MSAI 349 - Project Status Report

Google Analytics Customer Revenue Prediction

Mayank Malik and Amit Adate

1 The Task

Our task is to predict the total revenue generated per customer based on the customer dataset of a Google Analytics Merchandise Store. This project is our contribution to a live kaggle competition by Google. We are tackling a regression task.

2 The Data

We are provided with the training dataset of 1.7 million observations with 14 columns. However, a few columns are in JSON format and they require considerable preprocessing to convert into normal flattened CSV format. Each JSON column has many attributes in it . So, after expanding (flattening) these JSON into float columns, the total number of attributes are 60 in number.

Since, this is an online store, each observation actually is a session when a user comes online at the store to purchase something. However, a user may or may not cause any transaction, he can visit the store for browsing purposes. When a user doesn't purchase anything , the column totals.transactionRevenue is empty , else it generates some revenue. Also, each observation has other attributes / columns such as : device.browser , device.language, device.operatingSystem city, country, continent, date, visitStartTime, latitude , longitude etc.

The most important column for our analysis is **totals.transactionRevenue** . As this is our dependant variable that we need to predict for each observation(session) in the test dataset which has 0.4 million observations and it exactly has the same columns as the training dataset (except the dependant variable - totals.transactionRevenue)

Partitioning of data, 1.7 million examples in the training set and 0.4 million examples in the testing set, provided by the kaggle competition creators. Additionally, we have divided the training set into training and validation sets to perform 5 fold cross validation. Hence, we have generated 3 datasets: training, validation and testing.

3 Progress

We computed feature importance over the 60 features, the algorithm we used to compute importance was random forests. The results demonstrate that the most important features in the descending order of importance are: **Total.pageviews Total.hits Visitstarttime Visitnumber City Region**

The techniques we tried towards the task of regression are **Linear Regression, Polynomial Regression, Vanilla Decision Tree, Bagging - Random Forests.**

Results of implementing these techniques - **Linear Regression: 69.23, Polynomial Regression: 20.23, Vanilla Decision Tree: 25.78, Bagging - Random Forests: 9.67**

The metric used for evaluation is **RMSE of log of sum of transaction revenues**

The machine learning software packages we used for our experiments are, **Scikitlearn and Weka**

4 Future Plans

4.1 Concern

In our dataset, The dependant variable is highly imbalanced. The transactionrevenue is present only in 1.3 percent of all the sessions. For the rest, it is 0. So, the models don't have significant transaction revenue data to train. There have been good techniques on handling imbalanced data in the classification context such as - undersampling, oversampling, SMOTE etc. However, These techniques might not be used in the regression context. So, we have contemplated about it and now we want to try a new approach - The idea is to classify non-zero transactions first and use that to help our regressor get better results. We believe this will help us to improve our results further. Moreover, we have already scheduled some time with the professor to discuss on 21st Nov.

4.2 More Techniques

We are also planning to implement a few techniques (2 atleast if not more) from scratch. We think Linear Regression and KNN shouldn't be difficult to implement from scratch . We can write definitions for them, implementing them manually in python3.

Also, we intend to implement deep learning methods for comparison with above mentioned techniques. We will experiment on **Keras** - a wrapper over **Tensorflow**, using their **Sequential Model**. We intend to provide with a comparison between all of the various techniques that we have implemented.

4.3 Dimensionality Reduction

When we perform one-hot encoding, we believe our models might be getting affected from the curse of dimensionality. Our dimensions get larger when we one hot encode them, hence we would like to experiment with dimensionality reduction techniques such as, **t-SNE , PCA and LDA**.