Capstone Project-
Google Analytics Customer Revenue Prediction

Yifei Wang
2020-06-15

Introduction

The 80/20 rule states that 80% of the effort of anything is usually generated by 20% of the effort. In business this can be translated as 80% of a company's revenue being created by 20% of its customers. Machine learning is broadly used in the marketing industry to help companies to better understand their customers segmentations in order to design marketing campaigns that result in high revenues.

In this project, I analyzed a Google Merchandise Store (also known as GStore) customer dataset to predict revenue per customer. I applied a customer lifetime value (CLV) model to forecast how much a customer may spend in the store within a year given customer segments. Clustering and regression techniques were used in the project to develop feature clustering and predict CLV. My goal is to implement a continuous numerical output of a customer's lifetime value in order to help the GStore to find their most valuable customers and better use of their marketing budgets.

Problem Statement

1. What will be the tracking metric?
2. Customers have different needs and their own profiles. What will be the best features to describe them?
3. Companies invest in customers to generate revenues. Can I help these companies identify customers' behavior patterns, customers' profiles thus they can act accordingly to maximize their revenues?

Data Visualization and Exploration

I used the a GStore customer dataset provided by Kaggle's Google Analytics Customer Revenue Prediction Competition. The dataset has 903,653 rows and 29 columns, which contains user transactions from August 1st, 2016 to April 30th, 2018.

Data Fields

Data fields include,
- fullVisitorId- A unique identifier for each user of the Google Merchandise Store.
- channelGrouping - The channel via which the user came to the Store.
- date - The date on which the user visited the Store.
- device - The specifications for the device used to access the Store.
- geoNetwork - This section contains information about the geography of the user.
- socialEngagementType - Engagement type, either "Socially Engaged" or "Not Socially Engaged".
- totals - This section contains aggregate values across the session.
- 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).
- hits - This row and nested fields are populated for any and all types of hits. Provides a record of
all page visits.
- customDimensions - This section contains any user-level or session-level custom dimensions that are set for a session. This is a repeated field and has an entry for each dimension that is set.
- totals - This set of columns mostly includes high-level aggregate data.

Outcome Exploration

TransactionRevenue is used as the outcome variable in this project. TransactionRevenue is the total transaction revenue expressed as a value multiplied by 10^6. As I did not plan to provide insights on negative purchase values, I logged the outcome value to prevent the negative predictions.

From the data given, only 1.27% of the observations are actual purchases while 1.4 of the visitors are actually revenue generated customers.

The graph below shows the log of TransactionRevenue:

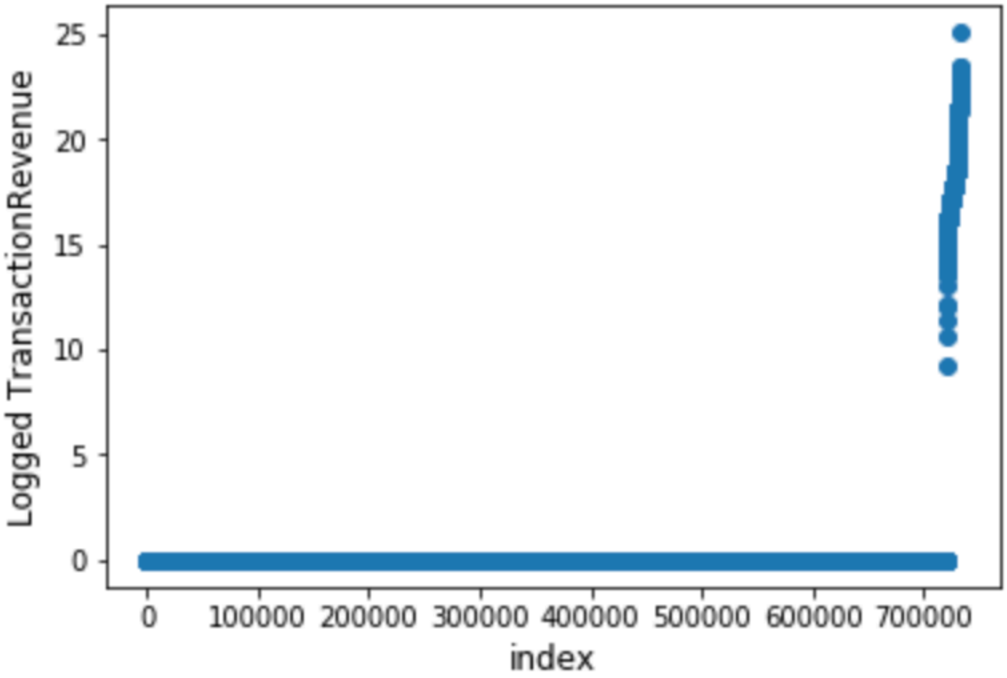


Figure 1 Transaction Revenue Distribution

As shown in the graph, a small proportion of the customers produce most of the revenue, which aligns with the 80/20 rule. These high-value customers are the customers GStore should target. In this project, I would try to use machine learning models to pinpoint these customer segments based on their identities and behavior patterns.

Control Features Exploration

I explored control features in order to see if they would be influential in the model.

Device Information

Device features provide insights into whether device browsers, operating systems and categories would impact on users 'willingness to purchase at the GStore.

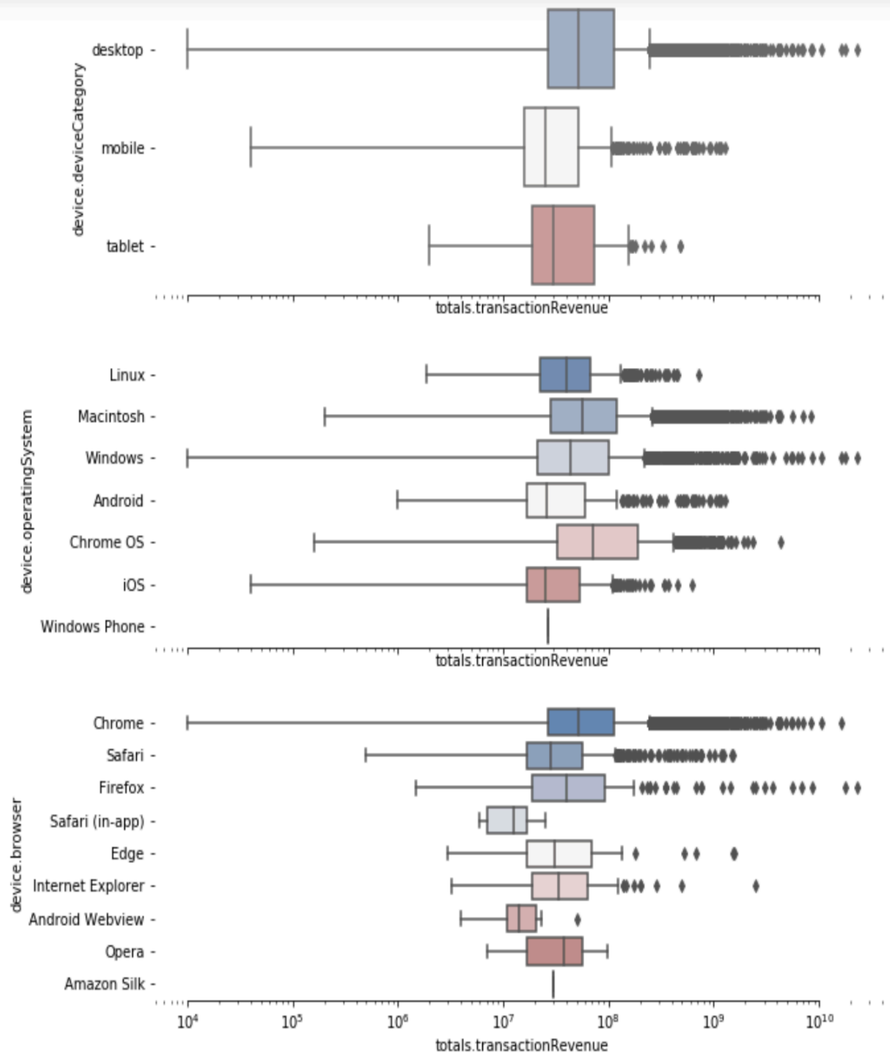


Figure 2 Device Information

From the boxplots above, we see that customers' transactions do differ by the device features. For example, desktop and tablet users have more purchases than mobile users. Unsurprisingly, the operating system graph shows that Chrome OS users have the highest transactions; while the device browser graph shows that Safari browser users have the lower median values in transactions.

As we do not have that many raw device feature values in the dataset, I kept all of them in the final projects without grouping or excluding.

Country Data

There are 222 unique countries in the dataset. The boxplot below shows the top 20 countries with the high median values in transaction revenues.

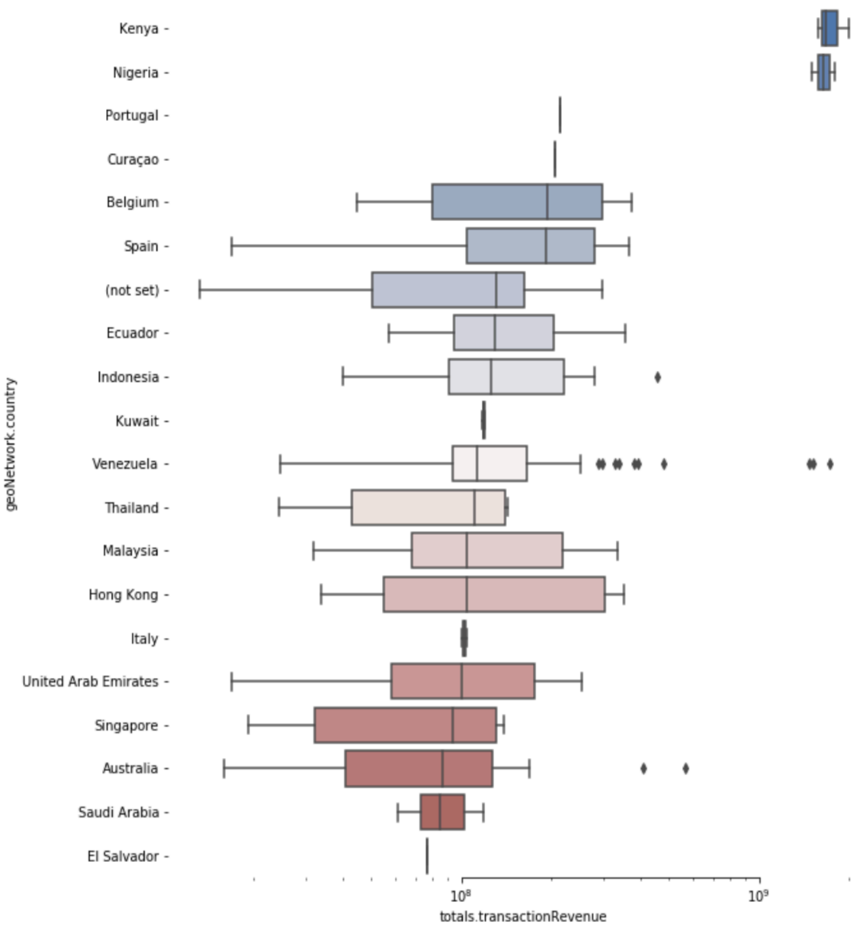


Figure 3 Country Information

From the graph above, we can see that there are clearly countries with very similar patterns in transaction revenue. Also, country has quite many categories and our model will not perform very efficiently with these many values. Thus, I would reduce the country values using Kmeans in the next part.

Channel Grouping

Channel grouping shows the specific medium by which an advertiser's message is conveyed to its audience. The graph below shows the transaction revenues by medium channels. It shows that Direct (mail, email and texting) and Organic (the act of getting your customers to come to your naturally) marketing strategies are the most impactful marketing strategies here.

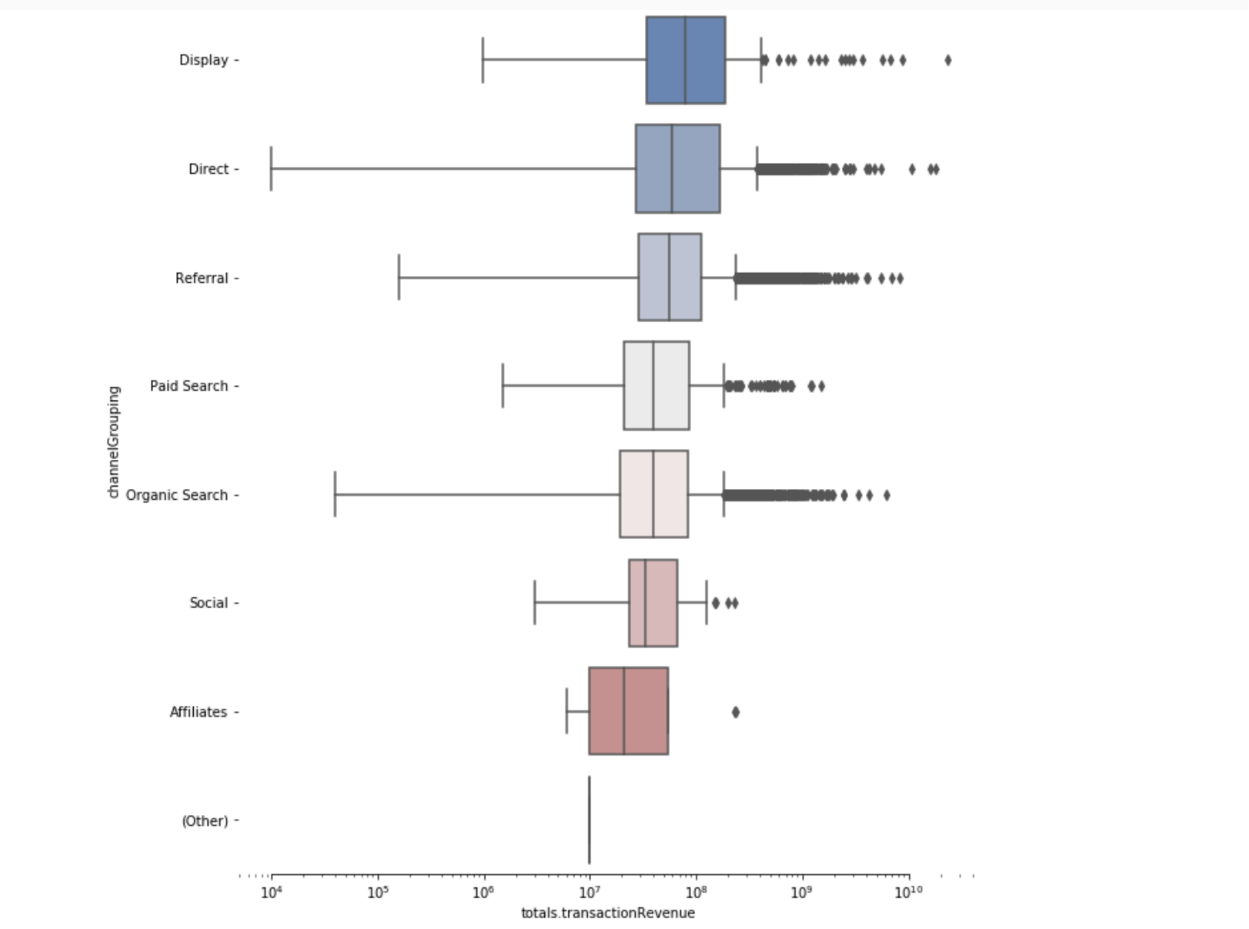


Figure 4 Channel Grouping

As we only have 8 unique media channels in the dataset, I kept all of them in the final model without grouping or excluding.

Pageviews Distribution

I used pageviews as an indirect outcome in this project. While direct outcome focuses on conversions and revenues, indirect outcome is about building brand awareness and creating brand familiarity with their potential customers. Direct and indirect outcome can be highly correlated. The graph below shows the distribution of pageviews in the dataset. We can see that pageviews have a very similar pattern as our outcome – the transaction data. It will be a good indirect indicator for our outcome data.

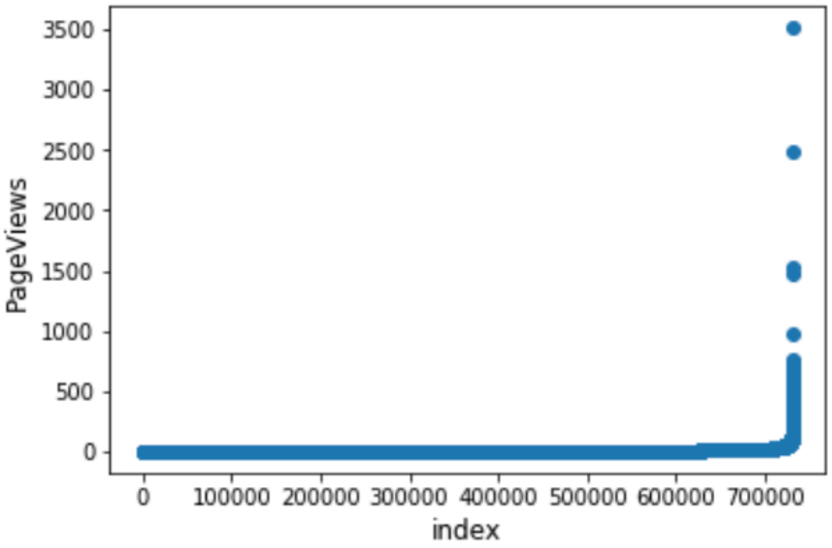


Figure 5 PageViews Distribution

Feature Engineering

I applied Recency- Frequency- Monetary (RFM) value to create customer segments, which divides customers into low value, mid value and high value segments based on their transaction revenue.

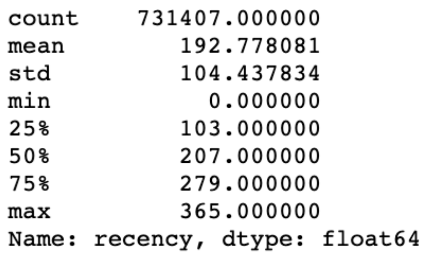
- Low Value: Customers who are less active than others, not very frequent buyer/visitor and generates very low or zero revenue.
- Mid Value: Often using our platform (but not as much as our High Values), fairly frequent and generates moderate revenue.
- High Value: The group we don't want to lose. High Revenue, Frequency and low Inactivity.

First, I need to select a time window. Since the dataset provides exactly one-year data, I choose 12 months as the modeled time window.

Theoretically, customer lifetime value is built on revenue; however, we will not have transaction data in the test dataset as our ultimate goal is to predict transaction revenue. As discussed above, pageviews is a good indicator for transaction, thus I use pageviews here to find the ideal clusters for RFM.

Recency

To measure recency, I subtract the most recent pageviews date of each customer from the maximum data in the dataset to calculate how many days they are inactive for.



```
count    731407.000000
mean        192.778081
std         104.437834
min           0.000000
25%         103.000000
50%         207.000000
75%         279.000000
max         365.000000
Name: recency, dtype: float64
```

We see that the average is 192-day recency, median 207.

Then I applied K-means clustering to assign customers a recency score. I used Elbow Method to find the optimal cluster number for optimal inertia for K-means algorithm.

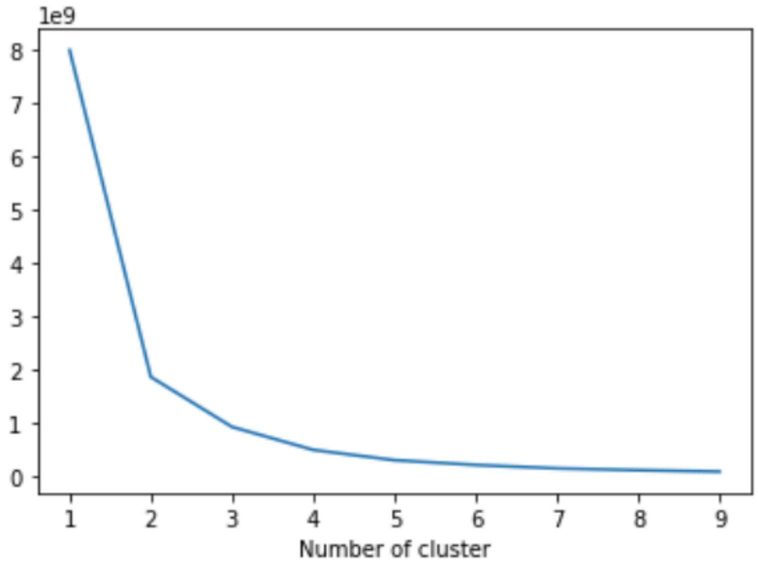
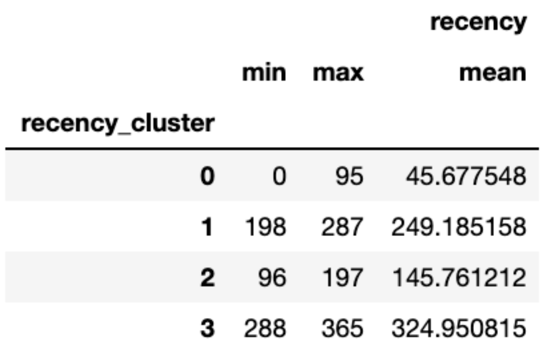


Figure 6 Elbow Graph for Recency

Here it looks like 4 is the optimal one. I calculated clusters and assigned them to each customer in the dataset.

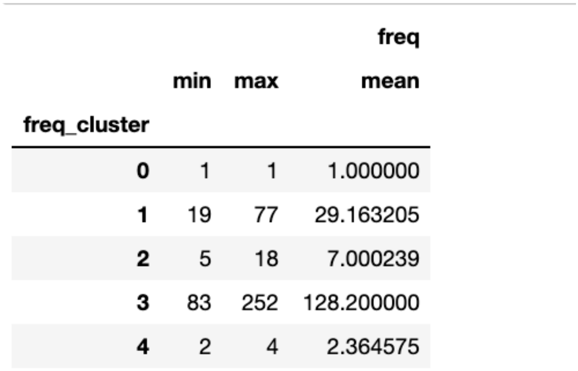


	recency		
	min	max	mean
recency_cluster			
0	0	95	45.677548
1	198	287	249.185158
2	96	197	145.761212
3	288	365	324.950815

We can see that our recency clusters have different characteristics. The customers in cluster 0 is very recent compared with cluster 3.

Frequency

To measure frequency, I found the total number of pageviews for each customer in the dataset. And then I applied the same logic for having frequency clusters and assign this to each customer.

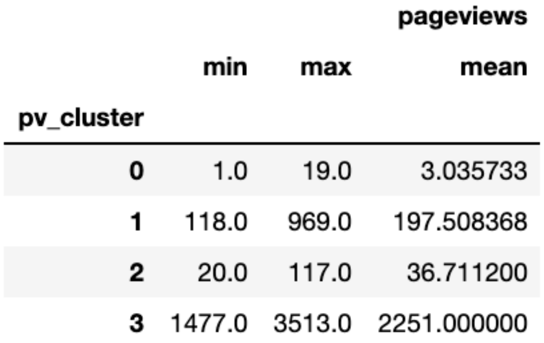


			freq
	min	max	mean
freq_cluster			
0	1	1	1.000000
1	19	77	29.163205
2	5	18	7.000239
3	83	252	128.200000
4	2	4	2.364575

As the same notation as recency clusters, high frequency number indicates better customers.

Pageviews

I calculated total page vies for each customer and applied the same clustering method.



			pageviews
	min	max	mean
pv_cluster			
0	1.0	19.0	3.035733
1	118.0	969.0	197.508368
2	20.0	117.0	36.711200
3	1477.0	3513.0	2251.000000

Country

As discussed before, we have too many country values in the dataset and we found very similar transaction revenue patterns in countries, so I applied the same K-means clustering method for countries as well.

Instead of applying the actual pageviews data, I calculated standard deviation, median, sum and mean of the pageviews for each country and used MinMax Scaler to normalize the values and then fed them into the Kmeans models as features.

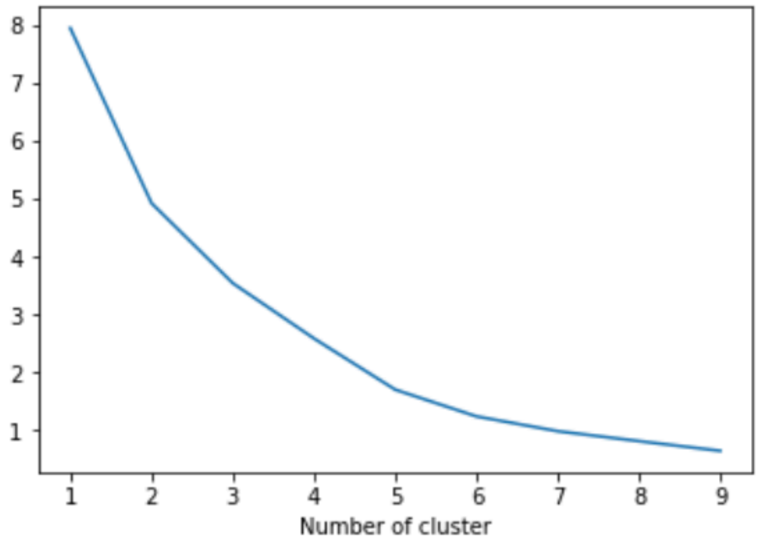


Figure 7 Elbow Method for Country Data

Here it looks like 6 is the optimal one. I calculated clusters and assigned them to each country in the dataset.

Modeling

Feature Selection

Before feeding data into the model, I applied one hot encoding to the control and clustering features to convert them into binary vectors.

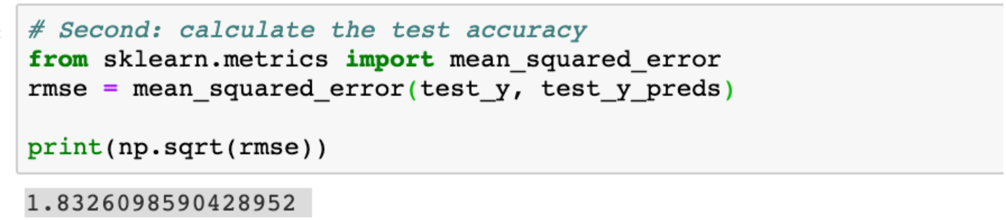
I selected the following features to feed into the model,

- country_cluster
- channelGrouping
- device.deviceCategory
- device.operatingSystem
- recency_cluster
- frequency_cluster
- pageviews_cluster
- totals.hits

Baseline Model: Linear Regression

I used sagemaker.sklearn.estimator with a train.py script called sklearn's LinearRegression to run the Linear Regression Model.

I used mean_squared_errors to measure the model accuracy.



```
# Second: calculate the test accuracy
from sklearn.metrics import mean_squared_error
rmse = mean_squared_error(test_y, test_y_preds)

print(np.sqrt(rmse))
```

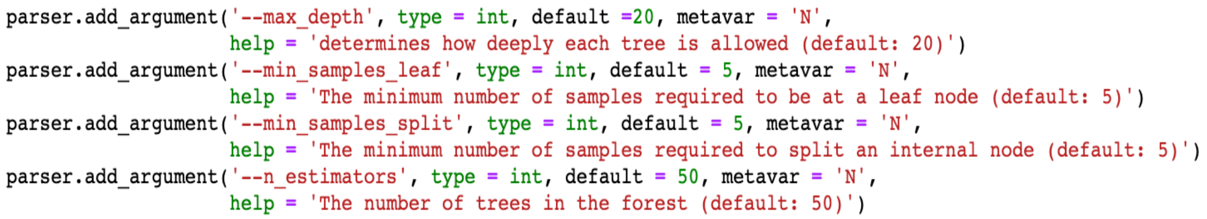
```
1.8326098590428952
```

Here I got a score of 1.8326.

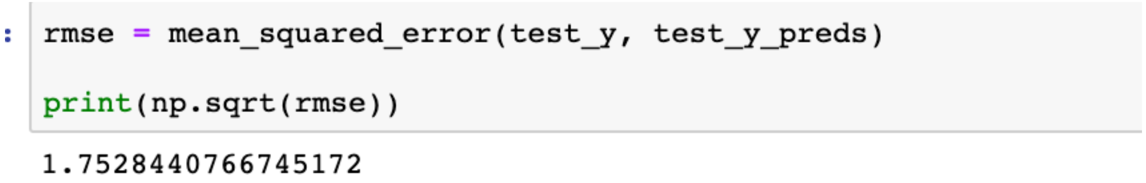
Final Model: Random Forest Regressor

Model Results

To improve the model accuracy, I decided to try a Random Forest Regressor. In my train-rf.py script, , I tuned the following hyperparamters, n_estimators, max_depth, min_samples_leaf, and min_samples_split and applied values as shown below.



```
parser.add_argument('--max_depth', type = int, default =20, metavar = 'N',
                    help = 'determines how deeply each tree is allowed (default: 20)')
parser.add_argument('--min_samples_leaf', type = int, default = 5, metavar = 'N',
                    help = 'The minimum number of samples required to be at a leaf node (default: 5)')
parser.add_argument('--min_samples_split', type = int, default = 5, metavar = 'N',
                    help = 'The minimum number of samples required to split an internal node (default: 5)')
parser.add_argument('--n_estimators', type = int, default = 50, metavar = 'N',
                    help = 'The number of trees in the forest (default: 50)')
```



```
rmse = mean_squared_error(test_y, test_y_preds)

print(np.sqrt(rmse))
```

```
1.7528440766745172
```

The model results got slightly improved. I got a model score of 1.7528 using root mean square error (RMSE)

Feature Importance

Here are the 10 most impactful features in the Random Forest model.

```
[('totals.hits', 0.6336648202425447),
 ('country_cluster_4', 0.1123653855535342),
 ('channelGrouping_Referral', 0.027625023802095236),
 ('freq_cluster_3', 0.025380133911585574),
 ('device.operatingSystem_Macintosh', 0.02252476405826257),
 ('recency_cluster_1', 0.020440476776506834),
 ('recency_cluster_3', 0.02016306617769529),
 ('recency_cluster_2', 0.020065506959174542),
 ('pv_cluster_1', 0.017621169694255895),
 ('freq_cluster_1', 0.016147598690270396)]
```

Unsurprisingly, total hits, as the indirect outcome, has the highest impact on the transactions. Country cluster 4 has the second highest impact. The cluster only contains the United States, which has the most transactions and highest transaction revenue in total.

Recency, pageviews and frequency clusters all have relatively high feature importance, which shows the importance of our lifetime customer value segments. Thus, the GStore can easily adapt their marketing actions based on customer groups.

Conclusion

Compared with the leaderboard on Kaggle with the lowest RMSE of 0.8814, there is a lot to be improved in my current model. Possible actions to improve include,

- Adding more control features
- Trying a different model, like XGBoost and Gradient Boosting Machine
- Applying automated hyperparameter tuning to find the optimal values for hyperparameters
- Adding more data to the model (GStore provided another test dataset)

I chose this project to learn about customer value prediction. Through the capstone project, I learned how to choose metrics to measure customer segments, cluster customer groups using RFM and predict customer values using Sagemaker. And now I can easily adapt these actions to other datasets to predict the future customer lifetime value and their revenue.

Reference

- https://towardsdatascience.com/data-driven-growth-with-python-part-2-customer-segmentation-5c019d150444
- https://www.kaggle.com/c/ga-customer-revenue-prediction/leaderboard