A Marketing Strategy For The Google Merchandise Store For The Holiday Season

Authors: Docena, Amel. Gadkari, Shruti. Kumar, Anurag. Mohanty, Manas
Contributor: Banik, Anusheela

I. Summary

In most businesses, it is believed and been proven that a small portion of customers generate the greater portion of revenues. Thus, one challenge of business strategy is to come up with a marketing strategy that is both effective and efficient. Under this context, we wish to take on developing a marketing strategy for the Google Merchandise Store for the holiday season because the store faces the same challenge. To aid us in devising this strategy, we applied the tools learned in class for our data analysis. The merchandise store data contain geographic information of each visitor, their record of activity for each session, (e.g., the number of page visits and hits), and the revenue per transaction, if any. We discovered some characteristics of online visitors of the store who were most likely to be customers by exploring the data and some insights gained from a logistic regression. Based on these findings, we drew our marketing strategy.

II. Methods

Our workflow started with fetching the data set, tidying and transforming it, and then exploring the variables. We then moved on to specifying a model based on our exploration and intuition, and then did some measures for evaluating the model performance.

Fetch data

We downloaded the raw Google Merchandise Store data set from Kaggle. The entire data set, however, is 25GB in size. Since we intended to read just a portion of the data set, we uploaded the data in R by reading the first 100,000 rows, iterating this step for the next 100,000 rows, while keeping only those observations whose dates are from 20 Nov 2016 through 5 Jan 2017.

Tidy data

A challenge we faced in tidying the raw data set lied in dealing with columns that are in JSON format. Pieces of information which are variables of their own are lumped in one column. Thus, our remedy was to split and/or separate these entries then create their respective variables. For those sub-columns whose entries are missing, we dropped them off.

Transform data

From the variables of the tidy data, we then moved on to transforming some of them toward a meaningful analysis.

Week: We created a categorical variable week, wherein the first week ranges from November 26 to Dec 2, the second week from Dec 3 to 9, and so on, until the last week of our scope. There are six week-values for this variable.

Transaction: This is a binary response variable whether a session has any transaction revenue or not.

Hour: This is a categorical variable for the hour of the day of each visit.

Source: There are some values that are spelled differently in other observations, but actually the same. For example, *bing* in one observation is *bing.com* in another. So, we further cleaned these values to make them uniform. Furthermore, we categorized sources that have total counts fewer than 100 as other sources.

Operating system: We categorized some operating systems with total counts fewer than 100 as others.

Browser: Similar to our treatment with operating system, browsers with total counts fewer than 100 we categorized as others.

Channel-grouping: There is a lone record with value as *(other)*. Since we could not identify its category, we treated it as NA.

Explore data

We explored among others the relationship between some demographics and level of activity, and then with revenue. We discovered some six facts about the store, and that only a small portion of visitors purchased merchandise.

Model data

This central fact led us to the business problem of understanding sessions that ended up with transactions. Who were those online visitors most likely to be customers? Stated alternatively, what is the probability that a visitor is actually a customer? Since this is a binary classification problem, a plausible model for our study is logistic regression. Our choice of predictors was based on the previous step of data exploration, exclusion of redundant and confounding variables, and intuition. Here is our model specification for the logistic regression:

$$P(y \quad 1|\, x) \quad G({}_0 + {}_1\textit{week} \quad {}_2\text{visits} + {}_3\text{visitNumber} + {}_4\text{ visithour} \quad {}_5\text{source} +$$
$$ {}_6\text{ pageviews} + {}_7\text{operatingSystem} + {}_8\text{medium} + {}_9\text{ hits} + {}_{10}\text{deviceCategory} +$$
$$ {}_{11}\text{continent} + {}_{12}\text{channelGrouping} + {}_{13}\text{browser}), \text{ where } 0 \; G(x) \; 1.$$

Down-sampling and Cross-validation

Since the ratio between the number of observations without transaction to those observations with transaction is extremely large, (in other words, the data is imbalanced), we implemented down-sampling and adjusted the ratio to 80:20.

We chose to perform cross-validation as our method for evaluating model performance. We performed a 5-fold cross-validation and ensured that for each iterated test and training subsets, the ratio of records from both the classes is balanced.

III. Results

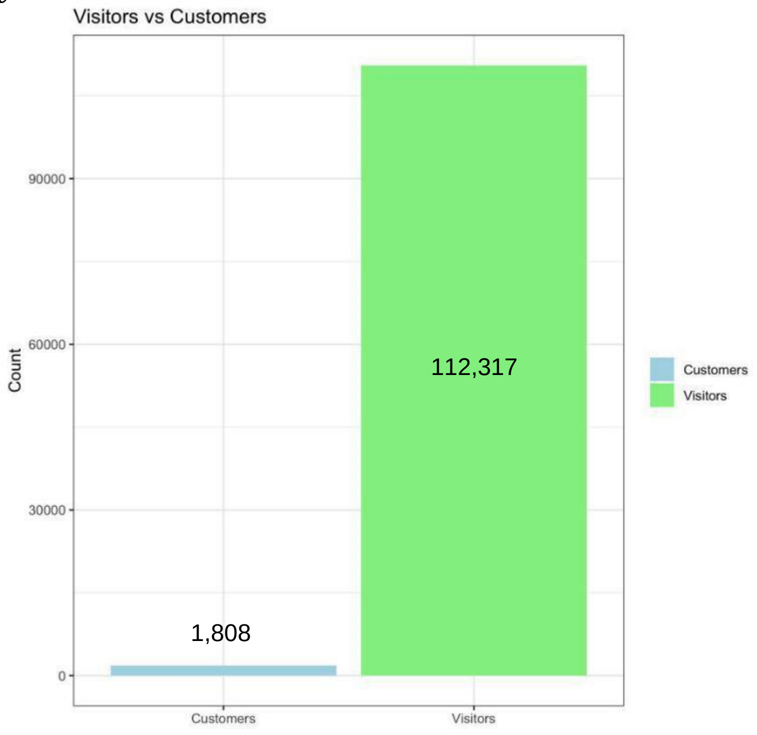
We discovered six interesting facts about the Google merchandise store. **Fact 1: Not all visits translate to revenue**

For the 2016 holiday season, the store was visited 131,909 times. But only 1,873 of these resulted in transactions, scantly 1.4%. The total revenue earned was $33,199. This seems to suggest that a targeted marketing would be a good strategy for the holiday season: Who are those visitors Google is most likely to earn revenue?

Fact 2: Not all visitors are customers

From those page visits, there were 114,125 unique visitors, (so there were visitors who checked out the store more than once). The chance of a visitor to be a customer was barely 1.6%. For every 100 visitors we would expect only one of them would be a customer. The revenue we would expect from any visitor is only $0.30. But from a customer-visitor, we would expect a greater $18.36. This further attests the case of a targeted marketing strategy.

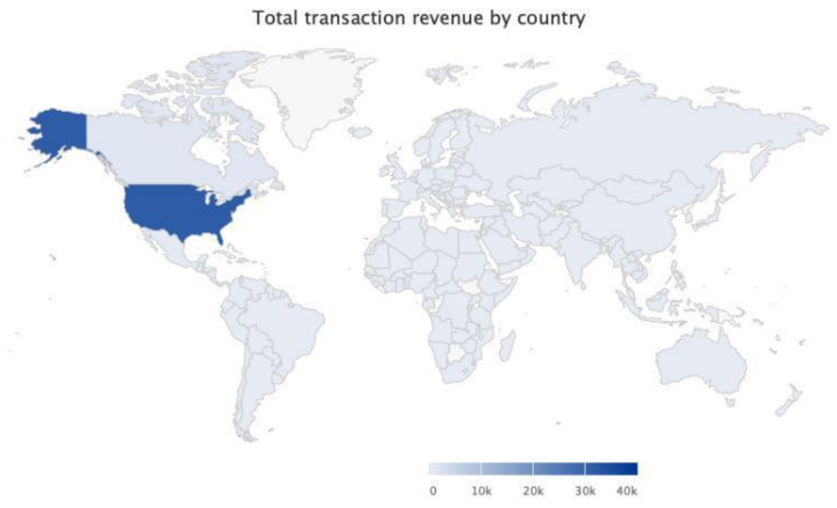
Number of visitors versus customer-visitors of the Google Merchandise Store during the 2016 holiday season



Fact 3: Certain countries we expect higher revenues

Google brand loyalists were from the United States一revenue comprised about 95% of the total revenue. But this is expected since Google is a US brand.

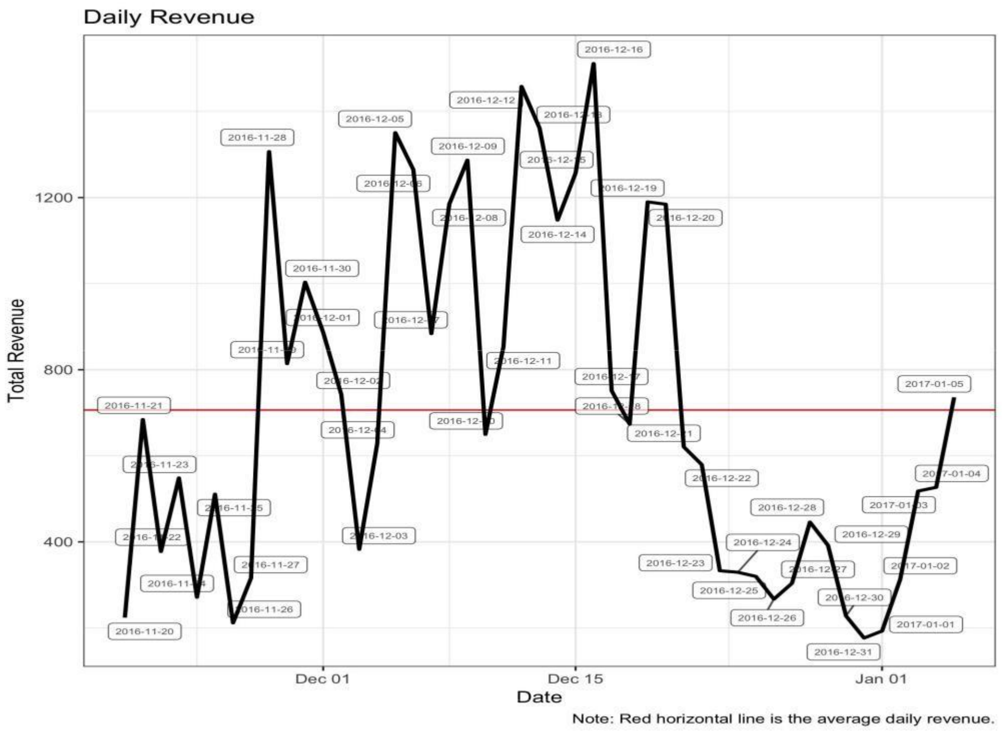
Revenue earned per country during the 2016 holiday season



Fact 4: Certain dates we expect higher revenues

The average daily revenue for the entire holiday season was $706. There were certain dates when the daily revenue was above this average. These were Nov 28, which was a Cyber Monday, and would extend through the pre-Christmas period, from Dec 5 to 20. This period captures consumer behavior, such as buying because of discounts and perks, and tradition of spending and gift-giving.

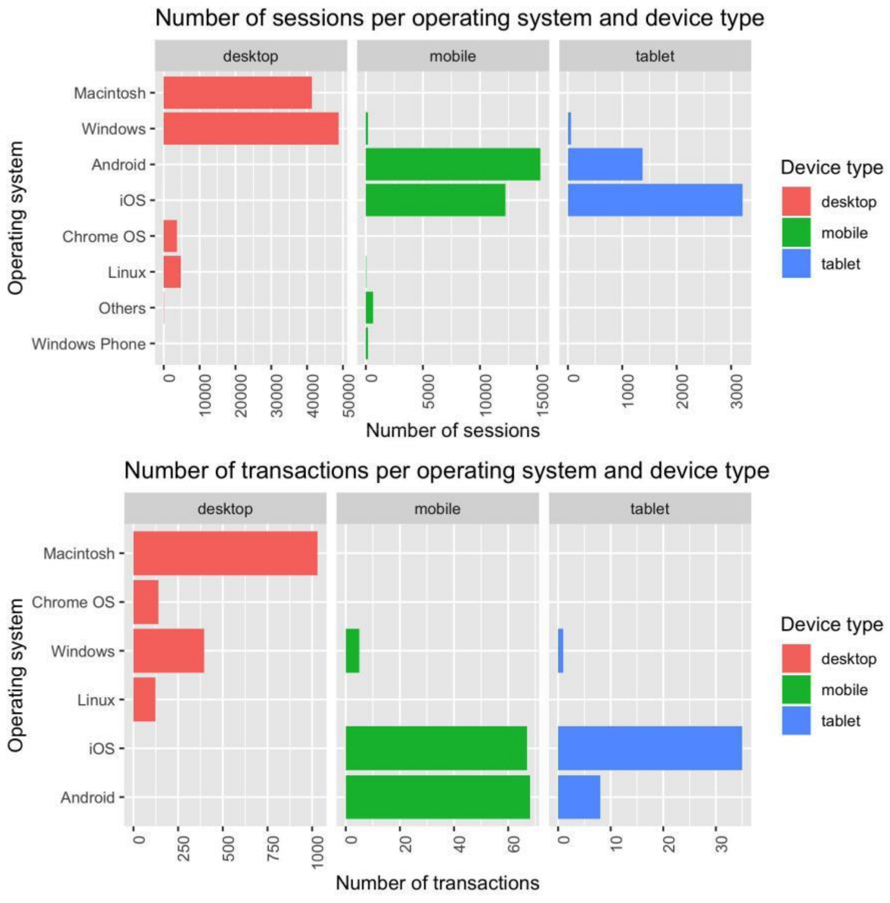
Daily revenue earned from 20 Nov 2016 through 5 Jan 2017



Fact 5: Certain customers generate higher revenues

Across the device types, we may say that Apple users were most likely to purchase Google merchandise. Does this suggest that users who *Think Different* associate themselves with the Google brand?

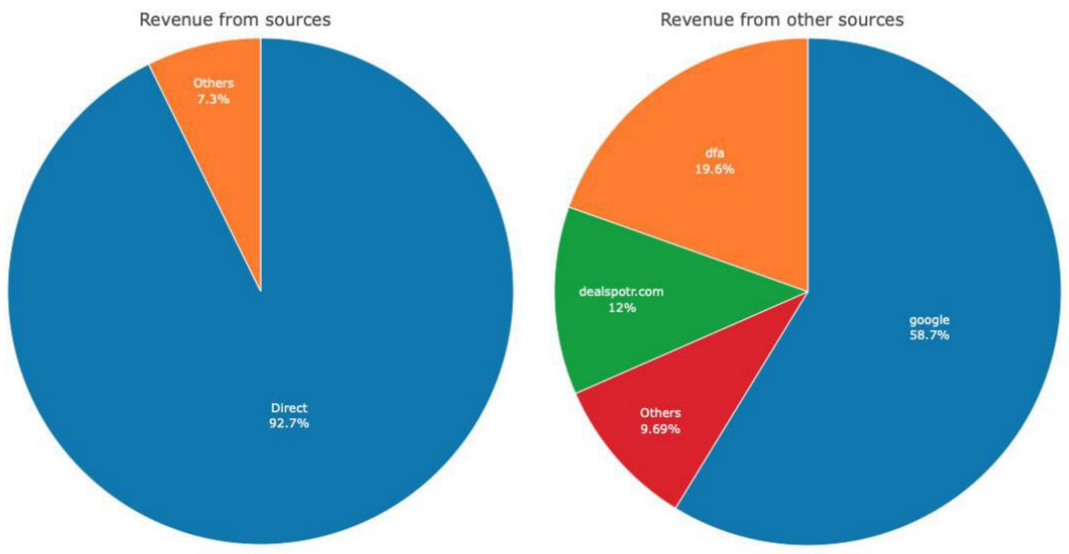
Number of sessions and transactions per operating system and device type



Fact 6: Third-party sources generate relatively lower revenue than direct-visits

Customers who would purchase Google merchandise are most likely to visit the store directly. Only 7.3% had been referred from third-party sources. From this third-party referrals, most would check out the store from ads shown in Google search; next to that from double-click for advertisers; from a shopping social network, *dealspotr.com*; and the smallest portion from other third-party sources. Can Google improve in their marketing of the store through third-party sources?

Composition of revenue earned from sources



Business Challenge

Since the expected revenue from customer-visitors is greatly larger than the expected revenue from an (ordinary) visitor, the challenge for Google is to target these customer-visitors for an effective and efficient allocation of marketing resources. What were some characteristics of online visitors most likely to be customers during the 2016 holiday season?

Results from the Model

We discovered from Fact 4 that daily revenue started to rise during the week of Thanksgiving and peaked on Cyber Monday. Our logistic model confirms our observation that post this Thanksgiving week, customers were still likely to purchase merchandise through the pre-Christmas period.

Another insight from the model about the behavior of customers is that purchases were less likely to occur in the morning as compared to midnight visits.

Compared to direct visits, customers were less likely to be referred from third-party sources. But this is not the case for referrals from *dealspotr.com*, which is a shopping social network that offers deals and discounts. Customers were more likely to purchase Google merchandise through this site. Does the marketing strategy of dealspotr.com actually work?

Results of the Logistic Regression

Cross-valided model accuracy

Overall accuracy	**True-positive accuracy**	**True-negative accuracy**
94%	90%	94%

Some statistically significant key predictors (excerpt from entire model summary)

Categorical predictor	**Coefficient**	**p-value**
Week [base = Nov 26-Dec 2]		
Dec 3-9	0.314	0.001153 **
Dec 10-16	0.53	7.89e-09 ***

Hour of the day [base = 12am-12:59am]		
4:00am-4:59am	-0.349	0.087383 .
5:00am-5:59am	-0.747	0.000999 ***
6:00am-6:59am	-0.772	0.003647 **
7:00am-7:59am	-1.2	0.000551 ***
9:00am-9:59am	-1.66	0.030311 *
11:00am-11:59am	-1.81	0.051700 .
11:00pm-11:59pm	-0.612	0.001668 **
Source [base = direct visit]		
dealspotr.com	1.16	0.001755 **
double-click for advertisers	-2.01	0.007907 **
facebook	-2.83	0.023222 *
google (search)	-1.4	3.34e-06 ***

Note on significant levels: *** at 0.1%; ** at 1%; * at 5%; . at 10%

IV. Discussions

In a nutshell, our findings confirm that only a small group of visitors are most likely to purchase Google merchandise. And this situation is a challenge. Thus alongside our analyses, we have come up with a marketing strategy tailored for this store.

A marketing strategy for the Google Merchandise Store

At the core, if Google wishes to expand its base of loyalists both nationally and internationally, it may consider strengthening its brand position. We know that the primary reason a person would purchase a merchandise and own it is because that person associates with the brand's identity. As of the moment, Google's slogan is "Don't be evil." But do people associate with this message?

For efficient allocation of marketing resources, Google should take on targeted marketing this holiday season. That is, Google should channel marketing resources to users who are most likely to be customers. Strategies like blast marketing, wherein every user receives a marketing ad, should be avoided for this would be inefficient.

Discounts and deals should be given out during the Thanksgiving throughout pre-Christmas for this period captures consumer behavior. We may attribute this behavior to the tradition of spending and gift-giving.

As an extension, Google may consider strengthening its ties with third-party sources. Some potential customers may be reached through these sites. Furthermore, Google may want to study the business model of shopping social networks. Some business insights may be gained from here.

VI. References

1. Creating Publication-ready Word Tables in R. Weston, S. and Yee, D.
Retrieved online: https://dmyee.files.wordpress.com/2016/03/table_workshop.pdf

2. Google Analytics. User Explorer. Website: https://analytics.google.com/analytics.

3. Google Analytics Customer Revenue Prediction. Overview, Discussions, and Data. Retrieved from Kaggle: https://www.kaggle.com/c/ga-customer-revenue-prediction.

4. Target Marketing. Retrieved from The Balance Small Business website: https://www.thebalancesmb.com/target-marketing-2948355

VII. Appendix
A. Entire Logistic Regression Model Summary

```
glm(formula = Transaction ~ Week + visits + visitNumber + visithour + source
    + pageviews + operatingSystem + medium + hits + deviceCategory +
    continent + channelGrouping + browser, family = binomial(link =
    "logit"), data = train_data)

Deviance Residuals:
    Min       1Q    Median      3Q      Max
-5.6033  -0.1126  -0.0245  0.0000   4.4443

Coefficients: (6 not defined because of singularities)
                           Estimate Std. Error z value Pr(>|z|)
(Intercept)               -1.840e+01  3.481e+02  -0.053 0.957853
WeekWeek   2               3.138e-01  9.655e-02   3.250 0.001153  **
WeekWeek   3               5.295e-01  9.175e-02   5.771 7.89e-09 ***
WeekWeek   4               1.549e-01  1.042e-01   1.488 0.136820
WeekWeek   5               3.390e-02  1.465e-01   0.231 0.817033
WeekWeek   6              -2.180e-01  1.429e-01  -1.525 0.127138
visits                           NA         NA      NA       NA
visitNumber               -1.134e-02  4.640e-03 -2.445 0.014495 *
visithour1                -1.266e-01  1.874e-01 -0.676 0.499357
visithour2                -1.740e-01  1.924e-01 -0.904 0.365843
visithour3                -8.352e-02  1.955e-01 -0.427 0.669244
visithour4                -3.487e-01  2.040e-01 -1.709 0.087383 .
visithour5                -7.468e-01  2.269e-01 -3.291 0.000999 ***
visithour6                -7.721e-01  2.656e-01 -2.907 0.003647 **
visithour7                -1.196e+00  3.461e-01 -3.455 0.000551 ***
visithour8                -4.769e-01  3.304e-01 -1.443 0.148918
visithour9                -1.660e+00  7.664e-01 -2.166 0.030311 *
visithour10               -1.272e-01  4.370e-01 -0.291 0.770955
visithour11               -1.813e+00  9.317e-01  -1.946 0.051700 .
visithour12                7.372e-04  3.413e-01   0.002 0.998277
visithour13               -2.177e-02  2.575e-01 -0.085 0.932629
visithour14               -3.377e-02  2.240e-01  -0.151 0.880183
visithour15                2.151e-02  1.905e-01   0.113 0.910105
visithour16               -1.554e-01  1.792e-01 -0.867 0.385885
visithour17               -1.042e-01  1.736e-01 -0.601 0.548152
visithour18               -1.737e-01  1.685e-01  -1.030 0.302843
visithour19                2.048e-02  1.655e-01   0.124 0.901523
visithour20                1.001e-01  1.661e-01   0.603 0.546813
visithour21               -3.208e-01  1.745e-01 -1.838 0.066014 .
visithour22               -2.400e-01  1.753e-01 -1.369 0.170951
visithour23               -6.120e-01  1.947e-01 -3.144 0.001668 **
sourcebaidu               -1.531e+01  7.006e+02 -0.022 0.982560
sourcebing                -2.387e-01  7.037e-01 -0.339 0.734468
sourceblog.golang.org       -1.607e+01  1.713e+03  -0.009 0.992515
sourcedealspotr.com          1.163e+00  3.718e-01   3.129 0.001755 **
sourcedfa                 -2.008e+00  7.559e-01 -2.656 0.007907 **
sourcefacebook            -2.830e+00  1.247e+00 -2.270 0.023222 *
sourcegoogle              -1.399e+00  3.010e-01 -4.649 3.34e-06 ***
sourceOthers              -2.829e-01  5.281e-01  -0.536 0.592152
sourcePartners            -1.669e+01  4.415e+02 -0.038 0.969844
sourceqiita.com           -1.380e+01  1.183e+03 -0.012 0.990694
sourcequora               -1.405e+00  1.246e+00 -1.127 0.259624
sourcereddit              -1.738e+01  1.279e+03 -0.014 0.989156
sourcesiliconvalley.about.com -1.677e+01  9.821e+02  -0.017 0.986379
sourceyahoo               -  3.740e-03  5.652e-01  -0.007 0.994721
sourceyoutube             -6.537e+01  3.902e+05   0.000 0.999866
pageviews                 4.245e-01  1.415e-02  30.008 < 2e-16 ***
operatingSystemChrome OS      2.389e+00  5.678e-01   4.207 2.58e-05 ***
operatingSystemiOS           6.651e-02  2.669e-01  0.249 0.803214
operatingSystemLinux         2.053e+00  5.661e-01   3.626 0.000288 ***
operatingSystemMacintosh      2.414e+00  5.580e-01   4.326 1.52e-05 ***
operatingSystemOthers        -8.209e+00  4.683e+02  -0.018 0.986014
operatingSystemWindows        2.274e+00  5.590e-01   4.069 4.72e-05 ***
operatingSystemWindows Phone  -1.288e+01  1.144e+03  -0.011 0.991015
```

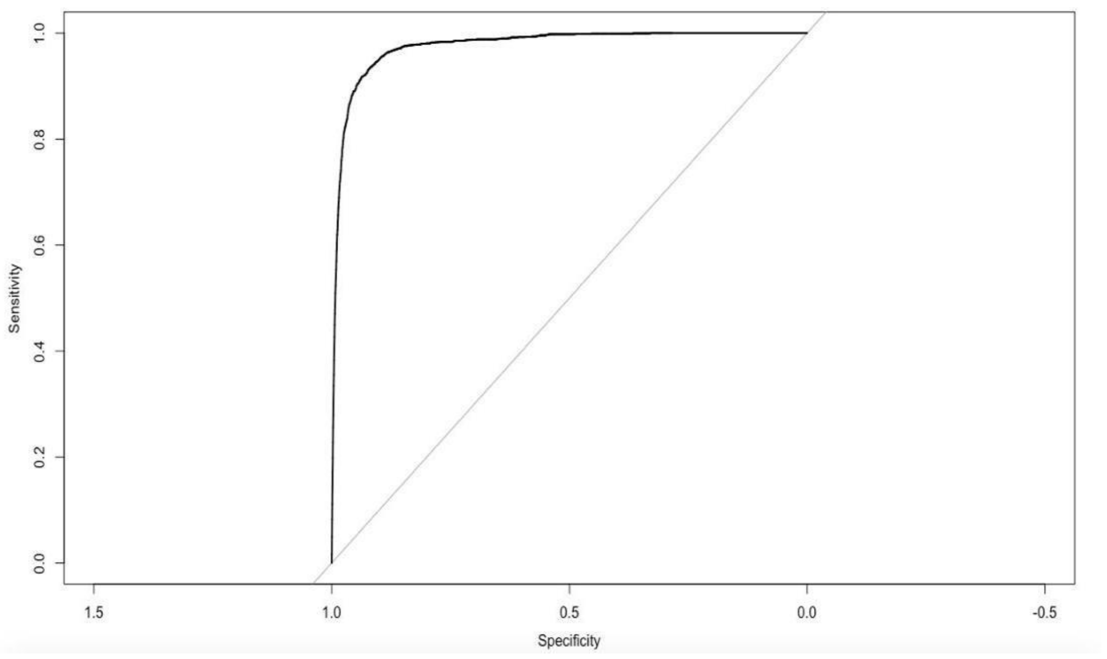
```
mediumaffiliate                      NA         NA     NA       NA
mediumcpc                     4.576e-01  7.625e-01  0.600 0.548387
mediumcpm                          NA         NA     NA       NA
mediumorganic                 1.250e+00  3.783e-01  3.305 0.000948 ***
mediumreferral                       NA         NA     NA       NA
hits                        -2.326e-01  1.024e-02 -22.727  < 2e-16 ***
deviceCategorymobile          1.105e+00   5.326e-01  2.076 0.037940 *
deviceCategorytablet          1.321e+00   5.704e-01  2.316 0.020568 *
continentAmericas             1.415e+01  3.481e+02  0.041 0.967587
continentAsia                 1.138e+01  3.481e+02  0.033 0.973920
continentEurope              1.074e+01   3.481e+02   0.031 0.975393
continentOceania              1.045e+01  3.481e+02   0.030 0.976058
channelGroupingDirect        -1.917e+00  7.177e-01  -2.671 0.007556 **
channelGroupingDisplay               NA         NA     NA       NA
channelGroupingOrganic Search -2.197e+00  7.173e-01  -3.063 0.002195 **
channelGroupingPaid Search   -1.959e+00   7.274e-01 -2.693 0.007078 **
channelGroupingReferral      -1.225e+00  7.100e-01 -1.726 0.084417 .
channelGroupingSocial                NA         NA     NA       NA
browserChrome               -1.244e+00  7.500e-01 -1.658 0.097308 .
browserEdge                 -1.479e+00  8.511e-01  -1.737 0.082335 .
browserFirefox              -9.317e-01  7.801e-01  -1.194 0.232361
browserInternet Explorer     -9.610e-01  8.029e-01 -1.197 0.231373
browserOpera                -7.956e-01  1.261e+00 -0.631 0.528185
browserOpera Mini           -1.057e+01   4.366e+02 -0.024 0.980680
browserOthers               -1.367e+01  6.985e+02 -0.020 0.984390
browserSafari               -8.260e-01  7.683e-01 -1.075 0.282304
browserSafari (in-app)       -1.699e+00  1.265e+00  -1.344 0.179092
browserUC Browser           -1.310e+01   1.036e+03 -0.013 0.989909
browserYaBrowser             -1.162e+01  8.593e+02  -0.014 0.989208
---
Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 15643.2  on 105436  degrees of freedom
Residual deviance:  8154.5  on 105360  degrees of freedom
  (187 observations deleted due to missingness)
AIC: 8308.5

Number of Fisher Scoring iterations: 19
```

B. ROC Curve of the Logistic Regression



C. R Codes used for Data Tidying through Cross-validation of the Logistic Regression Model

~~~~
```{r}
library(data.table)
library(readr)
library(dplyr)
library(tidyr)
library(magrittr)
library(lubridate)
library(purrr)
library(ggplot2)
library(gridExtra)
library(countrycode)
library(highcharter)
library(ggExtra)
```

################################# Data tidying and data wrangling #############################

```{r}

train_Data <- read.csv("~/Desktop/training_RAW.csv")

# Separating the data in customDimension column
train_custom <- data.frame(train_Data$customDimensions)

colnames(train_custom) <- c("AllValues")

train_custom$AllValues <- lapply(train_custom$AllValues, as.character)

train_custom$AllValues <- gsub("[\\[\"]", "", train_custom$AllValues)
train_custom$AllValues <- gsub("]", "", train_custom$AllValues)
train_custom$AllValues <- gsub("[\\{}\"]", "", train_custom$AllValues)
train_custom$AllValues <- gsub("[\\''\"]", "", train_custom$AllValues)

train_custom <- data.frame(separate(train_custom, AllValues,c("index", "value"), sep = ","))

train_custom <- data.frame(apply(train_custom, 2, function(y) (gsub("(.*: )", "", y))))
```


```{r}
# Separating the data in device column
train_device <- data.frame(train_Data$device)

colnames(train_device) <- c("AllValues")

train_device$AllValues <- lapply(train_device$AllValues, as.character)

train_device$AllValues <- gsub("[\\{}\"]", "", train_device$AllValues)

train_device <- data.frame(separate(train_device, AllValues,c("browser",
"browserVersion","browserSize","operatingSystem","operatingSystemVersion","isMobile","mobileDeviceBranding","mobile
DeviceModel","modelInputSelector","mobileDeviceInfo","mobileDeviceMarketingTeam","flashVersion","language","screenC
olors","screenResolution","deviceCategory"), sep = ","))
train_device <- apply(train_device, 2, function(y) (gsub("(.*: )", "",

y))) train_device[ train_device == "not available in demo dataset" ] <- NA

train_device <- as.data.frame(train_device)

train_device <- train_device[ , -which(names(train_device) %in%
c("browserVersion","operatingSystemVersion","browserSize","mobileDeviceBranding","mobileDeviceModel","mobileDeviceI
nfo","mobileDeviceMarketingTeam","modelInputSelector","flashVersion","language","screenColors","screenResolution",
"isMobile"))]
train_device <- as.data.frame(train_device)

```


```{r}
# Separating the data in geoNetwork column
train_geoNetwork <- data.frame(train_Data$geoNetwork)

colnames(train_geoNetwork) <- c("AllValues")

train_geoNetwork$AllValues <- lapply(train_geoNetwork$AllValues, as.character)

train_geoNetwork$AllValues <- gsub("[\\{}\"]", "", train_geoNetwork$AllValues)

train_geoNetwork$AllValues <- gsub(",MA,", "-MA,", train_geoNetwork$AllValues)
train_geoNetwork$AllValues <- gsub(",IA,", "-IA,", train_geoNetwork$AllValues)

train_geoNetwork <- data.frame(separate(train_geoNetwork, AllValues,c("continent",
"subContinent","country","region","metro","city","cityId","networkDomain","latitude","longitude","networkLocation"),
sep = ","))
train_geoNetwork <- apply(train_geoNetwork, 2, function(y) (gsub("(.*: )", "", y)))
train_geoNetwork[ train_geoNetwork == "not available in demo dataset" ] <- NA
train_geoNetwork[ train_geoNetwork == "(not set)" ] <- NA
train_geoNetwork[ train_geoNetwork == "unknown.unknown" ] <- NA


train_geoNetwork <- as.data.frame(train_geoNetwork)
~~~~

```
train_geoNetwork <- train_geoNetwork[ , -which(names(train_geoNetwork)
%in% c("cityId","latitude","longitude","networkLocation"))]
```

```{r}
# Separating the data in trafficSource column
train_trafficSource<- data.frame(train_Data$trafficSource)

colnames(train_trafficSource) <- c("AllValues")

train_trafficSource$AllValues <- lapply(train_trafficSource$AllValues, as.character)

train_trafficSource$AllValues <- gsub("[\\{}\"]", "", train_trafficSource$AllValues)

train_trafficSource <- as.data.frame(train_trafficSource)
train_trafficSource$campaign <- str_match(train_trafficSource$AllValues, "campaign: (.*?),")[,2]
train_trafficSource$source <- str_match(train_trafficSource$AllValues, "source: (.*?),")[,2]
train_trafficSource$medium <- str_match(train_trafficSource$AllValues, "medium: (.*?),")[,2]
train_trafficSource$criteriaParameters <- str_match(train_trafficSource$AllValues, "criteriaParameters:
(.*?)(?:,|$)")[,2]
train_trafficSource$gclId <- str_match(train_trafficSource$AllValues, "gclId: (.*?),")[,2]
train_trafficSource$adNetworkType <- str_match(train_trafficSource$AllValues, "adNetworkType: (.*?),")[,2]
train_trafficSource$isVideoAd <- str_match(train_trafficSource$AllValues, "isVideoAd: (.*?)$")[,2]
train_trafficSource$isTrueDirect <- str_match(train_trafficSource$AllValues, "isTrueDirect: (.*?)$")[,2]
train_trafficSource$referralPath <- str_match(train_trafficSource$AllValues, "referralPath: (.*?),")[,2]

train_trafficSource <- train_trafficSource[ , -which(names(train_trafficSource) %in% c("AllValues"))]
train_trafficSource[ train_trafficSource == "not available in demo dataset" ] <- NA
train_trafficSource[ train_trafficSource == "(not set)" ] <- NA
train_trafficSource[ train_trafficSource == "(not provided)" ] <- NA

train_trafficSource <- train_trafficSource[ , -which(names(train_trafficSource) %in% c("criteriaParameters"))]

```

```{r}
# Separating the data in totals column
train_totals <- data.frame(train_Data$totals)

colnames(train_totals) <- c("AllValues")

train_totals$AllValues <- lapply(train_totals$AllValues, as.character)

train_totals$AllValues <- gsub("[\\{}\"]", "", train_totals$AllValues)

train_totals$visits <- str_match(train_totals$AllValues, "visits: (.*?),")[,2]
train_totals$hits <- str_match(train_totals$AllValues, "hits: (.*?),")[,2]
train_totals$pageviews <- str_match(train_totals$AllValues, "pageviews: (.*?),")[,2]
train_totals$bounces <- str_match(train_totals$AllValues, "bounces: (.*?),")[,2]
train_totals$newVisits <- str_match(train_totals$AllValues, "newVisits: (.*?),")[,2]
train_totals$sessionQualityDim <- str_match(train_totals$AllValues, "sessionQualityDim: (.*?)^")[,2]
train_totals$timeOnSite <- str_match(train_totals$AllValues, "timeOnSite: (.*?),")[,2]
train_totals$transactionRevenue <- str_match(train_totals$AllValues, "transactionRevenue: (.*?),")[,2]

train_totals <- train_totals[ , -which(names(train_totals) %in% c("AllValues"))]

```

```{r}

#Removing columns which have been separated/which is not required for modeling
training <- train_Data[ , -which(names(train_Data) %in% c("customDimensions", "device", "geoNetwork",
"trafficSource", "totals", "hits"))]
#Merging the separated columns
training <- mutate(training, serial = 1:nrow(train_Data))
train_custom <- mutate(train_custom, serial = 1:nrow(train_custom))
train_device <- mutate(train_device, serial = 1:nrow(train_device))
train_geoNetwork <- mutate(train_geoNetwork, serial = 1:nrow(train_geoNetwork))
train_trafficSource <- mutate(train_trafficSource, serial = 1:nrow(train_trafficSource))
train_totals <- mutate(train_totals, serial = 1:nrow(train_totals))

training <- merge(training, train_custom, by.x = c("serial") ,by.y = c("serial"), all = TRUE)
training <- merge(training, train_device, by.x = c("serial") ,by.y = c("serial"), all = TRUE)
training <- merge(training, train_geoNetwork, by.x = c("serial"),by.y = c("serial"), all = TRUE)
training <- merge(training, train_trafficSource, by.x = c("serial"),by.y = c("serial"), all = TRUE)
training <- merge(training, train_totals, by.x = c("serial"),by.y = c("serial"), all = TRUE)
```

```{r}
#Updating the datatype of the columns as required
training <- transform(training, date = as.Date(date))
training$transactionRevenue <- as.numeric(training$transactionRevenue)
training$fullVisitorId <- as.character(training$fullVisitorId)
training$visitId <- as.character(training$visitId)
training$pageviews <- as.integer(training$pageviews)
training$bounces <- as.integer(training$bounces)
training$newVisits <- as.integer(training$newVisits)
training$timeOnSite <- as.integer(training$timeOnSite)
training$isVideoAd <- as.logical(training$isVideoAd)
training$isTrueDirect <- as.logical(training$isTrueDirect)

```

```{r}
#Keeping necessary columns
training <- training[ , -which(names(training) %in%
c("X","serial","socialEngagementType","index","value","isVideoAd"))]
#Adding response variable based on availability of transaction revenue
```

```
training <- mutate(training, Transaction = ifelse(is.na(transactionRevenue),0,1))

training <- transform(training, as.factor(training$Transaction))
```

```{r}
#Adding transformed variable week
training$Week <- as.character(trunc(difftime(training$date,strptime("25.11.2016", format
= "%d.%m.%Y"),units="weeks"))+1)
training <- transform(training, Week = ifelse(is.na(Week),Week,paste("Week ",Week)))

training <- transform(training, Week = as.factor(Week))

```

```{r}
#Adding transformed variable visithour
training$visitStartTime <- as_datetime(training$visitStartTime)

training$visithour <- hour(as.POSIXct(training$visitStartTime))

training <- transform(training, visithour = as.factor(visithour))

```

```{r}
#Updating the source column to take care of redundant values
training <- transform(training, source = as.character(source))
training$tempSource <- str_detect(training$source, "google")
training <- transform(training, source = ifelse(tempSource, "google", source))
training$tempSource <- str_detect(training$source, "facebook")
training <- transform(training, source = ifelse(tempSource, "facebook", source))
training$tempSource <- str_detect(training$source, "youtube")
training <- transform(training, source = ifelse(tempSource, "youtube", source))
training$tempSource <- str_detect(training$source, "quora")
training <- transform(training, source = ifelse(tempSource, "quora", source))
training$tempSource <- str_detect(training$source, "baidu")
training <- transform(training, source = ifelse(tempSource, "baidu", source))
training$tempSource <- str_detect(training$source, "reddit")
training <- transform(training, source = ifelse(tempSource, "reddit", source))
training$tempSource <- str_detect(training$source, "bing")
training <- transform(training, source = ifelse(tempSource, "bing", source))
training$tempSource <- str_detect(training$source, "amazon")
training <- transform(training, source = ifelse(tempSource, "amazon", source))
training$tempSource <- str_detect(training$source, "yahoo")
training <- transform(training, source = ifelse(tempSource, "yahoo", source))
training$tempSource <- str_detect(training$source, "github")
training <- transform(training, source = ifelse(tempSource, "github", source))
training$tempSource <- str_detect(training$source, "pinterest")
training <- transform(training, source = ifelse(tempSource, "pinterest", source))
training$tempSource <- str_detect(training$source, "live.com")
training <- transform(training, source = ifelse(tempSource, "live.com", source))
training$tempSource <- str_detect(training$source, "ask.com")
training <- transform(training, source = ifelse(tempSource, "ask.com", source))
training$tempSource <- str_detect(training$source, "vk")
training <- transform(training, source = ifelse(tempSource, "vk", source))
training_bySource <- summarise(group_by(training,source),count = n())

training <- left_join(training,training_bySource,by="source")

training <- transform(training, source = ifelse(count > 100, source, "Others"))

training <- transform(training, source = as.factor(source))

training <- training[ , -which(names(training) %in% c("tempSource"))]

```

```{r}
#Updating operating system column value to as "Others" for categories with frequency<100
training <- transform(training, operatingSystem = as.character(operatingSystem))
training_byOS <- summarise(group_by(training,operatingSystem),count = n())

training <- left_join(training,training_byOS,by="operatingSystem")

training <- transform(training, operatingSystem = ifelse(count > 100, operatingSystem, "Others"))

training <- transform(training, operatingSystem = as.factor(operatingSystem))

```

```{r}
#Updating browser column value to as "Others" for categories with
frequency<100 training <- transform(training, browser = as.character(browser))
```

```
training_byBrowser <- summarise(group_by(training,browser),count = n())

training <- left_join(training,training_byBrowser,by="browser")

training <- transform(training, browser = ifelse(count > 100, browser, "Others"))

training <- transform(training, browser = as.factor(browser))

```

```{r}
#Updating channel grouping column as NA if it is (Other), as there is only one record with the
value training <- transform(training, channelGrouping = as.character(channelGrouping))
training <- transform(training, channelGrouping = ifelse(channelGrouping!="(Other)", channelGrouping, NA))

```

################################ Metadata analysis #############################

```{r}
training <- training[ , -which(names(training) %in% c("count.x","count.x.x","count.y","count.y.y"))]

metadata1 <- data.frame(sapply(training, function(x) sum(is.na(x))))
colnames(metadata1) <- c("NAs")

metadata1$column <- rownames(metadata1)
rownames(metadata1) <- 1:nrow(metadata1)

metadata1 <- metadata1[,c(2,1)]

metadata2 <- data.frame(sapply(training, function(x) sum(!is.na(x))))
colnames(metadata2) <- c("Availables")

metadata2$column <- rownames(metadata2)
rownames(metadata2) <- 1:nrow(metadata2)

metadata3 <- data.frame(sapply(training, function(x) typeof(x)))
colnames(metadata3) <- c("Type")

metadata3$column <- rownames(metadata3)
rownames(metadata3) <- 1:nrow(metadata3)

metadata <- inner_join(metadata1, metadata2, by = "column")
metadata <- inner_join(metadata, metadata3, by = "column")

metadata <- mutate(metadata, total = NAs + metadata$Available)

metadata <- gather(metadata, "NAs", "Availables", key = "AvailType", value = "Number")

ggplot(metadata,aes(x = column,fill = AvailType, y = Number)) +
   geom_bar(position = "fill", stat = "identity") + coord_flip()
```

################################ Exploratory data analysis #############################

```{r}
#Plotting visitors vs customers

data <- training %>% select(fullVisitorId, transactionRevenue) %>% mutate(revenue =
ifelse(is.na(transactionRevenue),0,transactionRevenue))
data <- data %>% select(-transactionRevenue)

new_data <- data %>% mutate(visitorid = format(fullVisitorId, digits = 20)) %>% group_by(visitorid) %>%
summarise(visits = n(), Total_Revenue=sum(as.numeric(revenue)))
xx <- new_data %>% transmute(flag = ifelse(Total_Revenue == 0, "Visitors", "Customers")) %>% group_by(flag) %>% count(flag)

ggplot(xx, aes(x=flag, y=n, fill=flag)) + geom_col(position = "stack")
  + scale_fill_manual(values=c("lightblue", "lightgreen")) +
theme_bw() +
  xlab("")+
ylab("Count") +
ggtitle("Visitors vs Customers") +
labs(fill="")

```

```{r}
#Plotting daily revenues for the period in context for comparision
TotalRev <- summarise(group_by(training,date), Total_revenue = sum(as.numeric(log(transactionRevenue))))

colnames(TotalRev) <- c("Date", "Total Revenue")

revenue_gt_7.5 <- ifelse(TotalRev$`Total Revenue` >=1000, "More than 750$","Less than 750$")

avg_rev <- mean(TotalRev$`Total Revenue`)
library(ggrepel)


ggplot(filter(TotalRev, !is.na(`Total Revenue`)), aes(x=Date, y=`Total
Revenue`))+ geom_hline(yintercept =avg_rev, color="red") +
geom_label_repel(aes(label =as.character.Date(Date)), nudge_y = 1, alpha = 0.7, size=2)
+ scale_fill_manual(values=c("black", "red")) +
geom_line(size=1) +
theme_bw() +
xlab(colnames(TotalRev)[1]) +
ylab(colnames(TotalRev)[2]) +
labs(
  title = "Daily Revenue",
  caption = "Note: Red horizontal line is the average daily revenue."
)

```

```{r}
#Plotting comparison of number of sessions against number of transaction per device and operating
system g1 <- training %>%
  filter(!is.na(operatingSystem), !is.na(deviceCategory))%>%
  group_by(operatingSystem,deviceCategory)%>%
  summarise(n = n())%>%
  ggplot(aes(x=reorder(operatingSystem, n), y=n, fill = deviceCategory)) +
  geom_bar(stat='identity') + facet_wrap(~deviceCategory,scales = "free_x")+
  theme(axis.text.x = element_text(angle = 90, hjust = 1))+coord_flip() +
  labs(x="Operating system",y = "Number of sessions", title="Number of sessions per operating system and device
type", fill = "Device type")
g2 <-  training %>%
  filter(!is.na(operatingSystem), !is.na(deviceCategory), !is.na(transactionRevenue))%>%
  group_by(operatingSystem,deviceCategory)%>%
  summarise(n = n())%>%
  ggplot(aes(x=reorder(operatingSystem, n), y=n, fill = deviceCategory)) +
  geom_bar(stat='identity') + facet_wrap(~deviceCategory,scales = "free_x")+
  theme(axis.text.x = element_text(angle = 90, hjust = 1))+coord_flip() +
  labs(x="Operating system",y = "Number of transactions", title="Number of transactions per operating system and
device type", fill = "Device type")
listPlots <- list(c("g1","g2"))
l = mget(listPlots[[1]])

ggsave("~/Desktop/plot5.jpeg", arrangeGrob(grobs = l))

```

```{r}
#Plotting pie chart of transaction revenue from different sources

training_model_source <- transform(training, source = as.character(source))

training_model_AllSources <- transform(training_model_source, source = ifelse(source == "(direct)","Direct","Others"))

training_model_otherSource <- filter(training_model_source, source!="(direct)")

revenue_by_source <- summarise(group_by(filter(training_model_AllSources,
!is.na(transactionRevenue)),source), revenue = sum(as.numeric(transactionRevenue)))
revenue_by_otherSources <- summarise(group_by(filter(training_model_otherSource,
!is.na(transactionRevenue)),source), revenue = sum(as.numeric(transactionRevenue)))
revenue_by_source$Percentage <- round(revenue_by_source$revenue / sum(revenue_by_source$revenue),3)

revenue_by_otherSources$Percentage <- round(revenue_by_otherSources$revenue / sum(revenue_by_otherSources$revenue),3)

revenue_by_otherSources <- transform(revenue_by_otherSources,source = ifelse(Percentage>.1,source,"Others"))

revenue_by_otherSources <- summarise(group_by(revenue_by_otherSources,source), Percentage = sum(Percentage))

library(plotly)


p1 <- plot_ly(revenue_by_otherSources, labels = ~source, values = ~Percentage, type =
       'pie', textposition = 'inside',
       textinfo = 'label+percent',
       insidetextfont = list(color = '#FFFFFF'),
       hoverinfo = 'text',
       text = ~paste( Percentage, ' %'),
       marker = list(colors = colors,
                  line = list(color = '#FFFFFF', width = 1)),
                  #The 'pull' attribute can also be used to create space between the
       sectors showlegend = FALSE) %>%
  layout(title = 'Revenue from other sources',
        xaxis = list(showgrid = FALSE, zeroline = FALSE, showticklabels = FALSE),
        yaxis = list(showgrid = FALSE, zeroline = FALSE, showticklabels = FALSE))


p2 <- plot_ly(revenue_by_source, labels = ~source, values = ~Percentage, type = 'pie',
       textposition = 'inside',
       textinfo = 'label+percent',
       insidetextfont = list(color = '#FFFFFF'),
       hoverinfo = 'text',
       text = ~paste( Percentage, ' %'),
       marker = list(colors = colors,
                  line = list(color = '#FFFFFF', width = 1)),
                  #The 'pull' attribute can also be used to create space between the
       sectors showlegend = FALSE) %>%
  layout(title = 'Revenue from sources',
        xaxis = list(showgrid = FALSE, zeroline = FALSE, showticklabels = FALSE),
        yaxis = list(showgrid = FALSE, zeroline = FALSE, showticklabels = FALSE))

```

```{r}
#Plotting heatmap for transaction revenues generated across the countries in the world
by_country <- training%>%
           filter(!is.na(country))%>%
           group_by(country)%>%
           summarise(revenue =sum(log(as.numeric(transactionRevenue)), na.rm=TRUE))
```

```
training <- transform(training, transactionRevenue = as.numeric(transactionRevenue))

training$transactionRevenue <- ifelse(is.na(training$transactionRevenue),0,log(training$transactionRevenue))

by_country <- summarise(group_by(filter(training,!is.na(country)),country), revenue = sum(transactionRevenue))

by_country$iso3 <- countrycode(by_country$country, origin='country.name', destination='iso3c')

highchart() %>%
    hc_add_series_map(worldgeojson, by_country, value = 'revenue', joinBy = 'iso3') %>%
    hc_title(text = 'Total transaction revenue by country') %>%
    hc_tooltip(useHTML = TRUE, headerFormat = "",
        pointFormat = "{point.country}: ${point.revenue:.0f}")
```

~~~
################################ Cross-validation and modeling #############################

```{r}

training_revenue <- filter(training, !is.na(training_model$transactionRevenue))

training_noRevenue <- filter(training, is.na(training_model$transactionRevenue))

number_of_folds <- 5

overall_accuracy <- 0

true_positive_proportion <- 0

true_negative_proportion <- 0

rmseTotal <- 0

set.seed(1)

splitted_revenue <- split(training_revenue, sample(1:number_of_folds, nrow(training_revenue), replace=T))

splitted_noRevenue <- split(training_noRevenue, sample(1:number_of_folds, nrow(training_noRevenue), replace=T))

  for(i in 1:number_of_folds)
  {
    test_data_revenue  <- splitted_revenue[[i]]
    test_data_noRevenue <- splitted_noRevenue[[i]]

    test_data <- rbind(test_data_revenue, test_data_noRevenue)

    k = 1

    for(j in 1:number_of_folds)
    {
      if(j == i){next}


      if(k==1)
      {
        train_data_revenue <- splitted_revenue[[j]]
        train_data_noRevenue <- splitted_noRevenue[[j]]
      }
      else
      {
        train_data_revenue <- rbind(train_data_revenue, splitted_revenue[[j]])
        train_data_noRevenue <- rbind(train_data_noRevenue, splitted_noRevenue[[j]])
      }

      k = k + 1

    }

    train_data <- rbind(train_data_revenue, train_data_noRevenue)

    fit_linear <- lm(log(transactionRevenue) ~ Week + visits + visitNumber + visithour + source + pageviews + operatingSystem
+ medium + hits + deviceCategory + continent + channelGrouping + browser, data = train_data_revenue)
    fit_logit <- glm(Transaction ~ Week + visits + visitNumber    + visithour + source + pageviews + operatingSystem + medium
+ hits + deviceCategory + continent + channelGrouping +
browser, family=binomial(link="logit"), data=train_data)


    test_data$Predicted <- predict(fit_logit, test_data, type="response")

    test_data <- transform(test_data, Predicted = ifelse(test_data$Predicted > 0.03, 1, 0))

    overall_accuracy <- overall_accuracy + summarise(filter(test_data,!is.na(Transaction), !is.na(Predicted)),
mean(Predicted == Transaction))
    true_negative_proportion <- true_negative_proportion +
summarise(filter(test_data, !is.na(Transaction), !is.na(Predicted), Transaction == 0), mean(Predicted == Transaction))

    true_positive_proportion <- true_positive_proportion +
summarise(filter(test_data, !is.na(Transaction), !is.na(Predicted), Transaction == 1), mean(Predicted == Transaction))

    rmseTotal <- rmseTotal + rmse(fit_linear, filter(test_data, !is.na(Transaction), !is.na(Predicted), Transaction
== 1, Transaction == Predicted))
  }

  overall_accuracy <- overall_accuracy/number_of_folds

  true_positive_proportion <- true_positive_proportion/number_of_folds

  true_negative_proportion <- true_negative_proportion/number_of_folds
~~~

~~~
  rmseTotal <- rmseTotal/number_of_folds
```
~~~