Data Analysis Project

Spatial Modelling of Short-Term Rental Prices using Nonparametric Models

Zhi Rong Tan

INTRODUCTION

In this project, I obtained the complete list of Chicago's Airbnb's rental listings in October 2018, with plans to predict the daily rental prices based on the following variables – *type of room / apartment*, *privacy*, *host reputation*, and *location*. The parametric model used include both linear and polynomial regression. Thereafter, I attempted to use a nonparametric method – local linear regression, to predict Airbnb's rental prices using latitudes and longitudes, after controlling other variables by regressing on the residuals of the observations from the parametric model. In addition, another nonparametric model employed is the additive model.

The project has 2 aims – first: to understand how short-term rental prices (measured solely by Airbnb prices) are dependent on location when controlled for other variables, such as host quality, and apartment type; second: in serving as an informal guide to help homeowners determine how much they can rent their apartment or room for, if it is listed on Airbnb.

As an application of the project, I will try to predict how much are the following apartments worth if they are listed in Airbnb:

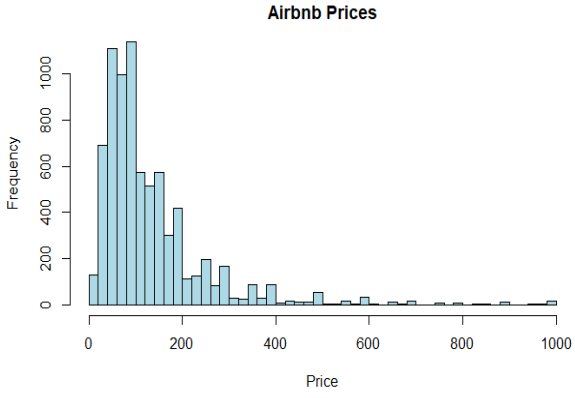
1. 4-Room Apartment in Hyde Park (My apartment)
2. 1-Room Studio Apartment in Downtown Loop (Friend 1's apartment)
3. 5-Room Multi-Level House in Gold Coast (Friend 2's family house)

1. DATA PROCESSING

The Airbnb data is obtained from the website http://insideairbnb.com/get-the-data.html, kindly scraped by the owner from the Airbnb server. It contains all Chicago Airbnb listings in October 2018, with over 50 variables, from details about images, location and host, to amenities.

After removing the unimportant information, I am left with the following variables:

Price per day (Response variable): This price excludes cleaning, taxes, and ignores long-term stay discounts. To remove outliers, I have deleted observations with price = $0, or price > $1000, the former which could be due to listing errors, and the latter, host ignorance about market norms.



I also created a new response variable, *log(price)* – observe from the histogram plots that log(price) has some normality, and may be the preferred response form that we should use in our model.

Residential Size: The type of residence is also an important factor in determining rental prices. This group of variables is likely to have the largest correlation with the response variables.

The variable, **room**, is the total number of bedrooms that the listing has, and this number excludes the living room (if it exists). I included a new variable, **room.den**, which is the people per room (determined by maximum number of people listing can accommodate divided by number of rooms). This variable measures the level of discomfort the guest may face – the higher the density, the more people will have to share a room.

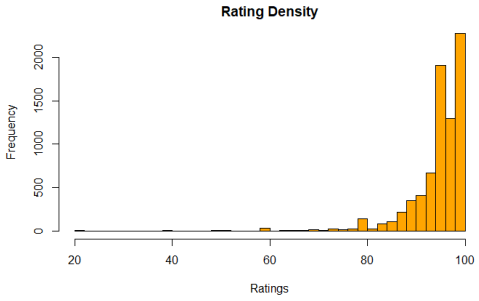
People measures the maximum number that the listing is able to accommodate. Notice that this is an interaction term between *room* and *room.den*. I understand that number of bedrooms and accommodation size have some correlation, so this interaction term is to account for this. It is likely that we only require 1-2 variables out of these 3 in our model, which we will determine later by stepwise regression in our parametric model.

Privacy & Living Space: An original variable is *Type of Apartment* = {shared, private room, entire apartment}. I transformed this variable into 2 categorical variables. The first is **no.private**. If *Type of Apartment* = {shared}, then no.private = 1, otherwise =0, and is a proxy for whether the listing room provides for some level of privacy, which is essential in determining the price. In addition, if *Type of Apartment* = {entire apartment}, then **living** = 1, otherwise =0. Living measures whether the listing provides both privacy, and additional facilities of a whole apartment (e.g. additional space, kitchen).

Host: A top host is likely to be able to warrant higher prices than a new and unknown host. Thus, the binary categorical variable **superhost** measures this. A superhost is determined by Airbnb, and must fulfil certain criteria, such as overall good listings, openness, etc.

In addition, the larger the number of reviews, **review.num**, the more experienced the host is, the more justifications for them to charge higher prices. I transform *review.num* into an ordinal variable **review.size**. This simplifies the model, and actual number of reviews do not matter as much as rough impression based on bins they are placed into (i.e. I treat a listing with 150 reviews and 250 reviews equally, likewise 0 reviews and 3 reviews).

review.num	<= 5	6-20	21-50	51-100	>100
review.size	0	1	2	3	4



Ratings: The higher the ratings are, the more the host may be able to charge. Since most ratings are above 60, I decided to normalize the variable by deducting 60 from the original rating, to form **ratings**. If ratings < 0, then set = 0. The original variable is renamed **ratings.old**. For missing data (i.e. listings with no prior guests), *I imputed it with the mean*.

Location: The location of the listing is given by 2 variables, **longitude** and **latitude**, which are both likely nonparametric in relationship to the response variable price. These 2 variables are selected compared to other location variables such as neighborhoods because the latter is too generic, and not all listing records them down.

To understand the density of the listings relative to their location, I did a rough density estimation (where bandwidth is chosen without cross validation) just to gauge the approximate distribution. From the plot on the next page, observe that the majority of listings are clustered in 3 main areas – Downtown, Wicker Park, and Gold Coast / Lincoln Park area. On the South Side, they are mainly clustered in Hype Park.

Summary of Data:

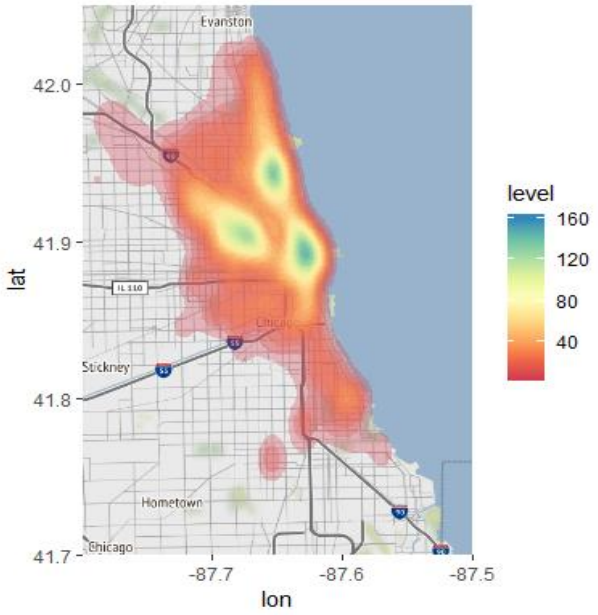
Our final dataset contains 2 potential response variables (price, price.log), with 11 covariates grouped into the following categories – apartment size (3), privacy (2), host reputation (2), review quality (2), and location (2).

Total observations: n = **7624** listings.

Correlation Matrix: *(To save space, the matrix is located at the end of the document)*

Price is most correlated with people [0.55], rooms [0.51], living [0.39] and no.private [-0.14]. Other variables like *longitude*, room density and *review size* have a correlation of around 0.13, with the smallest being latitude and review scores at 0.06.

Also observe that *people* is indeed correlated highly with *room* and *living*. Moreover, review size, superhost and ratings are also another cluster of variables that are somewhat correlated with each other.



1 ***Density Estimation of Airbnb Listings***

2. PARAMETRIC REGRESSION MODEL

I will approach the accuracy of my model by splitting it into a training and test set with a 3:1 ratio. Hence, my training set has 5718 observations, while my test set has 1906 observations. The model will be trained on the training set, and evaluated on the test set using Root Mean Square Error (RMSE), where the optimal model shall have the smallest error rate.

Model 1: Price ~ ALL covariates

In the first parametric model, we shall regress price on all variables (including longitude and latitude), in a linear regression model. Given the nature of the parameters, I only include 1 interaction term, which is people (room.den x rooms), and the other variables are assumed to have a linear relationship with price.

I shall start with the full model:

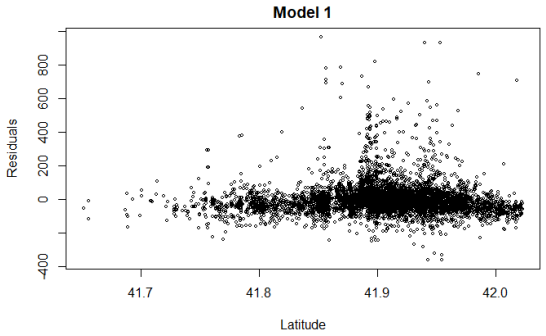
- *lm (price ~ latitude + longitude + people + rooms + review.num + ratings.old + superhost + no.private + living + room.den + review.size + ratings, data=train.set)*

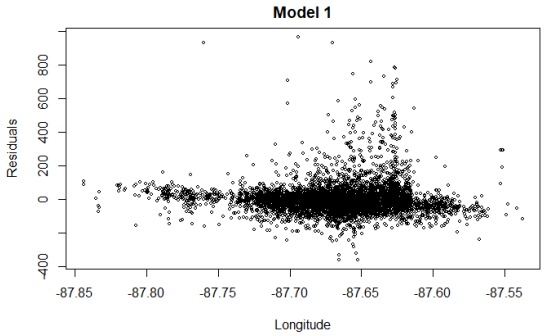
To evaluate which variables to use, I used backward stepwise regression, where i fix at 5% significance level to determine which variables to remove. I also ensured that variables that are highly correlated (e.g. *ratings* and *ratings.old*) are not included together in the final model. Based on the stepwise selection process, I obtained:

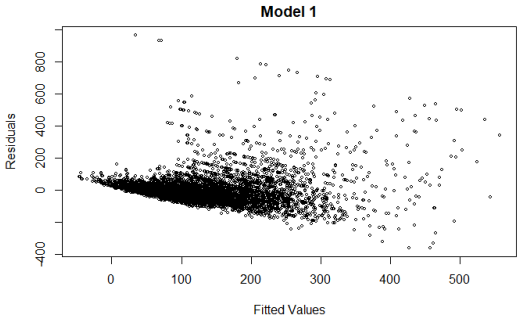
```
Coefficients:
                Estimate    Std. Error   t value          Pr(>|t|)
(Intercept) 49538.4518113 2778.3318073  17.83029       < 2.22e-16 ***
latitude      280.5272453   25.3606348  11.06152       < 2.22e-16 ***
longitude     699.5046282   35.4726443  19.71955       < 2.22e-16 ***
people         21.8943771    1.3835317  15.82499       < 2.22e-16 ***
rooms          12.8679163    3.1274869   4.11446 0.00003935515688 ***
ratings.old     1.0196668    0.1982752   5.14268 0.00000027987765 ***
no.private    -18.6817804    6.4317933  -2.90460        0.0036914 **
living         26.7021607    3.0517953   8.74966       < 2.22e-16 ***
room.den      -13.5486040    2.1214255  -6.38656 0.00000000018307 ***
review.size   -14.3216111    0.9677502 -14.79887       < 2.22e-16 ***
---
```

lm (price ~ latitude + longitude + people + rooms + ratings.old + no.private + living + room.den + review.size, data=train.set)

RMSE = **58.139**







Observe from the 3 plots that this model does not satisfy theour linear model assumptions:

(i) Lack of constant variance
(ii) Data points are not equally distributed around the 0 line
(iii) Fitted-residual plot has a weird shape to it

Hence, due to the non-linear nature of latitude and longitude variable, we want to investigate the possibility of using a polynomial model on these 2 variables.

Model 2: Price ~ ALL covariates (Longitude + Latitude Polynomial)

I shall include the term Longitude^2 and Latitude^2 into the regression model. I have experimented with the power 3, but it did not add any new predictive value. Similar to the previous model, we first fit a model with all the covariates, with longitude and latitude up to a polynomial power of 2. From stepwise regression, we choose the final model, presented below. Observe that in the final model, only Latitude^2 is included, not Longitude^2.

```
lm(formula = price ~ latitude + I(latitude^2) + longitude + people +
    rooms + ratings.old + living + room.den + review.size, data = train.set)

Residuals:
       Min         1Q     Median         3Q        Max
-355.73385  -43.87291  -11.72314   23.03323  956.93581

Coefficients:
                    Estimate       Std. Error   t value   Pr(>|t|)
(Intercept)   -6630708.5488344   455342.0914760 -14.56204 < 2.22e-16 ***
latitude        319406.1814678    21751.5587148  14.68429 < 2.22e-16 ***
I(latitude^2)    -3809.3022369      259.6479888 -14.67103 < 2.22e-16 ***
longitude          738.9370820       34.9122323  21.16556 < 2.22e-16 ***
people              20.8215011        1.3609086  15.29971 < 2.22e-16 ***
rooms               15.7553223        3.0773066   5.11984 3.1582e-07 ***
ratings.old          1.0159900        0.1947623   5.21656 1.8871e-07 ***
living              22.8716495        2.9621091   7.72141 1.3498e-14 ***
room.den           -12.6101912        2.0830819  -6.05362 1.5064e-09 ***
review.size        -14.4047936        0.9495868 -15.16954 < 2.22e-16 ***
```

Note that all the variables have extremely small p-values in the t-test. Hence, they are all significant.

By testing on the test set,

RMSE = **56.562**

From this, we determine that the polynomial model on latitude is slightly better than model 1, and will proceed with it.

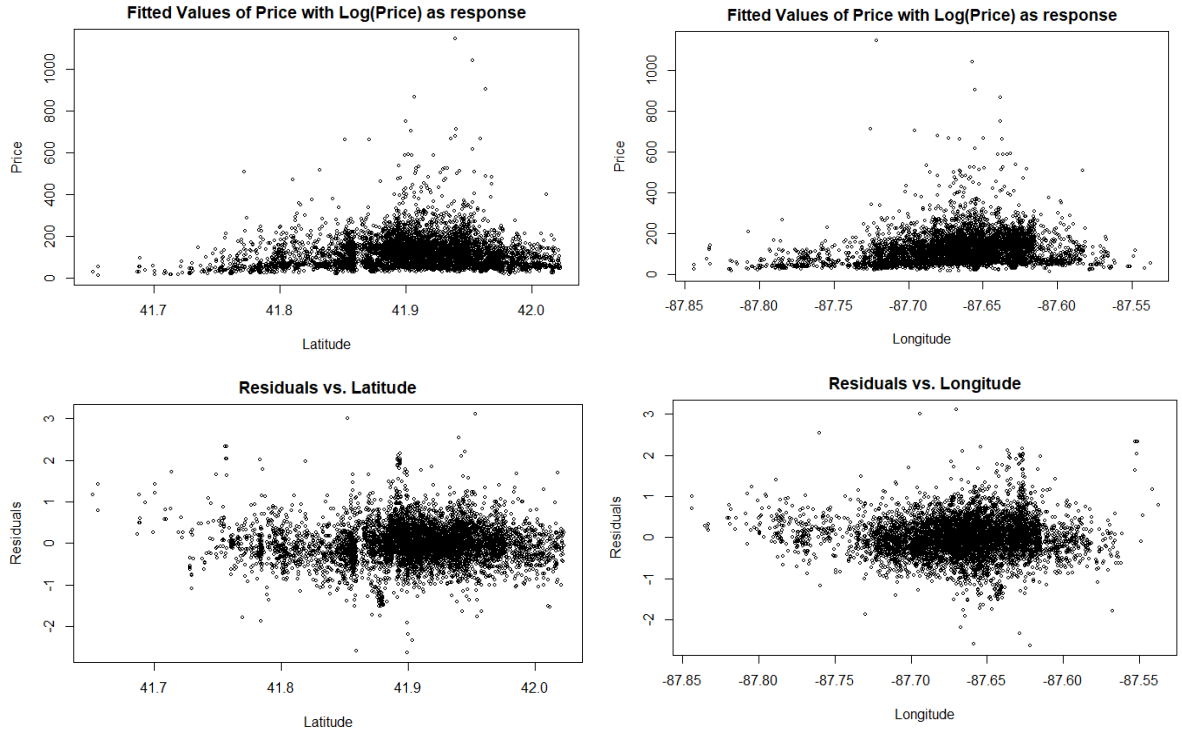
Model 3: Log(Price) ~ ALL covariates (Latitude Polynomial)

We will now test if we gain a better prediction by using Log(Price) as our response. The following steps are similar to what we have done in the previous 2 models. After stepwise selection, we obtained the final model shown below. With a lower RMSE and the Adjusted R^2 value improving by 0.2, we determine that Model 3 is the optimal model in parametric regression.

```
Coefficients:
                 Estimate       Std. Error   t value   Pr(>|t|)
(Intercept)   -47921.186919924  2371.220503615 -20.20950 < 2.22e-16 ***
latitude        2306.767683529   113.273542939  20.36458 < 2.22e-16 ***
I(latitude^2)    -27.510739129     1.352127457 -20.34626 < 2.22e-16 ***
longitude          4.916884997     0.180862226  27.18580 < 2.22e-16 ***
people             0.067404503     0.004030548  16.72341 < 2.22e-16 ***
rooms              0.144280155     0.009924082  14.53839 < 2.22e-16 ***
ratings.old        0.009103901     0.001008399   9.02807 < 2.22e-16 ***
no.private        -0.411319294     0.032888442 -12.50650 < 2.22e-16 ***
living             0.461256709     0.015609231  29.55025 < 2.22e-16 ***
review.size       -0.084110926     0.004912843 -17.12062 < 2.22e-16 ***
```

lm(price.log ~ latitude + I(latitude^2) + longitude + people + rooms + ratings.old + no.private + living + review.size, data=train.set)

RMSE = **51.559**



A linear model makes certain assumptions such as the following:

- *Errors are normal and independently distributed*
- *No or little multicollinearity*
- *Constant variance*

From the plots above (and by choices taken in stepwise selection), we have shown that the assumptions are not seriously contradicted.

<u>Proceeding to a Nonparametric Model</u>

It is clear that besides longitude and latitude, most of the variables are linear and/or categorical. Hence, I will proceed to find the residuals of a parametric linear regression model based on the linear and categorical variables (without location), before using the residuals as the response to determine a nonparametric model with longitude and latitude as the variables. This implies that the impact of location on price cannot be captured in the parametric model, but can be observed in the residuals.

For this method to work, we assume that location (latitude and longitude) is independent of other variables in determining prices. Hence, the coefficient of any interaction term with location is zero. The assumption is valid considering that the daily price range is too small for any interaction effect to be meaningfully reflected, especially for short-term rental prices. Moreover, the effect of location and other variables on price is additive linearly.

We will employ the 2 models above using either price or log(price) as the response, and simply omit the location variables in the parametric model now. We continue to consider both price and log(price) to understand which works better subsequently in a nonparametric model. To ensure that the omission of location will not cause any new variables to be significant again, the stepwise selection is performed again, and indeed, we arrive at the same variables selected (but with neither longitude nor latitude). For ease of reading, the presentation is omitted, but the RMSE for the model using price as response is now **60.44**, and model using log(price) is **59.74**. In the subsequent part, all residuals mentioned are based on residuals of these 2 models.

3. NONPARAMETRIC REGRESSION MODEL

To predict prices based solely on location, with other variables fixed, we shall proceed with Local Linear Regression, an extension of the Nadaraya–Watson Estimator, since it is the simplest and easiest method to interpret for our dataset. First recall that for a multivariate regression, the kernel smoother will be of the form

$$\hat{r}_n(x) = \sum_{i=1}^n l_i(x) Y_i \qquad l_i(x) = \frac{K^*(X_i)}{\sum_{j=1}^n K^*(X_j)}$$

The kernel will be *Bivariate Gaussian*, where we assume that the 2 variables (longitude and latitude) have zero correlation:

$$K^*(X_i) = K\left(\frac{x_1 - X_{i1}}{h}, \frac{x_2 - X_{i2}}{h}\right)$$

In the bandwidth selection, the LOOCV score is given by

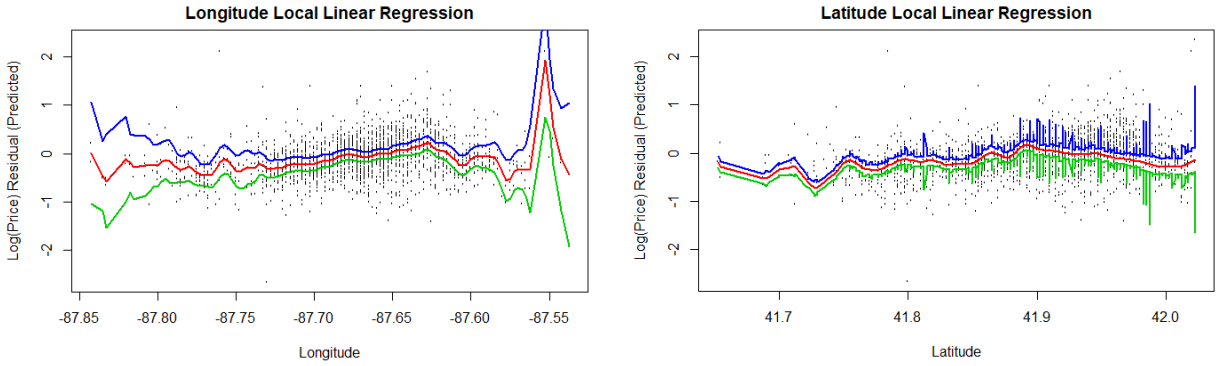
$$\hat{R}(h) = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{Y_i - \hat{r_n}(x_i)}{1 - L_{ii}}\right)^2$$

One-Dimensional Model

We first consider a one-dimensional **local linear regression** on latitude and longitude separately, regressing the residuals of Log(price) (from the parametric linear model) on either latitude or longitude. To proceed, we set up a grid to find the optimal bandwidth h. To save space, we will not publish the grid search process here.

In one-dimension, it is easy to visualize the nonparametric regression fit and confidence interval, plotted below for both latitude and longitude. The red line is the regression fit, and the blue and green lines are the upper and lower confidence interval respectively.

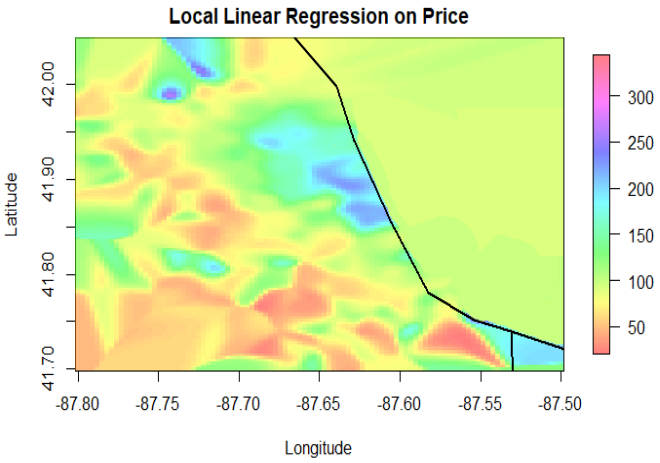
Observe that as the regression prediction approaches the boundaries where there are significantly less data points, the confidence intervals then increases significantly.



However, 1-dimensional local linear regression overgeneralizes the effect of one single variable of the location, since the effect of longitude and latitude are often not independent spatially. Having a 2-

dimensional regression will be more helpful and easier to interpret, allowing longitude and latitude to interact with each other in visualizing specific locations.

Because there are issues using the *locfit package* in R for 2-dimensional local linear/polynomial regression, I wrote my own functions instead of relying on the package, which can be found in Appendix B. To ensure that the functions work, I regressed price on location (longitude and latitude) *without controlling for other variables*, as an exploratory step. The results of the 2-dimensional local linear regression are presented here.

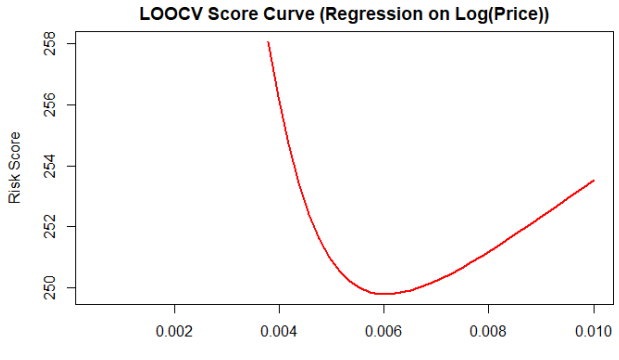


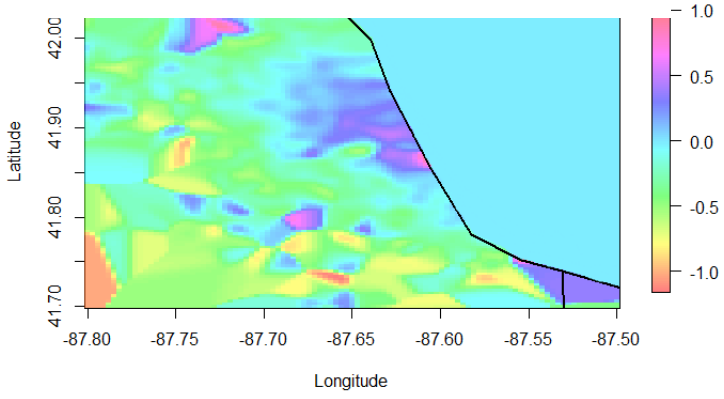
Since this map makes intuitive sense (the downtown Airbnb prices are more expensive, and Chicago South Side is less costly than average), the regression function should be accurate.

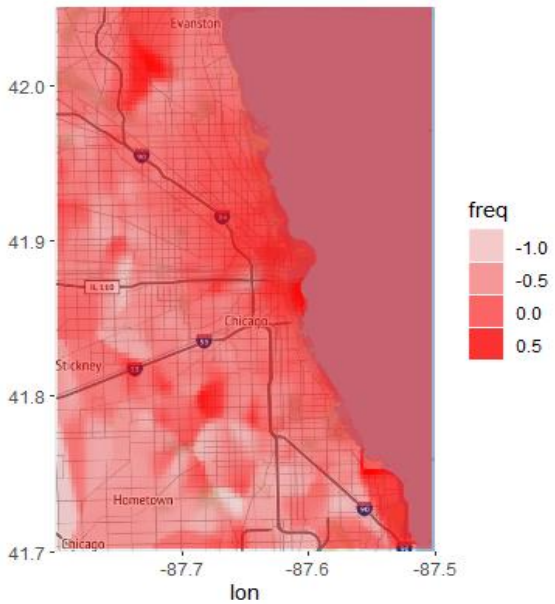
In addition, a quick look at this plot of different log(prices) in Chicago's Airbnb listings reveal a spatial pattern in terms of price that is not linear. Hence, we will move onto the 2-dimensional nonparametric regression of the *log(price) residuals* on both longitude and latitude. By using residuals as a response, we are able to *control the effect* of other categorical variables in the prediction of our prices based on location.

2-Dimensional Local Linear Regression

First, let us consider the residuals of Log(price) as a response variable. Through the bandwidth selection process, the optimal bandwidth size is h=0.00612.





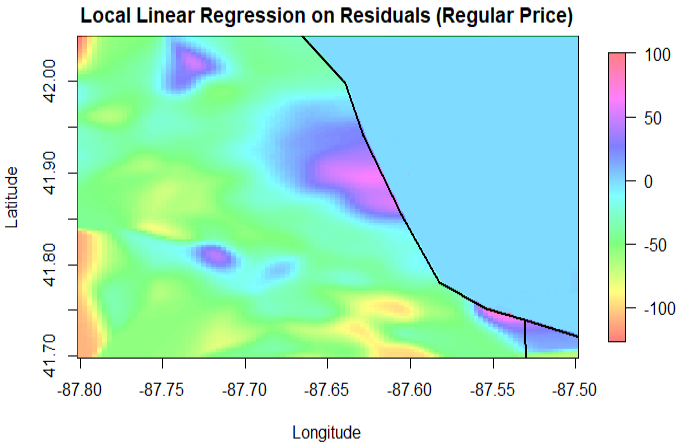


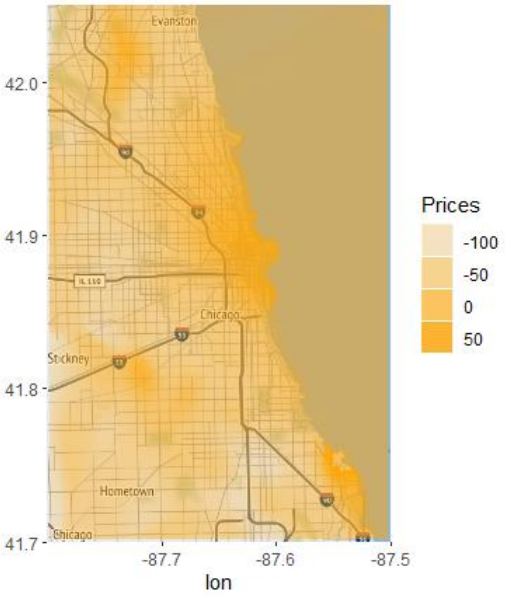
For better visualization, 2 plots are made from the regression predictions. The first plot is in rainbow color to contrast the difference in residuals between different neighborhoods. The other plot allows one to observe the pricier neighborhoods over an actual google map. The darker the red shade is, the more expensive it is, and the lighter, the cheaper. Hence, the regression map expresses the additional effect that a location has on an Airbnb's listing price. Since calculation of the Confidence interval is challenging in a 2-dimensional covariate setting, I will not consider it here.

Next, evaluate the model using RMSE. First, find the predicted residual based on the location, then add back the predicted value based on the parametric regression model. RMSE = **46.147**.

Regress on Residuals measured in Regular Price

Similarly, we also want to use the same method, except with the residuals of regular price as a response instead of residuals of log(price). The optimal bandwidth in this model is h = 0.00937, and the predicted residual values are presented below: From the plots, we noticed that the regression is much smoother than the once where we regress on the residuals of log(price).





The RMSE is found to be = **49.99**. Hence, based on minimizing the RMSE, the model using residuals of log(price) is found to be more accurate.

Visually, observe that the regions with residuals below 0 (implying that their locations actually cause the prices to be worth less given other features of the listing are constant) are located in the south and west side of Chicago, which correlates with the more violent regions of the city. On the other hand, 3 regions are found to be significantly more expensive. They are downtown for obvious reasons, and the area surrounding Midway Airport (middle left of plot). There is an interesting peak spot at the top left, which is the Lincolnwood village. Indeed, it was discovered that "*Lincolnwood home prices are ranked among the most expensive in America*", and is a significantly white-collar village.

Additive Model

Let us evaluate the dataset using a final method – additive model with the R built-in function, a method that decomposes the function into a sum of univariate regression functions over the variables, and in which the regression functions are arbitrary. This method is chosen since additive estimates tend to balance the strengths of fully nonparametric and parametric estimates – Additive estimates have a lower variance than fully nonparametric ones, and a lower bias than parametric estimates.

```
Call: gam(formula = price.log ~ s(people) + s(rooms) + s(ratings.old) +
    s(review.size) + s(longitude) + s(latitude) + no.private +
    living, data = train.set2)
Anova for Parametric Effects
                   Df     Sum Sq   Mean Sq    F value     Pr(>F)
s(people)         1.0000  958.73723 958.73723 4518.30365 < 2.22e-16 ***
s(rooms)          1.0000   37.19381  37.19381  175.28572 < 2.22e-16 ***
s(ratings.old)    1.0000   23.12534  23.12534  108.98428 < 2.22e-16 ***
s(review.size)    1.0000   73.12056  73.12056  344.60006 < 2.22e-16 ***
s(longitude)      1.0000  143.12484 143.12484  674.51381 < 2.22e-16 ***
s(latitude)       1.0000  148.59093 148.59093  700.27422 < 2.22e-16 ***
no.private        1.0000   65.16024  65.16024  307.08493 < 2.22e-16 ***
living            1.0000  130.35143 130.35143  614.31573 < 2.22e-16 ***
Residuals      5691.0001 1207.57127   0.21219
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Anova for Nonparametric Effects
             Npar Df     Npar F              Pr(F)
(Intercept)
s(people)          3  19.450290 0.0000000000015293 ***
s(rooms)           3  10.280351 0.0000009555050700 ***
s(ratings.old)     3  14.882544 0.0000000011947654 ***
s(review.size)     3   0.215012           0.88604
s(longitude)       3  43.281732         < 2.22e-16 ***
s(latitude)        3 226.186376         < 2.22e-16 ***
no.private
living
```

From the R output, we predicted the test set, and obtained a RMSE of **46.941**.

Observe that this is very similar to the RMSE obtained above of **46.147**, and this is because there is some resemblance – our earlier method also decomposes the regression function into its parametric and nonparametric components, before adding the various effects linearly – our arbitrary regression functions are local linear regression for the nonparametric model, and regular linear/polynomial regression for the parametric components.

4. DISCUSSION & CONCLUSION

Evaluation: Nonparametric regression differs from parametric regression in the shape of the functional relationship between the response and explanatory variables. In the former, this relationship is not pre-determined, but can be adjusted to capture unusual and unexpected features of the data, such as spatial information in our case, when the relationship to price is unknown and nonlinear.

In the analysis of Airbnb's dataset to predict listing prices, most of the variables we utilize are related linearly to the response variable price. Yet, longitude and latitude, 2 related variables that indicate location are nonparametric. It is possible to utilize other location-related variables such as Zip Code and/or Neighborhood as a variable to indicate location. However, the latter are categorical variables, and we have to determine how general or discrete we have to choose these categories. If we choose zip code, we may end up having too many nominal categories in the variable creating hassle in the regression process. If we choose another factor like neighborhood or region, it may be too general and lack specificity. Hence, the optimal method is still to use latitude and longitude in a nonparametric method.

Observe that the best fit is given by both the additive model and the combination of nonparametric model for the location variable and parametric model for the other variables, since they provided almost the same RMSE for the test set, reflecting how a prediction model may often end up combining various methodologies.

Finally, let us answer the question I first set out to resolve. The predicted prices (per day) that I can rent out the following properties for is:

4-Room Apartment in Hyde Park (My apartment)	$177.90
1-Room Studio Apartment in Downtown Loop (Friend 1's apartment)	$110.57
5-Room Multi-Level House in Gold Coast (Friend 2's family house)	$266.55

Improvements: The stepwise regression process in the parametric model also revealed the importance of understanding causality in our prediction. At first glance, it may seem that a large number of good reviews may actually indicate high prices because many good reviews may improve the reputation of hosts, allowing them to charge higher amounts. However, this is the wrong causal relationship. It is more likely, after looking at the results, that large number of good reviews is correlated with lower prices. It is the low prices that attracted guests in the first place, and let them be easily satisfied with the amenities because

of the value. Hence, interpreting the causal direction can also enable us to select variables in our model more efficiently.

A point worth mentioning is that the MSE is significantly large, and can definitely be lowered. One reason is because we included ALL listings in Chicago as part of our analysis. However, there may exist hosts who are first-time users, or lack knowledge in their pricing, causing the listing price to not be a function of anything reasonable. A possible solution is to find a subset of listings by superhosts who have much experiences – these listings will likely be set at reasonable market prices that can be well modelled. Since Chicago is a smaller market, we can explore the New York City Airbnb market, which has over 50 000 listings – even a small subset of that is significantly large enough.

Local Linear Regression is only one of the other regression methods we have learnt. I employed this method since it is the technique we had the most experience in for modelling in 2-dimensional space, and the least complex to implement. As future extensions, we may consider other nonparametric regression methods, such as Spline regression or wavelet regression if the prediction problem becomes more complex. Otherwise, a simple nonparametric regression method will suffice.

- END -

APPENDIX A – Additional Information

	price	latitude	longitude	people	rooms	review.num	ratings.old	superhost
price	1.000	0.060	0.168	0.559	0.520	-0.124	0.065	-0.009
latitude	0.060	1.000	-0.461	0.031	0.033	0.018	0.126	0.069
longitude	0.168	-0.461	1.000	-0.005	-0.058	-0.053	-0.078	-0.079
people	0.559	0.031	-0.005	1.000	0.780	0.032	0.004	0.054
rooms	0.520	0.033	-0.058	0.780	1.000	-0.013	0.022	0.039
review.num	-0.124	0.018	-0.053	0.032	-0.013	1.000	0.033	0.283
ratings.old	0.065	0.126	-0.078	0.004	0.022	0.033	1.000	0.292
superhost	-0.009	0.069	-0.079	0.054	0.039	0.283	0.292	1.000
no.private	-0.149	-0.121	-0.002	-0.162	-0.107	-0.055	-0.049	-0.055
living	0.399	0.099	0.084	0.493	0.357	-0.010	0.052	0.029
price.log	0.865	0.113	0.192	0.573	0.499	-0.113	0.108	0.019
room.den	0.136	-0.018	0.053	0.489	-0.047	0.087	-0.027	0.032
review.size	-0.135	0.036	-0.083	0.062	0.020	0.829	0.057	0.360
ratings	-0.012	0.088	-0.096	0.042	0.027	0.234	0.443	0.317

	no.private	living	price.log	room.den	review.size	ratings
price	-0.149	0.399	0.865	0.136	-0.135	-0.012
latitude	-0.121	0.099	0.113	-0.018	0.036	0.088
longitude	-0.002	0.084	0.192	0.053	-0.083	-0.096
people	-0.162	0.493	0.573	0.489	0.062	0.042
rooms	-0.107	0.357	0.499	-0.047	0.020	0.027
review.num	-0.055	-0.010	-0.113	0.087	0.829	0.234
ratings.old	-0.049	0.052	0.108	-0.027	0.057	0.443
superhost	-0.055	0.029	0.019	0.032	0.360	0.317
no.private	1.000	-0.265	-0.298	-0.118	-0.045	-0.013
living	-0.265	1.000	0.596	0.265	-0.007	0.037
price.log	-0.298	0.596	1.000	0.205	-0.120	0.005
room.den	-0.118	0.265	0.205	1.000	0.097	0.045
review.size	-0.045	-0.007	-0.120	0.097	1.000	0.381
ratings	-0.013	0.037	0.005	0.045	0.381	1.000

Table 1 – *Correlation Matrix of Airbnb's Variables*

APPENDIX B – 2-Dimensional Local Linear Regression R Code Functions

```
#Define our Local Linear Regression functions
#Write 2-variable functions for Gaussian Kernel
Gaussian_fn <- function(x, y, xa, ya, h) {
        x.new = sqrt((x- xa)^2 + (y-ya)^2)
        1/(2*pi)*exp((-1/2)*((x.new/h)^2))
}
L_ii <- function (x, y, xi, yi, h, response) {
        estimate1 <- gaussian.fn(x,y,xi,yi,h)
        estimate2 <- estimate1/sum(estimate1)
        estimate2*response
}

loocv.score = function(xi, yi, response, h.vector) {
        length.x = length(xi)
        matrix1 <- matrix(nrow = length.x, ncol = length.x)
        count = NULL

        #Find the values of kernels in the estimator function
        for (i in 1:length.x) {
                for (j in 1:length.x) {
                        matrix1[i,j] <- (xi[i]-xi[j])^2 + (yi[i]-yi[j])^2
                }
        }

        for (h in h.vector) {
                matrix2 <- apply(matrix1, c(0,1),
                function(x)(1/(2*pi))*exp((-1/2)*x*(1/h^2)))
                matrix3 <- matrix2/rowSums(matrix2)
                Li <- diag(matrix3)
                pred <- matrix3 %*% response
                result <- sum((( pred-response)/(1-Li))^2)
                count <- c(count, result)
        }
        return (count)
}

#Function to find actual estimator function
kernel.estimate <- function(xi, yi, x, y, h, response) {
        response[is.na(response)] <-0
        length.xseq = length(x)
        length.yseq = length(y)
        pred.matrix <- matrix(0, length.xseq, length.yseq)
        for (i in 1:length.xseq) {
                for (j in 1:length.yseq) {
                        temppred <- gaussian.fn(x[i], y[j], xi, yi, h)
                        temppred <- temppred/sum(temppred)
                        temppred[is.na(temppred)] <- 0
                        pred <- sum(temppred*response)
                        pred.matrix[i,j] <- pred
                }
        }
```

```
        pred.matrix
}

#Function to do final prediction
llr.predict <- function(xi, yi, xtest, ytest, h, response) {
        response[is.na(response)] <-0
        length.xseq = length(xtest)
        pred.list = NULL
        for (i in 1:length.xseq) {
                temppred <- gaussian.fn(xtest[i], ytest[i], xi, yi, h)
                temppred <- temppred/sum(temppred)
                pred.val <- sum(temppred*response)
                pred.list = c(pred.list, pred.val)
        }
        pred.list
}
```