MSIN0094 Second Assignment Answer Sheet

Candidate ID: NZQG9

Self-reported word count: 1497 words

1 Break-Even Response Rate (16pts)

Question 1. Discuss **break-even response rate** in general for targeted marketing campaigns.

The break-even response rate in marketing campaigns is the minimum percentage of customers who must respond to cover total costs. The calculation can help a marketing team identify what average customer response rate represents a successful campaign and who to target.

A simplistic marketing strategy would be to send out offers to individuals with a response rate of over 50%. However, the profit maximizing strategy would be to send offers to anyone over breakeven. By only targeting customers with a predicted response rate of 50%+, we would be leaving some profit on the table and so the correct strategy is to send offers to anyone over the break-even rate.

Question 2. Compute the break-even response rate for Tom's targeting campaign based on the cost information given and assign it into a variable named `breakeven_response_rate`.

```
# complete the code below.
# you can create additional intermediate variables if needed

average_COGS = 0.7 #Taken from average cogs in assignment brief
average_revenue = 40 #Taken from average revenue of goods purchased from Amazon Prime
Subscriber
shipping_cost = 6 #Taken from average shipping cost
subscription_fee = 8.99 #taken from 1 month subscription fee for Amazon Prime

cost_per_offer <- 1.5 #the cost of printing and mailing the offer

#profit per customer is the average revenue of goods purchased by new subscriber * 1-
the average COGS of those goods plus the subscription fee which the new customer will
have to pay and - the shipping cost which Amazon incurrs.
profit_per_customer <-  (average_revenue  *  (1-average_COGS)) +  subscription_fee  -
shipping_cost

#Breakeven response rate gives us the minimum percentage of customers that have to
respond to our offers in order for us to break even on the campaign. Thus it is the cost
for each offer we make divided by the calculated profit of an acquired customer.
breakeven_response_rate <- cost_per_offer/profit_per_customer
```

```
# pls do not modify the codes below
# these are for TAs to check results
print(paste("cost_per_offer is ", cost_per_offer))
```

```

```

```
[1] "cost_per_offer is  1.5"
```

```
print(paste("profit_per_customer is", profit_per_customer))
```

```
[1] "profit_per_customer is 14.99"
```

```
print(paste("breakeven_response_rate is", breakeven_response_rate))
```

```
[1] "breakeven_response_rate is 0.10006671114076"
```

Question 3. Compute the ROI of marketing for **blanket marketing** by sending offers to all 10,000 customers in the `data_full`.

Using the blanket marketing method on all 10,000 customers in the data_full data set, we can see that our calculated ROI is roughly −0.16. Given this information, Tom should not go ahead with blanket marketing as this ROI implies that for every dollar he spends on the campaign, the company loses $0.16. This is because blanket marketing incurs high costs due to a lower overall response rate of customers.

```
#The blanket method sends offers to everyone so we can calculate costs by using our
cost_per_offer variable and multiplying by the number of rows (or observations) in our
dataset
total_costs_of_mailing_blanket <- cost_per_offer * nrow(data_full)

#To calculate total profit, we use the profit_per_customer variable above and sum the
amount of observations where the subcribe variable is yes, recording these as 1 and no
as 0. This gives us a number of customers that actually subscribed to amazon prime.
total_profit_blanket <- profit_per_customer * sum(ifelse(data_full$subscribe == "yes",
1, 0))

#Once we have total costs and total profit of this method, we can calculate ROI by
subtracting profit by costs, and dividing again by costs.
ROI_blanket     <-     (total_profit_blanket     -     total_costs_of_mailing_blanket)/
total_costs_of_mailing_blanket
```

```
# do not modify the code below
print(paste("total_costs_of_mailing_blanket is ", total_costs_of_mailing_blanket))
```

```
[1] "total_costs_of_mailing_blanket is  15000"
```

```
print(paste("total_profit_blanket is ", total_profit_blanket))
```

```
[1] "total_profit_blanket is  12561.62"
```

```r
print(paste("ROI_blanket is ", ROI_blanket))
```

```
[1] "ROI_blanket is  -0.162558666666667"
```

2 Unsupervised Learning for Segmentation and Targeting (22 pts)

Question 4. Use `dplyr` data wrangling tools to compute the RFM variables for each customer in the dataset and assign them to the variables `recency`, `frequency`, and `monetary_value`. Report the summary statistics of these 3 variables. (Tips: the 3 variables can be generated based on existing variables in the dataset. frequency is about how many times the customer made purchases in the past year, and monetary value is about how much total spending the customer spent in the past year.)

```r
# create the RFM variables below
library(dplyr)

#Code below creates 3 variables recency, frequency, and monetary_value which represent
the RFM variables we will use in our k-means clustering analysis

# Recency Calcuations
recency <- data_full %>%
  mutate(recency = last) %>%  # We can directly use the 'last' column as recency variable
  select(user_id, recency)  # Selecting relevant columns

# Summary statistics for recency
summary(recency$recency)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   1.00    7.00   11.00   12.26   15.00   35.00
```

```r
# Frequency Calculations
frequency <- data_full %>%
  mutate(frequency = home + sports + clothes + health + books + digital + toys) %>%
  select(user_id, frequency)  # We select columns that indicate number of purchases for
each category and mutate them together.

# Summary statistics for frequency
summary(frequency$frequency)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   1.00    1.00    2.00    3.85    6.00   12.00
```

```r
#Monetary Value Calculations
monetary_value <- data_full %>%
  mutate(monetary_value = electronics + nonelectronics) %>%
  select(user_id, monetary_value)  # We add total spending in electronics and non-
electronics categories together and mutate.
```

```
# Summary statistics for Monetary Value
summary(monetary_value$monetary_value)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
   15.0   128.0   209.0   208.2   284.0   478.0 
```

Question 5. Use the three RFM variables to segment customers using the K-means clustering algorithm. Please use set.seed(888) for reproducibility. Specify `x` and `centers` in the `kmeans()` function, while using the default values for the remaining arguments.

Data pre-processing is important so that our k-means model performs as intended. Firstly, we have to create and add the recency, frequency, and monetary_value features so we can use these specific variables for clustering. Next, we want to standardize the data so that the relative scales of input features do not affect the clustering output. We want the features to affect clustering decisions at equal weights.

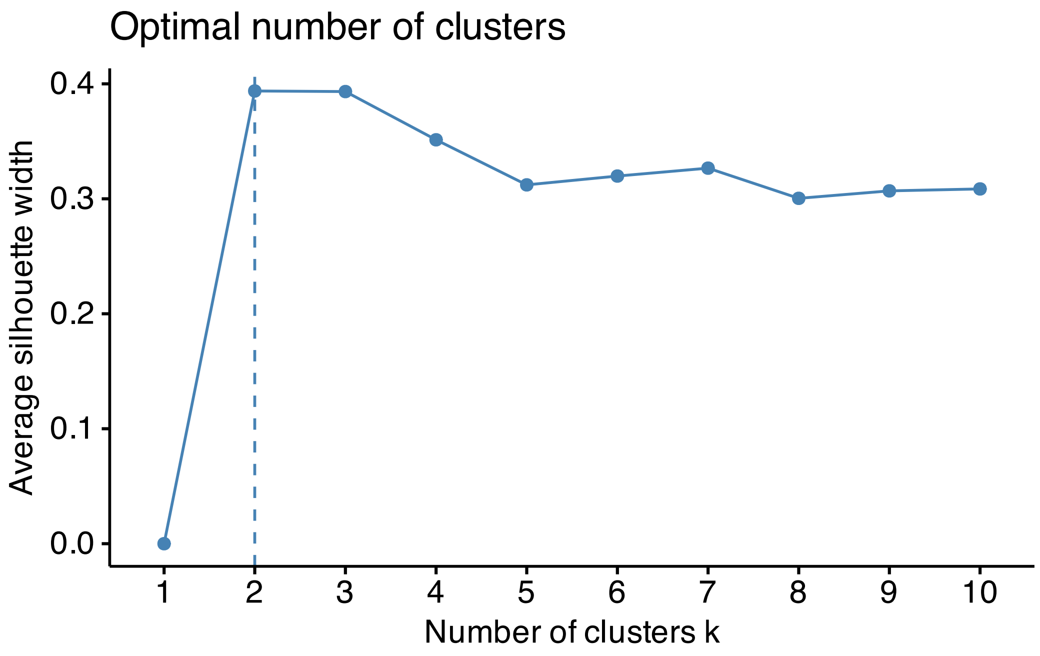
```
# Complete the code below to pre-process the data for k-means clustering

# Select only the relevant columns for clustering
data_kmeans <- data_full %>%
  mutate(
     #firstly we have to create new features that we will use for clustering based on customers behaviour. These three features are recency represents the time since the last customer's purchase, frequency which sums total num of purchases across all the product categories in our dataset, and monetary_value which sums toal spending across electronics and non-electronics
    recency = last,
    frequency = home + sports + clothes + health + books + digital + toys,
    monetary_value = electronics + nonelectronics
  ) %>%
   #After mutating these features to the data_kmeans dataset, we want to select the specific RFM variables for k means clustering
  select(recency, frequency, monetary_value)

# Standardize the data for k-means clustering
# Data standardization is important because the scale of input features can affect the clustering output. standardization will rescale features to a std. dev of 1 and a mean of 0 so they all equally impact the kmeans process.
data_kmeans <- scale(data_kmeans)
```

```
# Determine the optimal number of clusters using the Silhouette method below
library(factoextra)
```

```
Loading required package: ggplot2
```

```
Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```r
fviz_nbclust(data_kmeans, kmeans, method = "silhouette")
```



```r
# do not modify the seed below
set.seed(888)
```

```r
# implement k-means clustering below

# do not modify seeds
set.seed(888)
k_optimal <- 2 #Using the silhouette method we get the optimal number of clusters which
is 2
result_kmeans <- kmeans(data_kmeans, centers = k_optimal)
```

```r
# do no modify the code below, this is for TA to check the results

# use broom::tidy() to check the clusters.
pacman::p_load(broom)
tidy(result_kmeans)
```

```
# A tibble: 2 × 6
   recency frequency monetary_value  size withinss cluster
     <dbl>     <dbl>          <dbl> <int>    <dbl> <fct>
1 -0.00843    -0.547         -0.344  7263   13207. 1
2  0.0224      1.45           0.912  2737    5712. 2
```

Question 6. Among the segments from k-means, which segment should Tom target for the marketing campaign (i.e., send marketing offers to all customers in that segment)? (Tip: result_kmeans$cluster returns an R vector that tells us which segment each customer is in, in the same order as `data_kmeans`.)

The simplest way to determine which segment to target is by calculating the average response rate of each group. The average response rates were 0.068 and 0.12568 for segments 1 and 2 respectively. Thus, using this simple analysis of the response rates Tom should target segment 2 of the data set as this would result in a higher average customer response and therefore more profit.

```
# Write your codes here to reason which segment to target
# Show numbers from your coding to support your argument.
data_full <- data_full %>%
  mutate(
     #in the previous step we added the RFM values to the data_kmeans dataset, now we
want to add those variables into our original data_full dataset so that we can use
subscribe_numeric which is a new column based on the subscribe variable where yes is set
to 1 and no set to 0. This allows us to calculate the response rate of both segments
that were determined by kmeans clustering in the previous step. We assign the segments
with segment = result_kmeans$cluster
    recency = last,
    frequency = home + sports + clothes + health + books + digital + toys,
    monetary_value = electronics + nonelectronics,
    subscribe_numeric = ifelse(subscribe == "yes", 1, 0),
    segment = result_kmeans$cluster
    )

data_full %>%
  group_by(segment) %>%
  summarise(avg_response_rate = mean(subscribe_numeric, na.rm = T)) %>%
  ungroup()
```

```
# A tibble: 2 × 2
  segment avg_response_rate
    <int>             <dbl>
1       1            0.0680
2       2            0.126
```

```
# Compute the ROI of marketing if Tom conducts k-means targeted marketing to the segment
you selected
targeted_segment <- 2  #Segment with higher average response rate
target_customers <- data_full %>%
   filter(segment == targeted_segment) #filtering data to only include segment 2 which
we have determined has a higher response rate than segment 1

total_costs_of_mailing_kmeans <- nrow(target_customers) * cost_per_offer #calculating
costs of mailing offers to the customers in segment 2

total_profit_kmeans <- sum(target_customers$subscribe_numeric * profit_per_customer)
#calculating the total profit of customers in segment 2

ROI_kmeans <- (total_profit_kmeans - total_costs_of_mailing_kmeans) /
```

```r
total_costs_of_mailing_kmeans #using profit and costs, we can calculate ROI using the
(profit - costs)/costs formula
```

```r
# do not modify the code below

print(paste("ROI_kmeans is ", ROI_kmeans))
```

```
[1] "ROI_kmeans is  0.256012665935929"
```

Question 7. Should Tom use K-means clustering to conduct segmentation and targeting for this dataset? (**4pts**) (Tip: Think about the pros and cons of using K-means clustering)

On the positive side, k-means gives us an ROI of 0.256 as opposed to −0.162 using the blanket method. This means that while the blanket method lost money, k-means clustering was profitable. Tom was also able to identify a segment with a higher response rate than breakeven. There are also drawbacks to k-means clustering. Firstly, k-means assumes spherical clusters of data points, assigning them to the nearest centroid base. This means with data that form complex shapes, k-means may incorrectly classify customers. In Tom's case, I would suggest using a supervised learning model instead as the clusters are not natural. In addition, k-means groups customers solely on input variables, not accounting for subscription. As Tom's goal is to identify customers with a high average subscription rate, unsupervised methods like k-means might not be the most effective.

3 Decision Tree Analysis (24 pts)

Question 8. Complete the following data cleaning tasks

I first assign the existing data set to a new data set called data_tree so we can make changes without affecting future steps. An important first step is dividing the data into a training and a test set using a 75% training 25% test split. The reason for this is that we want to have some data to train our algorithm and a separate set of data where the subscribe variable is known to test the efficacy of our algorithm. With no split, we may run the risk of over-fitting, in which the model is very good at predicting outcomes for training data, but is less efficacious on future data.

```r
# set seed, please do not change the seed
set.seed(1314520)

# subscribe is a character, need to convert it into 1,0 variable
data_tree <- data_full %>%
    mutate(subscribe = ifelse(subscribe == "yes", 1,0))
#In the previous sections we created a new variable called subscribe_numeric to be a
numerical representation of yes and no, however the instructions for this step require
us to directly mutate the subscribe variable and so this is what is occuring in the code
above.

n_rows_data_tree <- nrow(data_tree)

# sample the index for training data
training_set_index <- sample(
  x = 1:n_rows_data_tree,
  size = 0.75 * n_rows_data_tree, #Here we are creating the training set by
  # taking 75% of the rows from the data_tree dataset
  replace = FALSE
)


# create data_training and data_test
data_training <- data_tree %>%
  slice(training_set_index) #our dataset for training slices the data tree set by only
the training set index set above

data_test <- data_tree %>%
  slice(-training_set_index) #using the - sign, we can set the test set to all the data
that is not in the training set
```

```
# Do not modify this code block.
# This is to print out first 5 customers
training_set_index[1:5]
```

```
[1] 3620   43 3574 4308 7387
```

```
str(data_training$recency)
```

```
 int [1:7500] 9 1 13 15 11 7 15 5 3 11 ...
```

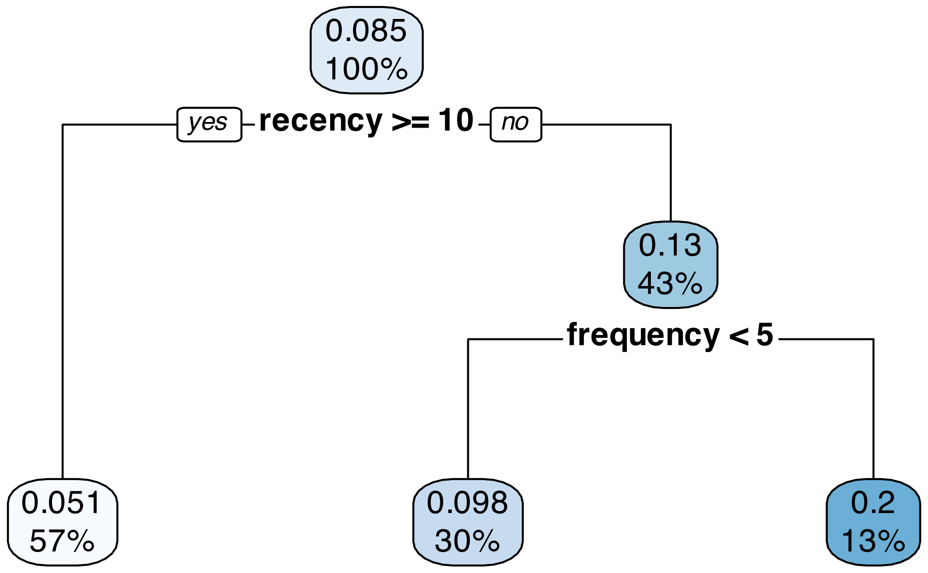
```
colnames(data_training) #This code is checking that the colnames we added in previous
steps for the RFM variables still exist in the training set that we have sliced.
```

```
 [1] "user_id"          "gender"           "first"
 [4] "last"             "electronics"      "nonelectronics"
 [7] "home"             "sports"           "clothes"
[10] "health"           "books"            "digital"
[13] "toys"             "subscribe"        "city"
[16] "recency"          "frequency"        "monetary_value"
[19] "subscribe_numeric" "segment"
```

Question 9. Complete the following steps to train a decision tree model

Decision trees start with a root node which includes 100% of the data set. At each node, the data splits based on which feature creates groups that are most different from each other in terms of the target outcome. We can see in this decision tree the first split occurs based on the recency variable being >= 10 or not for all data points. The left path ends with a leaf that has an average response rate of 0.051. The right path leads to another node, which splits the data by frequency being < 5 or not. This results in another two leaves with average response rates of 0.098 and 0.2. Given these splits, we are now able to target the groups of people who have a higher response rate than our break-even rate of 0.1. For future customers, the decision tree will split the data using the same predetermined rules. The algorithm leads customers to a leaf which we will either choose to target or ignore based on the average response rate.

```r
pacman::p_load(rpart, rpart.plot)
# train model tree1 below

tree1 <- rpart(
  formula = subscribe ~ recency + frequency + monetary_value,
  data = data_training,
  method = "anova"
)

# visualize tree1 below

rpart.plot(tree1)
```



Question 10. Compute the ROI from targeted marketing using `tree1`.

There is a difference in the ROI of the k-means method and our decision tree because they are fundamentally different approaches. K-means clustering is unsupervised learning, where the algorithm groups customers by similarity in our RFM variables without consideration of whether they are subscribed or not. In decision trees, we are using a supervised machine learning model in which we directly state we want the model to predict whether subscribed or not, based on the same RFM variables. While k-means groups customers by a combination of all RFM variables, decision trees make decisions on one input variable at each node, leading to fundamentally different outcomes.

```r
# Compute the ROI for tree1 below.
# First we use predict() to make predictions on the test set
# The predicted values will be treated as a continuous output as this is a regression/
ANOVA task
prediction_from_decision_tree <- predict(tree1, data_test)

# Add a new column in data_test for the predicted probability for each potential customer
data_test <- data_test %>%
  mutate(predicted_prob_decisiontree = prediction_from_decision_tree)

# Mutate a new binary indicator for whether to target a customer based on predicted
probability if it is higher than breakeven
# breakeven_respons_rate is predefined in first section
data_test <- data_test %>%
        mutate(is_target_decisiontree    =    ifelse(predicted_prob_decisiontree    >
breakeven_response_rate, 1, 0))

# We have to now calculate total marketing costs based on cost per offer and the amount
of people we choose to target based on decision tree.
total_costs_of_mailing_decisiontree           <-           cost_per_offer           *
sum(data_test$is_target_decisiontree)

# Filter out customers who receive the marketing offer
data_test_targeted_customers <- data_test %>%
  filter(is_target_decisiontree == 1)

# Calculate total profits from responding customers
total_profit_decisiontree     <-     sum(data_test_targeted_customers$subscribe)     *
profit_per_customer

# Compute ROI
ROI_decisiontree <- (total_profit_decisiontree - total_costs_of_mailing_decisiontree) /
total_costs_of_mailing_decisiontree

# Output the ROI
ROI_decisiontree
```

```
[1] 0.7185987
```

4 Random Forest (20 pts)

Question 11. Train a random forest model with the same set of predictors as `tree1`

For random forest. I set the formula to predict subscribers based on the RFM variables and set the data to the data_training data set. In this model, we set probability to true so that we get a probability output instead of the default binary classification. We also set the number of trees to 5000 as required.

```
pacman::p_load(ranger)
# train the random forest model below
set.seed(888)
randomforest <- ranger(
  formula = subscribe ~ recency + frequency + monetary_value,
```

```r
  data = data_training,
  probability = TRUE,  # To get class probabilities
  num.trees = 5000

)
#making a prediction on the test set based on our randomforest model that has been
trained on the data_training set
prediction_from_randomforest <- predict(randomforest, data_test)

#For randomforest as opposed to decision trees, we get a matrix and not a vector so we
have to take the second column
data_test <- data_test %>%
  mutate(predicted_prob_random_forest = prediction_from_randomforest$predictions[, 2])
# compute the ROI for random forest below below

#First we have to create a new binary indicator for targeting potential customers based
on what their predicted probability is
data_test <- data_test %>%
        mutate(is_target_randomforest    =    ifelse(predicted_prob_random_forest    >
breakeven_response_rate, 1, 0))

#Next we calculate total marketing costs

total_costs_of_mailing_randomforest           <-           cost_per_offer           *
sum(data_test$is_target_randomforest)

#Now we identify responding customers who were targeted by the random forest model
data_responding_targeted_customers <- data_test %>%
  filter(is_target_randomforest == 1) %>%  #Customers who are targeted
  filter(subscribe == 1) #customers who actually responded in test set

#Then we need to calculate the profits from responding customers
total_profit_randomforest      <-      nrow(data_responding_targeted_customers)      *
profit_per_customer

#ROI is simply the total profit - total costs divided by total costs for this chosen
method
ROI_randomforest <- (total_profit_randomforest - total_costs_of_mailing_randomforest) /
total_costs_of_mailing_randomforest


ROI_randomforest
```

```
[1] 0.4295828
```

```r
# do not modify the code below

print(paste("ROI_randomforest is ", ROI_randomforest))
```

```
[1] "ROI_randomforest is  0.429582760201743"
```

Question 12. Based on the results of different models you have obtained in this assignment so far, discuss the fundamental trade-offs of supervised learning

The first tradeoff is between bias and variance. These terms are tied in with over-fitting or under-fitting, which refers to how well models represent training data. Low bias implies that the model fits the training data well but may perform poorly on the test data because it is over-fitting and has high variance. High bias implies that the model performs worse on the training potentially due to under-fitting, often leading to poor performance on the test set. In our context, we could say that the random forest model has higher bias and lower variance as it produces a better generalization to new data given that it is an average of many trees, but may under-fit the training data. Alternatively, the single decision tree has low bias/high variance through creating rules that fit the training set well but may be prone to over-fitting.

Secondly, there is a trade-off between interpretability and complexity. In our context, the decision tree model is easier to interpret but less complex than the random-forest model as it is a single tree. The random-forest model is difficult to interpret, as it is an average of many trees, but has higher complexity potentially leading to improved accuracy. In our case, however, the random-forest model performed worse, showing that the random nature of these algorithms can produce unexpected results.

Question 13. Discuss at least two recommendations for Tom to improve the performance of the random forest model. (**8pts**)

One way Tom could improve the performance of the random forest model would be to use better variable selection. In our case, we only used the RFM variables in our model, however, Tom could add additional variables that may lead to a higher predicting power. Tom could also combine variables and create composite features, enhancing model accuracy. Another way Tom could improve the performance of the model would be to train it with a larger data set, this would lead to a reduction in variance and therefore better generalization of the model to future data.

5 A/B/N Testing (18 pts)

Question 14. Help Tom design the A/B/N testing based on the following structure. (This is a discussion question, please discuss each step below.)

Tom should include conditions where the marketing offer is sent off based on a decision tree, random forest or K-means clustering algorithm and a control group that receives no offer. Tom should use an individual unit of randomization for this as it is more granular and shows the effect of the treatments on a customer level. Most customers would also have their own Amazon account or subscription so this is in line with the reality of the product. Regarding crossover/spillover effects, some customers may live in households or have friends who are exposed to alternate treatment groups that received the offer. This could inflate the proportion of customers who subscribe to Amazon. Additionally, customers may check their Amazon basket without signing in or signing in to a friend's account that has prime delivery, skewing purchase data. These effects could be mitigated by requiring customers to sign in for purchases or excluding participants living in the same postcode. For randomization, Tom could assign a randomly generated number to all participants and divide them into 4 groups based on percentiles, with a larger proportion in the control group than in treatment groups. Tom would want to use a large sample size for this experiment and could calculate this based on a power calculation using R with his chosen level of significance. Tom should collect information on the outcome variable, subscribed or not, as well as other customer characteristics such as the RFM variables and more. For statistical analysis, Tom should do a T-test for significance between different treatment outcome variables to see which algorithm was most effective. He could also calculate basic summary statistics such as mean ROI and Std. Dev for further insights.