Google Analytics

Team 16

Agenda

- Explanation of the competition
  - Objective and key features
- Overview of the data
  - Business context and dataset description
- Critiques of other projects
- Solution and Challenges

Customer Revenue Prediction

- Analyze Google Store customer dataset
  - Predict total revenue per customer for future date
- How can data analytics support marketing teams?
  - Realign investment priorities with core customer features
  - Actionable insights can be derived to optimize promotional strategies
  - R functionality and open source software allow companies to leverage Google Analytics data
- Limitations of Google Analytics
  - Data is sampled after exceeding a session threshold
  - No access to raw data and limited data privacy
  - Not compliant with EU law

Key Features

- 80 / 20 Pareto Principle
  - Find key value adding features, retain customers, attract customers with these feature profiles
- Predict natural log of total sum transactions per user
  - forward looking problem
- Competition Evaluation

$$y_{user} = \sum_{i=1}^{n} transaction_{user_i}$$

$$target_{user} = \ln(y_{user} + 1)$$

Submissions are scored on the root mean squared error. RMSE is defined as:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2},$$

where y hat is the natural log of the predicted revenue for a customer and y is the natural log of the actual summed revenue value plus one.

Dataset Description

- Customer Data
  - Train set - user transactions from August 1st 2016 to April 30th 2018 - 25.41 GB
  - Test set - user transactions from May 1st 2018 to October 15th 2018 - 7.62 GB
  - Each row represents one virtual visit to the store
  - Dimensions:63264 rows, 54 columns
- Data Fields
  - Numeric features: fullVisitorId, visitId, visitNumber, visitStartTime, date
  - Categorical features: channelGrouping, customDimensions, device, trafficSource
  - Aggregate features: totals - per session and high-level

Data

- A 63264 * 54 dataset including categorical features such as browser, operating system, device, and numeric features such as visit number, date
- Null values:

	Column	Null_Count	Null_Percent
0	trafficSource.adContent	62466	98.738619
1	trafficSource.adwordsClickInfo.slot	61680	97.496206
2	trafficSource.adwordsClickInfo.page	61680	97.496206
3	trafficSource.adwordsClickInfo.gclId	61676	97.489884
4	trafficSource.referralPath	40167	63.491085
5	trafficSource.keyword	35175	55.600341

Goal

Dependent Variable: The natural log of the sum of all transactions per user

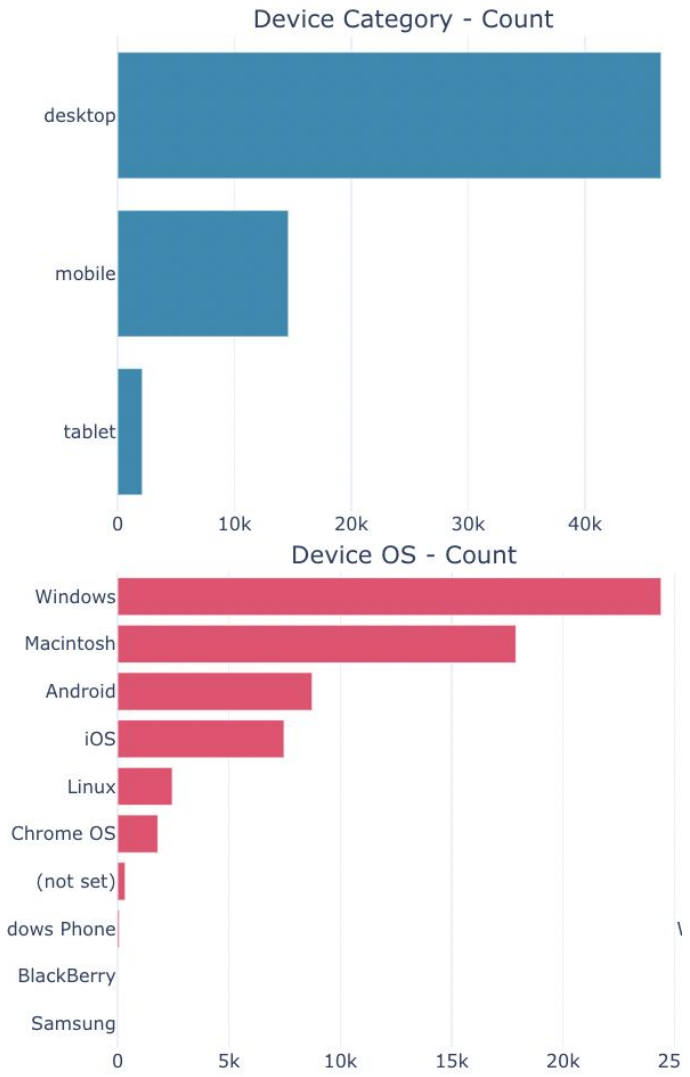
Evaluation metric: RMSE

Critiques

Tedious Visualization

Current: Spent too much time investigating the relationship between each category and their counts, which is not that related to our prediction purpose (the natural log of the sum of all transactions per user)

Recommendation: Delete useless visualizations, and explore more about relationships between different variables and the transactions



Lack of Tuning Method

```
params = {
        "objective" : "regression",
        "metric" : "rmse",
        "num_leaves" : 30,
        "min_child_samples" : 100,
        "learning_rate" : 0.01,
        "bagging_fraction" : 0.8,
        "feature_fraction" : 0.5,
        "bagging_frequency" : 5,
        "bagging_seed" : 2018,
        "verbosity" : -1
    }
```

Current: Set specific values for the hyperparameters

Recommendation: Use nested for loops to iterate through lists of possible hyperparameters, to find out the best combination

Lack of Feature Selection

```
cat_columns = list(df_train.dtypes[df_train.dtypes == 'object'].reset_index()['ind
ex'])
num_columns = num_cols = ["totals.hits", "totals.pageviews", "visitNumber", "visit
StartTime", 'totals.bounces',  'totals.newVisits']

for col in cat_columns:
    df_train[col] = LabelEncoder().fit_transform(df_train[col])

X = df_train[cat_columns + num_columns]
y = df_train['totals.transactionRevenue']
```

Current: Manually selected features according to visualization

Recommendation: Run a feature selection to get importance score of each feature, and select features based on that to avoid overfitting

Time Series Gap

Data Situation: The dataset does NOT contain data for December 1st 2018 to January 31st 2019

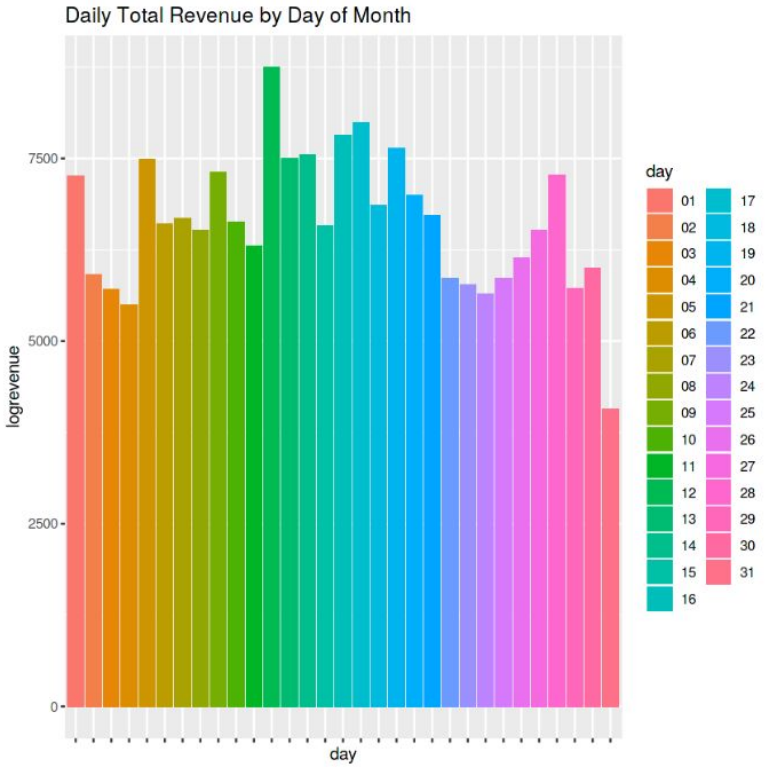
Current: Didn’t find ways to fill out the gap

Recommendation: Identify the unique fullVisitorIds in the provided test_v2.csv and make predictions for them for those unseen months

Unnecessary Visualizations

- Many visualizations are messy and do not provide much value
- They are difficult to read

Recommendation: Show the summary statistics on most of the variables and then show its relationship to Revenue without adding useless graphs.



Lack of Features and Modeling

```
print ('Regression of Hour of visit to LogRevenue')
summary(lm(logrevenue ~ as.factor(hour_visit) - 1,data=dfgoog_subset_2))
```

Ran linear regression model using only one feature at a time

Never provided a final model

Never gave RMSE which was a requirement of the competition

Only uses linear regression. While linear regression is simple, it most likely will not provide the best result

Splitting and Usage of Data

```
dfgoog = read.csv('../input/flat-train-data/flat_train.csv',colClasses = c(fullVisitorId="ch
aracter"))
dfgoog_subset = subset(dfgoog,select=-c(device_browserVersion,totals_visits,device_browserSi
ze,trafficSource_adwordsClickInfo.criteriaParameters,trafficSource_adwordsClickInfo.page,tra
fficSource_adwordsClickInfo.slot,trafficSource_adwordsClickInfo.gclId,trafficSource_adwordsC
lickInfo.adNetworkType,trafficSource_adwordsClickInfo.isVideoAd,trafficSource_adContent,traf
ficSource_campaignCode,device_operatingSystemVersion,device_mobileDeviceBranding,device_mobi
leDeviceModel,device_mobileInputSelector,device_mobileDeviceInfo,device_mobileDeviceMarketin
gName,device_flashVersion,device_language,device_screenColors,device_screenResolution,geoNet
work_cityId,geoNetwork_latitude,geoNetwork_longitude,geoNetwork_networkLocation))
dfgoog_subset = subset(dfgoog_subset,select=-c(socialEngagementType))
```

Dropped many columns because they did not want to deal with missing values.

Ran model on training data but never used the testing data to validate the model

Solution

STEPS:

- JSON formatting
- Aggregating
- Boosting
- Prediction

CHALLENGES:

- Size of the data
- mode() not working
- Variable types
- Deciding on gbm() parameters

JSON Formatting

```
data$customDimensions<-gsub("[]","[{}]",data$customDimensions,fixed=TRUE)
data$customDimensions<-gsub("'","\"",data$customDimensions,fixed=TRUE)
data$customDimensions<-gsub("[","",data$customDimensions,fixed=TRUE)
data$customDimensions<-gsub("]","",data$customDimensions,fixed=TRUE)
data$customDimensions <- factor(data$customDimensions)
data.customDimensions <- fromJSON(paste('[',paste(data$customDimensions,collapse = ','),']'),flat
data.customDimensions <- data.frame(apply(data.customDimensions, 2,function(x){as.factor(x)}))
```

```
                                                        trafficSource
1: {""referralPath"": ""(not set)"", ""campaign"": ""(not set)"", ""source"": ""google"", ""medium"":
 ""organic"", ""keyword"": ""(not provided)"", ""adContent"": ""(not set)"", ""adwordsClickInfo"": {""c
riteriaParameters"": ""not available in demo dataset""}, ""isTrueDirect"": true}
2:     {""referralPath"": ""(not set)"", ""campaign"": ""(not set)"", ""source"": ""(direct)"", ""mediu
m"": ""(none)"", ""keyword"": ""(not set)"", ""adContent"": ""(not set)"", ""adwordsClickInfo"": {""cri
teriaParameters"": ""not available in demo dataset""}, ""isTrueDirect"": true}
3: {""referralPath"": ""(not set)"", ""campaign"": ""(not set)"", ""source"": ""google"", ""medium"":
 ""organic"", ""keyword"": ""(not provided)"", ""adContent"": ""(not set)"", ""adwordsClickInfo"": {""c
riteriaParameters"": ""not available in demo dataset""}, ""isTrueDirect"": true}
      visitId visitNumber visitStartTime
1: 1526099341           2     1526099341
```

Aggregating

```r
calculate_mode <- function(x) {
  uniqx <- unique(x)
  uniqx[which.max(tabulate(match(x, uniqx)))]
}
```

```r
temp<-data %>%
  group_by(fullVisitorId) %>%
  summarise(browser = calculate_mode(browser))
tr <- merge(tr,temp,by="fullVisitorId",sort = FALSE,all.x=TRUE)
```

```r
temp<-data %>%
  group_by(fullVisitorId) %>%
  summarise(pageviews = mean(pageviews))
tr <- merge(tr,temp,by="fullVisitorId",sort = FALSE,all.x=TRUE)
```

```
       tablet  Americas Northern America United States                    California
                        metro                          city  campaign   source  medium
                   (not set)                   (not set) (not set)   google organic
San Francisco-Oakland-San Jose CA          San Francisco (not set) (direct)  (none)
   not available in demo dataset not available in demo dataset (not set)   google organic
                  Houston TX                   Houston (not set) (direct)  (none)
              Los Angeles CA                    Irvine (not set)   google organic
isTrueDirect adContent adwordsClickInfo.slot adwordsClickInfo.adNetworkType visitNumber.x
       TRUE (not set)    Google search: Top                Google Search             1
       TRUE (not set)    Google search: Top                Google Search           176
       TRUE (not set)    Google search: Top                Google Search             2
       TRUE (not set)    Google search: Top                Google Search             4
      FALSE (not set)    Google search: Top                Google Search             1
visitNumber.y pageviews sessionQualityDim timeOnSite transactions transactionRevenue
            1  2.500000                1  492.50000            0                   0
          176  2.333333                1   22.33333            0                   0
            2  3.000000                1   24.00000            0                   0
            4  4.000000                1   25.00000            0                   0
            1  4.000000                1   49.00000            0                   0
totalTransactionRevenue dayOfMonth month
                      0         11     5
                      0         11     6
                      0         11     5
                      0         11     5
                      0         11     5
```

Boosting

```r
set.seed(272)
train_test_split<-createDataPartition(data$transactionRevenue, p=0.7, list = FALSE)
train_data<-data[train_test_split,]
test_data<-data[-train_test_split,]


library(gbm)
set.seed(272)
# train GBM model
gbm.fit <- gbm(
  formula = transactionRevenue ~ .,
  distribution = "gaussian",
  data = train_data[,-c(1)],
  n.trees = 3500,
  interaction.depth = 5,
  shrinkage = 0.001,
  cv.folds = 3,
  n.cores = NULL,
  verbose = FALSE
)
print(gbm.fit)
```

Prediction

```
# predict values for test split
pred <- predict(gbm.fit, n.trees = best.iter, test_data)
# results
caret::RMSE(pred, test_data$transactionRevenue)


test_data$target <- ifelse(pred<0,0,pred)
tsf_data <- data.frame(fullVisitorId=unique(test_data$fullVisitorId))


temp<-test_data %>%
  group_by(fullVisitorId) %>%
  summarise(PredictedLogRevenue = log(sum(target)+1))
tsf_data <- merge(tsf_data,temp,by="fullVisitorId",sort = TRUE,all.x=TRUE)
head(tsf_data)
```

	fullVisitorId	PredictedLogRevenue
1	0004964234894022692	10.64558
2	0007577590016777945	10.63257
3	0007761730641712778	10.63257
4	0007975147778360613	10.63257
5	0008412824358840463	10.90492
6	0008558739771259 74	10.63257

For Future…

- Use the whole dataset
- Better data investigation
- More/less variables
- Different parameters in gbm()
- Data Visualizations