DATA MINING
Food Marketing

P1-Report

Group 6

Team Members

Font I Cabarrocas, Marc

Garcia Ayala, Jesus

Hernández Navarro, Arnau

Khalipskaya Soboleva, Aglaya

Mediavilla Jiménez, Alex Michel

Professor

Sergi Ramirez Mitjans

September 24, 2025

Contents

1 Working Plan

1.1 Gantt Diagram

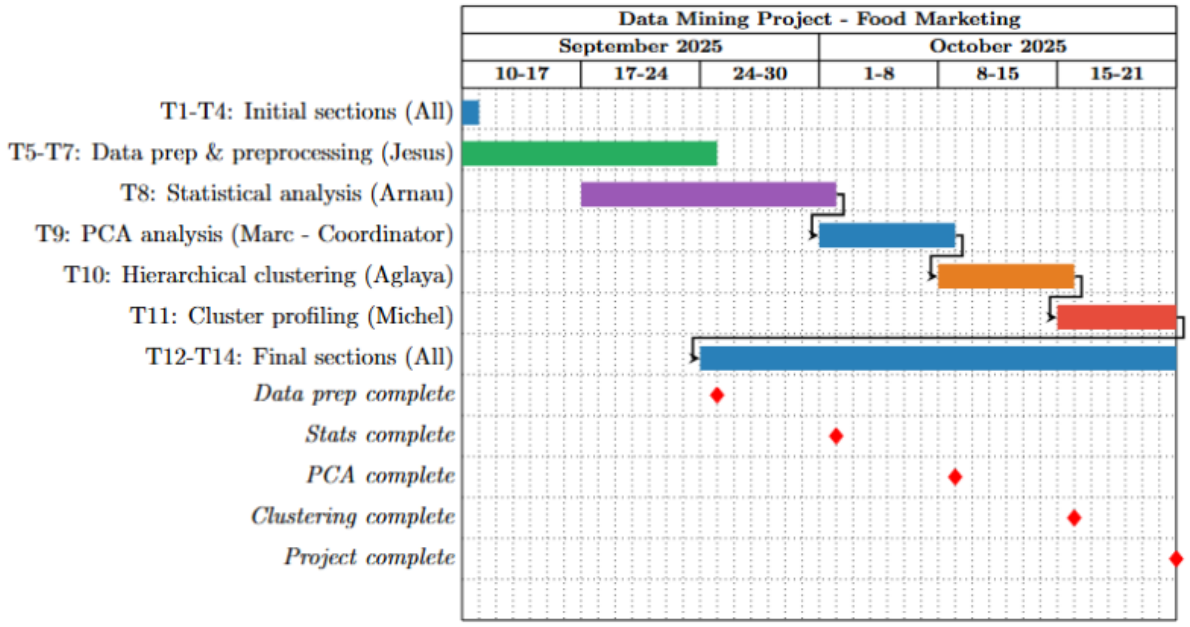


Figure 1: Gantt Diagram for Tasks

- Jesus Garcia Ayala - Data preparation
- Arnau Hernández Navarro - Statistical analysis
- Marc Font I Cabarrocas - PCA analysis & **Project Coordinator**
- Aglaya Khalipskaya Soboleva - Hierarchical clustering
- Alex Michel Mediavilla Jiménez - Cluster profiling
- All team members - Shared responsibilities

Task Dependencies:

- T8 (Statistical analysis) depends on completion of T5-T7 (Data preprocessing)
- T9 (PCA analysis) depends on T8 completion
- T10 (Hierarchical clustering) depends on T9 completion
- T11 (Cluster profiling) depends on T10 completion
- T12-T14 (Final sections) run in parallel but require integration of all analyses

Critical Path: T5-T7 → T8 → T9 → T10 → T11 (Total: 48 days including overlaps)

1.2 Risk Contingency Plan

The following risk assessment identifies potential project challenges and establishes prevention and management strategies:

Risk	Description	How to Prevent	How to Manage
Team Member Unavailability	Member becomes unavailable due to illness or personal issues	Regular communication, backup assignments for critical tasks	Redistribute work among remaining members, Jesus provides technical support to any task
Data Quality Issues	Missing values, outliers, or structural problems affecting analysis validity	Early data exploration, comprehensive preprocessing planning, document assumptions	Apply robust methods, imputation techniques, transformations. Document all decisions and impacts
Technical Difficulties	Software issues, R package conflicts, computational limitations	Test packages early, ensure compatible versions, backup environments	Use alternative tools (Python), simplify models, apply sampling techniques
Analysis Method Limitations	PCA/clustering methods unsuitable for dataset characteristics	Verify assumptions early, plan alternative methods (factor analysis, different algorithms)	Switch to non-parametric methods, apply transformations, document unsuitability reasons
Time Management	Tasks exceed estimated duration, especially complex analyses	Buffer time in schedule, early start of critical tasks, regular monitoring	Prioritize essential analyses, simplify sections, redistribute completed work
Integration Challenges	Difficulty combining PCA and clustering results coherently	Plan integration strategy early, common framework, regular team meetings	Focus on significant results, use visual methods, accept contradictions with discussion
Report Quality	Poor synthesis, unclear writing, insufficient analysis depth	Early writing assignments, multiple review cycles, detailed outlines	Peer review process, clarity focus, visual aids support
Data Access Delays	Late dataset availability or understanding	Confirm access early, request documentation in advance	Use similar public datasets, adjust scope, focus on methodological rigor

Table 1: Risk Assessment and Mitigation Strategies

Task Assignment Grid:

Task	Jesus	Arnau	Marc	Aglaya	Michel
T1-T4: Initial sections	X	X	X	X	X
T5-T7: Data prep	X				
T8: Statistical analysis		X			
T9: PCA analysis			X		
T10: Clustering				X	
T11: Profiling					X
T12-T14: Final sections	X	X	X	X	X

Table 2: Task Assignment and Coordination Matrix

2 Metadata

Metadata — iFood Dataset

> **How to use:** Place this `.Rmd` in the same folder as `ifood_base.csv` (or update the `data_path` below). Knit to HTML/Word to get a nice metadata table. It will also write `data_dictionary.csv` next to the Rmd.

1 1. Setup

```
required_pkgs <- c("readr","dplyr","tidyr","stringr","purrr","lubridate","knitr","kableExtra")
to_install <- setdiff(required_pkgs, rownames(installed.packages()))
if (length(to_install)) install.packages(to_install, repos = "https://cloud.r-project.org")

library(readr)
library(dplyr)
library(tidyr)
library(stringr)
library(purrr)
library(lubridate)
library(knitr)
library(kableExtra)

theme_kable <- function(tbl) {
  kbl(tbl, booktabs = TRUE, align = "l") |>
    kable_styling(full_width = FALSE, bootstrap_options = c("striped","hover","condensed"), font_size = 12)
}
```

2 2. Load data

```
# Adjust path if needed
data_path <- "ifood_base.csv"
stopifnot(file.exists(data_path))
df <- readr::read_csv(data_path, show_col_types = FALSE)
n_rows <- nrow(df); n_cols <- ncol(df)
glue_msg <- paste0("Loaded **", n_rows, "** rows x **", n_cols, "** columns from `", data_path, "`.")
glue_msg
```

```
## [1] "Loaded **2240** rows x **29** columns from `ifood_base.csv`."
```

3 3. Variable metadata (dictionary)

The table below combines **your provided definitions** with **computed summaries** from the data (min/max or levels).

```
# --- Your provided descriptions ---
descriptions <- c(
  Id = "The ID of the customer",
  Income = "Customer's yearly household income.",
  Kidhome = "Number of small children in the customer's household.",
  Teenhome = "Number of teenagers in the customer's household.",
  Recency = "Number of days since the last purchase.",
  MntWines = "Amount spent on wine in the last 2 years.",
  MntFruits = "Amount spent on fruits in the last 2 years.",
  MntMeatProducts = "Amount spent on meat products in the last 2 years.",
  MntFishProducts = "Amount spent on fish products in the last 2 years.",
  MntSweetProducts = "Amount spent on sweet products in the last 2 years.",
  MntGoldProds = "Amount spent on gold products in the last 2 years.",
  NumDealsPurchases = "Number of purchases made with a discount.",
  NumWebPurchases = "Number of purchases made through the website.",
  NumCatalogPurchases = "Number of purchases made using a catalog.",
  NumStorePurchases = "Number of purchases made directly in a store.",
  NumWebVisitsMonth = "Number of visits to the company's website by the customer in the last month.",
  AcceptedCmp1 = "Whether the customer accepted the offer in the 1st campaign.",
  AcceptedCmp2 = "Whether the customer accepted the offer in the 2nd campaign.",
  AcceptedCmp3 = "Whether the customer accepted the offer in the 3rd campaign.",
  AcceptedCmp4 = "Whether the customer accepted the offer in the 4th campaign.",
  AcceptedCmp5 = "Whether the customer accepted the offer in the 5th campaign.",
  Complain = "Whether the customer has filed a complaint in the last 2 years.",
  Z_CostContact = "Fixed cost of contacting the customer.",
  Z_Revenue = "Fixed revenue value.",
  Response = "Whether the customer accepted the offer in the last campaign.",
  Year_Birth = "Customer's year of birth.",
  Dt_Customer = "Date when the customer enrolled.",
  Marital_Status = "Marital status of the customer.",
  Education = "The level of studies the consumer has."
)

# --- Your provided variable type classification ---
var_type_provided <- c(
  Id = "ID",
  Income = "Numerical Continuous",
  Kidhome = "Numerical Discrete",
  Teenhome = "Numerical Discrete",
  Recency = "Numerical Discrete",
  MntWines = "Numerical Continuous",
  MntFruits = "Numerical Continuous",
  MntMeatProducts = "Numerical Continuous",
  MntFishProducts = "Numerical Continuous",
  MntSweetProducts = "Numerical Continuous",
```

```
  MntGoldProds = "Numerical Continuous",
  NumDealsPurchases = "Numerical Discrete",
  NumWebPurchases = "Numerical Discrete",
  NumCatalogPurchases = "Numerical Discrete",
  NumStorePurchases = "Numerical Discrete",
  NumWebVisitsMonth = "Numerical Discrete",
  AcceptedCmp1 = "Categorical Binary",
  AcceptedCmp2 = "Categorical Binary",
  AcceptedCmp3 = "Categorical Binary",
  AcceptedCmp4 = "Categorical Binary",
  AcceptedCmp5 = "Categorical Binary",
  Complain = "Categorical Binary",
  Z_CostContact = "Numerical Discrete",
  Z_Revenue = "Numerical Discrete",
  Response = "Categorical Binary",
  Year_Birth = "Numerical Discrete",
  Dt_Customer = "Date",
  Marital_Status = "Categorical Nominal",
  Education = "Categorical Nominal"
)

# --- Optional manual "possible values" overrides where they are fixed/known ---
manual_possible <- c(
  Id = "Random 3-5 digits",
  Income = "> 0",
  Kidhome = "0-5",
  Teenhome = "0-5",
  Recency = "0-365 (approx)",
  MntWines = "≥ 0",
  MntFruits = "≥ 0",
  MntMeatProducts = "≥ 0",
  MntFishProducts = "≥ 0",
  MntSweetProducts = "≥ 0",
  MntGoldProds = "≥ 0",
  NumDealsPurchases = "≥ 0",
  NumWebPurchases = "≥ 0",
  NumCatalogPurchases = "≥ 0",
  NumStorePurchases = "≥ 0",
  NumWebVisitsMonth = "≥ 0",
  AcceptedCmp1 = "0, 1",
  AcceptedCmp2 = "0, 1",
  AcceptedCmp3 = "0, 1",
  AcceptedCmp4 = "0, 1",
  AcceptedCmp5 = "0, 1",
  Complain = "0, 1",
  Z_CostContact = "3",
  Z_Revenue = "11",
  Response = "0, 1",
  Year_Birth = "1900-2020",
```

```
  Dt_Customer = "Date between 1900-01-01 and 2020-12-31",
  Marital_Status = "Divorced, Married, Single, Together, Widow",
  Education = "2nd Cycle, Basic, Graduation, Master, PhD"
)

# Rename common variations if present in the CSV
df <- df %>% rename(
  Year_Birth = dplyr::any_of(c("Year_Birth","Year_birth","year_birth","Year","YearBirth")),
  Dt_Customer = dplyr::any_of(c("Dt_Customer","DtCustomer","EnrollmentDate","Customer_Date","date_customer")),
  Marital_Status = dplyr::any_of(c("Marital_Status","Marital","marital_status")),
  Education = dplyr::any_of(c("Education","education"))
)

# Helper: summarize possible values from data
summ_possible <- function(v) {
  x <- df[[v]]
  # Try to coerce dates if they are character
  if (inherits(x, "character")) {
    x_try <- suppressWarnings(lubridate::ymd(x))
    if (!all(is.na(x_try))) x <- x_try
  }
  if (inherits(x, "Date")) {
    rng <- range(x, na.rm = TRUE)
    return(paste0(format(rng[1], "%Y-%m-%d"), " – ", format(rng[2], "%Y-%m-%d")))
  }
  nu <- dplyr::n_distinct(x, na.rm = TRUE)
  if (is.numeric(x)) {
    rng <- range(x, na.rm = TRUE)
    nonneg <- ifelse(all(x[!is.na(x)] >= 0), " (≥ 0)", "")
    return(paste0(format(rng[1], scientific = FALSE, trim = TRUE), " – ",
                  format(rng[2], scientific = FALSE, trim = TRUE), nonneg))
  }
  # Categorical-ish
  vals <- sort(unique(x))
  vals <- vals[!is.na(vals)]
  if (length(vals) == 0) return("—")
  if (length(vals) <= 10) return(paste(vals, collapse = ", "))
  paste0(paste(vals[1:10], collapse=", "), ", … (", length(vals), " levels)")
}

# Build data dictionary
vars <- colnames(df)
dtypes <- sapply(df, function(x) paste(class(x), collapse = "/"))
```

```
dictionary <- tibble::tibble(
  `Variable name` = vars,
  `R Type` = unname(dtypes),
  `Variable Type` = var_type_provided[vars] %||% "",
  `Description` = descriptions[vars] %||% ""
) |>
  mutate(`Possible values (from data)` = map_chr(vars, summ_possible),
         `Possible values (provided)` = manual_possible[vars] %||% NA_character_) |>
  mutate(`Possible values` = coalesce(`Possible values (provided)`, `Possible values (from data)`)) |>
  select(`Variable name`, `R Type`, `Variable Type`, Description, `Possible values`)

# Order by original column order
dictionary <- dictionary

dictionary |> theme_kable()
```

Variable name	R Type	Variable Type	Description	Possible values
ID	numeric	NA	NA	0 – 11191 (≥ 0)
Year_Birth	numeric	Numerical Discrete	Customer's year of birth.	1900–2020
Education	character	Categorical Nominal	The level of studies the consumer has.	2nd Cycle, Basic, Graduation, Master, PhD
Marital_Status	character	Categorical Nominal	Marital status of the customer.	Divorced, Married, Single, Together, Widow
Income	numeric	Numerical Continuous	Customer's yearly household income.	> 0
Kidhome	numeric	Numerical Discrete	Number of small children in the customer's household.	0–5
Teenhome	numeric	Numerical Discrete	Number of teenagers in the customer's household.	0–5
Dt_Customer	Date	Date	Date when the customer enrolled.	Date between 1900-01-01 and 2020-12-31
Recency	numeric	Numerical Discrete	Number of days since the last purchase.	0–365 (approx)
MntWines	numeric	Numerical Continuous	Amount spent on wine in the last 2 years.	≥ 0
MntFruits	numeric	Numerical Continuous	Amount spent on fruits in the last 2 years.	≥ 0
MntMeatProducts	numeric	Numerical Continuous	Amount spent on meat products in the last 2 years.	≥ 0
MntFishProducts	numeric	Numerical Continuous	Amount spent on fish products in the last 2 years.	≥ 0
MntSweetProducts	numeric	Numerical Continuous	Amount spent on sweet products in the last 2 years.	≥ 0
MntGoldProds	numeric	Numerical Continuous	Amount spent on gold products in the last 2 years.	≥ 0
NumDealsPurchases	numeric	Numerical Discrete	Number of purchases made with a discount.	≥ 0
NumWebPurchases	numeric	Numerical Discrete	Number of purchases made through the website.	≥ 0
NumCatalogPurchases	numeric	Numerical Discrete	Number of purchases made using a catalog.	≥ 0
NumStorePurchases	numeric	Numerical Discrete	Number of purchases made directly in a store.	≥ 0
NumWebVisitsMonth	numeric	Numerical Discrete	Number of visits to the company's website by the customer in the last month.	≥ 0
AcceptedCmp3	numeric	Categorical Binary	Whether the customer accepted the offer in the 3rd campaign.	0, 1
AcceptedCmp4	numeric	Categorical Binary	Whether the customer accepted the offer in the 4th campaign.	0, 1
AcceptedCmp5	numeric	Categorical Binary	Whether the customer accepted the offer in the 5th campaign.	0, 1
AcceptedCmp1	numeric	Categorical Binary	Whether the customer accepted the offer in the 1st campaign.	0, 1
AcceptedCmp2	numeric	Categorical Binary	Whether the customer accepted the offer in the 2nd campaign.	0, 1
Complain	numeric	Categorical Binary	Whether the customer has filed a complaint in the last 2 years.	0, 1
Z_CostContact	numeric	Numerical Discrete	Fixed cost of contacting the customer.	3
Z_Revenue	numeric	Numerical Discrete	Fixed revenue value.	11
Response	numeric	Categorical Binary	Whether the customer accepted the offer in the last campaign.	0, 1

3.1 Save as CSV

```
# Also write to a CSV next to the data for reuse
out_csv <- "data_dictionary.csv"
readr::write_csv(dictionary, out_csv)
paste0("Wrote: ", normalizePath(out_csv))
```

```
## [1] "Wrote: C:\\Users\\arnau\\Downloads\\data_dictionary.csv"
```

4 4. Notes

- **Variable Type** reflects your provided classification and may differ from a strict statistical interpretation (e.g., monetary amounts shown as "continuous" though often integer-valued in this dataset).
- **Possible values** prefer your provided ranges when available; otherwise they are inferred from the actual data at knit time.
- If your CSV uses different column names (e.g., `Year_birth` vs. `Year_Birth`), the Rmd attempts to standardize common variants.

2.1 Basic Initial Univariate Descriptive Statistics of Raw Variables

Basic Descriptive Statistics — iFood CSV (DOCX-safe)

1 Reference Descriptive Script (DOCX-safe)

Note: Use forward slashes in paths on Windows (e.g., `C:/Users/...`).
Place this `.Rmd` in the same folder as `ifood_base.csv` or update `data_path` below.

This version avoids HTML-only features so it **knits to Word (.docx) without errors**. If you knit to HTML, it will still show a floating ToC and nicer tables.

```
required_pkgs <- c("readr","dplyr","tidyr","stringr","purrr","lubridate","knitr")
to_install <- setdiff(required_pkgs, rownames(installed.packages()))
if (length(to_install)) install.packages(to_install, repos = "https://cloud.r-project.org")

library(readr); library(dplyr); library(tidyr); library(stringr); library(purrr)
library(lubridate); library(knitr)

# Safe Windows path
safe_path <- function(p) ifelse(is.na(p) || !nzchar(p), p, gsub("\\\\", "/", p))

# Table helper: uses HTML styling only when knitting to HTML; plain kable for Word/PDF
theme_table <- function(tbl) {
  if (knitr::is_html_output()) {
    if (!requireNamespace("kableExtra", quietly = TRUE)) {
      return(knitr::kable(tbl, align = "l"))
    }
    kableExtra::kbl(tbl, booktabs = TRUE, align = "l") |>
      kableExtra::kable_styling(full_width = FALSE, bootstrap_options = c("striped","hover","condensed"), font_si
ze = 11)
  } else {
    knitr::kable(tbl, align = "l")
  }
}

# Plot colors (enough for most categorical vars)
listOfColors <- grDevices::rainbow(40)
```

1.1 Read data

```
# If your CSV is elsewhere, replace with an absolute path using forward slashes
# e.g., data_path <- "ifood_base.csv"
data_path <- "ifood_base.csv"
data_path <- safe_path(data_path)

if (!file.exists(data_path)) {
  stop(paste0("CSV not found at: '", data_path, "'. ",
              "If you're on Windows and used backslashes (e.g., C:\\Users\\...), ",
              "replace them with forward slashes like C:/Users/..."))
}

dd <- readr::read_csv(data_path, show_col_types = FALSE)
```

1.2 Dimensions and variable names

```
dim(dd)
```

```
## [1] 2240   29
```

```
n <- nrow(dd); K <- ncol(dd)
n; K
```

```
## [1] 2240
```

```
## [1] 29
```

```
names(dd)
```

```
##  [1] "ID"                 "Year_Birth"         "Education"
##  [4] "Marital_Status"     "Income"             "Kidhome"
##  [7] "Teenhome"           "Dt_Customer"        "Recency"
## [10] "MntWines"           "MntFruits"          "MntMeatProducts"
## [13] "MntFishProducts"    "MntSweetProducts"   "MntGoldProds"
## [16] "NumDealsPurchases"  "NumWebPurchases"    "NumCatalogPurchases"
## [19] "NumStorePurchases"  "NumWebVisitsMonth"  "AcceptedCmp3"
## [22] "AcceptedCmp4"       "AcceptedCmp5"       "AcceptedCmp1"
## [25] "AcceptedCmp2"       "Complain"           "Z_CostContact"
## [28] "Z_Revenue"          "Response"
```

1.3 Optional: Declare/standardize types (recommended for iFood)

```
# Normalize common alternative names
dd <- dd %>% rename(
  Year_Birth    = dplyr::any_of(c("Year_Birth","Year_birth","year_birth","Year","YearBirth")),
  Dt_Customer   = dplyr::any_of(c("Dt_Customer","DtCustomer","EnrollmentDate","Customer_Date","date_customer")),
  Marital_Status= dplyr::any_of(c("Marital_Status","Marital","marital_status")),
  Education     = dplyr::any_of(c("Education","education"))
)

# Parse date column if present
if ("Dt_Customer" %in% names(dd)) {
  dd$Dt_Customer <- suppressWarnings(parse_date_time(dd$Dt_Customer,
                           orders = c("dmy","ymd","mdy","d-b-Y","Y-m-d","d/m/Y","m/d/Y")))
  dd$Dt_Customer <- as.Date(dd$Dt_Customer)
}

# Categorical text columns
for (col in c("Education","Marital_Status")) {
  if (col %in% names(dd)) dd[[col]] <- as.factor(dd[[col]])
}

# Binary 0/1 columns (if present)
bin_cols <- intersect(c("AcceptedCmp1","AcceptedCmp2","AcceptedCmp3","AcceptedCmp4",
                        "AcceptedCmp5","Complain","Response"), names(dd))
for (bc in bin_cols) {
  if (all(na.omit(unique(dd[[bc]])) %in% c(0,1))) {
    dd[[bc]] <- factor(dd[[bc]], levels = c(0,1))
  }
}

str(dd)
```

```
## spc_tbl_ [2,240 × 29] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ ID                 : num [1:2240] 5524 2174 4141 6182 5324 ...
##  $ Year_Birth         : num [1:2240] 1957 1954 1965 1984 1981 ...
##  $ Education          : Factor w/ 5 levels "2n Cycle","Basic",..: 3 3 3 3 5 4 3 5 5 5 ...
##  $ Marital_Status     : Factor w/ 8 levels "Absurd","Alone",..: 5 5 6 6 4 6 3 4 6 6 ...
##  $ Income             : num [1:2240] 58138 46344 71613 26646 58293 ...
##  $ Kidhome            : num [1:2240] 0 1 0 1 1 0 0 1 1 1 ...
##  $ Teenhome           : num [1:2240] 0 1 0 0 0 1 1 0 0 1 ...
##  $ Dt_Customer        : Date[1:2240], format: "2012-09-04" "2014-03-08" ...
##  $ Recency            : num [1:2240] 58 38 26 26 94 16 34 32 19 68 ...
##  $ MntWines           : num [1:2240] 635 11 426 11 173 520 235 76 14 28 ...
##  $ MntFruits          : num [1:2240] 88 1 49 4 43 42 65 10 0 0 ...
##  $ MntMeatProducts    : num [1:2240] 546 6 127 20 118 98 164 56 24 6 ...
##  $ MntFishProducts    : num [1:2240] 172 2 111 10 46 0 50 3 3 1 ...
##  $ MntSweetProducts   : num [1:2240] 88 1 21 3 27 42 49 1 3 1 ...
##  $ MntGoldProds       : num [1:2240] 88 6 42 5 15 14 27 23 2 13 ...
##  $ NumDealsPurchases  : num [1:2240] 3 2 1 2 5 2 4 2 1 1 ...
##  $ NumWebPurchases    : num [1:2240] 8 1 8 2 5 6 7 4 3 1 ...
##  $ NumCatalogPurchases: num [1:2240] 10 1 2 0 3 4 3 0 0 0 ...
##  $ NumStorePurchases  : num [1:2240] 4 2 10 4 6 10 7 4 2 0 ...
##  $ NumWebVisitsMonth  : num [1:2240] 7 5 4 6 5 6 6 8 9 20 ...
##  $ AcceptedCmp3       : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 2 ...
##  $ AcceptedCmp4       : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ AcceptedCmp5       : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ AcceptedCmp1       : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ AcceptedCmp2       : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Complain           : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Z_CostContact      : num [1:2240] 3 3 3 3 3 3 3 3 3 3 ...
##  $ Z_Revenue          : num [1:2240] 11 11 11 11 11 11 11 11 11 11 ...
##  $ Response           : Factor w/ 2 levels "0","1": 2 1 1 1 1 1 1 1 2 1 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   ID = col_double(),
##   ..   Year_Birth = col_double(),
##   ..   Education = col_character(),
##   ..   Marital_Status = col_character(),
##   ..   Income = col_double(),
##   ..   Kidhome = col_double(),
##   ..   Teenhome = col_double(),
##   ..   Dt_Customer = col_date(format = ""),
##   ..   Recency = col_double(),
##   ..   MntWines = col_double(),
##   ..   MntFruits = col_double(),
##   ..   MntMeatProducts = col_double(),
##   ..   MntFishProducts = col_double(),
##   ..   MntSweetProducts = col_double(),
##   ..   MntGoldProds = col_double(),
##   ..   NumDealsPurchases = col_double(),
##   ..   NumWebPurchases = col_double(),
##   ..   NumCatalogPurchases = col_double(),
##   ..   NumStorePurchases = col_double(),
##   ..   NumWebVisitsMonth = col_double(),
##   ..   AcceptedCmp3 = col_double(),
##   ..   AcceptedCmp4 = col_double(),
##   ..   AcceptedCmp5 = col_double(),
##   ..   AcceptedCmp1 = col_double(),
##   ..   AcceptedCmp2 = col_double(),
##   ..   Complain = col_double(),
##   ..   Z_CostContact = col_double(),
##   ..   Z_Revenue = col_double(),
##   ..   Response = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

1.4 Missingness overview

```
miss_tbl <- tibble::tibble(
  variable = names(dd),
  missing_n = sapply(dd, function(x) sum(is.na(x))),
  missing_pct = round(100 * sapply(dd, function(x) mean(is.na(x))), 2)
) %>% arrange(desc(missing_pct))

theme_table(miss_tbl)
```

variable	missing_n	missing_pct
Income	24	1.07
ID	0	0.00
Year_Birth	0	0.00
Education	0	0.00
Marital_Status	0	0.00
Kidhome	0	0.00
Teenhome	0	0.00
Dt_Customer	0	0.00
Recency	0	0.00
MntWines	0	0.00
MntFruits	0	0.00
MntMeatProducts	0	0.00
MntFishProducts	0	0.00
MntSweetProducts	0	0.00
MntGoldProds	0	0.00
NumDealsPurchases	0	0.00
NumWebPurchases	0	0.00
NumCatalogPurchases	0	0.00
NumStorePurchases	0	0.00
NumWebVisitsMonth	0	0.00
AcceptedCmp3	0	0.00
AcceptedCmp4	0	0.00
AcceptedCmp5	0	0.00
AcceptedCmp1	0	0.00
AcceptedCmp2	0	0.00
Complain	0	0.00
Z_CostContact	0	0.00
Z_Revenue	0	0.00
Response	0	0.00

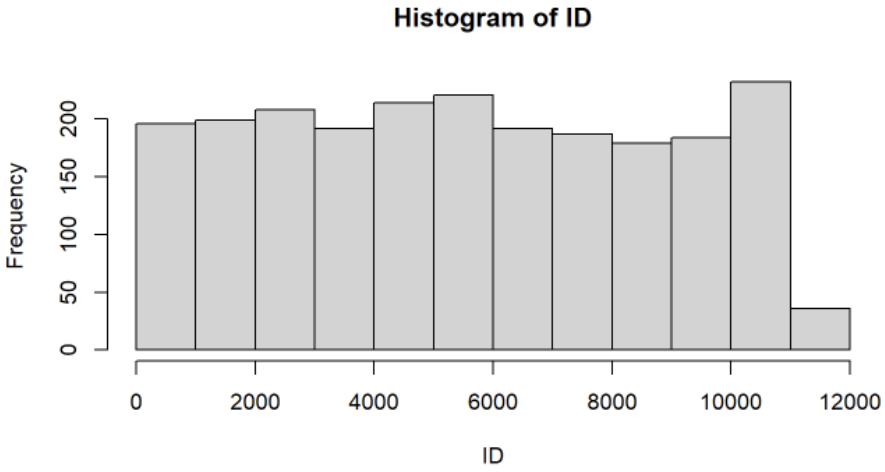
1.5 Descriptive function

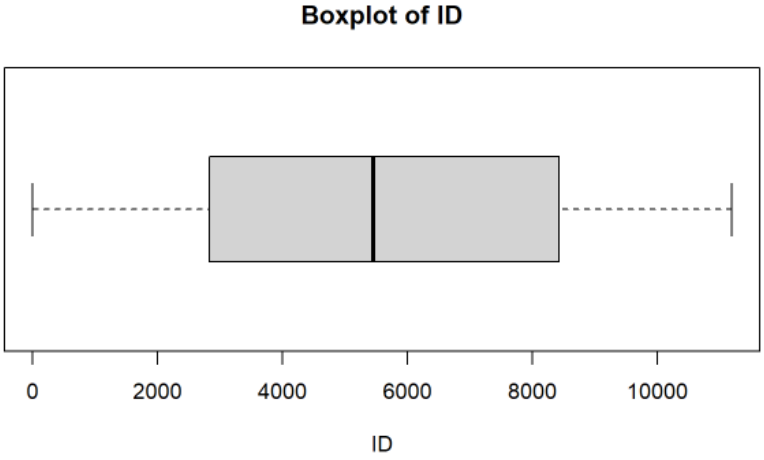
```
descriptiva <- function(X, nom, nrows_total) {
  if (!(is.numeric(X) || inherits(X, "Date"))) {
    # Categorical
    frecs <- table(as.factor(X), useNA = "ifany")
    proportions <- frecs / nrows_total
    pie(frecs, cex = 0.7, main = paste("Pie of", nom))
    barplot(frecs, las = 3, cex.names = 0.7, main = paste("Barplot of", nom), col = listOfColors)
    cat("\nNumber of modalities:", length(frecs), "\n")
    cat("\nFrequency table\n"); print(frecs)
    cat("\nRelative frequency table (proportions)\n"); print(round(proportions, 4))
    cat("\nFrequency table sorted\n"); print(sort(frecs, decreasing = TRUE))
    cat("\nRelative frequency table (proportions) sorted\n"); print(round(sort(proportions, decreasing = TRUE),
4))
  } else {
    if (inherits(X, "Date")) {
      # Date
      cat("\nExtended Summary Statistics (Date)\n"); print(summary(X))
      sd_days <- tryCatch(sd(as.numeric(X), na.rm = TRUE), error = function(e) NA_real_)
      cat("\nsd (days):", round(sd_days, 3), "\n")
      hist(X, breaks = "weeks", main = paste("Histogram (weekly) of", nom), xlab = nom)
    } else {
      # Numeric
      hist(X, main = paste("Histogram of", nom), xlab = nom)
      boxplot(X, horizontal = TRUE, main = paste("Boxplot of", nom), xlab = nom)
      cat("\nExtended Summary Statistics\n"); print(summary(X))
      sd_x <- sd(X, na.rm = TRUE)
      mn_x <- mean(X, na.rm = TRUE)
      cat("\nsd:", round(sd_x, 6), "\n")
      cat("vc (sd/mean):", ifelse(is.na(mn_x) || mn_x == 0, NA, round(sd_x / mn_x, 6)), "\n")
    }
  }
}
```

1.6 Run basic descriptives for all variables

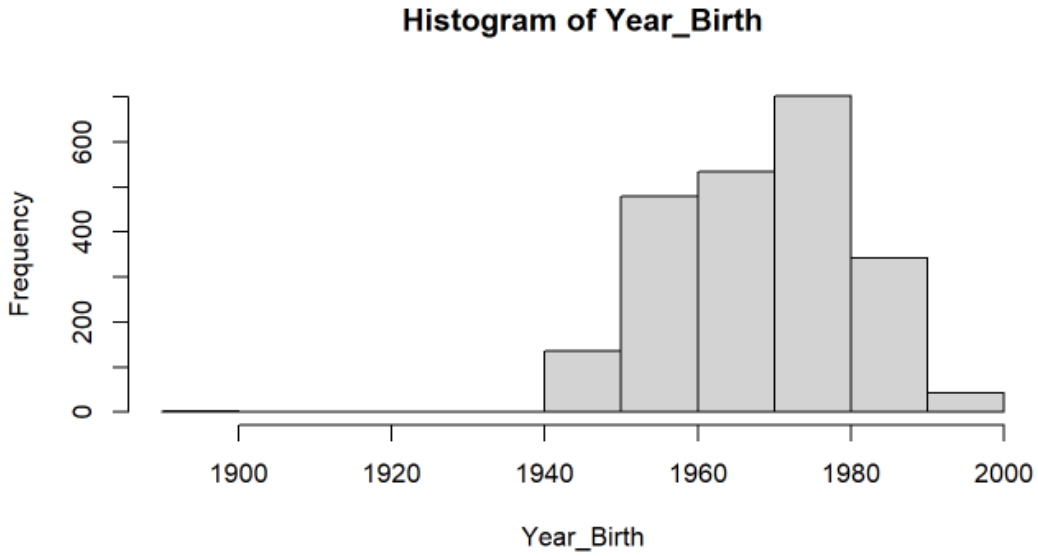
```
# Exclude known constants/IDs by default; adjust as needed
exclude <- intersect(c("Id","Z_CostContact","Z_Revenue"), names(dd))
actives <- setdiff(seq_len(ncol(dd)), match(exclude, names(dd)))

for (k in actives) {
  cat("\n\n## Variable", k, "–", names(dd)[k], "\n")
  descriptiva(dd[[k]], names(dd)[k], n)
}
```

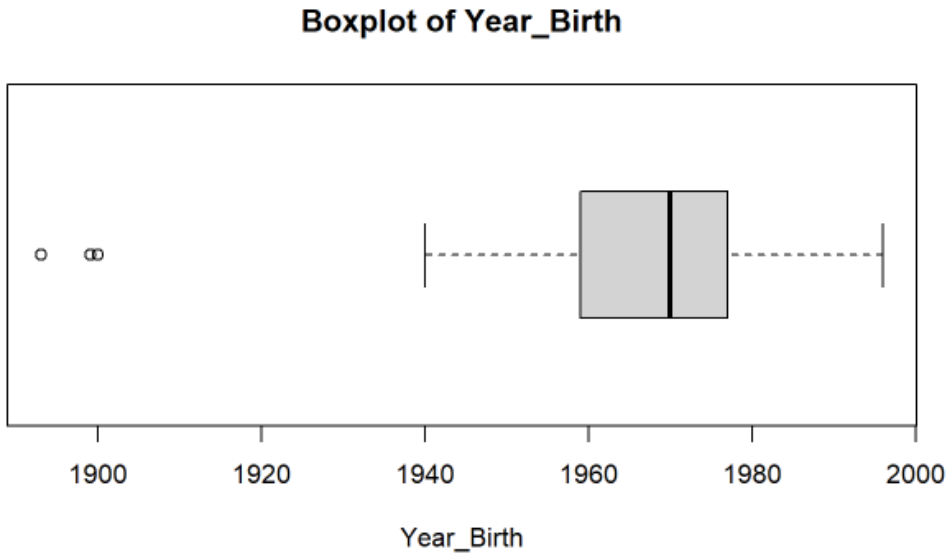
```
##
##
## ## Variable 1 – ID
```



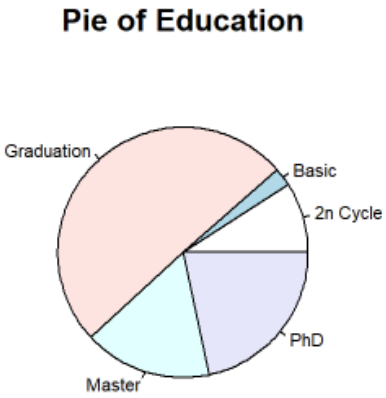


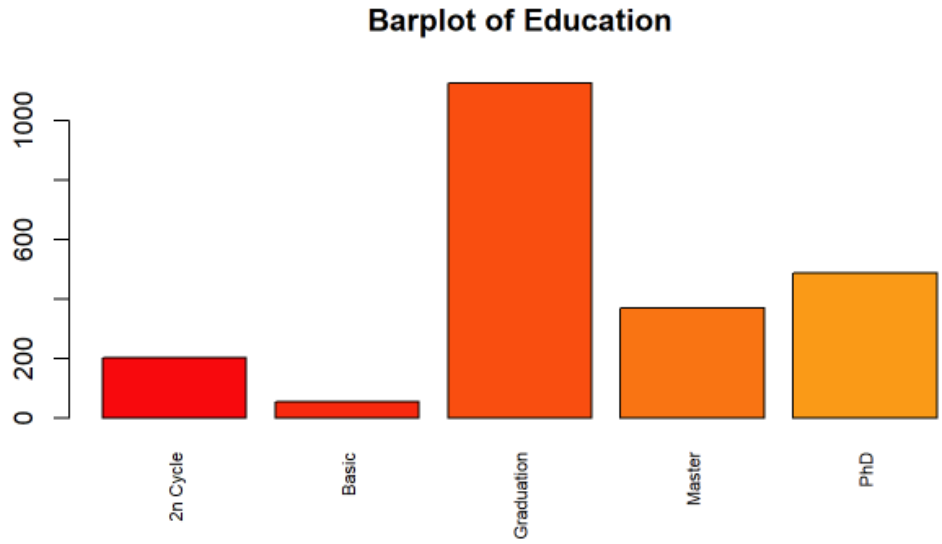

```
##
## Extended Summary Statistics
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       0    2828    5458    5592    8428   11191
##
## sd: 3246.662
## vc (sd/mean): 0.580574
##
##
## ## Variable 2 – Year_Birth
```



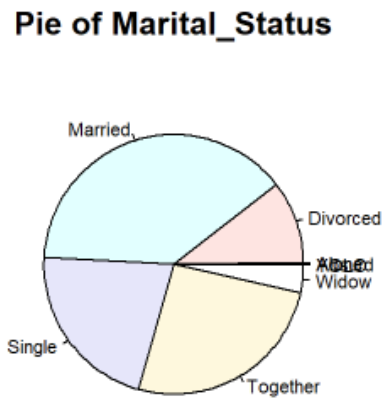


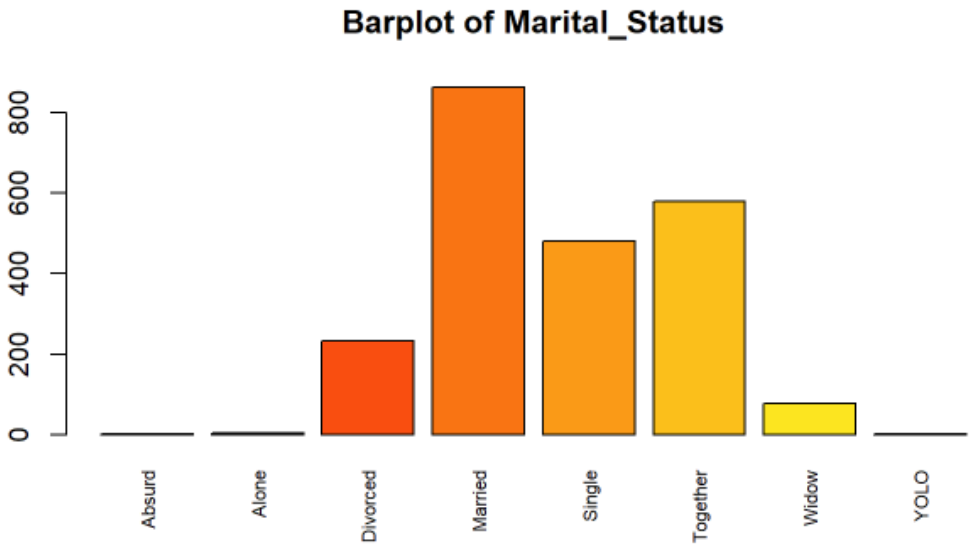

```
##
## Extended Summary Statistics
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1893    1959    1970    1969    1977    1996
##
## sd: 11.98407
## vc (sd/mean): 0.006087
##
##
## ## Variable 3 – Education
```


Pie of Education
Graduation
Basic
2n Cycle
PhD
Master

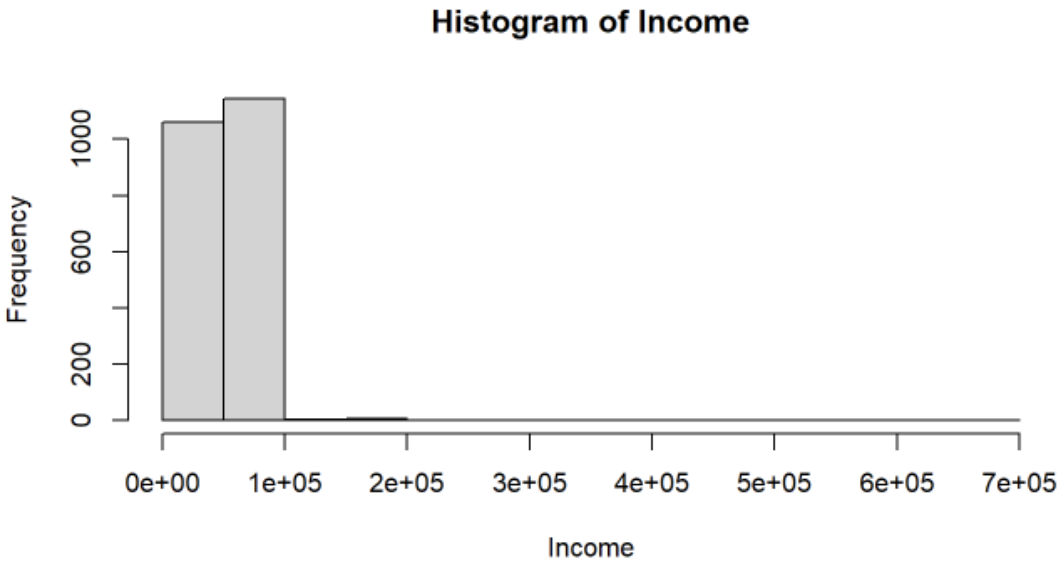

Barplot of Education
0
200
600
1000
2n Cycle
Basic
Graduation
Master
PhD

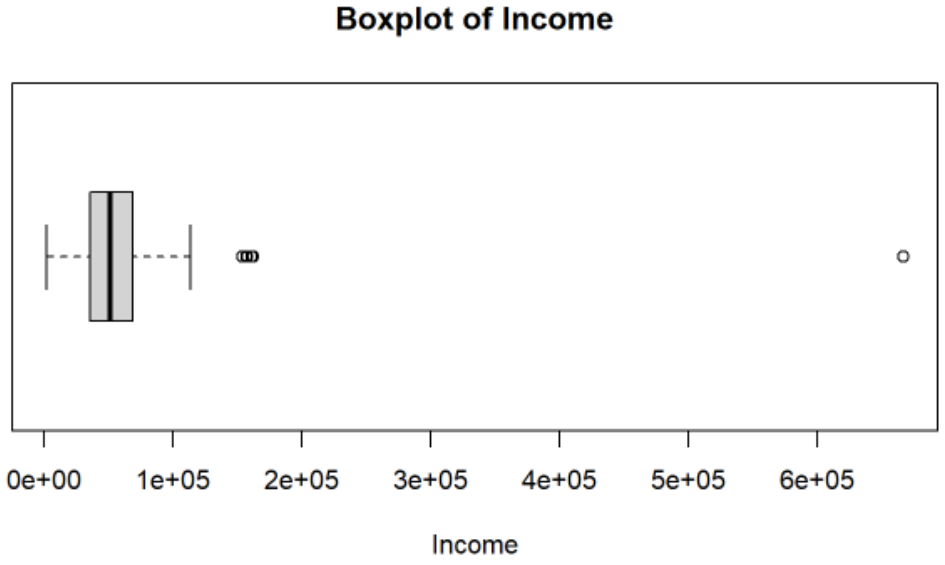
```
##
## Number of modalities: 5
##
## Frequency table
##
##   2n Cycle      Basic Graduation     Master        PhD
##        203         54       1127        370        486
##
## Relative frequency table (proportions)
##
##   2n Cycle      Basic Graduation     Master        PhD
##     0.0906     0.0241     0.5031     0.1652     0.2170
##
## Frequency table sorted
##
## Graduation        PhD     Master   2n Cycle      Basic
##       1127        486        370        203         54
##
## Relative frequency table (proportions) sorted
##
## Graduation        PhD     Master   2n Cycle      Basic
##     0.5031     0.2170     0.1652     0.0906     0.0241
##
##
## ## Variable 4 – Marital_Status
```


Pie of Marital_Status
Married
Divorced
Widow
Single
Together

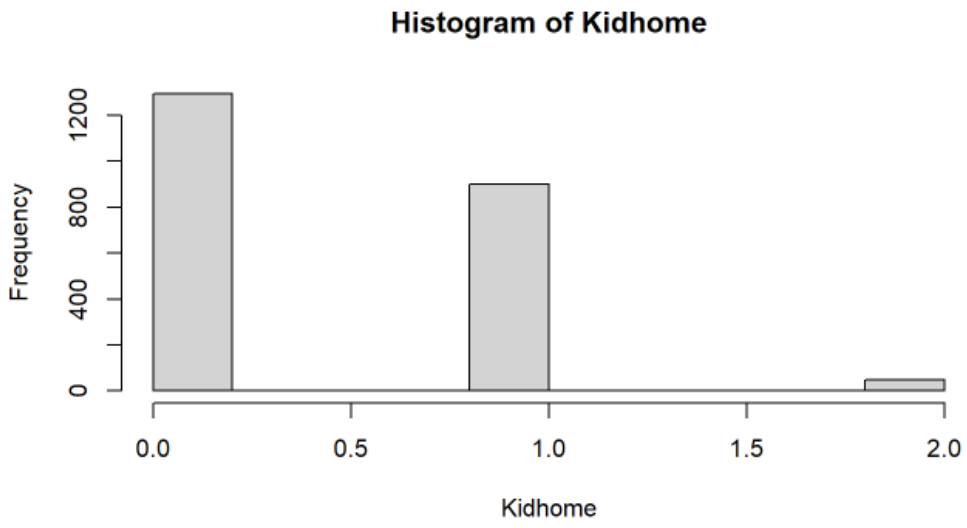

Barplot of Marital_Status
0
200
400
600
800
Absurd
Alone
Divorced
Married
Single
Together
Widow
YOLO

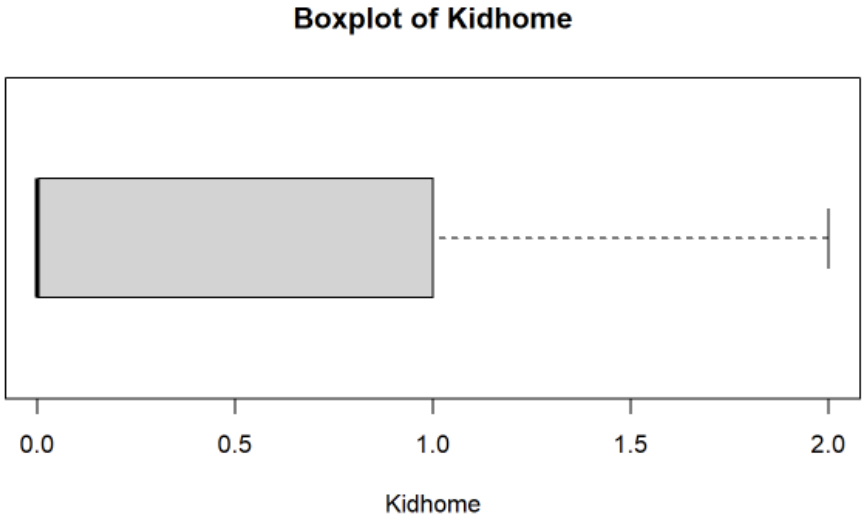
```
##
## Number of modalities: 8
##
## Frequency table
##
##   Absurd    Alone Divorced  Married   Single Together    Widow     YOLO
##       2        3      232      864      480      580       77        2
##
## Relative frequency table (proportions)
##
##   Absurd    Alone Divorced  Married   Single Together    Widow     YOLO
##   0.0009   0.0013   0.1036   0.3857   0.2143   0.2589   0.0344   0.0009
##
## Frequency table sorted
##
##  Married Together   Single Divorced    Widow    Alone   Absurd     YOLO
##       864      580      480      232       77        3        2        2
##
## Relative frequency table (proportions) sorted
##
##  Married Together   Single Divorced    Widow    Alone   Absurd     YOLO
##   0.3857   0.2589   0.2143   0.1036   0.0344   0.0013   0.0009   0.0009
##
##
## ## Variable 5 — Income
```



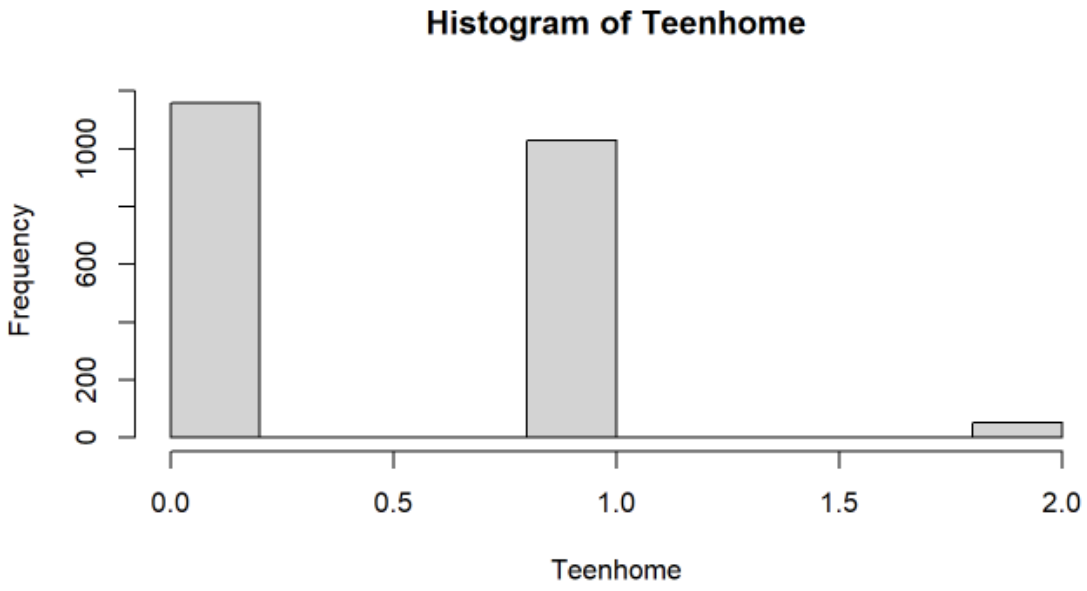


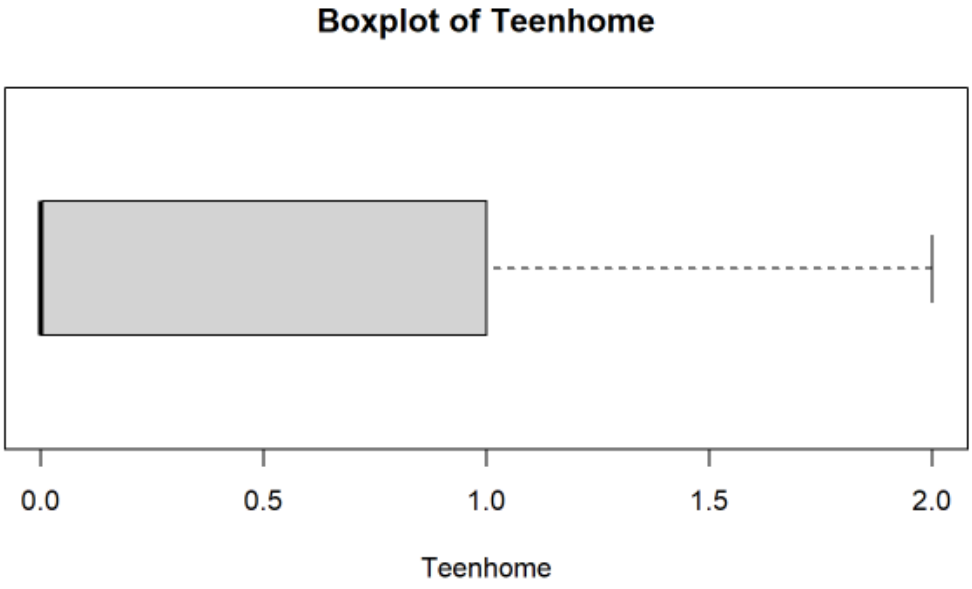

```
##
## Extended Summary Statistics
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##    1730   35303   51382   52247   68522  666666      24
##
## sd: 25173.08
## vc (sd/mean): 0.481807
##
##
## ## Variable 6 – Kidhome
```



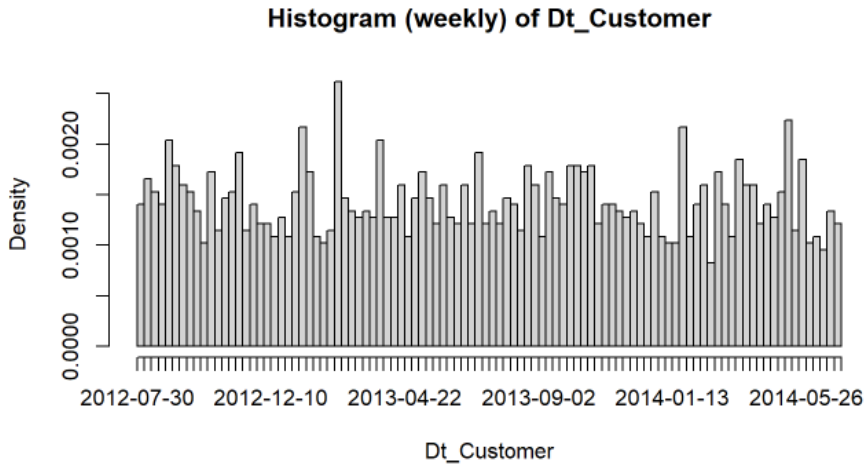



```
##
## Extended Summary Statistics
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0000  0.0000  0.0000  0.4442  1.0000  2.0000
##
## sd: 0.538398
## vc (sd/mean): 1.212072
##
##
## ## Variable 7 – Teenhome
```

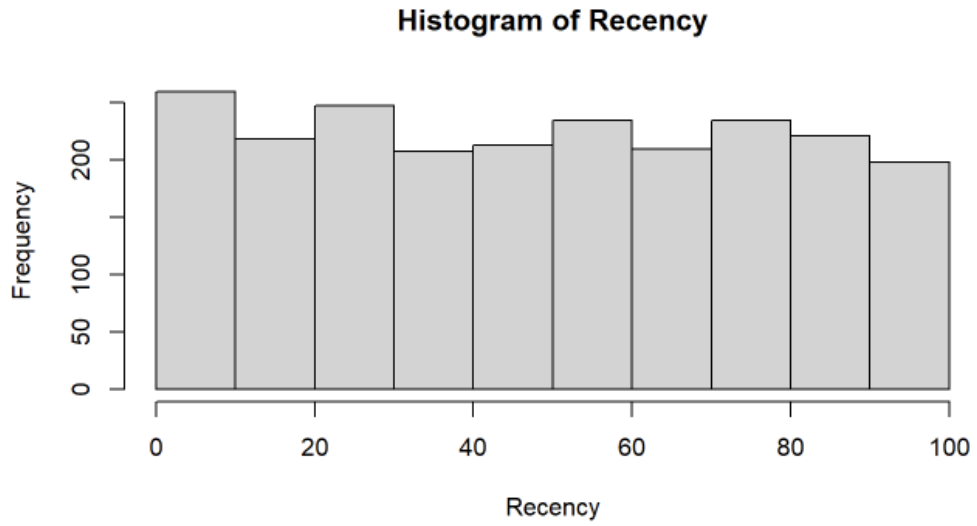

Histogram of Teenhome
Frequency
0
200
600
1000
0.0
0.5
1.0
1.5
2.0
Teenhome

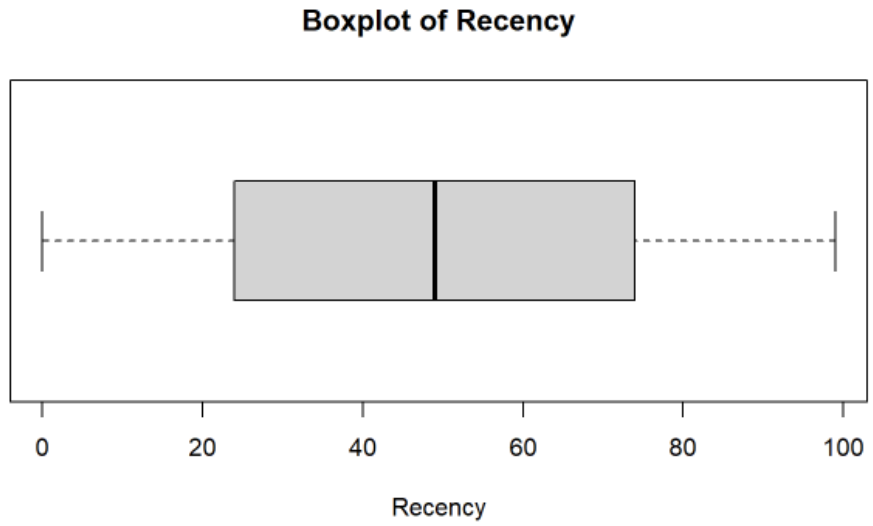

Boxplot of Teenhome
0.0
0.5
1.0
1.5
2.0
Teenhome

```
##
## Extended Summary Statistics
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0000  0.0000  0.0000  0.5062  1.0000  2.0000
##
## sd: 0.544538
## vc (sd/mean): 1.075631
##
##
## ## Variable 8 — Dt_Customer
##
## Extended Summary Statistics (Date)
##         Min.      1st Qu.       Median         Mean      3rd Qu.         Max.
## "2012-07-30" "2013-01-16" "2013-07-08" "2013-07-10" "2013-12-30" "2014-06-29"
##
## sd (days): 202.123
```

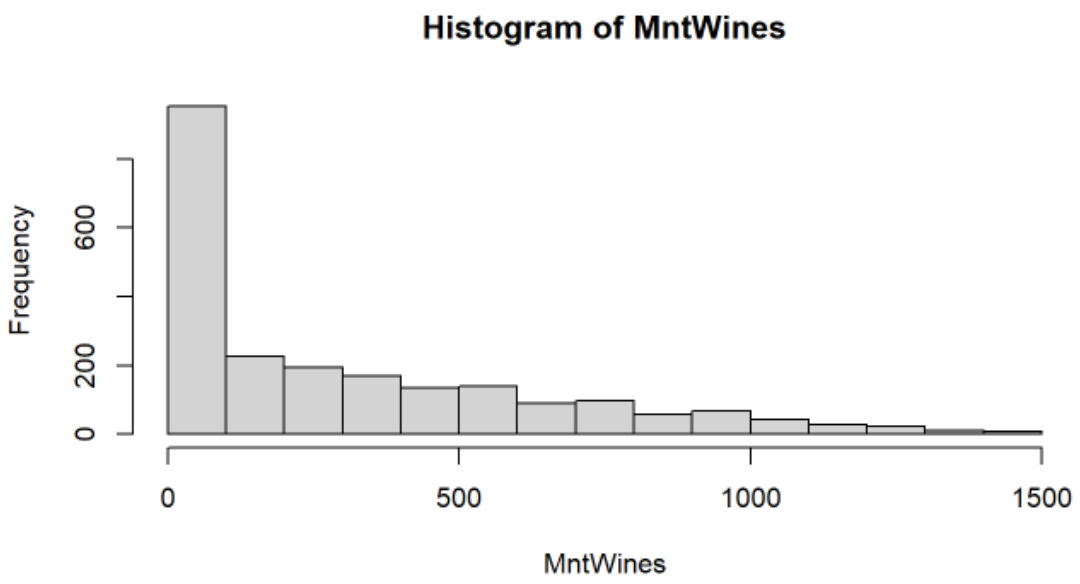


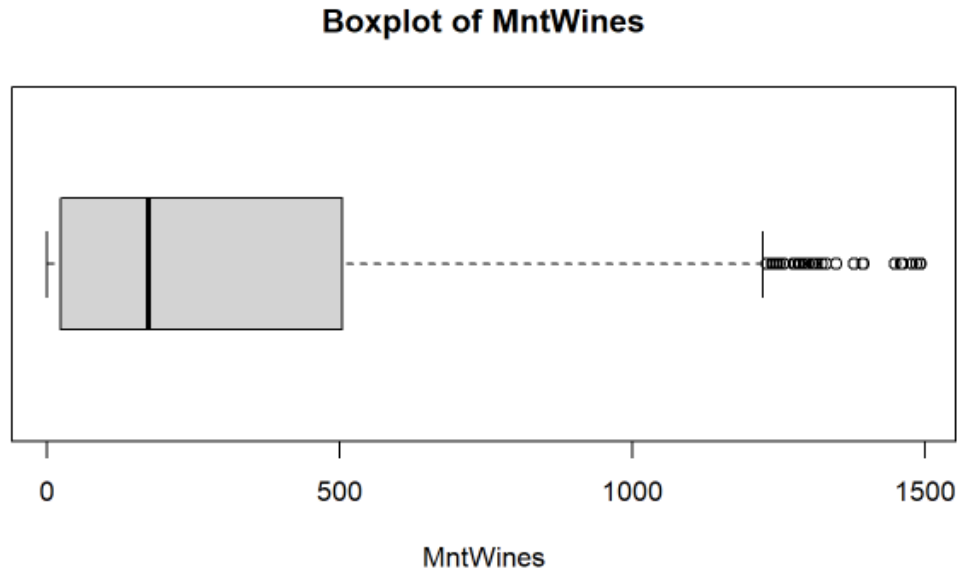

```
##
##
## ## Variable 9 — Recency
```



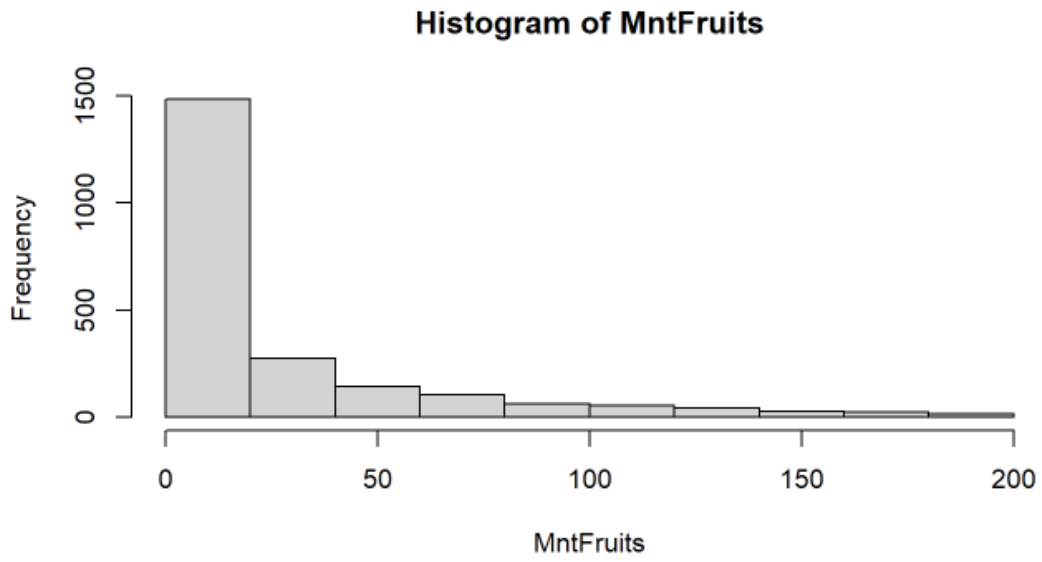



```
##
## Extended Summary Statistics
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00   24.00   49.00   49.11   74.00   99.00
##
## sd: 28.96245
## vc (sd/mean): 0.589754
##
##
## ## Variable 10 – MntWines
```

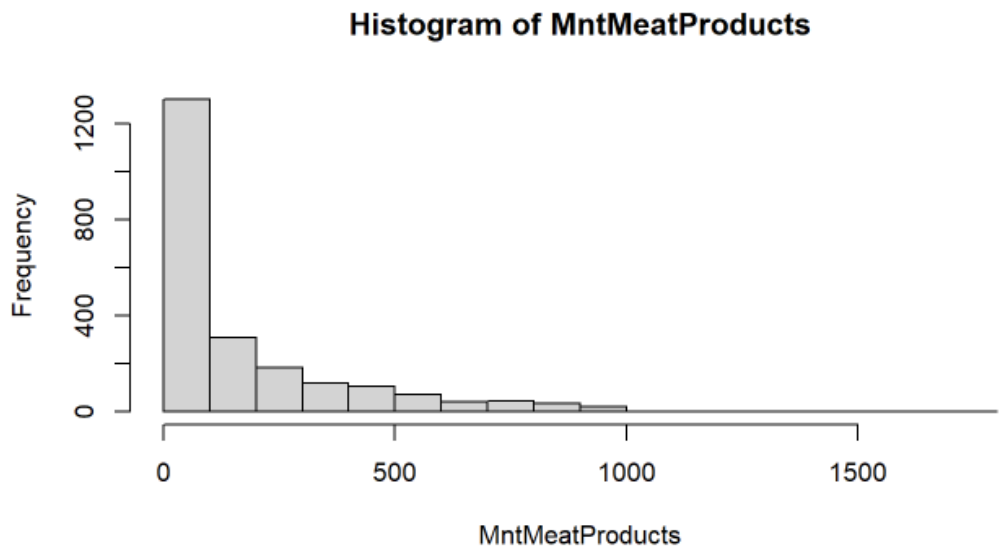


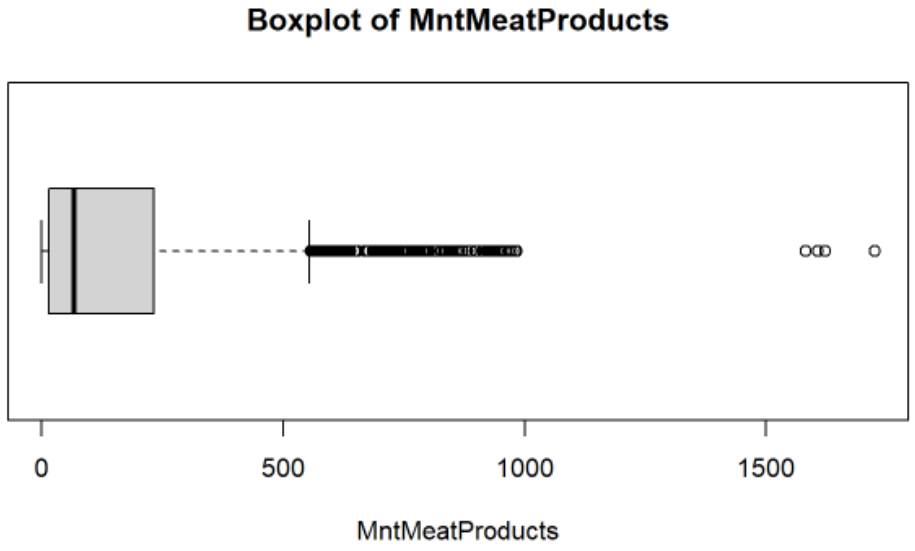



```
##
## Extended Summary Statistics
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00   23.75  173.50  303.94  504.25 1493.00
##
## sd: 336.5974
## vc (sd/mean): 1.107462
##
##
## ## Variable 11 — MntFruits
```

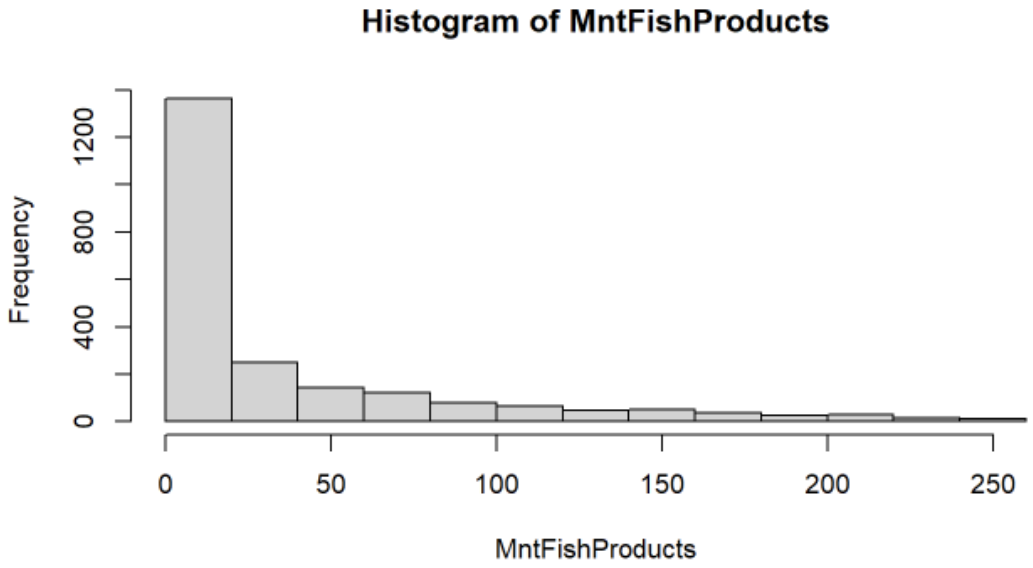


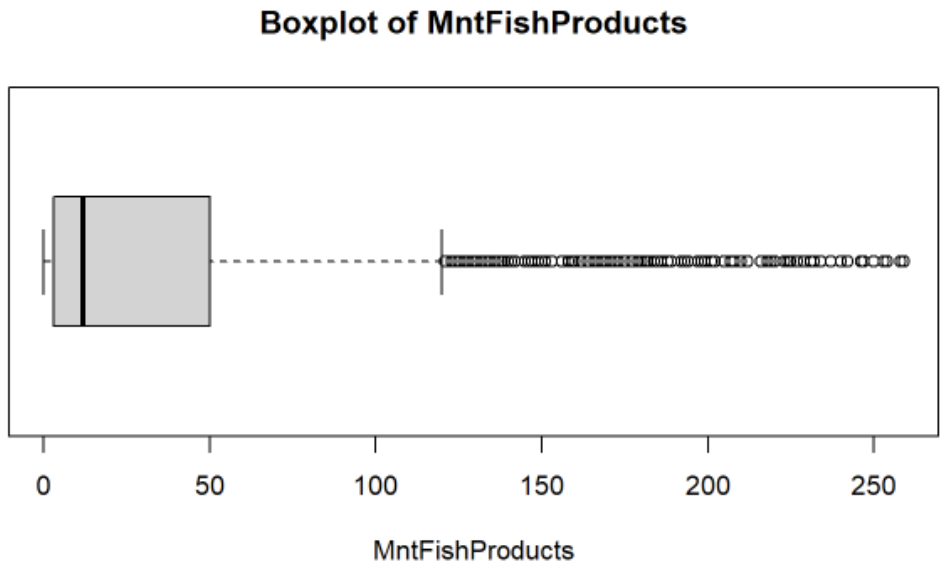

```
##
## Extended Summary Statistics
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     0.0     1.0     8.0    26.3    33.0   199.0
##
## sd: 39.77343
## vc (sd/mean): 1.51217
##
##
## ## Variable 12 – MntMeatProducts
```



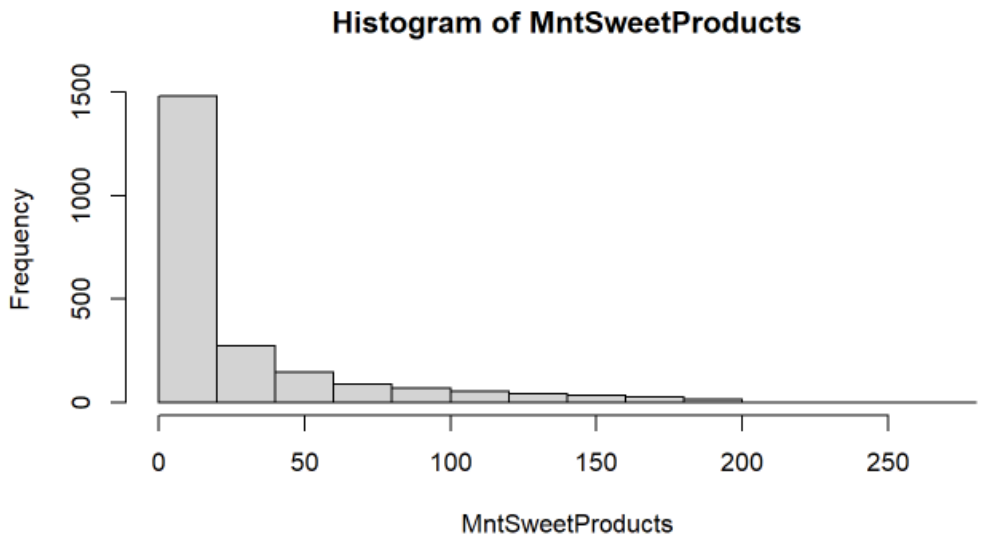


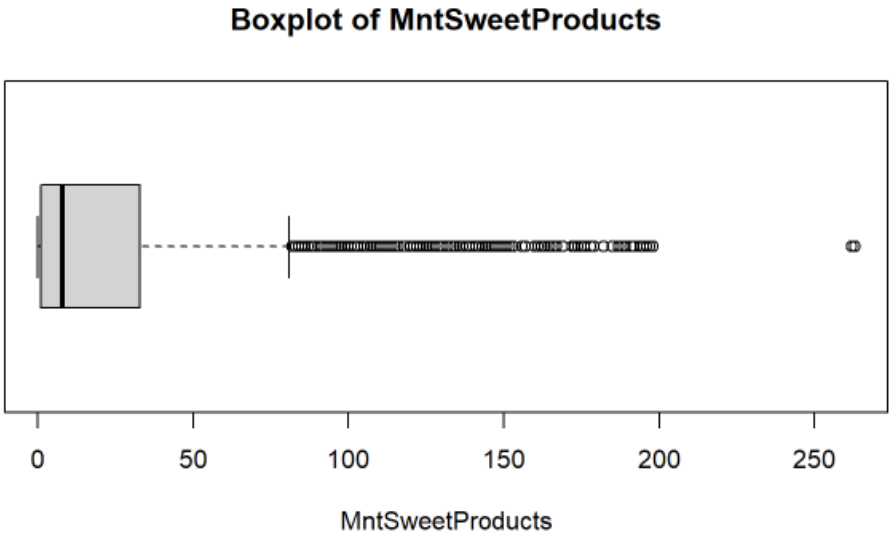

```
##
## Extended Summary Statistics
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     0.0    16.0    67.0   166.9   232.0  1725.0
##
## sd: 225.7154
## vc (sd/mean): 1.351994
##
##
## ## Variable 13 — MntFishProducts
```



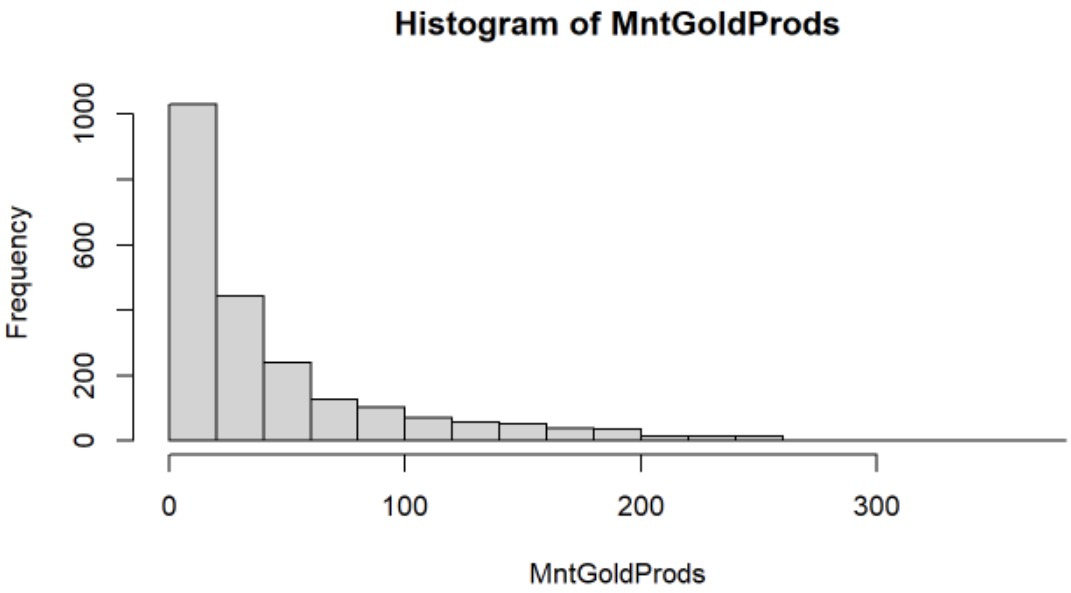


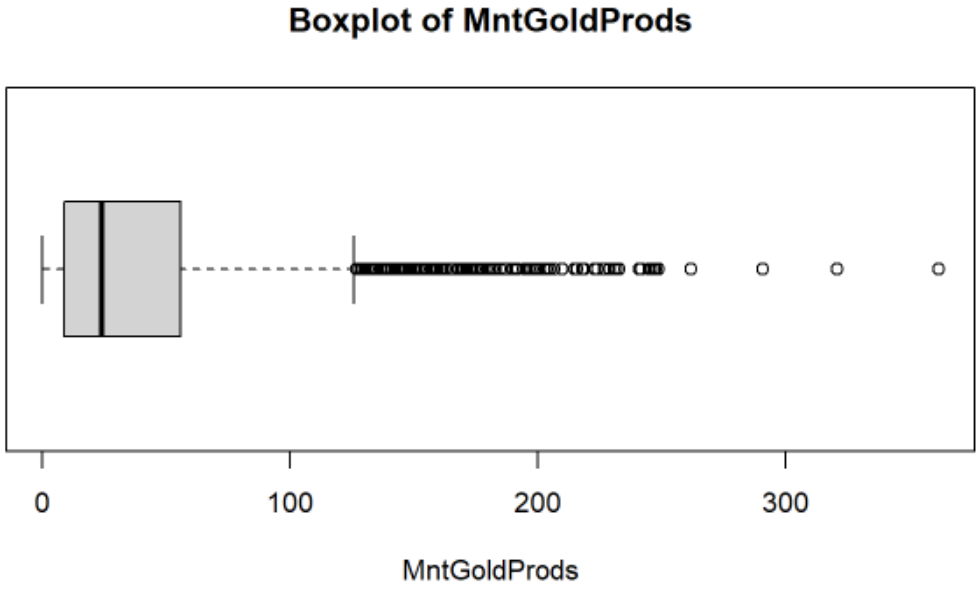

```
##
## Extended Summary Statistics
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00    3.00   12.00   37.53   50.00  259.00
##
## sd: 54.62898
## vc (sd/mean): 1.455785
##
##
## ## Variable 14 – MntSweetProducts
```



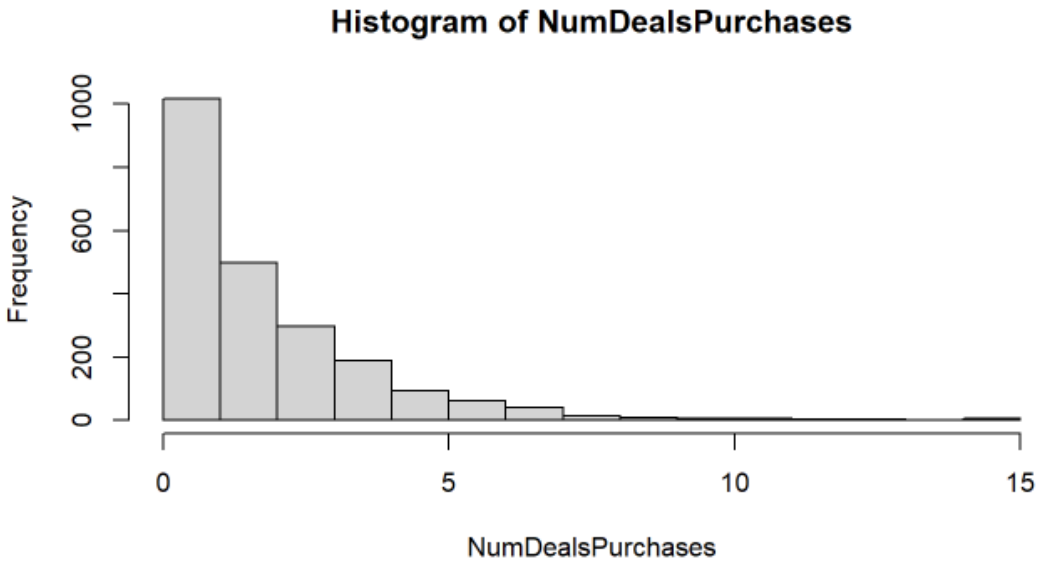


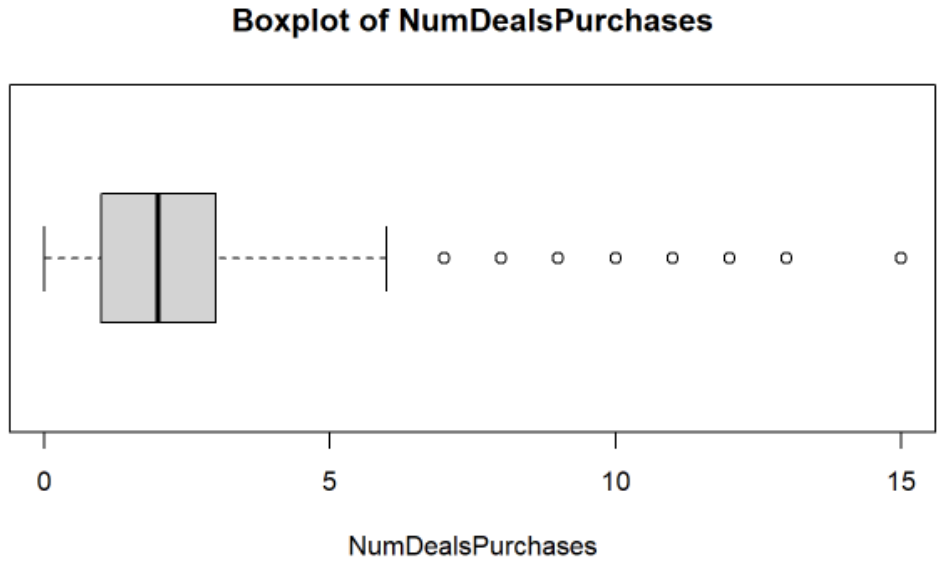

```
##
## Extended Summary Statistics
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00    1.00    8.00   27.06   33.00  263.00
##
## sd: 41.2805
## vc (sd/mean): 1.525351
##
##
## ## Variable 15 – MntGoldProds
```



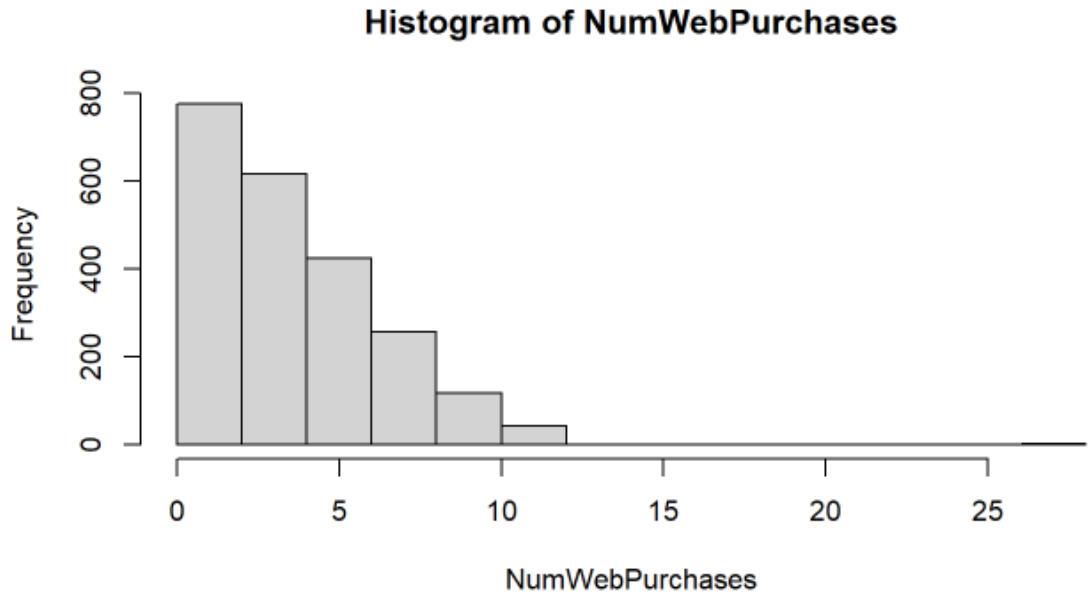


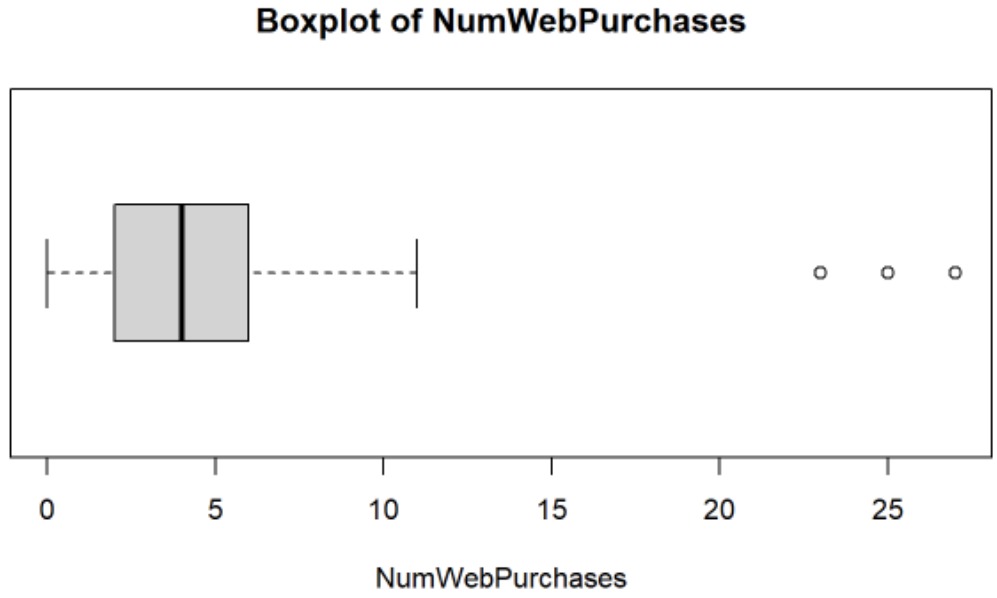

```
##
## Extended Summary Statistics
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00    9.00   24.00   44.02   56.00  362.00
##
## sd: 52.16744
## vc (sd/mean): 1.185034
##
##
## ## Variable 16 – NumDealsPurchases
```



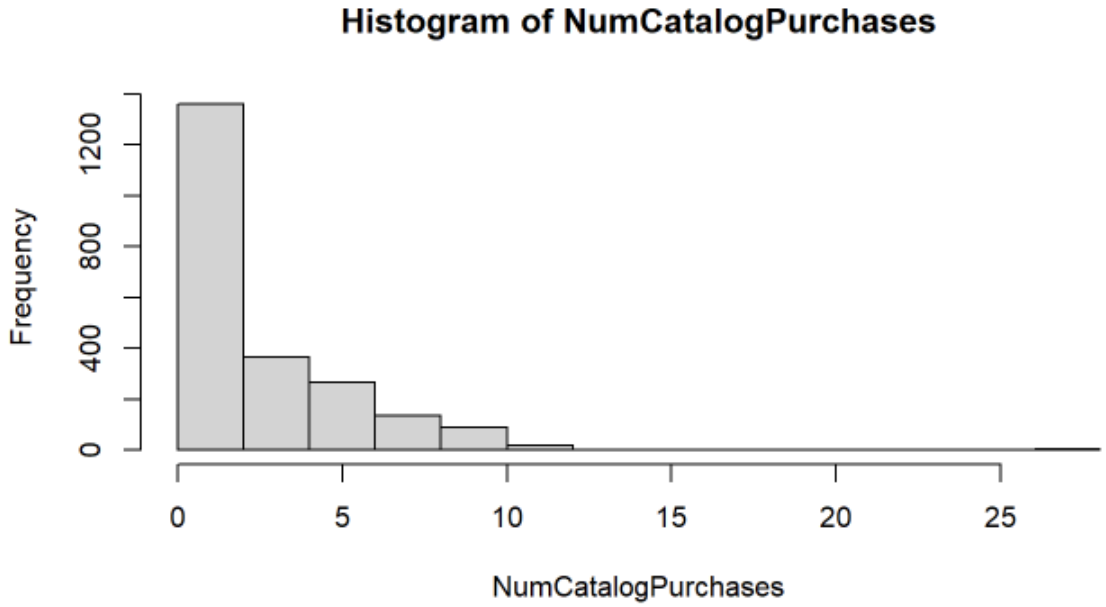


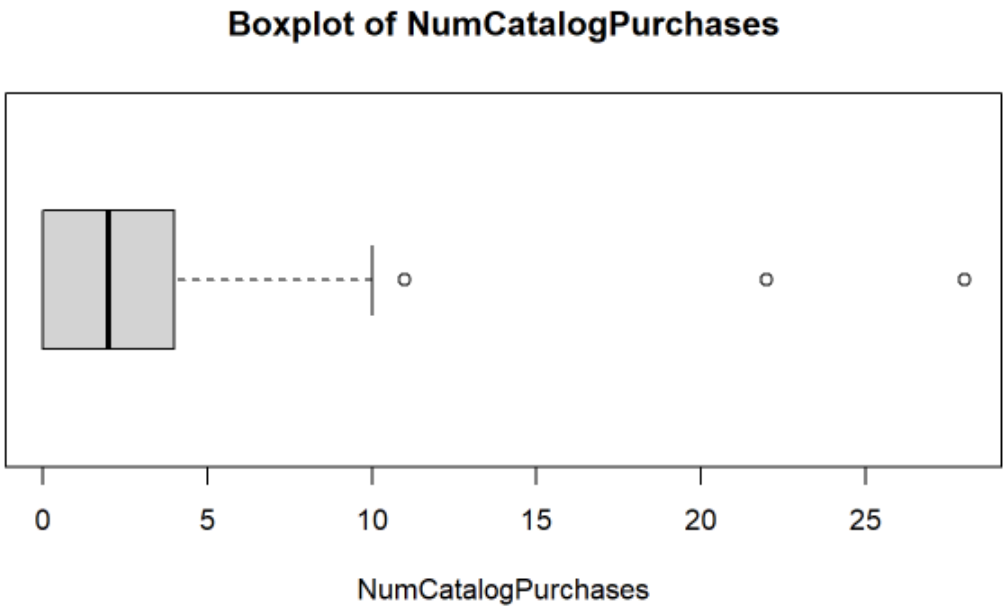

```
##
## Extended Summary Statistics
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   1.000   2.000   2.325   3.000  15.000
##
## sd: 1.932238
## vc (sd/mean): 0.83107
##
##
## ## Variable 17 – NumWebPurchases
```





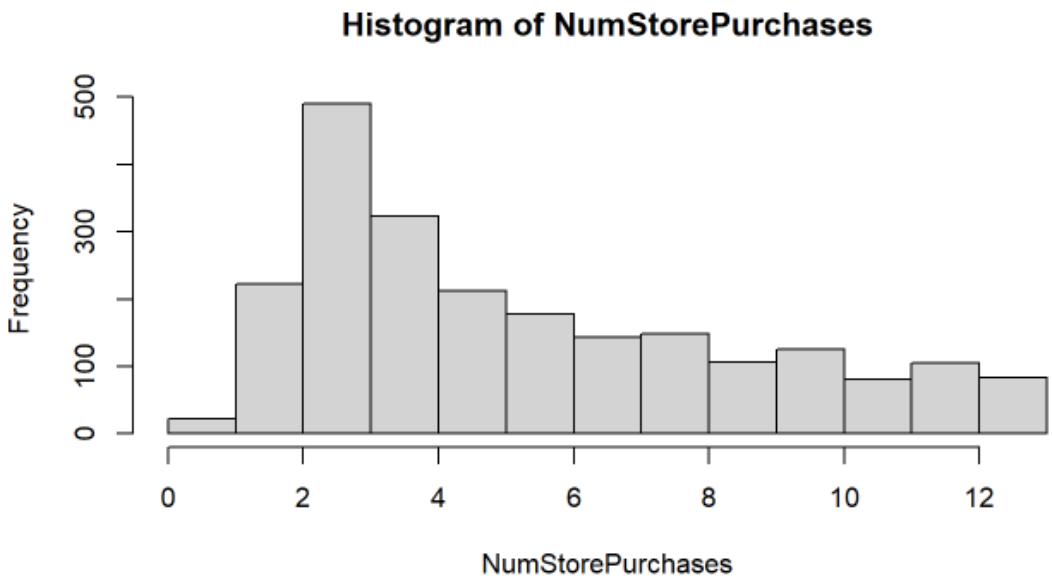

```
##
## Extended Summary Statistics
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   2.000   4.000   4.085   6.000  27.000
##
## sd: 2.778714
## vc (sd/mean): 0.680254
##
##
## ## Variable 18 — NumCatalogPurchases
```

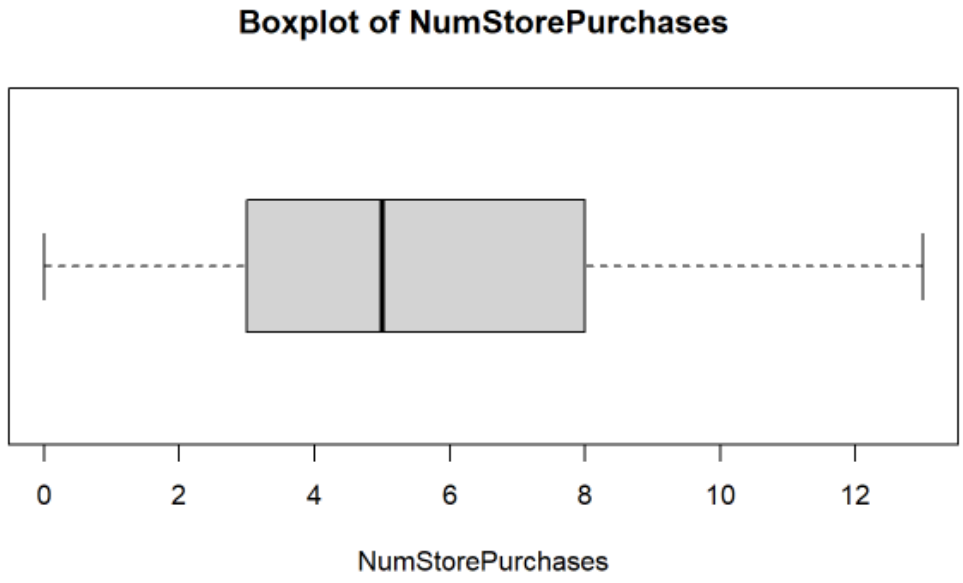




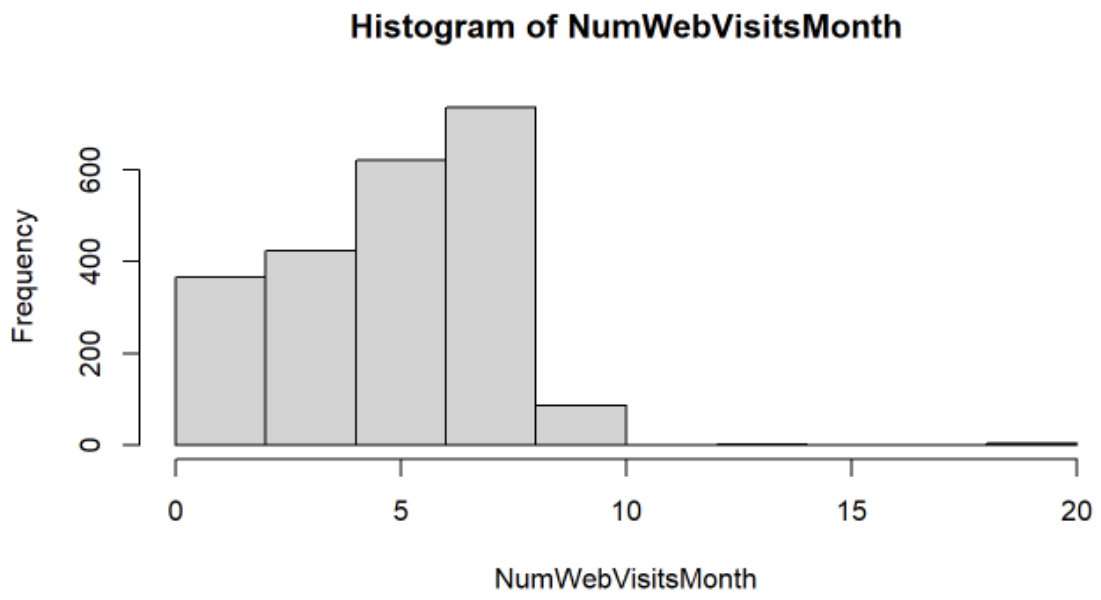
-

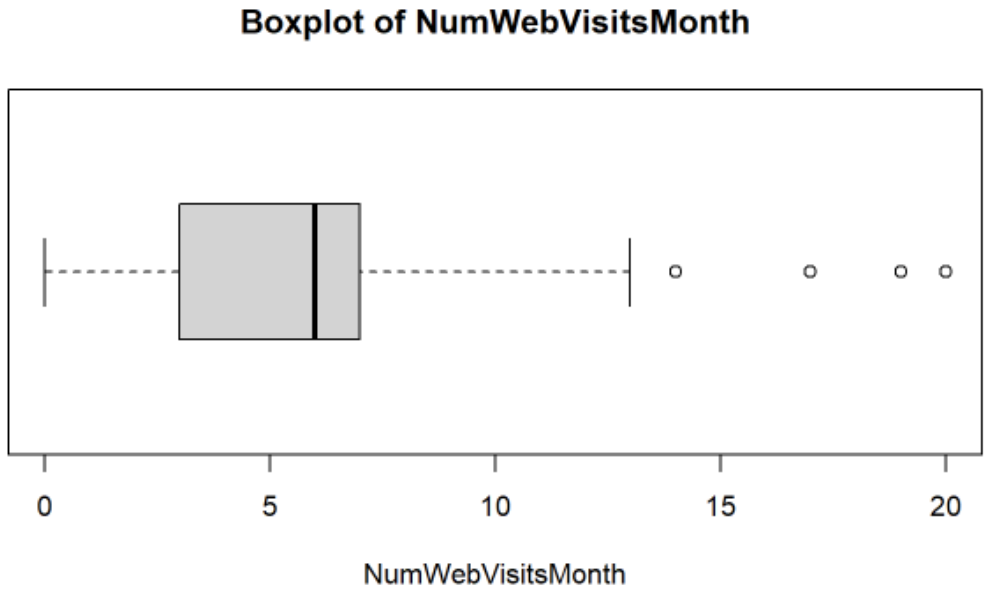
```
##
## Extended Summary Statistics
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   0.000   2.000   2.662   4.000  28.000
##
## sd: 2.923101
## vc (sd/mean): 1.098062
##
##
## ## Variable 19 – NumStorePurchases
```



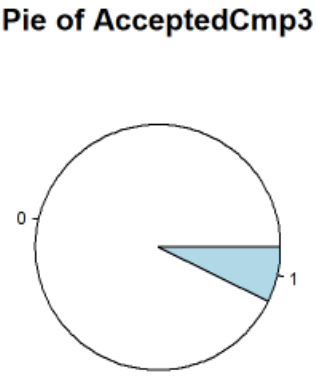


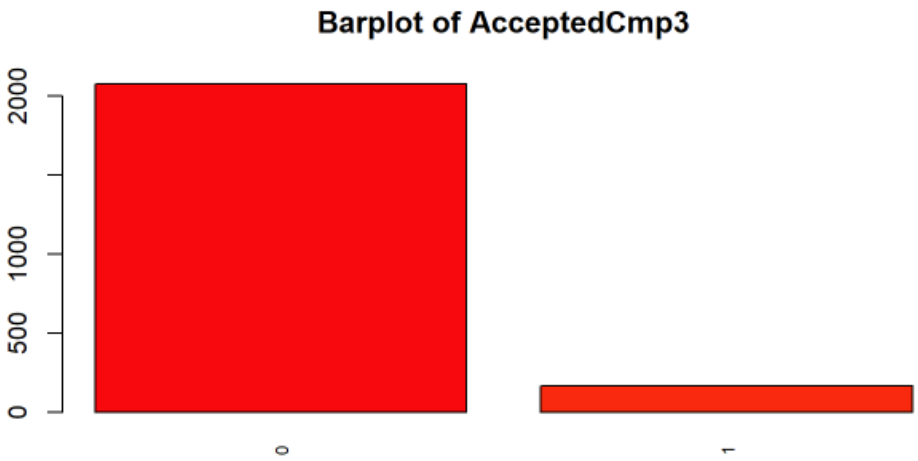

```
##
## Extended Summary Statistics
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00    3.00    5.00    5.79    8.00   13.00
##
## sd: 3.250958
## vc (sd/mean): 0.561461
##
##
## ## Variable 20 — NumWebVisitsMonth
```



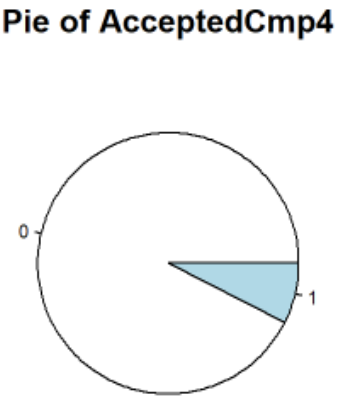


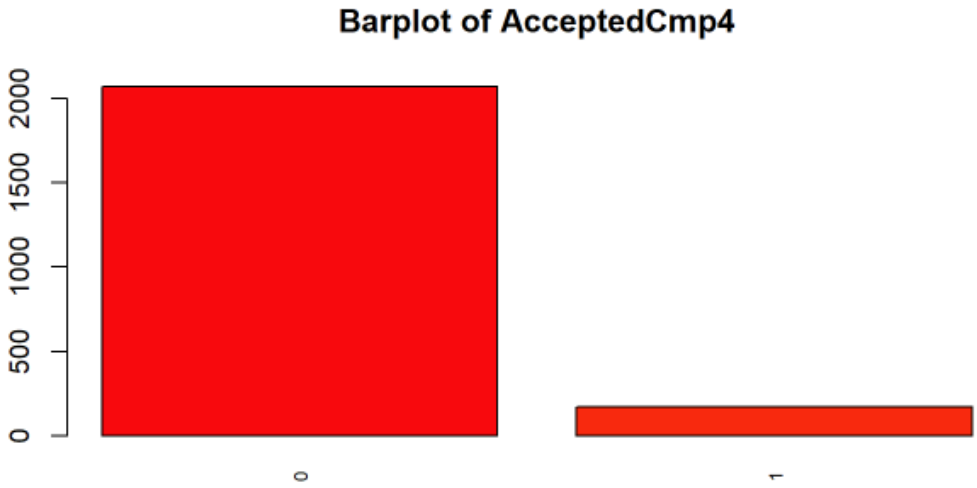

```
##
## Extended Summary Statistics
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   3.000   6.000   5.317   7.000  20.000
##
## sd: 2.426645
## vc (sd/mean): 0.456435
##
##
## ## Variable 21 – AcceptedCmp3
```


Pie of AcceptedCmp3
0
1

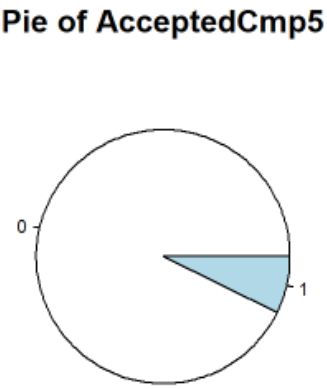

Barplot of AcceptedCmp3
0
500
1000
2000
0
1

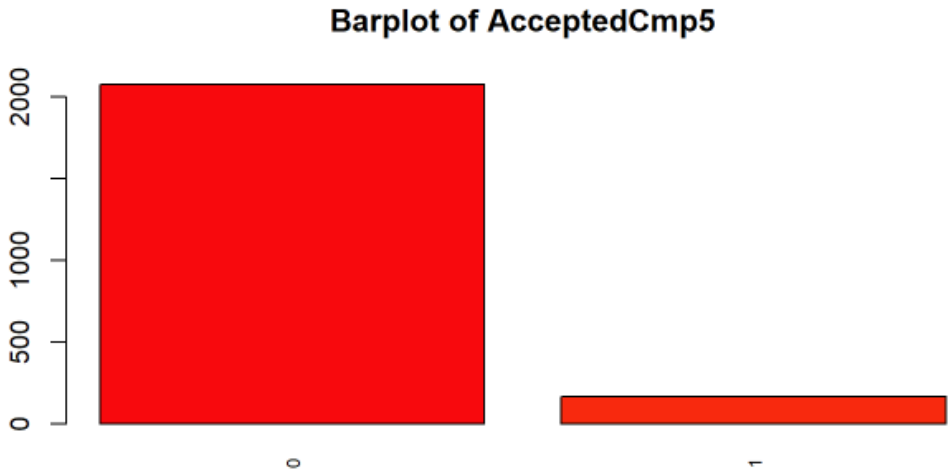
```
##
## Number of modalities: 2
##
## Frequency table
##
##    0    1
## 2077  163
##
## Relative frequency table (proportions)
##
##      0      1
## 0.9272 0.0728
##
## Frequency table sorted
##
##    0    1
## 2077  163
##
## Relative frequency table (proportions) sorted
##
##      0      1
## 0.9272 0.0728
##
##
## ## Variable 22 – AcceptedCmp4
```


Pie of AcceptedCmp4
0
1

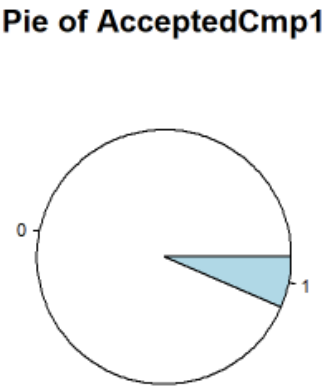

Barplot of AcceptedCmp4
0
500
1000
1500
2000
0
1

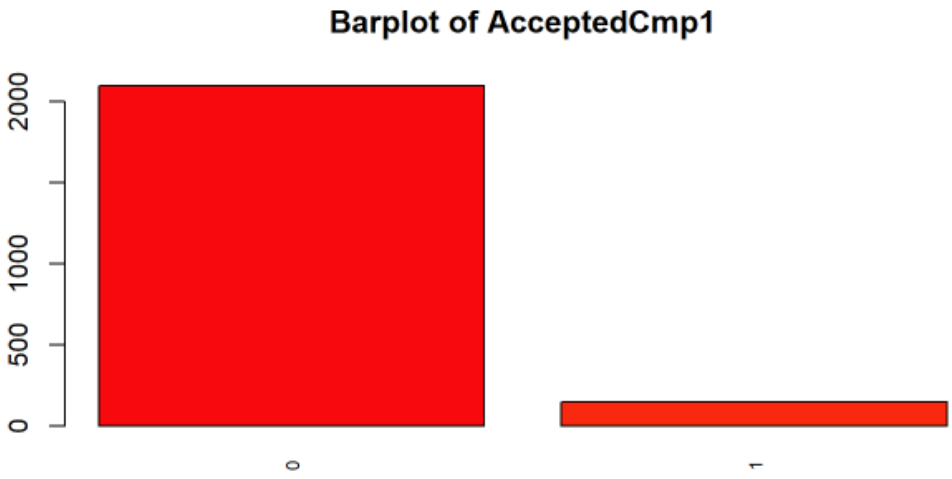
```
##
## Number of modalities: 2
##
## Frequency table
##
##    0    1
## 2073  167
##
## Relative frequency table (proportions)
##
##      0      1
## 0.9254 0.0746
##
## Frequency table sorted
##
##    0    1
## 2073  167
##
## Relative frequency table (proportions) sorted
##
##      0      1
## 0.9254 0.0746
##
##
## ## Variable 23 - AcceptedCmp5
```


Pie of AcceptedCmp5
0
1

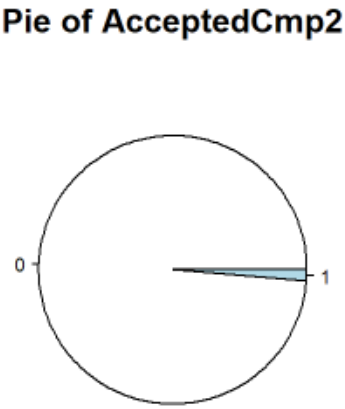

Barplot of AcceptedCmp5
0
500
1000
2000
0
1

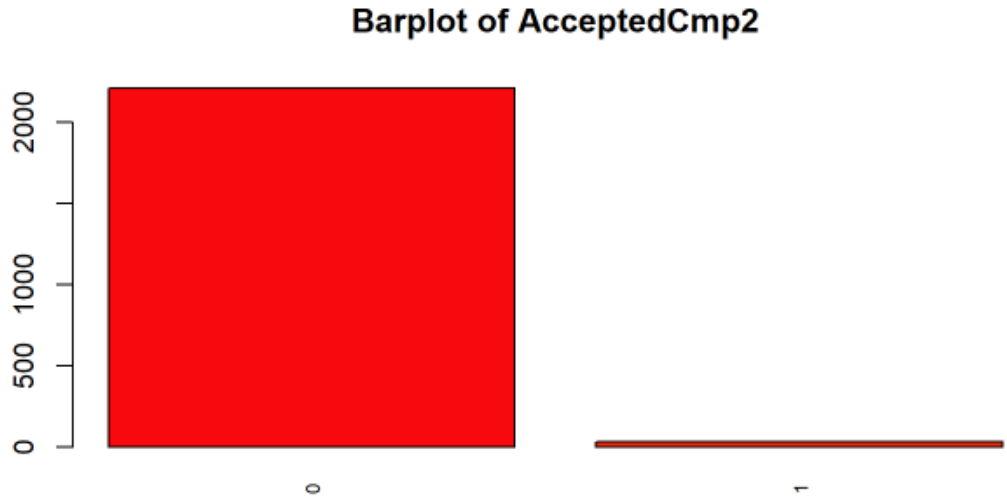
```
##
## Number of modalities: 2
##
## Frequency table
##
##    0    1
## 2077  163
##
## Relative frequency table (proportions)
##
##      0      1
## 0.9272 0.0728
##
## Frequency table sorted
##
##    0    1
## 2077  163
##
## Relative frequency table (proportions) sorted
##
##      0      1
## 0.9272 0.0728
##
##
## ## Variable 24 — AcceptedCmp1
```


Pie of AcceptedCmp1
0
1

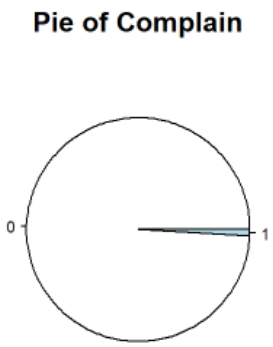

Barplot of AcceptedCmp1
0
500
1000
2000
0
1

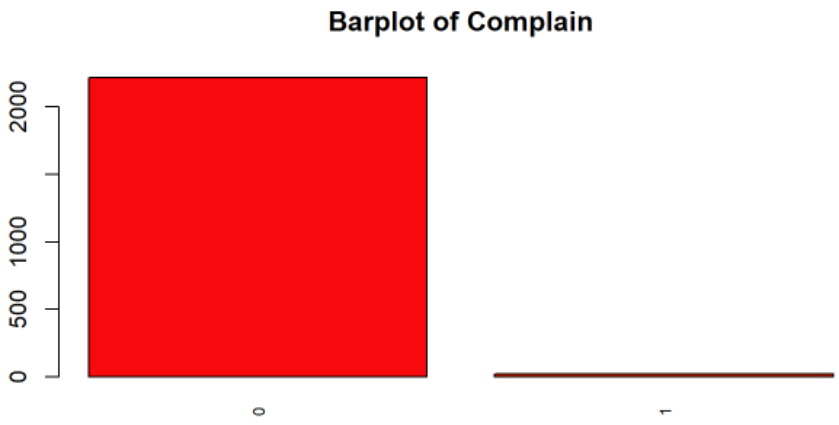
```
##
## Number of modalities: 2
##
## Frequency table
##
##    0    1
## 2096  144
##
## Relative frequency table (proportions)
##
##      0      1
## 0.9357 0.0643
##
## Frequency table sorted
##
##    0    1
## 2096  144
##
## Relative frequency table (proportions) sorted
##
##      0      1
## 0.9357 0.0643
##
##
## ## Variable 25 — AcceptedCmp2
```


Pie of AcceptedCmp2
0
1

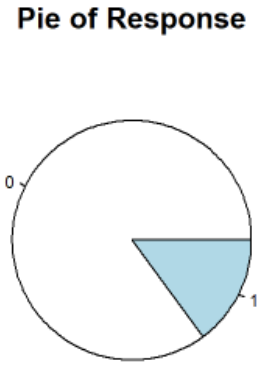

Barplot of AcceptedCmp2
0
500
1000
2000
0
1

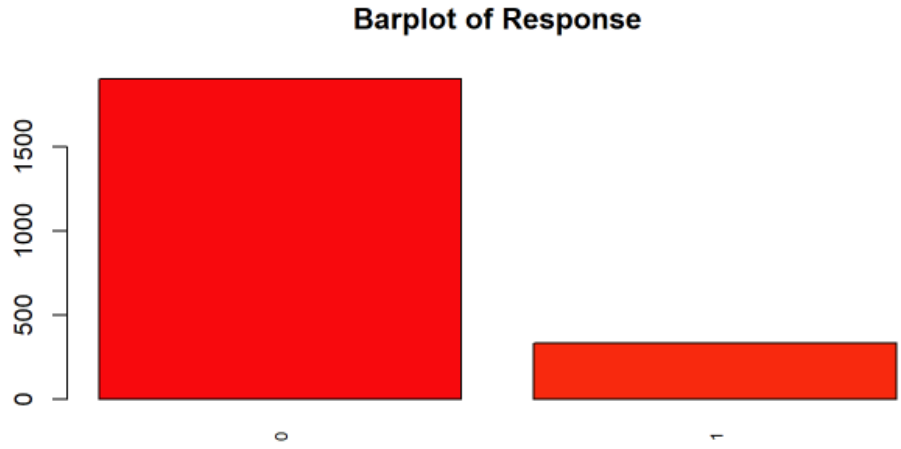
```
##
## Number of modalities: 2
##
## Frequency table
##
##    0    1
## 2210   30
##
## Relative frequency table (proportions)
##
##      0      1
## 0.9866 0.0134
##
## Frequency table sorted
##
##    0    1
## 2210   30
##
## Relative frequency table (proportions) sorted
##
##      0      1
## 0.9866 0.0134
##
##
## ## Variable 26 – Complain
```





```
##
## Number of modalities: 2
##
## Frequency table
##
##    0    1
## 2219   21
##
## Relative frequency table (proportions)
##
##      0      1
## 0.9906 0.0094
##
## Frequency table sorted
##
##    0    1
## 2219   21
##
## Relative frequency table (proportions) sorted
##
##      0      1
## 0.9906 0.0094
##
##
## ## Variable 29 - Response
```






```
##
## Number of modalities: 2
##
## Frequency table
##
##    0    1
## 1906  334
##
## Relative frequency table (proportions)
##
##      0      1
## 0.8509 0.1491
##
## Frequency table sorted
##
##    0    1
## 1906  334
##
## Relative frequency table (proportions) sorted
##
##      0      1
## 0.8509 0.1491
```

1.7 Appendix: Quick numeric summary table

```
num_vars <- names(dd)[sapply(dd, is.numeric)]
if (length(num_vars)) {
  num_summary <- dd %>%
    summarise(across(all_of(num_vars),
                     list(min = ~min(.x, na.rm = TRUE),
                          q1  = ~quantile(.x, 0.25, na.rm = TRUE),
                          mean= ~mean(.x, na.rm = TRUE),
                          median= ~median(.x, na.rm = TRUE),
                          q3  = ~quantile(.x, 0.75, na.rm = TRUE),
                          max = ~max(.x, na.rm = TRUE),
                          sd  = ~sd(.x, na.rm = TRUE)), .names = "{.col}_{.fn}")) %>%
    pivot_longer(everything(), names_to = c("variable","stat"), names_sep = "_") %>%
    pivot_wider(names_from = stat, values_from = value)
  theme_table(num_summary)
} else {
  cat("No numeric variables detected.")
}
```

variable	min	q1	mean	median	q3	max	sd	Birth	CostContact	Revenue
ID	0	2828.25	5592.16	5458.5	8427.75	11191	3246.662	NULL	NULL	NULL
Year	NULL	NULL	NULL	NULL	NULL	NULL	NULL	1893.00000, 1959.00000, 1968.80580, 1970.00000, 1977.00000, 1996.00000, 11.98407	NULL	NULL
Income	1730	35303	52247.25	51381.5	68522	666666	25173.08	NULL	NULL	NULL
Kidhome	0	0	0.4441964	0	1	2	0.5383981	NULL	NULL	NULL
Teenhome	0	0	0.50625	0	1	2	0.5445382	NULL	NULL	NULL
Recency	0	24	49.10938	49	74	99	28.96245	NULL	NULL	NULL
MntWines	0	23.75	303.9357	173.5	504.25	1493	336.5974	NULL	NULL	NULL
MntFruits	0	1	26.30223	8	33	199	39.77343	NULL	NULL	NULL
MntMeatProducts	0	16	166.95	67	232	1725	225.7154	NULL	NULL	NULL
MntFishProducts	0	3	37.52545	12	50	259	54.62898	NULL	NULL	NULL
MntSweetProducts	0	1	27.06295	8	33	263	41.2805	NULL	NULL	NULL
MntGoldProds	0	9	44.02188	24	56	362	52.16744	NULL	NULL	NULL
NumDealsPurchases	0	1	2.325	2	3	15	1.932238	NULL	NULL	NULL
NumWebPurchases	0	2	4.084821	4	6	27	2.778714	NULL	NULL	NULL
NumCatalogPurchases	0	0	2.662054	2	4	28	2.923101	NULL	NULL	NULL
NumStorePurchases	0	3	5.790179	5	8	13	3.250958	NULL	NULL	NULL
NumWebVisitsMonth	0	3	5.316518	6	7	20	2.426645	NULL	NULL	NULL
Z	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	3, 3, 3, 3, 3, 3, 0	11, 11, 11, 11, 11, 11, 0

3 Preprocessing

These are the steps we followed to carry out the preprocessing of the data. The goal of this stage is to clean, transform, and prepare the dataset, ensuring that it is consistent, reliable, and ready for the subsequent data mining tasks.

3.1 Step 1: Getting data

The source of the dataset used in our project comes from kaggle.com, an online platform that provides datasets, competitions, and tools for data science and machine learning. Our dataset focuses on marketing data, which will be analyzed to extract valuable insights.

3.2 Step 2: Visualizing data

To visualize our data, we represented it using different types of graphs, in addition to the metadata from the previous section.

3.3 Step 3: Filtering variables selection

To make them easier to see and understand, we had to transform or rename a lot of variables.

- Age Calculation: ifood$Age <- 2020 - ifood$Year_Birth creates a new column Age by subtracting the Year_Birth from the year 2020. The original Year_Birth column is then dropped to avoid data redundancy.
- Days as a Customer: ifood$CustDays <- as.numeric(reference_date - as.Date(ifood$Dt_Customer, format="%Y-%m-%d")) calculates the number of days a customer has been registered. It subtracts the customer's registration date (Dt_Customer) from a set reference_date (December 31, 2020), converting a date variable into a useful numerical one. The original Dt_Customer column is also removed.

Finally, we shortened most variable names to make them easier to represent in graphs, tables, etc.

Load required libraries

```
# Suppress startup messages of library dplyr
suppressPackageStartupMessages(library(dplyr))
# Loading required libraries
library(dplyr, quietly = TRUE)
library(class, quietly = TRUE)
```

0. Load raw dataset

```
ifood <- read.csv("ifood_base.csv", sep=",", header=TRUE, stringsAsFactors = FALSE)
```

1. Remove irrelevant columns

```
ifood <- ifood[, !names(ifood) %in% c("ID", "Z_CostContact", "Z_Revenue")]
```

2. Transform date-related variables

```
ifood$Age <- 2020 - ifood$Year_Birth
ifood <- ifood[, !names(ifood) %in% c("Year_Birth")]
reference_date <- as.Date("2020-12-31")
ifood$CustDays <- as.numeric(reference_date - as.Date(ifood$Dt_Customer, format="%Y-%m-%d"))
ifood <- ifood[, !names(ifood) %in% c("Dt_Customer")]
```

3. Rename columns for easier access

```
colnames(ifood) <- gsub("NumDealsPurchases", "DealsPurc", colnames(ifood))
colnames(ifood) <- gsub("NumWebPurchases", "WebPurc", colnames(ifood))
colnames(ifood) <- gsub("NumStorePurchases", "StorePurc", colnames(ifood))
colnames(ifood) <- gsub("NumWebVisitsMonth", "WebVisits", colnames(ifood))
colnames(ifood) <- gsub("AcceptedCmpOverall", "CmpOverall", colnames(ifood))
colnames(ifood) <- gsub("MntWines", "WineExp", colnames(ifood))
colnames(ifood) <- gsub("MntFruits", "FruitExp", colnames(ifood))
colnames(ifood) <- gsub("MntMeatProducts", "MeatExp", colnames(ifood))
colnames(ifood) <- gsub("MntFishProducts", "FishExp", colnames(ifood))
colnames(ifood) <- gsub("MntSweetProducts", "SweetExp", colnames(ifood))
colnames(ifood) <- gsub("MntGoldProds", "GoldExp", colnames(ifood))
colnames(ifood) <- gsub("Marital_Status", "MaritalSts", colnames(ifood))
colnames(ifood) <- gsub("NumCatalogPurchases", "CatalogPurc", colnames(ifood))
colnames(ifood) <- gsub("AcceptedCmp1", "AccCmp1", colnames(ifood))
colnames(ifood) <- gsub("AcceptedCmp2", "AccCmp2", colnames(ifood))
colnames(ifood) <- gsub("AcceptedCmp3", "AccCmp3", colnames(ifood))
colnames(ifood) <- gsub("AcceptedCmp4", "AccCmp4", colnames(ifood))
colnames(ifood) <- gsub("AcceptedCmp5", "AccCmp5", colnames(ifood))
```

3.4 Step 4: Missing detection and treatment

During the preprocessing stage, we identified missing or inconsistent values by visually exploring the data to detect anomalies. For instance, in the variable Marital_Status we found incorrect responses such as 'YOLO' and 'Absurd', which we removed to ensure data consistency.

Due to the fact that the Income cannot be less than 12500, we have had to impute the missing values using the KNN method.

4. Handle outliers

```
ifood$Age <- ifelse(ifood$Age > 80, 80, ifood$Age)
```

5. Handle missing values

```
ifood <- ifood[!ifood$MaritalSts %in% c("YOLO", "Absurd"),]
ifood$MaritalSts[ifood$MaritalSts == "Alone"] <- "Single"
```

6. Impute missing Income using KNN

```
ifood$Income <- ifelse(ifood$Income < 12500, NA, ifood$Income)

num_vars <- sapply(ifood, is.numeric)
complete_vars <- colnames(ifood)[num_vars]
missing_threshold <- 0.2 * nrow(ifood)
complete_vars <- complete_vars[colSums(is.na(ifood[, complete_vars])) < missing_threshold]
aux <- ifood[, complete_vars]

var <- "Income"
aux1 <- aux[!is.na(ifood[[var]]), , drop = FALSE]
aux2 <- aux[is.na(ifood[[var]]), , drop = FALSE]

cols_na <- colnames(aux2)[colSums(is.na(aux2)) > 0]
if (length(cols_na) > 0) {
  aux1 <- aux1[, !(colnames(aux1) %in% cols_na), drop = FALSE]
  aux2 <- aux2[, !(colnames(aux2) %in% cols_na), drop = FALSE]
}

knn_impute <- knn(aux1, aux2, ifood[[var]][!is.na(ifood[[var]])], k = 1)
ifood[[var]][is.na(ifood[[var]])] <- as.numeric(as.character(knn_impute))
```

3.5 Step 5: Outlier detection and treatment

To handle the outliers with older people, we decided to set a limit of 80 years for individuals, so that anyone older than that age is included in the 80-year-old group.

7. Correct calculation of `TotAccCmp`

```
ifood$TotAccCmp <- ifood$AccCmp1 + ifood$AccCmp2 + ifood$AccCmp3 + ifood$AccCmp4 + ifood$AccCmp5
```

8. Remove duplicate records

```
ifood <- ifood %>% arrange(desc(Response)) %>% distinct_at(vars(-Response), .keep_all = TRUE)
```

9. Create `TotalExp` before using it

```
ifood$TotalExp <- rowSums(ifood[, c("WineExp", "FruitExp", "MeatExp", "FishExp", "SweetExp", "GoldExp")], na.rm = TRUE)
```

10. Save cleaned dataset

```
write.csv(ifood, "ifood_cleaned.csv", row.names = FALSE)
```

3.6 Step 6: Feature selection

To clean our variables we decided to remove irrelevant columns:
- ID: The ID column is a unique identifier for each record. It does'nt provide predictive value or relevant information about customer behavior or characteristics.
- Z_CostContact and Z_Revenue: Columns starting with the letter 'Z' are often metadata variables used internally in a database, usually represents auxiliary metrics or internal

calculations that are not useful for predictive analysis. In this case, both are synthetic or aggregated variables, likely related to internal company costs and revenues, which are not relevant to understanding customer behavior.

1. Remove irrelevant columns

```
ifood <- ifood[, !names(ifood) %in% c("ID", "Z_CostContact", "Z_Revenue")]
```

3.7 Step 7: Transformation and new variables

These are the new variables we have created or modified from the existing ones to obtain new information.

Variable Creation

Second-Generation

Total Purchases

```
ifood$TotalPurchases <- ifood$DealsPurc + ifood$WebPurc + ifood$CatalogPurc + ifood$StorePurc
```

Purchase Frequency

```
ifood$PurchaseFrequency <- ifelse(ifood$CustDays > 0, ifood$TotalPurchases / (ifood$CustDays / 30), 0)
```

Preferred Product Category

```
product_categories <- c("WineExp", "FruitExp", "MeatExp", "FishExp", "SweetExp", "GoldExp")
max_index <- apply(ifood[ , product_categories], 1, which.max)
ifood$PreferredProductCategory <- product_categories[max_index]
ifood$PreferredProductCategory <- as.factor(ifood$PreferredProductCategory)
```

Preferred Purchase Channel

```
channels <- c("DealsPurc", "WebPurc", "CatalogPurc", "StorePurc")
max_ch_index <- apply(ifood[ , channels], 1, which.max)
ifood$PreferredChannel <- channels[max_ch_index]
ifood$PreferredChannel <- as.factor(ifood$PreferredChannel)
```

Average Spend Per Purchase

```
ifood$AvgSpendPerPurchase <- ifelse(ifood$TotalPurchases > 0, ifood$TotalExp / ifood$TotalPurchases, 0)
```

HasChildren

```
ifood$HasChildren <- ifelse(ifood$Kidhome + ifood$Teenhome > 0, 1, 0)
```

IncomeSegment

```
income_quantiles <- quantile(ifood$Income, probs = c(0.33, 0.66), na.rm = TRUE)
ifood$IncomeSegment <- cut(ifood$Income, breaks = c(-Inf, income_quantiles[1], income_quantiles[2], Inf),
                           labels = c("Low", "Medium", "High"))
```

CustomerTenure

```
ifood$CustomerTenure <- ifood$CustDays / 365
```

CampaignAcceptanceRate

```
ifood$CampaignAcceptanceRate <- ifelse(ifood$TotAccCmp > 0, ifood$TotAccCmp / 5, 0)
```

Third-Generation

Third-Generation Feature 1: Customer Segmentation via Clustering

Prepare data for clustering: use Recency, TotalPurchases (frequency), and TotalExp (monetary)

```
cluster_data <- ifood %>% select(Recency, TotalPurchases, TotalExp)
```

Scale the data for clustering

```
cluster_data_scaled <- scale(cluster_data)
```

Perform k-means clustering with 3 clusters (as an example)

```
set.seed(123)  # for reproducibility
k3 <- kmeans(cluster_data_scaled, centers = 3, nstart = 25)  # nstart for better convergence
```

Add the cluster assignment as a new feature

```
ifood$CustomerSegment <- as.factor(k3$cluster)
```

(Customers are now labeled 1, 2, or 3 based on their cluster segment)

Third-Generation Feature 2: Propensity Score via Logistic Regression

Fit a logistic regression model to predict campaign response (Response) using relevant features

```
propensity_model <- glm(Response ~ Income + Recency + TotalExp + TotalPurchases + TotAccCmp + Age + MaritalSts,
                        data = ifood, family = binomial)
```

Get predicted probabilities (propensity to respond)

```
ifood$PropensityScore <- predict(propensity_model, ifood, type = "response")
```

(PropensityScore is the model's predicted probability of Response=1 for each customer)

Quick summary of PropensityScore range

```
summary(ifood$PropensityScore)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.00000 0.03488 0.07725 0.15313 0.18046 0.99312
```

Third-Generation Feature 3: Engagement Index

Normalize components between 0 and 1

Note: For Recency, a lower value means more recent (more engaged), so we invert it.

```
recency_norm   <- (max(ifood$Recency) - ifood$Recency) / max(ifood$Recency)          # invert recency
frequency_norm <- ifood$TotalPurchases / max(ifood$TotalPurchases)                   # purchases normalized
monetary_norm  <- ifood$TotalExp / max(ifood$TotalExp)                               # spending normalized
campaign_norm  <- (ifood$TotAccCmp + ifood$Response) / 6                             # campaign acceptance (out of 6 campaig
ns total including last response)
webvisit_norm  <- ifood$WebVisits / max(ifood$WebVisits)                             # web visits normalized
```

Calculate engagement index as average of all five components, scaled to 0-100

```
ifood$EngagementIndex <- (recency_norm + frequency_norm + monetary_norm + campaign_norm + webvisit_norm) / 5 * 100
```

Preview EngagementIndex distribution

```
summary(ifood$EngagementIndex)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.892  20.859  27.401  28.453  35.209  62.573
```

Save enriched dataset

```
write.csv(ifood, "ifood_enriched.csv", row.names = FALSE)
```