Making Marketing Strategy Recommendations using Machine Learning on Google Analytics Data

Yusuf Ahmed
Chemical Engineering
Queens University
Kingston, Canada
13ya9@queensu.ca

Jesse Khaira
Chemical Engineering
Queens University
Kingston, Canada
13jk59@queensu.ca

Calvin Tam
Chemical Engineering
Queens University
Kingston, Canada
calvin.tam@queensu.ca

Relevant code for all sections can be found in the accompanying Jupyter Notebook markdown attached to this paper.

I. Introduction

This paper investigates the analysis and mining of a Google Merchandise Store's (GStore) customer data. The motivation for this is that many businesses see the bulk of their revenue coming from a small proportion of their customers. Marketing teams are consistently challenged with identifying their most profitable customers to effectively allocate their budget. Therefore, the aim of this paper is to gain domain expertise through combining insights gained from exploratory data analysis along with feature importance from machine learning models to provide actionable business recommendations for this store.

II. Related Work

Most revenue forecasting models can be split into two approaches: regression and time series.

A. Regression

Within regression, a common application is sales forecasting. This assumes that the patterns in historical data will be repeated. Stacking approach is common here, where the results of multiple models on the validation set serve as inputs for another set of models [1]. A study by EY on predicting company profitability found that Gradient Boosting was superior to Random Forest models, as measured by RMSE [2]. Another comparison of predictor models found that LightGBM was superior, followed by Neural Network and XGBoost. LightGBM and XGBoost have become the state-of-the-art models as a result of the additional processing speed and prediction performance, which is particularly useful for massive datasets [3].

B. Time Series

Alternatively, forecasting can be done through a time series model. This differs from most machine learning models in that data points are recorded at equally spaced points in time. For sales forecasting applications, this has involved tools based on autoregression and moving averages [4]. A comparison of time series and regression models for revenue forecasts showed that regression models minimized total forecast error [5].

III. Dataset and Features

The training data consisted of millions of rows of data and the testing data consisted of hundreds of thousands of rows of data. Each row in the dataset represented a customer's information and interaction with the store's website. The data pre-processing steps and exploratory analysis that was conducted on this dataset will be discussed in detail in the following sub-sections.

A. Random Subsampling Function

The size of the dataset posed a significant problem. This was due to both computational and time constraints, with computers not being able to read this dataset into memory and a lack of time to explore more complex solutions to load the dataset into memory. It was decided that to proceed, a 1% portion of the training and test sets would be randomly sampled without replacement into memory. A function was created for this purpose, being initialized through a random seed to ensure reproducibility of results.

B. JSON Parsing Function

Additionally, many of the features were present in JSON queries, posing an additional difficulty as these features could not be used directly. To address this issue, a JSON parsing function was successfully defined in order to access the data contained in the fields more efficiently.

C. Exploratory Analysis

The first step completed in the exploratory analysis was a constant columns and missing values analysis. The analysis found that there were many features in the dataset that had completely constant values, as well as many features with over 95% missing values, including the target customer revenue. This will be discussed further in feature engineering.

Trend analysis was the next step completed in exploratory analysis, with the goal of gaining insights into the domain. Only 1% of customers who visit the website end up paying. Seasonality is apparent in the customers' data, with clear revenue spikes just before the Christmas holiday season, as can be seen in Figure 1.

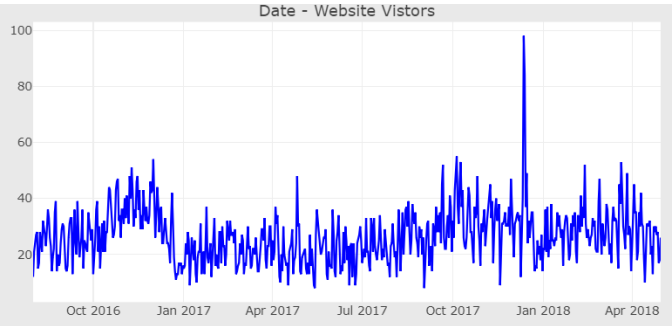

Figure 1: Graph of website visitor count versus date from Oct 2016 – Apr 2018

Although this website attracts global attention, North American customers are the most profitable, as measured by a ratio of paying customers to visiting customers. Western and Eastern Africa have the largest mean revenue generated, albeit at a minute sample size. This is shown in Figure 2.

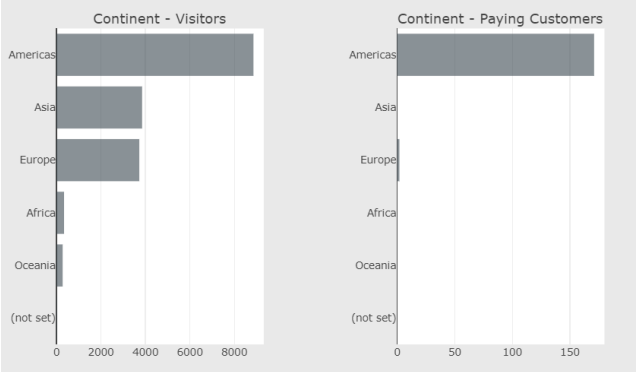

Figure 2: Graphs of website visitor count versus categorized global geographies

The customers most likely to make purchases come to the website through CPM ads, or ads that are run with a price based on cost per 1,000 impressions. Despite this, most of the traffic comes from organic and referral sources, with referred customers bringing in most of the store's revenue. Customers coming from YouTube account for the second largest amount of store visits, but an insignificant amount of revenue. The majority of paying customers access the website using their desktop via Chrome. As one could expect, more page hits and increased session quality result in higher paying customers, on average. This is shown in Figure 3.

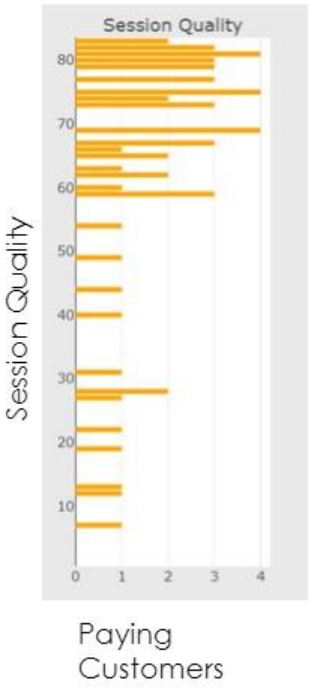

Figure 3: Graph of session quality versus number of paying customers

Clear trends have been identified that demonstrate that most of the revenue is derived from a small subset of GStore visitors. Primarily, almost all revenue comes from North America, from customers that reported a high "session quality". It is expected that these same characteristics will be seen in the feature importances derived from the models later in the paper.

D. Feature Engineering

A feature engineering function was included as the final data preprocessing step before the predictive models. In this function, features that were deemed to be similar to the target feature of total transaction revenue were removed to ensure the validity of the results obtained. In addition, missing values for the target were imputed with a zero, with the justification that missing values indicated customers that had come to the store and not bought anything. Features found in the exploratory analysis that had constant values and missing values (95% or more) were removed. Imputing values for these features was considered but ultimately ruled out with the rationale that imputing values for essentially an entire feature would likely add no analytical value. Features with intrinsically no analytical value were removed, such as customerID. Further preprocessing was conducted on the remaining features in the interest of preparing them for the predictive models. This included one-hot encoding the categoric features and converting the numeric features to float.

No feature transformation was performed within this step as the objective of the project was to obtain domain expertise and subsequently provide actionable business recommendations. Feature transformation is more suited to the task of building the most accurate predictive models.

IV. Methods

The two models implemented for this regression task were the XGBoost and LightGBM models. The primary reason these models were chosen was because they produce reliable feature importances, along with also producing top-tier predictive results [6]. The feature importances and the results obtained from both of these models will be compared in the Discussion section of the report.

A. Gradient Boosting Decision Tree (GBDT) Algorithm

Both XGBoost and the LightGBM models implement the GBDT algorithm with modifications. GBDT is defined as a sequential ensemble of decision trees, meaning subsequent decision trees in the ensemble attempt to improve upon previous decision trees errors. Within each iteration, the decision trees are learned through fitting the residual errors, also known as negative gradients. All models in the ensemble are summed to make the final prediction [7] [8].

B. XGBoost

The objective function for XGBoost can be expressed mathematically as follows:

$$Objective = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega\,(f_k)$$

In this objective function, l represents the training loss function. The training loss function is customizable, meaning it can take on many forms such as MSE, logistic regression, and pairwise ranking among others. This function takes its input as the real target values y_i and the predicted target values from the model $\hat{y}_i$. This term measures overall how well the model fits on the training data. The second term in the equation Ω represents the regularization term. This function takes its input as function f_k, each of which coincides to an independent tree structure and leaf weights. This measures the complexity of the decision trees being built, allowing the model to avoid overfitting issues [8] [9].

C. LightGBM

A major drawback to GBDT is that for each feature in a dataset, all the data instances must be analyzed in the interest of determining the best possible split point. This basically means that for larger datasets, GBDT becomes very time consuming to run. The LightGBM addresses this issue by building upon the GBDT algorithm with two techniques called Gradient-based One Side Sampling (GOSS) and Exclusive Feature Bundling (EFB). GOSS is a sampling method for the GBDT that LightGBM implements that achieves a favourable balance between reducing the number of data instances while retaining prediction accuracy for learned decision trees. EFB acts to reduce the number of features, relying on the sparsity of high-dimensional data to bundle together mutually exclusive features into one dimension. Experimental results have shown that implementing both techniques on top of the GBDT has allowed the LightGBM to drastically outperform XGBoost in terms of computational time [7].

V. Discussion

A. Validation Set

A validation set was constructed in the interest of mitigating overfitting issues in the final model along with performing hyperparameter tuning. This set was constructed through dividing on the date feature with a typical 80/20 split into training and validation set respectively.

B. Hyperparameter Tuning

Hyperparameter tuning was completed on the training set with a model in the scikit learn library in Python called RandomizedSearchCV, selected for use because of its flexibility and efficiency. When hyperparameter tuning with this model, two cross validation (CV) folds with 50 iterations each were used for the XGBoost, with three CV folds and 100 iterations each used for LightGBM. The difference between the two models stood out here, as tuning took 9 minutes for LightGBM, as opposed to the 29 minutes required by XGBoost for a smaller number of folds. The final optimized hyperparameters can be found in Table 2 and Table 3 in the Appendix.

Additional CV folds with more iterations could have been used when tuning the hyperparameters for both models but was decided against for two reasons. The first and most important reason is that hyperparameter tuning will most likely not have a massive impact on the final results obtained for the feature importances, as the most predictive and important features will remain the most predictive and important features regardless of a slight difference in hyperparameters. The second was due to a time constraint. This constraint showed up as it was initally attempted to use the same number of CV folds and iterations for both models, but it was found that the XGBoost ran for an extraordinarily long time when fed with the same parameters for hyperparameter tuning as the LightGBM.

C. Model Construction

Both of the models were built using early stopping with the training set and validation set. Early stopping was used as a technique to mitigate overfitting, while performing additional hyperparameter tuning through setting the number of estimators in the model. The other hyperparameters determined in the previous subsection remain the same. A figure of the early stopping process for the LightGBM is shown in Figure 4 with the final number of estimators at 950. Early stopping for the XGBoost model resulted in 59 estimators.

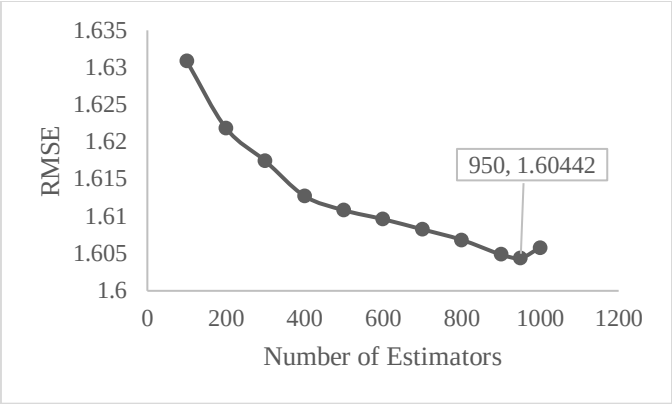

Figure 4: RMSE versus Number of Estimators for LightGBM showing Early Stopping at 950 Rounds

D. Model Evaluation

After the model was built, predictions were made using the optimal model on the test set as determined by early stopping. Both models were evaluated on three metrics: mean square error (MSE), mean absolute error (MAE), root mean square error (RMSE). The primary metric to note, as outlined in the challenge, is RMSE. The final results are shown in Table 1.

Table 1: Table with model performance for XGBoost and LightGBM on testing and validation sets

XGBoost	**Testing Set**			**Validation Set**		
	MSE	MAE	RMSE	MSE	MAE	RMSE
	3.65	0.35	1.91	2.61	0.34	1.62
LGBM	**Testing Set**			**Validation Set**		
	MSE	MAE	RMSE	MSE	MAE	RMSE
	3.41	0.48	1.85	2.57	0.40	1.60

There are a few things to notice from the results. In addition to being the faster model to run, LightGBM shows better accuracy on MSE and RMSE, the relevant metric in this case, but performs worse on MAE. Another key observation is that both models perform only slightly worse on the test set, indicating that there aren't large issues stemming from overfit. It is also important to note that neither the LightGBM and XGBoost models seemed to generalize significantly better than the other, as prediction accuracy decreased almost equally from the validation set to the testing set for both models. Interestingly, during early stopping, the LGBM model produced an optimum value of 950 estimators, compared to 59 estimators produced by the XGBoost model.

E. Feature Importance

The feature importances obtained from both of the models are shown in Figure 5 and Figure 6. The LightGBM model returned time on site, network domain and session quality as the most important features. The XGBoost model returned session quality, hits and time on site as the top 3 features.

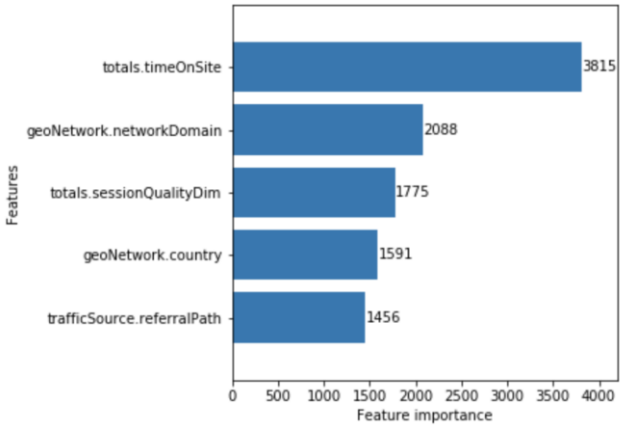

Figure 5: LightGBM Feature Importances

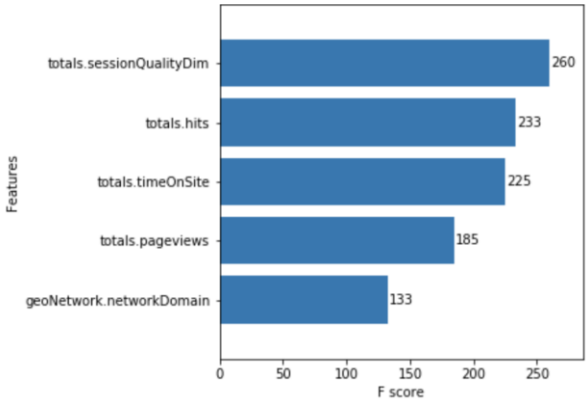

Figure 6: XGBoost Feature Importances

There was general agreement between the two models, with session quality, time spent on site, and network domain scoring in the top 5 features for both models.

F. Business Reccomendations

The models suggested that the most important features for customer revenue are their geographic location, and the way they interacted with the website. Exploratory analysis of the data determined that there are significant trends observable in the time of year. Profitable customers were also observed to come from referrals. As a result, there are 5 actionable changes recommended to the GStore.

1. *Website optimization*

Time spent on the website and the session quality are two of the most predictive features for customer spend. By improving the customer's web experience and increasing the time spent on the website, the GStore can expect an increase in customer spend. This can be done by focusing on the user interface of the website to ensure each page is engaging. Common practices also include frequent linking to other products for easier navigation and increased usage.

2. *Seasonal promotions*

Customers of the GStore used the website the most just before the holiday season. The store can take advantage of this by offering promotions for other holidays, like Easter weekend and Cyber Monday.

3. *Referral program*

The bulk of the store's revenue comes from customers that have been referred. This can become the primary channel for traffic through the creation of a referral program. This would give existing customers a discount if they refer a friend who makes a purchase. The discount provided will be made up by bringing the most profitable segment of customers.

4. *North American focus*

Although the store attracts global attention, users from outside of North America appear to be browsing but not purchasing anything. As such, the store's marketing focus should be placed on North American customers.

VI. Conclusion

After using the LightGBM and XGBoost models, it was apparent that the most important features were the customer's location and user experience. Exploratory analysis also highlighted the significance of referred customers and seasonal promotions to store traffic. The GStore can capitalize on this by improving their UI/UX, suggesting product recommendations, marketing to North American customers, implementing a referral program and offering multiple seasonal promotions throughout the year.

The LightGBM model was deemed to be superior as it scored better on MSE and RMSE. It also computed a larger number of folds in significantly less time. This is especially crucial for future work, which would involve modelling at much larger data set sizes. With greater computational resources and more time, the problem could have been explored using parallel dataframes and the full dataset. Notably, due to computational limitations, a Kaggle submission could not be made and so this project's results on the leaderboard were not determined.

Finally, it would be interesting to investigate the same problem using unsupervised techniques, such as high-dimensional clustering. This could highlight profitable groups of customers that may have been missed. It would also be interesting to investigate whether the findings for the marketing team can be applied to other parts of Google's business, such as the Google Play Store.

VII. Contributions

Yusuf Ahmed: Exploratory analysis, Business recommendations, related work, feature importance

Jesse Khaira: Data pre-processing, feature engineering, construction of predictive models, evaluation of model results, feature importance

Calvin Tam: Introduction, future work, related work, Business recommendations, exploratory analysis

VIII. References

[1] B. Pavlyshenko, "Machine-Learning Models for Sales Time Series Forecasting," in *2018 IEEE Second International Conference on Data Stream Mining & Processing (DSMP)*, Lviv, Ukraine, 2018.

[2] M. Wintgens, "Predicting the profitability level of companies regarding the five comparability factors," EY, Vrije Universiteir Amsterdam, 2017.

[3] H. Li, "An Ensemble Approach to Streaming Service Churn Prediction," 2018. [Online]. Available: https://wsdm-cup-2018.kkbox.events/pdf/9_WSDM%20Cup-VinaKago-An_Ensemble_Approach_to_Streaming_Service_Churn_Prediction-v2.pdf.

[4] G. Shine, "Sales Prediction with Time Series Modelling," [Online].

[5] A. Gajewar, "Revenue Forecasting for Enterprise Products," [Online]. Available: https://arxiv.org/ftp/arxiv/papers/1701/1701.06624.pdf.

[6] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," 2016. [Online]. Available: https://www.kdd.org/kdd2016/papers/files/rfp0697-chenAemb.pdf. [Accessed 20 March 2019].

[7] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye and T.-Y. Liu, "LightGBM: A Highly Efficient Gradient Boosting Decision Tree," 4 December 2017. [Online]. Available: https://papers.nips.cc/paper/6907-lightgbm-a-highly-efficient-gradient-boosting-decision-tree.pdf. [Accessed 20 March 2019].

[8] Y. Tian, *CISC 351 Lecture 5: Gradient Boosting Machine, XGBoost*, Kingston, Ontario: Queen's University, School of Computing, 2019.

[9] XGBoost Developers, "Introduction to Boosted Trees," 2016. [Online]. Available: https://xgboost.readthedocs.io/en/latest/tutorials/model.html. [Accessed 20 March 2019].

IX. APPENDIX

A. Final Tuned Hyperparameters

Table 2: Tabulated Optimized Hyperparameters for XGBoost

Colsample_bytree	0.5454951393008803
Gamma	4.834820481469037
Learning_rate	0.050158252068833425
Max_depth	36
Min_child_weight	1
Reg_alpha	100
Subsample	0.9278827742521691
Objective	Reg:linear
Eval_metric	Rmse
Silent	True
Random_State	40

Table 3: Tabulated Optimized Hyperparameters for Light GBM

Colsample_bytree	0.8539721005885168
Learning_rate	0.05722266438555726
Min_child_samples	453
Min_child_weight	1
Num_leaves	35
Subsample	0.6218338710274947
Objective	Regression
Metric	Rmse
Verbosity	-1