Las Vegas Airbnb Data Process and Analysis

Marcos

Setting up my environment

```
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✓ dplyr     1.1.4     ✓ readr     2.1.5
## ✓ forcats   1.0.0     ✓ stringr   1.5.1
## ✓ ggplot2   3.5.2     ✓ tibble    3.2.1
## ✓ lubridate 1.9.4     ✓ tidyr     1.3.1
## ✓ purrr     1.0.4
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becom
e errors
```

```
library(dplyr)
library(ggplot2)
library(lubridate)
library(scales)
```

```
##
## Attaching package: 'scales'
##
## The following object is masked from 'package:purrr':
##
##     discard
##
## The following object is masked from 'package:readr':
##
##     col_factor
```

Importing the date into R

```
# re-naming the data
LV_Airbnb<-read.csv("LV_listings.csv")
LV_Calendar<-read.csv("LV_calendar.csv.gz")
```

Exploring the data from LV_Airbnb and LV_Calendar

```
glimpse(LV_Airbnb)
```

```
## Rows: 15,396
## Columns: 18
## $ id                              <dbl> 44701, 113019, 114140, 133084, 143096, …
## $ name                            <chr> "Jan 4-11,2025 CES:  Clean, Classy and …
## $ host_id                         <int> 189245, 575684, 575684, 653641, 694506,…
## $ host_name                       <chr> "Christine", "LasVegasSuites", "LasVega…
## $ neighbourhood_group             <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,…
## $ neighbourhood                   <chr> "Unincorporated Areas", "Unincorporated…
## $ latitude                        <dbl> 36.11689, 36.10905, 36.10736, 36.16085,…
## $ longitude                       <dbl> -115.1626, -115.1664, -115.1659, -115.1…
## $ room_type                       <chr> "Entire home/apt", "Entire home/apt", "…
## $ price                           <int> 280, 118, 148, 121, NA, 80, 150, 114, 2…
## $ minimum_nights                  <int> 7, 2, 2, 30, 28, 30, 2, 1, 7, 2, 3, 3, …
## $ number_of_reviews               <int> 4, 200, 153, 2, 243, 36, 345, 81, 0, 21…
## $ last_review                     <chr> "2024-01-14", "2024-04-15", "2024-03-03…
## $ reviews_per_month               <dbl> 0.04, 1.28, 1.03, 0.01, 1.51, 0.28, 2.2…
## $ calculated_host_listings_count <int> 1, 11, 11, 1, 1, 2, 1, 1, 1, 6, 2, 1, 1…
## $ availability_365                <int> 164, 62, 63, 281, 277, 180, 271, 128, 1…
## $ number_of_reviews_ltm           <int> 1, 10, 2, 0, 4, 0, 40, 23, 0, 1, 4, 38,…
## $ license                         <chr> "", "", "", "", "", "", "", "", "", "",…
```

```
glimpse(LV_Calendar)
```

```
## Rows: 5,619,375
## Columns: 7
## $ listing_id     <dbl> 44701, 44701, 44701, 44701, 44701, 44701, 44701, 44701,…
## $ date           <chr> "2024-09-19", "2024-09-20", "2024-09-21", "2024-09-22",…
## $ available      <chr> "t", "t", "t", "f", "f", "f", "f", "f", "f", "f", "f", …
## $ price          <chr> "$280.00", "$280.00", "$280.00", "$280.00", "$280.00", …
## $ adjusted_price <chr> "", "", "", "", "", "", "", "", "", "", "", "", "", "",…
## $ minimum_nights <int> 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7…
## $ maximum_nights <int> 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7…
```

Data Cleaning

Exploring the data in LV_Airbnb we can remove sensitive and any irrelevant information we do not need for this analysis. We are also removing price from LV_Airbnb because we already have price in LV_Calendar which are data specific

```
RM_LV_listing<- LV_Airbnb[, !(names(LV_Airbnb) %in%
                                c("host_name","neighbourhood_group",
                                  "number_of_reviews", "last_review",
                                  "reviews_per_month", "number_of_reviews_ltm",
                                  "license", "price", "name", "minimum_nights"))]

# self check to make sure those column where removed from RM_LV_listings
glimpse(RM_LV_listing)
```

```
## Rows: 15,396
## Columns: 8
## $ id                            <dbl> 44701, 113019, 114140, 133084, 143096, …
## $ host_id                       <int> 189245, 575684, 575684, 653641, 694506,…
## $ neighbourhood                 <chr> "Unincorporated Areas", "Unincorporated…
## $ latitude                      <dbl> 36.11689, 36.10905, 36.10736, 36.16085,…
## $ longitude                     <dbl> -115.1626, -115.1664, -115.1659, -115.1…
## $ room_type                     <chr> "Entire home/apt", "Entire home/apt", "…
## $ calculated_host_listings_count <int> 1, 11, 11, 1, 1, 2, 1, 1, 1, 6, 2, 1, 1…
## $ availability_365              <int> 164, 62, 63, 281, 277, 180, 271, 128, 1…
```

```
# now check for any empty data Within RM_LV_listing and LV_Calendar
colSums(is.na(RM_LV_listing))
```

```
##                              id                         host_id
##                               0                               0
##                   neighbourhood                        latitude
##                               0                               0
##                       longitude                       room_type
##                               0                               0
## calculated_host_listings_count                availability_365
##                               0                               0
```

```
# since we only have 3 missing data we leave it
colSums(is.na(LV_Calendar))
```

```
##     listing_id           date      available          price adjusted_price
##              0              0              0              0              0
## minimum_nights maximum_nights
##              3              3
```

```
# Next we move on to the LV_Calendar dataset which contains the majority of the
# data required for our analysis
str(LV_Calendar$date)
```

```
##  chr [1:5619375] "2024-09-19" "2024-09-20" "2024-09-21" "2024-09-22" ...
```

```
# reformat date
LV_Calendar$date<- as.Date(LV_Calendar$date)

# reformat price gsub function removes $ and ,
LV_Calendar$price<- as.numeric(gsub("[$,]", "", LV_Calendar$price))

# check that conversion was made
class(LV_Calendar$price)
```

```
## [1] "numeric"
```

```
class(LV_Calendar$date)
```

```
## [1] "Date"
```

```
# find earliest and latest date we see the earliest date is Sept 2024
# and the lastest is Sept 2025
min(LV_Calendar$date)
```

```
## [1] "2024-09-18"
```

```
max(LV_Calendar$date)
```

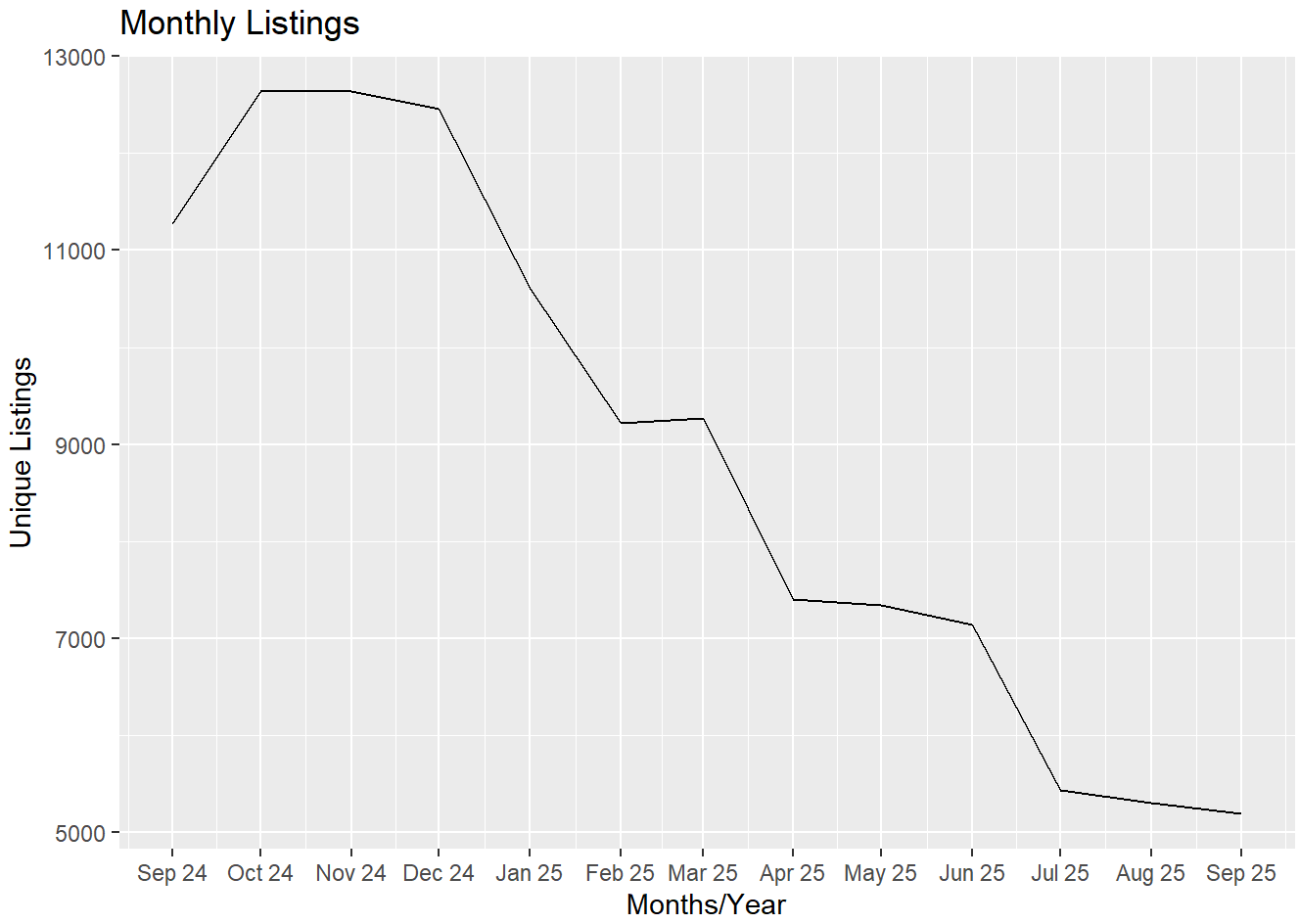
```
## [1] "2025-09-18"
```

```
# now we merge both dataset into one.
merged_data<- merge(LV_Calendar, RM_LV_listing, by.x = "listing_id",
                    by.y = "id", all.x = TRUE)


#check that the data was merged into on
glimpse(merged_data)
```

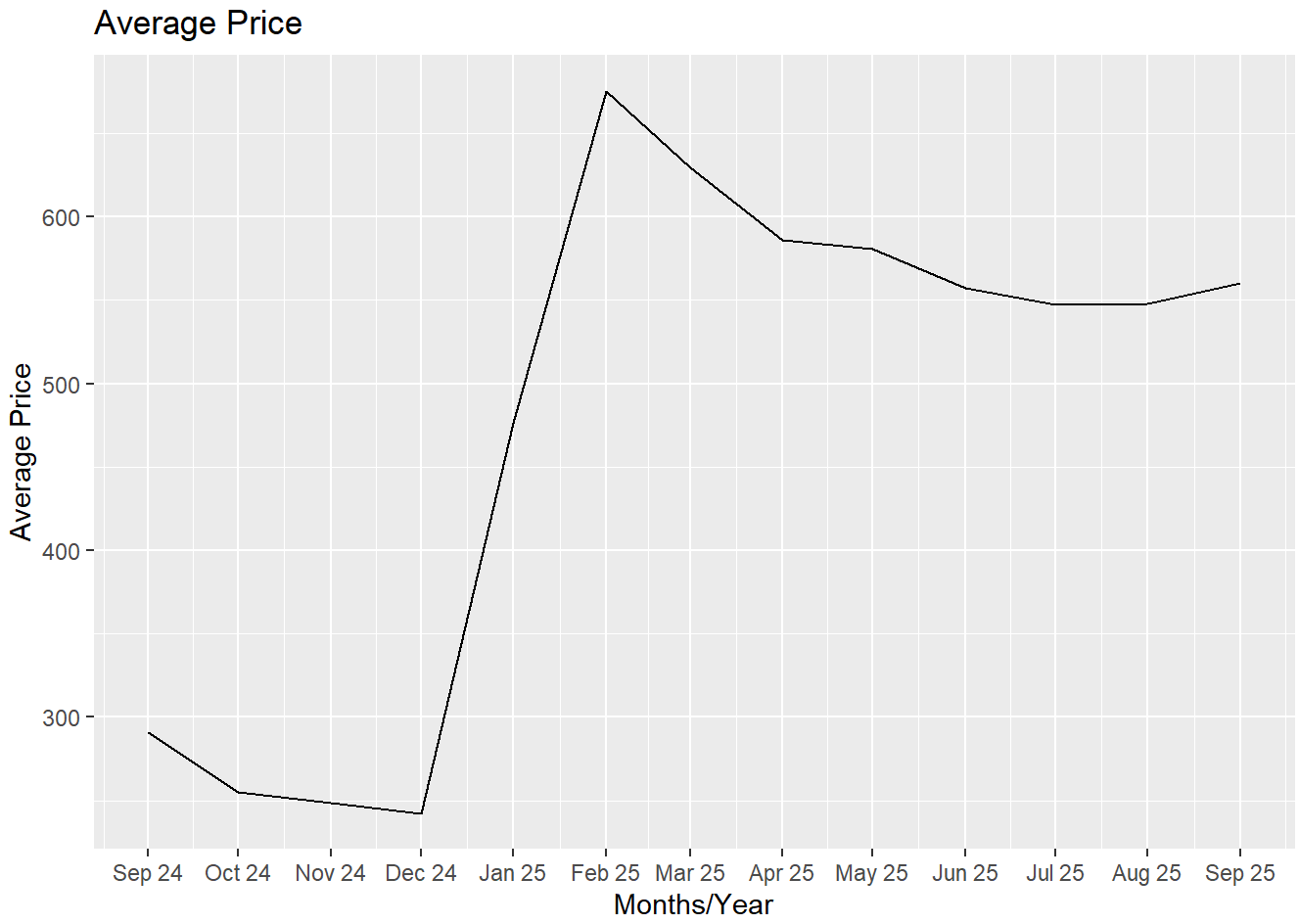
```
## Rows: 5,619,375
## Columns: 14
## $ listing_id                    <dbl> 44701, 44701, 44701, 44701, 44701, 4470…
## $ date                          <date> 2024-09-19, 2024-09-20, 2024-09-21, 20…
## $ available                     <chr> "t", "t", "t", "f", "f", "f", "f", "f",…
## $ price                         <dbl> 280, 280, 280, 280, 280, 280, 280, 280,…
## $ adjusted_price                <chr> "", "", "", "", "", "", "", "", "", "",…
## $ minimum_nights                <int> 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, …
## $ maximum_nights                <int> 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, …
## $ host_id                       <int> 189245, 189245, 189245, 189245, 189245,…
## $ neighbourhood                 <chr> "Unincorporated Areas", "Unincorporated…
## $ latitude                      <dbl> 36.11689, 36.11689, 36.11689, 36.11689,…
## $ longitude                     <dbl> -115.1626, -115.1626, -115.1626, -115.1…
## $ room_type                     <chr> "Entire home/apt", "Entire home/apt", "…
## $ calculated_host_listings_count <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, …
## $ availability_365              <int> 164, 164, 164, 164, 164, 164, 164, 164,…
```

Analyzing and Sharing

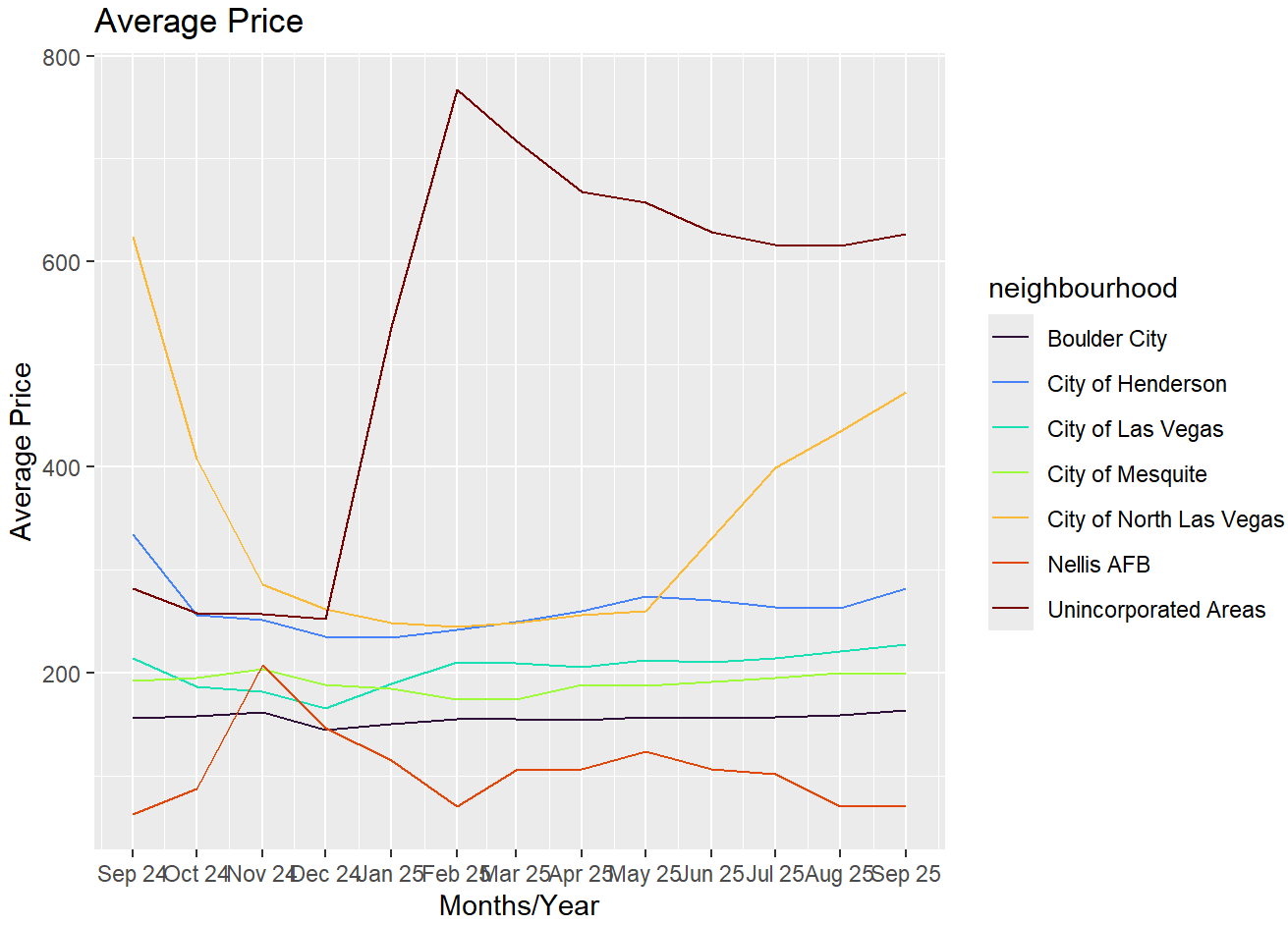
```
# here we are counting how many Airbnb are available = TRUE in each month
monthly_counts <- merged_data %>%
  mutate(month = floor_date(date, "month")) %>%
  filter(available == "t") %>%
  group_by(month) %>%
  summarise(unique_listings = n_distinct(listing_id))


# Looking at the line graph we have here we can see, during the winter months there
# are a lot of available Airbnb. We see the number of rooms available stars
# to decline during the summer
ggplot(data = monthly_counts) +
  geom_line(mapping = aes(x= month, y = unique_listings)) +
  # scale force the ggplot to print the months from  Sept 2024 to Sept 2025
  scale_x_date(
    breaks = seq(as.Date("2024-09-01"), as.Date("2025-09-01"), by = "1 month"),
    # this part will print out Month as Month / year
    labels = date_format("%b %y") # used to format how dates appear in x-axis
  ) +
  labs(title = "Monthly Listings", x = "Months/Year", y = "Unique Listings")
```




```
# we check the average price of each Airbnb that has been booked. Any Airbnb that is
# available we do not count. This shows us a Demand trend
ave_price <- merged_data %>%
  mutate(month = floor_date(date, "month")) %>%
 filter(available == "f") %>%
  group_by(month) %>%
  summarise(average_price = mean( price, na.rm = TRUE))



# As we see in the line graph the average price of Airbnb skyrockets to the moon
# during Dec 2024 and Jan 2025
ggplot( data =  ave_price) +
  geom_line( mapping = aes( x = month, y = average_price)) +
  scale_x_date(
    breaks = seq(as.Date("2024-09-01"), as.Date("2025-09-01"), by = "1 month"),
    labels = date_format("%b %y")
  ) +
  labs(title = "Average Price", x = "Months/Year", y = "Average Price")
```




```
# Know we check the average price of each Neighborhood in Las Vegas
ave_neighber_prices <- merged_data %>%
  mutate(month = floor_date(date, "month")) %>%
  #using false to count only those Airbnb that have been booked
  filter(available == "f") %>%
  group_by(month, neighbourhood) %>%
  #this .group tell summarize we do not want this data to be grouped anymore
  # and we only need the clean date frame. If we do not do this it leads to an error
  summarise(average_price = mean(price, na.rm = TRUE), .groups = "drop")


ggplot( data =ave_neighber_prices) +
  geom_line( mapping = aes(x = month, y = average_price, color = neighbourhood)) +
  #this line of code makes the color more vibrant
  scale_color_viridis_d( option = "turbo") +
  scale_x_date(
    breaks = seq(as.Date("2024-09-01"), as.Date("2025-09-01"), by = "1 month"),
    labels = date_format("%b %y")
  ) +
  labs(title = "Average Price", x = "Months/Year", y = "Average Price")
```



When analyzing the average price across neighborhoods in Las Vegas, we can find that the most expensive areas to stay in are located within unincorporated regions. These neighborhoods are Paradise, Spring Valley, Enterprise, Winchester and Whitney.

Extracting the data for futher analysis

```
# using the write function export a csv file to use in Tableau
write.csv(merged_data, "LV_Airbnb_Analysis.csv", row.names = FALSE)
```