The MIBE-LSU challenge

Om Ajay Karthik

3/10/2021

This report consists of detailed analyses of food delivery scene in London evaluated based on both the datasheets, one with the restaurant data and another with restaurant delivery time details. Using R, I have tried to analyze various relationships and extract the required details. I am presenting the detailed analysis below.

Restaurants Information In this section, you will analyze the Restaurant information dataset with the objective of answering the following questions:

1. Present a column chart with the top 10 neighborhoods by the number of restaurants.
2. Present a column chart with the top 10 neighborhoods by restaurant review score.
3. Compute the top 10 biggest chains. Present the results in a tabular format. (Use the column rest_brand to determine restaurants of the same brand)
4. Compute the average menu price and the number of menu items for each restaurant. (The rest_menu_item_price column is a list of characters. You might want to use the by_row and map functions)
5. Present in a bar chart the number of items on the menu for the five most expensive and cheapest restaurants. (The number of items can be determined by counting the elements in the rest_menu_item_price column)

Installing necessary packages

```
#install.packages('ggplot2')
#install.packages('dplyr')
#install.packages("tidyverse")

#install.packages("xlsx")
#install.packages('expss')
#install.packages('gridExtra')
#install.packages('purrrlyr')
#install.packages('ggpubr')
#install.packages('data.table')
```

Importing necessary libraries

```
library(ggpubr)
```

```
## Loading required package: ggplot2
```

```
library(ggplot2)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --
```

```
## v tibble  3.0.6     v purrr   0.3.4
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x gridExtra::combine() masks dplyr::combine()
## x dplyr::filter()      masks stats::filter()
## x dplyr::lag()         masks stats::lag()
```

```
library(purrrlyr)
library(data.table)
```

```
##
## Attaching package: 'data.table'
```

```
## The following object is masked from 'package:purrr':
##
##     transpose
```

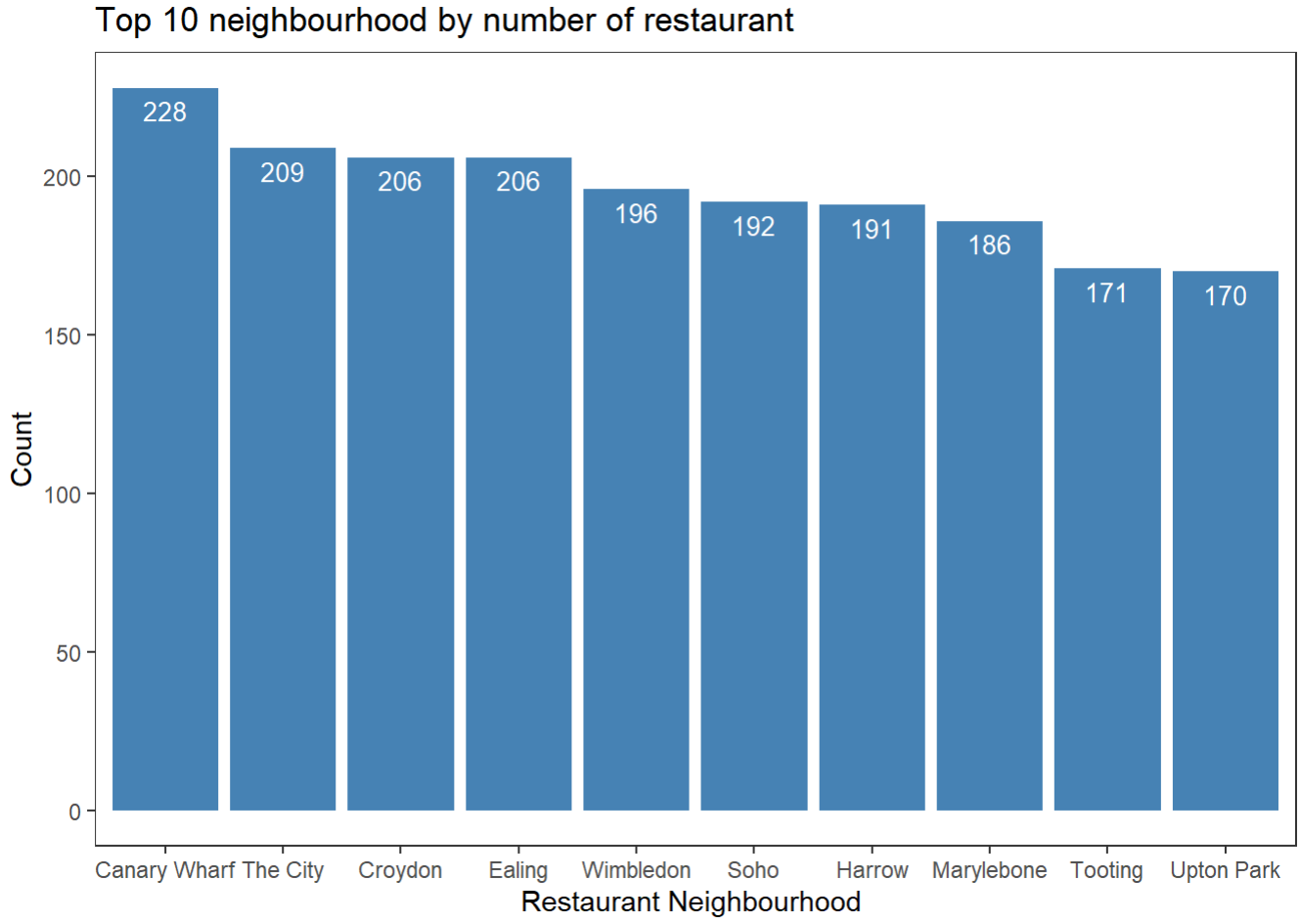
```
## The following objects are masked from 'package:dplyr':
##
##     between, first, last
```

Imporing datasets

```
resturants_mibe <- readRDS("C:/Users/Om Ajay Karthik/Desktop/LSU/resturants-mibe.rds")
delivery_mibe <- readRDS("C:/Users/Om Ajay Karthik/Desktop/LSU/delivery-mibe.rds")
```

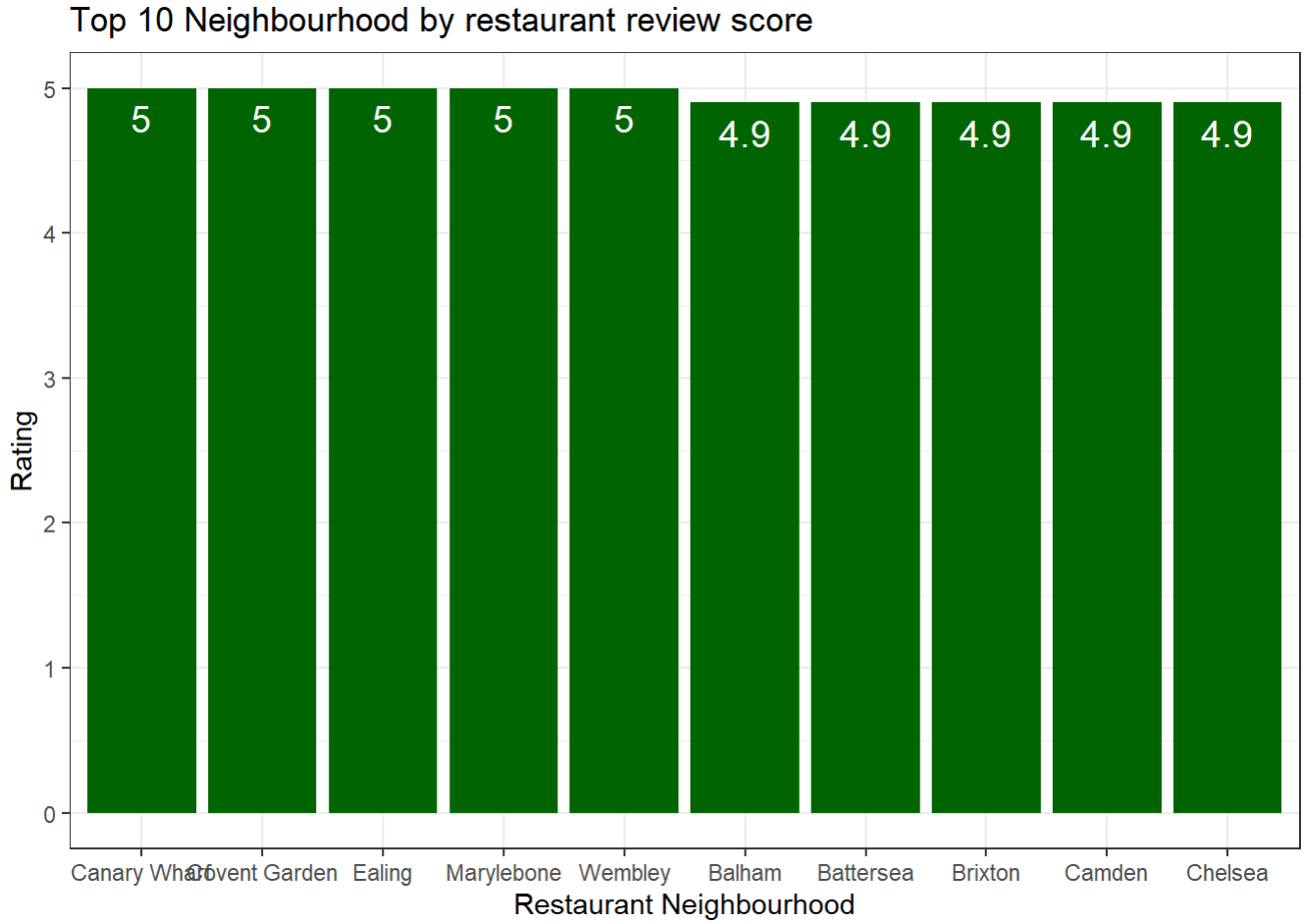
1. Present a column chart with the top 10 neighborhoods by the number of restaurants.

```
freq_rest_neighborhood= count(resturants_mibe, rest_neighborhood)
freq_rest_neighborhood_desc <-freq_rest_neighborhood[order(-freq_rest_neighborhood$n),]
freq_rest_neighborhood_desc_top_n = head(freq_rest_neighborhood_desc, n=10)

ggplot(freq_rest_neighborhood_desc_top_n, aes(x = reorder(rest_neighborhood, -n), y = n)) +
  geom_bar(stat = "identity") +
  geom_bar(stat="identity", fill="steelblue")+
  geom_text(aes(label=n), vjust=1.6, color="white", size=3.5)+
  labs(title = "Top 10 neighbourhood by number of restaurant") +
  xlab("Restaurant Neighbourhood ") +
  ylab("Count") +
  theme_test()
```



2. Present a column chart with the top 10 neighborhoods by restaurant review score.

```
topn_by_review = head(resturants_mibe  %>%
  filter(!is.na(rest_rating))  %>% arrange(-rest_rating,rest_neighborhood) %>% select(rest_neigh
borhood,rest_rating) %>% unique(),n=10)

ggplot(topn_by_review, aes(x = reorder(rest_neighborhood, -rest_rating), y = rest_rating)) +
  geom_bar(stat="identity", fill="darkgreen")+
  geom_text(aes(label=rest_rating), vjust=1.6, color="white", size=5)+
  labs(title = "Top 10 Neighbourhood by restaurant review score") +
  xlab("Restaurant Neighbourhood") +
  ylab("Rating") +
  theme_bw()
```



3. Compute the top 10 biggest chains. Present the results in a tabular format. (Use the column rest_brand to determine restaurants of the same brand)

```
graphics.off()
biggest_chains = resturants_mibe %>% filter(!is.na(rest_brand)) %>%  count(.,rest_brand)
names(biggest_chains)[1] <- "Restaurant_Brand"
names(biggest_chains)[2] <- "Count"
biggest_chains_desc <-biggest_chains[order(-biggest_chains$Count),]
biggest_chains_desc_top_n = head(biggest_chains_desc, n=10)
```

Custom display of of the table since Rmd has a bug while displaying this code:::grid.draw(biggest_chains_desc_top_n):::

```
writeLines("td, th { padding : 6px } th { background-color : brown ; color : white; border : 1px solid white; } td { color : brown ; border : 1px solid brown }", con = "mystyle.css")
dset1 <- head(biggest_chains_desc_top_n,n=10)
knitr::kable(dset1, format = "html")
```

Restaurant_Brand	Count
Get drinks delivered	42
KFC	42
PizzaExpress	42
Pret A Manger	33
Burger King	22
itsu	22
Pure	21
Wasabi	20
LEON	19
Papa John’s	18

4. Compute the average menu price and the number of menu items for each restaurant (The rest_menu_item_price column is a list of characters. You might want to use the by_row and map functions)

```
resturants_mibe_rest_menu_item_price_mean = resturants_mibe %>%
  select(rest_name,rest_brand,rest_menu_item_price) %>%
  by_row(..f = function(this_row) {
    this_row[3] %>% unlist %>% mean})

names(resturants_mibe_rest_menu_item_price_mean)[4] <- "Average_price"

resturants_mibe_rest_menu_item_price_length = resturants_mibe %>%
  select(rest_name,rest_brand,rest_menu_item_price) %>%
  by_row(..f = function(this_row) {
    this_row[3] %>% unlist %>% length()})

names(resturants_mibe_rest_menu_item_price_length)[4] <- "Count_of_menu_items"
```

Average menu price for each resturant(printing only head of 5(top 5))

```
dset1 <- head(resturants_mibe_rest_menu_item_price_mean,n=5)
knitr::kable(dset1, format = "html")
```

rest_name	rest_brand	rest_menu_item_price	Average_price

rest_name	rest_brand	rest_menu_item_price	Average_price
Baba Wali Hendon Broadway	NA	2.80, 4.20, 5.60, 4.20, 5.60, 14.00, 16.80, 4.20, 14.00, 14.00, 14.00, 200.00, 168.00, 84.00, 14.00, 14.00, 11.20, 11.20, 49.00, 14.00, 16.80, 16.80, 11.20, 12.60, 16.80, 42.00, 50.40, 14.00, 12.60, 11.20, 11.20, 12.59, 8.39, 13.99, 9.79, 9.79, 12.59, 2.80, 1.40, 2.10, 2.10, 1.40, 1.40, 1.40, 4.20, 2.80, 2.80, 2.80, 1.40, 5.00, 10.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00	17.05414
Burger & Lobster	Burger & Lobster	0.0, 1.0, 1.5, 1.5, 5.0, 3.0, 25.0, 0.0, 38.0, 0.0, 0.0, 12.5, 9.5, 0.0, 1.5, 1.5, 1.5, 5.0, 5.0, 12.0, 0.0, 75.0, 0.0, 0.0, 27.0, 8.5, 0.0, 0.0, 1.0, 5.0, 3.0, 17.0, 0.0, 33.0, 22.0, 10.0, 0.0, 0.0, 1.5, 1.0, 5.0, 15.0, 0.0, 6.5, 1.5, 0.0, 5.0, 20.0, 0.0, 1.5, 0.0, 1.0, 6.0, 0.0, 1.5, 0.0, 1.5, 0.0, 0.0, 0.0, 0.0	6.42623
Afta Eats	NA	7.49, 5.19, 0.00, 7.49, 5.19, 0.65, 7.49, 5.19, 1.30, 1.49, 5.19, 5.19, 0.00, 5.19, 6.49, 0.00, 5.19, 5.19, 5.19, 6.49, 5.19, 6.49, 5.19, 5.19, 5.19, 5.19, 5.19, 5.19, 5.19, 3.89, 3.89, 3.89, 3.89, 3.89, 3.89, 3.89, 3.89, 5.19, 5.19, 5.19, 5.19, 5.19, 5.19, 1.68, 1.81, 2.98, 3.24, 3.63, 3.89	4.34898
Europa 2 Go Pizza	Europa 2 Go Pizza	9.99, 13.95, 17.95, 17.95, 17.95, 24.95, 14.99, 17.99, 4.49, 4.49, 5.49, 5.49, 3.49, 3.49, 4.49, 4.49, 4.49, 4.49, 5.49, 5.49, 5.49, 1.95, 2.95, 3.00, 3.50, 3.95, 4.95, 3.95, 3.95, 4.50, 4.50, 2.50, 2.95, 3.95, 2.50, 1.50, 0.95, 2.50, 3.00, 3.00, 3.50, 3.50, 3.50, 3.50, 3.50, 3.50, 3.50, 3.50, 3.50, 3.50, 3.50, 3.99, 3.99, 3.99, 3.99, 3.99, 3.99, 3.99, 3.99, 3.99, 3.99, 3.99, 3.99, 3.99, 3.99, 5.99, 6.45, 6.45, 6.45, 6.45, 6.45, 6.99, 6.99, 6.99, 6.99, 6.99, 6.99, 7.49, 7.49, 7.49, 7.49, 7.49, 7.49, 7.49, 7.49, 7.49, 7.49, 7.49, 7.49, 7.49, 7.49, 7.99, 9.45, 9.45, 9.45, 9.45, 9.45, 9.99, 9.99, 9.99, 9.99, 9.99, 9.99, 10.49, 10.49, 10.49, 10.49, 10.49, 10.49, 10.49, 10.49, 10.49, 10.49, 10.49, 10.49, 10.49, 10.49, 9.99, 11.45, 11.45, 11.45, 11.45, 11.45, 11.99, 11.99, 11.99, 11.99, 11.99, 11.99, 12.99, 12.99, 12.99, 12.99, 12.99, 12.99, 12.99, 12.99, 12.99, 12.99, 12.99, 12.99, 12.99, 12.99, 3.00, 3.20, 4.00, 1.95, 3.00, 3.00, 3.00, 3.00, 1.50, 3.00, 1.70, 3.00, 3.50, 3.50, 3.50, 1.10, 4.50, 5.49, 4.50, 2.50, 3.95, 3.95, 3.95, 0.95, 0.95, 0.50, 5.95, 3.99, 3.99, 2.50, 2.50, 2.50, 0.90, 0.90, 0.90, 0.90, 0.90, 0.90, 0.90, 0.90, 0.00, 0.50, 1.00, 1.50, 0.50, 0.50, 0.50, 0.60, 1.50, 2.00, 0.60, 0.60, 0.60, 0.80, 2.00, 2.50, 0.80, 0.80, 0.80, 1.00, 2.50, 3.00, 1.00, 1.00, 1.00, 0.00, 1.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 1.00, 1.90, 3.90, 8.40, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 1.60, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00	3.42838

rest_name	rest_brand	rest_menu_item_price	Average_price
Julia Domna Cafe	NA	5.53, 11.48, 13.79, 7.70, 8.33, 6.93, 9.03, 7.35, 8.33, 6.30, 4.90, 5.46, 8.12, 8.33, 9.10, 15.40, 8.12, 7.63, 7.28, 9.10, 12.60, 6.30, 6.65, 6.65, 8.05, 6.93, 8.05, 8.05, 8.33, 7.00, 9.10, 9.10, 9.10, 6.65, 7.35, 7.00, 9.10, 9.59, 9.10, 8.40, 8.05, 8.75, 8.05, 8.38, 7.70, 14.70, 14.70, 14.70, 8.40, 5.60, 5.95, 4.20, 4.55, 4.55, 4.55, 3.86, 4.20, 4.20, 4.20, 3.50, 3.50, 4.20, 3.85, 5.60	7.675781

No of items on the menu for each resturant (printing only head of 5(top 5))

```
dset1 <- head(resturants_mibe_rest_menu_item_price_length,n=5)
knitr::kable(dset1, format = "html")
```

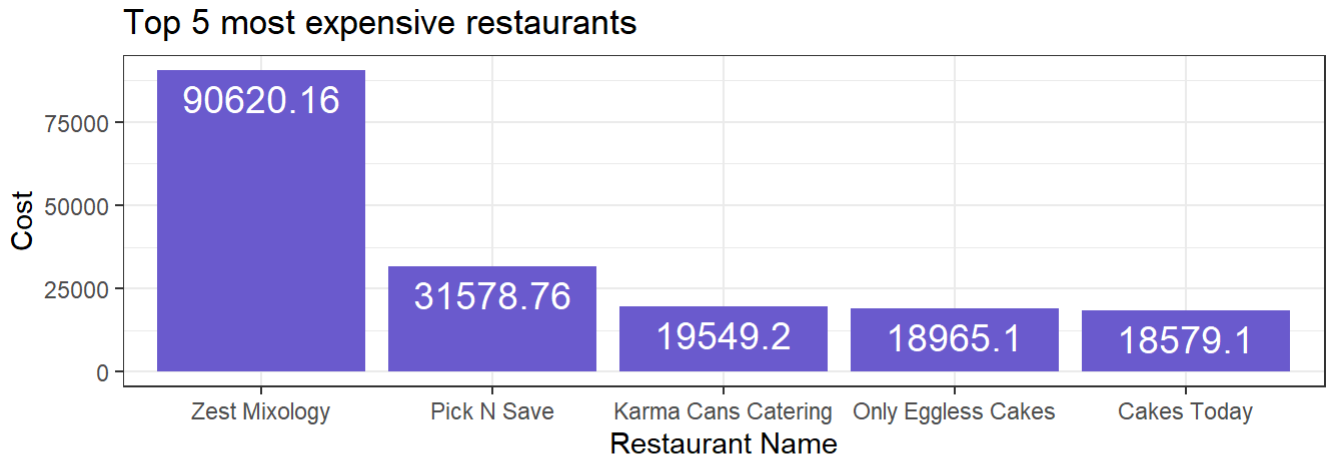
rest_name	rest_brand	rest_menu_item_price	Count_of_menu_items
Baba Wali Hendon Broadway	NA	2.80, 4.20, 5.60, 4.20, 5.60, 14.00, 16.80, 4.20, 14.00, 14.00, 14.00, 200.00, 168.00, 84.00, 14.00, 14.00, 11.20, 11.20, 49.00, 14.00, 16.80, 16.80, 11.20, 12.60, 16.80, 42.00, 50.40, 14.00, 12.60, 11.20, 11.20, 12.59, 8.39, 13.99, 9.79, 9.79, 12.59, 2.80, 1.40, 2.10, 2.10, 1.40, 1.40, 1.40, 4.20, 2.80, 2.80, 2.80, 1.40, 5.00, 10.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00	58
Burger & Lobster	Burger & Lobster	0.0, 1.0, 1.5, 1.5, 5.0, 3.0, 25.0, 0.0, 38.0, 0.0, 0.0, 12.5, 9.5, 0.0, 1.5, 1.5, 1.5, 5.0, 5.0, 12.0, 0.0, 75.0, 0.0, 0.0, 27.0, 8.5, 0.0, 0.0, 1.0, 5.0, 3.0, 17.0, 0.0, 33.0, 22.0, 10.0, 0.0, 0.0, 1.5, 1.0, 5.0, 15.0, 0.0, 6.5, 1.5, 0.0, 5.0, 20.0, 0.0, 1.5, 0.0, 1.0, 6.0, 0.0, 1.5, 0.0, 1.5, 0.0, 0.0, 0.0, 0.0	61
Afta Eats	NA	7.49, 5.19, 0.00, 7.49, 5.19, 0.65, 7.49, 5.19, 1.30, 1.49, 5.19, 5.19, 0.00, 5.19, 6.49, 0.00, 5.19, 5.19, 5.19, 6.49, 5.19, 6.49, 5.19, 5.19, 5.19, 5.19, 5.19, 5.19, 5.19, 5.19, 3.89, 3.89, 3.89, 3.89, 3.89, 3.89, 3.89, 3.89, 5.19, 5.19, 5.19, 5.19, 5.19, 5.19, 1.68, 1.81, 2.98, 3.24, 3.63, 3.89	49

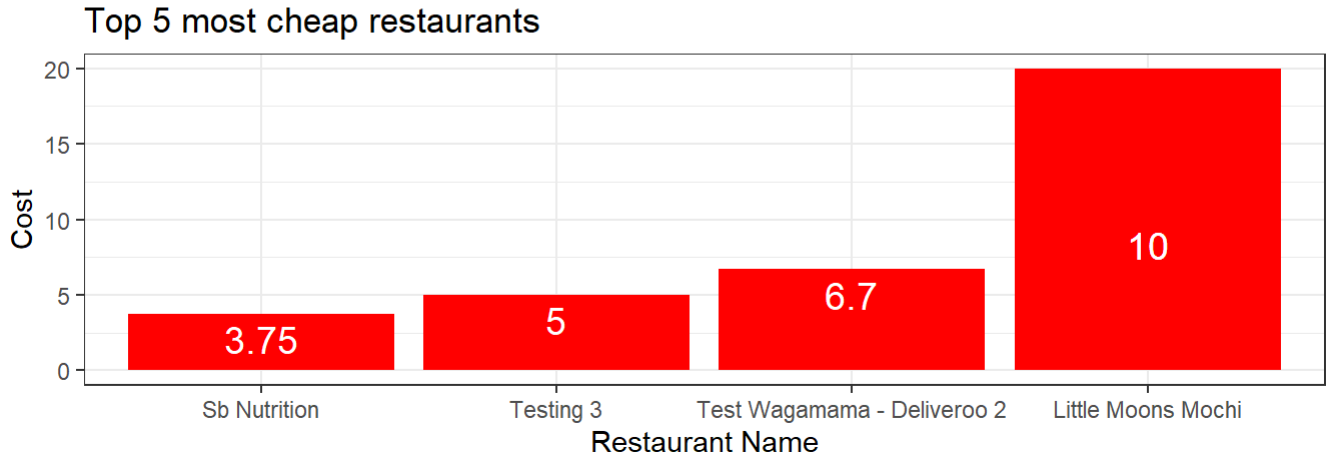
rest_name	rest_brand	rest_menu_item_price	Count_of_menu_items
Europa 2 Go Pizza	Europa 2 Go Pizza	9.99, 13.95, 17.95, 17.95, 17.95, 24.95, 14.99, 17.99, 4.49, 4.49, 5.49, 5.49, 3.49, 3.49, 4.49, 4.49, 4.49, 4.49, 5.49, 5.49, 5.49, 1.95, 2.95, 3.00, 3.50, 3.95, 4.95, 3.95, 3.95, 4.50, 4.50, 2.50, 2.95, 3.95, 2.50, 1.50, 0.95, 2.50, 3.00, 3.00, 3.50, 3.50, 3.50, 3.50, 3.50, 3.50, 3.50, 3.50, 3.50, 3.50, 3.50, 3.99, 3.99, 3.99, 3.99, 3.99, 3.99, 3.99, 3.99, 3.99, 3.99, 3.99, 3.99, 3.99, 3.99, 5.99, 6.45, 6.45, 6.45, 6.45, 6.45, 6.99, 6.99, 6.99, 6.99, 6.99, 6.99, 7.49, 7.49, 7.49, 7.49, 7.49, 7.49, 7.49, 7.49, 7.49, 7.49, 7.49, 7.49, 7.49, 7.49, 7.99, 9.45, 9.45, 9.45, 9.45, 9.45, 9.99, 9.99, 9.99, 9.99, 9.99, 9.99, 10.49, 10.49, 10.49, 10.49, 10.49, 10.49, 10.49, 10.49, 10.49, 10.49, 10.49, 10.49, 10.49, 10.49, 9.99, 11.45, 11.45, 11.45, 11.45, 11.45, 11.99, 11.99, 11.99, 11.99, 11.99, 11.99, 12.99, 12.99, 12.99, 12.99, 12.99, 12.99, 12.99, 12.99, 12.99, 12.99, 12.99, 12.99, 12.99, 12.99, 12.99, 3.00, 3.20, 4.00, 1.95, 3.00, 3.00, 3.00, 3.00, 1.50, 3.00, 1.70, 3.00, 3.50, 3.50, 3.50, 1.10, 4.50, 5.49, 4.50, 2.50, 3.95, 3.95, 3.95, 0.95, 0.95, 0.50, 5.95, 3.99, 3.99, 2.50, 2.50, 2.50, 0.90, 0.90, 0.90, 0.90, 0.90, 0.90, 0.90, 0.90, 0.00, 0.50, 1.00, 1.50, 0.50, 0.50, 0.50, 0.60, 1.50, 2.00, 0.60, 0.60, 0.60, 0.80, 2.00, 2.50, 0.80, 0.80, 0.80, 1.00, 2.50, 3.00, 1.00, 1.00, 1.00, 0.00, 1.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 1.00, 1.90, 3.90, 8.40, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 1.60, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00, 0.00	389
Julia Domna Cafe	NA	5.53, 11.48, 13.79, 7.70, 8.33, 6.93, 9.03, 7.35, 8.33, 6.30, 4.90, 5.46, 8.12, 8.33, 9.10, 15.40, 8.12, 7.63, 7.28, 9.10, 12.60, 6.30, 6.65, 6.65, 8.05, 6.93, 8.05, 8.05, 8.33, 7.00, 9.10, 9.10, 9.10, 6.65, 7.35, 7.00, 9.10, 9.59, 9.10, 8.40, 8.05, 8.75, 8.05, 8.38, 7.70, 14.70, 14.70, 14.70, 8.40, 5.60, 5.95, 4.20, 4.55, 4.55, 4.55, 3.86, 4.20, 4.20, 4.20, 3.50, 3.50, 4.20, 3.85, 5.60	64

5. Present in a bar chart the number of items on the menu for the five most expensive and cheapest restaurants.(The number of items can be determined by counting the elements in the

rest_menu_item_price column)

```
menu_item_price_sum = resturants_mibe %>% select(rest_name,rest_neighborhood,rest_menu_item_pric
e) %>% filter(!is.na(rest_menu_item_price )) %>% by_row(..f = function(this_row) {
  this_row[3] %>% unlist %>% sum()})

names(menu_item_price_sum)[4] <- "sum_of_price"
menu_item_price_sum$sum_of_price = as.numeric(as.character(menu_item_price_sum$sum_of_price))


menu_item_sum_top_5 = head(arrange(menu_item_price_sum,desc(sum_of_price)), n = 5)
menu_item_sum_bottom_5 = head(arrange(menu_item_price_sum,sum_of_price), n = 5)

expensive = ggplot(menu_item_sum_top_5, aes(x = reorder(rest_name, -sum_of_price), y = sum_of_pr
ice)) +
  geom_bar(stat="identity", fill="slateblue")+
  geom_text(aes(label=sum_of_price), vjust=1.5, color="white", size=5)+
  labs(title = "Top 5 most expensive restaurants") +
  xlab("Restaurant Name") +
  ylab("Cost") +
  theme_bw()

cheapest = ggplot(menu_item_sum_bottom_5, aes(x = reorder(rest_name, sum_of_price), y = sum_of_p
rice)) +
  geom_bar(stat="identity", fill="red")+
  geom_text(aes(label=sum_of_price), vjust=1.4, color="white", size=5)+
  labs(title = "Top 5 most cheap restaurants") +
  xlab("Restaurant Name") +
  ylab("Cost") +
  theme_bw()


grid.arrange(expensive, cheapest)
```





Resturant Delivery Times

Imporing datasets

```
resturants_mibe <- readRDS("C:/Users/Om Ajay Karthik/Desktop/LSU/resturants-mibe.rds")
delivery_mibe <- readRDS("C:/Users/Om Ajay Karthik/Desktop/LSU/delivery-mibe.rds")
```

Right outer join table and create a duplicate dataframe

```
resturant_delivery = merge(x = resturants_mibe, y = delivery_mibe, by.x='restaurant_id', by.y='r
est_key', all.y = TRUE)

rest_delivery_merged_subset = resturant_delivery %>% select(restaurant_id,rest_name,rest_neighbo
rhood,rest_postcode,neighborhood_name,rest_rating,rest_delivery_time_min)
```

1. Count the number of neighborhoods where each restaurant delivers. (Each restaurant can deliver to multiple neighborhoods, independently from here the restaurants is physically located)

```
neighbourhoods_delivery_count = rest_delivery_merged_subset %>%
  select(rest_name,neighborhood_name)   %>%
  group_by(rest_name,neighborhood_name) %>%
  unique() %>%
  summarise(n = n()) %>% count()
```

```
## `summarise()` has grouped output by 'rest_name'. You can override using the `.groups` argumen
t.
```

```
names(neighbourhoods_delivery_count)[1] <- "Restaurant Name"
names(neighbourhoods_delivery_count)[2] <- "No of Neighbourhood Delivery by Resturant"
```

```
writeLines("td, th { padding : 6px } th { background-color : brown ; color : white; border : 1px
solid white; } td { color : brown ; border : 1px solid brown }", con = "mystyle.css")
dset1 <- head(neighbourhoods_delivery_count,n=10)
knitr::kable(dset1, format = "html")
```

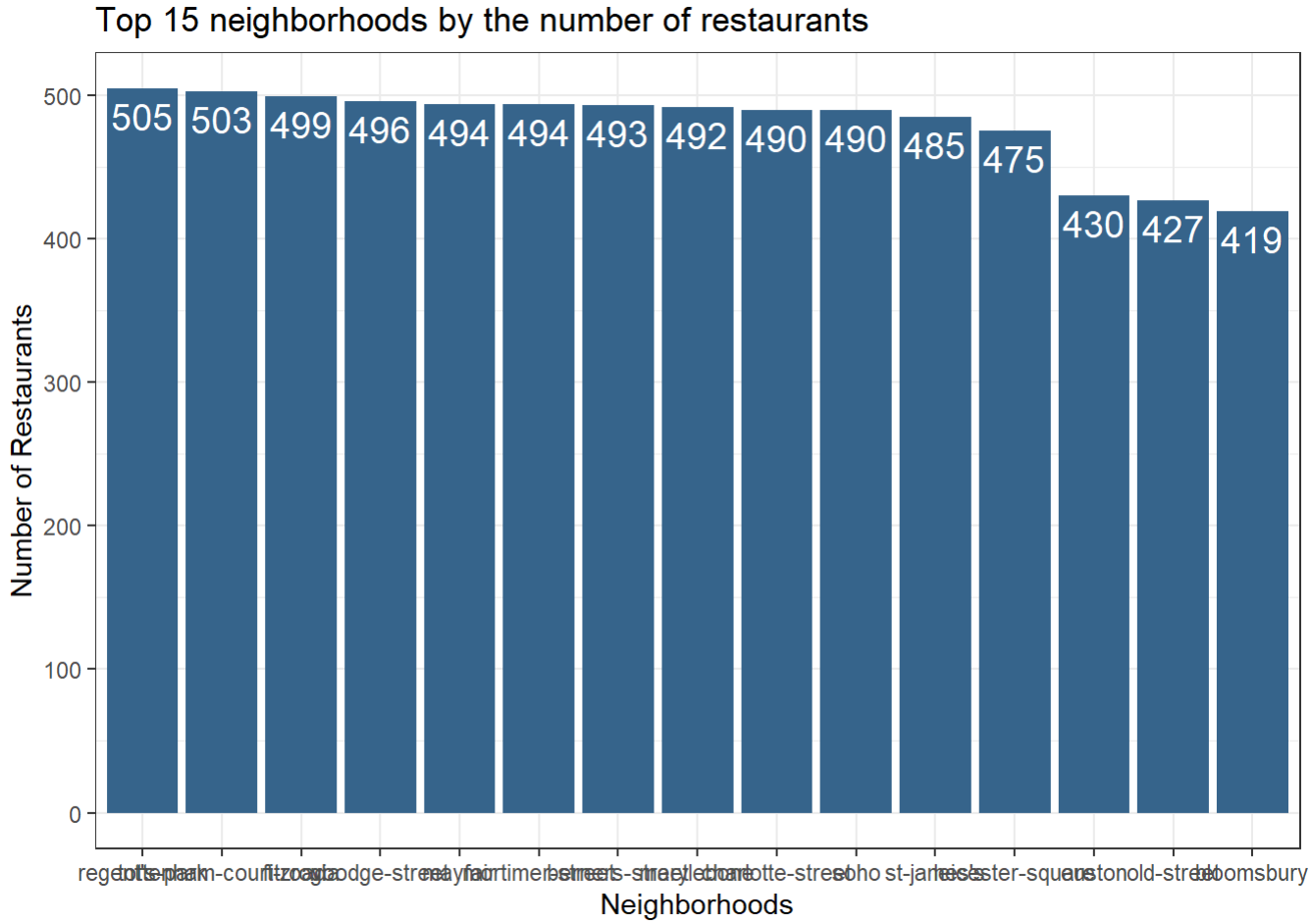
Restaurant Name	No of Neighbourhood Delivery by Resturant
&Cake – Wembley	6
<U+0001F1EF><U+0001F1F5><U+0001F1E7><U+0001F1F7> YOOBI <U+0001F363>	16
<U+0001F336> BOMBAY BICYCLE CHEF <U+0001F336>	16
<U+0001F3C4> Honi Poke - Mayfair<U+0001F3C4>	31
<U+0001F3C4> Honi Poke - Soho<U+0001F3C4>	10
<U+0001F451> Benito’s Hat <U+0001F451>	18
<U+5927><U+534E><U+5C0F><U+5403> Da Hua Street Food	5
<U+5927><U+534E><U+5FEB><U+9910>(Dahua Fast Food)	5
<U+5A46><U+5A46> Popo Chinese, Oriental, Thai & Dim Sum Cuisine	5
<U+6771><U+5712><U+5C0F><U+9928> - East Garden	2

2. Present in a bar chart the top 15 neighborhoods by the number of restaurants where restaurants make deliveries.(You will need to count the number of restaurants that deliver to each neighborhood)

```
restaurants_per_neighbourhood = rest_delivery_merged_subset %>%
  select(rest_name,neighborhood_name)   %>%
  group_by(neighborhood_name,rest_name) %>%
  unique() %>%
  summarise(n = n()) %>%
  count()%>%
  arrange(-n)
```

```
## `summarise()` has grouped output by 'neighborhood_name'. You can override using the `.groups`
argument.
```

```
names(restaurants_per_neighbourhood)[2] <- "No_of_Restaurants"

top_n_restaurants_per_neighbourhood = head(restaurants_per_neighbourhood,n=15)
ggplot(top_n_restaurants_per_neighbourhood, aes(x = reorder(neighborhood_name, -No_of_Restaurant
s), y = No_of_Restaurants)) +
  geom_bar(stat="identity", fill="steelblue4")+
  geom_text(aes(label=No_of_Restaurants), vjust=1.5, color="white", size=5)+
  labs(title = "Top 15 neighborhoods by the number of restaurants") +
  xlab("Neighborhoods") +
  ylab("Number of Restaurants") +
  theme_bw()
```



3. Compute the average delivery time for each restaurant. (Compute the average of all neighborhoods the restaurant delivers)

```
resturant_avg_delivery_time = rest_delivery_merged_subset %>%
  select(rest_name,neighborhood_name,rest_delivery_time_min)   %>%
  filter(!is.na(rest_delivery_time_min)) %>%
  group_by(rest_name) %>%
  summarise(average_delivery_time = mean(rest_delivery_time_min))
```

```
writeLines("td, th { padding : 6px } th { background-color : brown ; color : white; border : 1px
solid white; } td { color : brown ; border : 1px solid brown }", con = "mystyle.css")
dset1 <- head(resturant_avg_delivery_time,n=50)
knitr::kable(dset1, format = "html")
```

rest_name	average_delivery_time
&Cake – Wembley	22.77778
<U+0001F1EF><U+0001F1F5><U+0001F1E7><U+0001F1F7> YOOBI <U+0001F363>	20.62500
<U+0001F336> BOMBAY BICYCLE CHEF <U+0001F336>	36.91176
<U+0001F3C4> Honi Poke - Mayfair<U+0001F3C4>	21.07143
<U+0001F3C4> Honi Poke - Soho<U+0001F3C4>	11.62500
<U+0001F451> Benito’s Hat <U+0001F451>	24.22535
<U+5927><U+534E><U+5C0F><U+5403> Da Hua Street Food	25.00000
<U+5927><U+534E><U+5FEB><U+9910>(Dahua Fast Food)	25.00000
<U+5A46><U+5A46> Popo Chinese, Oriental, Thai & Dim Sum Cuisine	22.66667
<U+6771><U+5712><U+5C0F><U+9928> - East Garden	21.66667
<U+7C73>Kome Japanese & Korean Sushi	18.51562
<U+98DF><U+5168><U+5C0F><U+53A8> FODAL Oriental Kitchen	17.50000
11 Coffee and Co	27.14286
14228 - Burger King	21.87500
2 Girls’ Cafe	21.58537
201 Asian Kitchen	27.50000
24/7 Supermarket	16.00000
36 Streets Food	27.36111
4021 Firezza	15.00000
491094Adam’s Cafe	28.23529
5 Star Pizza	23.25000
500 Degrees Pizzeria	17.33333
696 mangal	16.83333
805 Restaurant	30.00000
A’La Pizza	26.17647
A Burgers by Taster	23.43750
A K Chicken	21.11111
A&Y Wines Beers	13.98438
A.D Food & Wine	16.18750
A1 Kebabish	21.66667
Aachis Masala Chennai	25.00000
Abbotshill Wine Bar & Deli	16.66667
Abby’s Food and Wine	23.57143
Abshar	29.16667
Abu Zaad	25.00000
Abu Zaad - Ealing	20.00000
Acacus	39.00000
Acai Berry	12.30159
Açaí Sisters	31.56716
Adam’s Ethiopian Restaurant	51.94444
Adana	22.26190
Ading’s Kitchen	31.56250

rest_name	average_delivery_time
AFC Fried Chicken & Pizza	31.30952
Afters	20.95238
Afters Original	21.66667
Aglio e Olio	31.25000
Agora Greek Bakery	16.62500
Aha Wok	23.12500
Ahaa Dosa	25.29412
Ahi Poké	12.68657

4. Present in tabular format the top 20 restaurants by fasted average delivery time.In the same table, present the rating score, and postcode.

```
resturant_avg_delivery_time_rating_postcode = rest_delivery_merged_subset %>%
select(rest_name,neighborhood_name,rest_rating,rest_delivery_time_min,rest_postcode)   %>%
group_by(rest_name) %>%
unique() %>%
summarise(rest_name,neighborhood_name,rest_rating,rest_delivery_time_min,rest_postcode,average_d
elivery_time = mean(rest_delivery_time_min))
```

```
## `summarise()` has grouped output by 'rest_name'. You can override using the `.groups` argumen
t.
```

```
resturant_avg_delivery_time_rating_postcode_desc <-resturant_avg_delivery_time_rating_postcode[o
rder(resturant_avg_delivery_time_rating_postcode$average_delivery_time),]
resturant_avg_delivery_time_rating_postcode_desc_top_n = resturant_avg_delivery_time_rating_post
code_desc %>% select(rest_name,rest_postcode,neighborhood_name,rest_rating,average_delivery_tim
e) %>% head(., n=20)
```

```
writeLines("td, th { padding : 6px } th { background-color : brown ; color : white; border : 1px
solid white; } td { color : brown ; border : 1px solid brown }", con = "mystyle.css")
dset1 <- head(resturant_avg_delivery_time_rating_postcode_desc_top_n,n=50)
knitr::kable(dset1, format = "html")
```

rest_name	rest_postcode	neighborhood_name	rest_rating	average_delivery_time
Vagabond Wines - Fulham	SW61AX	fulham	4.8	8.000000
Vagabond Wines - Fulham	SW61AX	parsons-green	4.8	8.000000
Vagabond Wines - Fulham	SW61AX	earl’s-court	4.8	8.000000
Vagabond Wines - Fulham	SW61AX	west-kensington	4.8	8.000000
Vagabond Wines - Fulham	SW61AX	south-kensington	4.8	8.000000
OREE	SW109PZ	knightsbridge	4.8	8.333333
OREE	SW109PZ	south-kensington	4.8	8.333333
OREE	SW109PZ	chelsea	4.8	8.333333
OREE	SW109PZ	parsons-green	4.8	8.333333
OREE	SW109PZ	belgravia	4.8	8.333333
OREE	SW109PZ	fulham	4.8	8.333333
Amorino - Old Compton St	W1D6HF	st-james’s	4.8	10.000000

rest_name	rest_postcode	neighborhood_name	rest_rating	average_delivery_time
Amorino - Old Compton St	W1D6HF	leicester-square	4.8	10.000000
Amorino - Old Compton St	W1D6HF	soho	4.8	10.000000
Amorino - Old Compton St	W1D6HF	westminster	4.8	10.000000
Amorino - Old Compton St	W1D6HF	mayfair	4.8	10.000000
Andys Quality Fish & Chips	W139RJ	ealing	4.6	10.000000
Bill’s Breakfast	W52XA	ealing	4.6	10.000000
Brew Café	SW195DX	wimbledon	4.5	10.000000
Chic-o-land Kebab	KT12AA	molesey	NA	10.000000

Open analysis

How can we widen the food delivery market and improve customer experience.

Importing necessary libraries

```
library(ggpubr)
library(ggplot2)
library(dplyr)
library(gridExtra)
library(tidyverse)
library(purrrlyr)
library(data.table)
```

Imporing datasets

```
resturants_mibe <- readRDS("C:/Users/Om Ajay Karthik/Desktop/LSU/resturants-mibe.rds")
delivery_mibe <- readRDS("C:/Users/Om Ajay Karthik/Desktop/LSU/delivery-mibe.rds")

resturant_delivery = merge(x = resturants_mibe, y = delivery_mibe, by.x='restaurant_id', by.y='r
est_key', all.y = TRUE)

rest_delivery_merged_subset = resturant_delivery %>% select(restaurant_id,rest_name,rest_neighbo
rhood,rest_postcode,neighborhood_name,rest_rating,rest_delivery_time_min)
```

```
#summary(resturants_mibe)
```

restaurant_id rest_name rest_brand rest_postcode rest_neighborhood rest_rating
Min. : 3 Length:5786 Length:5786 Length:5786 Length:5786 Min. :1.800
1st Qu.: 62488 Class :character Class :character Class :character Class :character 1st Qu.:4.200
Median :116654 Mode :character Mode :character Mode :character Mode :character Median :4.400
Mean :112608 Mean :4.355
3rd Qu.:167852 3rd Qu.:4.600
Max. :223928 Max. :5.000
NA’s :850

resturant_delivery = merge(x = resturants_mibe, y = delivery_mibe, by.x='restaurant_id', by.y='rest_key', all.y = TRUE)

```
resturant_avg_delivery_time_rating_postcode = rest_delivery_merged_subset %>%
select(rest_name,neighborhood_name,rest_rating,rest_delivery_time_min,rest_postcode)   %>%
group_by(rest_name) %>%
unique() %>%
summarise(rest_name,neighborhood_name,rest_rating,rest_delivery_time_min,rest_postcode,average_d
elivery_time = mean(rest_delivery_time_min))
```

```
## `summarise()` has grouped output by 'rest_name'. You can override using the `.groups` argumen
t.
```

```
resturant_avg_delivery_time_rating_postcode_desc <-resturant_avg_delivery_time_rating_postcode[o
rder(-resturant_avg_delivery_time_rating_postcode$average_delivery_time),]
resturant_avg_delivery_time_rating_postcode_desc_top_n = resturant_avg_delivery_time_rating_post
code_desc %>% select(rest_name,rest_postcode,neighborhood_name,rest_rating,average_delivery_tim
e) %>% head(., n=20)
```

```
writeLines("td, th { padding : 6px } th { background-color : brown ; color : white; border : 1px
solid white; } td { color : brown ; border : 1px solid brown }", con = "mystyle.css")
dset1 <- head(resturant_avg_delivery_time_rating_postcode_desc_top_n,n=50)
knitr::kable(dset1, format = "html")
```

rest_name	rest_postcode	neighborhood_name	rest_rating	average_delivery_time
Peppercorn Food	W127GF	marylebone	NA	120
Peppercorn Food	W127GF	holland-park	NA	120
Peppercorn Food	W127GF	kilburn	NA	120
Peppercorn Food	W127GF	queen's-park	NA	120
Peppercorn Food	W127GF	paddington	NA	120
Peppercorn Food	W127GF	west-kensington	NA	120
Peppercorn Food	W127GF	south-kensington	NA	120
Peppercorn Food	W127GF	shepherd's-bush	NA	120
Peppercorn Food	W127GF	south-hampstead	NA	120
Peppercorn Food	W127GF	chelsea	NA	120
Peppercorn Food	W127GF	hammersmith	NA	120
Peppercorn Food	W127GF	barnes	NA	120
Peppercorn Food	W127GF	brondesbury	NA	120
Peppercorn Food	W127GF	knightsbridge	NA	120
Peppercorn Food	W127GF	acton	NA	120
Peppercorn Food	W127GF	parsons-green	NA	120
Peppercorn Food	W127GF	st-john's-wood	NA	120
Peppercorn Food	W127GF	earl's-court	NA	120
Peppercorn Food	W127GF	south-acton	NA	120
Peppercorn Food	W127GF	westbourne	NA	120

Based on the table above, which shows the average delivery time of 20 restaurants that have highest average delivery time, and the top 20 is from the restaurant chain too that take more than 120 minutes to deliver. While conducting the analysis we realize that there are 957 restaurants that have average delivery time more than 60 minutes. So, to improve the customer experience, focus should be on getting these food chains and restaurants to deliver within an hour so that they can retain customers and also attract new customers.

Also, another analysis on total price versus average rating (considering the total menu items) shows that there are restaurants that are too expensive and also have low rating. Improving the price structure and improving the quality of food and better promotion to the restaurants that are cheap can fetch better business by improving customer experience.