Introduction

In this assignment, a large dataset pertaining to the XYZ organization, was provided to help determine which customers to send mailers to for XYZ's 17th campaign. The intent of this assignment is to determine the approximate net profit XYZ can expect from those customers that XYZ sends the mailer to. Three different models were developed to derive a strong predictive model that could be leveraged. Furthermore, the best performing model was used to determine the potential net profit XYZ could realize.

Data Overview

The dataset used in this assignment was provided in a pre-processed format (herein referred to as the raw data). Dr. Miller had taken the raw data from XYZ's database and converted it into a 'tidy' format - where each row is an observation and each column is a variable (Wickham, 2014). The raw data consisted of 30,779 observations across 554 variables (i.e., predictors). Upon importing the dataset, there were 345 character variables, 48 integer variables, and 161 numeric variables.

In order to reduce the large number of potential predictors, a subjective analysis was done to determine if a variable should be kept or not. This analysis relied on the provided descriptions in the data dictionary, a quick overview of the unique values within each variable, and the amount of missing observations. If a description could not be found, and/or if the variable was determined to have excessively high number of unique values, and/or if the variable was missing more than 10% of the observations then it was removed from the variable list[1]. This initial run-through resulted in 191 variables being selected. However, this number was further reduced through feature engineering (which resulted in the removal of some of the variables) and a construction of a naïve random forest model to select only the most important variables. These steps are discussed further in the next sections.

Data Pre-Processing

Pre-processing and missing data imputation was done in junction with the EDA. However, for this report, they are presented in a pseudo-seqential manner. For instance, all of the character variables were converted to factors. Furthermore, integer based variables were explored to determine if they needed to be factors or integers. As an example, the variable CHANNEL_ACQUISITION was initially a character variable that was converted to factors as it consisted of only three different types. On the other hand, the variable ZPETSP is an integer that was converted to a factor as it is ordinal.

Exploratory Data Analysis

A comprehensive exploratory data analysis was completed for this analysis (a small sample of the charts constructed can be found in the Appendix). Many of the values were found to have missing observations. In fact, of the numerous predictors, 300 of them had missing values. From this process, variables that had more than 10% missing observations were removed from the final dataset used to construct the training and validation datasets. Table 2.1 summarizes the overall response for campaign 16 given that a customer had received a mailer for the 16th campaign. Note the imbalance between customers who had responded to a mailer and customers who had not (approximately 10% of the customers responded to mailers).

[1] A separate document was kept for a deeper analysis and can be provided upon request.

Table 2.1: Summary Table on Response16

Response	Count	Percent
0	13482	0.9035
1	1440	0.0965

It was also found that multiple mailers were sent to customers in certain campaigns. It was not clear from the analysis and the data dictionaries as to why a customer may receive more than one mailer. One interesting observation was found in the variable related to the median travel time to work. In Figure 2.1, it is interesting to note that the customers who had a positive response were seeing 200 - 300 minutes of travel time. Intuitively, this could mean that customers who have a longer commute have more time to dwell on their buying decision and may be responsive to making a purchase. This enables the potential consumer ot absorb the information from a mailer and engage in purchasing as it may pass the time.

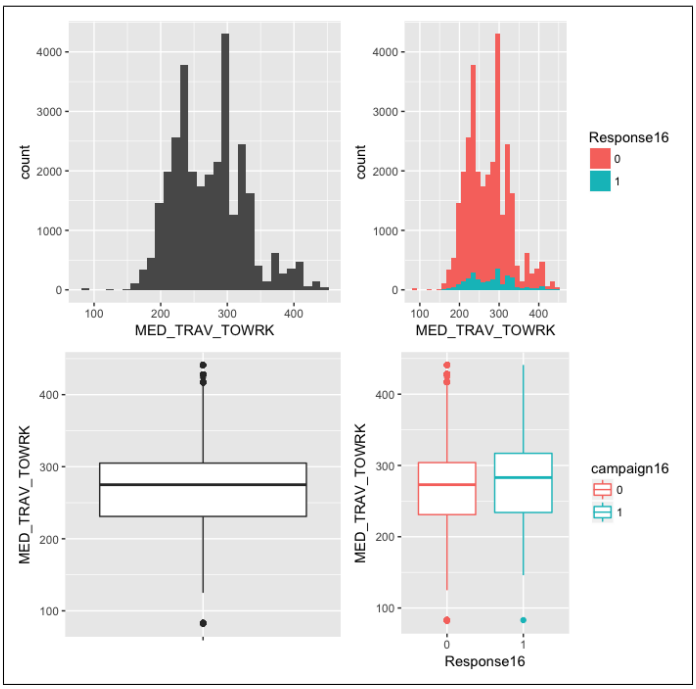

Figure 2.1: Histogram and Boxplot for Total Amount Prior to Campaign 16

Missing Data Imputation

Many of the variables had missing values or observations. To simplify matters, all factors that were either missing values or had the letter U[2], were classified as "Unknown". For any numeric variable, the missing values were imputed with the median value (the median value was calculated by removing the missing values first).

Feature Engineering

Nine feature variables were created for this analysis. Table 7.1 in the Appendix summarizes the new variables that were constructed. Of note is the variable *high.value.acct*. This variable makes use of the 90th

[2]the character U was construed as meaning "unknown" and this was used consistently in the provided data dictionary

percentile of total revenue amount prior to the 16th campaign. However, for simplicity's sake, the value was rounded down to $500.

Naïve Random Forest Model

Since the initial variable reduction (completed subjectively) resulted in 163 potential predictors, a naïve random forest model was constructed to help highlight the top 25 predictors using the gini impurity index. Figure 2.2 lists the top 25 variables.

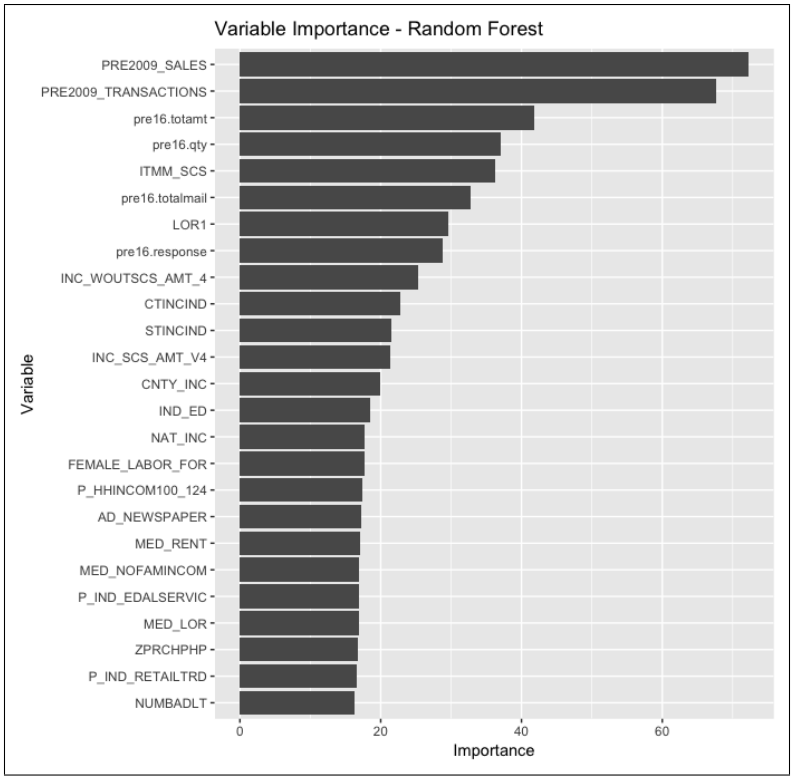

Figure 2.2: Variable Importance Plot

Note the steep drop off in variable importance after the first two variables. It was concluded that these top 25 variables would be used in the model development phase as it would offer suitable trade-off between accuracy and performance.

Model Development

Three different models were constructed for this analysis: Random Forest, Generalized Linear Model, and Naïve Bayes. Each model was developed using the 16th campaign response (RESPONSE16[3]) as the response variable. Two distinct datasets were constructed: one for use to modeling the responses to the 16th campaign and another to predict which accounts should get the 17th campaign mailers. In order to model the 16th campaign, a subset of the overall data was created by setting the ANY_MAIL_16 variable equal to 1. Once this was done, this dataset was split into a training partition (consisting of 80% of the observations) and a validation partition (consisting of the remaining 20% of the observations). Once a

[3]this variable has 2 classes: 0 for no mailer and 1 for if mailer was sent

preferred model was derived, a new dataset was created by setting the ANY_MAIL_16 variable equal to 0[4]. This dataset acted like the test dataset for which predictions would be made against.

All three models were used to classify the appropriate response as well as the probabilities for each class. Each model was assessed using several key attributes: the accuracy, sensitivity, specificity, and Kappa[5] for the classification and AUC (area under the curve) for the probabilities. All metrics were compared holistically to determine the best performing model and are discussed more in Model Results & Conclusion.

Model 1: Random Forest

The package 'ranger' was used to develop the random forest model. 1000 trees were constructed and five random variables were considered at each split within each tree. The confusion matrices for both the in-sample (i.e., the training dataset) and out-of-sample (i.e., the validation dataset) are in Table 3.1.

Table 3.1: Random Forest: In-Sample & Out-of-Sample Confusion Matrix

	In-Sample	
	Observed: 0	**Observed: 1**
Predicted: 0	10786	0
Predicted: 1	0	1152

	Out-of-Sample	
	Observed: 0	**Observed: 1**
Predicted: 1	2693	286
Predicted: 1	3	2

Note how the random forest model performed exceptionally well in the in-sample dataset, but performed poorly in the out-of-sample dataset. Table 3.4 in the Model Results & Conclusion summarizes the model metrics for the random forest model (along with the others).

Figure 3.1 highlights the in-sample and out-of-sample ROC curves. The benefit of the AUC curve, especially in a binary outcome problem such as the one in this analysis, is that it is much more robust to class imbalances. Note how the in-sample ROC curve is a perfect 1.

[4]this implies that the account never received a mailer
[5]known as Cohen's Kappa

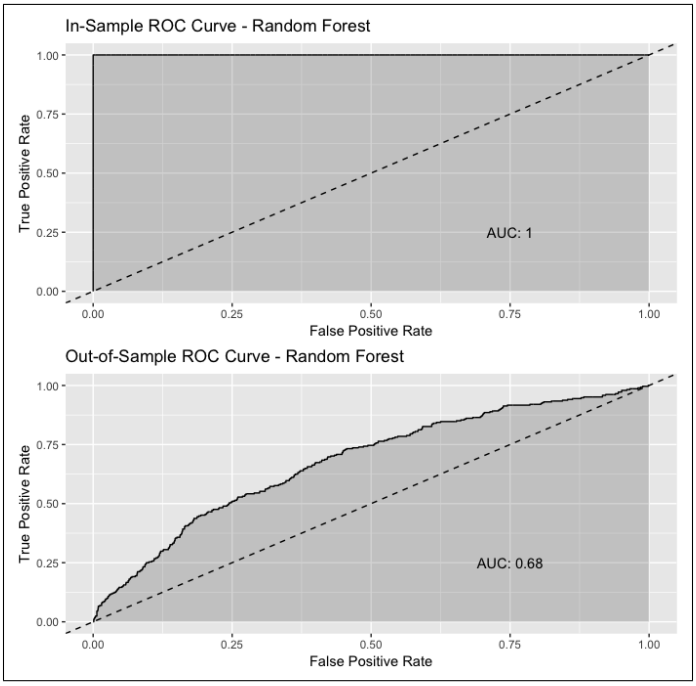

Figure 3.1: Random Forest AUC Curves

Model 2: GLMNet

The GLMNet model was actually an elastic-net model that was constructed. In this model, the regularization was a balance of the lasso and ridge regression techniques. Essentially, the model blended the ability of automated variable selection (lasso) and reducing the coefficients of unimportant variables to zero. This was first achieved by using cross-validation to determine the best α and then another cross-validation to determine the best λ value.

The confusion matrices for both the in-sample (i.e., the training dataset) and out-of-sample (i.e., the validation dataset) are in Table 3.2

Table 3.2: GLMNet: In-Sample & Out-of-Sample Confusion Matrix

	In-Sample	
	Observed: 0	**Observed: 1**
Predicted: 0	10766	1138
Predicted: 1	20	14

	Out-of-Sample	
	Observed: 0	**Observed: 1**
Predicted: 0	2691	287
Predicted: 1	5	1

Note how the out-of-sample performance is not much better than the random forest model. Figure 3.2 highlights the in-sample and out-of-sample ROC curves.

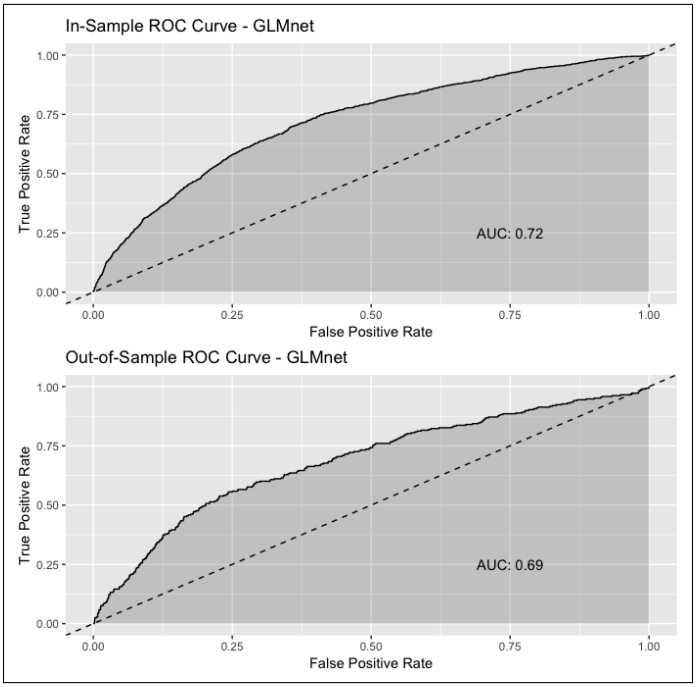

Figure 3.2: GLMNet AUC Curves

Similar to the Random Forest model, the out-of-sample AUC (Area Under the Curve) is similar, but the in-sample curve is not 'perfect'. Table 3.4 in the Model Results & Conclusion summarizes the model metrics for the glmnet model (along with the others).

Model 3: Naïve Bayes

For this model, a 5-fold cross validation was completed using the "naivebayes" method in the package "caret". The confusion matrices for both the in-sample (i.e., the training dataset) and out-of-sample (i.e., the validation dataset) are in Table 3.3.

Table 3.3: Naïve Bayes: In-Sample & Out-of-Sample Confusion Matrix

	In-Sample	
	Observed: 0	**Observed: 1**
Predicted: 0	9999	856
Predicted: 1	787	296

	Out-of-Sample	
	Observed: 0	**Observed: 1**
Predicted: 0	2500	225
Predicted: 1	196	63

Note how the out-of-sample classifications are substantially better than both GLMNet and Random Forest. Figure 3.3 highlights the in-sample and out-of-sample ROC curves.

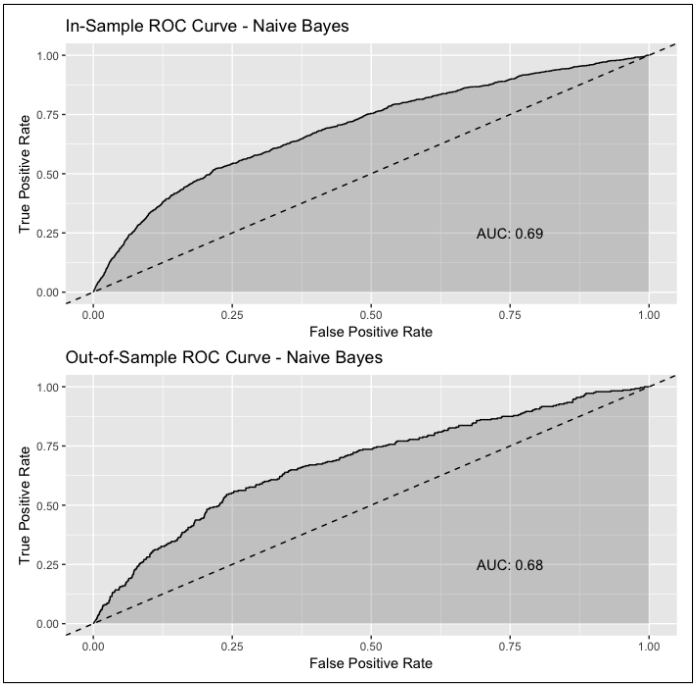

Figure 3.3: GLMNet AUC Curves

The in-sample ROC curves for the Naïve Bayes model is better than the GLMNet model, but nowhere near as well as the Random Forest. However, the out-of-sample AUC for the Naïve Bayes model is very slightly better than both the GLMnet and Random Forest models. Table 3.4 in the Model Results & Conclusion summarizes the model metrics for the Naïve Bayes model (along with the others).

Model Results & Conclusion

Table 3.4 summarizes each of the three models' statistics. All three models perform well in terms of accuracy on the out-of-sample dataset, but the same cannot be said about sensitivity. Recall that sensitivity is the true positive rate and that specificity is the rate of correctly predicted negatives. Note how the Naïve Bayes model had the best sensitivity and the random forest model had the best specificity.

Table 3.4: Model Statistics Summary

	Random Forest		**GLMNet**		**Naïve Bayes**	
	In-Sample	**Out-of-Sample**	**In-Sample**	**Out-of-Sample**	**In-Sample**	**Out-of-Sample**
accuracy	1	0.903	0.903	0.9025	0.8202	0.859
sensitivity	1	0.007	0.014	0.0104	0.39	0.219
specificity	1	0.999	0.998	0.998	0.866	0.927
Kappa	1	0.011	0.0209	0.0144	0.1989	0.1529
AUC	1	0.676	0.719	0.685	0.694	0.679

In terms of AUC, all three models used a probability of 0.5 as the threshold to determine if a classification should be 1 (mailer sent) or 0 (no mailer sent). Typically in an imbalanced dataset such as this, the probability threshold will have to different. This threshold could be determined by using the derived ROC curves, and has a direct impact on accuracy and sensitivity of the confusion matrices. Hence, in the test dataset, different probability thresholds will be assessed to determine the optimal net revenue.

Looking at the various metrics holistically, it appears that the Naïve Bayes model performs well. For instance, it has a much more robust Kappa value than the other two models and it's sensitivity is much stronger than the others. Therefore, the Naïve model will be used to assess the test dataset and project net revenue.

Net Revenue Summary

The average net profit of sending one mailer was assumed to be approximately $269. This was a derived value based on the total amount of money spent by customers who had received a mailer for the 16th campaign. In other words, the training dataset was used to derive the aforementioned profit. Within this calculation, the cost of the mailer, assumed to be $3 per mailer, was included.

Leveraging the Naïve Bayes model, the probabilities of a customer (in the test data set) being sent a mailer was derived. Different probability thresholds were tested to estimate the projected profit. The estimated (i.e., projected profit) was calculated by taking the probability (assuming it was equal to or greater than the threshold) and multiplying by the average profit. Table 4.1 summarizes the expected profit by different probability thresholds.

Table 4.1: Summary of Expected Profit by Probability Threshold

Probability Threshold	Projected Number of Mailers to Send	Expected Proift
0.0	15857	143421.16
0.1	782	115803.83
0.2	579	108262.98
0.3	484	102055.09
0.4	422	96303.79
0.5	388	92204.96
0.6	351	86736.64
0.7	317	80828.39
0.8	289	75134.06
0.9	252	66655.31
1.0	124	33323.7

From an initial perspective, it appears that by sending a mailer to every customer, the XYZ organization can realize maximum profit - given the assumptions made. However, by sending mailers to only 782 (less than 5% of the overall total of 15,857 accounts) accounts, the company only loses out on about $27,000 in profit. This is suggestive of the fact that many of the accounts have a probability of less than 10% and may not be worthwhile pursuing.

Limitations & Assumptions

Several assumptions were made to achieve the results highlighted in this analysis. For instance, the initial variable reduction technique was highly subjective. This may have resulted in variables being discarded that would have been useful for the overall modeling process. Furthermore, with an inadequate data dictionary, it was not readily apparent on how to develop deep contextual understanding of many of the variables. This, in turn, prevented the ability to select variables that may prove to be more valuable. Another assumption was made on the type of variables chosen. It is possible that some variables may have been marked as factors (i.e., categoricals) when in reality they should have been numeric. It is recommended that these

results be reviewed by a subject matter expert so that a deeper contextual understanding can be achieved prior to any key decision making process.

The key limitation of this analysis is that it may not be transferable to other regions or even different customer bases. All of the consumers in this analysis were concentrated in the Greater Chicago-land area and their purchasing habits may be similar. Furthermore, there is no clear context into the actual purchase that a customer made. For example, a customer may have purchased a single high ticket item or numerous items that were of low value but were equal. Another limitation is that the key time data had been removed and this removal may have resulted in the loss of seasonality. Depending on the actual products purchased and in what season, the predicted outcomes may vary. For instance, customers may be willing to purchase more clothing in spring and fall rather than in winter (as an example).

Conclusion & Next Steps

Three different models were constructed for this analysis: random forest, GLMnet, and Naïve Bayes. All three models were assessed using both classification oriented metrics (such as accuracy & sensitivity) in addition to the area under the curve (AUC). Due to the imbalanced dataset, it was essential to look at each metric holistically to ensure that the most impactful model would be chosen. In the end, the Naïve Bayes model was selected due to its overall performance on the training and validation datasets.

Many of the models constructed use a default probability of 0.5 to determine if the binary response variable should be 1 or 0. However, in this analysis different probability thresholds were explored to understand the impact on total profit. Unsurprisingly, the organization can realize a maximum profit by sending a mailer to every account. However, the amount of revenue gained this way versus using a minimal threshold of 10% (which essentially means the probability of a customer making a purchase given that they receive a mailer) is marginal at best.

For next steps, it is suggested that a more robust data dictionary be created. It is also important for the XYZ organization to work more towards data quality to ensure that the end result continue to be relevant. These methods will, undoubtedly, result in improved variable selection that could further improve the overall models.

References

Wickham, H. (2014, 8). Tidy data. *Journal of Statistical Software*, *59*(10).

Appendix

EDA

The figure below illustrates the frequency chart for the variable (i.e., predictor) Channel Acquisition. Note how the values are heavily skewed for each type of channel acquisition. This is due to the impbalance in the dataset pertaining to the response of the accounts in the 16th campaign.

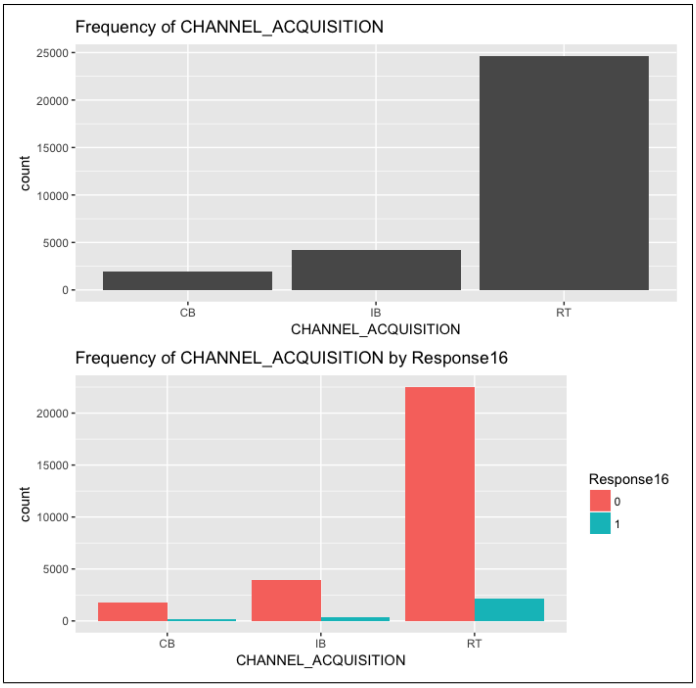

Figure 7.1: Frequency Chart for Channel Acquisition

In the figure below, note the difference in the distribution between a response of 0 or 1 in campaign 16. Unsurprisingly, the histograms are skewed heavily to the right due to no revenue for the large number of mailers sent.

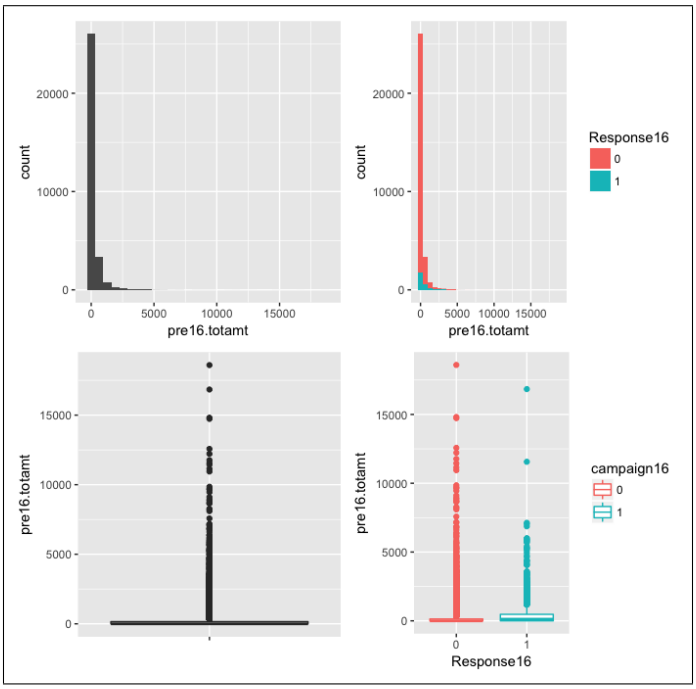

Figure 7.2: Histogram and Boxplot for Total Amount Prior to Campaign 16

The figure below highlights the county income percentile. Note how the median customers who had a positive response to the campaign seem to be from a slightly higher percentile than ones from a negative response. Furthermore, it is interesting to note that many of the respondents come from a high county income percentile.

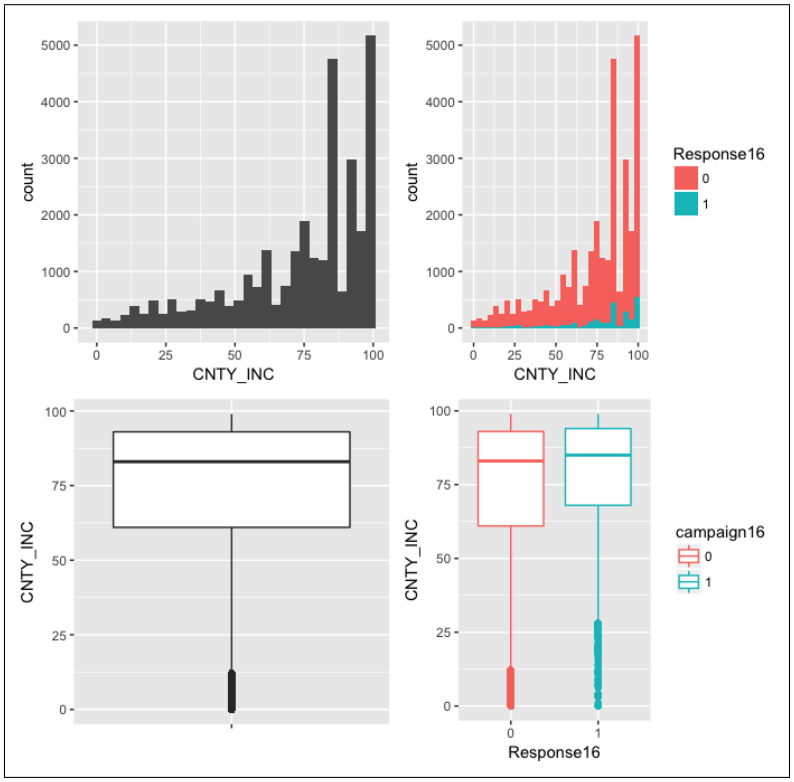

Figure 7.3: Histogram and Boxplot for Total Amount Prior to Campaign 16

Feature Engineering

Table 7.1 lists the features that were created for the analysis along with a brief description.

Table 7.1: Descriptions of Feature Variables

New Variable	Type	Description
pre16.qty	numeric	number of items ordered before the 16th campaign
pre16.totamt	numeric	total revenue before the 16th campaign
high.value.acct	factor	true if the account had $500 or more in sales prior to the 16th campaign
pre16.response	numeric	total number of responses prior to the 16th campaign
pre16.totalmail	numeric	total number of mailers prior to the 16th campaign
pre16.buyer.status	factor	true if the account had purchased at least once prior to the 16th campaign
pre16.got.mailer	factor	true if the account had received any mailer prior to the 16th campaign
pre16.bought.with.mailer	factor	true if the account had purchased anything if sent a mailer
pre16.bought.without.mailer	factor	true if the account had purchased anything if no mailer was sent

R Code

The R code for this assignment was development using a modular approach. Several files were created and the file called ”01-executor.R” was used to load and process the dataset. Each file is documented below.

01-executor.R

```
# Nikhil Agarwal
# PREDICT 450
# Northwestern University


# THIS FILE IS USED TO LOAD THE APPROPRIATE FILES

# load libraries
library(tidyverse)  # ez coding
library(caret)      # model development
library(gridExtra)  # multiple grids

# Load the script that loads data frames ----

source('scripts/02-load-data.R')
source('scripts/03-condense.R')
source('scripts/03a-campaign17-df.R')
source('scripts/04-convert-data-type.R')
source('scripts/06-impute.R')
source('scripts/07-data-split.R')

# activate Multicore processing (macs only)
library(doParallel)
registerDoParallel(cores=2)
```

02-load-data.R

```
# Nikhil Agarwal

# Load the data files ----

load('data/XYZ_complete_customer_data_frame.RData')


# store dataframe in a new named dataframe

customer.data <- complete.customer.data.frame


# need to replace the "" and " " with NA's
# assume that all U's are unknowns and equal to NAs

customer.data[customer.data == "" | customer.data == " " | customer.data == 'U'] <- NA
```

03-condense.R

```
# data consolidation


# first conslidate the number of columns

keep.vars <- c(
'ACCTNO',
'ANY_MAIL_16',
'CHANNEL_ACQUISITION',
'ZIP',
'INC_SCS_AMT_V4',
'INC_WIOUTSCS_V4',
'INC_WITHSCS_V4',
'INC_WOUTSCS_AMT_4',
'FIPSCNTY',
'P_HHINCOLES10M',
'P_HHINCOM10_14',
'P_HHINCOM15_19',
'P_HHINCOM20_24',
'P_HHINCOM25_29',
'P_HHINCOM30_34',
'P_HHINCOM35_39',
'P_HHINCOM40_44',
'P_HHINCOM45_49',
'P_HHINCOM50_59',
'P_HHINCOM60_74',
'P_HHINCOM75_99',
'P_HHINCOM100_124',
'P_HHINCOM125_149',
'P_HHINCOM150_199',
'P_HHINCOM20_UP',
'MED_INC',
'MED_FAMINCOM',
'MED_NOFAMINCOM',
'P_CAPITA_INCOM',
'FEMALE_LABOR_FOR',
'P_IND_AGRICULTUR',
'P_IND_MINING',
'P_IND_CONSTRUCT',
'P_IND_MANUFACT',
'P_IND_WHOLESALE',
'P_IND_RETAILTRD',
'P_IND_TRANSPORT',
'P_IND_UTILITIE',
'P_IND_INFORMAT',
'P_IND_FINALNCE',
'P_IND_REALESTAT',
'P_IND_PROFFES',
'P_IND_MANAGE',
'P_IND_ADM_SUPPOT',
'P_IND_EDALSERVIC',
'P_IND_HEALTHCARE',
'P_IND_ARTS',
'P_IND_ACCOMMOD',
'P_IND_OTHERSERV',
```

```
'P_IND_PUBADMIS',
'PCARPOOL',
'MED_TRAV_TOWRK',
'MED_HOME',
'MED_RENT',
'MED_DWELL_AGE',
'MED_LOR',
'HOUSE_STABILITY',
'CUR_EST_MED_INC',
'CENSUS_SEG1',
'EXAGE',
'ESTAGE',
'ADULT1_G',
'MARRIED',
'RELIGION',
'ETHNIC_COUNTRY',
'ADD_TYPE',
'LOR1',
'DUS',
'NUM_CHILD',
'C_00_03',
'C_04_06',
'C_07_09',
'C_10_12',
'C_13_18',
'NUMBADLT',
'IND_ED',
'CNTY_INC',
'NAT_INC',
'MAILPREF',
'ITMM',
'ITMM_SCS',
'NEWCAR',
'USEDCAR',
'AD_WEB',
'AD_MAGAZINE',
'AD_NEWSPAPER',
'AD_RADIO',
'AD_TV',
'STINCIND',
'CTINCIND',
'ESTHMVL',
'ECHVINX',
'ZGOLFERP',
'ZDONORSP',
'ZPETSP',
'ZARTSP',
'ZMOBP',
'ZFITNESP',
'ZOUTDOOP',
'ZTRAVANP',
'ZINVESTP',
'ZAUTOOWP',
'ZGARDENP',
'ZCOLLECP',
'ZCRUISEP',
'ZSPORTSP',
'ZSWEEPSP',
'ZPOLITIP',
```

```
'ZMUSICP',
'ZREADP',
'ZCHLDPRP',
'ZDIYP',
'ZSELFIPP',
'ZRELIGOP',
'ZGRANDPP',
'ZCLOTHNP',
'ZDONENVP',
'ZMUTUALP',
'ZWGHTCOP',
'ZPRCHPHP',
'ZPRCHTVP',
'ZMOBMULP',
'ZDOGSP',
'ZCATSP',
'ZHEALTHP',
'ZAUTOINP',
'ZSKIINGP',
'ZASTRLGP',
'ZBOATSP',
'ZCELLP',
'ZCOMMCOP',
'ZHMDECOP',
'ZHOMEENP',
'ZKITCHEP',
'ZMOBAVP',
'ZMOBBOOP',
'ZMOBCLTP',
'ZMOBFINP',
'ZMOBGIFP',
'ZMOBGRDP',
'ZMOBJWLP',
'ZMUSCLAP',
'ZMUSCNTP',
'ZMUSCRSP',
'ZMUSOLDP',
'ZMUSROCP',
'ZPBCAREP',
'ZPHOTOP',
'ZPRCHONP',
'ZTENNISP',
'ZTRAVDOP',
'ZTRAVFOP',
'ZVOLUNTP',
'PRE2009_SALES',
'PRE2009_TRANSACTIONS',
'TOTAL_MAIL_1',
'TOTAL_MAIL_2',
'TOTAL_MAIL_3',
'TOTAL_MAIL_4',
'TOTAL_MAIL_5',
'TOTAL_MAIL_6',
'TOTAL_MAIL_7',
'TOTAL_MAIL_8',
'TOTAL_MAIL_9',
'TOTAL_MAIL_10',
'TOTAL_MAIL_11',
'TOTAL_MAIL_12',
```

```
'TOTAL_MAIL_13',
'TOTAL_MAIL_14',
'TOTAL_MAIL_15',
'QTY',
'TOTAMT',
'QTY16',
'TOTAMT16',
'RESPONSE0',
'RESPONSE1',
'RESPONSE2',
'RESPONSE3',
'RESPONSE4',
'RESPONSE5',
'RESPONSE6',
'RESPONSE7',
'RESPONSE8',
'RESPONSE9',
'RESPONSE10',
'RESPONSE11',
'RESPONSE12',
'RESPONSE13',
'RESPONSE14',
'RESPONSE15',
'RESPONSE16'
)

customer.data.clean <- customer.data %>%
select(keep.vars)

# condense the total mail, qty, totamt, response to *.pre2009

customer.data.clean <- customer.data.clean %>%
mutate(
pre16.qty = QTY - QTY16,
pre16.totamt = TOTAMT - TOTAMT16,
high.value.account = ifelse(pre16.totamt > 500, 1, 0), # roughly 90%
pre16.response = RESPONSE0 + RESPONSE1 + RESPONSE2 + RESPONSE3 + RESPONSE4 + RESPONSE5 +
    RESPONSE6 + RESPONSE7 + RESPONSE8 + RESPONSE9 + RESPONSE10 + RESPONSE11 + RESPONSE12 +
    RESPONSE13 + RESPONSE14 + RESPONSE15,
pre16.totalmail = TOTAL_MAIL_15,
pre16.buyer.status = ifelse(pre16.totamt > 0, 1, 0),
pre16.got.mailer = ifelse(pre16.totalmail > 0, 1, 0),
pre16.bought.with.mailer = ifelse(pre16.got.mailer == 1 & pre16.totamt > 0, 1, 0), # had
    mailer and bought
pre16.bought.withnomailer = ifelse(pre16.got.mailer == 0 & pre16.totamt > 0, 1, 0), # had no
     mailer and bought
campaign16 = RESPONSE16
) %>%
select(-starts_with('RESPONSE'), -starts_with('TOTAL_MAIL'), -starts_with('TOTAMT'), -QTY, -
    QTY16)


# combine the EXAGE and ESTAGE

customer.data.clean <- customer.data.clean %>%
mutate(derived.age = ifelse(is.na(EXAGE), ESTAGE, EXAGE)) %>%
select(-EXAGE, -ESTAGE)
```

```
# the MARRIED variable has 2 bytes. the 1st byte tells us the predicted likelihood (5 is
    extremely likely, 1 is likely, 0 unknown). See the Experian data dictionary provided by
     the professor.

customer.data.clean <- customer.data.clean %>%
mutate(
predicted.marital.status = case_when(
substring(MARRIED, 1,1) == '1' ~ 'LIKELY_MARRIED',
substring(MARRIED, 1, 1) == '5' ~ 'STRONGLY_MARRIED',
TRUE ~ 'UNKNOWN'
)
) %>%
select(-MARRIED)
```

04-convert-data-type.R

```
# impute missing values before converting to appropriate types

customer.data.clean <- customer.data.clean %>%
transmute(
ACCTNO = as.character(ACCTNO),
ANY_MAIL_16,
CHANNEL_ACQUISITION = as.factor(CHANNEL_ACQUISITION),
ZIP = as.factor(ZIP),
INC_SCS_AMT_V4 = as.numeric(INC_SCS_AMT_V4),
INC_WIOUTSCS_V4 = as.factor(INC_WIOUTSCS_V4),
INC_WITHSCS_V4 = as.factor(INC_WITHSCS_V4),
INC_WOUTSCS_AMT_4 = as.numeric(INC_WOUTSCS_AMT_4),
FIPSCNTY = as.factor(FIPSCNTY),
P_HHINCOLES10M = as.numeric(P_HHINCOLES10M),
P_HHINCOM10_14 = as.numeric(P_HHINCOM10_14),
P_HHINCOM15_19 = as.numeric(P_HHINCOM15_19),
P_HHINCOM20_24 = as.numeric(P_HHINCOM20_24),
P_HHINCOM25_29 = as.numeric(P_HHINCOM25_29),
P_HHINCOM30_34 = as.numeric(P_HHINCOM30_34),
P_HHINCOM35_39 = as.numeric(P_HHINCOM35_39),
P_HHINCOM40_44 = as.numeric(P_HHINCOM40_44),
P_HHINCOM45_49 = as.numeric(P_HHINCOM45_49),
P_HHINCOM50_59 = as.numeric(P_HHINCOM50_59),
P_HHINCOM60_74 = as.numeric(P_HHINCOM60_74),
P_HHINCOM75_99 = as.numeric(P_HHINCOM75_99),
P_HHINCOM100_124 = as.numeric(P_HHINCOM100_124),
P_HHINCOM125_149 = as.numeric(P_HHINCOM125_149),
P_HHINCOM150_199 = as.numeric(P_HHINCOM150_199),
P_HHINCOM20_UP = as.numeric(P_HHINCOM20_UP),
MED_INC = as.numeric(MED_INC),
MED_FAMINCOM = as.numeric(MED_FAMINCOM),
MED_NOFAMINCOM = as.numeric(MED_NOFAMINCOM),
P_CAPITA_INCOM = as.numeric(P_CAPITA_INCOM),
FEMALE_LABOR_FOR = as.numeric(FEMALE_LABOR_FOR),
P_IND_AGRICULTUR = as.numeric(P_IND_AGRICULTUR),
P_IND_MINING = as.numeric(P_IND_MINING),
P_IND_CONSTRUCT = as.numeric(P_IND_CONSTRUCT),
P_IND_MANUFACT = as.numeric(P_IND_MANUFACT),
P_IND_WHOLESALE = as.numeric(P_IND_WHOLESALE),
P_IND_RETAILTRD = as.numeric(P_IND_RETAILTRD),
P_IND_TRANSPORT = as.numeric(P_IND_TRANSPORT),
```

```
P_IND_UTILITIE = as.numeric(P_IND_UTILITIE),
P_IND_INFORMAT = as.numeric(P_IND_INFORMAT),
P_IND_FINALNCE = as.numeric(P_IND_FINALNCE),
P_IND_REALESTAT = as.numeric(P_IND_REALESTAT),
P_IND_PROFFES = as.numeric(P_IND_PROFFES),
P_IND_MANAGE = as.numeric(P_IND_MANAGE),
P_IND_ADM_SUPPOT = as.numeric(P_IND_ADM_SUPPOT),
P_IND_EDALSERVIC = as.numeric(P_IND_EDALSERVIC),
P_IND_HEALTHCARE = as.numeric(P_IND_HEALTHCARE),
P_IND_ARTS = as.numeric(P_IND_ARTS),
P_IND_ACCOMMOD = as.numeric(P_IND_ACCOMMOD),
P_IND_OTHERSERV = as.numeric(P_IND_OTHERSERV),
P_IND_PUBADMIS = as.numeric(P_IND_PUBADMIS),
PCARPOOL = as.numeric(PCARPOOL),
MED_TRAV_TOWRK = as.numeric(MED_TRAV_TOWRK),
MED_HOME = as.numeric(MED_HOME),
MED_RENT = as.numeric(MED_RENT),
MED_DWELL_AGE = as.numeric(MED_DWELL_AGE),
MED_LOR = as.numeric(MED_LOR),
HOUSE_STABILITY = as.numeric(HOUSE_STABILITY),
CUR_EST_MED_INC = as.numeric(CUR_EST_MED_INC),
CENSUS_SEG1 = as.factor(CENSUS_SEG1),
ADULT1_G = as.factor(ADULT1_G),
RELIGION = as.factor(RELIGION),
ETHNIC_COUNTRY = as.factor(ETHNIC_COUNTRY),
ADD_TYPE = as.factor(ADD_TYPE),
LOR1 = as.numeric(LOR1),
DUS = as.factor(DUS),
NUM_CHILD = as.numeric(NUM_CHILD),
C_00_03 = as.factor(C_00_03),
C_04_06 = as.factor(C_04_06),
C_07_09 = as.factor(C_07_09),
C_10_12 = as.factor(C_10_12),
C_13_18 = as.factor(C_13_18),
NUMBADLT = as.numeric(NUMBADLT),
IND_ED = as.factor(IND_ED),
CNTY_INC = as.numeric(CNTY_INC),
NAT_INC = as.numeric(NAT_INC),
MAILPREF = as.factor(MAILPREF),
ITMM = as.factor(ITMM),
ITMM_SCS = as.numeric(ITMM_SCS),
NEWCAR = as.factor(NEWCAR),
USEDCAR = as.factor(USEDCAR),
AD_WEB = as.factor(AD_WEB),
AD_MAGAZINE = as.factor(AD_MAGAZINE),
AD_NEWSPAPER = as.factor(AD_NEWSPAPER),
AD_RADIO = as.factor(AD_RADIO),
AD_TV = as.factor(AD_TV),
STINCIND = as.numeric(STINCIND),
CTINCIND = as.numeric(CTINCIND),
ESTHMVL = as.numeric(ESTHMVL),
ECHVINX = as.numeric(ECHVINX),
ZGOLFERP = as.factor(ZGOLFERP),
ZDONORSP = as.factor(ZDONORSP),
ZPETSP = as.factor(ZPETSP),
ZARTSP = as.factor(ZARTSP),
ZMOBP = as.factor(ZMOBP),
ZFITNESP = as.factor(ZFITNESP),
ZOUTDOOP = as.factor(ZOUTDOOP),
```

```
ZTRAVANP = as.factor(ZTRAVANP),
ZINVESTP = as.factor(ZINVESTP),
ZAUTOOWP = as.factor(ZAUTOOWP),
ZGARDENP = as.factor(ZGARDENP),
ZCOLLECP = as.factor(ZCOLLECP),
ZCRUISEP = as.factor(ZCRUISEP),
ZSPORTSP = as.factor(ZSPORTSP),
ZSWEEPSP = as.factor(ZSWEEPSP),
ZPOLITIP = as.factor(ZPOLITIP),
ZMUSICP = as.factor(ZMUSICP),
ZREADP = as.factor(ZREADP),
ZCHLDPRP = as.factor(ZCHLDPRP),
ZDIYP = as.factor(ZDIYP),
ZSELFIPP = as.factor(ZSELFIPP),
ZRELIGOP = as.factor(ZRELIGOP),
ZGRANDPP = as.factor(ZGRANDPP),
ZCLOTHNP = as.factor(ZCLOTHNP),
ZDONENVP = as.factor(ZDONENVP),
ZMUTUALP = as.factor(ZMUTUALP),
ZWGHTCOP = as.factor(ZWGHTCOP),
ZPRCHPHP = as.factor(ZPRCHPHP),
ZPRCHTVP = as.factor(ZPRCHTVP),
ZMOBMULP = as.factor(ZMOBMULP),
ZDOGSP = as.factor(ZDOGSP),
ZCATSP = as.factor(ZCATSP),
ZHEALTHP = as.factor(ZHEALTHP),
ZAUTOINP = as.factor(ZAUTOINP),
ZSKIINGP = as.factor(ZSKIINGP),
ZASTRLGP = as.factor(ZASTRLGP),
ZBOATSP = as.factor(ZBOATSP),
ZCELLP = as.factor(ZCELLP),
ZCOMMCOP = as.factor(ZCOMMCOP),
ZHMDECOP = as.factor(ZHMDECOP),
ZHOMEENP = as.factor(ZHOMEENP),
ZKITCHEP = as.factor(ZKITCHEP),
ZMOBAVP = as.factor(ZMOBAVP),
ZMOBBOOP = as.factor(ZMOBBOOP),
ZMOBCLTP = as.factor(ZMOBCLTP),
ZMOBFINP = as.factor(ZMOBFINP),
ZMOBGIFP = as.factor(ZMOBGIFP),
ZMOBGRDP = as.factor(ZMOBGRDP),
ZMOBJWLP = as.factor(ZMOBJWLP),
ZMUSCLAP = as.factor(ZMUSCLAP),
ZMUSCNTP = as.factor(ZMUSCNTP),
ZMUSCRSP = as.factor(ZMUSCRSP),
ZMUSOLDP = as.factor(ZMUSOLDP),
ZMUSROCP = as.factor(ZMUSROCP),
ZPBCAREP = as.factor(ZPBCAREP),
ZPHOTOP = as.factor(ZPHOTOP),
ZPRCHONP = as.factor(ZPRCHONP),
ZTENNISP = as.factor(ZTENNISP),
ZTRAVDOP = as.factor(ZTRAVDOP),
ZTRAVFOP = as.factor(ZTRAVFOP),
ZVOLUNTP = as.factor(ZVOLUNTP),
PRE2009_SALES = as.numeric(PRE2009_SALES),
PRE2009_TRANSACTIONS = as.numeric(PRE2009_TRANSACTIONS),

pre16.qty = as.numeric(pre16.qty),
pre16.totamt = as.numeric(pre16.totamt),
```

```
high.value.account = as.factor(high.value.account),
pre16.response = as.numeric(pre16.response),
pre16.totalmail = as.numeric(pre16.totalmail),
pre16.buyer.status = as.factor(pre16.buyer.status),
pre16.got.mailer = as.factor(pre16.got.mailer),
pre16.bought.with.mailer = as.factor(pre16.bought.with.mailer),
pre16.bought.withnomailer = as.factor(pre16.bought.withnomailer),

campaign16 = as.factor(campaign16),
derived.age = as.numeric(derived.age),
predicted.marital.status = as.factor(predicted.marital.status)
)
```

05-eda.R

```
# Nikhil Agarwal

# THIS FILE IS USED FOR EDA


# check for missing values ----

customer.data %>%
map_df(function(x) sum(is.na(x))) %>%
gather(key = 'variable', value = 'missing.count') %>%
arrange(desc(missing.count)) %>%
mutate(
missing.pct = round(missing.count / nrow(customer.data),3)
) %>%
filter(missing.pct > 0.49) %>%
View

customer.data %>%
map_df(function(x) sum(is.na(x))) %>%
gather(key = 'variable', value = 'missing.count') %>%
arrange(desc(missing.count)) %>%
mutate(
missing.pct = round(missing.count / nrow(customer.data),3)
) %>%
filter(missing.count > 0) %>%
View


customer.data.clean %>%
map_df(function(x) sum(is.na(x))) %>%
gather(key = 'variable', value = 'missing.count') %>%
# arrange(desc(missing.count)) %>%
mutate(
missing.pct = round(missing.count / nrow(customer.data.clean),3)
) %>%
# filter(missing.pct > 0.49) %>%
View

## quite a few missing values


customer.data %>%
```

```
filter(ANY_MAIL_16 > 0) %>%
select(RESPONSE16) %>%
group_by(RESPONSE16) %>%
summarize(
cnt = n()
)



1# determine the number of numeric & character columns ----

customer.data %>%
select_if(is.numeric) %>%
summarize(
numeric.fields = length(colnames(.))
)

customer.data %>%
select_if(is.character) %>%
summarize(
character.fields = length(colnames(.))
)

# both add up to 554, so we know we only have two types of fields thus far: numeric &
    character

# do a quick glimpse ----

glimpse(customer.data)


# Verify that account number column (ACCTNO) is unique ----

length(unique(customer.data$ACCTNO))

# ACCTNO has all unique values (30,779) w/ no repeating values

# random data exploration ----

customer.data %>%
select(ACCTNO, starts_with('TOTAMT'), starts_with('ANY_MAIL')) %>%
transmute(
pre16.totamt = TOTAMT - TOTAMT16
) %>%
summarize_all(funs(quantile), probs = 0.9)
head


# summarized table values ----


getNumericSummary <- function(df){
df <- df %>%
select_if(is.numeric)


mean.values <- df %>%
summarize_all(mean, na.rm = T) %>%
gather(key = variable, value = avg.value) %>%
```

```
mutate(avg.value = round(avg.value, 3))
min.values <- df %>%
summarise_all(min, na.rm = T) %>%
gather(key = variable, value = minimum.value)
max.values <- df %>%
summarize_all(max, na.rm = T) %>%
gather(key = variable, value = maximum.value)
q05.values <- df %>%
summarize_all(funs(quantile), probs = 0.05, na.rm = T) %>%
gather(key = variable, value = p05.value)
q10.values <- df %>%
summarize_all(funs(quantile), probs = 0.10, na.rm = T) %>%
gather(key = variable, value = p10.value)
q25.values <- df %>%
summarize_all(funs(quantile), probs = 0.25, na.rm = T) %>%
gather(key = variable, value = p25.value)
median.values <- df %>%
summarize_all(median, na.rm = T) %>%
gather(key = variable, value = median.value)
q75.values <- df %>%
summarize_all(funs(quantile), probs = 0.75, na.rm = T) %>%
gather(key = variable, value = p75.value)
q90.values <- df %>%
summarize_all(funs(quantile), probs = 0.90, na.rm = T) %>%
gather(key = variable, value = p90.value)
q95.values <- df %>%
summarize_all(funs(quantile), probs = 0.95, na.rm = T) %>%
gather(key = variable, value = p95.value)
missing.values <- df %>%
summarize_all(funs(sum(is.na(.)))) %>%
gather(key = variable, value = missing)

sd.values <- df %>%
summarize_all(sd, na.rm = T) %>%
gather(key = variable, value = sd) %>%
mutate(sd = round(sd,3))
var.values <- df %>%
summarize_all(var, na.rm = T) %>%
gather(key = variable, value = var) %>%
mutate(var = round(var,3))

descriptive.stats <- min.values %>%
inner_join(median.values, by = 'variable') %>%
inner_join(mean.values, by = 'variable') %>%
inner_join(max.values, by = 'variable') %>%
inner_join(q05.values, by = 'variable') %>%
inner_join(q10.values, by = 'variable') %>%
inner_join(q25.values, by = 'variable') %>%
inner_join(q75.values, by = 'variable') %>%
inner_join(q90.values, by = 'variable') %>%
inner_join(q95.values, by = 'variable') %>%
inner_join(sd.values, by = 'variable') %>%
inner_join(var.values, by = 'variable') %>%
inner_join(missing.values, by = 'variable')

return(descriptive.stats)
}

numericValues.stats <- getNumericSummary(customer.data.clean)
```

```
characterValue.uniqueCount <- customer.data.clean %>%
select_if(is.factor) %>%
summarize_all(funs(length(unique(.)))) %>%
gather(key = variable, value = unique.value.count)

characterValue.missingSummary <- customer.data.clean %>%
select_if(is.factor) %>%
map_df(function(x) sum(is.na(x))) %>%
gather(key = variable, value = missing.cnt) %>%
mutate(
missing.pct = round(missing.cnt / nrow(customer.data.clean),3)
)

characterValue.Summary <- inner_join(characterValue.uniqueCount, characterValue.
    missingSummary, by = c('variable' = 'variable'))


# correlation matrix

cor.matrix <- as.data.frame(cor(x = customer.data %>% select_if(is.numeric) %>% select(-
    TOTAMT16), y = customer.data$TOTAMT16, method = 'pearson'))

cor.matrix <- rownames_to_column(cor.matrix)

cor.matrix %>%
rename(
variable = rowname,
correlation = V1
) %>%
mutate(
correlation = round(correlation, 3)
) %>%
arrange(desc(correlation)) %>%
head(n = 10)









# graphs ----

customer.data.clean %>%
ggplot(aes(x = ZARTSP)) +
geom_bar(stat = 'count') +
ggtitle('Frequency of ZARTSP')


customer.data.clean %>%
ggplot(aes(x = ZARTSP, fill = campaign16)) +
geom_bar(stat = 'count', position = 'dodge') +
scale_fill_discrete(name = 'Response16') +
ggtitle('Frequency of ZARTSP by Response16')
```

```
library(gridExtra)
# factor graphs ----
getFactorGraphs <- function(df, var) {
var <- enquo(var)
a <- df %>%
ggplot(aes_string(x = rlang::quo_text(var))) +
geom_bar(stat = 'count') +
ggtitle(paste('Frequency of ', rlang::quo_text(var), sep = ''))

b <- df %>%
ggplot(aes_string(x = rlang::quo_text(var), fill = 'campaign16')) +
geom_bar(stat = 'count', position = 'dodge') +
scale_fill_discrete(name = 'Response16') +
ggtitle(paste('Frequency of ', rlang::quo_text(var), ' by Response16', sep = ''))

grid.arrange(a, b, nrow = 2)
}

getFactorGraphs(customer.data.clean, CHANNEL_ACQUISITION)
getFactorGraphs(customer.data.clean, ZIP)
getFactorGraphs(customer.data.clean, INC_WIOUTSCS_V4)
getFactorGraphs(customer.data.clean, INC_WITHSCS_V4)
getFactorGraphs(customer.data.clean, FIPSSTCD)
getFactorGraphs(customer.data.clean, FIPSCNTY)
getFactorGraphs(customer.data.clean, CENSUS_SEG1)
getFactorGraphs(customer.data.clean, ADULT1_G)
getFactorGraphs(customer.data.clean, RELIGION)
getFactorGraphs(customer.data.clean, ETHNIC_COUNTRY)
getFactorGraphs(customer.data.clean, ADD_TYPE)
getFactorGraphs(customer.data.clean, DUS)
getFactorGraphs(customer.data.clean, C_00_03)
getFactorGraphs(customer.data.clean, C_04_06)
getFactorGraphs(customer.data.clean, C_07_09)
getFactorGraphs(customer.data.clean, C_10_12)
getFactorGraphs(customer.data.clean, C_13_18)
getFactorGraphs(customer.data.clean, IND_ED)
getFactorGraphs(customer.data.clean, MAILPREF)
getFactorGraphs(customer.data.clean, ITMM)
getFactorGraphs(customer.data.clean, NEWCAR)
getFactorGraphs(customer.data.clean, USEDCAR)
getFactorGraphs(customer.data.clean, AD_WEB)
getFactorGraphs(customer.data.clean, AD_MAGAZINE)
getFactorGraphs(customer.data.clean, AD_NEWSPAPER)
getFactorGraphs(customer.data.clean, AD_RADIO)
getFactorGraphs(customer.data.clean, AD_TV)
getFactorGraphs(customer.data.clean, ZGOLFERP)
getFactorGraphs(customer.data.clean, ZDONORSP)
getFactorGraphs(customer.data.clean, ZPETSP)
getFactorGraphs(customer.data.clean, ZARTSP)
getFactorGraphs(customer.data.clean, ZMOBP)
getFactorGraphs(customer.data.clean, ZFITNESP)
getFactorGraphs(customer.data.clean, ZOUTDOOP)
getFactorGraphs(customer.data.clean, ZTRAVANP)
getFactorGraphs(customer.data.clean, ZINVESTP)
getFactorGraphs(customer.data.clean, ZAUTOOWP)
getFactorGraphs(customer.data.clean, ZGARDENP)
getFactorGraphs(customer.data.clean, ZCOLLECP)
getFactorGraphs(customer.data.clean, ZCRUISEP)
getFactorGraphs(customer.data.clean, ZSPORTSP)
```

```
getFactorGraphs(customer.data.clean, ZSWEEPSP)
getFactorGraphs(customer.data.clean, ZPOLITIP)
getFactorGraphs(customer.data.clean, ZMUSICP)
getFactorGraphs(customer.data.clean, ZREADP)
getFactorGraphs(customer.data.clean, ZCHLDPRP)
getFactorGraphs(customer.data.clean, ZDIYP)
getFactorGraphs(customer.data.clean, ZSELFIPP)
getFactorGraphs(customer.data.clean, ZRELIGOP)
getFactorGraphs(customer.data.clean, ZGRANDPP)
getFactorGraphs(customer.data.clean, ZCLOTHNP)
getFactorGraphs(customer.data.clean, ZDONENVP)
getFactorGraphs(customer.data.clean, ZMUTUALP)
getFactorGraphs(customer.data.clean, ZWGHTCOP)
getFactorGraphs(customer.data.clean, ZPRCHPHP)
getFactorGraphs(customer.data.clean, ZPRCHTVP)
getFactorGraphs(customer.data.clean, ZMOBMULP)
getFactorGraphs(customer.data.clean, ZDOGSP)
getFactorGraphs(customer.data.clean, ZCATSP)
getFactorGraphs(customer.data.clean, ZHEALTHP)
getFactorGraphs(customer.data.clean, ZAUTOINP)
getFactorGraphs(customer.data.clean, ZSKIINGP)
getFactorGraphs(customer.data.clean, ZASTRLGP)
getFactorGraphs(customer.data.clean, ZBOATSP)
getFactorGraphs(customer.data.clean, ZCELLP)
getFactorGraphs(customer.data.clean, ZCOMMCOP)
getFactorGraphs(customer.data.clean, ZHMDECOP)
getFactorGraphs(customer.data.clean, ZHOMEENP)
getFactorGraphs(customer.data.clean, ZKITCHEP)
getFactorGraphs(customer.data.clean, ZMOBAVP)
getFactorGraphs(customer.data.clean, ZMOBBOOP)
getFactorGraphs(customer.data.clean, ZMOBCLTP)
getFactorGraphs(customer.data.clean, ZMOBFINP)
getFactorGraphs(customer.data.clean, ZMOBGIFP)
getFactorGraphs(customer.data.clean, ZMOBGRDP)
getFactorGraphs(customer.data.clean, ZMOBJWLP)
getFactorGraphs(customer.data.clean, ZMUSCLAP)
getFactorGraphs(customer.data.clean, ZMUSCNTP)
getFactorGraphs(customer.data.clean, ZMUSCRSP)
getFactorGraphs(customer.data.clean, ZMUSOLDP)
getFactorGraphs(customer.data.clean, ZMUSROCP)
getFactorGraphs(customer.data.clean, ZPBCAREP)
getFactorGraphs(customer.data.clean, ZPHOTOP)
getFactorGraphs(customer.data.clean, ZPRCHONP)
getFactorGraphs(customer.data.clean, ZTENNISP)
getFactorGraphs(customer.data.clean, ZTRAVDOP)
getFactorGraphs(customer.data.clean, ZTRAVFOP)
getFactorGraphs(customer.data.clean, ZVOLUNTP)
getFactorGraphs(customer.data.clean, campaign16)
getFactorGraphs(customer.data.clean, predicted.marital.status)
getFactorGraphs(customer.data.clean, pre16.buyer.status)


# numeric graphcs

customer.data.clean %>%
select(derived.age) %>%
na.omit() %>%
ggplot(aes(x = derived.age)) +
geom_histogram(bins = 30)
```

```
customer.data.clean %>%
select(derived.age, campaign16) %>%
na.omit() %>%
ggplot(aes(x = derived.age, fill = campaign16)) +
geom_histogram(bins = 30)


customer.data.clean %>%
select(derived.age) %>%
na.omit() %>%
ggplot(aes(x = '', y = derived.age)) +
geom_boxplot()

customer.data.clean %>%
select(derived.age, campaign16) %>%
na.omit() %>%
ggplot(aes(x = campaign16, y = derived.age, color = campaign16)) +
geom_boxplot()


#https://stackoverflow.com/questions/28777626/how-do-i-combine-aes-and-aes-string-options
`+.uneval` <- function(a,b) {
`class<-`(modifyList(a,b), "uneval")
}

getNumericGraphs <- function(df, var) {
var <- enquo(var)

a <- df %>%
select(!!var) %>%
na.omit() %>%
ggplot(aes_string(x = rlang::quo_text(var))) +
geom_histogram(bins = 30)

b <- df %>%
select(!!var, campaign16) %>%
na.omit() %>%
ggplot(aes_string(x = rlang::quo_text(var), fill = 'campaign16')) +
geom_histogram(bins = 30) +
scale_fill_discrete(name = 'Response16')

c <- df %>%
select(!!var) %>%
na.omit() %>%
ggplot(aes_string(y = rlang::quo_text(var)) + aes(x = '')) +
geom_boxplot() +
labs(x = '')

d <- df %>%
select(!!var, campaign16) %>%
na.omit() %>%
ggplot(aes(x = campaign16) + aes_string(y = rlang::quo_text(var)) + aes(color = campaign16))
     +
geom_boxplot() +
labs(x = 'Response16') +
scale_fill_discrete('Response16')

grid.arrange(a,b,c,d, ncol = 2, nrow = 2)
```

```
}

getNumericGraphs(customer.data.clean, INC_SCS_AMT_V4)
getNumericGraphs(customer.data.clean, INC_WOUTSCS_AMT_4)
getNumericGraphs(customer.data.clean, P_HHINCOLES10M)
getNumericGraphs(customer.data.clean, P_HHINCOM10_14)
getNumericGraphs(customer.data.clean, P_HHINCOM15_19)
getNumericGraphs(customer.data.clean, P_HHINCOM20_24)
getNumericGraphs(customer.data.clean, P_HHINCOM25_29)
getNumericGraphs(customer.data.clean, P_HHINCOM30_34)
getNumericGraphs(customer.data.clean, P_HHINCOM35_39)
getNumericGraphs(customer.data.clean, P_HHINCOM40_44)
getNumericGraphs(customer.data.clean, P_HHINCOM45_49)
getNumericGraphs(customer.data.clean, P_HHINCOM50_59)
getNumericGraphs(customer.data.clean, P_HHINCOM60_74)
getNumericGraphs(customer.data.clean, P_HHINCOM75_99)
getNumericGraphs(customer.data.clean, P_HHINCOM100_124)
getNumericGraphs(customer.data.clean, P_HHINCOM125_149)
getNumericGraphs(customer.data.clean, P_HHINCOM150_199)
getNumericGraphs(customer.data.clean, P_HHINCOM20_UP)
getNumericGraphs(customer.data.clean, MED_INC)
getNumericGraphs(customer.data.clean, MED_FAMINCOM)
getNumericGraphs(customer.data.clean, MED_NOFAMINCOM)
getNumericGraphs(customer.data.clean, P_CAPITA_INCOM)
getNumericGraphs(customer.data.clean, FEMALE_LABOR_FOR)
getNumericGraphs(customer.data.clean, P_IND_AGRICULTUR)
getNumericGraphs(customer.data.clean, P_IND_MINING)
getNumericGraphs(customer.data.clean, P_IND_CONSTRUCT)
getNumericGraphs(customer.data.clean, P_IND_MANUFACT)
getNumericGraphs(customer.data.clean, P_IND_WHOLESALE)
getNumericGraphs(customer.data.clean, P_IND_RETAILTRD)
getNumericGraphs(customer.data.clean, P_IND_TRANSPORT)
getNumericGraphs(customer.data.clean, P_IND_UTILITIE)
getNumericGraphs(customer.data.clean, P_IND_INFORMAT)
getNumericGraphs(customer.data.clean, P_IND_FINALNCE)
getNumericGraphs(customer.data.clean, P_IND_REALESTAT)
getNumericGraphs(customer.data.clean, P_IND_PROFFES)
getNumericGraphs(customer.data.clean, P_IND_MANAGE)
getNumericGraphs(customer.data.clean, P_IND_ADM_SUPPOT)
getNumericGraphs(customer.data.clean, P_IND_EDALSERVIC)
getNumericGraphs(customer.data.clean, P_IND_HEALTHCARE)
getNumericGraphs(customer.data.clean, P_IND_ARTS)
getNumericGraphs(customer.data.clean, P_IND_ACCOMMOD)
getNumericGraphs(customer.data.clean, P_IND_OTHERSERV)
getNumericGraphs(customer.data.clean, P_IND_PUBADMIS)
getNumericGraphs(customer.data.clean, PCARPOOL)
getNumericGraphs(customer.data.clean, MED_TRAV_TOWRK)
getNumericGraphs(customer.data.clean, MED_HOME)
getNumericGraphs(customer.data.clean, MED_RENT)
getNumericGraphs(customer.data.clean, MED_DWELL_AGE)
getNumericGraphs(customer.data.clean, MED_LOR)
getNumericGraphs(customer.data.clean, HOUSE_STABILITY)
getNumericGraphs(customer.data.clean, CUR_EST_MED_INC)
getNumericGraphs(customer.data.clean, LOR1)
getNumericGraphs(customer.data.clean, NUM_CHILD)
getNumericGraphs(customer.data.clean, NUMBADLT)
getNumericGraphs(customer.data.clean, CNTY_INC)
getNumericGraphs(customer.data.clean, NAT_INC)
getNumericGraphs(customer.data.clean, ITMM_SCS)
```

```
getNumericGraphs(customer.data.clean, STINCIND)
getNumericGraphs(customer.data.clean, CTINCIND)
getNumericGraphs(customer.data.clean, ESTHMVL)
getNumericGraphs(customer.data.clean, ECHVINX)
getNumericGraphs(customer.data.clean, PRE2009_SALES)
getNumericGraphs(customer.data.clean, PRE2009_TRANSACTIONS)
getNumericGraphs(customer.data.clean, pre2009.qty)
getNumericGraphs(customer.data.clean, pre2009.totamt)
getNumericGraphs(customer.data.clean, pre2009.response)
getNumericGraphs(customer.data.clean, pre2009.totalmail)
getNumericGraphs(customer.data.clean, derived.age)
getNumericGraphs(customer.data.clean, pre16.totamt)
```

06-impute.R

```
# need to impute the missing values in each one

customer.data.clean <- customer.data.clean %>%
transmute(
ACCTNO,
ANY_MAIL_16,
CHANNEL_ACQUISITION = fct_explicit_na(CHANNEL_ACQUISITION, na_level = 'Unknown'),
ZIP = fct_explicit_na(ZIP, na_level = 'Unknown'),
INC_SCS_AMT_V4,
INC_WIOUTSCS_V4 = fct_explicit_na(INC_WIOUTSCS_V4, na_level = 'Unknown'),
INC_WITHSCS_V4 = fct_explicit_na(INC_WITHSCS_V4, na_level = 'Unknown'),
INC_WOUTSCS_AMT_4,
FIPSCNTY = fct_explicit_na(FIPSCNTY, na_level = 'Unknown'),
P_HHINCOLES10M,
P_HHINCOM10_14,
P_HHINCOM15_19,
P_HHINCOM20_24,
P_HHINCOM25_29,
P_HHINCOM30_34,
P_HHINCOM35_39,
P_HHINCOM40_44,
P_HHINCOM45_49,
P_HHINCOM50_59,
P_HHINCOM60_74,
P_HHINCOM75_99,
P_HHINCOM100_124,
P_HHINCOM125_149,
P_HHINCOM150_199,
P_HHINCOM20_UP,
MED_INC,
MED_FAMINCOM,
MED_NOFAMINCOM,
P_CAPITA_INCOM,
FEMALE_LABOR_FOR,
P_IND_AGRICULTUR,
P_IND_MINING,
P_IND_CONSTRUCT,
P_IND_MANUFACT,
P_IND_WHOLESALE,
P_IND_RETAILTRD,
P_IND_TRANSPORT,
P_IND_UTILITIE,
```

```
P_IND_INFORMAT,
P_IND_FINALNCE,
P_IND_REALESTAT,
P_IND_PROFFES,
P_IND_MANAGE,
P_IND_ADM_SUPPOT,
P_IND_EDALSERVIC,
P_IND_HEALTHCARE,
P_IND_ARTS,
P_IND_ACCOMMOD,
P_IND_OTHERSERV,
P_IND_PUBADMIS,
PCARPOOL,
MED_TRAV_TOWRK,
MED_HOME,
MED_RENT,
MED_DWELL_AGE,
MED_LOR,
HOUSE_STABILITY,
CUR_EST_MED_INC,
CENSUS_SEG1 = fct_explicit_na(CENSUS_SEG1, na_level = 'Unknown'),
ADULT1_G = fct_explicit_na(ADULT1_G, na_level = 'Unknown'),
RELIGION = fct_explicit_na(RELIGION, na_level = 'Unknown'),
ETHNIC_COUNTRY = fct_explicit_na(ETHNIC_COUNTRY, na_level = 'Unknown'),
ADD_TYPE = fct_explicit_na(ADD_TYPE, na_level = 'Unknown'),
LOR1,
DUS = fct_explicit_na(DUS, na_level = 'Unknown'),
NUM_CHILD = median(NUM_CHILD, na.rm = T),
C_00_03 = fct_explicit_na(C_00_03, na_level = 'Unknown'),
C_04_06 = fct_explicit_na(C_04_06, na_level = 'Unknown'),
C_07_09 = fct_explicit_na(C_07_09, na_level = 'Unknown'),
C_10_12 = fct_explicit_na(C_10_12, na_level = 'Unknown'),
C_13_18 = fct_explicit_na(C_13_18, na_level = 'Unknown'),
NUMBADLT,
IND_ED = fct_explicit_na(IND_ED, na_level = 'Unknown'),
CNTY_INC,
NAT_INC,
MAILPREF = fct_explicit_na(MAILPREF, na_level = 'Unknown'),
ITMM = fct_explicit_na(ITMM, na_level = 'Unknown'),
ITMM_SCS,
NEWCAR = fct_explicit_na(NEWCAR, na_level = 'Unknown'),
USEDCAR = fct_explicit_na(USEDCAR, na_level = 'Unknown'),
AD_WEB = fct_explicit_na(AD_WEB, na_level = 'Unknown'),
AD_MAGAZINE = fct_explicit_na(AD_MAGAZINE, na_level = 'Unknown'),
AD_NEWSPAPER = fct_explicit_na(AD_NEWSPAPER, na_level = 'Unknown'),
AD_RADIO = fct_explicit_na(AD_RADIO, na_level = 'Unknown'),
AD_TV = fct_explicit_na(AD_TV, na_level = 'Unknown'),
STINCIND,
CTINCIND,
ESTHMVL = median(ESTHMVL, na.rm = T),
ECHVINX = median(ECHVINX, na.rm = T),
ZGOLFERP = fct_explicit_na(ZGOLFERP, na_level = 'Unknown'),
ZDONORSP = fct_explicit_na(ZDONORSP, na_level = 'Unknown'),
ZPETSP = fct_explicit_na(ZPETSP, na_level = 'Unknown'),
ZARTSP = fct_explicit_na(ZARTSP, na_level = 'Unknown'),
ZMOBP = fct_explicit_na(ZMOBP, na_level = 'Unknown'),
ZFITNESP = fct_explicit_na(ZFITNESP, na_level = 'Unknown'),
ZOUTDOOP = fct_explicit_na(ZOUTDOOP, na_level = 'Unknown'),
ZTRAVANP = fct_explicit_na(ZTRAVANP, na_level = 'Unknown'),
```

```
ZINVESTP = fct_explicit_na(ZINVESTP, na_level = 'Unknown'),
ZAUTOOWP = fct_explicit_na(ZAUTOOWP, na_level = 'Unknown'),
ZGARDENP = fct_explicit_na(ZGARDENP, na_level = 'Unknown'),
ZCOLLECP = fct_explicit_na(ZCOLLECP, na_level = 'Unknown'),
ZCRUISEP = fct_explicit_na(ZCRUISEP, na_level = 'Unknown'),
ZSPORTSP = fct_explicit_na(ZSPORTSP, na_level = 'Unknown'),
ZSWEEPSP = fct_explicit_na(ZSWEEPSP, na_level = 'Unknown'),
ZPOLITIP = fct_explicit_na(ZPOLITIP, na_level = 'Unknown'),
ZMUSICP = fct_explicit_na(ZMUSICP, na_level = 'Unknown'),
ZREADP = fct_explicit_na(ZREADP, na_level = 'Unknown'),
ZCHLDPRP = fct_explicit_na(ZCHLDPRP, na_level = 'Unknown'),
ZDIYP = fct_explicit_na(ZDIYP, na_level = 'Unknown'),
ZSELFIPP = fct_explicit_na(ZSELFIPP, na_level = 'Unknown'),
ZRELIGOP = fct_explicit_na(ZRELIGOP, na_level = 'Unknown'),
ZGRANDPP = fct_explicit_na(ZGRANDPP, na_level = 'Unknown'),
ZCLOTHNP = fct_explicit_na(ZCLOTHNP, na_level = 'Unknown'),
ZDONENVP = fct_explicit_na(ZDONENVP, na_level = 'Unknown'),
ZMUTUALP = fct_explicit_na(ZMUTUALP, na_level = 'Unknown'),
ZWGHTCOP = fct_explicit_na(ZWGHTCOP, na_level = 'Unknown'),
ZPRCHPHP = fct_explicit_na(ZPRCHPHP, na_level = 'Unknown'),
ZPRCHTVP = fct_explicit_na(ZPRCHTVP, na_level = 'Unknown'),
ZMOBMULP = fct_explicit_na(ZMOBMULP, na_level = 'Unknown'),
ZDOGSP = fct_explicit_na(ZDOGSP, na_level = 'Unknown'),
ZCATSP = fct_explicit_na(ZCATSP, na_level = 'Unknown'),
ZHEALTHP = fct_explicit_na(ZHEALTHP, na_level = 'Unknown'),
ZAUTOINP = fct_explicit_na(ZAUTOINP, na_level = 'Unknown'),
ZSKIINGP = fct_explicit_na(ZSKIINGP, na_level = 'Unknown'),
ZASTRLGP = fct_explicit_na(ZASTRLGP, na_level = 'Unknown'),
ZBOATSP = fct_explicit_na(ZBOATSP, na_level = 'Unknown'),
ZCELLP = fct_explicit_na(ZCELLP, na_level = 'Unknown'),
ZCOMMCOP = fct_explicit_na(ZCOMMCOP, na_level = 'Unknown'),
ZHMDECOP = fct_explicit_na(ZHMDECOP, na_level = 'Unknown'),
ZHOMEENP = fct_explicit_na(ZHOMEENP, na_level = 'Unknown'),
ZKITCHEP = fct_explicit_na(ZKITCHEP, na_level = 'Unknown'),
ZMOBAVP = fct_explicit_na(ZMOBAVP, na_level = 'Unknown'),
ZMOBBOOP = fct_explicit_na(ZMOBBOOP, na_level = 'Unknown'),
ZMOBCLTP = fct_explicit_na(ZMOBCLTP, na_level = 'Unknown'),
ZMOBFINP = fct_explicit_na(ZMOBFINP, na_level = 'Unknown'),
ZMOBGIFP = fct_explicit_na(ZMOBGIFP, na_level = 'Unknown'),
ZMOBGRDP = fct_explicit_na(ZMOBGRDP, na_level = 'Unknown'),
ZMOBJWLP = fct_explicit_na(ZMOBJWLP, na_level = 'Unknown'),
ZMUSCLAP = fct_explicit_na(ZMUSCLAP, na_level = 'Unknown'),
ZMUSCNTP = fct_explicit_na(ZMUSCNTP, na_level = 'Unknown'),
ZMUSCRSP = fct_explicit_na(ZMUSCRSP, na_level = 'Unknown'),
ZMUSOLDP = fct_explicit_na(ZMUSOLDP, na_level = 'Unknown'),
ZMUSROCP = fct_explicit_na(ZMUSROCP, na_level = 'Unknown'),
ZPBCAREP = fct_explicit_na(ZPBCAREP, na_level = 'Unknown'),
ZPHOTOP = fct_explicit_na(ZPHOTOP, na_level = 'Unknown'),
ZPRCHONP = fct_explicit_na(ZPRCHONP, na_level = 'Unknown'),
ZTENNISP = fct_explicit_na(ZTENNISP, na_level = 'Unknown'),
ZTRAVDOP = fct_explicit_na(ZTRAVDOP, na_level = 'Unknown'),
ZTRAVFOP = fct_explicit_na(ZTRAVFOP, na_level = 'Unknown'),
ZVOLUNTP = fct_explicit_na(ZVOLUNTP, na_level = 'Unknown'),
PRE2009_SALES,
PRE2009_TRANSACTIONS,
pre16.qty,
pre16.totamt,
high.value.account,
pre16.response,
```

```
pre16.totalmail,
pre16.buyer.status,
pre16.got.mailer,
pre16.bought.with.mailer,
pre16.bought.withnomailer,
campaign16 = fct_explicit_na(campaign16, na_level = 'Unknown'),
derived.age = median(derived.age, na.rm = T),
predicted.marital.status = fct_explicit_na(predicted.marital.status, na_level = 'Unknown')
)
```

07-data-split.R

```
# create 2 distinct datasets. One that will be used to model campaign16 (ANY_MAIL_16 = 1).
    This is why high accuracy is super important, we want as minimal false positives. The 2
    nd dataset (referring to campaign17) will use ANY_MAIL_16 = 0. The model constructed
    will then be applied to the campaign17 dataset.
```

```
# used for training & validation
campaign16.data <- customer.data.clean %>%
filter(ANY_MAIL_16 > 0) %>%
select(-ANY_MAIL_16)
```

```
# used for test dataset
campaign17.data <- customer.data.clean %>%
filter(ANY_MAIL_16 == 0) %>%
select(-ANY_MAIL_16)
```

```
# split the data

set.seed(360)
trainIndex <- createDataPartition(campaign16.data$campaign16, p = 0.80, list = F, times = 1)

customer.train.initial <- campaign16.data[trainIndex,]
customer.valid.initial <- campaign16.data[-trainIndex,]
```

```
# after running the initial random forest model, only the top 25 variables are kept

customer.train <- customer.train.initial %>%
select(
ACCTNO,
PRE2009_SALES,
PRE2009_TRANSACTIONS,
pre16.totamt,
pre16.qty,
ITMM_SCS,
pre16.totalmail,
LOR1,
pre16.response,
INC_WOUTSCS_AMT_4,
```

```
CTINCIND,
STINCIND,
INC_SCS_AMT_V4,
CNTY_INC,
IND_ED,
NAT_INC,
FEMALE_LABOR_FOR,
P_HHINCOM100_124,
AD_NEWSPAPER,
MED_RENT,
MED_NOFAMINCOM,
P_IND_EDALSERVIC,
MED_LOR,
ZPRCHPHP,
P_IND_RETAILTRD,
NUMBADLT,
campaign16
)

customer.valid <- customer.valid.initial %>%
select(
ACCTNO,
PRE2009_SALES,
PRE2009_TRANSACTIONS,
pre16.totamt,
pre16.qty,
ITMM_SCS,
pre16.totalmail,
LOR1,
pre16.response,
INC_WOUTSCS_AMT_4,
CTINCIND,
STINCIND,
INC_SCS_AMT_V4,
CNTY_INC,
IND_ED,
NAT_INC,
FEMALE_LABOR_FOR,
P_HHINCOM100_124,
AD_NEWSPAPER,
MED_RENT,
MED_NOFAMINCOM,
P_IND_EDALSERVIC,
MED_LOR,
ZPRCHPHP,
P_IND_RETAILTRD,
NUMBADLT,
campaign16
)
```

08a-ranger.R

```
# ranger model

# ranger ----
library(ranger)
```

```
# create ranger model for classification
ranger.model1.class <- ranger(campaign16 ~ .,
data = customer.train %>% select(-ACCTNO),
num.trees = 1000,
mtry = 5,
classification = T,
importance = 'impurity',
seed = 2018
)


# create ranger model for probability
ranger.model1.prob <- ranger(campaign16 ~ .,
data = customer.train %>% select(-ACCTNO),
num.trees = 1000,
mtry = 5,
classification = T,
importance = 'impurity',
seed = 2018,
probability = T
)


ranger.model1.class$confusion.matrix
ranger.model1.class$variable.importance

enframe(ranger.model1.class$variable.importance) %>%
rename(
variable = name,
importance = value
) %>%
arrange(desc(importance)) %>%
top_n(25) %>%
ggplot(aes(x = reorder(variable, importance), y = importance)) +
geom_bar(stat = 'identity') +
coord_flip() +
ggtitle('Variable Importance - Random Forest') +
labs(y = 'Importance', x = 'Variable')

#in sample predictions
pred1.is.class <- predict(ranger.model1.class, data = customer.train, type = 'response')
pred1.is.prob <- predict(ranger.model1.prob, data = customer.train, type = 'response')

confusionMatrix(data = pred1.is.class$predictions, reference = customer.train$campaign16,
    positive = '1')
postResample(pred1.is.class$predictions, customer.train$campaign16)

library(ROCR)
rocr_pred <- prediction(pred1.is.prob$predictions[,2], customer.train$campaign16)
rocr_perf <- performance(rocr_pred, measure="tpr", x.measure="fpr")
auc <- performance(rocr_pred, measure='auc')
auc <- auc@y.values[[1]]

rocr.data <- data.frame(fpr=unlist(rocr_perf@x.values),
tpr=unlist(rocr_perf@y.values))

insample.plot <- ggplot(rocr.data, aes(x=fpr, ymin=0, ymax=tpr)) +
geom_ribbon(alpha=0.2) +
geom_line(aes(y=tpr)) +
```

```
geom_abline(linetype="dashed") +
#ggtitle(expression(atop("Naive Bayes", atop("In-Sample ROC Curve")))) +
ggtitle("In-Sample ROC Curve - Random Forest") +
labs(x="False Positive Rate", y="True Positive Rate") +
annotate("text", x=0.75, y=0.25, label=paste("AUC:", round(auc,2)))

# out of sample predictions
pred1.oos.class <- predict(ranger.model1.class, data = customer.valid, type = 'response')
pred1.oos.prob <- predict(ranger.model1.prob, data = customer.valid, type = 'response')


confusionMatrix(pred1.oos.class$predictions, customer.valid$campaign16, positive = '1')
postResample(pred1.oos.class$predictions, customer.valid$campaign16)


library(ROCR)
rocr_pred <- prediction(pred1.oos.prob$predictions[,2], customer.valid$campaign16)
rocr_perf <- performance(rocr_pred, measure="tpr", x.measure="fpr")
auc <- performance(rocr_pred, measure='auc')
auc <- auc@y.values[[1]]

rocr.data <- data.frame(fpr=unlist(rocr_perf@x.values),
tpr=unlist(rocr_perf@y.values))

outofsample.plot <- ggplot(rocr.data, aes(x=fpr, ymin=0, ymax=tpr)) +
geom_ribbon(alpha=0.2) +
geom_line(aes(y=tpr)) +
geom_abline(linetype="dashed") +
#ggtitle(expression(atop("Naive Bayes", atop("In-Sample ROC Curve")))) +
ggtitle("Out-of-Sample ROC Curve - Random Forest") +
labs(x="False Positive Rate", y="True Positive Rate") +
annotate("text", x=0.75, y=0.25, label=paste("AUC:", round(auc,2)))



grid.arrange(insample.plot, outofsample.plot, nrow = 2)
```

08c-glmnet.R

```
# glmmnet model

library(glmnet)

# need to create a model matrix of the xvariables

glm.train <- customer.train %>%
select(-ACCTNO)

glm.train.x <- model.matrix(campaign16 ~ ., glm.train)[,-1]
glm.train.y <- customer.train$campaign16

glm.valid <- customer.valid %>%
select(-ACCTNO)

glm.valid.x <- model.matrix(campaign16 ~ ., glm.valid)[,-1]
glm.valid.y <- customer.valid$campaign16
```

```
# find the best alpha (hyper paramter tuning for the elasticnet)
# https://www.r-bloggers.com/variable-selection-with-elastic-net/
a <- seq(0.10, 0.9, 0.05)
search <- foreach(i = a, .combine = rbind) %dopar% {
cv <- cv.glmnet(glm.train.x, glm.train.y, family = "binomial", nfold = 10, type.measure = "
    deviance", parallel = TRUE, alpha = i)
data.frame(cvm = cv$cvm[cv$lambda == cv$lambda.1se], lambda.1se = cv$lambda.1se, alpha = i)
}

#search <- readRDS(file = 'data/glmnet-alpha-hyperparameter-results.RDS')
cv3 <- search[search$cvm == min(search$cvm), ]
cv3

fit.elnet <- glmnet(glm.train.x, glm.train.y, family="binomial", alpha=cv3$alpha, lambda =
    cv3$lambda.1se)
# fit.elnet <- glmnet(glm.train.x, glm.train.y, family = 'binomial', alpha = cv3$alpha)
#plot(fit.elnet, xvar="lambda")

# cross validation: https://stats.stackexchange.com/questions/72251/an-example-lasso-
    regression-using-glmnet-for-binary-outcome
cv.elnet1 <- cv.glmnet(glm.train.x, y=glm.train.y, alpha=cv3$alpha, family = 'binomial',
    parallel = T)
plot(cv.elnet1)
cv.elnet1$lambda.min
cv.elnet1$lambda.1se


coef(fit.elnet)


# in sample prediction
pred.insample.elnet <- predict(fit.elnet, newx = glm.train.x, type = 'response', s = cv.
    elnet1$lambda.1se)
insample.predictions <- rep(0,nrow(customer.train))
insample.predictions[pred.insample.elnet > 0.5] <-1
table(pred = insample.predictions, true = glm.train.y)

confusionMatrix(data = insample.predictions, reference = glm.train.y, positive = '1')

rocr_pred <- prediction(pred.insample.elnet, customer.train$campaign16)
rocr_perf <- performance(rocr_pred, measure="tpr", x.measure="fpr")
auc <- performance(rocr_pred, measure='auc')
auc <- auc@y.values[[1]]

rocr.data <- data.frame(fpr=unlist(rocr_perf@x.values),
tpr=unlist(rocr_perf@y.values))

insample.plot <- ggplot(rocr.data, aes(x=fpr, ymin=0, ymax=tpr)) +
geom_ribbon(alpha=0.2) +
geom_line(aes(y=tpr)) +
geom_abline(linetype="dashed") +
#ggtitle(expression(atop("Naive Bayes", atop("In-Sample ROC Curve")))) +
ggtitle("In-Sample ROC Curve - GLMnet") +
labs(x="False Positive Rate", y="True Positive Rate") +
annotate("text", x=0.75, y=0.25, label=paste("AUC:", round(auc,2)))
```

```
# out of sample predictions
pred.oos.elnet <- predict(fit.elnet, newx = glm.valid.x, type = 'response', s = cv.
    elnet1$lambda.1se)
oos.predictions <- rep(0, nrow(customer.valid))
oos.predictions[pred.oos.elnet > 0.5] <- 1
table(pred=oos.predictions, true = glm.valid.y)

confusionMatrix(data = oos.predictions, reference = glm.valid.y, positive = '1')

rocr_pred <- prediction(pred.oos.elnet, customer.valid$campaign16)
rocr_perf <- performance(rocr_pred, measure="tpr", x.measure="fpr")
auc <- performance(rocr_pred, measure='auc')
auc <- auc@y.values[[1]]

rocr.data <- data.frame(fpr=unlist(rocr_perf@x.values),
tpr=unlist(rocr_perf@y.values))

outofsample.plot <- ggplot(rocr.data, aes(x=fpr, ymin=0, ymax=tpr)) +
geom_ribbon(alpha=0.2) +
geom_line(aes(y=tpr)) +
geom_abline(linetype="dashed") +
#ggtitle(expression(atop("Naive Bayes", atop("In-Sample ROC Curve")))) +
ggtitle("Out-of-Sample ROC Curve - GLMnet") +
labs(x="False Positive Rate", y="True Positive Rate") +
annotate("text", x=0.75, y=0.25, label=paste("AUC:", round(auc,2)))


grid.arrange(insample.plot, outofsample.plot, nrow = 2)



pred.elnet <- predict(fit.elnet, newx = glm.valid.x, type = 'response', s = cv.elnet1$lambda
    .1se)
elasticnet.predictions <- rep(0,nrow(customer.valid))
elasticnet.predictions[pred.elnet > 0.5] <- 1
table(pred = elasticnet.predictions, true = glm.valid.y)
postResample(elasticnet.predictions, glm.valid.y)[[1]]

i = 0
for(i in seq(0.1, 0.9, 0.1)) {
ep <- rep(0, nrow(customer.valid))
ep[pred.insample.elnet > i] <- 1
print(postResample(ep, glm.valid.y)[[1]])
}


confusionMatrix(elasticnet.predictictions, customer.valid$campaign16, positive = '0')

library(pROC)
roc(glm.valid.y, elasticnet.predictictions)
plot(roc(glm.valid.y, elasticnet.predictictions))




# verify that all coded factors have at least 2 levels
# https://stackoverflow.com/questions/18171246/error-in-contrasts-when-defining-a-linear-
    model-in-r
l <- sapply(glm.train, function(x) is.factor(x))
```

```
l
m <- glm.train[, l]
ifelse(n <- sapply(m, function(x) length(levels(x))) == 1, "DROP", "NODROP")
```

08d-naivebayes.R

```
# Naive Bayes Classifer

# library(naivebayes)
#
# nb.model <- naive_bayes(
#   x = customer.train %>% select(-ACCTNO, -campaign16),
#   y = customer.train$campaign16
#     )
#
# nb.insample.class <- predict(nb.model, newdata = customer.train, type = 'class')
# postResample(nb.insample.class, customer.train$campaign16)
#
# nb.predict.class <- predict(nb.model, newdata = customer.valid, type = 'class')
#
# confusionMatrix(nb.predict.class, customer.valid$campaign16)
# postResample(nb.predict.class, customer.valid$campaign16)
#
#
# plot(roc(response = customer.valid$campaign16, predictor = as.numeric(nb.predict.class)))


# using caret
tControl <- trainControl(
method = 'cv',
number = 5,
allowParallel = T
# classProbs = T,
# summaryFunction = twoClassSummary,
# savePredictions = 'final',
# sampling = 'down'
)

fitNB <- train(
y = customer.train$campaign16,
x = customer.train %>% select(-campaign16, -ACCTNO),
method = 'naive_bayes',
trControl = tControl,
metric = 'Kappa',
preProc = c("center", "scale")
)


# in sample
fitNB.insample.pred <- predict(object = fitNB, newdata = customer.train)
confusionMatrix(data = fitNB.insample.pred, reference = customer.train$campaign16, positive
    = '1')
fitNB.insample.prob <- predict(object = fitNB, newdata = customer.train, type = 'prob')

library(ROCR)
rocr_pred <- prediction(fitNB.insample.prob[,2], customer.train$campaign16)
rocr_perf <- performance(rocr_pred, measure="tpr", x.measure="fpr")
```

```
auc <- performance(rocr_pred, measure='auc')
auc <- auc@y.values[[1]]

rocr.data <- data.frame(fpr=unlist(rocr_perf@x.values),
tpr=unlist(rocr_perf@y.values))

insample.plot <- ggplot(rocr.data, aes(x=fpr, ymin=0, ymax=tpr)) +
geom_ribbon(alpha=0.2) +
geom_line(aes(y=tpr)) +
geom_abline(linetype="dashed") +
#ggtitle(expression(atop("Naive Bayes", atop("In-Sample ROC Curve")))) +
ggtitle("In-Sample ROC Curve - Naive Bayes") +
labs(x="False Positive Rate", y="True Positive Rate") +
annotate("text", x=0.75, y=0.25, label=paste("AUC:", round(auc,2)))


# out of sample
fitNB.oos.pred <- predict(object = fitNB, newdata = customer.valid)
fitNB.oos.prob <- predict(object = fitNB, newdata = customer.valid, type = 'prob')
confusionMatrix(data = fitNB.oos.pred, reference = customer.valid$campaign16, positive =
    '1')

rocr_pred <- prediction(fitNB.oos.prob[,2], customer.valid$campaign16)
rocr_perf <- performance(rocr_pred, measure="tpr", x.measure="fpr")
auc <- performance(rocr_pred, measure='auc')
auc <- auc@y.values[[1]]

rocr.data <- data.frame(fpr=unlist(rocr_perf@x.values),
tpr=unlist(rocr_perf@y.values))

outofsample.plot <- ggplot(rocr.data, aes(x=fpr, ymin=0, ymax=tpr)) +
geom_ribbon(alpha=0.2) +
geom_line(aes(y=tpr)) +
geom_abline(linetype="dashed") +
#ggtitle(expression(atop("Naive Bayes", atop("In-Sample ROC Curve")))) +
ggtitle("Out-of-Sample ROC Curve - Naive Bayes") +
labs(x="False Positive Rate", y="True Positive Rate") +
annotate("text", x=0.75, y=0.25, label=paste("AUC:", round(auc,2)))

grid.arrange(insample.plot, outofsample.plot, nrow = 2)
```

value.R

```
catalog.cost <- 3

#campaign 16 revenue
customer.data %>%
filter(ANY_MAIL_16 == 0) %>%
select(ACCTNO, starts_with('TOTAMT'), TOTAL_MAIL_16) %>%
transmute(
ACCTNO,
total.sales = TOTAMT1 + TOTAMT2 + TOTAMT3 + TOTAMT4 + TOTAMT5 + TOTAMT6 + TOTAMT7 + TOTAMT8
    + TOTAMT9 + TOTAMT10 + TOTAMT11 + TOTAMT12 + TOTAMT13 + TOTAMT14 + TOTAMT15 + TOTAMT16,
TOTAL_MAIL_16
) %>%
filter(TOTAL_MAIL_16 > 0) %>%
```

```
mutate(
sales.per.mail = total.sales / TOTAL_MAIL_16
) %>%
summarize(
total.mailers = sum(TOTAL_MAIL_16),
all.sales = sum(total.sales),
total_accounts = length(ACCTNO),
total.costs = catalog.cost * total.mailers,
revenue.per.customer = all.sales / total_accounts,
profit.per.customer = (all.sales - total.costs) / total_accounts
)



test.data.summary <- customer.data %>%
filter(ANY_MAIL_16 == 1) %>%
select(ACCTNO, starts_with('TOTAMT'), TOTAL_MAIL_16) %>%
transmute(
ACCTNO,
total.sales = TOTAMT1 + TOTAMT2 + TOTAMT3 + TOTAMT4 + TOTAMT5 + TOTAMT6 + TOTAMT7 + TOTAMT8
    + TOTAMT9 + TOTAMT10 + TOTAMT11 + TOTAMT12 + TOTAMT13 + TOTAMT14 + TOTAMT15 + TOTAMT16,
TOTAL_MAIL_16
) %>%
filter(TOTAL_MAIL_16 > 0) %>%
mutate(
sales.per.mail = total.sales / TOTAL_MAIL_16
) %>%
summarize(
total.mailers = sum(TOTAL_MAIL_16),
all.sales = sum(total.sales),
total_accounts = length(ACCTNO),
total.costs = catalog.cost * total.mailers,
revenue.per.customer = all.sales / total_accounts,
profit.per.customer = (all.sales - total.costs) / total_accounts
)

profit <- round(test.data.summary$profit.per.customer,2)
profit

# profit per customer is 268.74


campaign17.data %>%
summarize(
num.accounts = n()
) %>%
mutate(
total.profit = num.accounts * profit
)


fitNB.test.prob <- predict(object = fitNB, newdata = campaign17.data, type = 'prob')


test.data <- tibble(success.prob = round(fitNB.test.prob[,2],3))

# now let's pick a probability threshold and create a classification

threshold <- 0
```

```
predicted.values <- test.data %>%
mutate(
predicted.class = ifelse(success.prob >= threshold, 1, 0),
expected.profit = ifelse(predicted.class == 1, success.prob * profit, 0)
)

predicted.values %>%
group_by(predicted.class) %>%
summarize(
obs.count = n(),
total.revenue = sum(expected.profit)
)

i = 0
tmp <- tibble(threshold = as.numeric(), obs.count = as.numeric(), total.revenue = as.numeric
    ())
for(i in seq(0,1,0.1)) {
# tmp <- tibble(threshold = as.numeric(), obs.count = as.numeric(), total.revenue = as.
    numeric())
threshold <- i
a <- test.data %>%
mutate(
predicted.class = ifelse(success.prob >= threshold, 1, 0),
expected.profit = ifelse(predicted.class == 1, success.prob * profit, 0)
)

b <- a %>%
group_by(predicted.class) %>%
filter(predicted.class == 1) %>%
summarize(
obs.count = n(),
total.revenue = sum(expected.profit)
)

tmp <- add_row(tmp, threshold = i, obs.count = b$obs.count, total.revenue = b$total.revenue)

}
```