Machine Learning Project Report

On

Kaggle's

Google Analytics Customer Revenue Prediction

Hieu Trung Tran: txt171930
Rutali Bandivadekar: rdb170002
Pallavi Pandey: pxp170009

Contents

Topic 1

Introduction and Problem Description

We have participated in the Kaggle Competition ‘Google Analytics Customer Revenue Prediction’. The competition provides data from the Google Merchandise Store (also known as GStore). Our goal is to develop a model that will help to predict the revenue of a customer.

We believe that this is a great opportunity for us to showcase how digital marketing can make use of data science to answer any business problem. Digital marketing can take important data insights learned from the given datato improve their marketing strategies and business to maximize their rate of return (ROI).

In our approach, we are using the exploratory data analysis for applying data science to business problem.

Competition link: https://www.kaggle.com/c/ga-customer-revenue-prediction

Topic 2

Related Work

We know that the 80/20 rule holds true for a lot of businesses. It says that it is only a small percentage of the customers that make up most of the revenue. Owing to this, the marketers face a tough task of coming up with various promotional/marketing techniques.

Through our project, the outcome that we have provided will consider better actionable changes which help the marketing managers to imply various strategies on Gstore.

We have referred a few machine learning tasks so that we have a deep insight into which machine learning tasks we should perform.

- For preprocessing and analyzing the data, we referred a few of the kernels available online on www.kaggle.com.
- The initial steps required a very large and complex data to be preprocessed and thus we referred to DataCamp's course "Preprocessing for Machine Learning in Python" in order to clean our data for modeling.

Topic 3

Dataset Description

The customer dataset consists of**1708337** records. Out of these, **903653 are training samples** and remaining **804684 are the test samples**. We have total **12 features and 1 target** which is transaction revenue. The featuresas shown below:

fullVisitorId- A unique identifier for each user of the Google Merchandise Store.

[x] channelGrouping - The channel via which the user came to the Store.

[x] date - The date on which the user visited the Store.

[x] device - The specifications for the device used to access the Store.

[x] geoNetwork - This section contains information about the geography of the user.

sessionId - A unique identifier for this visit to the store.

[x] socialEngagementType - Engagement type, either "Socially Engaged" or "Not Socially Engaged".

[x] totals - This section contains aggregate values across the session.

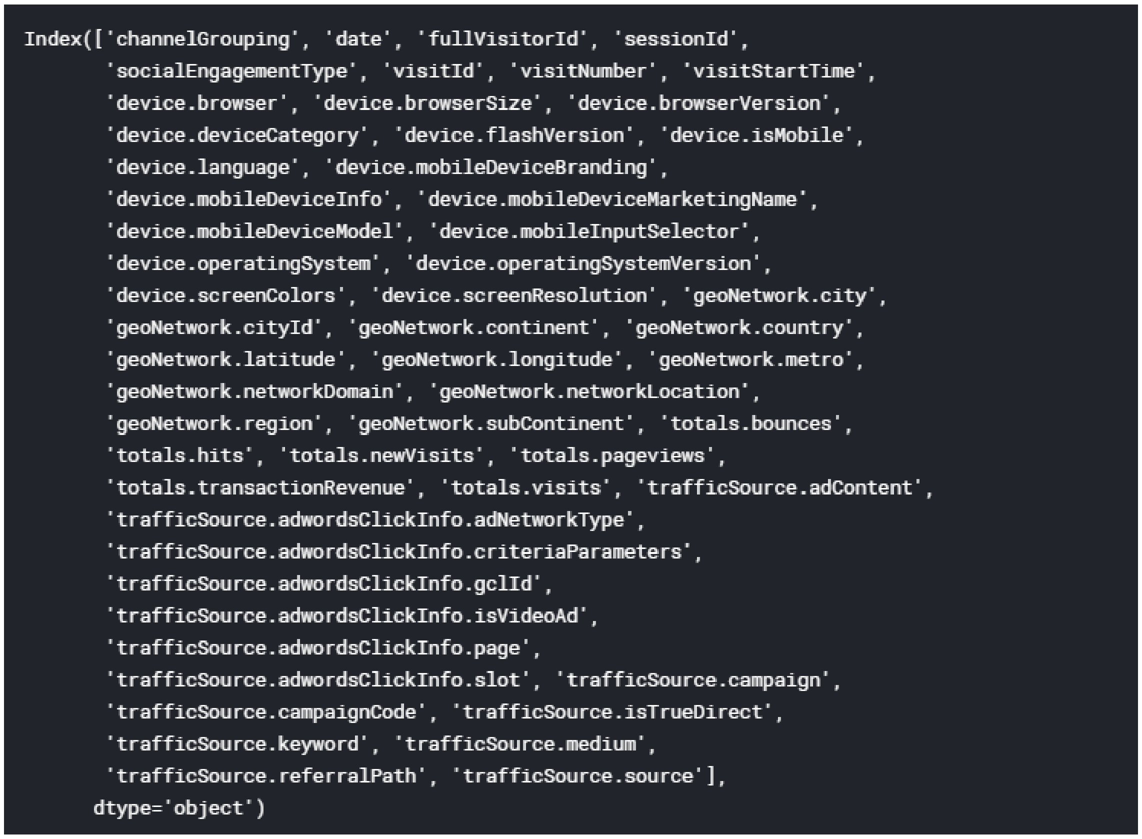
[x] trafficSource - This section contains information about the Traffic Source from which the session originated.

visitId - An identifier for this session. This is part of the value usually stored as the _utmb cookie. This is only unique to the user. For a completely unique ID, you should use a combination of fullVisitorId and visitId.

visitNumber - The session number for this user. If this is the first session, then this is set to 1.

visitStartTime - The timestamp (expressed as POSIX time).

Train dataset columns



```
Index(['channelGrouping', 'date', 'fullVisitorId', 'sessionId',
       'socialEngagementType', 'visitId', 'visitNumber', 'visitStartTime',
       'device.browser', 'device.browserSize', 'device.browserVersion',
       'device.deviceCategory', 'device.flashVersion', 'device.isMobile',
       'device.language', 'device.mobileDeviceBranding',
       'device.mobileDeviceInfo', 'device.mobileDeviceMarketingName',
       'device.mobileDeviceModel', 'device.mobileInputSelector',
       'device.operatingSystem', 'device.operatingSystemVersion',
       'device.screenColors', 'device.screenResolution', 'geoNetwork.city',
       'geoNetwork.cityId', 'geoNetwork.continent', 'geoNetwork.country',
       'geoNetwork.latitude', 'geoNetwork.longitude', 'geoNetwork.metro',
       'geoNetwork.networkDomain', 'geoNetwork.networkLocation',
       'geoNetwork.region', 'geoNetwork.subContinent', 'totals.bounces',
       'totals.hits', 'totals.newVisits', 'totals.pageviews',
       'totals.transactionRevenue', 'totals.visits', 'trafficSource.adContent',
       'trafficSource.adwordsClickInfo.adNetworkType',
       'trafficSource.adwordsClickInfo.criteriaParameters',
       'trafficSource.adwordsClickInfo.gclId',
       'trafficSource.adwordsClickInfo.isVideoAd',
       'trafficSource.adwordsClickInfo.page',
       'trafficSource.adwordsClickInfo.slot', 'trafficSource.campaign',
       'trafficSource.campaignCode', 'trafficSource.isTrueDirect',
       'trafficSource.keyword', 'trafficSource.medium',
       'trafficSource.referralPath', 'trafficSource.source'],
      dtype='object')
```

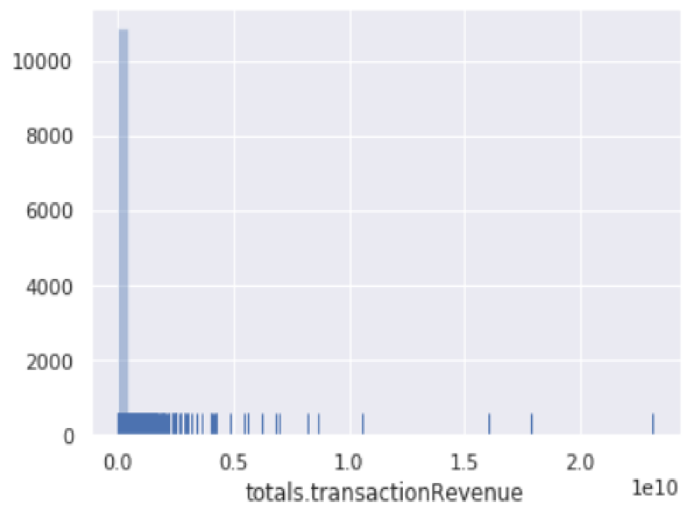
After splitting up each and all JSON columns, we have **55 features in training sample** and **53 in test sample**.

Topic 4

Data Preprocessing Techniques

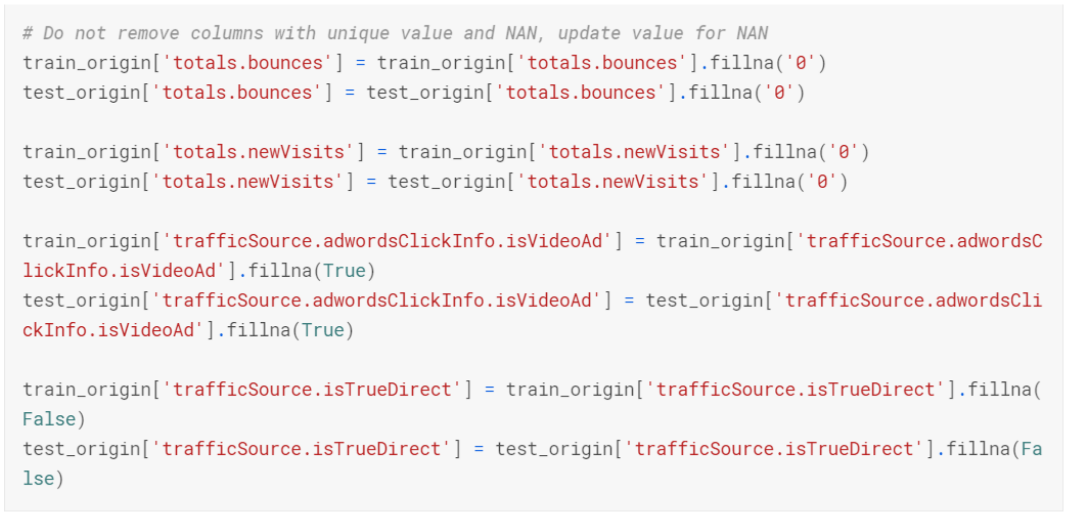
The target we want to predict, transactionRevenue, is contained in one of the JSON columns. While loading the dataset, we renamed it to totals.transactionRevenue. The target only contains a few non-null values. Only 11515 sessions have done a transaction which is approximately 1.3% of the total observations. Therefore, we fill the NAs with zero for the column totals.transactionRevenue.

Total.transactionRevenue plot



Next, we searched the columns which contain all unique and Null values. Most of the unique value columns contain Boolean data, so we replaced the null values with other Boolean data accordingly.

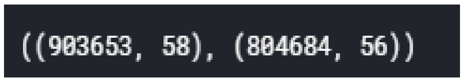
Fill Nan value for other attributes



```
# Do not remove columns with unique value and NAN, update value for NAN
train_origin['totals.bounces'] = train_origin['totals.bounces'].fillna('0')
test_origin['totals.bounces'] = test_origin['totals.bounces'].fillna('0')

train_origin['totals.newVisits'] = train_origin['totals.newVisits'].fillna('0')
test_origin['totals.newVisits'] = test_origin['totals.newVisits'].fillna('0')

train_origin['trafficSource.adwordsClickInfo.isVideoAd'] = train_origin['trafficSource.adwordsC
lickInfo.isVideoAd'].fillna(True)
test_origin['trafficSource.adwordsClickInfo.isVideoAd'] = test_origin['trafficSource.adwordsCli
ckInfo.isVideoAd'].fillna(True)

train_origin['trafficSource.isTrueDirect'] = train_origin['trafficSource.isTrueDirect'].fillna(
False)
test_origin['trafficSource.isTrueDirect'] = test_origin['trafficSource.isTrueDirect'].fillna(Fa
lse)
```

We transferred the visitStartTime into understandable formats: [day of week(0-6), hour(0-24) and day(0-30)] Thus, the number of columns got increased to 58 for training data and 56 for the test data.

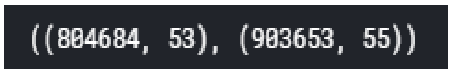
<u>Train Original Shape, Test Original Shape</u>



```
((903653, 58), (804684, 56))
```

After converting visitStartTime into understandable format we have:

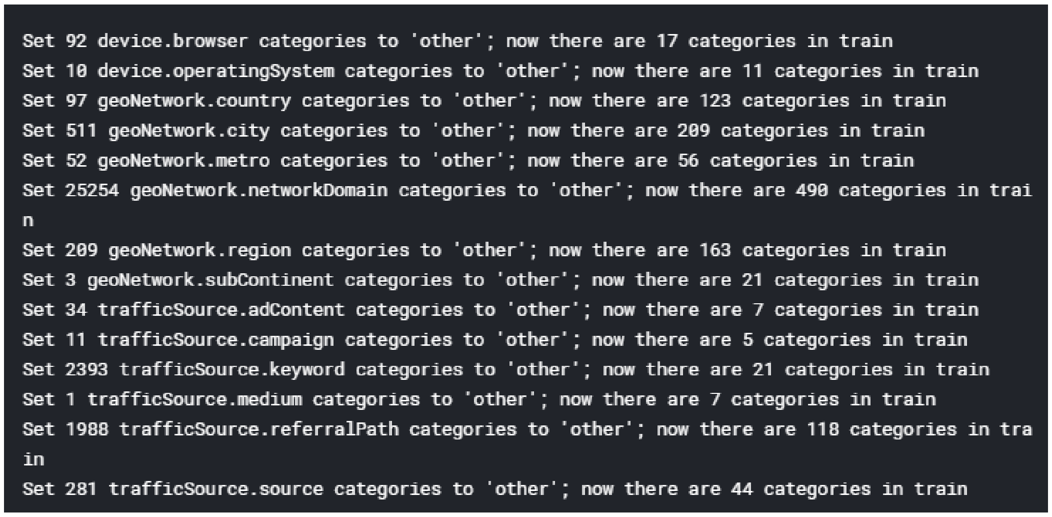
<u>Train Shape, Test Shape</u>



```
((804684, 53), (903653, 55))
```

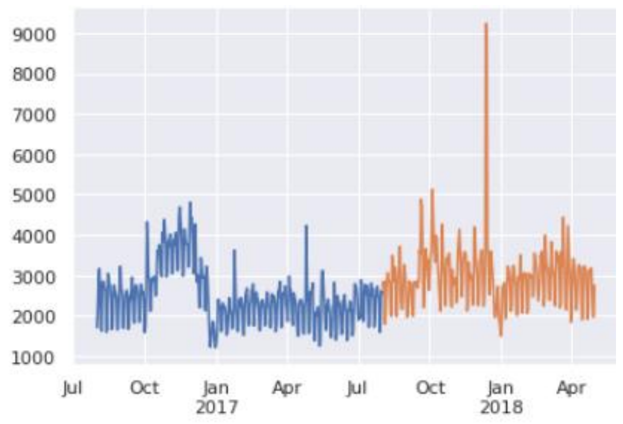
We also reduced number of rare categories for following columns:

[‘device.browser’, ‘device.operatingSystem’, ‘geoNetwork.country’, ‘geoNetwork.city’, ‘geoNetwork.metro’, ‘geoNetwork.networkDomain’, ‘geoNetwork.region’, ‘geoNetwork.subContinent’, ‘trafficSource.adContent’, ‘trafficSource.campaign’, ‘trafficSource.keyword’, ‘trafficSource.medium’, ‘trafficSource.referralPath’, ‘trafficSource.source’]

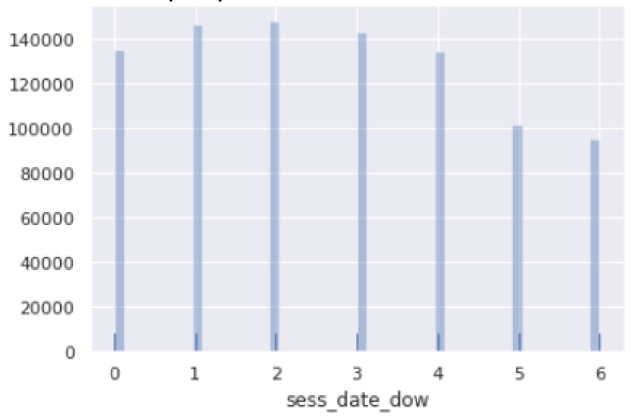


```
Set 92 device.browser categories to 'other'; now there are 17 categories in train
Set 10 device.operatingSystem categories to 'other'; now there are 11 categories in train
Set 97 geoNetwork.country categories to 'other'; now there are 123 categories in train
Set 511 geoNetwork.city categories to 'other'; now there are 209 categories in train
Set 52 geoNetwork.metro categories to 'other'; now there are 56 categories in train
Set 25254 geoNetwork.networkDomain categories to 'other'; now there are 490 categories in trai
n
Set 209 geoNetwork.region categories to 'other'; now there are 163 categories in train
Set 3 geoNetwork.subContinent categories to 'other'; now there are 21 categories in train
Set 34 trafficSource.adContent categories to 'other'; now there are 7 categories in train
Set 11 trafficSource.campaign categories to 'other'; now there are 5 categories in train
Set 2393 trafficSource.keyword categories to 'other'; now there are 21 categories in train
Set 1 trafficSource.medium categories to 'other'; now there are 7 categories in train
Set 1988 trafficSource.referralPath categories to 'other'; now there are 118 categories in tra
in
Set 281 trafficSource.source categories to 'other'; now there are 44 categories in train
```

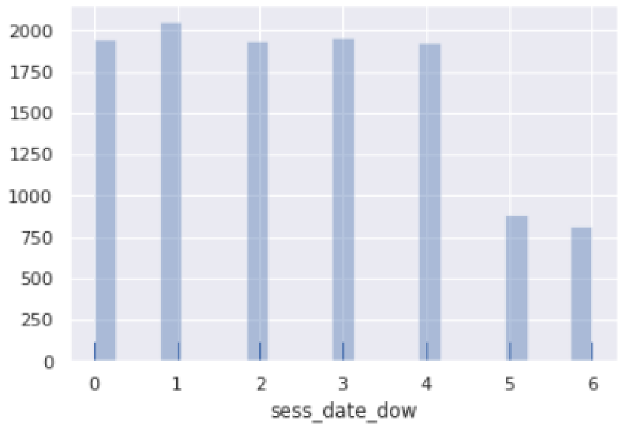
Analyzed transaction time of dataset



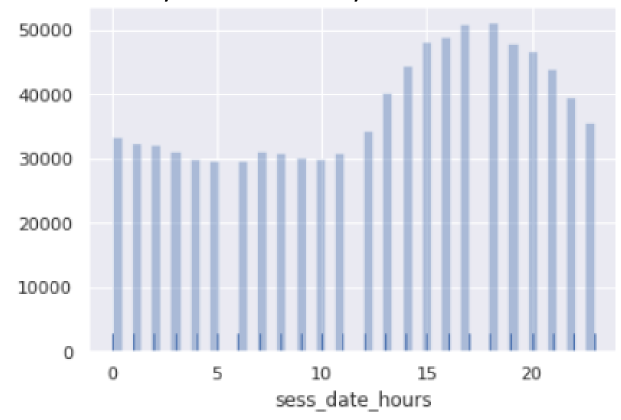
Distribution by day of week



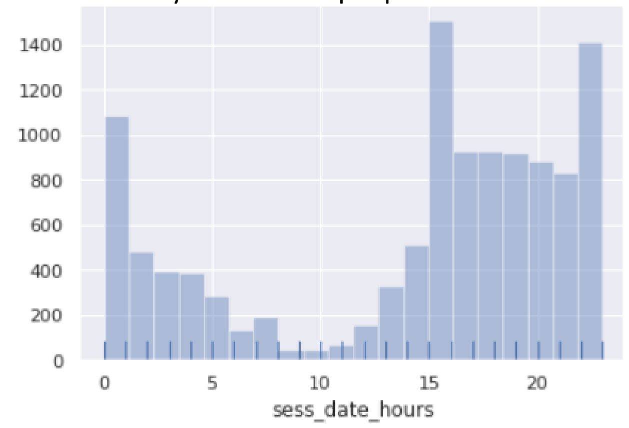
Distribution by date of week when people have done transactions



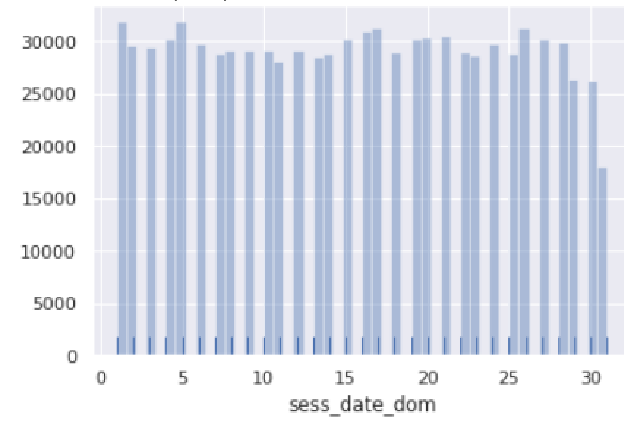
Distribution by hour of the day



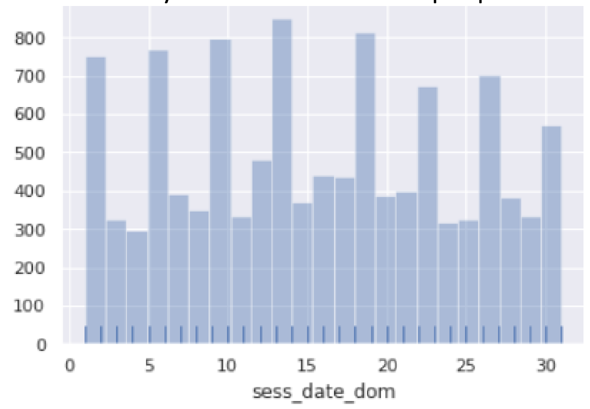
Distribution by hours when people have done transactions



Distribution by day of month



Distribution by date of month when people have done transactions



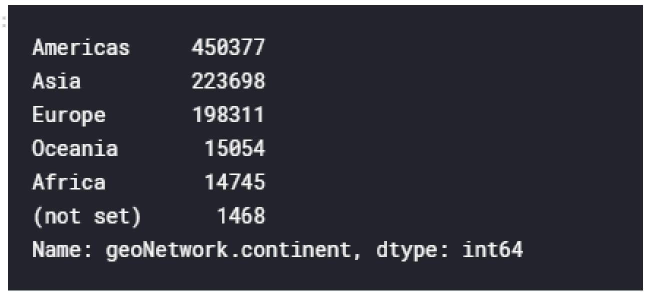
Next, we removed the columns which were either unique for all the customers or they were present only in the training data but not in the test data (only one column trafficSource.campaignCode is not present in test data). Thus, we now only have **31 features** in the training data which are given below.

['channelGrouping', 'device.browser', 'device.deviceCategory', 'device.isMobile', 'device.operatingSystem', 'geoNetwork.city', 'geoNetwork.continent', 'geoNetwork.country', 'geoNetwork.metro', 'geoNetwork.networkDomain', 'geoNetwork.region', 'geoNetwork.subContinent', 'totals.bounces', 'totals.hits', 'totals.newVisits', 'totals.pageviews', 'trafficSource.adContent', 'trafficSource.adwordsClickInfo.adNetworkType', 'trafficSource.adwordsClickInfo.gclId', 'trafficSource.adwordsClickInfo.isVideoAd', 'trafficSource.adwordsClickInfo.page', 'trafficSource.adwordsClickInfo.slot', 'trafficSource.campaign', 'trafficSource.isTrueDirect', 'trafficSource.keyword', 'trafficSource.medium', 'trafficSource.referralPath', 'trafficSource.source', 'sess_date_dow', 'sess_date_hours', 'sess_date_dom']

Topic 5

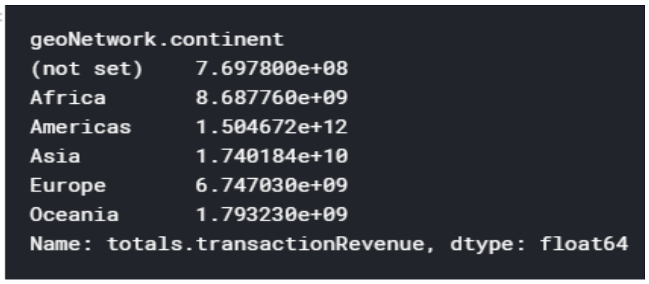
Data Visualization

Continent wise data count



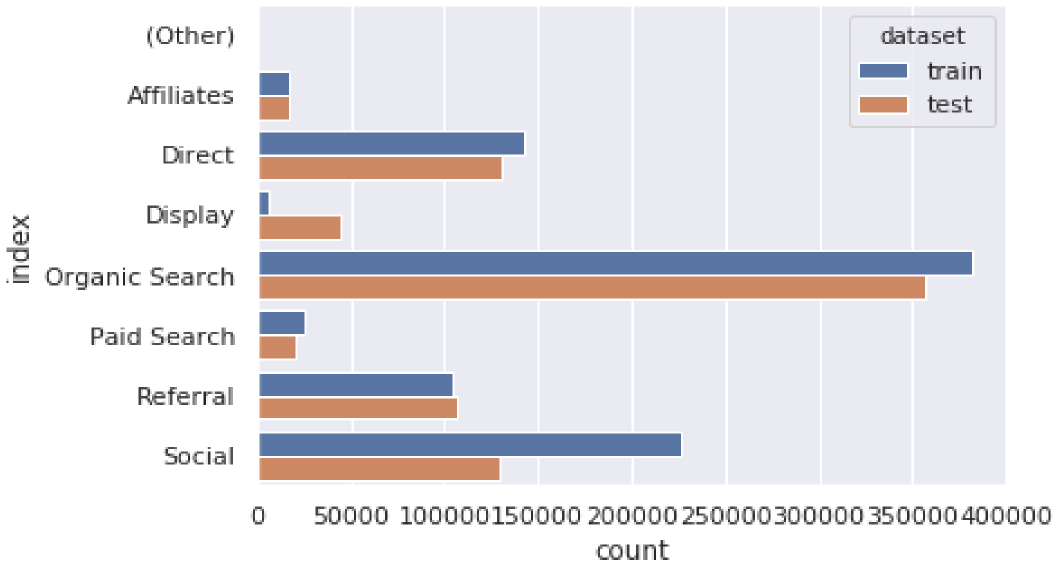
```
Americas      450377
Asia          223698
Europe        198311
Oceania        15054
Africa         14745
(not set)       1468
Name: geoNetwork.continent, dtype: int64
```

Total revenue by continent

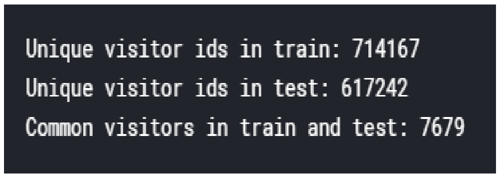


```
geoNetwork.continent
(not set)     7.697800e+08
Africa        8.687760e+09
Americas      1.504672e+12
Asia          1.740184e+10
Europe        6.747030e+09
Oceania       1.793230e+09
Name: totals.transactionRevenue, dtype: float64
```

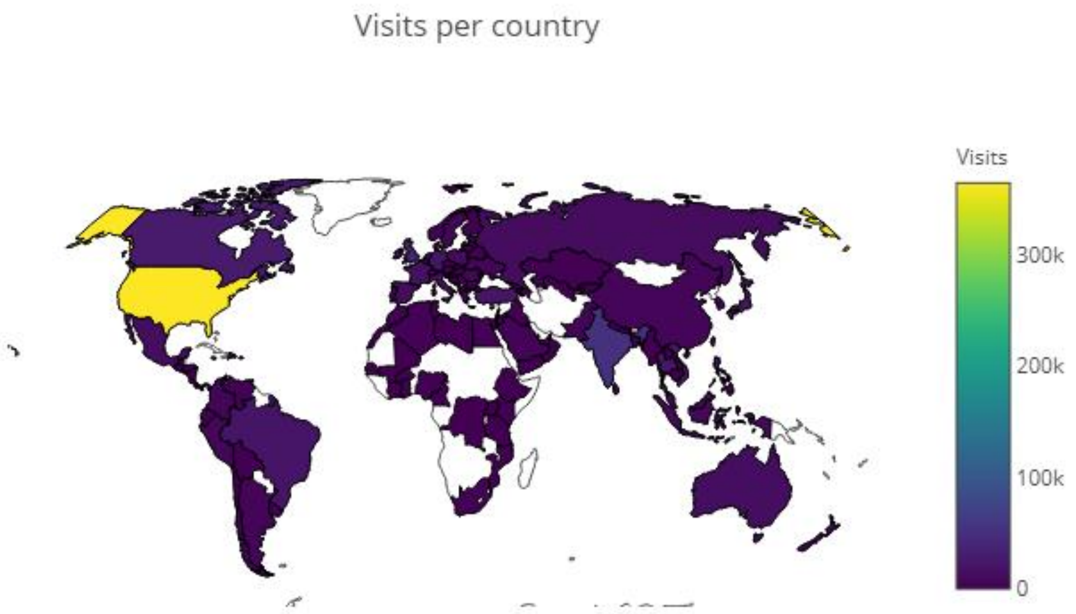
Visualization of channel grouping column

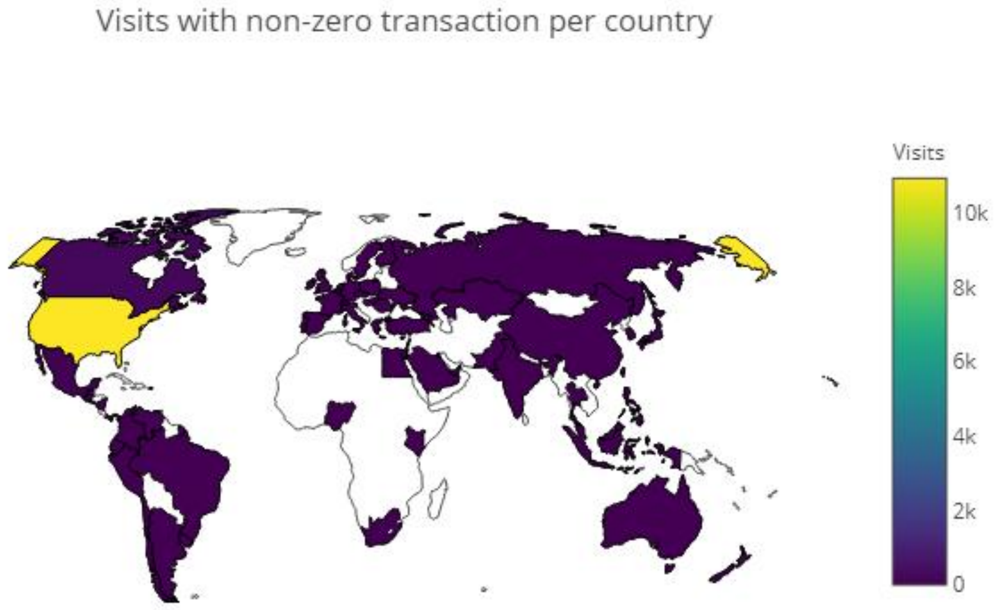


Country wise visitors count

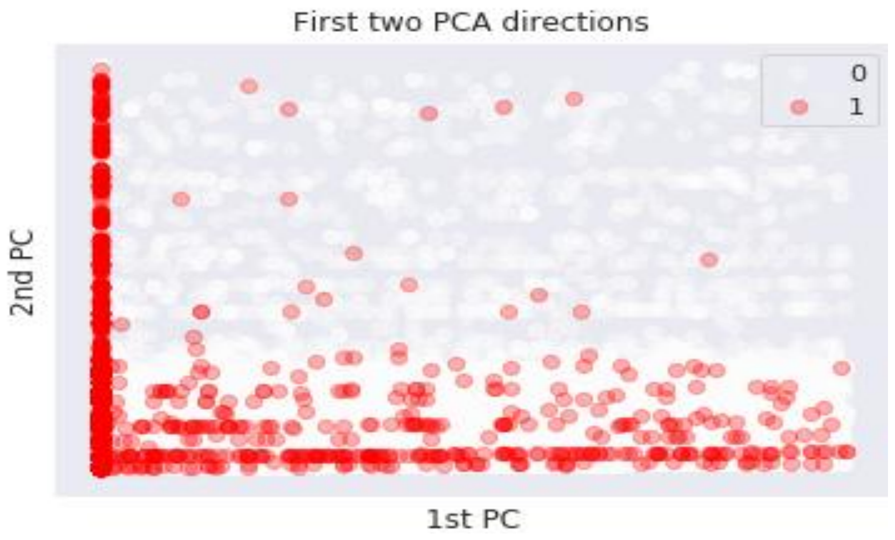


```
Unique visitor ids in train: 714167
Unique visitor ids in test: 617242
Common visitors in train and test: 7679
```





We have used PCA to visualize the dataset.



We have used PCA to reduce the dimensionality of the data as dimensionality plays a crucial role when the data consists of many features.

Topic 6

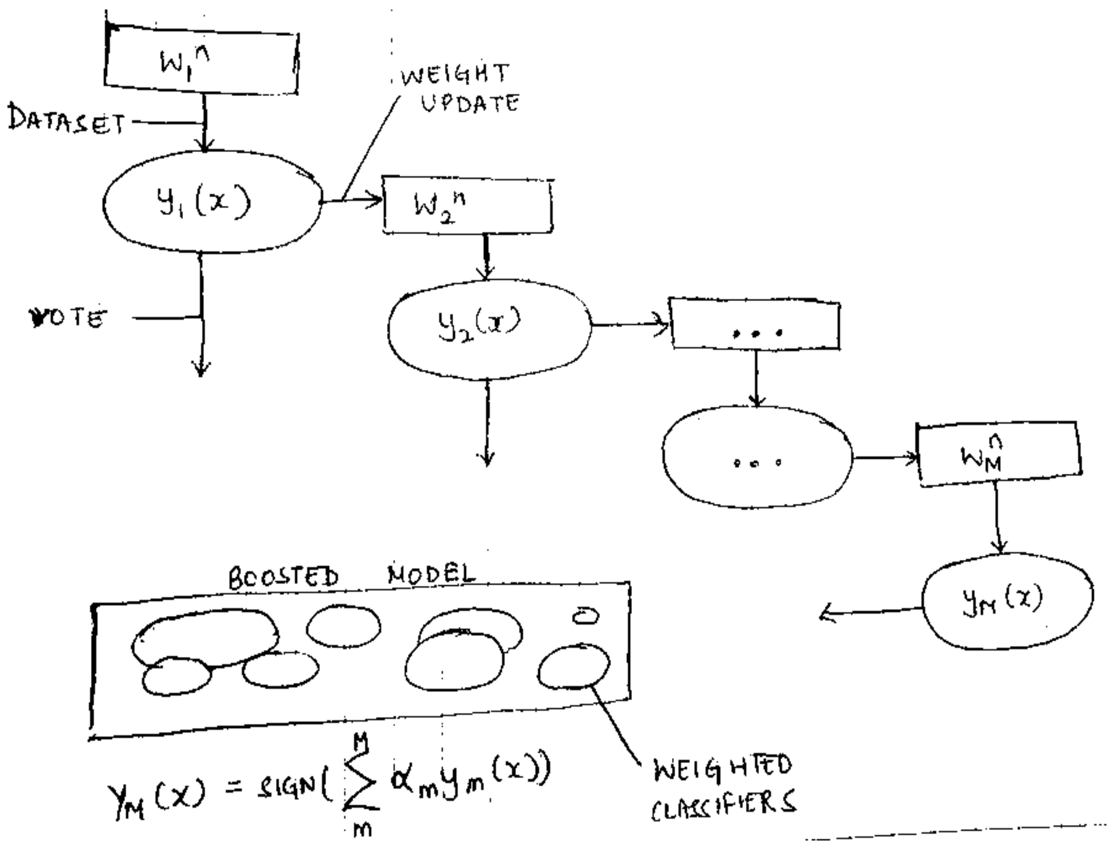
Proposed Solutions and Methods

We have used the following regressors to train our model

XG Boost Classifier
Random Forest
Ensemble learning of Classifiers RF + XGB
Light GBM Regressor
CAT Regressor
Light GBM+ CAT

Why have we used XG Boost Classifier?

XG Boost makes use of a tree learning algorithm in addition to a linear model solver. Due to this, XG Boost is able to achieve parallel computation on a single device which makes this classifier powerful.

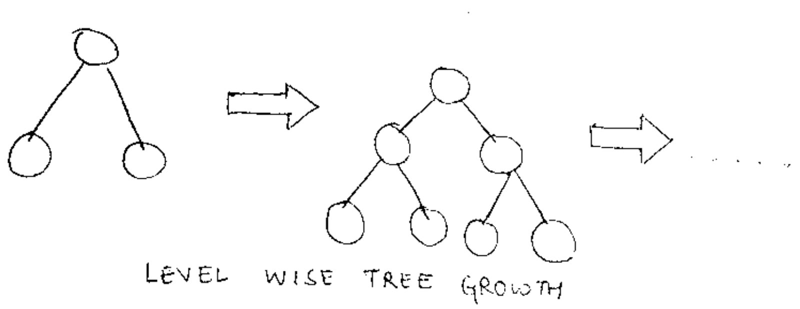


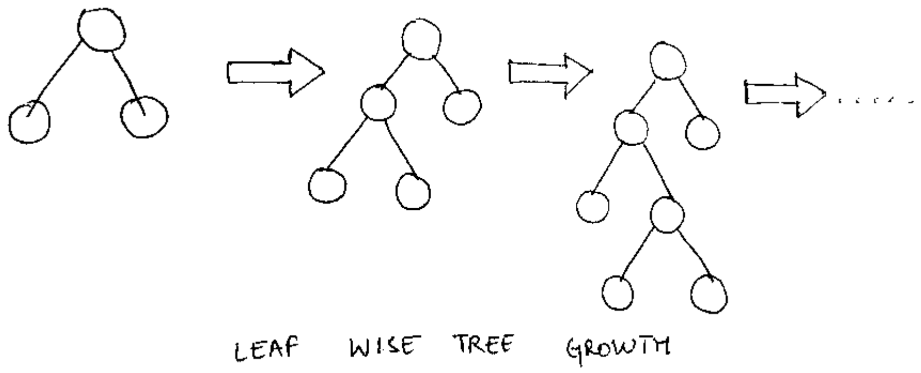
Why have we used CAT + light GBM?

As we already know, CAT Boosting prevents overfitting and reduces bias at a minimal cost of a varience. LightGBM consumes low memory and is faster.

Why have we used Random Forest?

A Random Forest model creates an ensemble of trees which are pretty weak learner so as to construct a relatively stronger learner for the model.



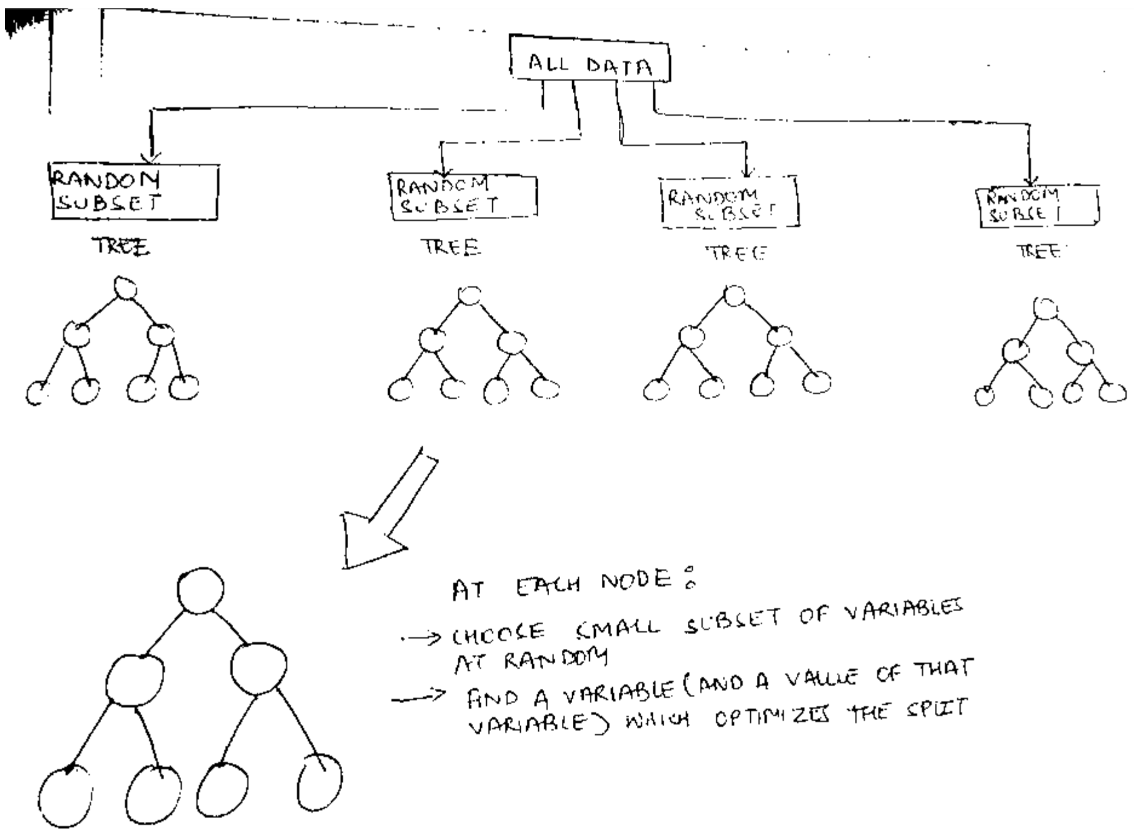


Why have we used Ensemble learning of Classifiers RF + XGB?

Both XG Boost and Random Forest perform random sampling and average across numerous models. This helps them reduce overfitting. Averaging across numerous models helps in fighting overfitting.

Why have we used Light GBM Regressor?

Light GBM Regressor has many features as a classifier. It is a high performance, distributed, and fast boosting model that uses the DT algorithms for classification. Compared to other boosting algorithms that use level split on trees, it makes use of the leaf split so as to provide the best fit for the model. This leaf split is known to reduce the loss considerably well compared to all other boosting method. below diagram explains the difference between leaf and level split.


ALL DATA
RANDOM SUBSET
TREE
RANDOM SUBSET
TREE
RANDOM SUBSET
TREE
RANDOM SUBSET
TREE
AT EACH NODE:
→ CHOOSE SMALL SUBSET OF VARIABLES AT RANDOM
→ FIND A VARIABLE (AND A VALUE OF THAT VARIABLE) WHICH OPTIMIZES THE SPLIT

Why have we used CAT Regressor?

The CAT Regressor makes the model very generalized by reducing overfitting. It also reduces the call for tuning extensive hyper parmeters to our model.

Topic 7

Experimental Results and Analysis

XGBoost Classifier

Parameters used:
'n_estimators':[100, 500, 1000]

Best Estimator:
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1, colsample_bytree=1, gamma=0, learning_rate=0.1, max_delta_step=0, max_depth=3, min_child_weight=1, missing=None, n_estimators=1000, n_jobs=1, nthread=None, objective='binary:logistic', random_state=0, reg_alpha=0, reg_lambda=1, scale_pos_weight=1, seed=None, silent=True, subsample=1)

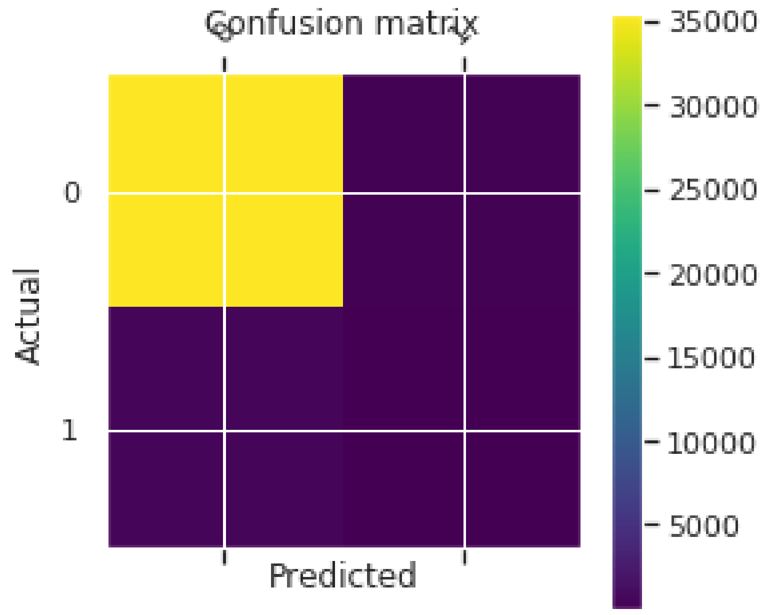
f1 score:
Train data: 0.399274321566256
Test data: 0.33219919579338075

Accuracy Score:
Train data: 0.9890071128005511
Test data: 0.9880540693074238

ROC AUC Score:
Train data: 0.641907814723072
Test data: 0.6170391116659173

Confusion Matrix:

Predicted	0	1
Actual		
0	35507	185
1	459	1



Predicted Customer Revenue:

fullVisitorId	Lgbpred
9999882818693474736	0.693147
9999860794386137754	0.000000
6059383810968229466	0.000000
2376720078563423631	0.000000
000018122977590134	0.000000

Random Forest

Parameters used:
'n_estimators':[100, 500, 1000]

Best Estimator:
RandomForestClassifier(bootstrap=True, class_weight=None, criterion='gini',
max_depth=None, max_features='auto', max_leaf_nodes=None,
min_impurity_decrease=0.0, min_impurity_split=None, min_samples_leaf=1,

```
n_samples_split=2, min_weight_fraction_leaf=0.0, n_estimators=1000, n_jobs=1,
oob_score=False, random_state=None, verbose=0, warm_start=False)
```

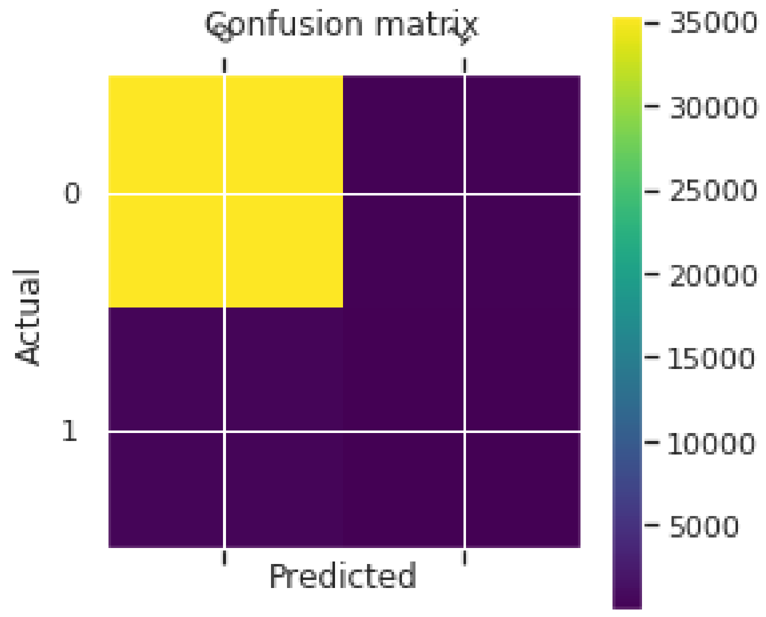
f1 score:
Train data: 1.0
Test data: 0.2831746031746032

Accuracy Score:
Train data: 1.0
Test data: 0.987506293884281

ROC AUC Score:
Train data: 1.0
Test data: 0.5969814709072429

Confusion Matrix

```
Predicted      0          1
Actual
0              35499      193
1              459        1
```



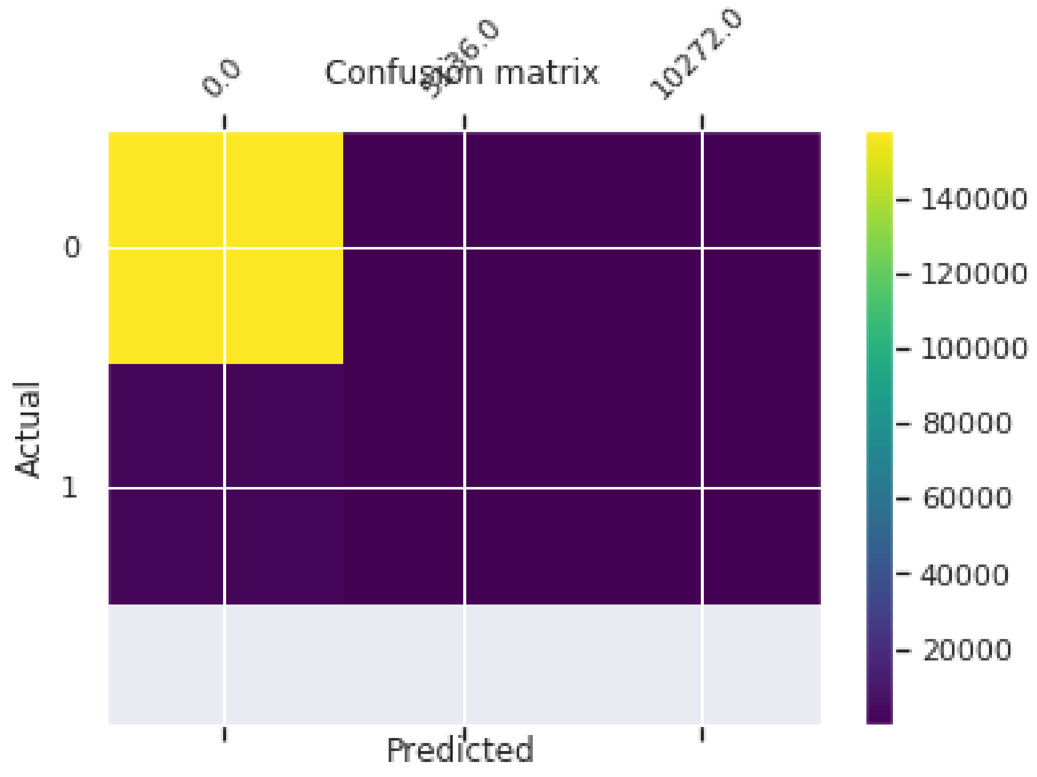
Predicted Customer Revenue:

fullVisitorId	Lgbpred
9999882818693474736	0.693147
9999860794386137754	0.000000
6059383810968229466	0.000000
2376720078563423631	0.000000
000018122977590134	0.693147

Ensemble Learning of classifiers Random Forest+ XGBoost

Confusion Matrix

Predicted	0.0	5136.0	10272.0
Actual			
0	158502	359	175
1	2014	4	5



Predicted Customer Revenue:

fullVisitorId	**Lgbpred**
9999882818693474736	9.237274
9999860794386137754	0.000000
6059383810968229466	0.000000
2376720078563423631	0.000000
000018122977590134	8.544225

Accuracy of XGBoost and Random Forest algorithm is very high on tets data. The model seem to overfit the data. When we combine both the models, revenue increases by a large factor. Above three models do not provide accurate prediction.

LightGBM regressor

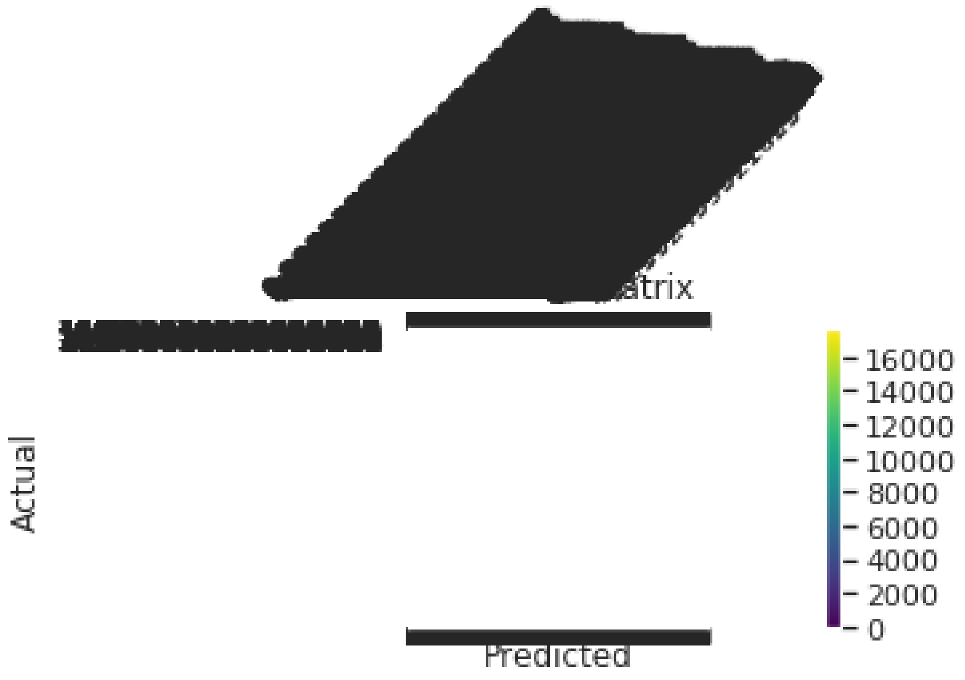
Parameters used:
n_estimators=1000, objective="regression",metric="rmse", num_leaves=31,min_child_samples=100,learning_rate=0.03,bagging_fraction=0.7,feature_fraction=0.5,bagging_frequency=5,bagging_seed=2019, subsample=.9, colsample_bytree=.9, use_best_model=True

Root Mean Squared Error: 1.6178543838098116

Confusion Matrix:

```
Predicted  0.000000e+00  2.033870e-07     ...      1.553394e+01  1.576198e+01
Actual                                 ...
0.000000        17690           1     ...               1           1
14.503646           1           0    ...               0           0
14.910784           1           0    ...               0           0
15.065413           0           0    ...               0           0
15.068274           0           0    ...               0           0
15.196793           0           0    ...               0           0
...             ...         ...     ...             ...         ...
19.895604           1           0    ...               0           0
19.935610           0           0    ...               0           0
20.011751           0           0    ...               0           0
20.049333           1           0    ...               0           0
20.630492           0           0    ...               0           0
20.640058           1           0    ...               0           0
20.647680           1           0    ...               0           0
```

```
20.705845        1          0     ...           0          0
20.742480        0          0     ...           0          0
20.797445        1          0     ...           0          0
20.816702        1          0     ...           0          0
20.971477        0          0     ...           0          0
20.981777        1          0     ...           0          0
21.030015        1          0     ...           0          0
21.177521        1          0     ...           0          0
21.253058        0          0     ...           0          0
22.427650        0          0     ...           0          0

[376 rows x 11396 columns]
```



Predicted Customer Revenue:

fullVisitorId	**Lgbpred**
6167871330617112363	0.000000
0643697640977915618	0.000127
6059383810968229466	0.000289
2376720078563423631	0.000000

2314544520795440038	0.000000

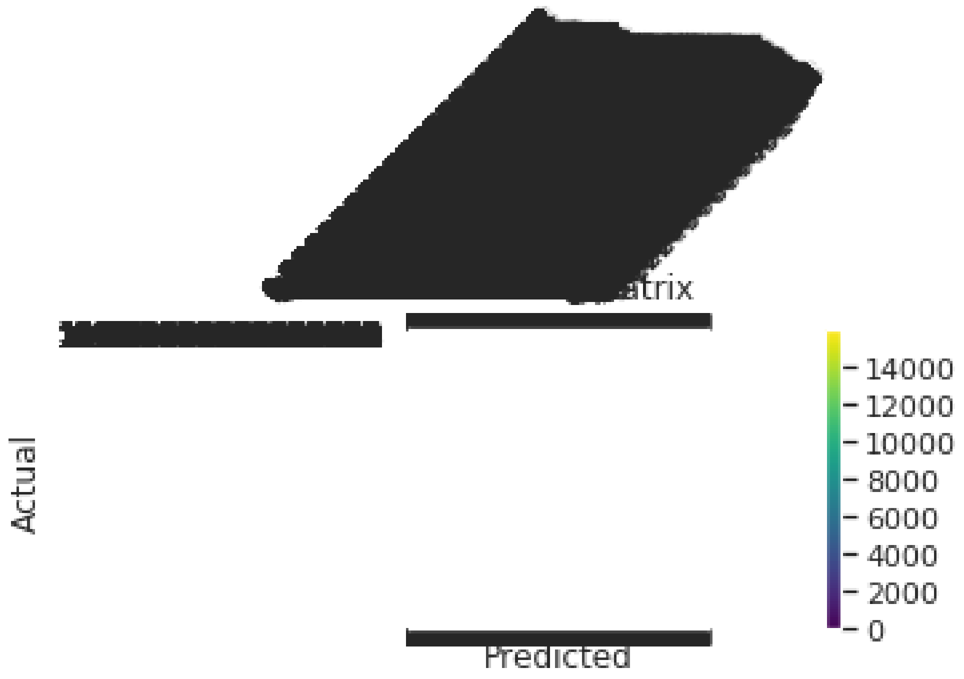
CAT regression model

Parameters used:
Iterations:1000, learning_rate=0.2, depth=5, random_seed=2019

Root Mean Squared Error:1.6274764445009735

Confusion Matrix

```
Predicted  0.000000   0.000002    ...     17.055803  18.407936
Actual                           ...
0.000000       15945       1    ...            1         1
16.210675        1         0    ...           0         0
16.212496        0         0    ...           0         0
16.219749        0         0    ...           0         0
16.229637        1         0    ...           0         0
16.297078        1         0    ...           0         0
16.340439        0         0    ...           0         0
16.378920        1         0    ...           0         0
16.379690        1         0    ...           0         0
16.393452        0         0    ...           0         0
...             ...       ...   ...          ...       ...
19.895604        0         0    ...           0         0
19.935610        1         0    ...           0         0
20.981777        0         0    ...           0         0
21.030015        0         0    ...           0         0
21.177521        1         0    ...           0         0
21.253058        0         0    ...           0         0
22.427650        0         0    ...           0         0

[376 rows x 20199 columns]
```



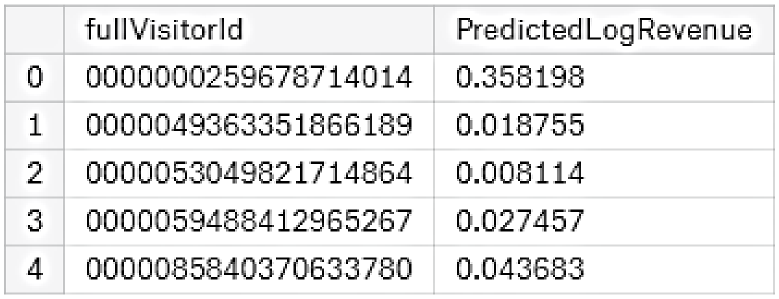
Predicted Customer Revenue:

fullVisitorId	catpred
6167871330617112363	0.003411
0643697640977915618	0.001367
6059383810968229466	0.004039
2376720078563423631	0.000000
2314544520795440038	0.023078

Light GBM + CAT

Root Mean Squared Error:1.6193379375212846

Predicted Revenue per customer:



	fullVisitorId	PredictedLogRevenue
0	0000000259678714014	0.358198
1	0000049363351866189	0.018755
2	0000053049821714864	0.008114
3	0000059488412965267	0.027457
4	0000085840370633780	0.043683

LightGBM, CAT and LightGBM + CAT give less Root Mean Squared Error than previous three results.

Topic 8

Conclusion

What are we predicting?

We are predicting the natural log of the sum of all transactions per user. Once the data is updated, as noted above, this will be for all users in test_v1.csv for December 1st, 2018 to January 31st, 2019. For every user in the test set, the target is:

$$y_{user} = \sum_{i=1}^{n} transaction_{user_i}$$

$$target_{user} = \ln(y_{user} + 1)$$

Note that the dataset does NOT contain data for December 1st 2018 to January 31st 2019. We are identifying unique fullVisitorIds in the provided test_v1.csv and making predictions for them for those unseen months.

Predicted Customer Revenue (Top 5 Customers):

	Full Visitor id	PredictedLogRevenue
1	0000000259678714014	0.358198
2	0000049363351866189	0.018755
3	0000053049821714864	0.008114
4	0000059488412965267	0.027457
5	0000085840370633780	0.043683

Root mean squared error (RMSE) is used to evaluate our models.

$$RMSE = \sqrt[2]{\frac{1}{n}\sum_{i=1}^{n}(y_i - \check{y}_i)^2}$$

Where y hat is the natural log of the predicted revenue for a customer and y is the natural log of the actual summed revenue value plus one.

Models	**RMSE**
XGBoost Classifier	**1.68**
Random Forest	**1.67**

Random Forest + XG BoostClassifier	**1.64**
LightGBM regressor	**1.618**
CAT regression	**1.623**
LightGBM + CAT	**1.613**

**Note: for experiments using classifier models, we implemented classifier data and assign non-null value predicted by mean revenue.*

When we combined LightGBM + CAT, we got the minimum RMSE error.
This model is predicting the revenue more accurately.

Topic 9

Contribution of Team Members

Project idea proposal:	Hieu, Rutali, Pallavi
Understanding the project:	Hieu, Rutali, Pallavi
Defining the scope of the project:	Hieu, Rutali, Pallavi
Research:	Hieu, Rutali, Pallavi
Project Plan:	Hieu, Rutali, Pallavi
Data Preprocessing and Analysis:	Hieu, Rutali, Pallavi
Project Report:	Hieu, Rutali, Pallavi
Running the code:	Hieu, Rutali, Pallavi

Topic 10

References

- http://blog.citizennet.com/blog/2012/11/10/random-forests-ensembles-and-performance-metrics
- https://www.python-course.eu/Boosting.php
- https://www.analyticsvidhya.com/blog/2017/06/which-algorithm-takes-the-crown-light-gbm-vs-xgboost/
- https://www.kaggle.com/saodem74/gacrp/data
- https://www.datacamp.com/courses/preprocessing-for-machine-learning-in-python
- https://www.kaggle.com/gpreda/google-analytics-customer-revenue-extensive-eda
- For Map usage - https://www.kaggle.com/gpreda/google-analytics-customer-revenue-extensive-eda