Analytics Programming Assignment

Manan Bhatia

2023-05-16

- We hold a copy of this assignment that we can produce if the original is lost or damaged.
- We hereby certify that no part of this assignment/product has been copied from any other student's work or from any other source except where due acknowledgement is made in the assignment.
- No part of this assignment/product has been written/produced for us by another person except where such collaboration has been authorised by the subject lecturer/tutor concerned.
- We are aware that this work may be reproduced and submitted to plagiarism detection software programs for the purpose of detecting possible plagiarism (which may retain a copy on its database for future plagiarism checking).
- We hereby certify that we have read and understand what the School of Computing, Engineering and Mathematics defines as minor and substantial breaches of misconduct as outlined in the learning guide for this unit

```r
product = read.csv("product_hierarchy.csv")
sale = read.csv("sales_ug.csv")
store = read.csv("store_cities.csv")
```

Comment: Before starting the first question, I first defined the 3 data set csv files given with product, sale and store.

```r
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.2     v tibble    3.2.1
## v lubridate 1.9.4     v tidyr     1.3.1
## v purrr     1.0.4
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```r
library(conflicted)
library(dplyr)
```

1. Write the code to compute the total revenue of each store at the end of each day. Is there a noted difference between the days? Write also the code to calculate the total revenue over the seven day period. Plot the latter on a graph.

Viewing the overall information about the dataset sale (daily sales record of data over a seven day period)

```r
#viewing the dataset
head(sale, 10) #head(..., 10) shows the first 10 rows of dataset a
```

```
##    product_id store_id       date sales revenue stock  price promo_type_1
## 1       P0001    S0002 2017-07-03     0       0     1   6.75         PR14
## 2       P0001    S0038 2017-07-03     0       0     1   6.75         PR14
## 3       P0001    S0040 2017-07-03     0       0     2   6.75         PR14
## 4       P0001    S0050 2017-07-03     0       0     1   6.75         PR14
## 5       P0001    S0103 2017-07-03     0       0    10   6.75         PR14
## 6       P0001    S0105 2017-07-03     0       0     5   6.75         PR14
## 7       P0002    S0038 2017-07-03     0       0    24 349.00         PR14
## 8       P0002    S0085 2017-07-03     0       0    25 349.00         PR14
## 9       P0004    S0085 2017-07-03     0       0     7   4.50         PR14
## 10      P0005    S0001 2017-07-03     0       0     3  33.90         PR14
##    promo_bin_1 promo_discount_2 promo_discount_type_2
## 1                            NA                    NA
## 2                            NA                    NA
## 3                            NA                    NA
## 4                            NA                    NA
## 5                            NA                    NA
## 6                            NA                    NA
## 7                            NA                    NA
## 8                            NA                    NA
## 9                            NA                    NA
## 10                           NA                    NA
```

```r
#structure of the dataset
str(sale) #show the type of data of the variables
```

```
## 'data.frame':    104000 obs. of  11 variables:
##  $ product_id           : chr  "P0001" "P0001" "P0001" "P0001" ...
##  $ store_id             : chr  "S0002" "S0038" "S0040" "S0050" ...
##  $ date                 : chr  "2017-07-03" "2017-07-03" "2017-07-03" "2017-07-03" ...
##  $ sales                : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ revenue              : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ stock                : num  1 1 2 1 10 5 24 25 7 3 ...
##  $ price                : num  6.75 6.75 6.75 6.75 6.75 6.75 349 349 4.5 33.9 ...
##  $ promo_type_1         : chr  "PR14" "PR14" "PR14" "PR14" ...
##  $ promo_bin_1          : chr  "" "" "" "" ...
##  $ promo_discount_2     : logi  NA NA NA NA NA NA ...
##  $ promo_discount_type_2: logi  NA NA NA NA NA NA ...
```

Total revenue of each store at the end of each day

To calculate the revenue of each store at the end of each day, using `aggregate()` is the best choice of algorithm, as it can split data into subsets and compute summary statistics for each.

The function below summarise the statistic of revenue based on the store_id and date variables. In this case, it sums the total revenue made based on the store_id and date.

```
revenue_each_day <- aggregate(revenue ~ store_id + date, #calculate revenue based on
                              #store_id and date variables
                              data = sale,
                              FUN = sum) #summation is abbreviated to sum
head(revenue_each_day, 10)
```

```
##    store_id       date revenue
## 1     S0001 2017-07-03  767.99
## 2     S0002 2017-07-03  346.82
## 3     S0003 2017-07-03   94.43
## 4     S0004 2017-07-03  461.42
## 5     S0006 2017-07-03   56.45
## 6     S0008 2017-07-03  221.52
## 7     S0009 2017-07-03   19.50
## 8     S0010 2017-07-03  255.77
## 9     S0011 2017-07-03  102.58
## 10    S0012 2017-07-03  216.28
```

The above table demonstrates the total revenue of each store profited by the end of each day, starting from date 3 June to 9 June of 2017.

The stores are shown by `store_id` while the `date` shows the days for which the `revenue` is shown. For example:

- Store with unique identifier number of S0001 obtained a total revenue of 767.99 on the date 2017-07-03.
- Store with unique identifier number of S0002 obtained a total revenue of 346.82 on the date 2017-07-03.
- Store with unique identifier number of S0115 obtained a total revenue of 908.29 on the date 2017-07-03. And so on.

Differences in revenues between the day?

To see the difference in revenues between the day, we can use `tapply()` to provide mathematical function to columns that use the function. In this example, `diff` is a function value that is used to calculated the differences in revenues obtained between each row where `store_id` is matched with the previous row.

```
tapply(revenue_each_day$revenue,
       revenue_each_day$store_id,
       diff) %>% #each array element represents the difference in revenue between
  head(10)       #the current day and the next day
```

```
## $S0001
## [1]  528.37 -290.51 -112.30  354.33  299.45  -82.10
##
## $S0002
## [1] -120.64  -50.70   87.11 -121.13  444.79 -202.29
##
## $S0003
## [1]  27.28  -9.50 -71.73  55.07 -35.48  19.24
##
## $S0004
## [1] -324.83   -9.83  -14.94   29.68  182.01 -156.84
##
## $S0006
## [1] -29.64  43.70  -1.36 -21.83 -11.78  -6.33
##
## $S0008
## [1] -27.40 -87.07 100.93  57.08 -15.42 -55.36
##
## $S0009
## [1]  -3.02  38.41 -10.17 -19.56  10.57  37.89
##
## $S0010
## [1]   9.11 -87.39 -10.11  74.18 173.72 131.48
##
## $S0011
## [1]  16.62  16.72 -15.13  -7.99 -59.78  34.35
##
## $S0012
## [1] -115.96   39.98    5.28  -44.74  188.43 -150.29
```

The table above shows the differences in revenues of each store between the day. For example:

- Store with `store_id` S0001 has 6 returned values:
    - The first value means the difference in revenues between day 1 and day 2 is $528.37, implying that day 2 total revenue obtained is about $528.37 more than day 1.
    - The second value means the difference in revenues between day 2 and day 3 is $-290.51, meaning that day 3 total revenue obtained is about 290.51 less than day 2.

In this example, `tapply()` returns values in the form of arrays. It is a poor way to arrange data, however this is the only current available option for my personal choice of algorithm.

```r
class(tapply(revenue_each_day$revenue, revenue_each_day$store_id, diff))
```

```
## [1] "array"
```

```r
#returns values in the form of arrays.
```

Total revenue generated from each store over seven days

We will use `aggregate()` function to calculate the total revenue obtained in corresponds with each store's `store_id`.

```r
#summarise the total revenue made from each store_id over the seven days
seven <- aggregate(revenue ~ store_id,
                           data = sale,
                           FUN = sum) #use sum to calculate the total revenue
head(seven, 10)
```

```
##    store_id revenue
## 1     S0001 8224.19
## 2     S0002 2122.74
## 3     S0003  603.76
## 4     S0004 1468.27
## 5     S0006  334.99
## 6     S0008 1439.65
## 7     S0009  270.10
## 8     S0010 2069.12
## 9     S0011  731.68
## 10    S0012 1131.57
```

The above table portrays the first 10 values of the total revenue of each store over the seven day period. For example:

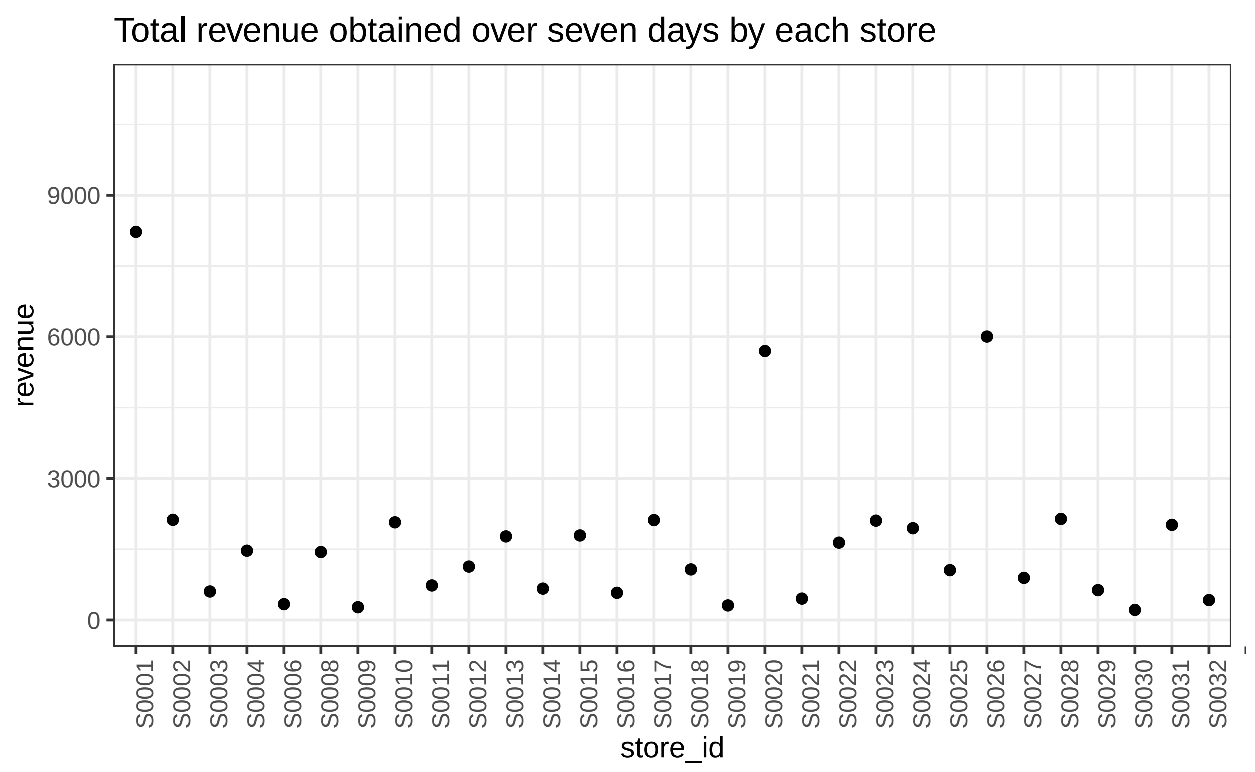
- Store with `store_id` (unique identifier number) of S0001 has gained a total revenue of 8224.19.
- Store with `store_id` of S0002 has gained a total revenue of 2122.74.
- Store with `store_id` of S0056 has gained a total revenue of 2175.47.

And so on

Plotting:

We could use `ggplotly` to interact with graph in other form of document (html) but not in any word or pdf document. However, we still include it to see the overall plotting of points of revenue by each store.

```r
#plotting the total revenue over the seven day period
ggplot(seven, aes(store_id, revenue)) +
  geom_point() +
  theme_bw() +
  theme(axis.text.x = element_text(angle = 90)) +
  coord_cartesian(xlim = c(1, 30)) + #showing the revenues obtained by the first 30 stores
  labs(title = "Total revenue obtained over seven days by each store",
       caption = "The plot shows only the first 30 stores' revenues due to overloading of data.
       Note: revenue - daily total sales revenue
             store_id - unique identifier of a store")
```



The plot shows only the first 30 stores' revenues due to overloading of data.
Note: revenue − daily total sales revenue
store_id − unique identifier of a store

Most of the time, we see that most stores' revenue accumulate below the mark of $3000. However, some stores are distinct, where revenues obtained could go higher than the mark of $3000 and potentially could reach the mark of $9000 in total revenue. For instance, in the total revenue table above (section 1 - part 3), the store with store_id of S0001 has gained a total of $8224.19 in term of total revenue over the past seven days.

2. What's the most popular product type (hierarchy 1) sold in all stores over a week? How much revenue did the stores receive for that product during the week? How does that compare with the second most popular product? Provide a table that shows the product type ranked from most to least popular. For each product type provide: how many subtypes (hierarchy 2) are there, how many products are in this product type, what's the sales quantity, and the revenue generated.

Viewing information about the dataset product (product_hierarchy data)

```r
#viewing the dataset
head(product, 10) #shows the first 10 variables of dataset b
```

```
##    product_id product_length product_depth product_width cluster_id
## 1       P0000            5.0            20          12.0
## 2       P0001           13.5            22          20.0  cluster_5
## 3       P0002           22.0            40          22.0  cluster_0
## 4       P0004            2.0            13           4.0  cluster_3
## 5       P0005           16.0            30          16.0  cluster_9
## 6       P0006            8.5            15          15.0  cluster_0
## 7       P0007            2.0            22           9.5  cluster_4
## 8       P0008            5.0            16           5.0  cluster_0
## 9       P0009            5.0            18          14.0  cluster_6
## 10      P0010            2.0            22           3.0  cluster_0
##    hierarchy1_id hierarchy2_id hierarchy3_id hierarchy4_id hierarchy5_id
## 1            H00         H0004       H000401     H00040105   H0004010534
## 2            H01         H0105       H010501     H01050100   H0105010006
## 3            H03         H0315       H031508     H03150800   H0315080028
## 4            H03         H0314       H031405     H03140500   H0314050003
## 5            H03         H0312       H031211     H03121109   H0312110917
## 6            H03         H0316       H031608     H03160817   H0316081708
## 7            H03         H0313       H031305     H03130519   H0313051904
## 8            H00         H0000       H000004     H00000400   H0000040017
## 9            H00         H0002       H000201     H00020100   H0002010012
## 10           H01         H0108       H010801     H01080109   H0108010917
```

```r
#structure of the dataset
str(product) #shows the structure of b and its datax
```

```
## 'data.frame':    699 obs. of  10 variables:
##  $ product_id    : chr  "P0000" "P0001" "P0002" "P0004" ...
##  $ product_length: num  5 13.5 22 2 16 8.5 2 5 5 2 ...
##  $ product_depth : num  20 22 40 13 30 15 22 16 18 22 ...
##  $ product_width : num  12 20 22 4 16 15 9.5 5 14 3 ...
##  $ cluster_id    : chr  "" "cluster_5" "cluster_0" "cluster_3" ...
##  $ hierarchy1_id : chr  "H00" "H01" "H03" "H03" ...
##  $ hierarchy2_id : chr  "H0004" "H0105" "H0315" "H0314" ...
##  $ hierarchy3_id : chr  "H000401" "H010501" "H031508" "H031405" ...
##  $ hierarchy4_id : chr  "H00040105" "H01050100" "H03150800" "H03140500" ...
##  $ hierarchy5_id : chr  "H0004010534" "H0105010006" "H0315080028" "H0314050003" ...
```

The most popular product type (hierarchy 1) sold in all stores over a week

Joining two datasets a and b based on their corresponding variables

In this case the corresponding key is product_id, and the joining variables are hierarchy1_id and hierarchy2_id.

```r
merged_sale_product_tab <- product %>%
  select("product_id", "hierarchy1_id", "hierarchy2_id") %>%
  right_join(sale)
```

```
## Joining with `by = join_by(product_id)`
```

To check for the popularity ranking of the product type (hierarchy 1) in terms of selling, we use `sort()` to sort table values. By using `decreasing = TRUE` as additional argument, it sorts table values from the highest to the lowest.

```r
sort(table(merged_sale_product_tab$hierarchy1_id), decreasing = TRUE)
```

```
##
##   H00   H01   H03   H02
## 52395 29748 21494   363
```

As it can be seen in the above table, the most sold product type in all stores is `H03` with over 52395 items sold over the week. And the second most popular product type sold is `H01` with 29748 items sold over the week.

How much revenue did the stores receive for that product during the week?

To calculate Revenue received from that product during the week, again, we will use `aggregate()` to summarise the summation statistic of revenue based on the `store_id` and `date`.

```r
#revenue made
stores_rev_made1 <-
  merged_sale_product_tab[which(merged_sale_product_tab$hierarchy1_id == "H00"),] #select rows
#where hiearchy1_id is "H00"

aggregate(revenue ~ store_id + date,
          data = stores_rev_made1,
          sum) %>%
  head(10)#shows the first 10 values of revenues made from products with hierarchy1_id of "H00"
```

```
##    store_id       date revenue
## 1     S0001 2017-07-03  315.09
## 2     S0002 2017-07-03  210.99
## 3     S0003 2017-07-03   85.18
## 4     S0004 2017-07-03  397.83
## 5     S0006 2017-07-03   17.91
## 6     S0008 2017-07-03  117.56
## 7     S0009 2017-07-03   19.50
## 8     S0010 2017-07-03   85.05
## 9     S0011 2017-07-03   74.53
## 10    S0012 2017-07-03  110.24
```

As shown in the table above, Each store has received a various amount of revenue on each day. For instance, Store with the `store_id` of S0001 has made a total of $315.09 on the date of 3/7/2017. While store with the `store_id` of S0006 has only made a total of $17.91 on the date of 3/7/2017 on the same product as the store with `store_id` of S0001. Therefore, the revenues generated by each store are unique.

How does that compare with the second most popular product?

The second most popular product is "H01" according to the sorted table above in task 2, question 1. In the below table, it shows the revenues obtained on each day in each store, by selling the second most popular product "H01".

```r
stores_rev_made2 <- merged_sale_product_tab[which(merged_sale_product_tab$hierarchy1_id == "H01"),] #se
#rows where hierarchy1_id is "H01"

#total revenue made in each store from the products with hierarchy1_id "H01"
aggregate(revenue ~ store_id + date,
          data = stores_rev_made2,
          sum) %>%
  head(10)
```

```
##    store_id       date revenue
## 1     S0001 2017-07-03  184.85
## 2     S0002 2017-07-03   64.96
## 3     S0003 2017-07-03    0.00
## 4     S0004 2017-07-03   41.61
## 5     S0006 2017-07-03    0.00
## 6     S0008 2017-07-03   76.14
## 7     S0009 2017-07-03    0.00
## 8     S0010 2017-07-03   81.91
## 9     S0011 2017-07-03   10.08
## 10    S0012 2017-07-03   83.51
```

In some store, they gained no revenue on this product type, for example:

- S0003 made zero revenue on 3/7/2017.
- S0006 made zero revenue on 3/7/2017, and so on.

Comparison

Assess the number of rows of each aggregated dataset

```r
stores_rev_made1 <- aggregate(revenue ~ store_id + date, data = stores_rev_made1, sum)
stores_rev_made2 <- aggregate(revenue ~ store_id + date, data = stores_rev_made2, sum)
```

We notice that number of rows of each assigned data frame is different due to the lack of recording of information on the date.

```
nrow(stores_rev_made1) #showing the row numbers of stores_rev_made1
```

```
## [1] 886
```

```
nrow(stores_rev_made2) #showing the row numbers of stores_rev_made2
```

```
## [1] 884
```

Merging dataset:

Since the number of rows is different for each set of data, when doing a merging process, we use `full_join` on `store_id` and `date` to have a complete set of data from both sides. Even though there will be NULL variables in some case, but we can set it as 0 since there is no record available. However, we cannot remove NULL variables because there might be records from the other dataset,

```
store_rev_made_12binded <- stores_rev_made1 %>% #joins stores_rev_made1 to stores_rev_made2
  full_join(stores_rev_made2,
            by = c("store_id", "date")) # by "store_id" and "date"

#fix column names
colnames(store_rev_made_12binded) <-
  c("store_id", "date", "H00.revenue", "H01.revenue")

#assign 0 to NA values
store_rev_made_12binded[is.na(store_rev_made_12binded)] <- 0

#shows final result
head(store_rev_made_12binded, 10)
```

```
##    store_id       date H00.revenue H01.revenue
## 1     S0001 2017-07-03      315.09      184.85
## 2     S0002 2017-07-03      210.99       64.96
## 3     S0003 2017-07-03       85.18        0.00
## 4     S0004 2017-07-03      397.83       41.61
## 5     S0006 2017-07-03       17.91        0.00
## 6     S0008 2017-07-03      117.56       76.14
## 7     S0009 2017-07-03       19.50        0.00
## 8     S0010 2017-07-03       85.05       81.91
## 9     S0011 2017-07-03       74.53       10.08
## 10    S0012 2017-07-03      110.24       83.51
```

```
store_rev_made_12binded <- aggregate(cbind(store_rev_made_12binded$H00.revenue,
                                           store_rev_made_12binded$H01.revenue),
                                     by = list(store_rev_made_12binded$store_id),
                                     FUN = sum)

#changes colnames
colnames(store_rev_made_12binded) <- c("store_id","H00.revenue","H01.revenue")

#assign a new column with differences in revenues to store_rev_made_12binded
store_rev_made_12binded[,"revenue.differences"] <-
```

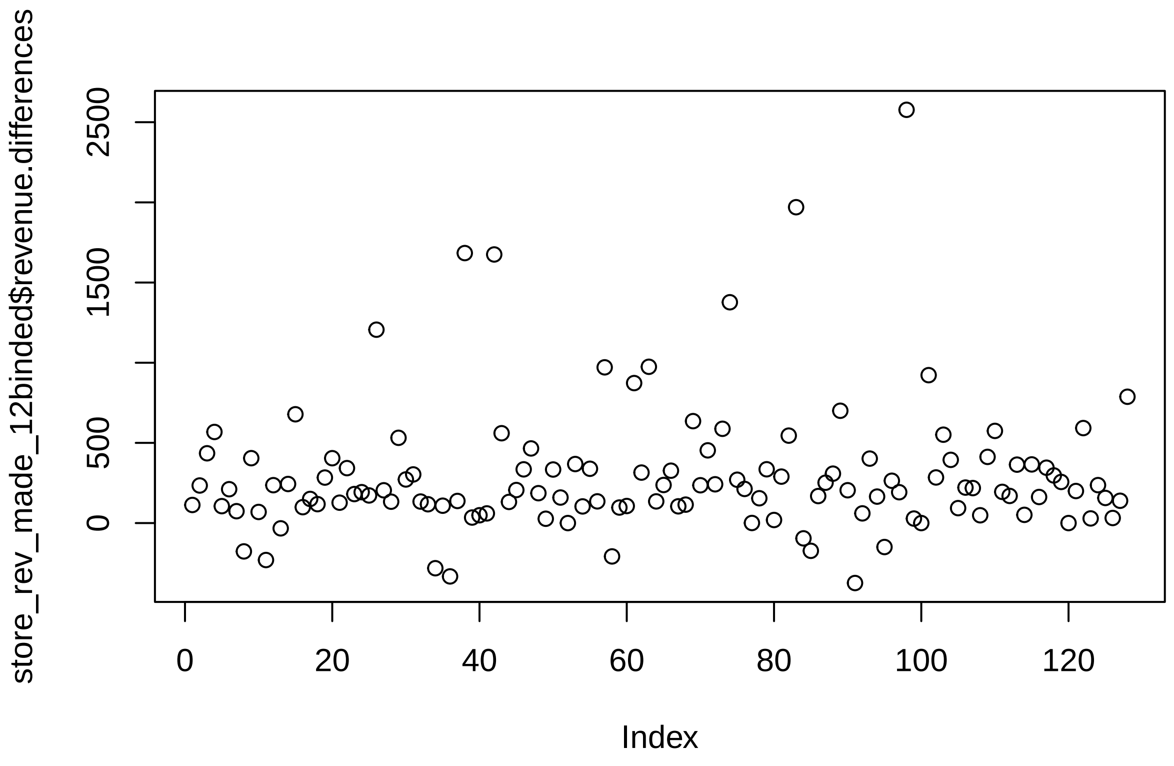
```
  store_rev_made_12binded$H00.revenue - store_rev_made_12binded$H01.revenue

#shows final result
head(store_rev_made_12binded, 10)
```

```
##    store_id H00.revenue H01.revenue revenue.differences
## 1     S0001     2837.56     2724.72              112.84
## 2     S0002     1045.75      811.57              234.18
## 3     S0003      480.16       45.12              435.04
## 4     S0004      921.20      353.10              568.10
## 5     S0006      134.00       28.64              105.36
## 6     S0008      676.91      465.73              211.18
## 7     S0009      156.59       82.66               73.93
## 8     S0010      681.31      858.13             -176.82
## 9     S0011      479.39       74.78              404.61
## 10    S0012      530.59      461.39               69.20
```

Plotting the revenue differences:

```
plot(store_rev_made_12binded$revenue.differences)
```



4) Provide a table showing the product type ranked from most to least popular

Again, we use sort to sort out the ranking of product types based on the number of product they have.

```r
sort(table(merged_sale_product_tab$hierarchy1_id), decreasing = TRUE)
```

```
##
##   H00   H01   H03   H02
## 52395 29748 21494   363
```

The table above shows the ranking of product type from most to least, where the most and least popular product types are H00 and H01.

5) For each product: how many subtypes products are there?

To see how many subtypes products are available and the amount of products in these subtype products, we use `table` to tabulate the occurrence frequency of a data in a variable. In this case, we want to see how often the number of hierarchy2_id occurs, in order to calculate the amount of available products in that subcategory.

```r
matx_1 <- table(product$hierarchy1_id, product$hierarchy2_id)
matx_1
```

```
##
##       H0000 H0001 H0002 H0003 H0004 H0105 H0106 H0107 H0108 H0209 H0210 H0311
##   H00    32    38    54    53    38     0     0     0     0     0     0     0
##   H01     0     0     0     0     0    17    28    40    96     0     0     0
##   H02     0     0     0     0     0     0     0     0     0     4     7     0
##   H03     0     0     0     0     0     0     0     0     0     0     0    51
##
##       H0312 H0313 H0314 H0315 H0316 H0317
##   H00     0     0     0     0     0     0
##   H01     0     0     0     0     0     0
##   H02     0     0     0     0     0     0
##   H03    61   101    28    40     5     6
```

As described in the description of variables, each product has subtype products corresponded to and is categorised into levels of hierarchy. According to the hierarchy table shown above:

- There are 5 subtype products of H00: H0000, H0001, H0002, H0003, H0004.
- There are 4 subtype products of H01: H0105, H0106, H0107, H0108.
- There are 2 subtype products of H02: H0209, H0311.
- There are 7 subtype products of H03: H0311, H0312, H0313, H0314, H0315, H0316, H0317.

6) How many products are in this product type?

As shown in the matrix table `matx_1` above:

- There are 32 items in H0000 (subset of H00).
- There are 38 items in H0001 (subset of H00).
- And so on.

7) Sales quantity:

We use `aggregate()` to calculate the summation of `sales` quantity in correspondence with `hierarchy1_id` subset.

```r
#hierarchy1_id:
aggregate(sales ~ hierarchy1_id, data = merged_sale_product_tab, sum)
```

```
##   hierarchy1_id     sales
## 1           H00 40256.818
## 2           H01  5797.000
## 3           H02  1141.983
## 4           H03  4266.000
```

There are four product types, and each made a unique quantity of sales over the seven days:

- H00 has made a total sale of 4.0256818×10^4.
- H01 has made a total sale of 5797.
- H02 has made a total sale of 1141.983.
- H03 has made a total sale of 4266.

The table below shows the summmation of sales quantity that corresponds to `hierarchy1_id` and `hierarchy2_id`subsets

```r
#hierarchy2_id:
sale_hier2 <- aggregate(sales ~ hierarchy1_id + hierarchy2_id,
                        data = merged_sale_product_tab,
                        sum) %>%
              head(10)
sale_hier2
```

```
##    hierarchy1_id hierarchy2_id     sales
## 1            H00         H0000 13093.000
## 2            H00         H0001  2481.000
## 3            H00         H0002  2955.000
## 4            H00         H0003 17920.000
## 5            H00         H0004  3807.818
## 6            H01         H0105   787.000
## 7            H01         H0106  1888.000
## 8            H01         H0107  1438.000
## 9            H01         H0108  1684.000
## 10           H02         H0209  1133.513
```

Total sale made based on the second level of hierarchy (hierarchy2_id). For instance:

- In a week, the total sale produced by selling products where the first level of hierarchy is H00 and the second level of hierarchy is H0000, was 13093.000.
- Meanwhile, the total sale produced by selling products where the first hierarchy level is H00 and the second hierarchy level is H0001, was 2481.000.

Insight:

Re-ordering dataframe `sale_hier2` to see which the maximum sales of each type of product, going from the highest sales to lowest sales of each type.

```r
sale_hier2[order(sale_hier2$hierarchy1_id, - sale_hier2$sales),]
```

```
##    hierarchy1_id hierarchy2_id     sales
## 4            H00         H0003 17920.000
## 1            H00         H0000 13093.000
## 5            H00         H0004  3807.818
## 3            H00         H0002  2955.000
## 2            H00         H0001  2481.000
## 7            H01         H0106  1888.000
## 9            H01         H0108  1684.000
## 8            H01         H0107  1438.000
## 6            H01         H0105   787.000
## 10           H02         H0209  1133.513
```

The most popular subtype of H00 sold in all stores is H0003 with a total sale of 17,920 made over the seven days. And the second most popular subtype of H00 sold in all stores is H0000 with a total sale of 13,093 made over the seven days.

8) Revenue generated from each product type:

As same as for calculating sales quantity, we use `aggregate()` with `sum` as a function to calculate the revenue generated from each product type.

```r
#hierarchy1_id
aggregate(revenue ~ hierarchy1_id,
          data = merged_sale_product_tab,
          sum)
```

```
##   hierarchy1_id   revenue
## 1           H00 100165.44
## 2           H01  61773.15
## 3           H02  12221.22
## 4           H03  25377.66
```

The total revenue obtained by each product type over the seven day period shows that:

- The top ranked product type is H00,which has obtained a total revenue of $100,165.44 over seven days.
- Meanwhile, the second-ranked product type is H01,which has obtained a total revenue of $61,773.15.
- And, the last ranked product type is H02,which has obtained a total revenue of $12,221.22.

```r
#hierarchy2_id:
aggregate(revenue ~ hierarchy1_id + hierarchy2_id,
          data = merged_sale_product_tab,
          sum) %>%
  head(10)
```

```
##    hierarchy1_id hierarchy2_id  revenue
## 1            H00         H0000 35413.54
## 2            H00         H0001  9207.45
## 3            H00         H0002 11134.93
## 4            H00         H0003 24249.76
## 5            H00         H0004 20159.76
## 6            H01         H0105  7698.96
## 7            H01         H0106 21503.25
## 8            H01         H0107 16386.22
## 9            H01         H0108 16184.72
## 10           H02         H0209 12180.40
```

Total revenue made based on the second level of hierarchy (hierarchy2_id).

- The most sold item in H00 is H0000 with a total of $35,413.54 made over the week.
- And the least sold item in H00 is H0001, with a total of $9,207.45 made over the week.

3. Compare the sales volumes between the two most common store types in the data set. How do they compare in terms of total revenue? Is there a relationship between a store's size and its revenue?

View information about the dataset store (store_cities data)

```r
#Viewing the first 10 values of the dataset
head(store, 10)
```

```
##    store_id storetype_id store_size city_id
## 1     S0091         ST04         19    C013
## 2     S0012         ST04         28    C005
## 3     S0045         ST04         17    C008
## 4     S0032         ST03         14    C019
## 5     S0027         ST04         24    C022
## 6     S0088         ST04         20    C009
## 7     S0095         ST02         44    C014
## 8     S0055         ST04         24    C014
## 9     S0099         ST03         14    C014
## 10    S0078         ST04         19    C036
```

```r
#structure of the dataset
str(store)
```

```
## 'data.frame':    144 obs. of  4 variables:
##  $ store_id    : chr  "S0091" "S0012" "S0045" "S0032" ...
##  $ storetype_id: chr  "ST04" "ST04" "ST04" "ST03" ...
##  $ store_size  : int  19 28 17 14 24 20 44 24 14 19 ...
##  $ city_id     : chr  "C013" "C005" "C008" "C019" ...
```

Compare the Sales volumes between the two most common store types in the data set.

Sorting store types accross the stores cities data set:

```r
sort(table(store$storetype_id), decreasing = TRUE)
```

```
## 
## ST04 ST03 ST01 ST02 
##   83   53    4    4
```

Ranking from most to least, there are:

- ST04 is the most common storetype with over 83 stores accross cities.
- ST02 and ST01 are the least common storetypes accross cities, with only 4 stores for each.

Joining two datasets a and d together

```r
#right join dataset d and a according to the corresponding id key:
merged_store_sale_tab <- store %>%
  select("store_id", "storetype_id", "store_size") %>%
  right_join(sale)
```

```
## Joining with ‘by = join_by(store_id)‘
```

```r
head(merged_store_sale_tab, 10)
```

```
##    store_id storetype_id store_size product_id       date sales revenue stock
## 1     S0091         ST04         19      P0015 2017-07-03     0       0     6
## 2     S0091         ST04         19      P0017 2017-07-03     0       0    20
## 3     S0091         ST04         19      P0035 2017-07-03     0       0     3
## 4     S0091         ST04         19      P0042 2017-07-03     0       0     5
## 5     S0091         ST04         19      P0046 2017-07-03     0       0     7
## 6     S0091         ST04         19      P0051 2017-07-03     0       0    22
## 7     S0091         ST04         19      P0054 2017-07-03     0       0     6
## 8     S0091         ST04         19      P0055 2017-07-03     0       0    12
## 9     S0091         ST04         19      P0057 2017-07-03     0       0     6
## 10    S0091         ST04         19      P0067 2017-07-03     0       0     4
##    price promo_type_1 promo_bin_1 promo_discount_2 promo_discount_type_2
## 1   2.85         PR14                           NA                    NA
## 2   1.49         PR12    veryhigh               NA                    NA
## 3   4.25         PR14                           NA                    NA
## 4   5.50         PR14                           NA                    NA
## 5  34.50         PR14                           NA                    NA
## 6   0.70         PR14                           NA                    NA
## 7   3.95         PR14                           NA                    NA
## 8   3.50         PR14                           NA                    NA
## 9  14.90         PR14                           NA                    NA
## 10 16.90         PR14                           NA                    NA
```

Calculating Sales volume using `aggregate()` with `sum` as an additional function.

```r
#sales volume of ST03 and ST04
sale_ST <- aggregate(sales ~ storetype_id,
                     data = merged_store_sale_tab,
                     sum)[c(3,4),] #[c(3,4),] is to display only the values of sales
sale_ST
```

```
##   storetype_id     sales
## 3         ST03  7980.007
## 4         ST04 35566.554
```

In terms of sales, Stores with `Storetype_id` ST03 has gained a total of 7980 in sale volume while stores with the store_id ST04 has gained a total of 35,556 in sale volume over the seven days. This means that stores with the storetype_id ST04 is more potential than the other, since the difference in the volume of sale made over a week is at least 4.4569577 times approximately over the other.

How do they compare in terms of total revenue?

```r
#Total revenue of ST03 and ST04
rev_ST <- aggregate(revenue ~ storetype_id,
          data = merged_store_sale_tab,
          sum)[c(3,4),] #shows only the values of revenue from ST03 and ST04
rev_ST
```

```
##   storetype_id   revenue
## 3         ST03  21776.75
## 4         ST04 144628.73
```

In terms of revenue achieved over the seven days period, Stores with `Storetype_id` as ST03 has gained a total of $21,776 while stores with `storetype_id` ST04 gained a total of $144,628. This means stores that is ST04 has made a total revenue that is at least 6.6414286 times approximately over the ST03 stores' total revenue.

Is there a relationship betwen a store's size and its revenue?

We will perform a hypothesis test on correlation to see if there is a relationship between a store's size and its revenue. Let the hypothesis be:

- H0: p = 0
- Ha: p != 0

```r
rev_rel <- aggregate(revenue ~ store_id + store_size, data = merged_store_sale_tab, sum)
nrow(rev_rel) #nrow of observations
```

```
## [1] 128
```

```r
cor.test(rev_rel$store_size,rev_rel$revenue) #perform pearson correlation testing
```

```
##
##  Pearson's product-moment correlation
##
## data:  rev_rel$store_size and rev_rel$revenue
## t = 11.043, df = 126, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.6008880 0.7799116
## sample estimates:
##      cor
## 0.701293
```

As stated in the correlation test above, since:

- There seems to be a moderate positive correlation between `store_size` and `revenue` as the correlation coefficient `cor` is 0.701293.
- 95% CI between 0.60 to 0.77 for correlation coefficient.

- the number of observations is large enough, with 128 rows.
- p-value is smaller than 0.05(default significance level).

We reject the null hypothesis. In conclusion, there is sufficient evidence to conclude that there is a significant linear relationship between `store_size` and `revenue`.

Lets see would a linear regression line be able to fit in the graph.

Hypothesi:

- H0: B = 0. There is no sufficient evidence of a linear relationship between `store_size` and `revenue`.
- Ha: B != 0. There is sufficient evidence of a linear relationship between `store_size` and `revenue`.

```
summary(lm(revenue~store_size, data = rev_rel))
```

```
##
## Call:
## lm(formula = revenue ~ store_size, data = rev_rel)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4955.0  -553.4  -217.1   272.1  6453.8
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -611.964    225.231  -2.717  0.00751 **
## store_size    89.635      8.117  11.043  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1244 on 126 degrees of freedom
## Multiple R-squared:  0.4918, Adjusted R-squared:  0.4878
## F-statistic: 121.9 on 1 and 126 DF,  p-value: < 2.2e-16
```

As described by the table, the explantory variable,`store_size` seems to have a statistically significant positive relationship with the response variable `revenue`, because:

- p-value of `store_size` is smaller than 0.05
- Standard error is small.

Overall,the linear regression model fits slightly well with the data, since:

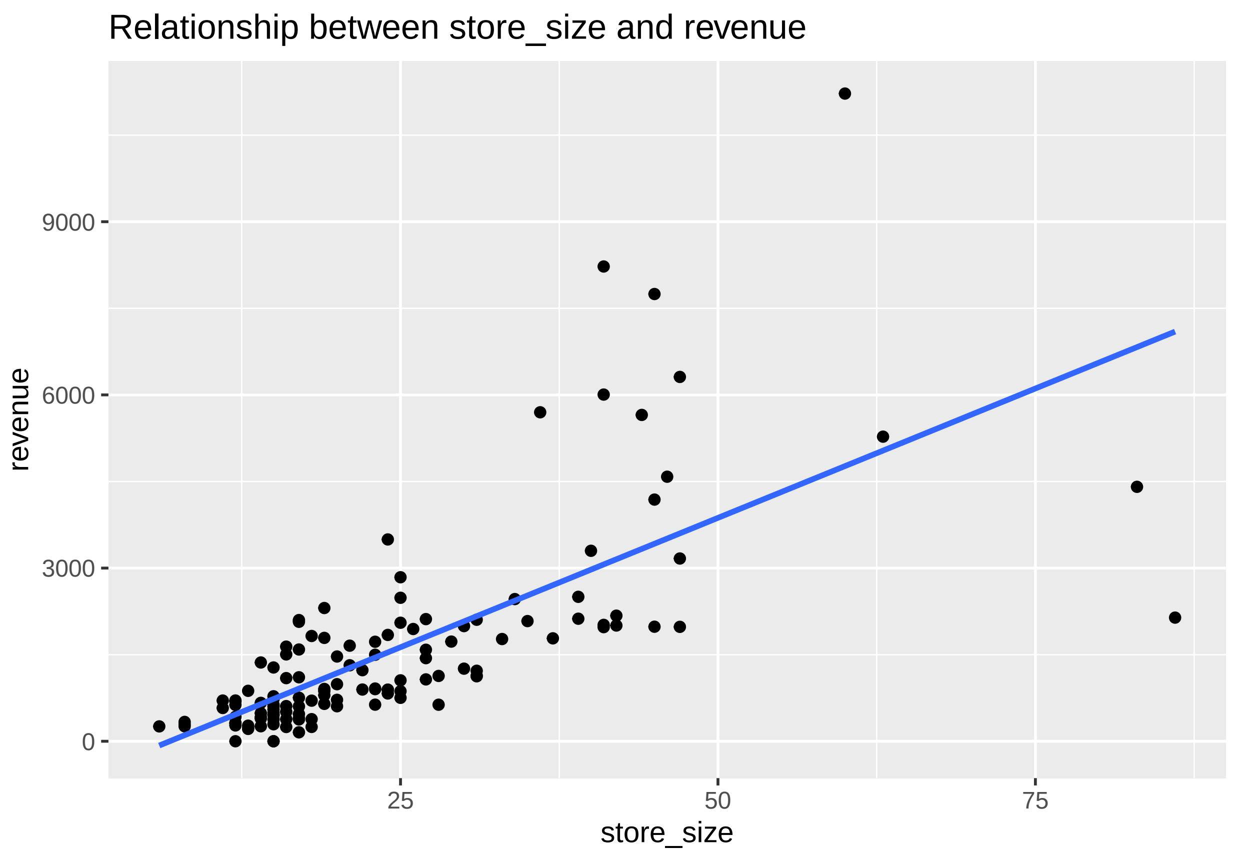
- R-squared refers to the 48% of the variance in the response variale `revenue`, is explained y the model, promoting moderate linear relationship.
- p-value is less than 0.05.
- Slope of the parameter is not equal to 0.
- RSE (residual standard error) is high, which explains why the scatterplot (in the next part) spread like a big fan-shaped.

Therefore, we reject the null hypothesis and conclude that there is a linear relationship between `store_size` and `revenue`.

Visualisation of the linear regression model on the graph of `store_size` and `revenue`.

```r
ggplot(data = rev_rel, aes(x = store_size, y = revenue)) +
  labs(title = "Relationship between store_size and revenue") +
  geom_point() +
  geom_smooth(method = "lm",
              se = FALSE) # se = FALSE removes the confidence interval lines
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



4. Several different types of promotions were applied to the products during the period with various level of promotion rates. For each promotion type, display the different levels of promotion used during the period. Analyse the effectiveness of the promotion on the sales of the products.

For each promotion type, display the different levels of promotion during the period

We will check promotion rate (promo_bin_1) by using `table()` function

```
#Different levels of promotion
table(sale$promo_type_1, sale$promo_bin_1)
```

```
##
##                 high   low moderate veryhigh verylow
##   PR03      0      0     0        0        0     286
##   PR05      0    123   744       14        0     240
##   PR06      0      0   175        0        0     481
##   PR08      0      0     0        0      126       0
##   PR09      0    190  1638        0        0       0
##   PR10      0      0     0        0        0      58
##   PR12      0      0     0        0     3196    1804
##   PR13      0      0     0        0        0      26
##   PR14 94899      0     0        0        0       0
```

Each promotion type has a unique level of ranking rate, from very high to very low. Except for promotion type PR14, it has a single promotion rate and is not categorised to any rate level like other promotion types.

To assess the effectiveness of using promotion, we will check on how many promotion used per day in stores across cities. The table below shows the amount and type of promotions that was used over seven days.

```
#Uses of promotion accross the seven day period
table(sale$date, sale$promo_type_1)
```

```
##
##                 PR03  PR05  PR06  PR08  PR09  PR10  PR12  PR13  PR14
##   2017-07-03      52   236    93     0   263     9   704     0 13422
##   2017-07-04      52    85    93     0   262     9   710     0 13616
##   2017-07-05      52    86    95     0   260     8   715     0 13605
##   2017-07-06      52   103    94     0   262     8   716     0 13652
##   2017-07-07      52   104    93     0   260     8   716     0 13668
##   2017-07-08      13   252    94    66   259     8   720    12 13476
##   2017-07-09      13   255    94    60   262     8   719    14 13460
```

However, as it can be seen, the most commonly used promotion across the seven days was PR14, with more over 13400 promotions were used on each day in every stores across cities.

If we dwell deeper into how many subtypes products are sold per day, we will have:

- These are the total products, subtype products sold from all stores on each day, from day 1 to day 7.

```
#shows products, their subtype products, and how many sold per day over 7 days.
table(sale$promo_type_1, sale$promo_bin_1, sale$date) %>%
  head(2) #shows 2 days instead of 7 days to minimise the display of data.
```

```
## , ,  = 2017-07-03
##
##
##             high low moderate veryhigh verylow
##   PR03    0    0   0        0        0      52
##   PR05    0  123  87        2        0      24
##
## , ,  = 2017-07-04
##
##
##             high low moderate veryhigh verylow
##   PR03    0    0   0        0        0      52
##   PR05    0    0  59        2        0      24
##
## , ,  = 2017-07-05
##
##
##             high low moderate veryhigh verylow
##   PR03    0    0   0        0        0      52
##   PR05    0    0  60        2        0      24
##
## , ,  = 2017-07-06
##
##
##             high low moderate veryhigh verylow
##   PR03    0    0   0        0        0      52
##   PR05    0    0  59        2        0      42
##
## , ,  = 2017-07-07
##
##
##             high low moderate veryhigh verylow
##   PR03    0    0   0        0        0      52
##   PR05    0    0  60        2        0      42
##
## , ,  = 2017-07-08
##
##
##             high low moderate veryhigh verylow
##   PR03    0    0   0        0        0      13
##   PR05    0    0 208        2        0      42
##
## , ,  = 2017-07-09
##
##
##             high low moderate veryhigh verylow
##   PR03    0    0   0        0        0      13
##   PR05    0    0 211        2        0      42
```

Analyse the effectiveness of the promotion on the sales of the products

To analyse the effectiveness of the promotion on the sales of products, We will use aggregate to see how much sales were made on each type of promotion, along with the revenue obtained, over the seven-day period. In the example below, I use `cbind` to bind columns sales and revenue from the dataset `a`, then I use `list` (as required to group dataframe by column variables) to aggregate `sales` and`revenue` by `promo_type_1` and `date`, with function `sum`.

```
x1 <- aggregate(cbind(sale$sales, sale$revenue),
                by = list(sale$promo_type_1, sale$date), #aggregated by these variables
                sum) #returns sales and revenue
colnames(x1) <- c("promo_type_1", "date", "sales", "revenue")
head(x1[order(x1$sales, decreasing = TRUE),], 10)
```

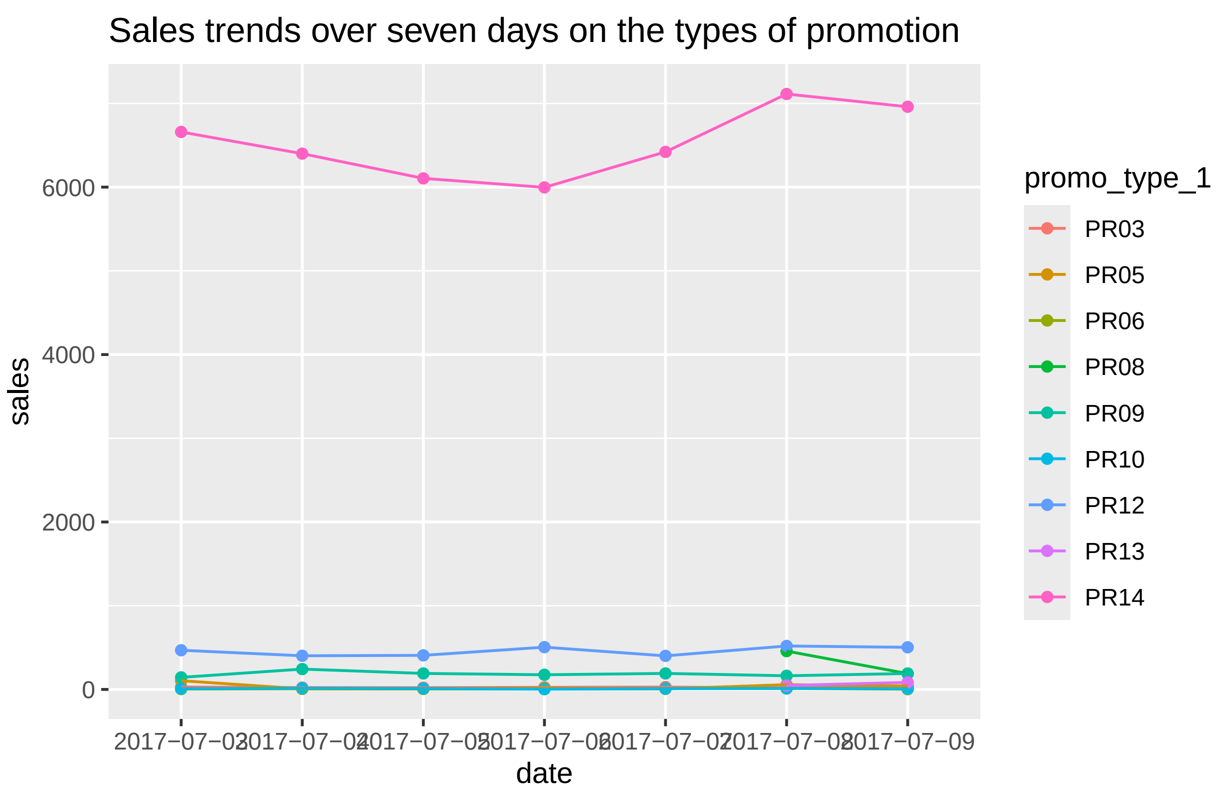
```
##    promo_type_1       date    sales  revenue
## 44         PR14 2017-07-08 7112.630 25845.64
## 53         PR14 2017-07-09 6960.143 26321.40
## 7          PR14 2017-07-03 6659.139 25170.12
## 35         PR14 2017-07-07 6421.084 23237.66
## 14         PR14 2017-07-04 6399.828 23925.97
## 21         PR14 2017-07-05 6104.809 22337.32
## 28         PR14 2017-07-06 5997.168 23066.07
## 42         PR12 2017-07-08  519.000  1086.82
## 27         PR12 2017-07-06  505.000  1191.49
## 51         PR12 2017-07-09  503.000  1261.38
```

As we can see from the ordered dataframe `x1`, we notice that the type of promotion have a significant effect on the sales of items within stores. For examples:

- The promotion type PR14 achieved the most sales among others (sales = 25845), and which also produced the highest revenues.
- While the promotion type PR06 only achieved the most sales of 3, which also produced the lowest revenues among others.

However, the ability to obtain sufficient amount of sales also varies depending on the date which the promotions were being promoted, meaning the shops might get a different amount of sales everyday in the seven days. To visualise the table of sales of each promotion type we will plot the sales trends of each type of promotion over seven day period, by using `ggplot`.

```
pl1 <- ggplot(data = x1,
              aes(x = date, y = sales, color = promo_type_1)) +
       geom_point() +
       geom_line(group = x1$promo_type_1) +
       coord_trans()
pl1 +
  labs(title = "Sales trends over seven days on the types of promotion",
       caption = "*Note: scaling is not efficient, so subgraphs of sales trends
                will be provided to reinforce the visualisation on trends data")
```

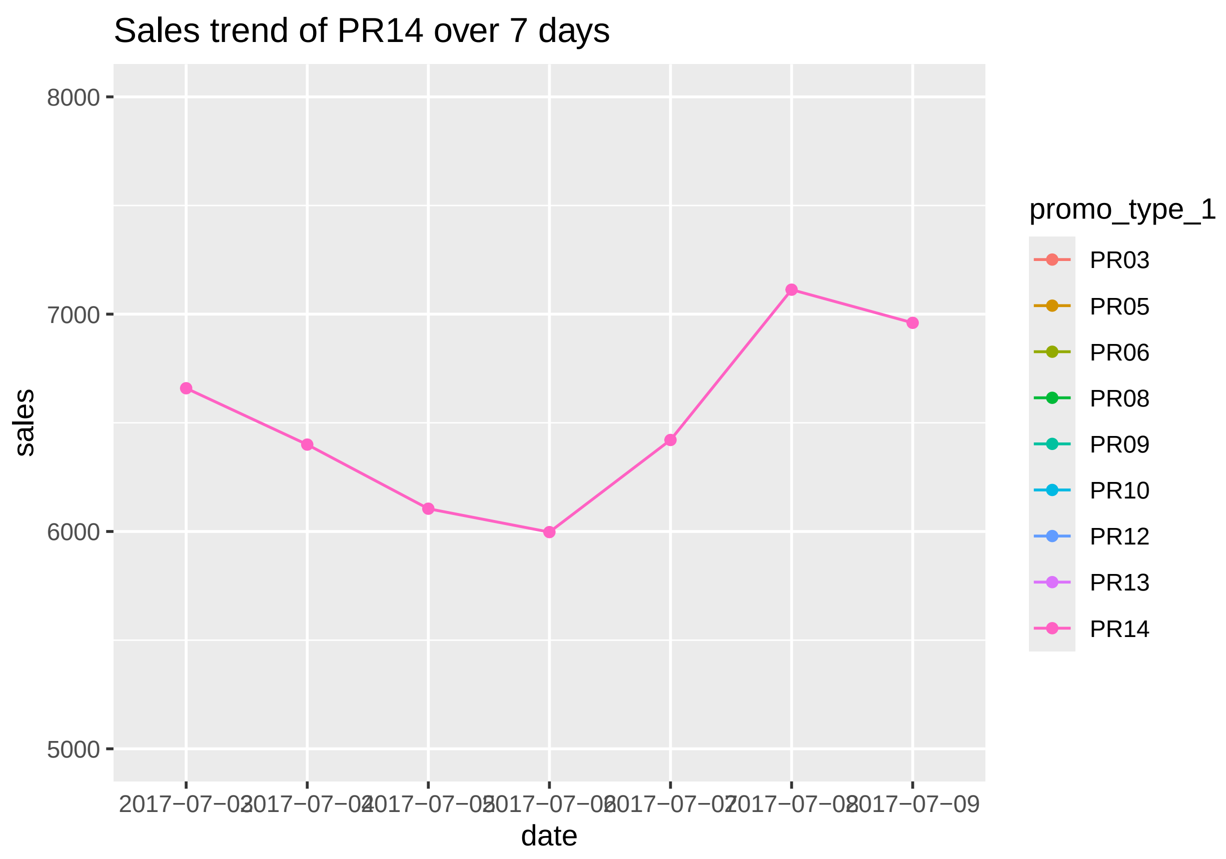


*Note: scaling is not efficient, so subgraphs of sales trends
will be provided to reinforce the visualisation on trends data

Subgraphs of pl1

```
pl1 +
  coord_cartesian(ylim = c(5000,8000)) +
  labs(title = "Sales trend of PR14 over 7 days")
```

```
## Coordinate system already present. Adding new coordinate system, which will
## replace the existing one.
```

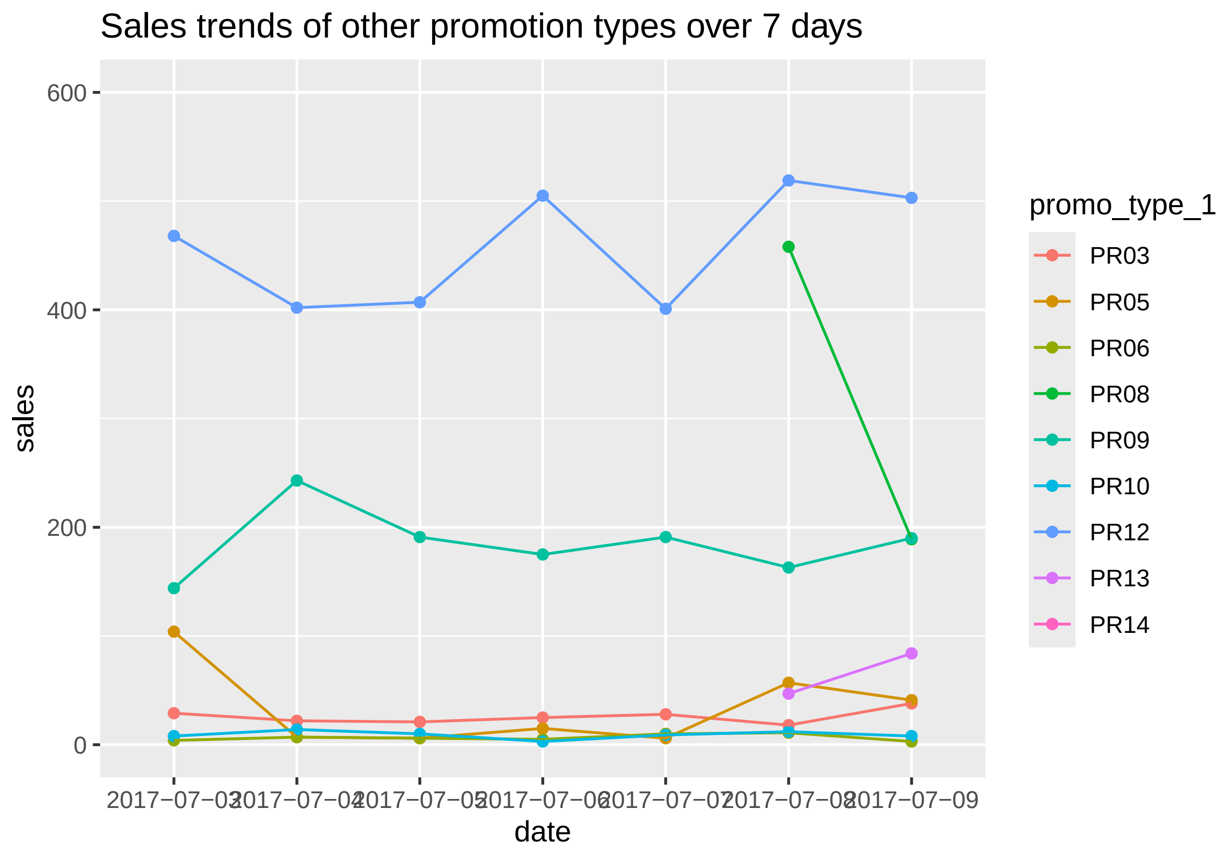


As mentioned earlier, the sales trend would vary depending on the date that the items with sales promotions were sold. In the Sales trend of PR14 above, the highest sales achieved was on the date 7/8/2017 and the lowest sales during the seven days was on 6/7/2017.

- The path that the sales trend followed is parabolic, it shows that most sales occurred on the weekend and lowest in midweek. Furthermore, the variation in sales between each day is significant.

```
pl1 +
  coord_cartesian(ylim = c(0,600)) +
  labs(title = "Sales trends of other promotion types over 7 days")
```

```
## Coordinate system already present. Adding new coordinate system, which will
## replace the existing one.
```



From the sales trends above, it could be noticed that some of these sales trends followed the linear trend throughout the whole week. For example, `PR03`,`PR08`,`PR06`. These promotion types did not vary much in terms of sales across the seven days. Moreover, Promotions such as `PR08` and `PR13` did not even achieve any sales since day 1 (3/7/2017) until day 6 (8/7/2017). Uniquely, promotion types like `PR05` and `PR12` followed the parabolic trend as `PR14`, despite there are not much variations within their sales quantity.

However, from the analysis above, we can assure that the uses of promotions can affect the sales of products. Especially, with products that were promoted with promotion type `PR14`, where the number of slaes throughout the week was higher than sales with other promo types.