Google Analytics Customer Revenue Prediction

EE 660 Project Type: Individual

Anfeng Yu, anfengyu@usc.edu

Date:12/3/2018

**Please organize your report along the lines of this template**; you may use any word processing software you like, as long as you submit your report as the required pdf file described below.

**Your report must be typewritten and submitted as a pdf document**, in machine readable form (no scans or screen shots).

**Please submit your code as a second pdf file**, all code in the one file, also required to be machine readable (no scans or screenshots).

# Abstract

This project is based on a Kaggle competition, Google Analytics Customer Revenue Prediction. In this competition, we are challenged to analyze a Google Merchandise Store (also known as GStore, where Google swag is sold) customer dataset to predict revenue per customer.

This basically is a regression machine learning project.

To complete this project, I firstly cut train dataset into preprocessing set, training set and validation set. To test, I use all test set.

For training and model selection, I used Bayes regression, linear regression, Lasso and Ridge regression, lightGBM (a gradient boosting tree-based algorithm), random forest, and found random forest gives best result.

# Introduction

# Problem Type, Statement and Goals

This is a Regression problem, by using each entry’s (each row’s) data, like userID, date, device, location, number of page views, etc.) to predict revenue of per customer in future, which is the aggregation of per user’s “transactionRevenue” (group by userID).

Problem Difficulty:

1. High dimensionality of feature space. ( Need to do dimension reduction)
2. Sparsity. ( Most entries’ revenue is 0, most customers didn’t purchase anything.)
3. Nonlinear behaviors. ( There is no strong linear relation between features and target.)
4. Significant amounts of preprocessing required. ( Raw data’s features are not able to train, need to be transformed. Mostly features’ type.)

# Literature Review (Optional)

# Prior and Related Work (Mandatory)

Prior and Related Work - None.

# Overview of Approach

**Methods I used:**

Bayes regression: Easy to implement and train, although usually doesn’t give a very good result.

linear regression: Easy to implement and train, can be useful for strong linear model data, but not good for nonlinear model data.

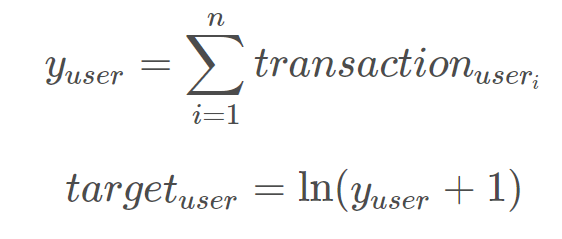
Lasso and Ridge regression: Linear Regression with regularizer, Lasso is easier to produce sparse parameters.

LightGBM: Use python package lightgbm, which is a gradient boosting tree-based algorithm, which is similar to decision tree, easy to train but not very good at prediction. This model can give features’ importance based on tree algorithm, which can be used as feature selection for random forest.

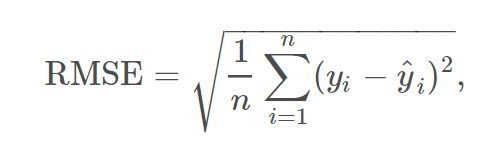
Randomforest**:**

**Metric to measure the result:**

“transactionRevenue” (group by userID).



Use “Root Mean Squared Error (RMSE)” as metric to measure the result.



# Implementation

Report your implementation details and results in the following subsections. You should mention which libraries and functions you used but avoid including code in your report. Your description of what your system does should be readable and understandable to a reader that isn’t familiar with the functions and libraries you used, but is familiar with the algorithms and techniques that were covered in EE 660. (For example, stating “we standardized all real-valued features, and recast all categorical features using one-hot encoding” and also stating the functions used in your code for this, is fine; stating only the functions used in your code is not fine.)

## Data Set

Number of data points:

Train set: 1708337 data points

Test set: 401589 data points

Totally 39 features ( 38 input variables, 1 output variable(target)).

|  |  |  |  |
| --- | --- | --- | --- |
| Features’ Name | Type | Cardinality /Range | Description |
| channelGrouping | string | 8 | The channel via which the user came to the Store. |
| date,visitStartTime | datetime | from 2016-08-01 to 2018-10-15. | The date on which the user visited the Store. |
| fullVisitorId,visitId ,  sessionId | string | 1323730 | A unique identifier for each user of the Google Merchandise Store. |
| visitNumber | int | 457 | The session number for this user. If this is the first session, then this is set to 1. |
| Device  (device\_browser,  device\_deviceCategory,  device\_isMobile ,  device\_operatingSystem) | String  (categorical) | 129 | The specifications for the device used to access the Store |
| geoNetwork  (geoNetwork\_city ,  geoNetwork\_continent ,  geoNetwork\_country ,  geoNetwork\_metro ,  geoNetwork\_networkDomain ,  geoNetwork\_region ,  geoNetwork\_subContinent) | String  (categorical) | 956 | This section contains information about the geography of the user. |
| trafficSource  (trafficSource\_adContent  trafficSource\_adwordsClickInfo.adNetworkType  trafficSource\_adwordsClickInfo.gclId  trafficSource\_adwordsClickInfo.isVideoAd  trafficSource\_adwordsClickInfo.page  trafficSource\_adwordsClickInfo.slot trafficSource\_campaign  trafficSource\_isTrueDirect trafficSource\_keyword  trafficSource\_medium  trafficSource\_referralPath  trafficSource\_source ) | String  (categorical) | 59009 | This section contains information about the Traffic Source from which the session originated |
| totals\_bounces | int | 2 | Number of bounces |
| totals\_hits | int | 297 | Number of hits |
| totals\_newVisits | int | 2 | Number of new visits |
| totals\_pageviews | int | 231 | Number of page views in this session |
| totals\_sessionQualityDim | int | 101 | Number of session quality dimension |
| totals\_timeOnSite | int | 4775 | Time used in this session |
| totals\_transactionRevenue | float | 7252 | Target, the amount of revenue. |

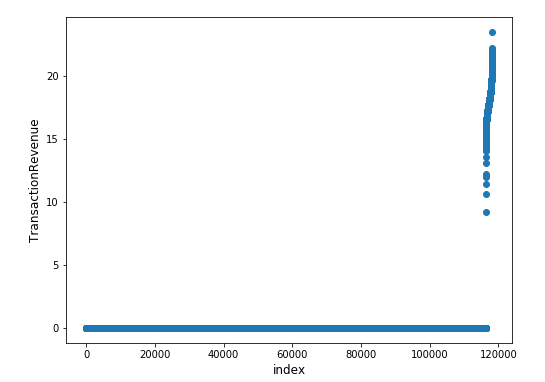
## Preprocessing, Feature Extraction, Dimensionality Adjustment

Describe in detail the pre-processing and feature extraction techniques you used. If you used any dimensionality reduction or sparse coding methods, explain in here as well. If the dataset has missing data, explain how you dealt with it (removing samples, removing features or data filling) and justify.

If you used different pre-processing for different machine learning methods, or if you tested the same machine learning method with different pre-processed inputs, state so. A table can be useful in these cases.

**Pre-processing:**

1. Change “date” column from type “object” into “datetime”.
2. Fill NaN (missing data) with 0 in the numeric features’ columns, like “totals\_transactionRevenue”, “totals\_hits”, “totals\_pageviews”, etc.
3. Change categorical features (in string) into factorized features using LabelEncoder.
4. Calculate the correlation coefficient between target and each features, found most features don’t have a strong correlation coefficient with target (revenue).
5. Draw graph to do data exploration with preprocess set. Draw a scatter plot as below, find most entries’ revenue is 0, only a few have high revenue.



**Feature Extraction:**

Because the random forest is very slow to train for such a large amount of data and features, I use LightGBM to show most important features for decision tree

('channelGrouping','device\_operatingSystem','geoNetwork\_networkDomain','totals\_hits','totals\_pageviews', 'visitStartTime', 'visitNumber'),

then select these features to train random forest.

## Dataset Methodology

The Training data is from 2016-08-01 to 2018-04-30, while the Testing data is from 2018-05-01 to 2018-10-15.

I divided the training data into

\* Preprocessing Set ==> 2016/08/01 ~ 2016/09/30 (2 month) ,145791

\* Training Set ==> 2016/10/01 ~ 2017/12/31 (13 month),1219462

\* Validation Set ==> 2018/01/01 ~ 2018/04/30 (4 month), 343084

And original testing set ==> 2018/05/01 ~ 2015/10/15 (5.5 month) ,401589.

How validation set used:

For each model ( or one model with different hyper parameter), test on validation set, and each get a validation score (RMSE).

In validation phase, the Random Forest get the best result,

RMSE = 3.4324486325215005

How test set used:

Test the best model (Random Forest) on the test set, get

RMSE = 4.094964691288565

Testing set used only once.

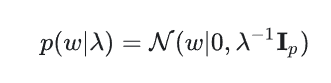
## Training Process

1. **Bayes regression.**

the output y is assumed to be Gaussian distributed around X\*w:

1543985638(1)

With the prior for the parameter w is given by a spherical Gaussian:



The reason for choosing: I chose Bayes regression as a simple baseline model, it’s easy and fast to train.

Hyper parameters: chosen as default1543986008(1)

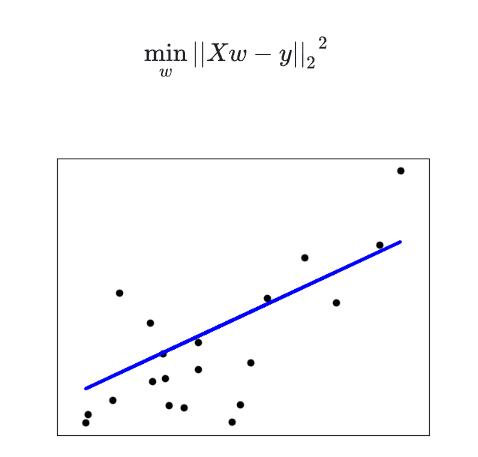
The complexity of your hypothesis set:only 1 model with 4 parameters already set.

Overfitting or underfitting: Not very likely for such a huge amount of data and a simple model.

Result of RMSE: **8.648827446056496**

1. **Linear Regression.**

A simple linear model to minimize the



The reason for choosing: I chose linear regression to find if there is a significant linear relation between target and features.

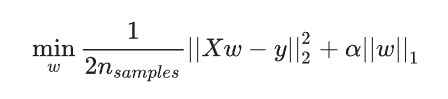
Hyper parameters: No need to choose parameters.

The complexity of your hypothesis set:only 1 model

Overfitting or underfitting: Not very likely for such a huge amount of data and a simple model.

Result of RMSE: **8.664401065570745**

1. **Lasso Regression.**

The reason for choosing: Lasso can give us a more sparse linear model to minimize target function

Hyper parameters: Choose alpha from 0.01 to 100. ([0.01,0.1,1,10,100])

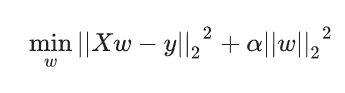
The complexity of your hypothesis set: 5, 1 model with 5 different parameter.

Overfitting or underfitting: Not very likely for such a huge amount of data and a simple model.

Result of RMSE:

|  |  |
| --- | --- |
| Alpha = 0.01 | 8.664327116930957 |
| Alpha = 0.1 | 8.664330868284313 |
| Alpha = 1 | 8.664357106723406 |
| Alpha = 10 | 8.664520874909705 |
| Alpha = 100 | 8.665590489375765 |
| **Best: Alpha = 0.01** | **8.664327116930957** |

1. **Ridge Regression.**

The reason for choosing: Ridge regression can give us a more sparse linear model to minimize target function 

Hyper parameters: Choose alpha from 0.01 to 100. ([0.01,0.1,1,10,100])

The complexity of your hypothesis set:5, 1 model with 5 different parameter.

Overfitting or underfitting: Not very likely for such a huge amount of data and a simple model.

Result of RMSE:

|  |  |
| --- | --- |
| Alpha = 0.01 | 8.664401071490563 |
| Alpha = 0.1 | 8.66440108367824 |
| Alpha = 1 | 8.664396296517642 |
| Alpha = 10 | 8.664436670639697 |
| Alpha = 100 | 8.664089500399202 |
| **Best: Alpha = 100** | **8.664089500399202** |

1. **Light GBM**

The reason for choosing:

By the models and results before, it seems linear model cannot produce a good fit. So, I choose to use decision tree. And light GBM is a gradient boosting tree-based algorithm,

which is similar to decision tree, easy to train but not very good at prediction. This model can give features’ importance based on tree algorithm, which can be used as feature selection for random forest.

Meanwhile light GBM uses histogram-based algorithms[1][2][3], speeding up training and reduces memory usage.

This algorithm is very suitable as a baseline to train non-linear algorithms.

Hyper parameters: As a baseline method, parameters chosen by heuristic

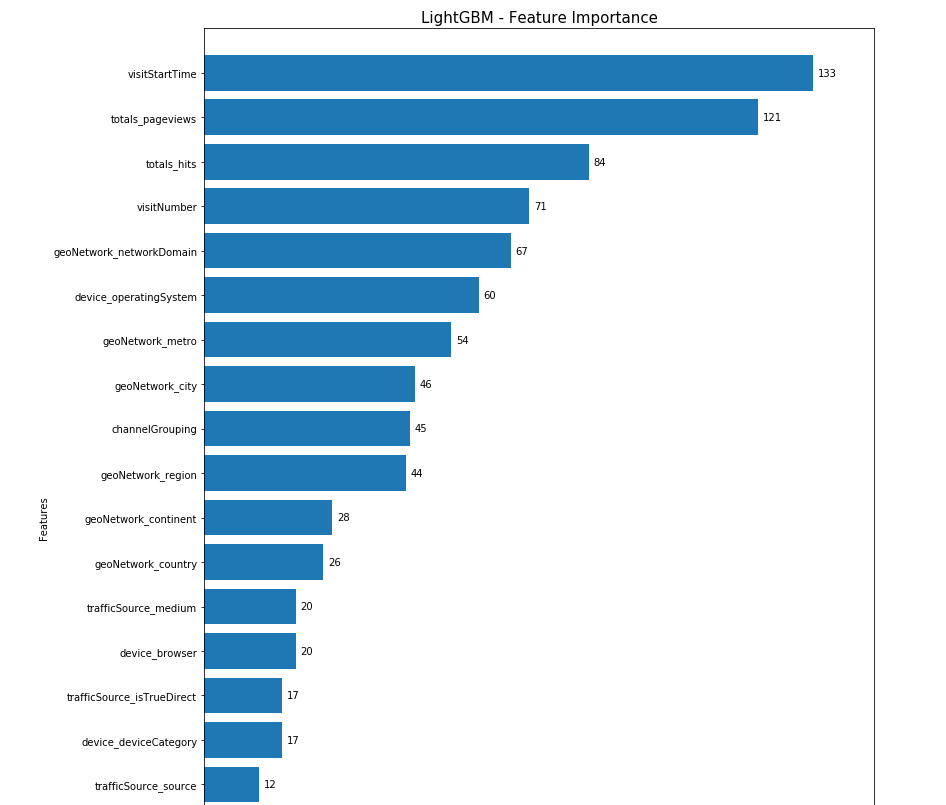
"objective" : "regression", "metric" : "rmse", "num\_leaves" : 60, "min\_child\_samples" : 200, "learning\_rate" : 0.1, "bagging\_fraction" : 0.7,"feature\_fraction" : 0.5, "bagging\_frequency" : 5,"bagging\_seed" : 3000, "verbosity" : -1

The complexity of your hypothesis set:only 1 model with parameters already set.

Overfitting or underfitting: Not very likely for such a huge amount of data and a simple model.

Result of RMSE: **12.797163214541467**

The importance of each features



Choose features with importance bigger than 30.

['channelGrouping', 'device\_operatingSystem', 'geoNetwork\_city', 'geoNetwork\_metro','geoNetwork\_networkDomain', 'geoNetwork\_region', 'totals\_hits','totals\_pageviews','visitStartTime', 'visitNumber']

1. **Random Forest**

The reason for choosing: I chose Bayes regression as a simple baseline model, it’s easy and fast to train.

Hyper parameters: choose n\_estimators = 20 (default is 10), for a larger forest and better result by heuristic, other parameters are chosen as default.

The complexity of your hypothesis set:only 1 model with 4 parameters already set.

Overfitting or underfitting: Not very likely for such a huge amount of data and a simple model.

Result of RMSE: **3.733930166412385**

## Model Selection and Comparison of Results

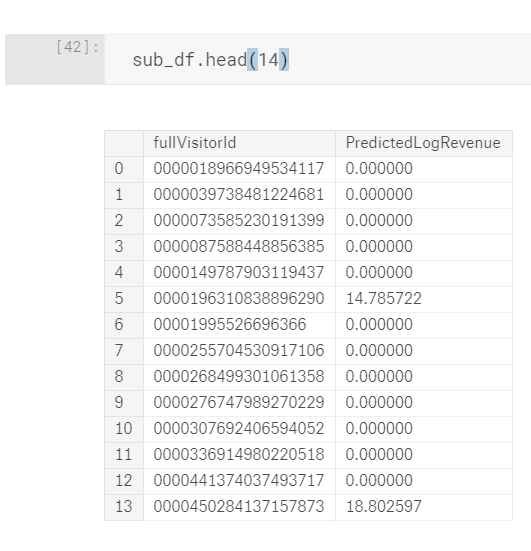
For all models’ validation RMSE

|  |  |
| --- | --- |
| Bayes regression | 8.648827446056496 |
| Linear Regression | 8.664401065570745 |
| Lasso Regression (best one) | 8.664327116930957 |
| Ridge Regression (best one) | 8.664089500399202 |
| lightGBM | 12.797163214541467 |
| **Random Forest** | **3.733930166412385** |

Choose Random Forest ( with 20 trees) as my final model.

# Final Results and Interpretation

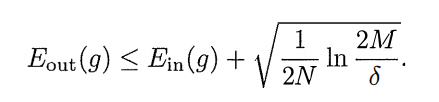
Build a submission file , 296530 rows(users) and 2 columns



Test the Random Forest ( with 20 trees) on test set, get

**RMSE = 4.436840917185571**

Estimate of out of sample performance:

Using function With 296530 data points, N = 296530 , M = 1, δ = 0.05,

1543994000(1), ε = 2.49\* (10^(-3)), which means the out of sample error is nearly the test error.

**If you are working on an online competition, report the performance of your best submission and compare it to others on the leader board.** If you want to compare your results with other work, do so here.

Unfortunately, this competition’s test data has been leaked.

And the Google and Kaggle just give all competitors the test data, so almost all the socres in leader board is 0.00, which is meaningless.

Before the data leak, I think my submission is likely to be 300/1000.

Interpretation: For this project, I found the my final model, random forest, is much better than other model is because the original data with too many factorized categorical features is non-linear. So no matter how I tried the linear model, the validation result is very similar. But for tree-based algorithm and random forest, they can deal with non-linear data very well. Although lightGBM didn’t give me a very good result, but it helped me to find important features. Finally, with these selected features and random forest, I got a much better result.

# Contributions of each team member

Not a team work.

# Summary and conclusions

Briefly summarize key findings, and optionally state what would be interesting or useful to do next.

# References

[1] Ranka, Sanjay, and V. Singh. “CLOUDS: A decision tree classifier for large datasets.” Proceedings of the 4th Knowledge Discovery and Data Mining Conference. 1998.

[2] Machado, F. P. “Communication and memory efficient parallel decision tree construction.” (2003).

[3] Li, Ping, Qiang Wu, and Christopher J. Burges. “Mcrank: Learning to rank using multiple classification and gradient boosting.” Advances in Neural Information Processing Systems 20 (NIPS 2007).

# Your code

**Please submit your code in a separate file.**  All your code should be in one pdf file, machine readable (no screen shots or scans).