Airbnb Booking Price Prediction

Yuanyuan Lin

11/25/2019

```
library(randomForest)
library(lmerTest)
library(car)
library(ggplot2)
library(gridExtra)
library(stats)
library(lmtest)
```

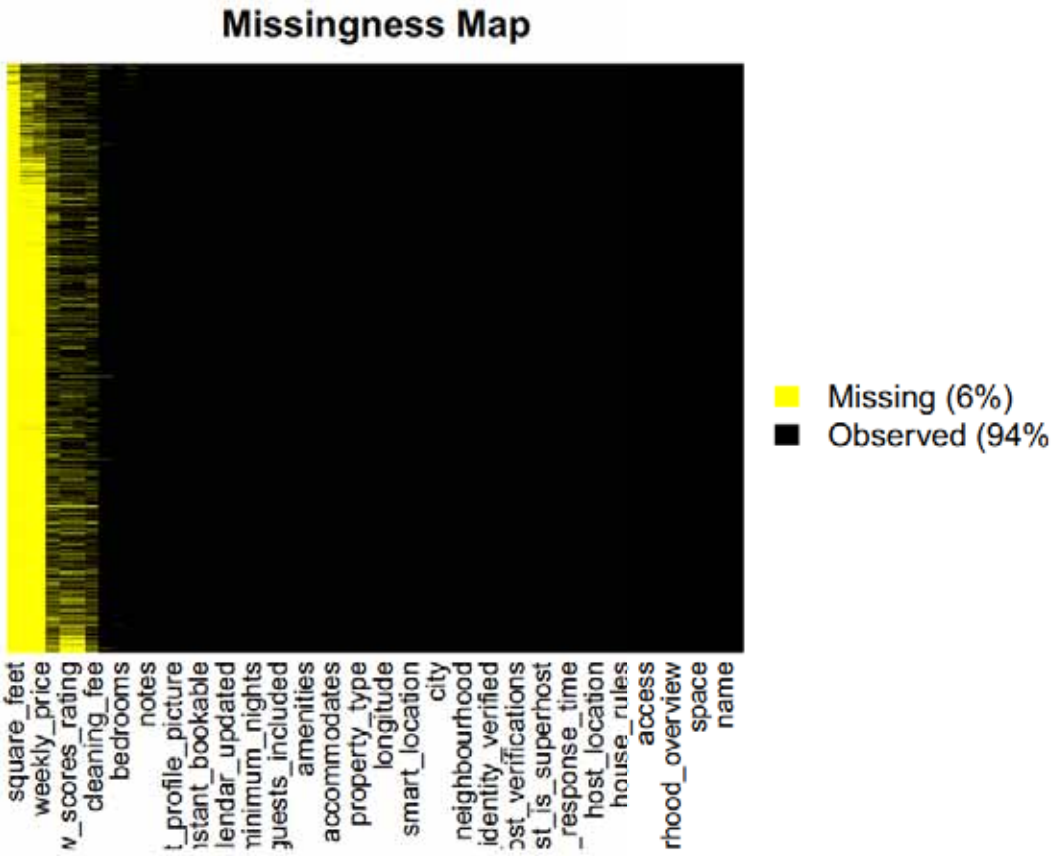
Data Cleaning

```
list=read.csv("listings.csv")
list<-list[,-c(2,3,4,9,16,17,18,19,20,21,23,28,30,31,33,34,41,43,44,47,48,70,71,72,73,74,75,77,78,79,80
```

```
list_sum<-read.csv("listings_sum.csv")
#review_final<-read.csv("reviews_final.csv")
# Select 5000 random rows
#review<-review_final[sample(nrow(review_final), 5000), ]
#write.csv(review,file="review.csv")
```

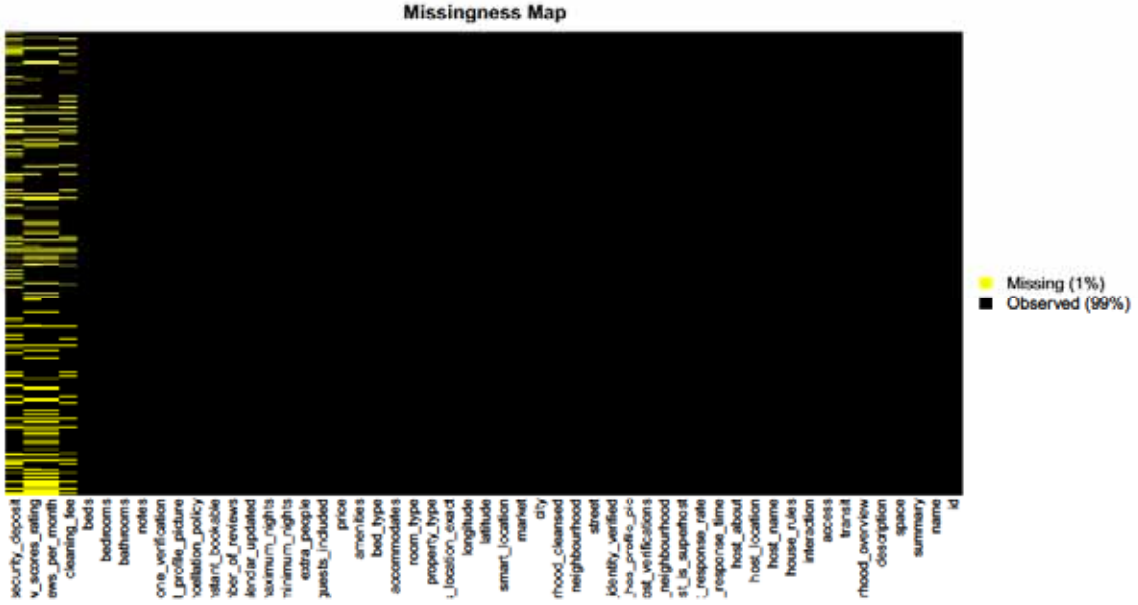
```
review<-read.csv("review.csv")
```

```
library(Amelia)
#Check any NA
missmap(list,col=c('yellow','black'),y.at=1,y.labels='',legend=TRUE)
```




```
#drop irrelevant columns
list<-list[,-c(39,41,42)]
```

```
#check proportition of NA in whole dataset
missmap(list,col=c('yellow','black'),y.at=1,y.labels='',legend=TRUE)
```



```
list$host_since<-list_sum$host_since
list[list==""]<-NA
#drop missing values completely
list<-na.omit(list)
```

```
write.csv(list,file="list.csv")
```

##Text Analysis on Review

analyzing customers review could help to know customer's booking patterns and trends.

```
#remove rows that have N/A in "host_response_time","host_response_rate""
library(dplyr)
list = filter(list, host_response_time != "N/A" & host_response_rate != "N/A")
```

```
#write.csv(list,file="airbnb.csv")
```

```
#Display the data dimensions
dim(list)
```

```
## [1] 6700   54
```

```
# Display the column names
colnames(list)
```

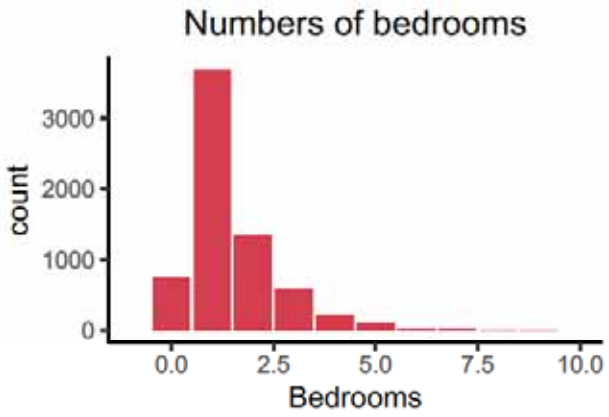
```
##  [1] "id"                                "name"
##  [3] "summary"                           "space"
##  [5] "description"                       "neighborhood_overview"
##  [7] "notes"                             "transit"
##  [9] "access"                            "interaction"
## [11] "house_rules"                       "host_name"
## [13] "host_location"                     "host_about"
## [15] "host_response_time"                "host_response_rate"
## [17] "host_is_superhost"                 "host_neighbourhood"
## [19] "host_verifications"                "host_has_profile_pic"
## [21] "host_identity_verified"            "street"
## [23] "neighbourhood"                     "neighbourhood_cleansed"
## [25] "city"                              "market"
## [27] "smart_location"                    "latitude"
## [29] "longitude"                         "is_location_exact"
## [31] "property_type"                     "room_type"
## [33] "accommodates"                      "bathrooms"
## [35] "bedrooms"                          "beds"
## [37] "bed_type"                          "amenities"
## [39] "price"                             "security_deposit"
## [41] "cleaning_fee"                      "guests_included"
## [43] "extra_people"                      "minimum_nights"
## [45] "maximum_nights"                    "calendar_updated"
## [47] "number_of_reviews"                 "review_scores_rating"
## [49] "instant_bookable"                  "cancellation_policy"
## [51] "require_guest_profile_picture"     "require_guest_phone_verification"
## [53] "reviews_per_month"                 "host_since"
```

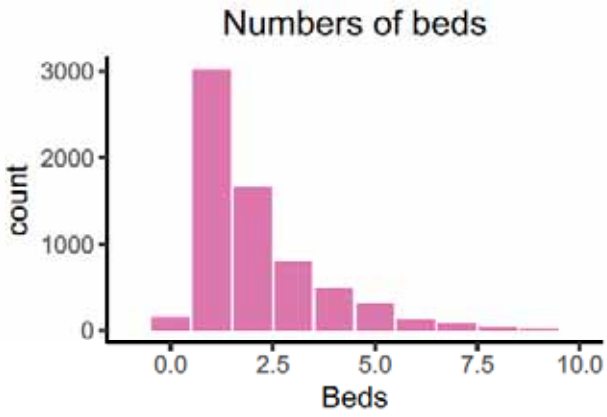
```r
# Display the data structures
str(list)
```

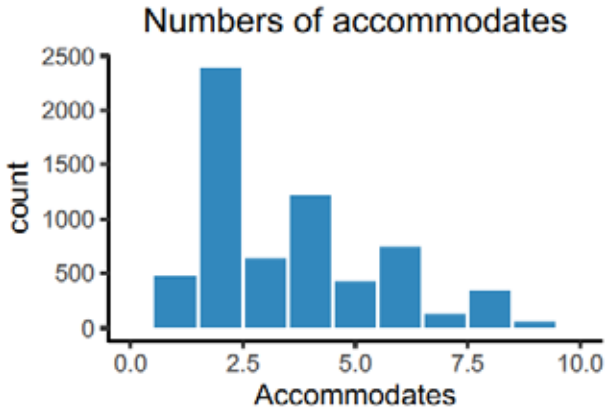
```
## 'data.frame':    6700 obs. of  54 variables:
##  $ id                              : int  344 2708 5728 5729 5843 6931 7992 15089 15440 15766 ...
##  $ name                            : Factor w/ 44172 levels "","1 bedroom modern apt ",..: 16629 16
##  $ summary                         : Factor w/ 39058 levels "","          This apartment is ideal for
##  $ space                           : Factor w/ 30038 levels "","\tBUILT BY HOWARD HUGHES!!  ATTENTIO
##  $ description                     : Factor w/ 41394 levels "","\tBUILT BY HOWARD HUGHES!!  ATTENTIO
##  $ neighborhood_overview           : Factor w/ 24792 levels "","  Prime location near Beverly Hills S
##  $ notes                           : Factor w/ 18497 levels "","\tThe loft has an independent entran
##  $ transit                         : Factor w/ 22728 levels "","  The apartment is within close prox
##  $ access                          : Factor w/ 22703 levels "","\tThe loft has an independent entran
##  $ interaction                     : Factor w/ 22942 levels "","  Available 24/7 via Airbnb,Text & ph
##  $ house_rules                     : Factor w/ 25580 levels "","_ My room is downstairs and your sui
##  $ host_name                       : Factor w/ 8933 levels "","\"M\"Ac",""Quincy"",..: 5536 1506 708
##  $ host_location                   : Factor w/ 1009 levels "","  California, United States",..: 122 5
##  $ host_about                      : Factor w/ 15726 levels "","\n","\n\n",..: 13628 15396 14648 146
##  $ host_response_time              : Factor w/ 6 levels "","a few days or more",..: 4 6 6 6 6 6 6 5
##  $ host_response_rate              : Factor w/ 71 levels "","0","0.1","0.13",..: 22 70 70 70 70 70 7
##  $ host_is_superhost               : Factor w/ 3 levels "","f","t": 2 3 3 3 3 3 3 2 3 2 ...
##  $ host_neighbourhood              : Factor w/ 409 levels "","Ala Moana/Kakaako",..: 46 142 83 83 83
##  $ host_verifications              : Factor w/ 516 levels "['email', 'facebook', 'reviews', 'jumio',
##  $ host_has_profile_pic            : Factor w/ 3 levels "","f","t": 3 3 3 3 3 3 3 3 3 3 ...
##  $ host_identity_verified          : Factor w/ 3 levels "","f","t": 3 3 2 2 2 3 3 3 2 3 ...
##  $ street                          : Factor w/ 430 levels " Encino, CA, United States",..: 39 160 16
##  $ neighbourhood                   : Factor w/ 165 levels "","Alhambra",..: 22 61 34 34 34 61 9 87 1
##  $ neighbourhood_cleansed          : Factor w/ 265 levels "Acton","Adams-Normandie",..: 32 102 57 57
##  $ city                            : Factor w/ 410 levels "","  Encino","  Los Angeles",..: 37 141 141
##  $ market                          : Factor w/ 30 levels "","Cabo San Lucas",..: 14 14 14 14 14 14 1
##  $ smart_location                  : Factor w/ 430 levels " Encino, CA",..: 39 160 160 160 160 160 1
##  $ latitude                        : num  34.2 34.1 34 34 34 ...
##  $ longitude                       : num  -118 -118 -118 -118 -118 ...
##  $ is_location_exact               : Factor w/ 2 levels "f","t": 2 2 2 2 2 2 2 2 2 2 ...
##  $ property_type                   : Factor w/ 45 levels "Aparthotel","Apartment",..: 26 2 38 23 26
##  $ room_type                       : Factor w/ 4 levels "Entire home/apt",..: 1 3 3 3 1 3 1 1 1 1 ..
##  $ accommodates                    : int  6 1 2 3 5 1 4 2 2 2 ...
##  $ bathrooms                       : num  1 1.5 1 1 1 1.5 1 1 1 1 ...
##  $ bedrooms                        : int  3 1 1 1 2 1 1 1 0 2 ...
##  $ beds                            : int  3 1 1 2 2 1 2 1 1 2 ...
##  $ bed_type                        : Factor w/ 5 levels "Airbed","Couch",..: 5 5 5 1 5 5 5 5 5 5 ...
##  $ amenities                       : Factor w/ 41546 levels "{\"Air conditioning\",\"Carbon monoxide
##  $ price                           : int  168 79 75 105 303 99 90 89 140 120 ...
##  $ security_deposit                : int  0 480 100 150 200 400 100 500 250 0 ...
##  $ cleaning_fee                    : int  100 87 25 50 100 87 35 99 0 150 ...
##  $ guests_included                 : int  6 1 1 2 2 1 2 2 2 1 ...
##  $ extra_people                    : int  0 0 15 15 15 41 10 50 0 0 ...
##  $ minimum_nights                  : int  2 28 1 1 1 28 1 3 2 3 ...
##  $ maximum_nights                  : int  14 366 14 29 1120 366 1250 365 31 30 ...
##  $ calendar_updated                : Factor w/ 86 levels "1 week ago","10 months ago",..: 75 39 83 8
##  $ number_of_reviews               : int  6 21 287 196 104 17 207 54 766 3 ...
##  $ review_scores_rating            : int  93 98 96 95 92 96 99 96 99 100 ...
##  $ instant_bookable                : Factor w/ 2 levels "f","t": 2 2 2 2 2 2 2 1 2 1 ...
```

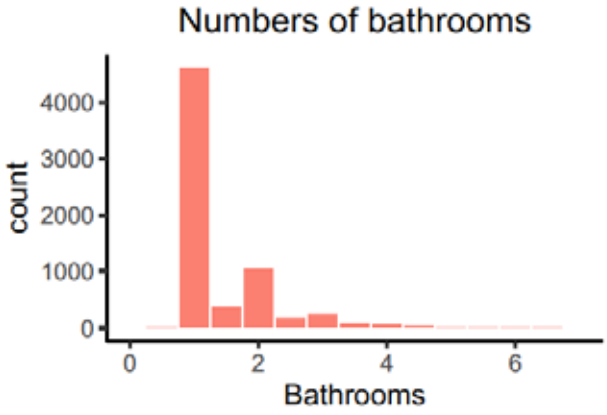
```
##  $ cancellation_policy            : Factor w/ 9 levels "flexible","luxury_moderate",..: 1 7 5 5 5 7
##  $ require_guest_profile_picture  : Factor w/ 2 levels "f","t": 1 1 1 1 1 1 1 2 1 1 ...
##  $ require_guest_phone_verification: Factor w/ 2 levels "f","t": 1 1 1 1 1 1 1 2 2 1 ...
##  $ reviews_per_month              : num  0.15 0.33 2.32 1.6 1.03 0.14 2.16 0.45 6.51 0.22 ...
##  $ host_since                     : Factor w/ 3434 levels "","2008-03-03",..: 3 5 16 16 16 5 35 77 
##  - attr(*, "na.action")= 'omit' Named int  1 4 5 9 11 13 14 15 16 17 ...
##   ..- attr(*, "names")= chr  "1" "4" "5" "9" ...
```

```
#review/summaries text analysis
library(gridExtra)
library(grid)
par(mfrow=c(4,4))
library(plotly)
library(ggthemes)

g<-ggplot(data = list) +
  geom_bar(aes(x = bedrooms),fill="#D53E4F") +
  xlab('Bedrooms') +
  labs(title = "Numbers of bedrooms")+xlim(-1, 10)+
  theme(plot.title = element_text(hjust = 0.5,size=13),panel.grid.major =element_blank(), panel.grid.mi
panel.background = element_blank(),axis.line = element_line(colour = "black"))


h<-ggplot(data = list) +
  geom_bar(aes(x = beds),fill="#DE77AE") +
  xlab('Beds') +
  labs(title = "Numbers of beds")+xlim(-1, 10)+
  theme(plot.title = element_text(hjust = 0.5,size=13),panel.grid.major =element_blank(), panel.grid.mi
panel.background = element_blank(),axis.line = element_line(colour = "black"))

j<-ggplot(data = list) +
  geom_bar(aes(x = accommodates),fill="#3288BD") +
  xlab('Accommodates') +
  labs(title = "Numbers of accommodates")+xlim(0, 10)+
  theme(plot.title = element_text(hjust = 0.5,size=13),panel.grid.major =element_blank(), panel.grid.mi
panel.background = element_blank(),axis.line = element_line(colour = "black"))

i<-ggplot(data = list) +
  geom_bar(aes(x = bathrooms),fill="#FB8072") +
  xlab('Bathrooms') +
  labs(title = "Numbers of bathrooms")+xlim(0, 7)+
  theme(plot.title = element_text(hjust = 0.5,size=13),panel.grid.major =element_blank(), panel.grid.mi
panel.background = element_blank(),axis.line = element_line(colour = "black"))



grid.arrange(g, h, j,i ,ncol=2)
```

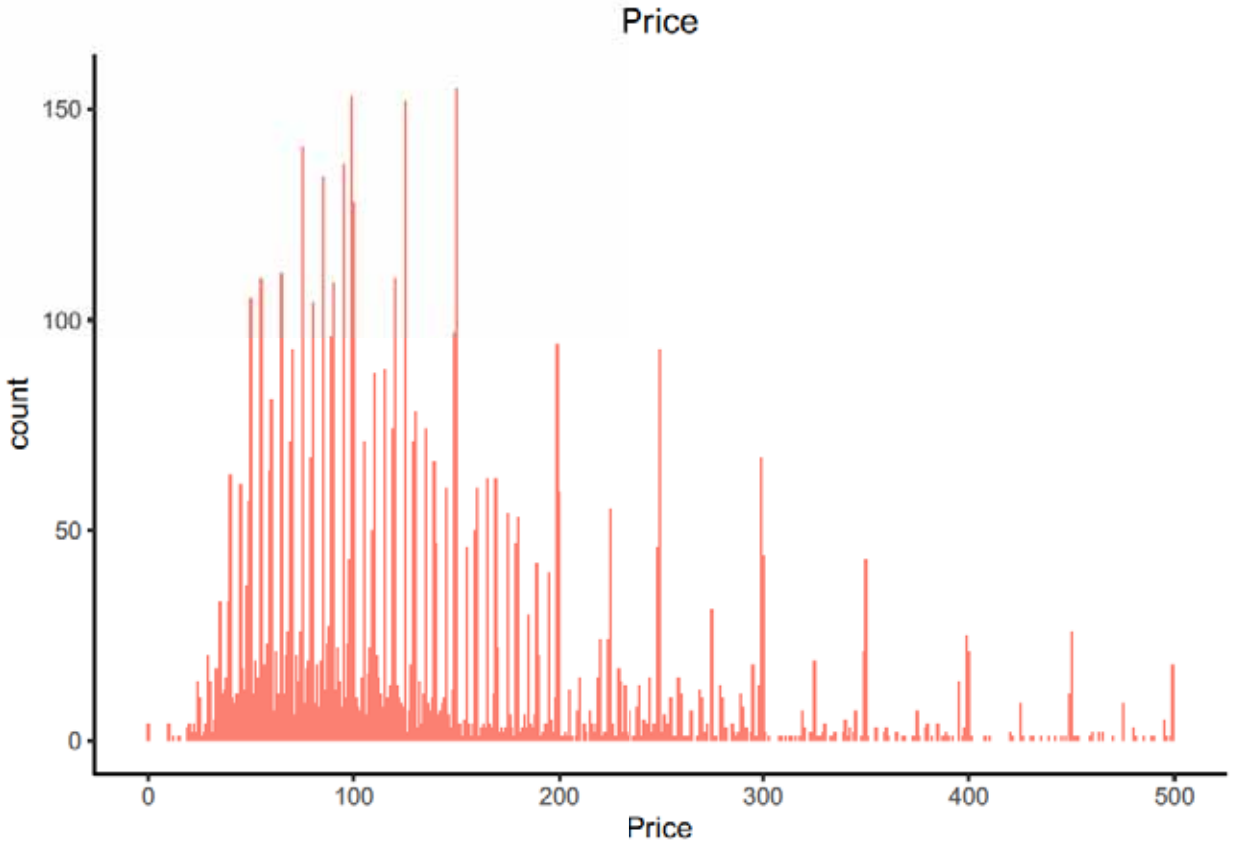









```
#price analysis
ggplot(data = list) +
  geom_bar(aes(x = price),fill="#FB8072") +
  xlab('Price') +
  labs(title = "Price")+xlim(-1, 500)+
  theme(plot.title = element_text(hjust = 0.5,size=13),panel.grid.major =element_blank(), panel.grid.mi
panel.background = element_blank(),axis.line = element_line(colour = "black"))
```




```
#neighborhood analysis
library(dplyr)
library(kableExtra)
list1<-list%>%group_by(list$neighbourhood) %>% summarise(number = n())%>%arrange(desc(number))
head(list1)
```

```
## # A tibble: 6 x 2
##   `list$neighbourhood` number
##   <fct>                 <int>
## 1 Hollywood               697
## 2 Mid-Wilshire            557
## 3 Venice                  512
## 4 Long Beach              343
## 5 Downtown                267
## 6 West Hollywood          185
```

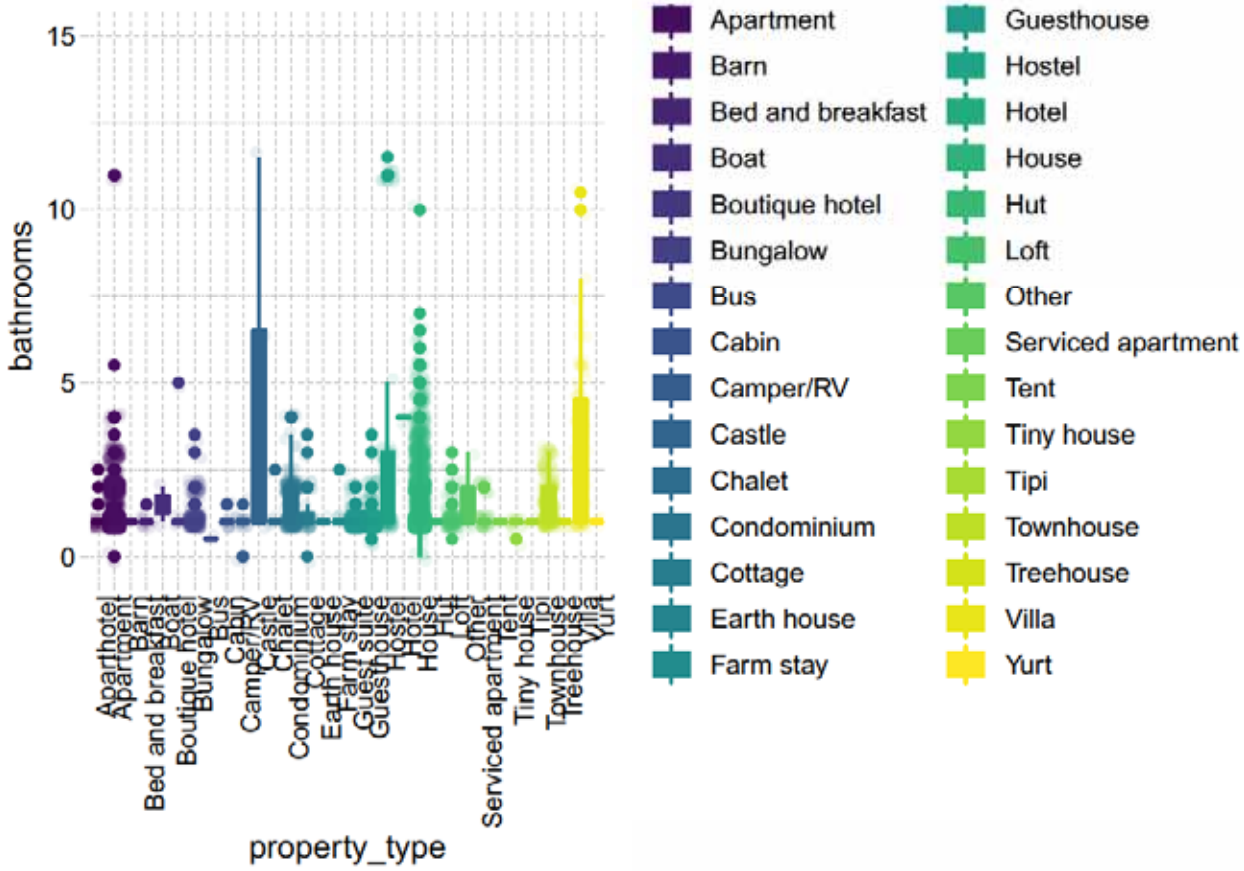
```
Latitude<-list[,28]
Latitude<-data.frame(Latitude)
Latitude$long<-list[,29]
```

```
Latitude_sample<-Latitude[sample(nrow(Latitude), 100), ]
```

```
# get the location of housing
#library(leaflet)
#Latitude_sample %>%
```

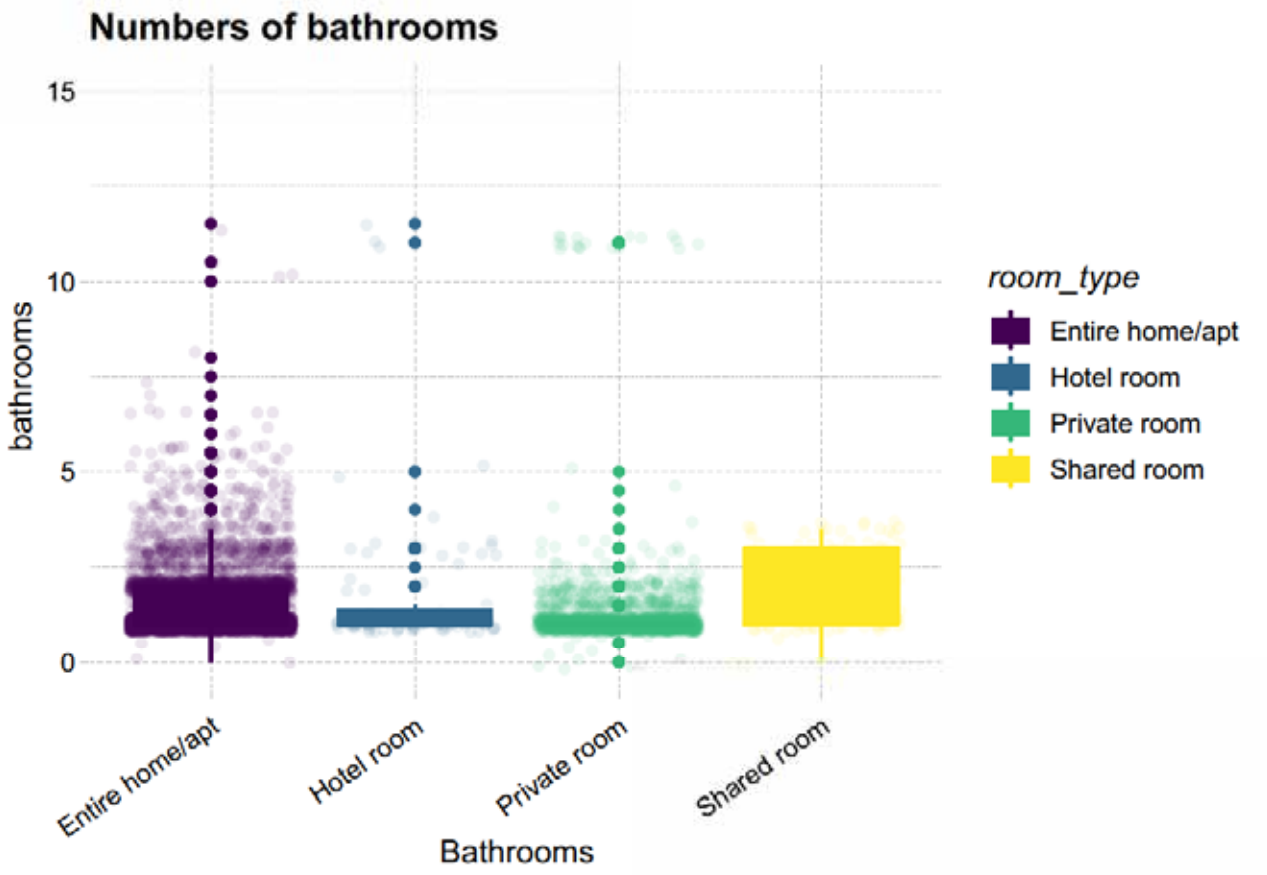
```
  #leaflet() %>%
 # addTiles() %>%
 # addTiles() %>%
#addMarkers(popup="sites")
```

```
#property type analysis

list$property_type = as.factor(list$property_type)
ggplot(aes(x = property_type, y = bathrooms,color=property_type,fill=property_type), data = list) +
  geom_boxplot() +
  geom_jitter(alpha = 0.1)+
  coord_cartesian(ylim=c(0,15))+
theme( axis.text.x  = element_text(angle=90, hjust=1, vjust=0.9))+
 scale_fill_viridis_d(option = "viridis") +
 scale_color_viridis_d(option = "viridis") +
 theme_pander()
```

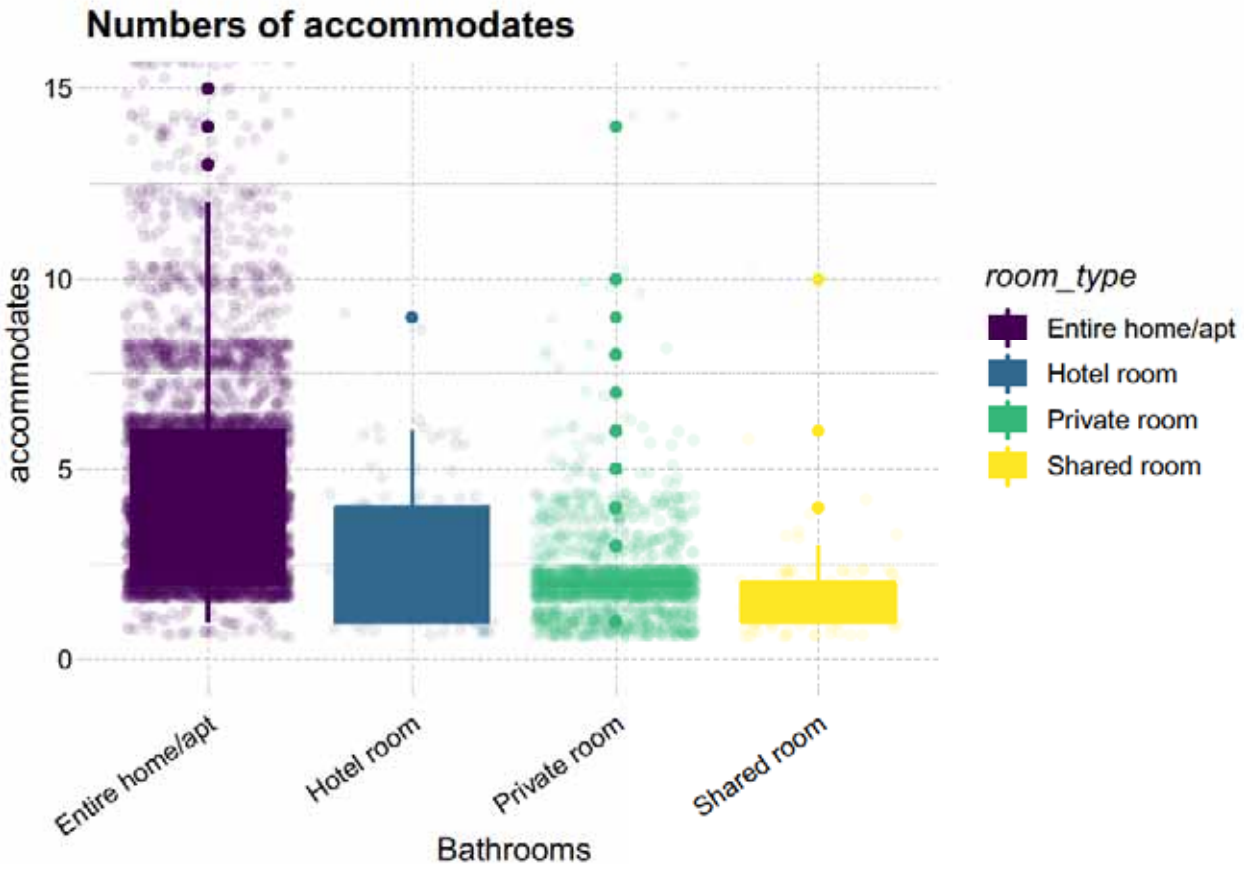



```
#room type analysis
par(mfrow=c(4,4))
list$room_typee = as.factor(list$room_type)
ggplot(aes(x = room_type, y = bathrooms,color=room_type,fill=room_type), data = list) +
  geom_boxplot() +
  geom_jitter(alpha = 0.1)+
  coord_cartesian(ylim=c(0,15))+
theme( axis.text.x  = element_text(angle=35, hjust=1, vjust=0.9))+
```

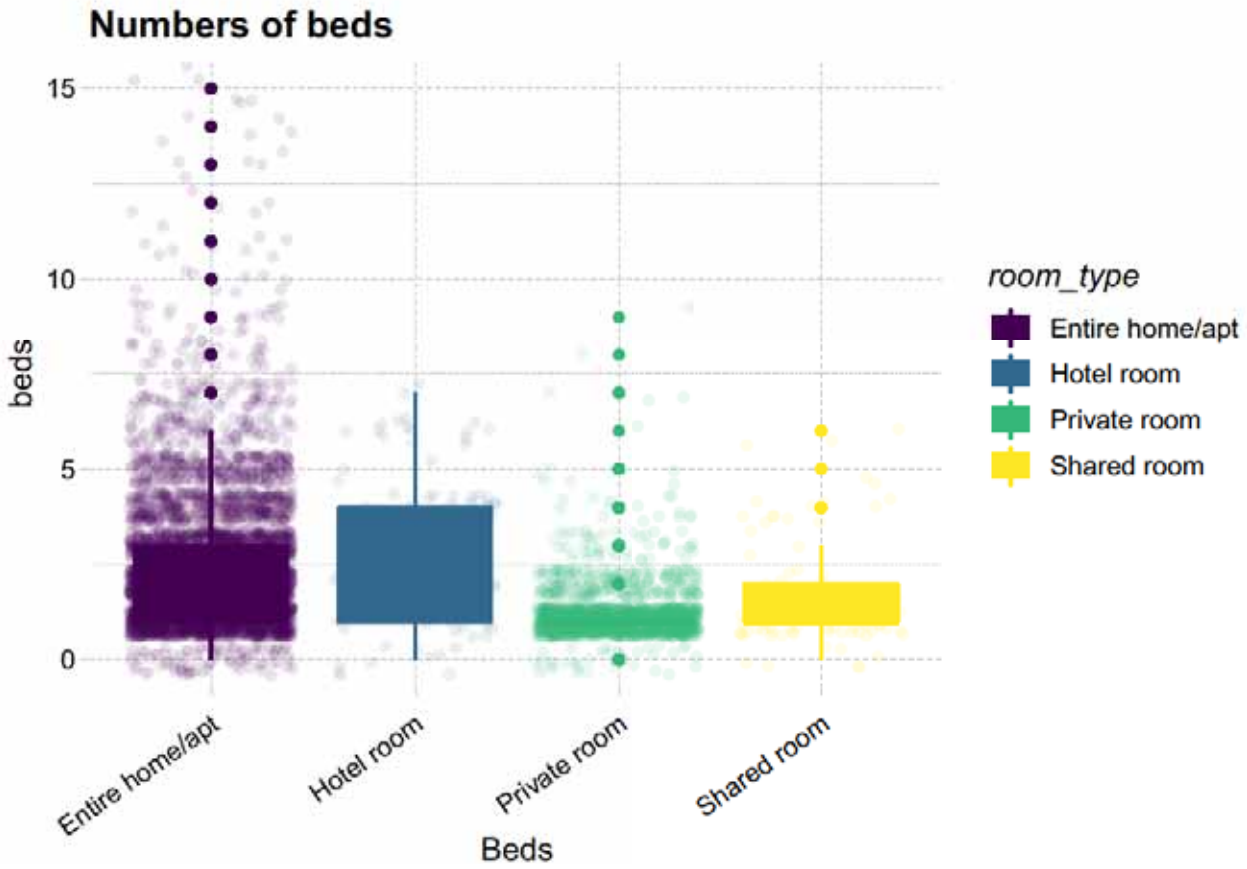
```
 xlab('Bathrooms') +
  labs(title = "Numbers of bathrooms")+
  scale_fill_viridis_d(option = "viridis") +
 scale_color_viridis_d(option = "viridis") +
 theme_pander()
```



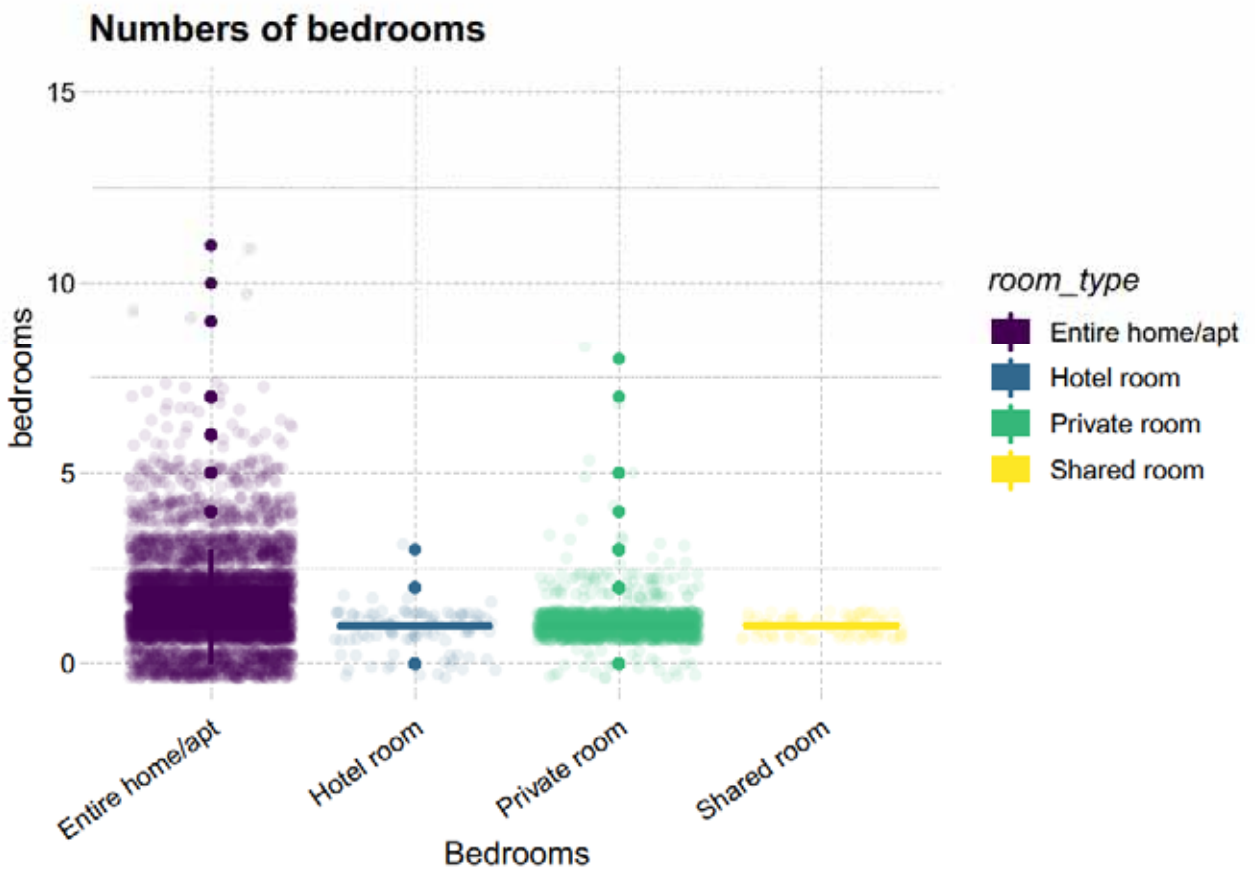

```
ggplot(aes(x = room_type, y = accommodates,color=room_type,fill=room_type), data = list) +
  geom_boxplot() +
  geom_jitter(alpha = 0.1)+
  coord_cartesian(ylim=c(0,15))+
theme( axis.text.x  = element_text(angle=35, hjust=1, vjust=0.9))+
 xlab('Bathrooms') +
  labs(title = "Numbers of accommodates")+
  scale_fill_viridis_d(option = "viridis") +
 scale_color_viridis_d(option = "viridis") +
 theme_pander()
```



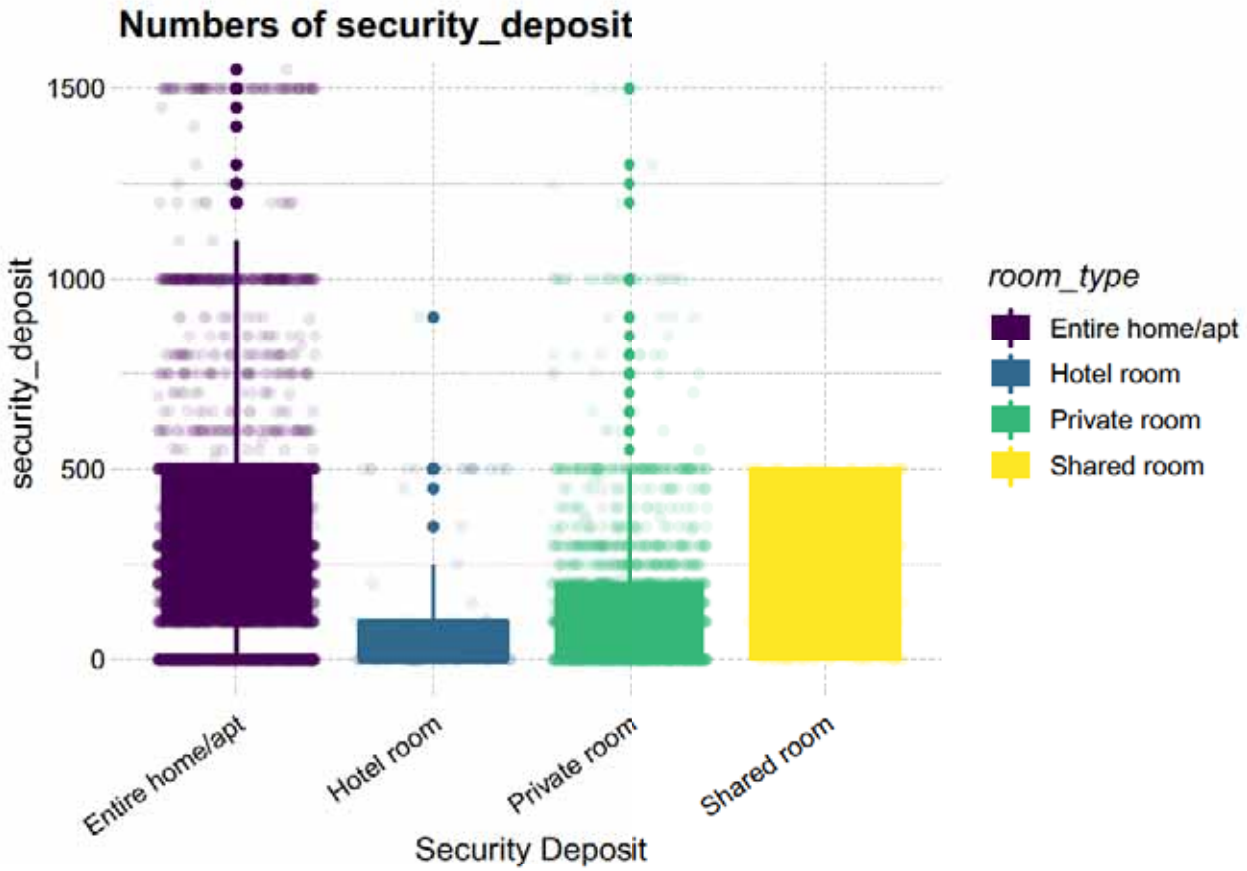

```
ggplot(aes(x = room_type, y = beds,color=room_type,fill=room_type), data = list) +
  geom_boxplot() +
  geom_jitter(alpha = 0.1)+
  coord_cartesian(ylim=c(0,15))+
theme( axis.text.x  = element_text(angle=35, hjust=1, vjust=0.9))+
 xlab('Beds') +
  labs(title = "Numbers of beds")+
  scale_fill_viridis_d(option = "viridis") +
 scale_color_viridis_d(option = "viridis") +
 theme_pander()
```



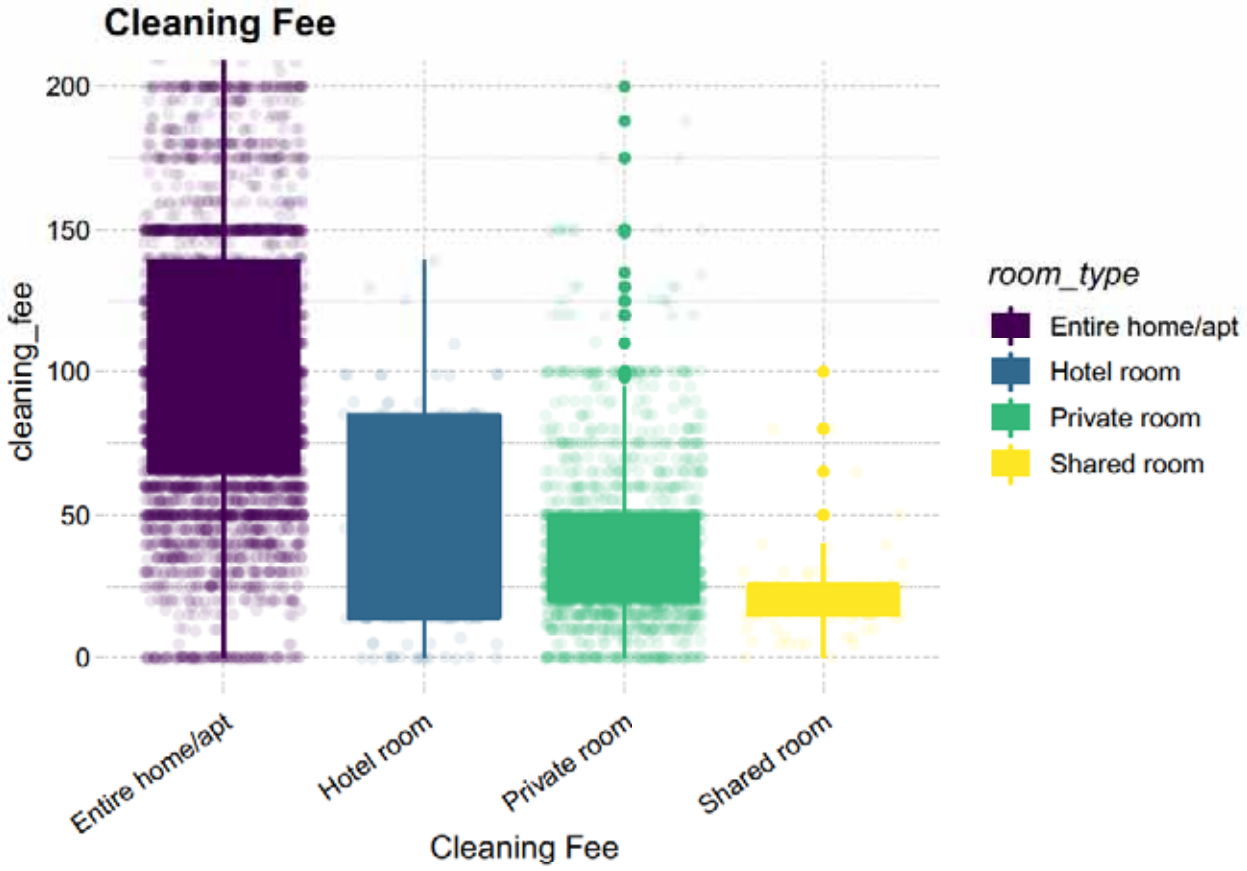

```
ggplot(aes(x = room_type, y = bedrooms,color=room_type,fill=room_type), data = list) +
  geom_boxplot() +
  geom_jitter(alpha = 0.1)+
  coord_cartesian(ylim=c(0,15))+
theme( axis.text.x  = element_text(angle=35, hjust=1, vjust=0.9))+
 xlab('Bedrooms') +
  labs(title = "Numbers of bedrooms")+
  scale_fill_viridis_d(option = "viridis") +
 scale_color_viridis_d(option = "viridis") +
 theme_pander()
```



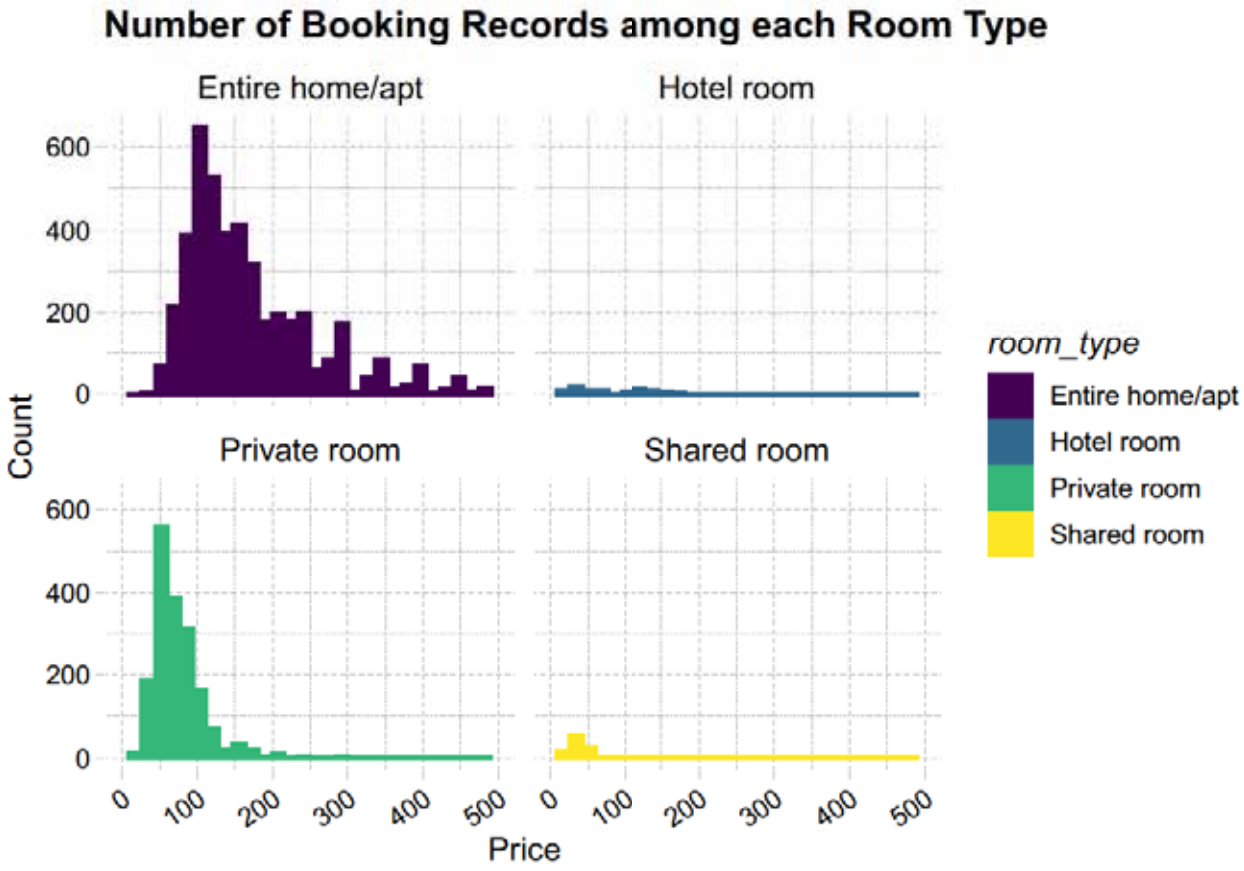

```
ggplot(aes(x = room_type, y = security_deposit,color=room_type,fill=room_type), data = list) +
  geom_boxplot() +
  geom_jitter(alpha = 0.1)+
  coord_cartesian(ylim=c(0,1500))+
theme( axis.text.x  = element_text(angle=35, hjust=1, vjust=0.9))+
 xlab('Security Deposit') +
  labs(title = "Numbers of security_deposit")+
  scale_fill_viridis_d(option = "viridis") +
 scale_color_viridis_d(option = "viridis") +
 theme_pander()
```




```
ggplot(aes(x = room_type, y = cleaning_fee,color=room_type,fill=room_type), data = list) +
  geom_boxplot() +
  geom_jitter(alpha = 0.1)+
  coord_cartesian(ylim=c(0,200))+
theme( axis.text.x  = element_text(angle=35, hjust=1, vjust=0.9))+
 xlab('Cleaning Fee') +
  labs(title = "Cleaning Fee")+
  scale_fill_viridis_d(option = "viridis") +
 scale_color_viridis_d(option = "viridis") +
 theme_pander()
```




```
#number of booking records based on room type
m<-ggplot(aes(x = price,fill=room_type,color=room_type), data = list) +
  geom_histogram()+
  facet_wrap(~room_type)+xlim(0, 500)+
  theme( axis.text.x  = element_text(angle=35, hjust=1, vjust=0.9))+
 labs(x = "Price", y = " Count") +
  ggtitle("Number of Booking Records among each Room Type")+
  scale_fill_viridis_d(option = "viridis") +
 scale_color_viridis_d(option = "viridis") +
 theme_pander()
m
```



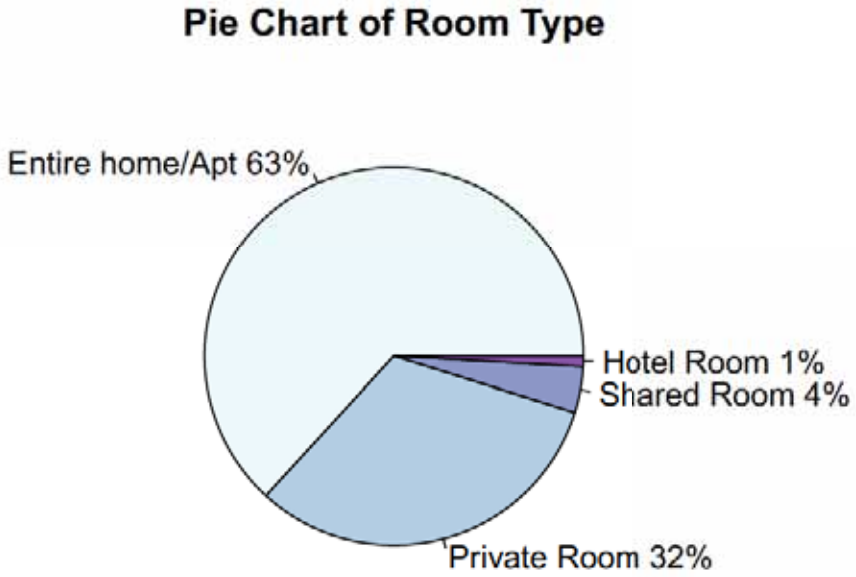

```
#room type numbers in each category
library(dplyr)
```

```
list_roomtype<-list%>%group_by(list$room_type) %>% summarise(number = n())%>%arrange(desc(number))
library(knitr)
kable(list_roomtype,format = "markdown")
```

list$room_type	number
Entire home/apt	4693
Private room	1805
Hotel room	102
Shared room	100

```
library("RColorBrewer")
#pie chart
# Pie Chart with Percentages
slices <- c(28468, 14410, 1769, 406)
lbls <- c("Entire home/Apt", "Private Room", "Shared Room", "Hotel Room")
pct <- round(slices/sum(slices)*100)
lbls <- paste(lbls, pct) # add percents to labels
lbls <- paste(lbls,"%",sep="") # ad % to labels
coul <- brewer.pal(5, "BuPu")
pie(slices,labels = lbls, col=coul,
```

```
  main="Pie Chart of Room Type")
```




```
library(tidyverse)
library(tidytext)
library(knitr)
library(textdata)
library(magrittr)

summary<-data.frame(list$summary)
summary$list.summary<-as.character(summary$list.summary)
tidy_word <- summary %>%
  unnest_tokens(word,list.summary)
```

```
#find the most frequently used words in summary
library(wordcloud)
library(magrittr)
tidy_word %>%
  anti_join(stop_words) %>%
  count(word) %>%
  with(wordcloud(word, n, max.words = 20,colors = brewer.pal(7, 'Dark2'), random.order = FALSE,rot.per=
```


```
review<-na.omit(review)
```

```
comment<-data.frame(review$comments)
comment$review.comments<-as.character(comment$review.comments)
tidy_word_com <- comment %>%
  unnest_tokens(word,review.comments)
```

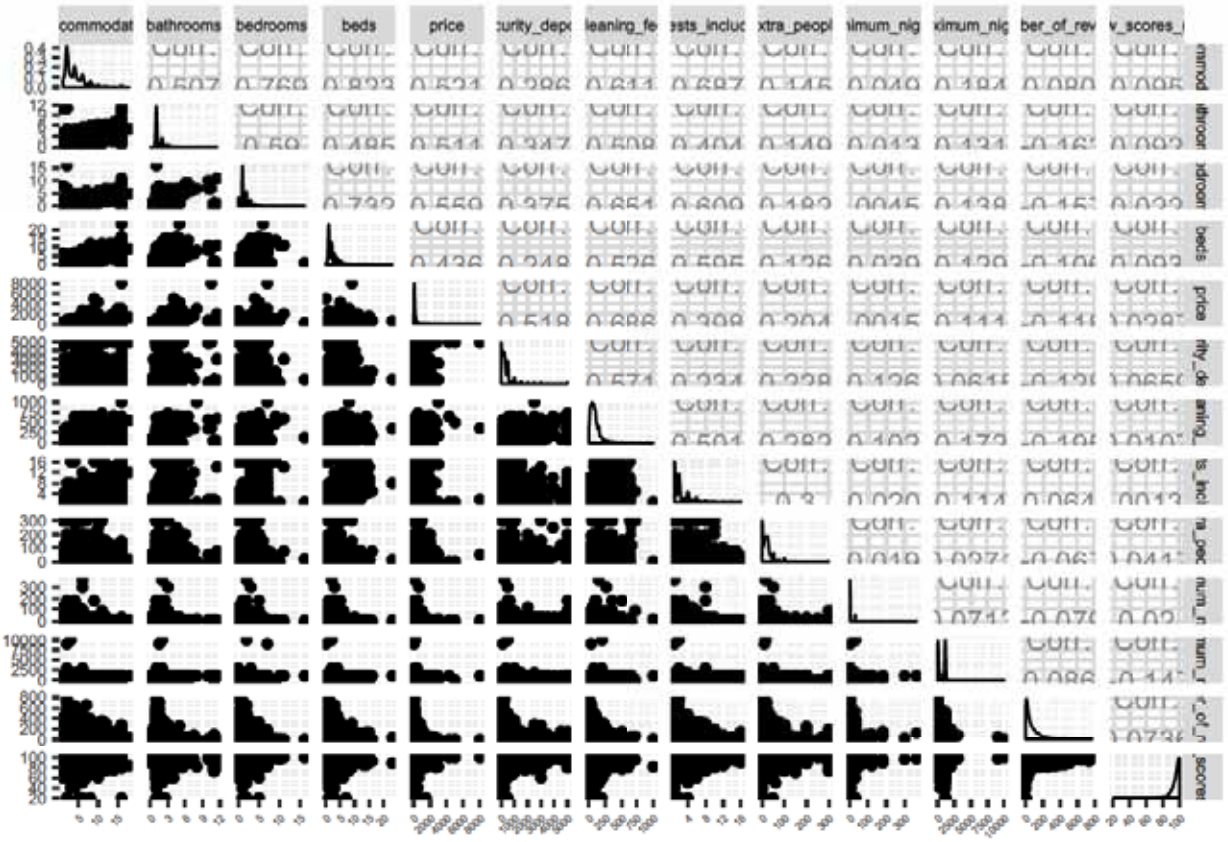
```
tidy_word_com %>%
  anti_join(stop_words) %>%
  count(word) %>%
  with(wordcloud(word, n, max.words = 100,colors = brewer.pal(7, 'Dark2'), random.order = FALSE,rot.per
```


```
#get relevant columns in the dataset for regression
model_data<-list[,c(33:36,39:45,47,48)]
head(model_data)
```

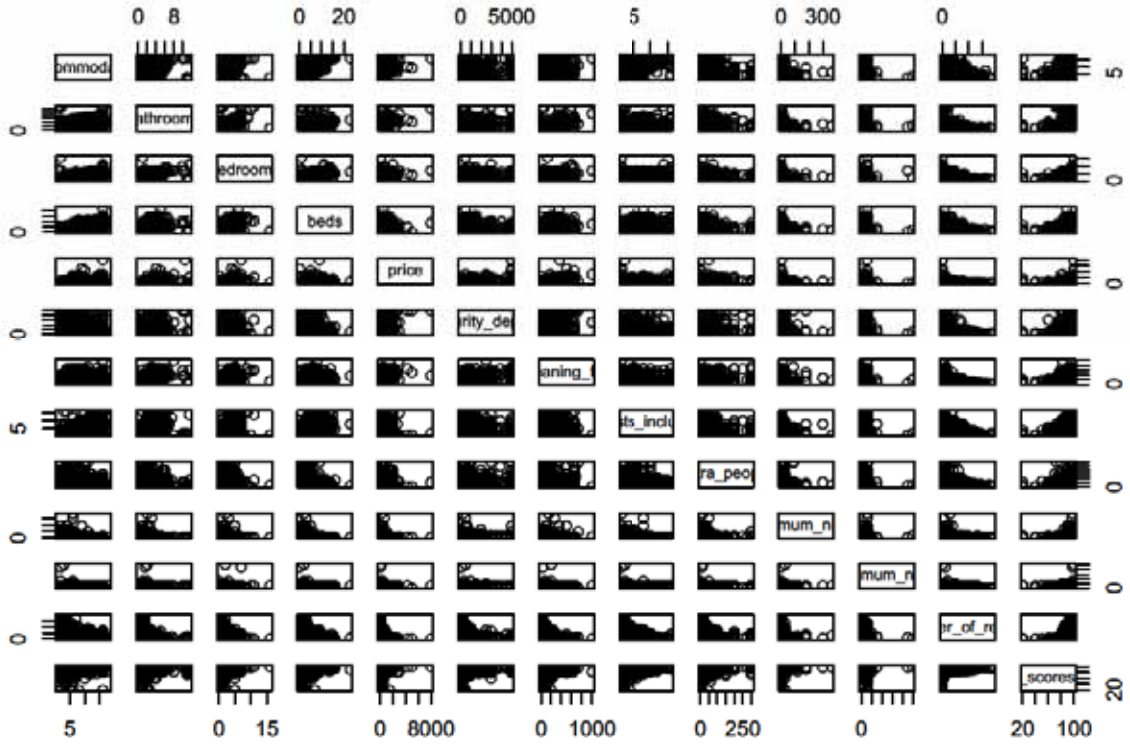
```
##   accommodates bathrooms bedrooms beds price security_deposit cleaning_fee
## 1            6       1.0        3    3   168                0           100
## 2            1       1.5        1    1    79              480            87
## 3            2       1.0        1    1    75              100            25
## 4            3       1.0        1    2   105              150            50
## 5            5       1.0        2    2   303              200           100
## 6            1       1.5        1    1    99              400            87
##   guests_included extra_people minimum_nights maximum_nights number_of_reviews
## 1               6            0              2             14                 6
## 2               1            0             28            366                21
## 3               1           15              1             14               287
## 4               2           15              1             29               196
## 5               2           15              1           1120               104
## 6               1           41             28            366                17
##   review_scores_rating
## 1                   93
## 2                   98
## 3                   96
## 4                   95
## 5                   92
## 6                   96
```

```
#Find the correlation among each variable
set.seed(200)
library(GGally)
ggpairs(model_data,cardinality_threshold = 100) +
```

```
  theme(text = element_text(size = 8)) +
  theme(axis.text.x = element_text(angle = 45, vjust = 1, hjust = 1, size = 4))
```




```
#correlation plot
pairs(model_data)
```

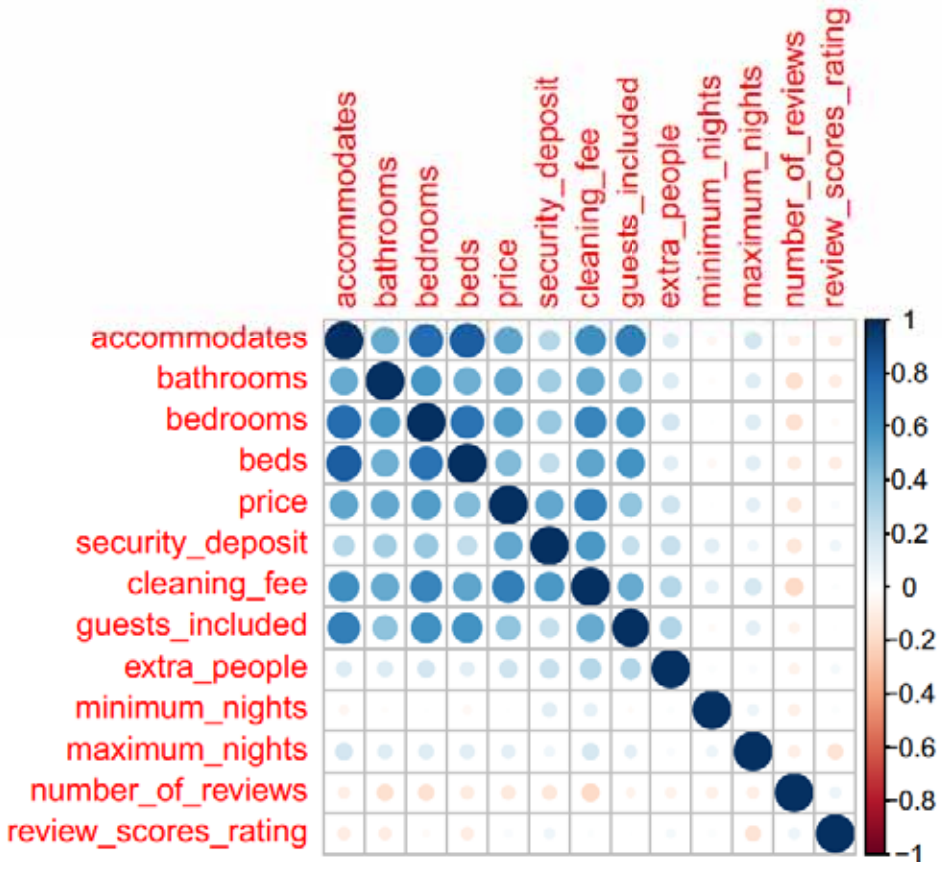



```
#Correlogram
library(corrgram)
library(PerformanceAnalytics)
corMat <- cor(model_data, use = "complete")
round(corMat, 3)
```

```
##                       accommodates bathrooms bedrooms   beds  price
## accommodates                 1.000     0.507    0.769  0.823  0.521
## bathrooms                    0.507     1.000    0.590  0.485  0.511
## bedrooms                     0.769     0.590    1.000  0.732  0.559
## beds                         0.823     0.485    0.732  1.000  0.436
## price                        0.521     0.511    0.559  0.436  1.000
## security_deposit             0.286     0.347    0.375  0.248  0.518
## cleaning_fee                 0.611     0.508    0.651  0.526  0.686
## guests_included              0.687     0.404    0.609  0.595  0.398
## extra_people                 0.145     0.149    0.182  0.126  0.204
## minimum_nights              -0.050    -0.013    0.005 -0.040  0.002
## maximum_nights               0.184     0.131    0.138  0.129  0.111
## number_of_reviews           -0.080    -0.163   -0.157 -0.106 -0.118
## review_scores_rating        -0.096    -0.093   -0.023 -0.093  0.029
##                       security_deposit cleaning_fee guests_included extra_people
## accommodates                     0.286         0.611           0.687        0.145
## bathrooms                        0.347         0.508           0.404        0.149
## bedrooms                         0.375         0.651           0.609        0.182
## beds                             0.248         0.526           0.595        0.126
## price                            0.518         0.686           0.398        0.204
## security_deposit                 1.000         0.571           0.234        0.228
## cleaning_fee                     0.571         1.000           0.501        0.282
## guests_included                  0.234         0.501           1.000        0.300
## extra_people                     0.228         0.282           0.300        1.000
```

```
## minimum_nights                0.126        0.102         -0.021        0.019
## maximum_nights                0.061        0.172          0.114        0.027
## number_of_reviews            -0.128       -0.195         -0.064       -0.067
## review_scores_rating          0.066        0.011          0.001        0.042
##                     minimum_nights maximum_nights number_of_reviews
## accommodates                -0.050          0.184            -0.080
## bathrooms                   -0.013          0.131            -0.163
## bedrooms                     0.005          0.138            -0.157
## beds                        -0.040          0.129            -0.106
## price                        0.002          0.111            -0.118
## security_deposit             0.126          0.061            -0.128
## cleaning_fee                 0.102          0.172            -0.195
## guests_included             -0.021          0.114            -0.064
## extra_people                 0.019          0.027            -0.067
## minimum_nights               1.000          0.071            -0.079
## maximum_nights               0.071          1.000            -0.086
## number_of_reviews           -0.079         -0.086             1.000
## review_scores_rating         0.020         -0.147             0.074
##                     review_scores_rating
## accommodates                      -0.096
## bathrooms                         -0.093
## bedrooms                          -0.023
## beds                              -0.093
## price                              0.029
## security_deposit                   0.066
## cleaning_fee                       0.011
## guests_included                    0.001
## extra_people                       0.042
## minimum_nights                     0.020
## maximum_nights                    -0.147
## number_of_reviews                  0.074
## review_scores_rating               1.000
```

```
library(corrplot)
corrplot(cor(model_data), method = "circle")
```




```
#multiple regression
linear<-lm(price~extra_people+maximum_nights+number_of_reviews+review_scores_rating+guests_included,dat
summary(linear)
```

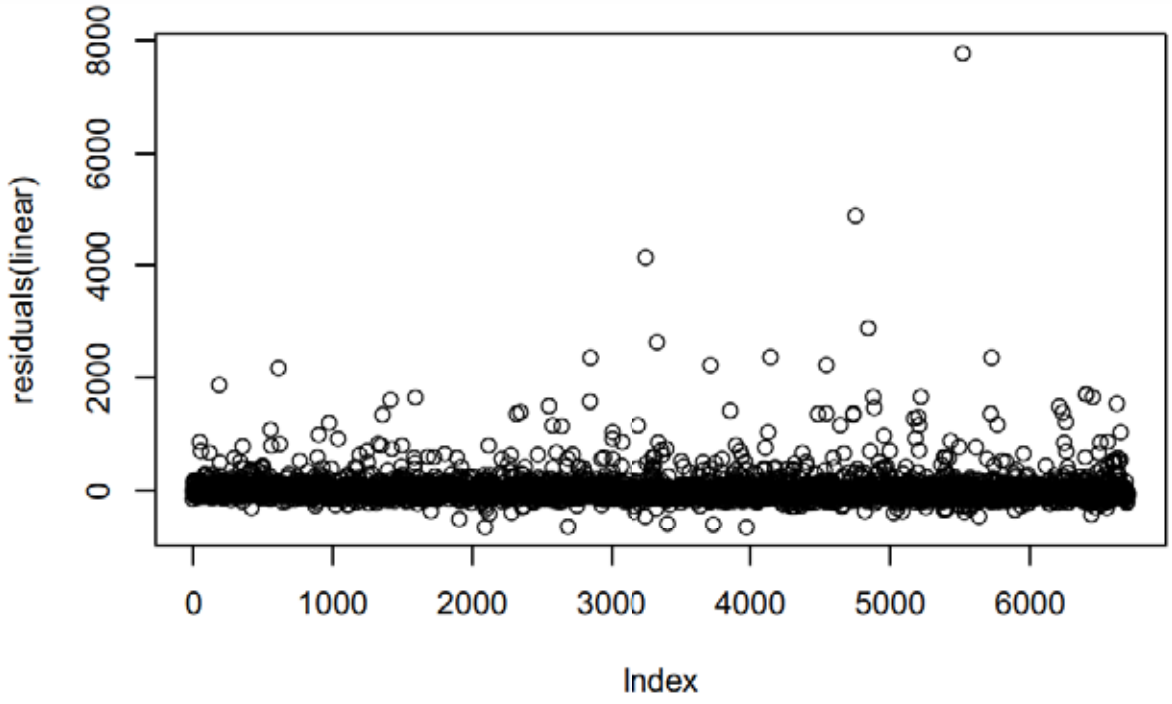
```
##
## Call:
## lm(formula = price ~ extra_people + maximum_nights + number_of_reviews +
##     review_scores_rating + guests_included, data = model_data)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -657.8  -75.1  -33.7   19.6 7774.5
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)           -1.088e+02  4.655e+01  -2.337 0.019472 *
## extra_people           7.583e-01  1.010e-01   7.509 6.73e-14 ***
## maximum_nights         2.942e-02  5.052e-03   5.823 6.05e-09 ***
## number_of_reviews     -2.560e-01  3.302e-02  -7.752 1.04e-14 ***
## review_scores_rating   1.748e+00  4.836e-01   3.614 0.000304 ***
## guests_included        4.441e+01  1.446e+00  30.706  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 220.9 on 6694 degrees of freedom
## Multiple R-squared:  0.1794, Adjusted R-squared:  0.1788
```

```
## F-statistic: 292.6 on 5 and 6694 DF,  p-value: < 2.2e-16
```

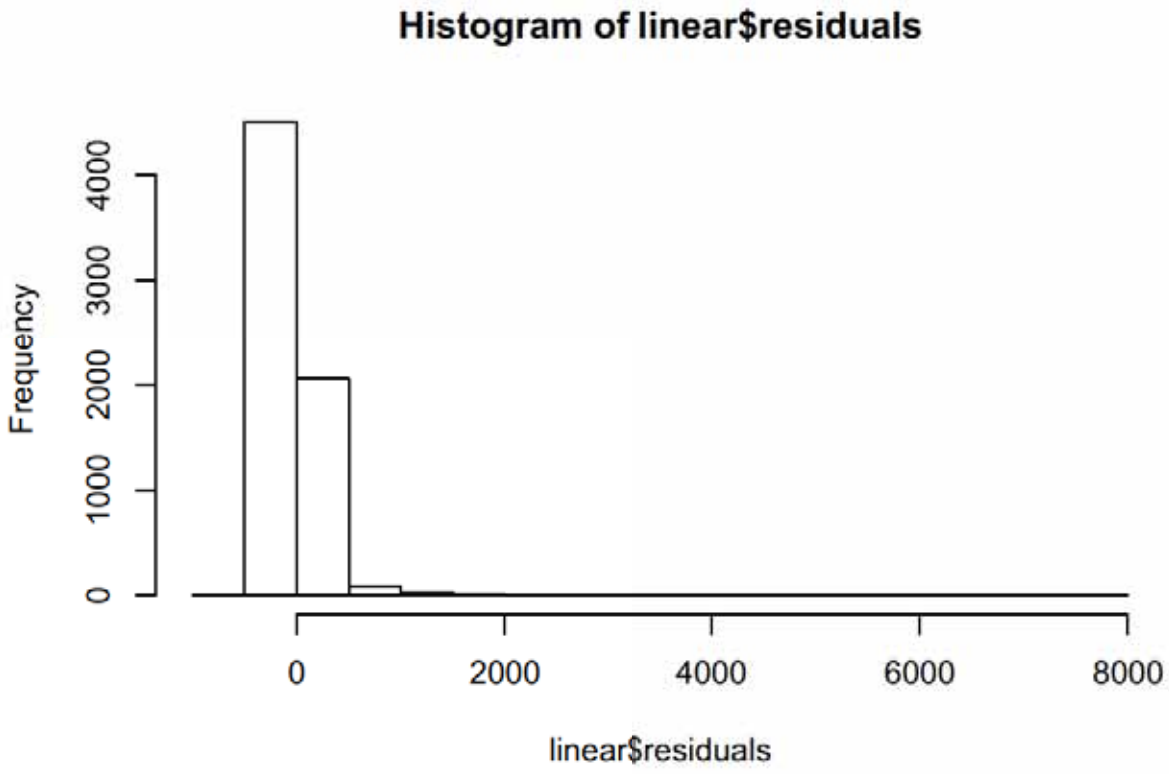
```
#calculate AIC
AIC(linear)
```

```
## [1] 91351.96
```

```
plot(residuals(linear))
```




```
#residual plot
hist(linear$residuals)
```




```
#calcualte VIF
vif(linear)
```

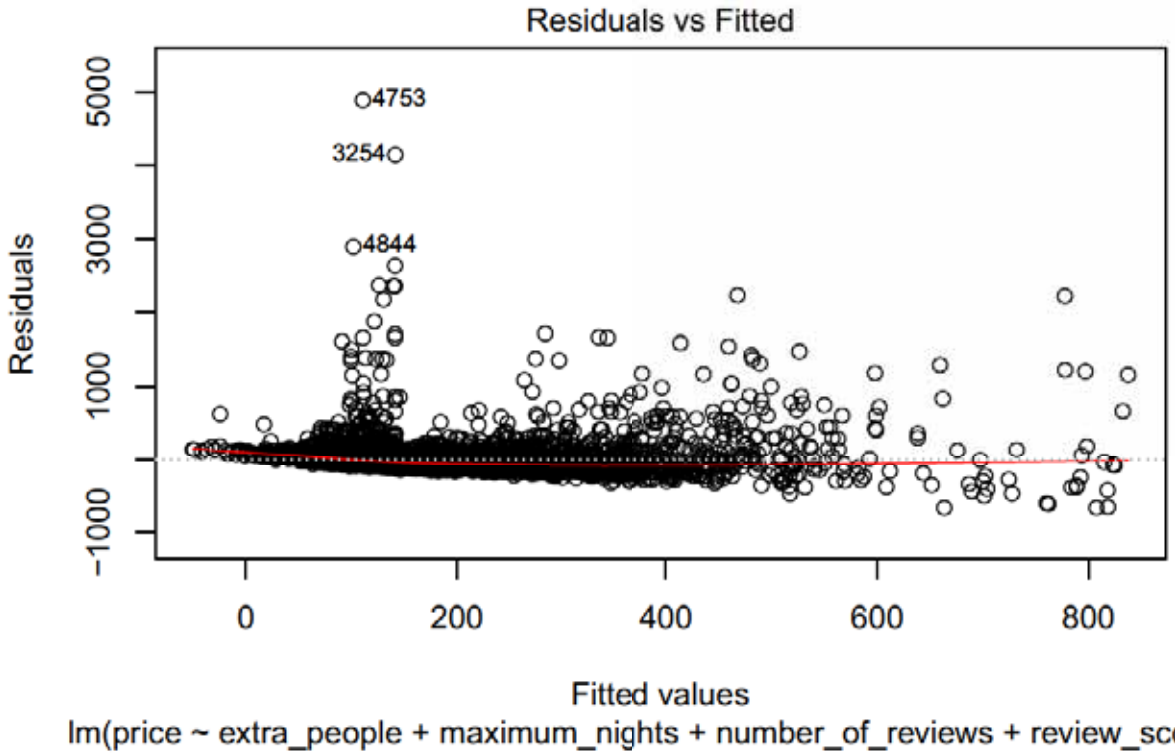
```
##         extra_people       maximum_nights    number_of_reviews
##             1.103855             1.041036             1.017540
## review_scores_rating      guests_included
##             1.028649             1.113982
```

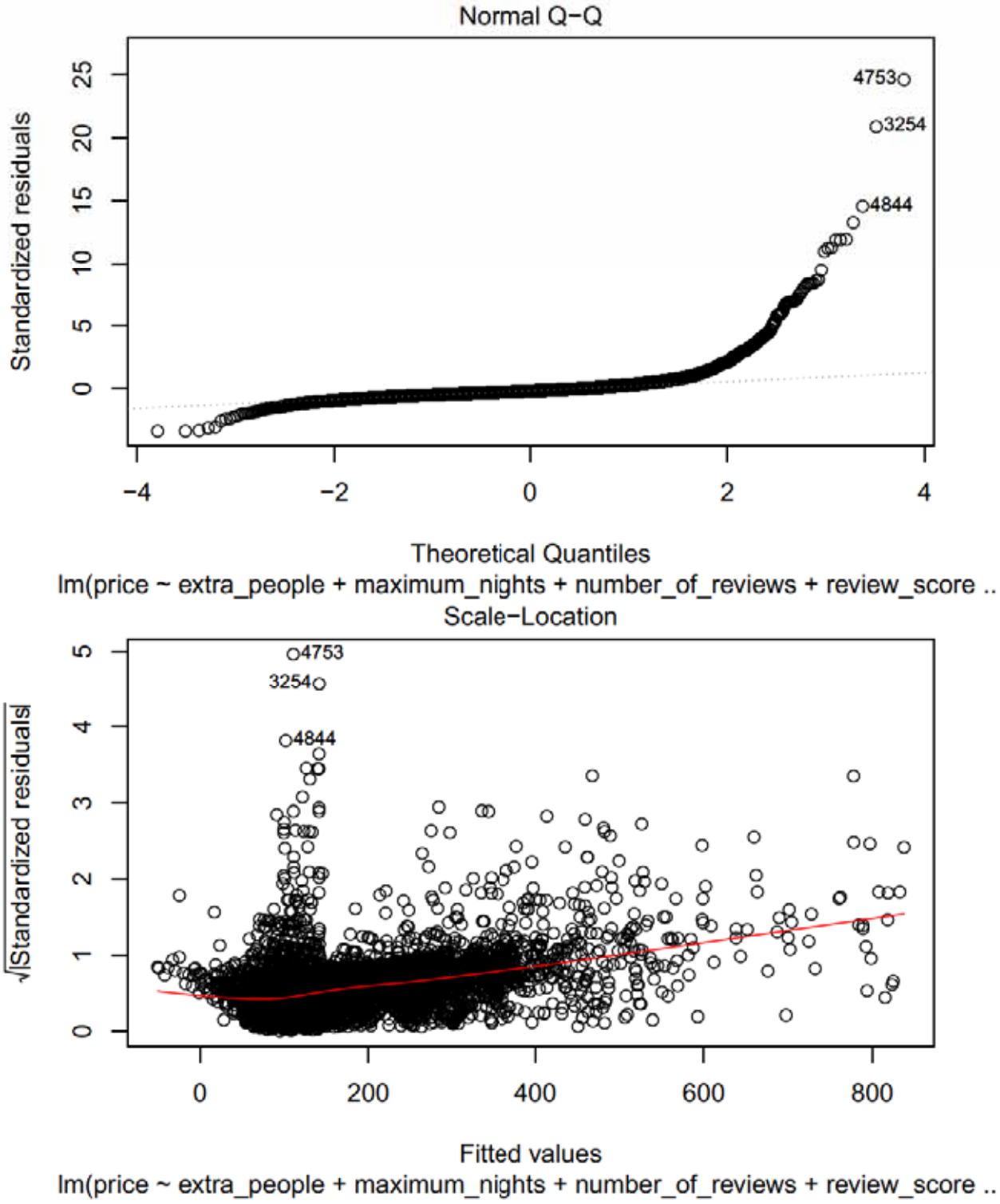
```
#drop some observations(might be outlier)
model_data<-model_data[-c(5519,6209),]
adj.linear<-lm(price~extra_people+maximum_nights+number_of_reviews+review_scores_rating+guests_included
summary(adj.linear)
```

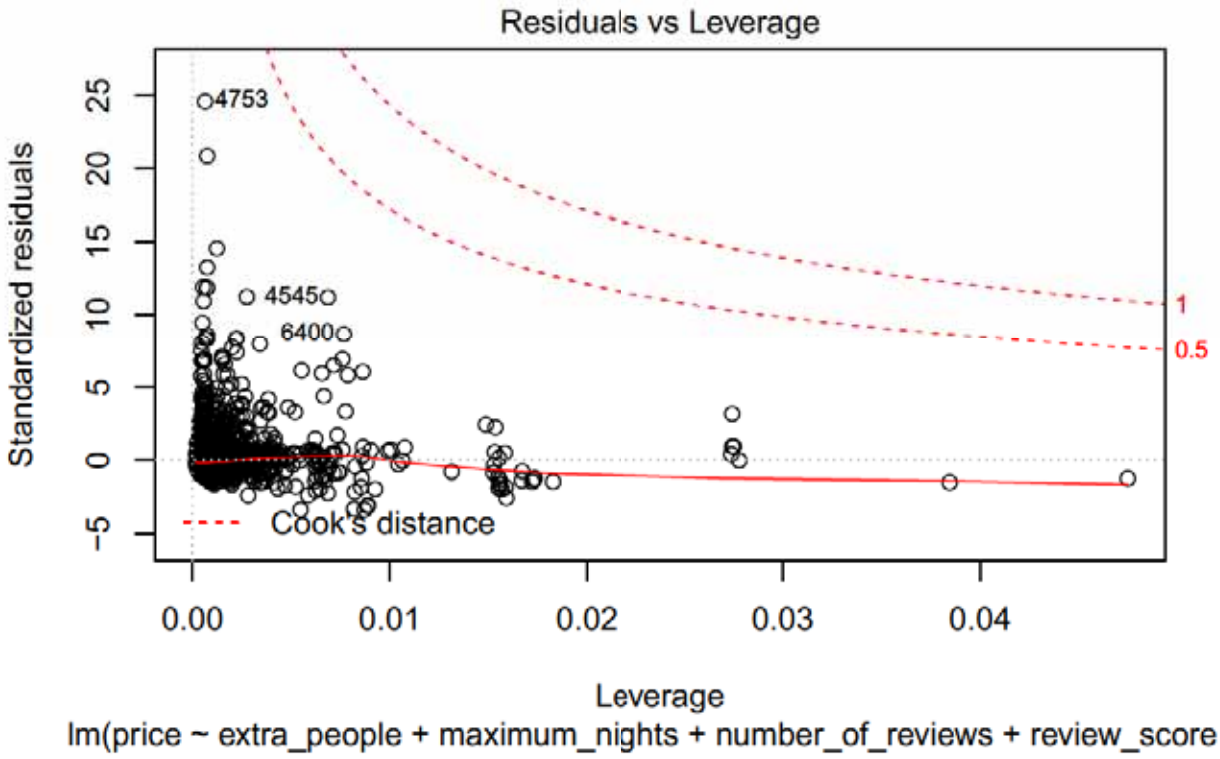
```
##
## Call:
## lm(formula = price ~ extra_people + maximum_nights + number_of_reviews +
##     review_scores_rating + guests_included, data = model_data)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -663.5  -73.3  -32.2   20.3 4889.1
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)         -1.412e+02  4.241e+01  -3.330 0.000872 ***
## extra_people         7.116e-01  9.079e-02   7.838 5.29e-15 ***
## maximum_nights       2.728e-02  4.542e-03   6.006 2.00e-09 ***
## number_of_reviews   -2.446e-01  2.969e-02  -8.240  < 2e-16 ***
```

```
## review_scores_rating  2.076e+00  4.406e-01   4.711 2.51e-06 ***
## guests_included       4.490e+01  1.300e+00  34.528  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 198.6 on 6692 degrees of freedom
## Multiple R-squared:  0.2124, Adjusted R-squared:  0.2118
## F-statistic:   361 on 5 and 6692 DF,  p-value: < 2.2e-16
```

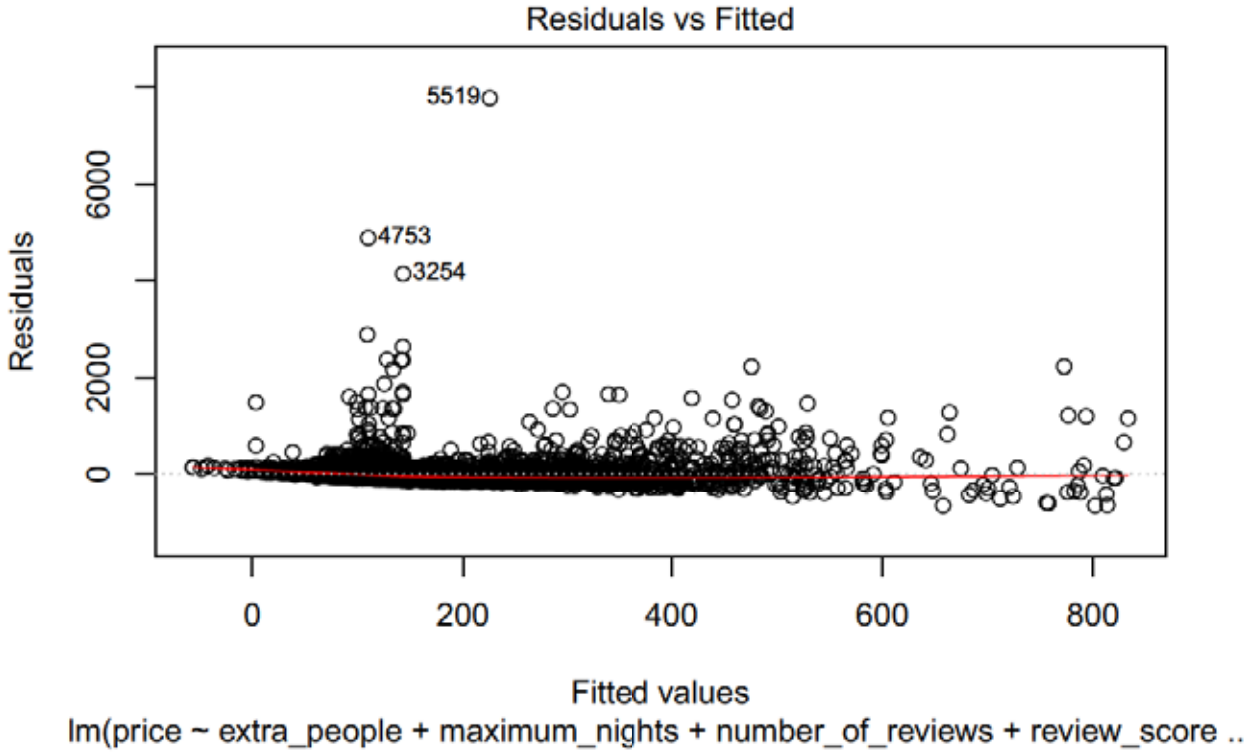
```
#model check( adjusted model)
plot(adj.linear)
```

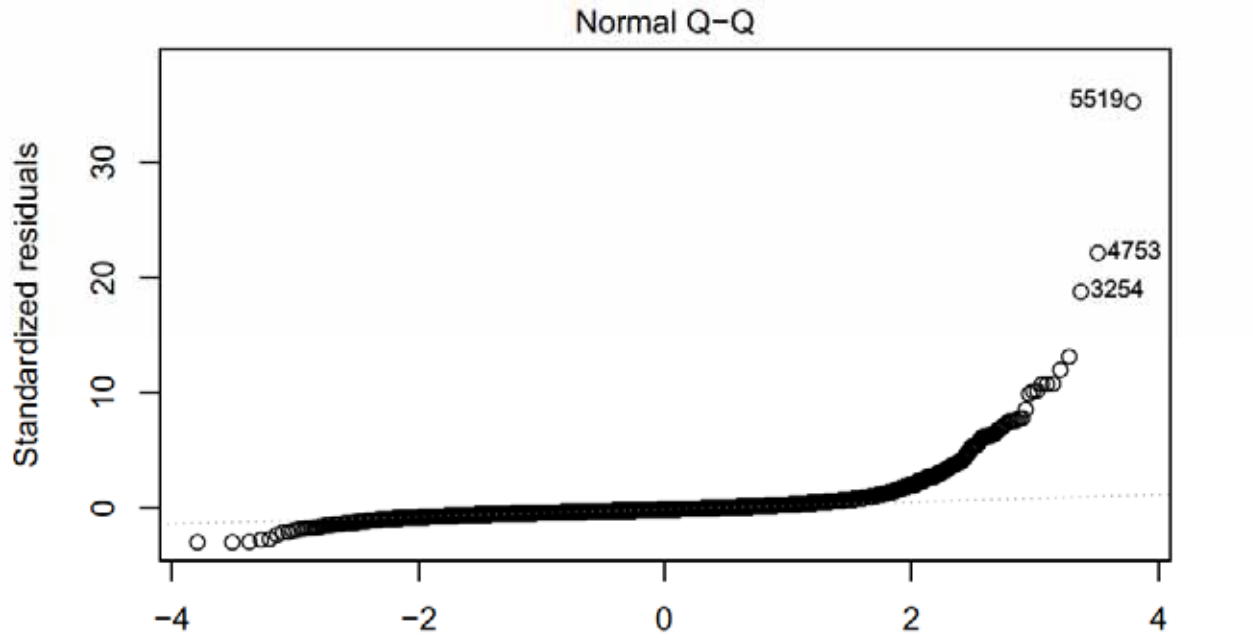


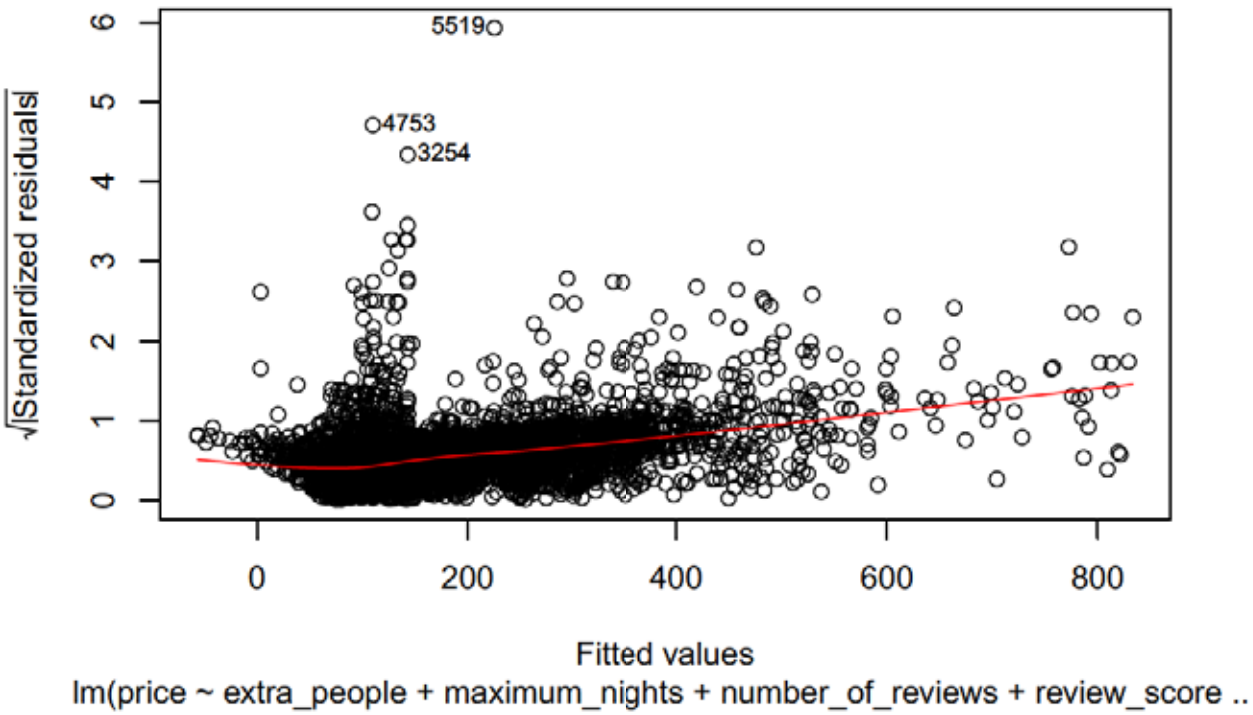

Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(price ~ extra_people + maximum_nights + number_of_reviews + review_score ..
4753
3254
4844
Scale−Location
√|Standardized residuals|
Fitted values
lm(price ~ extra_people + maximum_nights + number_of_reviews + review_score ..

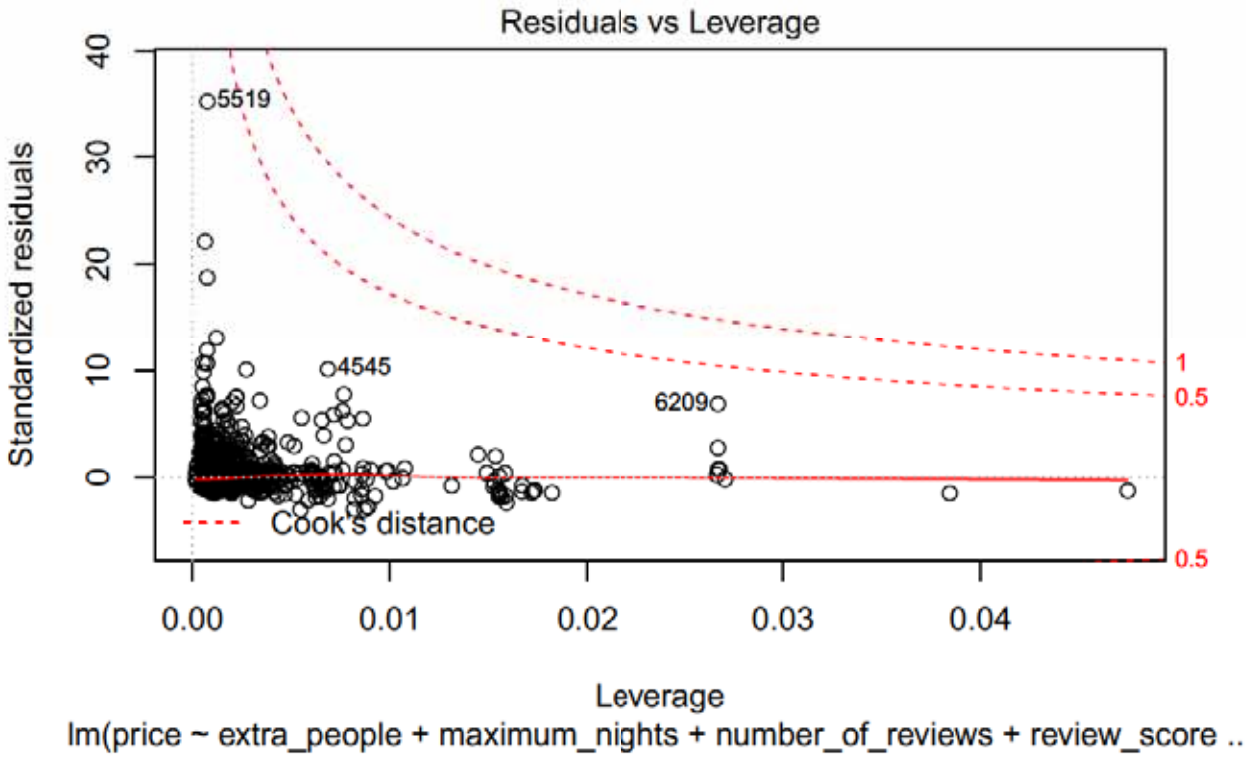



```
#model check(old model)
plot(linear)
```




Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(price ~ extra_people + maximum_nights + number_of_reviews + review_score ..
5519
4753
3254


Scale−Location
√|Standardized residuals|
Fitted values
lm(price ~ extra_people + maximum_nights + number_of_reviews + review_score ..
5519
4753
3254



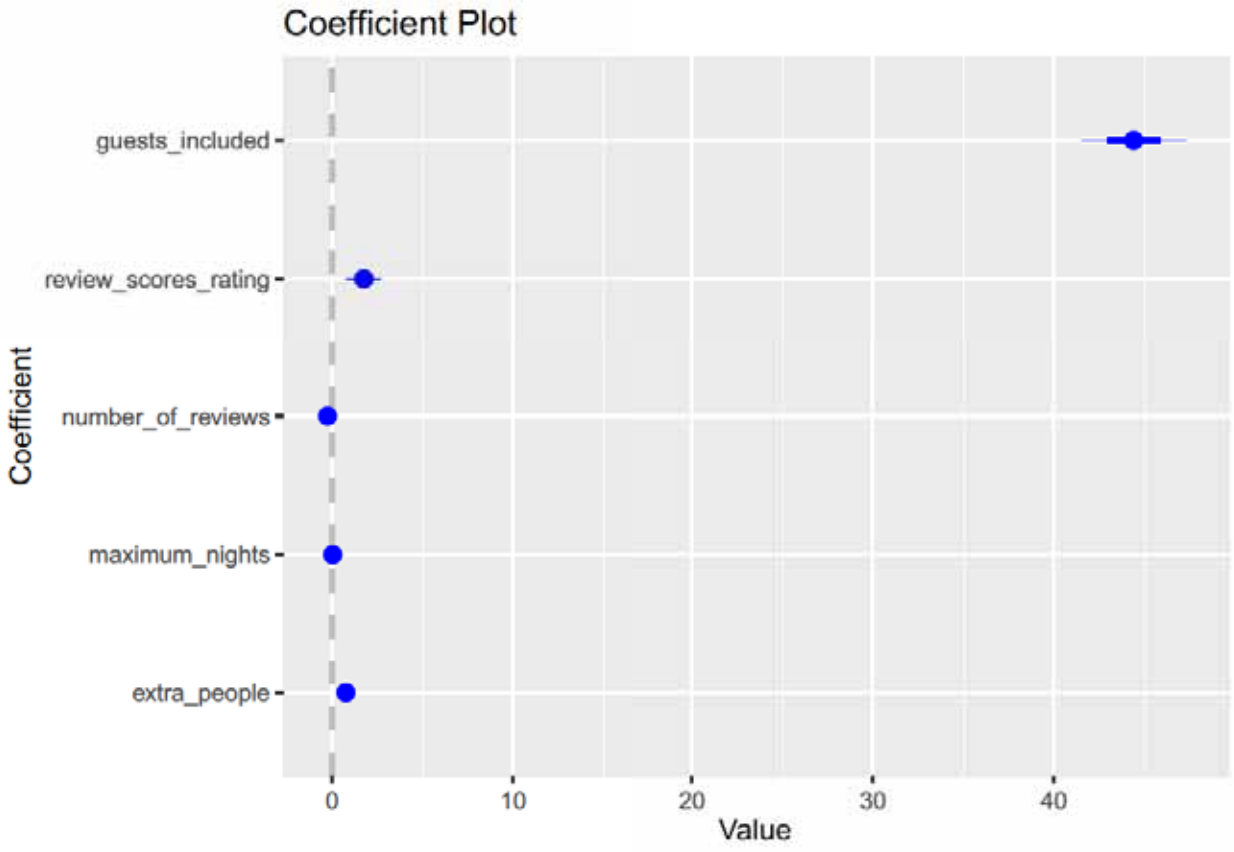

```
# F-statistic
summary(linear)$fstatistic
```

```
##     value     numdf     dendf
##  292.6285    5.0000 6694.0000
```

```
# confidence interval
confint(linear)
```

```
##                                2.5 %       97.5 %
## (Intercept)          -200.02716891 -17.53042155
## extra_people            0.56030208   0.95619933
## maximum_nights          0.01951284   0.03931848
## number_of_reviews      -0.32068267  -0.19122714
## review_scores_rating    0.79961995   2.69562707
## guests_included        41.57347355  47.24373803
```

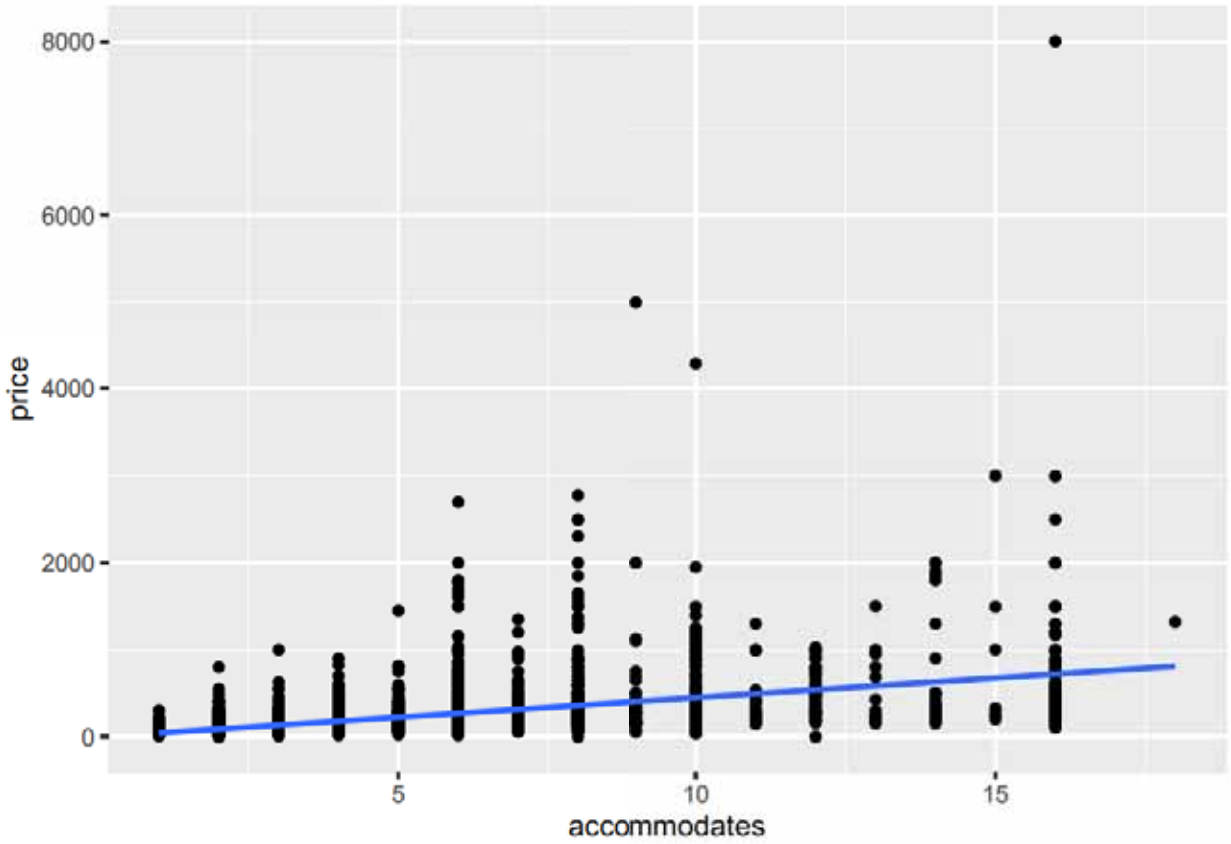
```
# visualize the confidence intervals
library(coefplot)
coefplot(linear, intercept = FALSE)
```




```
dwt(linear)
```

```
##  lag Autocorrelation D-W Statistic p-value
##    1      0.01684466      1.966227   0.126
##  Alternative hypothesis: rho != 0
```

```
#make a scatter plot
ggplot(list)+aes(x=accommodates,y=price)+
  geom_point()+geom_smooth(method="lm",se=FALSE)
```




```
model2<-lm(price~accommodates+bathrooms+bedrooms+beds+cleaning_fee+security_deposit,data=list)
summary(model2)
```

```
##
## Call:
## lm(formula = price ~ accommodates + bathrooms + bedrooms + beds +
##     cleaning_fee + security_deposit, data = list)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -969.4  -44.7    1.8   38.5 6599.3
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      -69.693792   3.966570 -17.570  < 2e-16 ***
## accommodates      12.092988   1.445667   8.365  < 2e-16 ***
## bathrooms         40.729041   2.667429  15.269  < 2e-16 ***
## bedrooms          17.648386   3.298630   5.350 9.08e-08 ***
## beds             -10.105630   2.154877  -4.690 2.79e-06 ***
## cleaning_fee       1.224411   0.039402  31.075  < 2e-16 ***
## security_deposit   0.075177   0.004278  17.573  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 165.9 on 6693 degrees of freedom
## Multiple R-squared:  0.5374, Adjusted R-squared:  0.537
## F-statistic:  1296 on 6 and 6693 DF,  p-value: < 2.2e-16
```

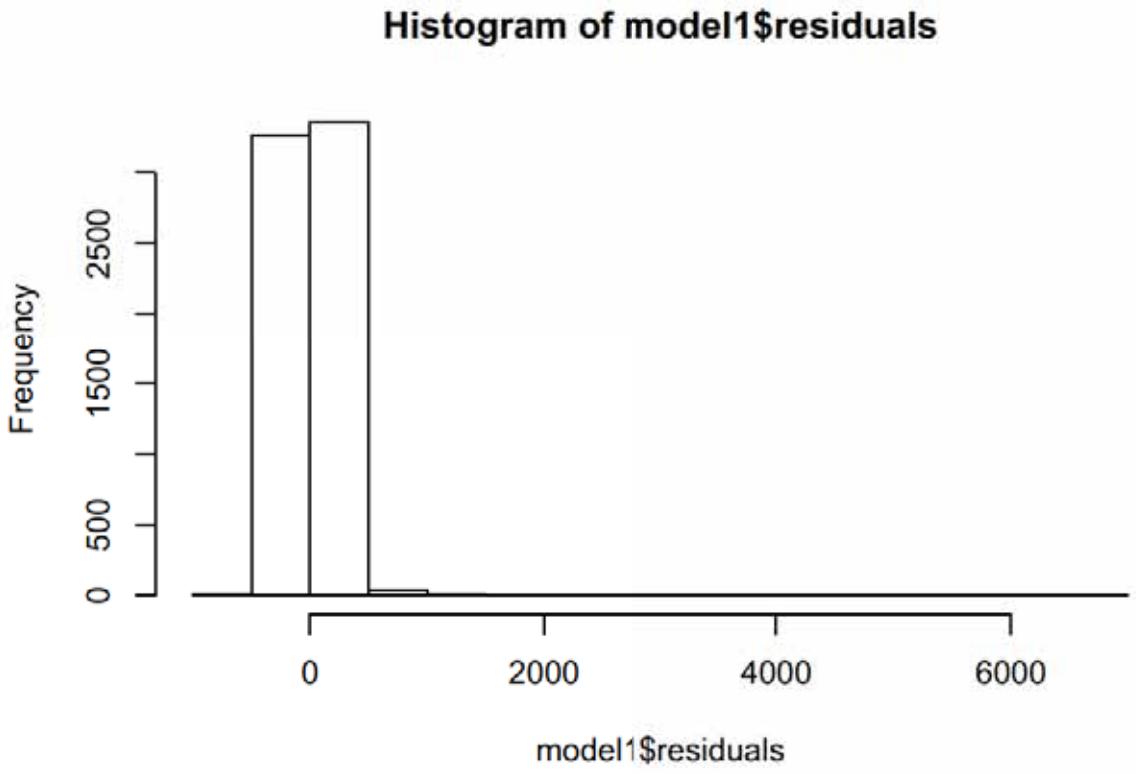
```r
AIC(model2)
```

```
## [1] 87513.49
```

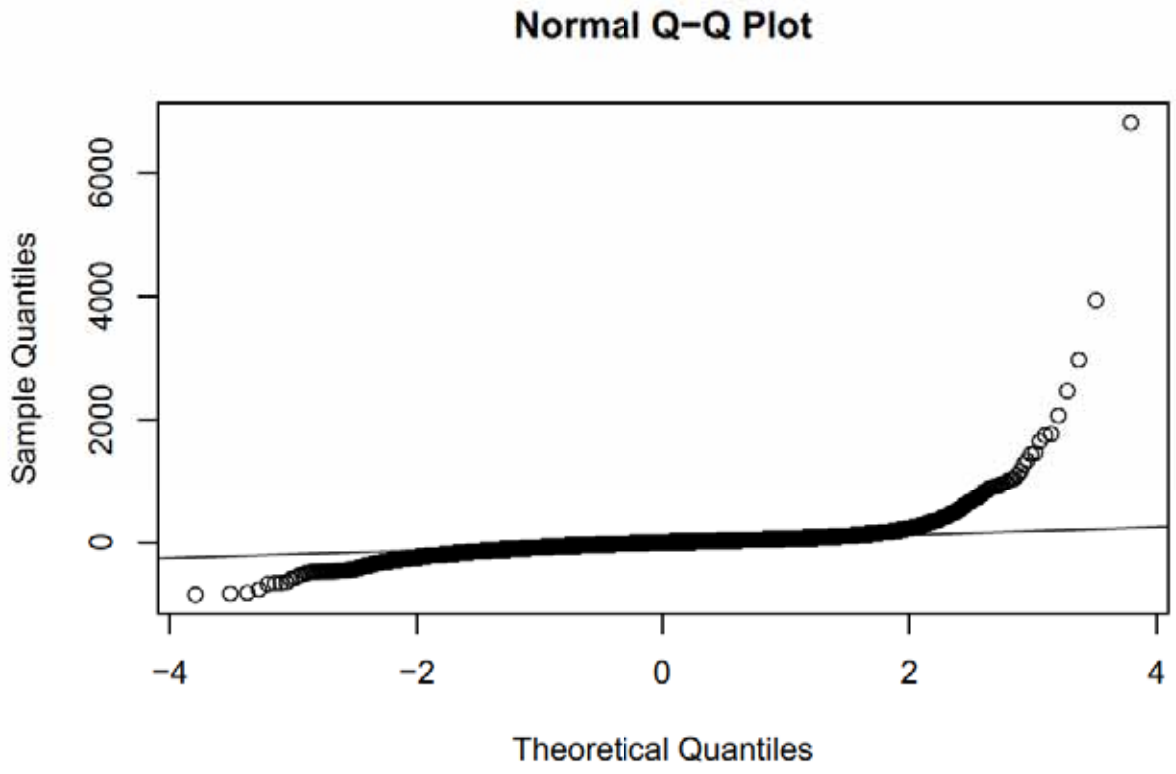
```r
model1<-lm(price~accommodates+bathrooms+bedrooms+beds+cleaning_fee+guests_included,data=list)
summary(model1)
```

```
##
## Call:
## lm(formula = price ~ accommodates + bathrooms + bedrooms + beds +
##     cleaning_fee + guests_included, data = list)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -846.8  -46.2    1.5   38.0 6811.8
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -70.59287    4.07950 -17.304  < 2e-16 ***
## accommodates     11.22253    1.55683   7.209 6.27e-13 ***
## bathrooms        45.26468    2.71468  16.674  < 2e-16 ***
## bedrooms         22.13985    3.38735   6.536 6.78e-11 ***
## beds            -11.27561    2.20244  -5.120 3.15e-07 ***
## cleaning_fee      1.55083    0.03589  43.206  < 2e-16 ***
## guests_included  -3.91695    1.47644  -2.653    0.008 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 169.6 on 6693 degrees of freedom
## Multiple R-squared:  0.5166, Adjusted R-squared:  0.5161
## F-statistic:  1192 on 6 and 6693 DF,  p-value: < 2.2e-16
```

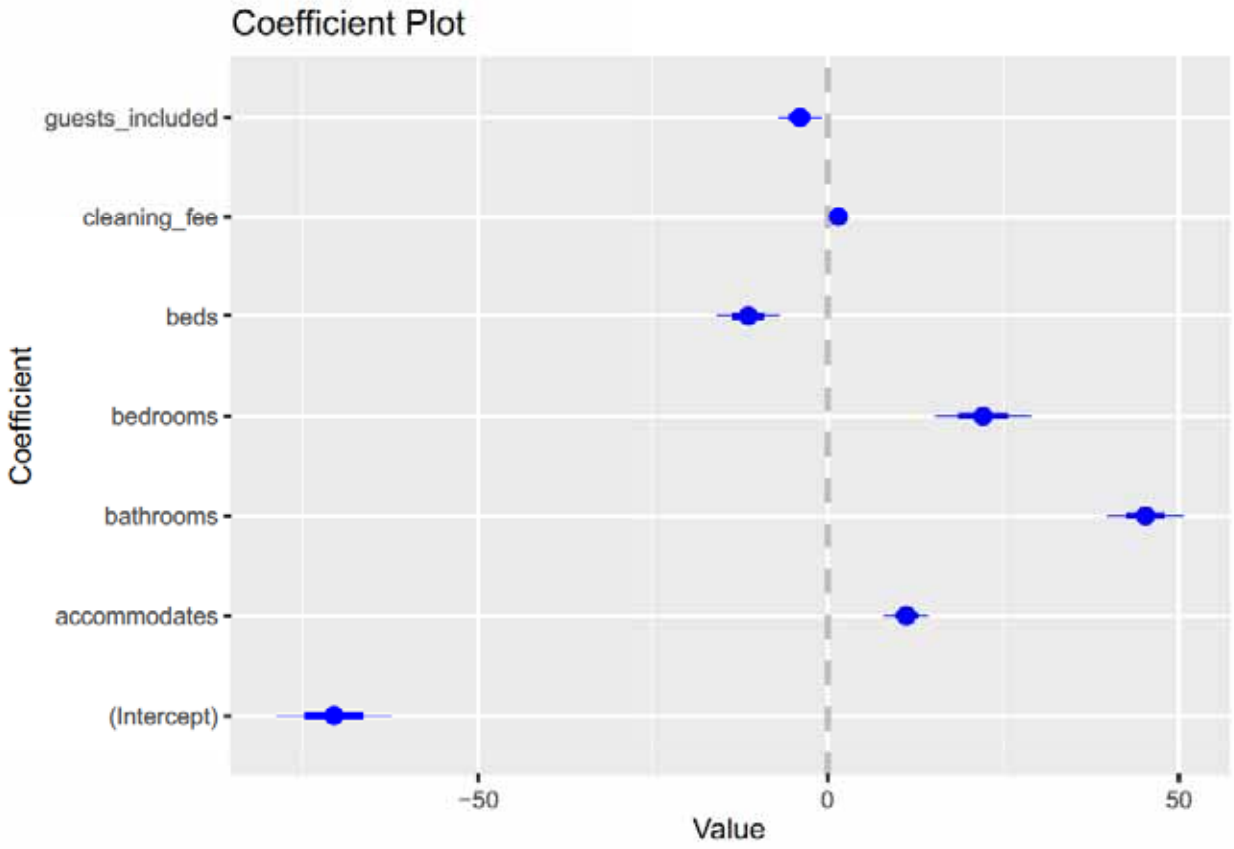
```r
hist(model1$residuals)
```



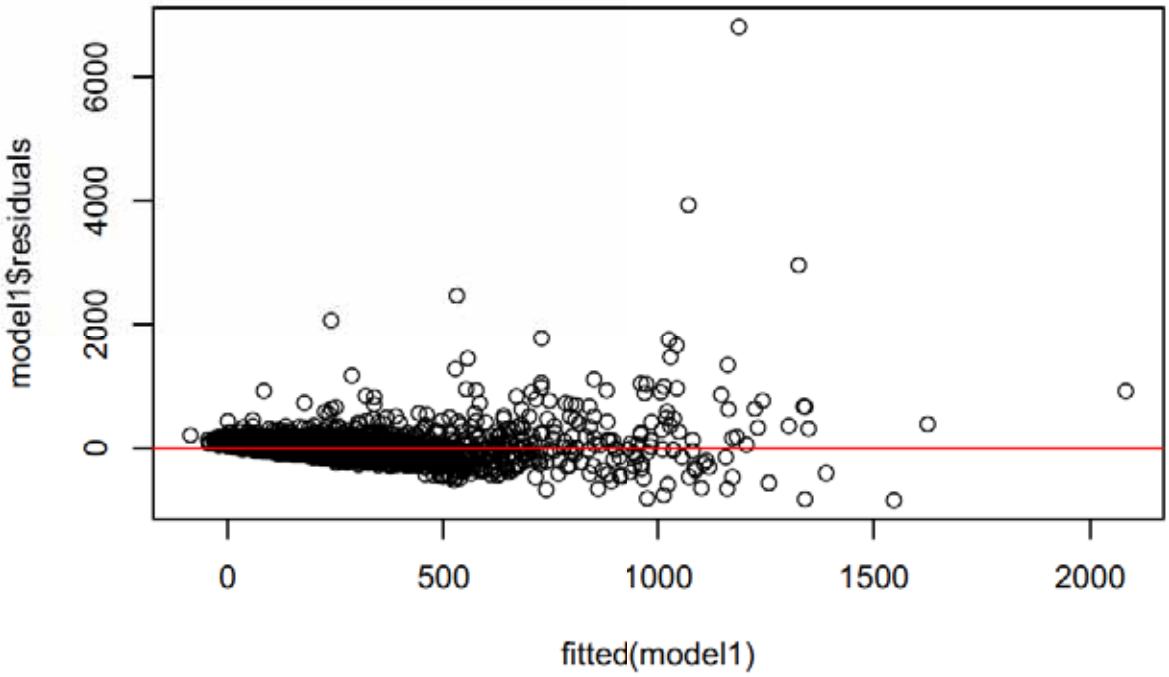

```r
qqnorm(model1$residuals)
qqline(model1$residuals)
```



```
library(coefplot)
coefplot(model1)
```




```
plot(fitted(model1),model1$residuals)
abline(0,0,col="red")
```




```
library(tidyverse)
library(gridExtra)
library(car)
#Checking the assumption of independence
dwt(model1)
```

```
##  lag Autocorrelation D-W Statistic p-value
##    1      0.03198436      1.936024   0.022
##  Alternative hypothesis: rho != 0
```

```
# VIF
vif(model1)
```

```
##    accommodates       bathrooms        bedrooms            beds    cleaning_fee
##        4.445607        1.609950        3.304916        3.362097        1.912772
## guests_included
##        1.970408
```

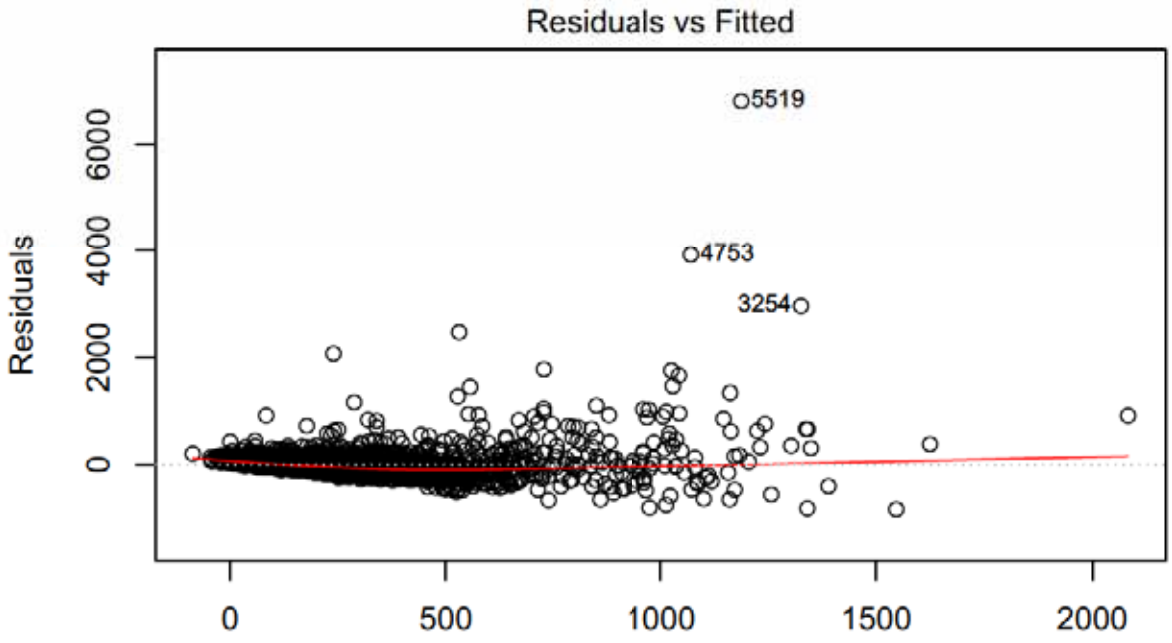
```
# tolerance
1/vif(model1)
```

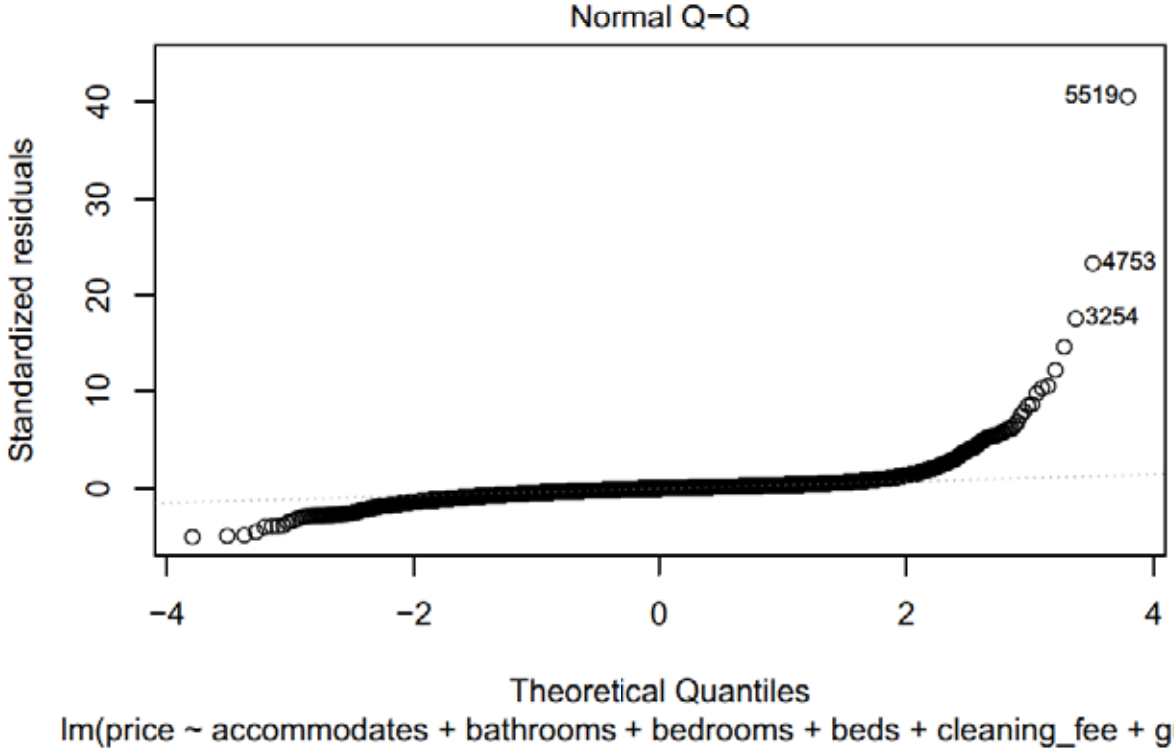
```
##    accommodates       bathrooms        bedrooms            beds    cleaning_fee
##       0.2249412       0.6211375       0.3025795       0.2974334       0.5228014
## guests_included
##       0.5075090
```

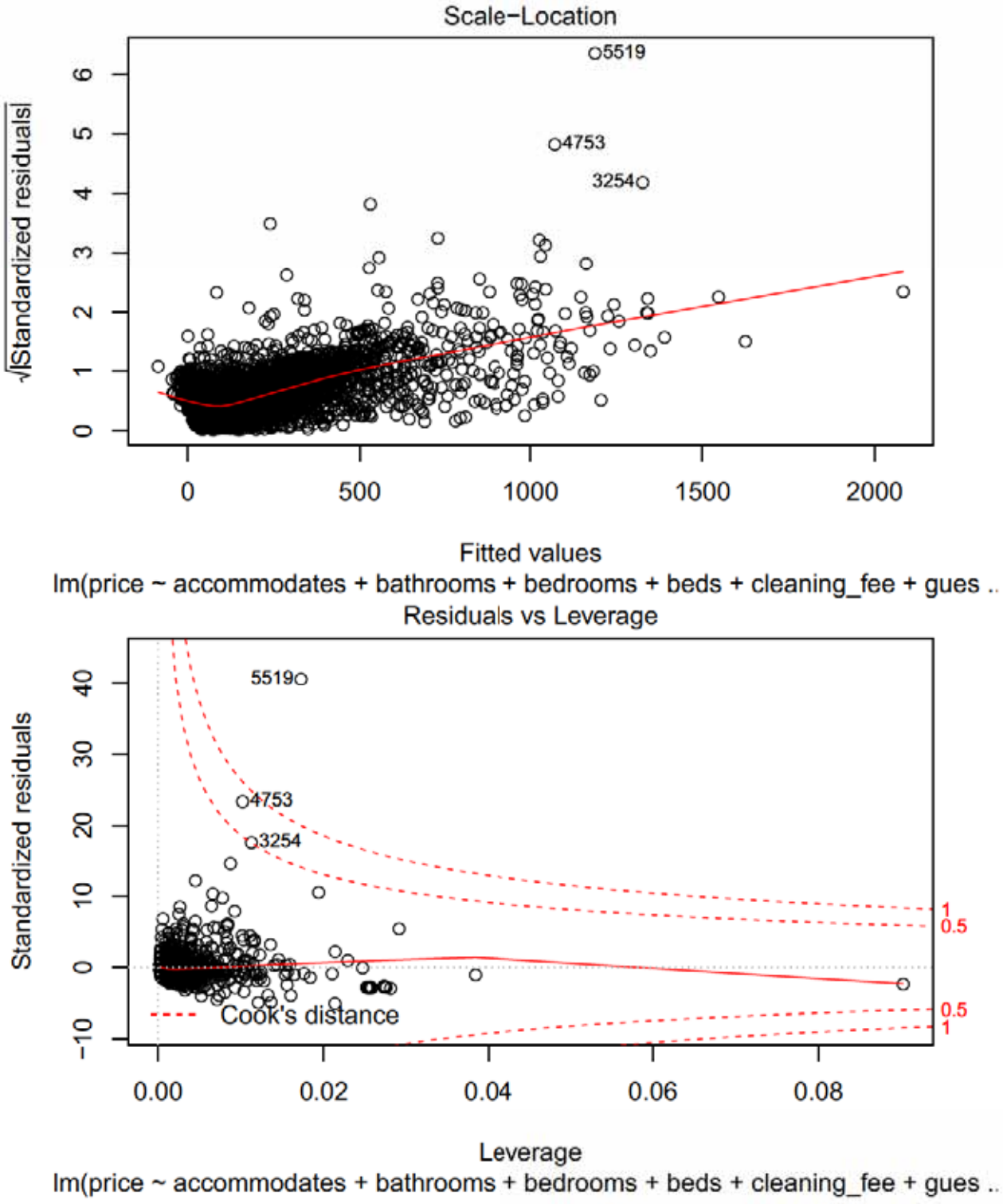
```
# mean VIF
mean(vif(model1))
```

```
## [1] 2.767625
```

```
plot(model1)
```


Residuals vs Fitted
Residuals
Fitted values
lm(price ~ accommodates + bathrooms + bedrooms + beds + cleaning_fee + gues ..
5519
4753
3254
0
2000
4000
6000
0
500
1000
1500
2000


Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(price ~ accommodates + bathrooms + bedrooms + beds + cleaning_fee + gues ..
5519
4753
3254
0
10
20
30
40
−4
−2
0
2
4



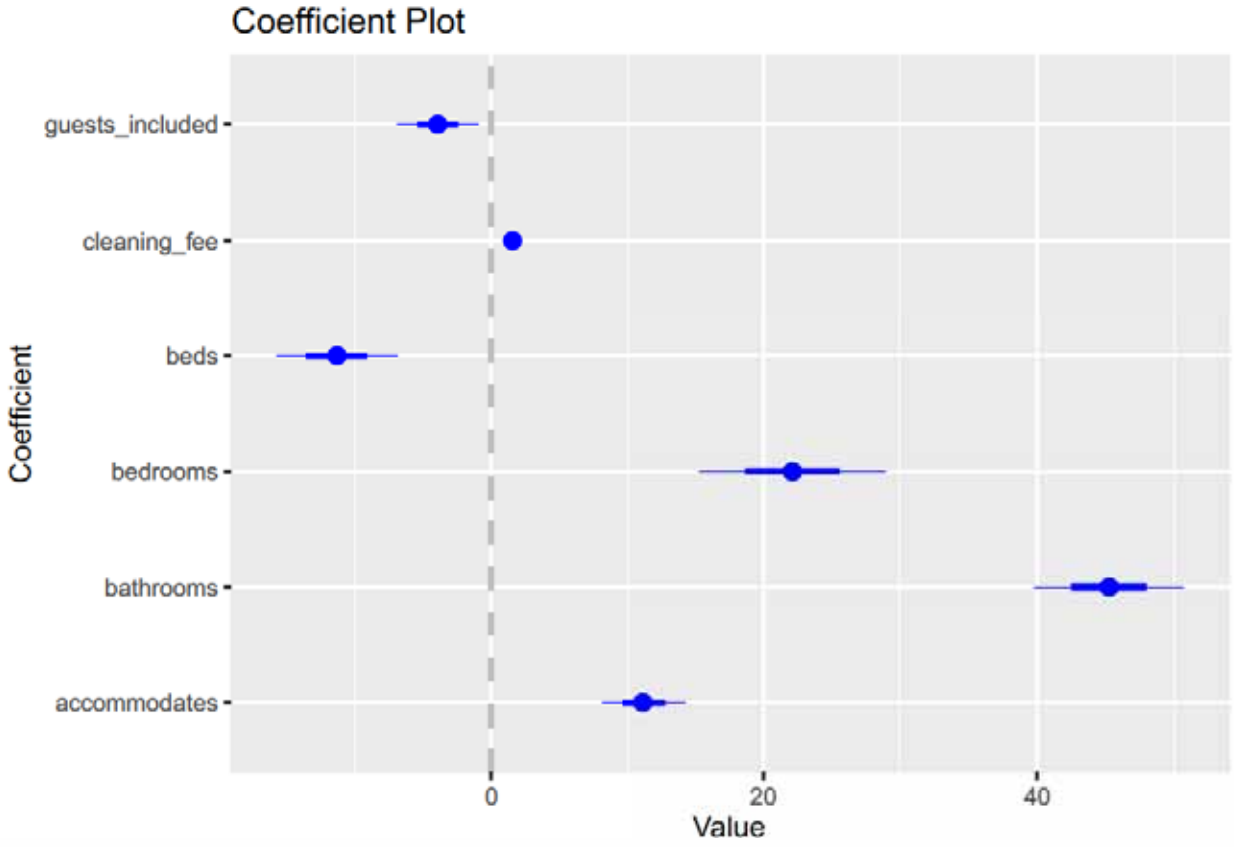

```
# F-statistic
summary(model1)$fstatistic
```

```
##    value    numdf    dendf
## 1191.932    6.000 6693.000
```

```r
# confidence interval
confint(model1)
```

```
##                       2.5 %     97.5 %
## (Intercept)     -78.589996 -62.595753
## accommodates      8.170648  14.274418
## bathrooms        39.943043  50.586310
## bedrooms         15.499566  28.780141
## beds            -15.593084  -6.958128
## cleaning_fee      1.480469   1.621196
## guests_included  -6.811233  -1.022663
```

```r
# visualize the confidence intervals
library(coefplot)
coefplot(model1, intercept = FALSE)
```




```r
library(MASS)
#Shapiro-Wilk Normality Test
## Distribution of studentized residuals
student_residuals <- studres(model1)
shapiro.test(sample(student_residuals, size = 5000))
```

```
##
##  Shapiro-Wilk normality test
##
```

```
## data:  sample(student_residuals, size = 5000)
## W = 0.4101, p-value < 2.2e-16
```

```
#p-value is less than 0.05, reject the null hypothesis that residuals are normally distributed.
```

```
#change price into binary outcome
library(magrittr)
library(tidyverse)
log_list<-list
log_list$price<-as.factor(ifelse(log_list$price>400,1,0))
```

```
#get confusion matrix
(table(log_list$price))
```

```
##
##    0    1
## 6292  408
```

```
6759/(6759+421)
```

```
## [1] 0.9413649
```

```
library(caTools)
#Splitting Training & Testing Data
# Randomly split data
set.seed(6888)
split = sample.split(log_list$price, SplitRatio = 0.94)
# Create training and testing sets
priceTrain = subset(log_list, split == TRUE)
priceTrain<-data.frame(priceTrain)
priceTest = subset(log_list, split == FALSE)
priceTest<-data.frame(priceTest)
```

```
nrow(priceTrain)
```

```
## [1] 6298
```

```
nrow(priceTest)
```

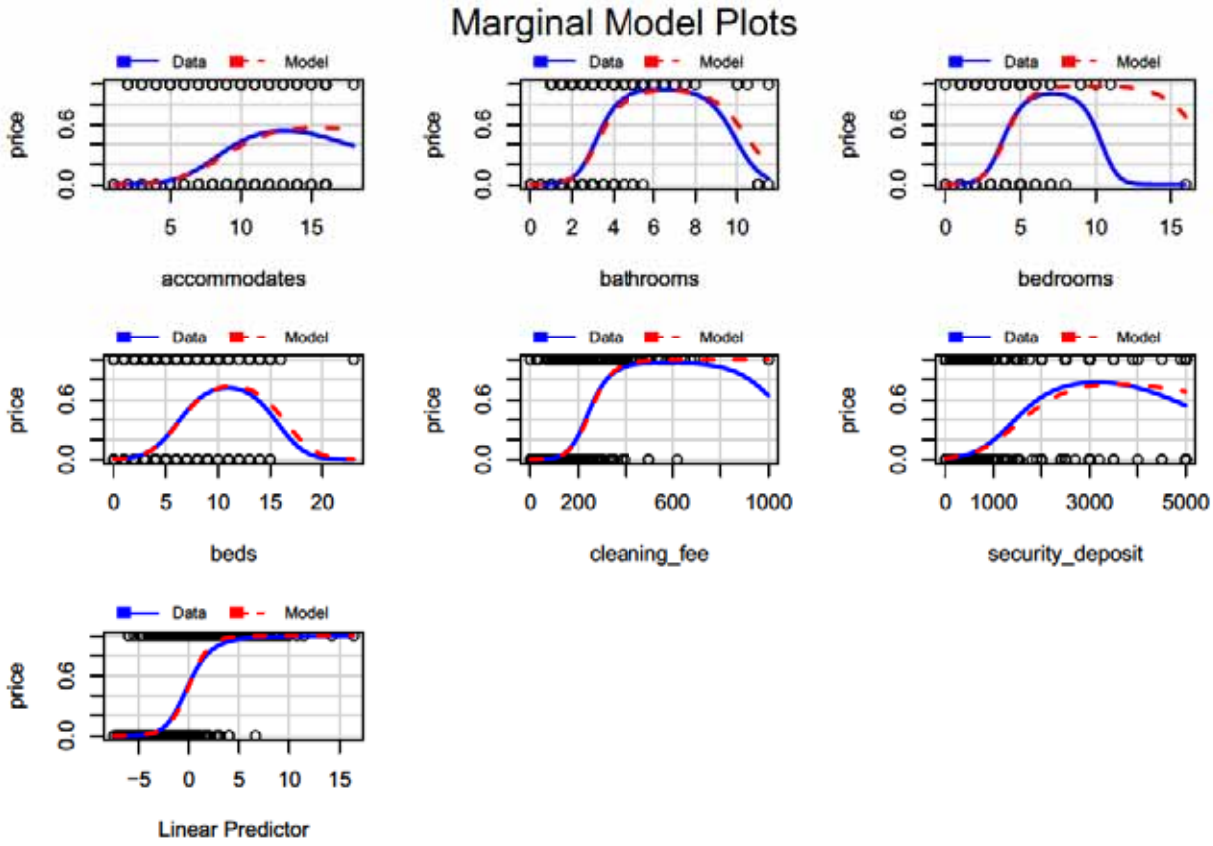
```
## [1] 402
```

```
#logistic regression
logistic.model1=glm(price~accommodates+bathrooms+bedrooms+beds+cleaning_fee+security_deposit,data=price
summary(logistic.model1)
```

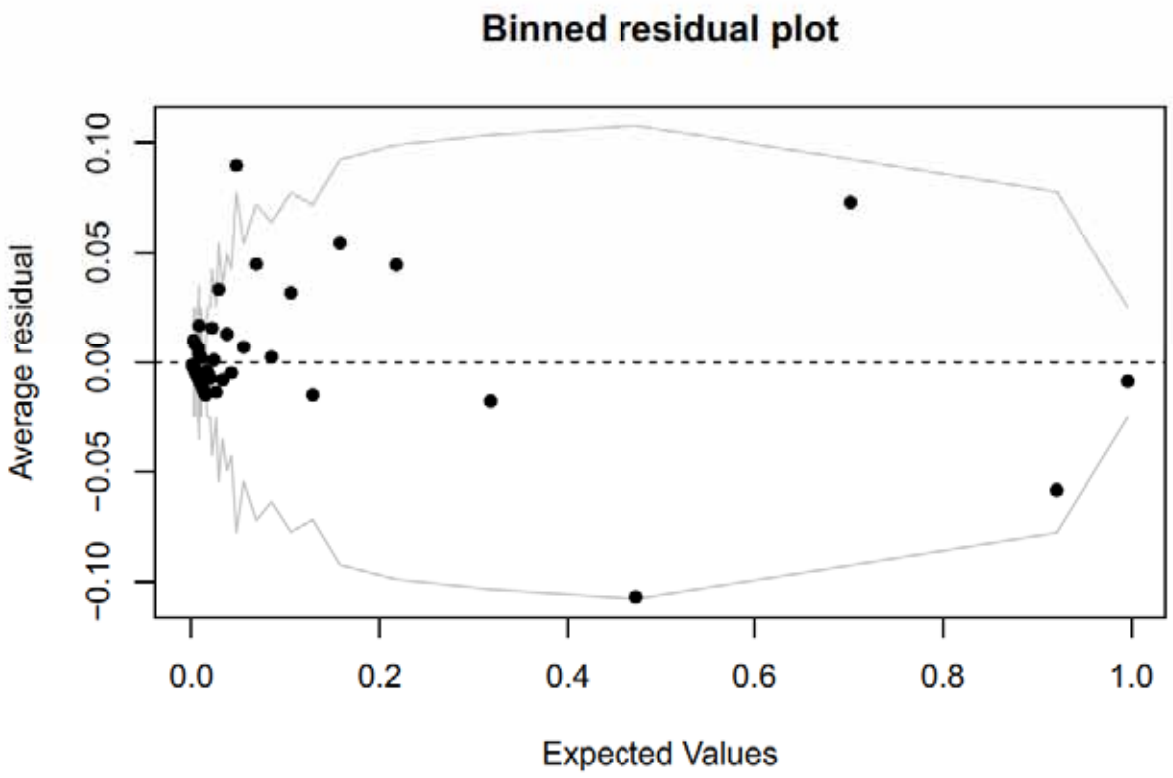
```
##
## Call:
## glm(formula = price ~ accommodates + bathrooms + bedrooms + beds +
##     cleaning_fee + security_deposit, family = binomial, data = priceTrain)
```

```
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.6431  -0.1679  -0.0948  -0.0689   3.4766
##
## Coefficients:
##                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)      -7.6763591  0.2520458 -30.456  < 2e-16 ***
## accommodates      0.1201032  0.0376849   3.187  0.00144 **
## bathrooms         0.4638019  0.0528759   8.772  < 2e-16 ***
## bedrooms          0.5032084  0.0978814   5.141 2.73e-07 ***
## beds             -0.0789813  0.0577402  -1.368  0.17135
## cleaning_fee      0.0144916  0.0011873  12.205  < 2e-16 ***
## security_deposit  0.0004381  0.0001036   4.228 2.35e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2892.5  on 6297  degrees of freedom
## Residual deviance: 1220.5  on 6291  degrees of freedom
## AIC: 1234.5
##
## Number of Fisher Scoring iterations: 7
```

```r
#model check
library(car)
marginalModelPlots(logistic.model1)
```




```
#binned residual plot
library(arm)
binnedplot(fitted(logistic.model1),residuals(logistic.model1,type="response"))
```




```
#use model to get prediction
predictTrain = predict(logistic.model1, type="response")
summary(predictTrain)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## 0.000542 0.002736 0.005362 0.060972 0.021434 1.000000
```

```
tapply(predictTrain, priceTrain$price, mean)
```

```
##          0          1
## 0.02768651 0.57359890
```

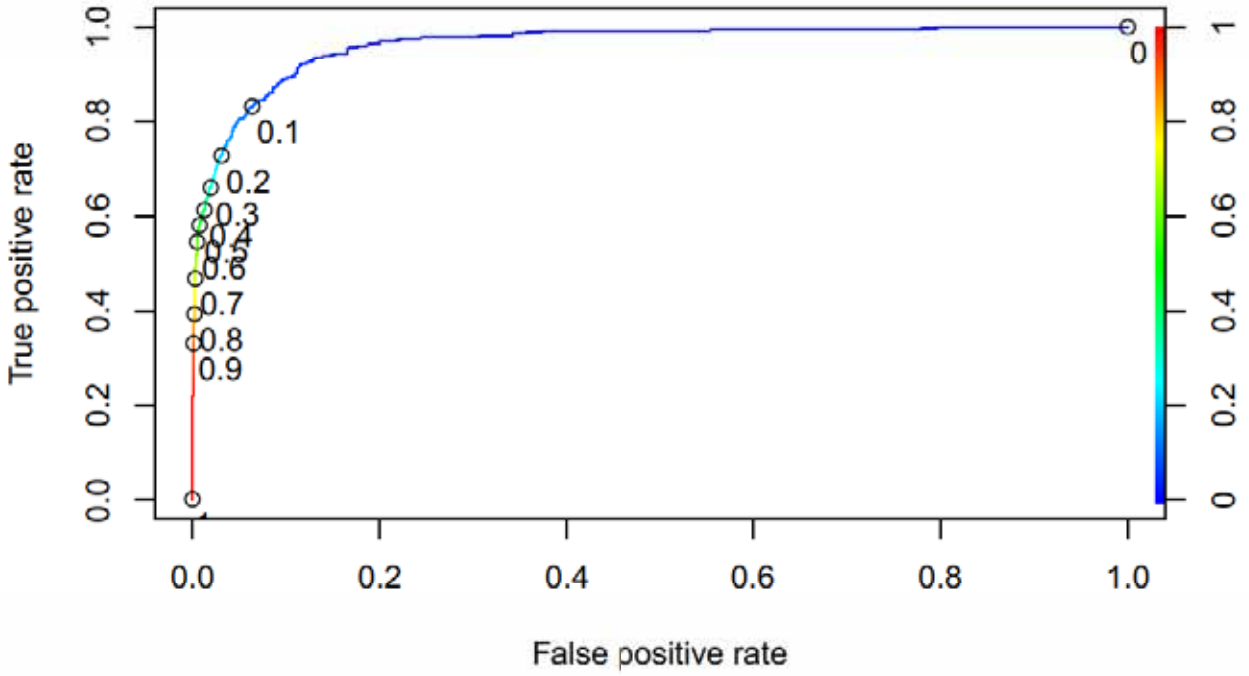
```
#Confusion matrix
table(priceTrain$price, predictTrain > 0.5)
```

```
##
##     FALSE TRUE
##   0  5869   45
##   1   161  223
```

```
#sensitivity
82/(339+82)
```

```
## [1] 0.1947743
```

```r
#Iload ROCR package
library(ROCR)
ROCpred = prediction(predictTrain, priceTrain$price)
# Performance function
ROCperf = performance(ROCpred, "tpr", "fpr")
# Add threshold labels
plot(ROCperf, colorize=TRUE, text.adj=c(-0.1,1.6),print.cutoffs.at=seq(0,1,by=0.1) )
```




```r
predictTest = predict(logistic.model1, type = "response", newdata = priceTest)
table(priceTest$price,predictTest >= 0.3)
```

```
##
##     FALSE TRUE
##   0   373    5
##   1     9   15
```

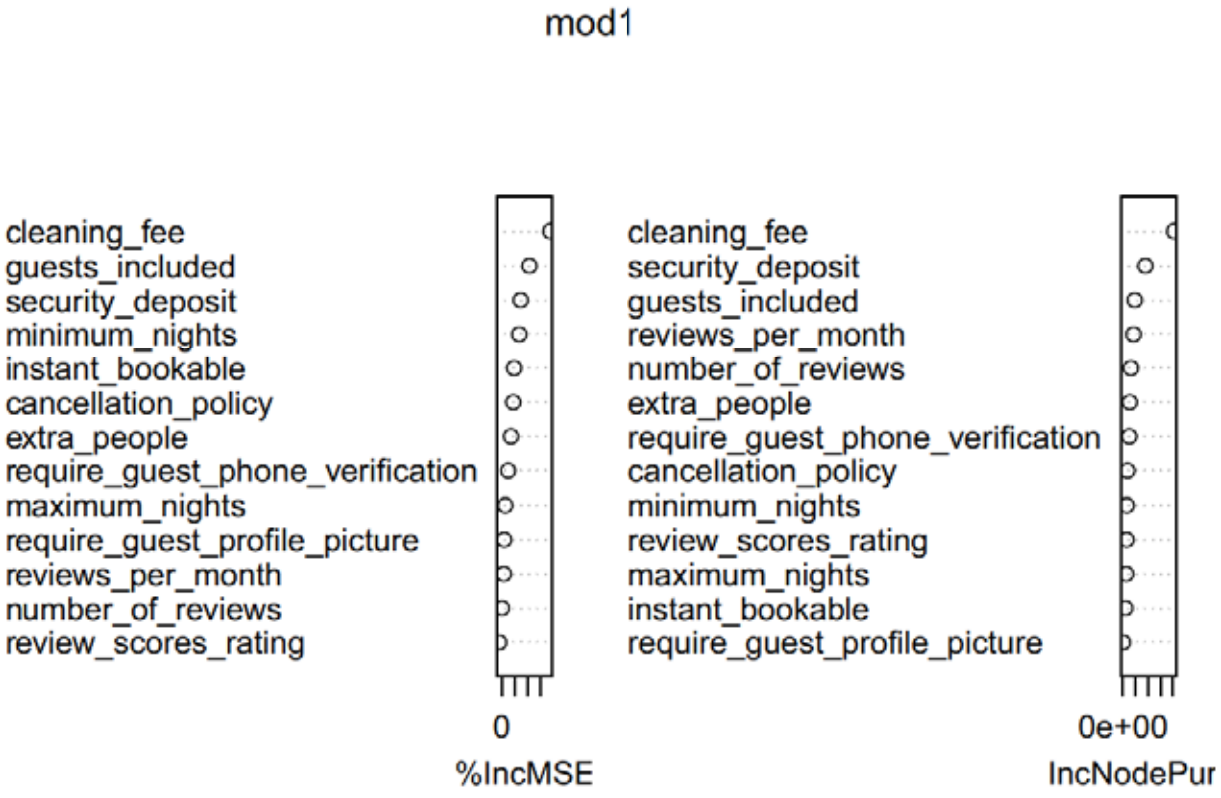
```r
# Accuracy
(400+9)/(409+22)
```

```
## [1] 0.9489559
```

```r
#multi-level regression
data<-read.csv("list.csv")
neigh<-data$neighbourhood
data<-data[,c(40:55)]
#data$neigh<-neigh
data$host_since<-NULL
data$calendar_updated<-NULL
data$cancellation_policy<-as.factor(data$cancellation_policy)
data$require_guest_phone_verification<-as.factor(data$require_guest_phone_verification)
```

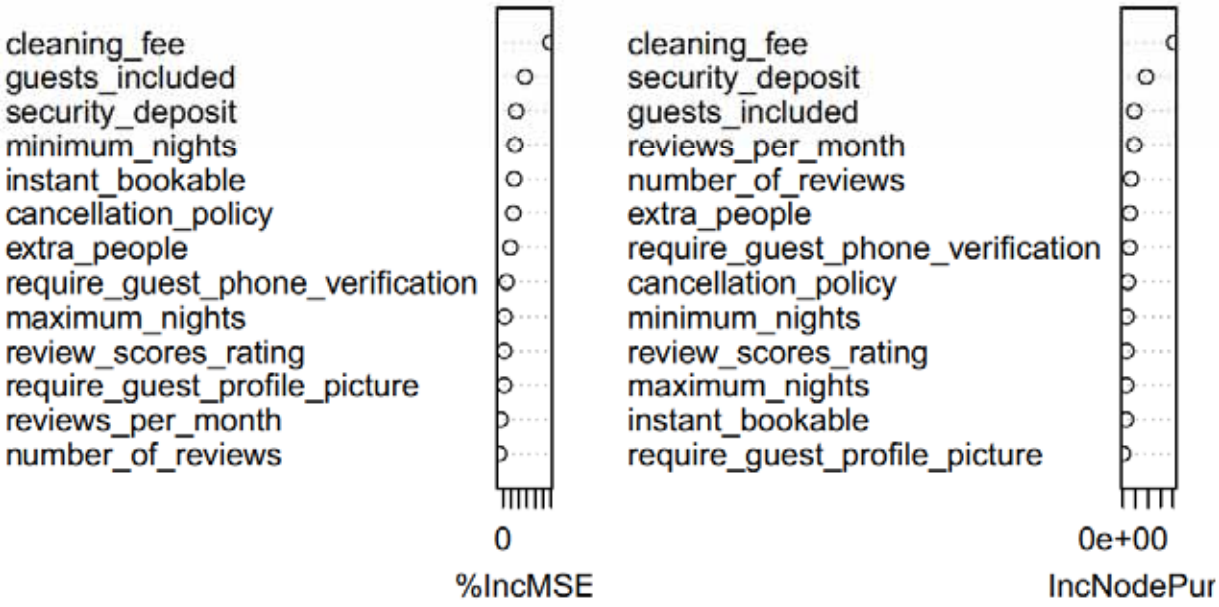
```
data$require_guest_profile_picture<-as.factor(data$require_guest_profile_picture)
index<-sample(7180,7180/2,replace = F)
variable<-data[index,]
modelset<-data[-index,]
# function for leave-one-out cv
looCv<- function(model){
mean(residuals(model)^2/(1-hatvalues(model))^2)
}
```

```
# random forest
mod1<-randomForest(price~.,data = variable,importance=T,ntree=500)
mod2<-randomForest(price~.,data = variable,importance=T,ntree=1000)

varImpPlot(mod1)
```



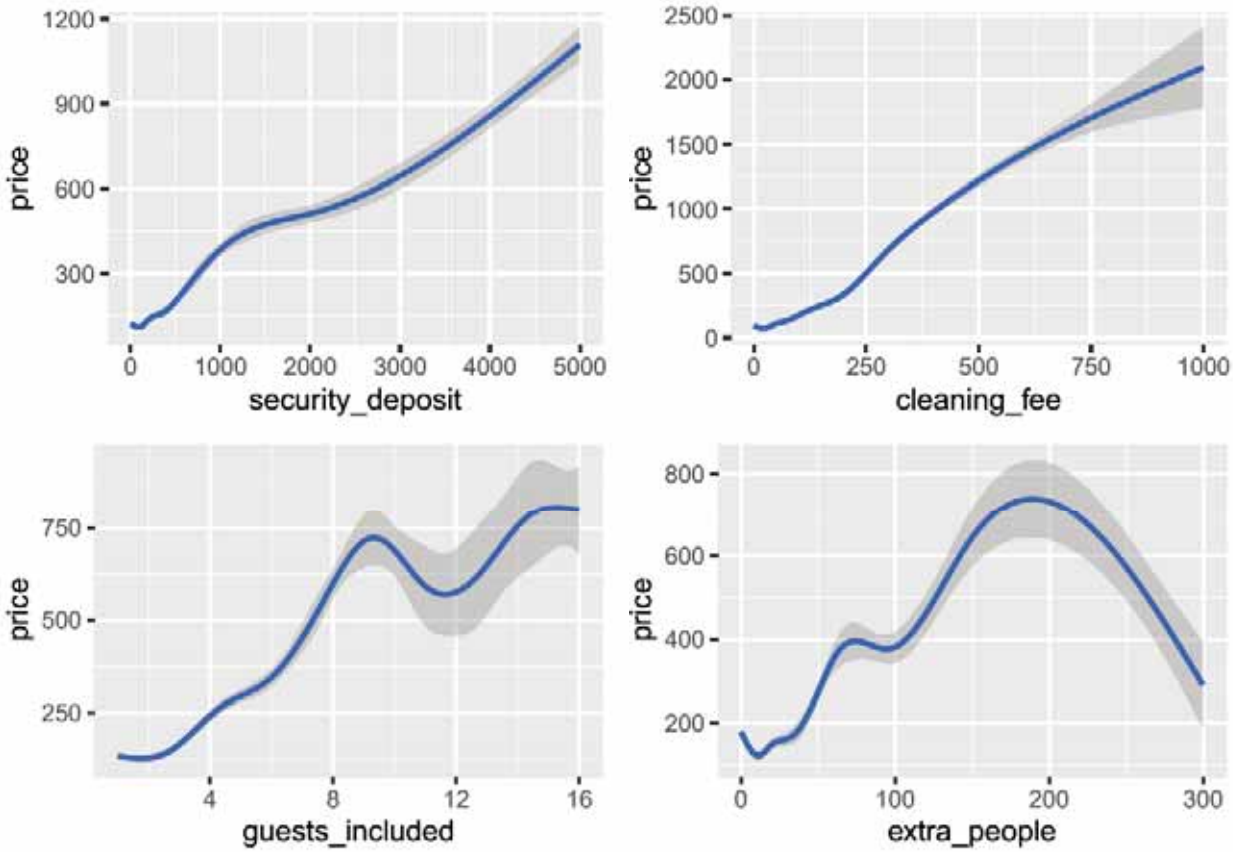

```
varImpPlot(mod2)
```

mod2

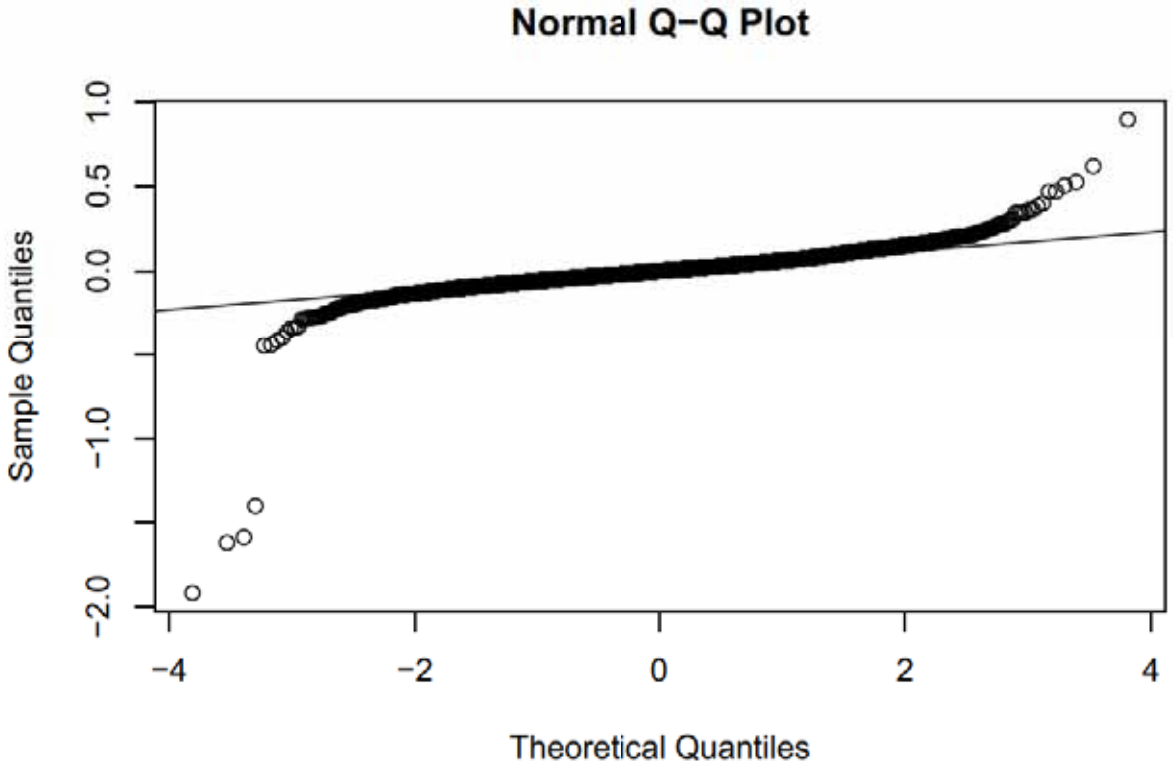



```
# select top 6
finaldata<-data[,c(1,2,3,4,5)]
finaldata$neigh<-neigh
finaldata$cancelPolicy<-data$cancellation_policy
finaldata$instant_bookable<-data$instant_bookable
```

```
#find relationship between price and each variable
p1<-ggplot(finaldata, aes(x=security_deposit, y=price)) +
  geom_smooth()
p2<-ggplot(finaldata, aes(x=cleaning_fee, y=price)) +
  geom_smooth()
p3<-ggplot(finaldata, aes(x=guests_included, y=price)) +
  geom_smooth()
p4<-ggplot(finaldata, aes(x=extra_people, y=price)) +
  geom_smooth()
grid.arrange(p1,p2,p3,p4,nrow=2)
```




```
# random intercept
finaldata<-na.omit(finaldata)
mod1<-lmer(price^0.1~security_deposit+cleaning_fee+guests_included+(~extra_people^2)+cancelPolicy+insta
# normality of residual
qqnorm(residuals(mod1))
qqline(residuals(mod1))
```




```
summary(mod1)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: price^0.1 ~ security_deposit + cleaning_fee + guests_included +
##     (-extra_people^2) + cancelPolicy + instant_bookable + (1 |      neigh)
##    Data: finaldata
##
## REML criterion at convergence: -15261.8
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -23.4915  -0.4870  -0.0133   0.4678  10.9907
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  neigh    (Intercept) 0.001400 0.03742
##  Residual             0.006661 0.08162
## Number of obs: 7180, groups:  neigh, 147
##
## Fixed effects:
##                                          Estimate Std. Error        df t value
## (Intercept)                             1.486e+00  4.791e-03 3.414e+02 310.174
## security_deposit                        1.287e-05  2.056e-06 7.119e+03   6.261
## cleaning_fee                            7.978e-04  1.766e-05 7.156e+03  45.165
## guests_included                         1.658e-02  5.915e-04 7.163e+03  28.025
## cancelPolicymoderate                    6.961e-03  3.422e-03 7.155e+03   2.034
## cancelPolicystrict_14_with_grace_period 3.465e-04  3.391e-03 7.156e+03   0.102
## cancelPolicysuper_strict_30             3.792e-02  4.125e-02 7.069e+03   0.919
```

```
## cancelPolicysuper_strict_60             1.756e-01  2.544e-02 7.053e+03   6.902
## instant_bookablet                       6.266e-03  2.027e-03 7.137e+03   3.091
##                                        Pr(>|t|)
## (Intercept)                             < 2e-16 ***
## security_deposit                       4.06e-10 ***
## cleaning_fee                            < 2e-16 ***
## guests_included                         < 2e-16 ***
## cancelPolicymoderate                      0.042 *
## cancelPolicystrict_14_with_grace_period   0.919
## cancelPolicysuper_strict_30               0.358
## cancelPolicysuper_strict_60            5.55e-12 ***
## instant_bookablet                         0.002 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) scrty_ clnng_ gsts_n cnclPl cP_14_ cP__30 cP__60
## secrty_dpst -0.015
## cleaning_fe -0.042 -0.494
## gusts_ncldd -0.139  0.056 -0.461
## cnclPlcymdr -0.527 -0.017 -0.027 -0.052
## cnclP_14___ -0.515 -0.025 -0.111 -0.024  0.801
## cnclPlc__30 -0.043 -0.007 -0.008 -0.009  0.067  0.071
## cnclPlc__60 -0.064 -0.061 -0.099  0.082  0.112  0.131  0.010
## instnt_bkbl -0.184  0.089  0.020 -0.062  0.040  0.023  0.008 -0.021
```

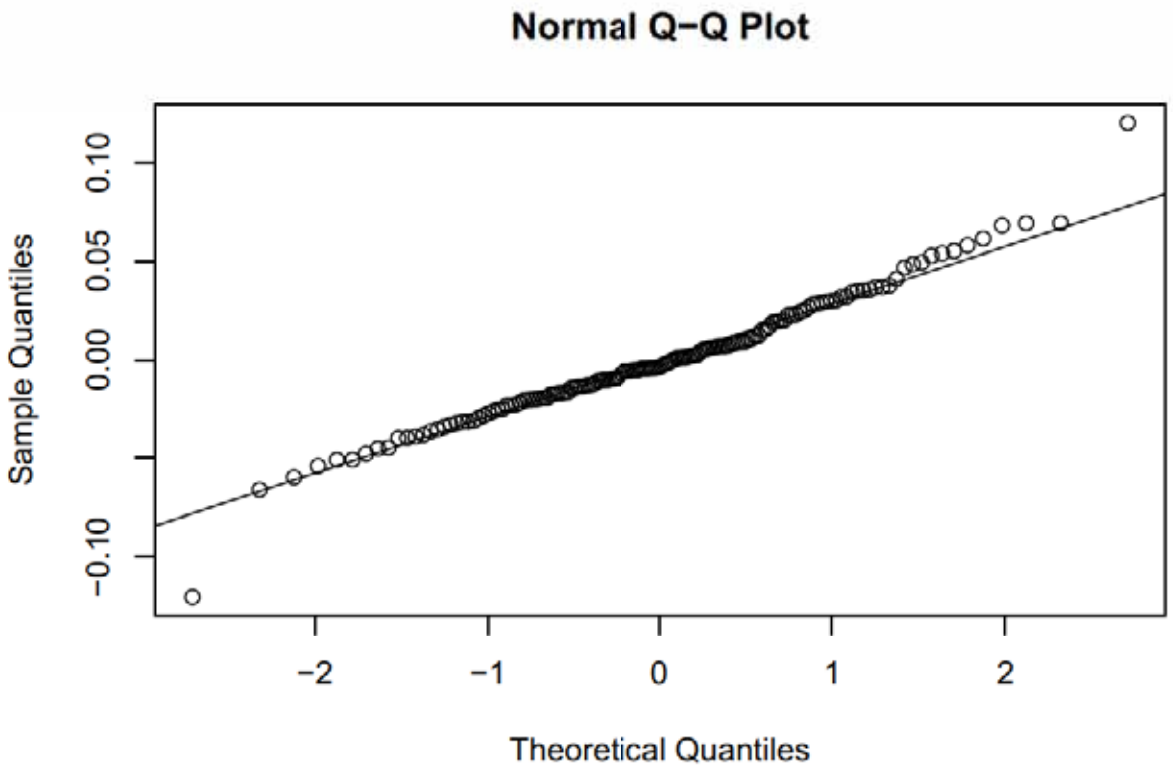
```
#head(coef(mod1))
```

```
library(arm)
display(mod1)
```

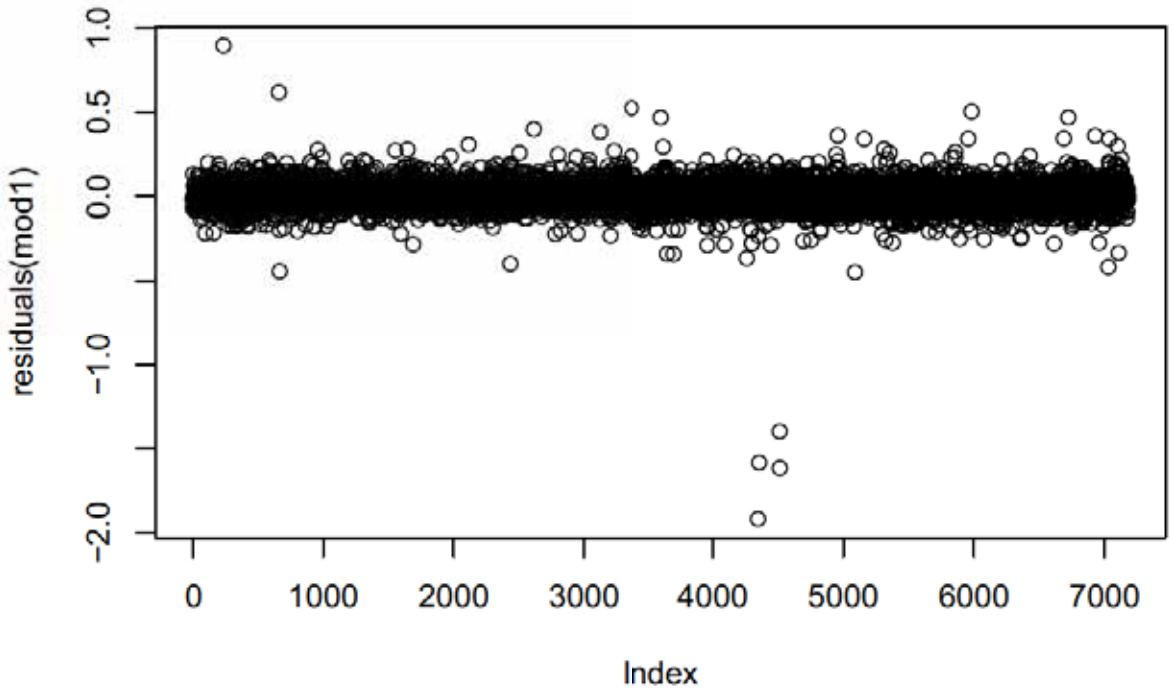
```
## lmer(formula = price^0.1 ~ security_deposit + cleaning_fee +
##     guests_included + (-extra_people^2) + cancelPolicy + instant_bookable +
##     (1 | neigh), data = finaldata)
##                                        coef.est coef.se
## (Intercept)                            1.49     0.00
## security_deposit                       0.00     0.00
## cleaning_fee                           0.00     0.00
## guests_included                        0.02     0.00
## cancelPolicymoderate                   0.01     0.00
## cancelPolicystrict_14_with_grace_period 0.00    0.00
## cancelPolicysuper_strict_30            0.04     0.04
## cancelPolicysuper_strict_60            0.18     0.03
## instant_bookablet                      0.01     0.00
##
## Error terms:
##  Groups   Name        Std.Dev.
##  neigh    (Intercept) 0.04
##  Residual             0.08
## ---
## number of obs: 7180, groups: neigh, 147
## AIC = -15239.8, DIC = -15478.1
## deviance = -15369.9
```

```r
# normality of parameters
para<-data.frame(ranef(mod1))
```

```r
qqnorm(para[,4])
qqline(para[,4])
```




```r
# independentce of residual
plot(residuals(mod1))
# constant variable
plot(residuals(mod1))
```



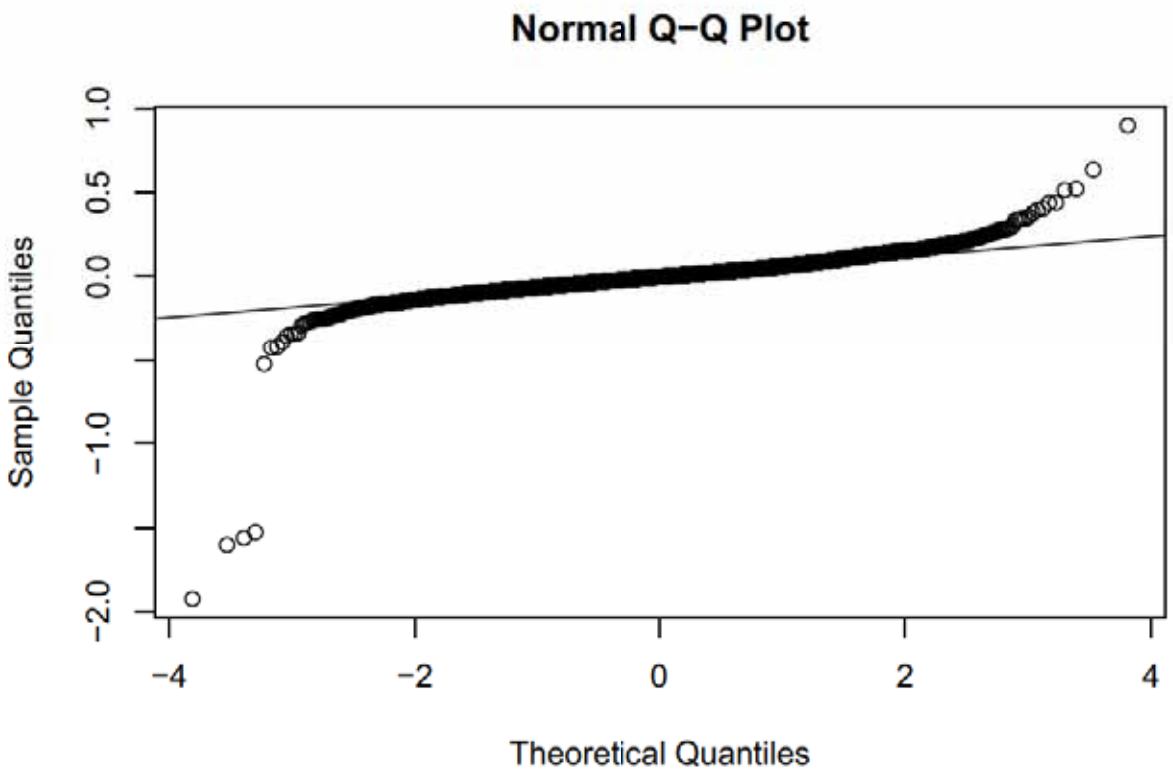

```
# check multicollinearity
vif(mod1)
```

```
##                       GVIF Df GVIF^(1/(2*Df))
## security_deposit 1.451470  1        1.204770
## cleaning_fee     1.869159  1        1.367172
## guests_included  1.355484  1        1.164252
## cancelPolicy     1.076269  4        1.009230
## instant_bookable 1.016724  1        1.008327
```

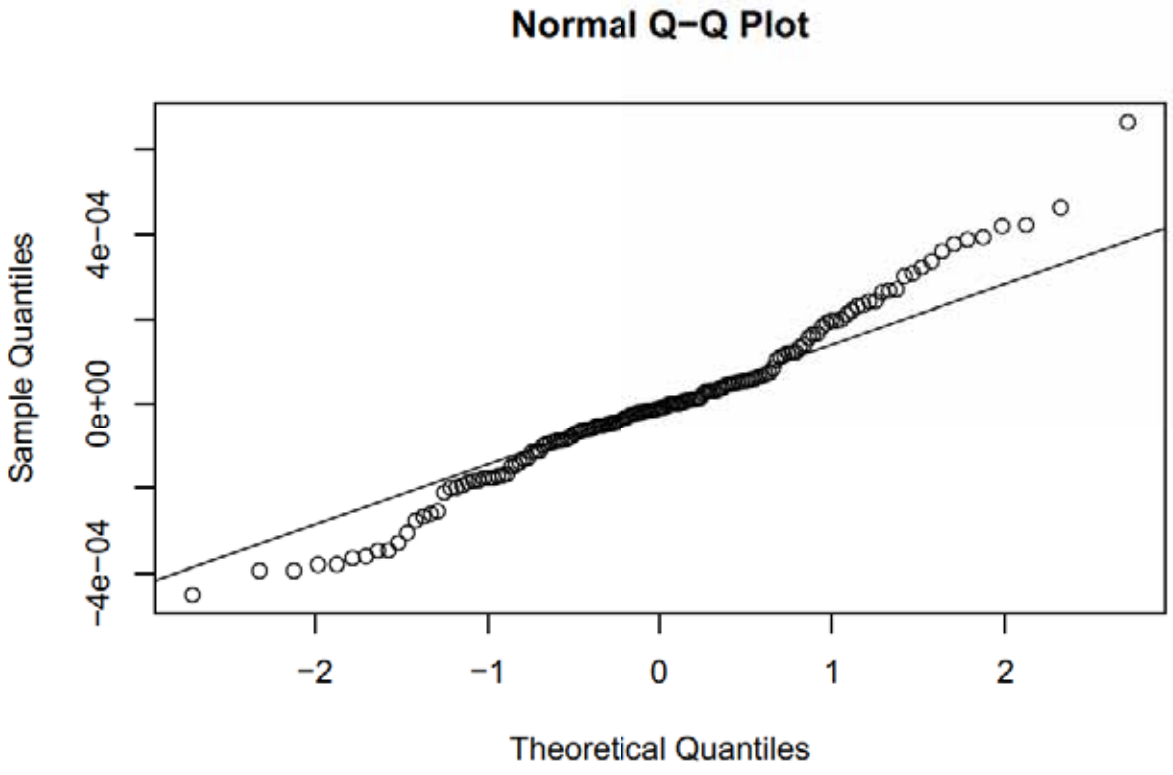
```
#calculate MSE after cross validation for model 1
mse1<-looCv(mod1)
mse1
```

```
## [1] 0.006778095
```

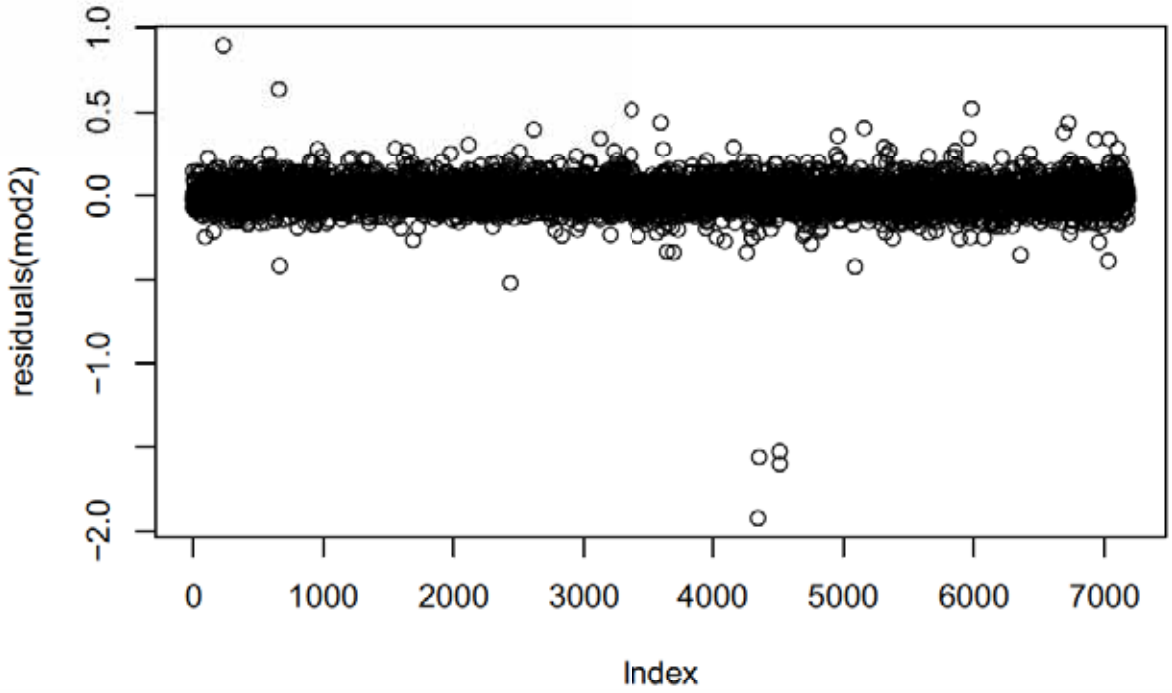
```
# random slope
mod2<-lmer(price^0.1~security_deposit+cleaning_fee+guests_included+cancelPolicy+instant_bookable+(0+cle
# normality of residual
qqnorm(residuals(mod2))
qqline(residuals(mod2))
```




```
# normality of parameters
para<-data.frame(ranef(mod2))
qqnorm(para[,4])
qqline(para[,4])
```



```r
# independentce of residual
plot(residuals(mod2))
# constant variance
plot(residuals(mod2))
```




```r
# check multicollinearity
vif(mod2)
```

```
##                      GVIF Df GVIF^(1/(2*Df))
## security_deposit 1.081429  1        1.039918
## cleaning_fee     1.174056  1        1.083539
## guests_included  1.107925  1        1.052580
## cancelPolicy     1.022733  4        1.002814
## instant_bookable 1.015915  1        1.007926
```

```r
summary(mod2)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: price^0.1 ~ security_deposit + cleaning_fee + guests_included +
##     cancelPolicy + instant_bookable + (0 + cleaning_fee | neigh)
##    Data: finaldata
##
## REML criterion at convergence: -15030.7
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -23.1078  -0.5090  -0.0122   0.4734  10.7987
##
## Random effects:
```

```
##  Groups   Name         Variance  Std.Dev.
##  neigh    cleaning_fee 7.136e-08 0.0002671
##  Residual              6.943e-03 0.0833226
## Number of obs: 7180, groups:  neigh, 147
##
## Fixed effects:
##                                          Estimate Std. Error        df t value
## (Intercept)                             1.499e+00  3.339e-03 7.148e+03 448.901
## security_deposit                        1.365e-05  2.150e-06 7.170e+03   6.348
## cleaning_fee                            6.917e-04  3.556e-05 1.547e+02  19.450
## guests_included                         1.641e-02  6.213e-04 7.062e+03  26.406
## cancelPolicymoderate                    9.727e-03  3.454e-03 7.125e+03   2.816
## cancelPolicystrict_14_with_grace_period 4.008e-03  3.417e-03 7.132e+03   1.173
## cancelPolicysuper_strict_30             3.248e-02  4.197e-02 7.089e+03   0.774
## cancelPolicysuper_strict_60             1.704e-01  2.646e-02 7.087e+03   6.442
## instant_bookablet                       5.816e-03  2.051e-03 7.150e+03   2.835
##                                         Pr(>|t|)
## (Intercept)                              < 2e-16 ***
## security_deposit                        2.31e-10 ***
## cleaning_fee                             < 2e-16 ***
## guests_included                          < 2e-16 ***
## cancelPolicymoderate                     0.00487 **
## cancelPolicystrict_14_with_grace_period  0.24086
## cancelPolicysuper_strict_30              0.43891
## cancelPolicysuper_strict_60             1.26e-10 ***
## instant_bookablet                        0.00459 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) scrty_ clnng_ gsts_n cnclPl cP_14_ cP__30 cP__60
## secrty_dpst -0.014
## cleaning_fe -0.079 -0.246
## gusts_ncldd -0.167  0.044 -0.295
## cnclPlcymdr -0.778 -0.021 -0.020 -0.044
## cnclP_14___ -0.766 -0.029 -0.061 -0.015  0.800
## cnclPlc__30 -0.062 -0.006 -0.002 -0.012  0.066  0.071
## cnclPlc__60 -0.096 -0.058 -0.031  0.069  0.106  0.124  0.008
## instnt_bkbl -0.285  0.087  0.025 -0.068  0.043  0.023  0.007 -0.020
## convergence code: 0
## Model failed to converge with max|grad| = 0.0433245 (tol = 0.002, component 1)
## Model is nearly unidentifiable: very large eigenvalue
##  - Rescale variables?
```

```
#coef(mod2)
```

```
display(mod2)
```

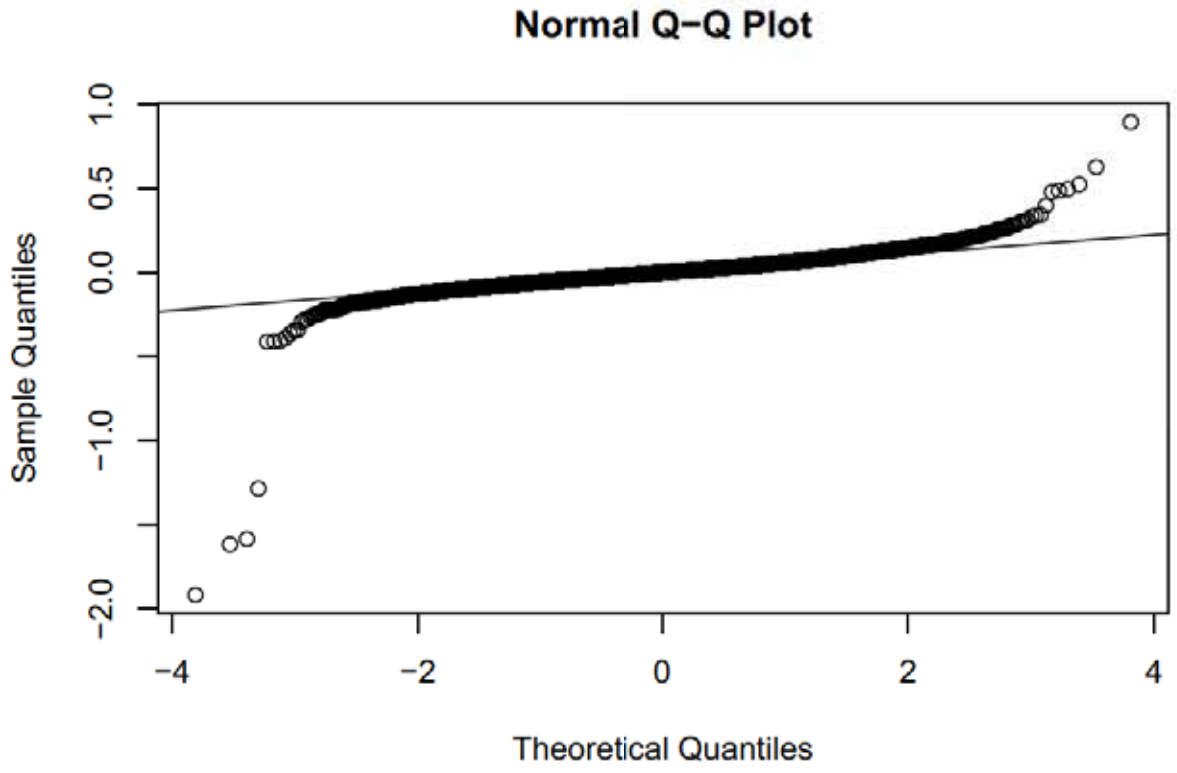
```
## lmer(formula = price^0.1 ~ security_deposit + cleaning_fee +
##     guests_included + cancelPolicy + instant_bookable + (0 +
##     cleaning_fee | neigh), data = finaldata)
##                                         coef.est coef.se
## (Intercept)                             1.50     0.00
```

```
## security_deposit                          0.00     0.00
## cleaning_fee                              0.00     0.00
## guests_included                           0.02     0.00
## cancelPolicymoderate                      0.01     0.00
## cancelPolicystrict_14_with_grace_period 0.00     0.00
## cancelPolicysuper_strict_30               0.03     0.04
## cancelPolicysuper_strict_60               0.17     0.03
## instant_bookablet                         0.01     0.00
##
## Error terms:
##  Groups   Name          Std.Dev.
##  neigh    cleaning_fee 0.00
##  Residual               0.08
## ---
## number of obs: 7180, groups: neigh, 147
## AIC = -15008.7, DIC = -15246.3
## deviance = -15138.5
```

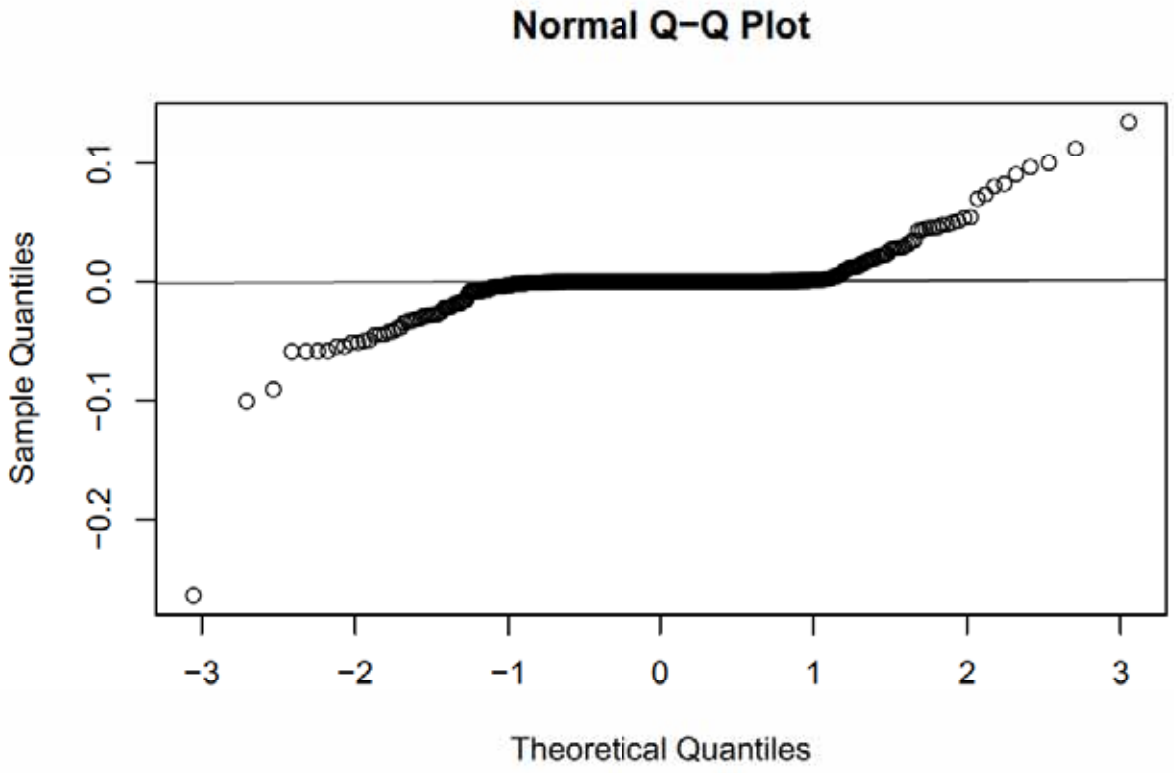
```
#calculate MSE after cross validation for model 2
mse2<-looCv(mod2)
mse2
```

```
## [1] 0.00709287
```

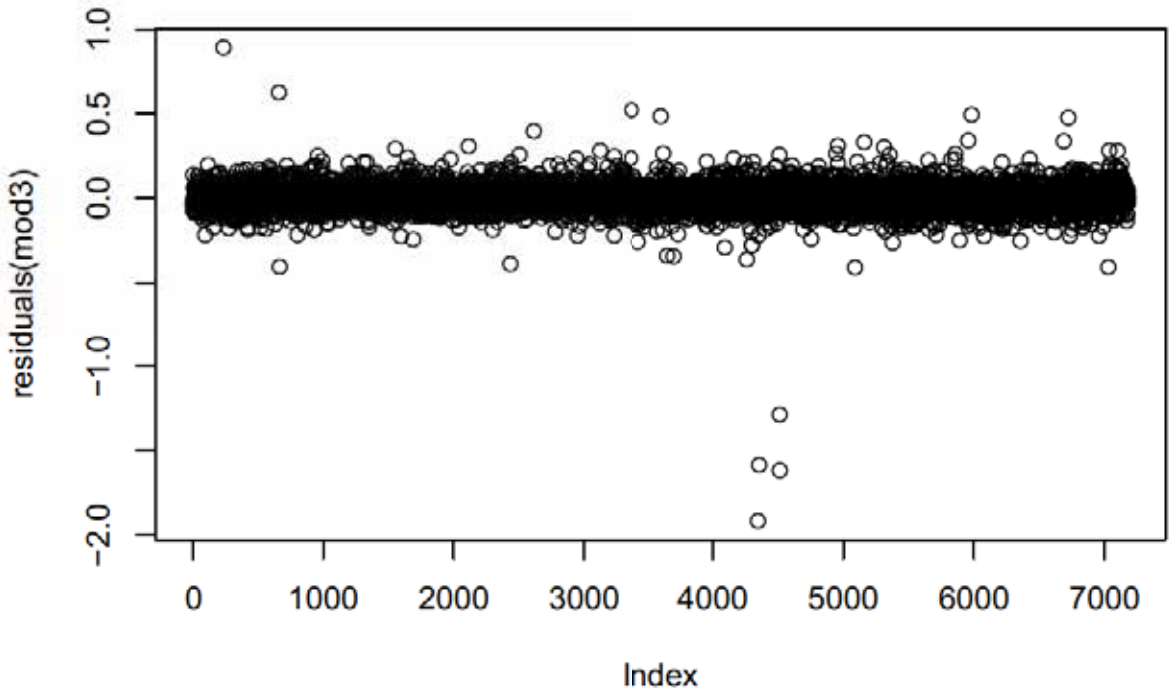
```
# random intercept and slope
mod3<-lmer(price^0.1~security_deposit+cleaning_fee+guests_included+extra_people+cancelPolicy+instant_bo
# normality of residual
qqnorm(residuals(mod3))
qqline(residuals(mod3))
```



```r
# normality of parameters
para<-data.frame(ranef(mod3))
qqnorm(para[,4])
qqline(para[,4])
```



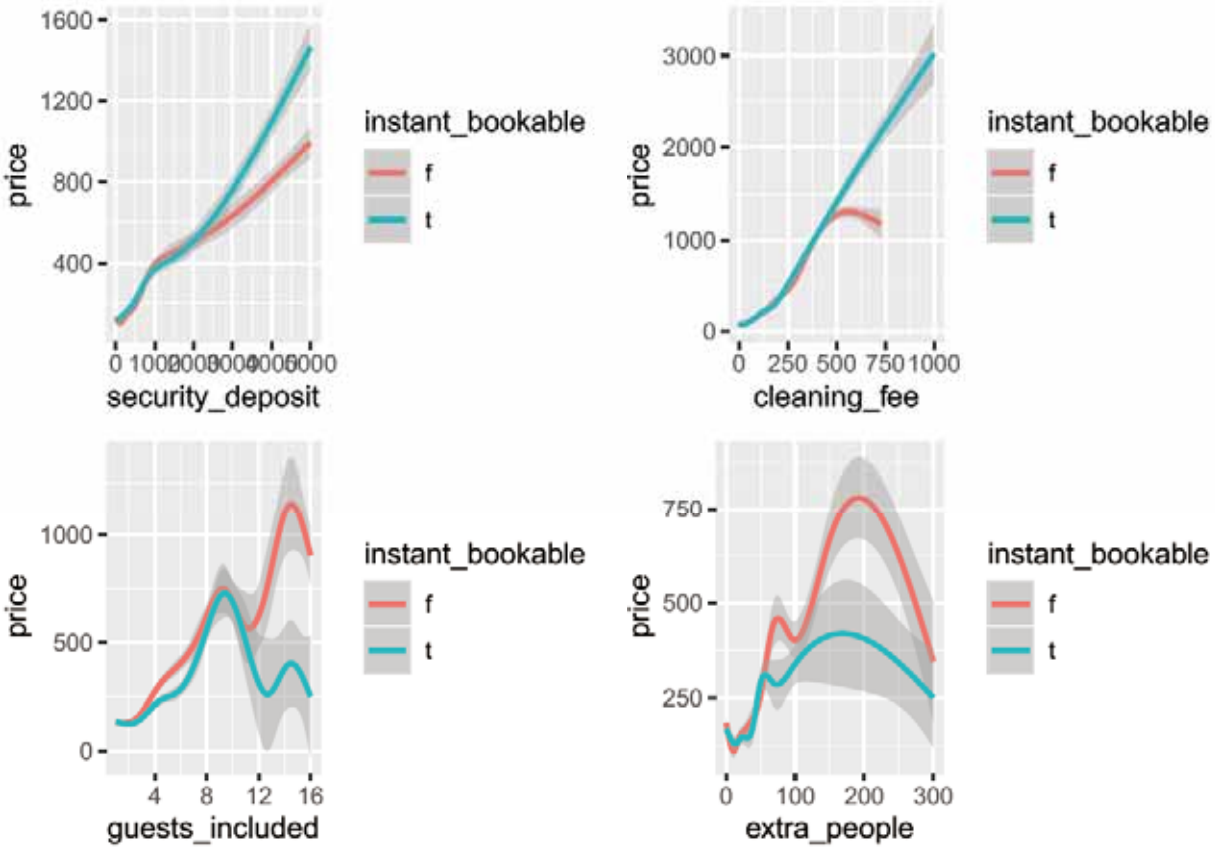

```r
# independentce of residual
plot(residuals(mod3))
# constant variable
plot(residuals(mod3))
```




```
# check multicollinearity
vif(mod3)
```

```
##                      GVIF Df GVIF^(1/(2*Df))
## security_deposit 1.013516  1        1.006735
## cleaning_fee     1.006826  1        1.003407
## guests_included  1.059053  1        1.029103
## extra_people     1.052307  1        1.025820
## cancelPolicy     1.012975  4        1.001613
## instant_bookable 1.009998  1        1.004987
```

```
#graph relationship between price and each variable grouped by different category of instant_bookable
p1<-ggplot(finaldata, aes(x=security_deposit, y=price,color=instant_bookable)) +
geom_smooth()
p2<-ggplot(finaldata, aes(x=cleaning_fee, y=price,color=instant_bookable)) +
  geom_smooth()
p3<-ggplot(finaldata, aes(x=guests_included, y=price,color=instant_bookable)) +
  geom_smooth()
p4<-ggplot(finaldata, aes(x=extra_people, y=price,color=instant_bookable)) +
  geom_smooth()
grid.arrange(p1,p2,p3,p4,nrow=2)
```




```
#calculate MSE after cross validation for model 3
mse3<-looCv(mod3)
mse3
```

```
## [1] 0.006991225
```

```
compare <- cbind(mse1,mse2,mse3)%>%as.data.frame()
knitr::kable(compare)%>%kableExtra::kable_styling(bootstrap_options = c("striped", "hover"))
```

mse1	mse2	mse3
0.0067781	0.0070929	0.0069912

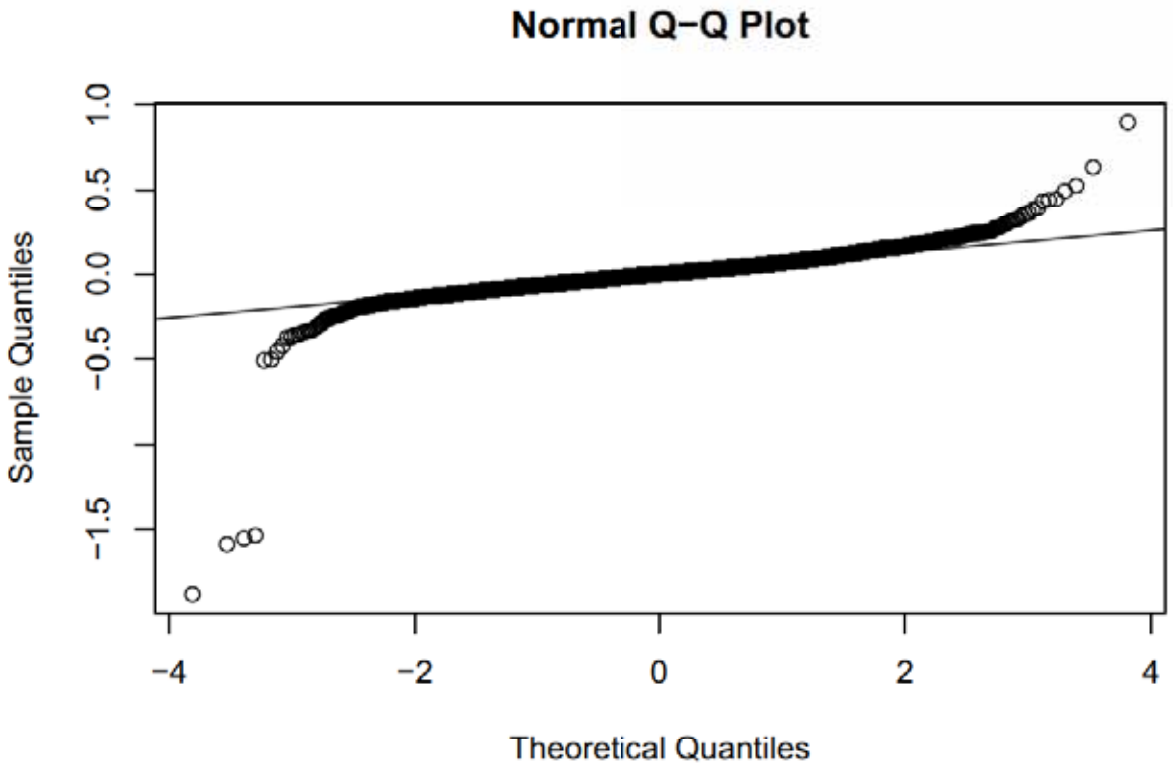
```
# random intercept
finaldata<-na.omit(finaldata)
mod4<-lmer(price^0.1~security_deposit+cleaning_fee+guests_included+(-extra_people^2)+cancelPolicy+(1|in
```

```
display(mod4)
```

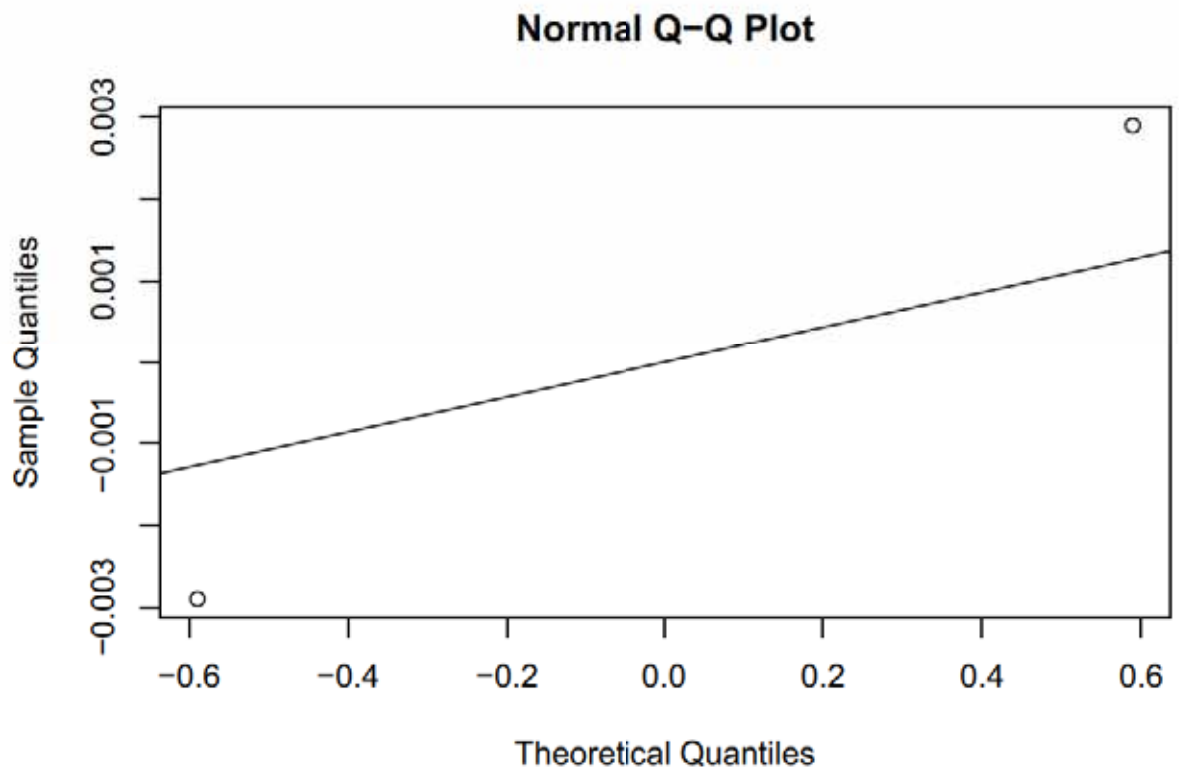
```
## lmer(formula = price^0.1 ~ security_deposit + cleaning_fee +
##     guests_included + (-extra_people^2) + cancelPolicy + (1 |
##     instant_bookable), data = finaldata)
##                                     coef.est coef.se
## (Intercept)                         1.50     0.00
## security_deposit                    0.00     0.00
```

```
## cleaning_fee                              0.00     0.00
## guests_included                           0.02     0.00
## cancelPolicymoderate                      0.01     0.00
## cancelPolicystrict_14_with_grace_period 0.01     0.00
## cancelPolicysuper_strict_30               0.05     0.04
## cancelPolicysuper_strict_60               0.20     0.03
##
## Error terms:
##  Groups           Name        Std.Dev.
##  instant_bookable (Intercept) 0.00
##  Residual                     0.09
## ---
## number of obs: 7180, groups: instant_bookable, 2
## AIC = -14635.2, DIC = -14850.8
## deviance = -14753.0
```

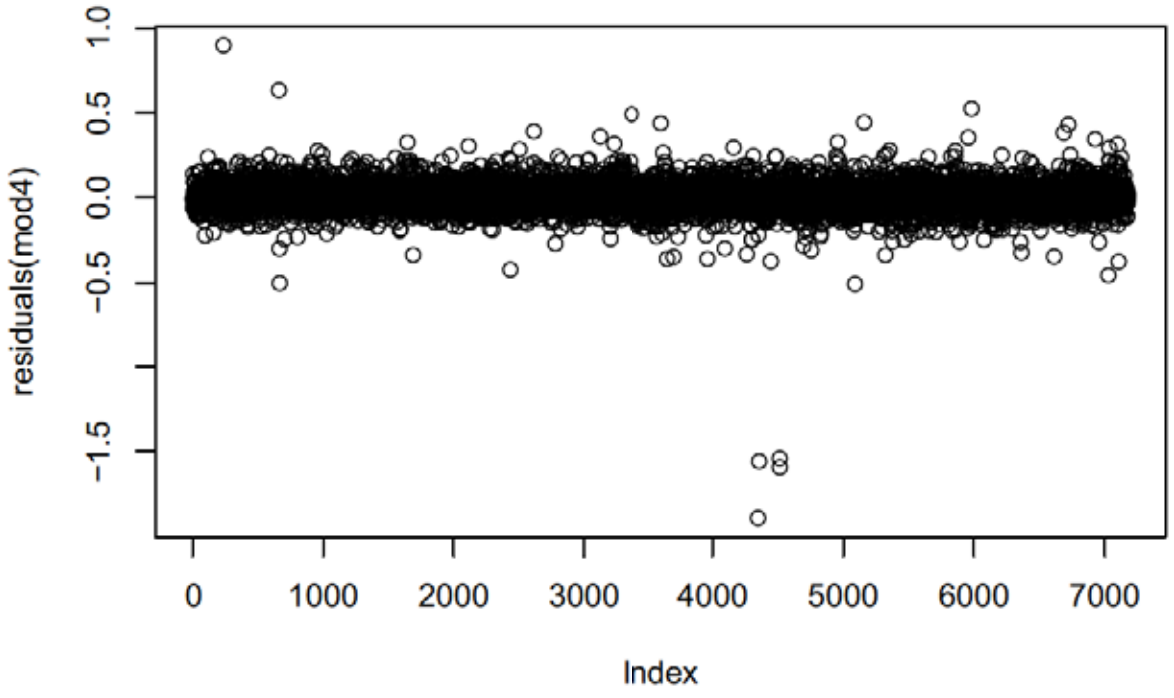
```
# normality of residual
qqnorm(residuals(mod4))
qqline(residuals(mod4))
```




```
# normality of parameters
para<-data.frame(ranef(mod4))
qqnorm(para[,4])
qqline(para[,4])
```




```r
# independentce of residual
plot(residuals(mod4))

# constant variable

plot(residuals(mod4))
```



```r
# check multicollinearity
vif(mod4)
```

```
##                       GVIF Df GVIF^(1/(2*Df))
## security_deposit 1.507307  1        1.227724
## cleaning_fee     1.954425  1        1.398007
## guests_included  1.355431  1        1.164230
## cancelPolicy     1.085778  4        1.010340
```

```r
mse4<-looCv(mod4)
mse4
```

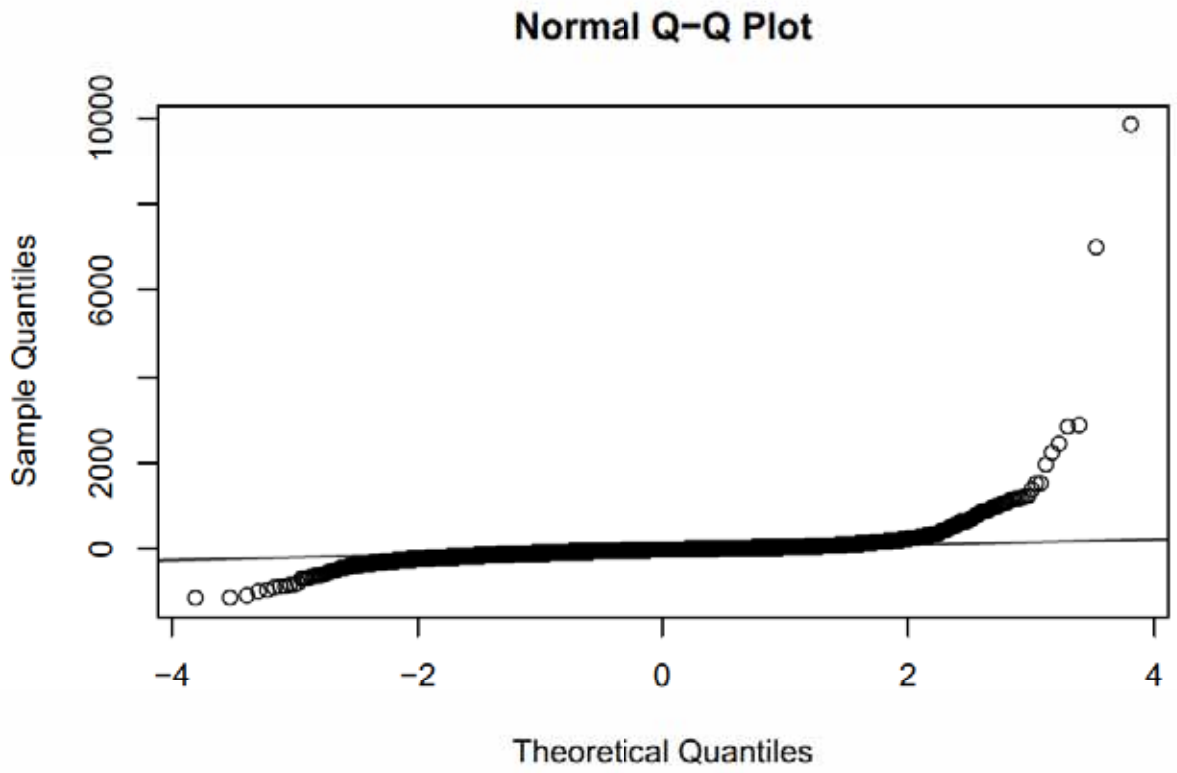
```
## [1] 0.007533975
```

```r
# random slope
glmerControl(optimizer="bobyqa", optCtrl = list(maxfun = 10000000))
```

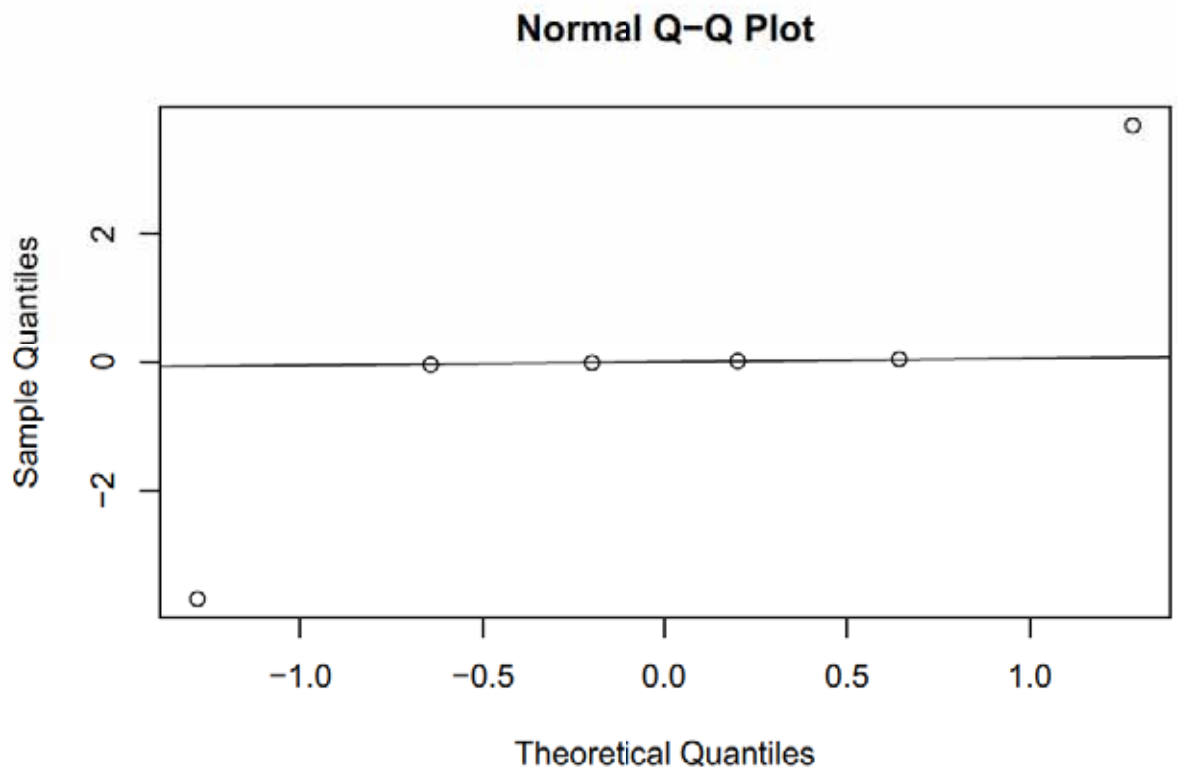
```
## $optimizer
## [1] "bobyqa" "bobyqa"
##
## $calc.derivs
## [1] TRUE
##
## $use.last.params
## [1] FALSE
##
## $restart_edge
## [1] FALSE
##
## $boundary.tol
## [1] 1e-05
##
## $tolPwrss
## [1] 1e-07
##
## $compDev
## [1] TRUE
##
## $nAGQ0initStep
## [1] TRUE
##
## $checkControl
## $checkControl$check.nobs.vs.rankZ
## [1] "ignore"
##
## $checkControl$check.nobs.vs.nlev
## [1] "stop"
##
## $checkControl$check.nlev.gtreq.5
## [1] "ignore"
##
## $checkControl$check.nlev.gtr.1
```

```
## [1] "stop"
##
## $checkControl$check.nobs.vs.nRE
## [1] "stop"
##
## $checkControl$check.rankX
## [1] "message+drop.cols"
##
## $checkControl$check.scaleX
## [1] "warning"
##
## $checkControl$check.formula.LHS
## [1] "stop"
##
## $checkControl$check.response.not.const
## [1] "stop"
##
##
## $checkConv
## $checkConv$check.conv.grad
## $checkConv$check.conv.grad$action
## [1] "warning"
##
## $checkConv$check.conv.grad$tol
## [1] 0.001
##
## $checkConv$check.conv.grad$relTol
## NULL
##
##
## $checkConv$check.conv.singular
## $checkConv$check.conv.singular$action
## [1] "message"
##
## $checkConv$check.conv.singular$tol
## [1] 1e-04
##
##
## $checkConv$check.conv.hess
## $checkConv$check.conv.hess$action
## [1] "warning"
##
## $checkConv$check.conv.hess$tol
## [1] 1e-06
##
##
##
## $optCtrl
## $optCtrl$maxfun
## [1] 1e+07
##
##
## attr(,"class")
## [1] "glmerControl" "merControl"
```

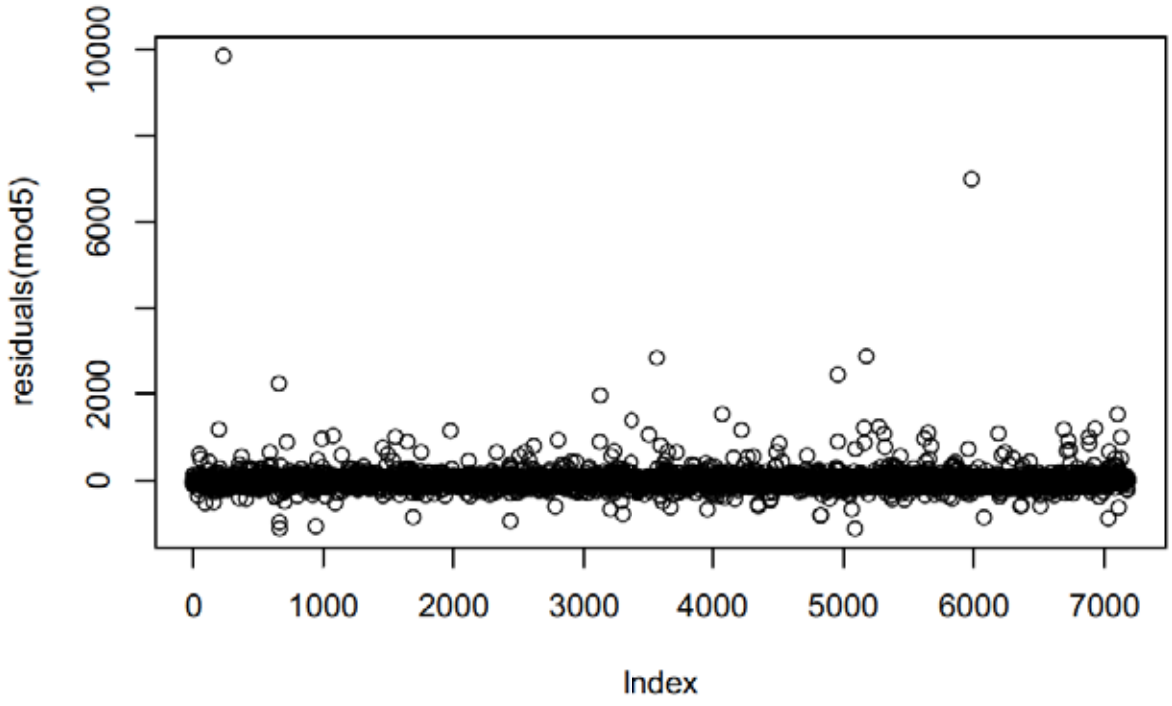
```r
mod5<-lmer(price~security_deposit+cleaning_fee+guests_included+cancelPolicy+(0+security_deposit+cleanin
# normality of residual
qqnorm(residuals(mod5))
qqline(residuals(mod5))
```




```r
# normality of parameters
para<-data.frame(ranef(mod5))
qqnorm(para[,4])
qqline(para[,4])
```




```r
# independentce of residual
plot(residuals(mod5))

# constant variance

plot(residuals(mod5))
```



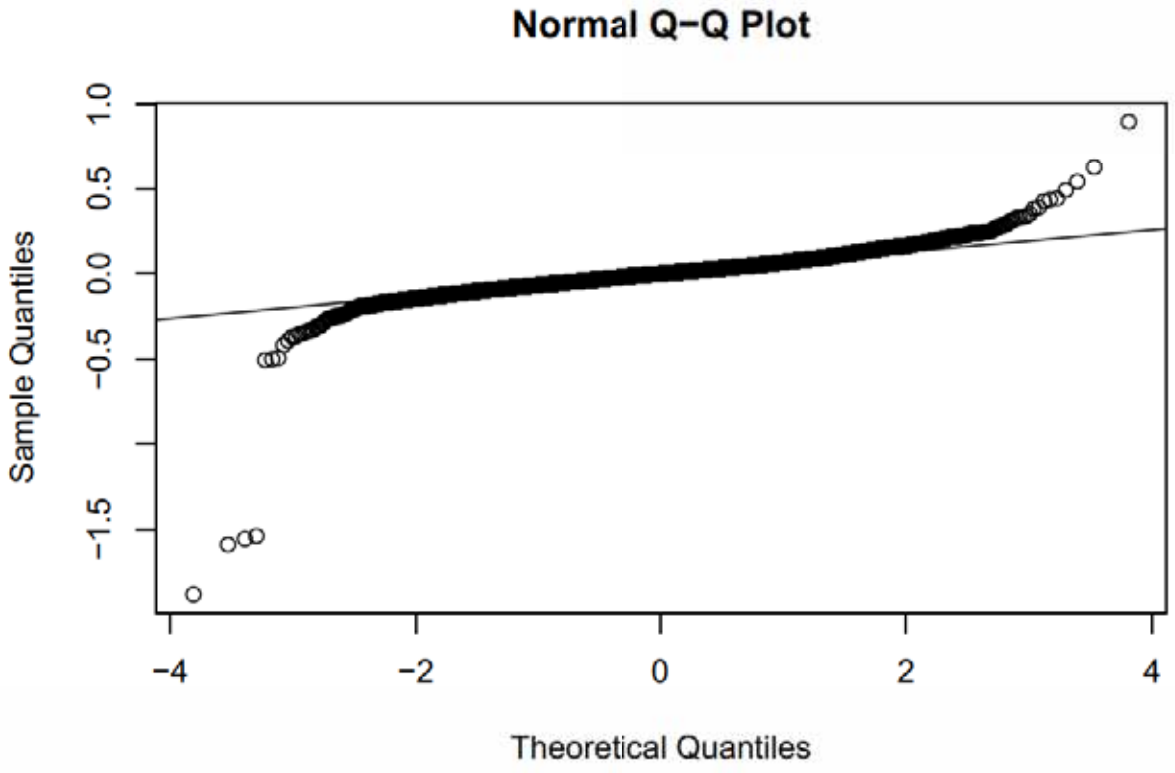
```r
# check multicollinearity
vif(mod5)
```

```
##                       GVIF Df GVIF^(1/(2*Df))
## security_deposit 104.127835  1       10.204305
## cleaning_fee      68.753588  1        8.291778
## guests_included   53.518046  1        7.315603
## cancelPolicy       1.021105  4        1.002614
```

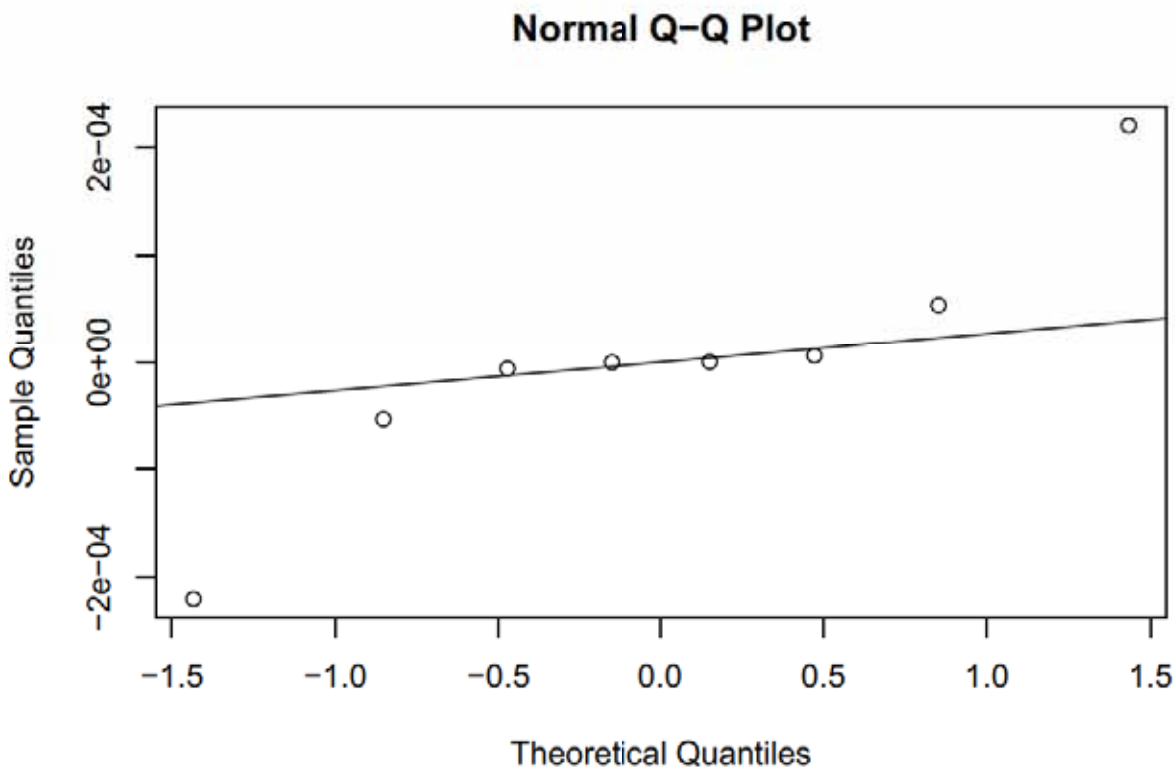
```r
mse5<-looCv(mod5)
mse5
```

```
## [1] 42346.08
```

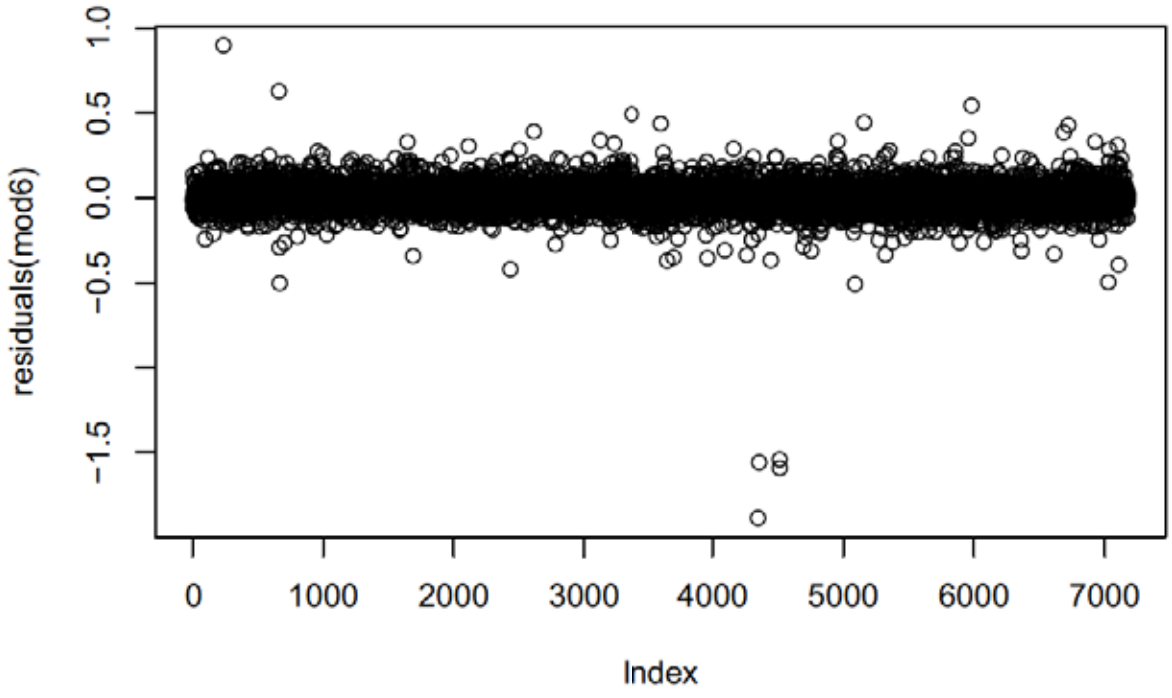
```r
# random intercept and slope
mod6<-lmer(price^0.1~security_deposit+cleaning_fee+guests_included+cancelPolicy+(1+security_deposit+cle
# normality of residual
qqnorm(residuals(mod6))
qqline(residuals(mod6))
```




```r
# normality of parameters
para<-data.frame(ranef(mod6))
qqnorm(para[,4])
qqline(para[,4])
```




```
# independentce of residual
plot(residuals(mod6))

# constant variable

plot(residuals(mod6))
```



```r
# check multicollinearity
vif(mod6)
```

```
##                          GVIF Df GVIF^(1/(2*Df))
## security_deposit 1226.330215  1       35.018998
## cleaning_fee     1248.855162  1       35.339145
## guests_included    67.529283  1        8.217620
## cancelPolicy        1.062702  4        1.007631
```

```r
mse6<-looCv(mod6)
mse6
```

```
## [1] 0.007528396
```

```r
compare_new <- cbind(mse4,mse5,mse6)%>%as.data.frame()
knitr::kable(compare_new)%>%kableExtra::kable_styling(bootstrap_options = c("striped", "hover"))
```

mse4	mse5	mse6
0.007534	42346.08	0.0075284

```r
#library(knitr)
#purl("airbnb_project.Rmd")
```