Mohammad Javad Ranjbar

810101173

Project

Statistical Inference, Fall 2022

Contents

Note:

In the R code I used couple of libraries that are not installed by default. If you want to run the code, please check and uncomment the install.pakges(“”) in my code.

Question 0:

A) Airbnb is a service that helps house owners to rent out their properties to people who need a temporary place to stay. Airbnb does not own any of these houses, it profits by receiving a commission from each booking. This service has grown larger every year since 2008.
This dataset contains these data:

Field	Type	Description
Id	Integer	Airbnb's unique identifier for the listing.
Name	Text	The property name.
Host id	Integer	Airbnb's unique identifier for the owners.
Host identify verified	Text or Boolean(t=true; f=false)	Airbnb hosts will have the option of requiring that all their guests verify their identification before booking a reservation. Hosts who choose this option will be required to verify their own identification as well.
Host name	Text	Name of the host. Usually just the first name(s).
Neighborhood group	Text	The neighborhood group as geocoded using the latitude and longitude against neighborhoods as defined by open or public digital shapefiles.
Neighborhood	Text	The neighborhood as geocoded using the latitude and longitude against neighborhoods as defined by open or public digital shapefiles.
Lat (latitude)	Numeric	Uses the World Geodetic System (WGS84) projection for latitude and longitude.
Long (longitude)	numeric	Uses the World Geodetic System (WGS84) projection for latitude and longitude.
Country	Text	The country where the house is located.
Instant bookable	boolean	[t=true; f=false]. Whether the guest can automatically book the listing without the host requiring to accept their booking request. An indicator of a commercial listing.
cancellation_policy	Text	There are 10 cancellation policies as followed: Flexible Moderate Firm Strict Strict Long term Flexible Long term Non-refundable option Super Strict 30 days Super Strict 60 days Special cases However, this dataset only contains 3 of these policies:

		Flexible: Guests can cancel until 24 hours before check-in for a full refund, and you won't be paid. If they cancel after that, you'll be paid for each night they stay, plus 1 additional night. Moderate: Guests can cancel until 5 days before check-in for a full refund, and you won't be paid. If they cancel after that, you'll be paid for each night they stay, plus 1 additional night, plus 50% for all unspent nights. Strict: To receive a full refund, guests must cancel within 48 hours of booking, and the cancellation must occur at least 14 days before check-in. If they cancel between 7 and 14 days before check-in, you'll be paid 50% for all nights. If they cancel after that, you'll be paid 100% for all nights.
Room type	Text	All homes are grouped into the following three room types: Entire place Private room Shared room Entire place: Entire places are best if you're seeking a home away from home. With an entire place, you'll have the whole space to yourself. This usually includes a bedroom, a bathroom, a kitchen, and a separate, dedicated entrance. Hosts should note in the description if they'll be on the property or not (ex: "Host occupies first floor of the home"), and provide further details on the listing. Private rooms: Private rooms are great for when you prefer a little privacy, and still value a local connection. When you book a private room, you'll have your own private room for sleeping and may share some spaces with others. You might need to walk through indoor spaces that another host or guest may occupy to get to your room. Shared rooms: Shared rooms are for when you don't mind sharing a space with others. When you book a shared room, you'll be sleeping in a space that is shared with others and share the entire space with other people. Shared rooms are popular among flexible travelers looking for new friends and budget-friendly stays.
Construction year	Integer	The year the house was constructed.
Price	Integer	daily price in local currency. ($ sign may be used despite local.)
service fee	Integer	To help Airbnb run smoothly and to cover the cost of services like 24/7 customer support, Airbnb charges a service fee when a booking is confirmed.
minimum nights	Integer	minimum number of night stay for the listing (calendar rules may be different).
number of reviews	Integer	The number of reviews the listing has over the lifetime of the listing.

reviews per month	Numeric	The average number of reviews that the listing get in each month.
review rate number	Integer	Average rating of the location.
calculated host listings count	integer	The number of listings the host has in the current scrape, in the city/region geography.
availability 365	integer	avaliability_x. The availability of the listing x days in the future as determined by the calendar. (listing may be available because it has been booked by a guest or blocked by the host)

Why studying the Airbnb dataset can be interesting?

This dataset can give various insights in matters such as:

- What features are more important for people to choose their preferred house?
    - Does neighborhood play a role in choosing houses?
        - Are some Neighborhoods cheaper?
        - Are neighborhoods that are close to tourist attractions more popular?
        - Do houses located in some neighborhoods get more visitors? And Which neighborhoods are more popular?
    - What range or price is more acceptable for people?
        - Do people prefer to go to cheaper houses or rating matters most?
        - Do people care about service fees and if it is expensive is it a deal breaker?
    - Does house rating affect people's decisions?
        - Do more expensive houses get a better rating?
        - Does the number of previous visitors and their rating matter to people?
    - What kind of house do people prefer to stay in?
        - Does the Construction year matter?
        - What kind of Room type is more popular among renters? Do people prefer to get a more expensive house to get more privacy?
        - What kind of policies drive people away?
            - Does Instance booking matter?
            - Does availability 365 matter?
            - Are people ok with strict cancellation policies?
- Property owners prefer to provide what kind of services?
    - Do property owners prefer to be strict on their policies?
    - Do property owners prefer to verify their identification? And does it affects their visitors?
    - Do people in more poor Neighborhoods tend to rent part of their home?
    - What is the range of price preferred for property owners of each area?

Also, we can use these data in combination whit users' data to provide a good house recommendation for them.

B) This dataset has 23 columns. Six of these columns are useless and do not have any practical application. These columns are the entry's id, name of the place, name of the owner, the host id, id of the place, and the country (All of the houses are in America, obviously). Therefore, we have 17 features and 30000 cases.

```
> df <- read.csv(file = 'Airbnb_Open_Data.csv')
> df <- df[ , ! names(df) %in% c("X","NAME","id","host.id","host.name","country")]
> n_col<- ncol(df)
> n_row<- nrow(df)
> n_col
[1] 17
> n_row
[1] 30000
```

It should be mentioned that there are some duplicate data in this dataset after deleting them we will have 29957 cases.

```
> df<-df[!duplicated(df), ]
> n_row<- nrow(df)
> n_row
[1] 29957
```

C) First off, we need to analyze if the values are correct in each feature:

```
> summary(df)
 host_identity_verified neighbourhood.group neighbourhood           lat             long         instant_bookable   cancellation_policy
 Length:29957           Length:29957        Length:29957       Min.   :40.51   Min.   :-74.24   Length:29957       Length:29957
 Class :character       Class :character    Class :character   1st Qu.:40.69   1st Qu.:-73.98   Class :character   Class :character
 Mode  :character       Mode  :character    Mode  :character   Median :40.72   Median :-73.95   Mode  :character   Mode  :character
                                                               Mean   :40.73   Mean   :-73.95
                                                               3rd Qu.:40.76   3rd Qu.:-73.93
                                                               Max.   :40.91   Max.   :-73.71

  room.type         Construction.year     price          service.fee     minimum.nights      number.of.reviews reviews.per.month
 Length:29957       Min.   :2003      Min.   :  50.0   Min.   : 10.0   Min.   :-1223.000   Min.   :  0.00    Min.   : 0.010
 Class :character   1st Qu.:2008      1st Qu.: 339.0   1st Qu.: 68.0   1st Qu.:    2.000   1st Qu.:  1.00    1st Qu.: 0.220
 Mode  :character   Median :2012      Median : 627.0   Median :125.0   Median :    3.000   Median :  7.00    Median : 0.750
                    Mean   :2012      Mean   : 625.8   Mean   :125.2   Mean   :    8.187   Mean   : 27.57    Mean   : 1.377
                    3rd Qu.:2017      3rd Qu.: 911.0   3rd Qu.:182.0   3rd Qu.:    5.000   3rd Qu.: 31.00    3rd Qu.: 2.020
                    Max.   :2022      Max.   :1200.0   Max.   :240.0   Max.   : 5645.000   Max.   :884.00    Max.   :65.740
                    NA's   :66        NA's   :76       NA's   :93      NA's   :119         NA's   :58        NA's   :4640
 review.rate.number calculated.host.listings.count availability.365
 Min.   :1.000      Min.   :  1.000                Min.   :-10.0
 1st Qu.:2.000      1st Qu.:  1.000                1st Qu.:  3.0
 Median :3.000      Median :  1.000                Median : 95.0
 Mean   :3.276      Mean   :  7.843                Mean   :139.7
 3rd Qu.:4.000      3rd Qu.:  2.000                3rd Qu.:264.0
 Max.   :5.000      Max.   :332.000                Max.   :426.0
 NA's   :102        NA's   :100                    NA's   :120
```

It is obvious in the above figure. There are some values that are inherently wrong:

- The minimum nights feature is inherently positive and should not have a case with a negative value.
- For the minimum nights feature great values, such as 5645 are highly unlikely. (It does not make sense to rent the house for 5645 days on minimum)
- Availability 365 is inherently positive and should not have a case with a negative value.
- Availability 365 means the availability of the listing 365 days in the future as determined by the calendar. Cases with values more than 365 are unlikely to be the correct values.
- For reviews per month features, cases with a value of more than 30 do not really make any sense. Because it means more than 30 people have stayed there in 30 days. Which is not possible. However, it is possible that people stayed there in groups of more than 1 person. But it is still unlikely that every person who stayed there wrote a review. (I checked the actual place on the Airbnb website that had 65.74 reviews per month and it only has 973 reviews as of right now. Which, confirms that the data is incorrect or it was just a temporary spike.). I am suspecting most of the cases with more than 10 reviews per month are incorrect but I keep these data for sake of having some outliers for future parts of this project.

For correction of the incorrect values, we can give them Nan values and fix them with other Nan values later, or we can just guess what was the correct value. We use the first approach:

- The minimum nights:

Most of the data is between 1 and 30. However, Some of the values of more than 30 can make sense. For example, People might want to rent their house for the whole summer to go on vacation (There is a spike in 90 days which might be caused by the explained theory).

```
> table(df$minimum.nights)

 -1223   -10     1     2     3     4     5     6     7     8     9    10    11    12    13    14    15    16    17    18    19
     1     1  7391  6875  4727  1983  1780   431  1206    70    46   275    12    57    35   311   160    10     8    12     5
    20    21    22    23    24    25    26    27    28    29    30    31    32    33    35    37    39    40    42    44    45
   127    63     6     6     6    46    11    13   136   158  3369   135     7     3    12     1     1     5     1     1    23
    47    48    50    53    55    56    57    59    60    62    64    65    70    75    80    81    85    87    88    90    99
     3     1     7     1     5     2     1     3    68     1     1     1     2     5    11     1     2     1     1    79     1
   100   105   110   115   120   129   133   134   144   145   150   175   180   182   183   185   210   240   265   270   273
     8     1     1     3    21     1     1     1     1     5     4     1    20     2     2     1     1     1     1     3     1
   300   350   360   364   365   366   452   458   500   954  1000  2645  5645
     7     1     4     2    13     1     1     1     1     1     1     1     1
```

Nevertheless, we exclude the values outside this range by replacing them with Nans.

```
> df['minimum.nights'][df['minimum.nights'] < 0]=NA
> df['minimum.nights'][df['minimum.nights'] >31]=NA
```

- Availability 365:
  Most of the data is between 1 and 365. we exclude the values outside this range by replacing them with Nans.

```
> table(df$availability.365)

 -10   -9   -8   -7   -6   -5   -4   -3   -2   -1    0    1    2    3    4    5    6    7    8    9   10   11   12   13   14 
  13   15   11    9   12   17   10   11   15   11 6875  215  138  191  131  166  127  124  119  117   93   97   87   93  112 
  15   16   17   18   19   20   21   22   23   24   25   26   27   28   29   30   31   32   33   34   35   36   37   38   39 
  87  105   84   62   77   86   82   71   84   75   46   78   64   87   73   97   97   90   64   89  100   83   98   90   69 
  40   41   42   43   44   45   46   47   48   49   50   51   52   53   54   55   56   57   58   59   60   61   62   63   64 
  69   54   62   65   64   62   63   61   72   58   69   62   48   75   76   78   63   78   72   69   43   78   83  106   75 
  65   66   67   68   69   70   71   72   73   74   75   76   77   78   79   80   81   82   83   84   85   86   87   88   89 
  71   91   72   66   57   76   62   63   51   58   66   60   64   63   51   73   59   58  117   74   70   72   88  144  217 
  90   91   92   93   94   95   96   97   98   99  100  101  102  103  104  105  106  107  108  109  110  111  112  113  114 
 179   51   49   43   49   61   50   39   49   36   36   35   30   43   49   47   33   30   31   37   37   28   53   36   36 
 115  116  117  118  119  120  121  122  123  124  125  126  127  128  129  130  131  132  133  134  135  136  137  138  139 
  48   30   41   36   33   45   44   30   40   47   65   48   40   53   56   35   42   47   37   34   38   47   47   30   33 
 140  141  142  143  144  145  146  147  148  149  150  151  152  153  154  155  156  157  158  159  160  161  162  163  164 
  50   43   41   41   42   42   50   45   32   40   42   58   50   52   61   67   70   78   69   42   52   47   46   49   59 
 165  166  167  168  169  170  171  172  173  174  175  176  177  178  179  180  181  182  183  184  185  186  187  188  189 
  46   48   59   57   44   60   56   46   84   56   66   53   85   67  202  189   31   38   23   35   51   30   23  103   79 
 190  191  192  193  194  195  196  197  198  199  200  201  202  203  204  205  206  207  208  209  210  211  212  213  214 
  38   29   43   35   21   26   31   38   28   36   33   45   33   41   31   29   36   32   42   23   22   46   30   34   32 
 215  216  217  218  219  220  221  222  223  224  225  226  227  228  229  230  231  232  233  234  235  236  237  238  239 
  42   42   38   36   51   53   35   30   38   30   39   25   27   30   32   46   43   37   39   37   34   25   37   39   38 
 240  241  242  243  244  245  246  247  248  249  250  251  252  253  254  255  256  257  258  259  260  261  262  263  264 
  32   43   39   44   39   39   52   58   50   71   49   34   43   33   29   49   35   38   42   35   35   36   34   45   49 
 265  266  267  268  269  270  271  272  273  274  275  276  277  278  279  280  281  282  283  284  285  286  287  288  289 
  37   34   46   43   54   67   41   42   31   35   52   29   59   45   35   50   55   35   45   48   34   37   31   48   36 
 290  291  292  293  294  295  296  297  298  299  300  301  302  303  304  305  306  307  308  309  310  311  312  313  314 
  36   38   42   43   51   33   36   55   43   46   39   49   45   33   39   45   47   51   66   60   76   91   60   44   44 
 315  316  317  318  319  320  321  322  323  324  325  326  327  328  329  330  331  332  333  334  335  336  337  338  339 
  44   36   45   47   45   49   39   49   57   77   70   50   63   50   67   63   60   87   74   50   62   59   68  102   83 
 340  341  342  343  344  345  346  347  348  349  350  351  352  353  354  355  356  357  358  359  360  361  362  363  364 
  52  107  103   60   53   69   55   67   60   56   65   77   53   73   62   65   64   71  116   79   75   77  104  143  292 
 365  366  367  368  369  370  371  372  373  374  375  376  377  378  379  380  381  382  383  384  385  386  387  388  389 
 757    7   13   14   19   14   13   11   15   14    9   21   14   11   15   18   22   11   13   21   11    9   11   14   12 
 390  391  392  393  394  395  396  397  398  399  400  401  402  403  404  405  406  407  408  409  410  411  412  413  414 
   9    6   13    8   11   13   13   11   10   12   19    8    9   12    9   13   16   10   11   13   13   16    8   13   15 
 415  416  417  418  419  420  421  422  423  424  425  426 
  15   24   13   12   11   10   13   14   16   13   11   13 
```

```
> df['availability.365'][df['availability.365'] < 0]=NA
> df['availability.365'][df['availability.365'] >366]=NA
```

- Reviews per month:
  we exclude the values outside the range of 0 to 30 by replacing them with Nans.

```
> df['reviews.per.month'][df['reviews.per.month'] >31]=NA
```

In addition to the above problems. Some features that are in text format have some incorrect or missing values.

- The neighborhood group:
  In this feature, there are 13 cases with missing values (because it is in text format df summary did not show this). Also, in some cases, Manhattan is misspelled.

```
> df['reviews.per.month'][df['reviews.per.month'] >31]=NA
> table(df$neighbourhood.group)

                  Bronx      Brooklyn      manhatan     Manhattan        Queens Staten Island 
           13       797         12201             1         12770          3897           278 
```

- The neighborhood:
  In this feature, there are 5 cases with missing values.

```
> table(df$neighbourhood)[0:5]

                     Allerton Arden Heights      Arrochar       Arverne 
               5           25             5            14            73 
```

- cancellation_policy:
  In this feature, there are 26 cases with missing values.

```
> table(df$cancellation_policy)

          flexible moderate   strict 
       26    10009    10009     9913 
> df['cancellation_policy'][df['cancellation_policy']== '']=NA
```

- host_identity_verified:
  In this feature, there are 92 cases with missing values.

```
> table(df$host_identity_verified)

          unconfirmed     verified 
       92       14936        14929 
> df['host_identity_verified'][df['host_identity_verified']== '']=NA
```

- instant_bookable:
  In this feature, there are 39 cases with missing values.

```
> table(df$instant_bookable)

        False  True
   39 15099 14819
> df['instant_bookable'][df['instant_bookable']== '']=NA
```

After deleting incorrect data our dataset would look like this:

```
> summary(df)
 host_identity_verified neighbourhood.group neighbourhood          lat             long        instant_bookable   cancellation_policy
 Length:29957           Length:29957        Length:29957       Min.   :40.51   Min.   :-74.24   Length:29957       Length:29957
 Class :character       Class :character    Class :character   1st Qu.:40.69   1st Qu.:-73.98   Class :character   Class :character
 Mode  :character       Mode  :character    Mode  :character   Median :40.72   Median :-73.95   Mode  :character   Mode  :character
                                                               Mean   :40.73   Mean   :-73.95
                                                               3rd Qu.:40.76   3rd Qu.:-73.93
                                                               Max.   :40.91   Max.   :-73.71

  room.type         Construction.year     price          service.fee     minimum.nights   number.of.reviews reviews.per.month
 Length:29957       Min.   :2003      Min.   :  50.0   Min.   : 10.0   Min.   : 1.000   Min.   :  0.00    Min.   : 0.010
 Class :character   1st Qu.:2008      1st Qu.: 339.0   1st Qu.: 68.0   1st Qu.: 1.000   1st Qu.:  1.00    1st Qu.: 0.220
 Mode  :character   Median :2012      Median : 627.0   Median :125.0   Median : 3.000   Median :  7.00    Median : 0.750
                    Mean   :2012      Mean   : 625.8   Mean   :125.2   Mean   : 6.625   Mean   : 27.57    Mean   : 1.375
                    3rd Qu.:2017      3rd Qu.: 911.0   3rd Qu.:182.0   3rd Qu.: 5.000   3rd Qu.: 31.00    3rd Qu.: 2.020
                    Max.   :2022      Max.   :1200.0   Max.   :240.0   Max.   :31.000   Max.   :884.00    Max.   :25.230
                    NA's   :66        NA's   :76       NA's   :93      NA's   :487      NA's   :58        NA's   :4641
 review.rate.number calculated.host.listings.count availability.365
 Min.   :1.000      Min.   :  1.000                Min.   :  0.0
 1st Qu.:2.000      1st Qu.:  1.000                1st Qu.:  3.0
 Median :3.000      Median :  1.000                Median : 90.0
 Mean   :3.276      Mean   :  7.843                Mean   :133.4
 3rd Qu.:4.000      3rd Qu.:  2.000                3rd Qu.:249.0
 Max.   :5.000      Max.   :332.000                Max.   :366.0
 NA's   :102        NA's   :100                    NA's   :1025
```

There are missing data in 14 features.

```
> sum(colSums(is.na(df))>0)
[1] 14
> names(which(colSums(is.na(df))>0))
 [1] "host_identity_verified"         "neighbourhood.group"            "neighbourhood"                  "instant_bookable"
 [5] "cancellation_policy"            "Construction.year"              "price"                          "service.fee"
 [9] "minimum.nights"                 "number.of.reviews"              "reviews.per.month"              "review.rate.number"
[13] "calculated.host.listings.count" "availability.365"
```

The construction year has 66 nulls which represent 0.2203158% of the total rows. The price has 76 nulls which represent 0. 2536970% of the total rows. The service fee has 93 nulls which represent 0.3104450% of the total rows. Minimum nights has 487 nulls which represent 1.6256635% of the total rows. The number of reviews has 58 nulls which represent 0. 1936108% of the total rows. Reviews per month has 4641 nulls which represent 15.49220549% of the total rows. The review rate number has 102 nulls which represent 0.34048803% of the total rows. The calculated host listings count has 100 nulls which represent 0.3338118% of the total rows. Availability 365 has 1025 nulls which represent 3.4215709% of the total rows. The neighborhood group has 14 nulls which represent 0.04339553% of the total rows. The neighborhood has 5 nulls which represent 0.01669059% of the total rows. The instant bookable has 39 nulls which represent 0.13018660% of the total rows. The cancellation policy has 26 nulls which represent 0.08679107% of the total rows. The host_identity_verified has 92 nulls which represent 0.30710685% of the total rows.

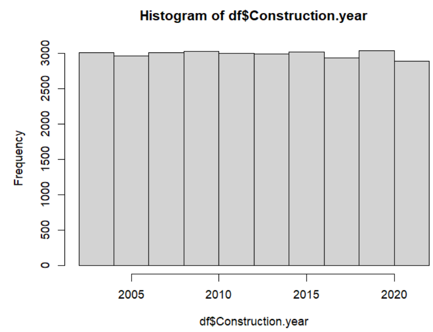
Other features do not have any missing data.
The portion of missing values for each variable is as followed:

```
> na_count <-sapply(df, function(y) sum(length(which(is.na(y)))))/n_row*100
> na_count
        host_identity_verified            neighbourhood.group                  neighbourhood                            lat
                    0.30710685                     0.04339553                     0.01669059                     0.00000000
                          long               instant_bookable            cancellation_policy                      room.type
                    0.00000000                     0.13018660                     0.08679107                     0.00000000
             Construction.year                          price                    service.fee                 minimum.nights
                    0.22031579                     0.25369697                     0.31044497                     1.62566345
             number.of.reviews              reviews.per.month             review.rate.number calculated.host.listings.count
                    0.19361084                    15.49220549                     0.34048803                     0.33381180
              availability.365
                    3.42157092
```

For handling missing values in each column we should analyze the column based on its real-world meaning. We can replace these nan values with an educated guess or we can just dispose of any case with a nan value. but the second option is not a good choice because in such a way we lose the information. Here, we use the first approach and replace these data:

- Construction year:

```
> summary(df$Construction.year)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's 
   2003    2008    2012    2012    2017    2022      66 
> hist(df$Construction.year, breaks = 7)
```

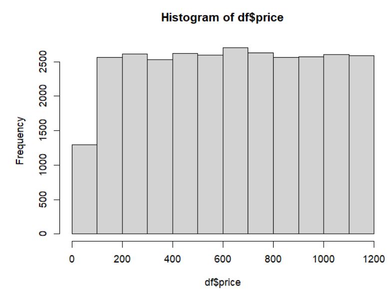


The construction year feature has an almost even distribution in the span of 2022 to 2003 and we can simply replace any nan values in this variable with the median of this feature.

```
> df$Construction.year[is.na(df$Construction.year)]<-median(df$Construction.year,na.rm=TRUE)
> # Price Nan values
> summary(df$price)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's 
   50.0   339.0   627.0   625.8   911.0  1200.0      76 
```

- Price:

```
> hist(df$price)
> summary(df$price)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's 
   50.0   339.0   627.0   625.8   911.0  1200.0      76 
```

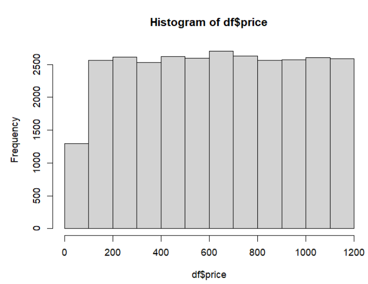


The price feature has an almost even distribution between 50 to 1200 and there are not many outliers to skew our mean. Therefore, we can simply replace any nan values in this variable with the mean or median of this feature.

```
> df$price[is.na(df$price)]<-median(df$price,na.rm=TRUE)
> summary(df$price)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
   50.0   341.0   627.0   625.8   910.0  1200.0 
```

- Service fee:

```
> summary(df$service.fee)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's 
   10.0    68.0   125.0   125.2   182.0   240.0      93 
> hist(df$service.fee)
```

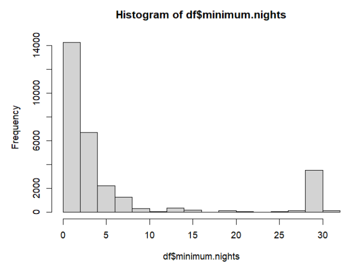


The service fee has an almost even distribution between 10 to 240 and there are not many outliers to skew our mean. Therefore, we can simply replace any nan values in this variable with the mean or median of this feature.

```
> df$service.fee[is.na(df$service.fee)]<-median(df$service.fee,na.rm=TRUE)
> summary(df$service.fee)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
   10.0    68.0   125.0   125.2   182.0   240.0 
```

- Minimum nights

```
> hist(df$minimum.nights,breaks=15)
> summary(df$minimum.nights)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's 
  1.000   1.000   3.000   6.625   5.000  31.000     487 
```

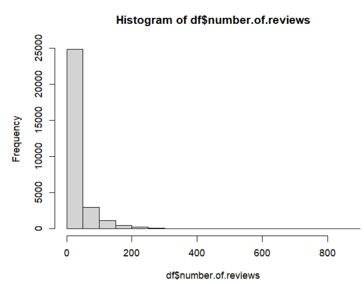


The minimum nights feature is extremely skewed to right. For handling the nan values for this feature. We know that 1 night is the most repeated number so we replace all the nan values with 1. Which, makes sense people need to stay at least one night.

```
> df$minimum.nights[is.na(df$minimum.nights)]<-1 > table(df$minimum.nights)[0:5]
> summary(df$minimum.nights)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.      1    2    3    4    5 
  1.000   1.000   3.000   6.534   5.000  31.000   7391 6875 4727 1983 1780 
```

- Number of reviews:

```
> summary(df$number.of.reviews)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
   0.00    1.00    7.00   27.52   31.00  884.00 
```



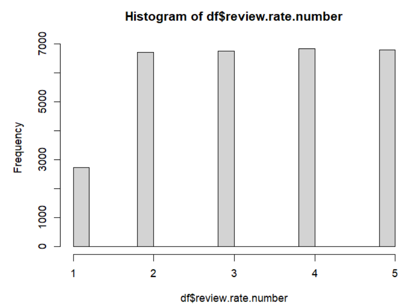

```
> table(df$number.of.reviews)[0:5]

   0    1    2    3    4 
4650 3033 2042 1572 1215 
```

The Number of reviews is extremely skewed to right. For handling the nan values for this feature. We know that 0 reviews is the most common number so we replace all the nan values with 0.

```
> df$number.of.reviews[is.na(df$number.of.reviews)]<-0
> summary(df$number.of.reviews)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
   0.00    1.00    7.00   27.52   31.00  884.00 
```

- Review rate number:

```
> summary(df$review.rate.number)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's 
  1.000   2.000   3.000   3.276   4.000   5.000     102 
> hist(df$review.rate.number)
```

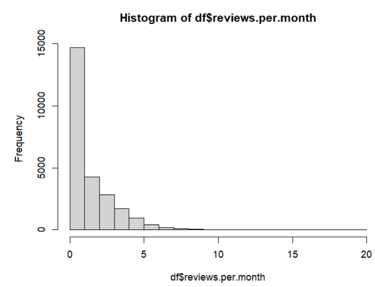


The rating has an almost even distribution. Therefore, we can simply replace any nan values in this variable with the median of this feature.

```
> df$review.rate.number[is.na(df$review.rate.number)]<-median(df$review.rate.number,na.rm=TRUE)
> summary(df$review.rate.number)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  1.000   2.000   3.000   3.275   4.000   5.000 
```

- Reviews per month:

```
> summary(df$reviews.per.month)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's 
  0.010   0.220   0.750   1.372   2.020  19.750    4644 
> hist(df$reviews.per.month)
```

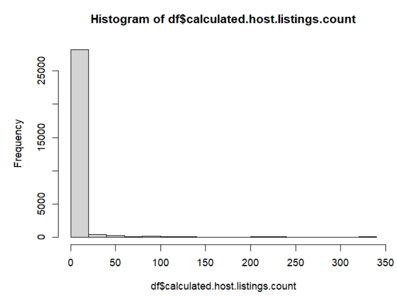


The Reviews per month feature is extremely skewed to right. For handling the nan values for this feature. We replace the Nan values with the median of this variable.

```
> df$reviews.per.month[is.na(df$reviews.per.month)]<-median(df$reviews.per.month,na.rm=TRUE)
> summary(df$reviews.per.month)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  0.010   0.280   0.750   1.276   1.730  19.750 
```

- Calculated host listings count:

```
> summary(df$calculated.host.listings.count)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's 
  1.000   1.000   1.000   7.843   2.000 332.000     100 
> hist(df$calculated.host.listings.count)
```

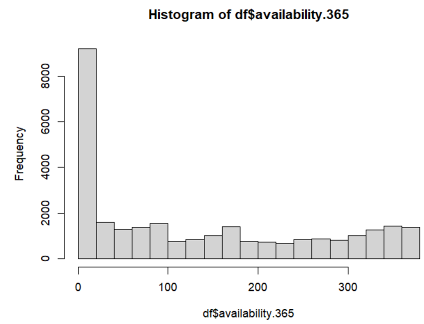


The calculated host listing count month feature is extremely skewed to right. For handling the nan values for this feature. We replace the Nan values with the median of this variable.

```
> df$calculated.host.listings.count[is.na(df$calculated.host.listings.count)]<-median(df$calculated.host.listings.count,
+                                                                                       na.rm=TRUE)
> summary(df$calculated.host.listings.count)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
   1.00    1.00    1.00    7.82    2.00  332.00 
```

- Availability 365:

```
> summary(df$availability.365)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's 
    0.0     3.0    90.0   133.4   249.0   366.0    1025 
> hist(df$availability.365)
```



The availability 365 feature has a spike around 0 and has an almost even distribution in other bins. We know that 0 is the most common number. Therefore, we replace all the nan values with 0.

```
> summary(df$availability.365)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
    0.0     0.0    87.0   128.9   245.0   366.0 
```

For the below categorical variables we choose the most common category for each variable to replace the nan values.

- host_identity_verified:

  Unconfirmed is the most common status.

```
> tail(names(sort(table(df$host_identity_verified))), 1)
[1] "unconfirmed"
> df$host_identity_verified[is.na(df$host_identity_verified)]<-tail(names(sort(table(df$host_identity_verified))), 1)
```

- neighborhood.group:

  Manhattan is the most common neighborhood group.

```
> tail(names(sort(table(df$neighbourhood.group))), 1)
[1] "Manhattan"
> df$neighbourhood.group[is.na(df$neighbourhood.group)]<-tail(names(sort(table(df$neighbourhood.group))), 1)
```

- neighbourhood:

  Bedford-Stuyvesant is the most common neighborhood.

```
> tail(names(sort(table(df$neighbourhood))), 1)
[1] "Bedford-Stuyvesant"
> df$neighbourhood[is.na(df$neighbourhood)]<-tail(names(sort(table(df$neighbourhood))), 1)
```

- cancellation_policy:

  The moderate is the most common cancellation policy.

```
> tail(names(sort(table(df$cancellation_policy))), 1)
[1] "moderate"
> df$cancellation_policy[is.na(df$cancellation_policy)]<-tail(names(sort(table(df$cancellation_policy))), 1)
```

- instant_bookable:

  False is the most common Status.

```
> tail(names(sort(table(df$instant_bookable))), 1)
[1] "False"
> df$instant_bookable[is.na(df$instant_bookable)]<-tail(names(sort(table(df$instant_bookable))), 1)
```

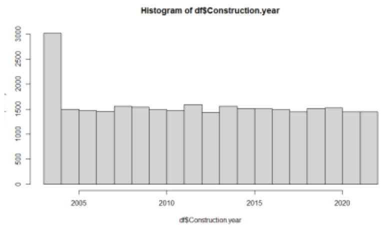
D) I believe the sum of price and service fee will have the main role in this dataset. Because normally people decide based on their budget and they will have some exaptation based on the price of the house. Therefore, the price of a house defiantly has an impact on what people thought of the house and as a result, their rating. Also, based on the range of price it can be guessed what neighborhood this house is located and what is its' room type.

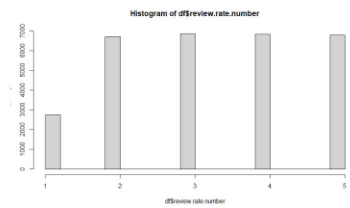
Note:

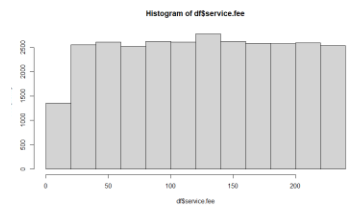
As you can see in the below histograms.

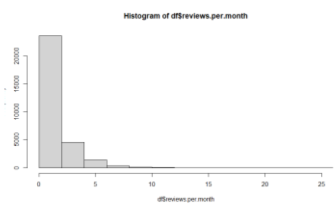
Most of the important features have an uniform distribution and only longitude and latitude have a close to normal distribution. Therefore, in the following questions you can see there is little to no relationship between variables, and we can not infer anything based on this dataset.

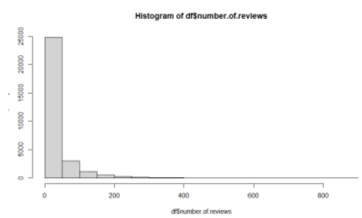
I have answered all the questions based on this dataset. But, I need to mention that this dataset is not correct and has been changed by someone (it is obvious just by looking at the price distribution that it should be skewed toward zero like real world data). The correct dataset is in [New York City Airbnb Open Data](#).

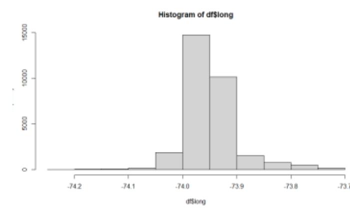


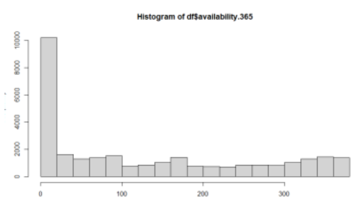


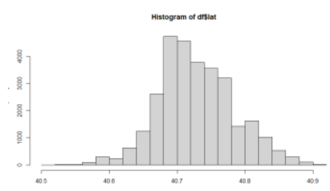


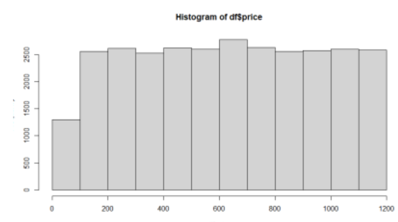


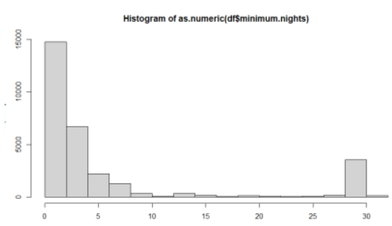










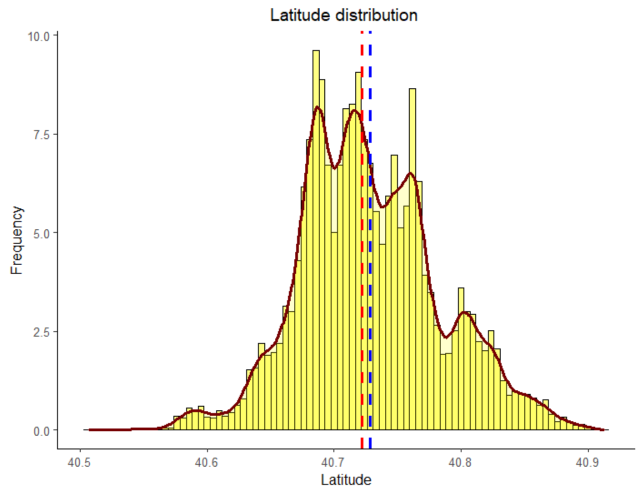


Question 1:

I will answer this question for two variables one of them is longitude which has a better distribution. The second one is price which is more important for this dataset.

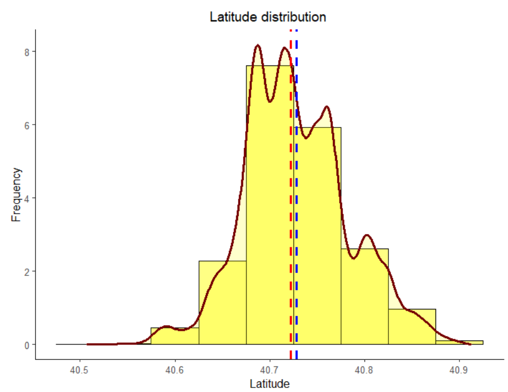
1- Longitude:

A) The distribution for this variable can be approximated as normal.




```
> bw <- 2 * IQR(df$lat) / length(df$lat)^(1/3)
> ggplot(df, aes(x=lat)) + geom_histogram(aes(y=..density..),binwidth =bw, color="black", fill="#ffff84") +
+     geom_density(aes(y=stat(density)),alpha = .2, fill= "yellow", color = '#760002',size=1)+
+     geom_vline(xintercept= c(mean(df$lat),median(df$lat)),size=1,linetype ="dashed",color=c('blue','red'))+ labs(title ="Latitude
 distribution ")+
+     xlab("Latitude") + ylab("Frequency")+theme(plot.title = element_text(hjust = 0.5),plot.subtitle = element_text(hjust = 0.5),
+                                            panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
+ .                                          panel.background = element_blank(), axis.line = element_line(colour = "black"))
```

For choosing the bin size we used Freedman-Diaconis. However, because our chased variable is longitude, we can consider a bin size that is aligned with our data. For example, bin size=0.05 makes sense. Because, we can understand the frequency of rental houses in distances of 5 km, and our hisogram would look like this:



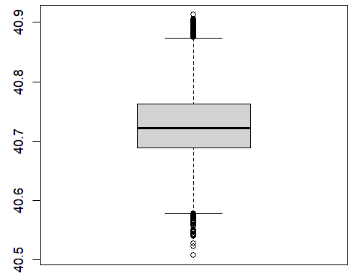
B) The distribution is skewed to right. In the figure above you can see the difference of mean and median. The histogram is also unimodal.

```
> summary(df$lat)
  Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
 40.51   40.69   40.72   40.73   40.76   40.91 
```

C)

- The upper whisker reach: min(max(x), Q3 + 1.5 * IQR)= 40.79976
- The lower whisker reach: max(min(x), Q1 – 1.5 * IQR)= 40.57818
- IQR=Q3-Q1= 0.07386
- The upper quartile: 40.76283
- The lower quartile: 40.68897

```
> PC_plot<-boxplot(df$lat)
> PC_plot$stats
         [,1]
[1,] 40.57821
[2,] 40.68897
[3,] 40.72224
[4,] 40.76283
[5,] 40.87305
> IQR(df$lat)
[1] 0.07386
> quantile(df$lat)
      0%      25%      50%      75%     100% 
40.50708 40.68897 40.72224 40.76283 40.91234 
```



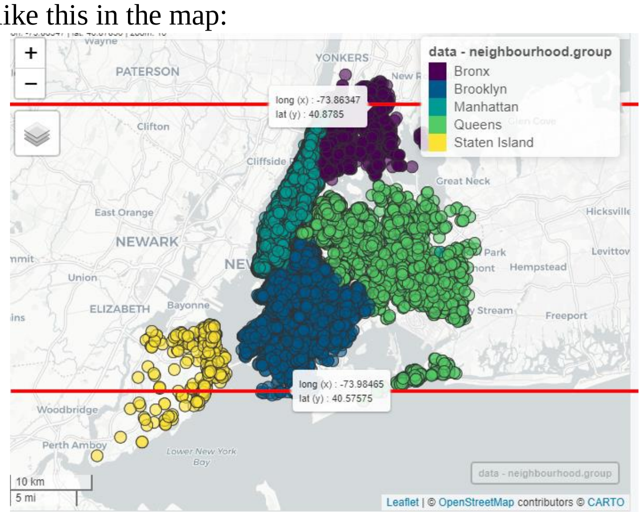
D) Cases with longitudes more than 40.79976 and less than 40.57818 are considered outliers

```
> PC_plot$out
  [1] 40.88107 40.88316 40.88165 40.89600 40.88393 40.89702 40.88805 40.56033 40.57556 40.88526 40.87618 40.57707 40.57753
 [14] 40.87858 40.89245 40.54106 40.56614 40.88985 40.88546 40.57093 40.57476 40.88796 40.88467 40.88377 40.88253 40.89600
 [27] 40.87821 40.88698 40.87491 40.90484 40.87910 40.87740 40.88534 40.57636 40.89890 40.87666 40.88393 40.89814 40.87886
 [40] 40.89279 40.87905 40.54250 40.57380 40.87871 40.88798 40.57592 40.87498 40.57629 40.88399 40.90281 40.87953 40.88010
 [53] 40.89981 40.87820 40.88511 40.88169 40.87739 40.89429 40.89649 40.90020 40.57808 40.56153 40.88453 40.87925 40.57762
 [66] 40.87540 40.57646 40.90505 40.57710 40.57762 40.55762 40.57458 40.88830 40.56546 40.87896 40.89528 40.89981 40.88455
 [79] 40.52211 40.88271 40.89121 40.89121 40.87663 40.57578 40.57721 40.88116 40.88297 40.57095 40.88990 40.87886 40.88192
 [92] 40.57609 40.88777 40.87414 40.57552 40.91234 40.87583 40.89382 40.57647 40.56251 40.89308 40.54639 40.88151 40.87925
[105] 40.87910 40.54615 40.87621 40.88143 40.57411 40.50708 40.88249 40.89279 40.57753 40.87886 40.57458 40.57294 40.87396
[118] 40.88316 40.87618 40.90260 40.56933 40.87621 40.88017 40.88364 40.87647 40.54857 40.87850 40.54106 40.87701 40.87618
[131] 40.87810 40.57455 40.89768 40.57577 40.56629 40.57576 40.88805 40.57582 40.88354 40.57413 40.87900 40.52700 40.55105
[144] 40.89400 40.87851 40.89685 40.88542 40.57044 40.88855 40.89502 40.88909 40.88304 40.87847 40.54878 40.54550 40.88500
[157] 40.89156 40.57750 40.87697 40.88283 40.57759 40.89984 40.56028 40.57589 40.52211 40.87695 40.89800 40.88166 40.57641
[170] 40.90154 40.88068 40.57210 40.87906 40.88316 40.87749 40.88204 40.57548 40.56464 40.56251 40.87934 40.57567 40.88032
[183] 40.89010 40.88332 40.55616 40.89637 40.90112 40.87829 40.54106 40.89279 40.89581 40.88832 40.57575 40.56506 40.89124
[196] 40.87900 40.89385 40.87877 40.87530 40.87870 40.90168 40.88546 40.54268 40.87924 40.88944 40.90154 40.88366 40.87712
[209] 40.88229 40.88297 40.57201 40.89681 40.87570 40.57095 40.57756 40.57674 40.87842 40.56464 40.87698 40.57569 40.87699
[222] 40.88058 40.90356 40.88066 40.56434 40.53987 40.88634 40.57650 40.90059 40.56082 40.56482 40.89691 40.89121 40.88296
[235] 40.57491 40.54652 40.87850 40.57543 40.56649 40.88340 40.54889 40.88166 40.88485 40.57717 40.89781 40.57575 40.89393
[248] 40.87553 40.89600 40.89743 40.57573 40.87804 40.88493 40.87548 40.89216
```

We have 235 outliers in our data:

```
> length(PC_plot$out)
[1] 255
```

These data will look like this in the map:



Based on the above figure it is obvious that these cases are far away from the center of New York City. Therefore, they are considered outliers.

E)

- Mean=40.72828
- Median=40.72224
- Variance=0.00309331
- Standard deviation=0.05561753

```
> mean(df$lat)
[1] 40.72828
> median(df$lat)
[1] 40.72224
> var(df$lat)
[1] 0.00309331
> sd(df$lat)
[1] 0.05561753
```

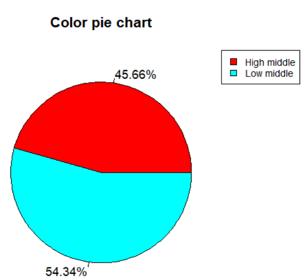
As it is understandable from the map. Our data is not much spread out from the mean. Which caused the low variance and low standard deviation. and because our data is not much skewed, the mean and median are close.

F) It is not obvious what is the meaning of this question. I will answer for two possible meanings:

1- Splitting data into 4 categories:

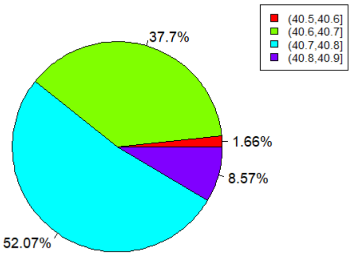
   a. Lower than half of mean
   b. Bigger than half of the mean and lower than mean
   c. Bigger than mean and lower than 1.5*mean
   d. Bigger than 1.5*mean

As mentioned in past parts of this project. Our data is very dense around the mean. Therefore, there is no data present in categories of d and a.



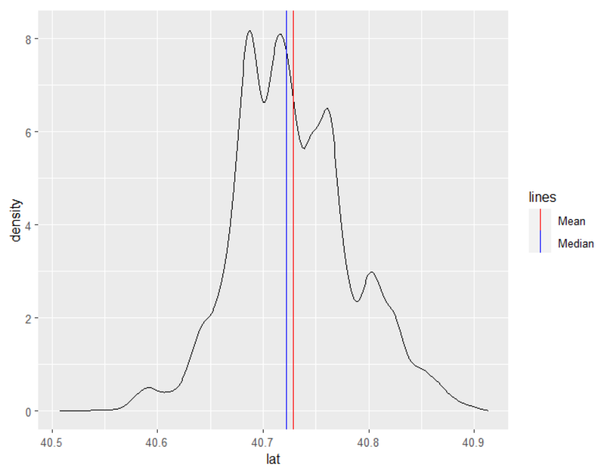

```
> dat <- df['lat']
> dat <- within(dat, {place <- NA
+ place[dat$lat<=0.5*mean(dat$lat)] <- "Low"
+ place[dat$lat>0.5*mean(dat$lat) & dat$lat<=mean(dat$lat)] <- "Low middle"
+ place[dat$lat<=1.5*mean(dat$lat) & dat$lat>mean(dat$lat)] <- "High middle"
+ place [dat$lat>1.5*mean(dat$lat)]<- "High"} )
> dat$place <- factor(dat$place)
> pie( table(dat$place), labels = paste0(round(100 * table(dat$place)/sum(table(dat$place)), 2), "%")
+      , main = "Color pie chart", col=rainbow(length(levels(dat$place))))
> legend("topright", levels(dat$place), cex = 0.8,fill = rainbow(length(levels(dat$place))))
```

2- Splitting our data into 4 groups using cut:




```
> dat=cut(df$lat,breaks=4)
> pie(table(dat),labels = paste0(round(100 *table(dat)/sum(table(dat)), 2), "%"),col=rainbow(length(levels(dat))))
> legend("topright", levels(dat), cex = 0.8,fill = rainbow(length(levels(dat))))
```

G) The density plot will be as followed:

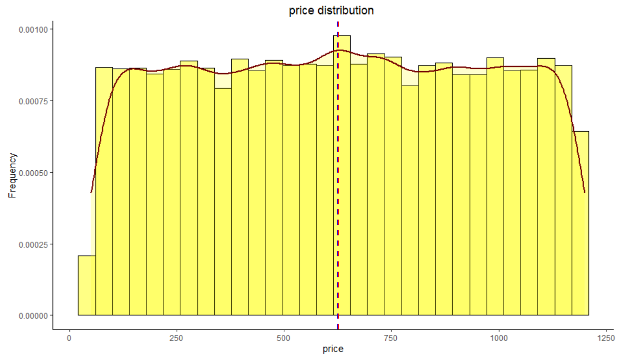



```
ggplot(df, aes(x=lat)) +
geom_density()+geom_vline(aes(xintercept= mean(df$lat),
                              linetype = 'Mean'), colour = 'red') +

geom_vline(aes(xintercept = median(df$lat),
                         linetype = 'Median'), colour = 'blue') +
scale_linetype_manual(name = 'lines',
                      values = c('Mean' = 1,
                                 'Median' = 1),
                      guide = guide_legend(override.aes = list(colour = c('red',
                                                                          'blue'))))
```

Based on the plot we understand that Mean>Median

Density has a spike near the mean. Which means, most of the data are around the mean.

Price:



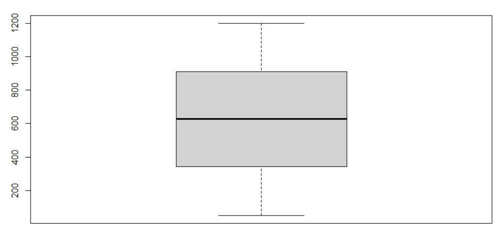
For this variable, the distribution is not normal and it is almost uniform.

B) the distribution modality is uniform. It is almost symmetric and there is no skewness.

```
> summary(df$price)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
   50.0   341.0   627.0   625.8   910.0  1200.0 
```

C)

- The upper whisker reach: min(max(x), Q3 + 1.5 * IQR)= 1200
- The lower whisker reach: max(min(x), Q1 – 1.5 * IQR)= 50
- IQR=Q3-Q1= 569
- The upper quartile: 910
- The lower quartile: 341




```
> PC_plot$stats
     [,1]
[1,]   50
[2,]  341
[3,]  627
[4,]  910
[5,] 1200
```

```
> IQR(df$price)
[1] 569
> quantile(df$price)
  0%  25%  50%  75% 100% 
  50  341  627  910 1200 
```

D) As it is seen in the boxplot there is no outliers.

E)

```
> mean(df$price)
[1] 625.7875
> median(df$price)
[1] 627
> var(df$price)
[1] 109328.5
> sd(df$price)
[1] 330.6486
```

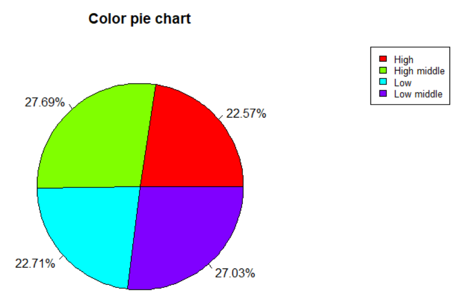
The distribution is uniform from 50 to 1200. Therefore, it makes sense for the mean and the median to be almost exactly in the middle of our range. Also, we have a big range that has an almost distribution. Hence, the large value of variance and standard deviation is valid.

F) It is not obvious what is the meaning of this question. I will answer for two possible meanings:

3- Splitting data into 4 categories:

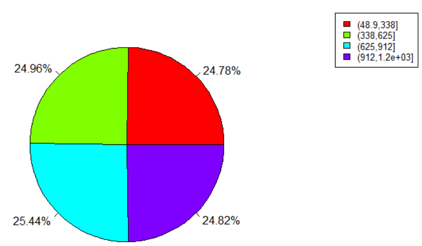
a. Lower than half of mean
b. Bigger than half of the mean and lower than mean
c. Bigger than mean and lower than 1.5*mean
d. Bigger than 1.5*mean

As mentioned in past parts of this project. Our data is very dense around the mean. Therefore, there is no data present in categories of d and a.



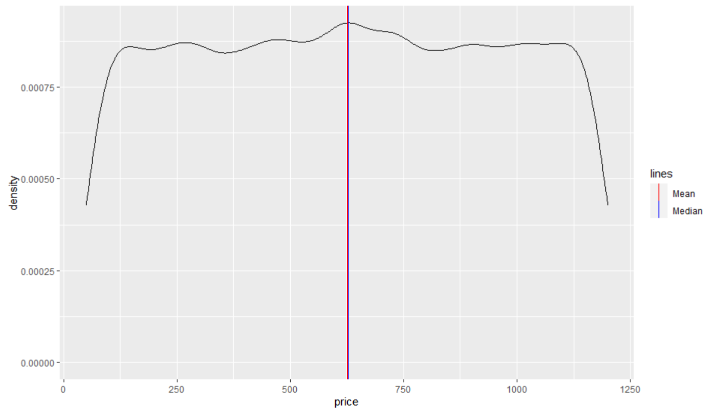

```
> dat <- df['price']
> dat <- within(dat, {place <- NA
+ place[dat$price<=0.5*mean(dat$price)] <- "Low"
+ place[dat$price>0.5*mean(dat$price) & dat$price<=mean(dat$price)] <- "Low middle"
+ place[dat$price<=1.5*mean(dat$price) & dat$price>mean(dat$price)] <- "High middle"
+ place [dat$price>1.5*mean(dat$price)]<- "High"} )
> dat$place <- factor(dat$place)
> pie( table(dat$place), labels = paste0(round(100 * table(dat$place)/sum(table(dat$place)), 2), "%")
+       , main = "Color pie chart", col=rainbow(length(levels(dat$place))))
> legend("topright", levels(dat$place), cex = 0.8,fill = rainbow(length(levels(dat$place))))
```

4- Splitting our data into 4 groups using cut:




```
> dat=cut(df$price,breaks=4)
> pie(table(dat),labels = paste0(round(100 *table(dat)/sum(table(dat)), 2), "%"),col=rainbow(length(levels(dat))))
> legend("topright", levels(dat), cex = 0.8,fill = rainbow(length(levels(dat))))
```

G)



Mean and median are almost equal.

Question 2:

I chose the neighborhood group variable:

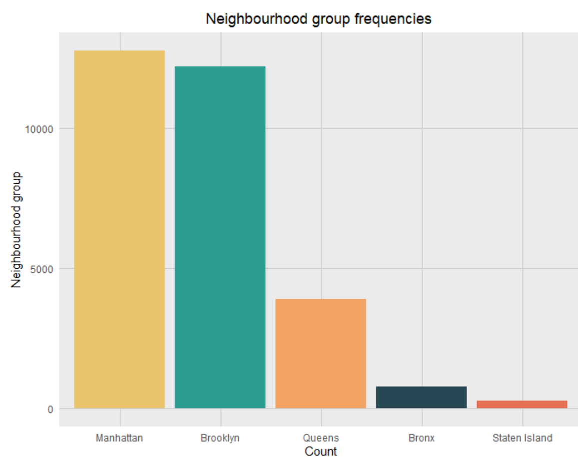
A) The table is like this:

Neighborhood Group	Bronx	Brooklyn	Manhattan	Queens	Staten Island
Frequency	797	12201	12784	3897	278

In R:

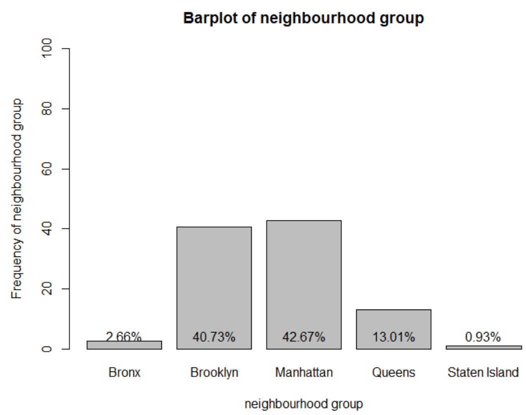
```
> table(df$neighbourhood.group)

        Bronx      Brooklyn     Manhattan        Queens Staten Island
          797         12201         12784          3897           278
```

B)



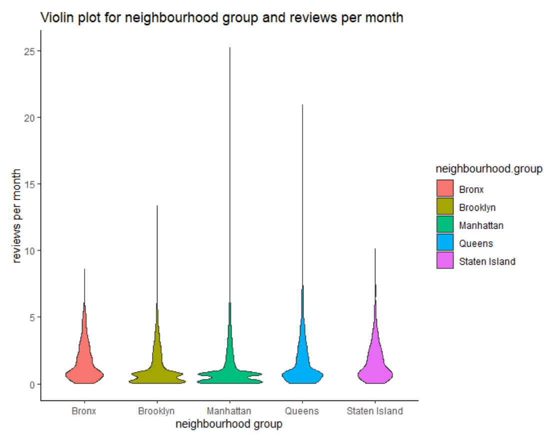

```
> ggplot(df, aes(x=reorder(neighbourhood.group,neighbourhood.group, function(x)-length(x)),fill=neighbourhood.group)) +
+   geom_bar()+ labs(title ="Neighbourhood group frequencies ")+
+   xlab("Count") + ylab("Neighbourhood group")+scale_fill_manual(values = c("#264653","#2a9d8f","#e9c46a","#f4a261","#e76f51"), guide = "none")
 +
+   theme(panel.grid.major = element_line(colour = "grey80"), panel.grid.minor = element_blank(),
+         panel.background = element_rect(),axis.ticks.x=element_blank(),axis.ticks.y=element_blank(),
+         plot.title = element_text(hjust = 0.5),plot.subtitle = element_text(hjust = 0.5))
```

C)

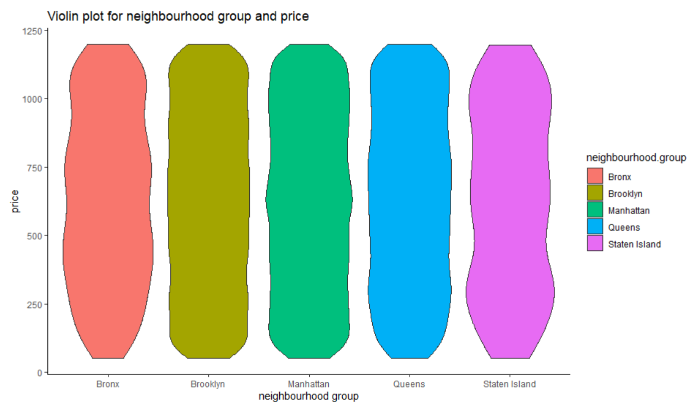



```
> bp<-barplot(meds, ylim=c(0,100),
+             main="Barplot of neighbourhood group",
+             xlab="neighbourhood group",
+             ylab="Frequency of neighbourhood group",)
> text(bp, 0, paste(round(meds, 2), "%", sep=""), cex=1, pos=3)
```

D) We chose reviews per month as the second numerical variable. The violin plot is as followed:



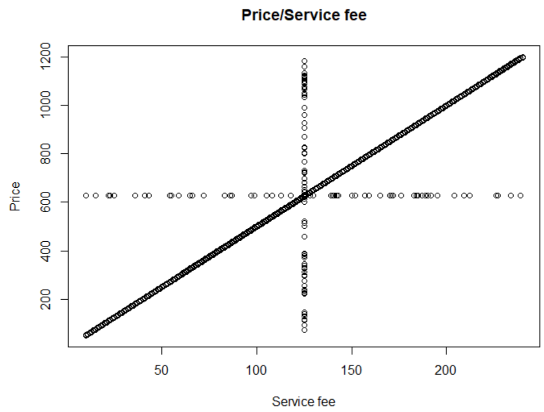
I will show this plot for price/neighborhood too to prove my point about this dataset having incorrect data:



As you can see here. The price distribution in all the neighborhood groups are almost equal. This obviously is not usual because in every city some neighborhood groups are more expensive than others. In this scenario, Manhattan and Brooklyn are two of the most renowned neighborhood and most expensive neighborhood in NYC. But here all the neighborhood have almost the same price.

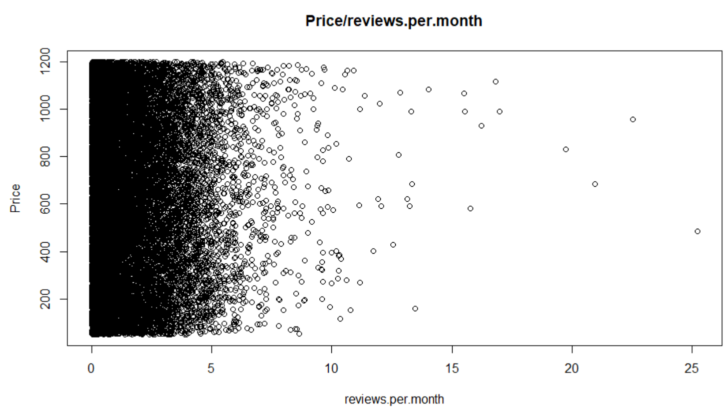
Question 3:

A) For price and service fee the scatter plot would look like this:



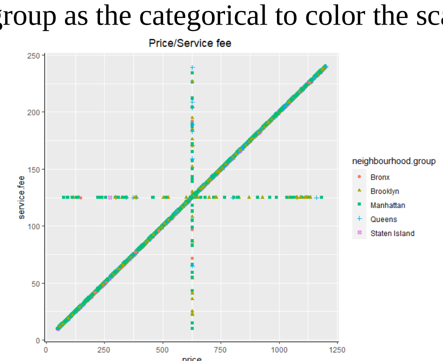
In most cases, there is a positive linear relationship between price and service fee. as one increases, the other increases.

We can draw scatter plot for other variables too for example:



There is not really a relationship between these two variables. But, It is understandable that most of the houses get under the 10 reviews. Per month which is valid.

B) We chose the neighborhood group as the categorical to color the scatter plot.



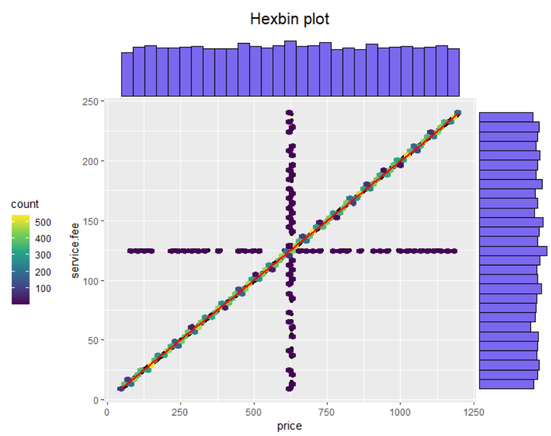
The relation between price and service fee still holds for different categories.

C) They are highly correlated:

```
> val<-cor.test(df$price,df$service.fee)
> val

        Pearson's product-moment correlation

data:  df$price and df$service.fee
t = 2515.4, df = 29955, p-value < 2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.9975873 0.9976940
sample estimates:
      cor
0.9976413
```

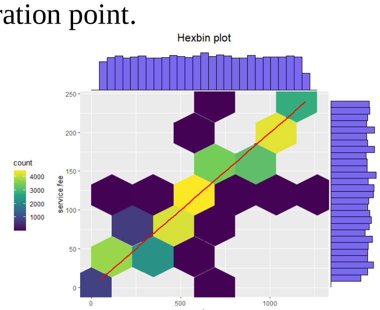
A p-value is a probability that the null hypothesis is true. In our case, it represents the probability that the correlation between price and service fee in the sample data occurred by chance. A p-value of 2.2e-16 means that there is only $2.2 \times 10^{-14}\%$ chance that results from our sample occurred due to chance.

D)

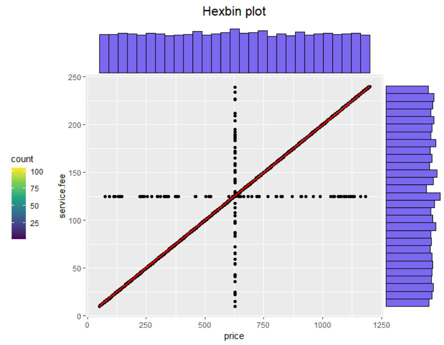


Based on the above figure, we understand that most of the cases abide by the relationship between price and service fee. Meaning that only small amount cases have a high price while having low service fees and vice versa.

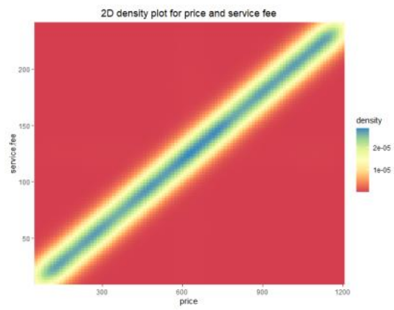
If the bin size is really small, The concentration of the data will not be clear, and a large range may be presented as the data concentration point.



If the bin size is really big, the Hexbin plot does not differ from a simple scatter plot and will not give us any useful additional information.



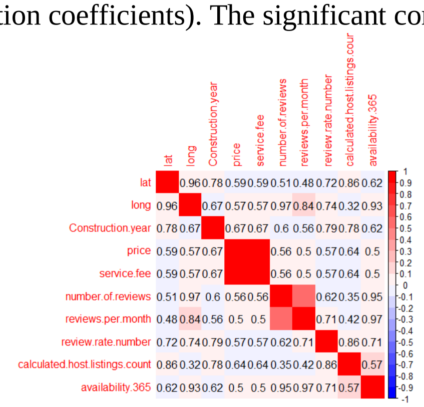
E)

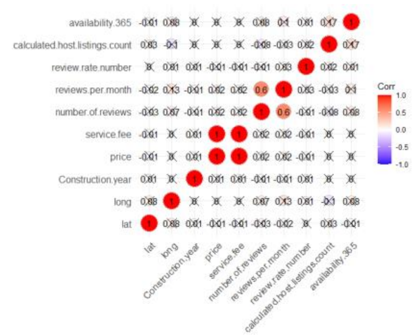


Hexibin plot shows data in more accurate way. In general it leaves room for interpretation, we can find the exact outliers and their density. A data is a dot on a specific spot. However, 2d density plots shows some specific information in more understandable way. In addition, if the data set is so large that you literally can't make sense of a scatter plot due to overdrawing/occlusion, then we have to start looking for solutions like the chart on the left. Hexagon bin plots are another good option. Also, determining the number of bins in the Hexibin plot is something that should be discussed.

Question 4:

A) The heatmap correlograms for our variable will be like this (containing p-value and Pearson's correlation coefficients). The significant correlation is highlighted with the color red.



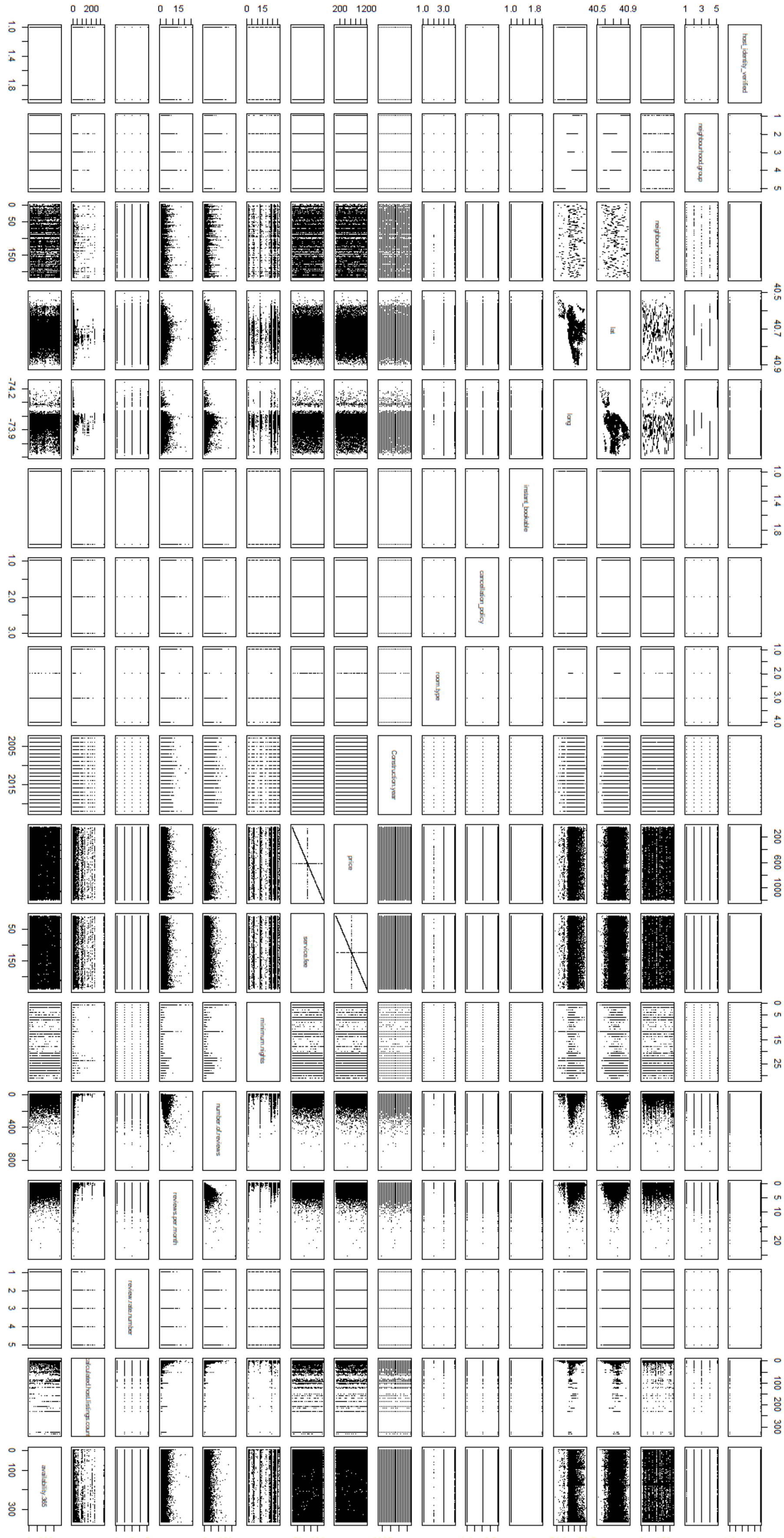



```
> airbnb_cor <- df[, sapply(df, is.numeric)]
> airbnb_cor <- airbnb_cor[complete.cases(airbnb_cor), ]
> correlation_matrix <- cor(airbnb_cor)
> p.mat <- cor_pmat(correlation_matrix)
> corrplot(correlation_matrix, p.mat = p.mat,method = "color", insig = "p-value",
+          col= colorRampPalette(c("blue","white", "red"))(20))
> ggcorrplot(correlation_matrix, method = "circle",
+            colors = c("blue", "white", "red"),
+            lab = TRUE, p.mat=p.mat,)
```

B) The correlogram is on the next page:

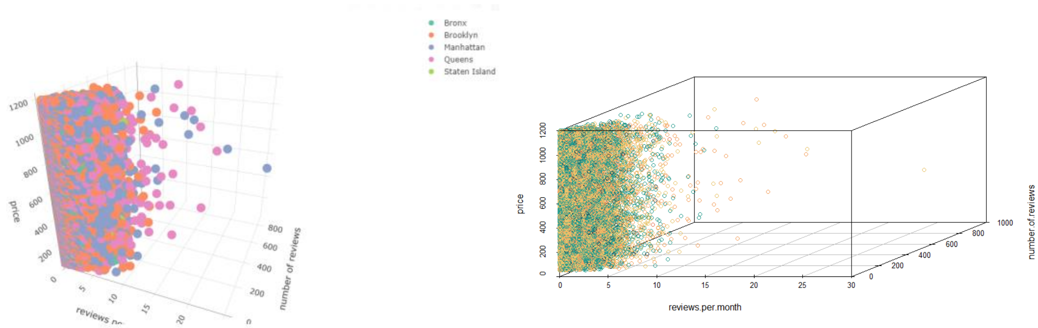
In most variables, there is no meaningful pattern between features. In other words, cases are distributed almost even in every class. Here I mention a couple of these patterns that I could spot:

- In most cases, there is a positive linear relationship between price and service fee. as one increases, the other increases.
- Based on the latitude and longitude(in addition to information from previous parts of this project). It is understandable these houses are concentrated around the center of NYC.
- The listing with the most number of reviews has the most number of reviews per month. Which makes sense. Also, most cases have reviews per month<15. (I explained why this is logical in Question 0.)



host_identity_verified
neighbourhood.group
neighbourhood
lat
long
instant_bookable
cancellation_policy
room.type
Construction.year
price
service.fee
minimum.nights
number.of.reviews
reviews.per.month
review.rate.number
calculated.host.listings.count
availability.365

C) I Choose the price, reviews per month, and number of reviews for numerical variables and the neighborhood group as the categorical variables.




```
> #install.packages("scatterplot3d") # Install
> library("scatterplot3d") # load
> library(plotly)
> col_names<-c('reviews.per.month','number.of.reviews','price')
> df$neighbourhood.group = as.factor(df$neighbourhood.group)
> shapes = c("#264653","#2a9d8f","#e9c46a","#f4a261","#e76f51")
> shapes <- shapes[as.numeric(df$neighbourhood.group)]
> scatterplot3d(df[col_names],color=shapes)
> plot_ly(x=df$reviews.per.month, y=df$number.of.reviews, z=df$price, type="scatter3d",
+         mode="markers", color=df$neighbourhood.group)%>%
+   layout(scene = list(xaxis = list(title = "reviews per month"),
+                       yaxis = list(title = "number of reviews"),
+                       zaxis = list(title = "price")))
```

The price, reviews per month, and the number of reviews distribution is almost even in a different neighborhood group. However, there is a slight increase in price for houses with a larger number of reviews. Also, a larger number of reviews per month caused a slight increase in the number of reviews.

Question 5:

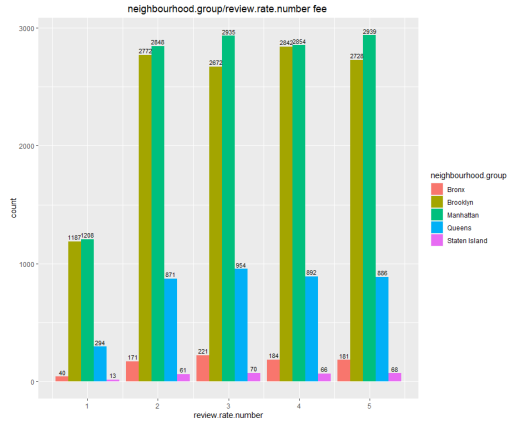
A) The Contingency table for the neighborhood group and review rate number is like this:

neighborhood group / review rate number	1	2	3	4	5	Total
Bronx	40	171	221	184	181	797
Brooklyn	1187	2772	2672	2842	2728	12201
Manhattan	1208	2848	2935	2854	2939	12784
Queens	294	871	954	892	886	3897
Staten Island	13	61	70	66	68	278
Total	2742	6723	6852	6838	6802	29957

In R:

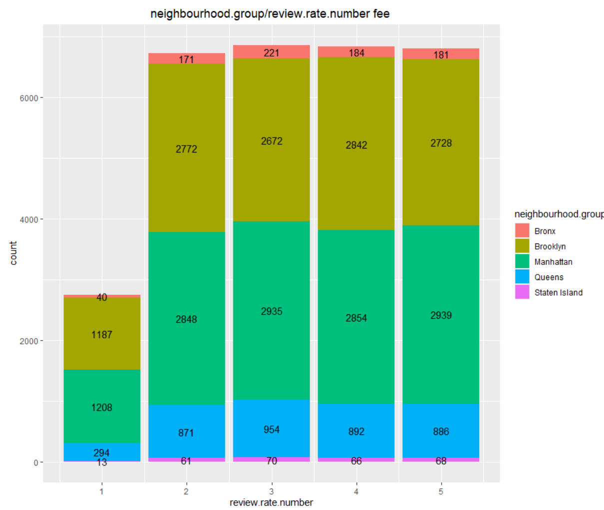
```
> addmargins(table(df$neighbourhood.group,df$review.rate.number))

                  1     2     3     4     5   Sum
  Bronx          40   171   221   184   181   797
  Brooklyn     1187  2772  2672  2842  2728 12201
  Manhattan    1208  2848  2935  2854  2939 12784
  Queens        294   871   954   892   886  3897
  Staten Island  13    61    70    66    68   278
  Sum          2742  6723  6852  6838  6802 29957
```

B)



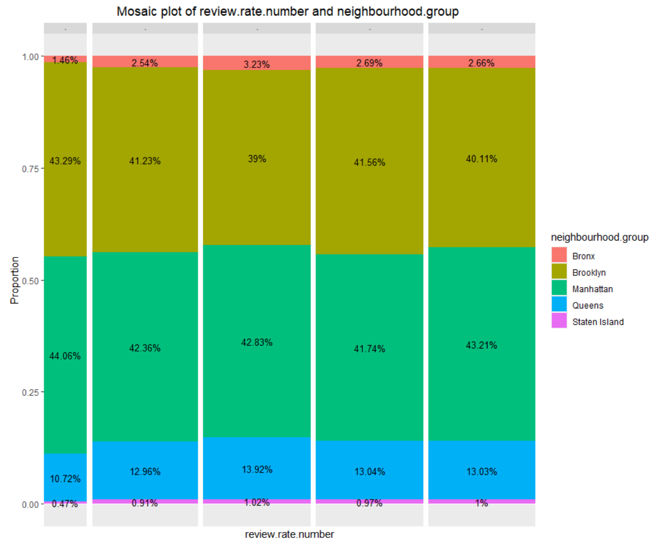

```
> ggplot(df, aes(fill=neighbourhood.group, x=review.rate.number)) +
+   geom_bar(position=position_dodge(), stat="count")+
+   geom_text(stat='count', aes(label=..count..),position=position_dodge(0.9),vjust=-0.4,size=3)+
+   labs(title ="neighbourhood.group/review.rate.number fee ")+
+   theme(plot.title = element_text(hjust = 0.5))
```

C)




```
> ggplot(df, aes(fill=neighbourhood.group, x=review.rate.number)) +
+   geom_bar(position="stack", stat="count")+
+   geom_text(stat='count', aes(label=..count..),position = position_stack(vjust = 0.5))+
+   labs(title ="neighbourhood.group/review.rate.number fee ")+
+   theme(plot.title = element_text(hjust = 0.5))
```

D)




```
> v <- aggregate(df$neighbourhood.group~df$review.rate.number + neighbourhood.group, data = df, FUN = length)
> colnames(v) <- c('review.rate.number','neighbourhood.group','counts')
> v<-transform(v, rel1 = round(ave(counts, review.rate.number, FUN = prop.table),digit=4))
> v<-transform(v, grpsize = aggregate(v$counts, by=list(review.rate.number =v$review.rate.number ), FUN=sum))
> colnames(v) <- c('review.rate.number','neighbourhood.group','counts','rel1','test','grpSize')
> names=c('lv hypertrophy'="",'normal'="",'st-t abnormality'="")
> graphics.off()
> ggplot(v, aes(x=review.rate.number,y=rel1,fill=neighbourhood.group,width = grpSize)) +
+   geom_bar(stat='identity') +
+   scale_x_discrete(expand = c(0, 0)) +
+   facet_grid(~review.rate.number, scales = "free", space = "free",labeller = as_labeller(names))+
+   geom_text(aes(label = paste(round(100*rel1, 2), "%", sep="")),size=3.34,position = position_stack(vjust = 0.5)) +
+   labs(title ="Mosaic plot of review.rate.number and neighbourhood.group ")+
+   xlab("review.rate.number") + ylab("Proportion")+
+   theme(plot.title = element_text(hjust = 0.5), plot.subtitle = element_text(hjust = 0.5),
+         strip.text = element_text(size = 1))
```

Question 6:

In this dataset, there are no numerical variables that have a close to the normal distribution, except longitude and latitude. But, these two variables both have outliers which, prevents us from using the CLT-based test. Also, these two variables are unimportant for our research. However, for the sake of doing the tests, and showing the process of t-test and CLT I assume there are no outliers and ignore them in these two variables and I will do the test. It should be mentioned that, because these variables do not met conditions of test. The result of these test might be incorrect.

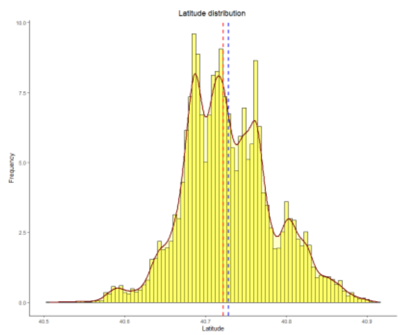
I answer this question for two possibilities:

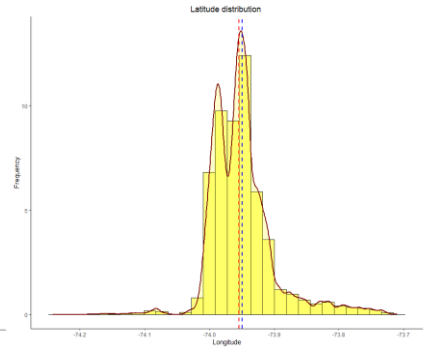
1- Two completely different numerical values:

A) I chose the latitude and longitude as our numerical values

   For choosing between the t-test and the z-test we need to check the conditions:

   - Our observations are independent.
     - We chose 25 samples which are less than 10% of the whole population.
   - The population distribution is not extremely skewed.
     - However, we have chosen 25 samples which are less than 30
     - There are outliers and therefore we can not use CLT but for the sake of doing the test we assume, there are no outliers.





Based on the above explanation, we need to use t-test.

B)

```
> t.test(x,y, mu =0,df=24, conf.lev=.95)

        Welch Two Sample t-test

data:  x and y
t = 8457.9, df = 44.973, p-value < 2.2e-16
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 114.6434 114.6980
sample estimates:
mean of x mean of y
 40.71876 -73.95192
```

$Difference = 40.73893 - (-73.95599) = 114.67068$

$p - value < \alpha = 0.05$. Therefore, We reject H_0 meaning that there is a significant difference between the mean of our variables.

As I mentioned in the beginning CLT based tests should not be used for this variable because of the presence of outliers. Here, we got a wrong answer for our hypothesis. The difference value is in our confidence interval. The confidence interval does not support rejecting the H_0. Which is wrong and the result of the t-test and CLT should be the same.

2- One numerical variable and two different groups:

For example, I chose to check if there is a significant difference between the mean latitude of two neighborhoods.


```
> t.test(x,y, mu =0, conf.lev=.95)

        Welch Two Sample t-test

data:  x and y
t = 25.237, df = 41.814, p-value < 2.2e-16
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 0.219954 0.258194
sample estimates:
mean of x mean of y
 40.84936  40.61029
```

$Difference = 40.84936 - (40.61029) = 0.23907$

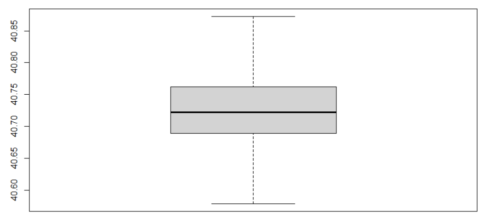
$p - value < \alpha = 0.05$. Therefore, We reject H_0 meaning that there is a significant difference between the mean of our variables.

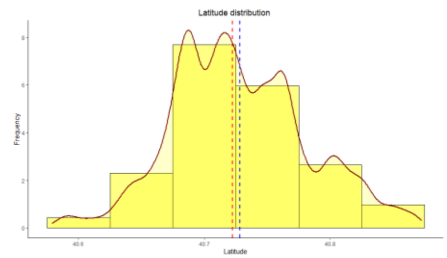
As I mentioned in the beginning CLT based tests should not be used for this variable because of the presence of outliers. Here, we got a wrong answer for our hypothesis. The difference value is in our confidence interval. The confidence interval does not support rejecting the H_0. Which is wrong and based on the map above it is clearly obvious that the means should be different.

Question 7:

Based on the variables of this dataset, I can not answer this question. Because None of the variables met the condition to use CLT.

However, for the sake of doing this task, I assume there are no outliers for the latitude variable and I will delete them to use CLT.






```
> IQR <- IQR(data_no_outlier$lat)
> quartiles <- quantile(data_no_outlier$lat, probs=c(.25, .75), na.rm = FALSE)
> Lower <- quartiles[1] - 1.5*IQR
> Upper <- quartiles[2] + 1.5*IQR
> data_no_outlier <- subset(data_no_outlier, data_no_outlier$lat > Lower & data_no_outlier$lat < Upper)
> boxplot(data_no_outlier$lat)
```

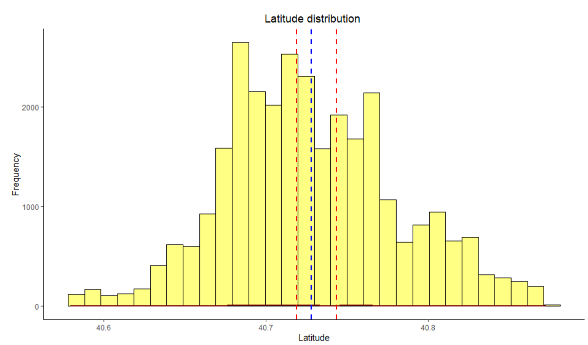
A) We calculate the interval using the below formula:

$$\bar{x} - z^{*}SE < \mu < \bar{x} + z^{*}SE$$

```
> upper_limit<-sample_mean+qnorm((1-0.98)/2)*Me
> #A)
> Our_sample<-data_no_outlier[sample(nrow(df),100, replace = TRUE),]
> Our_sample<-Our_sample$lat
> sample_mean<-mean(Our_sample)
> sample_sigma<-sd(Our_sample)
> Me<-sample_sigma/sqrt(length(Our_sample))
> upper_limit<-sample_mean+qnorm((1-0.98)/2)*Me
> lower_limit<-sample_mean-qnorm((1-0.98)/2)*Me
```

B) We are 98% confident that rental houses in NYC on average are located between 40.71879 to 40.74338.

C)



D) We suppose that house in NYC are located on average in latitude of 40.7 and we test if based on our sample this assumption is true or not.

$$H_0: lat = 40.7$$
$$H_A: lat > 40.7$$

If we consider $\alpha = 0.02$. The p-value is very small therefore, we reject the null hypothesis. This means, our sample do not provide enough evidence to prove houses in NYC are located on average in 40.7

p-value = P(observed or more extreme outcome | H_0 true)
in our cases, p-value is the probability of the location of housed on average be in 40.7 and considering observation of our sample.

E) Yes the confidence interval supports our p-value result. The latitude of 40.7 is outside of the range of CI. Therefore, CI rejects the null hypothesis too.

F)

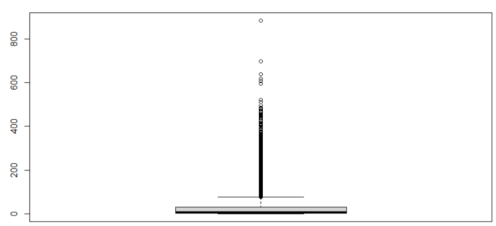
$$\text{Power} = 1 - \text{ Type II error} = 1 - \beta = 1 - p(z \leq z_a - \frac{40.72866 - 40.7}{0.005089511}) \Rightarrow \text{power} = 1 - p(z \leq 1.96 - \frac{40.72866 - 40.7}{0.005089511}) = 1 - p(z \leq 2.326348) = 1 - 0.6535786 = 0.01$$

$$\text{Type II error} = 0.99$$

G) The magnitude of an effect size greatly impacts statistical power. Large effect sizes increase statistical power and small effect sizes decrease power.

Question 8:

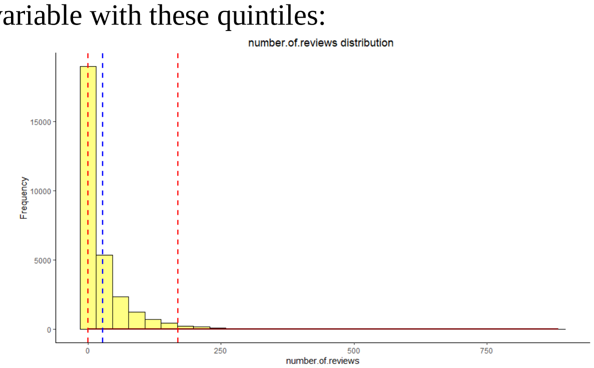
I used the number of reviews for this question:



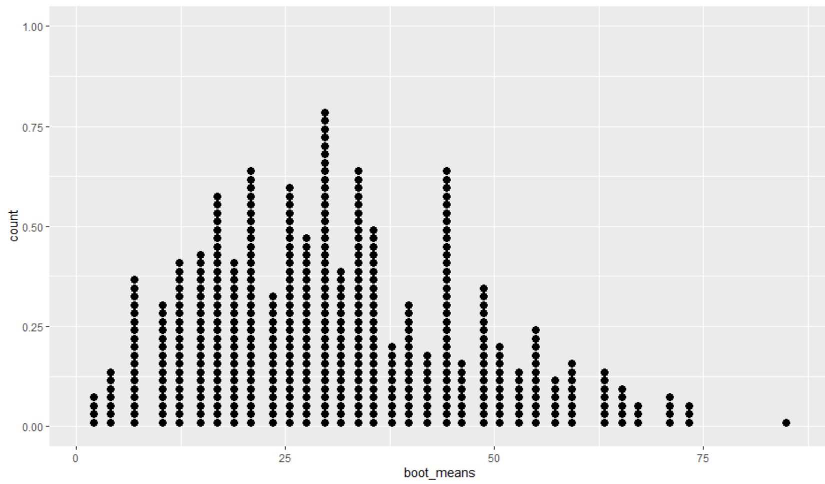
A) The 95% confidential interval using the quintiles of this variable are as followed:

```
> quantile(df$number.of.reviews,0.025)
2.5%
   0
> quantile(df$number.of.reviews,0.975)
97.5%
  169
```

The histogram of this variable with these quintiles:

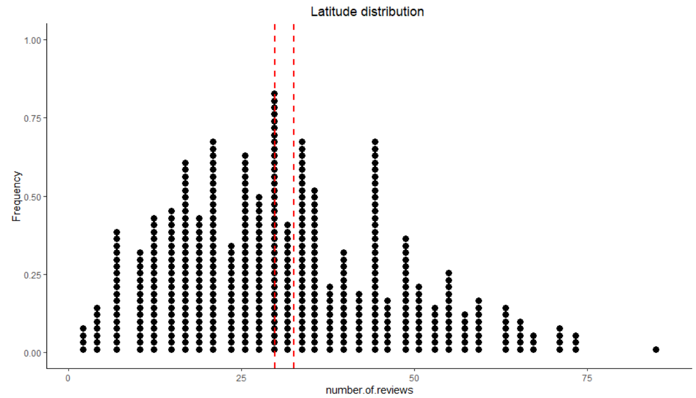


B) Now we take 20 samples from the original data and we will resample from this sample for 500 times with replacement to build the bootstrap distribution.
The bootstrap distribution is as followed:



Now for this distribution we calculate the CI:

$$\bar{x} - z^{*}SE < \mu < \bar{x} + z^{*}SE$$



C) Yes, there is noticeable difference between two methods.
The quintile method is used on the original population and is not the correct method for estimating the mean. However, with bootstrapping we can calculate a correct confidence interval to estimate the mean value.

Question 9:

Again, we do not have any variables to that meet our conditions. However, for the sake of doing the tests, and showing the process of ANOVA I will show the process. It should be mentioned that, because these variables do not met conditions of test. The result might be incorrect.

I will use neighborhood group and latitude as my categorical and numerical variable. We check if latitude is different in different neighborhood groups. (It obviously is)

I also check for the price and neighborhood group variable. Because, price is a more important variable for this dataset.

1- Latitude:
A)
Hypothesis
H_0: The mean latitude is the same across all the mentioned neighborhood group.

$$\mu_1 = \mu_2 = \mu_3 = \mu_4 = \mu_5$$

H_A: The mean latitude differs between at least one pair of the mentioned neighborhood group.

```
                        Df Sum Sq Mean Sq F value Pr(>F)    
neighbourhood.group      4  50.84  12.710   10796 <2e-16 ***
Residuals            29680  34.94   0.001                   
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

```
> df$neighbourhood.group <- factor(df$neighbourhood.group)
> result <- aov( lat~neighbourhood.group, data = data_no_outlier)
> summary(result)
```

B)

Based on the calculated $p - value < \alpha = 0.05$. We reject the null hypothesis. In other words, we conclude that there is at least one group that has a different mean latitude from others.

Now we need to do a pairwise comparison to find this group.

Our hypothesis for each pair wise comparison would be like:

H_0: The mean latitude in neighborhood group i and neighborhood group j is the same.

$$\mu_i - \mu_j = 0$$

H_A: The mean latitude in neighborhood group i and neighborhood group j is not the same.

$$\mu_i - \mu_j \neq 0$$

For the above pairwise test, we need to modify α based on Bonferroni correction therefore we have:

$$\alpha^* = \frac{\alpha}{5} = 0.01$$

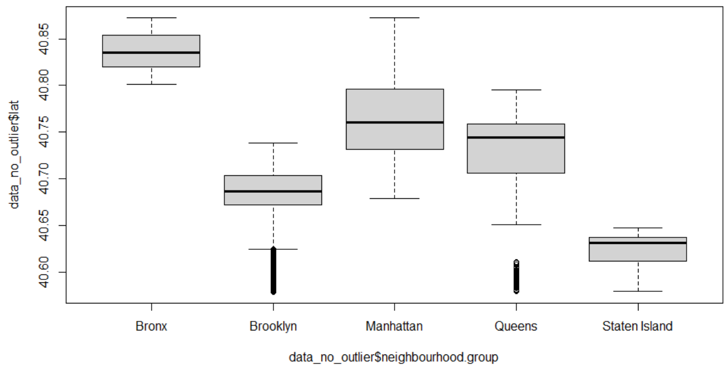
However we could just use the bulletin Bonferroni correction like this:

Our test would be like this:

```
> pairwise.t.test(data_no_outlier$lat, data_no_outlier$neighbourhood.group, p.adj = "bonf", pool.sd = FALSE)

	Pairwise comparisons using t tests with non-pooled SD 

data:  data_no_outlier$lat and data_no_outlier$neighbourhood.group 

              Bronx  Brooklyn Manhattan Queens
Brooklyn      <2e-16 -        -         -     
Manhattan     <2e-16 <2e-16   -         -     
Queens        <2e-16 <2e-16   <2e-16    -     
Staten Island <2e-16 <2e-16   <2e-16    <2e-16

P value adjustment method: bonferroni 
```

Based on the p-value in the pairwise comparison of each group, because in all the comparison $p - value < \alpha$. We reject the null hypothesis for each pair. Therefore, we conclude that the mean latitude is not equal in any of the groups.

The boxplot for this variable supports our findings too:



2- Price:

A)

Hypothesis

H_0: The mean price is the same across all the mentioned neighborhood group.

$$\mu_1 = \mu_2 = \mu_3 = \mu_4 = \mu_5$$

H_A: The mean price differs between at least one pair of the mentioned neighborhood group.

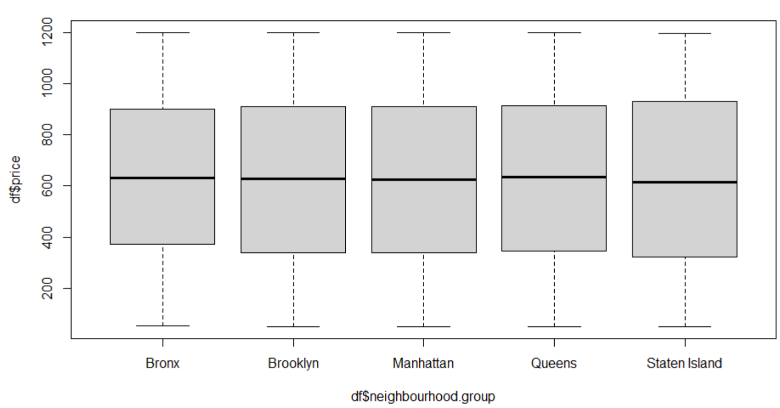
Our test would look like this:

```
> result <- aov( price~neighbourhood.group, data = df)
> summary(result)
                       Df    Sum Sq Mean Sq F value Pr(>F)
neighbourhood.group     4 2.699e+05   67464   0.617   0.65
Residuals           29952 3.275e+09  109334
```

B)

Based on the calculated $p - value > \alpha = 0.05$. We reject the H_A hypothesis. In other words, we conclude that the data suggests that the mean price of all the neighborhood group are equal.

This result is supported based on the boxplot too:



We can to a pairwise comparison. But because ANOVA test resulted in means being equal we expect the same result in each test.

Our hypothesis for each pair wise comparison would be like:

H_0: The mean price in neighborhood group i and neighborhood group j is the same.

$$\mu_i - \mu_j = 0$$

H_A: The mean price in neighborhood group i and neighborhood group j is not the same.

$$\mu_i - \mu_j \neq 0$$

For the above pairwise test, we need to modify α based on Bonferroni correction therefore we have:

$$\alpha^* = \frac{\alpha}{5} = 0.01$$

However we could just use the bulletin Bonferroni correction like this:

Our test would be like this:

```
> pairwise.t.test(df$price, df$neighbourhood.group, p.adj = "bonf", pool.sd = FALSE)

        Pairwise comparisons using t tests with non-pooled SD

data:  df$price and df$neighbourhood.group

              Bronx Brooklyn Manhattan Queens
Brooklyn      1     -        -         -
Manhattan     1     1        -         -
Queens        1     1        1         -
Staten Island 1     1        1         1

P value adjustment method: bonferroni
```

Based on the p-value in the pairwise comparison of each group, because in all the comparison $p-value > \alpha$. We reject the null hypothesis for each pair. Therefore, we conclude, this data suggests that the mean price in all the groups are equal.