Project Report

Contents

Team members

- Ahmed Ashraf Mohamed
  - ID: 2022446758
- Abdelrhman Mohamed Abdelhady Hodid
  - ID : 2022513643
- Antonuose Gerges Nageh
  - ID : 20221903971

Project Description

Getting and preparing the data set

importing the library we are going to use

```
library(dplyr)
library(ggplot2)
library(forcats)
library(arules)
library(hrbrthemes)
```

Reading the data and checking the frist 10 rows of it

```
dataPath <- readline("Enter the path to the data set : ")
grc <- as_tibble(read.csv(dataPath,stringsAsFactors = FALSE))
# displaying first 10 rows of our data
print(grc,n = 10, width = 80)
```

```
## # A tibble: 9,835 x 8
##    items                      count total   rnd customer   age city   paymentType
##    <chr>                      <int> <int> <int> <chr>    <int> <chr>  <chr>
##  1 citrus fruit,semi-finish~     4  1612     9 Maged       60 Hurgh~ Cash
##  2 tropical fruit,yogurt,co~     3   509    12 Eman        23 Aswan  Cash
##  3 whole milk                    1  2084     8 Rania       37 Dakah~ Cash
##  4 pip fruit,yogurt,cream c~     4   788     8 Rania       37 Dakah~ Cash
##  5 other vegetables,whole m~     4  1182    14 Magdy       36 Sohag  Cash
##  6 whole milk,butter,yogurt~     5  1771     3 Ahmed       30 Giza   Credit
##  7 rolls/buns                    1  2196     7 Huda        39 Gharb~ Cash
##  8 other vegetables,UHT-mil~     5  1657     6 Walaa       29 Cairo  Cash
##  9 pot plants                    1  2373     2 Mohamed     25 Alexa~ Credit
## 10 whole milk,cereals            2   343     5 Shimaa      55 Port ~ Cash
## # ... with 9,825 more rows
```

Preparing the data for k-means and apriori algorithms

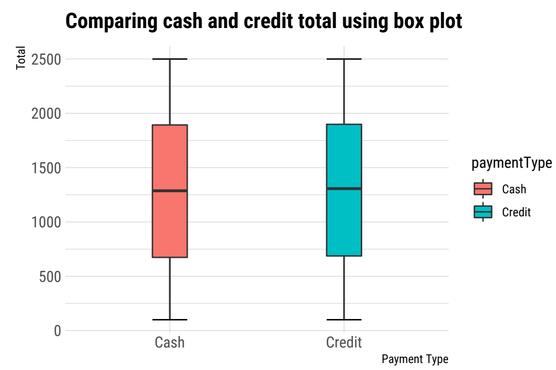
```
# creating a data framing containing customers, their ages and total spending
grc_customers <- grc %>%
  select(customer,age,total) %>%
  group_by(customer)%>%
  mutate(total = sum(total))%>%
  unique()
# making the data frame suitable for k-means
grc_k <- data.frame(grc_customers[,2:3],row.names = grc_customers$customer)
# splitting the items to be suitable for apriori algorithms
tdata <- strsplit(as.vector(grc$items), ',')
tdata <- transactions(tdata)
```

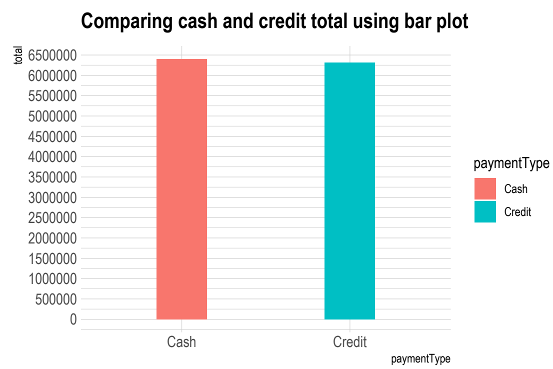
Visualizing our Data

Comparison between cash and creadit total spending using box plot

```
boxplot_cashCredit <- ggplot(
  grc,
  aes(x = paymentType, y = total, fill = paymentType)) +
  stat_boxplot(geom = "errorbar", width = .2) +
  geom_boxplot(width = .2,
               outlier.color = "orange",
               outlier.size = 2)+
  theme_ipsum_rc() +
  xlab("Payment Type") +
  ylab("Total") +
  theme(
    plot.title = element_text(size=16)) +
  ggtitle("Comparing cash and credit total using box plot")
```

```
barplot_cashCredit <- ggplot(grc,
                             aes(x = paymentType,
                                 y = total,
                                 fill = paymentType)) +
  geom_col(width = .3) +
  scale_y_continuous(n.breaks = 12) +
  theme_ipsum() +
  theme(
    plot.title = element_text(size = 16)) +
  ggtitle("Comparing cash and credit total using bar plot")

print(boxplot_cashCredit)
print(barplot_cashCredit)
```





Observations

After brief moments of seeing the figure, it is quite easy to see that people nearly equally pay with Cash or credit money.

Compare each age and sum of total spending.

Before visualizing

let's create a contingency table using the data prepared and look at a table containing customers, their ages and their total individual spending

```
table(grc_customers$age)
```

```
##
## 22 23 25 29 30 35 36 37 39 50 55 60
##  2  1  1  1  1  1  1  2  1  1  2  1
```

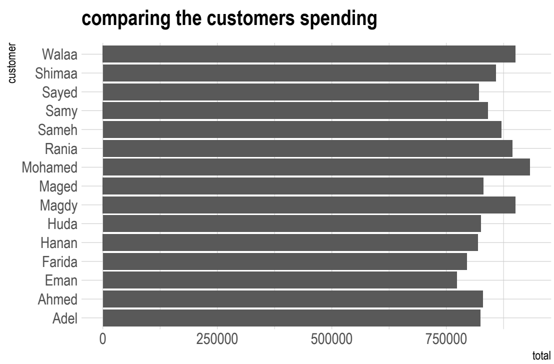
```
arrange(grc_customers,desc(total))
```

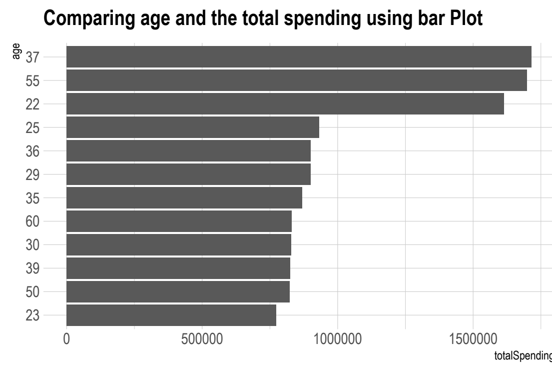
```
## # A tibble: 15 x 3
## # Groups:   customer [15]
##    customer   age  total
##    <chr>     <int>  <int>
##  1 Mohamed      25 932250
##  2 Magdy        36 901010
##  3 Walaa        29 900797
```

```
## 4 Rania 37 893789
## 5 Sameh 35 869668
## 6 Shimaa 55 857901
## 7 Samy 55 841167
## 8 Maged 60 831272
## 9 Ahmed 30 829587
## 10 Huda 39 825147
## 11 Adel 50 824064
## 12 Sayed 37 820900
## 13 Hanan 22 819231
## 14 Farida 22 794570
## 15 Eman 23 772871
```

It becomes clear that there are more people aged (22,37 and 55) than the other age groups

```
customersBarPlot<- grc_customers %>%
  ggplot(
    aes(x = customer, y = total)
  ) +
  geom_col() +
  coord_flip()+
  theme_ipsum() +
  theme(
    plot.title = element_text(size=16))+
  ggtitle("comparing the customers spending")
grc_age <- select(grc,age,total)
grc_age <- grc_age %>%
  group_by(age) %>%
  summarise(totalSpending = sum(total))
grc_age <- mutate(grc_age,age = fct_reorder(as.factor(age),totalSpending))
barPlotAgeSum<-ggplot(
  grc_age,
  aes(x = age, y = totalSpending)) +
  geom_col() +
  coord_flip() +
  theme_ipsum() +
  theme(
    plot.title = element_text(size=16),
    legend.position = "none")+
  ggtitle("Comparing age and the total spending using bar Plot")
customersBarPlot
barPlotAgeSum
```





Observations

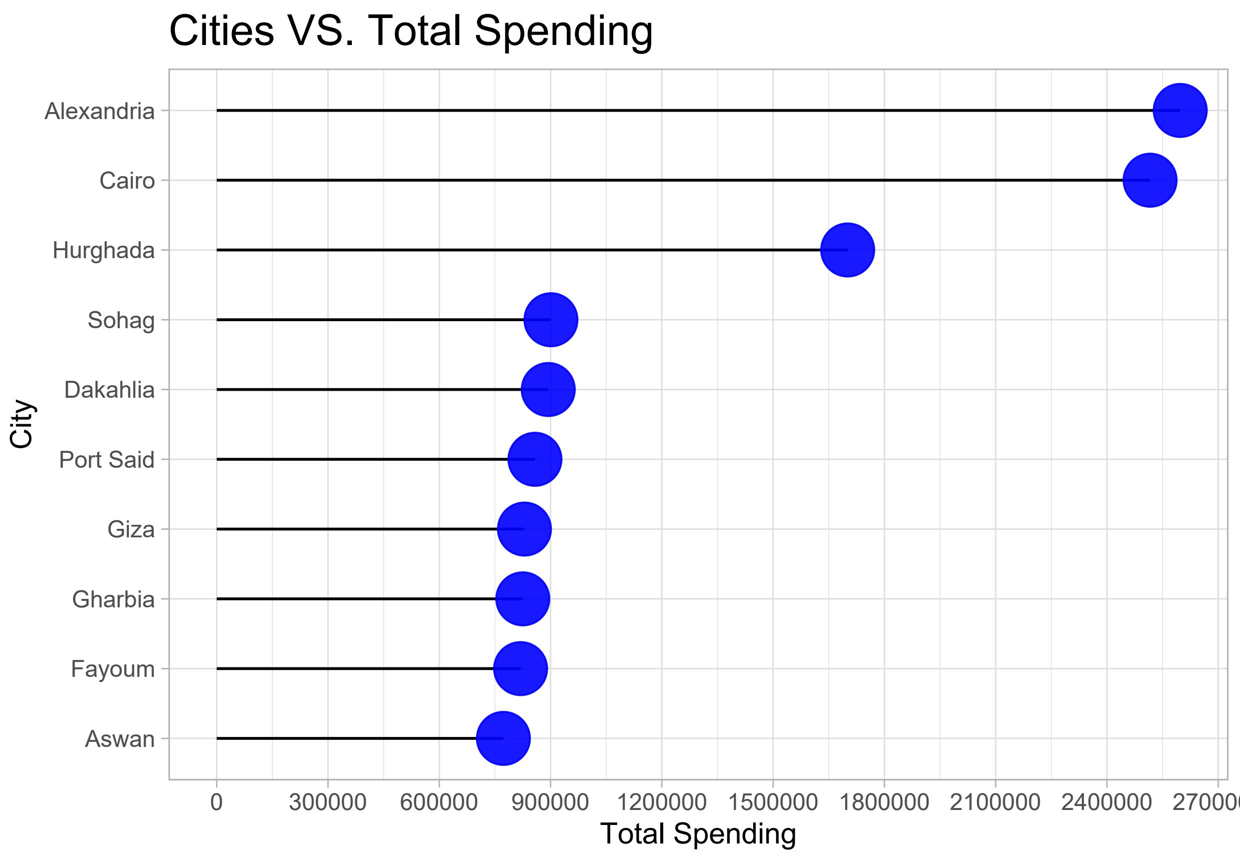
Since there are more customers aged (22, 37 and 55), it makes sense that the total spending of these groups is higher than the rest. But, looking at the data again we can see that if we look at the customers and their individual spending, we find that customers aged between 20 and 30 -even though there are single individual of these groups- are the highest spending.

```
###Cleaning the data in order to be prepared for data visualization
C_Vs_To <-grc %>%
  select(city,total) %>%
  group_by(city) %>%
  summarise(totalspending = sum(total))
C_Vs_To <- mutate(C_Vs_To, city = fct_reorder(as.factor(city),totalspending))
print(C_Vs_To)
```

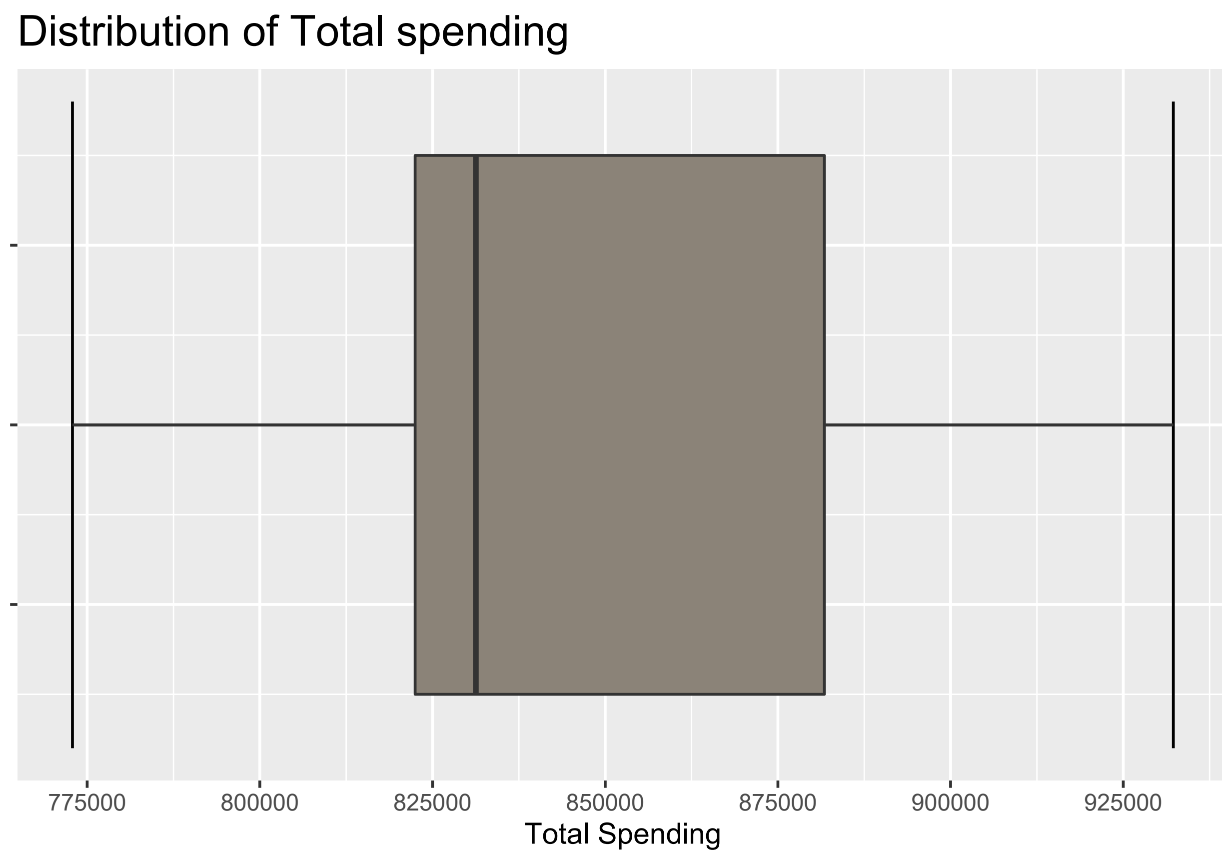
```
## # A tibble: 10 x 2
##    city       totalspending
##    <fct>              <int>
##  1 Alexandria       2597481
##  2 Aswan             772871
##  3 Cairo            2516267
##  4 Dakahlia          893789
##  5 Fayoum            819231
##  6 Gharbia           825147
##  7 Giza              829587
##  8 Hurghada         1700940
##  9 Port Said         857901
## 10 Sohag             901010
```

```
###Visualizing

CityandTotalspending<- ggplot(C_Vs_To,aes(city,totalspending)) +
  geom_segment( aes(xend=city,y = 0,yend=totalspending)) +
  scale_y_continuous(n.breaks = 10) +
  geom_point( color="blue", size=9, alpha=.9) +
  theme_light() +
  xlab("City") +
  ylab("Total Spending") +
  coord_flip() +
  theme(
    plot.title = element_text(size=16))+
```

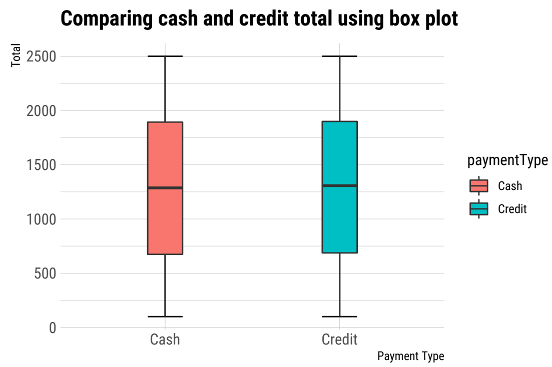
```
  ggtitle("Cities VS. Total Spending")

print(CityandTotalspending)
```

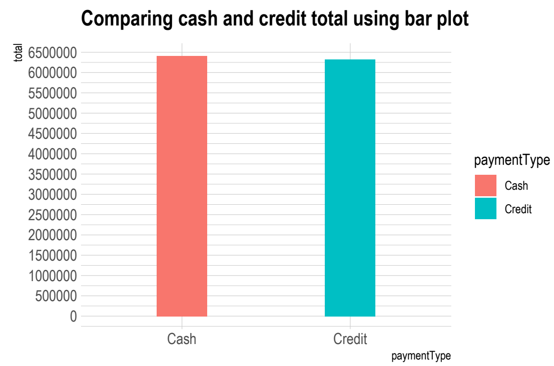


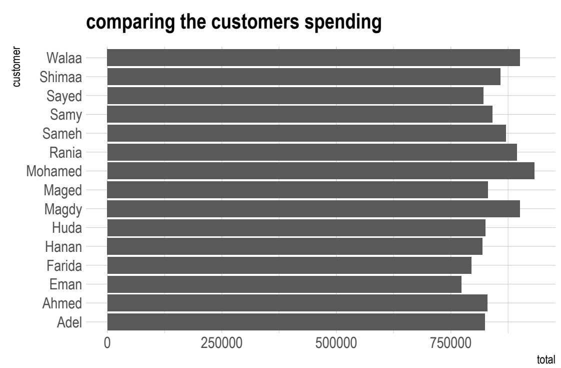

```
## Display the distribution of total spending.
Distribution_of_total_spending<-ggplot(grc_customers,aes(total)) +
  stat_boxplot(geom = "errorbar",width = .9) +
  geom_boxplot(fill = "antiquewhite4",) +
  scale_x_continuous(n.breaks = 8) +
  theme_grey() +
  theme(
    plot.title = element_text(size=16),
    axis.text.y = element_blank()) +
  xlab("Total Spending")  +
  ggtitle("Distribution of Total spending")
print(Distribution_of_total_spending)
```

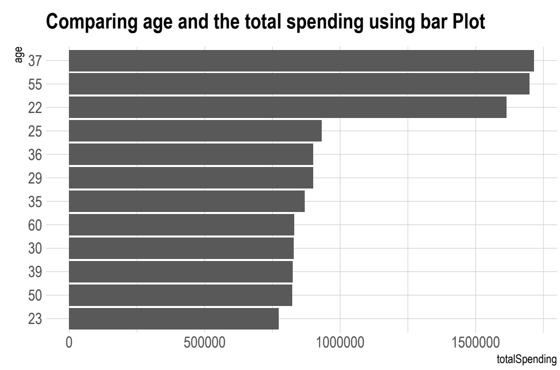

Distribution of Total spending
775000
800000
825000
850000
875000
900000
925000
Total Spending

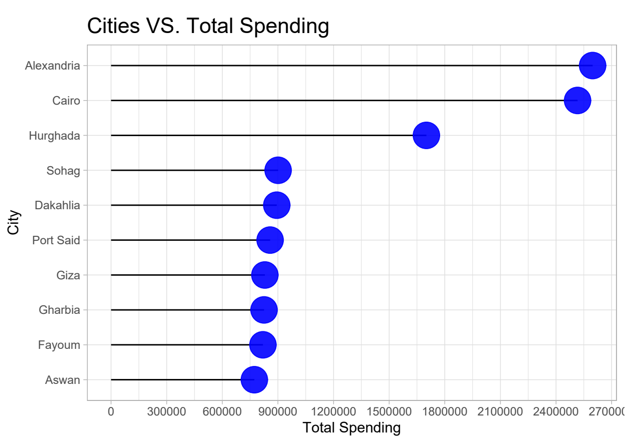
Dashboard

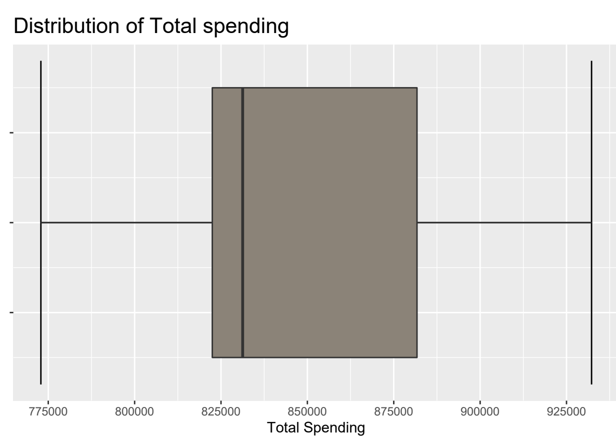












K-means

```
No_of_clusters<-as.numeric(readline("Enter the number of clusters: "))
Kmeans_Algorithm<-kmeans(grc_kmeans,centers = No_of_clusters)
grc_kmeans<-mutate(grc_kmeans,cluster=Kmeans_Algorithm$cluster)
print(grc_kmeans)
```

```
##          age  total cluster
## Maged     60 831272       2
## Eman      23 772871       2
```

```
## Rania    37 893789       1
## Magdy    36 901010       1
## Ahmed    30 829587       2
## Huda     39 825147       2
## Walaa    29 900797       1
## Mohamed  25 932250       1
## Shimaa   55 857901       1
## Farida   22 794570       2
## Hanan    22 819231       2
## Sayed    37 820900       2
## Adel     50 824064       2
## Sameh    35 869668       1
## Samy     55 841167       2
```

Generating of association rules

Brief explanition of Apriori algorithm for generating the rules

Apriori algorithm is an iterative approach for discovering the most frequent item sets. The frequent item sets generated by the algorithm can be used to determine association rules that highlight general trends in the data-set, it is especially useful in the analysis of super-market items in our data set

Implementing the alogrithm

Reading both minimum support and minimum confidance from the user

```
min_support <- as.numeric(readline("Enter the minimum Support : "))
min_conf <- as.numeric(readline("Enter the minimum Confidance : "))
```

implementing the algorithm using the built-in function

```
apriori_rules <- apriori(
  tdata,
  parameter = list(supp = min_support, conf = min_conf, minlen = 2))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen maxlen target  ext
##         0.5    0.1    1 none FALSE            TRUE       5    0.01      2     10  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 98
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[169 item(s), 9835 transaction(s)] done [0.00s].
## sorting and recoding items ... [88 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 done [0.00s].
## writing ... [15 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```r
# Displaying at most 100 rows of the rules
as_tibble(DATAFRAME(apriori_rules,separate = TRUE, setStart = "", setEnd = "")) %>%
  print(n = 100, width = 90)
```

```
## # A tibble: 15 x 7
##    LHS                                RHS         support confidence coverage  lift count
##    <fct>                              <fct>         <dbl>      <dbl>    <dbl> <dbl> <int>
##  1 curd,yogurt                        whole milk   0.0101      0.582   0.0173  2.28    99
##  2 butter,other vegetables            whole milk   0.0115      0.574   0.0200  2.24   113
##  3 domestic eggs,other vegetables     whole milk   0.0123      0.553   0.0223  2.16   121
##  4 whipped/sour cream,yogurt          whole milk   0.0109      0.525   0.0207  2.05   107
##  5 other vegetables,whipped/sour cream whole milk  0.0146      0.507   0.0289  1.98   144
##  6 other vegetables,pip fruit         whole milk   0.0135      0.518   0.0261  2.03   133
##  7 citrus fruit,root vegetables       other vege~  0.0104      0.586   0.0177  3.03   102
##  8 root vegetables,tropical fruit     other vege~  0.0123      0.585   0.0210  3.02   121
##  9 root vegetables,tropical fruit     whole milk   0.0120      0.570   0.0210  2.23   118
## 10 tropical fruit,yogurt              whole milk   0.0151      0.517   0.0293  2.02   149
## 11 root vegetables,yogurt             other vege~  0.0129      0.5     0.0258  2.58   127
## 12 root vegetables,yogurt             whole milk   0.0145      0.563   0.0258  2.20   143
## 13 rolls/buns,root vegetables         other vege~  0.0122      0.502   0.0243  2.59   120
## 14 rolls/buns,root vegetables         whole milk   0.0127      0.523   0.0243  2.05   125
## 15 other vegetables,yogurt            whole milk   0.0223      0.513   0.0434  2.01   219
```