 (that is with treatment on users)

Lost cause

If the visit =0 in treatment 0 dataset and predicted visit =0 (treatment 1)
this shows that these users even if they have treatment or not, they will not visit and hence they are a lost cause and should not be treated

Sure Thing

If the visit =1 in treatment 0 dataset and predicted visit =1 (treatment 1)
this shows that these users even if they have treatment or not, they will visit and hence they are a Sure thing and should not be treated

Do not Disturb

If the visit =1 in treatment 0 dataset and predicted visit =0 (treatment 1)
this shows that these users when they are treated, they will not visit and if they are not treated left alone, they will visit hence treatment has a negative impact on these users and should not be disturbed and should never be treated.

Persuadable

If the visit =0 in treatment 0 dataset and predicted visit =1 (treatment 1)
this shows that these users when they are treated, they will visit and if they are not treated left alone, they will not visit hence treatment has a positive impact on these users and should always be treated.

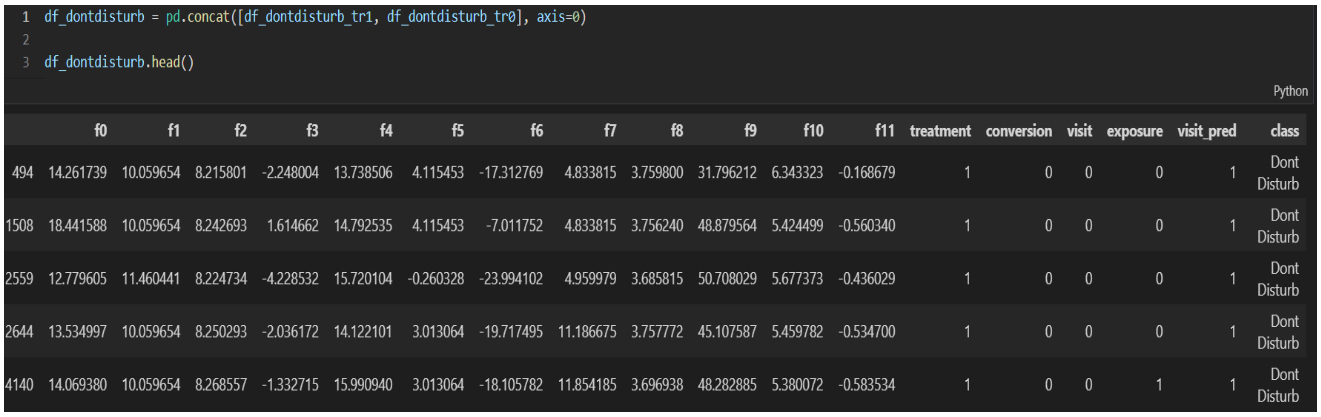
We must target persuadable only

5.5 Combining the datasets

Now we have combined the datasets of 4 classes for treatment=0 and treatment=1 datasets

5.5.1 Making final dataset for don't disturb class customers

Combining dataset for don't disturb for treatment=0 and treatment=1



```
df_dontdisturb = pd.concat([df_dontdisturb_tr1, df_dontdisturb_tr0], axis=0)

df_dontdisturb.head()
```

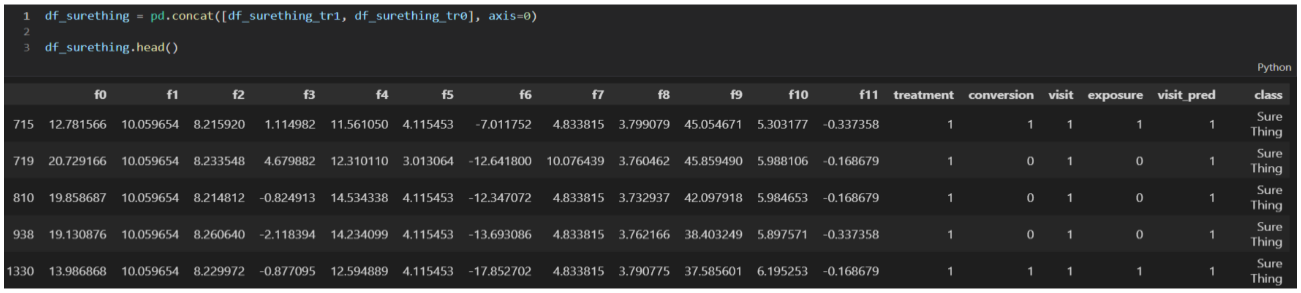
Python

	f0	f1	f2	f3	f4	f5	f6	f7	f8	f9	f10	f11	treatment	conversion	visit	exposure	visit_pred	class
494	14.261739	10.059654	8.215801	-2.248004	13.738506	4.115453	-17.312769	4.833815	3.759800	31.796212	6.343323	-0.168679	1	0	0	0	1	Dont Disturb
1508	18.441588	10.059654	8.242693	1.614662	14.792535	4.115453	-7.011752	4.833815	3.756240	48.879564	5.424499	-0.560340	1	0	0	0	1	Dont Disturb
2559	12.779605	11.460441	8.224734	-4.228532	15.720104	-0.260328	-23.994102	4.959979	3.685815	50.708029	5.677373	-0.436029	1	0	0	0	1	Dont Disturb
2644	13.534997	10.059654	8.250293	-2.036172	14.122101	3.013064	-19.717495	11.186675	3.757772	45.107587	5.459782	-0.534700	1	0	0	0	1	Dont Disturb
4140	14.069380	10.059654	8.268557	-1.332715	15.990940	3.013064	-18.105782	11.854185	3.696938	48.282885	5.380072	-0.583534	1	0	0	1	1	Dont Disturb

Figure 5-7 Combined Dataset of do not disturb class users

5.5.2 Making final dataset for Sure thing class customers

Combining dataset for sure thing for treatment=0 and treatment=1



```python
df_surething = pd.concat([df_surething_tr1, df_surething_tr0], axis=0)

df_surething.head()
```

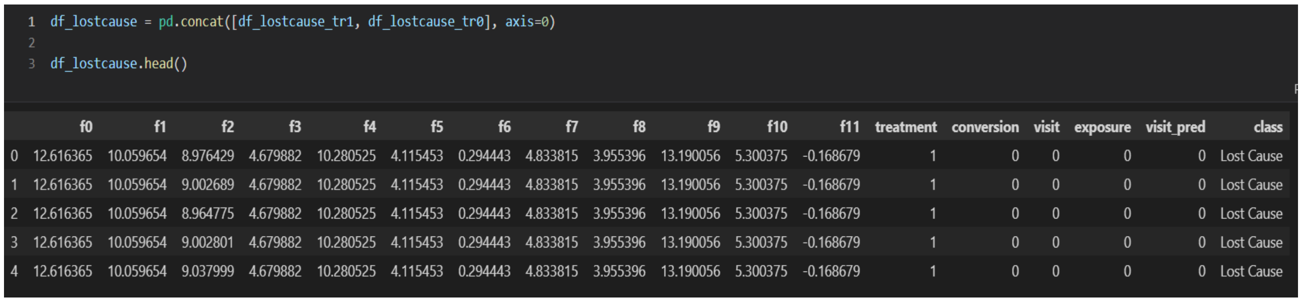
Python

	f0	f1	f2	f3	f4	f5	f6	f7	f8	f9	f10	f11	treatment	conversion	visit	exposure	visit_pred	class
715	12.781566	10.059654	8.215920	1.114982	11.561050	4.115453	-7.011752	4.833815	3.799079	45.054671	5.303177	-0.337358	1	1	1	1	1	Sure Thing
719	20.729166	10.059654	8.233548	4.679882	12.310110	3.013064	-12.641800	10.076439	3.760462	45.859490	5.988106	-0.168679	1	0	1	0	1	Sure Thing
810	19.858687	10.059654	8.214812	-0.824913	14.534338	4.115453	-12.347072	4.833815	3.732937	42.097918	5.984653	-0.168679	1	0	1	0	1	Sure Thing
938	19.130876	10.059654	8.260640	-2.118394	14.234099	4.115453	-13.693086	4.833815	3.762166	38.403249	5.897571	-0.337358	1	0	1	0	1	Sure Thing
1330	13.986868	10.059654	8.229972	-0.877095	12.594889	4.115453	-17.852702	4.833815	3.790775	37.585601	6.195253	-0.168679	1	1	1	1	1	Sure Thing

Figure 5-8 Combined dataset of sure thing class users

5.5.3 Making final dataset for Lost cause class customers

Combining dataset for lost cause for treatment=0 and treatment=1



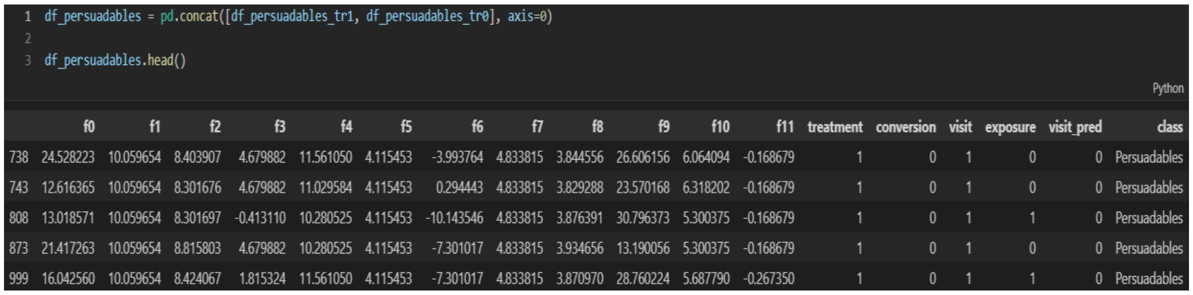
```python
df_lostcause = pd.concat([df_lostcause_tr1, df_lostcause_tr0], axis=0)

df_lostcause.head()
```

	f0	f1	f2	f3	f4	f5	f6	f7	f8	f9	f10	f11	treatment	conversion	visit	exposure	visit_pred	class
0	12.616365	10.059654	8.976429	4.679882	10.280525	4.115453	0.294443	4.833815	3.955396	13.190056	5.300375	-0.168679	1	0	0	0	0	Lost Cause
1	12.616365	10.059654	9.002689	4.679882	10.280525	4.115453	0.294443	4.833815	3.955396	13.190056	5.300375	-0.168679	1	0	0	0	0	Lost Cause
2	12.616365	10.059654	8.964775	4.679882	10.280525	4.115453	0.294443	4.833815	3.955396	13.190056	5.300375	-0.168679	1	0	0	0	0	Lost Cause
3	12.616365	10.059654	9.002801	4.679882	10.280525	4.115453	0.294443	4.833815	3.955396	13.190056	5.300375	-0.168679	1	0	0	0	0	Lost Cause
4	12.616365	10.059654	9.037999	4.679882	10.280525	4.115453	0.294443	4.833815	3.955396	13.190056	5.300375	-0.168679	1	0	0	0	0	Lost Cause

Figure 5-9 Combined dataset for lost cause class users

5.5.4 Making final dataset for persuadable class customers

Combining dataset for persuadable for treatment=0 and treatment=1



```python
df_persuadables = pd.concat([df_persuadables_tr1, df_persuadables_tr0], axis=0)

df_persuadables.head()
```

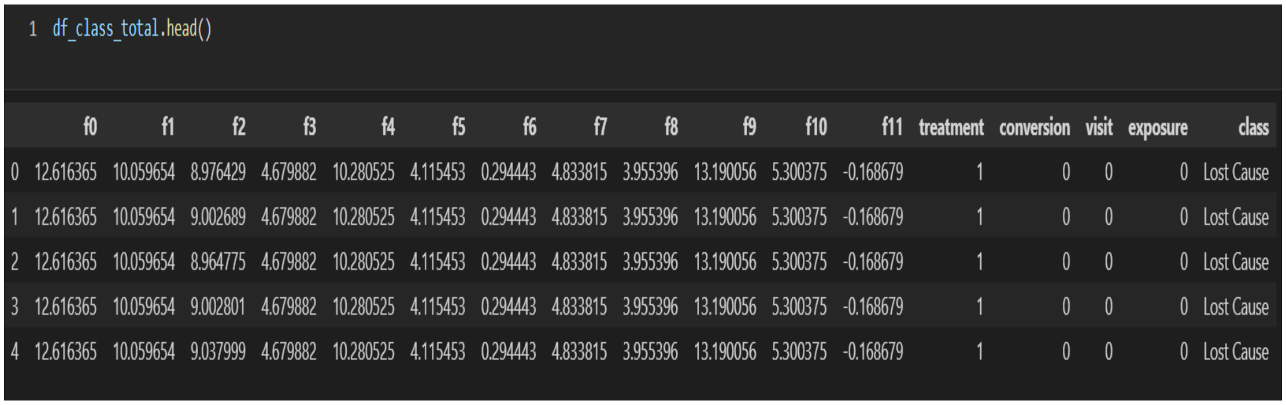
Python

	f0	f1	f2	f3	f4	f5	f6	f7	f8	f9	f10	f11	treatment	conversion	visit	exposure	visit_pred	class
738	24.528223	10.059654	8.403907	4.679882	11.561050	4.115453	-3.993764	4.833815	3.844556	26.606156	6.064094	-0.168679	1	0	1	0	0	Persuadables
743	12.616365	10.059654	8.301676	4.679882	11.029584	4.115453	0.294443	4.833815	3.829288	23.570168	6.318202	-0.168679	1	0	1	0	0	Persuadables
808	13.018571	10.059654	8.301697	-0.413110	10.280525	4.115453	-10.143546	4.833815	3.876391	30.796373	5.300375	-0.168679	1	0	1	1	0	Persuadables
873	21.417263	10.059654	8.815803	4.679882	10.280525	4.115453	-7.301017	4.833815	3.934656	13.190056	5.300375	-0.168679	1	0	1	0	0	Persuadables
999	16.042560	10.059654	8.424067	1.815324	11.561050	4.115453	-7.301017	4.833815	3.870970	28.760224	5.687790	-0.267350	1	0	1	1	0	Persuadables

Figure 5-10 Combined dataset for persuadable class user

5.5.5 Making final dataset of all customers with all classes

Finally joining all the combined datasets of 4 classes to make a complete dataset with classes of users



```
df_class_total.head()
```

	f0	f1	f2	f3	f4	f5	f6	f7	f8	f9	f10	f11	treatment	conversion	visit	exposure	class
0	12.616365	10.059654	8.976429	4.679882	10.280525	4.115453	0.294443	4.833815	3.955396	13.190056	5.300375	-0.168679	1	0	0	0	Lost Cause
1	12.616365	10.059654	9.002689	4.679882	10.280525	4.115453	0.294443	4.833815	3.955396	13.190056	5.300375	-0.168679	1	0	0	0	Lost Cause
2	12.616365	10.059654	8.964775	4.679882	10.280525	4.115453	0.294443	4.833815	3.955396	13.190056	5.300375	-0.168679	1	0	0	0	Lost Cause
3	12.616365	10.059654	9.002801	4.679882	10.280525	4.115453	0.294443	4.833815	3.955396	13.190056	5.300375	-0.168679	1	0	0	0	Lost Cause
4	12.616365	10.059654	9.037999	4.679882	10.280525	4.115453	0.294443	4.833815	3.955396	13.190056	5.300375	-0.168679	1	0	0	0	Lost Cause

This dataset will be used for the part 2 implementation

6. Implementation of our Approach part 2

In part 2 we will work on combined dataset with classes

6.1 Data Exploration and analysis

As dataset is nearly same as original dataset and only 1 column of class is different, so data exploration is done similarly as done for original dataset and it has similar results (so not mentioned here)

6.1.1 Dataset Size

The size of the csv is **3.309 GB**
The no. of rows = 13979592
Number of columns = 17

6.1.2 Data types

The features f0 to f11 are of float type while the rest
Treatment is integer (0 or 1)
Exposure is integer (0 or 1)
Conversion is integer (0 or 1)
Visit is integer (0 or 1)
Class is object

6.1.3 Finding correlation matrix

For analysis and study, we did one hot encoding on dataset to find the correlation of each user class and make our correlation matrix

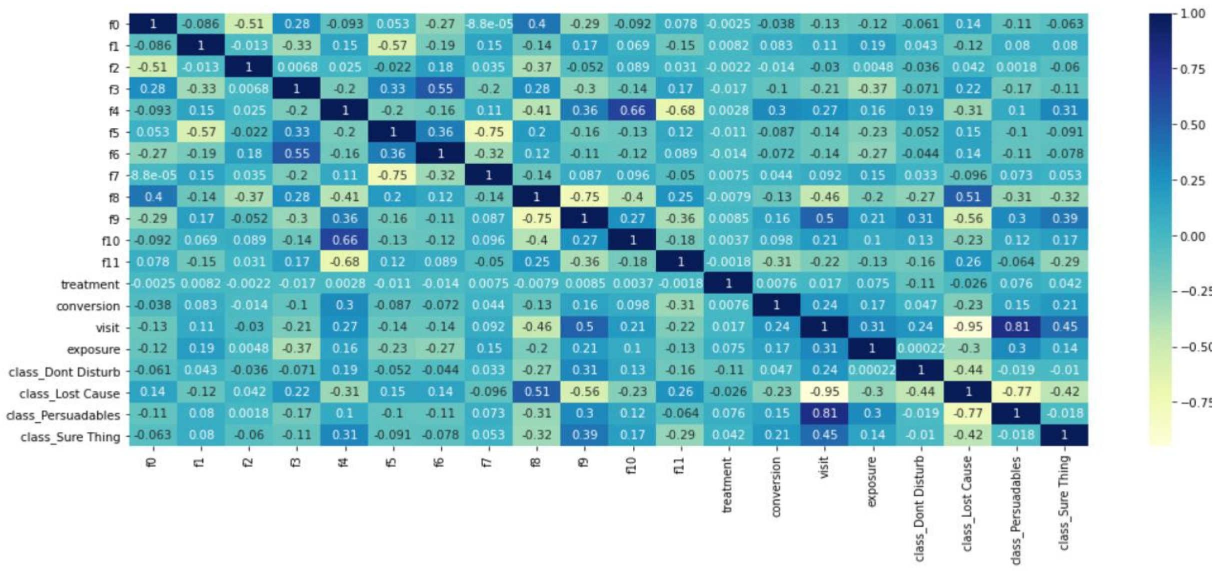
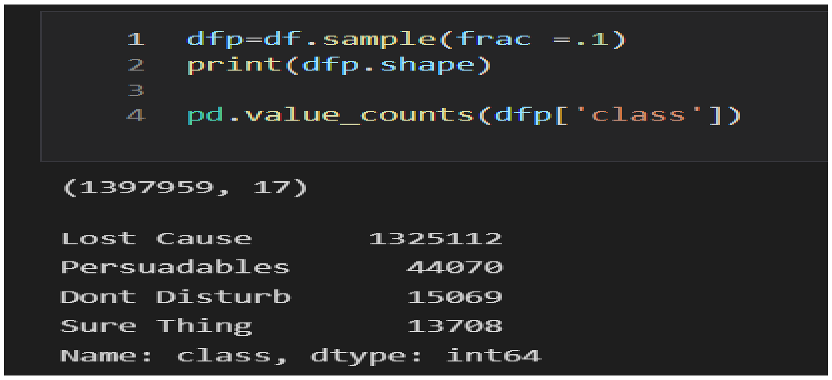


Figure 6-1 Correlation matrix of our combined dataset with classes

6.2 Finding the winner final-model

As size of dataset is very big, we created a smaller sample of dataset.
A random sample 10% size of full dataset was created



```
dfp=df.sample(frac =.1)
print(dfp.shape)

pd.value_counts(dfp['class'])
```

```
(1397959, 17)

Lost Cause        1325112
Persuadables        44070
Dont Disturb        15069
Sure Thing          13708
Name: class, dtype: int64
```

Figure 6-2 Sample random 10% dataset

It can be seen in the figure above that all the classes are present in this sample

Next, we used train test split on this sampled dataset for our initial analysis of choosing best model

the following results of ROC AUC: of different models obtained

Logistic :	0.9421809450740963
Decision Tree :	0.9567119916019335
Random Forest :	0.9626564673164548
Graident Boosting :	0.9633181808365909
Hist-Gradient Boosting :	0.9651601462857793
Naive Bayes Gaussian :	0.9339242229009771
k-NN :	0.8172690444914528
Scaled k-NN :	0.8600238102551588

6.2.1 1st best model

Hist-Gradient boosting gave 1st best result

6.2.2 2nd best model

Gradient boosting gave 2nd best result
GradientBoostingClassifier(max_depth=2,n_estimators=50, verbose=2)
AUC 0.9633181808365909

6.2.3 3rd best model

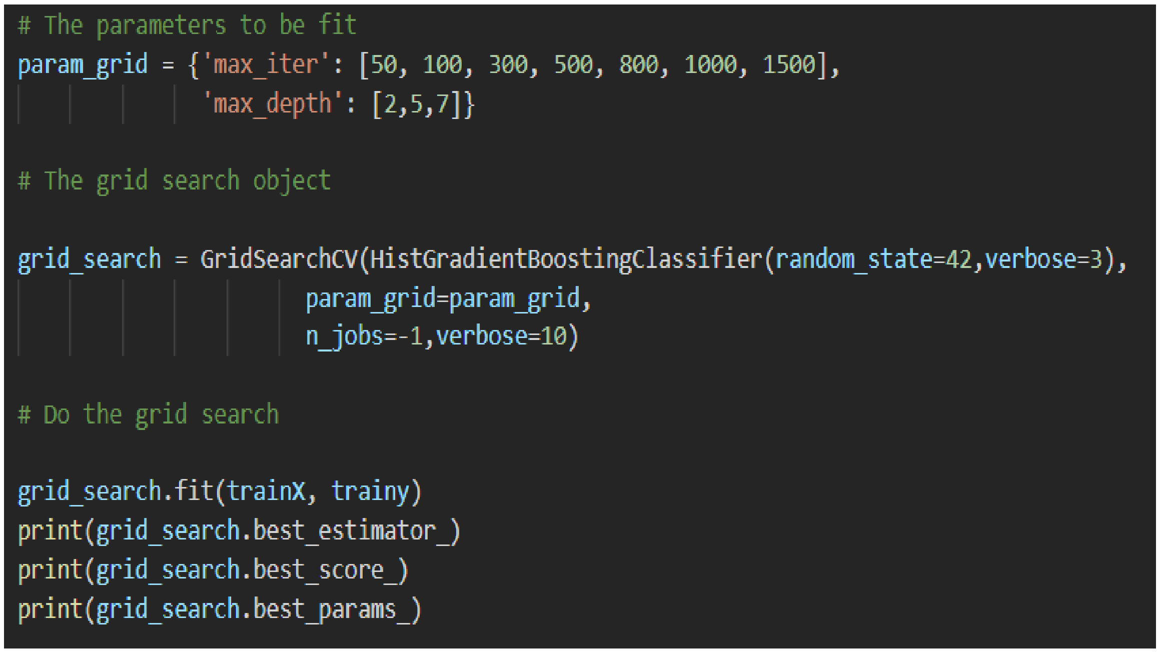
Random forest gave 3rd best result
RandomForestClassifier(max_depth=5,n_estimators=50, verbose=2)
AUC 0.9626564673164548

6.3 Hyper parameter tuning of best final-model

For finding the parameters of our best model hist gradient we used GridSearch

6.3.1 GridSearchCV

Applied grid search CV to find parameters for our best model of histgradient



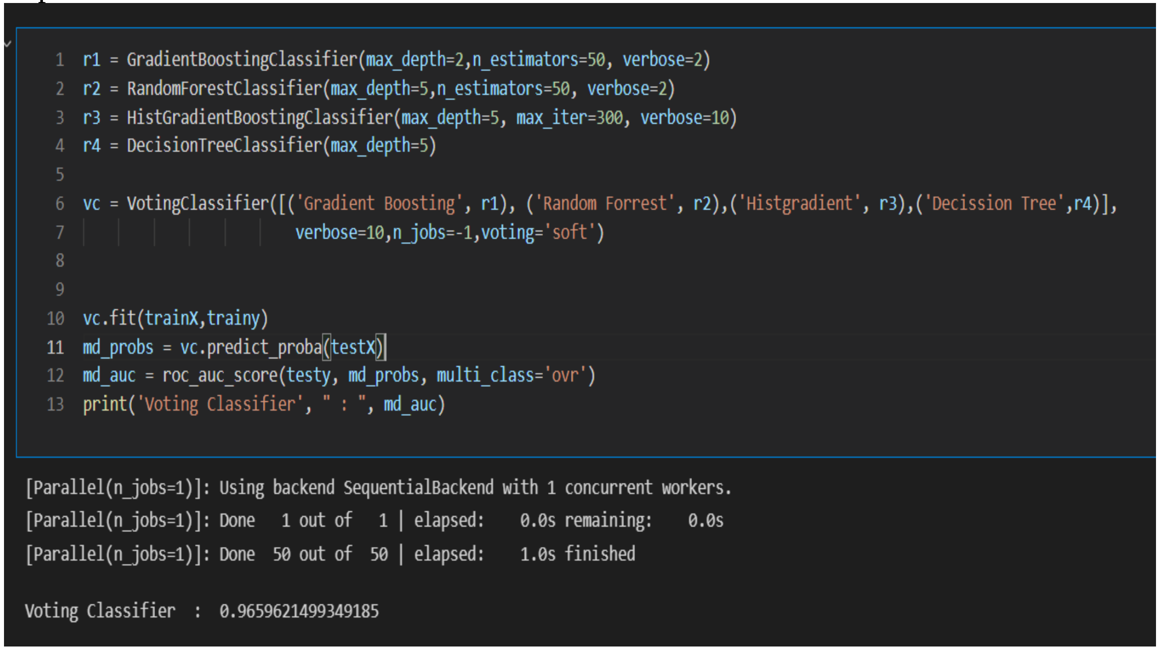
```
# The parameters to be fit
param_grid = {'max_iter': [50, 100, 300, 500, 800, 1000, 1500],
              'max_depth': [2,5,7]}

# The grid search object

grid_search = GridSearchCV(HistGradientBoostingClassifier(random_state=42,verbose=3),
                          param_grid=param_grid,
                          n_jobs=-1,verbose=10)

# Do the grid search

grid_search.fit(trainX, trainy)
print(grid_search.best_estimator_)
print(grid_search.best_score_)
print(grid_search.best_params_)
```

Figure 6-3 Gridsearch for finding best paramters

The following results of parameters obtained by gridsearch
{'max_depth': 5, 'max_iter': 300}
AUC 0.9676611853222854

6.4 Voting classifier

Finally Voting classifier was applied on best results of histagragient boosting, gradient boosting, decision tree, random forest but the result of AUC (gave 0.96596 AUC) did not improve as shown below



```
r1 = GradientBoostingClassifier(max_depth=2,n_estimators=50, verbose=2)
r2 = RandomForestClassifier(max_depth=5,n_estimators=50, verbose=2)
r3 = HistGradientBoostingClassifier(max_depth=5, max_iter=300, verbose=10)
r4 = DecisionTreeClassifier(max_depth=5)

vc = VotingClassifier([('Gradient Boosting', r1), ('Random Forrest', r2),('Histgradient', r3),('Decission Tree',r4)],
                        verbose=10,n_jobs=-1,voting='soft')


vc.fit(trainX,trainy)
md_probs = vc.predict_proba(testX)
md_auc = roc_auc_score(testy, md_probs, multi_class='ovr')
print('Voting Classifier', " : ", md_auc)
```

```
[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent workers.
[Parallel(n_jobs=1)]: Done   1 out of   1 | elapsed:    0.0s remaining:    0.0s
[Parallel(n_jobs=1)]: Done  50 out of  50 | elapsed:    1.0s finished

Voting Classifier  :  0.9659621499349185
```

Figure 6-4 Voting Classifier

Hence hist-gradient boosting is the final model which gave best AUC of 0.9676

6.5 Training final-model on full dataset

Now final-model is trained on full dataset

Best final-model
HistGradientBoostingClassifier(max_depth=5, max_iter=300, verbose=10)
AUC (trained on full data and tested on full dataset) 0.9685990334910384

The result of our tarinned fitted final model on full dataset is stored as pred_final.sav file using python pickle library. This can be called anytime to predict user classes

6.6 Predicting the class of customer on test data

For our project the test data will be the input we will take and predict the class to which user belongs

7. Building Web API

The front end of our project is a web API using streamlit

7.1 Using Streamlit for building web API

A simple web API is made using stremlit library of python, this web API will predict results class 0f customer after taking inputs from user

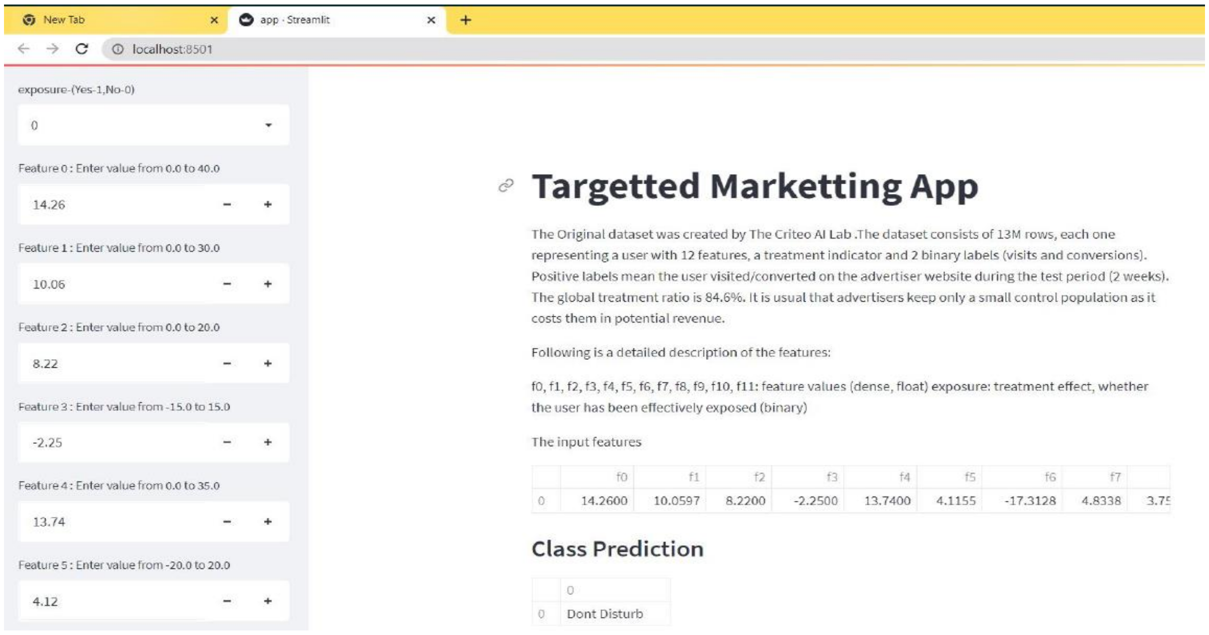
7.2 Taking inputs in API from user for our final prediction of customer class

We have taken f0 to f11 and exposure as our 13 user inputs
These inputs will make a test dataset which will predict results using final-model which was saved

The final model pred_final.sav is imported and the predictions are made

7.3 Running the final Web API

Web API was run successfully, and a snapshot of API is below



exposure-(Yes-1,No-0)

0

Feature 0 : Enter value from 0.0 to 40.0

14.26

Feature 1 : Enter value from 0.0 to 30.0

10.06

Feature 2 : Enter value from 0.0 to 20.0

8.22

Feature 3 : Enter value from -15.0 to 15.0

-2.25

Feature 4 : Enter value from 0.0 to 35.0

13.74

Feature 5 : Enter value from -20.0 to 20.0

4.12

Targetted Marketting App

The Original dataset was created by The Criteo AI Lab .The dataset consists of 13M rows, each one representing a user with 12 features, a treatment indicator and 2 binary labels (visits and conversions). Positive labels mean the user visited/converted on the advertiser website during the test period (2 weeks). The global treatment ratio is 84.6%. It is usual that advertisers keep only a small control population as it costs them in potential revenue.

Following is a detailed description of the features:

f0, f1, f2, f3, f4, f5, f6, f7, f8, f9, f10, f11: feature values (dense, float) exposure: treatment effect, whether the user has been effectively exposed (binary)

The input features

	f0	f1	f2	f3	f4	f5	f6	f7	
0	14.2600	10.0597	8.2200	-2.2500	13.7400	4.1155	-17.3128	4.8338	3.7

Class Prediction

	0
0	Dont Disturb

Figure 7-1 Web API of projec

8. Results

8.1 All winner models comparison

The best results of winner model are shown in table below

Winner model	Model Description	ROC AUC
Model-1	RandomForestClassifier(max_depth=10, n_estimators=800, verbose=2, max_features=7)	0.944895442
Model-2	GradientBoostingClassifier(max_depth=2,n_estimators=200,verbose=2)	0.932731664
Final-Model	HistGradientBoostingClassifier(max_depth=5, max_iter=300, verbose=10)	0.968599033

8.2 The persuadable customers in our dataset

For our original dataset we can have the results
Out of total 13979592 customers there are only 440543 persuadable customers should be targeted that is 3.15% of total customers

Toral customers =13979592
persuadable customers= 440543
Percentage of customers to be targeted in terms of total customers= 3.15%

Customers who were treated(treatment=1) = 11882655
Percentage of customers to be targeted in terms of treated customers= 3.71%

8.3 Extra Customers treated (didn’t need treatment)

Extra Customers treated= 11442112
Percentage of Extra Customers treated = 96.29%

That is nearly 96.29% of extra customers were treated

8.4 Do not disturb customers treated (treatment had negative impact)

Do not disturb customers treated= 69757
Percentage of do not disturb customers targeted= 0.59%

9. Benefits

This project will give following benefits

9.1 Cost Saving

By using our model approximately 96% of reduction in number of customers to be treated this will decrease the overall budget and cost of marketing and save money

9.2 Increase Marketing efficiency

With the reduction in cost also the efficiency of marketing will increase as we have also removed do not disturb customers (on which treatment had a negative impact).
Therefore, greater efficient marketing will be achieved by this project

9.3 Identify customers to target

Using the final model company can identify which user should be treated, that is targeting customers for advertisements.

10. Conclusion

This project has used marketing data, classified users found customers which must be targeted, improved efficiency of marketing and reduced overall marketing cost

Using these models and this project people can identify customer class then develop a better marketing strategy and plan with best future results.

11. Problems Faced

- As the size of dataset of is very big so Gridsearch could not be utilized to full extent
- The big size of dataset made finding best parameter time consuming.
- As dataset had only few features so feature selection had minimum impact and was not useful

12. Future works

- More features can be added to make model better and improve prediction power

- The model can be used with different features and in various sectors for targeted marketing plans.

- CuasalML tools and uplift model using pylift library of python can also be applied on dataset to study and analyze dataset more

References

Lendave, Vijaysinh. n.d. https://analyticsindiamag.com/what-is-uplift-modelling-and-how-can-it-be-done-with-causalml/#:~:text=Uplift%20modelling%20is%20a%20predictive,as%20targeting%20promotions%20and%20advertisements.

Möbius-Kaggle. n.d. *kaggle.* https://www.kaggle.com/datasets/arashnic/uplift-modeling?select=criteo-uplift-v2.1.csv.

- https://analyticsindiamag.com/what-is-uplift-modelling-and-how-can-it-be-done-with-causalml/#:~:text=Uplift%20modelling%20is%20a%20predictive,as%20targeting%20promotions%20and%20advertisements.

- https://www.kaggle.com/datasets/arashnic/uplift-modeling?select=criteo-uplift-v2.1.csv