Project

Trinath Sai Subhash Reddy Pittala, Uma Maheswara R Meleti, Hemanth Vasireddy

2023-04-04

Introduction

Airbnb Price Determinants in Europe

We want to work on Airbnb's dataset from kaggle.com. It provides information about hotel rooms in Europe.

Each major city has its dataset for weekends and weekdays Variables included in the dataset: Host ID (Id) The total price of listing (realSum) Room type: private, shared, entire home, apt (room_type) Whether or not a room is shared (room_shared) Max number of people allowed in property (person_capacity) Whether or not the host is superhost (host_is_superhost) Whether or not it is multiple rooms (multi) Whether for business or family use (biz) Distance from the city center (dist) Distance from nearest metro (metro_dist) Latitude and longitude (lat long) Guest satisfaction (guest_satisfaction_overall) Cleanliness (cleanliness_rating) The total quantity of bedrooms available among all properties for a single host (bedrooms)

Questions we can answer with the dataset: Price Forecasting: use pricing, room type, and amenities to predict potential rental prices in the future. Hotspots: use listing location in relation to business and tourism centers and correlate this with pricing to determine where Airbnb rentals would be most profitable Customer Sentiment Analysis: analyze customer comments and satisfaction ratings to evaluate listing on overall customer experience and use it to optimize hosts' services to improve user satisfaction ratings.

How can this information be used: Data can help travelers find accommodation that meets their needs without exceeding budget. Can help hosts set competitive pricing and optimize listings to get more bookings. Help investors evaluate the value of investing in real estate in different European cities based on pricing trends.

Pre Processing and Cleaning the Data

Data loading

```
# Set the relative directory path
my_dir <- "./archive"

# List all the files in the directory
files <- list.files(path = my_dir, full.names = TRUE)
```

Combining the Data from all Files

```r
# Get a list of all the csv files in the directory
file_list <- list.files(path = my_dir, pattern = "*.csv", full.names = TRUE)

# Initialize an empty list to store the data frames
df_list <- list()

# Loop through each file and read it into a data frame
for (i in seq_along(file_list)) {
    df <- read.csv(file_list[i])

    # Add a new column with the city_day
    df$city_day <- basename(file_list[i])

    # Append the data frame to the list
    df_list[[i]] <- df
}

# Combine all the data frames into a single dataset
my_data <- bind_rows(df_list)

# Removing the .csv ext
my_data$city_day <- gsub("\\.csv", "", my_data$city_day)

# Print the first few rows of the data
head(my_data)
```

```
##   X  realSum    room_type room_shared room_private person_capacity
## 1 0 194.0337 Private room       False         True               2
## 2 1 344.2458 Private room       False         True               4
## 3 2 264.1014 Private room       False         True               2
## 4 3 433.5294 Private room       False         True               4
## 5 4 485.5529 Private room       False         True               2
## 6 5 552.8086 Private room       False         True               3
##   host_is_superhost multi biz cleanliness_rating guest_satisfaction_overall
## 1             False     1   0                 10                         93
## 2             False     0   0                  8                         85
## 3             False     0   1                  9                         87
## 4             False     0   1                  9                         90
## 5              True     0   0                 10                         98
## 6             False     0   0                  8                        100
##   bedrooms      dist metro_dist attr_index attr_index_norm rest_index
## 1        1 5.0229638  2.5393800   78.69038        4.166708   98.25390
## 2        1 0.4883893  0.2394039  631.17638       33.421209  837.28076
## 3        1 5.7483119  3.6516213   75.27588        3.985908   95.38695
## 4        2 0.3848620  0.4398761  493.27253       26.119108  875.03310
## 5        1 0.5447382  0.3186926  552.83032       29.272733  815.30574
## 6        2 2.1314201  1.9046682  174.78896        9.255191  225.20166
##   rest_index_norm     lng      lat           city_day
## 1        6.846473 4.90569 52.41772 amsterdam_weekdays
## 2       58.342928 4.90005 52.37432 amsterdam_weekdays
## 3        6.646700 4.97512 52.36103 amsterdam_weekdays
## 4       60.973565 4.89417 52.37663 amsterdam_weekdays
## 5       56.811677 4.90051 52.37508 amsterdam_weekdays
```

```
## 6       15.692376 4.87699 52.38966 amsterdam_weekdays
```

```r
print(unique(my_data[my_data$room_shared == my_data$room_private,
    ]$room_type))  # if the room is shared
```

```
## [1] "Entire home/apt"
```

```r
print(unique(my_data[my_data$room_private == "False", ]$room_type))
```

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

```r
print(unique(my_data[my_data$room_shared == "True", ]$room_type))
```

```
## [1] "Shared room"
```

```r
print(unique(my_data[my_data$room_shared == "False", ]$room_type))
```

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

The room_shared and room_private information is already embedded in room_type. The variables are multi-collinear, so we can remove room_shared and room_private.

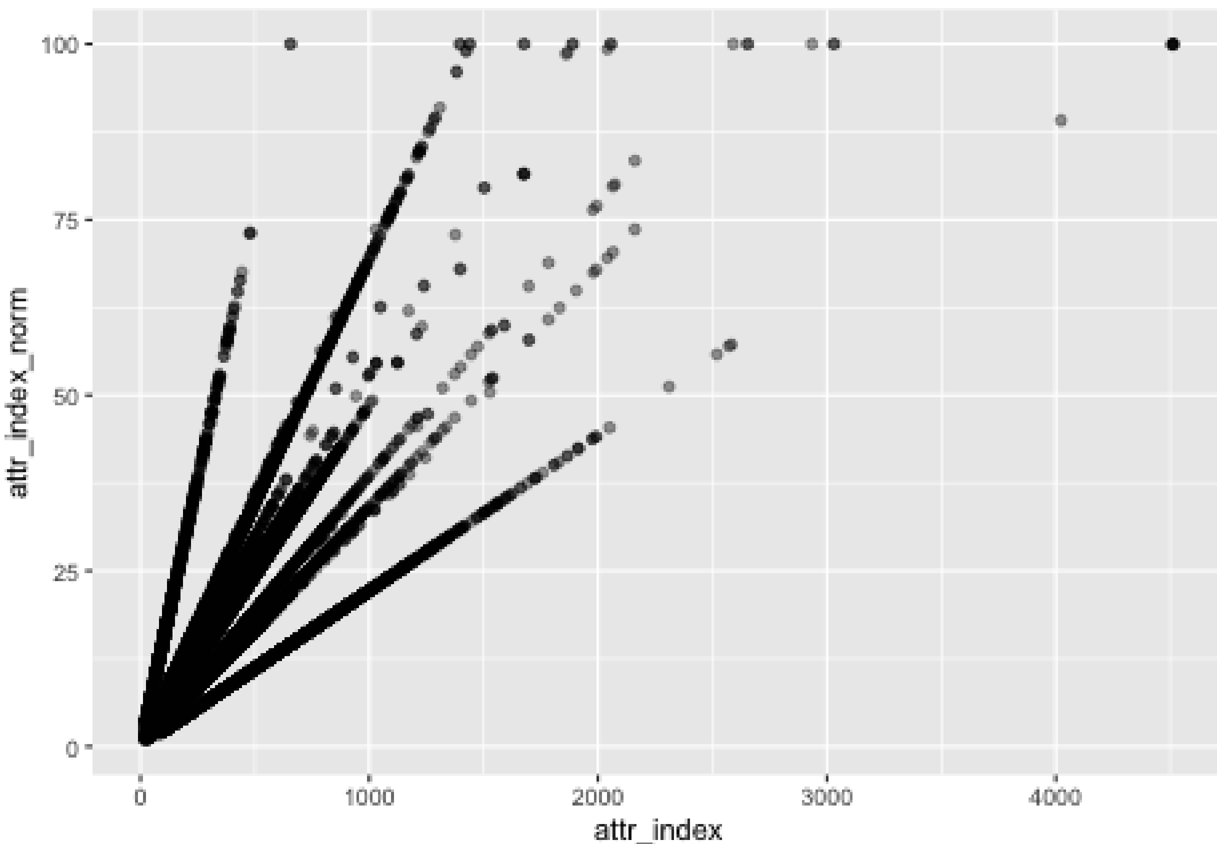
Dropping columns of room_shared and room_private

```r
my_data = select(my_data, -c(room_shared, room_private))
head(my_data)
```

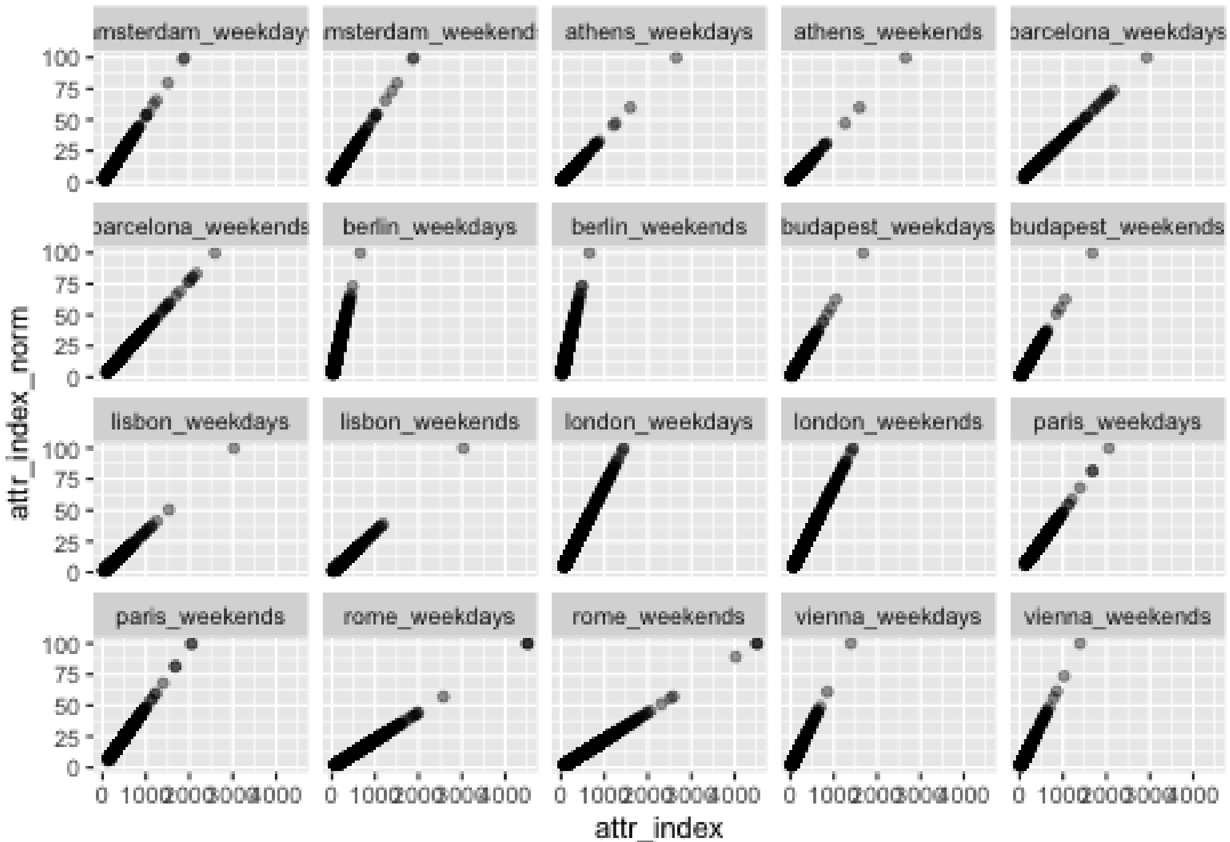
```
##   X  realSum    room_type person_capacity host_is_superhost multi biz
## 1 0 194.0337 Private room               2             False     1   0
## 2 1 344.2458 Private room               4             False     0   0
## 3 2 264.1014 Private room               2             False     0   1
## 4 3 433.5294 Private room               4             False     0   1
## 5 4 485.5529 Private room               2              True     0   0
## 6 5 552.8086 Private room               3             False     0   0
##   cleanliness_rating guest_satisfaction_overall bedrooms      dist metro_dist
## 1                 10                         93        1 5.0229638  2.5393800
## 2                  8                         85        1 0.4883893  0.2394039
## 3                  9                         87        1 5.7483119  3.6516213
## 4                  9                         90        2 0.3848620  0.4398761
## 5                 10                         98        1 0.5447382  0.3186926
## 6                  8                        100        2 2.1314201  1.9046682
##   attr_index attr_index_norm rest_index rest_index_norm     lng      lat
## 1   78.69038        4.166708   98.25390        6.846473 4.90569 52.41772
## 2  631.17638       33.421209  837.28076       58.342928 4.90005 52.37432
## 3   75.27588        3.985908   95.38695        6.646700 4.97512 52.36103
## 4  493.27253       26.119108  875.03310       60.973565 4.89417 52.37663
## 5  552.83032       29.272733  815.30574       56.811677 4.90051 52.37508
## 6  174.78896        9.255191  225.20166       15.692376 4.87699 52.38966
```

```
##              city_day
## 1 amsterdam_weekdays
## 2 amsterdam_weekdays
## 3 amsterdam_weekdays
## 4 amsterdam_weekdays
## 5 amsterdam_weekdays
## 6 amsterdam_weekdays
```

```r
ggplot() + geom_point(data = my_data, aes(x = attr_index, y = attr_index_norm),
    alpha = 0.4)
```

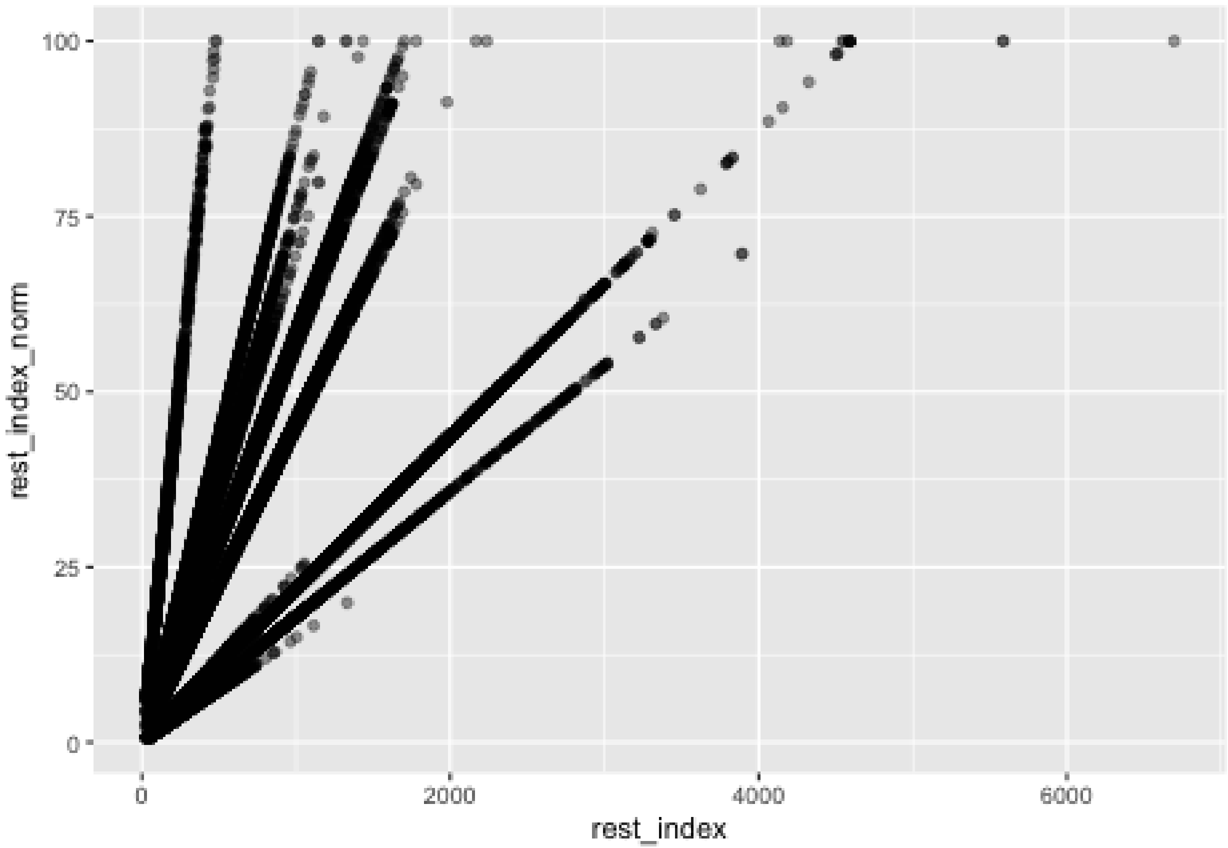



```r
ggplot() + geom_point(data = my_data, aes(x = attr_index, y = attr_index_norm),
    alpha = 0.4) + facet_wrap(~city_day)
```

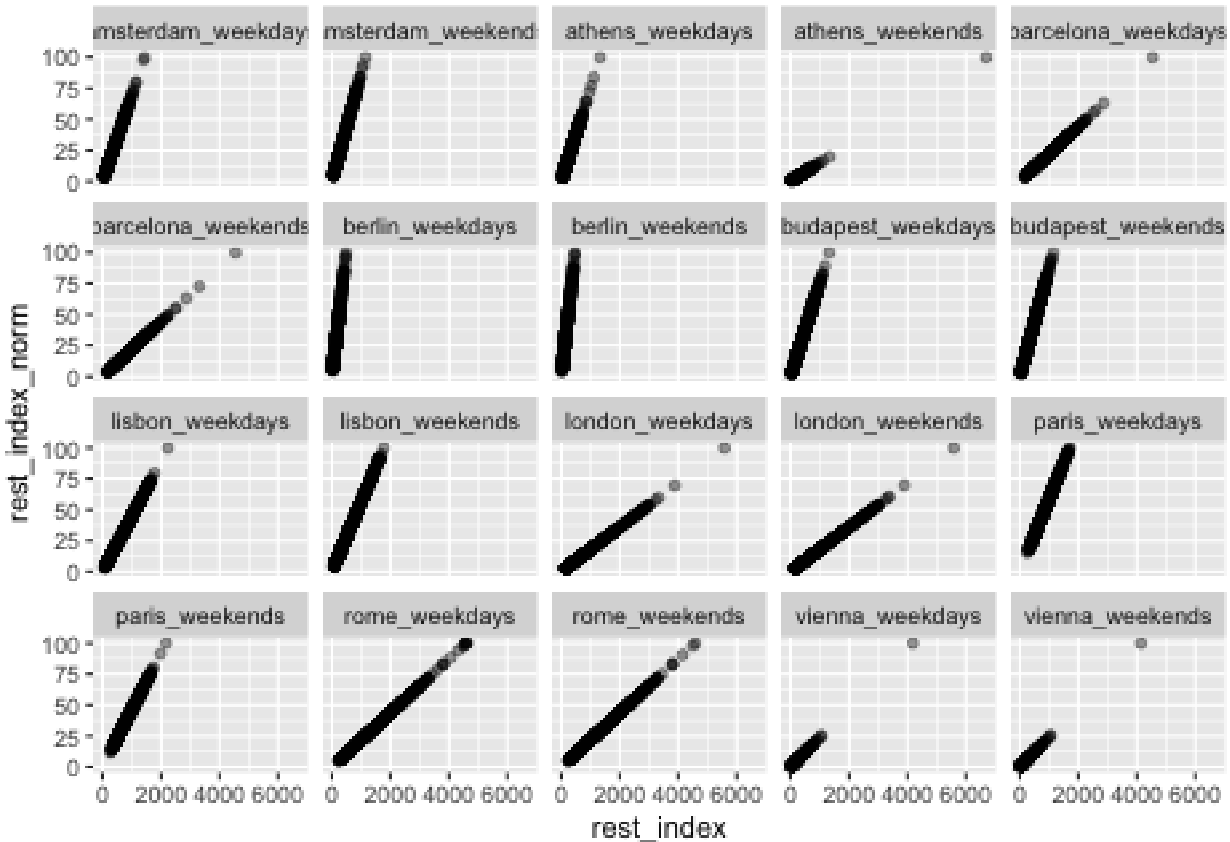


attr_index and attr_index_norm are same, attr_index_norm is just normalized attr_index

```
ggplot() + geom_point(data = my_data, aes(x = rest_index, y = rest_index_norm),
    alpha = 0.4)
```




```r
ggplot() + geom_point(data = my_data, aes(x = rest_index, y = rest_index_norm),
    alpha = 0.4) + facet_wrap(~city_day)
```



rest_index and rest_index_norm are same, rest_index_norm is just normalized rest_index.

removing attr_index and rest_index

```
my_data = select(my_data, -c(attr_index, rest_index))
head(my_data)
```

```
##   X  realSum    room_type person_capacity host_is_superhost multi biz
## 1 0 194.0337 Private room               2             False     1   0
## 2 1 344.2458 Private room               4             False     0   0
## 3 2 264.1014 Private room               2             False     0   1
## 4 3 433.5294 Private room               4             False     0   1
## 5 4 485.5529 Private room               2              True     0   0
## 6 5 552.8086 Private room               3             False     0   0
##   cleanliness_rating guest_satisfaction_overall bedrooms      dist metro_dist
## 1                 10                         93        1 5.0229638  2.5393800
## 2                  8                         85        1 0.4883893  0.2394039
## 3                  9                         87        1 5.7483119  3.6516213
## 4                  9                         90        2 0.3848620  0.4398761
## 5                 10                         98        1 0.5447382  0.3186926
## 6                  8                        100        2 2.1314201  1.9046682
##   attr_index_norm rest_index_norm     lng      lat           city_day
## 1        4.166708        6.846473 4.90569 52.41772 amsterdam_weekdays
## 2       33.421209       58.342928 4.90005 52.37432 amsterdam_weekdays
## 3        3.985908        6.646700 4.97512 52.36103 amsterdam_weekdays
## 4       26.119108       60.973565 4.89417 52.37663 amsterdam_weekdays
## 5       29.272733       56.811677 4.90051 52.37508 amsterdam_weekdays
## 6        9.255191       15.692376 4.87699 52.38966 amsterdam_weekdays
```

Outliers using IQR Range

Filtering out the Outliers from Data Out of IQR Ranges

```
# Initialize an empty list to store the outliers
outliers_list <- list()

# Initialize an empty list to store the filtered data
# frames
df_list_filtered <- list()

# Loop through each file and read it into a data frame
# after removing outliers
for (i in seq_along(file_list)) {
    df_filtered <- read.csv(file_list[i])

    # Add a new column with the city_day
    df_filtered$city_day <- gsub("\\.csv", "", basename(file_list[i]))

    iqr_var1 <- IQR(df_filtered$realSum)

    # Calculate the upper and lower bounds for each
    # variable
    upper_var1 <- quantile(df_filtered$realSum, 0.75) + 1.5 *
        iqr_var1
    lower_var1 <- quantile(df_filtered$realSum, 0.25) - 1.5 *
        iqr_var1

    # Filter the data based on the upper and lower bounds
    # for each variable
    filtered_data <- filter(df_filtered, realSum > lower_var1 &
        realSum < upper_var1)

    # Append the filtered data frame to the list
    df_list_filtered[[i]] <- filtered_data

    # Get the rows that were removed while filtering
    outliers <- anti_join(df_filtered, filtered_data)

    # Append the outliers to the list
    outliers_list[[i]] <- outliers
}


# Combine all the filtered data frames into a single
# dataset
my_data_filtered <- bind_rows(df_list_filtered)

# Removing the .csv ext
my_data_filtered$city_day <- gsub("\\.csv", "", my_data_filtered$city_day)

summary(my_data_filtered)
```

```
##        X           realSum          room_type          room_shared
##  Min.   :   0   Min.   :  34.78   Length:48970       Length:48970
##  1st Qu.: 645   1st Qu.: 145.23   Class :character   Class :character
##  Median :1340   Median : 204.27   Mode  :character   Mode  :character
##  Mean   :1621   Mean   : 244.35
##  3rd Qu.:2385   3rd Qu.: 295.27
##  Max.   :5378   Max.   :1229.11
##  room_private       person_capacity host_is_superhost      multi
##  Length:48970       Min.   :2.00    Length:48970       Min.   :0.0000
##  Class :character   1st Qu.:2.00    Class :character   1st Qu.:0.0000
##  Mode  :character   Median :3.00    Mode  :character   Median :0.0000
##                     Mean   :3.08                       Mean   :0.2953
##                     3rd Qu.:4.00                       3rd Qu.:1.0000
##                     Max.   :6.00                       Max.   :1.0000
##       biz        cleanliness_rating guest_satisfaction_overall    bedrooms
##  Min.   :0.000   Min.   : 2.000     Min.   : 20.00             Min.   : 0.000
##  1st Qu.:0.000   1st Qu.: 9.000     1st Qu.: 90.00             1st Qu.: 1.000
##  Median :0.000   Median :10.000     Median : 95.00             Median : 1.000
##  Mean   :0.342   Mean   : 9.384     Mean   : 92.57             Mean   : 1.118
##  3rd Qu.:1.000   3rd Qu.:10.000     3rd Qu.: 98.00             3rd Qu.: 1.000
##  Max.   :1.000   Max.   :10.000     Max.   :100.00             Max.   :10.000
##       dist             metro_dist          attr_index      attr_index_norm
##  Min.   : 0.01506   Min.   : 0.002301   Min.   :  15.15   Min.   :  0.9263
##  1st Qu.: 1.48598   1st Qu.: 0.250718   1st Qu.: 133.75   1st Qu.:  6.2341
##  Median : 2.66962   Median : 0.416955   Median : 228.54   Median : 11.1929
##  Mean   : 3.24072   Mean   : 0.691774   Mean   : 285.15   Mean   : 13.0064
##  3rd Qu.: 4.31533   3rd Qu.: 0.749700   3rd Qu.: 374.37   3rd Qu.: 16.9444
##  Max.   :25.28456   Max.   :14.273577   Max.   :4513.56   Max.   :100.0000
##    rest_index      rest_index_norm         lng                lat
##  Min.   :  19.58   Min.   :  0.5928   Min.   :-9.22634   Min.   :37.95
##  1st Qu.: 245.42   1st Qu.:  8.5601   1st Qu.:-0.07277   1st Qu.:41.40
##  Median : 512.42   Median : 17.1799   Median : 4.87234   Median :47.51
##  Mean   : 611.32   Mean   : 22.2861   Mean   : 7.40027   Mean   :45.66
##  3rd Qu.: 818.44   3rd Qu.: 32.0321   3rd Qu.:13.52350   3rd Qu.:51.47
##  Max.   :6696.16   Max.   :100.0000   Max.   :23.78602   Max.   :52.64
##    city_day
##  Length:48970
##  Class :character
##  Mode  :character
##
##
##
```

```r
# Combine all the outliers into a single dataset
my_outliers <- bind_rows(outliers_list)

# Removing the .csv ext
my_outliers$city_day <- gsub("\\.csv", "", my_outliers$city_day)

summary(my_outliers)
```

```
##        X           realSum          room_type          room_shared
##  Min.   :   0   Min.   :  279.4   Length:2737        Length:2737
##  1st Qu.: 666   1st Qu.:  469.2   Class :character   Class :character
```

```
##  Median :1237   Median :  691.9   Mode  :character   Mode  :character
##  Mean   :1614   Mean   :  915.5
##  3rd Qu.:2310   3rd Qu.:  996.3
##  Max.   :5374   Max.   :18545.5
##  room_private       person_capacity host_is_superhost      multi
##  Length:2737        Min.   :2.000   Length:2737        Min.   :0.000
##  Class :character   1st Qu.:4.000   Class :character   1st Qu.:0.000
##  Mode  :character   Median :5.000   Mode  :character   Median :0.000
##                     Mean   :4.628                      Mean   :0.221
##                     3rd Qu.:6.000                      3rd Qu.:0.000
##                     Max.   :6.000                      Max.   :1.000
##       biz         cleanliness_rating guest_satisfaction_overall    bedrooms
##  Min.   :0.0000   Min.   : 2.000     Min.   : 20.00             Min.   :0.000
##  1st Qu.:0.0000   1st Qu.: 9.000     1st Qu.: 91.00             1st Qu.:1.000
##  Median :0.0000   Median :10.000     Median : 97.00             Median :2.000
##  Mean   :0.4965   Mean   : 9.509     Mean   : 93.65             Mean   :1.886
##  3rd Qu.:1.0000   3rd Qu.:10.000     3rd Qu.:100.00             3rd Qu.:2.000
##  Max.   :1.0000   Max.   :10.000     Max.   :100.00             Max.   :6.000
##       dist            metro_dist         attr_index     attr_index_norm
##  Min.   : 0.01504   Min.   :0.006171   Min.   :  20.5   Min.   :  1.468
##  1st Qu.: 1.04119   1st Qu.:0.218081   1st Qu.: 225.1   1st Qu.: 11.719
##  Median : 1.89579   Median :0.352339   Median : 385.0   Median : 17.958
##  Mean   : 2.30674   Mean   :0.498426   Mean   : 456.2   Mean   : 20.892
##  3rd Qu.: 3.00820   3rd Qu.:0.576430   3rd Qu.: 610.6   3rd Qu.: 25.953
##  Max.   :21.29515   Max.   :8.918036   Max.   :2040.4   Max.   :100.000
##    rest_index     rest_index_norm        lng                lat
##  Min.   :  27.9   Min.   :  0.667   Min.   :-9.22476   Min.   :37.96
##  1st Qu.: 408.5   1st Qu.: 14.187   1st Qu.:-0.06677   1st Qu.:41.41
##  Median : 739.9   Median : 30.001   Median : 4.88384   Median :47.51
##  Mean   : 904.9   Mean   : 31.734   Mean   : 7.88764   Mean   :45.93
##  3rd Qu.:1269.7   3rd Qu.: 45.426   3rd Qu.:13.44666   3rd Qu.:51.50
##  Max.   :4183.1   Max.   :100.000   Max.   :23.75400   Max.   :52.58
##    city_day
##  Length:2737
##  Class :character
##  Mode  :character
##
##
##
```

Percentage of Outliers outside of IQR range.

```
# Create empty table
outliers_table <- data.frame(City_day = character(), Data_Length = numeric(),
    Percent_Outliers = numeric(), stringsAsFactors = FALSE)

# Loop through city_data and fill in table
for (city_day in unique(my_data$city_day)) {
    x = my_data[my_data$city_day == city_day, ]$realSum
    q1 <- quantile(x, 0.25)
    q3 <- quantile(x, 0.75)
    iqr <- IQR(x)
```

```
    upper_bound <- q3 + 1.5 * iqr
    lower_bound <- q1 - 1.5 * iqr
    x_no_outliers <- x[x >= lower_bound & x <= upper_bound]
    percent_outliers <- ((length(x) - length(x_no_outliers))/length(x)) *
        100

    # Add row to table
    outliers_table <- rbind(outliers_table, data.frame(City_day = city_day,
        Data_Length = length(x), Percent_Outliers = percent_outliers))
}

# Format table using kable
kable(outliers_table, format = "markdown")
```

City_day	Data_Length	Percent_Outliers
amsterdam_weekdays	1103	5.077063
amsterdam_weekends	977	5.629478
athens_weekdays	2653	5.767056
athens_weekends	2627	5.405405
barcelona_weekdays	1555	7.524116
barcelona_weekends	1278	8.059468
berlin_weekdays	1284	6.308411
berlin_weekends	1200	6.166667
budapest_weekdays	2074	5.930569
budapest_weekends	1948	5.544148
lisbon_weekdays	2857	3.360168
lisbon_weekends	2906	3.475568
london_weekdays	4614	5.353273
london_weekends	5379	5.521472
paris_weekdays	3130	6.134185
paris_weekends	3558	5.368184
rome_weekdays	4492	5.031167
rome_weekends	4535	5.005513
vienna_weekdays	1738	4.257767
vienna_weekends	1799	4.113396

Spilt Training and Testing Data

```
set.seed(123456789)
my_data_train <- my_data[sample(nrow(my_data), 0.7 * nrow(my_data)),
    ]
my_data_test <- my_data[setdiff(1:nrow(my_data), rownames(my_data_train)),
    ]
dim(my_data_train)
```

```
## [1] 36194    17
```

```
dim(my_data_test)
```

```
## [1] 15513    17
```

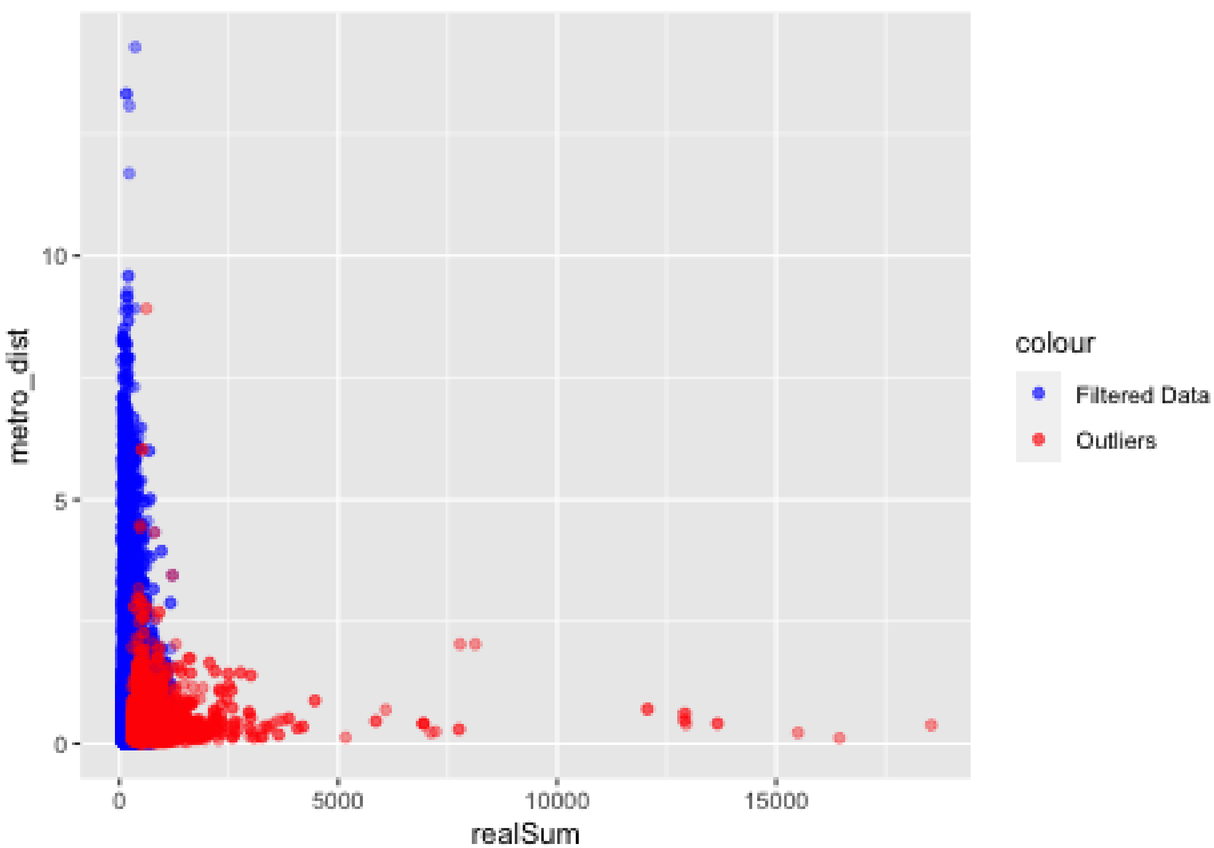
Exploratory Data Analysis

Outlier Analysis

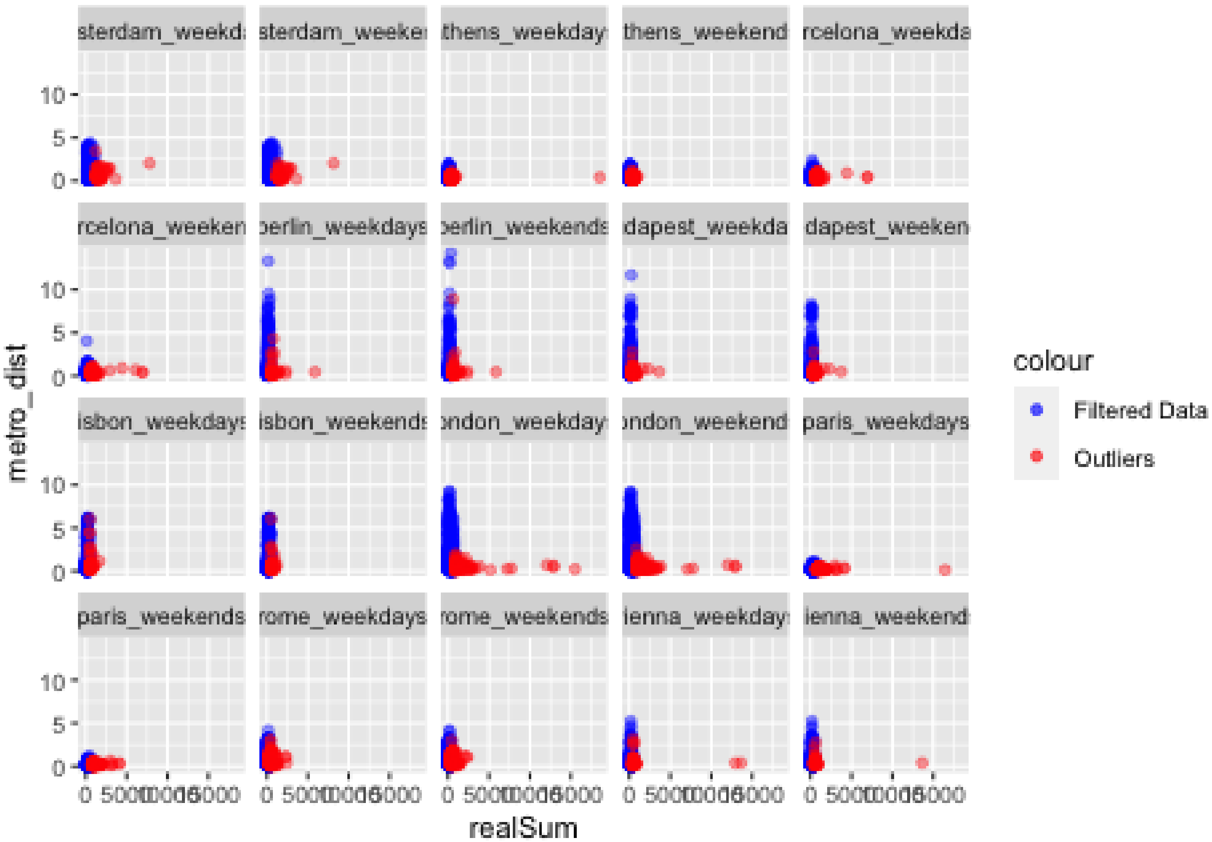
Metro Dist vs Real Sum

We have planned to analyse the filtered data along with outlier data. Here outlier data represents the hotel rooms with high prices.

```
ggplot() + geom_point(data = my_data_filtered, aes(x = realSum,
    y = metro_dist, color = "Filtered Data"), alpha = 0.4) +
    geom_point(data = my_outliers, aes(x = realSum, y = metro_dist,
        color = "Outliers"), alpha = 0.4) + scale_color_manual(values = c(`Filtered Data` = "blue",
    Outliers = "red"))
```



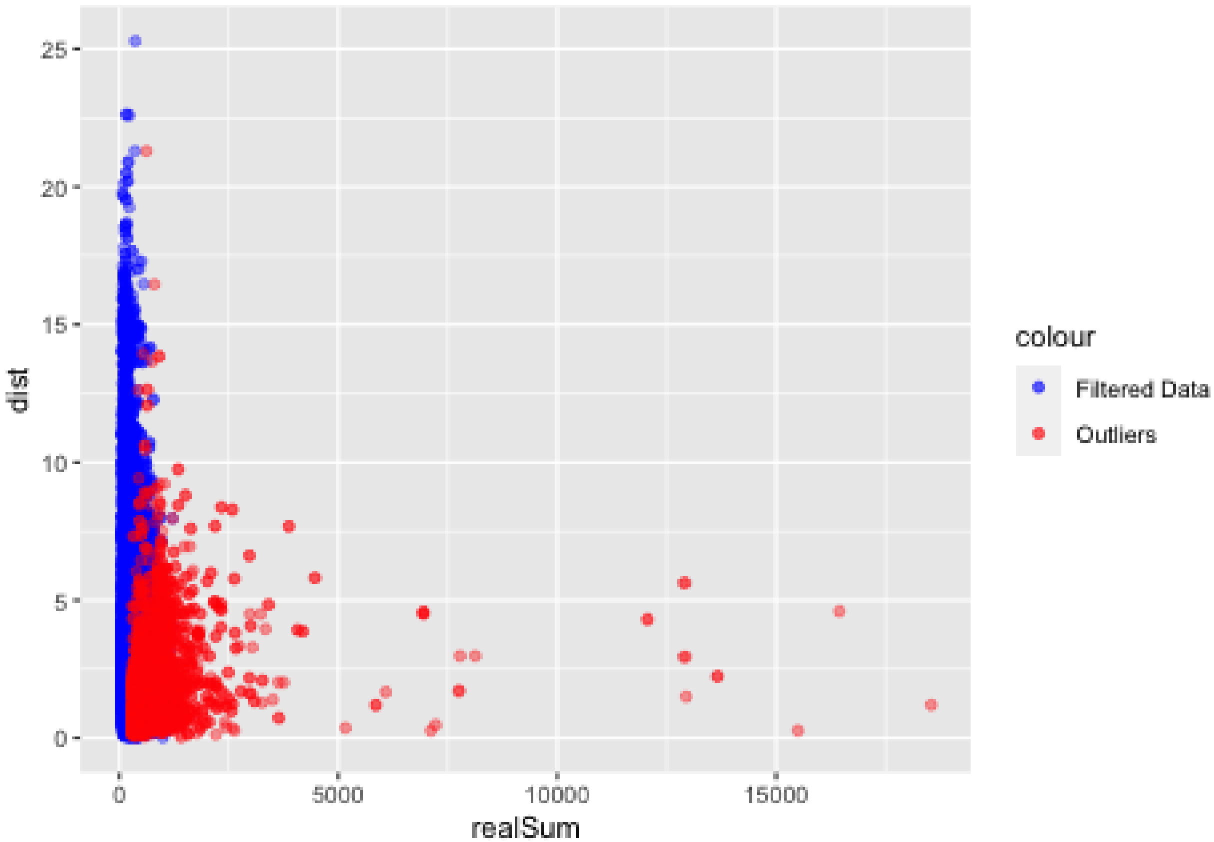

```
ggplot() + geom_point(data = my_data_filtered, aes(x = realSum,
    y = metro_dist, color = "Filtered Data"), alpha = 0.4) +
    geom_point(data = my_outliers, aes(x = realSum, y = metro_dist,
        color = "Outliers"), alpha = 0.4) + scale_color_manual(values = c(`Filtered Data` = "blue",
    Outliers = "red")) + facet_wrap(~city_day)
```



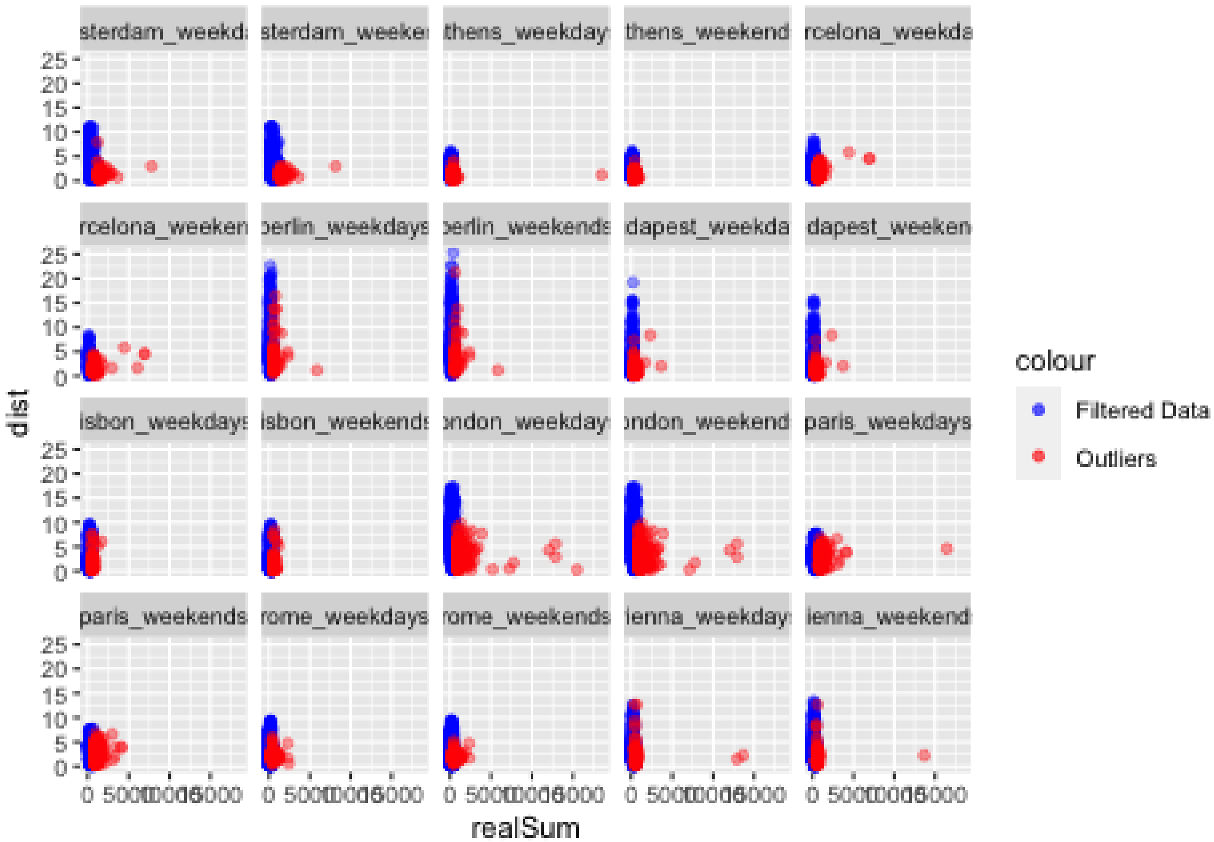
In general the rooms that are closer to metro have comparatively higher prices. But, in Rome city the distance to metro is almost same for both categories of price.

Real Sum vs Distance

```
ggplot() + geom_point(data = my_data_filtered, aes(x = realSum,
    y = dist, color = "Filtered Data"), alpha = 0.4) + geom_point(data = my_outliers,
    aes(x = realSum, y = dist, color = "Outliers"), alpha = 0.4) +
    scale_color_manual(values = c(`Filtered Data` = "blue", Outliers = "red"))
```



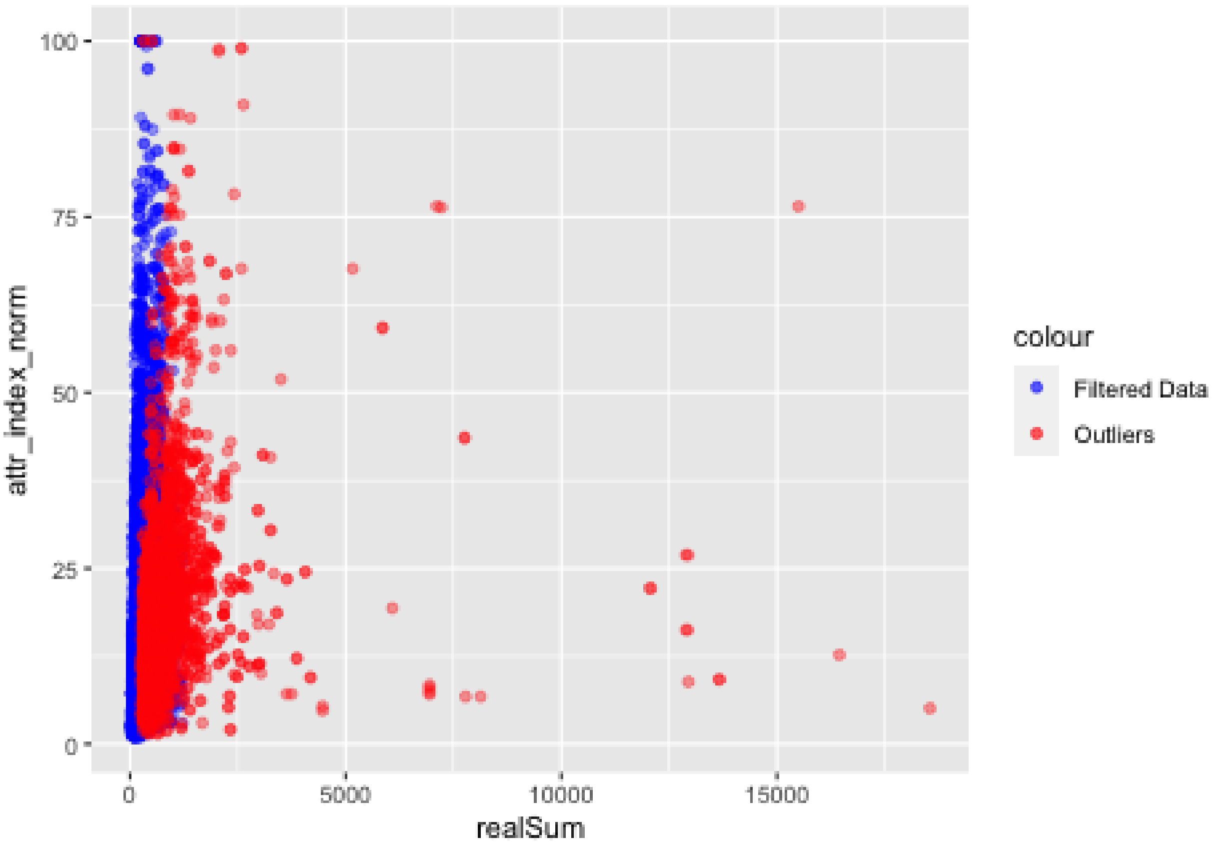

```
ggplot() + geom_point(data = my_data_filtered, aes(x = realSum,
    y = dist, color = "Filtered Data"), alpha = 0.4) + geom_point(data = my_outliers,
    aes(x = realSum, y = dist, color = "Outliers"), alpha = 0.4) +
    scale_color_manual(values = c(`Filtered Data` = "blue", Outliers = "red")) +
    facet_wrap(~city_day)
```



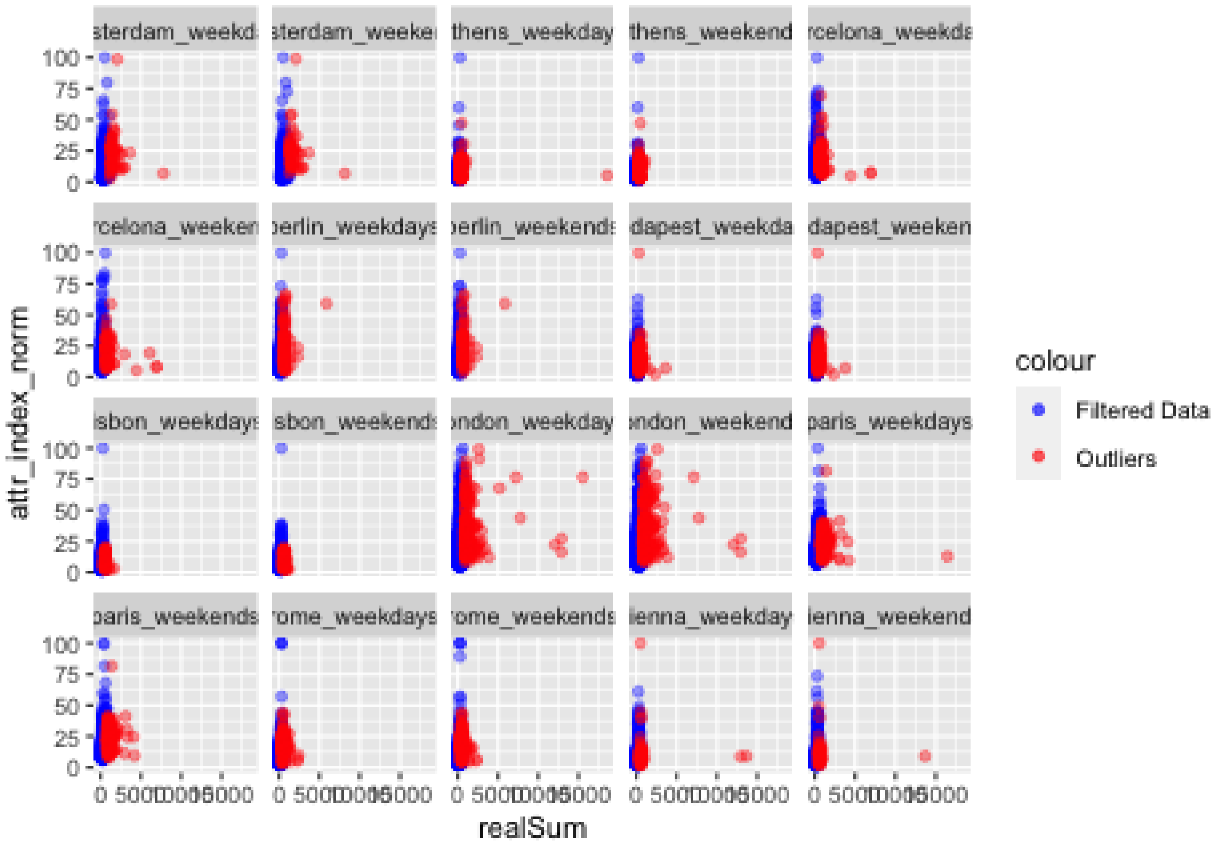
In general the pricey rooms are near to the centre of the city.

Real Sum vs Attraction Index Normal

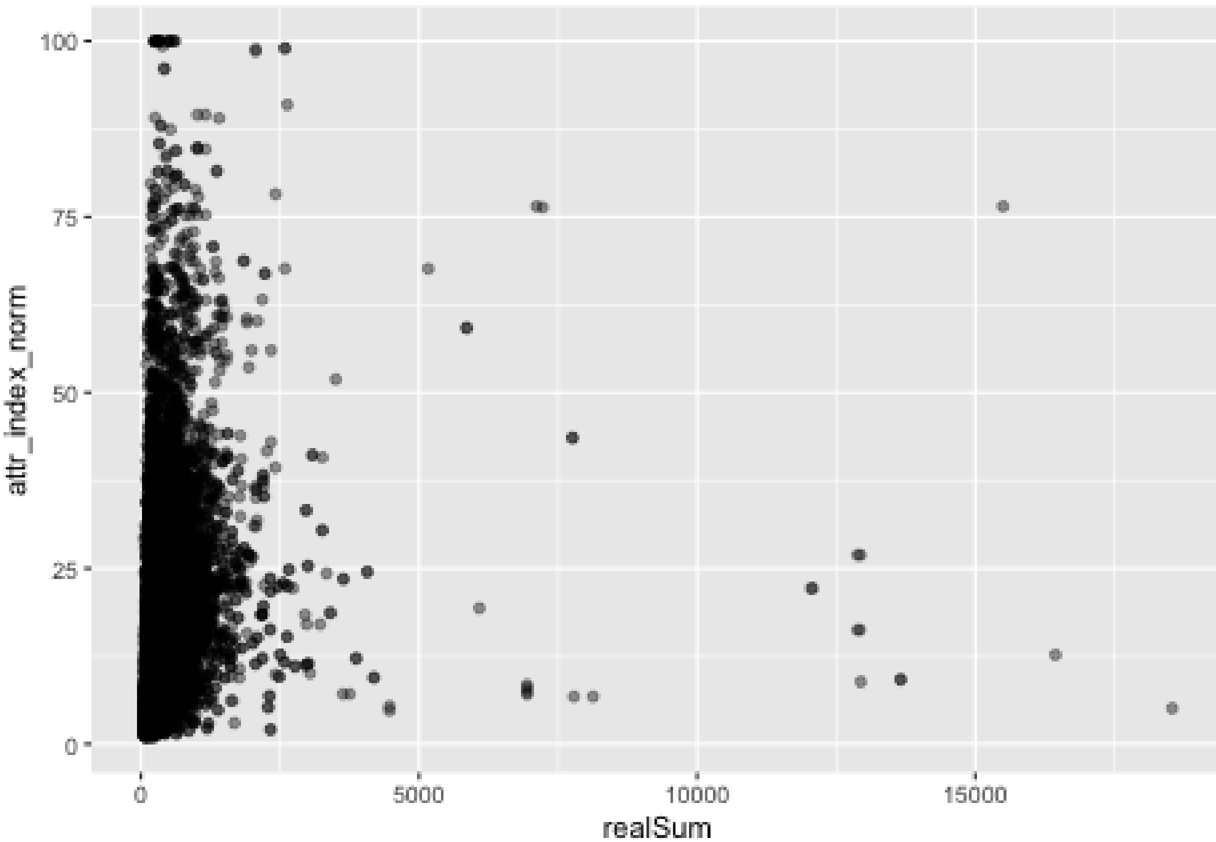
```
ggplot() + geom_point(data = my_data_filtered, aes(x = realSum,
    y = attr_index_norm, color = "Filtered Data"), alpha = 0.4) +
    geom_point(data = my_outliers, aes(x = realSum, y = attr_index_norm,
        color = "Outliers"), alpha = 0.4) + scale_color_manual(values = c(`Filtered Data` = "blue",
    Outliers = "red"))
```




```
ggplot() + geom_point(data = my_data_filtered, aes(x = realSum,
    y = attr_index_norm, color = "Filtered Data"), alpha = 0.4) +
    geom_point(data = my_outliers, aes(x = realSum, y = attr_index_norm,
        color = "Outliers"), alpha = 0.4) + scale_color_manual(values = c(`Filtered Data` = "blue",
    Outliers = "red")) + facet_wrap(~city_day)
```



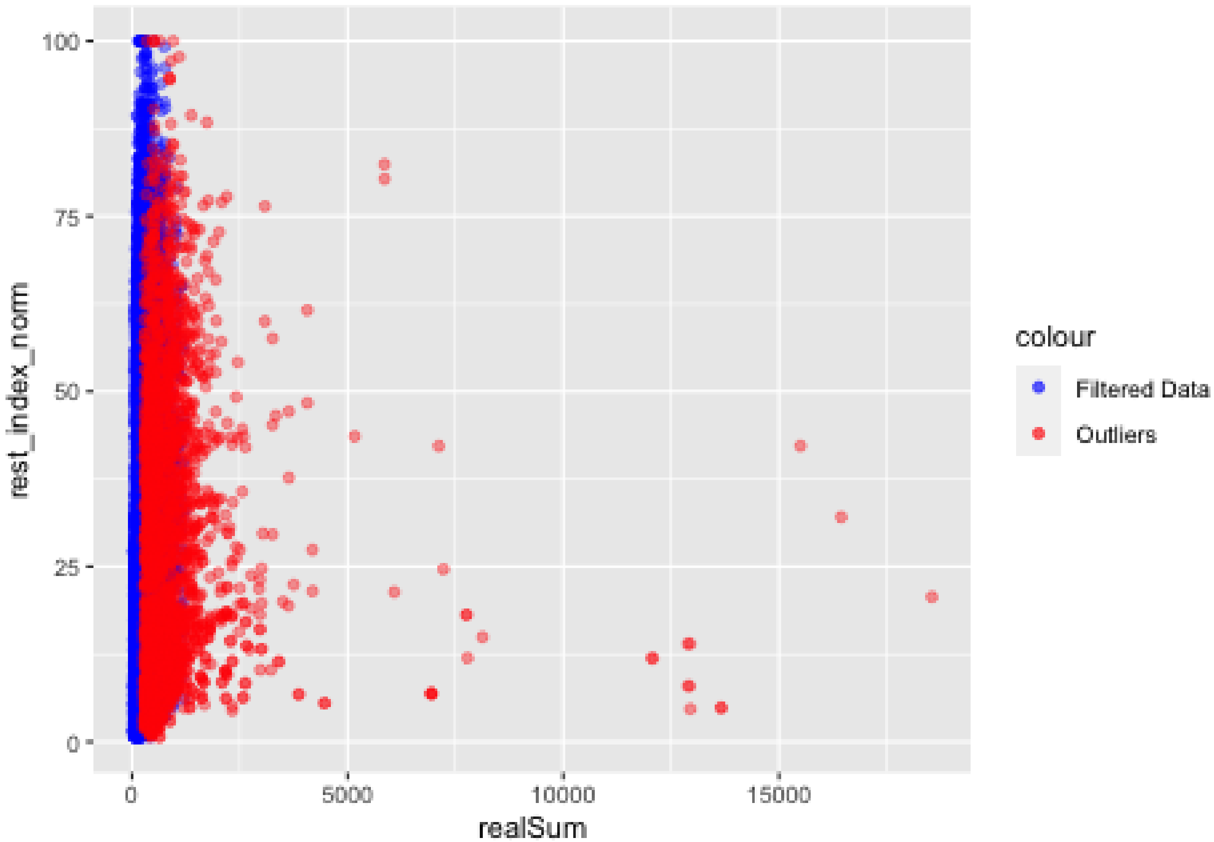

```r
ggplot() + geom_point(data = my_data_filtered, aes(x = realSum,
    y = attr_index_norm), alpha = 0.4) + geom_point(data = my_outliers,
    aes(x = realSum, y = attr_index_norm), alpha = 0.4)
```



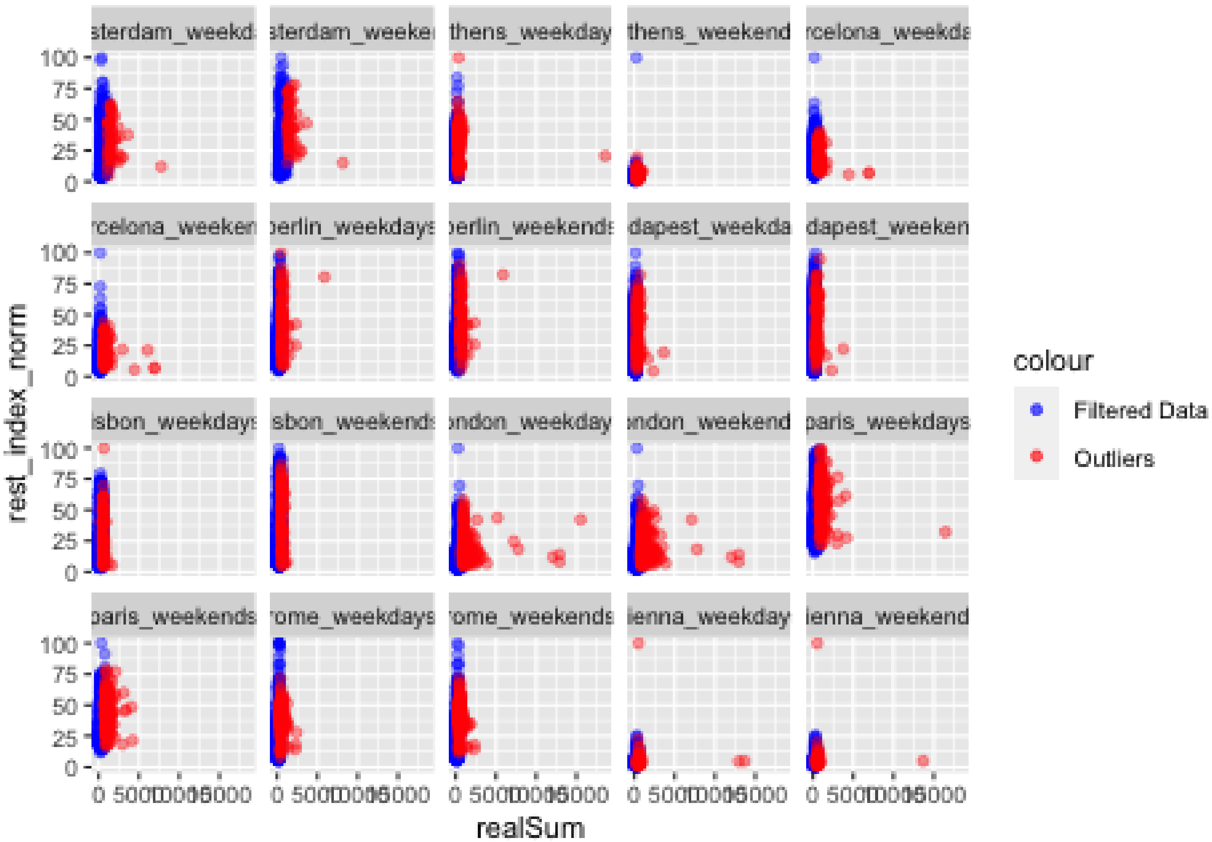
The range of values falling b/w outliers and normal data is almost same . So there isn't a relationship b/w attr_index and realSum.

Real Sum vs Restaurant Index Normal

```
ggplot() + geom_point(data = my_data_filtered, aes(x = realSum,
    y = rest_index_norm, color = "Filtered Data"), alpha = 0.4) +
    geom_point(data = my_outliers, aes(x = realSum, y = rest_index_norm,
        color = "Outliers"), alpha = 0.4) + scale_color_manual(values = c(`Filtered Data` = "blue",
    Outliers = "red"))
```



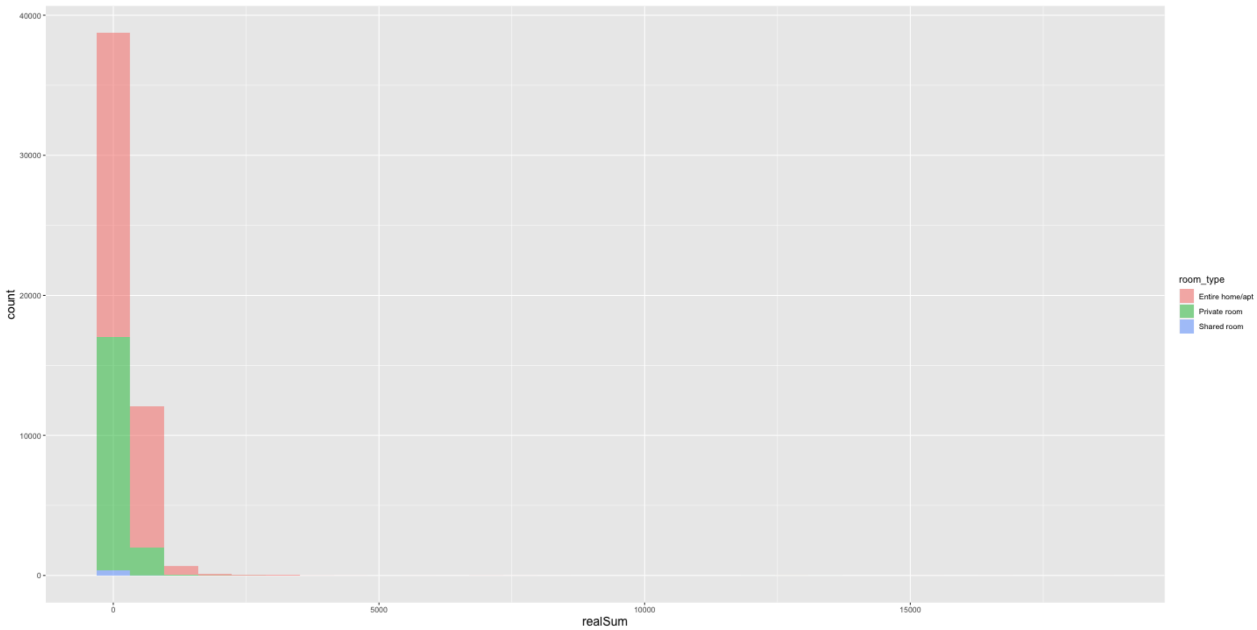

```
ggplot() + geom_point(data = my_data_filtered, aes(x = realSum,
    y = rest_index_norm, color = "Filtered Data"), alpha = 0.4) +
    geom_point(data = my_outliers, aes(x = realSum, y = rest_index_norm,
        color = "Outliers"), alpha = 0.4) + scale_color_manual(values = c(`Filtered Data` = "blue",
    Outliers = "red")) + facet_wrap(~city_day)
```



There is no relationship between outliers and rest_index

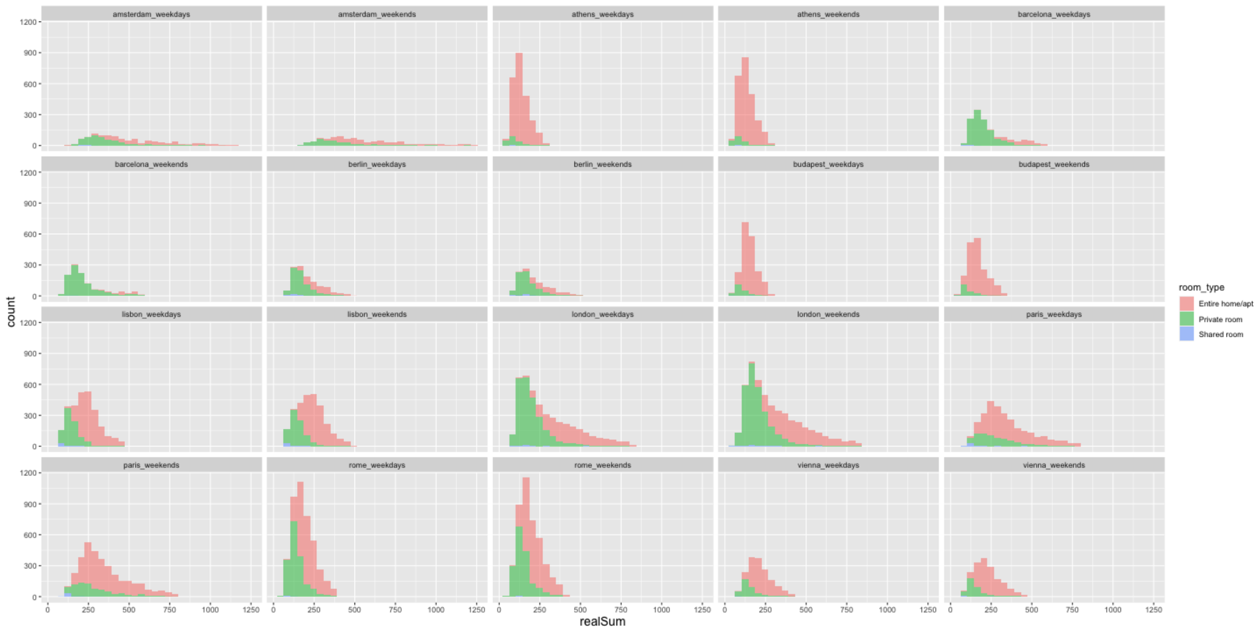
Room Type Vs Real Sum

```
ggplot(my_data, aes(x = realSum, fill = room_type, group = room_type)) +
    geom_histogram(alpha = 0.5, nbins = 20) + theme(axis.title.x = element_text(size = 14),
    axis.title.y = element_text(size = 14))
```




```
ggplot(my_data_filtered, aes(x = realSum, fill = room_type, group = room_type)) +
    geom_histogram(alpha = 0.5, nbins = 20) + theme(axis.title.x = element_text(size = 14),
    axis.title.y = element_text(size = 14))
```

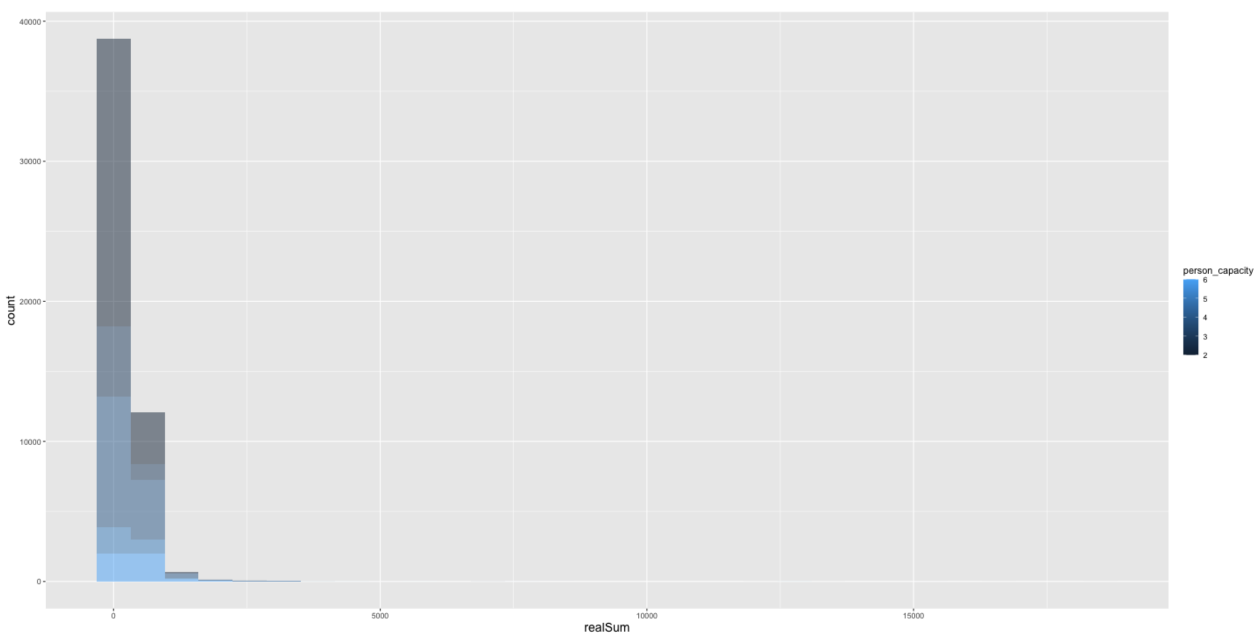

```
ggplot(my_data_filtered, aes(x = realSum, fill = room_type, group = room_type)) +
    geom_histogram(alpha = 0.5, nbins = 20) + theme(axis.title.x = element_text(size = 14),
    axis.title.y = element_text(size = 14)) + facet_wrap(~city_day)
```



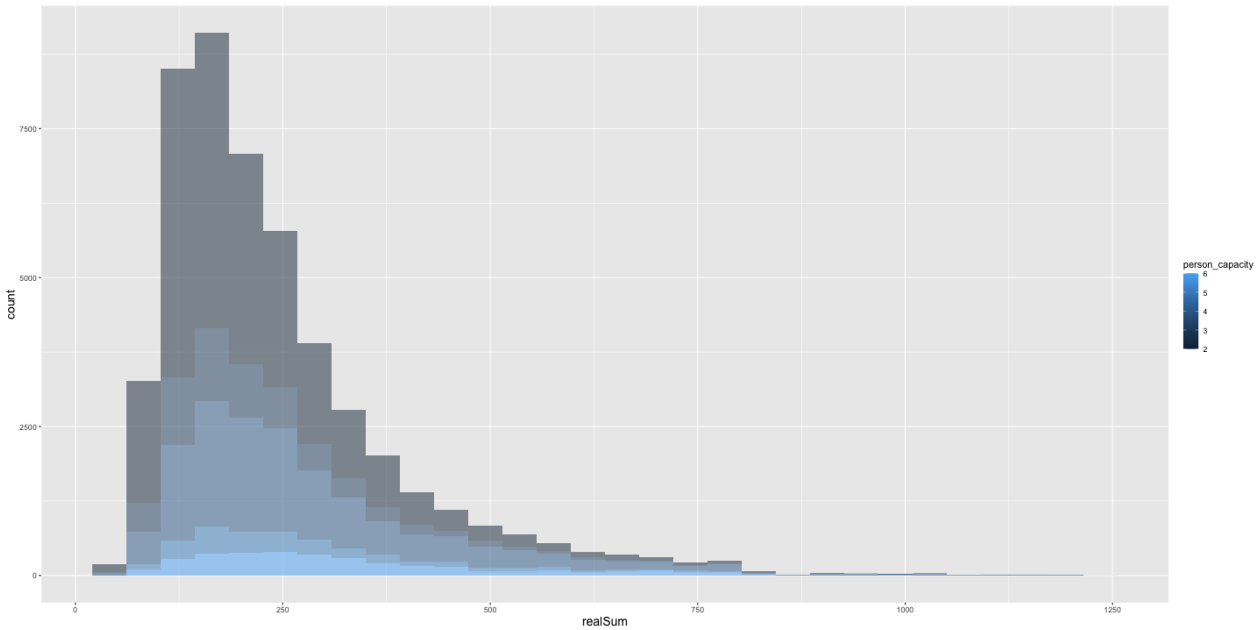
The price of entire home/apt tend to be higher compared to other two categories. And the count of entire home /apt is also more.

Room Type Vs Person Capacity

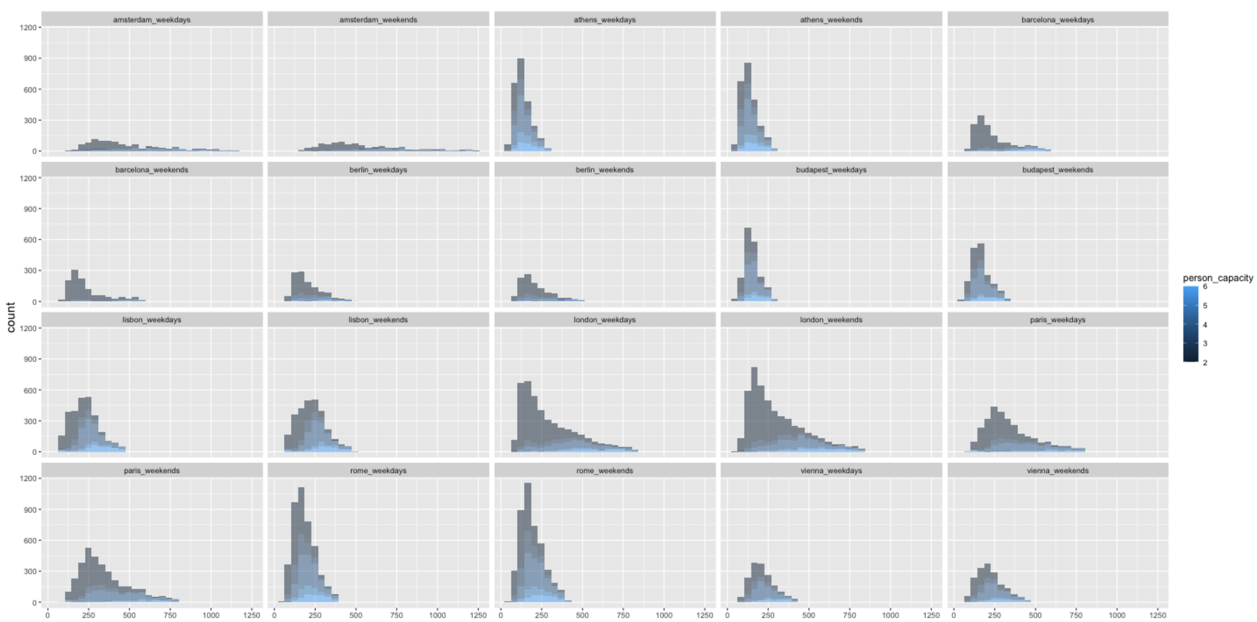
```
ggplot(my_data, aes(x = realSum, fill = person_capacity, group = person_capacity)) +
    geom_histogram(alpha = 0.5, nbins = 20) + theme(axis.title.x = element_text(size = 14),
    axis.title.y = element_text(size = 14))
```




```
ggplot(my_data_filtered, aes(x = realSum, fill = person_capacity,
    group = person_capacity)) + geom_histogram(alpha = 0.5, nbins = 20) +
    theme(axis.title.x = element_text(size = 14), axis.title.y = element_text(size = 14))
```



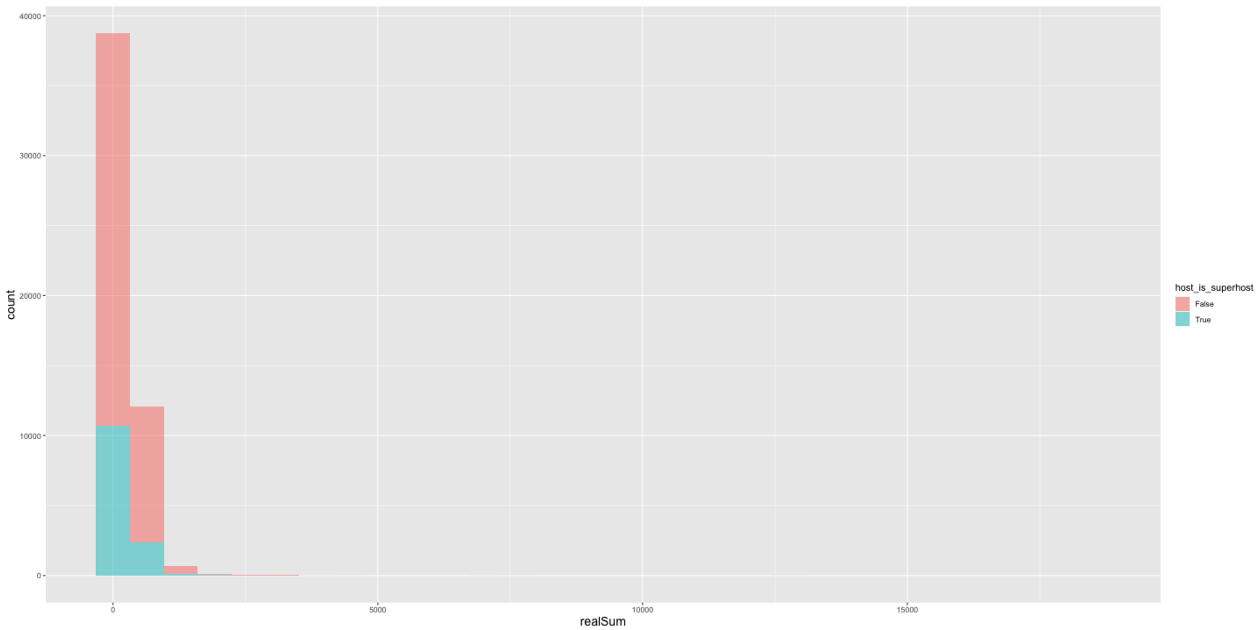

```
ggplot(my_data_filtered, aes(x = realSum, fill = person_capacity,
    group = person_capacity)) + geom_histogram(alpha = 0.5, nbins = 20) +
    theme(axis.title.x = element_text(size = 14), axis.title.y = element_text(size = 14)) +
    facet_wrap(~city_day)
```



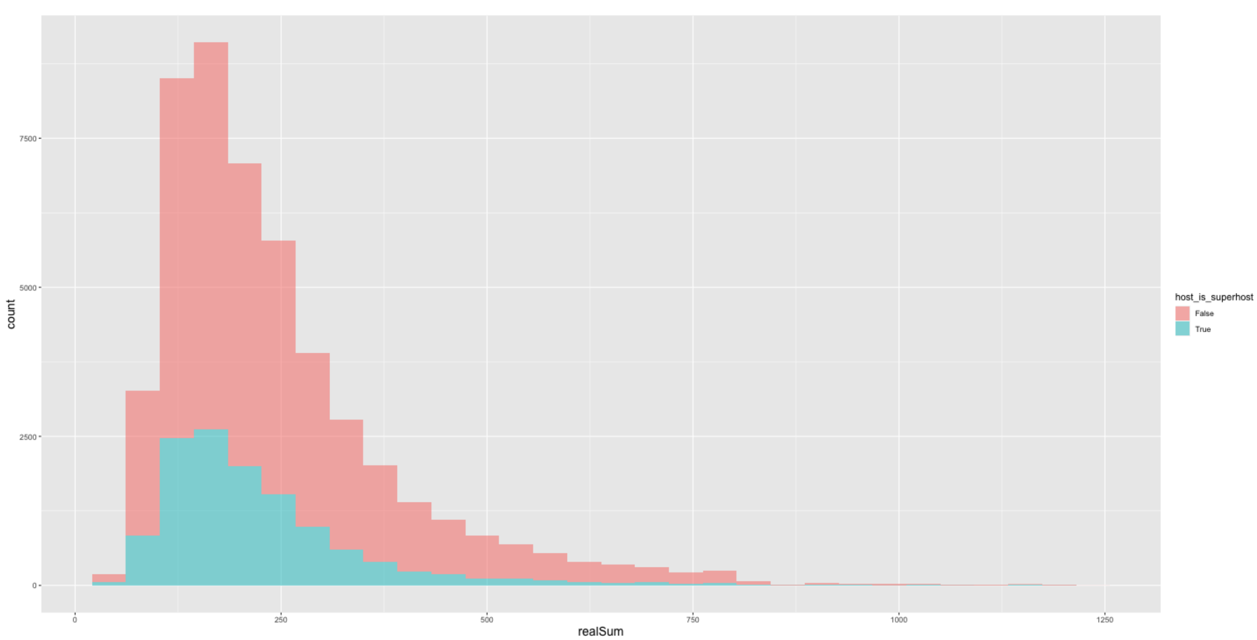
The overall price is distributed similarly across the spectrum irrespective of person_capacity. But for some cities like london, london_weekdays, lisbon the price is higher with person capacity. So, person capacity along with city will be an important variable for determining price.

Real Sum Vs host_is_superhost

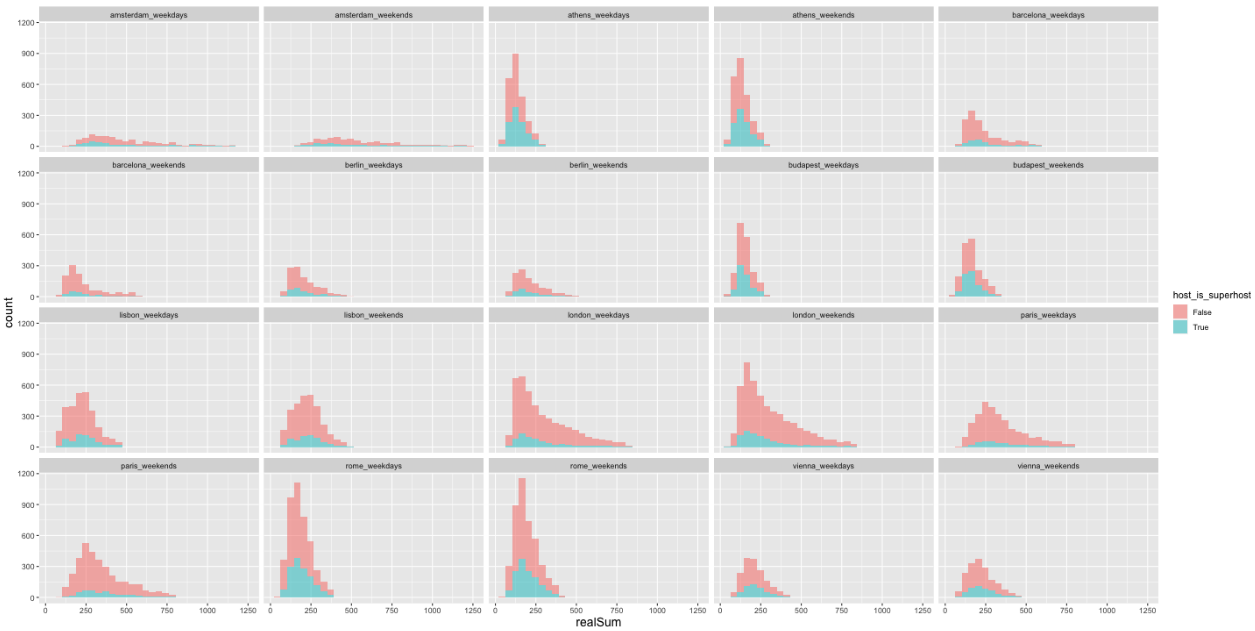
```
ggplot(my_data, aes(x = realSum, fill = host_is_superhost, group = host_is_superhost)) +
    geom_histogram(alpha = 0.5, nbins = 20) + theme(axis.title.x = element_text(size = 14),
    axis.title.y = element_text(size = 14))
```




```
ggplot(my_data_filtered, aes(x = realSum, fill = host_is_superhost,
    group = host_is_superhost)) + geom_histogram(alpha = 0.5,
    nbins = 20) + theme(axis.title.x = element_text(size = 14),
    axis.title.y = element_text(size = 14))
```



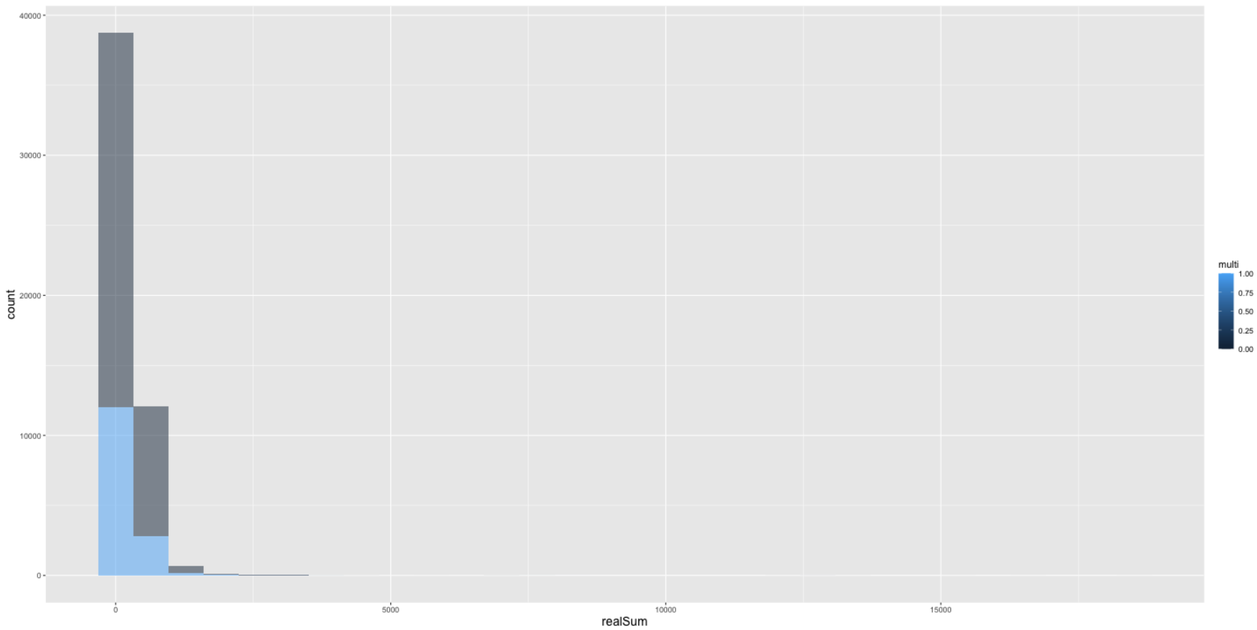
```r
ggplot(my_data_filtered, aes(x = realSum, fill = host_is_superhost,
    group = host_is_superhost)) + geom_histogram(alpha = 0.5,
    nbins = 20) + theme(axis.title.x = element_text(size = 14),
    axis.title.y = element_text(size = 14)) + facet_wrap(~city_day)
```



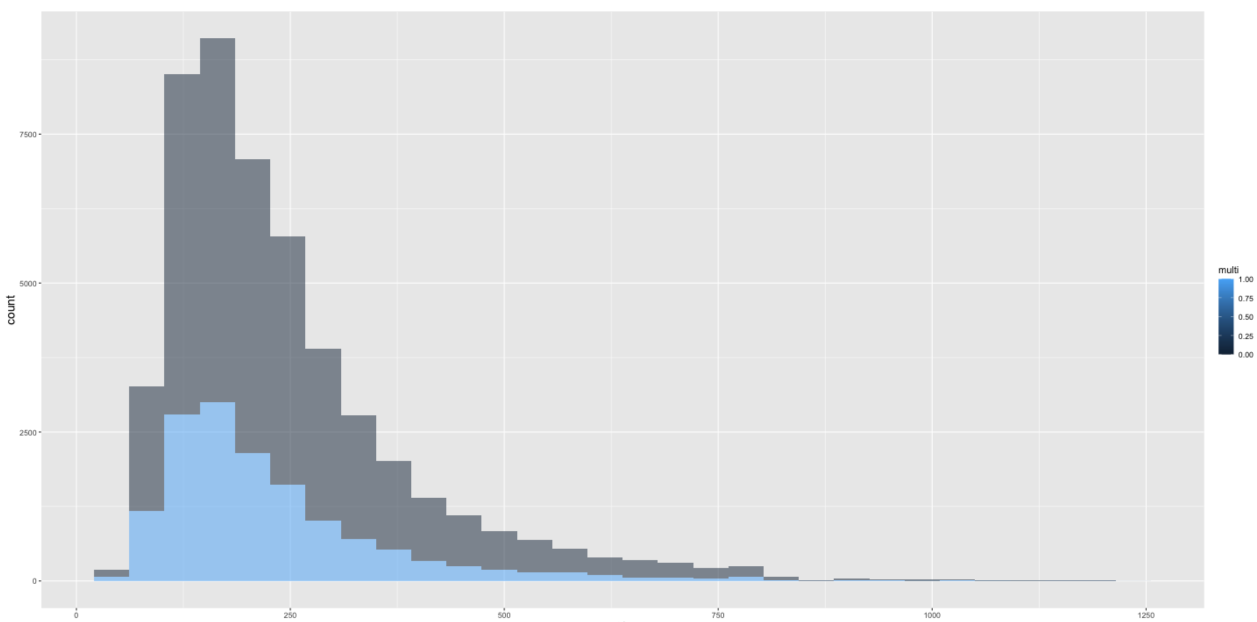
The prices are spread across all the spectrum irrespective of super_host or not.

Real Sum Vs multi

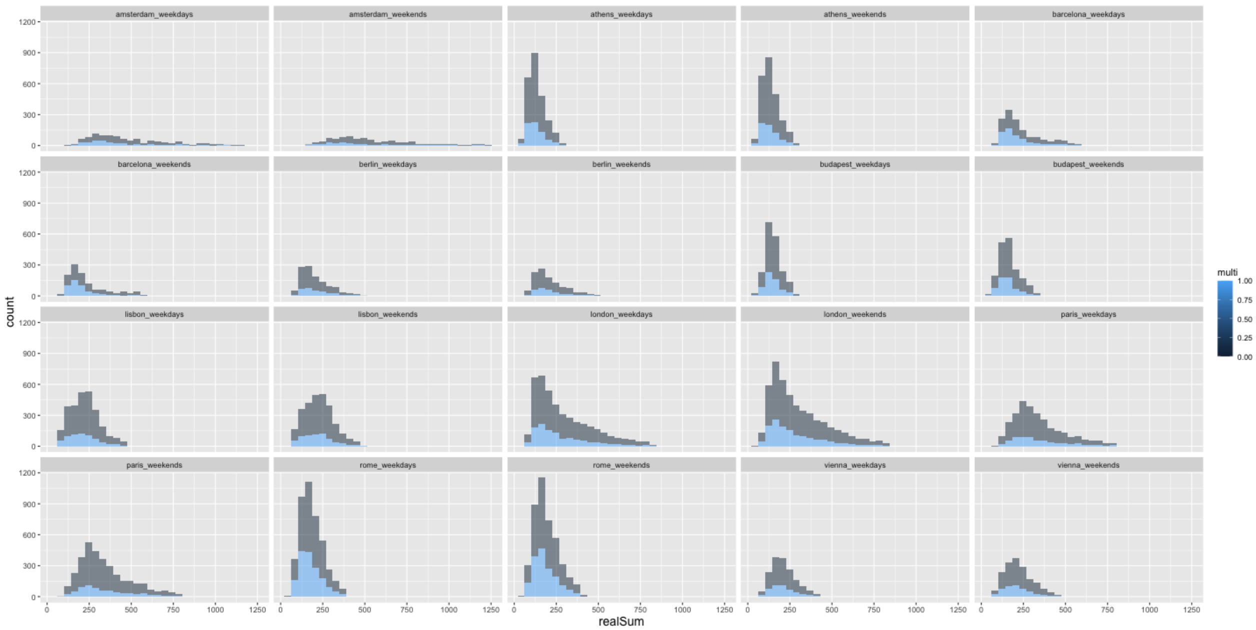
```r
ggplot(my_data, aes(x = realSum, fill = multi, group = multi)) +
    geom_histogram(alpha = 0.5, nbins = 20) + theme(axis.title.x = element_text(size = 14),
    axis.title.y = element_text(size = 14))
```




```
ggplot(my_data_filtered, aes(x = realSum, fill = multi, group = multi)) +
    geom_histogram(alpha = 0.5, nbins = 20) + theme(axis.title.x = element_text(size = 14),
    axis.title.y = element_text(size = 14))
```



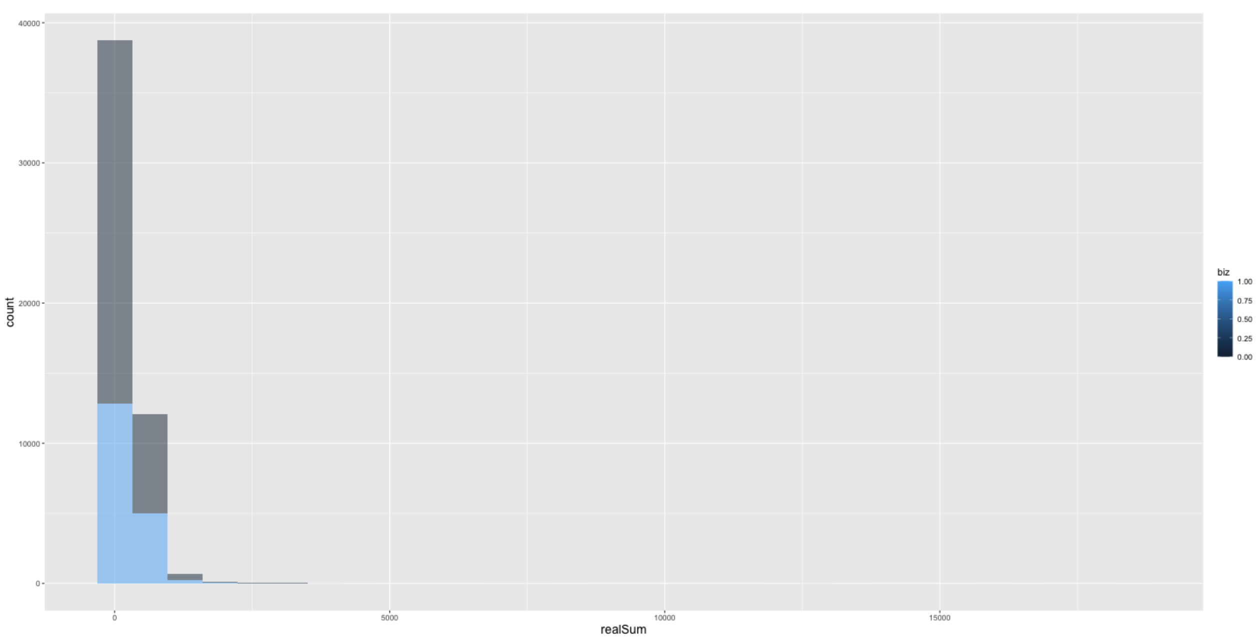

```
ggplot(my_data_filtered, aes(x = realSum, fill = multi, group = multi)) +
    geom_histogram(alpha = 0.5, nbins = 20) + theme(axis.title.x = element_text(size = 14),
    axis.title.y = element_text(size = 14)) + facet_wrap(~city_day)
```



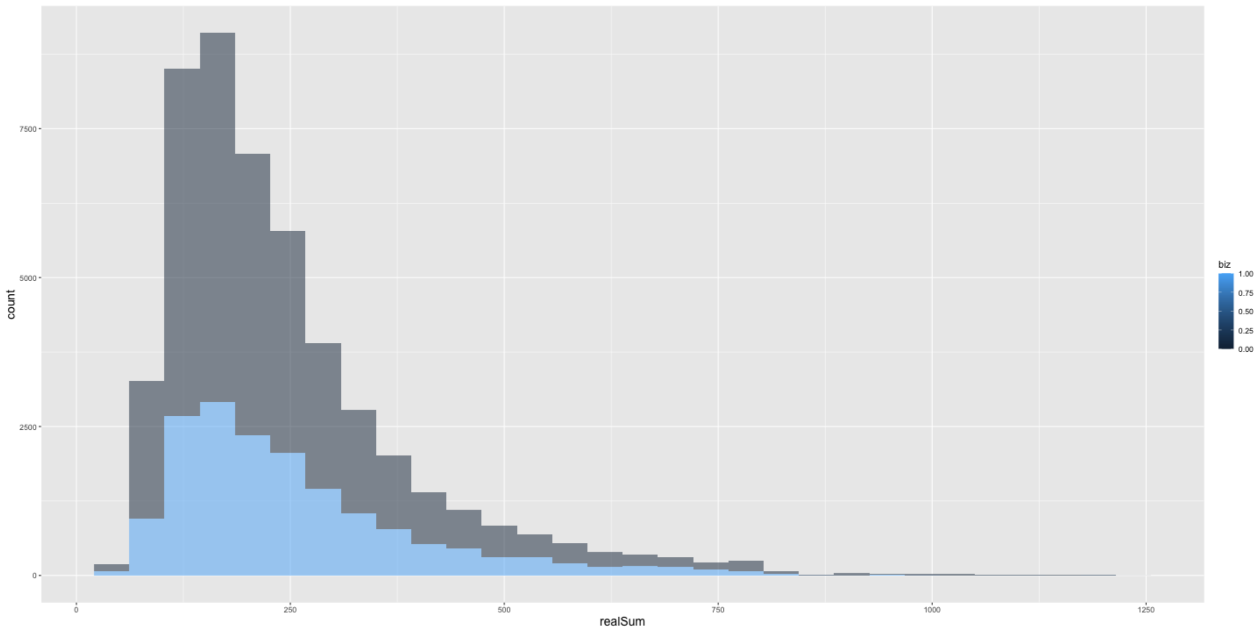
The prices are similar irrespective of multi or not.

Real Sum Vs biz

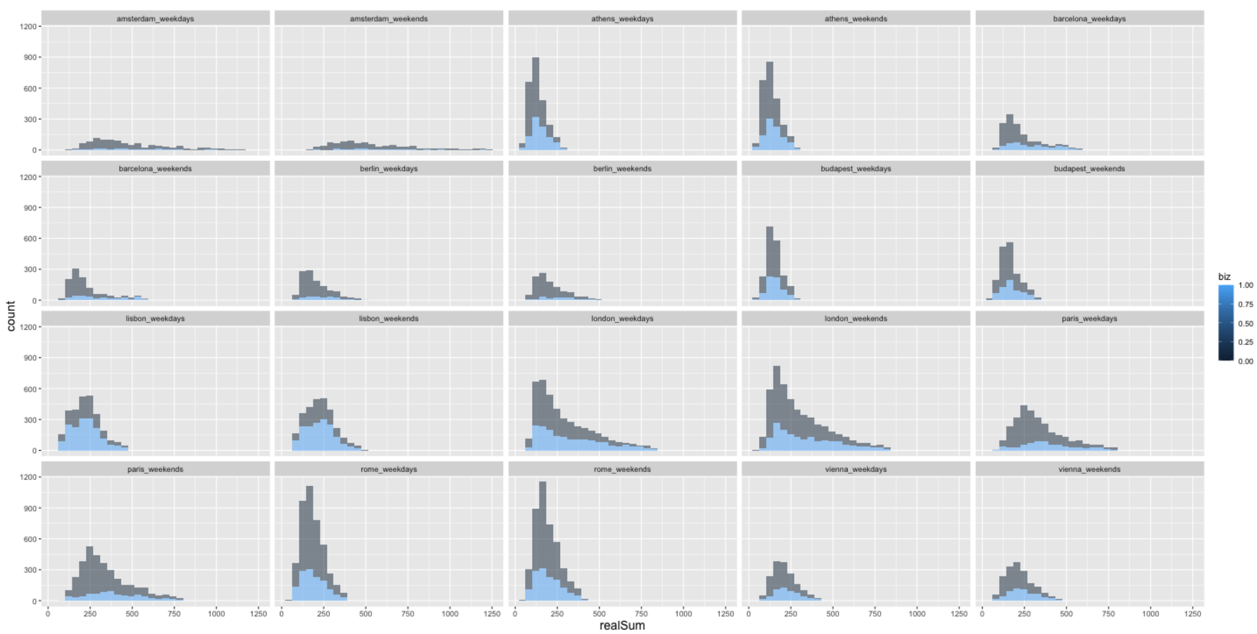
```r
ggplot(my_data, aes(x = realSum, fill = biz, group = biz)) +
    geom_histogram(alpha = 0.5, nbins = 20) + theme(axis.title.x = element_text(size = 14),
    axis.title.y = element_text(size = 14))
```




```r
ggplot(my_data_filtered, aes(x = realSum, fill = biz, group = biz)) +
    geom_histogram(alpha = 0.5, nbins = 20) + theme(axis.title.x = element_text(size = 14),
    axis.title.y = element_text(size = 14))
```



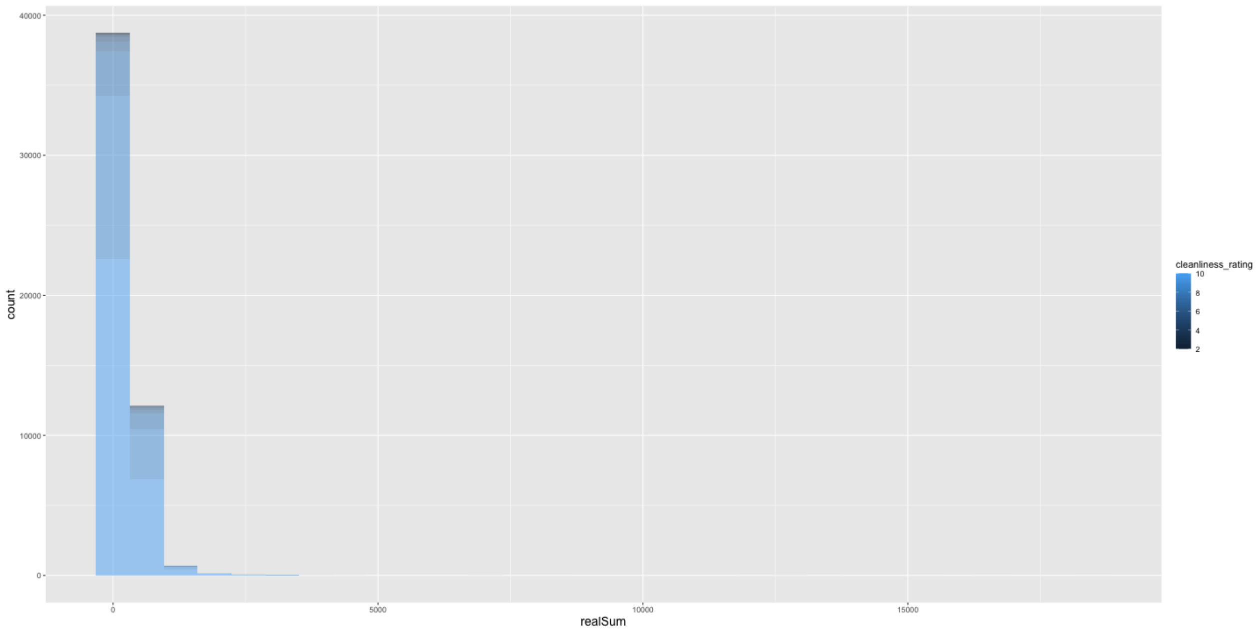

```
ggplot(my_data_filtered, aes(x = realSum, fill = biz, group = biz)) +
    geom_histogram(alpha = 0.5, nbins = 20) + theme(axis.title.x = element_text(size = 14),
    axis.title.y = element_text(size = 14)) + facet_wrap(~city_day)
```



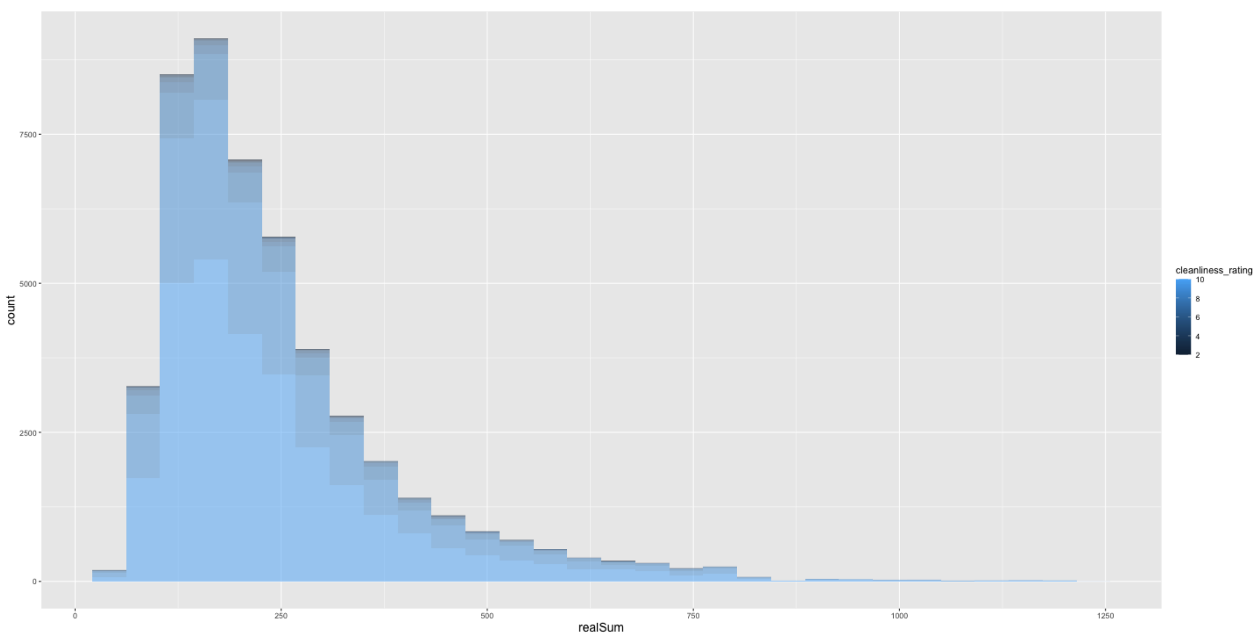
The prices are similar irrespective of biz or not.

Real Sum vs Cleanliness

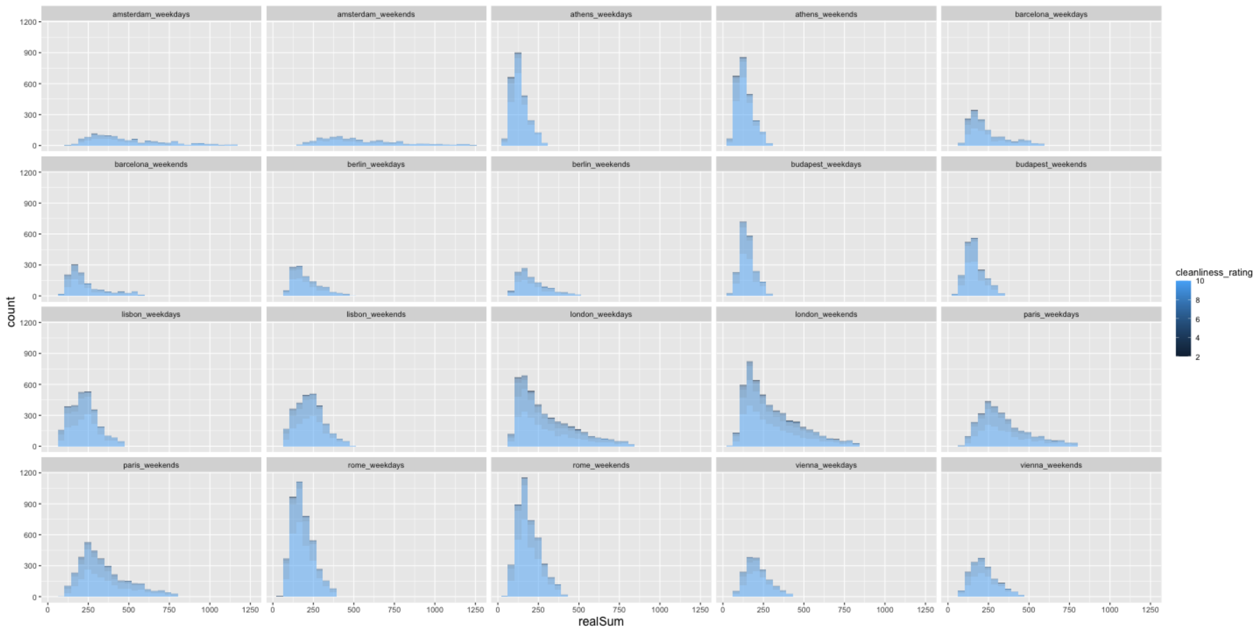
```
ggplot(my_data, aes(x = realSum, fill = cleanliness_rating, group = cleanliness_rating)) +
    geom_histogram(alpha = 0.5, nbins = 20) + theme(axis.title.x = element_text(size = 14),
    axis.title.y = element_text(size = 14))
```




```
ggplot(my_data_filtered, aes(x = realSum, fill = cleanliness_rating,
    group = cleanliness_rating)) + geom_histogram(alpha = 0.5,
    nbins = 20) + theme(axis.title.x = element_text(size = 14),
    axis.title.y = element_text(size = 14))
```



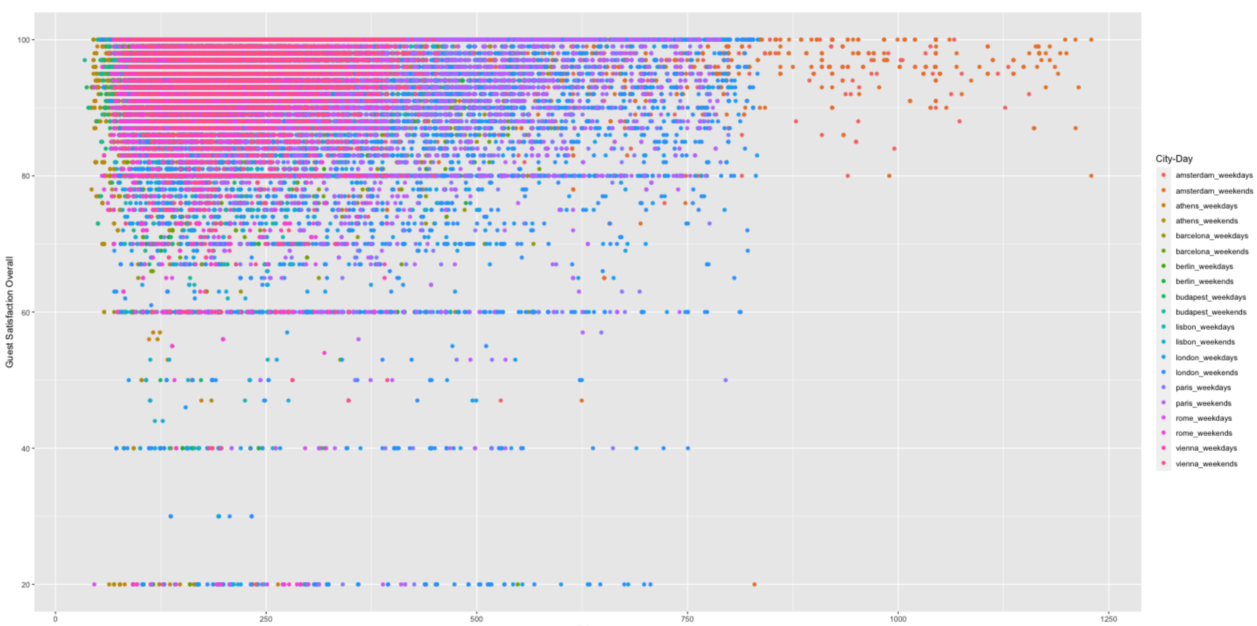

```
ggplot(my_data_filtered, aes(x = realSum, fill = cleanliness_rating,
    group = cleanliness_rating)) + geom_histogram(alpha = 0.5,
    nbins = 20) + theme(axis.title.x = element_text(size = 14),
    axis.title.y = element_text(size = 14)) + facet_wrap(~city_day)
```



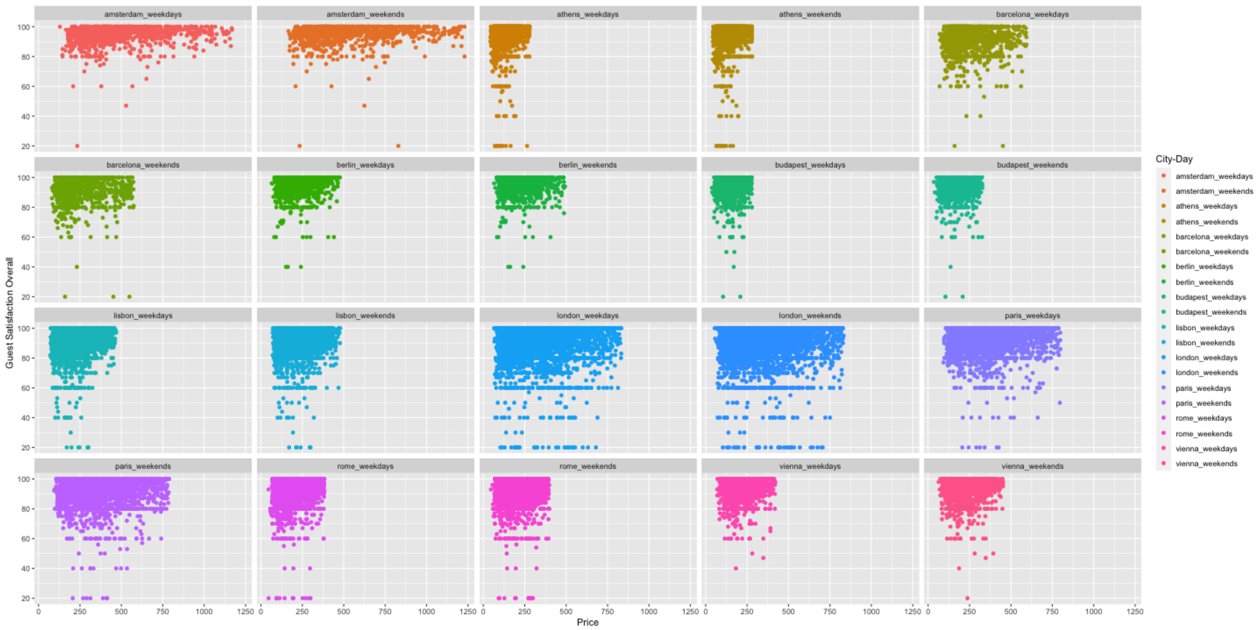
The cleanliness rating doesn't really have an effect on price

Scatterplot of Price vs Guest Satisfaction filtered by city

```
ggplot(my_data_filtered, aes(x = realSum, y = guest_satisfaction_overall,
    color = city_day)) + geom_point() + xlab("Price") + ylab("Guest Satisfaction Overall") +
    scale_color_discrete(name = "City-Day")
```



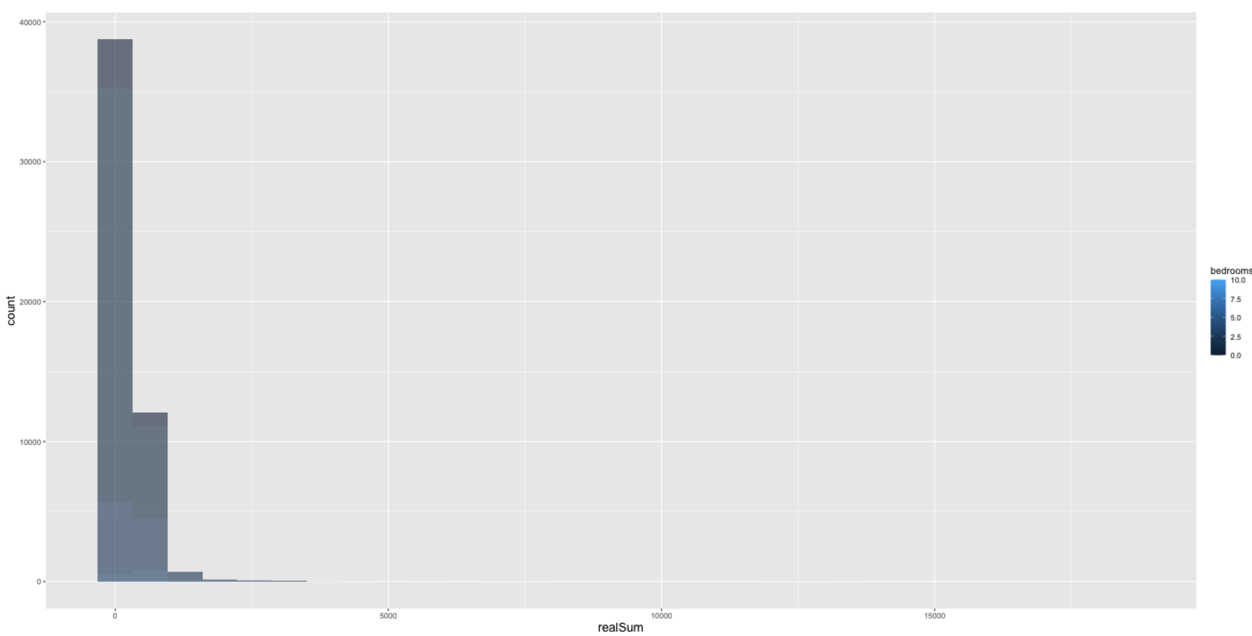

```
ggplot(my_data_filtered, aes(x = realSum, y = guest_satisfaction_overall,
    color = city_day)) + geom_point() + xlab("Price") + ylab("Guest Satisfaction Overall") +
    scale_color_discrete(name = "City-Day") + facet_wrap(~city_day)
```



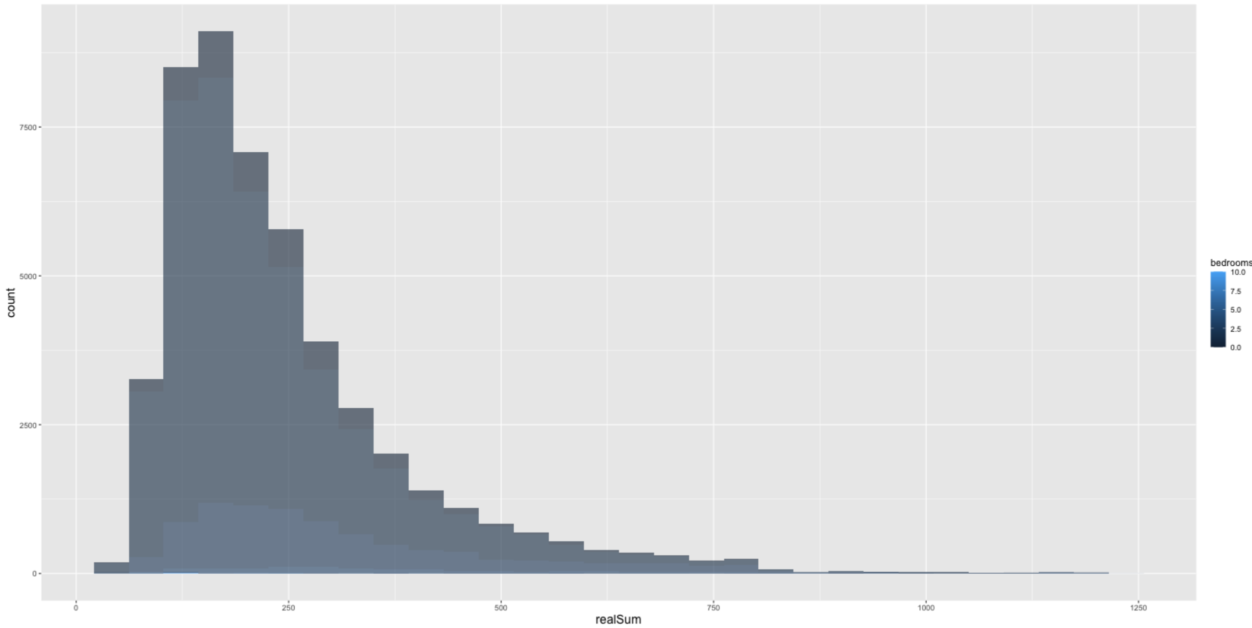
The plot depicts that there is no correlation of price with guest satisfaction, good satisfaction rate is found across all the prices. In some cities like lonon, we can see a group of reviews with low guest satisfaction.

Real Sum Vs Bedroom Count

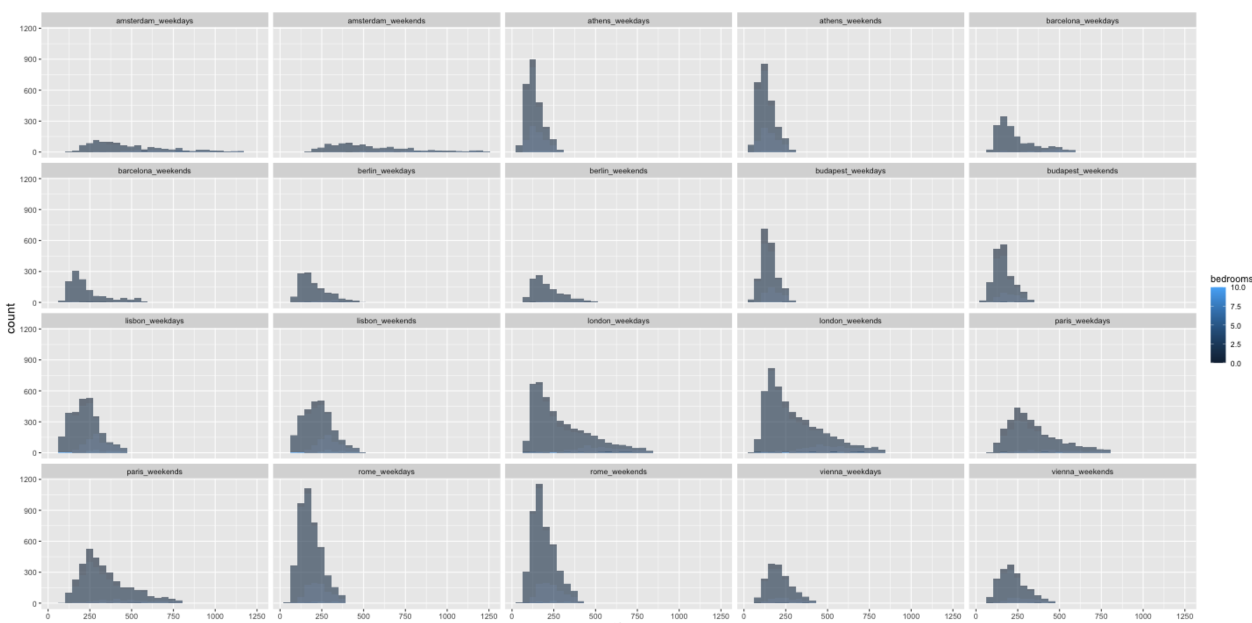
```r
ggplot(my_data, aes(x = realSum, fill = bedrooms, group = bedrooms)) +
    geom_histogram(alpha = 0.6) + theme(axis.title.x = element_text(size = 14),
    axis.title.y = element_text(size = 14))
```



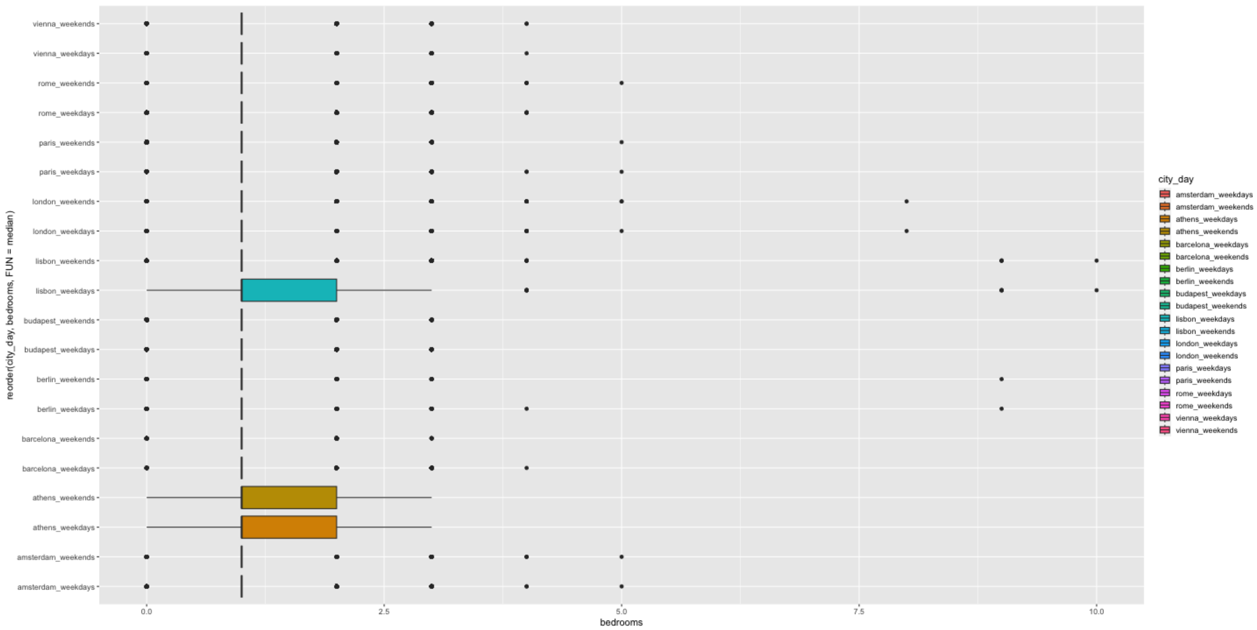

```r
ggplot(my_data_filtered, aes(x = realSum, fill = bedrooms, group = bedrooms)) +
    geom_histogram(alpha = 0.6) + theme(axis.title.x = element_text(size = 14),
    axis.title.y = element_text(size = 14))
```




```r
ggplot(my_data_filtered, aes(x = realSum, fill = bedrooms, group = bedrooms)) +
    geom_histogram(alpha = 0.6) + theme(axis.title.x = element_text(size = 14),
    axis.title.y = element_text(size = 14)) + facet_wrap(~city_day)
```




```r
ggplot(my_data_filtered, aes(x = reorder(city_day, bedrooms,
    FUN = median), y = bedrooms, fill = city_day)) + geom_boxplot() +
    coord_flip() + theme(legend.key.height = unit(0.5, "cm"),
    legend.key.size = unit(1, "lines"))
```



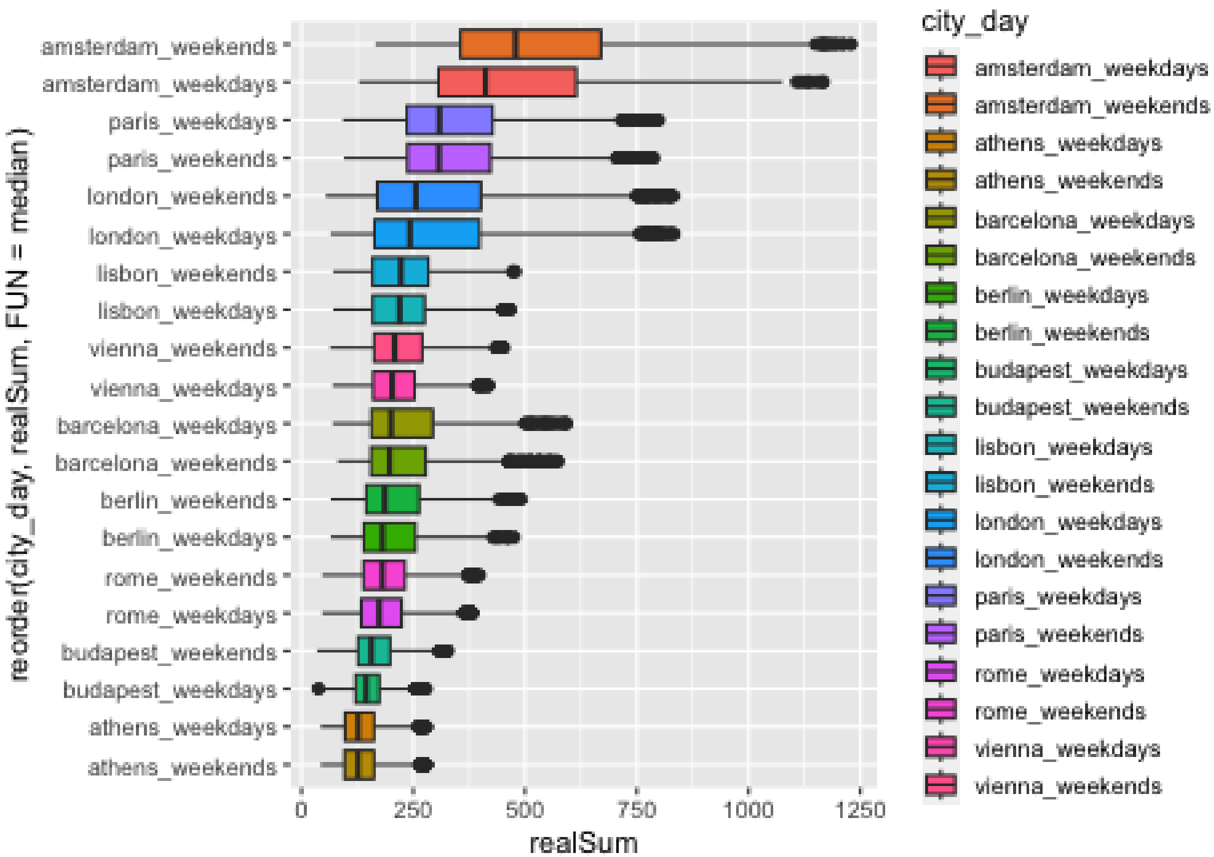

```
cor(as.numeric(factor(my_data$multi)), as.numeric(factor(my_data$biz)))
```

```
## [1] -0.4707248
```

Non Outlier Analysis

Boxplot of Price Vs City

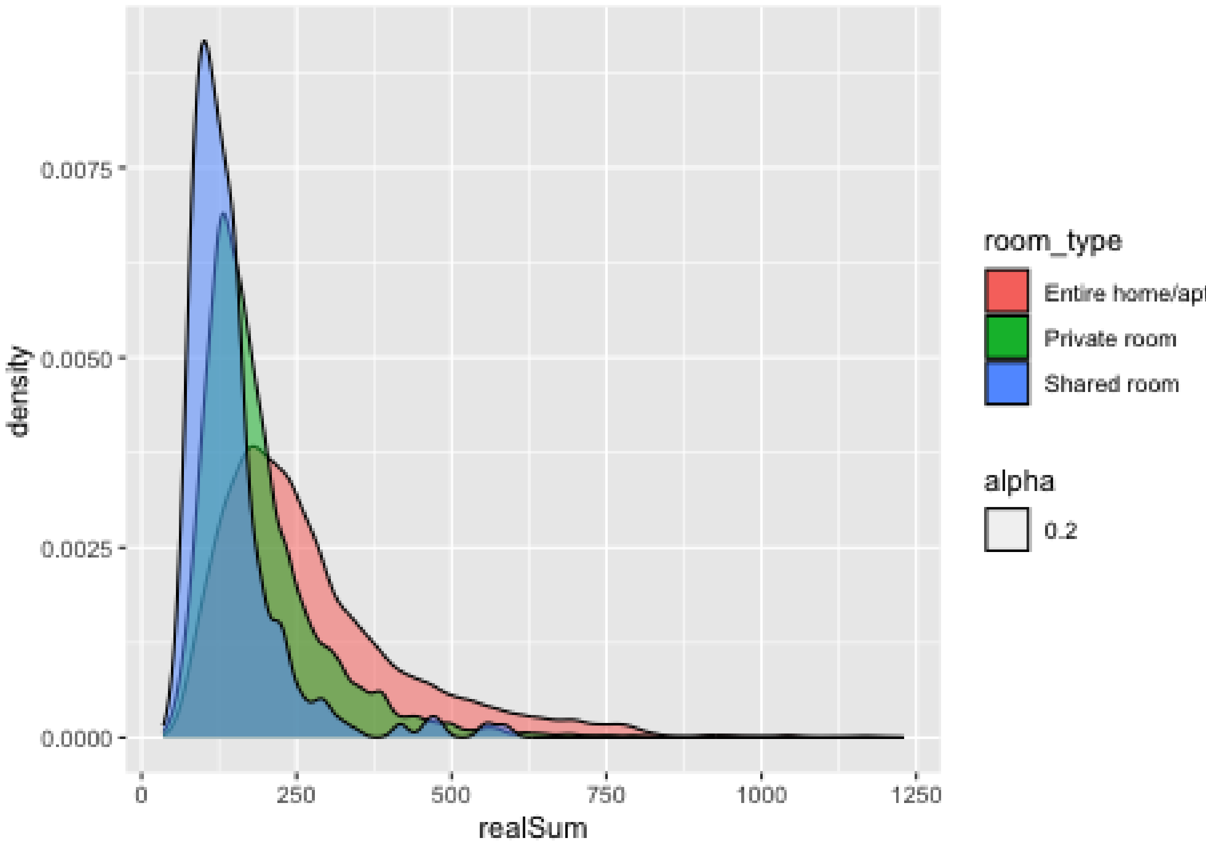
```
ggplot(my_data_filtered, aes(x = reorder(city_day, realSum, FUN = median),
    y = realSum, fill = city_day)) + geom_boxplot() + coord_flip() +
    theme(legend.key.height = unit(0.5, "cm"), legend.key.size = unit(1,
        "lines"))
```



The highest prices in Europe are found in Amsterdam.

Density plot of Price vs Room type

```
ggplot(my_data_filtered, aes(x = realSum, group = room_type,
    fill = room_type, alpha = 0.2)) + geom_density()
```



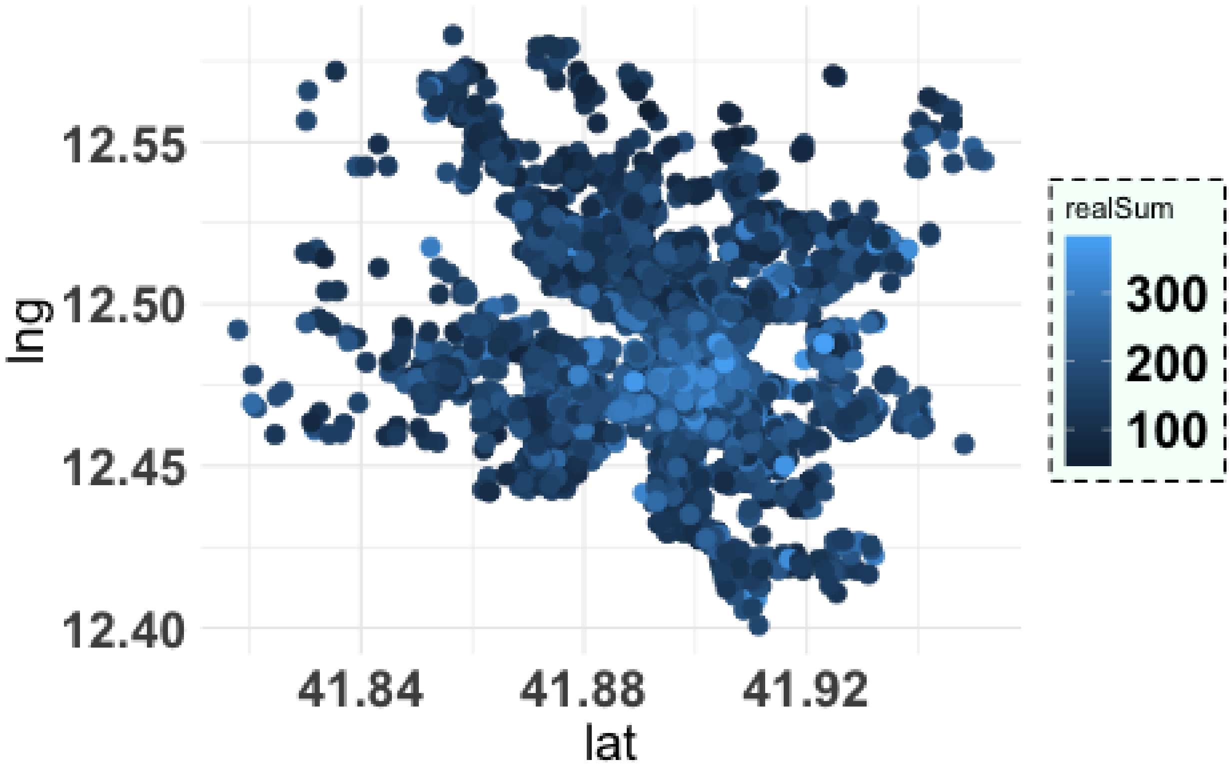
The prices of entire home are high comparatively

Scatterplot of Prices in Rome w.r.t Latitude and Longitude during weekdays

```
tema <- theme(plot.title = element_text(size = 23, hjust = 0.5),
    axis.text.x = element_text(size = 19, face = "bold"), axis.text.y = element_text(size = 19,
        face = "bold"), axis.title.x = element_text(size = 19),
    axis.title.y = element_text(size = 19), legend.text = element_text(colour = "black",
        size = 19, face = "bold"), legend.background = element_rect(fill = "#F5FFFA",
        size = 0.5, linetype = "dashed", colour = "black"))

rome_data <- my_data_filtered %>%
    subset(city_day == "rome_weekdays")

ggplot(data = rome_data, mapping = aes(x = lat, y = lng)) + theme_minimal() +
    scale_fill_identity() + geom_point(mapping = aes(color = realSum),
    size = 3) + ggtitle("") + tema
```

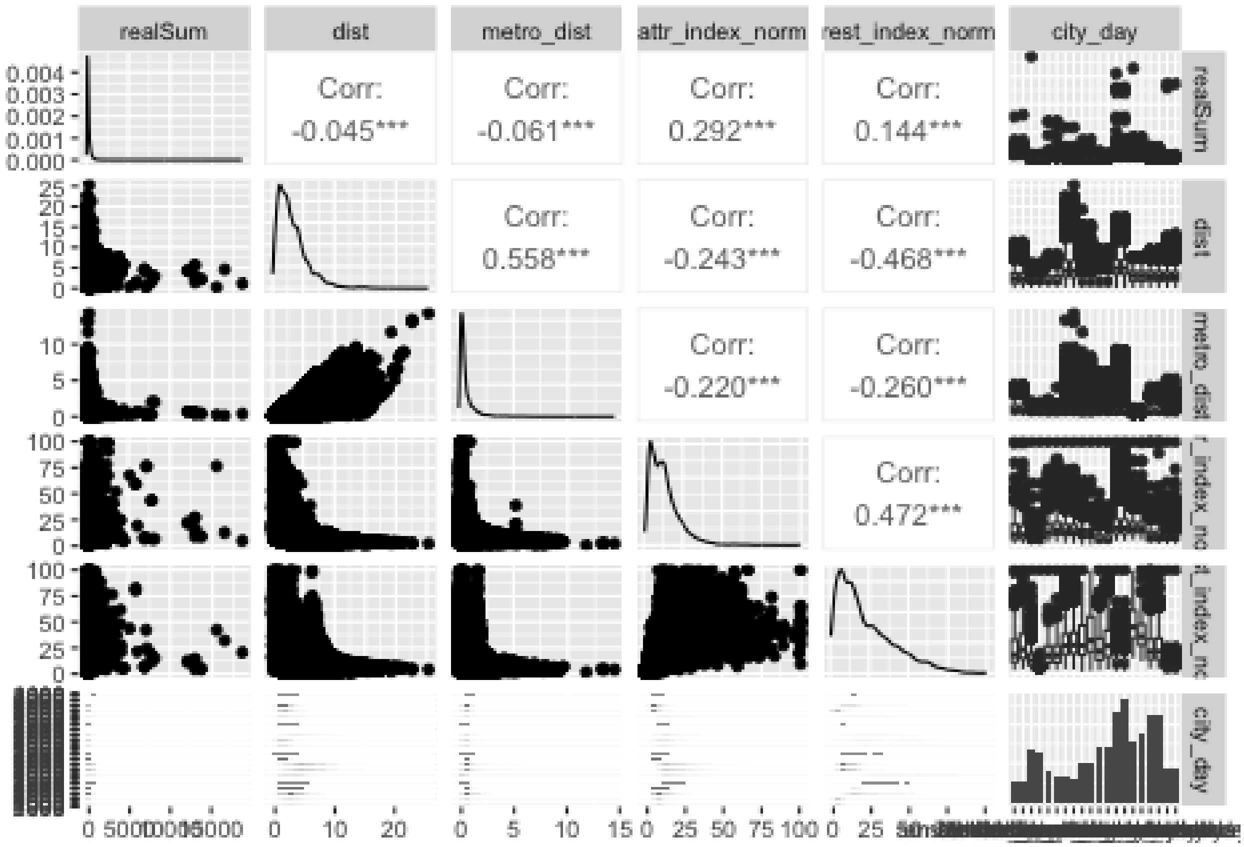


This plot is within expectations of general trends, which suggests similar types of establishments (price and hospitality) tend be in clusters.

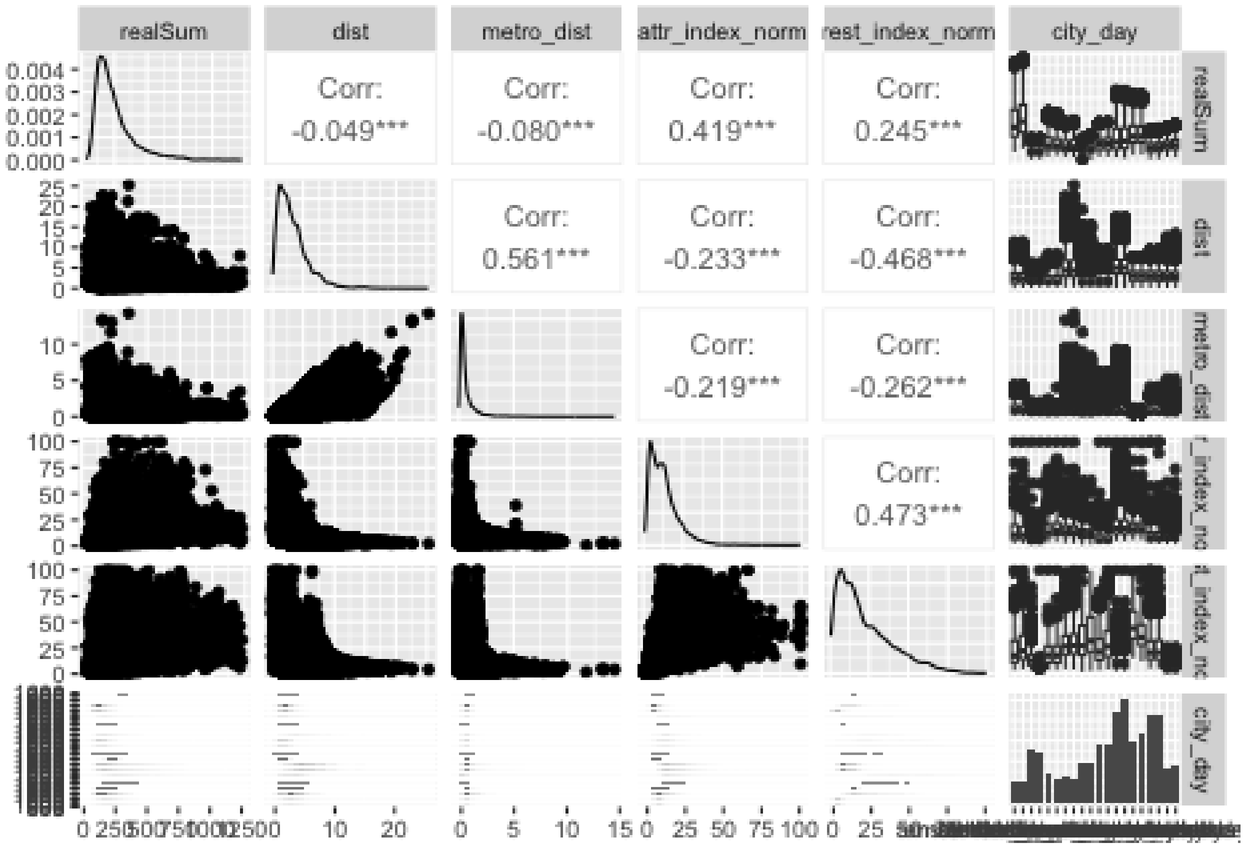
Different Model Selection and Training

Checking for correlations between different attributes

```
ggpairs(my_data[c("realSum", "dist", "metro_dist", "attr_index_norm",
    "rest_index_norm", "city_day")], cardinality = 20, cardinality_threshold = 999)
```



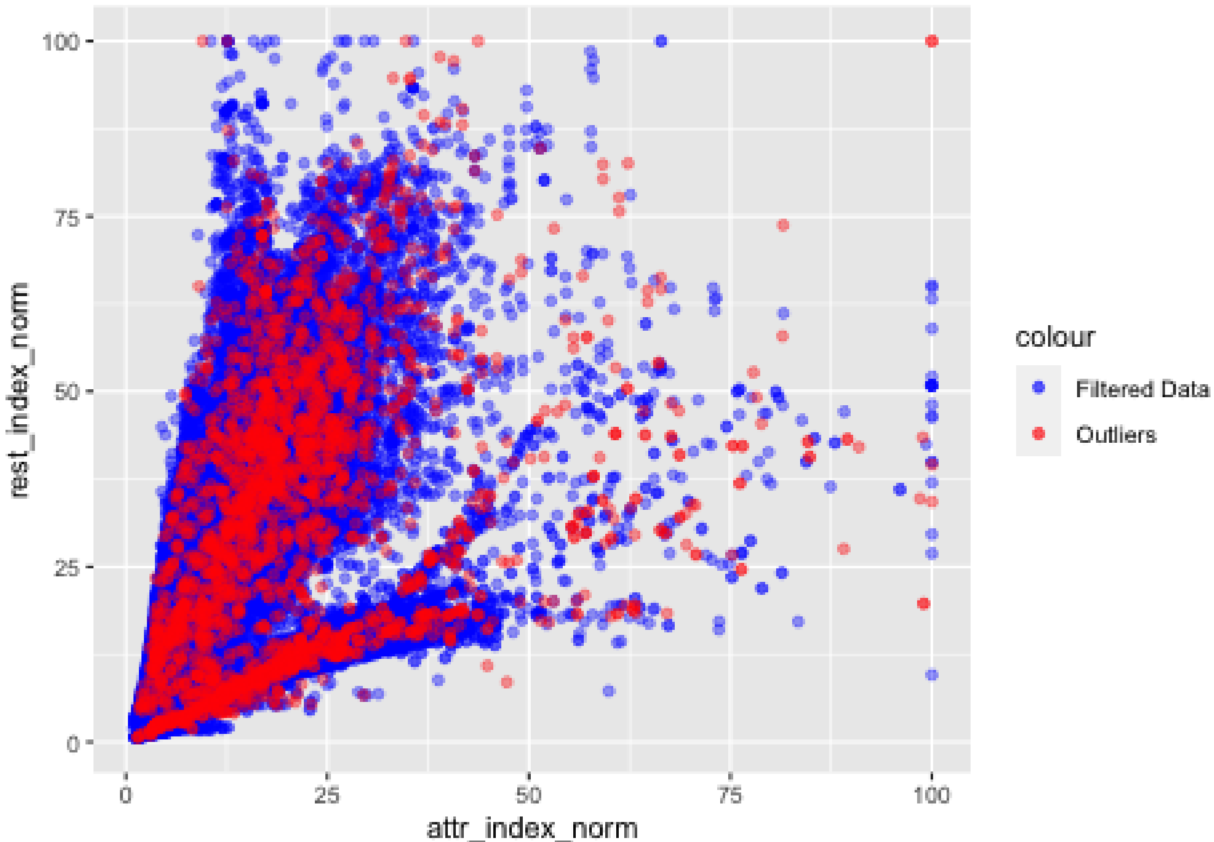

```r
ggpairs(my_data_filtered[c("realSum", "dist", "metro_dist", "attr_index_norm",
    "rest_index_norm", "city_day")], cardinality = 20, cardinality_threshold = 999)
```



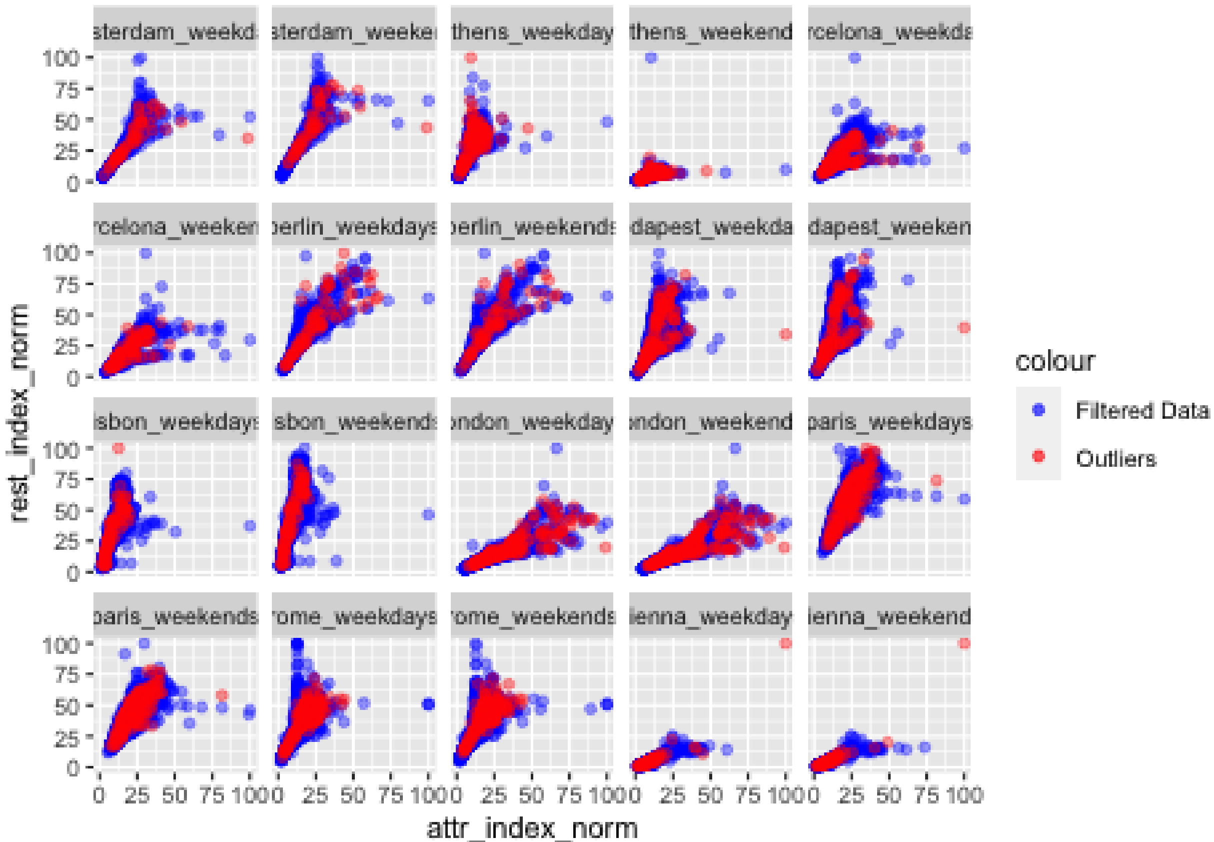

```r
cor(my_data$attr_index, my_data$rest_index)
```

```
## [1] 0.4721427
```

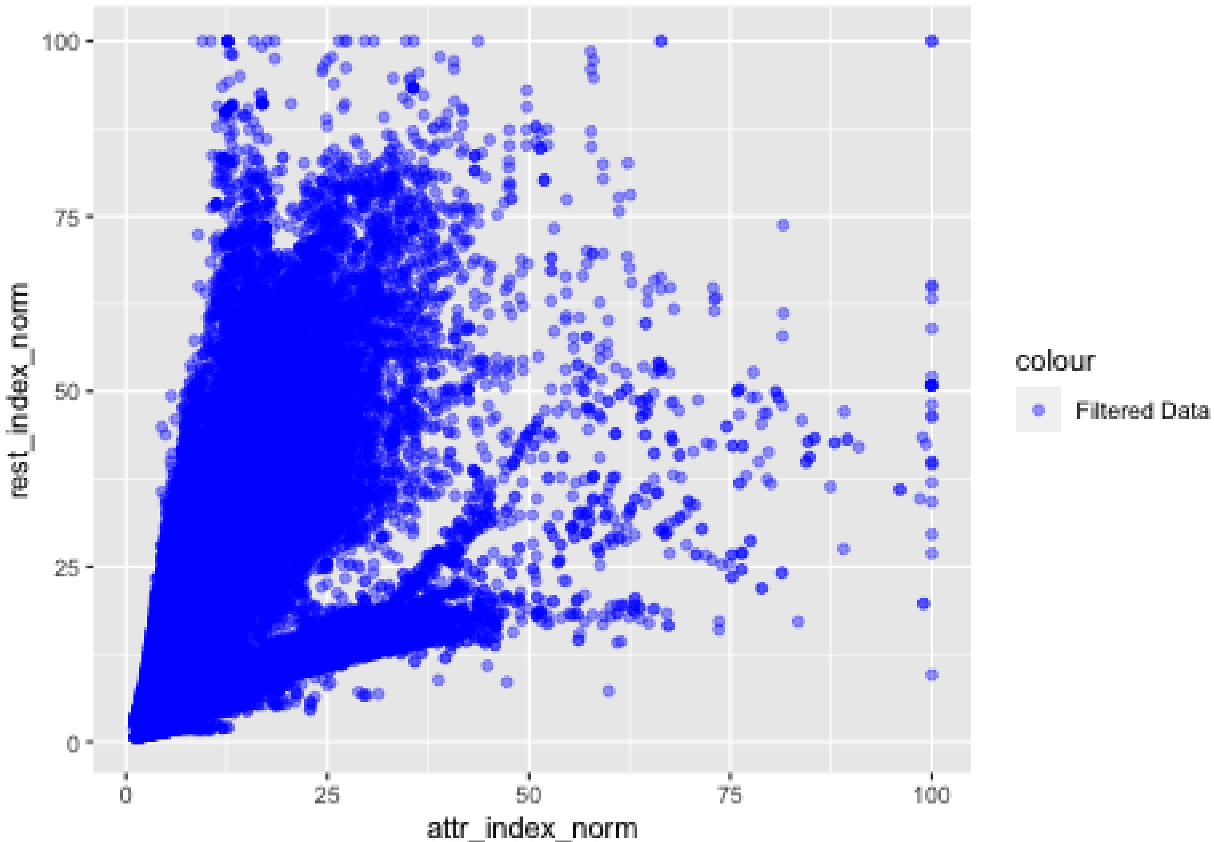
```r
ggplot() + geom_point(data = my_data_filtered, aes(x = attr_index_norm,
    y = rest_index_norm, color = "Filtered Data"), alpha = 0.4) +
    geom_point(data = my_outliers, aes(x = attr_index_norm, y = rest_index_norm,
        color = "Outliers"), alpha = 0.4) + scale_color_manual(values = c(`Filtered Data` = "blue",
    Outliers = "red"))
```



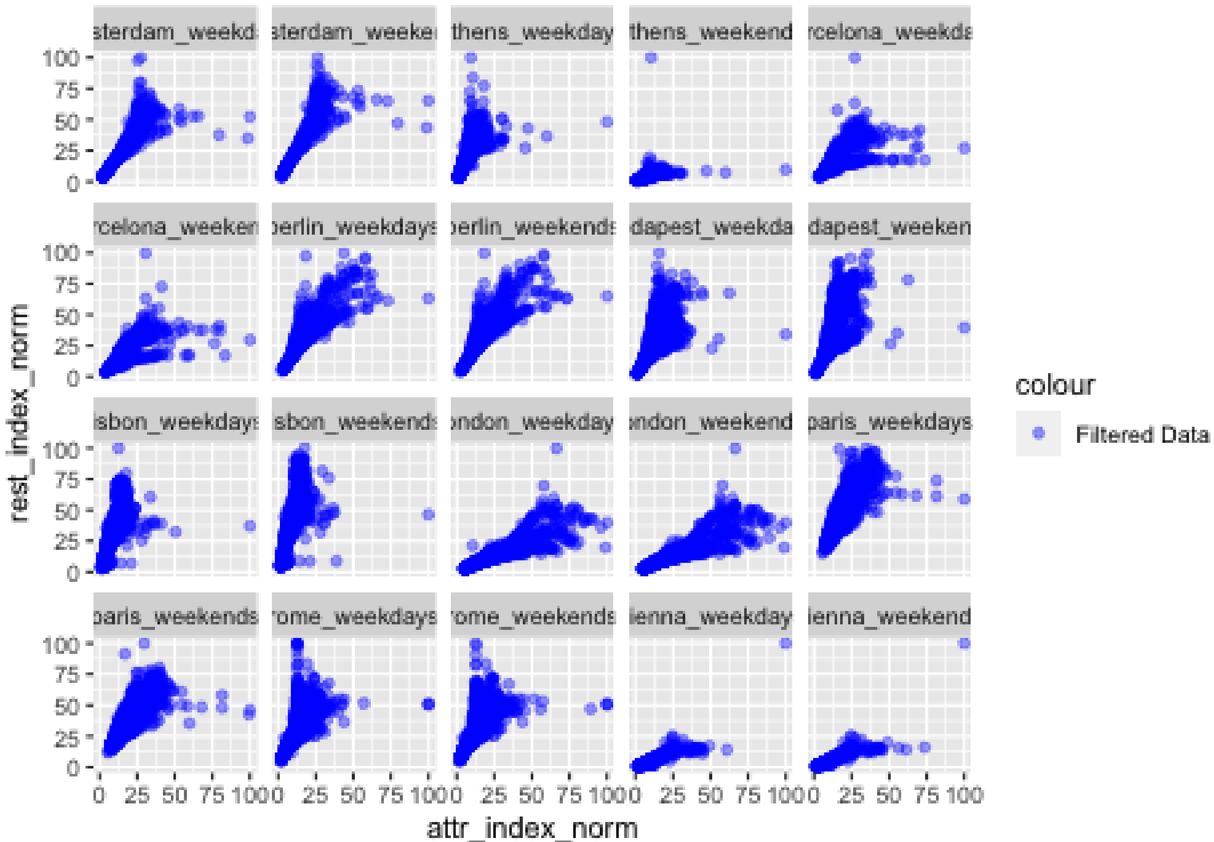

```
ggplot() + geom_point(data = my_data_filtered, aes(x = attr_index_norm,
    y = rest_index_norm, color = "Filtered Data"), alpha = 0.4) +
    geom_point(data = my_outliers, aes(x = attr_index_norm, y = rest_index_norm,
        color = "Outliers"), alpha = 0.4) + scale_color_manual(values = c(`Filtered Data` = "blue",
    Outliers = "red")) + facet_wrap(~city_day)
```




```r
ggplot() + geom_point(data = my_data, aes(x = attr_index_norm,
    y = rest_index_norm, color = "Filtered Data"), alpha = 0.4) +
    scale_color_manual(values = c(`Filtered Data` = "blue"))
```




```
ggplot() + geom_point(data = my_data, aes(x = attr_index_norm,
    y = rest_index_norm, color = "Filtered Data"), alpha = 0.4) +
    scale_color_manual(values = c(`Filtered Data` = "blue")) +
    facet_wrap(~city_day)
```




```
cor(my_data$attr_index_norm, my_data$rest_index_norm)
```

```
## [1] 0.4721427
```

Linear Regression

```
M1 <- lm(realSum ~ . - realSum, data = my_data_train)
summary(M1)
```

```
##
## Call:
## lm(formula = realSum ~ . - realSum, data = my_data_train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
##  -757.5   -84.1   -21.0    43.0 18422.1
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                -4.916e+03  3.997e+03  -1.230 0.218657
## X                           1.026e-03  1.605e-03   0.639 0.522652
## room_typePrivate room      -1.144e+02  4.282e+00 -26.703  < 2e-16 ***
## room_typeShared room       -2.043e+02  1.894e+01 -10.790  < 2e-16 ***
```

```
## person_capacity            2.396e+01  1.765e+00  13.578  < 2e-16 ***
## host_is_superhostTrue       1.152e+00  3.936e+00   0.293 0.769759
## multi                       9.638e+00  4.133e+00   2.332 0.019704 *
## biz                         3.332e+01  4.189e+00   7.954 1.85e-15 ***
## cleanliness_rating          5.045e+00  2.415e+00   2.089 0.036740 *
## guest_satisfaction_overall  7.759e-01  2.615e-01   2.968 0.003003 **
## bedrooms                    8.599e+01  3.189e+00  26.965  < 2e-16 ***
## dist                       -1.538e+00  1.263e+00  -1.218 0.223192
## metro_dist                 -3.879e+00  2.509e+00  -1.546 0.122147
## attr_index_norm             6.375e+00  2.946e-01  21.636  < 2e-16 ***
## rest_index_norm            -1.736e-01  1.781e-01  -0.975 0.329728
## lng                        -2.619e+02  4.023e+01  -6.510 7.63e-11 ***
## lat                         1.224e+02  7.653e+01   1.599 0.109884
## city_dayamsterdam_weekends  6.796e+01  1.600e+01   4.247 2.17e-05 ***
## city_dayathens_weekdays     6.283e+03  1.389e+03   4.522 6.15e-06 ***
## city_dayathens_weekends     6.271e+03  1.390e+03   4.513 6.41e-06 ***
## city_daybarcelona_weekdays  4.051e+02  8.379e+02   0.483 0.628751
## city_daybarcelona_weekends  4.231e+02  8.379e+02   0.505 0.613579
## city_dayberlin_weekdays     1.940e+03  3.424e+02   5.666 1.47e-08 ***
## city_dayberlin_weekends     1.950e+03  3.423e+02   5.697 1.23e-08 ***
## city_daybudapest_weekdays   3.883e+03  7.075e+02   5.488 4.09e-08 ***
## city_daybudapest_weekends   3.910e+03  7.075e+02   5.527 3.28e-08 ***
## city_daylisbon_weekdays    -2.311e+03  1.144e+03  -2.020 0.043372 *
## city_daylisbon_weekends    -2.302e+03  1.144e+03  -2.013 0.044164 *
## city_daylondon_weekdays    -1.407e+03  2.062e+02  -6.823 9.06e-12 ***
## city_daylondon_weekends    -1.409e+03  2.062e+02  -6.833 8.47e-12 ***
## city_dayparis_weekdays     -4.046e+02  2.785e+02  -1.453 0.146310
## city_dayparis_weekends     -4.238e+02  2.787e+02  -1.521 0.128294
## city_dayrome_weekdays       2.929e+03  8.819e+02   3.322 0.000896 ***
## city_dayrome_weekends       2.934e+03  8.819e+02   3.327 0.000879 ***
## city_dayvienna_weekdays     3.216e+03  5.832e+02   5.515 3.51e-08 ***
## city_dayvienna_weekends     3.215e+03  5.833e+02   5.511 3.59e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 305.1 on 36158 degrees of freedom
## Multiple R-squared:  0.2151, Adjusted R-squared:  0.2143
## F-statistic:   283 on 35 and 36158 DF,  p-value: < 2.2e-16
```

The r^2 and adjusted r^2 values are too low for the Linear regression model to be considered a competent one in this case.

Lasso Step Regression

```
M1_step = step(M1, direction = "backward")
```

```
## Start:  AIC=414133.9
## realSum ~ (X + room_type + person_capacity + host_is_superhost +
##     multi + biz + cleanliness_rating + guest_satisfaction_overall +
##     bedrooms + dist + metro_dist + attr_index_norm + rest_index_norm +
##     lng + lat + city_day) - realSum
```

```
##
##                              Df Sum of Sq        RSS    AIC
## - host_is_superhost           1      7973 3365094746 414132
## - X                           1     38033 3365124806 414132
## - rest_index_norm             1     88412 3365175185 414133
## - dist                        1    138090 3365224863 414133
## <none>                                   3365086773 414134
## - metro_dist                  1    222399 3365309172 414134
## - lat                         1    237878 3365324651 414134
## - cleanliness_rating          1    406023 3365492796 414136
## - multi                       1    506134 3365592907 414137
## - guest_satisfaction_overall  1    819629 3365906402 414141
## - lng                         1   3943764 3369030537 414174
## - biz                         1   5888249 3370975022 414195
## - person_capacity             1  17158129 3382244902 414316
## - attr_index_norm             1  43565736 3408652509 414598
## - bedrooms                    1  67668883 3432755656 414853
## - room_type                   2  74087785 3439174558 414918
## - city_day                   19 129073720 3494160493 415458
##
## Step:  AIC=414132
## realSum ~ X + room_type + person_capacity + multi + biz + cleanliness_rating +
##     guest_satisfaction_overall + bedrooms + dist + metro_dist +
##     attr_index_norm + rest_index_norm + lng + lat + city_day
##
##                              Df Sum of Sq        RSS    AIC
## - X                           1     37006 3365131752 414130
## - rest_index_norm             1     87739 3365182485 414131
## - dist                        1    137403 3365232149 414131
## <none>                                   3365094746 414132
## - metro_dist                  1    224011 3365318757 414132
## - lat                         1    236846 3365331592 414133
## - cleanliness_rating          1    422853 3365517599 414135
## - multi                       1    513494 3365608240 414136
## - guest_satisfaction_overall  1    846062 3365940808 414139
## - lng                         1   3943452 3369038198 414172
## - biz                         1   5880284 3370975030 414193
## - person_capacity             1  17156242 3382250987 414314
## - attr_index_norm             1  43564593 3408659339 414596
## - bedrooms                    1  67662885 3432757630 414851
## - room_type                   2  74127027 3439221773 414917
## - city_day                   19 129091002 3494185748 415457
##
## Step:  AIC=414130.4
## realSum ~ room_type + person_capacity + multi + biz + cleanliness_rating +
##     guest_satisfaction_overall + bedrooms + dist + metro_dist +
##     attr_index_norm + rest_index_norm + lng + lat + city_day
##
##                              Df Sum of Sq        RSS    AIC
## - rest_index_norm             1     99078 3365230830 414129
## - dist                        1    136528 3365268280 414130
## <none>                                   3365131752 414130
## - metro_dist                  1    238734 3365370486 414131
## - lat                         1    240263 3365372016 414131
```

```
## - cleanliness_rating          1    420972 3365552724 414133
## - multi                       1    509297 3365641049 414134
## - guest_satisfaction_overall  1    845143 3365976896 414138
## - lng                         1   3980694 3369112446 414171
## - biz                         1   5868776 3371000528 414191
## - person_capacity             1  17162129 3382293881 414313
## - attr_index_norm             1  43529070 3408660822 414594
## - bedrooms                    1  67709766 3432841518 414849
## - room_type                   2  74127155 3439258907 414915
## - city_day                   19 130900670 3496032422 415474
##
## Step:  AIC=414129.5
## realSum ~ room_type + person_capacity + multi + biz + cleanliness_rating +
##     guest_satisfaction_overall + bedrooms + dist + metro_dist +
##     attr_index_norm + lng + lat + city_day
##
##                              Df Sum of Sq        RSS    AIC
## - dist                        1     86683 3365317513 414128
## <none>                                    3365230830 414129
## - metro_dist                  1    272120 3365502950 414130
## - lat                         1    273162 3365503992 414130
## - cleanliness_rating          1    419524 3365650354 414132
## - multi                       1    494022 3365724852 414133
## - guest_satisfaction_overall  1    837305 3366068136 414136
## - lng                         1   3973420 3369204250 414170
## - biz                         1   5789568 3371020398 414190
## - person_capacity             1  17121733 3382352564 414311
## - attr_index_norm             1  54609498 3419840328 414710
## - bedrooms                    1  67904086 3433134916 414851
## - room_type                   2  74039649 3439270479 414913
## - city_day                   19 131071208 3496302038 415474
##
## Step:  AIC=414128.4
## realSum ~ room_type + person_capacity + multi + biz + cleanliness_rating +
##     guest_satisfaction_overall + bedrooms + metro_dist + attr_index_norm +
##     lng + lat + city_day
##
##                              Df Sum of Sq        RSS    AIC
## <none>                                    3365317513 414128
## - lat                         1    292396 3365609909 414130
## - cleanliness_rating          1    414092 3365731605 414131
## - multi                       1    495582 3365813095 414132
## - metro_dist                  1    614735 3365932248 414133
## - guest_satisfaction_overall  1    838098 3366155612 414135
## - lng                         1   3887405 3369204918 414168
## - biz                         1   5842686 3371160199 414189
## - person_capacity             1  17112579 3382430092 414310
## - bedrooms                    1  67830738 3433148251 414849
## - room_type                   2  74300129 3439617642 414915
## - attr_index_norm             1  86947824 3452265337 415050
## - city_day                   19 133905560 3499223073 415503
```

```
summary(M1_step)
```

```
##
## Call:
## lm(formula = realSum ~ room_type + person_capacity + multi +
##     biz + cleanliness_rating + guest_satisfaction_overall + bedrooms +
##     metro_dist + attr_index_norm + lng + lat + city_day, data = my_data_train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
##  -756.8   -84.1   -21.0    42.9 18422.9
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)                -5594.6616  3968.4120  -1.410 0.158608
## room_typePrivate room       -114.3509     4.2751 -26.748  < 2e-16 ***
## room_typeShared room        -203.5745    18.9242 -10.757  < 2e-16 ***
## person_capacity               23.9228     1.7642  13.560  < 2e-16 ***
## multi                          9.5185     4.1248   2.308 0.021024 *
## biz                           33.0599     4.1724   7.924 2.37e-15 ***
## cleanliness_rating             5.0657     2.4015   2.109 0.034916 *
## guest_satisfaction_overall     0.7802     0.2600   3.001 0.002693 **
## bedrooms                      86.0342     3.1867  26.998  < 2e-16 ***
## metro_dist                    -5.4925     2.1370  -2.570 0.010170 *
## attr_index_norm                6.3667     0.2083  30.566  < 2e-16 ***
## lng                         -257.3918    39.8246  -6.463 1.04e-10 ***
## lat                          134.7768    76.0355   1.773 0.076312 .
## city_dayamsterdam_weekends    67.1363    15.9781   4.202 2.65e-05 ***
## city_dayathens_weekdays     6380.0626  1384.2639   4.609 4.06e-06 ***
## city_dayathens_weekends     6370.9186  1384.2104   4.603 4.19e-06 ***
## city_daybarcelona_weekdays   554.5173   831.2840   0.667 0.504737
## city_daybarcelona_weekends   572.4049   831.2994   0.689 0.491101
## city_dayberlin_weekdays     1895.1920   338.3471   5.601 2.14e-08 ***
## city_dayberlin_weekends     1904.8977   338.2674   5.631 1.80e-08 ***
## city_daybudapest_weekdays   3880.0408   704.0633   5.511 3.59e-08 ***
## city_daybudapest_weekends   3906.3502   704.0708   5.548 2.91e-08 ***
## city_daylisbon_weekdays    -2077.4612  1130.8273  -1.837 0.066201 .
## city_daylisbon_weekends    -2069.6484  1130.8363  -1.830 0.067229 .
## city_daylondon_weekdays    -1373.5039   204.1097  -6.729 1.73e-11 ***
## city_daylondon_weekends    -1375.1043   204.1230  -6.737 1.65e-11 ***
## city_dayparis_weekdays      -354.6025   276.2027  -1.284 0.199203
## city_dayparis_weekends      -371.8162   276.2127  -1.346 0.178271
## city_dayrome_weekdays       3026.1639   878.2040   3.446 0.000570 ***
## city_dayrome_weekends       3031.0164   878.2293   3.451 0.000559 ***
## city_dayvienna_weekdays     3219.2260   580.2311   5.548 2.91e-08 ***
## city_dayvienna_weekends     3217.6304   580.3196   5.545 2.97e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 305.1 on 36162 degrees of freedom
## Multiple R-squared:  0.215,  Adjusted R-squared:  0.2143
## F-statistic: 319.5 on 31 and 36162 DF,  p-value: < 2.2e-16
```

The stepwise regression removed - host_is_superhost, rest_index_norm, dist

```r
lm_pred <- predict(M1_step, newdata = my_data_test)
y_test <- my_data_test$realSum

RMSE = sqrt(mean((y_test - lm_pred)^2))
RMSE
```

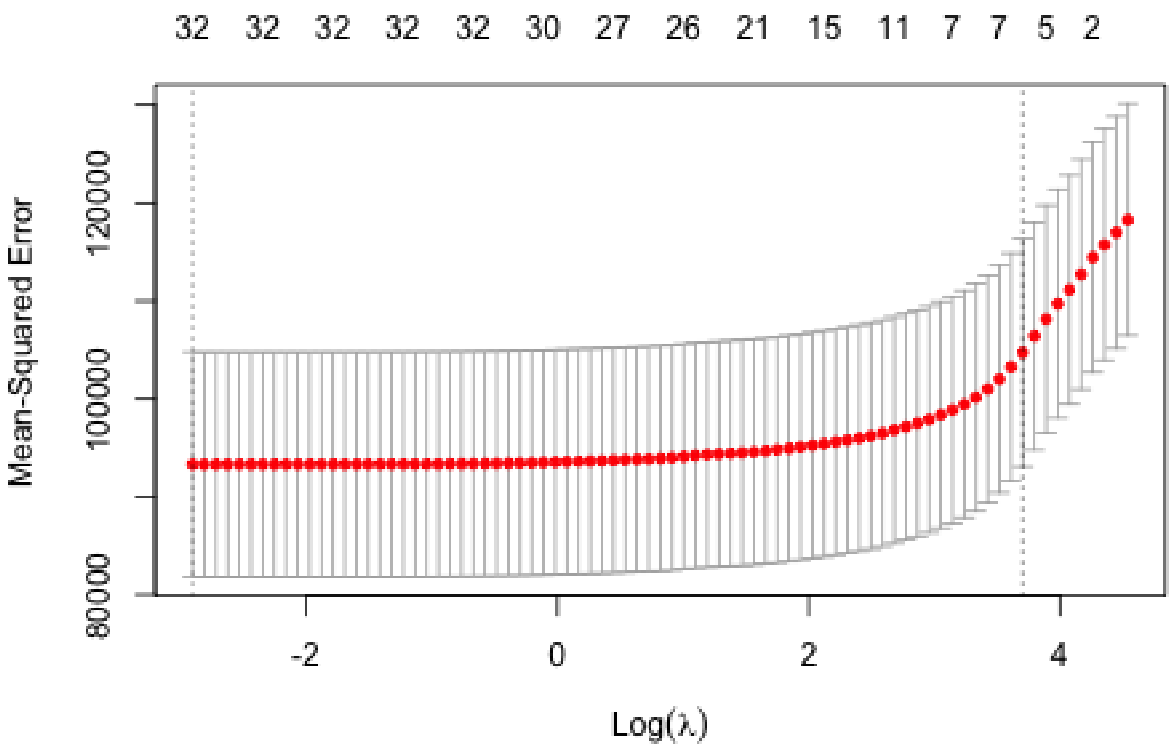
```
## [1] 233.2787
```

```r
# Prepare the predictors and response variable
x_train <- model.matrix(realSum ~ room_type + person_capacity +
    host_is_superhost + multi + biz + cleanliness_rating + guest_satisfaction_overall +
    bedrooms + dist + metro_dist + attr_index_norm + rest_index_norm +
    lng + lat + city_day, data = my_data_train)[, -1]
y_train <- my_data_train$realSum

# Fit a Lasso regression model
lasso_model <- glmnet(x_train, y_train, alpha = 1)

# Select the best lambda value using cross-validation
cv_model <- cv.glmnet(x_train, y_train, alpha = 1, nfolds = 5)

# Plot the cross-validation results
plot(cv_model)
```




```r
# Select the lambda value that minimizes the mean
# cross-validation error
best_lambda <- cv_model$lambda.min

# Fit a Lasso regression model with the selected lambda
```

```r
# value
lasso_model_best <- glmnet(x_train, y_train, alpha = 1, lambda = best_lambda)

# Calculate R-squared and multiple R-squared
y_train_pred <- predict(lasso_model_best, newx = x_train)
y_train_mean <- mean(y_train)
SST <- sum((y_train - y_train_mean)^2)
SSR <- sum((y_train - y_train_pred)^2)
R_squared <- 1 - SSR/SST
multiple_R_squared <- cor(y_train_pred, y_train)^2
n <- length(y_train)
p <- ncol(x_train)
adj_R_squared <- 1 - (SSR/(n - p - 1))/(SST/(n - 1))


# Print the R-squared and multiple R-squared values
cat("R-squared:", round(R_squared, 3), "\n")
```

```
## R-squared: 0.214
```

```r
cat("Multiple R-squared:", round(multiple_R_squared, 3), "\n")
```

```
## Multiple R-squared: 0.214
```

```r
cat("Adjusted R-squared:", round(adj_R_squared, 3), "\n")
```

```
## Adjusted R-squared: 0.213
```

```r
# Predict on the test data
x_test <- model.matrix(realSum ~ room_type + person_capacity +
    host_is_superhost + multi + biz + cleanliness_rating + guest_satisfaction_overall +
    bedrooms + dist + metro_dist + attr_index_norm + rest_index_norm +
    lng + lat + city_day, data = my_data_test)[, -1]
y_test <- my_data_test$realSum
lasso_pred <- predict(lasso_model_best, newx = x_test)

# Evaluate the model performance
rmse <- sqrt(mean((y_test - lasso_pred)^2))
cat("RMSE on test set:", round(rmse, 3), "\n")
```

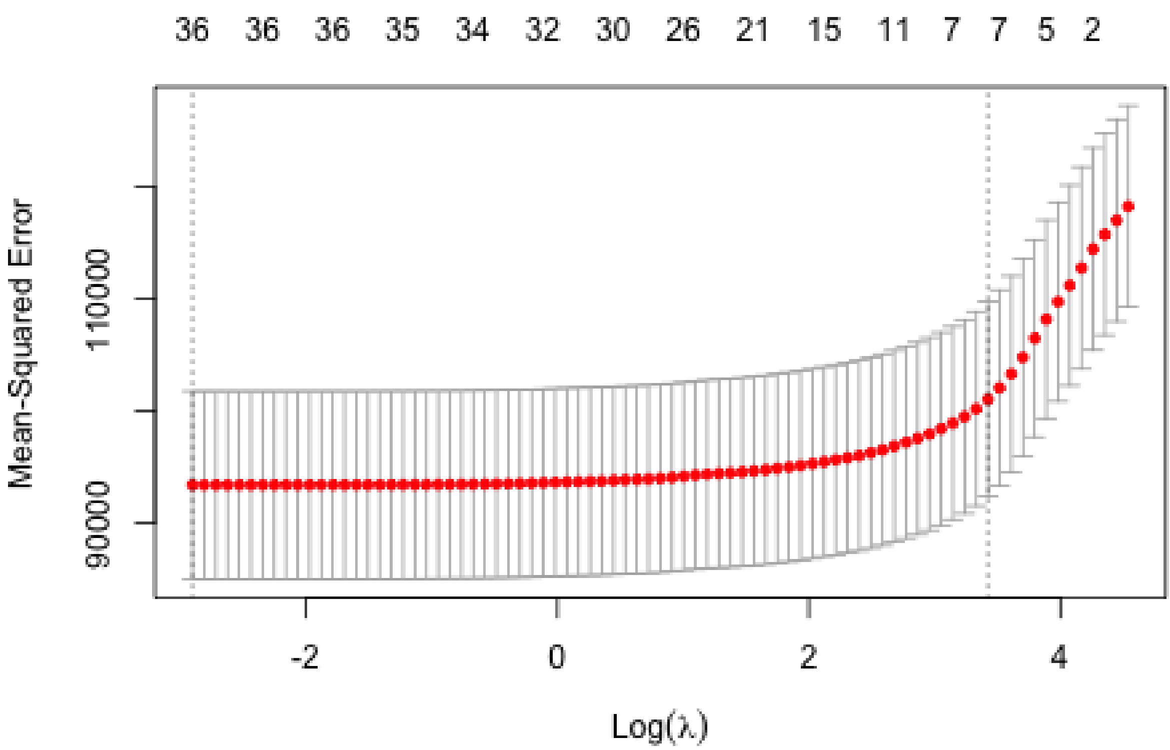
```
## RMSE on test set: 233.534
```

```r
# Prepare the predictors and response variable
x_train <- model.matrix(realSum ~ room_type + person_capacity +
    host_is_superhost + multi + biz + cleanliness_rating + guest_satisfaction_overall +
    bedrooms + poly(dist, 2) + poly(metro_dist, 2) + poly(attr_index_norm,
    2) + poly(rest_index_norm, 2) + lng + lat + city_day, data = my_data_train)[,
    -1]
y_train <- my_data_train$realSum
```

```r
# Fit a Lasso regression model
lasso_model <- glmnet(x_train, y_train, alpha = 1)

# Select the best lambda value using cross-validation
cv_model <- cv.glmnet(x_train, y_train, alpha = 1, nfolds = 5)

# Plot the cross-validation results
plot(cv_model)
```




```r
# Select the lambda value that minimizes the mean
# cross-validation error
best_lambda <- cv_model$lambda.min

# Fit a Lasso regression model with the selected lambda
# value
lasso_model_best <- glmnet(x_train, y_train, alpha = 1, lambda = best_lambda)

# Calculate R-squared and multiple R-squared
y_train_pred <- predict(lasso_model_best, newx = x_train)
y_train_mean <- mean(y_train)
SST <- sum((y_train - y_train_mean)^2)
SSR <- sum((y_train - y_train_pred)^2)
R_squared <- 1 - SSR/SST
multiple_R_squared <- cor(y_train_pred, y_train)^2
n <- length(y_train)
p <- ncol(x_train)
adj_R_squared <- 1 - (SSR/(n - p - 1))/(SST/(n - 1))
```

```r
# Print the R-squared and multiple R-squared values
cat("R-squared:", round(R_squared, 3), "\n")
```

```
## R-squared: 0.214
```

```r
cat("Multiple R-squared:", round(multiple_R_squared, 3), "\n")
```

```
## Multiple R-squared: 0.214
```

```r
cat("Adjusted R-squared:", round(adj_R_squared, 3), "\n")
```

```
## Adjusted R-squared: 0.213
```

```r
# Predict on the test data
x_test <- model.matrix(realSum ~ room_type + person_capacity +
    host_is_superhost + multi + biz + cleanliness_rating + guest_satisfaction_overall +
    bedrooms + poly(dist, 2) + poly(metro_dist, 2) + poly(attr_index_norm,
    2) + poly(rest_index_norm, 2) + lng + lat + city_day, data = my_data_test)[,
    -1]
y_test <- my_data_test$realSum
lasso_pred <- predict(lasso_model_best, newx = x_test)

# Evaluate the model performance
rmse <- sqrt(mean((y_test - lasso_pred)^2))
# Print the R-squared and multiple R-squared values
cat("RMSE on test set:", round(rmse, 3), "\n")
```

```
## RMSE on test set: 235.35
```

Even Lasso regression is not good because of extremely low value of R^2 even in polynominal model of power 2 and 3.

Conclusion at this Point in Time

Even though EDA has given us good insights in price determinants, both Linear and Lasso Step Regression are not good for this case which is to be expected since all common data tends to be generally skewed Normal or Guassian(to be tested).

Further modelling is required and will be conducted which includes trying of different linear techniques and also models from different family.