CMPE 255 - Data Mining
Google Store Customer Revenue Prediction

GROUP 9

SAN JOSÉ STATE
UNIVERSITY

Gayatri Yadkikar 014562574
Ramin Mousivand 012499474
Mayura Dhivya Nehruji 012434461
Venkata Lakshmi Pravallika Ramachandruni 015297308

TABLE OF CONTENTS

1. Introduction

1.1 Motivation

In today's world online shops are very popular. It would be very helpful for online stores if they can predict their revenue. That will give them better ideas on how to budget. As a matter of fact, most of the revenue is usually coming from a small percentage of the customers. Therefore, the prediction helps to identify which customers are more contributing to the revenue generation. In this project online google store data is used to predict revenue from customers.

1.2 Objectives

The objective of this project is to analyze a Google Merchandise Store (also known as GStore, where Google swag is sold) customer dataset to predict revenue per customer using different regression models. The metrics used for evaluating the models is the “Root Mean Square Error” (RSME).

1.3 Literature review

Price prediction has always been one of most important applications of learning algorithms. Prices are usually volatile in nature and difficult to forecast. One of the most popular methods for price prediction is to use supervised machine learning techniques. Such techniques help to find the price patterns based on previous historical records. The prices can be treated as time series to train models. CLassification models such as SVM can be used to predict the price patterns [1], but for price prediction, various regression models can be used. There are various regression models. Here is the list of some of the most well-known algorithms: Linear Regression, Support Vector Regression (SVR), Ridge Regression (RR), Lasso Regression (LR), Elastic Net Regression (ER), XGBoost Regression (XGBR), Light Gradient Boosting Machine Regression (LGBMR) , Adaboost Regression (AR) and Gradient Boosting Regression(GBR). Qu and Zhang explained the SVR that uses a kernel function to overcome non-linearity [2]. ANN (Artificial Neural Network) or Deep learning is another type of forecasting method inspired by the human brain. This is really helpful with non-linear and complex data. J. B. Heaton and his colleagues used ANN application for price forecasting [3]. This method is helpful against overfitting and noise in the data, but a little difficult to implement. Some of the works used Random Forest Regressor, which is an ensemble method. It is also used in the market. The speed is not slower than linear regression but the performance is usually better.

2. System Design/Implementation

This section will discuss the algorithms selected, technology and tools used, system design, and use cases.

2.1. Selected Algorithms

The prediction of the total revenue generated by each customer, based on the Google store's existing transactions, is a regression problem, and hence regression algorithms are implemented in this project.

2.1.1. Linear Regression

The Linear Regression model's approach finds a linear relationship between the independent variable x and dependent variable y. i.e., predict the output variable based on the input variable x. The model finds the best fit line or plane for the dataset and aims to reduce the error between the predicted value and actual value. Linear Regression is the baseline approach or model.

2.1.2. Ridge Regression

The Ridge regression model addresses the overfitting issue of linear regression by the regularization method that aims to reduce the variance at the cost of bias introduction. Applying a penalty minimizes the magnitude of the coefficients. The ridge regression model uses the l2 norm for regularization and minimizes the mean squared error and model complexity.

2.1.3. Lasso Regression

The Lasso Regression model is similar to the ridge regression model with few differences. The lasso model applies L1 regularization, i.e., penalty, on the sum of absolute values. The lasso model offers feature selection as it forces the coefficients of features of low importance to zero and selects the other features

2.1.4. Elastic Net Regression

The ElasticNet regression model applies both L1 and L2 regularization. The parameter l1_ratio controls the choice of penalty to be l1 or l2. When the l1_ratio value is 0, ridge penalty(l2) is applied. When l1_ratio = 1, lasso penalty(l1) is applied. A value between 0 and 1 for the l1_ratio denotes a combination of ridge and lasso penalty

2.1.5. LightGBM Regressor

The Light Gradient Boosting Machine (lightGBM) Regressor is a boosting model that bases its learning on decision trees. The growth of the tree in lightGBM is leaf-wise as opposed to the more common level-wise growth. It offers distributed learning with faster training speed, low memory usage.

2.1.6. XGBoost Regressor

The Extreme Gradient Boosting (XGBoost) model offers both decision trees and linear models as base learners. It also provides hyper tuning parameters such as the objective, max_depth, subsample, and regularization parameters alpha, beta, and gamma. The objective parameter denotes the loss function for regression.

2.1.7. Random Forest Regressor

The Random Forest Regressor builds a random subset of samples from the dataset and creates a decision tree based on it. Each tree in the forest predicts every new record, and the final value is obtained by averaging the values. The random forest model works on the dataset that has missing values, categorical and numerical features.

2.2. Technology, Tools and Libraries Used

The tools, software libraries, and technologies used in the project are listed as follows.

1. Language - Python3
2. Google Colab, a cloud-based web IDE with GPU/TPU resources for shared code development
3. Scikit-Learn Library for preprocessing, metrics and models
4. Plotly Python Graphing Library for interactive data visualization graphs
5. Seaborn library for data visualization graphs such as distplot
6. Matplotlib library for data visualization

2.3. Process Flow

The process flow for the project is illustrated below in Fig.1

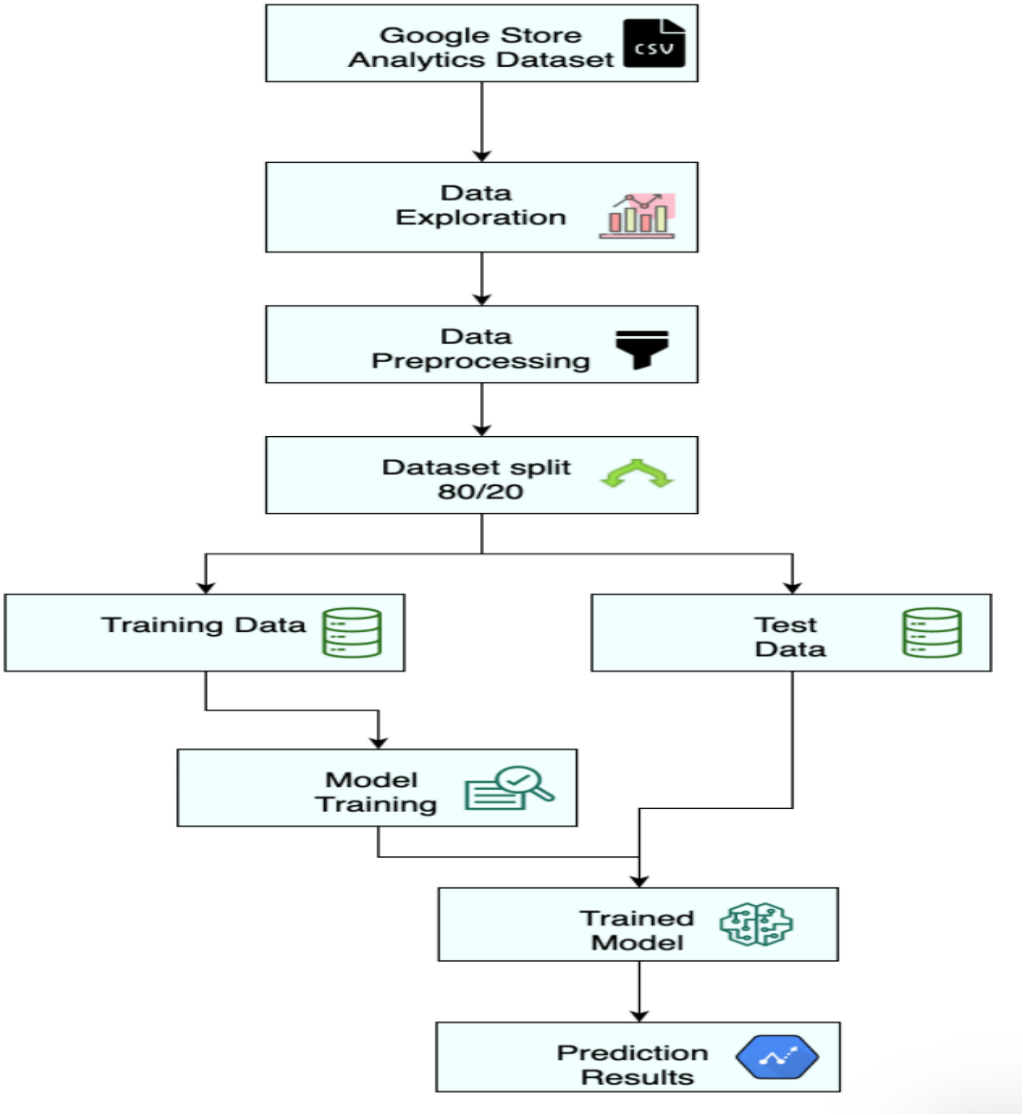


Fig. 1: Process flow diagram

2.4. Use Cases

The prediction of revenue generation per customer will help with the following identified use cases:

- Increase revenue by focusing on existing customers
- Proper budgeting for promotional events

3. Experiments

3.1. Dataset

Data provided is of Google Merchandise Store (where Google swag is sold) and customer transactions are given from August 2016 till August 2017(approx 1.4 GB). Data consists of 903653 rows and 55 columns.There are multiple columns which contain JSON blobs of varying depth. In one of those JSON columns, totals, the sub-column transactionRevenue contains the revenue information which is the target variable.

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

3.2. Data Preprocessing

There were 55 columns in raw data. Below steps are followed to clean the dataset

Imputing null values

- Out of 55 given columns, 24 columns have only one value in the column which is not offering much information. Hence such columns were removed
- Columns where null values are greater than 80% of the total number of values are removed
- Null values of remaining columns are filled using mode strategy

Feature Engineering

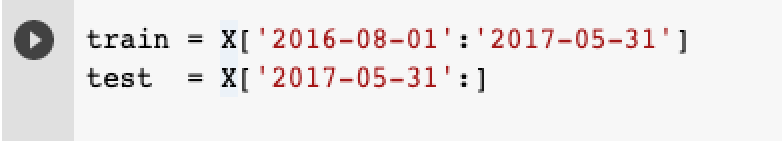
- Column ‘VisitID’ is id for the session and Column ‘VisitStarttime’ is a timestamp expressed as POSIX time. Most of the values are the same in both the columns and hence a new input feature is created where the difference of the two values is not equal to zero.
- ‘Date’ column is formatted to python DateTime object and new input features Quarter month and Weekday are created to train the model
- ‘Visittime’ column is formatted to create new input feature visit hour in a day
- Column ‘hits’ provides a record of all page visits.New input feature mean hits per day is created from this column

Dimensionality Reduction

- PCA technique is used to preserve 95% variance
- It was observed that there is no reduction in number of columns in data to preserve required variance ratio

3.3. Methodology

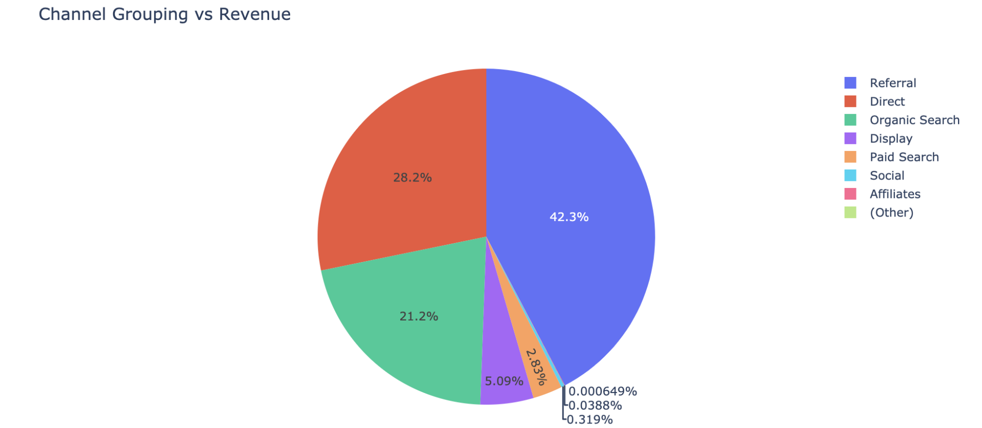
The csv data containing JSON blobs was read and converted to a dataframe.Data was preprocessed and new input features that are more useful to analyse the data are derived using existing columns. Revenue from each customer is the target variable and log of the customer revenue is considered to train the model.For training the model, data was split into 80/20.Since data has customer transaction records from August 2016 till August 2017, data is sorted in ascending order using date column and top 80% of the records are used for training model(records from August 2016 till May 2017). Remaining 20% records are used as test data sets(records from June 2017 till August 2017).

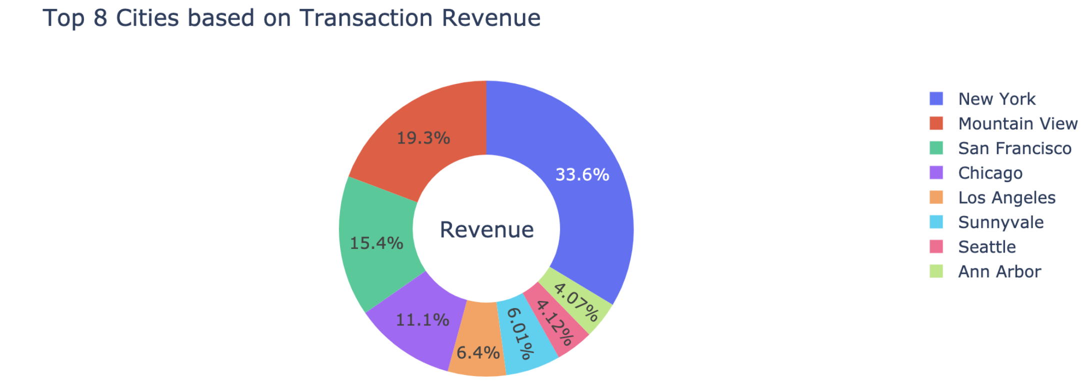


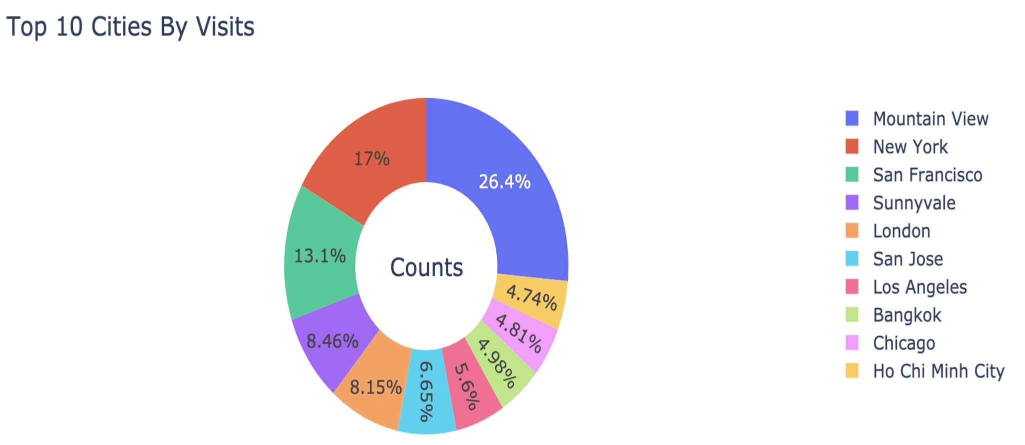
```
train = X['2016-08-01':'2017-05-31']
test  = X['2017-05-31':]
```

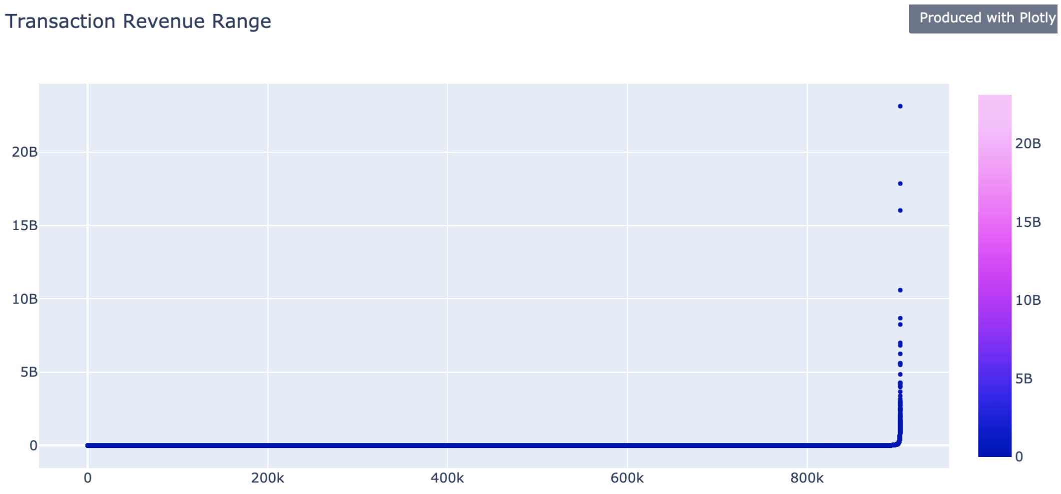
As the target variable is continuous regression models were chosen predicting the target variable.Each model was evaluated using Root Mean Square Error(RMSE) from sklearn.metrics. Hyper parameters of models were fine tuned to further increase efficiency of the model.

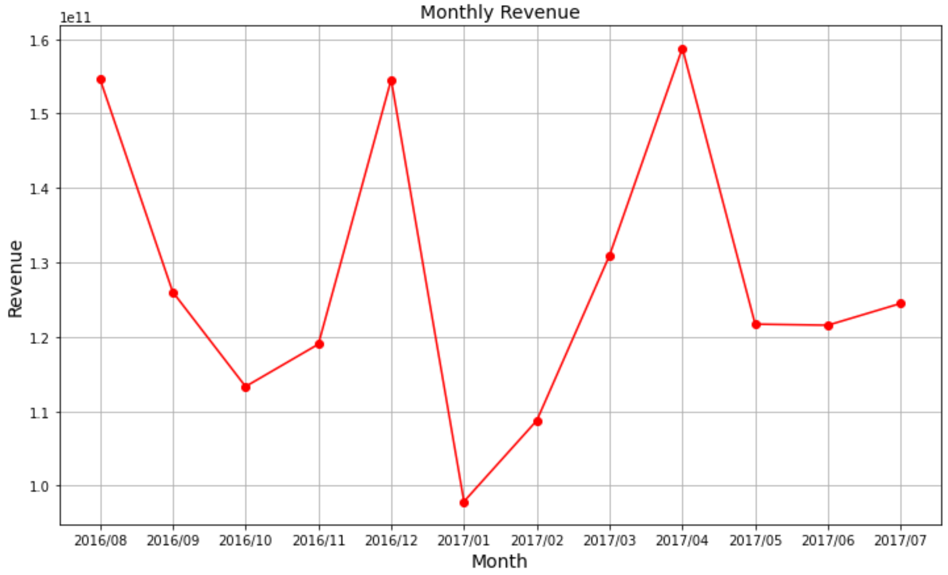
3.4. Data Visualization

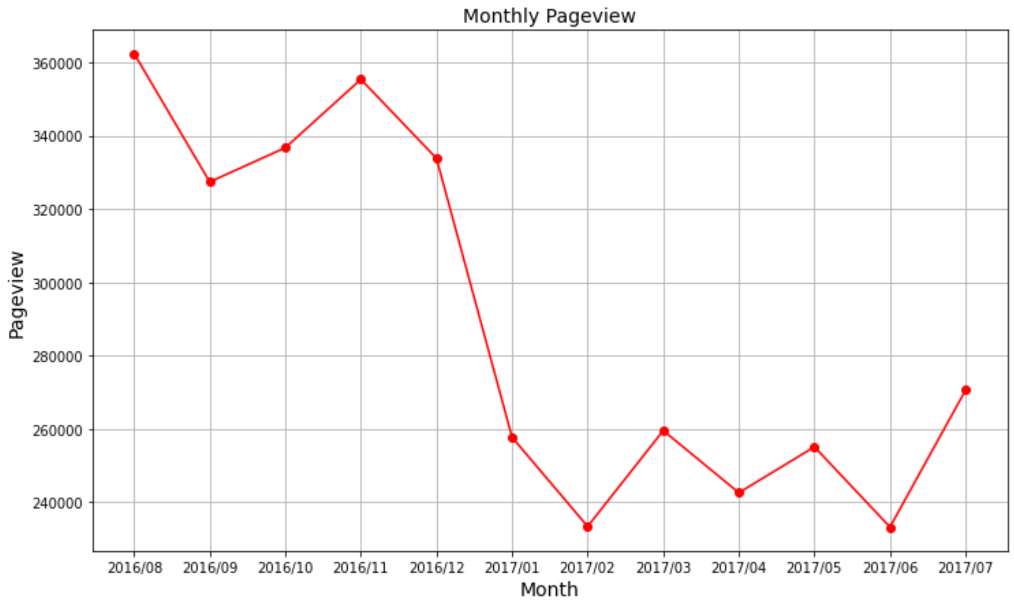




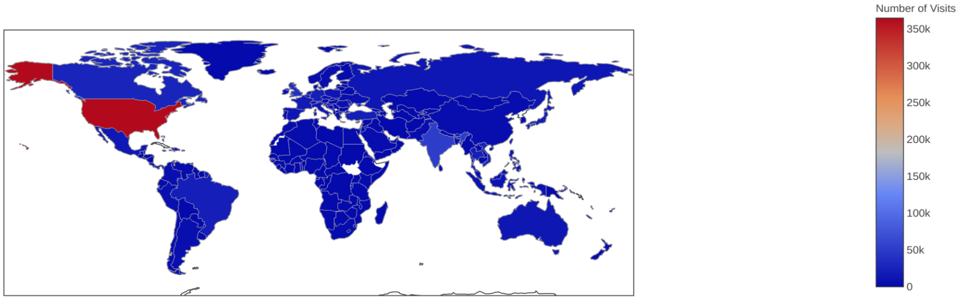

Top 10 Cities By Visits
Counts
26.4%
17%
13.1%
8.46%
8.15%
6.65%
5.6%
4.98%
4.81%
4.74%
Mountain View
New York
San Francisco
Sunnyvale
London
San Jose
Los Angeles
Bangkok
Chicago
Ho Chi Minh City


Transaction Revenue Range
Produced with Plotly
20B
15B
10B
5B
0
0
200k
400k
600k
800k
20B
15B
10B
5B
0


Monthly Revenue
1e11
1.6
1.5
1.4
1.3
1.2
1.1
1.0
Revenue
2016/08
2016/09
2016/10
2016/11
2016/12
2017/01
2017/02
2017/03
2017/04
2017/05
2017/06
2017/07
Month



Visits per Country



Observations :

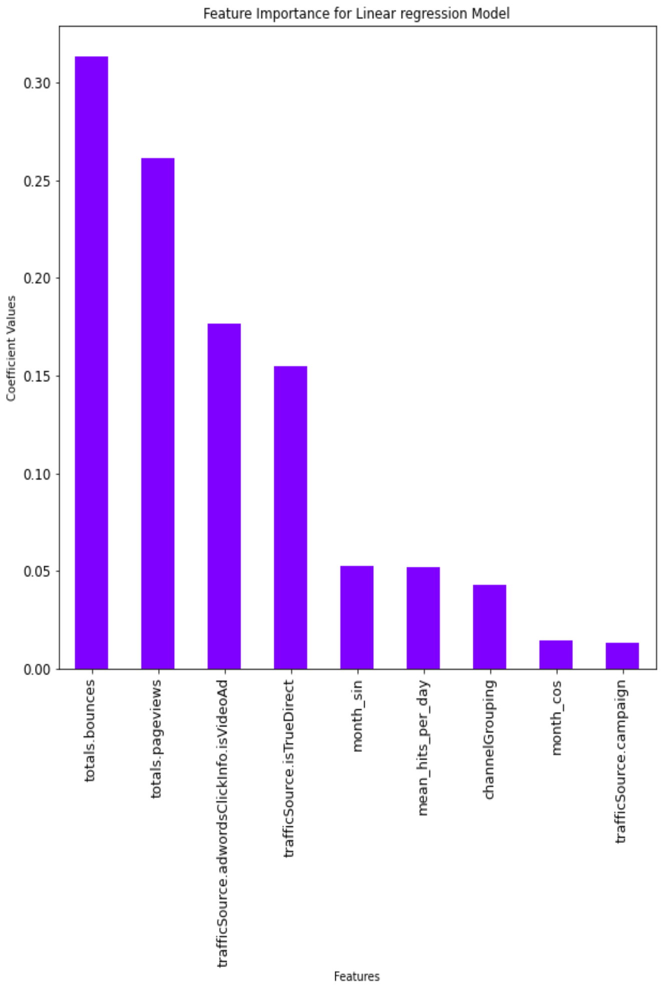
Seasonality and cyclicity are observed in monthly revenue. Revenue increases in a few months followed by a drop and again followed by an increase.New York contributes to approximately 33% of total revenue.Mountain View has more number of page views followed by Newyork. Referral channel is the highest contributor to the transaction revenue followed by Direct.

3.5. Evaluation

After training the model, it is evaluated using test data on Root Mean Square Error. Hyper parameters of each model were tuned to further decrease RMSE of the model.

Linear Regression Model

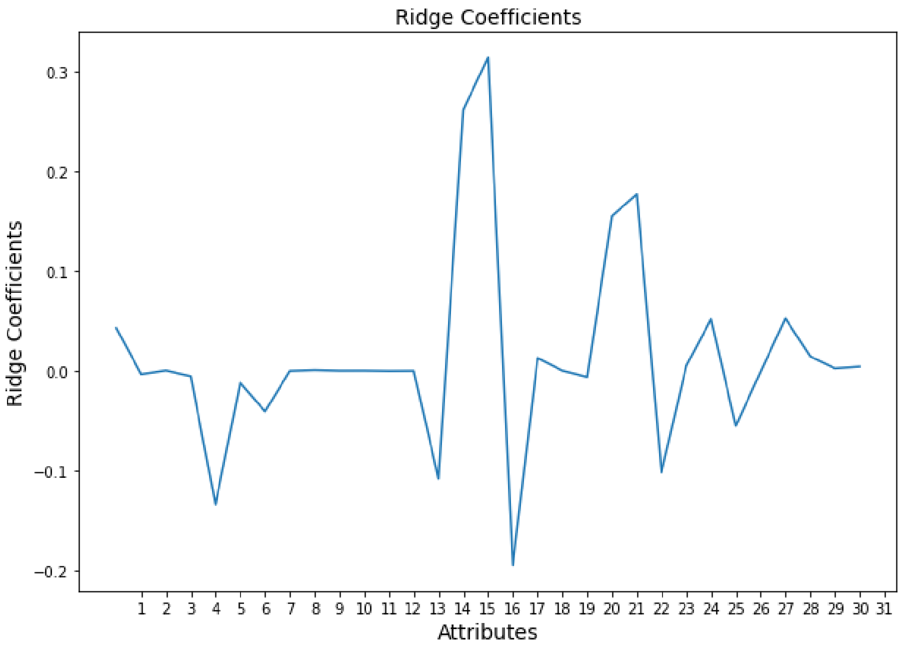
RMSE for this model is 1.92 and coefficient of determination as 0.19.Feature importance for linear regression is as follows

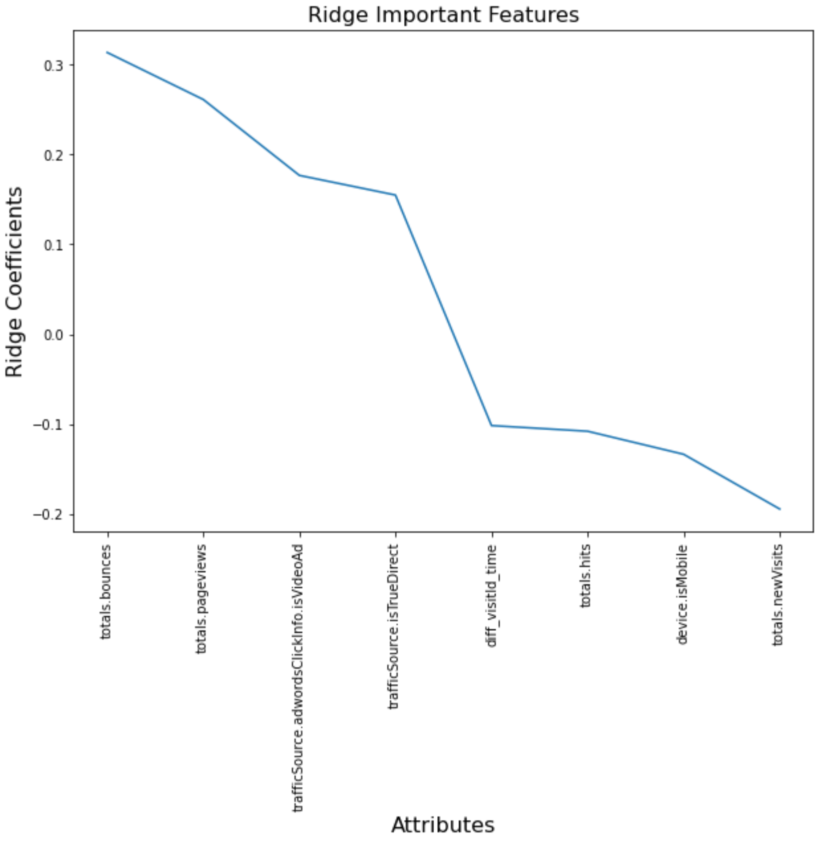


Ridge Regression Model

Model is finetuned using various values of alpha and least RMSE is achieved for an alpha value of 0.01

Alpha	RMSE
0.5	1.92
0.1	1.92
0.01	1.92

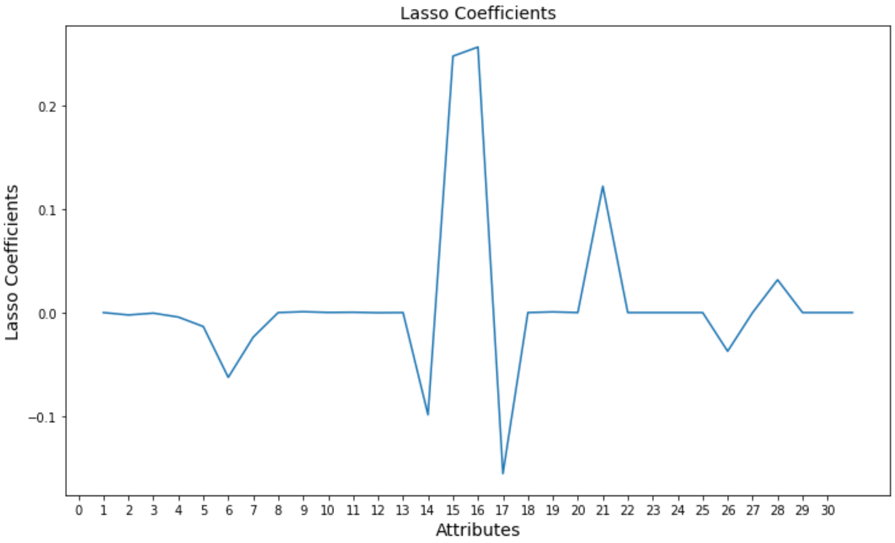

Ridge Coefficients
Ridge Coefficients
Attributes
0.3
0.2
0.1
0.0
−0.1
−0.2
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

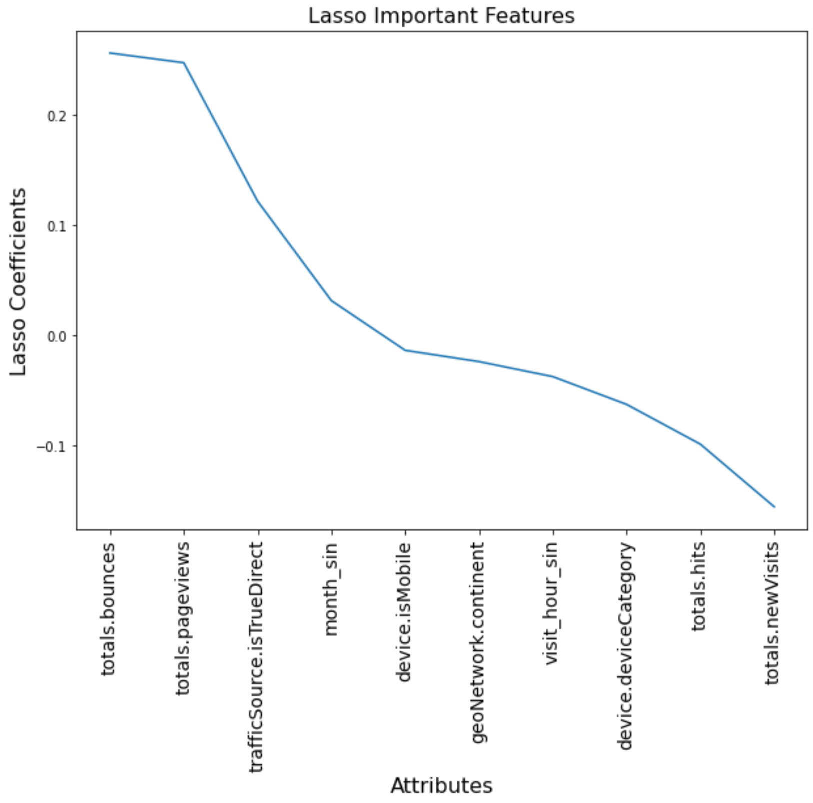

Ridge Important Features
Ridge Coefficients
Attributes
0.3
0.2
0.1
0.0
−0.1
−0.2
totals.bounces
totals.pageviews
trafficSource.adwordsClickInfo.isVideoAd
trafficSource.isTrueDirect
diff_visitId_time
totals.hits
device.isMobile
totals.newVisits

Lasso Regression Model

Model is fine tuned with various values of alpha and the least RMSE is achieved for alpha value of 0.01. As lasso regression performs L1 form of normalization, coefficients of few features are reduced to zero and the model is built with only remaining input parameters.

Alpha	RMSE
0.5	1.95
0.1	1.94
0.01	1.92

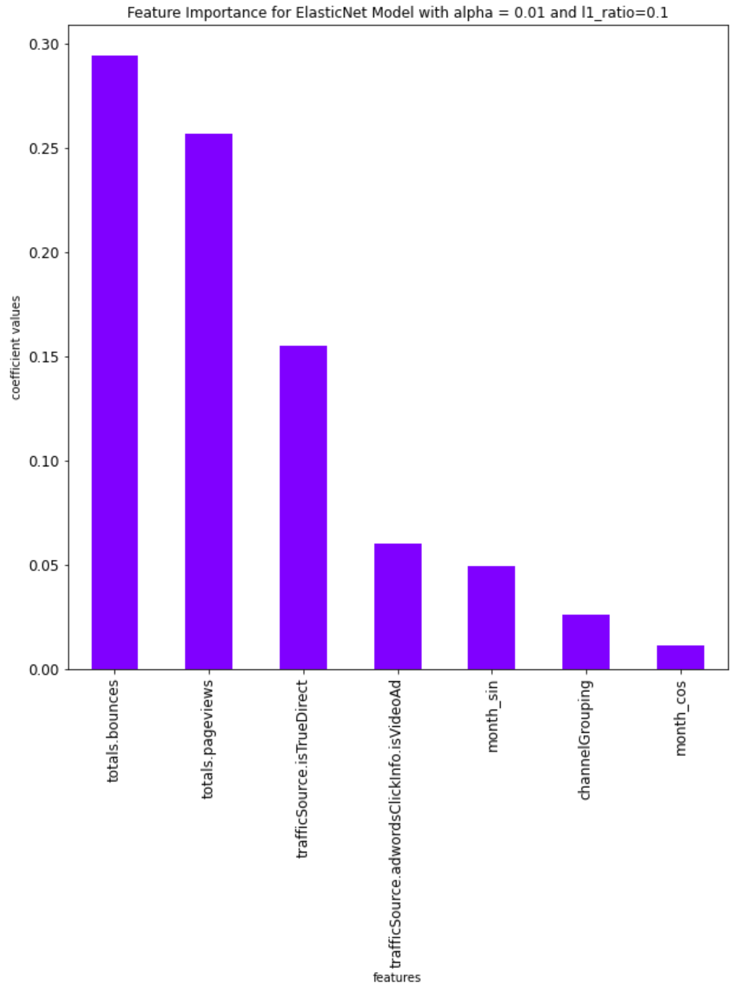




Elastic Net Regression Model
This regression model combines L1 and L2 priors as regularizers.When l1_ratio = 1, lasso penalty(l1) is applied. A value between 0 and 1 for the l1_ratio denotes a combination of ridge and penalty

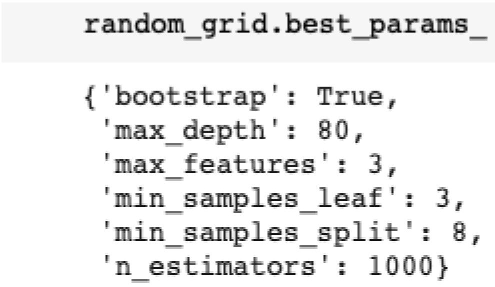
Alpha	L1 Ratio	RMSE
Default(1.0)	Default(0.5)	1.95
0.5	0.3	1.94
0.1	0.3	1.93
0.01	0.1	1.92

Feature Importance for Elastic Net (alpha = 0.01 and l1 = 0.1)



Random Forest regressor Model

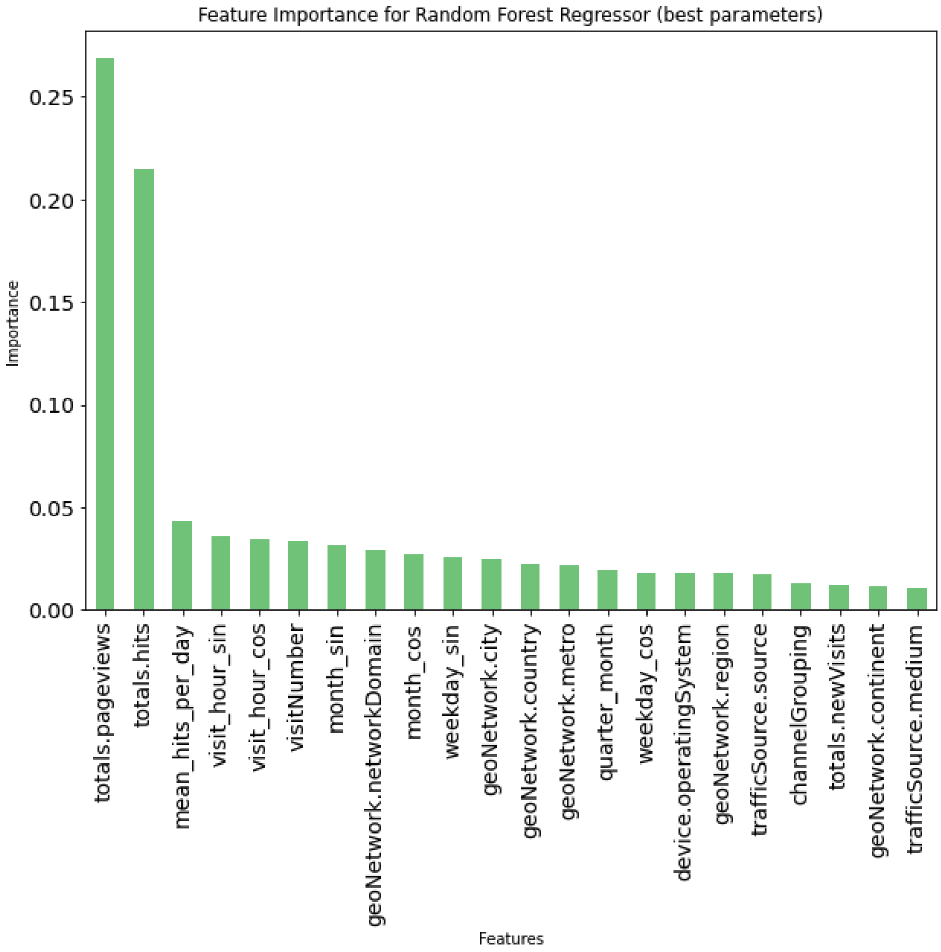
Random Forest is also a Tree-based algorithm that uses the qualities features of multiple Decision Trees for making decisions.Parameters are hypertuned using sklearn gridsearchCV method and below parameters were found to give better accuracy of model



```
random_grid.best_params_
```

```
{'bootstrap': True,
 'max_depth': 80,
 'max_features': 3,
 'min_samples_leaf': 3,
 'min_samples_split': 8,
 'n_estimators': 1000}
```

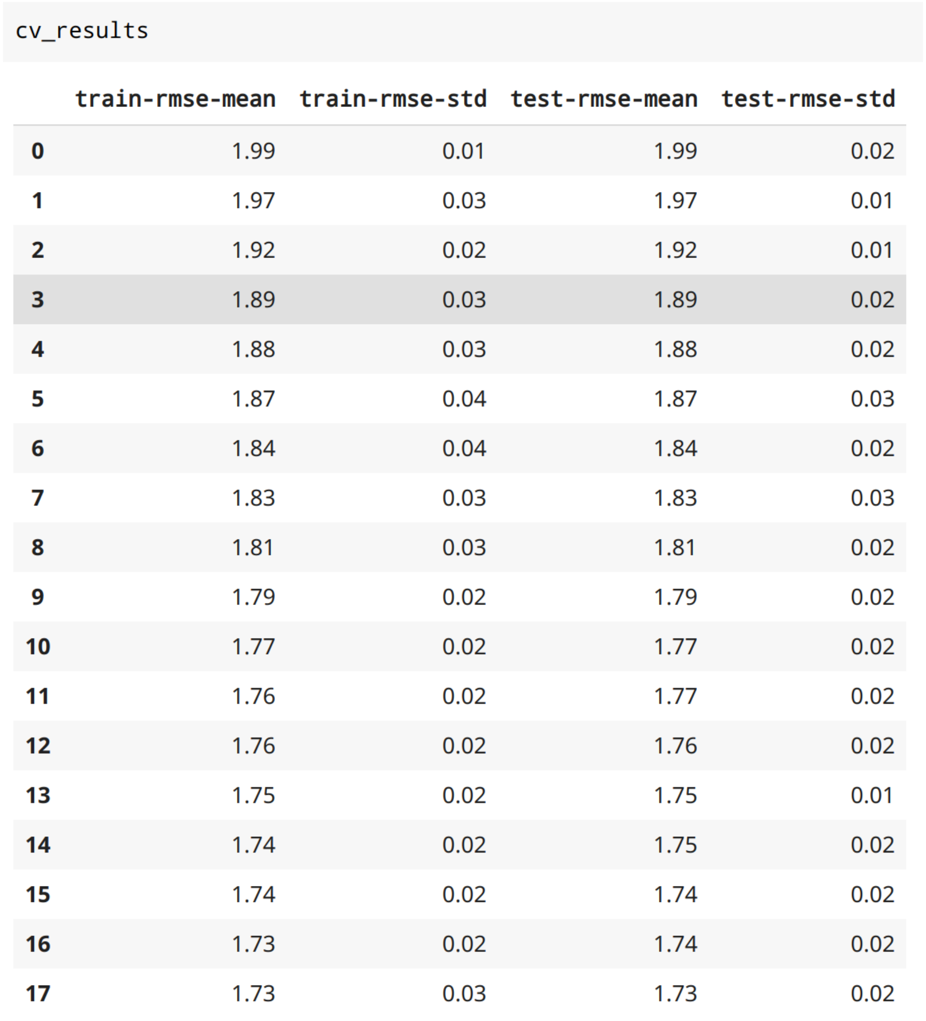
RMSE of the predictions using above parameters is 1.74.Feature importance for this model is as follows



Feature Importance for Random Forest Regressor (best parameters)

XG Boosting

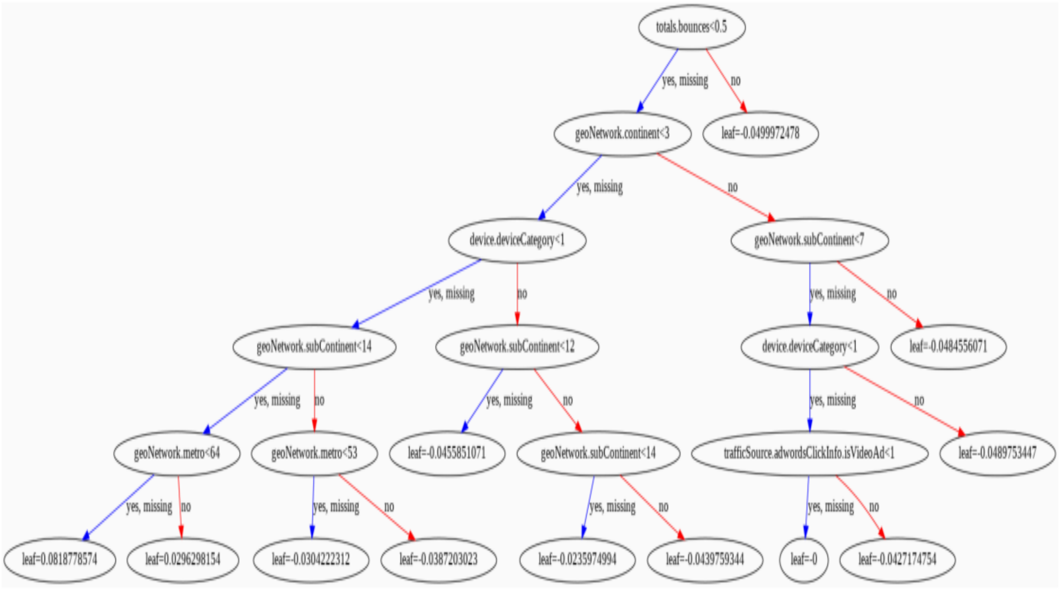
XGBRegressor() class from the XGBoost library is used with the hyper-parameters passed as arguments. Dataset into an optimized data structure called Dmatrix that XGBoost supports .k-fold cross validation technique where all the entries in the original training dataset are used for both training as well as validation is used.Train and Test RMSE metrics for each boosting round are as follows



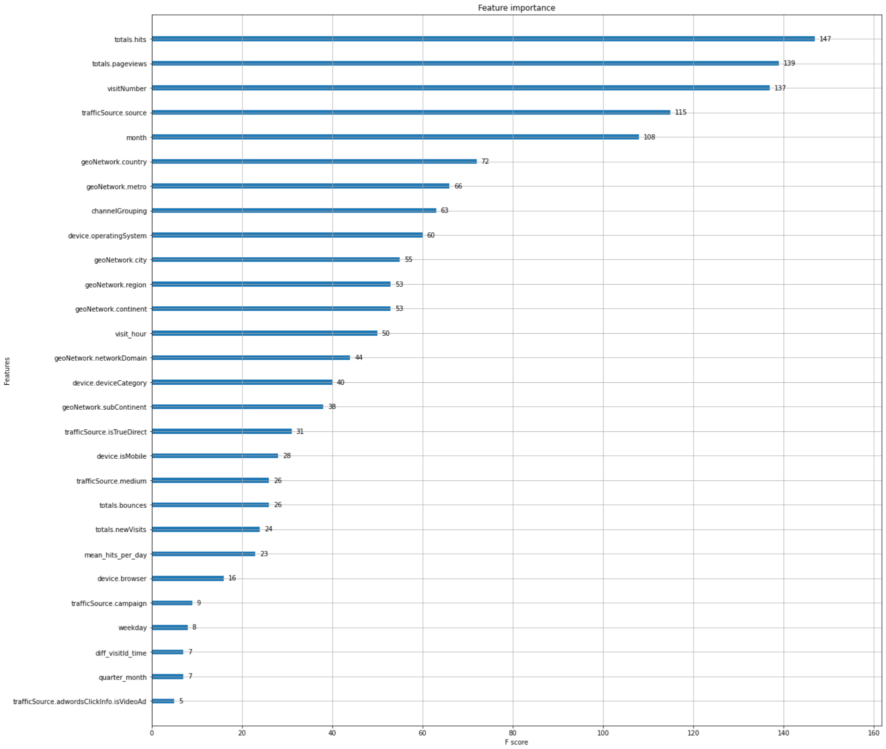
```
cv_results
```

	train-rmse-mean	train-rmse-std	test-rmse-mean	test-rmse-std
0	1.99	0.01	1.99	0.02
1	1.97	0.03	1.97	0.01
2	1.92	0.02	1.92	0.01
3	1.89	0.03	1.89	0.02
4	1.88	0.03	1.88	0.02
5	1.87	0.04	1.87	0.03
6	1.84	0.04	1.84	0.02
7	1.83	0.03	1.83	0.03
8	1.81	0.03	1.81	0.02
9	1.79	0.02	1.79	0.02
10	1.77	0.02	1.77	0.02
11	1.76	0.02	1.77	0.02
12	1.76	0.02	1.76	0.02
13	1.75	0.02	1.75	0.01
14	1.74	0.02	1.75	0.02
15	1.74	0.02	1.74	0.02
16	1.73	0.02	1.74	0.02
17	1.73	0.03	1.73	0.02

plot_tree() function is used for tree visualization



RMSE of the model is 1.74 and Feature importance of the model is as follows



Light Gradient Boosting (LGB) Model

Advantages of LGB Model

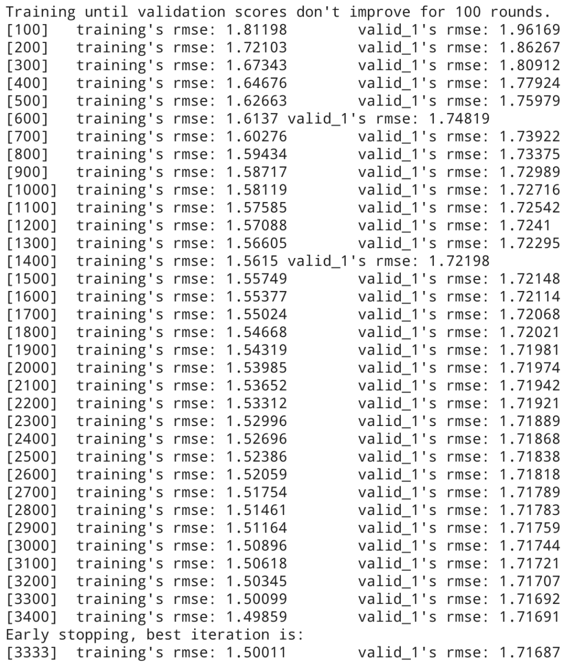
1. Faster training speed and higher efficiency
2. Lower memory usage
3. Better accuracy
4. Compatibility with Large Datasets

As the dataset of given customer transactions is larger, to reduce training time Light GBM model was explored. Parameters of model

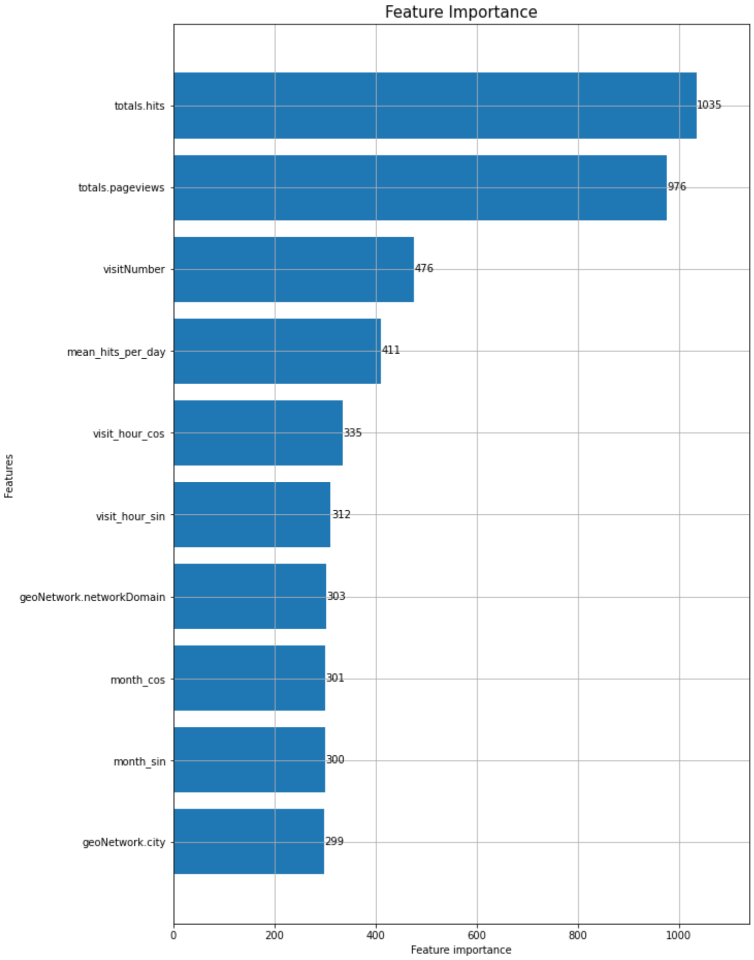
- task : options = train , prediction ; Specifies the task to perform
- application: regression/binary classification/multiclass classification
- num_leaves : number of leaves in one tree
- feature_fraction: specifies the fraction of features to be taken for each iteration
- bagging_fraction:specifies the fraction of data to be used for each iteration and is generally used to speed up the training and avoid overfitting.
- min_gain_to_split: min gain to perform splitting

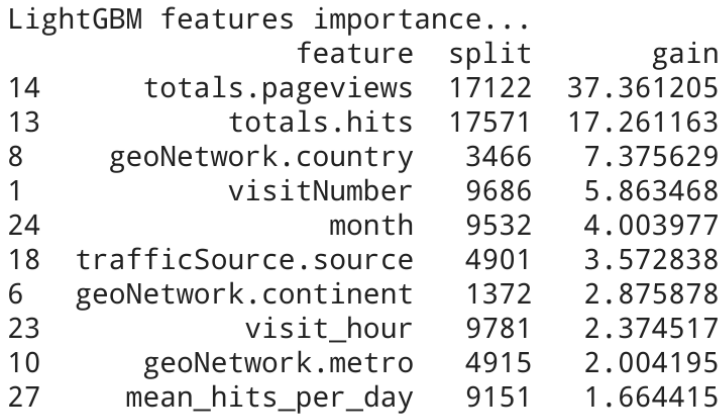
Model was trained with below values of hyper parameters

Num_ leaves	min_child _samples	Learning_ rate	Bagging_ fraction	Feature_ fraction	Bagging_ frequency	RMSE
30	100	0.1	0.7	0.5	5	1.75
40	0.005	0.6	0.6	6	42	1.74



```
Training until validation scores don't improve for 100 rounds.
[100]   training's rmse: 1.81198        valid_1's rmse: 1.96169
[200]   training's rmse: 1.72103        valid_1's rmse: 1.86267
[300]   training's rmse: 1.67343        valid_1's rmse: 1.80912
[400]   training's rmse: 1.64676        valid_1's rmse: 1.77924
[500]   training's rmse: 1.62663        valid_1's rmse: 1.75979
[600]   training's rmse: 1.6137 valid_1's rmse: 1.74819
[700]   training's rmse: 1.60276        valid_1's rmse: 1.73922
[800]   training's rmse: 1.59434        valid_1's rmse: 1.73375
[900]   training's rmse: 1.58717        valid_1's rmse: 1.72989
[1000]  training's rmse: 1.58119        valid_1's rmse: 1.72716
[1100]  training's rmse: 1.57585        valid_1's rmse: 1.72542
[1200]  training's rmse: 1.57088        valid_1's rmse: 1.7241
[1300]  training's rmse: 1.56605        valid_1's rmse: 1.72295
[1400]  training's rmse: 1.5615 valid_1's rmse: 1.72198
[1500]  training's rmse: 1.55749        valid_1's rmse: 1.72148
[1600]  training's rmse: 1.55377        valid_1's rmse: 1.72114
[1700]  training's rmse: 1.55024        valid_1's rmse: 1.72068
[1800]  training's rmse: 1.54668        valid_1's rmse: 1.72021
[1900]  training's rmse: 1.54319        valid_1's rmse: 1.71981
[2000]  training's rmse: 1.53985        valid_1's rmse: 1.71974
[2100]  training's rmse: 1.53652        valid_1's rmse: 1.71942
[2200]  training's rmse: 1.53312        valid_1's rmse: 1.71921
[2300]  training's rmse: 1.52996        valid_1's rmse: 1.71889
[2400]  training's rmse: 1.52696        valid_1's rmse: 1.71868
[2500]  training's rmse: 1.52386        valid_1's rmse: 1.71838
[2600]  training's rmse: 1.52059        valid_1's rmse: 1.71818
[2700]  training's rmse: 1.51754        valid_1's rmse: 1.71789
[2800]  training's rmse: 1.51461        valid_1's rmse: 1.71783
[2900]  training's rmse: 1.51164        valid_1's rmse: 1.71759
[3000]  training's rmse: 1.50896        valid_1's rmse: 1.71744
[3100]  training's rmse: 1.50618        valid_1's rmse: 1.71721
[3200]  training's rmse: 1.50345        valid_1's rmse: 1.71707
[3300]  training's rmse: 1.50099        valid_1's rmse: 1.71692
[3400]  training's rmse: 1.49859        valid_1's rmse: 1.71691
Early stopping, best iteration is:
[3333]  training's rmse: 1.50011        valid_1's rmse: 1.71687
```





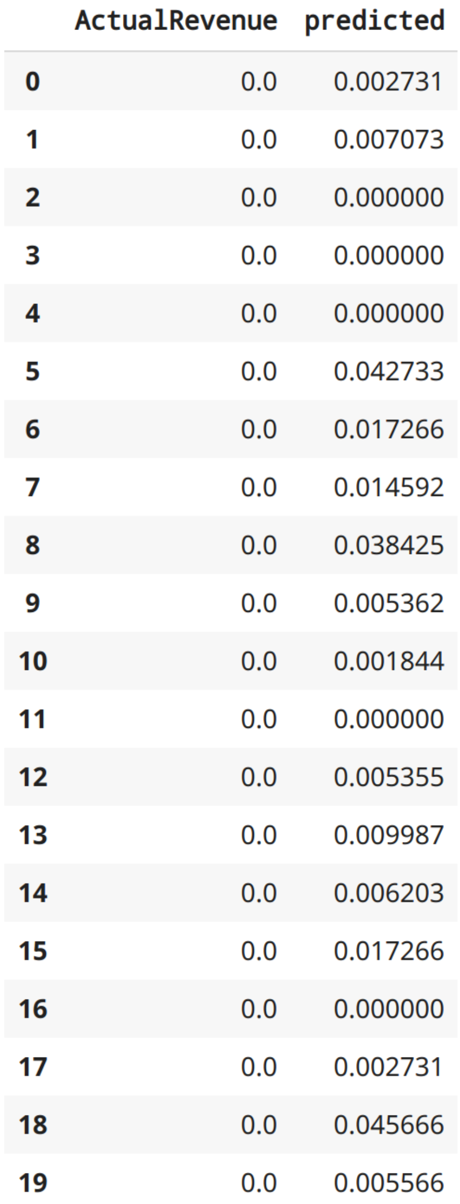
```
LightGBM features importance...
                 feature  split       gain
14      totals.pageviews  17122  37.361205
13           totals.hits  17571  17.261163
8      geoNetwork.country   3466   7.375629
1            visitNumber   9686   5.863468
24                 month   9532   4.003977
18  trafficSource.source   4901   3.572838
6   geoNetwork.continent   1372   2.875878
23            visit_hour   9781   2.374517
10      geoNetwork.metro   4915   2.004195
27      mean_hits_per_day   9151   1.664415
```

3.6. Results

Summary of Models

Model Name	RMSE
Linear Regression	1.92
Ridge Regression	1.92
Lasso Regression	1.92
Elastic Net	1.92
Random Forest	1.74
XG Boost	1.74
Light GBM	1.74

Sample Predictions



	ActualRevenue	predicted
0	0.0	0.002731
1	0.0	0.007073
2	0.0	0.000000
3	0.0	0.000000
4	0.0	0.000000
5	0.0	0.042733
6	0.0	0.017266
7	0.0	0.014592
8	0.0	0.038425
9	0.0	0.005362
10	0.0	0.001844
11	0.0	0.000000
12	0.0	0.005355
13	0.0	0.009987
14	0.0	0.006203
15	0.0	0.017266
16	0.0	0.000000
17	0.0	0.002731
18	0.0	0.045666
19	0.0	0.005566

Value 0.0 in Actual Revenue implies that no transactions are expected from the customer. From the above sample predictions it can be observed that the model also predicted accurately that customers might not bring any revenue in such cases.

4. Discussions And Conclusions

4.1. Decisions made

- The team discussed and researched about selecting the dataset that met all the criteria necessary for the project implementation.
- After finalizing the dataset, the team decided and narrowed down various tasks that were to be implemented during the project lifecycle.

- It was mentioned in the Kaggle competition overview, that our target variable, ’totalTransactionRevenue’, is inside the ’totals’ JSON column. So we have to flatten these JSON values and add them to our dataset to get value out of them.
- Finally, we made decisions about various regression models for target prediction.

4.2. Difficulties faced

- Null values: The given dataset has a large number of null values, so we removed all columns that are more than 80% empty.
- The team also faced difficulties in implementing algorithms and obtaining the best possible results considering all the factors such as the complexity of data, the volume of data, a large number of features, etc.
- Size of dataset:The training dataset size was too large , so initially we faced issues in cleaning and preprocessing.

4.3. Things that worked

- We tuned the various hyperparameters and achieved the least rmse score with all the regression models like Ridge,Lasso,Random forest, Elastic net regression model,LGB model etc.
- As the dataset was huge, with 900K+ rows and 55 dimensions, it gave us the opportunity to identify the pitfalls in handling such complex datasets.But with adequate research, the team was able to successfully work with the data.

4.4. Things that didn’t work well

- We started with linear regression, which gave the root mean score error as 1.92.So, we tried other regression models to achieve less RMSE score and improve our prediction accuracy.However, owing to insufficient data in more than 50% columns RMSE score could not be improved.

4.5. Conclusion

The success of machine learning in predicting customer’s revenue relies on the good use of the data and machine learning algorithms. Selecting the right machine learning method for the right problem is necessary to achieve the best results. However, the algorithm alone can not provide the best prediction results. Tuning hyperparameters and preprocessing data for machine learning, is also an important factor. The aim of this project was to compare method selection by their ability to improve the prediction results in terms of error and time.However, it is still hard to specifically tell how much money they will spend, since we cannot predict the item customer is looking for.Google recently provided a new version of the data set, which contains what items visitors look through and how long they stay at each page. The richer information can help make better predictions of revenue. Next we can use these data to do future analysis, expecting to have a lower error.

5. Project Plan

Task	Ownership
Project Proposal	All team members.
Data Preprocessing	All team members.
Exploratory Data Analysis	All team members
Feature Engineering	All team members.
Linear Regression	Gayatri Yadkikar
Ridge and Lasso Regression Model	Ramin Mousavind
LightGBM and XGBoost Model	Venkata Lakshmi Pravallika Ramachandruni
ElasticNet and Random Forest Model	Mayura Dhivya Nehruji
Report Documentation	All team members
Presentation	All team members.

6. Github Link - https://github.com/venkatalakshmi03/Data-Mining-255

7. References

1. K.-F. WONG, Y. XIA, R. XU, M. WU, and W. LI, “Pattern-Based Opinion Mining for Stock Market Trend Prediction,” International Journal of Computer Processing of Languages, vol. 21, no. 04, pp. 347–361, Dec. 2008.
2. Q. Ouyang, W. Lu, X. Xin, Y. Zhang, W. Cheng, and T. Yu, “Monthly Rainfall Forecasting Using EEMD-SVR Based on Phase-Space Reconstruction,” Water Resources Management, vol. 30, no. 7, pp. 2311–2325, Mar. 2016.
3. J. B. Heaton and N. Polson, “Deep Learning for Finance: Deep Portfolios,” SSRN Electronic Journal, 2016.

4. Data Source: Google Analytics Customer Revenue Prediction — Kaggle. https://www.kaggle.com/c/ga- customer-revenue-prediction
5. https://towardsdatascience.com/l1-and-l2-regularization-methods-ce25e7fc831c
6. https://medium.com/@pushkarmandot/https-medium-com-pushkarmandot-what-is-lightgbm-how-to-implement-it-how-to-fine-tune-the-parameters-60347819b7fc
7. https://www.kaggle.com/diogomenezesborges/full-depth-eda-analysis
8. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.StackingRegressor.html