R Notebook: iFood CRM Data Analyst Case

Business Analysis on Customer Relationship Management

Eduar Felipe Riaño Torres*

2021-03-25

Contents

*felipehuman@gmail.com

1 Executive Summary

- A pilot campaign involving **2.240 customers** was carried out, customers who bought the offer were properly labeled. The total cost of the sample campaign was **6.720MU** and the revenue generated by the customers who accepted the offer was **3.674MU**.
- Globally the campaign had a profit of **-3.046MU** and the success rate of the campaign was **15%**.
- Through an **Exploratory Data Analysis (EDA)**, it was studied the characteristic features of customers.
- Based on customers behaviors, it is necessary to create and describe a **customer segmentation**.
- To maximize the profit of the next marketing campaign profit, it was built a **predictive model** (classification).

2 Loading R libraries

I load a range of R libraries for general data wrangling, transformation, analyzing and visualization together with more specialized tools.

```
# summarization
library(skimr)
library(Hmisc)
library(knitr) # some tables and R Markdown
library(correlationfunnel) # correlation Analysis

# articulation with Python
library(reticulate)

# general visualization
library(ggplot2) # visualization
library(scales) # visualization
library(grid) # visualization
library(gridExtra) # visualization
library(RColorBrewer) # visualization
```

```r
library(corrplot) # visualization
library(reshape2) # visualization
library(hrbrthemes) # visualization

# general data manipulation
library(dplyr) # data manipulation
library(readr) # input/output
library(data.table) # data manipulation
library(tibble) # data wrangling
library(tidyr) # data wrangling
library(stringr) # string manipulation
library(forcats) # factor manipulation
library(tidyverse) # plotting, cleaning, etc
library(gdata)
library(plyr)

# specific visualization
library(alluvial) # visualization
library(ggrepel) # visualization
library(ggforce) # visualization
library(ggridges) # visualization
library(gganimate) # animations
library(gridExtra) # visualization
library(GGally) # visualization
library(ggExtra) # visualization
library(highcharter) # visualization
library(countrycode) # visualization
library(geofacet) # visualization
library(wesanderson) # color palettes
library(treemapify) # visualization
library(cluster) # visualization
library(gridExtra) # visualization
library(grid) # visualization

# specific data manipulation
library(lazyeval) # data wrangling
library(broom) # data wrangling
library(purrr) # string manipulation
library(reshape2) # data wrangling
library(rlang) # encoding

# analysis
library(lubridate)
library(tidyverse)
library(caret)
library(xgboost)
library(modeest)
library(NbClust)
library(factoextra)
library(tidymodels) # framework for ML
library(ranger)
library(randomForest)
library(vip)
library(DataExplorer)
```

```r
library(tidyquant)
```

3 Loading Python libraries

As I did before with R libraries, I import some of the most important libraries in Python.

```python
import numpy as np
import pandas as pd
import statistics as stat
import matplotlib.pyplot as plt

import seaborn as sns
sns.set()

# Load the main dataset in Python environment
raw_ifood = pd.read_csv('ml_project1_data.csv')
```

4 Setting working directory

```r
setwd("C:/Users/eduar/Downloads/My Things/Data sets R/iFood")
```

5 Loading the main dataset

The dataset contains socio-demographic and firmographic features about 2.240 customers who were contacted. Additionally, it contains a flag for those customers who responded the campaign, by buying the product.

```r
# Load the main data set in R environment
raw_ifood <- read.csv("ml_project1_data.csv")
```

6 Section 01: Exploratory Data Analysis

This is the first approach to the data set. I am going to analyze the map (dataset) and get a big picture perspective of it. In similar way, to ensure a holistic approach. Once data is imported, it is a good idea to to tidy it. Tidying data, means storing it in a consistent form that matches the semantics of the dataset with the way it stored.

```python
raw_ifood.head()
```

```
##      ID  Year_Birth   Education  ... Z_CostContact  Z_Revenue  Response
## 0  5524        1957  Graduation  ...             3         11         1
## 1  2174        1954  Graduation  ...             3         11         0
## 2  4141        1965  Graduation  ...             3         11         0
## 3  6182        1984  Graduation  ...             3         11         0
## 4  5324        1981         PhD  ...             3         11         0
##
## [5 rows x 29 columns]
```

```
raw_ifood.shape
```

```
## (2240, 29)
```

```
raw_ifood.describe()
```

```
##                   ID   Year_Birth  ...  Z_Revenue     Response
## count    2240.000000  2240.000000  ...     2240.0  2240.000000
## mean     5592.159821  1968.805804  ...       11.0     0.149107
## std      3246.662198    11.984069  ...        0.0     0.356274
## min         0.000000  1893.000000  ...       11.0     0.000000
## 25%      2828.250000  1959.000000  ...       11.0     0.000000
## 50%      5458.500000  1970.000000  ...       11.0     0.000000
## 75%      8427.750000  1977.000000  ...       11.0     0.000000
## max     11191.000000  1996.000000  ...       11.0     1.000000
##
## [8 rows x 26 columns]
```

```
raw_ifood %>% plot_missing(missing_only = FALSE, title = "Missing Values")
```

- Note: All NAs come from Income column. This means that there is missing Income data for 24 customers, data, and none of the other columns contain any missing data.

```
# Removing data points where Income = NA
print("Number of Datapoints removed Income=NA: ", raw_ifood.isnull().sum().sum())
```

```
## Number of Datapoints removed Income=NA:  24
```

```
raw_ifood <- raw_ifood %>% filter(.,!is.na(Income))
summary(raw_ifood$Income)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1730   35303   51382   52247   68522  666666
```

```
summary(raw_ifood)
```

```
##        ID          Year_Birth    Education         Marital_Status
##  Min.   :    0   Min.   :1893   Length:2216        Length:2216
##  1st Qu.: 2815   1st Qu.:1959   Class :character   Class :character
##  Median : 5458   Median :1970   Mode  :character   Mode  :character
##  Mean   : 5588   Mean   :1969
##  3rd Qu.: 8422   3rd Qu.:1977
##  Max.   :11191   Max.   :1996
##      Income          Kidhome          Teenhome      Dt_Customer
##  Min.   :  1730   Min.   :0.0000   Min.   :0.0000   Length:2216
##  1st Qu.: 35303   1st Qu.:0.0000   1st Qu.:0.0000   Class :character
##  Median : 51382   Median :0.0000   Median :0.0000   Mode  :character
##  Mean   : 52247   Mean   :0.4418   Mean   :0.5054
##  3rd Qu.: 68522   3rd Qu.:1.0000   3rd Qu.:1.0000
##  Max.   :666666   Max.   :2.0000   Max.   :2.0000
##     Recency         MntWines        MntFruits      MntMeatProducts
##  Min.   : 0.00   Min.   :   0.0   Min.   :  0.00   Min.   :   0.0
##  1st Qu.:24.00   1st Qu.:  24.0   1st Qu.:  2.00   1st Qu.:  16.0
##  Median :49.00   Median : 174.5   Median :  8.00   Median :  68.0
##  Mean   :49.01   Mean   : 305.1   Mean   : 26.36   Mean   : 167.0
##  3rd Qu.:74.00   3rd Qu.: 505.0   3rd Qu.: 33.00   3rd Qu.: 232.2
##  Max.   :99.00   Max.   :1493.0   Max.   :199.00   Max.   :1725.0
##  MntFishProducts  MntSweetProducts  MntGoldProds    NumDealsPurchases
##  Min.   :  0.00   Min.   :  0.00   Min.   :  0.00   Min.   : 0.000
##  1st Qu.:  3.00   1st Qu.:  1.00   1st Qu.:  9.00   1st Qu.: 1.000
##  Median : 12.00   Median :  8.00   Median : 24.50   Median : 2.000
##  Mean   : 37.64   Mean   : 27.03   Mean   : 43.97   Mean   : 2.324
##  3rd Qu.: 50.00   3rd Qu.: 33.00   3rd Qu.: 56.00   3rd Qu.: 3.000
##  Max.   :259.00   Max.   :262.00   Max.   :321.00   Max.   :15.000
##  NumWebPurchases  NumCatalogPurchases NumStorePurchases NumWebVisitsMonth
##  Min.   : 0.000   Min.   : 0.000      Min.   : 0.000    Min.   : 0.000
##  1st Qu.: 2.000   1st Qu.: 0.000      1st Qu.: 3.000    1st Qu.: 3.000
##  Median : 4.000   Median : 2.000      Median : 5.000    Median : 6.000
##  Mean   : 4.085   Mean   : 2.671      Mean   : 5.801    Mean   : 5.319
##  3rd Qu.: 6.000   3rd Qu.: 4.000      3rd Qu.: 8.000    3rd Qu.: 7.000
##  Max.   :27.000   Max.   :28.000      Max.   :13.000    Max.   :20.000
##   AcceptedCmp3      AcceptedCmp4      AcceptedCmp5     AcceptedCmp1
```

```
##  Min.   :0.00000   Min.   :0.00000   Min.   :0.0000   Min.   :0.00000
##  1st Qu.:0.00000   1st Qu.:0.00000   1st Qu.:0.0000   1st Qu.:0.00000
##  Median :0.00000   Median :0.00000   Median :0.0000   Median :0.00000
##  Mean   :0.07356   Mean   :0.07401   Mean   :0.0731   Mean   :0.06408
##  3rd Qu.:0.00000   3rd Qu.:0.00000   3rd Qu.:0.0000   3rd Qu.:0.00000
##  Max.   :1.00000   Max.   :1.00000   Max.   :1.0000   Max.   :1.00000
##   AcceptedCmp2        Complain        Z_CostContact   Z_Revenue
##  Min.   :0.00000   Min.   :0.000000   Min.   :3     Min.   :11
##  1st Qu.:0.00000   1st Qu.:0.000000   1st Qu.:3     1st Qu.:11
##  Median :0.00000   Median :0.000000   Median :3     Median :11
##  Mean   :0.01354   Mean   :0.009477   Mean   :3     Mean   :11
##  3rd Qu.:0.00000   3rd Qu.:0.000000   3rd Qu.:3     3rd Qu.:11
##  Max.   :1.00000   Max.   :1.000000   Max.   :3     Max.   :11
##     Response
##  Min.   :0.0000
##  1st Qu.:0.0000
##  Median :0.0000
##  Mean   :0.1503
##  3rd Qu.:0.0000
##  Max.   :1.0000
```

Throughout the data it is possible to see:

- 2240 customers (observations)
- 29 variables/columns
- 3 character variables (i.e. strings)
- 26 numeric variables (i.e. numbers / floats)

Additionally, I categorize the variables in terms of the 4 P's of marketing:People (customers), Products, Places (channels), and Promotions (discounts and campaigns). The purpose of this classification is to increase the analyst's perspective. It might also provide a great way to segment analytic steps in the Exploratory Data Analysis.

Table 1: 4 P's

People	Products	Place	Promotion
ID	MntWines	NumWebPurchases	NumDealsPurchases
Year_Birth	MntFruits	NumCatalogPurchases	AcceptedCmp1
Education	MntMeatProducts	NumStorePurchases	AcceptedCmp2
Marital_Status	MntFishProducts	NumWebVisitsMonth	AcceptedCmp3
Income	MntSweetProducts		AcceptedCmp4
Kidhome	MntGoldProds		AcceptedCmp5
Teenhome			Response
Dt_Customer			
Recency			
Complain			

```
glimpse(raw_ifood)
```

```
## Rows: 2,216
## Columns: 29
## $ ID                 <int> 5524, 2174, 4141, 6182, 5324, 7446, 965, 6177, ...
```

```
## $ Year_Birth          <int> 1957, 1954, 1965, 1984, 1981, 1967, 1971, 1985,...
## $ Education           <chr> "Graduation", "Graduation", "Graduation", "Grad...
## $ Marital_Status      <chr> "Single", "Single", "Together", "Together", "Ma...
## $ Income              <int> 58138, 46344, 71613, 26646, 58293, 62513, 55635...
## $ Kidhome             <int> 0, 1, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 0, 1,...
## $ Teenhome            <int> 0, 1, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1,...
## $ Dt_Customer         <chr> "2012-09-04", "2014-03-08", "2013-08-21", "2014...
## $ Recency             <int> 58, 38, 26, 26, 94, 16, 34, 32, 19, 68, 59, 82,...
## $ MntWines            <int> 635, 11, 426, 11, 173, 520, 235, 76, 14, 28, 6,...
## $ MntFruits           <int> 88, 1, 49, 4, 43, 42, 65, 10, 0, 0, 16, 61, 2, ...
## $ MntMeatProducts     <int> 546, 6, 127, 20, 118, 98, 164, 56, 24, 6, 11, 4...
## $ MntFishProducts     <int> 172, 2, 111, 10, 46, 0, 50, 3, 3, 1, 11, 225, 3...
## $ MntSweetProducts    <int> 88, 1, 21, 3, 27, 42, 49, 1, 3, 1, 1, 112, 5, 1...
## $ MntGoldProds        <int> 88, 6, 42, 5, 15, 14, 27, 23, 2, 13, 16, 30, 14...
## $ NumDealsPurchases   <int> 3, 2, 1, 2, 5, 2, 4, 2, 1, 1, 1, 1, 3, 1, 1, 3,...
## $ NumWebPurchases     <int> 8, 1, 8, 2, 5, 6, 7, 4, 3, 1, 2, 3, 6, 1, 7, 3,...
## $ NumCatalogPurchases <int> 10, 1, 2, 0, 3, 4, 3, 0, 0, 0, 0, 4, 1, 0, 6, 0...
## $ NumStorePurchases   <int> 4, 2, 10, 4, 6, 10, 7, 4, 2, 0, 3, 8, 5, 3, 12,...
## $ NumWebVisitsMonth   <int> 7, 5, 4, 6, 5, 6, 6, 8, 9, 20, 8, 2, 6, 8, 3, 8...
## $ AcceptedCmp3        <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,...
## $ AcceptedCmp4        <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ AcceptedCmp5        <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,...
## $ AcceptedCmp1        <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,...
## $ AcceptedCmp2        <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Complain            <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Z_CostContact       <int> 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,...
## $ Z_Revenue           <int> 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11,...
## $ Response            <int> 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0,...
```

6.1 Data cleaning before any further process

I have performed some transformations on the variables "Year_Birth", "Income" and "Dt_customer". The main reason is that the syntax of the data contained is not suitable for statistical analysis, such as currency formatting or date formatting.

- Based on Year_Birth column, bring out Age of Customers.
- Make Income column numeric and also reformat the data values to remove dollar sign and commas.
- Make Dt_customer a date column.

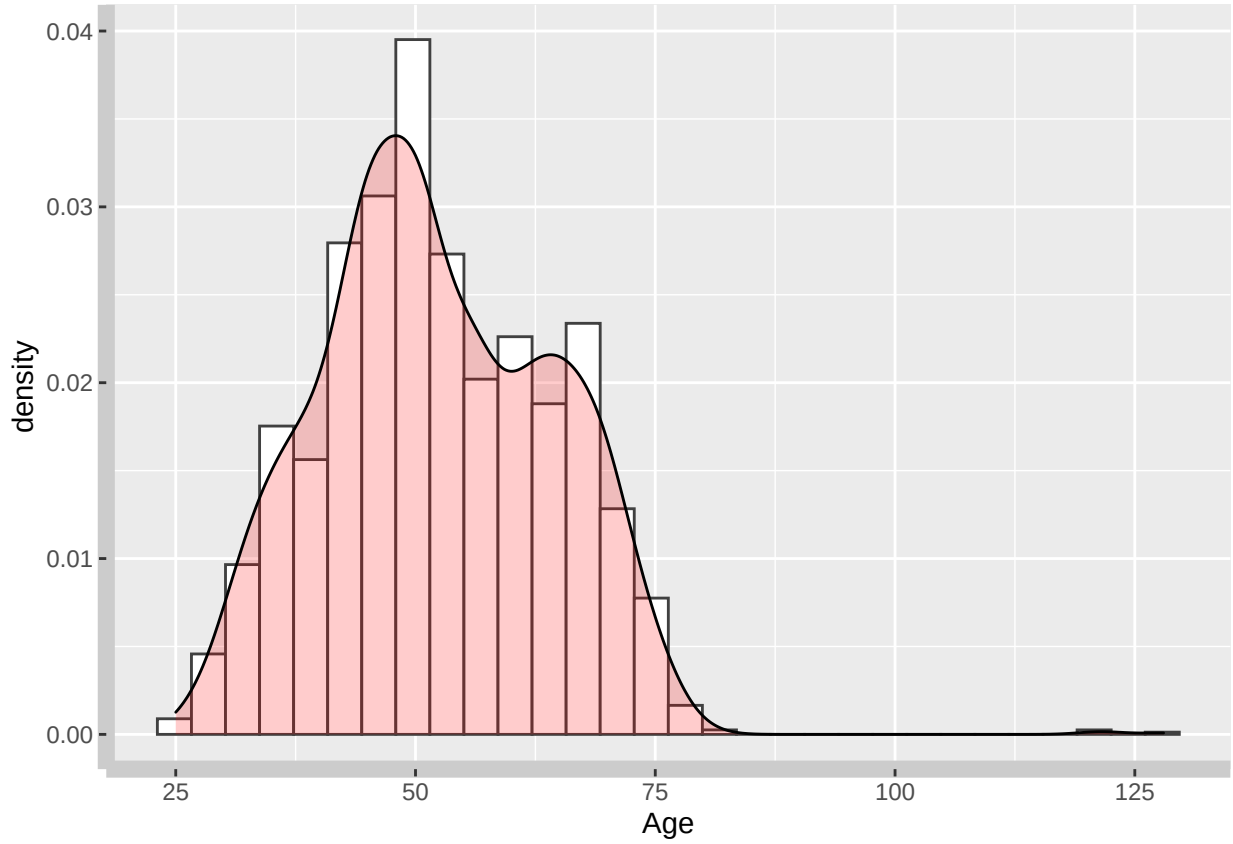
```
# Making income numeric
raw_ifood$Income <- as.numeric(raw_ifood$Income %>% gsub("[$,]","",.))
summary(raw_ifood$Income)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1730   35303   51382   52247   68522  666666
```

```
# Extracting Age of Customers. Note: current year = 2021
current_year = 2021
raw_ifood <-mutate(raw_ifood, Age = current_year - raw_ifood$Year_Birth)
summary(raw_ifood$Age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   25.00   44.00   51.00   52.18   62.00  128.00
```

```
raw_ifood %>%
              ggplot(aes(x = Age)) +
              geom_histogram(aes(y= ..density..), colour = "gray25", fill = "white") +
              geom_density(alpha=.2, fill ="red1") +
              theme(axis.line = element_line(size = 3, colour = "grey80"))
```



The Age Column has outliers, the previous histogram shows the distribution of the variable and shows that are some ages ahead of 100 years. Those middle age groups, 40 and 60 somethings, make up the majority of respondents. Even though, there is a notable number of respondents at ages of 60+.

```
#Remove data points where Age>100
paste0("Number of Datapoints removed for Age>100: ",
       sum(raw_ifood$Age > 100))
```

```
## [1] "Number of Datapoints removed for Age>100: 3"
```

```
raw_ifood <- raw_ifood %>%
             filter(.,Age<100)
```

```
summary(raw_ifood$Age)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   25.00   44.00   51.00   52.08   62.00   81.00
```

```
# Making Dt_customer a date column
raw_ifood$Dt_Customer <- ymd(raw_ifood$Dt_Customer)
summary(raw_ifood$Dt_Customer)
```

```
##         Min.      1st Qu.       Median         Mean      3rd Qu.         Max.
## "2012-07-30" "2013-01-16" "2013-07-08" "2013-07-10" "2013-12-31" "2014-06-29"
```

6.2 Analysis and Data Visualization of all features

For a better understanding of the data, I will plot each variable to visualize the distribution of the data and identify any outliers or imbalanced classes. Using mainly graphs like boxplots for quantitative variables and barplots (of value counts) for qualitative, categorical and binary (yes/no) variables.

6.2.1 Grouping by People

```
# Discrete variables
people_discrete <- raw_ifood %>% select(c('Education','Marital_Status',
                                          'Kidhome','Teenhome','Complain'))

# Continuous variables
people_continuous <- raw_ifood %>% select(c('Year_Birth','Income',
                                            'Dt_Customer','Recency'))
```

6.2.2 Grouping by Product

```
data_products <- raw_ifood %>% select(c('MntWines','MntFruits',
                                        'MntMeatProducts',
                                        'MntFishProducts',
                                        'MntSweetProducts',
                                        'MntGoldProds'))
```

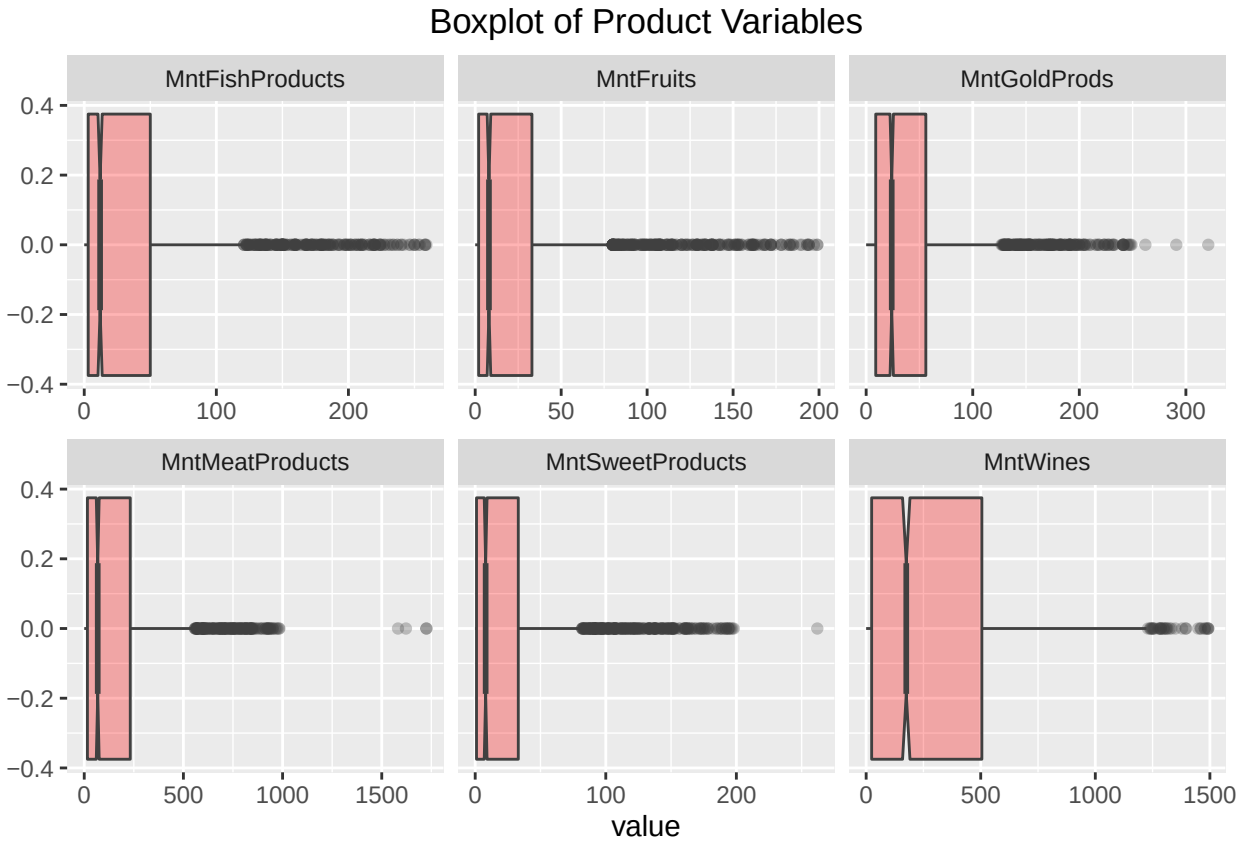
6.2.3 Grouping by Place

```
data_place <- raw_ifood %>% select(c('NumWebPurchases','NumCatalogPurchases',
                                     'NumStorePurchases','NumWebVisitsMonth'))
```

6.2.4 Grouping by Promotion

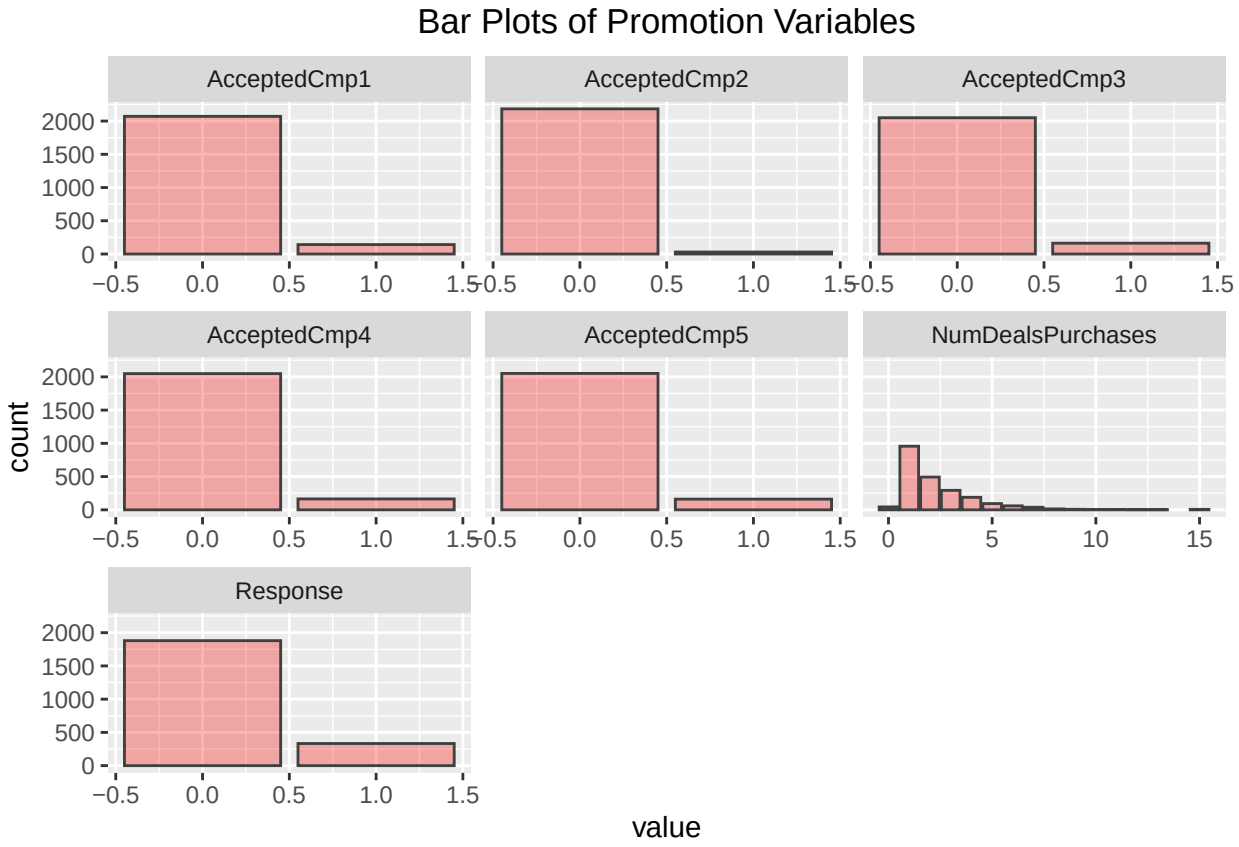
```
data_promotion <- raw_ifood %>% select(c('AcceptedCmp1','AcceptedCmp2','AcceptedCmp3',
                                         'AcceptedCmp4',
                                         'AcceptedCmp5',
                                         'Response',
                                         'NumDealsPurchases'))


# Creating a key-value data set for each category
product_key_value     <- gather(data_products)
place_key_value       <- gather(data_place)
people_key_value_disc <- gather(people_discrete)
people_key_value_cont <- gather(people_continuous)
promotion_key_value   <- gather(data_promotion)

# Plotting box plots for product variables
ggplot(product_key_value, aes(value))                                    +
geom_boxplot(alpha=.3, fill ="red1", colour = "gray25", notch = TRUE) +
```

```
facet_wrap(~key, scales = 'free_x', ncol = 3)                    +
labs(title = "Boxplot of Product Variables")                     +
theme(plot.title = element_text(hjust = 0.5))
```



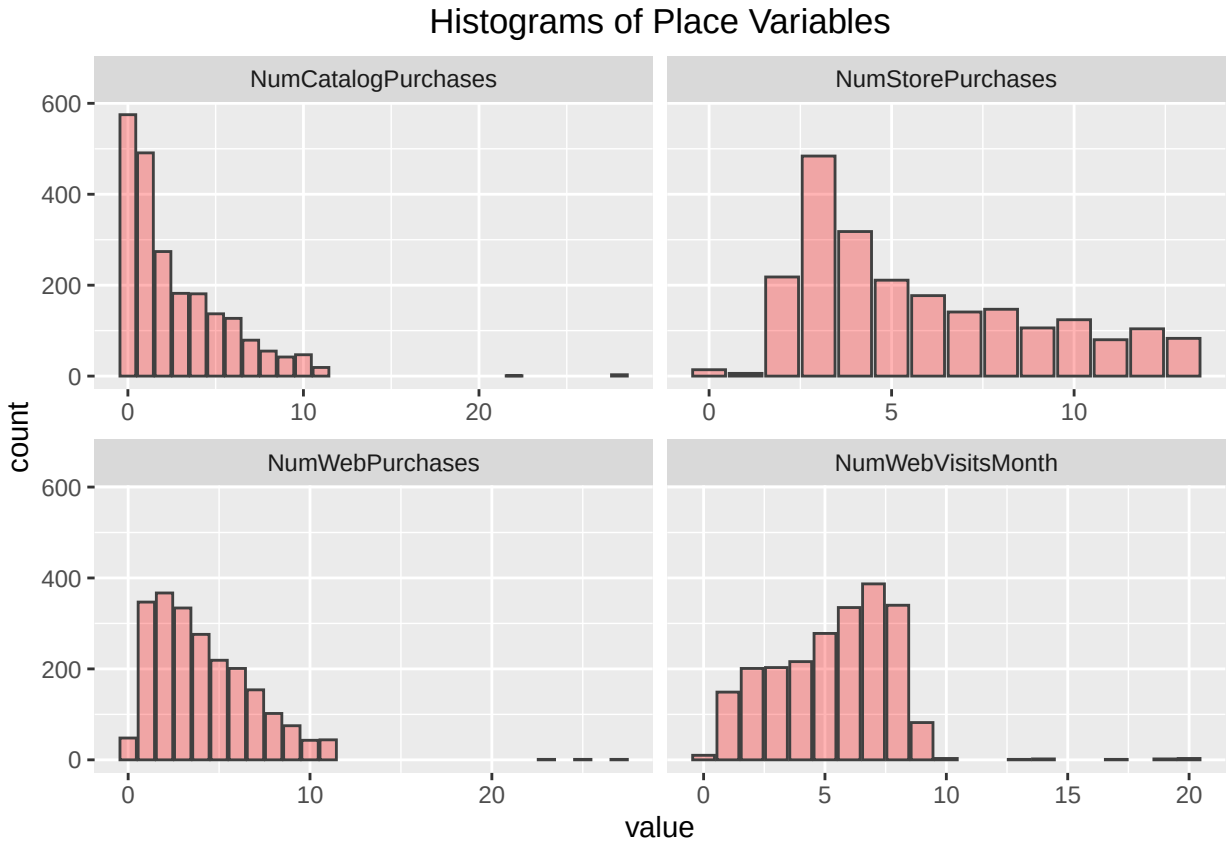
- It is important to notice that there seems to be nothing out of the expected in terms of the amount spent per product. Nevertheless, it is necessary to consider whether those customers who bought more than 1500 meat products over the past few years differ from the rest of the customers. In general, the outliers for these categories are all acceptable and valid values of purchases, and shall be kept as is.

```
# Plotting bar-plots for promotion variables
ggplot(promotion_key_value, aes(value)) +
geom_histogram(stat = 'count',alpha=.3, fill ="red1",
               colour = "gray25", position="identity") +
facet_wrap(~key, scales = 'free_x', ncol = 3) +
labs(title = "Bar Plots of Promotion Variables") +
theme(plot.title = element_text(hjust = 0.5))
```



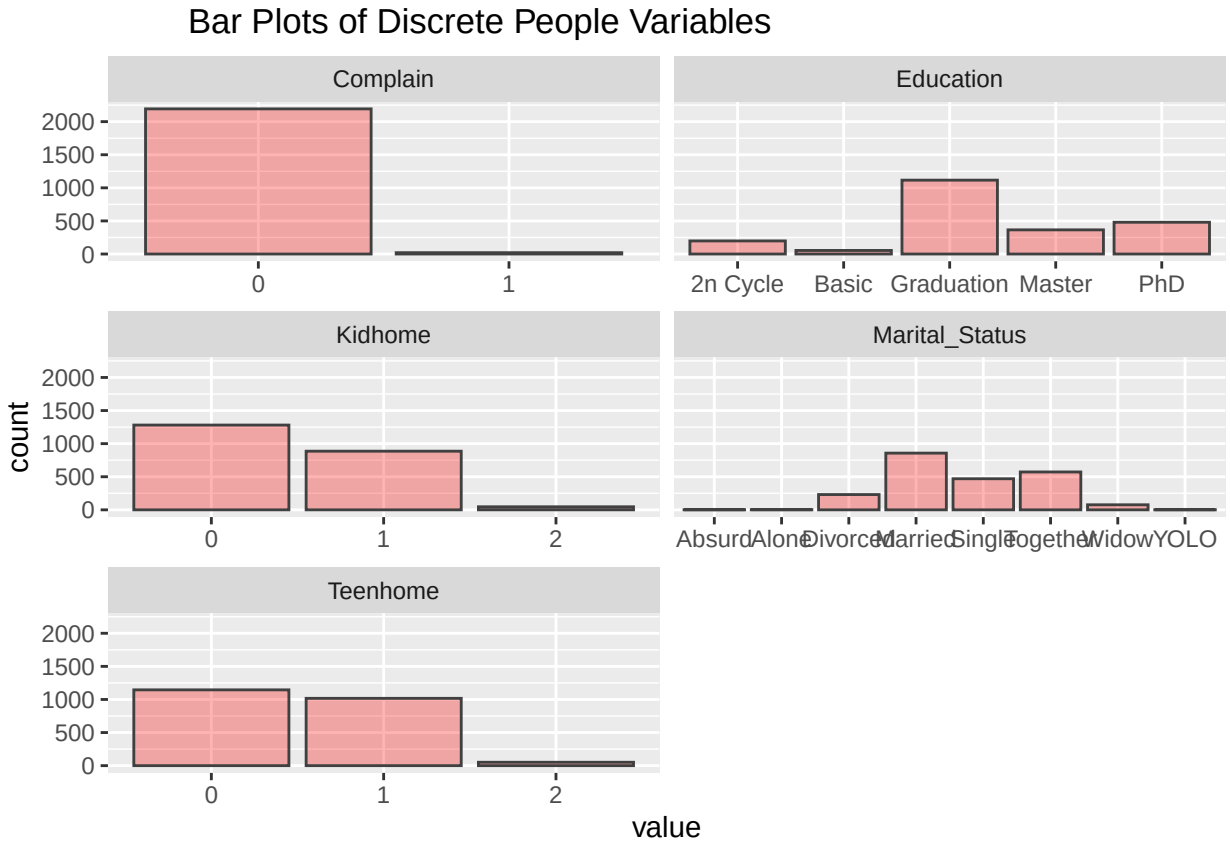
- It can be observed in Campaign Responses chart that campaigns one, three, four and five performed similarly, whereas campaign two performed poorly. The latest campaign (under variable 'Response') had the best performance.

```
# Plotting histograms for place variables
ggplot(place_key_value, aes(value)) +
geom_histogram(stat = 'count', alpha=.3, fill ="red1",
               colour = "gray25", position="identity",
               binwidth = 1) +
facet_wrap(~key, scales = 'free_x', ncol = 2) +
labs(title = "Histograms of Place Variables") +
theme(plot.title = element_text(hjust = 0.5))
```



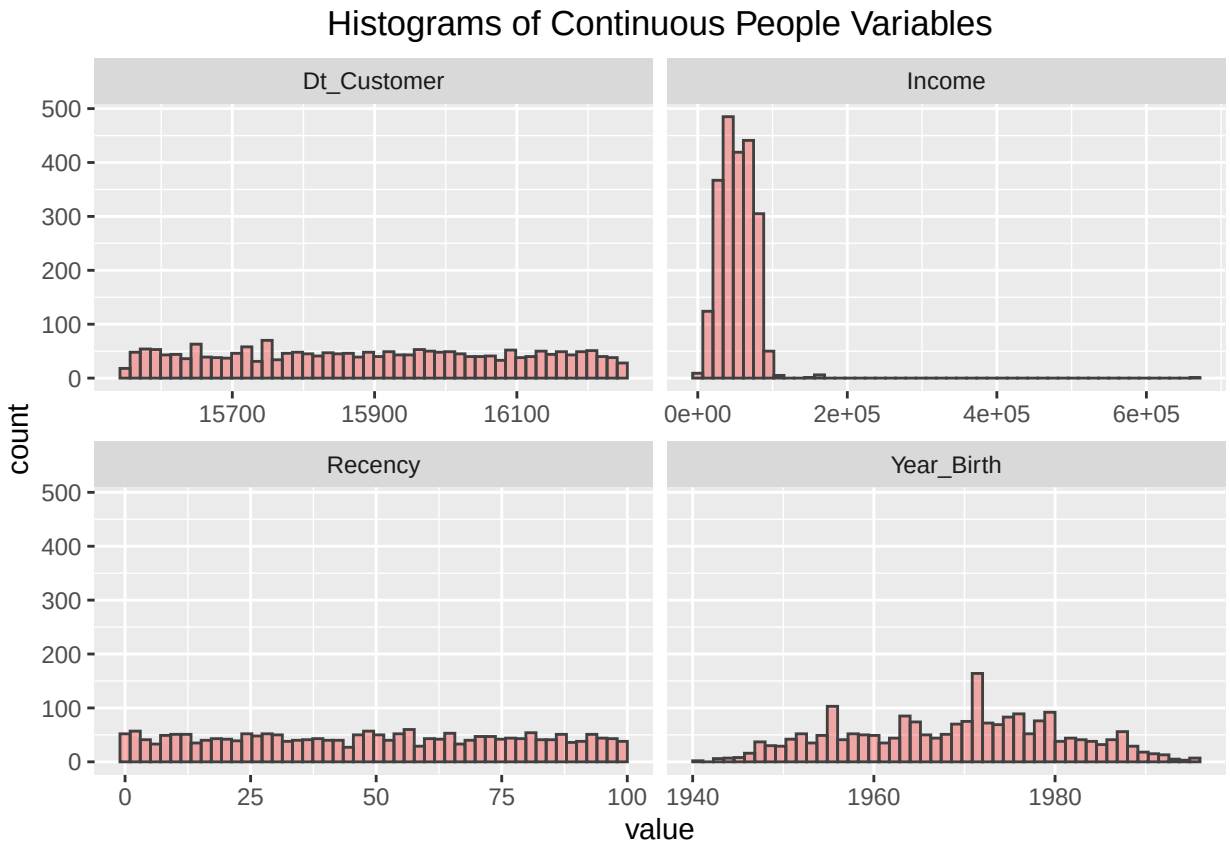
- Generally speaking, each channel performs well in the early stages, but few customers continue to use these channels to continue shopping.

```
# Ploting barplots for discrete people variables
ggplot(people_key_value_disc, aes(value))              +
geom_histogram(stat = 'count', alpha=.3, fill ="red1",
               colour = "gray25", position="identity") +
facet_wrap(~key, scales = 'free_x', ncol = 2)          +
labs(title = "Bar Plots of Discrete People Variables") +
theme(plot.title = element_text(hjust = 0.15))
```



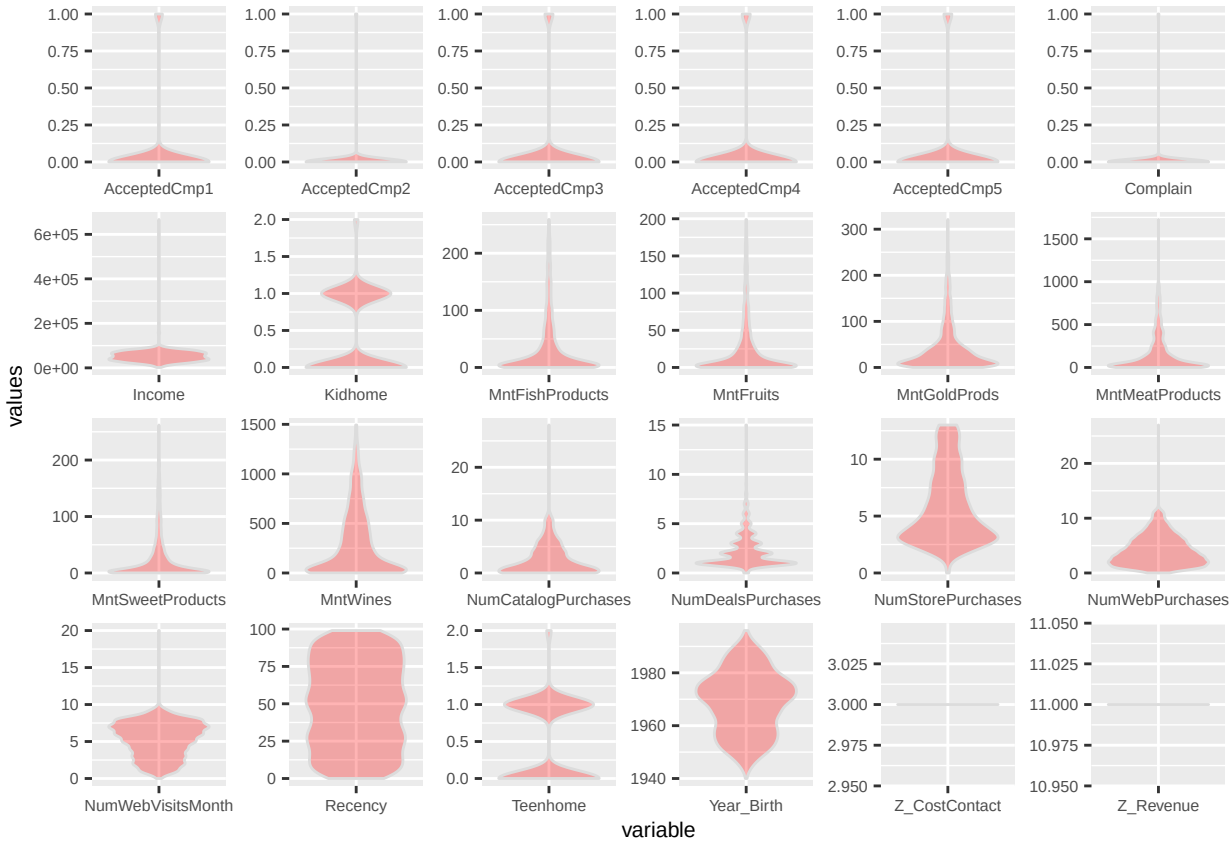
- Complain: Very few customers have filed any complaints.
- Education: The prevalent current highest level of education are Graduation, which is a little misleading as an education level, probably it refers to an "Undergraduate" education, Master degrees and Doctoral degrees.
- Marital_Status: There are three categories 'Alone', 'YOLO' and 'Absurd' with very low number of customers in each.
- Kidhome and Teenhome: The number of customers with two kid / teenager at home is very low, causing the classes to be very imbalanced.

```
# Plotting histograms for continuous people variables
ggplot(people_key_value_cont, aes(value)) +
geom_histogram(bins = 50, alpha=.3, fill ="red1",
               colour = "gray25", position="identity")    +
facet_wrap(~key, scales = 'free_x', ncol = 2) +
labs(title = "Histograms of Continuous People Variables") +
theme(plot.title = element_text(hjust = 0.5))
```



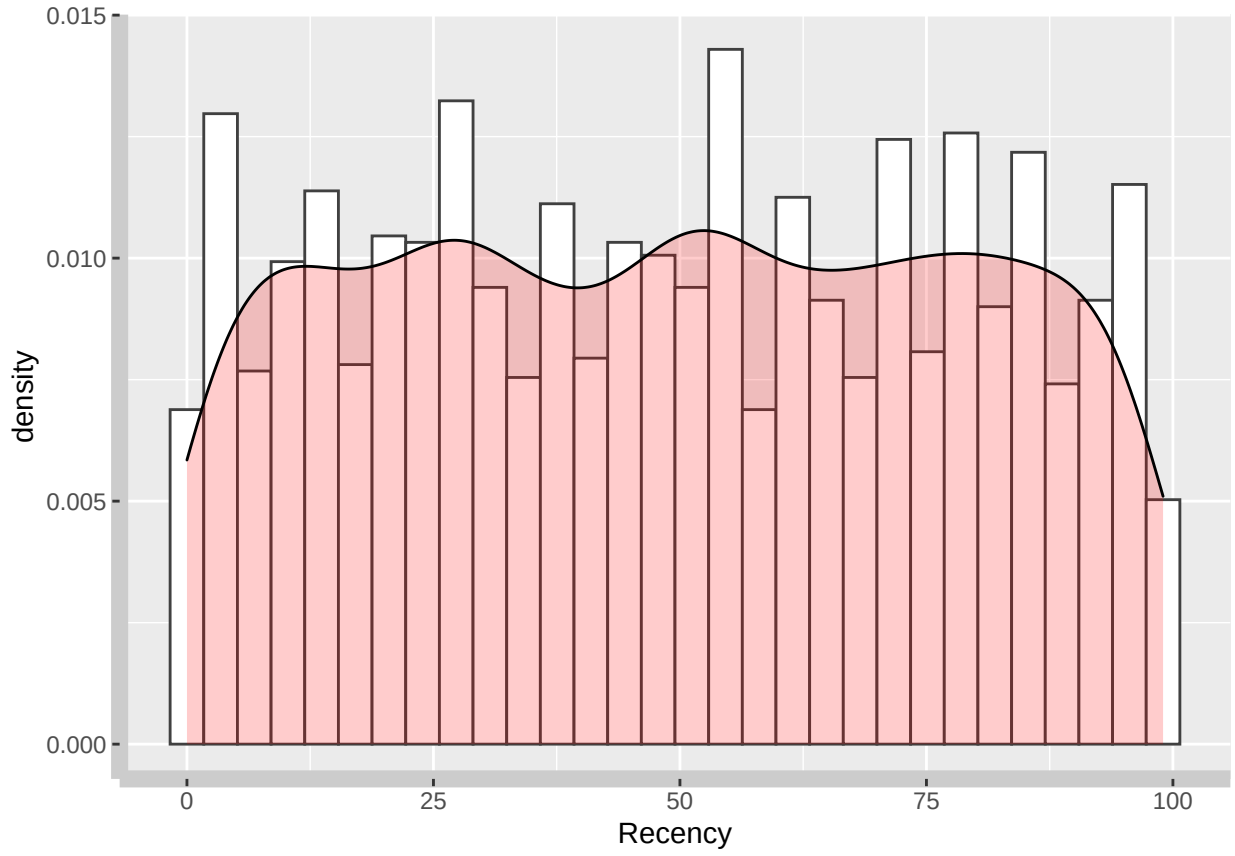
Before running a summary statistics we can actually visualize the range, central tendency and quartiles via a `geom_violin` call.

```
df_num   <-select_if(raw_ifood, is.numeric) %>% select(-ID)
df_num_p <-df_num %>% gather(variable, values, 1:24)

df_num_p %>% ggplot() +
             geom_violin(aes(x = variable, y = values),
                         alpha=.3, fill = "red1",
                         colour = "gainsboro")  +
             facet_wrap(~variable, ncol = 6, scales = "free") +
             theme(strip.text.x = element_blank(),
             text = element_text(size = 7.5))
```



- There are certain data points with high income.

```
raw_ifood %>%
              ggplot(aes(x = Recency)) +
              geom_histogram(aes(y= ..density..),
                             colour = "gray25",
                             fill = "white") +
              geom_density(alpha=.2, fill ="red1") +
              theme(axis.line = element_line(size = 3, colour = "grey80"))
```




```
raw_ifood %>%
  summarize(
    min = min(Income),
    max = max(Income)
    )
```

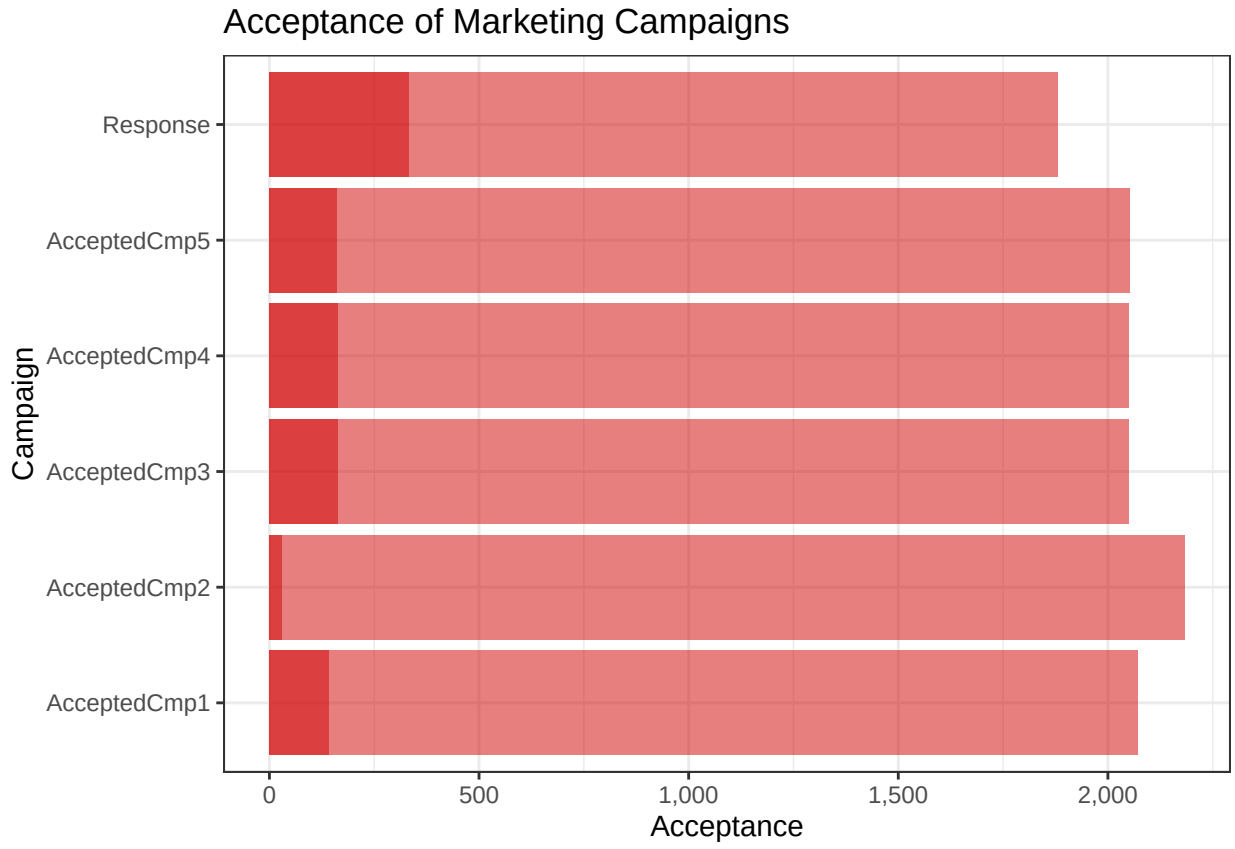
```
##    min    max
## 1 1730 666666
```

- Seems like I have customers earn a household income of more than $600,000.

6.2.5 The most successful marketing campaign

```
# Counting ones
campaign_takeup <- raw_ifood %>%
                   select('AcceptedCmp1', 'AcceptedCmp2', 'AcceptedCmp3',
                          'AcceptedCmp4','AcceptedCmp5', 'Response') %>%
                   colSums()
# Counting zeros
zero     <- function(x) sum(x == 0)
campaign <- raw_ifood %>%
            select('AcceptedCmp1', 'AcceptedCmp2', 'AcceptedCmp3',
                   'AcceptedCmp4','AcceptedCmp5', 'Response')
rechazo  <- numcolwise(zero)(campaign)
```

```
rechazo  <- as.data.frame(t(as.matrix(rechazo)))
colnames(rechazo) <- "no_aceptacion"

# Making a data frame with Acceptance and Rejected campaigns
campaign_takeup <- data.frame(campana = c('AcceptedCmp1', 'AcceptedCmp2',
                                          'AcceptedCmp3',
                                          'AcceptedCmp4',
                                          'AcceptedCmp5',
                                          'Response'),
                              aceptacion    = campaign_takeup[1:6],
                              no_aceptacion = rechazo)

campaign_takeup_long <- melt(campaign_takeup)

ggplot(campaign_takeup_long, aes(x = campana, y = value,
                                 fill = variable))+
geom_bar(stat = "identity", position= "dodge",
         fill = c("#D20000"), alpha=.50,show.legend = NA) +
theme_bw() +
labs(x = "Campaign", y = "Acceptance",
     title="Acceptance of Marketing Campaigns") +
coord_flip() +
scale_y_continuous(labels = scales::comma)
```



- From the bar plot, the most recent Marketing Campaign had the most success while Campaign2 had the least. In other words, based on the graph, we can conclude that the most recent campaign is the

most successful one. It can be observed that campaigns 1, 3, 4 and 5 performed similarly. The latest campaign (under variable 'Response') had the best performance.

6.2.6 Average customer profiling

```
average_customer_num <- raw_ifood %>%
                        select_if(names(.)=="Dt_Customer" | sapply(., is.numeric)) %>%
                        select(-ID) %>%
                        summarise_each(funs(mean)) %>%
                        t() %>%
                        as.data.frame() %>%
                        format(scientific = F, digits = 2) %>%
                        setnames("V1", "average_customer")

education      <- mlv(raw_ifood$Education, method="mfv")
status_marital <- mlv(raw_ifood$Marital_Status, method="mfv")
categ          <- data.frame(education,status_marital)

average_customer_categ <- categ %>%
                          t() %>%
                          as.data.frame() %>%
                          setnames("V1", "average_customer")


average_customer <- rbind(average_customer_num, average_customer_categ)
average_customer
```

```
##                      average_customer
## Year_Birth                   1968.917
## Income                       52236.58
## Kidhome                      0.441934
## Teenhome                    0.5056484
## Dt_Customer                2013-07-10
## Recency                      49.00768
## MntWines                     305.1536
## MntFruits                    26.32399
## MntMeatProducts              166.9625
## MntFishProducts              37.63534
## MntSweetProducts             27.03479
## MntGoldProds                 43.91143
## NumDealsPurchases             2.32535
## NumWebPurchases              4.087664
## NumCatalogPurchases          2.671487
## NumStorePurchases            5.805242
## NumWebVisitsMonth            5.321735
## AcceptedCmp3               0.07365567
## AcceptedCmp4               0.07410755
## AcceptedCmp5               0.07275192
## AcceptedCmp1               0.06416629
## AcceptedCmp2               0.01355626
## Complain                  0.009037506
## Z_CostContact                       3
## Z_Revenue                          11
```

```
## Response                   0.1504745
## Age                         52.08269
## education                 Graduation
## status_marital               Married
```

Taking the modal category of all categorical variables to obtain the average customer profile for this iFood. According to the above analysis, the average customer profile can be divided into different categories:

Demographic:

- Born between 1968 and 1969
- Income: $52.237
- 0.44 kids and 0.51 teens at home, for an average of 1 dependent at home
- Graduated
- Married

Expenditure in last two years:

- Wine: 305.153
- Fruits: 26.323
- Meat: 166.962
- Fish: 37.635
- Sweet: 27.034
- Gold: 43.911

Channels:

- Deals Purchases: 2.325
- Web Purchases: 4.087
- Catalog Purchases: 2.671
- Store Purchases: 5.805
- Number of Web Visits in past month: 5.321

Loyalty:

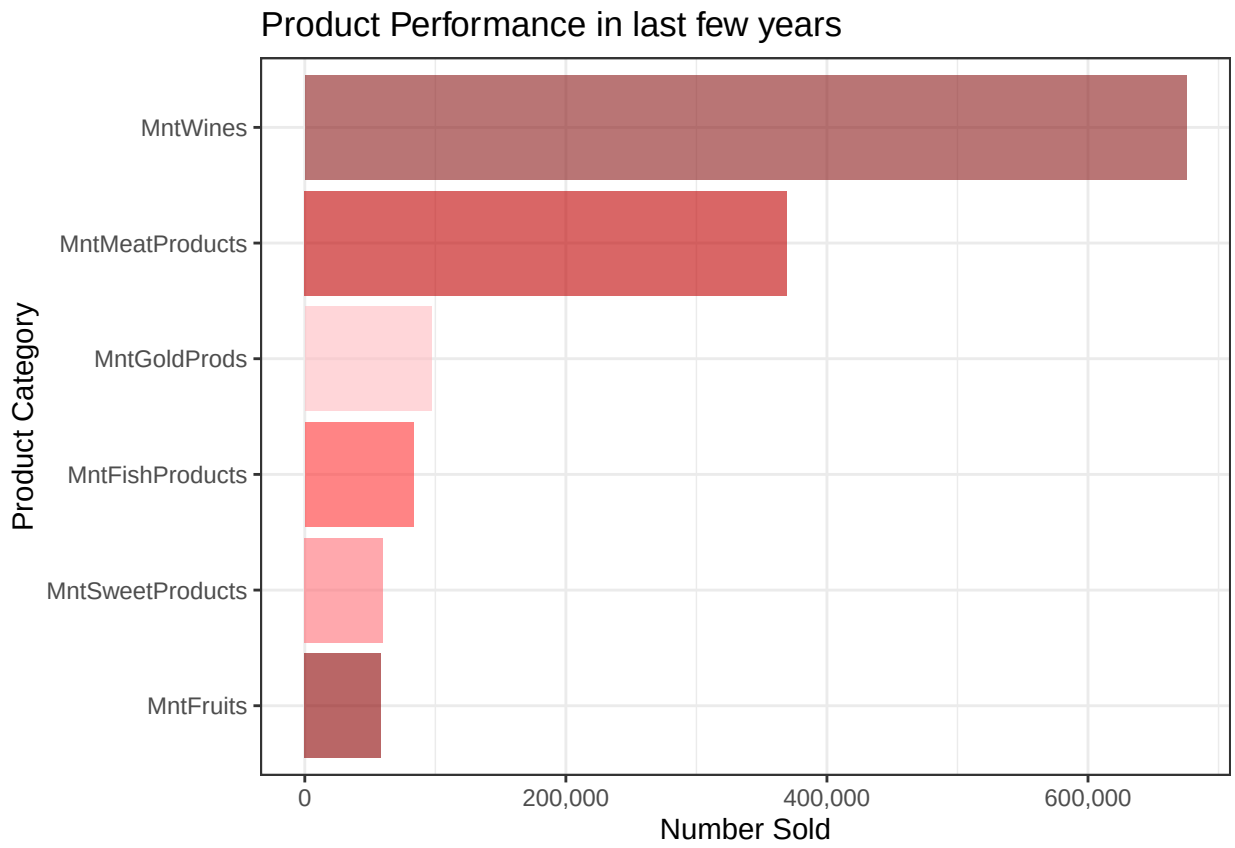
- Became a customer on 31-08-2012
- Last made a purchase 49 days ago

Interactions:

- Complains: 0.01
- Accepted latest campaign: 0.15
- Accepted Campaign 1: 0.064
- Accepted Campaign 2: 0.013
- Accepted Campaign 3: 0.073
- Accepted Campaign 4: 0.075
- Accepted Campaign 5: 0.073

6.2.7 The best performing product

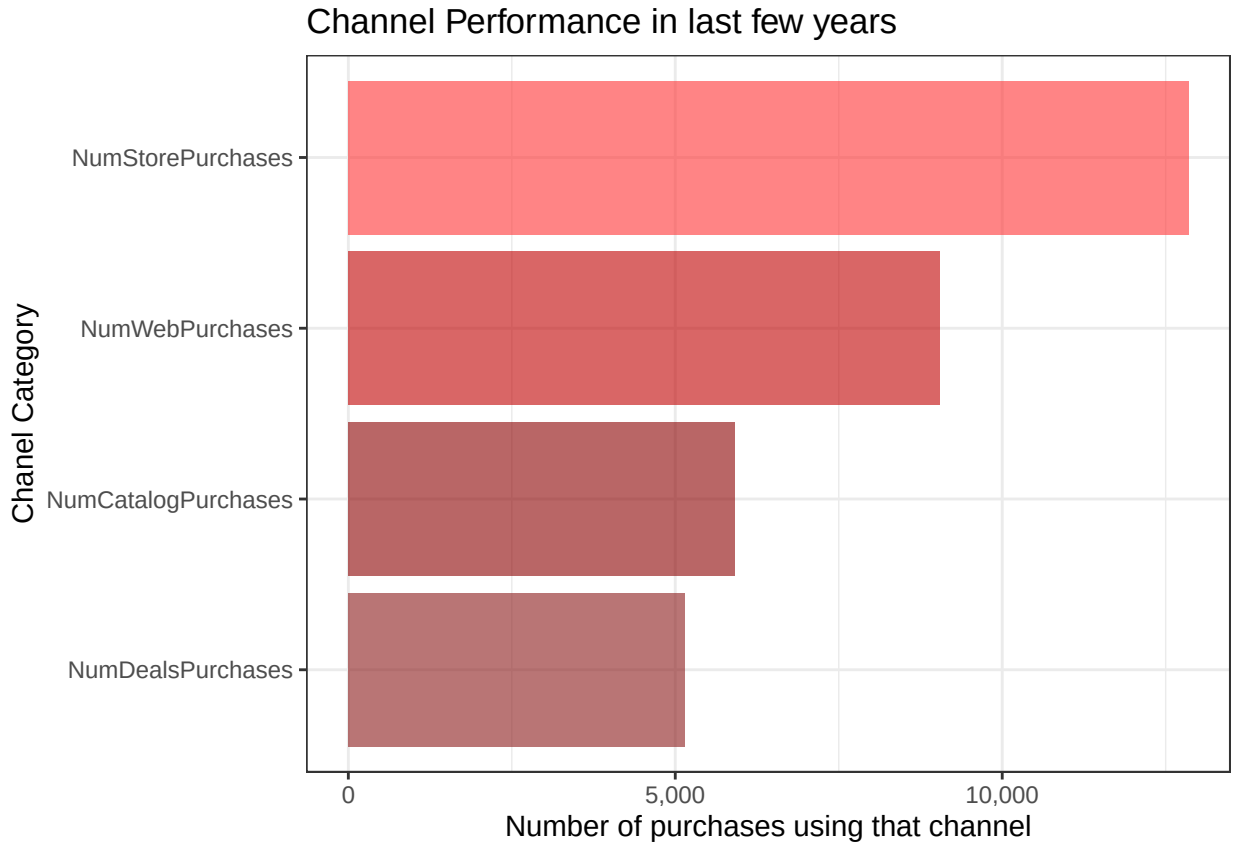
```
products <- raw_ifood %>%
            select('MntWines','MntFruits','MntMeatProducts',
                   'MntFishProducts','MntSweetProducts',
                   'MntGoldProds') %>%
            colSums()
products <- data.frame(producto = c('MntWines','MntFruits',
                                    'MntMeatProducts','MntFishProducts',
                                    'MntSweetProducts','MntGoldProds'),
                       suma_total = products[1:6])

ggplot(data = products) +
geom_bar(mapping = aes(x = reorder(producto, suma_total), y = suma_total),
         stat = "identity",
         fill = c("firebrick4","darkred","#C00000","#FF3334","#FF6F77","#FFBBC1"),
         alpha=.60) +
theme_bw() +
labs(x = "Product Category", y = "Number Sold", title="Product Performance in last few years") +
coord_flip() +
scale_y_continuous(labels = scales::comma)
```



- The best performing product is Wine followed by Meat Products. Based on the chart, we can see that wine performed the best, with the highest number of items sold, followed by meat products. On the other hand, Gold, Fish, Sweet and Fruits products are less popular, with similar number of items sold.

6.2.8 The best performing channel

```
channels <- raw_ifood %>%
            select('NumDealsPurchases', 'NumCatalogPurchases',
                   'NumWebPurchases',
                   'NumStorePurchases') %>%
            colSums()
channels <- data.frame(canal = c('NumDealsPurchases', 'NumCatalogPurchases',
                                 'NumWebPurchases',
                                 'NumStorePurchases'),
                       suma_total = channels[1:4])

ggplot(data = channels) +
geom_bar(mapping = aes(x = reorder(canal, suma_total), y = suma_total),
         stat = "identity",
         fill = c("firebrick4","darkred","#C00000","#FF3334"),
         alpha=.60) +
theme_bw() +
labs(x = "Chanel Category",y = "Number of purchases using that channel",
     title = "Channel Performance in last few years") +
coord_flip() +
scale_y_continuous(labels = scales::comma)
```



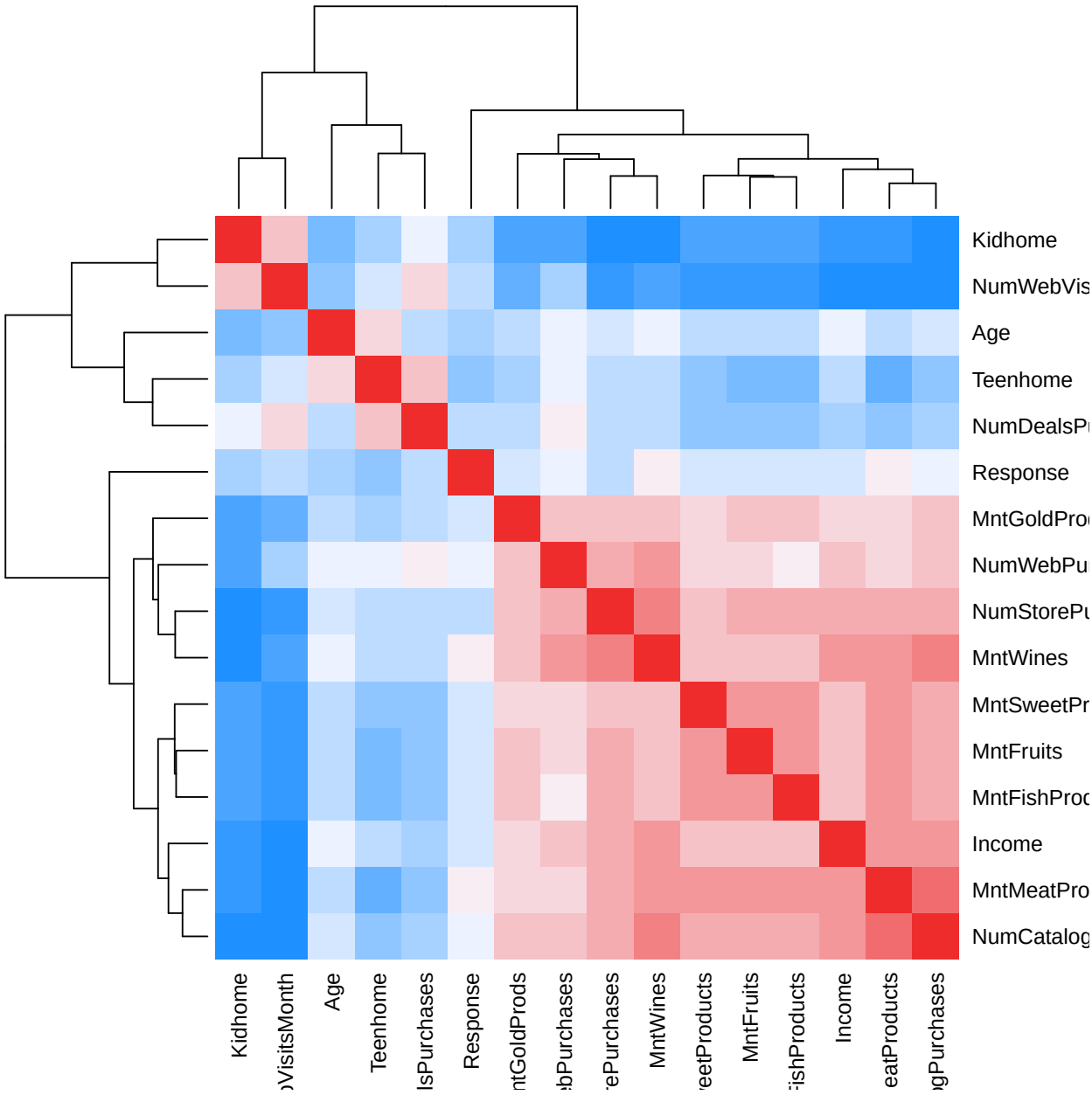
- Based on the chart, we can see that most customers preferred purchasing in physical stores, as it has the most number of items sold. This is followed by online website, catalog, and deals. Deals is the most under-performing channel.

6.3 Data exploration of dependent variables

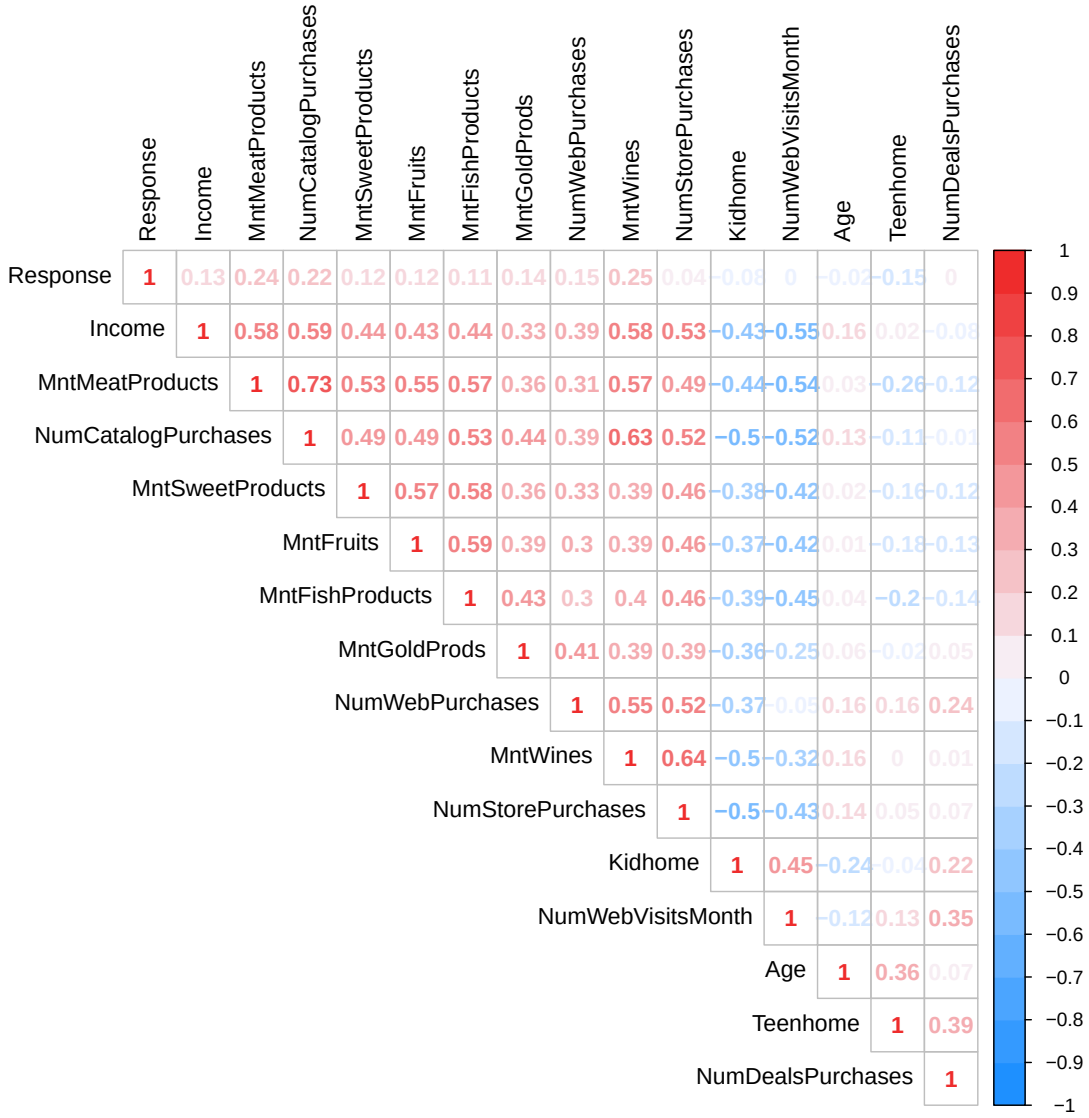
- I will use correlation matrix to find the correlations among features. The cor() method can be applied on dataframe and the results can also be visualized using a heatmap.

6.3.1 Features Correlation Analysis

```
data_numeric <- raw_ifood %>% select_if(., is.numeric) %>% select(-c("AcceptedCmp1","AcceptedCmp2",
                                                                     "AcceptedCmp3","AcceptedCmp4",
                                                                     "AcceptedCmp5","Recency",
                                                                     "Complain","ID","Z_CostContact",
                                                                     "Z_Revenue","Year_Birth"))
crr <- cor(data_numeric, use="complete.obs")

#Visualizing correlation in heatmap
colores <- colorRampPalette(c("dodgerblue", "ghostwhite", "firebrick2"))(20)
heatmap(crr, col = colores, symm=TRUE)
```




```
corrplot(corr = crr,
         method="number",
         col = colores,
         type="upper",
         tl.col="black",
         order="hclust")
```

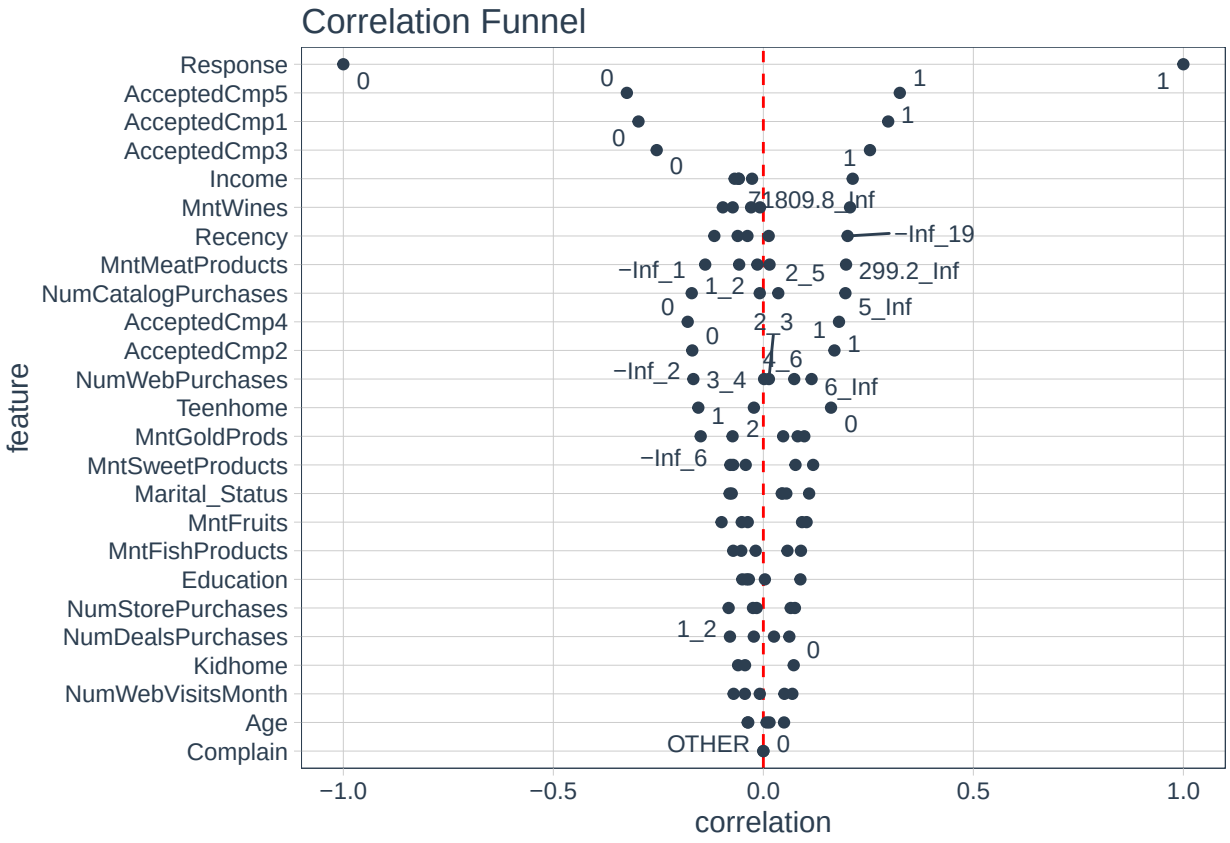


- Highly positively correlated variables are in Red and Highly negatively correlated variables are in Blue.
- We could see that income are positively correlated to Number of purchases and the amount of purchases.
- The different kind of purchases such as Meat purchases/ Sweet purchases/ Fish purchases/ Fruit purchases tend to be positively correlated to one another.
- There are negative correlations between having kids/ dependents at home and the amount or number of purchases.

6.3.2 Correlation Analysis - correlationfunnel

```
customer_ifoof_binarized <- raw_ifood %>%
                            select(-ID,-Year_Birth,-Z_CostContact,-Z_Revenue, -Dt_Customer) %>%
                            binarize(n_bins = 5, thresh_infreq = 0.01, name_infreq = "OTHER", one_hot =
```

```
customer_response_corr  <- customer_ifoof_binarized %>%
                           correlate(Response__1)

customer_response_corr %>%
  plot_correlation_funnel()
```



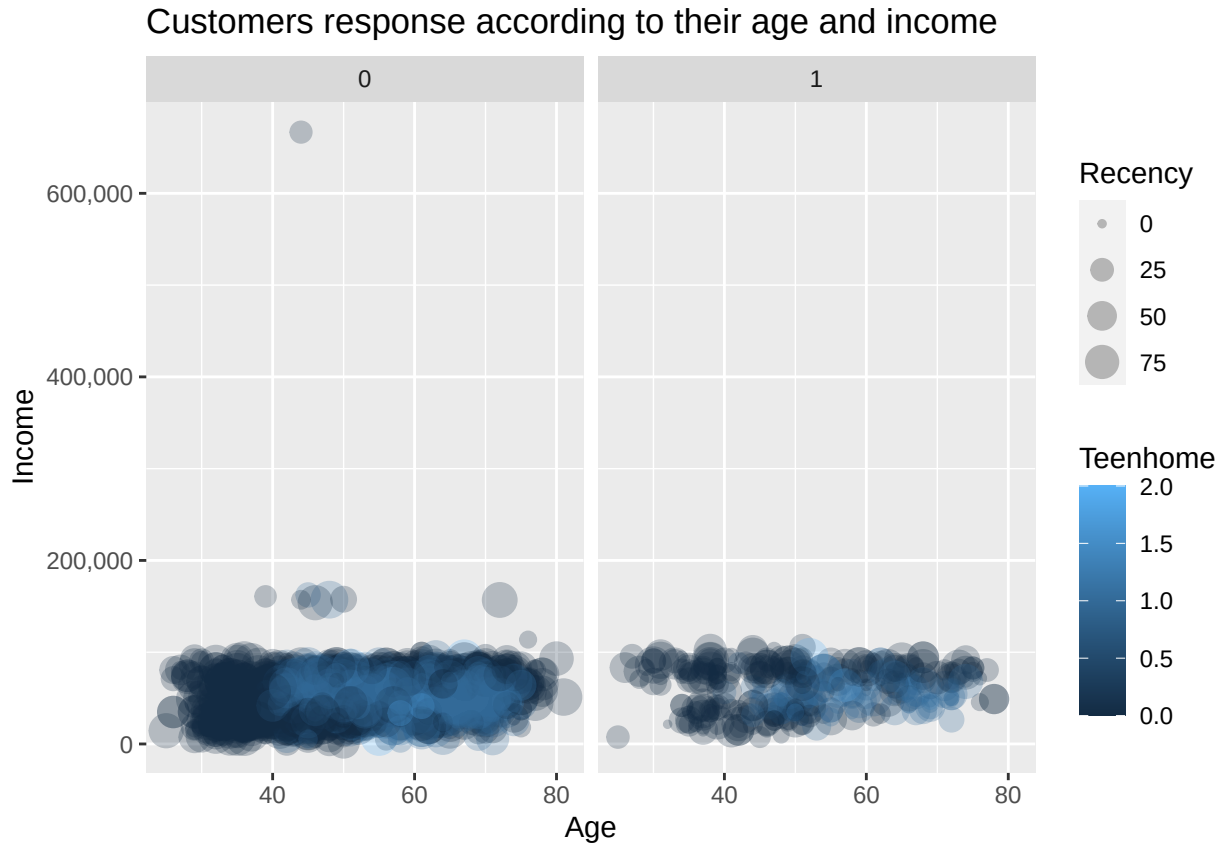
We can see that the correlation funnel graph produced uncovers insights by elevating high-correlation features and lowering low-correlation features:

- Income is a proxy for several other features, such as the amount spend, positively driven by meat and wine and it has a negative correlation with the number of kids home and the visits on the websites.
- The amount spend on wine is, besides being related to high income, to the amount spend on meat and it is purchased or in Catalog or Stores.
- The number of kids is negative related to income. Higher Income is also related to accept Campaigns.

7 Section 02: Customer segmentation

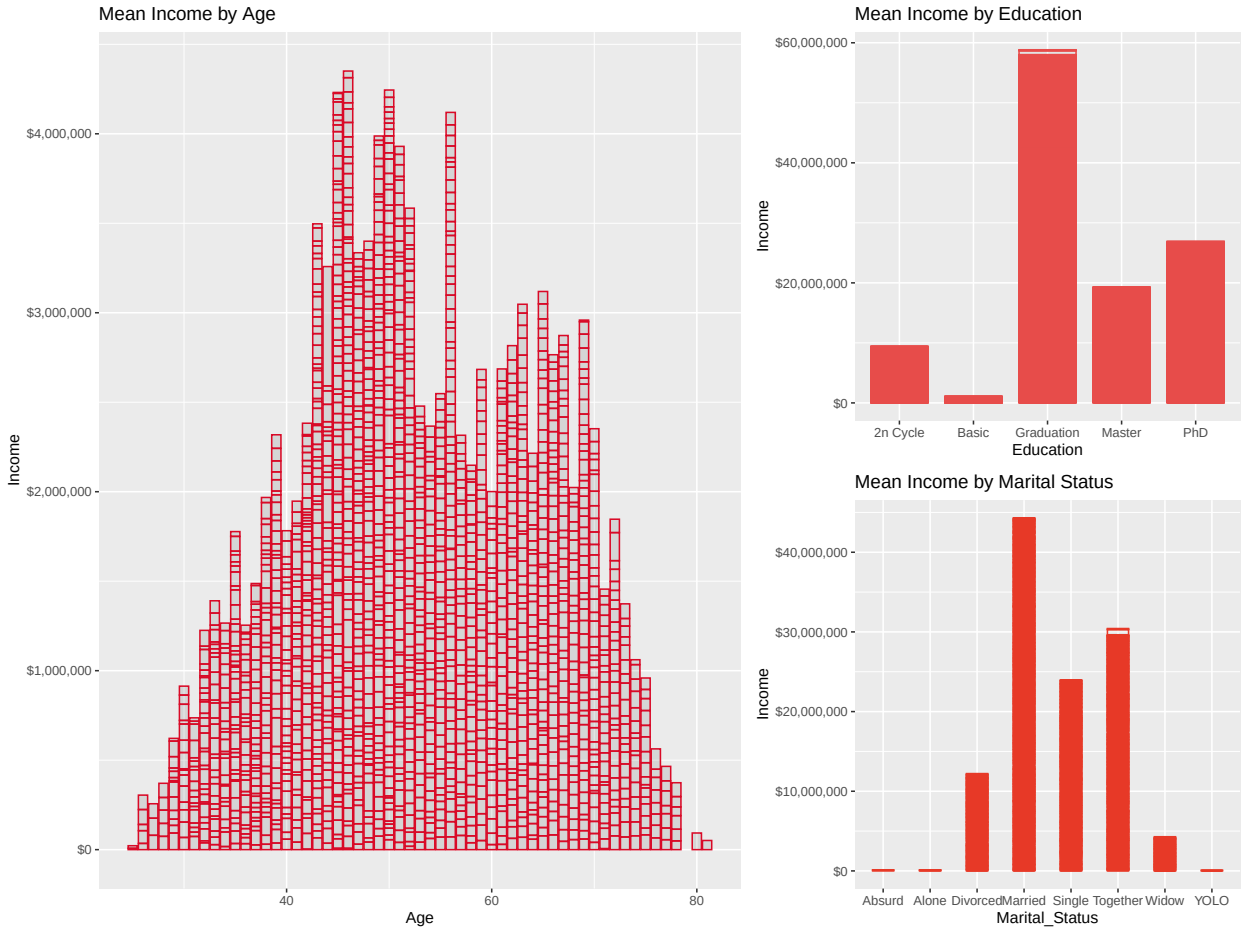
7.1 Customer Profile Analysis

```
raw_ifood %>%
  ggplot(aes(x = Age, y = Income, color = Teenhome, size = Recency)) +
  geom_point(alpha = 0.25) +
   labs(title = "Customers response according to their age and income") +
  facet_wrap(~Response) +
  scale_y_continuous(labels = scales::comma)
```



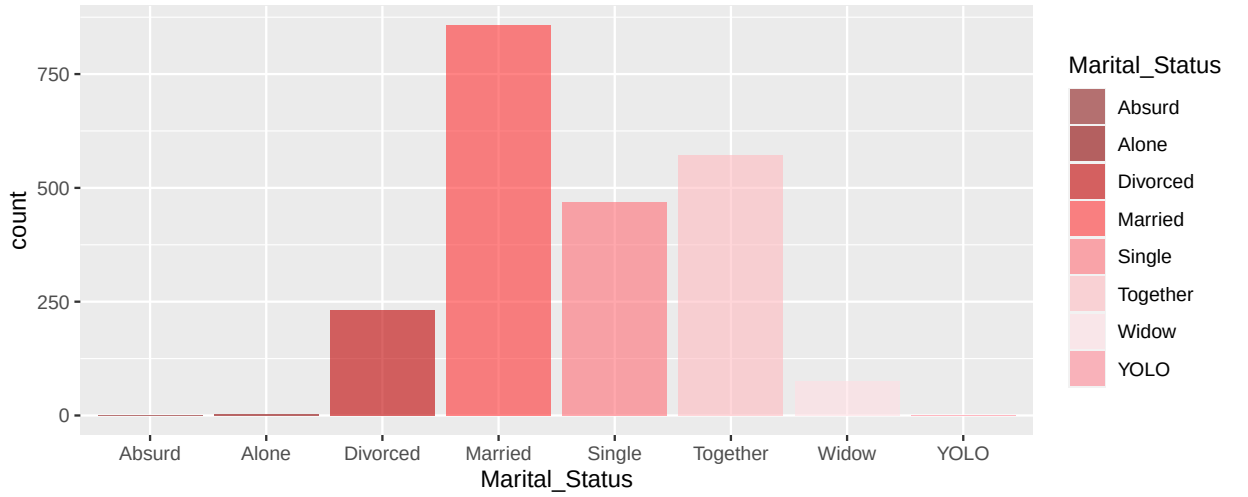

```
plot_1 <- raw_ifood %>%
          select(Income, Age) %>%
          group_by(Age) %>%
          ggplot(aes(x = Age, y = Income))   +
          geom_col(alpha=.15, colour = c("#D90A27"))  +
          scale_y_continuous(labels = scales::dollar) +
          ggtitle("Mean Income by Age") +
          theme(legend.title = element_blank())

plot_2 <- raw_ifood %>%
          select(Income, Education) %>%
          group_by(Education) %>%
          ggplot(aes(x = Education, y = Income))   +
          geom_col(alpha=.05, colour = c("#E74C4A"), width = 0.77) +
          scale_y_continuous(labels = scales::dollar) +
          ggtitle("Mean Income by Education") +
          theme(legend.title = element_blank())

plot_3 <-raw_ifood %>%
         select(Income, Marital_Status) %>%
         group_by(Marital_Status) %>%
         ggplot(aes(x = Marital_Status, y=Income))   +
         geom_col(alpha=.05, colour = c("#E73927"), width = 0.45)  +
         scale_y_continuous(labels = scales::dollar) +
         ggtitle("Mean Income by Marital Status") +
         theme(legend.title = element_blank())
```

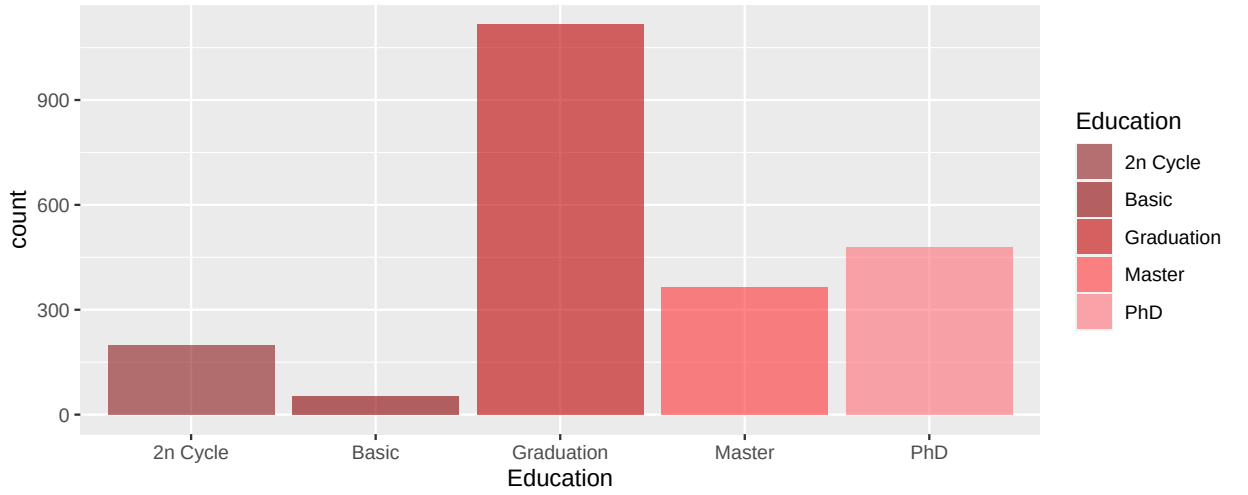
```
gs <- lapply(1:9, function(ii)
grobTree(rectGrob(gp=gpar(fill=ii, alpha=0.5)), textGrob(ii)))

lay <- rbind(c(1,1,1,2,2),
             c(1,1,1,3,3))
grid.arrange(plot_1, plot_2, plot_3, layout_matrix = lay)
```

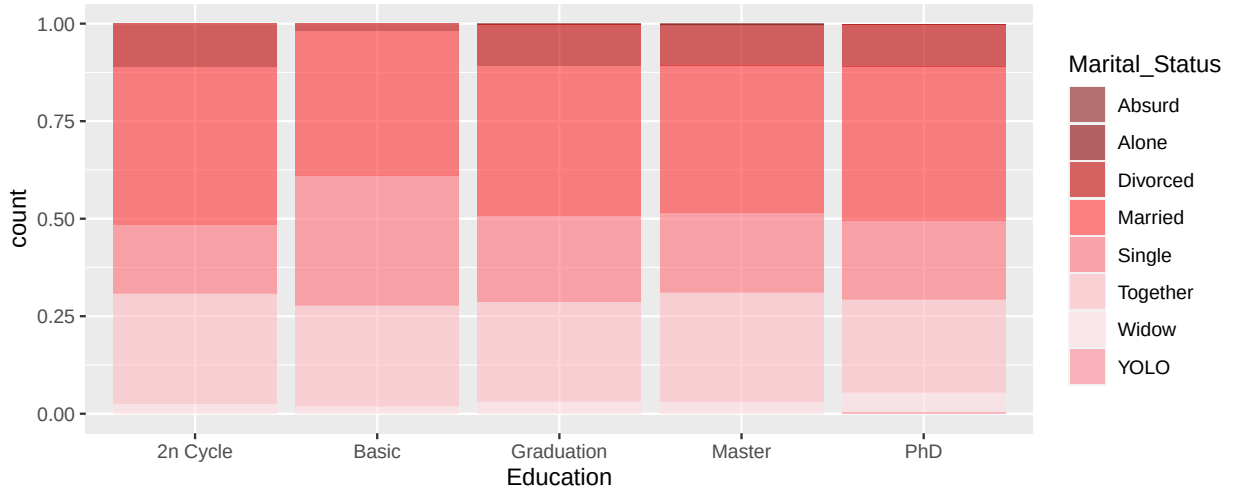



```
graph_1 <- ggplot(raw_ifood, aes(x = Marital_Status, fill= Marital_Status)) +
           geom_bar(alpha=.60)                                               +
           scale_fill_manual(values = c("firebrick4","darkred","#C00000",
                                        "#FF3334","#FF6F77","#FFBBC1","#FFDEE3",
                                        "#FF8896"))

graph_2 <- ggplot(raw_ifood, aes(x = Education, fill= Education)) +
           geom_bar(alpha=.60)                                    +
           scale_fill_manual(values = c("firebrick4","darkred","#C00000","#FF3334",
                                        "#FF6F77","#FFBBC1","#FFDEE3","#FF8896"))

graph_3 <- ggplot(raw_ifood, aes(x = Education, fill= Marital_Status)) +
           geom_bar(position = position_fill(), alpha=.60) +
           scale_fill_manual(values = c("firebrick4","darkred","#C00000","#FF3334",
                                        "#FF6F77","#FFBBC1","#FFDEE3","#FF8896"))
```

```
grid.arrange(graph_1, graph_2, graph_3, nrow = 3,
             bottom = textGrob("Customer Profile Information",
             gp = gpar(fontface = 3, fontsize = 9),
             hjust = 1,
             x = 1))
```







Customer Profile Information

7.2 Statistical Clustering - K-Means

The segmentation will be performed using K-Means clustering, which is a simple and elegant way of subsetting the customers into non-overlapping segments.

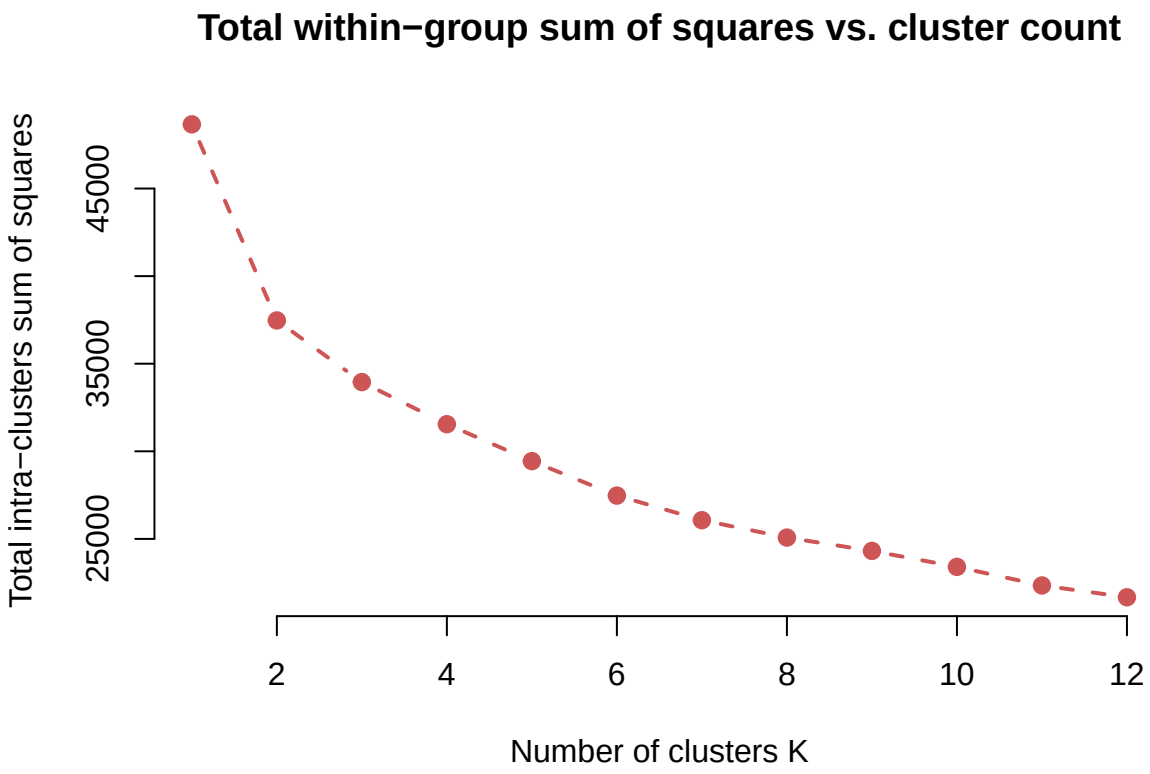
- I specify the number of clusters that I need to create.
- The algorithm selects k objects at random from the data set. This object is the initial cluster or mean.
- The closest centroid obtains the assignment of a new observation. I base this assignment on the Euclidean Distance between object and the centroid.

```
ifood_df_clustering = raw_ifood %>%
                      select(-ID,-Education,-Marital_Status,-Dt_Customer,
                             -Year_Birth,-Z_CostContact,-Z_Revenue,-Response)
```

- Importance of Scaling the Data before Performing K-Means: In the iFood data, the variables are measured in different units, where a unit increase or decrease in one day for the Recency (days inactive) is completely different than a unit increase or decrease in dollars for the Income feature. Therefore the importance of scaling the data, to represent the true distance among variables. The data has been scaled using the function scale() in the k-means algorithm.

7.2.1 Elbow method

```
# function to calculate total intra-cluster sum of square (euclidean distance)
set.seed(20)
ics <- function(k){
  kmeans(scale(ifood_df_clustering), k, iter.max = 100,
         nstart = 100, algorithm = "Lloyd")$tot.withinss
}

k_values   <- 1:12
ics_values <- map_dbl(k_values, ics)

plot(k_values, ics_values,
     type = "b", pch = 19, frame = FALSE, col="indianred3",
     cex=1, lwd = 2, lty = 2,
     xlab = "Number of clusters K",
     ylab = "Total intra-clusters sum of squares",
     main = 'Total within-group sum of squares vs. cluster count')
```



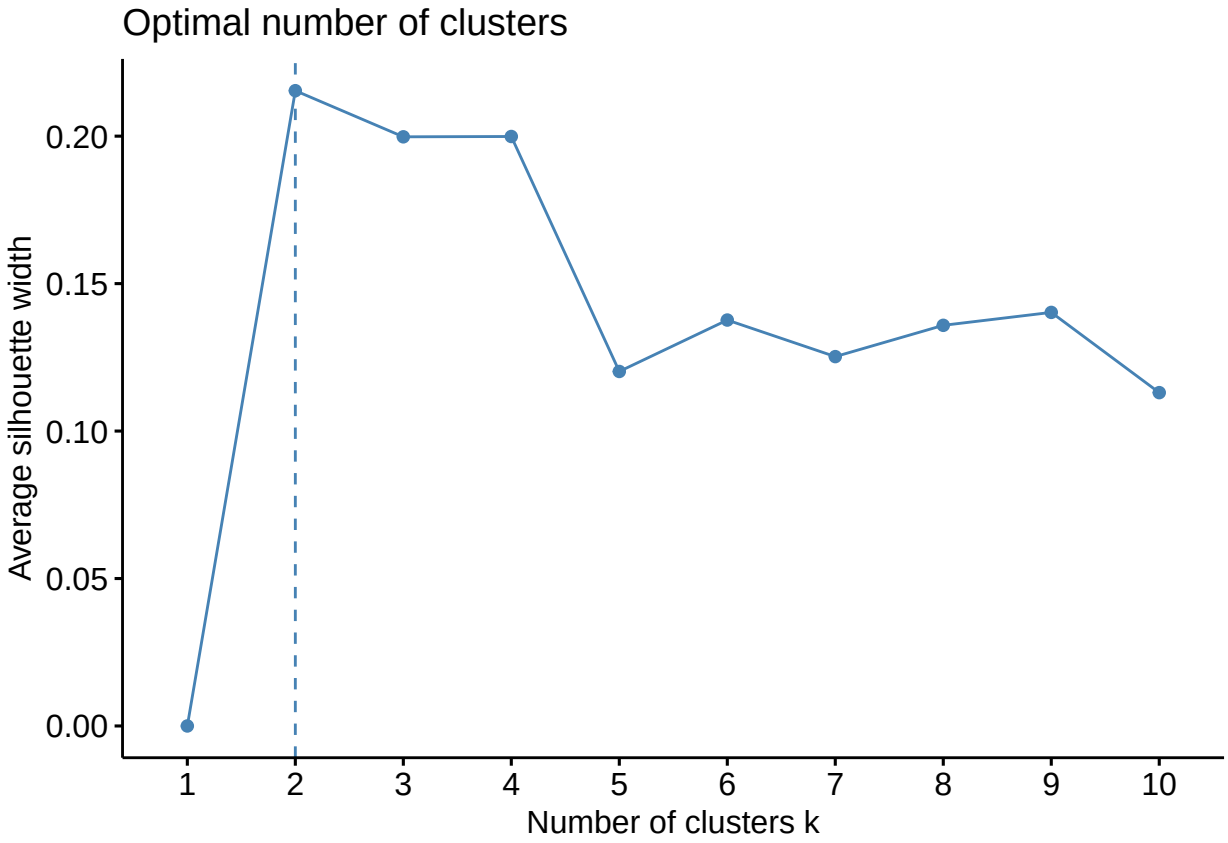
- Fitting k-means to the data set with K = 2 or 3. From the above graph, we conclude that 4 is the appropriate number of clusters since it seems to be appearing at the bend in the elbow plot.

7.2.2 Silhouette method

```
# average Silhouette method
k2 <-  kmeans(scale(ifood_df_clustering), 2,  iter.max = 100,
              nstart = 50, algorithm = "Lloyd")
k3 <-  kmeans(scale(ifood_df_clustering), 3,  iter.max = 100,
              nstart = 50, algorithm = "Lloyd")
k4 <-  kmeans(scale(ifood_df_clustering), 4,  iter.max = 100,
              nstart = 50, algorithm = "Lloyd")
k5 <-  kmeans(scale(ifood_df_clustering), 5,  iter.max = 100,
              nstart = 50, algorithm = "Lloyd")
k6 <-  kmeans(scale(ifood_df_clustering), 6,  iter.max = 100,
              nstart = 50, algorithm = "Lloyd")
k7 <-  kmeans(scale(ifood_df_clustering), 7,  iter.max = 100,
              nstart = 50, algorithm = "Lloyd")
k8 <-  kmeans(scale(ifood_df_clustering), 8,  iter.max = 100,
              nstart = 50, algorithm = "Lloyd")
k9 <-  kmeans(scale(ifood_df_clustering), 9,  iter.max = 100,
              nstart = 50, algorithm = "Lloyd")
k10 <- kmeans(scale(ifood_df_clustering), 10, iter.max = 100,
              nstart = 50, algorithm = "Lloyd")
```

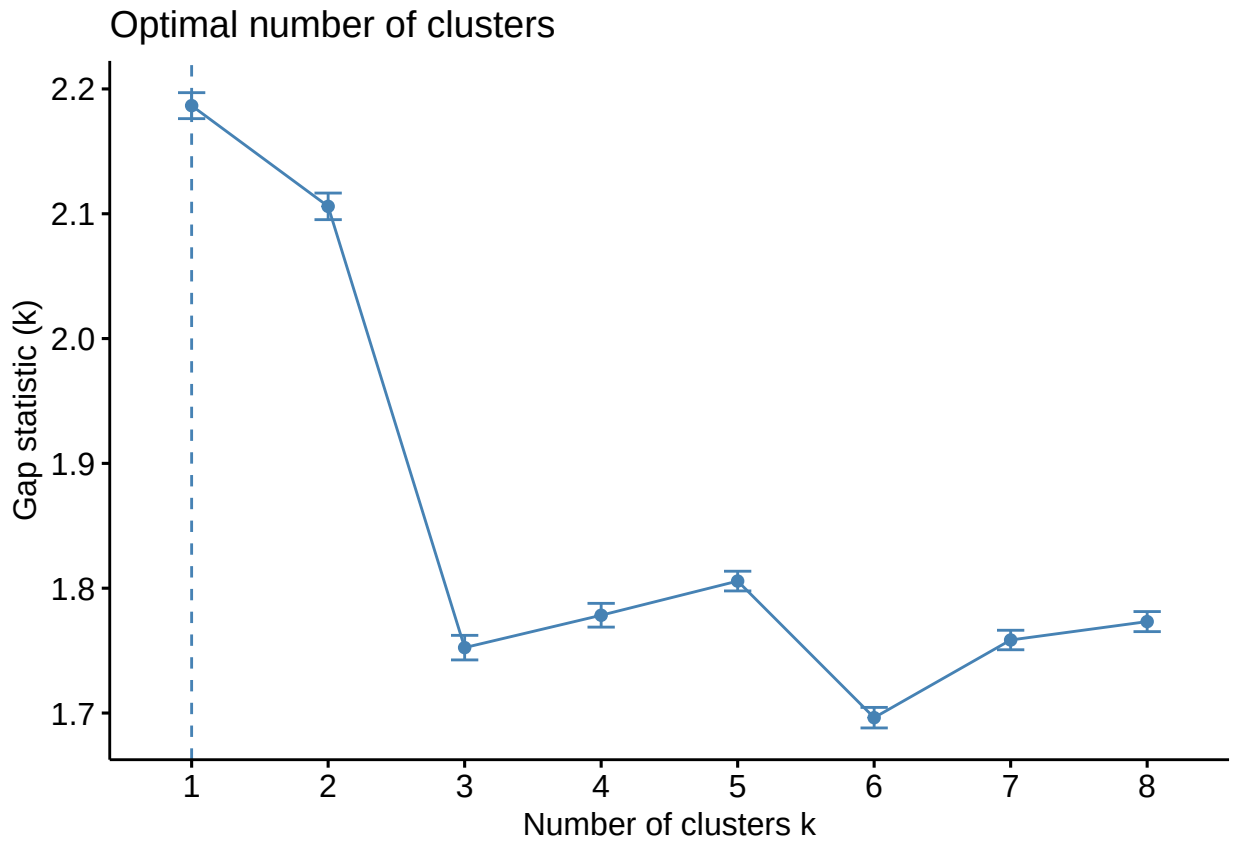
Now, I make use of the `fviz_nbclust()` function to determine and visualize the optimal number of clusters

```
fviz_nbclust(scale(ifood_df_clustering), kmeans, method = "silhouette")
```



7.2.3 Gap statistic method

```
set.seed(125)
stat_gap <- clusGap(ifood_df_clustering, FUN = kmeans, nstart = 25,
            K.max = 8, B = 60)
fviz_gap_stat(stat_gap)
```



Now, considering previous analysis, lets take K = 2 as optimal cluster

```
k2 <- kmeans(scale(ifood_df_clustering), 2,  iter.max = 100,
             nstart = 50, algorithm = "Lloyd")
k2
```

```
## K-means clustering with 2 clusters of sizes 892, 1321
##
## Cluster means:
##       Income    Kidhome    Teenhome      Recency   MntWines  MntFruits
## 1  0.7862173 -0.7019299 -0.14015279  0.014027958  0.8896340  0.7387051
## 2 -0.5308901  0.4739754  0.09463761 -0.009472323 -0.6007218 -0.4988077
##   MntMeatProducts MntFishProducts MntSweetProducts MntGoldProds
## 1       0.8586054       0.7577719        0.7402808    0.6194357
## 2      -0.5797699      -0.5116824       -0.4998717   -0.4182715
##   NumDealsPurchases NumWebPurchases NumCatalogPurchases NumStorePurchases
## 1        -0.1626575       0.6202264           0.9061898         0.8465529
## 2         0.1098338      -0.4188054          -0.6119011        -0.5716315
##   NumWebVisitsMonth AcceptedCmp3 AcceptedCmp4 AcceptedCmp5 AcceptedCmp1
## 1        -0.6587462   0.03131983    0.2220547    0.4147287    0.3556812
## 2         0.4448158  -0.02114859   -0.1499416   -0.2800439   -0.2401723
##   AcceptedCmp2     Complain        Age
## 1    0.1541848 -0.012571460  0.1564915
## 2   -0.1041127  0.008488828 -0.1056702
##
## Clustering vector:
```

```
##    [1] 1 2 1 2 2 1 1 2 2 2 2 1 2 2 1 2 2 1 2 2 1 2 2 2 2 2 2 1 2 2 2 2 1 2 1 2 2
##   [38] 1 1 2 2 2 1 2 2 1 1 1 2 1 2 1 1 2 2 1 2 1 2 1 2 2 1 1 1 1 1 2 2 2 1 1 2 2
##   [75] 2 2 2 2 1 2 2 2 1 2 2 2 2 2 2 1 2 2 2 1 1 1 2 2 1 2 1 1 1 2 1 2 2 1 1 2 2
##  [112] 1 2 2 2 1 1 1 2 2 1 2 1 2 1 2 2 2 2 1 1 1 1 2 2 2 2 2 2 2 2 2 2 1 1 2 2 2
##  [149] 1 2 1 2 1 1 2 1 2 1 2 2 2 2 2 2 1 1 2 2 1 2 2 1 2 2 2 2 2 2 1 2 2 1 2 2 2 1
##  [186] 1 1 2 2 1 1 1 2 2 2 2 2 1 2 1 2 2 1 2 2 1 2 1 2 1 1 2 1 2 2 2 1 2 2 1 2 2
##  [223] 1 2 2 1 2 1 1 2 1 1 2 1 1 1 1 2 2 1 2 1 2 1 2 2 2 2 1 2 2 2 2 1 2 1 2 1 2
##  [260] 2 2 2 1 1 1 1 1 2 2 2 2 2 2 2 1 1 1 2 2 2 1 2 2 1 2 2 1 1 2 1 2 2 2 1 2 1 2
##  [297] 2 2 2 1 2 2 2 2 2 2 1 2 2 1 2 1 2 2 2 2 2 2 1 2 2 1 1 2 1 1 2 1 2 2 1 2 1 2
##  [334] 2 1 1 1 1 1 2 2 1 1 2 1 2 2 2 2 1 1 2 1 2 2 2 2 1 2 2 2 2 1 2 2 2 2 2 2 2
##  [371] 2 1 2 1 1 2 1 2 1 1 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 1 1 2 1 1 2 1 1 2 2 2 2
##  [408] 1 1 2 1 1 2 1 1 2 1 1 2 2 1 2 2 2 2 2 2 2 2 2 2 1 2 2 1 1 2 2 1 2 1 2 2 1
##  [445] 1 1 2 1 2 1 1 2 2 2 1 2 1 2 2 1 2 2 2 2 2 2 2 2 2 1 1 1 1 2 2 1 2 1 2 1 1 2
##  [482] 1 1 1 2 2 2 1 2 1 1 1 2 1 2 1 2 1 2 1 2 2 1 1 2 1 2 1 2 2 1 2 1 2 1 1 2 2
##  [519] 2 2 2 1 2 2 2 2 2 2 1 1 2 1 2 2 2 2 2 2 2 2 1 2 1 1 2 1 2 1 1 1 2 2 1 2 2 2 2
##  [556] 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 1 1 2 2 2 2 1 1 2 1 2 2 2 2 2 2 1 1 2 2 2 2 2
##  [593] 2 1 2 2 2 2 2 1 2 2 2 2 2 2 2 1 2 1 2 1 1 2 2 2 1 1 1 2 1 2 1 1 1 1 1 1 1 2 1
##  [630] 2 1 2 1 1 1 2 1 2 2 2 2 2 1 2 2 2 2 1 2 2 2 2 2 2 2 2 1 2 1 1 2 2 1 1 2 1 1 1
##  [667] 2 2 1 1 1 1 1 1 1 2 1 2 2 2 2 2 2 2 1 1 1 1 1 1 2 1 2 2 2 2 2 2 1 2 1 2 1 1 2 1
##  [704] 2 1 1 2 1 2 2 1 1 2 1 2 2 2 2 2 1 1 1 2 2 1 1 2 2 2 1 1 2 1 2 1 1 2 1 1 1 1
##  [741] 1 1 2 2 2 2 1 2 1 2 1 1 2 2 1 1 2 2 2 2 2 2 1 2 1 1 2 2 2 2 2 2 2 1 2 2 1 1 2
##  [778] 2 2 2 1 1 1 2 2 2 2 1 1 2 2 2 2 2 1 2 2 2 1 1 1 2 2 1 1 2 1 2 1 2 1 2 1 1 1 2 1 2
##  [815] 2 1 2 2 2 2 1 2 1 2 1 2 2 2 2 2 1 1 1 1 2 2 2 2 2 1 2 2 1 2 2 2 2 1 2 2 2 2 2 2 2
##  [852] 2 1 2 1 1 2 2 2 1 1 2 2 1 2 2 2 2 2 2 1 1 2 2 2 2 1 2 2 1 2 1 2 1 1 1 1 2 2 1 2 1 2
##  [889] 2 1 1 2 2 2 1 1 1 2 1 1 1 1 2 1 2 1 2 2 1 2 1 1 1 1 1 2 1 2 1 2 1 1 1 1 1
##  [926] 1 1 1 2 1 1 2 2 1 2 2 2 2 2 2 2 1 1 2 2 1 2 2 2 2 2 1 1 2 2 2 1 2 2 1 1 1 1 2
##  [963] 2 1 2 2 2 2 1 1 2 1 1 1 2 1 2 2 1 2 2 1 2 1 2 1 2 1 1 1 2 2 1 1 1 2 2 1 2 2 2 2
## [1000] 1 1 2 2 2 2 2 2 1 2 2 2 1 2 2 2 2 1 1 1 1 2 1 2 2 2 2 2 1 1 2 2 1 2 2 2 2 1 2 1 1 2
## [1037] 1 2 2 1 2 2 2 1 1 2 2 1 1 2 2 2 2 1 1 2 1 2 2 1 1 2 2 2 1 1 2 2 2 1 2 1 2 1 1 2 1 2
## [1074] 1 1 2 1 2 2 2 2 1 1 2 2 1 1 2 2 2 2 2 1 2 2 2 2 2 1 1 2 1 2 1 2 2 2 2 2 1 2 2 2 2 2
## [1111] 1 2 2 2 1 1 2 2 2 1 1 2 2 2 1 2 2 2 1 2 2 2 2 2 2 2 1 1 2 1 2 2 2 2 1 1 1 2 2 2
## [1148] 1 2 1 2 2 2 1 1 2 2 2 1 2 2 2 2 2 2 2 1 2 2 2 1 2 2 2 2 2 1 2 2 2 1 1 2 2 2 1 2 1 1 1
## [1185] 2 1 2 2 2 1 2 1 2 2 2 2 2 1 1 1 2 2 2 1 2 1 2 1 2 2 2 2 1 2 2 2 1 1 2 2 2 2 2 2 2 2 2 1 2
## [1222] 1 2 2 2 2 2 1 1 2 2 2 2 2 2 1 1 1 1 1 1 2 1 1 2 1 2 1 2 1 1 2 2 1 2 2 2 2 1 1 1 2 2
## [1259] 2 1 2 2 2 1 2 2 1 1 2 2 2 2 2 2 2 1 1 2 2 2 2 2 2 2 2 1 1 2 2 1 2 2 2 1 1 2 1 1 1 1
## [1296] 1 2 1 2 2 2 2 2 2 2 1 2 1 2 2 2 1 2 2 2 1 2 2 2 1 1 1 2 1 2 2 2 2 2 2 2 2 2 2 2 2 2
## [1333] 1 1 1 1 2 2 2 1 1 2 2 1 1 2 2 2 2 1 2 1 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 1 2 2
## [1370] 1 2 2 2 2 2 2 1 2 2 2 1 2 2 1 2 2 2 2 2 2 1 2 2 2 1 1 2 1 2 2 2 2 2 2 2 2 2 1 1 2 2 2
## [1407] 2 2 2 1 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 1 1 2 2 1 1 1 1 2 1 1 2 2 1 2 2 2 1 1 2 1 2
## [1444] 2 2 1 2 1 2 2 1 2 2 2 2 1 1 2 1 2 2 2 1 1 1 2 2 2 1 1 1 2 1 2 1 2 1 2 2 2 1 2 2 2 1 1
## [1481] 2 2 2 2 2 1 1 2 1 1 1 1 2 1 1 2 2 2 1 2 2 2 2 2 1 1 2 2 2 2 2 1 2 1 2 1 2 1 2 2 2 2
## [1518] 1 1 1 2 1 1 2 2 2 2 2 2 1 2 2 2 1 1 1 1 2 2 2 2 2 1 2 2 2 2 1 2 2 2 1 1 2 2 1 2 1 2 1 2
## [1555] 2 2 2 1 2 2 2 2 1 1 2 1 2 2 1 2 1 2 1 2 2 2 2 2 2 2 1 2 2 2 1 1 2 2 2 1 2 2 2 2 1
## [1592] 2 1 2 2 2 1 2 2 2 2 1 2 2 2 1 2 2 2 2 1 1 2 2 2 1 2 2 2 2 2 2 2 2 1 2 2 2 2 1 2 2 2 1
## [1629] 1 2 2 2 1 2 1 2 2 2 2 1 1 1 2 2 2 1 2 2 2 2 1 2 1 1 2 1 1 1 1 2 2 2 2 2 2 1 2 2 2 2
## [1666] 2 1 1 1 1 2 1 2 2 2 2 1 2 1 2 1 1 2 2 2 2 2 2 1 2 1 2 1 1 2 1 2 2 1 2 2 1 2 1
## [1703] 1 1 2 2 2 2 2 1 1 2 2 2 2 1 1 1 2 1 2 2 2 2 2 2 1 2 1 2 1 1 1 1 2 2 2 2 2 2 2 2 2
## [1740] 1 1 2 1 1 2 1 2 1 2 2 2 2 2 1 1 2 2 2 2 2 2 1 2 2 1 1 2 2 2 2 2 1 1 2 2 1 2 2 2
## [1777] 2 2 1 1 1 1 2 2 2 1 1 2 1 1 2 1 1 1 1 2 1 1 2 2 1 2 2 1 1 2 2 1 1 2 2 2 1
## [1814] 2 2 2 1 2 1 1 2 1 2 1 2 1 2 2 2 2 2 1 2 1 1 1 1 2 2 1 2 1 2 2 2 1 1 2 2 2 2 1
## [1851] 2 1 2 2 2 2 2 1 1 1 1 1 2 2 1 2 2 2 2 1 1 2 1 1 2 1 1 2 2 1 1 2 2 2 2 2 2 2 1 1
## [1888] 2 2 2 1 1 1 1 1 1 2 2 2 2 2 2 1 1 1 1 2 1 1 1 1 2 2 2 2 2 1 2 2 2 2 2 2 1 2 2 1
## [1925] 2 1 1 1 2 2 2 1 1 1 2 1 2 2 1 2 1 1 2 2 2 2 1 1 1 1 1 1 2 2 2 2 2 1 1 2 2 2 2
## [1962] 2 1 2 2 2 2 2 2 2 2 1 2 1 1 2 1 1 2 2 2 2 2 2 2 2 2 2 1 2 2 2 1 1 2 1 1 1 2 2 2
```

```
## [1999] 2 2 2 2 2 1 1 2 2 2 1 2 1 1 2 1 2 2 1 1 2 1 1 1 2 2 2 2 2 1 1 1 2 2 1 2 2
## [2036] 2 1 1 1 2 1 2 1 1 2 1 2 2 2 1 2 1 1 1 2 2 2 2 1 1 1 1 2 2 2 2 2 1 1 2 1 2
## [2073] 1 1 2 1 2 1 2 2 2 1 2 1 2 1 2 2 2 1 2 1 1 2 2 2 2 1 2 1 2 1 1 2 2 1 2 2 1
## [2110] 2 2 2 2 2 2 2 2 2 1 1 2 2 2 2 2 2 1 2 2 2 2 2 2 1 2 1 2 2 1 2 2 1 1 1 2 2 1
## [2147] 1 1 1 1 1 2 2 2 2 2 2 2 2 1 1 1 1 2 1 2 2 1 1 2 2 2 2 2 2 2 1 2 1 2 2 1 2 2
## [2184] 2 2 1 2 1 2 2 2 1 2 2 2 1 2 2 1 2 2 2 2 2 2 1 2 1 2 1 2 1 1 2
##
## Within cluster sum of squares by cluster:
## [1] 23669.01 13800.91
##  (between_SS / total_SS =  23.0 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```
# principal component analysis
pcclust = prcomp(ifood_df_clustering, scale = TRUE)
summary(pcclust)
```

```
## Importance of components:
##                           PC1     PC2     PC3    PC4     PC5    PC6    PC7
## Standard deviation     2.5445 1.42719 1.31129 1.1402 1.02219 1.0038 0.9939
## Proportion of Variance 0.2943 0.09259 0.07816 0.0591 0.04749 0.0458 0.0449
## Cumulative Proportion  0.2943 0.38687 0.46503 0.5241 0.57162 0.6174 0.6623
##                           PC8     PC9    PC10    PC11    PC12    PC13   PC14
## Standard deviation     0.91756 0.86882 0.86167 0.78307 0.76982 0.75906 0.7504
## Proportion of Variance 0.03827 0.03431 0.03375 0.02787 0.02694 0.02619 0.0256
## Cumulative Proportion  0.70059 0.73490 0.76865 0.79652 0.82346 0.84965 0.8752
##                           PC15    PC16    PC17    PC18    PC19    PC20    PC21
## Standard deviation     0.68470 0.65682 0.64701 0.61388 0.56403 0.54675 0.48585
## Proportion of Variance 0.02131 0.01961 0.01903 0.01713 0.01446 0.01359 0.01073
## Cumulative Proportion  0.89656 0.91617 0.93519 0.95232 0.96678 0.98037 0.99110
##                           PC22
## Standard deviation     0.4425
## Proportion of Variance 0.0089
## Cumulative Proportion  1.0000
```

```
pcclust$rotation[,1:2]
```

```
##                                PC1          PC2
## Income               -0.294227437  0.059499916
## Kidhome               0.258946955 -0.081565453
## Teenhome              0.053994421  0.529269432
## Recency              -0.005051413  0.012871568
## MntWines             -0.306407633  0.187831056
## MntFruits            -0.268414709 -0.129236546
## MntMeatProducts      -0.315178049 -0.134187103
## MntFishProducts      -0.278970765 -0.139352823
## MntSweetProducts     -0.271883822 -0.115709222
## MntGoldProds         -0.222118037  0.094524736
## NumDealsPurchases     0.056309101  0.466082867
## NumWebPurchases      -0.213221598  0.362155105
```

```
## NumCatalogPurchases -0.320000373  0.011142830
## NumStorePurchases   -0.287772190  0.187038571
## NumWebVisitsMonth    0.251595218  0.202761856
## AcceptedCmp3        -0.015112339 -0.048372975
## AcceptedCmp4        -0.093927209  0.172214094
## AcceptedCmp5        -0.188267873 -0.117533075
## AcceptedCmp1        -0.168618266 -0.093824178
## AcceptedCmp2        -0.055898472  0.056673960
## Complain             0.014558835 -0.003846416
## Age                 -0.060604896  0.339606551
```

```
# Cluster centres
k2$centers
```

```
##       Income    Kidhome    Teenhome      Recency   MntWines  MntFruits
## 1  0.7862173 -0.7019299 -0.14015279  0.014027958  0.8896340  0.7387051
## 2 -0.5308901  0.4739754  0.09463761 -0.009472323 -0.6007218 -0.4988077
##   MntMeatProducts MntFishProducts MntSweetProducts MntGoldProds
## 1       0.8586054       0.7577719        0.7402808    0.6194357
## 2      -0.5797699      -0.5116824       -0.4998717   -0.4182715
##   NumDealsPurchases NumWebPurchases NumCatalogPurchases NumStorePurchases
## 1        -0.1626575       0.6202264           0.9061898         0.8465529
## 2         0.1098338      -0.4188054          -0.6119011        -0.5716315
##   NumWebVisitsMonth AcceptedCmp3 AcceptedCmp4 AcceptedCmp5 AcceptedCmp1
## 1        -0.6587462   0.03131983    0.2220547    0.4147287    0.3556812
## 2         0.4448158  -0.02114859   -0.1499416   -0.2800439   -0.2401723
##   AcceptedCmp2     Complain        Age
## 1    0.1541848 -0.012571460  0.1564915
## 2   -0.1041127  0.008488828 -0.1056702
```

7.3 Analyze customer segments

```
# Analyze clusters
colMeans(ifood_df_clustering[k2$cluster == 1, 1:22])
```

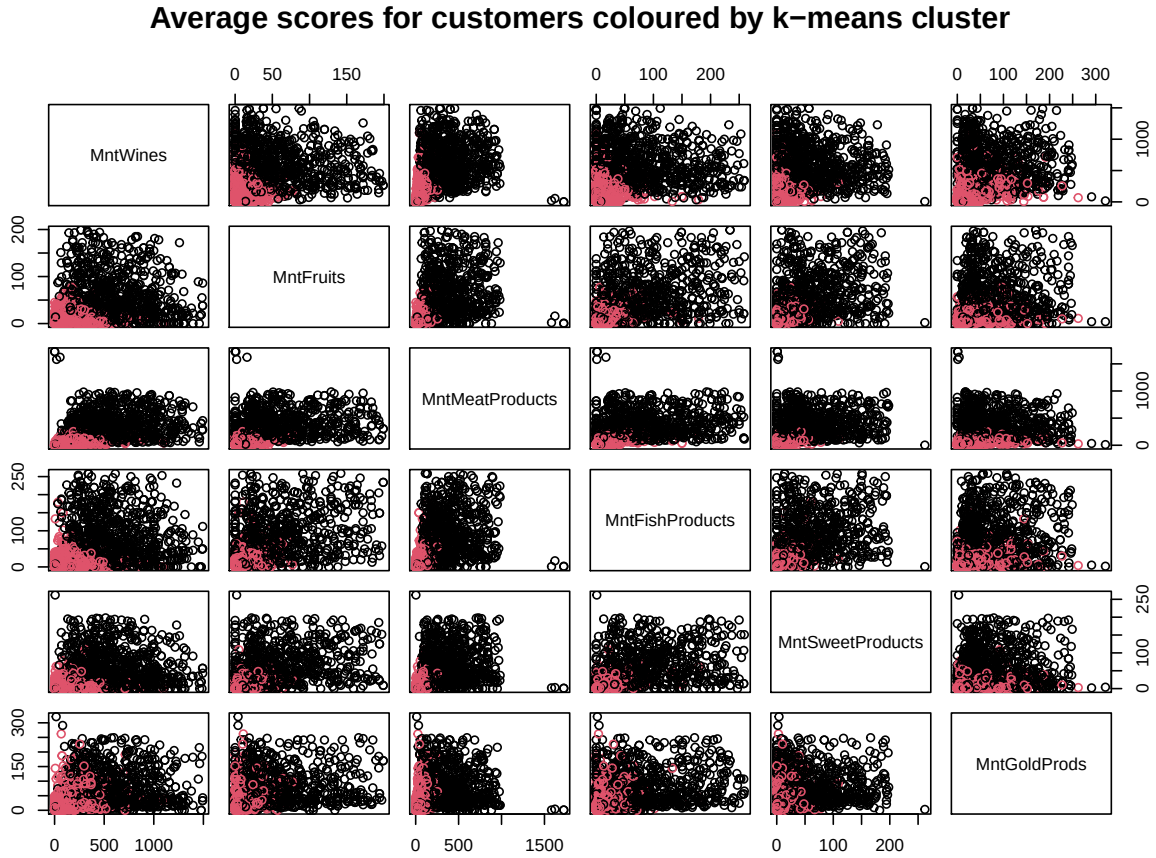
```
##              Income             Kidhome            Teenhome             Recency
##        7.203244e+04        6.502242e-02        4.293722e-01        4.941368e+01
##             MntWines            MntFruits     MntMeatProducts     MntFishProducts
##        6.052321e+02        5.567713e+01        3.594843e+02        7.913341e+01
##    MntSweetProducts        MntGoldProds   NumDealsPurchases     NumWebPurchases
##        5.744955e+01        7.593610e+01        2.012332e+00        5.788117e+00
## NumCatalogPurchases   NumStorePurchases   NumWebVisitsMonth        AcceptedCmp3
##        5.323991e+00        8.557175e+00        3.724215e+00        8.183857e-02
##        AcceptedCmp4        AcceptedCmp5        AcceptedCmp1        AcceptedCmp2
##        1.322870e-01        1.804933e-01        1.513453e-01        3.139013e-02
##            Complain                 Age
##        7.847534e-03        5.391368e+01
```

```
colMeans(ifood_df_clustering[k2$cluster == 2, 1:22])
```

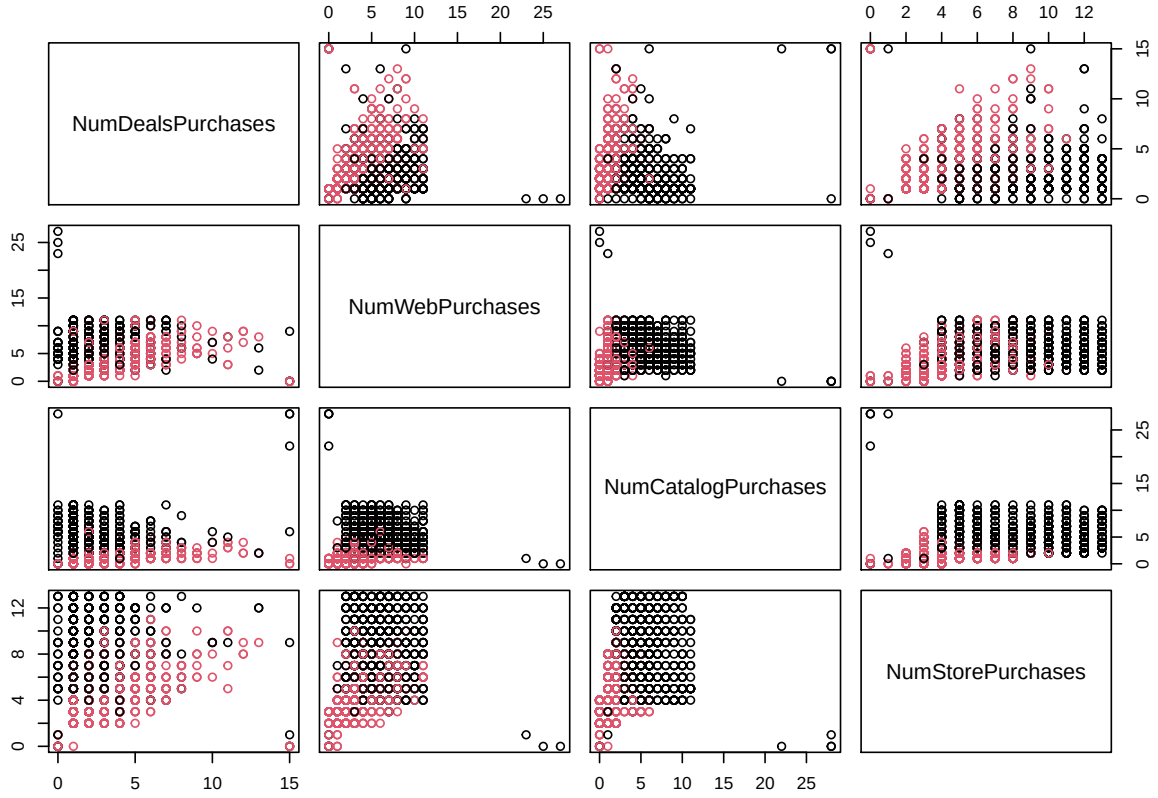
```
##              Income             Kidhome            Teenhome             Recency
##        3.886951e+04        6.964421e-01        5.571537e-01        4.873354e+01
```

```
##             MntWines            MntFruits     MntMeatProducts     MntFishProducts
##         1.025269e+02         6.503407e+00         3.696291e+01         9.613929e+00
##    MntSweetProducts         MntGoldProds   NumDealsPurchases     NumWebPurchases
##         6.497350e+00         2.228690e+01         2.536715e+00         2.939440e+00
## NumCatalogPurchases   NumStorePurchases   NumWebVisitsMonth         AcceptedCmp3
##         8.803936e-01         3.947010e+00         6.400454e+00         6.813020e-02
##         AcceptedCmp4         AcceptedCmp5         AcceptedCmp1         AcceptedCmp2
##         3.482210e-02         0.000000e+00         5.299016e-03         1.514005e-03
##             Complain                  Age
##         9.841030e-03         5.084633e+01
```

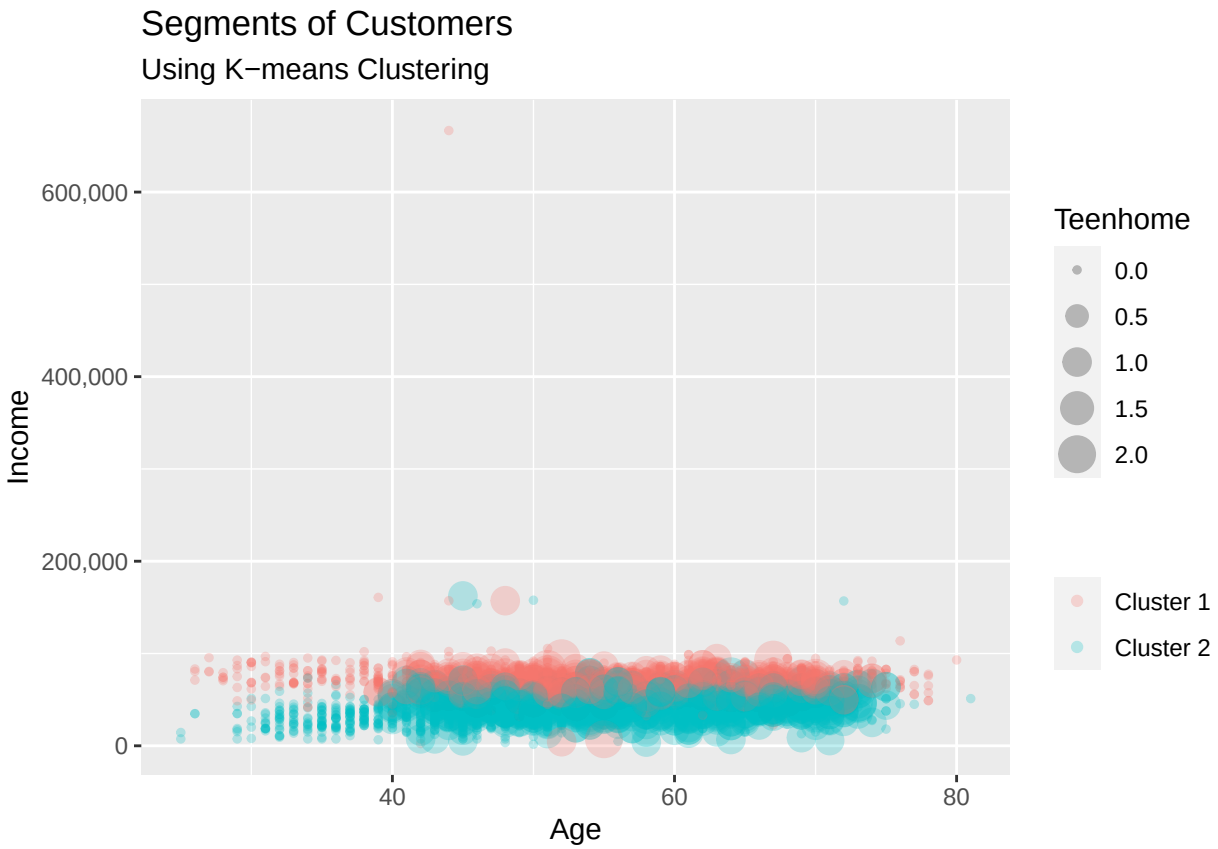
```
# Plot clusters
plot(ifood_df_clustering[,5:10], col = k2$cluster,
     main = "Average scores for customers coloured by k-means cluster")
```




```
# Plot clusters
plot(ifood_df_clustering[,11:14], col = k2$cluster,
     main = "Average scores for customers coloured by k-means cluster")
```




```
set.seed(15)
ggplot(raw_ifood, aes(x = Age, y = Income, size = Teenhome)) +
  geom_point(alpha = 0.25, stat = "identity",
             aes(color = as.factor(k2$cluster))) +
  scale_color_discrete(name=" ",
              breaks=c("1", "2"),
              labels=c("Cluster 1", "Cluster 2")) +
  ggtitle("Segments of Customers",
          subtitle = "Using K-means Clustering")  +
  scale_y_continuous(labels = scales::comma)
```




```
kCols = function(vec){cols=rainbow (length (unique (vec)))
return (cols[as.numeric(as.factor(vec))])}
digCluster <- k2$cluster
dignm      <-as.character(digCluster) # K-means clusters
plot(pcclust$x[,1:2], col = kCols(digCluster), pch = 19, xlab ="K-means", ylab = "classes")
legend("bottomleft", unique(dignm), fill = unique(kCols(digCluster)))
```

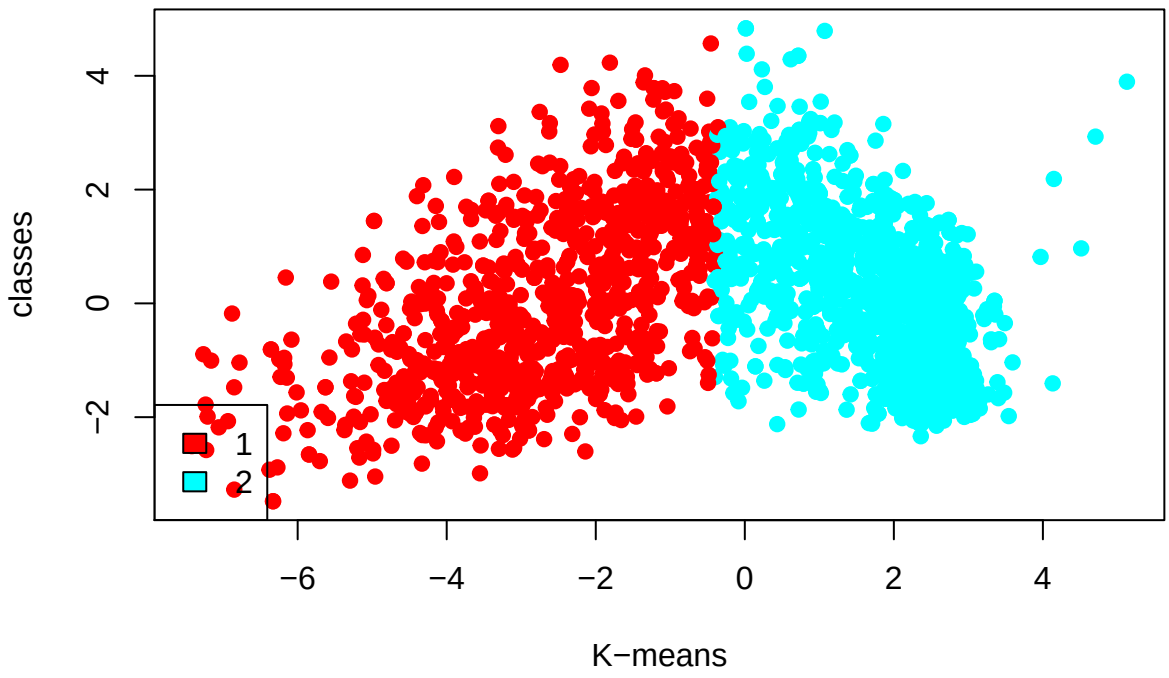


Table 2: Customers segmentation

Cluster 1 - **Low Value Customers**	Cluster 2 - **High Value Customers**
• Low or average level of income • The majority has one kid or teenager at home • Represents the most part of basic level of education • Low number of purchases through store purchase. They prefer web purchases or make catalog purchases • Negative effect of having kids and teens on advertising campaign acceptance	• High level of income • Meat and wine are preferred • The majority has no children • Low web visit and high store purchase • Number of store purchases decreases when there are kids • Selection of wines and fruits, as well as the attractive deals attract customers with higher income

8 Section 03: Predictive model (Classification)

```
# Selecting the data
ifood_ml <- raw_ifood %>%
            select(-ID,-Year_Birth,-Z_CostContact,-Z_Revenue) %>%
            mutate(Income = log10(Income)) %>%
            mutate_if(is.character, factor) %>%
```

```
            mutate(Response = as.factor(Response)) %>%
            mutate(AcceptedCmp1 = as.factor(AcceptedCmp1)) %>%
            mutate(AcceptedCmp2 = as.factor(AcceptedCmp2)) %>%
            mutate(AcceptedCmp3 = as.factor(AcceptedCmp3)) %>%
            mutate(AcceptedCmp4 = as.factor(AcceptedCmp4)) %>%
            mutate(AcceptedCmp5 = as.factor(AcceptedCmp5)) %>%
            mutate(Complain     = as.factor(Complain))
```

8.1 Build a model

```
set.seed(123)
ifood_ml_split <- initial_split(data =  ifood_ml, strata = Response)
ifood_ml_train <- training(ifood_ml_split)
ifood_ml_test  <- testing(ifood_ml_split)

# Computer performance
set.seed(234)
ifood_ml_boot <- bootstraps(ifood_ml_train)

# Setting a Random Forest
rf_spec <- rand_forest() %>%
           set_mode("classification") %>%
           set_engine("ranger", importance = "permutation") # "spark","randomForest"

ifood_ml_wf <- workflow() %>%
               add_formula(Response ~ .)
ifood_ml_wf

## == Workflow ====================================================================
## Preprocessor: Formula
## Model: None
##
## -- Preprocessor ----------------------------------------------------------------
## Response ~ .

# Training with the bootstraps
rf_rs <- ifood_ml_wf %>%
         add_model(rf_spec) %>%
         fit_resamples(resamples = ifood_ml_boot,
                       control = control_resamples(save_pred = TRUE, verbose = TRUE))
rf_rs

## # Resampling results
## # Bootstrap sampling
## # A tibble: 25 x 5
##    splits            id          .metrics       .notes         .predictions
##    <list>            <chr>       <list>         <list>         <list>
##  1 <split [1.7K/573~ Bootstrap01 <tibble [2 x ~ <tibble [0 x ~ <tibble [573 x 6~
##  2 <split [1.7K/621~ Bootstrap02 <tibble [2 x ~ <tibble [0 x ~ <tibble [621 x 6~
##  3 <split [1.7K/635~ Bootstrap03 <tibble [2 x ~ <tibble [0 x ~ <tibble [635 x 6~
##  4 <split [1.7K/619~ Bootstrap04 <tibble [2 x ~ <tibble [0 x ~ <tibble [619 x 6~
##  5 <split [1.7K/622~ Bootstrap05 <tibble [2 x ~ <tibble [0 x ~ <tibble [622 x 6~
```

```
##  6 <split [1.7K/632~ Bootstrap06 <tibble [2 x ~ <tibble [0 x ~ <tibble [632 x 6~
##  7 <split [1.7K/606~ Bootstrap07 <tibble [2 x ~ <tibble [0 x ~ <tibble [606 x 6~
##  8 <split [1.7K/611~ Bootstrap08 <tibble [2 x ~ <tibble [0 x ~ <tibble [611 x 6~
##  9 <split [1.7K/636~ Bootstrap09 <tibble [2 x ~ <tibble [0 x ~ <tibble [636 x 6~
## 10 <split [1.7K/605~ Bootstrap10 <tibble [2 x ~ <tibble [0 x ~ <tibble [605 x 6~
## # ... with 15 more rows
```

8.2 Evaluate modeling

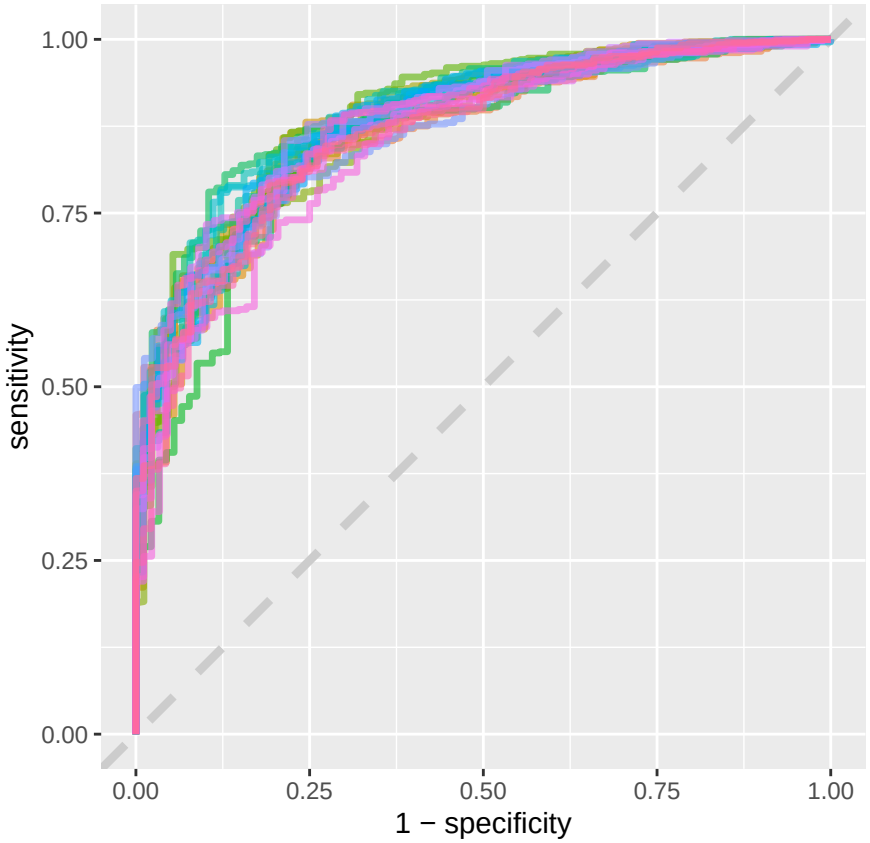
```
collect_metrics(rf_rs)
```

```
## # A tibble: 2 x 6
##   .metric  .estimator  mean     n std_err .config
##   <chr>    <chr>      <dbl> <int>   <dbl> <chr>
## 1 accuracy binary     0.870    25 0.00181 Preprocessor1_Model1
## 2 roc_auc  binary     0.876    25 0.00257 Preprocessor1_Model1
```

```
rf_rs %>%
  conf_mat_resampled()
```

```
## # A tibble: 4 x 3
##   Prediction Truth  Freq
##   <fct>      <fct> <dbl>
## 1 0          0     509.
## 2 0          1      68.9
## 3 1          0      10.5
## 4 1          1      24.2
```

```
rf_rs %>%
  collect_predictions() %>%
  group_by(id) %>%
  roc_curve(Response, .pred_0) %>%
  ggplot(aes(x = 1 - specificity, y = sensitivity, color = id)) +
  geom_abline(lty = 2, color = "gray80", size = 1.5) +
  geom_path(show.legend = FALSE, alpha = 0.6, size = 1.2) +
  coord_equal()
```




```
# Back to the testing data
ifood_ml_final <- ifood_ml_wf %>%
                  add_model(rf_spec) %>%
                  last_fit(ifood_ml_split)
ifood_ml_final
```

```
## # Resampling results
## # Manual resampling
## # A tibble: 1 x 6
##   splits        id           .metrics      .notes      .predictions    .workflow
##   <list>        <chr>        <list>        <list>      <list>          <list>
## 1 <split [1.7K~ train/test ~ <tibble [2 x~ <tibble [0~ <tibble [553 x~ <workflo~
```

```
collect_metrics(ifood_ml_final)
```

```
## # A tibble: 2 x 4
##   .metric  .estimator .estimate .config
##   <chr>    <chr>          <dbl> <chr>
## 1 accuracy binary         0.886 Preprocessor1_Model1
## 2 roc_auc  binary         0.900 Preprocessor1_Model1
```

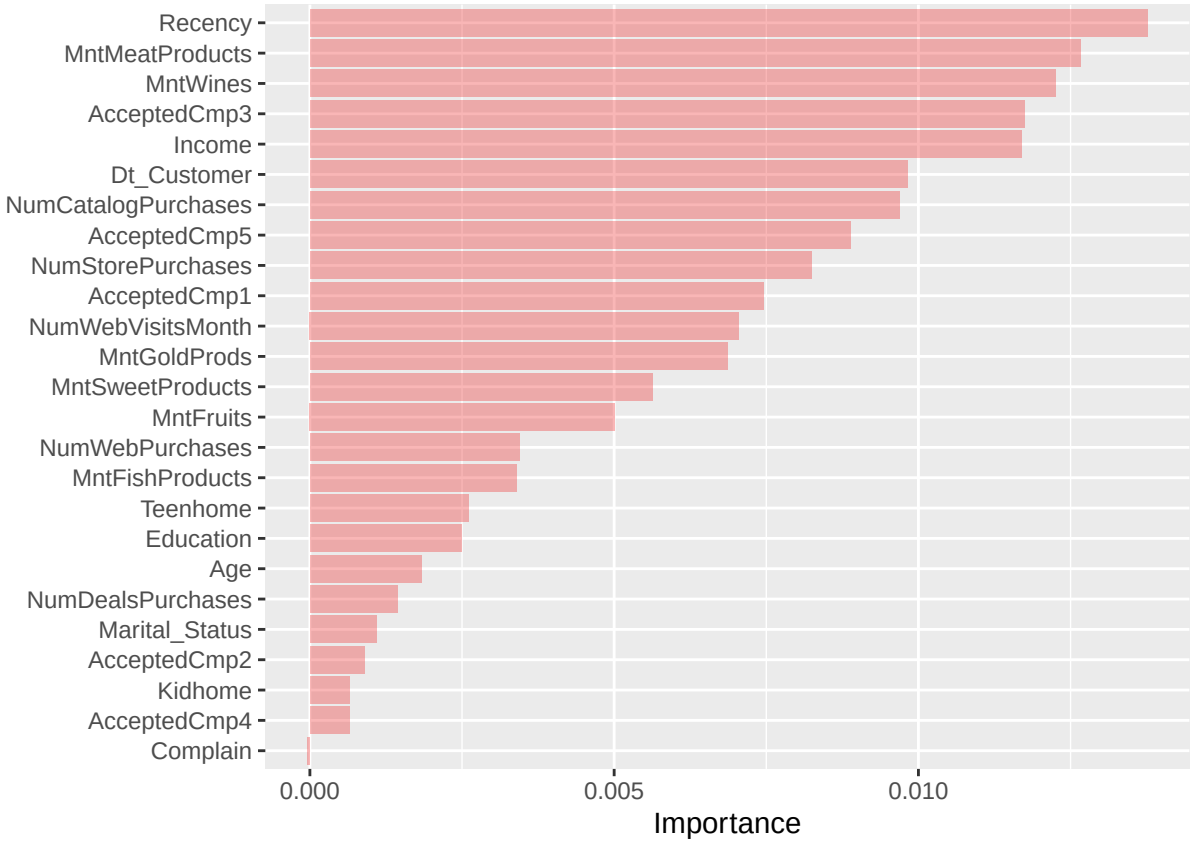
```
ifood_ml_final %>%
  collect_predictions() %>%
  conf_mat(Response, .pred_class)
```

```
##            Truth
## Prediction   0   1
##          0 462  55
##          1   8  28
```

```
ifood_ml_final$.workflow[[1]]
```

```
## == Workflow [trained] ==========================================================
## Preprocessor: Formula
## Model: rand_forest()
##
## -- Preprocessor ----------------------------------------------------------------
## Response ~ .
##
## -- Model -----------------------------------------------------------------------
## Ranger result
##
## Call:
##  ranger::ranger(x = maybe_data_frame(x), y = y, importance = ~"permutation",      num.threads = 1, v
##
## Type:                             Probability estimation
## Number of trees:                  500
## Sample size:                      1660
## Number of independent variables:  25
## Mtry:                             5
## Target node size:                 10
## Variable importance mode:         permutation
## Splitrule:                        gini
## OOB prediction error (Brier s.):  0.08675173
```

```
ifood_ml_final %>%
  pluck(".workflow", 1) %>%
  pull_workflow_fit()   %>%
  vip(aesthetics = list(alpha = 0.35, fill = "firebrick2"),
      num_features = 25)
```



- Among the most important variables for the proposed machine learning model, the number of days since the last purchase ('Recency') is very important.
- Because purchasing in store, on the web, or via the catalog ('NumStorePurchases', 'NumWebPurchases', 'NumCatalogPurchases') is positively correlated with 'Income'. Eventually, these variables become significant.
- The history of past campaigns are also crucial factors to this prediction.
- Nevertheless, variables such as 'Complain','Age' of the customer or 'Marital_Status' are not so crucial for the model.

9 Chief Marketing Officer Recommendations

Based on the findings in the sections above, I provide the following data-driven recommendations:

1. Channel Recommendations
    - Number of every kind of purchase is influenced by the income level of the customers, when there is increase in the number of web/catalog purchases, there is a reduction in store purchases.
    - The under-performing channels are deals and catalog purchases (the average customer made the fewest purchases via these channels).
    - The best performing channels are web and store purchases (the average customer made the most purchases via these channels). Focus advertising campaigns on the more successful channels, to reach more customers.
2. Product Recommendations

- Customers who spend more on Wines and Meat products tend to spend less on Fish products.
- Wines and Meat products are the top two best performing products in terms of sales. Deals and promotions should be carried out to increase the sales of other products. Focus advertising campaigns on boosting sales of the less popular items.

3. Campaign Recommendations

- Overall Campaigns have not done well for this company.
- Create two streams of targeted advertising campaigns, one aimed at high-income individuals without kids/teens and another aimed at lower-income individuals with kids/teens.

4. Customer Loyalty Recommendations

- CMO should consider some loyalty program or rewards for members to increase stickiness & purchase frequency. This is because engagement within members is rather low, as many have been members for over a year, but median recency (days since last purchase) is approximately 50 days. Considering the store has a large variety of fresh products (Fruits, Fish & Meat), they would prefer weekly / biweekly visits of customers.
- Another thing to take into account, is inventing a reward system unique to the store channel will enhance consumer engagement and is expected to increase the number of instore purchases.
- The marketing department should target customers with Teens in order to increase the number of store purchases. On the other hand should not target customers with with Kids because this will decrease the number of store purchases.

5. iFood Data Science Team

- To strengthen the machine learning model it would be useful to add data such as the country where the customer is located, rating in terms of perception of the company and satisfaction surveys.
- Since the description, data analysis and model building was done based on customer consumption habits, the model results may vary somewhat in the case where these habits vary drastically.
- In order to improve the previous analysis, it is important to have multidisciplinary iFood teams to understand the problem holistically. Therefore the intervention of the distribution and logistics, advertising, communication and image, public relations teams is timely.
- The methodology and algorithm used previously works well to be able to perform different experiments varying the amount of data. However, it is important to consider a scalable model using technologies such as Hadoop ecosystem with the Mahout machine-learning framework or Spark ecosystem with the MLlib machine-learning library.