Amir Sela

Homework 1

Warm up

Before we introduce the data, let's warm up with some simple exercises.

- Update the YAML, changing the author name to your name, and **knit** the document.

Packages

We'll use the **tidyverse** package for much of the data wrangling and visualization and the data lives in the **dsbox** package.

```
#Install and load tidyverse package
#install.packages("tidyverse")
library(tidyverse)

#Install and load devtools package - this is where the dataset is stored for this
#install.packages("devtools")
library(devtools)
#devtools::install_github("rstudio-education/dsbox")
library(dsbox)

# NOTE: comments should be added to install lines after running once when knitting
```

Data

The data can be found in the **dsbox** package, and it's called `edibnb`. Since the dataset is distributed with the package, we don't need to load it separately; it becomes available to us when we load the package.

You can view the dataset as a spreadsheet using the `View()` function. Note that you should not put this function in your R Markdown document, but instead type it directly in the Console, as it pops open a new window (and the concept of popping open a window in a static document doesn't really make sense…). When you run this in the console,

you'll see the following **data viewer** window pop up.

```
View(edibnb) # view the data frame
```

You can find out more about the dataset by inspecting its documentation, which you can access by running `?edibnb` in the Console or using the Help menu in RStudio to search for `edibnb`.

Exercises

Hint: The Markdown Quick Reference sheet has an example of inline R code that might be helpful. You can access it from the Help menu in RStudio.

1. How many observations (rows) does the dataset have? Instead of hard coding the number in your answer, use inline code. There are 13245 rows
2. Run `View(edibnb)` in your Console to view the data in the data viewer. What does each row in the dataset represent? *Each row represent an airbnb rental with an id, price per night, neighborhood, how many people it can accomodate, # of bathrooms, # of bedrooms, # of beds, the # of reviews and the avg rating, and also the url where its listed *

Each column represents a variable. We can get a list of the variables in the data frame using the `names()` function.

```
names(edibnb) # view column names
```

```
##  [1] "id"                  "price"               "neighbourhood"
##  [4] "accommodates"        "bathrooms"           "bedrooms"
##  [7] "beds"                "review_scores_rating" "number_of_reviews"
## [10] "listing_url"
```

You can find descriptions of each of the variables in the help file for the dataset, which you can access by running `?edibnb` in your Console.

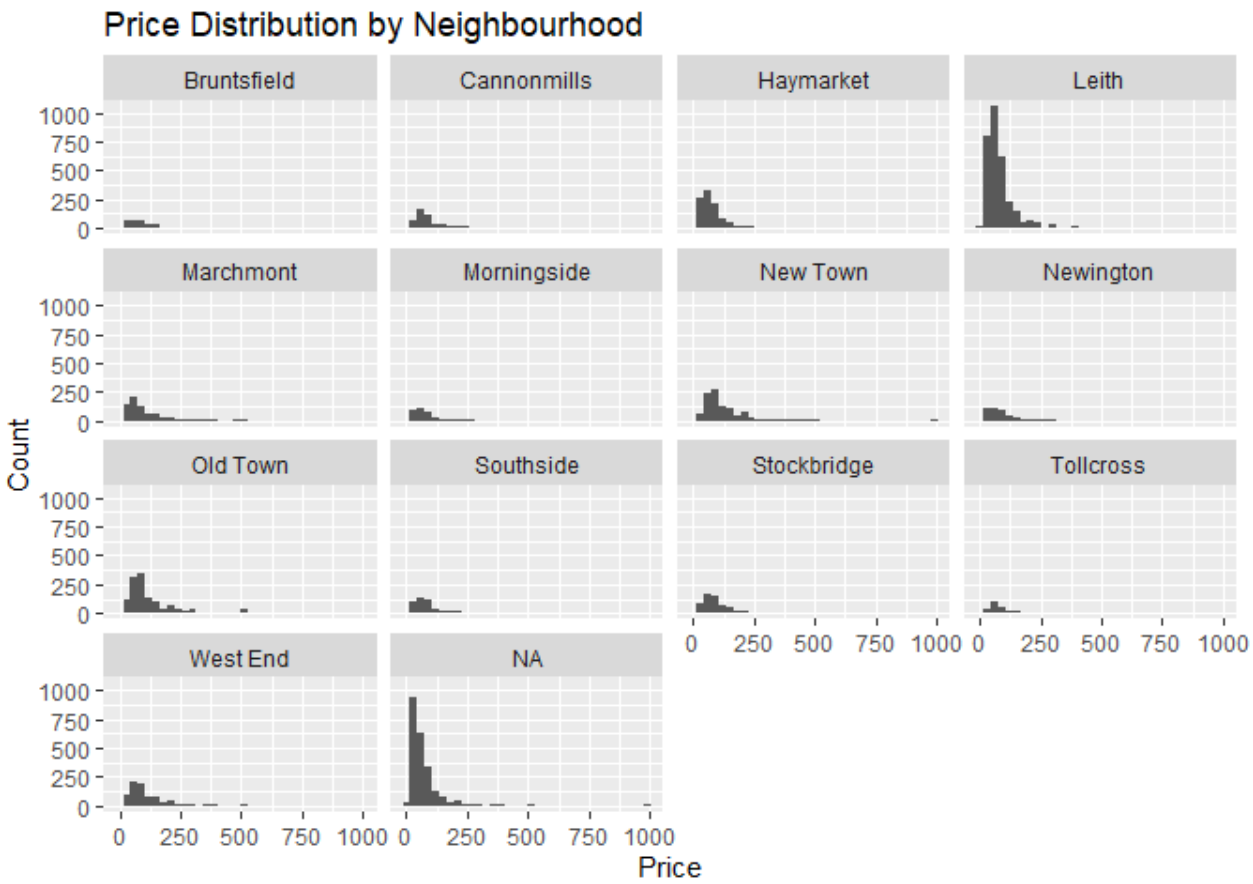
Note: The plot will give a warning about some observations with non-finite values for price being removed. Don't worry about the warning, it simply means that 199 listings in

the data didn't have prices available, so they can't be plotted.

3. Create a faceted histogram where each facet represents a neighbourhood and displays the distribution of Airbnb prices in that neighbourhood. Think critically about whether it makes more sense to stack the facets on top of each other in a column, lay them out in a row, or wrap them around. Along with your visualization, include your reasoning for the layout you chose for your facets.

```
ggplot(data = edibnb, mapping = aes(x = price)) +
  geom_histogram(binwidth = 30) +
  facet_wrap(~neighbourhood) + # facet wrap by neighbourhood
  labs(title = "Price Distribution by Neighbourhood",
       x = "Price",
       y = "Count")
```

```
## Warning: Removed 199 rows containing non-finite outside the scale range
## (`stat_bin()`).
```



In the end we also have a histogram where the neighbourhood names are missing and

the price values of those rows are graphed in the NA histogram. It might be a good idea to keep that just incase because we never know if the missing values are shared equally between the neighbourhoods or are they mostly from one neighbourhood. Because we dont really have much information on that it would be a good idea to keep that histogram for assumptions # the bin width 30 makes a good visualisation because we can still see a shape or a trend in the histogram, the binwidth its not too much or too low. I decided to wrap the histograms because it doesnt matter when change the ncol or nrow, it will still try to fit all of the histograms in a fixed size box, so wrapping them around gives is a goo visualization on the prices

Let's de-construct this code:

- `ggplot()` is the function we are using to build our plot, in layers.
- In the first layer we always define the data frame as the first argument. Then, we define the mappings between the variables in the dataset and the **aes**thetics of the plot (e.g. x and y coordinates, colours, etc.).
- In the next layer we represent the data with **geom**etric shapes, in this case with a histogram. You should decide what makes a reasonable bin width for the histogram by trying out a few options.
- In the final layer we facet the data by neighborhood.

4. Use a single pipeline to identity the neighborhoods with the top five median listing prices.

```
#code for  top five median listing prices
medianPrices <- edibnb %>%
  group_by(neighbourhood) %>%
  summarise(median_price = median(price, na.rm = TRUE )) %>%
  arrange(desc(median_price)) %>%
  slice_head(n = 5) #NOTE: do not use head(edibnb) to try and get the top five bec
medianPrices
```

```
## # A tibble: 5 × 2
##   neighbourhood median_price
##   <chr>                <dbl>
## 1 New Town               100
## 2 Old Town                90
## 3 West End                90
## 4 Stockbridge             85
## 5 Bruntsfield             80
```

Then, in another pipeline filter the data for these five neighborhoods and make ridge plots of the distributions of listing prices in these five neighborhoods.

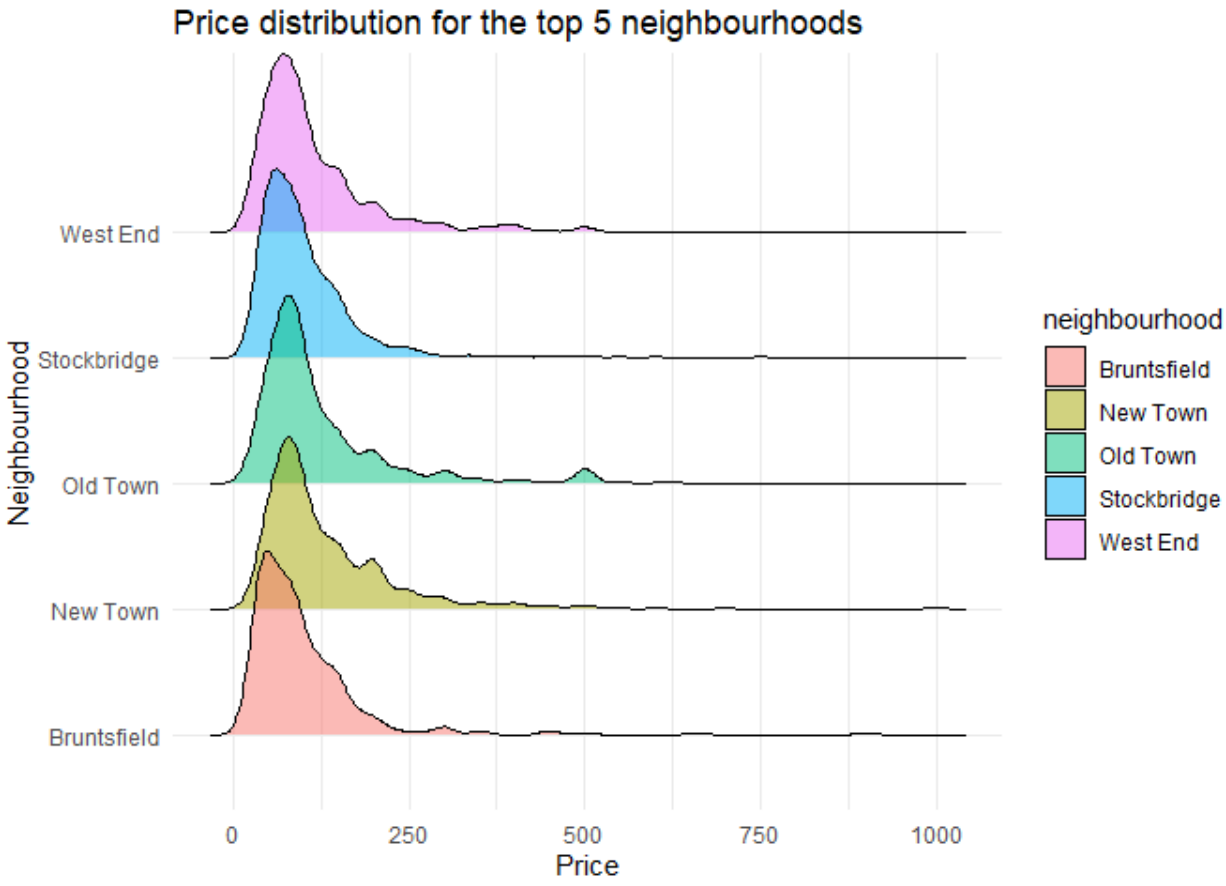
```
# first intall ggridges library
#install.packages("ggridges")
#load ggridges library
library(ggridges)
#filter the data
ridge_Data <- edibnb %>%
  filter(neighbourhood %in% medianPrices$neighbourhood)
# what the %in% does is that it it will only exctract the neighbourhood info that 
ridge_Data # we got a new df name ridge_Data where it only containts info on the t
```

```
## # A tibble: 4,040 × 10
##        id price neighbourhood accommodates bathrooms bedrooms  beds
##     <dbl> <dbl> <chr>                <dbl>     <dbl>    <dbl> <dbl>
##  1  15420    80 New Town                 2         1        1     1
##  2  48645    71 Old Town                 3         1        1     2
##  3  51505   175 New Town                 5         1        2     3
##  4  54188   150 West End                5         1        3     4
##  5  58682   190 West End               10         2        4     7
##  6  67706    85 New Town                 2         1        1     1
##  7 100285   120 New Town                 2         1        1     1
##  8 131342    80 West End                4         1        1     1
##  9 138010    80 Old Town                4         1        1     1
## 10 170756    55 West End                2         2        1     1
## # ℹ 4,030 more rows
## # ℹ 3 more variables: review_scores_rating <dbl>, number_of_reviews <dbl>,
## #   listing_url <chr>
```

```
ggplot(ridge_Data, aes(x = price, y = neighbourhood, fill = neighbourhood)) +
  geom_density_ridges(scale = 1.5,alpha = 0.5) + # scale makes the graphs touch ea
  labs(
    title = "Price distribution for the top 5 neighbourhoods",
    x = "Price",
    y = "Neighbourhood"
  ) +
  theme_minimal()
```

```
## Picking joint bandwidth of 13.8

## Warning: Removed 104 rows containing non-finite outside the scale range
## (`stat_density_ridges()`).
```



In a third pipeline calculate the minimum, mean, median, standard deviation, IQR, and maximum listing price in each of these neighborhoods. Use the visualization and the summary statistics to describe the distribution of listing prices in the neighborhoods. (Your answer will include three pipelines, one of which ends in a visualization, and a narrative.)

```
summarise_TopFiveNeighbourhoods <- edibnb %>%
  filter(neighbourhood %in% ridge_Data$neighbourhood)
summarise_TopFiveNeighbourhoods
```

```
## # A tibble: 4,040 × 10
##        id price neighbourhood accommodates bathrooms bedrooms  beds
##     <dbl> <dbl> <chr>                <dbl>     <dbl>    <dbl> <dbl>
##  1  15420    80 New Town                 2         1        1     1
```

```
##  2  48645    71 Old Town                3         1        1     2
##  3  51505   175 New Town                5         1        2     3
##  4  54188   150 West End                5         1        3     4
##  5  58682   190 West End               10         2        4     7
##  6  67706    85 New Town                2         1        1     1
##  7 100285   120 New Town                2         1        1     1
##  8 131342    80 West End                4         1        1     1
##  9 138010    80 Old Town                4         1        1     1
## 10 170756    55 West End                2         2        1     1
## # ℹ 4,030 more rows
## # ℹ 3 more variables: review_scores_rating <dbl>, number_of_reviews <dbl>,
## #   listing_url <chr>
```

```
# basically ridge_Data dataframe is the same the summarise_TopFiveNeighbourhoods b
summarise_TopFiveNeighbourhoods %>%
  group_by(neighbourhood) %>%
  summarise(
    minPrice = min(price, na.rm = TRUE),
    medianPrice = median(price, na.rm = TRUE),
    meanPrice = mean(price, na.rm = TRUE),
    SD_Price = sd(price, na.rm = TRUE),
    IQR_Price = IQR(price, na.rm =TRUE),
    max_Price = max(price, na.rm = TRUE)


  )
```

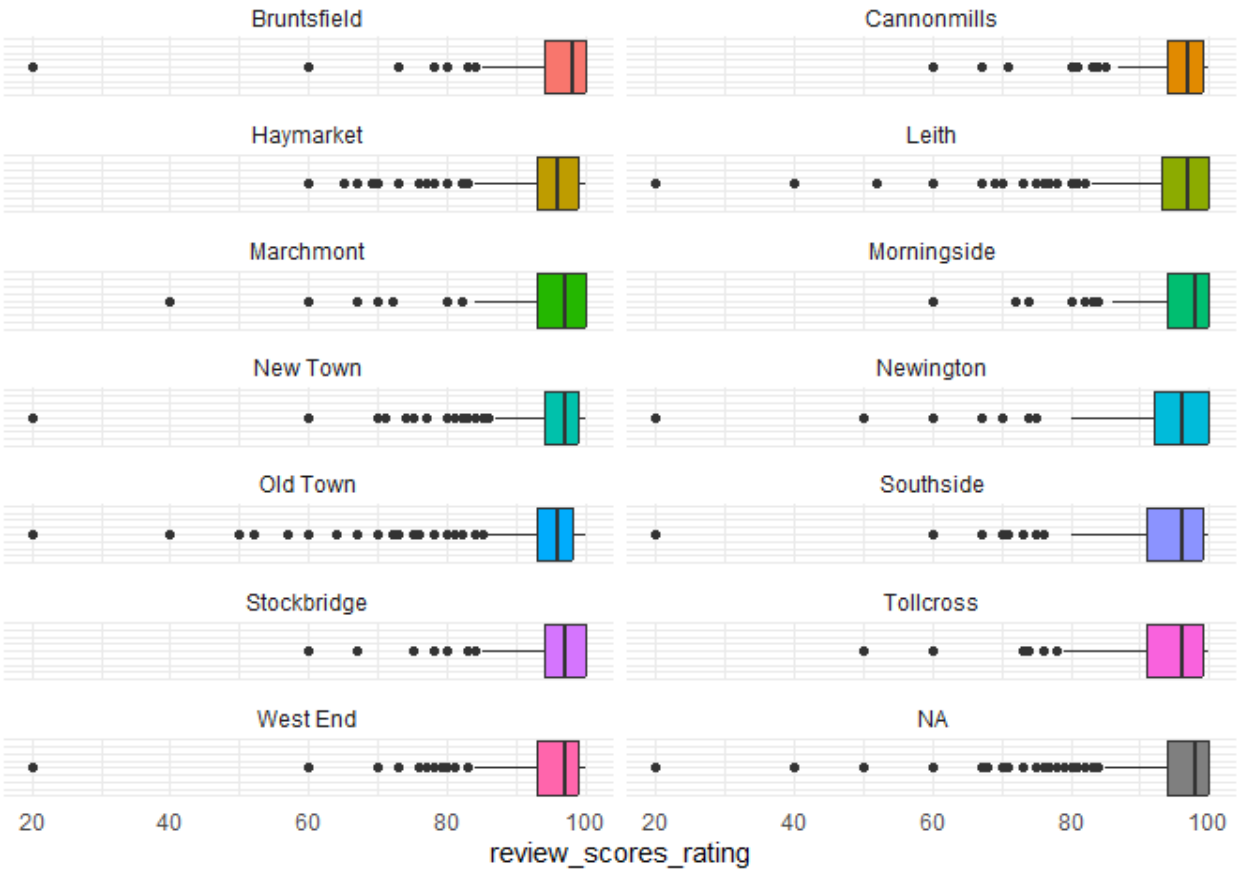
```
## # A tibble: 5 × 7
##   neighbourhood minPrice medianPrice meanPrice SD_Price IQR_Price max_Price
##   <chr>            <dbl>       <dbl>     <dbl>    <dbl>     <dbl>     <dbl>
## 1 Bruntsfield         10          80      99.4     90.2      72.5       900
## 2 New Town            12         100     136.     109.       86.5       999
## 3 Old Town            15          90     128.     110.       76         999
## 4 Stockbridge         21          85     104.      77.6      66         750
## 5 West End            19          90     116.      93.3      80         999
```

5. Create a visualization that will help you compare the distribution of review scores (`review_scores_rating`) across neighborhoods. You get to decide what type of visualization to create and there is more than one correct answer! In your answer, include a brief interpretation of how Airbnb guests rate properties in general and how the neighborhoods compare to each other in terms of their ratings.

```
ggplot(edibnb, aes(x = review_scores_rating, fill = neighbourhood)) +
```

```
  geom_boxplot()+
  facet_wrap(~neighbourhood, nrow = 10, ncol = 2) +
  theme_minimal() +
  theme(
    legend.position = "none",
    axis.text.y = element_blank())
```

```
## Warning: Removed 2177 rows containing non-finite outside the scale range
## (`stat_boxplot()`).
```




```
# i took away the legeng because we have the name of each neighbourhood displayed
# then i also took away the y-axis text because we dont need it
# but the reason i removed both of those features is because it gives the boxplots
```

Airbnb guest usually are biased when rating because they tend to rate higher. Both guest and host can rate eachother so they tend to give a good rate. Meaning that when a costumer leaves a bad review that means something really went wrong. Based on the boxplot, we can say that the distribution of the review score rates are rougly similiar between neighbourhoods, with some neighbourhoods having lower IQR(meaning that

50% of the data is concetrated in a small area) and some have slightly higher IQR. The boxplots are skewed to the left, meaning that a high percentage of the review score rate is high. A boxplot contains 99.3% of the data before it starts identifying outliers, so we can bassed that some neighbourhoods have more outliers points than others, one can take small precautions before booking in those neighbourhoods. The medians seems to be fairly close to eachother. Almost all boxplots have the same spread with mostly having a very low rate of 20 and a high rate close to a 100