Customer Revenue Prediction

First Draft for DSA 5013

Azadeh Gilanpour, Precious Jatau and Zachary White

Executive Summary

Marketing teams have always had to try to identify which customers are most likely to make a purchase. Google and Kaggle released a large dataset for a competition to see who could identify which customers would likely purchase and product. We took this dataset with the intention of trying to predict whether if a purchase would be made.

A major problem with the dataset is that it is approximately 17 million observations with 5 of the 12 variables in a JSON format and 1% of the observations were positive transactions. Through work done on a kernel on Kaggle, the JSON format variables were flattened to create 60 variables. The data was cleaned by removing any variables with more than 40% of its data missing, variables with its factors containing too many unique values, variables with near zero variance, and variables that were repetitive to other variables or had a high correlation. We created 4 new variables: *weekday* (the day of the week), *month*, *year*, and *TransactionRevenue* (the target variable that is 1 for a positive transaction and 0 for a non-transaction). The data could run on any method with its large size, so we reduced the data size by taking the entire 1% of the positive transactions and a random 4% sample of the non-transactions to make a new data that was had a 70/30 split to create training and testing data respectively.

Four models were chosen to run the dataset for comparison: logistic regression, decision trees, random forest, and gradient boosted decision trees. The method for comparing the four models was compute the accuracy percentage, to see how precise each model was, and the kappa percentage, to ensure it was meant to be accurate and the model did not get a high accuracy by chance. These methods were computed through the confusion matrix.

We found that all four models had a high accuracy and kappa percentage with random forest having the highest accuracy and kappa percentage. We feel that this could be from our sampling method of over sampling the minority case of positive transactions. A future direction for this problem could be adding more categorical models such as support vector machines, increase the non-transaction observations to the data set to see if that changes the accuracy and kappa results, and acquiring hardware build to run datasets with such large size to run the complete dataset and not sample it.

Ultimately, we feel that we succeeded in our comparison of different models with the dataset that we chose by evaluating the accuracy and kappa percentages of each model.

1. Problem Description

1.1 Background

The data set comes from the Kaggle.com competition *Google Analytics Customer Revenue Prediction* (Kaggle). It contains customer information from the Google Merchandise Store with the goal of predicting the revenue per customer, so the marketing team can make better decisions on marketing strategies. Our goal with this dataset is to determine the best method of prediction of whether a customer will make a purchase by comparing multiple categorical models.

1.2 Data Exploration

The dataset from the Kaggle competition contains customer information. It has 1,708,337 observations and 12 variables. There are two major issues when importing the data: the large amount of data and 5 of the 12 rows are in JSON format. The solution we found to import the dataset into R and flatten to JSON parameters were found in a kernel of the Kaggle Competition[2]. After the JSON parameters have been flattened, the dataset increases to 60 parameters. Further inspection of the dataset showed that 18,514 observations were positive transactions. This meant approximately 1% of the data represents a purchase and a majority of these purchases came from North America. After observing this, we began feature selection and transformation.

2. Analysis Plan

2.1 Feature Selection

The missingness within the dataset became apparent after the JSON variables were flattened. Missingness was defined as NA, as well as “Not Socially Engaged”, “not available in demo dataset”, and “(not set)”. Some variables with missing data could be address, the variable *newVisits* and *bounces* each have two values, 1 and NA, which defines presence and absence respectively. *newVisits* is converted to a factor with two levels: “NEW” and “REGULAR”. *bounces* is also converted to a factor with two levels: “BOUNCED” and “UNBOUNCED”. After that, variables missing more than 40% of their data were removed from the dataset (Table 1).

There are 4 variables with missing less than 40% of their data: *pageviews*, *OperatingSystem*, *continent*, and *subcontinent. pageviews*’ missing data was imputed with its median value and *OperatingSystem* was imputed with its most common value (“Windows”). It was observed that the location of the transactions played a big role in determining the outcome, with America having most of positive transactions. With this assumption about the origin of customers having seriously bias classification, listwise deletion was applied to *continent* and *subcontinent*.

Variables that were removed after this had little importance and other reasons. *visitStartTime*, *visitID*, *index*, *Date*, *Country*, *networkDomain*, and *browser* are removed for having too many unique values. *socialEngagementType* and *visits* were removed for having near zero variance. Source, medium, value, and isMobile were removed for displaying information already told by other variables.

Variables created in the data set were *weekDays* (telling day of the week), *month*, and *year* from *Date*. *TransactionRevenue* is the target variable created for the prediction and is a binary variable created from transaction where 1 is a positive transaction and 0 is no transition.

The remaining variables are the following:

channelGrouping	visitNumber	operatingSystem	deviceCategory	continent
subContinent	hits	pageviews	bounces	newVisits
revenueCondition	weekDays	month	year	

Even though the dataset went from 60 variables to 14 variables, the 17 million rows made creating models on home computers difficult. The solution we used around this was to create a dataset that was an over sampling of the minority case. This mean we took all 18,514 observation with positive transactions, which is approximately 1% of the data, and combined with a 4% random sample of the non-transactions. These samples were then each split, where 70% of each sample was put into a training set and 30% of each sample was put into test set. The last issue presented with the dataset was that the test set contained 2 observations where subcontinent “Polynesia” and were not represented in the training set. These observations were removed. The final train set contains 60,202 observations of 14 variables while the test set contains 25,794 observations of 14 variables.

2.2 Models

Four models were chosen for classification: logistic regression, decision trees, random forest, and gradient boosted decision tree. The goal is to see which model would be best to use to classify the dataset. A criteria is set up to compare the models. The accuracy of each model will be compared as well as the kappa coefficient. Since 80% of both the training data and test data are non-transaction, the kappa coefficient will ensure that a high accuracy is not by chance.

2.2.1 Logistic Regression

The ipflasso library was used for classification. A repeated k-fold cross-validation method, cvr.glmnet, was used to tune the logistic regression method, glmnet, to find lambda. It was repeated 5 times and had a 5 fold cross validation. The lambda with the lowest Mean Absolute Error was found to 7.845277e-05 (chart 1).

2.2.2 Decision Tree

The rpart library was used for classification. An initial tree was created with a cp = .0001 and was cross validated with xval = 20. When observing the graph of cp vs x-val relative error (chart 2), cp = 0.0078 was used to prune the decision tree (chart 3).

2.2.3 Random Forest

The randomforest library was used for classification. An initial tuning for mtry is performed using bootstrap aggregation. Mtry = 3 was found to be the have the lowest out of bag (OOB) error shown in chart 4. A finer tune is performed to optimize accuracy and the kappa statistic for mtry values between 3 and 10. The results are shown in the plot below. An mtry of 7 gives the simplest model with the best performance (chart 5). This is chosen as the final model with an accuracy of 99.9368% and kappa statistic of 99.8132%.

2.2.4 Gradient Boosted Decision Trees

The xgboost library was used for classification. Xgb.cv running 5 times with a max depth of 100 for 1000 iteration stopping after 100 iteration when it found the best ntreelimit was used to tune xgboost. The best number of iterations to run for Gboosted Tree is 11.

3. Results

Method	Package	Parameters	Accuracy %	Kappa%
Logistic Regression	lpflasso (cvr.glmnet)	Lambda = 7.845277e-05	94.35%	83.13%
Decision Trees	Rpart	cp = 0.0078	94.85%	85.19%
Random Forest	randomForest	mtry = 7	95.25%	86.77%
Gradient Boosted Decision Trees	xgboost	Nrounds = 11	95.08%	85.80%

4. Discussion

The results show that Random Forest have the best accuracy and kappa. Decision Trees and Gradient Boost Trees follow close in accuracy and in kappa. While Logistic Regression has the worst performance of the four, its accuracy and kappa imply that it is a decent method. The high performance of all these methods could be the sampling method used to test the method; the over sampling of the minority case could lead to high accuracy and kappa.

5. Conclusion

Google and Kaggle released a dataset with the goal of identifying which customers are most likely to buy a product. For this project, we took the dataset with the goal of comparing different methods of categorization to predict when a purchase would occur. The first problem encountered was finding a way to read the dataset and flattening 5 of the variables in JSON format so we could begin exploring and analyzing the dataset. A method was found in a kernel on Kaggle to flatten the variables (Kumar). After performing feature selection by removing variables with a definite majority of its data missing, near zero variance, repetitive data, and other criteria, we shrunk the data size by taking the 1% of positive transactions and a 4% random sample of non-transactions.

We tuned four categorical methods for prediction: logistic regression, decision trees, random forest, and gradient boosted decision tree. Our approach for comparing the 4 methods was to look at the accuracy percentage, to see how correct the predictions were on the test set,

and kappa percentage, to ensure that the accuracy was valid due to the skewness of sample. Our results showed that all the methods had a high accuracy with promising kappa values. However, we did find that random forest yielded the best results with the highest accuracy and highest kappa. The high results could have been due to the over sampling the minority case. A future direction that can be taken from this is to change the percentages of sampling method and observe any changes to the accuracy and kappa percentages. Another direction to add other classification methods such as support vector machine to see if that would bring better results. Obtaining better hardware that could store and run the large original dataset would ultimately solve all these problems. In conclusion, we met our goal of comparing the classification methods and found that the four methods chosen would work well with a sample with random forest ultimately bringing the best results.

Reference

Kaggle (2018) Google Analytics Customer Revenue Prediction. Retrieved From Kaggle: https://www.kaggle.com/c/ga-customer-revenue-prediction

Kumar, Amit., preprocessing of the v2 datasets. Retrieved From Kaggle: https://www.kaggle.com/flubber/preprocessing-of-the-v2-datasets

Appendix

Table 1.

Variables missing more that 40% data

```
[1] "browserVersion"                          "browserSize"
 [3] "operatingSystemVersion"                  "mobileDeviceBranding"
 [5] "mobileDeviceModel"                       "mobileInputSelector"
 [7] "mobileDeviceInfo"                        "mobileDeviceMarketingName"
 [9] "flashVersion"                            "language"
[11] "screenColors"                            "screenResolution"
[13] "region"                                  "metro"
[15] "city"                                    "cityId"
[17] "latitude"                                "longitude"
[19] "networkLocation"                         "sessionQualityDim"
[21] "timeOnSite"                              "totalTransactionRevenue"
[23] "campaign"                                "keyword"
[25] "referralPath"                            "isTrueDirect"
[27] "adContent"                               "campaignCode"
[29] "adwordsClickInfo.criteriaParameters" "adwordsClickInfo.page"
[31] "adwordsClickInfo.slot"                   "adwordsClickInfo.gclId"
[33] "adwordsClickInfo.adNetworkType"          "adwordsClickInfo.isVideoAd"
```

Chart 1: Lambda vs Mean Absolute Error for logistic regression

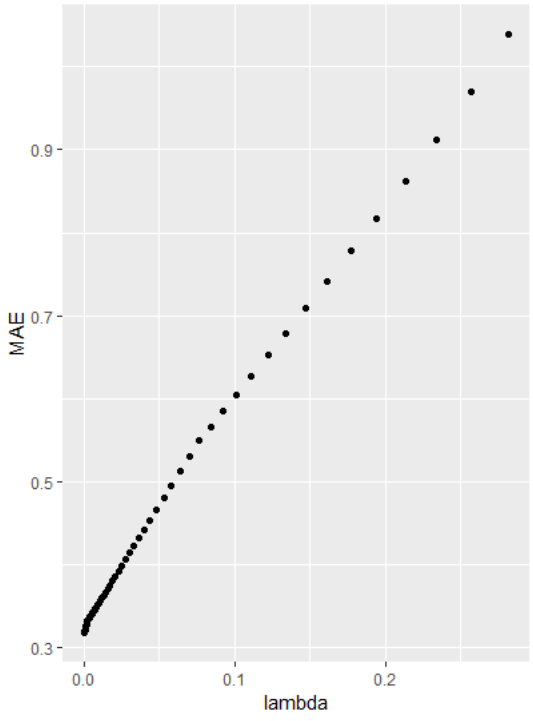

Chart 2: CP vs X-val for decision tree pruning

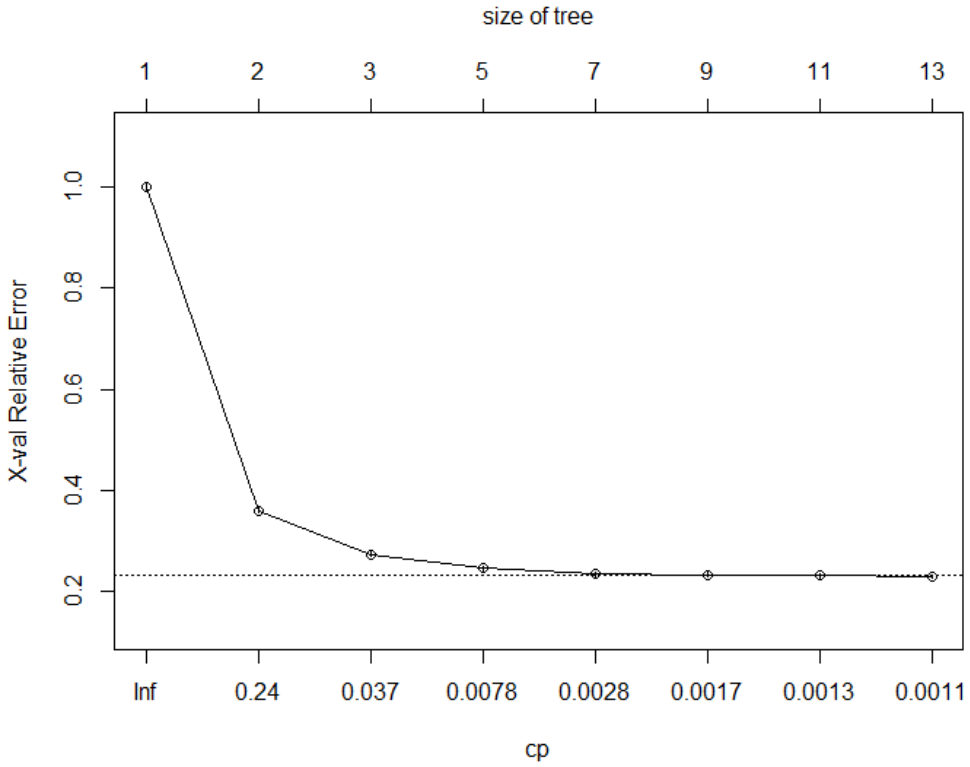

Chart 3: Pruned Tree with cp = 0.0078.

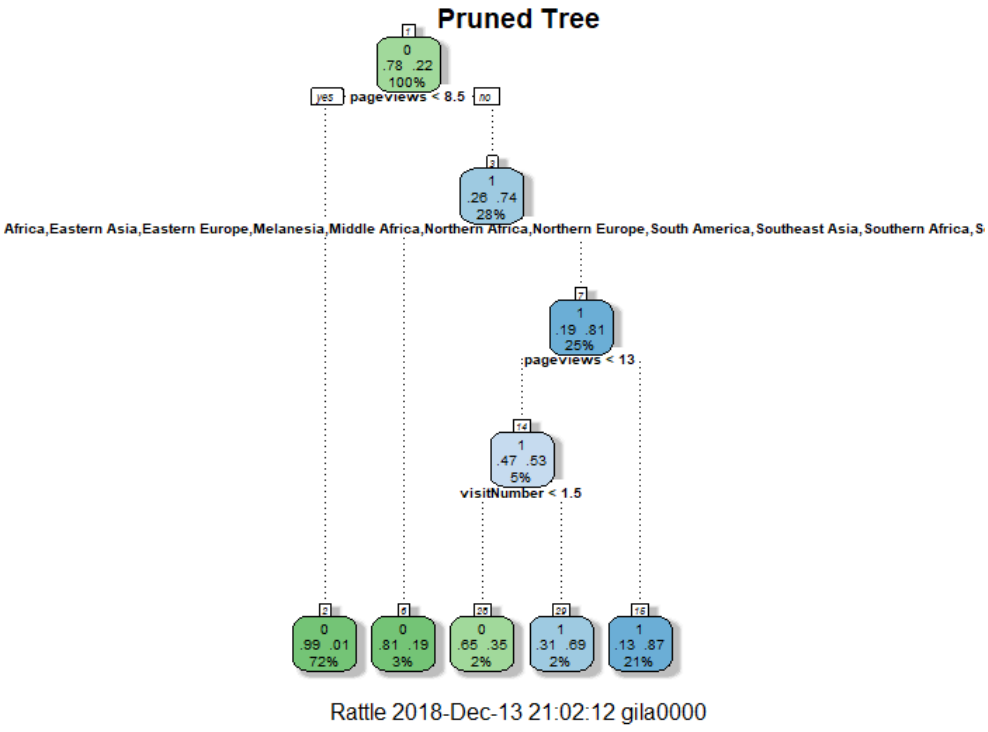

Chart 4 mtry vs OOBError for Random Forest tuning

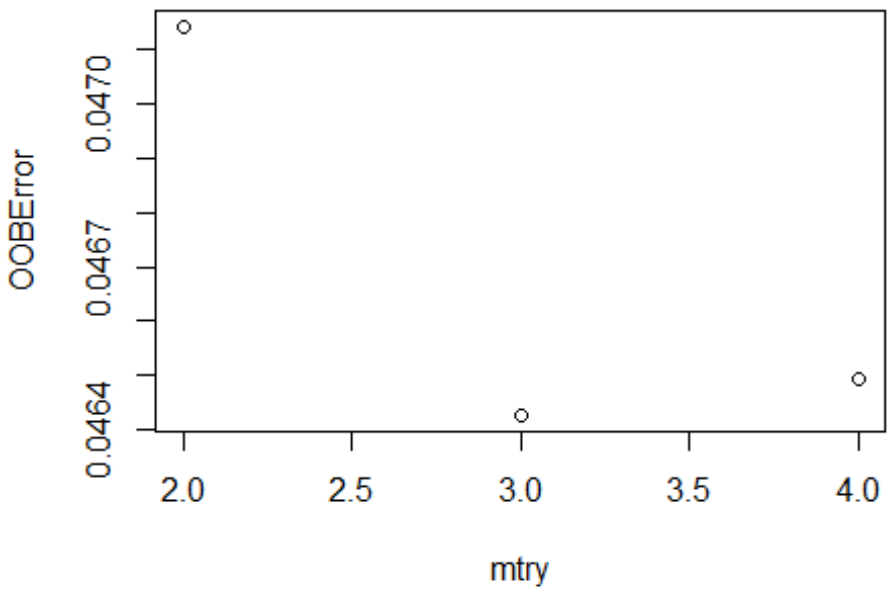

Chart 5: mtry vs accuracy and kappa