Terragon Project Report - Increasing The Click Through Rate Of Campaigns Using Predictive Analysis

May 2020

Executive Summary

This document presents a data science capstone project of the Information and Data Analytics Foundation (IDAF) for the Terragon Group. The project is about building a predictive model to forecast the likelihood of a customer to click an ad sent. At the initiation phase, Terragon's historical data of 66,739 customers with 31 variables were used for data analyses and model building.

Following the data science pipeline, after the business understanding and a clear business objective, we performed some exploratory data analysis for data understanding, then data preparation with visualisations; checking correlations, performing univariate and bivariate analysis, and dimensionality reduction before fitting the model into a Random forest algorithm - the optimal predictive model with precision score of 76.3% and recall score of 17.3%.

A flask API endpoint was built to access the model and it was tested before being deployed on Docker to serve predictions in real-time.

Business Challenge

- Our goal is to calculate the propensity of a profile to click on an ad served by training a machine learning algorithm on Tarragon's historical data of profiles who were previously served an ad.
- Achieve a minimum precision score of 75%,
- Build and test a machine learning flask API that handles bad request without breaking,
- Deploy the model on Docker to serve predictions in real-time.

Data Understanding

The initial exploration of the data began with some summary and descriptive statistics on variables with numerical values.

```
data.describe()
```

	location_city	spend_total	spend_vas	spend_voice	spend_data	sms_cost	xtra_data_talk_rev	customer_class	age
count	0.0	6.551000e+04	55455.000000	55455.000000	55455.000000	55455.000000	55455.000000	65509.000000	54708.000000
mean	NaN	3.231714e+03	239.270517	1583.266644	357.811002	51.925997	44.343161	60.177594	37.206186
std	NaN	1.339642e+04	1730.038459	2470.361416	861.130366	179.041100	488.071740	67.210053	11.366520
min	NaN	0.000000e+00	0.000000	0.000000	0.000000	0.000000	0.000000	20.000000	-71.000000
25%	NaN	2.999600e+02	0.000000	90.067500	0.000000	0.000000	0.000000	44.000000	29.000000
50%	NaN	1.522085e+03	20.000000	740.155000	0.000000	4.000000	0.000000	46.000000	36.000000
75%	NaN	3.892408e+03	220.500000	2067.697500	300.000000	48.000000	0.000000	46.000000	44.000000
max	NaN	2.037584e+06	352057.000000	69728.750000	24156.839700	27160.000000	17200.000000	1051.000000	1051.000000

Label

'**click**' - This is our target variable (a binary variable) for each row of the test data set.

Features

Demographic Information

'**gender**' (categorical) – A categorical variable with no ordering indicating the gender of the customer targeted where 'M' indicates male and 'F' indicates female.

'**age**' (integer) – An integer value that shows the age of the customer.

Behavioural Information

'**location_region**' – A categorical variable with no ordering indicating the geo-political zones in Nigeria where the customer resides.

'**location_state**' – A categorical variable with no ordering indicating the states in Nigeria where the customer resides.

'**customer_value**' – A categorical variable with hierarchy indicating the value placed on the customer based on their transactional information, having 'top' as the highest value and 'low' as the lowest value.

'**device_type**' (categorical) – A categorical variable with no ordering indicating the type of device used by the customer.

'**device_manufacturer**' (categorical) – A categorical variable with no ordering indicating the manufacturer of the device being used by the customer.

'spend_total' (integer) – A measure of the total amount spent (in naira) by the customer.

'spend_vas' (integer) – A numerical variable.

'spend_voice' (integer) – A measure of the total amount spent (in naira) by the customer on voice calls.

'spend_data' (integer) – A measure of the total amount (in naira) spent by the customer on data.

'xtra_data_talk_rev' (integer) – Measures amount of extra data used by the customer.

'customer_class' (integer**) –** Contains different numerical categories on customers' transactions.

'@timestamp' (date) – Date variable with exact dates and time when ads was sent.

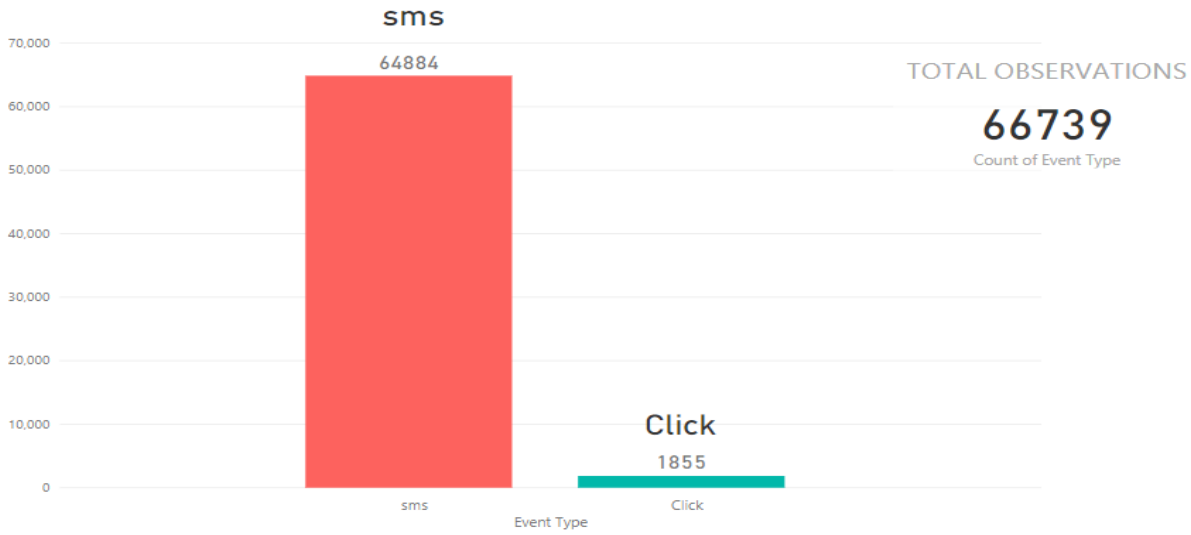
Exploratory Data Analysis

Univariate Analysis

We performed univariate analysis on the entire variable to examine them individually. For categorical features, we used bar charts to calculate the frequency of each category in a particular variable. For numerical features, probability density plots and box plots were used to look at the distribution of the variable.

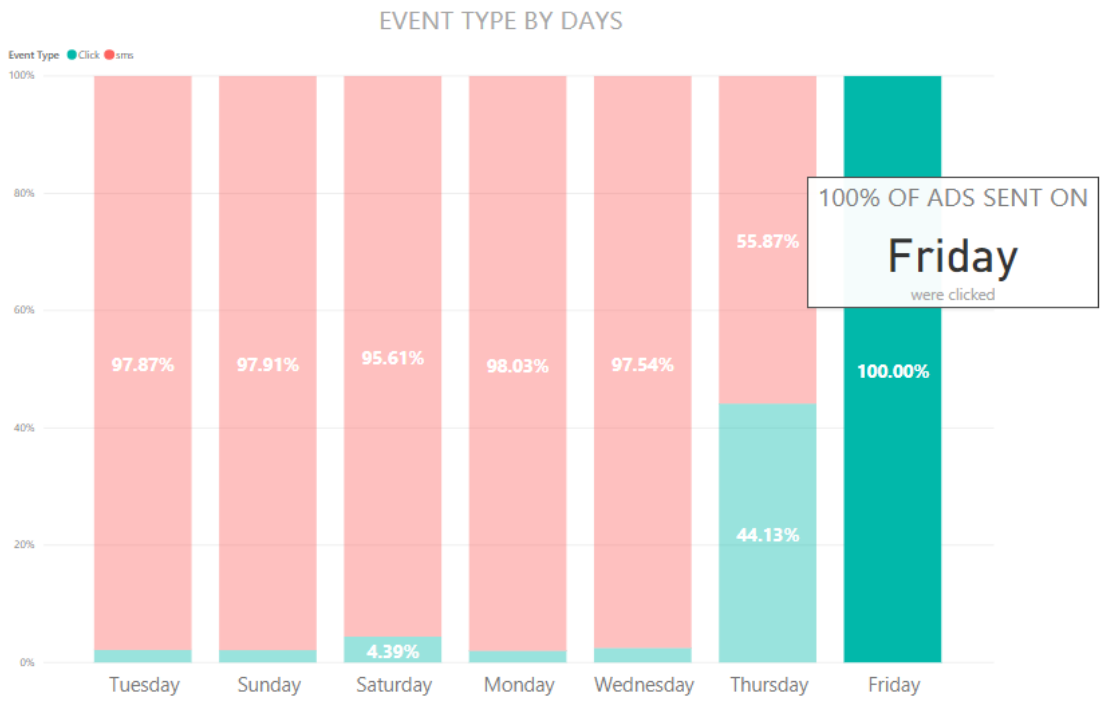
Target

Since '*event_type*' is the class of interest in this analysis, it is noted that there was a high imbalance between the number of *'clicks'* (the positive class) and *'sms'* (the negative class) with each of the classes having 97.22% and 2.78% respectively.



Date Variable

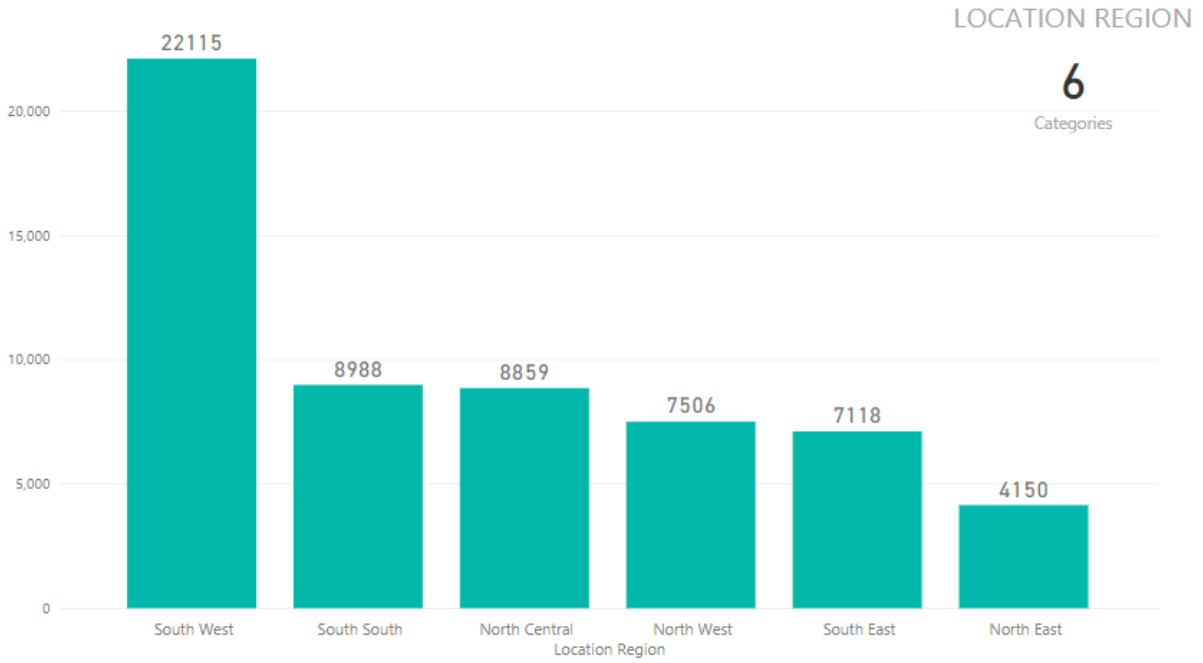
From the '@timestamp' variable, information like week days, hours of the day, months and years were extracted from the data string, and hence the visualization below;

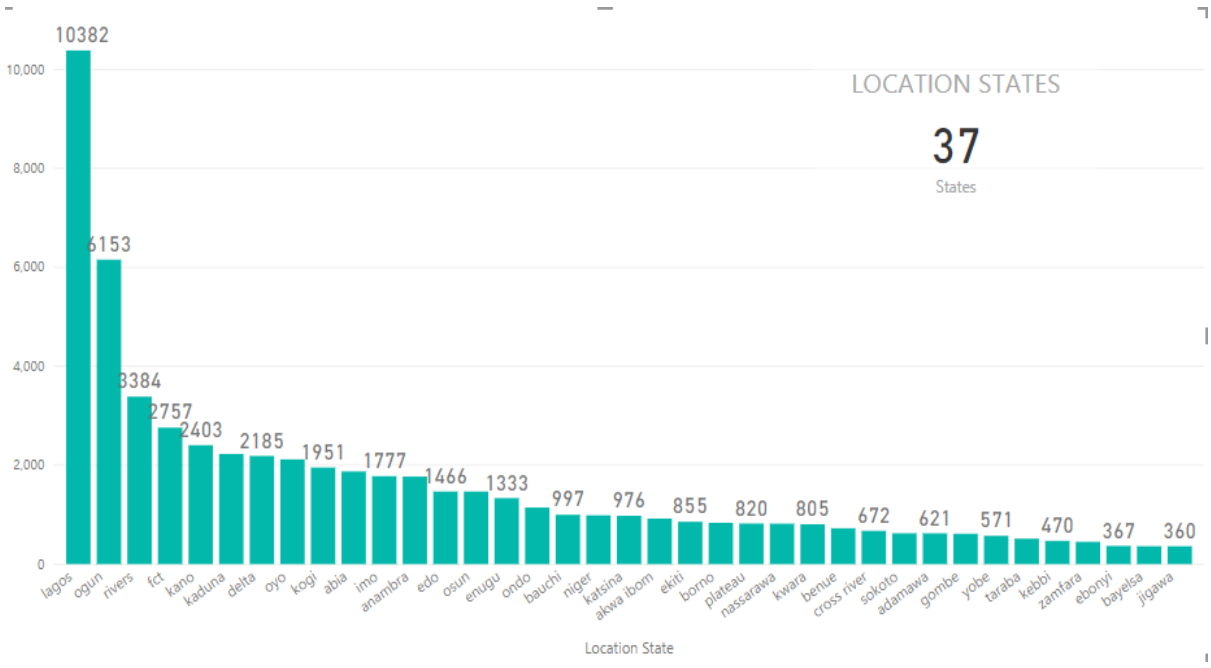


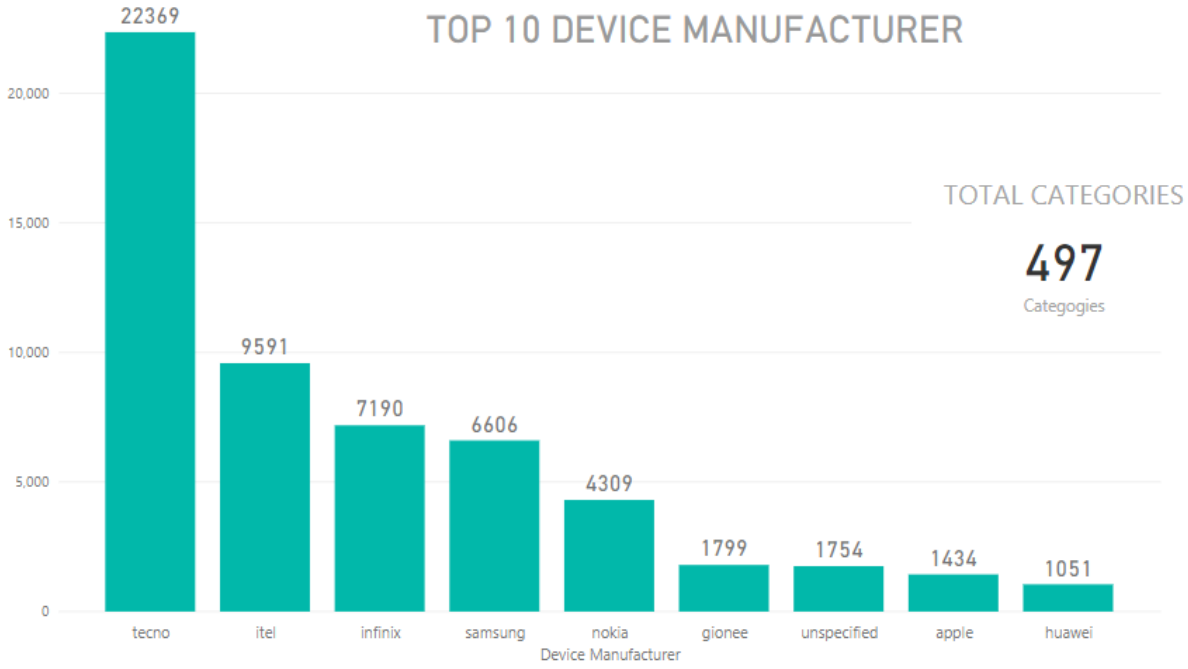
In can be inferred that all the 129 ads (100%) received on Fridays were clicked and about 44% of the ads received on Thursdays were clicked. This implies that there is a high conversion rate close to weekends. The worst days were Mondays followed by Sundays.

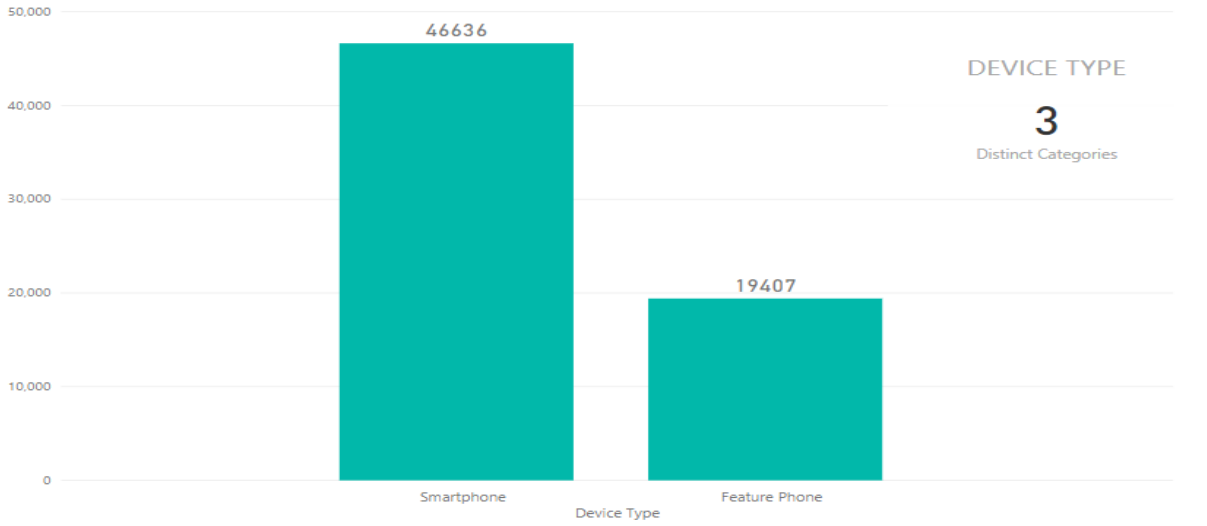
Categorical Variable

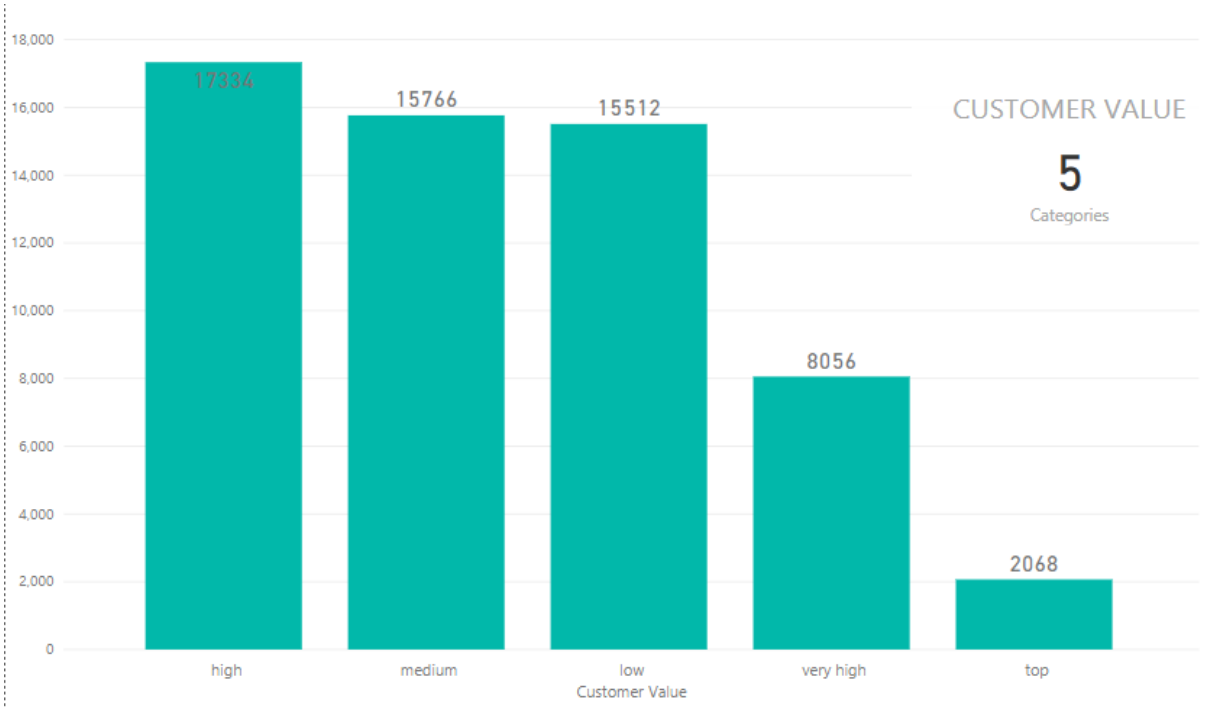
Analysis was performed on all categorical variables however, only the features used in the final modelling has been presented here. These features *are 'location_region', 'location_state', 'customer_value', 'gender', 'device_type',* and *'device_manufacturer'*

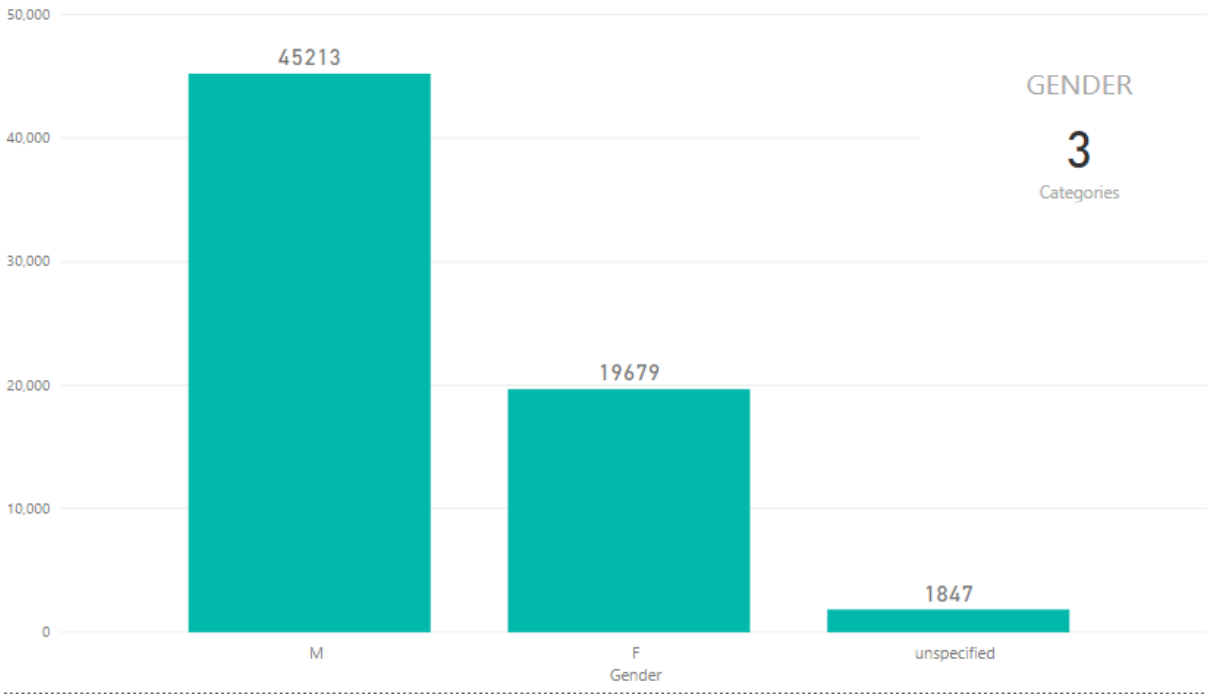

LOCATION REGION
6
Categories
22115
8988
8859
7506
7118
4150
20,000
15,000
10,000
5,000
0
South West
South South
North Central
North West
South East
North East
Location Region


LOCATION STATES
37
States
10382
6153
3384
2757
2403
2185
1951
1777
1466
1333
997
976
855
820
805
672
621
571
470
367
360
10,000
8,000
6,000
4,000
2,000
0
lagos
ogun
rivers
fct
kano
kaduna
delta
oyo
kogi
abia
imo
anambra
edo
osun
enugu
ondo
bauchi
niger
katsina
akwa ibom
ekiti
borno
plateau
nassarawa
kwara
benue
cross river
sokoto
adamawa
gombe
yobe
taraba
kebbi
zamfara
ebonyi
bayelsa
jigawa
Location State


TOP 10 DEVICE MANUFACTURER
22369
9591
7190
6606
4309
1799
1754
1434
1051
20,000
15,000
10,000
5,000
0
tecno
itel
infinix
samsung
nokia
gionee
unspecified
apple
huawei
Device Manufacturer
TOTAL CATEGORIES
497
Categogies


50,000
40,000
30,000
20,000
10,000
0
46636
19407
Smartphone
Feature Phone
Device Type
DEVICE TYPE
3
Distinct Categories

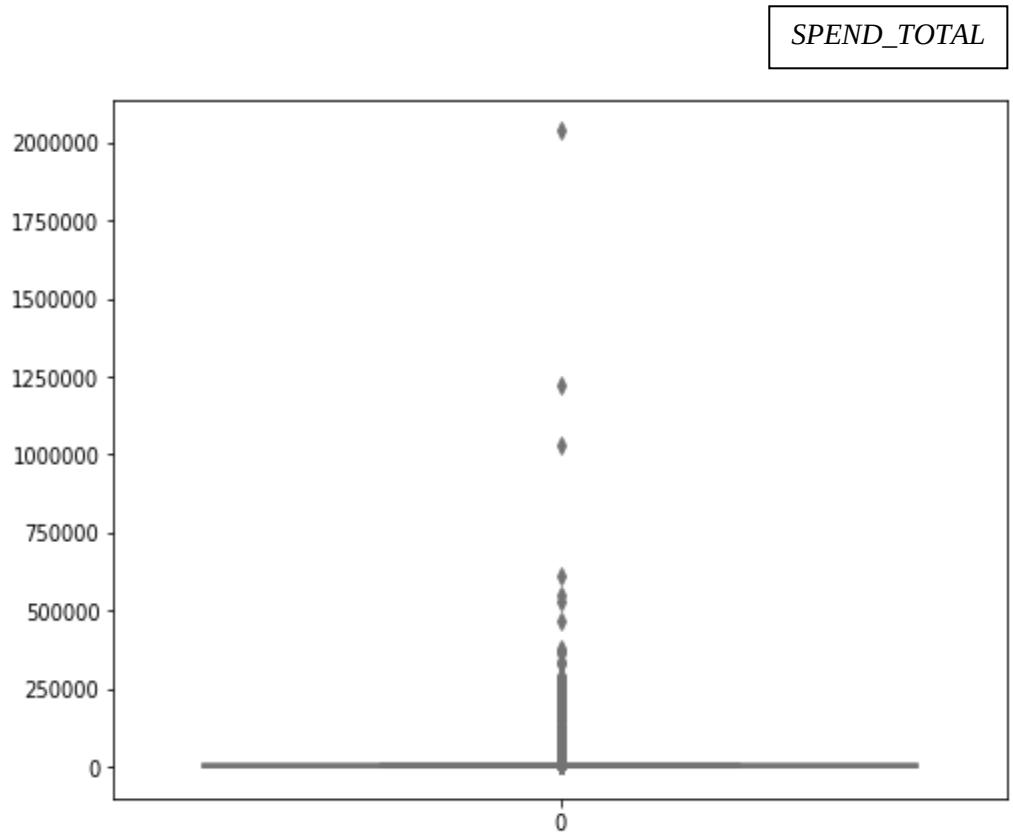

18,000
16,000
14,000
12,000
10,000
8,000
6,000
4,000
2,000
0
17334
15766
15512
8056
2068
high
medium
low
very high
top
Customer Value
CUSTOMER VALUE
5
Categories


50,000
40,000
30,000
20,000
10,000
0
45213
19679
1847
M
F
unspecified
Gender
GENDER
3
Categories

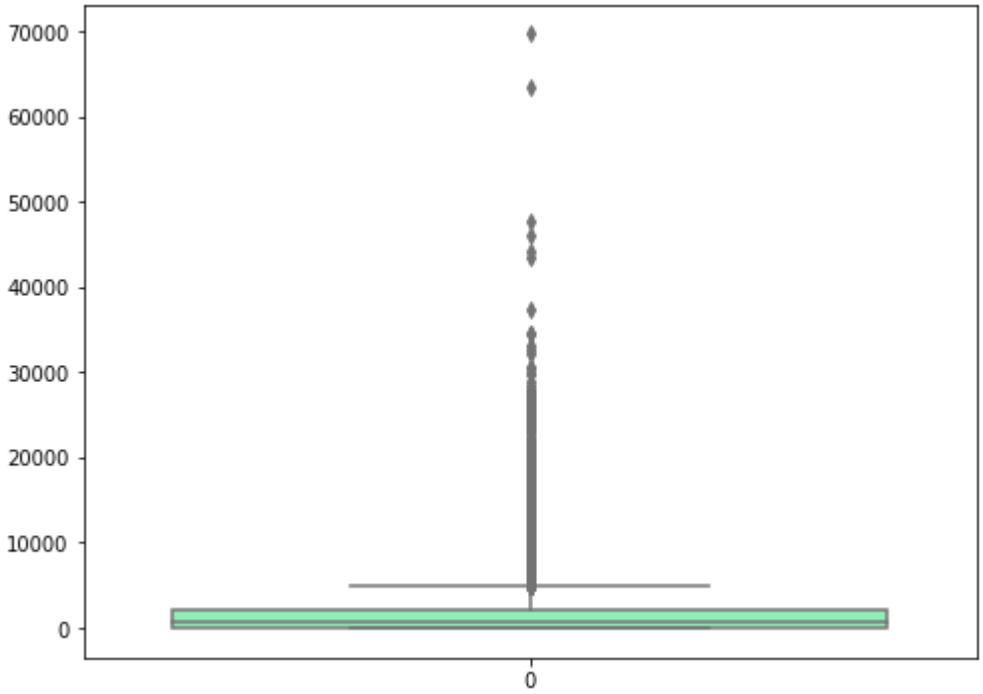
It can be inferred from the visualizations above that all classes in the categorical variables are not equally distributed. 0.8% of customer-value top-category clicked, 20% of customer of age 16 clicked and majority of the categorical variables are having imbalanced categories.

Numerical Variable

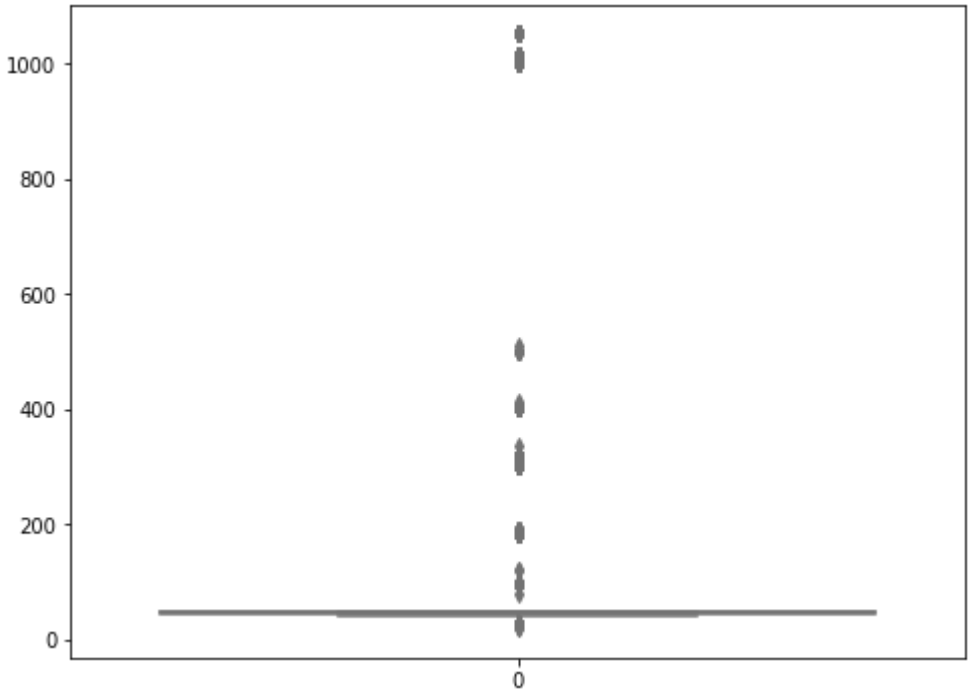
Data visualization was performed on all numerical variables using a box plot. These features are: *'spend_total', 'spend_vas', 'spend_voice', 'spend_data', 'xtra_data_talk_rev', 'customer_class', 'age'*.



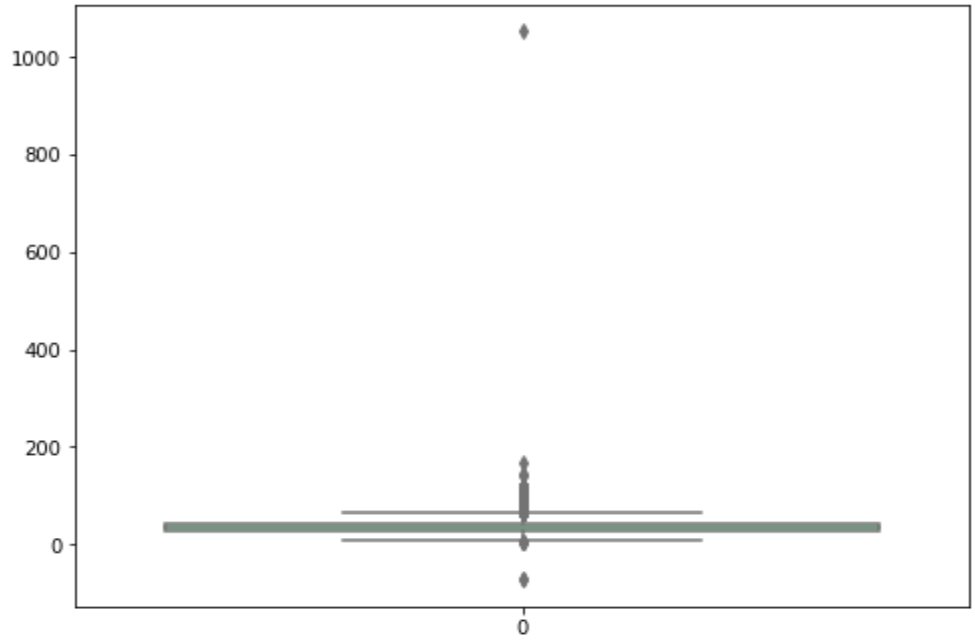
SPEND_VOICE



CUSTOMER CLASS



AGE

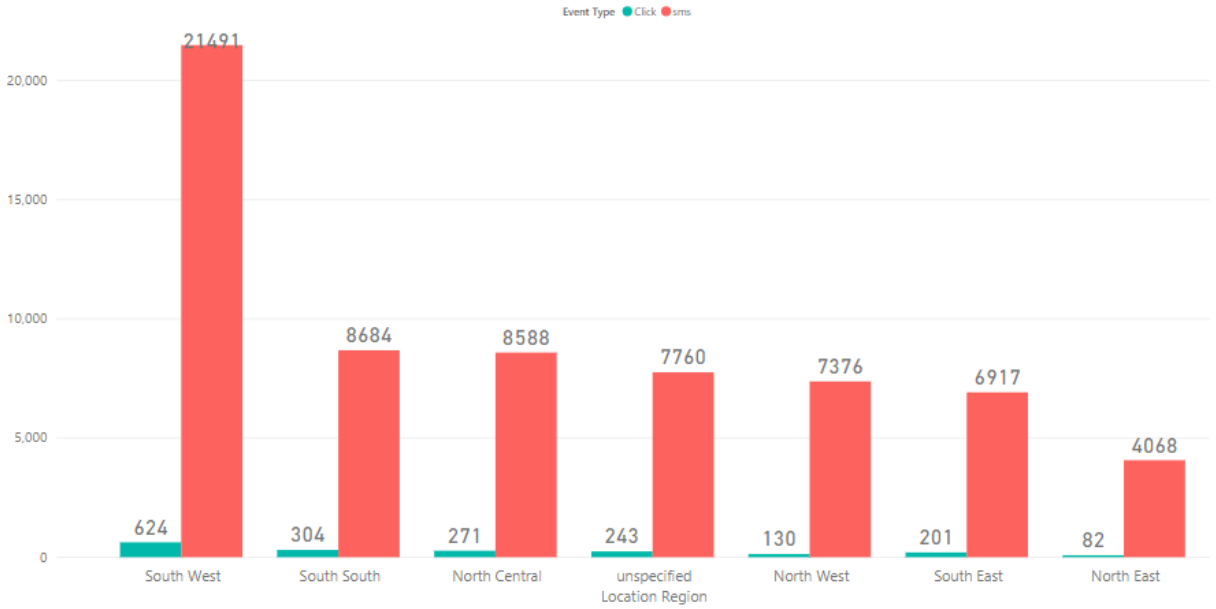


From the above summary statistics, variables like '*spent_vas'*, '*spent_voice'* ,*'spent_data'*, '*sms_cost'*, '*xtra_data_talk_rev'* and '*age*' are not normally distributed i.e are skewed due to the presence of outliers. It can be inferred from the above visualization that the presence of outliers have affected the distribution of the data.

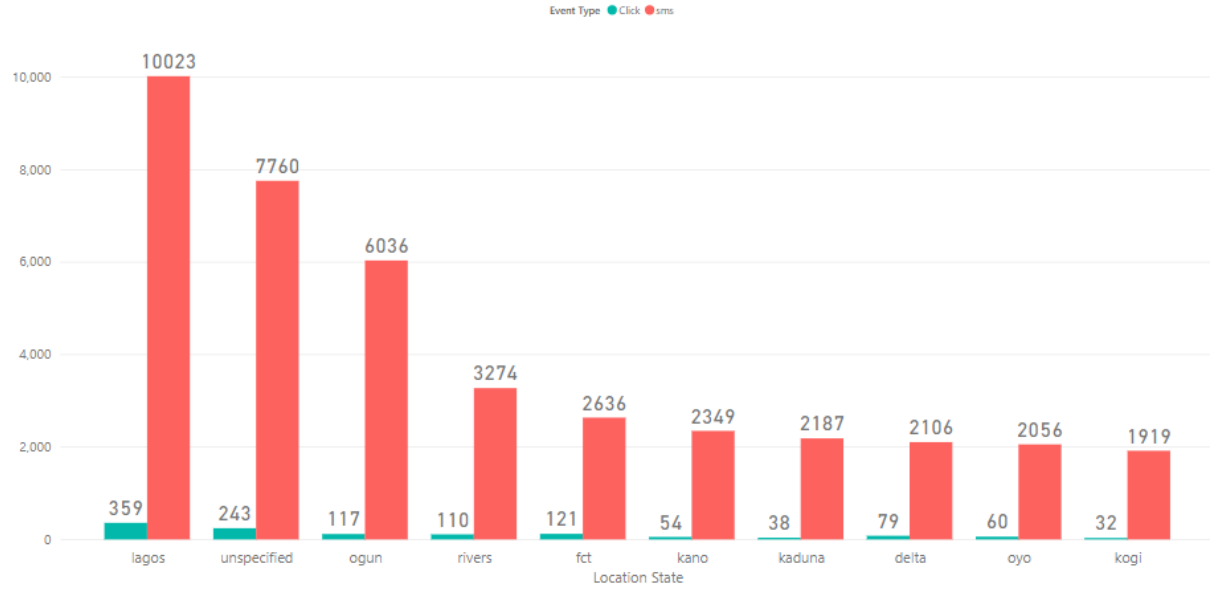
Multivariate Analysis

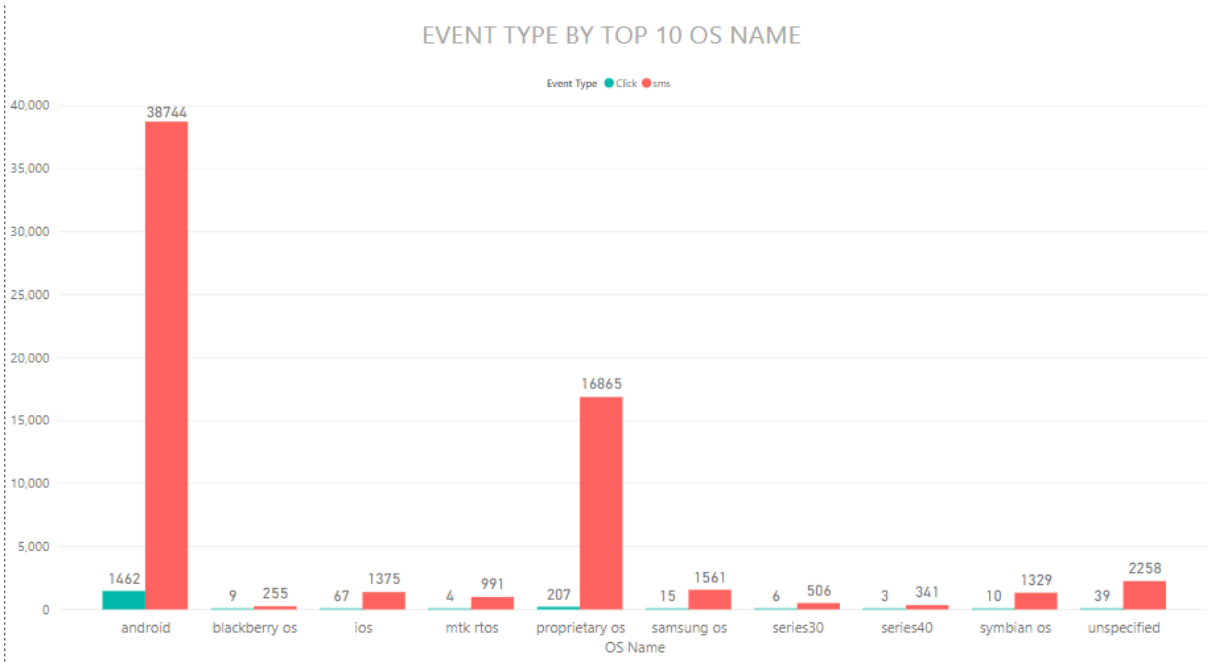
After looking at every variable individually in univariate analysis, we now visualize two or more variables with respect to the target variable.

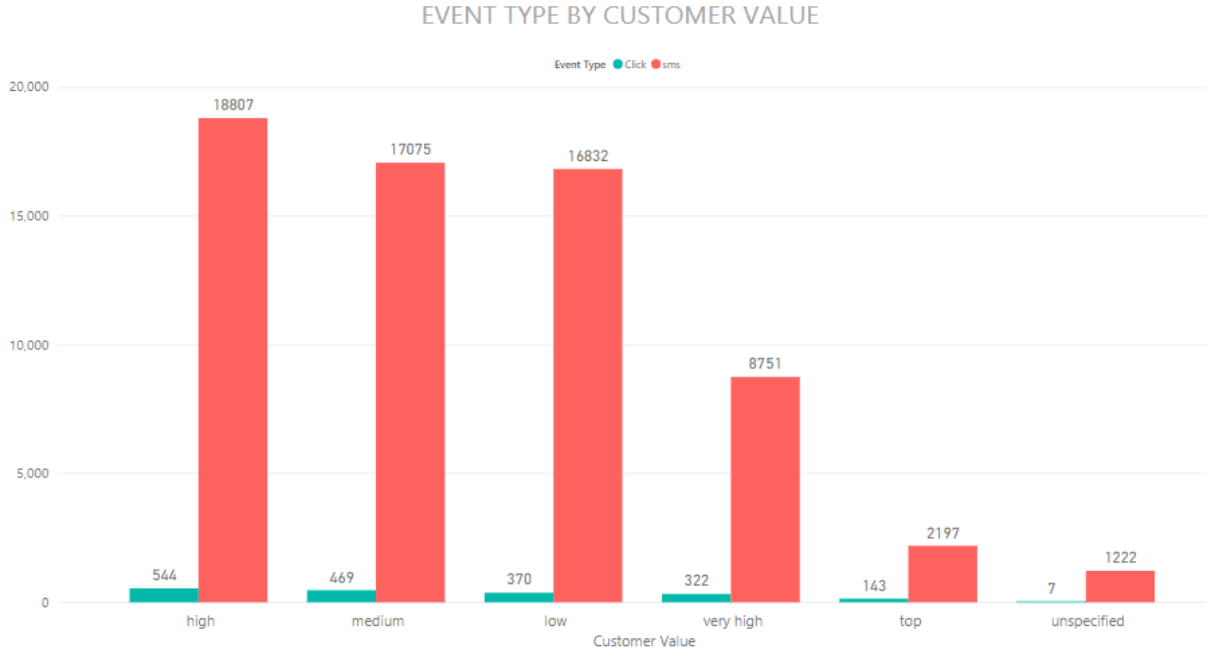
EVENT TYPE BY LOCATION REGION

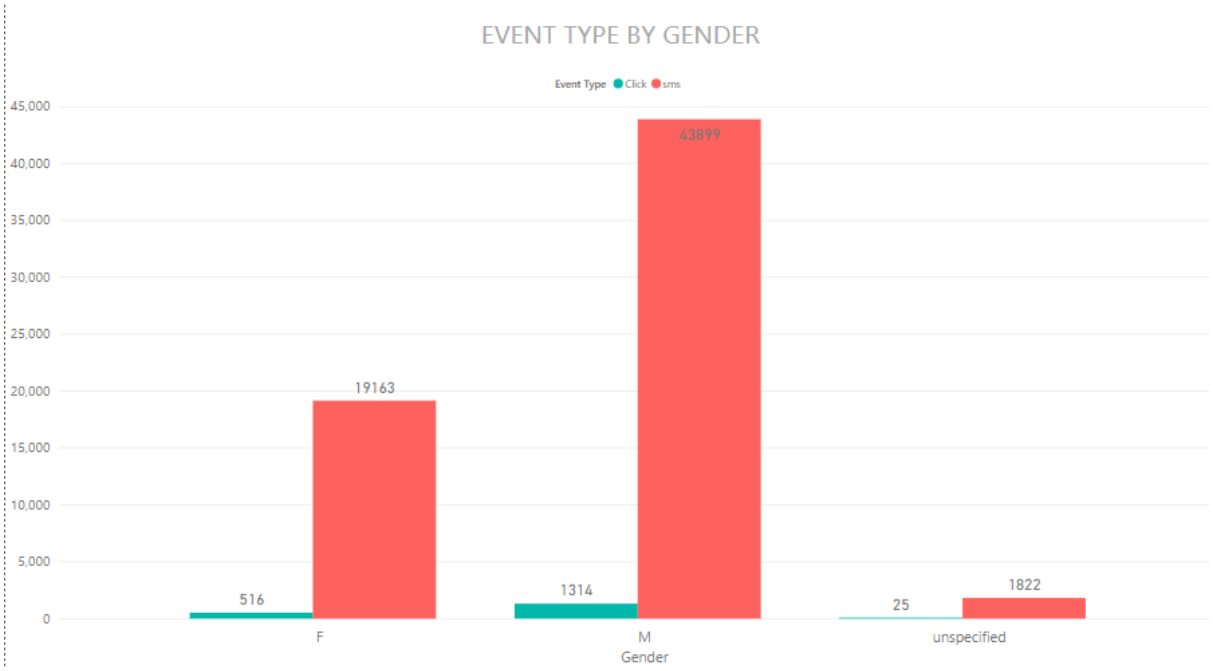


EVENT TYPE BY TOP 10 LOCATION STATES

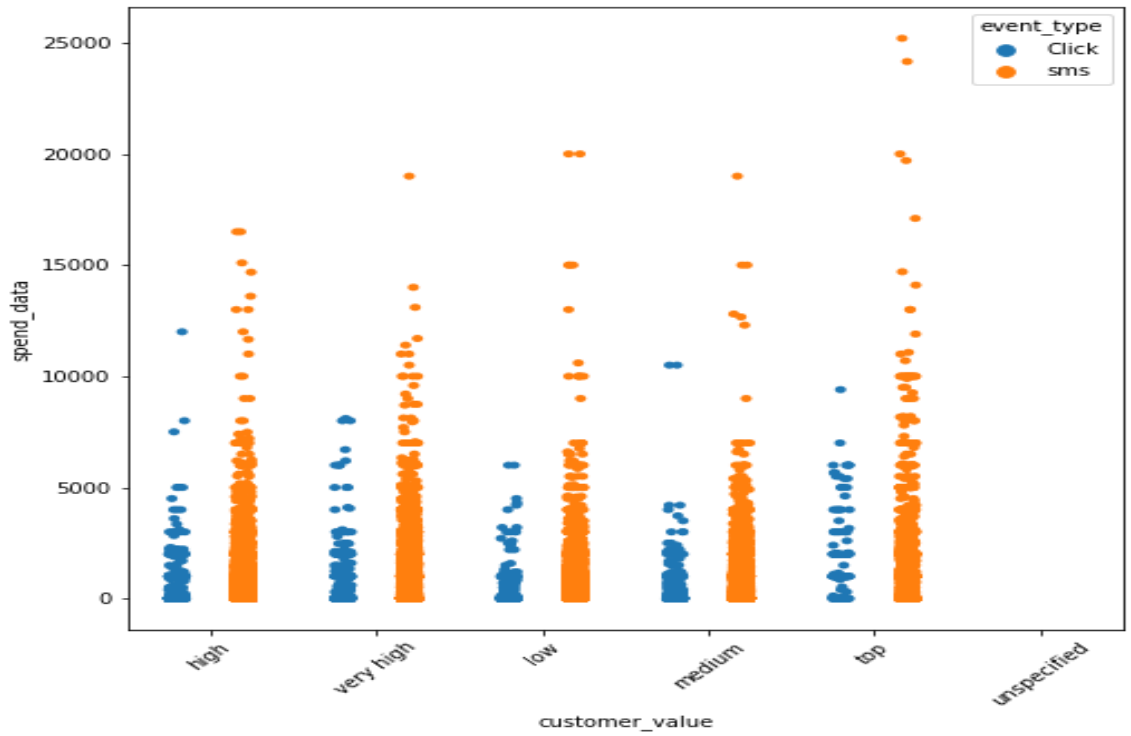


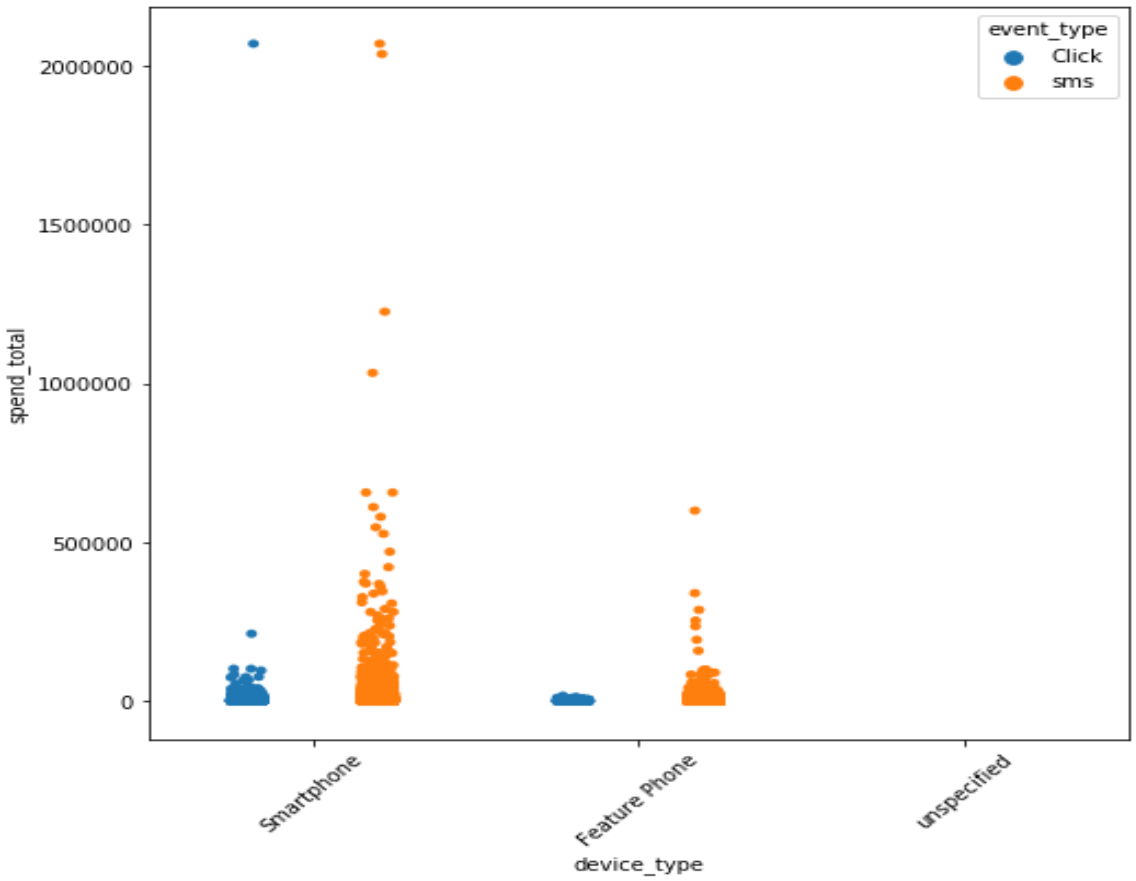

EVENT TYPE BY TOP 10 OS NAME
Event Type Click sms
40,000
35,000
30,000
25,000
20,000
15,000
10,000
5,000
0
38744
1462
9
255
67
1375
4
991
16865
207
15
1561
6
506
3
341
10
1329
39
2258
android
blackberry os
ios
mtk rtos
proprietary os
samsung os
series30
series40
symbian os
unspecified
OS Name

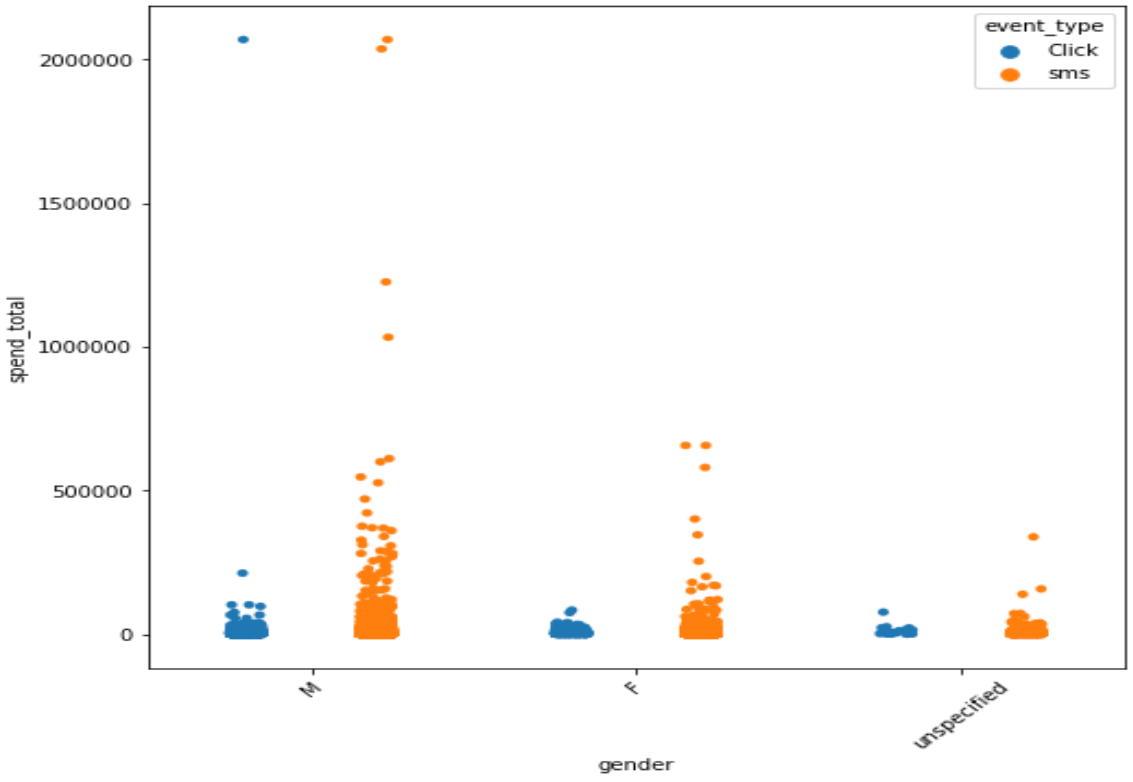

EVENT TYPE BY CUSTOMER VALUE
Event Type Click sms
20,000
15,000
10,000
5,000
0
18807
544
17075
469
16832
370
8751
322
2197
143
1222
7
high
medium
low
very high
top
unspecified
Customer Value

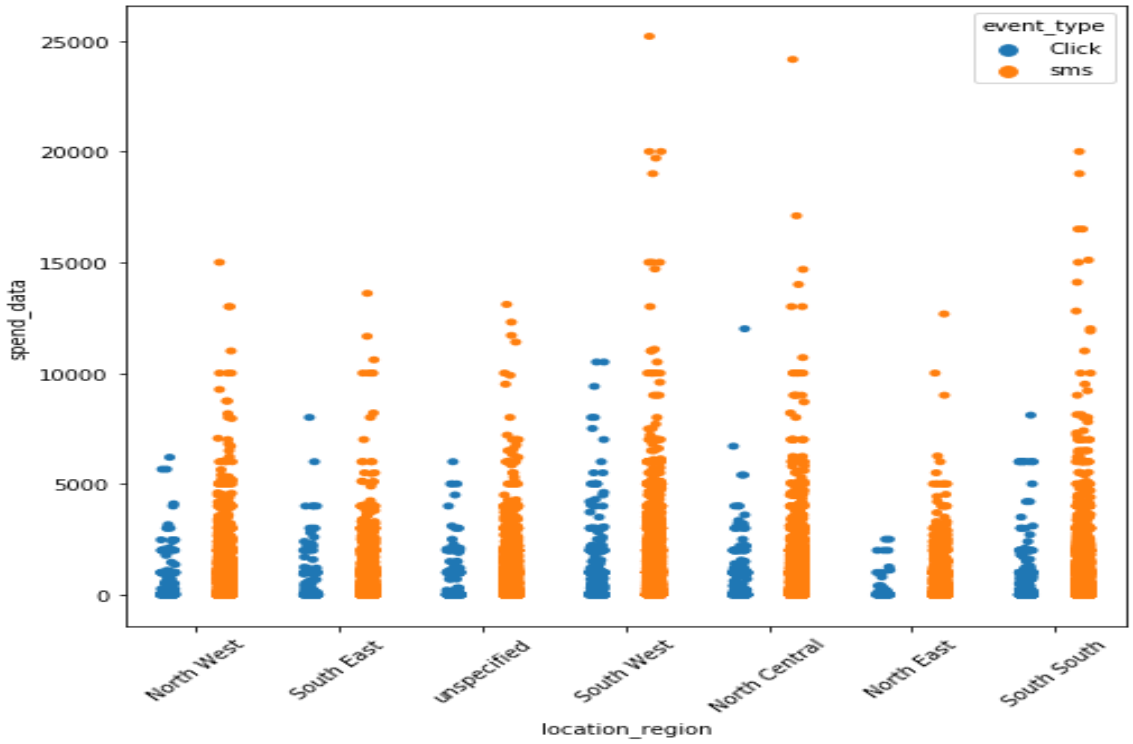


From the above visualizations, we have gained insight into categories that are very informative in predicting the target. Also some categories were poor in predicting due to lack of enough data on those categories.




event_type
Click
sms
2000000
1500000
1000000
500000
0
spend_total
Smartphone
Feature Phone
unspecified
device_type


event_type
Click
sms
2000000
1500000
1000000
500000
0
spend_total
M
F
unspecified
gender



Data Preparation and Pre-processing

After exploring the data and having a good knowledge of what the data entails we carried out some data cleaning by dropping some unique identities; addressing missing values and outliers which were spotted during data visualization

Handling missing values

Null values in the numerical variables were replaced with either zero (0), median or mode of the variable. While missing values in the categorical variables were given a new category named *'unspecified'* and a variable like *'location_city'* was dropped since the entire column was empty i.e 'Nan'.

Outlier Removal

Ages greater than 100 or Negative (-ve) were considered to be outliers and were replace with 100 and 12years respectively. We assumed that no customer who received an ad in the dataset was below the age of 12 or above age 100.

Data Transformation

Major transformation considered on the numerical variables were *rescaling* variables with different scales and *normalization* on variable data with large number of zeros (0) i.e sparse data.

Encoding

The various features in the data were encoded using various techniques. Domain knowledge of each feature was the most important factor considered. We used two techniques to encode each categorical variable of the dataset: *'one hot encoding'* and *'ordinal encoding'*. Variables which had a hierarchy was encoded with 'ordinal encoding' while variables which do not have a hierarchy was encoded using 'one hot encoder' Each variable encoding is explained below:

Variables encoded using ordinal encoding

'event_type' – this variable which is our target variable was encoded using label encoder. with 'sms' as **'0'** and 'click' as '**1'**.

'customer_value' **–** this variable was encoded using ordinal encoder with the highest category 'very high' as '5', 'high' as 4, 'medium' as 3, 'low' as 2 and the lowest label 'very low' as 1.

Variables encoded using one hot encoding

'gender' **–** we had each label mapped to a binary vector.

'os_name' **–** we had some labels of this feature replaced with *'others'* due to their sparse appearance among the observations. *'android'* and *'propriety os'* labels were retained while other labels were replaced with 'others'. Each label was then mapped to a binary vector.

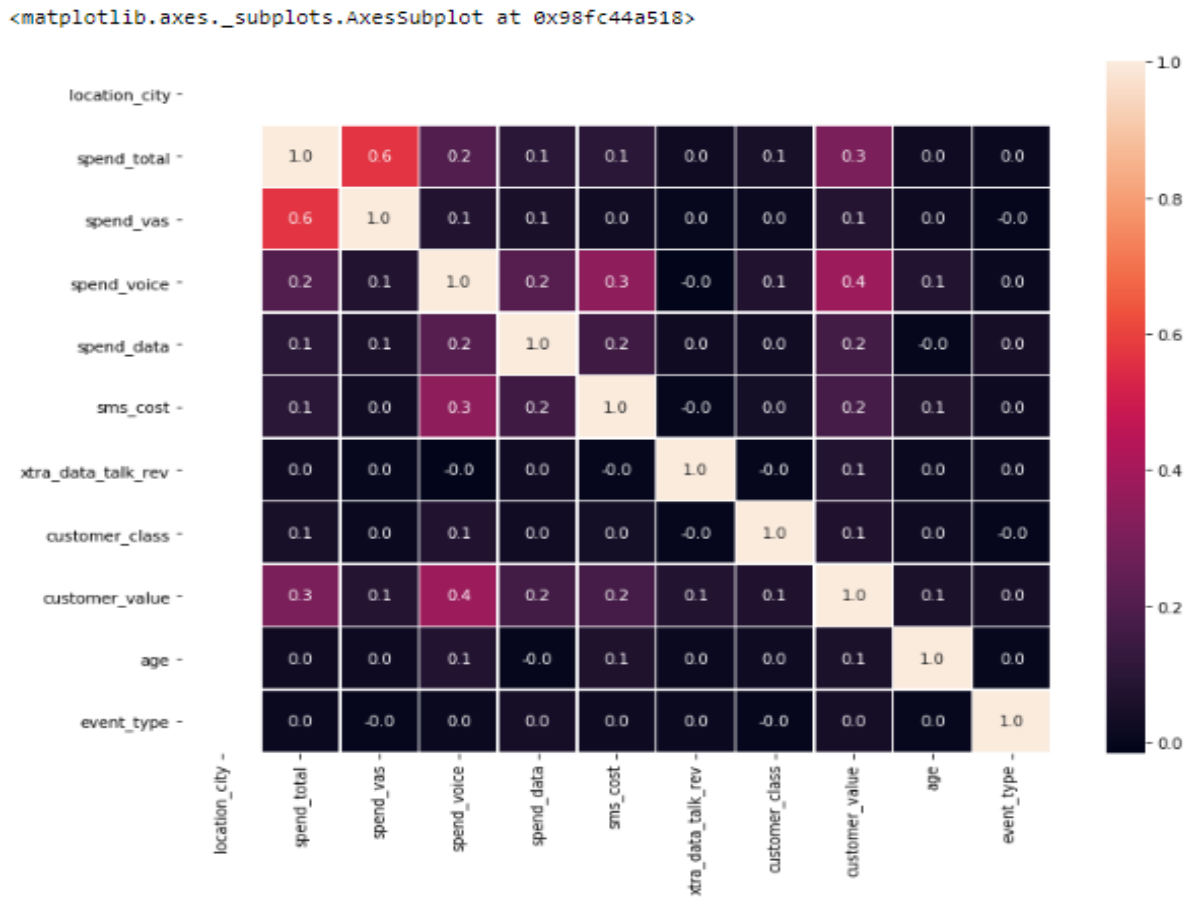
'device_manufacturer' **–** we had some labels of this feature replaced with 'others' due to their sparse appearance among the observations. *'tecno', 'itel', 'infinix', 'samsung', 'nokia'* and *'apple'* were retained while other labels were replaced with 'others'. Each label was then mapped to a binary vector.

'location_region' **–** we had each label mapped to a binary vector.

'device_type' **–** we had each label mapped to a binary vector

Correlation and Apparent Relationships

After encoding, the sci-kit learn plot package was used to check correlation and identify relationships between the features in the pre-processed data.



We then fit and train the pre-processed data on various algorithms

Model Building

Random Forest Algorithm and Evaluation Metrics

Several predictive model algorithms were used to learn the trained data (75% of the dataset) and make predictions with the test data (25% of the dataset). The following results were gotten:

Random Forest

Accuracy: 0.798

Precision: 0.763

Recall: 0.173

Logistic Regression

Accuracy: 0.972

Precision: 0.000

Recall: 00000

Support Vector Machine (SVM)

Accuracy: 0.972

Precision: 0.000

Recall: 00000

Decision Tree

Accuracy: 0.95

Precision: 0.166

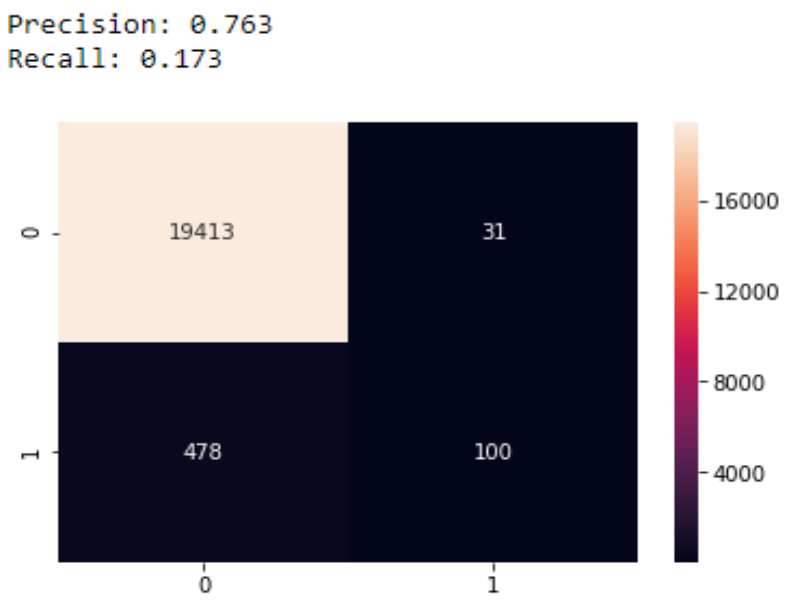
Recall: 0.201

Catboost

Accuracy: 0.97

Macro Precision: 0.49

Macro Recall: 0.50



Random Forest gave the optimal precision score of 76.7% and a low recall of 17.3% and so we chose this as our machine learning algorithm. Random Forest is also a good algorithm for training imbalanced classes. Algorithms like Catboost tend to overfit the model though it has a higher accuracy.

Dimensionality Reduction

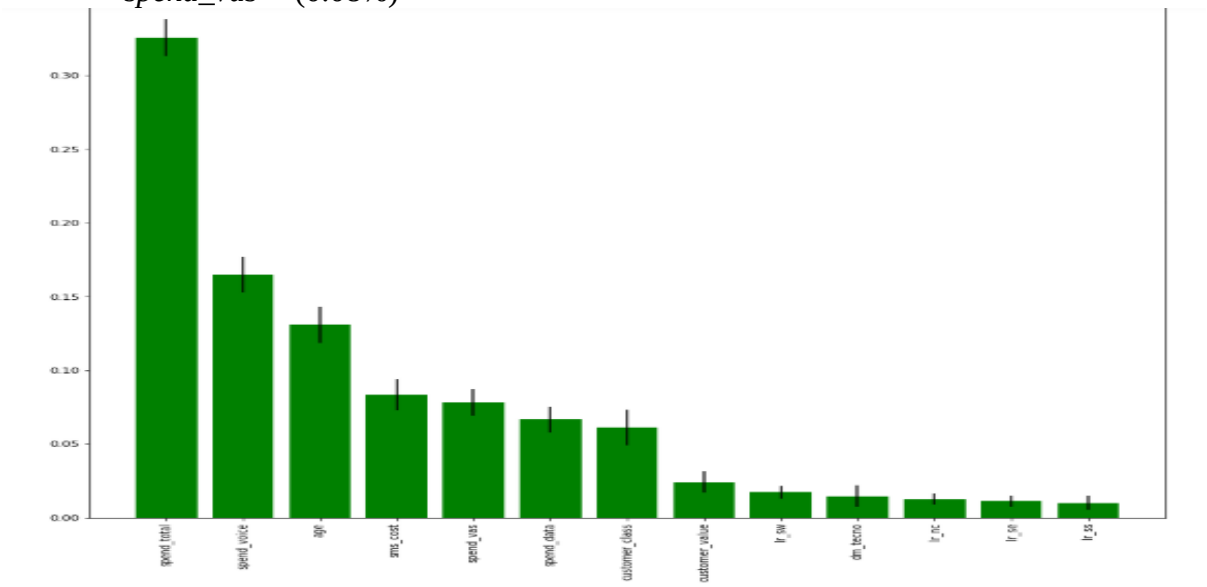
We then decided to reduce dimensions to see if Random Forest will improve its performance by selecting top ranked features that predicted the target well.

Important Feature

Numerical variables (transactional) like 'spend_total', 'spend_voice', 'sms_cost' are top informative predictors in predicting if a customer would click on an ad sent or not, demograghic variable - 'age', and categorical variable- 'customer_class' were found to be among the top predictors.

Top 5 Feature ranking:

- *'spent_total' - (0.19%)*
- *'spent_voice' - (0.12%)*
- *'age' - (0.11%)*
- *'sms_cost' - (0.08%)*
- *' spend_vas'* - (0.08%)



We used **Principle Component Analysis (PCA)** and **Linear Discriminant Analysis (LDA)** on the Random Forest algorithm.

With PCA and n component = 3, we got an Accuracy was 0.973. Precision was 0.557 and Recall was 16.9. Precision has increased a by 0.37. LDA did not seem to give us an improved performance in terms of precision.

Handling Imbalanced Classes

We also decided to oversample the minority class (our target class) in the dataset using **SMOTE().** PCA & LDA was also performed on the oversampled data. So Recall became better (0.540). We also under sampled our Dataset using **NEARMISS()**, to the results below:

Logistic Regression

Accuracy: 0.119

Precision: 0.029

Recall: 0.925

Random Forest

Accuracy: 0.057

Precision: 0.028

Recall: 0.985.

After under-sampling, PCA and LDA were also performed on Random Forest algorithm with the best scores: Accuracy of 0.060. Precision was 0.029 and Recall was 0.985.

Our model seems to have its optimal performance without performing any dimentionality reduction of the training data.

Deployment

In this phase, a machine learning application was built using flask and was tested using port 8000, before a Docker-image was being created for deployment on Amazon Web Services (AWS) or Docker.

The machine learning model used for our dataset was saved as an object and converted to bytes using a python package called pickle.

An API endpoint was built using flask to be able to access the serialized model. Our API is built in way as to be capable of handling any form of error, and also handling any form of pre-processing from a raw json/csv file. Our model was able to serve the responses in less than 100ms. When tested with Postman software, we achieved about 60-200ms after different tests and we also load tested using python ‘locust’ package to ensure we meet the required service time.

Docker was our preferred choice for deployment. Containerisation, we know, helps to lay the foundation for other steps if and when we have to deploy our models on cloud platforms or on-premise servers while also serving as a platform for deploying models.

We created a Docker-image for our model and a container from which the image would run.

For ease of access, a direct link through which the model can be accessed on Docker has been published below.

docker run -p 8080:8000 --name lanreosuntoye/terragon_repo

The image of our model being tested on postman is displayed below:

Flask API on port 8000

HTTP POST: http://127.0.0.1/8000/predict

INPUT

CSV/JSON with columns and values as keys value pairs


```
{
 "msisdn":"e723bb7947e52fdc65ca8272045c3856",
 "location_region":"South West",
 "location_state":"lagos",
 "location_lga":"somolu",
 "location_city": "None",
 "device_manufacturer":"asus",
 "os_name":"android",
 "os_version":7.1,
 "spend_total":100,
 "spend_vas":0,
 "spend_voice":828.833339,
 "spend_data":0,
 "sms_cost":0,
 "xtra_data_talk_rev":0,
 "customer_class":49,
 "customer_value":"low",
 "gender":"M",
 "age":54,
 "device_type":"Smartphone",
 "ad_id":"/110/1348/1428",
 "ad_name":"doubble_july1_5 SMS",
 "@timestamp":"2019-07-23T20:47:20.000Z"
}
```


OUTPUT

Prediction output


```
{
  "prediction": "[1]"
}
```