Bryce Richards
Cole Herndon
Jeya Lee

# Pricing Analysis of Singapore AirBnb Listings

## Abstract

Our analysis consists of pricing data from AirBnb units in Singapore. This data comes from Kaggle and was scrapped from AirBnb listings. We had over 7,000 observations that told us the price per night of the AirBnb, room type, number of reviews, reviews per month, location, and region. We wanted to determine what variables were significant in predicting how much a single AirBnb would cost per night. For this analysis our response variable was price per night, our continuous variable was the number of reviews per month, and our categorical variables was room type and geographical region. Room type was broken down to three types, Entire home/apt, Private room or Shared room while there were five groups in the region category. Our Null Hypothesis was that none of our predictors would be significant in determining the price of an AirBnb while our alternative Hypothesis is that at least one predictor is significant in determining the price. We chose to do a multiple linear regression analysis. After running our analysis, we found that our R squared value was .46 so our model does not accurately explain most of the variation in our model, which can be explained by the business model of AirBnb. Our model ouput p-value of 2.2e^-16 so we can reject the null hypothesis and conclude that at least one of our variables is significant in determining the price per night of an AirBnb in Singapore.

## Introduction

In this analysis, we determine what factors or criteria effect the price per night of an AirBnb in Singapore. This model could help predict reasonable AirBnb prices for unit owners and listers. Our response variable was price per night (continuous) and our explanatory variables were reviews per month (continuous), room type (categorical) and region (categorical). Our Null Hypothesis was that none of our predictors would be significant in determining the price of an AirBnb while our alternative Hypothesis is that at least one predictor is significant in determining the price.

## Methods

1. **Source:** Singapore AirBnbs data came from Kaggle "https://www.kaggle.com/jojoker/singapore-airbnb". Data was gathered by 'Inside AirBnb' Website (http://insideairbnb.com/) which website collect the reviews of AirBnb from all over the world to create data sets. Singapore AirBnb data was collected on 28 August 2019 and contains 7907 sample. The data includes 'room id', 'room names', 'host names', 'Singapore region', 'specific places', 'latitude', 'longitude', 'room type', 'price', 'minimum nights', 'number of reviews', 'reviews per month', 'total room or house in host catalog on Airbnb', and 'availability'.
2. **Tools and packages:** We used R as the program for our analysis. We used 'tidyverse', 'gg plot2', 'ggfortify', and 'car' libraries.
3. **Statistical Analysis**
    - **Select Model:** we assumed there exist linear relationships between the at least one covariables (region, number of reviews, and price). Due to the desire to investigate linear relationships between coefficients, we chose to use multiple linear regression.
    - **Independence of Residuals:** In order to perform our analysis, we need to determine whether residuals are independent or not.
    - **Linearity of x vs y:** we assumed there exist linear relationship between price and other variables (room type, number of reviews). To check this assumption, we used added variable plot. Added Variable plots show the relationship between log(price) and each predictor variables (room type, number of reviews).
    - **Normality of residuals and residuals centered at zero:** We assumed residuals are normally distributed and centered at zero to check this we used box plot for residuals. We checked this assumption using a Box Plot of the residuals which show how they are distributed.

- **Residuals are Homoscedastic:** We assumed residuals are homoscedastic to check whether the data is homoscedastic we used Brown-Forsythe test. The Brown-Forsythe test can tell us whether or not our residuals are homoscedastic.
- **The model describes all observation:** we assumed our model describes all observation. To check this, we calculated the DFFITS and DFBETAS. DEFFITS and DFBETAS use various cutoff values to determine possible influential points.
- **No multicollinearity:** We assumed there is no multicollinearity in our data. To check this, we calculated Variance Inflation Factors. VIFs will tell us whether our data is multicollinearity. If the VIF is > 10, then we have multicollinearity problem.

## Results

After performing the methods described above, we received the following linear regression model that allows us to estimate Average Log Price given values for our coefficients:

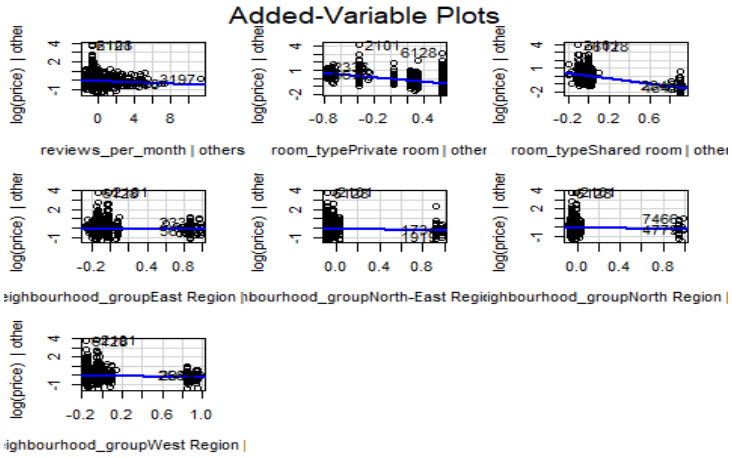
$\widehat{\text{Log(Price)}}$ = 5.296 + -0.03375**reviews_per_month + -0.8956**I(room_type = Private Room) + -1.5807**I(room_type = Shared Room) + -0.0950**I(neighbourhood_group = East Region) + -0.2882**I(neighbourhood_group = North Region) + -0.1852 ** I(neighbourhood_group = North-East Region) + -0.1873*I(neighbourhood_group = West Region)

The model summary information contained in the appendix details the significance of the effect that each of our coefficients has on average Log Price. With a P value of 0.18909 with an alpha of 0.05, the Neighbourhood Group East Region has no significant effect on Log Price. With a P value of 0.05597, with an alpha of 0.05, the Neighbourhood Group North-East Region has no significant effect on Log Price. All other coefficients have a P value less than 0.05, meaning that their impact on Log Price is significant and valuable future analysis. After receiving the output of our model, it was very important for us to check Multiple Linear Regression model assumptions to verify that our results are valid. In order to better

meet these assumptions, we performed a log y transformation on price. The beneath section will detail these assumptions and the tests performed on our model.

1. X vs Y is linear.

The below added variable plots demonstrate that there is a linear relationship between X and Y. We are looking for linear trends in each plot, and we do see this in our data, even if it isn't very strong.



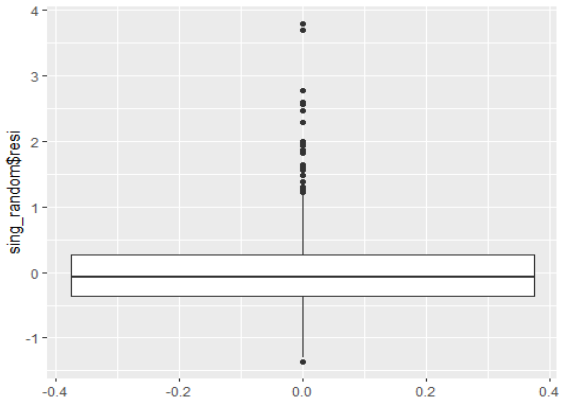

*See Supplementary Materia (R Code) for additional tests on this assumption.*

2. The residuals are independent across all values of y.

This assumption can be met because we are taking a random sample from our data set. This allows our residuals to be independent.

3. The residuals are normally distributed and centered at zero.

The blow graph is a boxplot of the residuals received from our model. You will see that there is some right skewness in our residuals; however, this is vastly improved from the original data before our Log transformation. We feel comfortable moving forward with our linear model from here, knowing that we did everything we could to minimize skewness. This will lead us to later be more cautious in interpreting our P values.




*See Supplementary Material for additional tests on this assumption.*

4. The residuals are homoscedastic.

With a P value of 0.04381 at an alpha of 0.05, we used the Brown-Foresythe test to confirm that the residuals are homoscedastic.

```
Levene's Test for Homogeneity of Variance (center = median)
       Df F value  Pr(>F)
group   1  4.0742 0.04381 *
      998
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

*See Supplementary Material for additional tests on this assumption.*

5. The model describes all observations

In order to test this assumption, we calculated the DFFITS and DFBETAS on our data to determine if they are any influential points in our data. We discovered that one observation was an influential point but felt comfortable proceeding with only one point like this in our data. *See supplementary material for influential observation (R code DFBETAS).*

*6.* No multicollinearity.

With no Variance Inflation Factors anywhere close to 10, we are confident that we have no multicollinearity in our data.

```
vif(sing_random_log_y.lm)
```

```
##                          GVIF Df GVIF^(1/(2*Df))
## reviews_per_month    1.021333  1        1.010610
## room_type            1.100557  2        1.024243
## neighbourhood_group 1.118002  4        1.014041
```

Our model assumptions are met and lead us to accept that our data is useful for multiple linear regression. We next looked at confidence intervals for each of our coefficients to help better determine the true effect of each coefficient on Log price.

```
                                        2.5 %       97.5 %
(Intercept)                          5.24228532  5.350439681
reviews_per_month                   -0.06260957 -0.004892606
room_typePrivate room               -0.97215909 -0.819078095
room_typeShared room                -1.74041693 -1.421084324
neighbourhood_groupEast Region      -0.23707729  0.046898853
neighbourhood_groupNorth-East Region -0.37522887  0.004725880
neighbourhood_groupNorth Region     -0.51768837 -0.058873721
neighbourhood_groupWest Region      -0.32839762 -0.046389428
```

Lastly, we need to evaluate our model and see how well it performed in explaining the variation in the data. To do this we looked at an Adjusted R-squared value. With an Adjusted R-squared of 0.4606 our linear model only explains 46.06% of the variation in log price of Singapore AirBnb listings, meaning that it is not incredibly accurate in doing so. At first glance, this can be troublesome, but it does make sense in the context of our analysis. AirBnb price listings are controlled by thousands of individuals who often can set prices without any major explanation behind their methods. Our model demonstrates that only 46% of log price can be explained by the variables we examined, but perhaps there are variables not collected in our data that would be helpful in better predicting log price or price.

**Conclusion**

This analysis indicated several variables which can influence AirBnb price; namely, reviews per month, room type (all categories), and neighbourhood group (north and west region only). This can prove useful, but with a low adjusted R-squared we see that in our data set most AirBnb prices were not made in a linear fashion and are done without much pattern, as is very plausible with AirBnb postings.

# Supplementary material

## Confidence Interval Interpretation

Our confidence intervals tell us a lot about our coefficients and their true effect. We are 95% confident that an AirBnb posting with Entire Room/Apartment as Room Type, in the Central Region, with 0 reviews per month will have a log price between 5.24 and 5.35 per night on average. We are 95% confident that, holding all else constant, for every one unit increase in reviews per month that Log price will decrease between 0.06 and 0.004 on average. We are 95% confident that, holding all else constant, an AirBnb with Private Room as Room Type will have a Log price between 0.97215909 and 0.819078095 less than a listing with Entire Room/Apartment as room type, on average. We are 95% confident that, holding all else constant, an AirBnb with Shared Room as Room Type will have a Log price between 1.74041693 and 1.421084324 less than a listing with Entire Room/Apartment as room type, on average. Our confidence interval for the East Region contains zero demonstrating that there is no significant difference in log price between a listing belonging to the Central Region and a listing belonging to the East Region, on average. Our confidence interval for the North-East Region contains zero demonstrating that there is no significant difference in log price between a listing belonging to the Central Region and a listing belonging to the North Region, on average. We are 95% confident, holding all else constant, that a listing belonging to the North-East region will have a log price between 0.51768837and 0.058873721 less than a listing belonging to the Central Region, on average. We are 95% confident, holding all else constant, that a listing belonging to the West region will have a log price between 0.32839762and 0.046389428 less than a listing belonging to the Central Region, on average.

**Model Summary Output**

```
## Call:
## lm(formula = log(price) ~ reviews_per_month + room_type + neighbourhood_gr
oup,
##     data = sing_random)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.3590 -0.3679 -0.0676  0.2681  3.7974
##
## Coefficients:
##                                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)                         5.29636    0.02756 192.195  < 2e-16
***
## reviews_per_month                  -0.03375    0.01471  -2.295  0.02194
*
## room_typePrivate room              -0.89562    0.03900 -22.962  < 2e-16
***
## room_typeShared room               -1.58075    0.08136 -19.428  < 2e-16
***
## neighbourhood_groupEast Region     -0.09509    0.07236  -1.314  0.18909
## neighbourhood_groupNorth-East Region -0.18525    0.09681  -1.914  0.05597
.
## neighbourhood_groupNorth Region    -0.28828    0.11690  -2.466  0.01383
*
## neighbourhood_groupWest Region     -0.18739    0.07185  -2.608  0.00925
**
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5716 on 992 degrees of freedom
## Multiple R-squared:  0.4644, Adjusted R-squared:  0.4606
## F-statistic: 122.9 on 7 and 992 DF,  p-value: < 2.2e-16
```

**R Code File**

link to our data set https://www.kaggle.com/jojoker/singapore-airbnb

```
library(tidyverse)
```

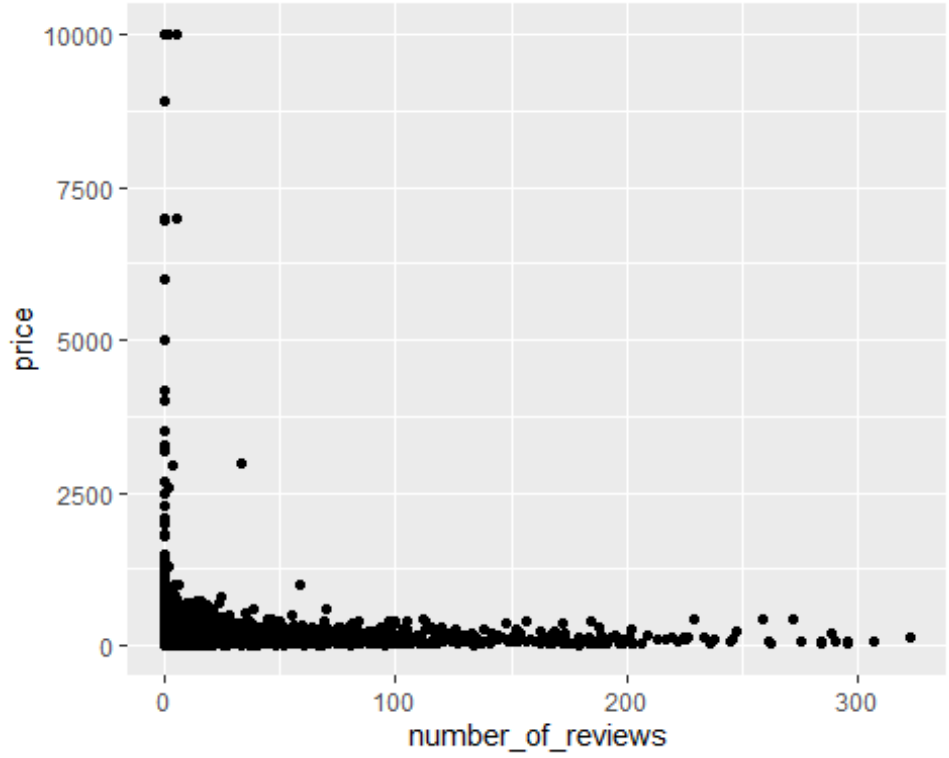
```
## -- Attaching packages ---------------------------------------------------
------------------------------------------------------- tidyverse 1.3.0 -
-

## v ggplot2 3.2.1     v purrr   0.3.3
## v tibble  2.1.3     v dplyr   0.8.3
## v tidyr   1.0.0     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0

## -- Conflicts ------------------------------------------------------------
------------------------------------------------- tidyverse_conflicts() -
-
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(ggplot2)
library(car)
```

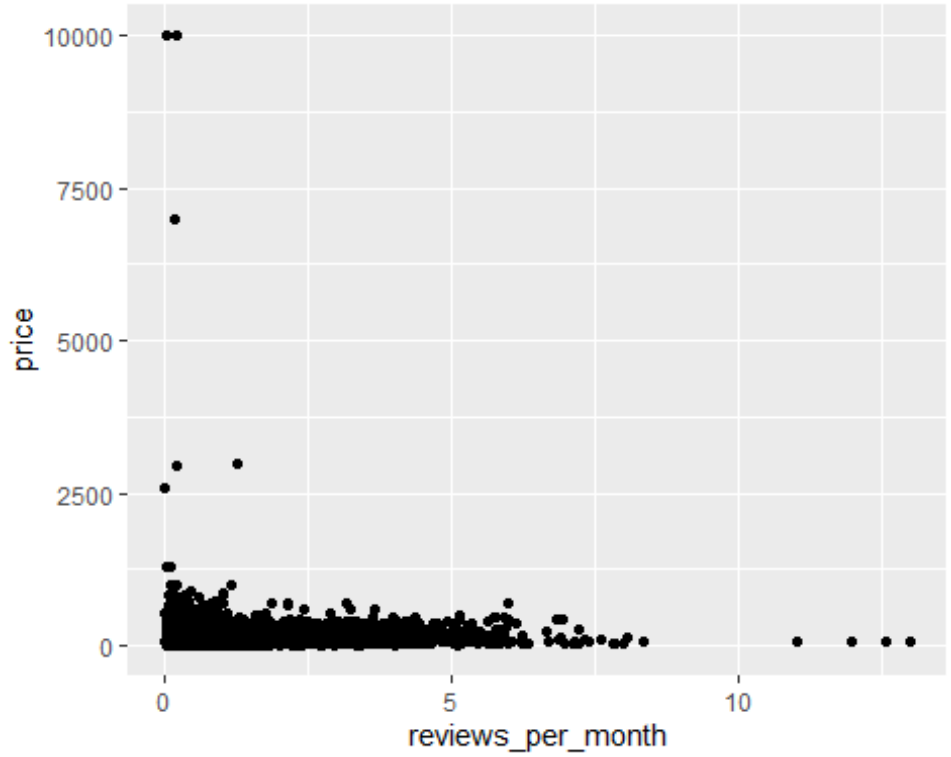
```
## Loading required package: carData

##
## Attaching package: 'car'

## The following object is masked from 'package:dplyr':
##
##     recode

## The following object is masked from 'package:purrr':
##
##     some
```

```
library(ggfortify)

sing_abb_all <- read.csv("C:\\Users\\coleh\\OneDrive\\Documents\\BYU Fall 201
9\\Stat 330\\Final\\singapore-airbnb\\listings.csv")

sing <- sing_abb_all %>%
  select(id, host_id, neighbourhood, neighbourhood_group, room_type, price, n
umber_of_reviews, reviews_per_month)
sing <- sing[-c(3169),]

highest_price <- sing %>%
  group_by(neighbourhood) %>%
  summarise(Avg_price = mean(price)) %>%
  arrange(desc(Avg_price))
```

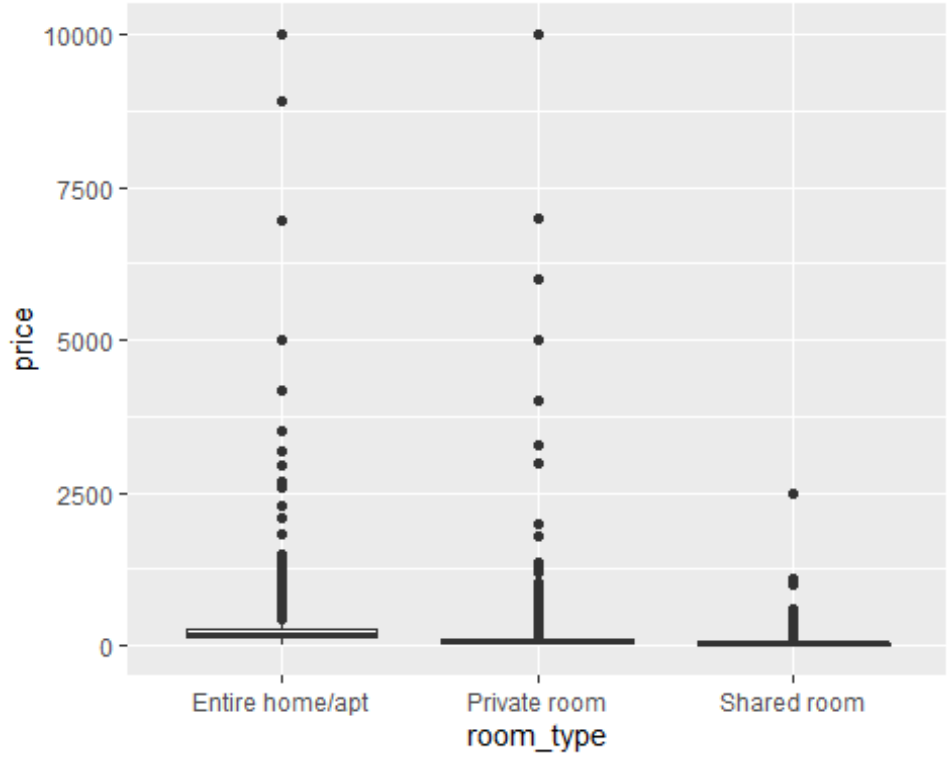
```
price_group <- sing_abb_all %>%
  group_by(neighbourhood_group) %>%
  summarise(Avg_price = mean(price), SD = sd(price)) %>%
  arrange(desc(Avg_price))

percent_group <- sing_abb_all %>%
  count(neighbourhood_group) %>%
  mutate(Percent = round(n/length(sing_abb_all$id), 2))

count_room_type <- sing_abb_all %>%
  count(room_type)

price_reviews_plot <- ggplot(data = sing, mapping = aes(y = price, x = number
_of_reviews)) + geom_point()
price_reviews_plot
```



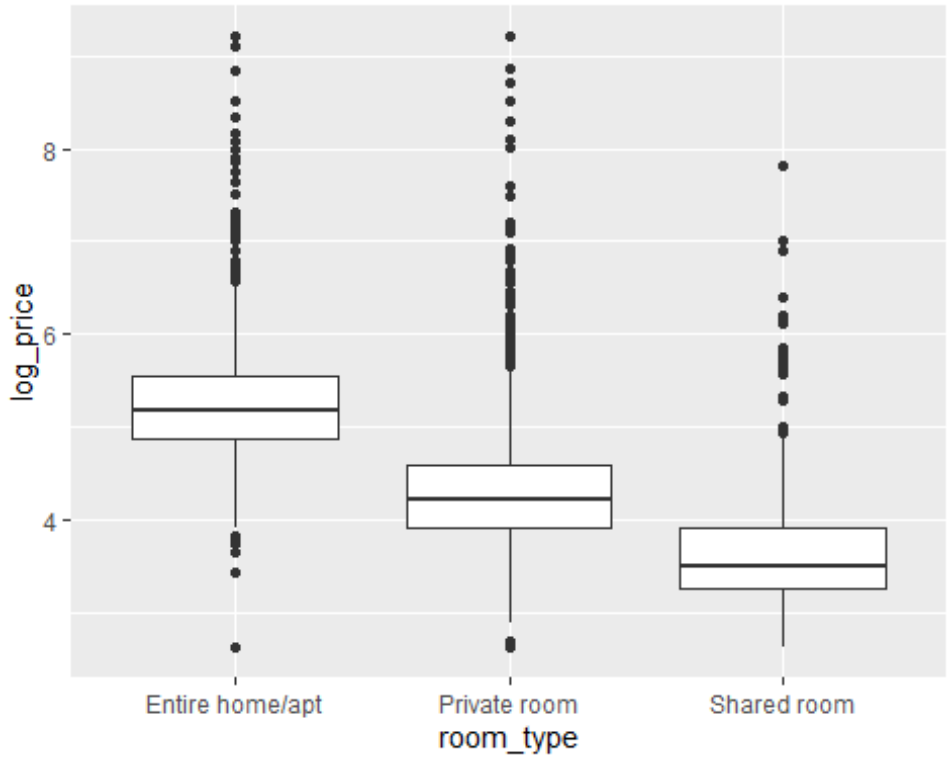

```
price_reviews_per_month_plot <- ggplot(data = sing, mapping = aes(y = price,
x = reviews_per_month)) + geom_point()
price_reviews_per_month_plot

## Warning: Removed 2758 rows containing missing values (geom_point).
```




```
price_room_type_box <- ggplot(data = sing, mapping = aes(y = price, x = room_
type)) + geom_boxplot()
price_room_type_box
```



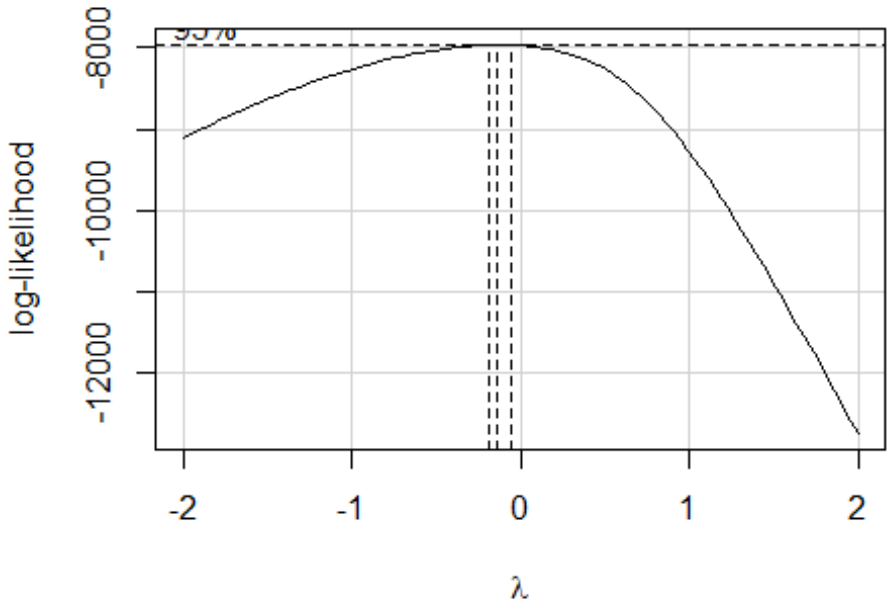
```r
sing$log_price <- log(sing$price)

price_room_type_box <- ggplot(data = sing, mapping = aes(y = log_price, x = r
oom_type)) + geom_boxplot()
price_room_type_box
```




```r
sing$room_private <- ifelse(sing$room_type == "Private room", 1, 0)
sing$room_entire <- ifelse(sing$room_type == "Entire home/apt", 1, 0)
sing$room_shared <- ifelse(sing$room_type == "Shared room", 1, 0)

sing.lm <- lm(data = sing, formula = price ~ number_of_reviews + room_entire
+ room_shared)
summary(sing.lm)

##
## Call:
## lm(formula = price ~ number_of_reviews + room_entire + room_shared,
##     data = sing)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -218.0  -76.0  -41.9    9.7 9883.5
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)       116.9195     6.0051   19.47  < 2e-16 ***
## number_of_reviews  -0.4395     0.1267   -3.47 0.000524 ***
```

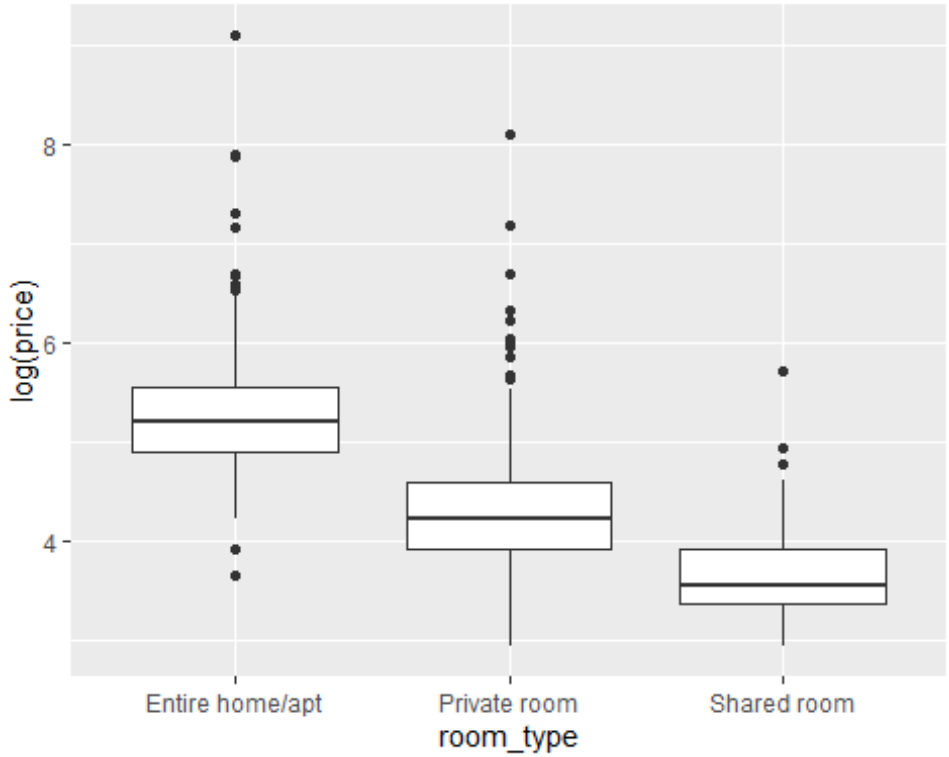
```
## room_entire       115.4884     7.7591   14.88  < 2e-16 ***
## room_shared       -45.7678    17.8062   -2.57 0.010178 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 334.5 on 7902 degrees of freedom
## Multiple R-squared:  0.03374,    Adjusted R-squared:  0.03338
## F-statistic: 91.98 on 3 and 7902 DF,  p-value: < 2.2e-16
```

```
set.seed(42)
random_sample_sz <- 1000
sing_no_na <- sing_abb_all

sing_no_na$reviews_per_month <- ifelse(is.na(sing_no_na$reviews_per_month) ==
TRUE, 0, sing_no_na$reviews_per_month )
sing_no_na$price <- ifelse(is.na(sing_no_na$price) == TRUE, 0, sing_no_na$pri
ce)
sing_no_na <- sing_no_na[-c(3169),]

sing_random <- sing_no_na[sample(nrow(sing_no_na), random_sample_sz),]

bc <-boxCox(sing_random$price ~ sing_random$reviews_per_month)
```

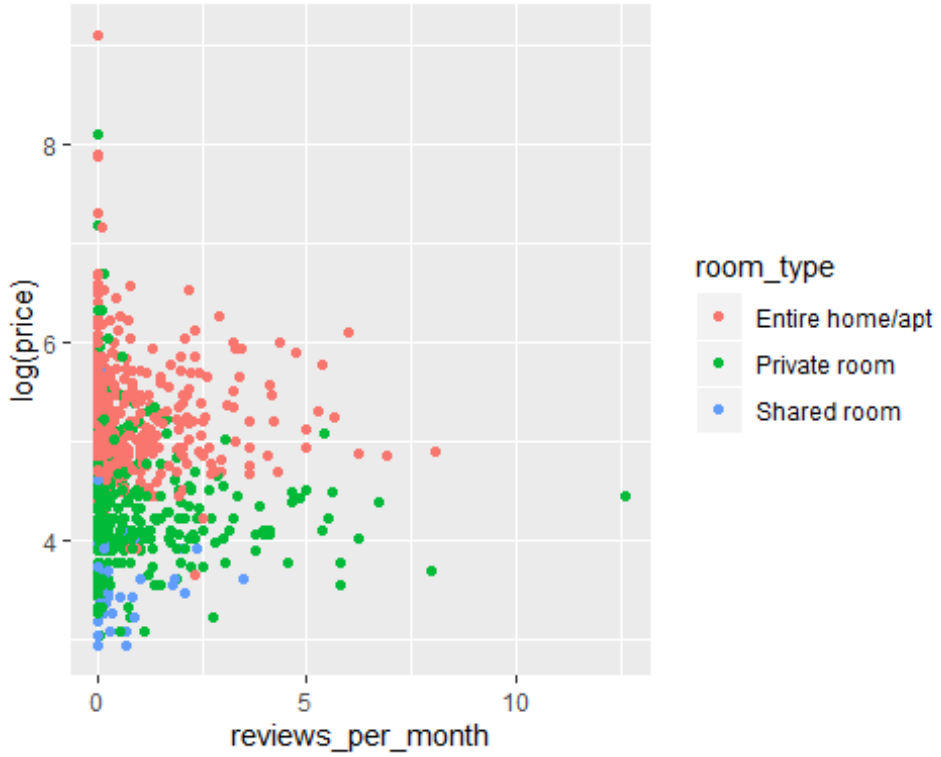



```
bc$x[which.max(bc$y)]

## [1] -0.1414141
```

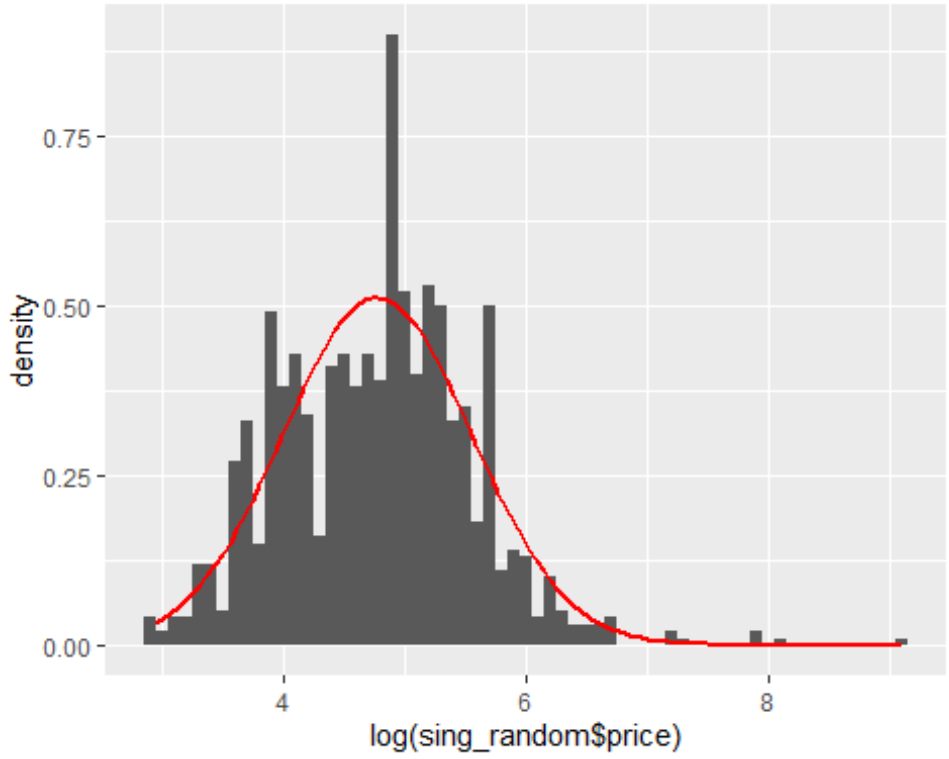
```
random_sample_boxplot <- ggplot(data = sing_random, mapping = aes(y = log(pri
ce), x = room_type)) + geom_boxplot()
random_sample_boxplot
```




```
r_s_plot_color <- ggplot(data = sing_random, mapping = aes(y = log(price), x
= reviews_per_month, color = room_type)) + geom_point()
r_s_plot_color
```




```
#frequency plot with log y transformation
freq_distribution <- ggplot(data = sing_random, mapping = aes(x = log(sing_ra
ndom$price))) +
  geom_histogram(mapping = aes(y = ..density..), binwidth = .1) +
  stat_function(fun = dnorm, color = "red", size = 1,
                args = list(mean = mean(log(sing_random$price)),
                            sd = sd(log(sing_random$price))))
freq_distribution
```



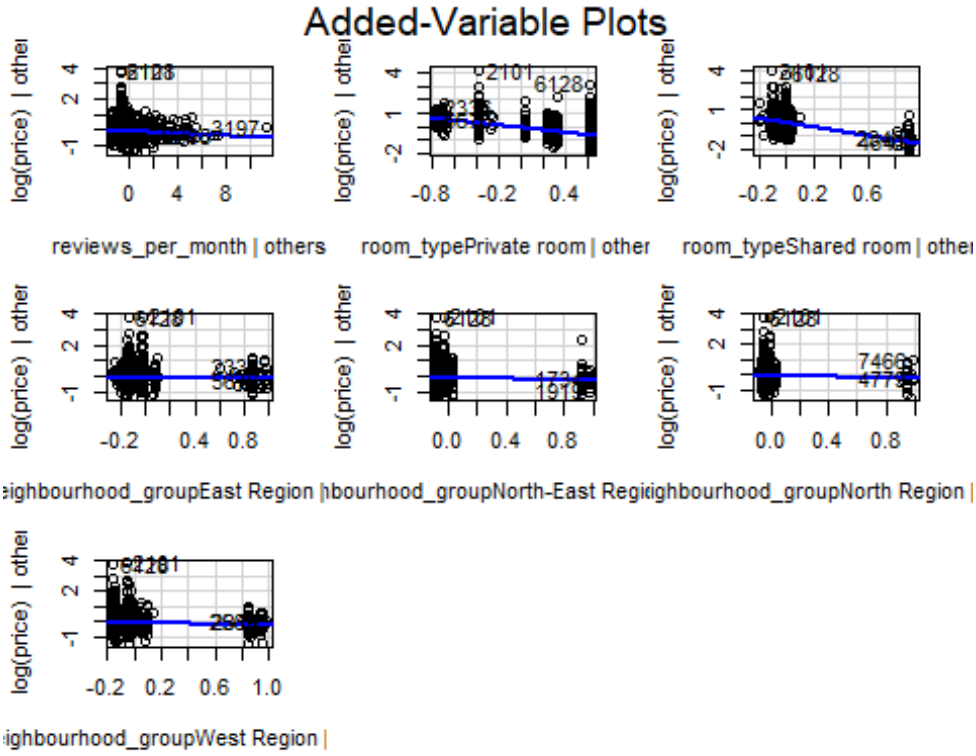

```
sing_random_log_y.lm <- lm(data = sing_random, formula = log(price) ~ reviews
_per_month + room_type + neighbourhood_group )
summary(sing_random_log_y.lm)

##
## Call:
## lm(formula = log(price) ~ reviews_per_month + room_type + neighbourhood_gr
oup,
##     data = sing_random)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.3590 -0.3679 -0.0676  0.2681  3.7974
##
## Coefficients:
##                                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)                               5.29636    0.02756 192.195  < 2e-16
***
## reviews_per_month                        -0.03375    0.01471  -2.295  0.02194
*
## room_typePrivate room                    -0.89562    0.03900 -22.962  < 2e-16
***
## room_typeShared room                     -1.58075    0.08136 -19.428  < 2e-16
***
## neighbourhood_groupEast Region           -0.09509    0.07236  -1.314  0.18909
## neighbourhood_groupNorth-East Region     -0.18525    0.09681  -1.914  0.05597
.
```

```
## neighbourhood_groupNorth Region      -0.28828    0.11690  -2.466  0.01383
*
## neighbourhood_groupWest Region       -0.18739    0.07185  -2.608  0.00925
**
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5716 on 992 degrees of freedom
## Multiple R-squared:  0.4644, Adjusted R-squared:  0.4606
## F-statistic: 122.9 on 7 and 992 DF,  p-value: < 2.2e-16
```

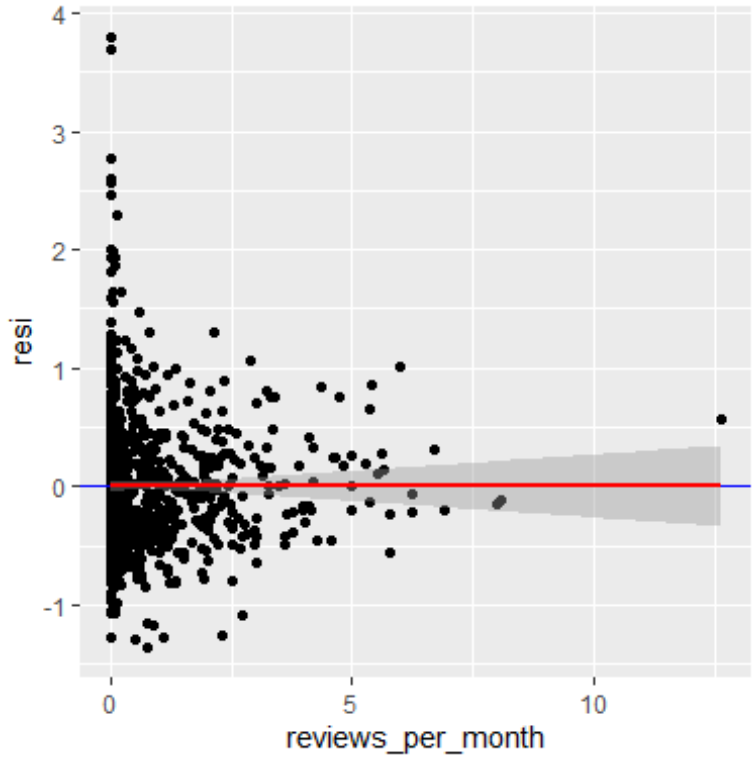
```
sing_random.lm <- lm(data = sing_random, formula = price ~ reviews_per_month
+ room_type + neighbourhood_group )
summary(sing_random_log_y.lm)
```

```
##
## Call:
## lm(formula = log(price) ~ reviews_per_month + room_type + neighbourhood_gr
oup,
##     data = sing_random)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.3590 -0.3679 -0.0676  0.2681  3.7974
##
## Coefficients:
##                                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)                        5.29636    0.02756 192.195  < 2e-16
***
## reviews_per_month                 -0.03375    0.01471  -2.295  0.02194
*
## room_typePrivate room             -0.89562    0.03900 -22.962  < 2e-16
***
## room_typeShared room              -1.58075    0.08136 -19.428  < 2e-16
***
## neighbourhood_groupEast Region       -0.09509    0.07236  -1.314  0.18909
## neighbourhood_groupNorth-East Region -0.18525    0.09681  -1.914  0.05597
.
## neighbourhood_groupNorth Region      -0.28828    0.11690  -2.466  0.01383
*
## neighbourhood_groupWest Region       -0.18739    0.07185  -2.608  0.00925
**
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5716 on 992 degrees of freedom
## Multiple R-squared:  0.4644, Adjusted R-squared:  0.4606
## F-statistic: 122.9 on 7 and 992 DF,  p-value: < 2.2e-16
```

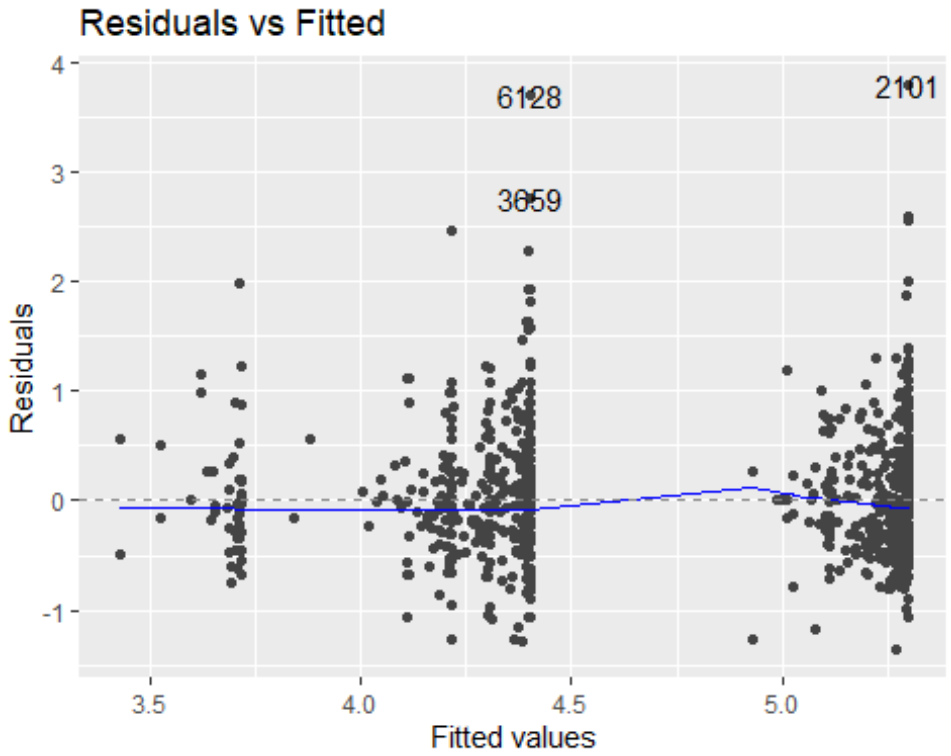
```r
sing_random$resi <- sing_random_log_y.lm$residuals
sing_random$predicted <- predict(sing_random_log_y.lm)


#check x vs y is linear
#partial regression plots
avPlots(sing_random_log_y.lm)
```



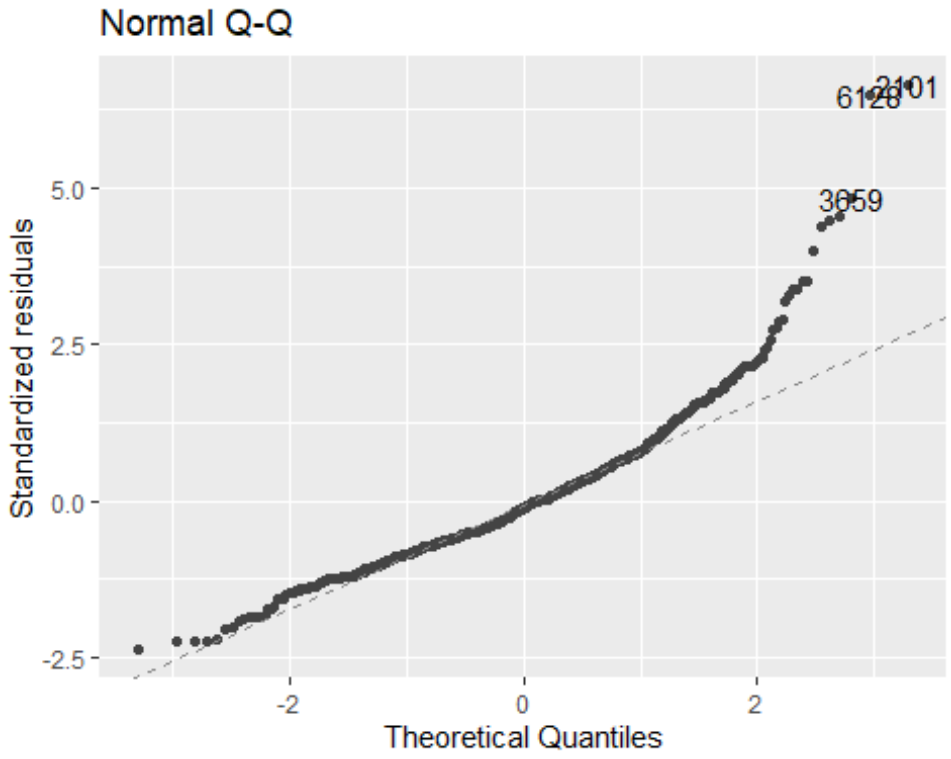

```r
#residuals vs predictor plots
b_rpm <- ggplot(sing_random, aes(x = reviews_per_month, y = resi)) +
  geom_point() +
  geom_hline(yintercept = 0, color = 'blue') +
  theme(aspect.ratio = 1) +
  geom_smooth(method = 'lm', color = 'red')

b_rpm
```



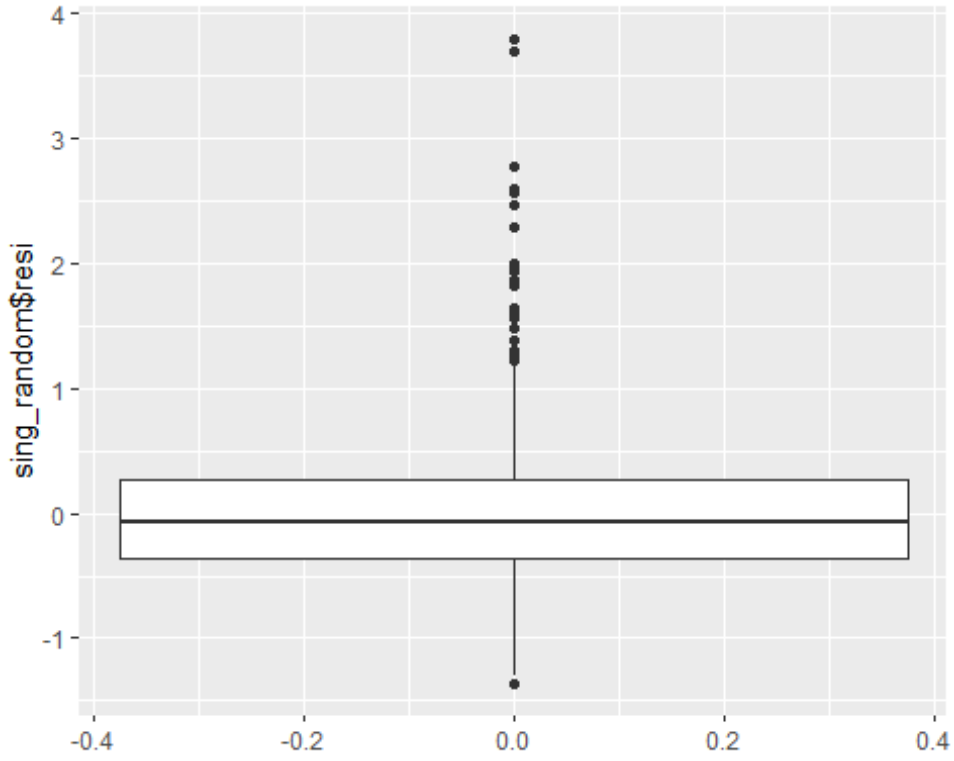

```r
#residuals vs fit values plot
autoplot(sing_random_log_y.lm, which = 1, ncol = 1, nrow = 1)
```



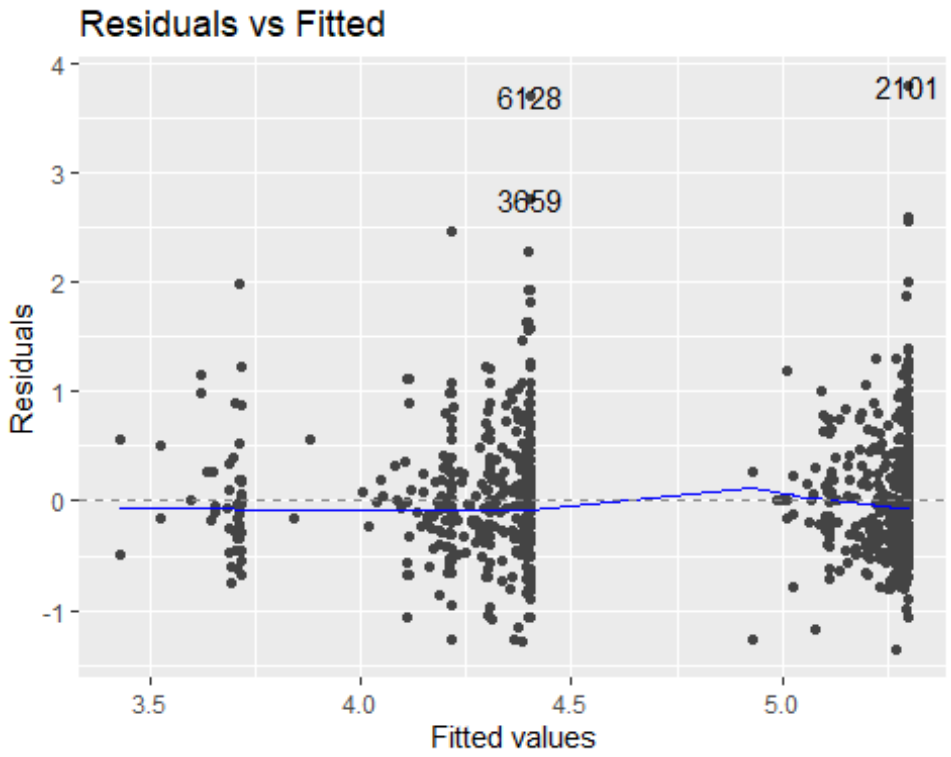
```
#The residuals are normally distributed and centered at zero

# normal probability plot
autoplot(sing_random_log_y.lm, which = 2, ncol = 1, nrow = 1)
```



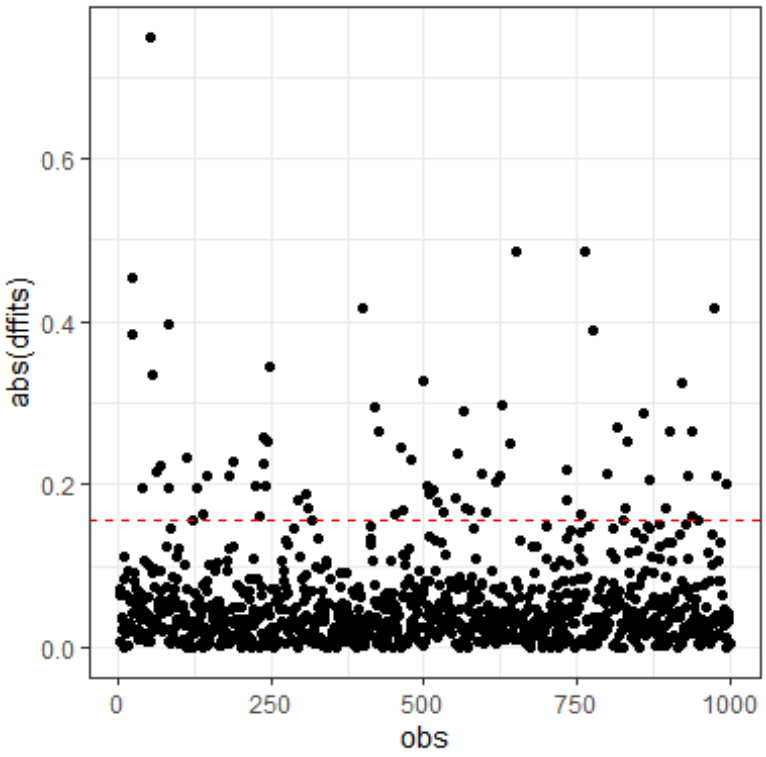

```
# shapiro wilk test
#shapiro.test(sing_random_log_y.lm)

#boxplot
sing_log_y_boxplot <- ggplot(sing_random_log_y.lm, mapping = aes(y = sing_ran
dom$resi)) + geom_boxplot()
sing_log_y_boxplot
```



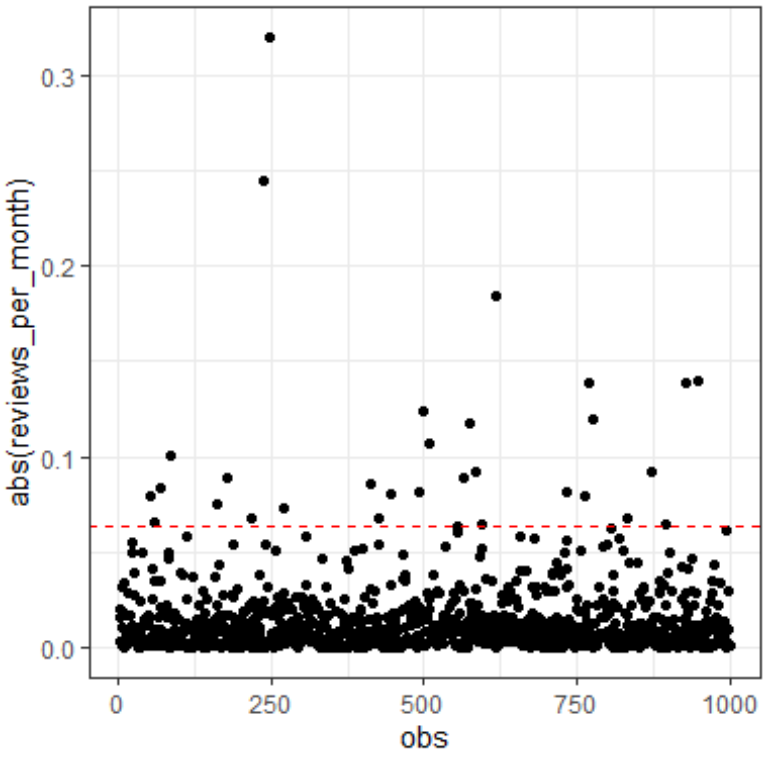

```
#The residuals are homoscedastic

#residuals vs fit values plot
autoplot(sing_random_log_y.lm, which = 1, ncol = 1, nrow = 1)
```



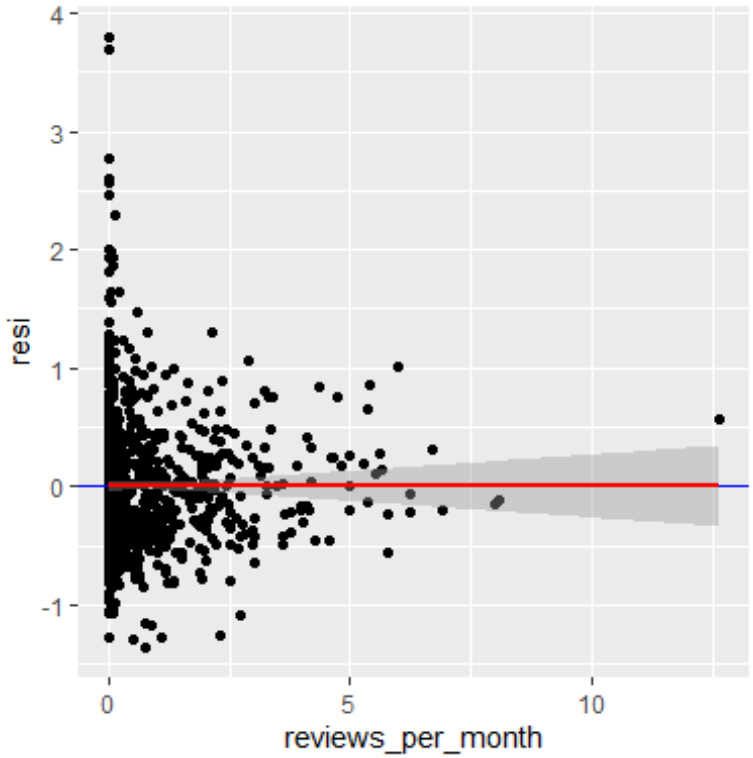

```
#brown-forsythe test
grp <- as.factor(c(rep("lower", floor(dim(sing_random)[1] / 2)),
                   rep("upper", ceiling(dim(sing_random)[1] / 2))))
leveneTest(sing_random$resi ~ grp, center = median)

## Levene's Test for Homogeneity of Variance (center = median)
##        Df F value Pr(>F)
## group   1  1.1398  0.286
##       998

#The model describes all observations - there are no influential points (use
the DFBETAS, DFFITS, partial regression plots

# DFFITS
sing_random.dffits <- data.frame("dffits" = dffits(sing_random_log_y.lm))
sing_random.dffits$obs <- 1:length(sing_random$price)

ggplot(data = sing_random.dffits) +
  geom_point(mapping = aes(x = obs, y = abs(dffits))) +
  # geom_hline(mapping = aes(yintercept = 1),
  #            color = "red", linetype = "dashed") +  # for n <= 30
  geom_hline(mapping = aes(yintercept = 2 * sqrt(6 / length(obs))),
             color = "red", linetype = "dashed") +  # for n > 30
  theme_bw() +
  theme(aspect.ratio = 1)
```




```
sing_random.dffits[abs(sing_random.dffits$dffits) > 1, ]

## [1] dffits obs
## <0 rows> (or 0-length row.names)

#DFFBETAS

sing_random.dfbetas <- as.data.frame(dfbetas(sing_random_log_y.lm))
sing_random.dfbetas$obs <- 1:length(sing_random$price)
# age
ggplot(data = sing_random.dfbetas) +
 geom_point(mapping = aes(x = obs, y = abs(reviews_per_month))) +
   geom_hline(mapping = aes(yintercept = 2 / sqrt(length(obs))),
              color = "red", linetype = "dashed") +  # for n > 30
   theme_bw() +
   theme(aspect.ratio = 1)
```




```
reviews.extreme.dfbetas <- sing_random.dfbetas[abs(sing_random.dfbetas$review
s_per_month) > .25, ]
reviews.extreme.dfbetas[order(reviews.extreme.dfbetas$reviews_per_month), ]

##      (Intercept) reviews_per_month room_typePrivate room room_typeShared r
oom
## 3197  -0.1292251          0.320289            0.01790715           0.01603
595
##      neighbourhood_groupEast Region neighbourhood_groupNorth-East Region
## 3197                      0.08304192                           -0.001366735
##      neighbourhood_groupNorth Region neighbourhood_groupWest Region obs
## 3197                      -0.01273612                     0.01881182 247

#partial pregression
b_rpm
```




```
#no mulitcollinlearnity

vif(sing_random_log_y.lm)
```

```
##                         GVIF Df GVIF^(1/(2*Df))
## reviews_per_month   1.021333  1        1.010610
## room_type           1.100557  2        1.024243
## neighbourhood_group 1.118002  4        1.014041
```

```
confint(sing_random_log_y.lm, level = .95)
```

```
##                                              2.5 %       97.5 %
## (Intercept)                               5.24228532  5.350439681
## reviews_per_month                        -0.06260957 -0.004892606
## room_typePrivate room                    -0.97215909 -0.819078095
## room_typeShared room                     -1.74041693 -1.421084324
## neighbourhood_groupEast Region           -0.23707729  0.046898853
## neighbourhood_groupNorth-East Region     -0.37522887  0.004725880
## neighbourhood_groupNorth Region          -0.51768837 -0.058873721
## neighbourhood_groupWest Region           -0.32839762 -0.046389428
```

```
#adjusted r squared
summary(sing_random_log_y.lm)
```

```
##
## Call:
## lm(formula = log(price) ~ reviews_per_month + room_type +
neighbourhood_group,
```

```
##     data = sing_random)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.3590 -0.3679 -0.0676  0.2681  3.7974
##
## Coefficients:
##                                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)                         5.29636    0.02756 192.195  < 2e-16
***
## reviews_per_month                  -0.03375    0.01471  -2.295  0.02194
*
## room_typePrivate room              -0.89562    0.03900 -22.962  < 2e-16
***
## room_typeShared room               -1.58075    0.08136 -19.428  < 2e-16
***
## neighbourhood_groupEast Region     -0.09509    0.07236  -1.314  0.18909
## neighbourhood_groupNorth-East Region -0.18525    0.09681  -1.914  0.05597
.
## neighbourhood_groupNorth Region    -0.28828    0.11690  -2.466  0.01383
*
## neighbourhood_groupWest Region     -0.18739    0.07185  -2.608  0.00925
**
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5716 on 992 degrees of freedom
## Multiple R-squared:  0.4644, Adjusted R-squared:  0.4606
## F-statistic: 122.9 on 7 and 992 DF,  p-value: < 2.2e-16
```