Data Visualization Project

Elva

Data Visualization Project

Objectives for this project

To complete this project you'll need to do a few things within this file. As you go through the notebook, you will have further instruction on how to complete these objectives.

1. Go through this notebook, reading along.
2. Fill in empty or incomplete code chunks when prompted to do so.
3. Run each code chunk as you come across it by clicking the tiny green triangles at the right of each chunk. You should see the code chunks print out various output when you do this.
4. At the very top of this file, put your own name in the `author:` place. Currently it says `"DataTrail Team"`. Be sure to put your name in quotes.
5. In the `Conclusions` section, write up responses to each of these questions posed here.
6. When you are satisfied with what you've written and added to this document you'll need to save it. In the menu, go to `File` > `Save`. Now the `nb.html` output resulting file will have your new output saved to it.
7. Open up the resulting `airbnb_project.nb.html` file and click `View in Web Browser`. Does it look good to you? Did all the changes appear here as you expected.
8. Upload your `Rmd` and your `nb.html` files to your assignment folder (this is something that will be dependent on what your instructors have told you – or if you are taking this on your own, just collect these projects in one spot, preferably a Google Drive)!
9. Pat yourself on the back for finishing this project!

The goal of this analysis

<Write here what the goal of this analysis is. What question are we trying to answer?>Clean and tidy data, as well as a visual presentation of data through plots.

Set up

Let's load these packages for use. Add or subtract from this list as you see fit.

Set up directories

Here we are going to make a `data` directory if it doesn't already exist.

```
if (!dir.exists("data")) {
  dir.create("data")
}
```

Get the data

The data we'll be using for this part of the project are about and can be downloaded from here: https://github.com/rfordatascience/tidytuesday/blob/master/data/2023/2023-01-24/readme.md

First, we'll read the data in from our `data/raw` directory. Use the `read_csv` function to do this. Call this new data frame `airbnb_df`.s

```
airbnb_df <- readr::read_csv("data/raw/Airbnb_Open_Data.csv")
```

```
## Warning: One or more parsing issues, call `problems()` on your data frame for details,
## e.g.:
##   dat <- vroom(...)
##   problems(dat)
```

```
## Rows: 102599 Columns: 26
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (13): NAME, host_identity_verified, host name, neighbourhood group, neig...
## dbl (11): id, host id, lat, long, Construction year, minimum nights, number ...
## lgl  (2): instant_bookable, license
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Use this chunk to do some exploring of this dataset to get an idea of what kind of information is here.

```
dim(airbnb_df)
```

```
## [1] 102599     26
```

```
nrow(airbnb_df)
```

```
## [1] 102599
```

```
ncol(airbnb_df)
```

```
## [1] 26
```

```
str(airbnb_df)
```

```
## spc_tbl_ [102,599 x 26] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ id                            : num [1:102599] 1e+06 1e+06 1e+06 1e+06 1e+06 ...
##  $ NAME                          : chr [1:102599] "Clean & quiet apt home by the park" "Skylit Midtou
##  $ host id                       : num [1:102599] 8.00e+10 5.23e+10 7.88e+10 8.51e+10 9.20e+10 ...
##  $ host_identity_verified        : chr [1:102599] "unconfirmed" "verified" NA "unconfirmed" ...
##  $ host name                     : chr [1:102599] "Madaline" "Jenna" "Elise" "Garry" ...
##  $ neighbourhood group           : chr [1:102599] "Brooklyn" "Manhattan" "Manhattan" "Brooklyn" ...
##  $ neighbourhood                 : chr [1:102599] "Kensington" "Midtown" "Harlem" "Clinton Hill" ...
##  $ lat                           : num [1:102599] 40.6 40.8 40.8 40.7 40.8 ...
##  $ long                          : num [1:102599] -74 -74 -73.9 -74 -73.9 ...
##  $ country                       : chr [1:102599] "United States" "United States" "United States" "Un
##  $ country code                  : chr [1:102599] "US" "US" "US" "US" ...
##  $ instant_bookable              : logi [1:102599] FALSE FALSE TRUE TRUE FALSE TRUE ...
##  $ cancellation_policy           : chr [1:102599] "strict" "moderate" "flexible" "moderate" ...
##  $ room type                     : chr [1:102599] "Private room" "Entire home/apt" "Private room" "En
##  $ Construction year             : num [1:102599] 2020 2007 2005 2005 2009 ...
##  $ price                         : chr [1:102599] "$966" "$142" "$620" "$368" ...
##  $ service fee                   : chr [1:102599] "$193" "$28" "$124" "$74" ...
##  $ minimum nights                : num [1:102599] 10 30 3 30 10 3 45 45 2 2 ...
```

```
##  $ number of reviews             : num [1:102599] 9 45 0 270 9 74 49 49 430 118 ...
##  $ last review                   : chr [1:102599] "10/19/2021" "5/21/2022" NA "7/5/2019" ...
##  $ reviews per month             : num [1:102599] 0.21 0.38 NA 4.64 0.1 0.59 0.4 0.4 3.47 0.99 ...
##  $ review rate number            : num [1:102599] 4 4 5 4 3 3 5 5 3 5 ...
##  $ calculated host listings count: num [1:102599] 6 2 1 1 1 1 1 1 1 1 ...
##  $ availability 365              : num [1:102599] 286 228 352 322 289 374 224 219 180 375 ...
##  $ house_rules                   : chr [1:102599] "Clean up and treat the home the way you'd like you
##  $ license                       : logi [1:102599] NA NA NA NA NA NA ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   id = col_double(),
##   ..   NAME = col_character(),
##   ..   `host id` = col_double(),
##   ..   host_identity_verified = col_character(),
##   ..   `host name` = col_character(),
##   ..   `neighbourhood group` = col_character(),
##   ..   neighbourhood = col_character(),
##   ..   lat = col_double(),
##   ..   long = col_double(),
##   ..   country = col_character(),
##   ..   `country code` = col_character(),
##   ..   instant_bookable = col_logical(),
##   ..   cancellation_policy = col_character(),
##   ..   `room type` = col_character(),
##   ..   `Construction year` = col_double(),
##   ..   price = col_character(),
##   ..   `service fee` = col_character(),
##   ..   `minimum nights` = col_double(),
##   ..   `number of reviews` = col_double(),
##   ..   `last review` = col_character(),
##   ..   `reviews per month` = col_double(),
##   ..   `review rate number` = col_double(),
##   ..   `calculated host listings count` = col_double(),
##   ..   `availability 365` = col_double(),
##   ..   house_rules = col_character(),
##   ..   license = col_logical()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

```
summary(airbnb_df)
```

```
##        id               NAME              host id
##  Min.   : 1001254   Length:102599      Min.   :1.236e+08
##  1st Qu.:15085814   Class :character   1st Qu.:2.458e+10
##  Median :29136603   Mode  :character   Median :4.912e+10
##  Mean   :29146235                      Mean   :4.925e+10
##  3rd Qu.:43201198                      3rd Qu.:7.400e+10
##  Max.   :57367417                      Max.   :9.876e+10
##
##  host_identity_verified  host name         neighbourhood group
##  Length:102599          Length:102599      Length:102599
##  Class :character       Class :character   Class :character
##  Mode  :character       Mode  :character   Mode  :character
##
##
```

```
##
##
##  neighbourhood           lat             long          country
##  Length:102599      Min.   :40.50   Min.   :-74.25   Length:102599
##  Class :character   1st Qu.:40.69   1st Qu.:-73.98   Class :character
##  Mode  :character   Median :40.72   Median :-73.95   Mode  :character
##                     Mean   :40.73   Mean   :-73.95
##                     3rd Qu.:40.76   3rd Qu.:-73.93
##                     Max.   :40.92   Max.   :-73.71
##                     NA's   :8       NA's   :8
##  country code       instant_bookable cancellation_policy  room type
##  Length:102599      Mode :logical    Length:102599       Length:102599
##  Class :character   FALSE:51474      Class :character    Class :character
##  Mode  :character   TRUE :51020      Mode  :character    Mode  :character
##                     NA's :105
##
##
##
##  Construction year    price           service fee        minimum nights
##  Min.   :2003      Length:102599      Length:102599      Min.   :-1223.000
##  1st Qu.:2007      Class :character   Class :character   1st Qu.:    2.000
##  Median :2012      Mode  :character   Mode  :character   Median :    3.000
##  Mean   :2012                                            Mean   :    8.136
##  3rd Qu.:2017                                            3rd Qu.:    5.000
##  Max.   :2022                                            Max.   : 5645.000
##  NA's   :214                                             NA's   :409
##  number of reviews last review        reviews per month review rate number
##  Min.   :   0.00   Length:102599      Min.   : 0.010    Min.   :1.000
##  1st Qu.:   1.00   Class :character   1st Qu.: 0.220    1st Qu.:2.000
##  Median :   7.00   Mode  :character   Median : 0.740    Median :3.000
##  Mean   :  27.48                      Mean   : 1.374    Mean   :3.279
##  3rd Qu.:  30.00                      3rd Qu.: 2.000    3rd Qu.:4.000
##  Max.   :1024.00                      Max.   :90.000    Max.   :5.000
##  NA's   :183                          NA's   :15879     NA's   :326
##  calculated host listings count availability 365 house_rules
##  Min.   :  1.000                Min.   : -10.0   Length:102599
##  1st Qu.:  1.000                1st Qu.:   3.0   Class :character
##  Median :  1.000                Median :  96.0   Mode  :character
##  Mean   :  7.937                Mean   : 141.1
##  3rd Qu.:  2.000                3rd Qu.: 269.0
##  Max.   :332.000                Max.   :3677.0
##  NA's   :319                    NA's   :448
##  license
##  Mode:logical
##  NA's:102599
##
##
##
##
##
```

```
max(airbnb_df$price, na.rm=T)
```

```
## [1] "$999"
```

```
unique(airbnb_df$'room type')
```

```
## [1] "Private room"    "Entire home/apt" "Shared room"     "Hotel room"
```

```
length(unique(airbnb_df$'room type'))
```

```
## [1] 4
```

Clean the data

Use the `janitor::clean_names` function to clean up some of these column names.

```
### Your code here
colnames(airbnb_df)
```

```
##  [1] "id"                             "NAME"
##  [3] "host id"                        "host_identity_verified"
##  [5] "host name"                      "neighbourhood group"
##  [7] "neighbourhood"                  "lat"
##  [9] "long"                           "country"
## [11] "country code"                   "instant_bookable"
## [13] "cancellation_policy"            "room type"
## [15] "Construction year"              "price"
## [17] "service fee"                    "minimum nights"
## [19] "number of reviews"              "last review"
## [21] "reviews per month"              "review rate number"
## [23] "calculated host listings count" "availability 365"
## [25] "house_rules"                    "license"
```

```
airbnb_df <-airbnb_df %>%
 clean_names()
```

Fix `neighbourhood_group` variable

Take a look at the `neighbourhood_group` variable. Try to summarize how many different neighborhood groups there are using the `summary()` function.

```
### summary(neighborhood_group)
summary(unique(airbnb_df$neighbourhood_group))
```

```
##    Length     Class      Mode
##         8 character character
```

You will find that this variable is a character, so we will need to check it as a factor to get meaningful information about the groups here. Thus run the `summary` function on a factor version of the `neighborhood_group` data, but don't yet change the variable to be a factor.

```
### Your code here
airbnb_df$neighbourhood_group <- as.factor(airbnb_df$neighbourhood_group)
summary(airbnb_df$neighbourhood_group)
```

```
##         Bronx       brookln      Brooklyn      manhatan     Manhattan
##          2712             1         41842             1         43792
##        Queens Staten Island          NA's
##         13267           955            29
```

#You'll notice there's some typos here `brookln` and `manhatan`. Fix these typos in this dataset by whatever method you find most intuitive. After you have fixed the typos, make sure that you coerce `neighbourhood_group` as a factor.

```
### Your code here

airbnb_df$neighbourhood_group[airbnb_df$neighbourhood_group == 'brookln'] <- 'Brooklyn'

airbnb_df$neighbourhood_group[airbnb_df$neighbourhood_group == 'manhatan'] <- 'Manhattan'

summary(unique(airbnb_df$neighbourhood_group))
```

```
##         Bronx       brookln      Brooklyn      manhatan     Manhattan
##             1             0             1             0             1
##        Queens Staten Island          NA's
##             1             1             1
```

Re-run `summary()` on the `neighbourhood_group` to see if your code successfully fixed these misspellings.

```
### Your code here
summary(airbnb_df$neighbourhood_group)
```

```
##         Bronx       brookln      Brooklyn      manhatan     Manhattan
##          2712             0         41843             0         43793
##        Queens Staten Island          NA's
##         13267           955            29
```

Fix numeric data

The `service_fee` and `price` columns are really numeric data but are being treated as characters because there are `$` and `,` included.

We'll need to get rid of these. **Hint** use this kind of code: `stringr::str_remove(service_fee, "\\$|,")` within a mutate to get rid of the dollar signs. After you get rid of the dollar signs from both `service_fee` and `price`, coerce both of these variables to be numeric.

```
### Your code here
#airbnb_df <- airbnb_df %>%
 # mutate(service_fee = as.numeric(stringr::str_remove(service_fee, #"\\$|,")),  price = as.numeric(str

# another way
airbnb_df <- airbnb_df %>%
 mutate(service_fee=stringr::str_remove(service_fee, "\\$|,"))

airbnb_df <- airbnb_df %>%
  mutate(price=stringr::str_remove(price, "\\$|,"))
```

Use summary to see if `price` and `service_fee` are now appropriately numeric datsa after removing dollor sign and comma. before character however after the as.numeric() function change to numeric. using class() function to verify to tell

```
### Your code here

class(airbnb_df$service_fee)
```

```
## [1] "character"
```

```
airbnb_df$service_fee <- as.numeric(airbnb_df$service_fee)

class(airbnb_df$price)
```

```
## [1] "character"
```

```
airbnb_df$price<- as.numeric(airbnb_df$price)
```

```
## Warning: NAs introduced by coercion
```

```
### Your code here the results in box plot min 1st qu,Median, mean, 3rd QU, max
summary(airbnb_df$service_fee)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##      10      68     125     125     183     240     273
```

```
summary(airbnb_df$price)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##    50.0   288.0   524.0   524.7   759.0   999.0   18150
```

Plot the data

Let's make a series of plots that explore how the price of an airbnb may be related to the other variables in this set.

To refresh your memory, a basic ggplot code looks like this:DATA_FRAME %>% ggplot(aes(VARIABLE, VARIABLE, color = OPTIONAL_VARIABLE)) + geom_PLOT_TYPE()

We recommend keeping the ggplot2 cheatsheet handy while you are going through this project: https://www.rstudio.com/resources/cheatsheets/

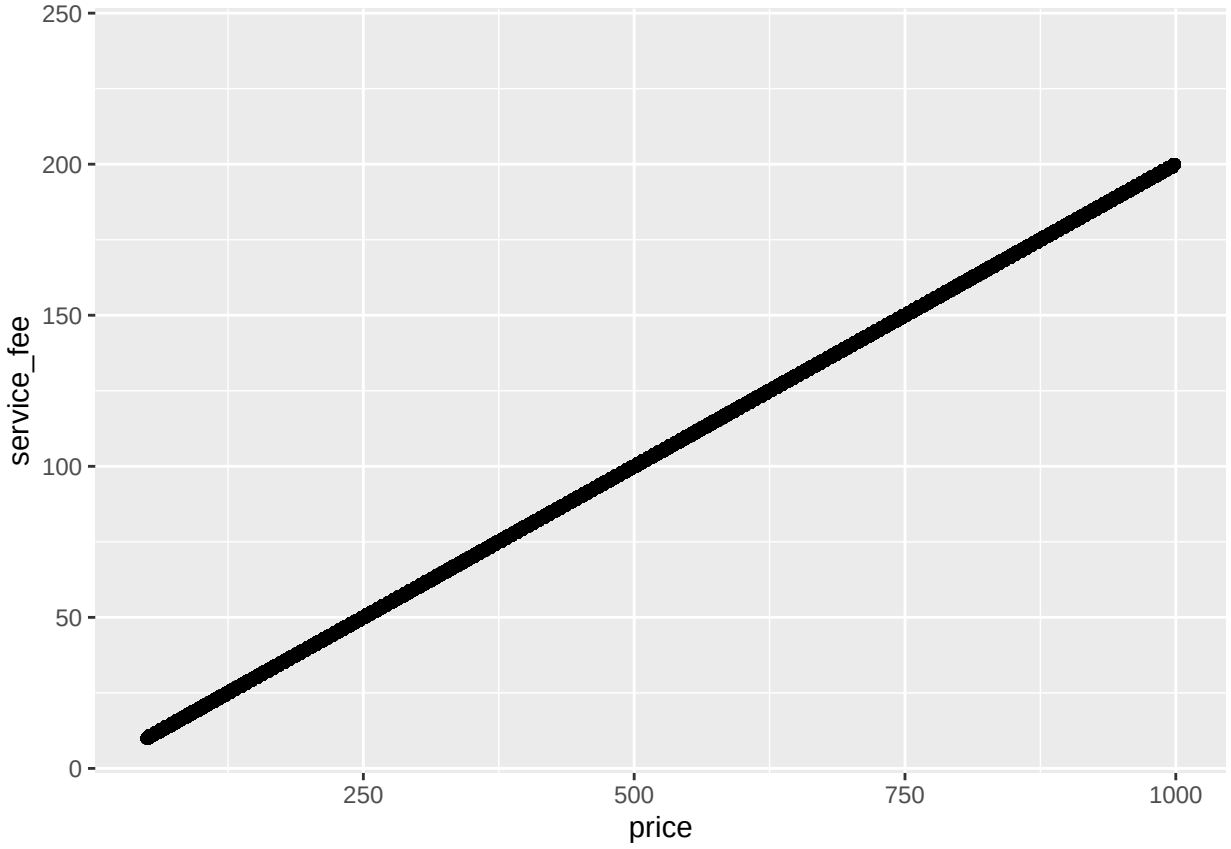
Scatter plot

Now make a scatter plot that explores if `price` is related to one of the other numeric variables in the dataset ie service fee, note linear price increase

```
### Your code here ggplot(data = DATASET) +
  ## geom_PLOT_TYPE(mapping = aes(VARIABLE(S)))

ggplot(data = airbnb_df) +
 geom_point(mapping = aes(x=price, y=service_fee))
```

```
## Warning: Removed 18336 rows containing missing values (`geom_point()`).
```



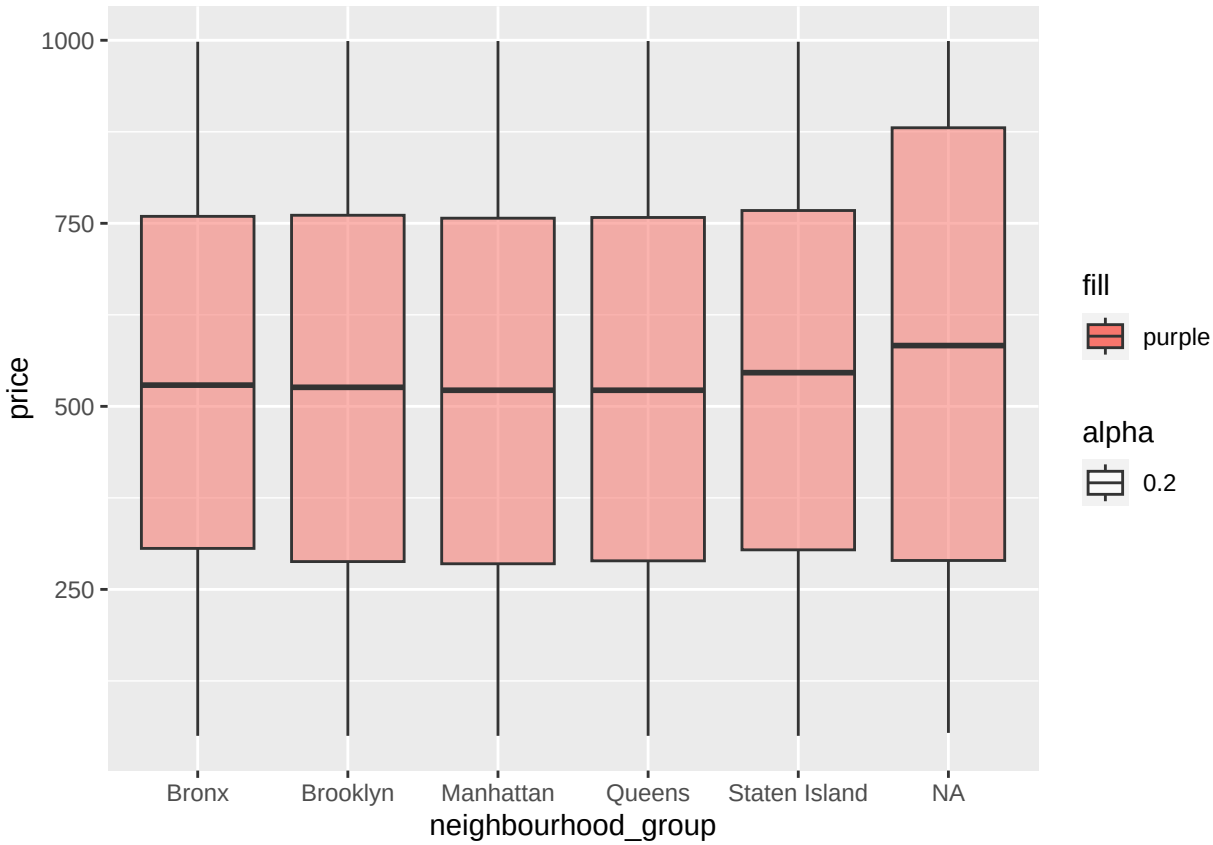
Box plot

Make a box plot for the price distribution of different neighborhood groups.(5 elements min value,1st quartile q1 25%,median q2,3rd quartile q3 75%, max each %25)

```
### Your code here

ggplot(data = airbnb_df) +
  geom_boxplot(aes(y = price, x = neighbourhood_group, fill =
        "purple", alpha = 0.2))
```

```
## Warning: Removed 18150 rows containing non-finite values (`stat_boxplot()`).
```




```
# Another really basic boxplot i.e.
# ggplot(mtcars, aes(x=as.factor(cyl), y=mpg)) +
    #geom_boxplot(fill="slateblue", alpha=0.2) +
   # xlab("cyl")

#ggplot(data = airbnb_df) +
# geom_boxplot(aes(y = price, x = neighbourhood_group),
 # fill = "slateblue", alpha = 0.2) +
 #   xlab("neighbourhood_groups")
```

Barplot

Now make a plot that explores if there's a particular `room_type` that has a higher price than others. You may choose to summarize the `price` by the `room_type`. Hint: you may need to `group_by` a variable.

```
### Your code here
?barplot

#airbnb_df %>%
  #group_by(room_type) %>%
  #summarize(price)

# Use summarize using the mean() for specifics, also tell R to ignore the NA's
airbnb_df %>%
   group_by(room_type) %>%
   summarize(price=mean(price, na.rm = T))
```

```
## # A tibble: 4 x 2
```

```
##   room_type       price
##   <chr>           <dbl>
## 1 Entire home/apt  525.
## 2 Hotel room       575.
## 3 Private room     524.
## 4 Shared room      525.
```

```
 #barplot(airbnb_df ~ room_type, data = airbnb_df())

# barplot(cbind(x=room_type, y=price)
```

Your choice plot!

Make one more plot that is completely your choice about what variables you'd like to see the relationships of.

See http://r-statistics.co/Top50-Ggplot2-Visualizations-MasterList-R-Code.html for inspiration and ideas.

```
### Your code here

#ggplot(data = airbnb_df) +
 #geom_area(mapping = aes(x=price, y=room_type))

#airbnb_df %>%
#ggplot(data = airbnb_df) +
  #geom_area(aes(y = price, x = room_type, fill =
# "blue", alpha = 0.4))
# xlab("NYC room_type")
```

Conclusion

Write up your thoughts about this data science project here and answer the following questions:

- What did we find out about our questions?First the function to clean up some of these column names, from typos, splitting data, renaming columns with the many variable. saw the opportunity to summarize how many different groups
- How did we explore our questions? In addition to cleaning the data, we explored different functions data as character vs numeric. we saw it was very general whereas when we entered it as a factor, the data was very specific, waying in on more details. We dived into the plots which allowed us to see our data visually.
- What did our explorations show us? We got to see how factor gives meaningful information about the groups, a notable difference. Distinguishing between numeric data vs characters. We further got to explore series of plots, exploring the different visuals references
- What follow up data science questions arise for you regarding this candy dataset now that we've explored it some? Further in-depth understanding and practice to grasp it fully

Print out session info

Session info is a good thing to print out at the end of your notebooks so that you (and other folks) referencing your notebooks know what software versions and libraries you used to run the notebook.

```
sessionInfo()
```

```
## R version 4.3.3 (2024-02-29)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 20.04.6 LTS
##
```

```
## Matrix products: default
## BLAS:   /usr/lib/x86_64-linux-gnu/atlas/libblas.so.3.10.3
## LAPACK: /usr/lib/x86_64-linux-gnu/atlas/liblapack.so.3.10.3;  LAPACK version 3.9.0
##
## locale:
##  [1] LC_CTYPE=C.UTF-8       LC_NUMERIC=C           LC_TIME=C.UTF-8
##  [4] LC_COLLATE=C.UTF-8     LC_MONETARY=C.UTF-8    LC_MESSAGES=C.UTF-8
##  [7] LC_PAPER=C.UTF-8       LC_NAME=C              LC_ADDRESS=C
## [10] LC_TELEPHONE=C         LC_MEASUREMENT=C.UTF-8 LC_IDENTIFICATION=C
##
## time zone: UTC
## tzcode source: system (glibc)
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] readr_2.1.4    ggthemes_4.2.4 dplyr_1.1.2    ggplot2_3.4.2  janitor_2.2.0
##
## loaded via a namespace (and not attached):
##  [1] utf8_1.2.3        generics_0.1.3    stringi_1.7.12    hms_1.1.3
##  [5] digest_0.6.33     magrittr_2.0.3    evaluate_0.21     grid_4.3.3
##  [9] timechange_0.2.0  fastmap_1.1.1     purrr_1.0.1       fansi_1.0.4
## [13] scales_1.2.1      cli_3.6.1         rlang_1.1.1       crayon_1.5.2
## [17] bit64_4.0.5       munsell_0.5.0     withr_2.5.0       yaml_2.3.7
## [21] tools_4.3.3       parallel_4.3.3    tzdb_0.4.0        colorspace_2.1-0
## [25] vctrs_0.6.3       R6_2.5.1          lifecycle_1.0.3   lubridate_1.9.2
## [29] snakecase_0.11.0  stringr_1.5.1     bit_4.0.5         vroom_1.6.3
## [33] pkgconfig_2.0.3   pillar_1.9.0      gtable_0.3.3      glue_1.6.2
## [37] xfun_0.39         tibble_3.2.1      tidyselect_1.2.0  highr_0.10
## [41] rstudioapi_0.15.0 knitr_1.43        farver_2.1.1      htmltools_0.5.5
## [45] rmarkdown_2.23    labeling_0.4.2    compiler_4.3.3
```