COMP828 Assignment
Semester 1 2024

Ricky Yang
ID Number: 23205919

03 May 2024

Load & extract the data

Download the csv file

```r
library(tidyverse)
sales_all <- read_csv("COMP828_sales_data.csv")


sales <- sales_all %>%
   filter(sku_id==222087)

#clean data
sales <- sales %>%
  mutate(start_of_month = as.Date(start_of_month, format = "%m/%d/%Y"))  %>%
  mutate(end_of_month = as.Date(end_of_month, format = "%m/%d/%Y"))  %>%
  mutate(week = as.Date(week, format = "%m/%d/%Y"))

print(head(sales,5))
```

```
## # A tibble: 5 × 15
##   record_ID week       store_id sku_id total_price base_price is_featured_sku
##       <dbl> <date>        <dbl>  <dbl>       <dbl>      <dbl>           <dbl>
## 1        30 2011-01-17     8095 222087        174.       192.               1
## 2        55 2011-01-17     8094 222087        211.       211.               0
## 3        79 2011-01-17     8063 222087        217.       217.               0
## 4       106 2011-01-17     8023 222087        170.       220.               1
## 5       131 2011-01-17     8058 222087        170.       211.               1
## # ℹ 8 more variables: is_display_sku <dbl>, units_sold <dbl>, year <dbl>,
## #   month <dbl>, day <dbl>, weekday <chr>, start_of_month <date>,
## #   end_of_month <date>
```

Question 2. Explore the sales data

(a) Date range

```r
#What is the date range of the sales data? Provide R code and output required to determine this
#and write your answer in a sentence


sales_with_date <- sales %>%
  unite(dym,year,month,day,sep = '-') %>%
  mutate(dym = as.Date(dym))

 max_date <- format(max(sales_with_date$dym), "%Y-%m-%d")
 min_date <- format(min(sales_with_date$dym), "%Y-%m-%d")
```

Date range of the sales data: 2013-07-09 to 2011-01-17

(b) Number of stores

```r
#How many different stores sell your product (sku_id)? Provide R code and output required to
#determine this and write your answer in a sentence

num_stores <- unique(sales$store_id) %>% length()
cat(num_stores," different stores sell my product")
```

```
## 72  different stores sell my product
```

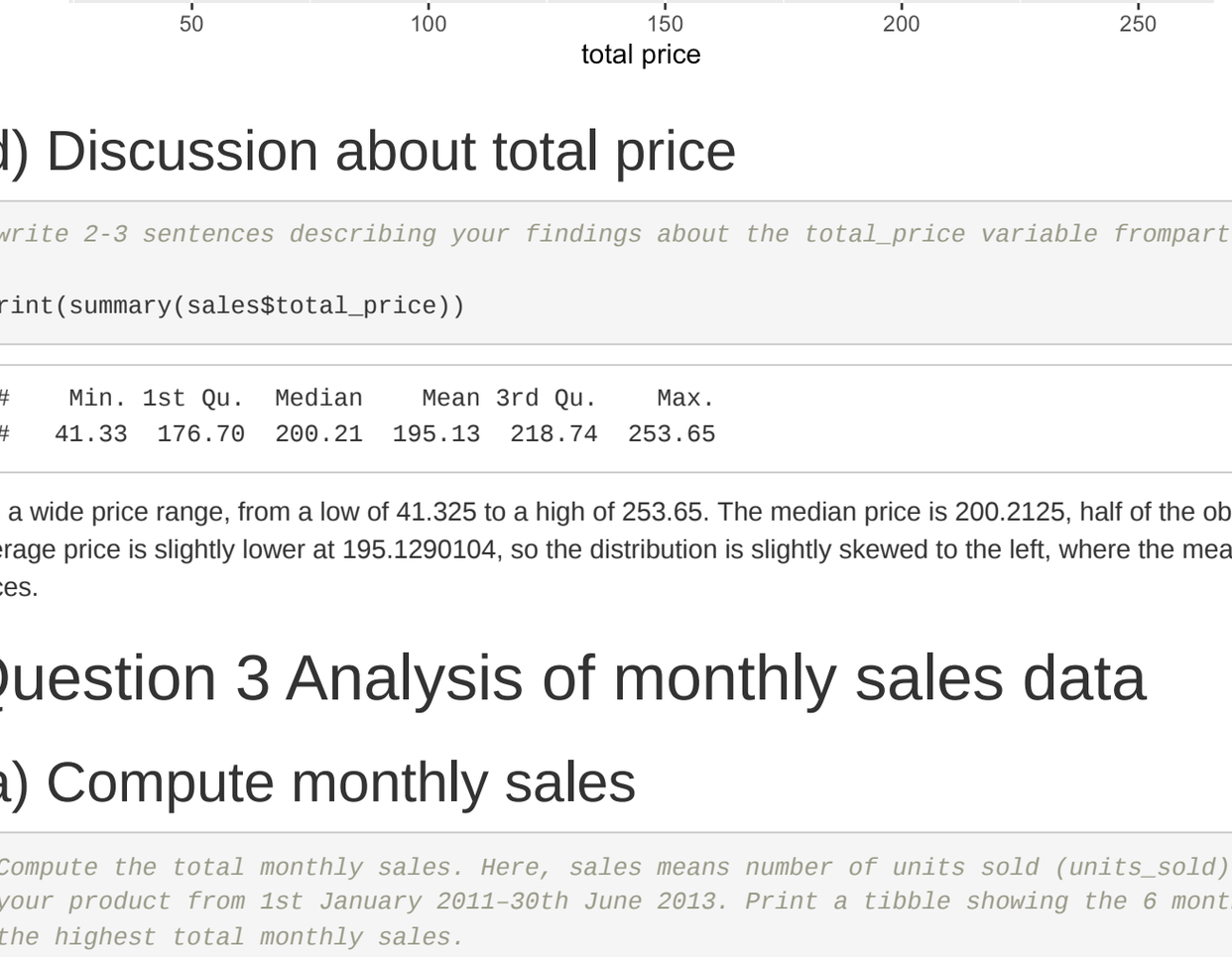
72 different stores sell my product

(c) Total price

```r
#compute some summary statistics and construct a histogram using the ggplot2 package to analyze
#the variable total_price for your product.

print(summary(sales$total_price))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   41.33  176.70  200.21  195.13  218.74  253.65
```

```r
sales %>% ggplot(aes(x=total_price)) +
  geom_histogram(binwidth = 10, fill = "green", color = "black") +
  labs(title ="total price analyze",x = "total price",y = "frequency")
```



(d) Discussion about total price

```r
#write 2-3 sentences describing your findings about the total_price variable frompart (c)

print(summary(sales$total_price))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   41.33  176.70  200.21  195.13  218.74  253.65
```

it is a wide price range, from a low of 41.325 to a high of 253.65. The median price is 200.2125, half of the observations are below this value. The average price is slightly lower at 195.1290104, so the distribution is slightly skewed to the left, where the mean is influenced by outliers with lower prices.

Question 3 Analysis of monthly sales data

(a) Compute monthly sales

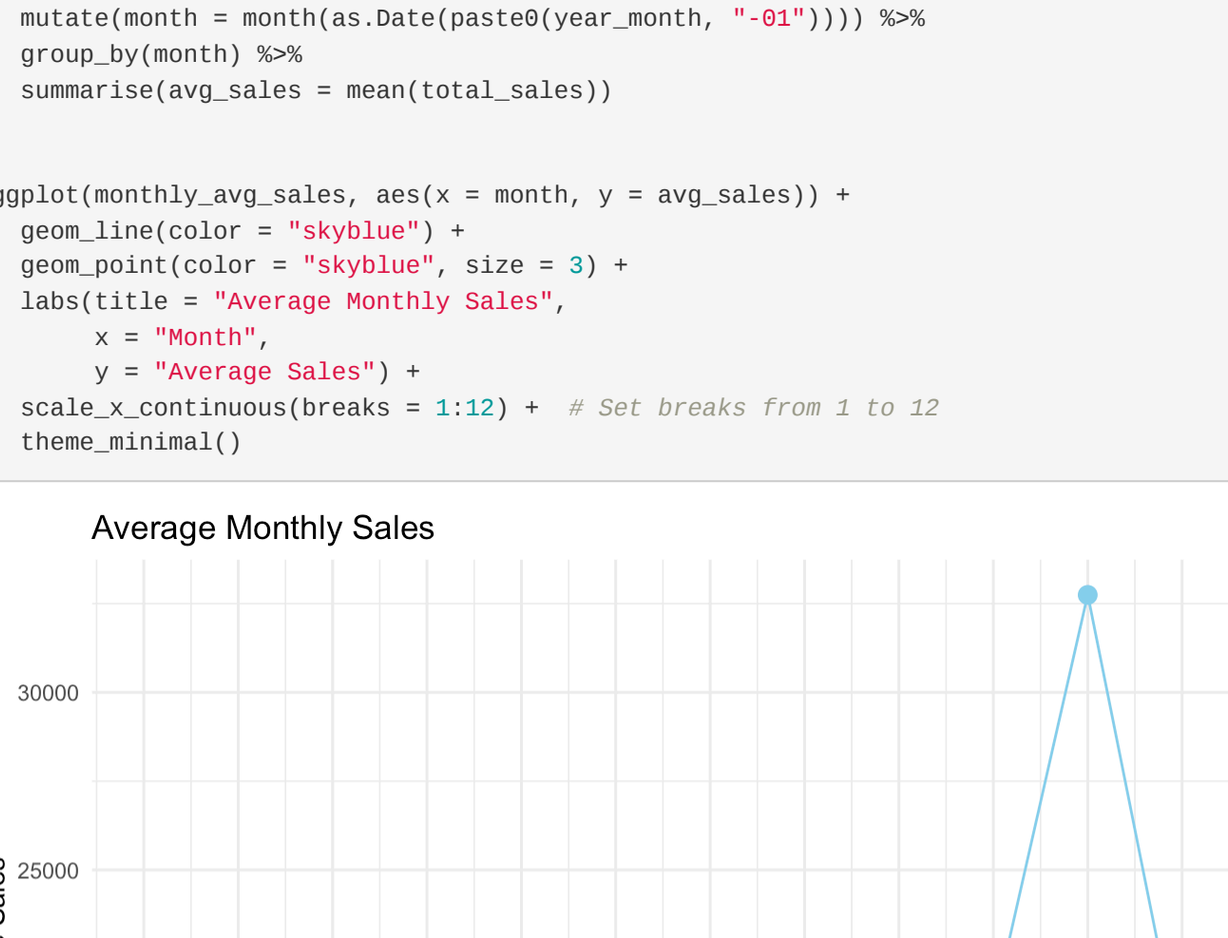
```r
#Compute the total monthly sales. Here, sales means number of units sold (units_sold). for
#your product from 1st January 2011–30th June 2013. Print a tibble showing the 6 months with
#the highest total monthly sales.


start_time <- as.Date("2011-01-01")
end_time <- as.Date("2013-06-30")

monthly_sales <- sales %>%
  unite(date_ymd,year,month,day,sep = '-',remove = FALSE) %>%
  mutate(date_ymd = as.Date(date_ymd)) %>%
  filter(date_ymd >= start_time & date_ymd <= end_time) %>%
  group_by(year_month = paste(year, month, sep = '-')) %>%
  summarise(total_sales = sum(units_sold))  %>%
  arrange(-total_sales)

  head(monthly_sales,6)
```

year_month <chr>	total_sales <dbl>
2012-11	40809
2013-4	29328
2011-8	28443
2011-5	28211
2013-1	26040
2011-10	25810

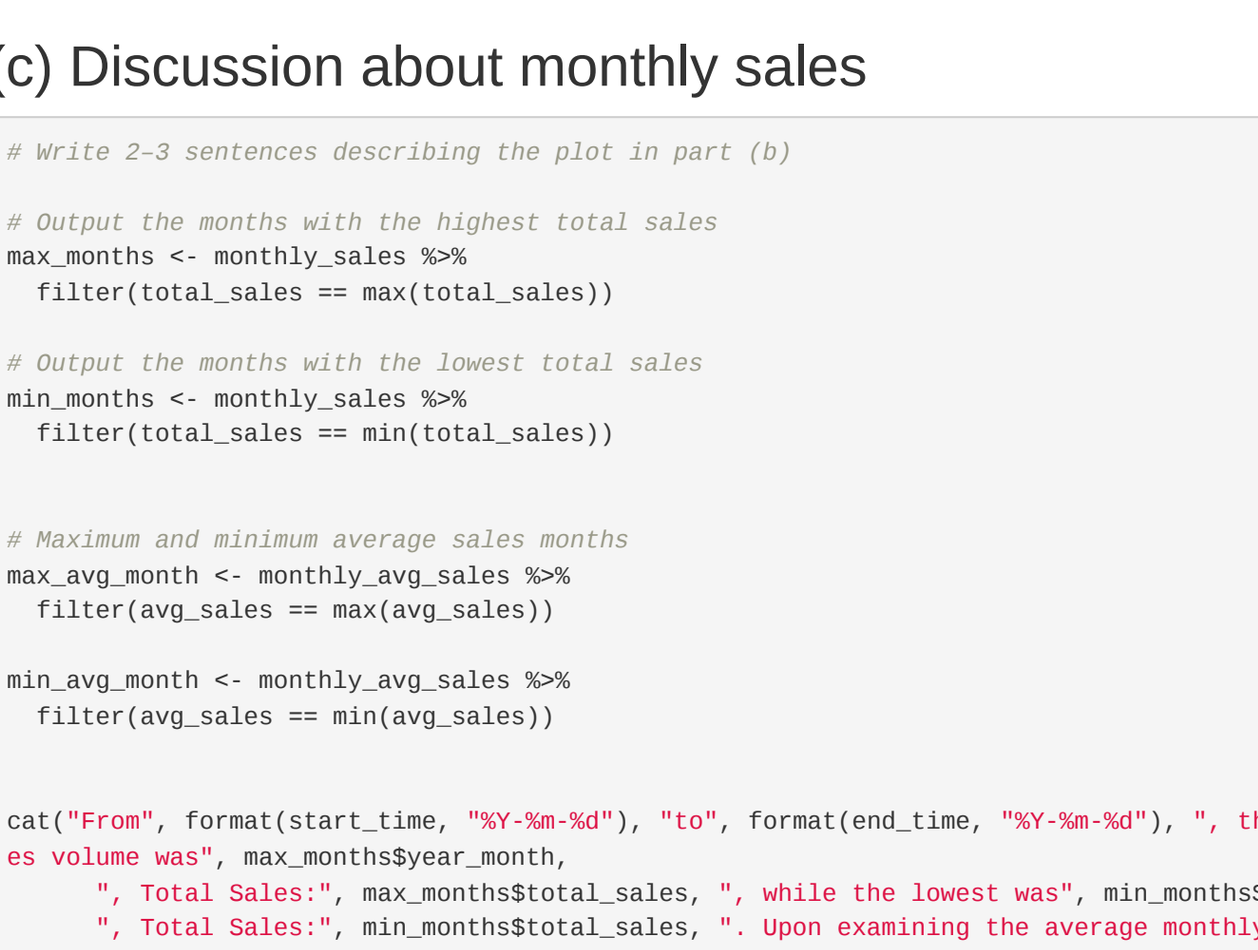
6 rows

(b) Plot of monthly sales

```r
#Use ggplot2 to present total monthly sales for your product in an appropriate plot. Ensure your
# graph has appropriate titles, labels, scales etc
ggplot(monthly_sales, aes(x = year_month, y = total_sales, group = 1)) +
  geom_line(color = "skyblue") +
  geom_point(color = "skyblue", size = 2) +
  labs(title = "Total Monthly Sales",
       x = "Year-Month",
       y = "Total Sales") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  scale_x_discrete(labels = function(x) format(as.Date(paste(x, "01", sep = "-")), "%Y-%m"))
```




```r
 # Calculate average monthly sales for each month (1 to 12)
monthly_avg_sales <- monthly_sales %>%
  mutate(month = month(as.Date(paste0(year_month, "-01")))) %>%
  group_by(month) %>%
  summarise(avg_sales = mean(total_sales))


ggplot(monthly_avg_sales, aes(x = month, y = avg_sales)) +
  geom_line(color = "skyblue") +
  geom_point(color = "skyblue", size = 3) +
  labs(title = "Average Monthly Sales",
       x = "Month",
       y = "Average Sales") +
  scale_x_continuous(breaks = 1:12) +  # Set breaks from 1 to 12
  theme_minimal()
```



(c) Discussion about monthly sales

```r
# Write 2–3 sentences describing the plot in part (b)

# Output the months with the highest total sales
max_months <- monthly_sales %>%
  filter(total_sales == max(total_sales))

# Output the months with the lowest total sales
min_months <- monthly_sales %>%
  filter(total_sales == min(total_sales))


# Maximum and minimum average sales months
max_avg_month <- monthly_avg_sales %>%
  filter(avg_sales == max(avg_sales))

min_avg_month <- monthly_avg_sales %>%
  filter(avg_sales == min(avg_sales))


cat("From", format(start_time, "%Y-%m-%d"), "to", format(end_time, "%Y-%m-%d"), ", the month with the highest sal
es volume was", max_months$year_month,
      ", Total Sales:", max_months$total_sales, ", while the lowest was", min_months$year_month,
      ", Total Sales:", min_months$total_sales, ". Upon examining the average monthly sales,",
      month.name[max_avg_month$month], "emerges as the month with the highest sales (", max_avg_month$avg_sales,
      "), whereas", month.name[min_avg_month$month], "records the lowest sales (", min_avg_month$avg_sales,
      "). Overall, there is considerable fluctuation in sales volume, indicating distinct periods of peak and off
-peak sales activity.")
```

```
## From 2011-01-01 to 2013-06-30 , the month with the highest sales volume was 2012-11 , Total Sales: 40809 , whi
le the lowest was 2012-9 , Total Sales: 12028 . Upon examining the average monthly sales, November emerges as the
month with the highest sales ( 32745 ), whereas September records the lowest sales ( 12916.5 ). Overall, there is
considerable fluctuation in sales volume, indicating distinct periods of peak and off-peak sales activity.
```

Question 4 Analysis of store performance

The GM Sales wants to know which stores are performing well, in terms of product sales. You should analyse the data for the product (sku_id) which has been assigned to you.

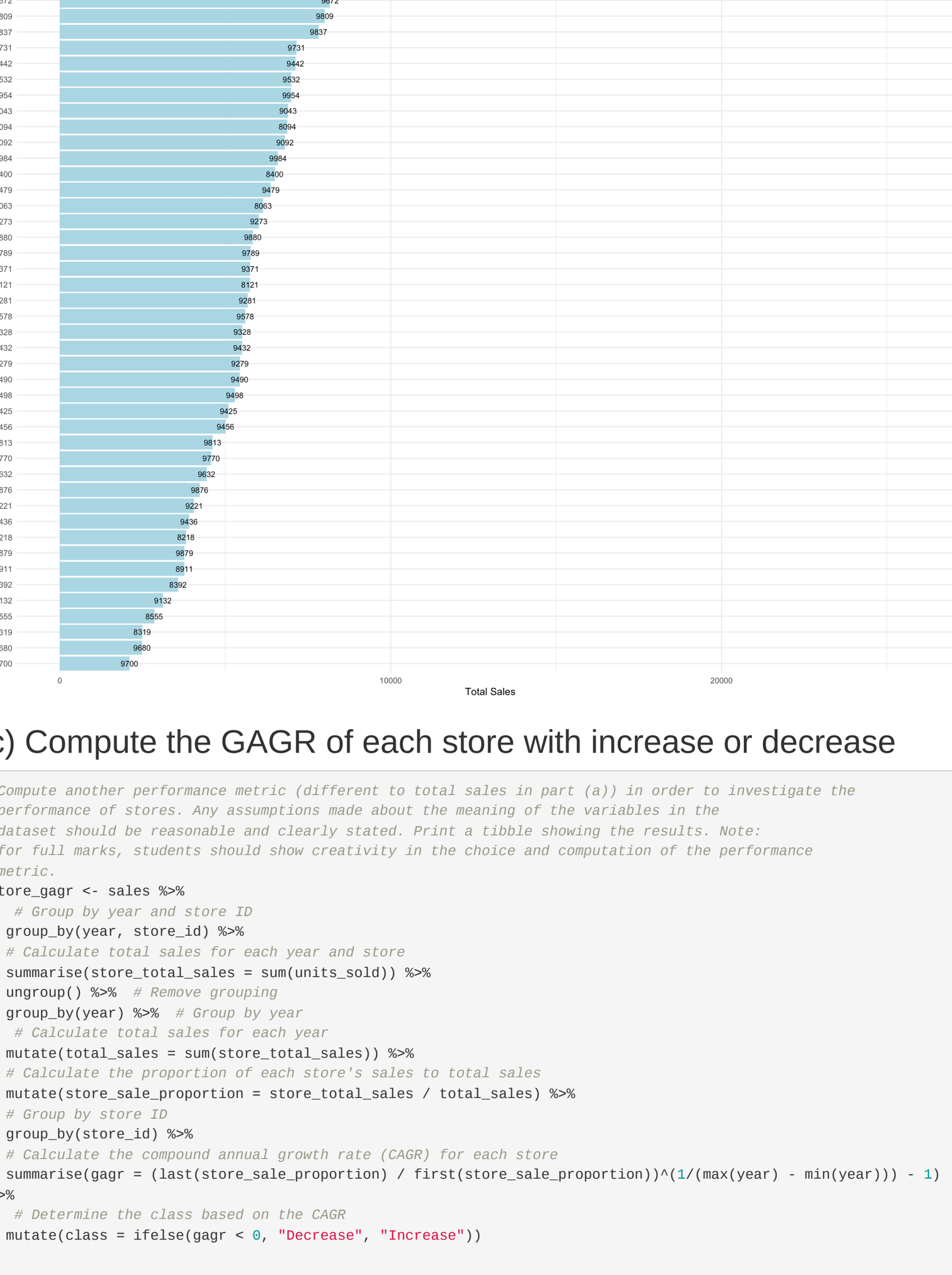
(a) Compute total sales per store

```r
#Use appropriate tidyverse functions to compute the total sales per store. Here, “sales” refers
#to the number of units sold (units_sold). Print a tibble showing total sales by store, sorted
#by total sales in decreasing order.

total_sales_by_store <- sales %>% group_by(store_id) %>%
  summarise(total_sales=sum(units_sold)) %>%
  arrange(-total_sales)


  print(total_sales_by_store)
```

```
## # A tibble: 72 × 2
##    store_id total_sales
##       <dbl>       <dbl>
##  1     8023       26020
##  2     9823       21655
##  3     9845       21539
##  4     9112       19272
##  5     8222       18169
##  6     8095       15161
##  7     9611       14363
##  8     9250       13899
##  9     9961       13757
## 10     9613       13680
## # ℹ 62 more rows
```

(b) Plot of total sales per store

```r
#Create an appropriate plot using ggplot2 to visualize the total sales per store from part (a). Hint:
#you may need to use a function like as_factor to ensure the store id is visualized correctly.


total_sales_by_store <- total_sales_by_store %>%
  arrange(total_sales)

total_sales_by_store %>%
  ggplot(aes(y = factor(store_id, levels = store_id), x = total_sales, label = store_id)) +
  geom_bar(stat = "identity", fill = "lightblue") +
  geom_text(aes(label = store_id), hjust = 0.5, vjust = 0.5, color = "black", size = 3) +
  labs(title = "Total Sales per Store",
       y = "Store ID",
       x = "Total Sales") +
  theme_minimal()
```



(c) Compute the GAGR of each store with increase or decrease

```r
#Compute another performance metric (different to total sales in part (a)) in order to investigate the
#performance of stores. Any assumptions made about the meaning of the variables in the
#dataset should be reasonable and clearly stated. Print a tibble showing the results. Note:
#for full marks, students should show creativity in the choice and computation of the performance
#metric.
store_gagr <- sales %>%
   # Group by year and store ID
  group_by(year, store_id) %>%
  # Calculate total sales for each year and store
  summarise(store_total_sales = sum(units_sold)) %>%
  ungroup() %>%  # Remove grouping
  group_by(year) %>%  # Group by year
   # Calculate total sales for each year
  mutate(total_sales = sum(store_total_sales)) %>%
  # Calculate the proportion of each store's sales to total sales
  mutate(store_sale_proportion = store_total_sales / total_sales) %>%
  # Group by store ID
  group_by(store_id) %>%
  # Calculate the compound annual growth rate (CAGR) for each store
  summarise(gagr = (last(store_sale_proportion) / first(store_sale_proportion))^(1/(max(year) - min(year))) - 1)
%>%
   # Determine the class based on the CAGR
  mutate(class = ifelse(gagr < 0, "Decrease", "Increase"))


# Merge the data , show the Total Sales and GAGR together.
merged_data <- merge(total_sales_by_store, store_gagr, by = "store_id") %>%
  # Arrange in descending order of CAGR
  arrange(-gagr)

merged_data
```

store_id <dbl>	total_sales <dbl>	gagr <dbl>	class <chr>
9770	4576	0.134943113	Increase
9221	4057	0.131149461	Increase
9490	5434	0.127751196	Increase
9879	3766	0.114109669	Increase
9613	13680	0.106207442	Increase
9147	8321	0.103528596	Increase
8869	11730	0.088321261	Increase
8095	15161	0.083694372	Increase
9845	21539	0.080056505	Increase
9043	6898	0.067246173	Increase

1-10 of 72 rows   Previous 1 2 3 4 5 6 ... 8 Next

(d) Plot of the GAGR of each store with color indicating increase or decrease

```r
#Create an appropriate plot using ggplot2 to visualize the performance metric in part (c).


ggplot(merged_data) +
  geom_bar(aes(x = factor(reorder(store_id, -gagr)), y = total_sales), fill = "lightblue", stat = "identity") +
  geom_text(aes(x = factor(reorder(store_id, -gagr)), y = total_sales, label = store_id), color = "black") +
  geom_point(aes(x = factor(reorder(store_id, -gagr)), y = gagr * max(total_sales*2), color = class), size = 3) +
  scale_y_continuous(sec.axis = sec_axis(~./max(merged_data$total_sales*2), name = "CAGR")) +
  labs(title = "Store Sales and Growth Analysis",
       x = NULL,
       y = "Total Sales") +
  theme_minimal() +
  theme(
        axis.text.x = element_text( hjust = 1,vjust=0.5),
        legend.position = "none")+
coord_flip()
```

(e)Discussion of store performance

```r
#Write 1-2 paragraphs for the GMSales discussing your findings from parts (a–d)
```

Due to variations in location and scale among stores, it was not accurate to simply compare sales volumes to assess store performance. Additionally, since the product sales experienced fluctuations, relying solely on month-over-month sales growth of stores was not sufficient. Instead, it was necessary to consider the growth of each store's sales proportion within the same market conditions. This meant comparing each store's performance to its own historical data to evaluate the growth of its sales performance.

With the utilization of Compound Annual Growth Rate (CAGR) (Gartner. n.d.) and total sales metrics, an insightful analysis revealed that certain stores exhibited commendable performance in terms of total sales, coupled with robust growth ratios, as exemplified by stores 9845, 8222, and 9112. Conversely, there were establishments demonstrating strong total sales figures yet experiencing negative growth trends, indicating areas for improvement, such as stores 8317. Additionally, 8023 maintained a stable performance and boasted the highest total sales.

Reference:

Gartner. (n.d.). Compound Annual Growth Rate (CAGR) [Online]. Available: https://www.gartner.com/en/information-technology/glossary/cagr-compound-annual-growth-rate. Accessed: April 30, 2024.

Appendix: R Environment

```r
format(Sys.time(), '%d %B %Y')
```

```
## [1] "03 May 2024"
```

```r
sessionInfo()
```

```
## R version 4.3.3 (2024-02-29)
## Platform: x86_64-apple-darwin20 (64-bit)
## Running under: macOS Sonoma 14.4.1
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.3-x86_64/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.3-x86_64/Resources/lib/libRlapack.dylib;  LAPACK version 3.
11.0
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## time zone: Pacific/Auckland
## tzcode source: internal
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] ggrepel_0.9.5   lubridate_1.9.3 forcats_1.0.0   stringr_1.5.1
##  [5] dplyr_1.1.4     purrr_1.0.2     readr_2.1.5     tidyr_1.3.1
##  [9] tibble_3.2.1    ggplot2_3.5.0   tidyverse_2.0.0 knitr_1.45
##
## loaded via a namespace (and not attached):
##  [1] sass_0.4.8        utf8_1.2.4        generics_0.1.3    stringi_1.8.3
##  [5] hms_1.1.3         digest_0.6.34     magrittr_2.0.3    evaluate_0.23
##  [9] grid_4.3.3        timechange_0.3.0  fastmap_1.1.1     jsonlite_1.8.8
## [13] fansi_1.0.6       scales_1.3.0      jquerylib_0.1.4   cli_3.6.2
## [17] rlang_1.1.3       crayon_1.5.2      bit64_4.0.5       munsell_0.5.0
## [21] withr_3.0.0       cachem_1.0.8      yaml_2.3.8        tools_4.3.3
## [25] parallel_4.3.3    tzdb_0.4.0        colorspace_2.1-0  vctrs_0.6.5
## [29] R6_2.5.1          lifecycle_1.0.4   bit_4.0.5         vroom_1.6.5
## [33] pkgconfig_2.0.3   pillar_1.9.0      bslib_0.6.1       gtable_0.3.4
## [37] glue_1.7.0        Rcpp_1.0.12       highr_0.10        xfun_0.42
## [41] tidyselect_1.2.1  rstudioapi_0.15.0 farver_2.1.1      htmltools_0.5.7
## [45] rmarkdown_2.26    labeling_0.4.3    compiler_4.3.3
```