Quantium Virtual Internship - Retail Strategy and Analytics - Task 1

Solution template for Task 1

This file is a solution template for the Task 1 of the Quantium Virtual Internship. It will walk you through the analysis, providing the scaffolding for your solution with gaps left for you to fill in yourself.

Look for comments that say "over to you" for places where you need to add your own code! Often, there will be hints about what to do or what function to use in the text leading up to a code block - if you need a bit of extra help on how to use a function, the internet has many excellent resources on R coding, which you can find using your favourite search engine.

Load required libraries and datasets

Note that you will need to install these libraries if you have never used these before.

```
#### Example code to install packages
#install.packages("data.table")
#### Load required libraries
library("data.table")
library(ggplot2)
library(ggmosaic)
library(readr)
#### Point the filePath to where you have downloaded the datasets to and
#### assign the data files to data.tables
# over to you! fill in the path to your working directory. If you are on a
↪  Windows machine, you will need to use forward slashes (/) instead of
↪  backshashes (\)
filePath <- paste0(getwd(),"/")
transactionData <- fread(file = paste0(filePath,"QVI_transaction_data.xlsx -
↪  in.csv"), header = TRUE)
customerData <- fread(file = paste0(filePath,"QVI_purchase_behaviour.csv"),
↪  header = TRUE)
```

Exploratory data analysis

The first step in any analysis is to first understand the data. Let's take a look at each of the datasets provided. ### Examining transaction data We can use `str()` to look at the format of each column and see a sample of the data. As we have read in the dataset as a `data.table` object, we can also run `transactionData` in the console to see a sample of the data or use `head(transactionData)` to look at the first 10 rows. Let's check if columns we would expect to be numeric are in numeric form and date columns are in date format.

```r
#### Examine transaction data
# Over to you! Examine the data using one or more of the methods described above.

str(transactionData)
```

```
## Classes 'data.table' and 'data.frame': 264836 obs. of 8 variables:
## $ DATE : int 43390 43599 43605 43329 43330 43604 43601 43601 43332 43330 ...
## $ STORE_NBR : int 1 1 1 2 2 4 4 4 5 7 ...
## $ LYLTY_CARD_NBR: int 1000 1307 1343 2373 2426 4074 4149 4196 5026 7150 ...
## $ TXN_ID : int 1 348 383 974 1038 2982 3333 3539 4525 6900 ...
## $ PROD_NBR : int 5 66 61 69 108 57 16 24 42 52 ...
## $ PROD_NAME : chr "Natural Chip Compny SeaSalt175g" "CCs Nacho Cheese 175g"
"Smiths Crinkle Cut Chips Chicken 170g" "Smiths Chip Thinly S/Cream&Onion 175g"
...
## $ PROD_QTY : int 2 3 2 5 3 1 1 1 1 2 ...
## $ TOT_SALES : num 6 6.3 2.9 15 13.8 5.1 5.7 3.6 3.9 7.2 ...
## - attr(*, ".internal.selfref")=<externalptr>
```

```r
transactionData
```

```
##           DATE STORE_NBR LYLTY_CARD_NBR TXN_ID PROD_NBR
##      1: 43390         1           1000      1        5
##      2: 43599         1           1307    348       66
##      3: 43605         1           1343    383       61
##      4: 43329         2           2373    974       69
##      5: 43330         2           2426   1038      108
##     ---
## 264832: 43533       272         272319 270088       89
## 264833: 43325       272         272358 270154       74
## 264834: 43410       272         272379 270187       51
## 264835: 43461       272         272379 270188       42
## 264836: 43365       272         272380 270189       74
##                                        PROD_NAME PROD_QTY TOT_SALES
##      1:   Natural Chip        Compny SeaSalt175g        2       6.0
##      2:                 CCs Nacho Cheese    175g        3       6.3
##      3:   Smiths Crinkle Cut  Chips Chicken 170g        2       2.9
##      4:   Smiths Chip Thinly  S/Cream&Onion 175g        5      15.0
##      5: Kettle Tortilla ChpsHny&Jlpno Chili 150g        3      13.8
##     ---
## 264832:  Kettle Sweet Chilli And Sour Cream 175g        2      10.8
## 264833:            Tostitos Splash Of  Lime 175g        1       4.4
## 264834:                 Doritos Mexicana    170g        2       8.8
## 264835:  Doritos Corn Chip Mexican Jalapeno 150g        2       7.8
## 264836:            Tostitos Splash Of  Lime 175g        2       8.8
```

We can see that the date column is in an integer format. Let’s change this to a date format.

```r
#### Convert DATE column to a date format
#### A quick search online tells us that CSV and Excel integer dates begin on 30
↪  Dec 1899

transactionData$DATE <- as.Date(transactionData$DATE, origin = "1899-12-30")
```

We should check that we are looking at the right products by examining PROD_NAME.

```
#### Examine PROD_NAME
# Over to you! Generate a summary of the PROD_NAME column.

transactionData$PROD_NAME[1:10]
```

```
##  [1] "Natural Chip        Compny SeaSalt175g"
##  [2] "CCs Nacho Cheese    175g"
##  [3] "Smiths Crinkle Cut  Chips Chicken 170g"
##  [4] "Smiths Chip Thinly  S/Cream&Onion 175g"
##  [5] "Kettle Tortilla ChpsHny&Jlpno Chili 150g"
##  [6] "Old El Paso Salsa   Dip Tomato Mild 300g"
##  [7] "Smiths Crinkle Chips Salt & Vinegar 330g"
##  [8] "Grain Waves         Sweet Chilli 210g"
##  [9] "Doritos Corn Chip Mexican Jalapeno 150g"
## [10] "Grain Waves Sour    Cream&Chives 210G"
```

Looks like we are definitely looking at potato chips but how can we check that these are all chips? We can do some basic text analysis by summarizing the individual words in the product name.

```
#### Examine the words in PROD_NAME to see if there are any incorrect entries
#### such as products that are not chips

# make dataframe of all chars in all data names
productWords <- data.table(unlist(strsplit(unique(transactionData[,
 ↪  PROD_NAME]), "")))

# set column name as words
setnames(productWords, 'words')
```

As we are only interested in words that will tell us if the product is chips or not, let’s remove all words with digits and special characters such as ‘&’ from our set of product words. We can do this using `grepl()`.