Improving Rental Price in the United States

Chun-Li Hou

Introduction

The client is the management in the rental industry. Comparing the United States and China, the people in the United States prefer renting a place than buying it for living. According to online research, it is only 35% of Americans under 35 years old who are willing to buy their own house. On the other hand, the proportion in China is more than 70%. We hereby know that the rental industry is relatively potential in the United States. Thus, how to set a profitable price and maximize profits are important issues to the management in the rental industry.

By analyzing the data, it can help us to clear several questions: (i) what regression model can we use to predict the most accurate rental price? (ii) what property attributes are important and interesting to decide the retail price?

By answering these questions, we can bring the management in the rental industry some interesting insights, which can help them to be profitable.

Hypothesis

Question i. What regression model can we use to predict the most accurate rental price?

H0: all coefficients are zero

H1: one of the coefficients is nonzero

Question ii. What property attributes are important and interesting to decide the retail price?

H0: the coefficient beta is zero

H1: the coefficient beta is nonzero

*For multiple linear regression model & polynomial non-linear regression model use

Data Description

The data comes from the machine learning repository in the University of California at Irvine (UCI). The data is about rental price in the United States. It is collected from the web crawling through 12 different rental websites in the United States, such as GoSection8.com, HomeRentals.com, Listanza.com, ListedBuy.com, RealEstateAgent.com, etc. The time point is on December 28, 2019. After cleaning the data with no empty in the column of price, it has 22 columns and 10,000 rows. The data is shown in a glimpse in Figure 1. The attribute information is shown in a table in Figure 2. Since there are still some attributes as a character type that cannot be used in the model, we will proceed with some data cleaning processes more. The detailed processes will be introduced in the next paragraph.

Data Preparation

We decide to break some columns for more useful attributes, drop some meaningless columns, and keep some important columns as the original.

Firstly, we take price as the dependent variable. Furthermore, we transform price based on price_type as monthly in our standard. The column of price_type has monthly, weekly, and monthly/weekly. After checking, we turn the weekly into monthly due to the simple typing error for the original data. The monthly/weekly is a mistake after making sure by searching on Google. We find the property with a price in weekly. Thus, we take the price and divide it by 7 then multiple it by 30.4, which is the average days per month. In summary, we correct price by standardizing with price_type.

Secondly, as for the usable independent variables, these are id, bathrooms, bedrooms, and square_feet, source. Thirdly, we perform some breakdown or transformation to generate extra variables. We break down category into three types, including cat_apart, cat_home, and cat_short.

Then, we extract amenities into special features that are more possible to be concerned when renting a property. The extra generated attributes are clubhouse, fireplace, gated, balcony, sport, gym, pool, parking, laundry, tv, elevator, and wood_floor. As for the rest features in amenities, they are considered as a standard facility for all kinds. The column of pet_allowed is factorized.

The column of metro_area is based on whether the property city is in the top 10 largest metropolitan areas, which refers to the website in reference. The web data is from the United States Bureau of Economic Analysis that estimates the GDP for each United States metropolitan area in 2016. The process also corrects some data with the null value. By using the column of latitude and

longitude on Google Map and the column of body with a manual check, we define 77 null values with their city name. The preparation processes conduct in Excel. These non-numeric variables will be set as a factor type to use in the model.

Data Preprocess

After data preparation, we can start importing data into R to preprocess data. Firstly, the data in a glimpse is in Figure 3 and the column description table is in Figure 4. After that, we check whether we have any missing data. We decide to drop the observations with missing values. Then, we pick up the near-zero variable to drop due to a lack of variance contributing to the model. Our ready-to-go data has 17 columns and 9,961 rows.

Finally, we remove source and id for no use in the model. Also, we remove the outliers. And, we do data partition to have a train set for modeling fitting and a test set for model evaluating. At the end, the train set has 7,968 rows and the test set has 1,990 rows with 15 columns.

Exploratory Data Analysis

Due to having more factor variables than numeric variables, it is not workable to perform the correlation matrix plot. Thus, instead of that, we use ggpairs function to show up all possible demonstrations showing in Figure 5. We can quickly know about the correlation between a pair of variables. For example, the strongest correlation with price is square_feet. Then, we take a few further steps to see in different plots with interesting variables.

Firstly, we see the distribution of each numeric variable. The distribution for price or square_feet is more normal distributed after log transformation in Figure 6 and Figure 7. The distribution for bedrooms or bathrooms is skewed but match up with our expectation in Figure 8 and Figure 9. Overall, there is no unexpected sign with these numeric variables.

As for the factor variables, we use two sample t-test to see the price difference. As setting the significant level as $p < 0.001$, with the two sample t-test, we find that gym, laundry, elevator, and pets_allowed are statistically the same on price between having or not having it.

Furthermore, we try to combine more dimensions with more variables together to plot out to process datamining. For example, based on the scatterplot in Figure 10, we can clearly see the correlation between bathrooms and bedrooms. Also, price in log and square_feet in log are correlated with either bedrooms or bathrooms.

Analysis

In the first step, we use seven different regression machine learning algorithms with the train set to find out the best regression model. We use the test set to evaluate the different regression models. From Figure 11, we know that the best regression model is the multiple linear regression model (MLR) and polynomial nonlinear regression model (PNLR) due to the lowest root mean square error (RMSE).

We start from the PNLR model. After checking assumptions, we find that the original PNLR model violates some of those. Therefore, we try to fix these violations in some ways. After deleting outliers, transforming the dependent variable, and removing independent variables, the PNLR model fits all of the assumptions, which are outlier, heteroscedasticity, collinearity, and normality. But, the model is more as the MLR model. The results demonstrate in Figure 12 and Figure 13.

This linear regression model is a log-linear regression model. We start to interpret the model. We can see the estimates and the coefficients in Figure 14. Only the estimates of bedrooms, fireplace, and pets_allowed are not significant. The multiple R squared is 0.332, which means only 33.2% of the dependent variable (price) can be explained by the model inputs (the independent variables). The adjusted R squared is 0.331. We take an estimate as an example of interpreting its coefficient to establish causality. As increasing the property's square feet by 100 units, the price of the property increases by 4%.

Lastly, we can start answer our business questions and provide insights. The first insight is as considering exercise facility, which kind does positively affect the price? We can see the estimates of sport and gym. We will know that having the outdoor sport facility (sport), such as basketball field or tennis court, the price decreases by 13.51%. On the other hand, having the indoor sport facility (gym), such as gym or fitness center, the price increases by 11.67%. Thus, we know that the indoor facility would bring more profit to the apartment value.

The second insight is to answer what apartment attribute affected the price is the most interesting. We know the location can affect the price easily. Our model also identify this point. The second highest coefficient is elevator, which means whether the apartment has an elevator or not. This is an interesting insight. In detail, having an elevator in the apartment, the price increases by 22.31%. As we know, living in which floors has its pros and cons. Therefore, having an elevator might not be an impact on price. However, from our model result, we can say having an elevator would bring positive impact on the price.

The last one insight is how much should the new releasing apartment set the price? We use this model for real-world validation. We search the crawling websites for a new releasing apartment.

The property is 808 square feet, with 1 bedroom and 1 bathroom, locates in Richardson, TX, has gym and parking area, and is pet allowed. The predicted price is $1,225. Thus, with this model, we can provide a reasonable price for a new releasing apartment.

Summary

After the analysis, we know the multiple linear regression as the log-linear regression model is our best regression model on this data. We can use this model to help the management in the rental industry to set a profitable price and maximize profits. For example, we can use this model to predict the accurate and reasonable rent for a new releasing apartment. Furthermore, we generate some interesting insights. Those can help the management to have better profits.

In conclusion, the rental industry is thriving regardless of economic fluctuations. The need for a profitable price is important to keep the industry thriving. In this research, we can connect the machine learning model selection and the best selected one model to help to identify the profitable price and to provide interesting insights for making necessary business decisions in the rental industry.

Appendix

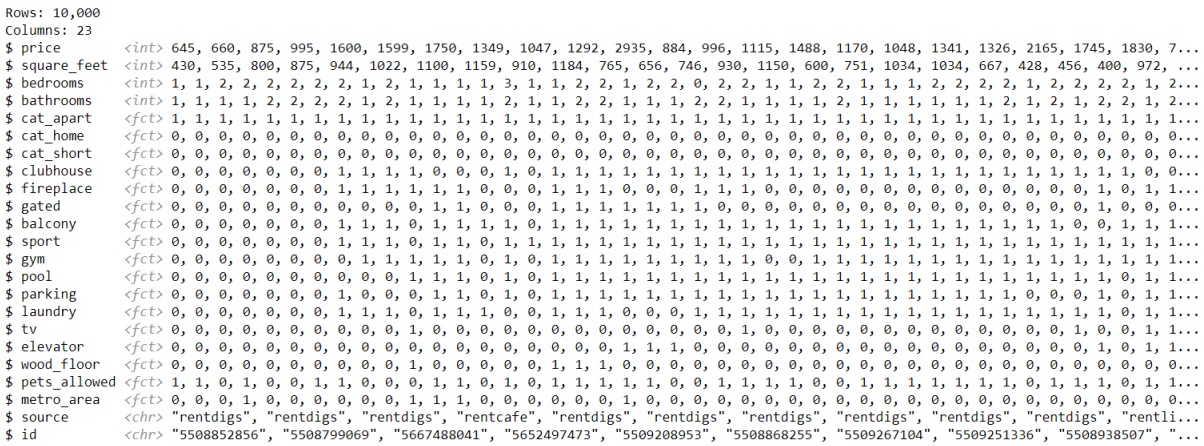
Figure 1. The original data has 22 columns and 10,000 rows.

```
Rows: 10,000
Columns: 22
$ id            <dbl> 5668626895, 5664597177, 5668626833, 5659918074, 5668626759, 5667891676, 5668627426, 5668626687, 5668610290, 5668627023, 566862...
$ category      <chr> "housing/rent/apartment", "housing/rent/apartment", "housing/rent/apartment", "housing/rent/apartment", "housing/rent/apartmen...
$ title         <chr> "Studio apartment 2nd St NE, Uhland Terrace NE, Washington, DC 20002", "Studio apartment 814 Schutte Road", "Studio apartment ...
$ body          <chr> "This unit is located at second St NE, Uhland Terrace NE, Washington, DC 20002, Washington, 20002, DCMonthly rental rates rang...
$ amenities     <chr> "null", "null", "null", "null", "null", "Dishwasher,Elevator,Patio/Deck,Pool,Storage", "null", "null", "Refrigerator", "null",...
$ bathrooms     <chr> "null", "null", "1", "1", "null", "1", "null", "null", "1", "null", "null", "null", "null", "null", "1", "1", "1", "1", "1", "...
$ bedrooms      <chr> "0", "1", "0", "0", "0", "0", "0", "0", "0", "0", "0", "0", "1", "0", "0", "1", "1", "1", "1", "1", "1", "1", "1", "1", "1", "...
$ currency      <chr> "USD", "USD", "USD", "USD", "USD", "USD", "USD", "USD", "USD", "USD", "USD", "USD", "USD", "USD", "USD", "USD", "USD", "USD", ...
$ fee           <chr> "No", "No", "No", "No", "No", "No", "No", "No", "No", "No", "No", "No", "No", "No", "No", "No", "No", "No", "No", "No", "No", ...
$ has_photo     <chr> "Thumbnail", "Thumbnail", "Thumbnail", "Thumbnail", "Thumbnail", "Thumbnail", "Thumbnail", "Thumbnail", "Thumbnail", "Thumbnai...
$ pets_allowed  <chr> "None", "None", "None", "None", "None", "null", "None", "None", "None", "None", "None", "None", "Cats,Dogs", "None", "None", "...
$ price         <int> 790, 425, 1390, 925, 880, 2475, 1800, 840, 1495, 890, 990, 840, 1795, 1090, 1695, 1560, 1560, 1000, 950, 625, 600, 544, 525, 4...
$ price_display <chr> "$790", "$425", "$1,390", "$925", "$880", "$2,475", "$1,800", "$840", "$1,495", "$890", "$990", "$840", "$1,795", "$1,090", "$...
$ price_type    <chr> "Monthly", "Monthly", "Monthly", "Monthly", "Monthly", "Monthly", "Monthly", "Monthly", "Monthly", "Monthly", "Monthly", "Mont...
$ square_feet   <int> 101, 106, 107, 116, 125, 130, 132, 136, 138, 141, 146, 150, 156, 178, 190, 200, 200, 200, 200, 200, 200, 200, 200, 215, 2...
$ address       <chr> "null", "814 Schutte Rd", "null", "1717 12th Avenue", "null", "350 West 50th St", "2432 Penmar Avenue", "null", "333 Hyde St",...
$ cityname      <chr> "Washington", "Evansville", "Arlington", "Seattle", "Arlington", "Manhattan", "Venice", "Washington", "San Francisco", "Washin...
$ state         <chr> "DC", "IN", "VA", "WA", "VA", "NY", "CA", "DC", "CA", "DC", "DC", "DC", "AZ", "DC", "CA", "NC", "NC", "TX", "CA", "GA", "NC", ...
$ latitude      <chr> "38.9057", "37.9680", "38.8910", "47.6160", "38.8738", "40.7629", "33.9932", "38.9328", "37.7599", "38.9118", "38.9118", "38.9...
$ longitude     <chr> "-76.9861", "-87.6621", "-77.0816", "-122.3275", "-77.1055", "-73.9885", "-118.4609", "-77.0297", "-122.4379", "-77.0132", "-7...
$ source        <chr> "RentLingo", "RentLingo", "RentLingo", "RentLingo", "RentLingo", "Listanza", "RentLingo", "RentLingo", "RentLingo", "RentLingo...
$ time          <int> 1577359415, 1577017063, 1577359410, 1576667743, 1577359401, 1577289784, 1577359461, 1577359393, 1577358313, 1577359424, 157735...
```

Figure 2. The original data attribute information shows in descriptive.

Name	Description
id	unique identifier of apartment
category	category of classified
title	title text of apartment
body	body text of apartment
amenities	like AC, basketball, cable, gym, internet access, pool, refrigerator etc.
bathrooms	number of bathrooms
bedrooms	number of bedrooms
currency	price in current
fee	fee
has_photo	photo of apartment
pets_allowed	what pets are allowed dogs/cats etc.
price	rental price of apartment
price_display	price converted into display for reader
price_type	price in USD
square_feet	size of the apartment
address	where the apartment is located
cityname	where the apartment is located
state	where the apartment is located
latitude	where the apartment is located
longitude	where the apartment is located
source	origin of classified
time	when classified was created

Figure 3. The prepared data has 23 columns and 10,000 rows.



```
Rows: 10,000
Columns: 23
$ price         <int> 645, 660, 875, 995, 1600, 1599, 1750, 1349, 1047, 1292, 2935, 884, 996, 1115, 1488, 1170, 1048, 1341, 1326, 2165, 1745, 1830, 7...
$ square_feet   <int> 430, 535, 800, 875, 944, 1022, 1100, 1159, 910, 1184, 765, 656, 746, 930, 1150, 600, 751, 1034, 1034, 667, 428, 456, 400, 972, ...
$ bedrooms      <int> 1, 1, 2, 2, 2, 2, 2, 1, 2, 1, 1, 1, 3, 1, 1, 2, 2, 1, 2, 2, 0, 2, 2, 1, 1, 2, 2, 1, 1, 1, 2, 2, 2, 1, 2, 2, 2, 1, 2...
$ bathrooms     <int> 1, 1, 1, 1, 2, 2, 2, 1, 2, 1, 1, 1, 1, 2, 1, 1, 2, 2, 1, 1, 1, 2, 2, 1, 1, 1, 2, 1, 1, 1, 1, 1, 2, 1, 2, 1, 2, 2, 1, 2...
$ cat_apart     <fct> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1...
$ cat_home      <fct> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
$ cat_short     <fct> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
$ clubhouse     <fct> 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0...
$ fireplace     <fct> 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1...
$ gated         <fct> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0...
$ balcony       <fct> 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1...
$ sport         <fct> 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1...
$ gym           <fct> 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1...
$ pool          <fct> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1...
$ parking       <fct> 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1...
$ laundry       <fct> 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1...
$ tv            <fct> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1...
$ elevator      <fct> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1...
$ wood_floor    <fct> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
$ pets_allowed  <fct> 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1...
$ metro_area    <fct> 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0...
$ source        <chr> "rentdigs", "rentdigs", "rentdigs", "rentcafe", "rentdigs", "rentdigs", "rentdigs", "rentdigs", "rentdigs", "rentdigs", "rentli...
$ id            <chr> "5508852856", "5508799069", "5667488041", "5652497473", "5509208953", "5508868255", "5509267104", "5509251336", "5508938507", "...
```

Figures 4. The prepared data attribute information shows in descriptive.

Name	Type	Description
price	int	price of apartment in dollar
square_feet	int	size of apartment
bedrooms	int	number of bedroom
bathrooms	int	number of bathroom
cat_apart	fct	apartment
cat_home	fct	home
cat_short	fct	short-term
clubhouse	fct	clubhouse
fireplace	fct	fireplace
gated	fct	gated
balcony	fct	patio/deck
sport	fct	field/court/playground
gym	fct	gym
pool	fct	pool
parking	fct	parking
laundry	fct	washer/dryer
tv	fct	tv
elevator	fct	elevator
wood_floor	fct	wood floor
pets_allowed	fct	pets allowed
metro_area	fct	metropolitan area
source	chr	origin website source
id	chr	unique identifier

Figure 5. The overall plot shows the general explanatory data analysis information.

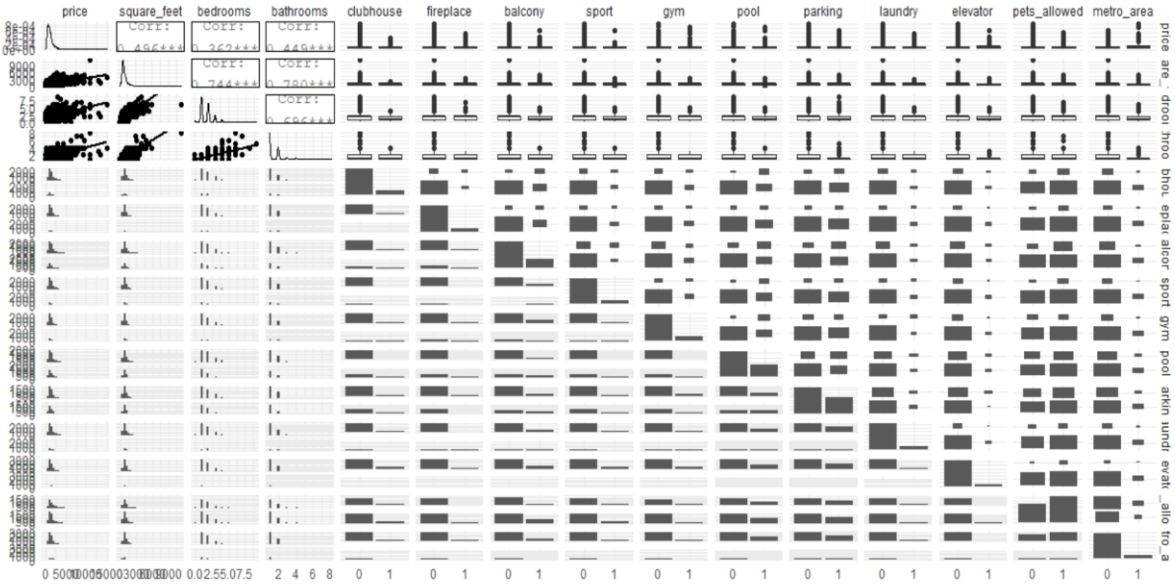


Figure 6. The distribution of price in log-transformation is normal distributed.

Figure 7. The distribution of square feet in log-transformation is normal distributed.

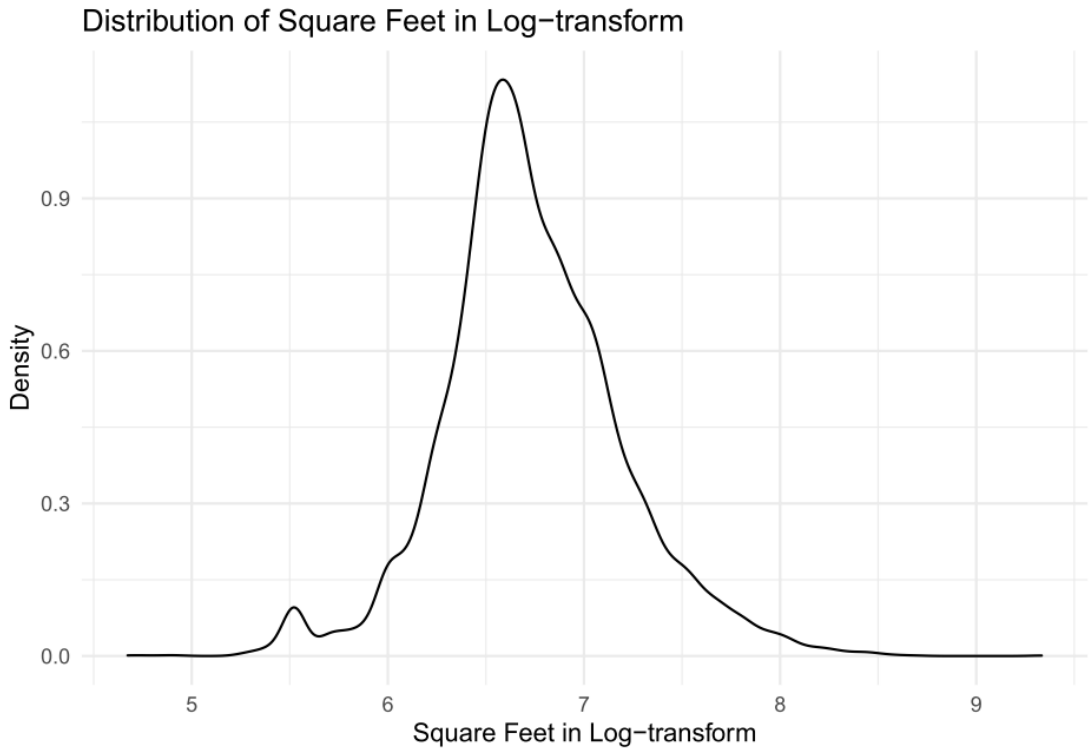


Figure 8. The barplot of number of bedrooms is skewed.

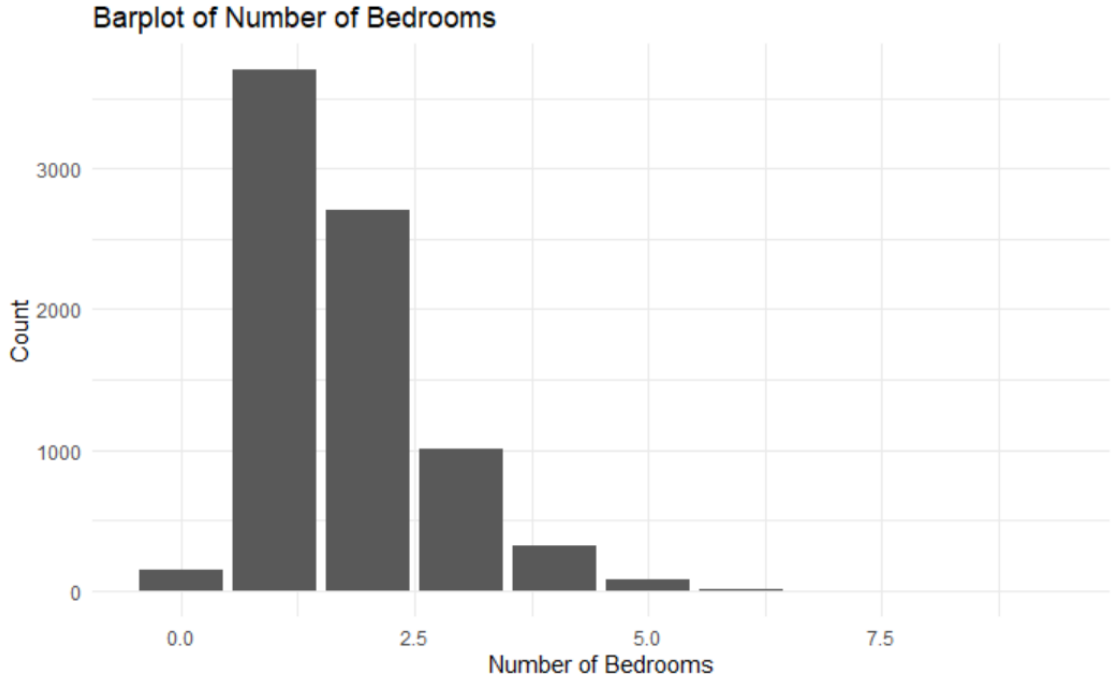


Figure 9. The barplot of number of bathrooms is skewed.

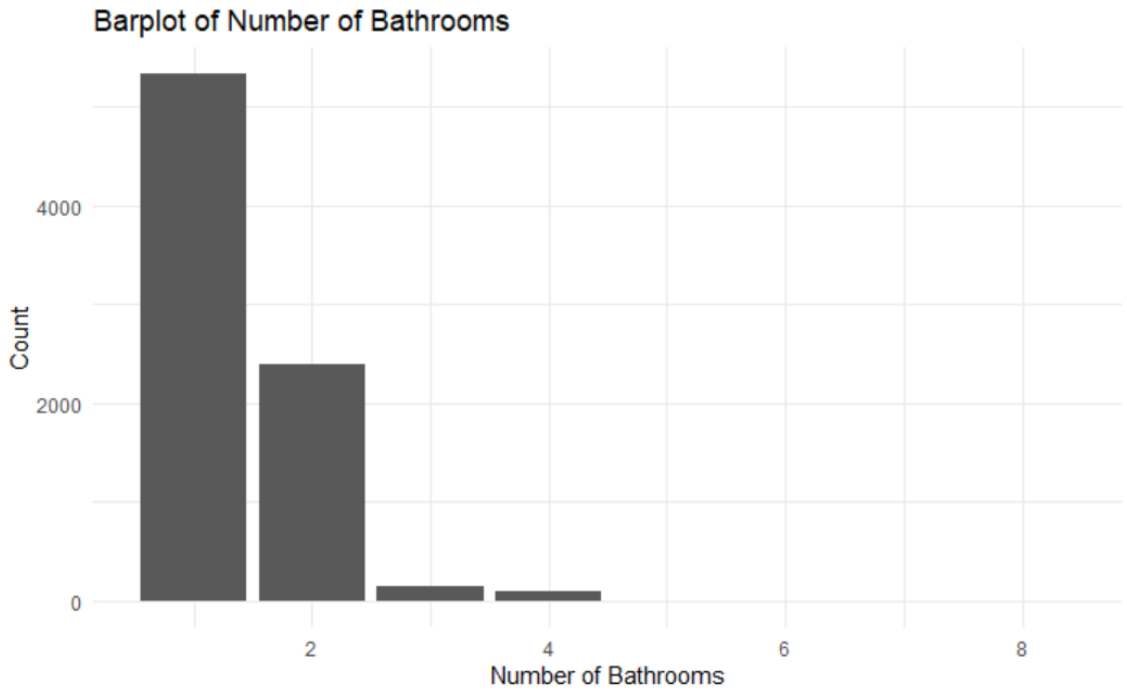


Figure 10. The combination of numeric variables plot shows a correlated relationship.

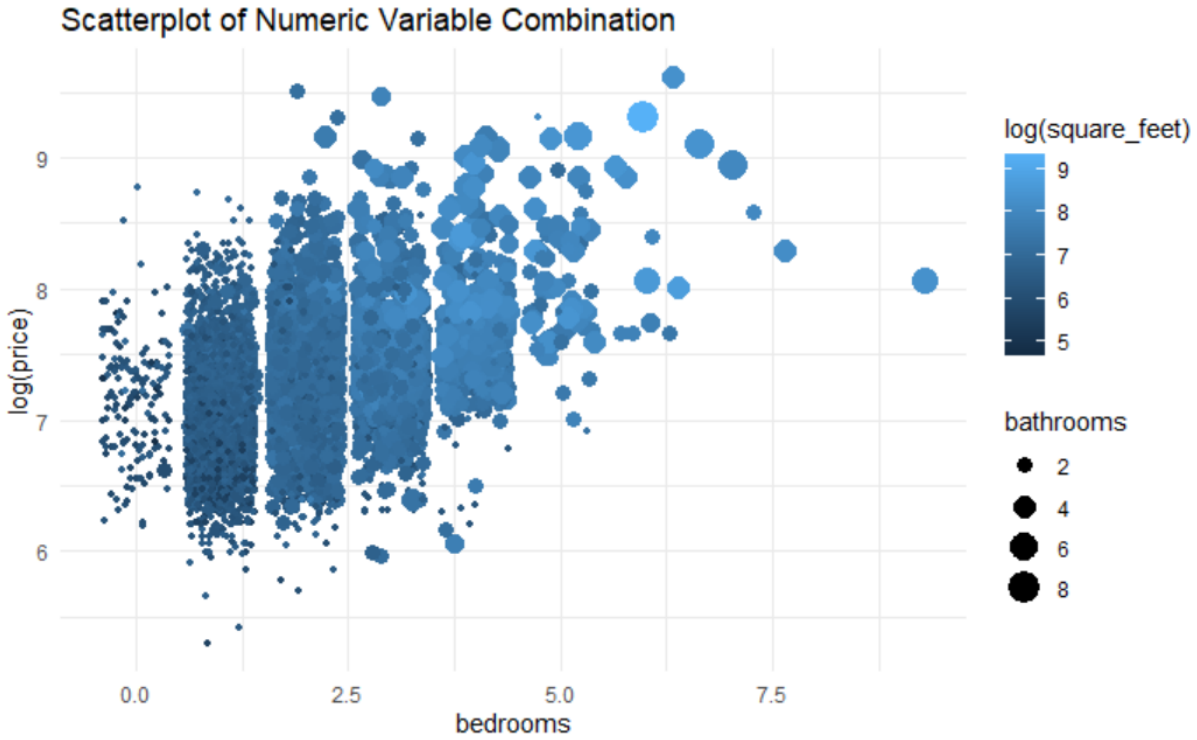


Figure 11. The table shows the result of the model evaluation.

	RMSE <dbl>
PNLR	705
MLR	705
RF	706
KSVR	717
SVR	742
DT	784
ANN	973

Figure 12. Plots show no issue with outlier, heteroscedasticity, and normality.

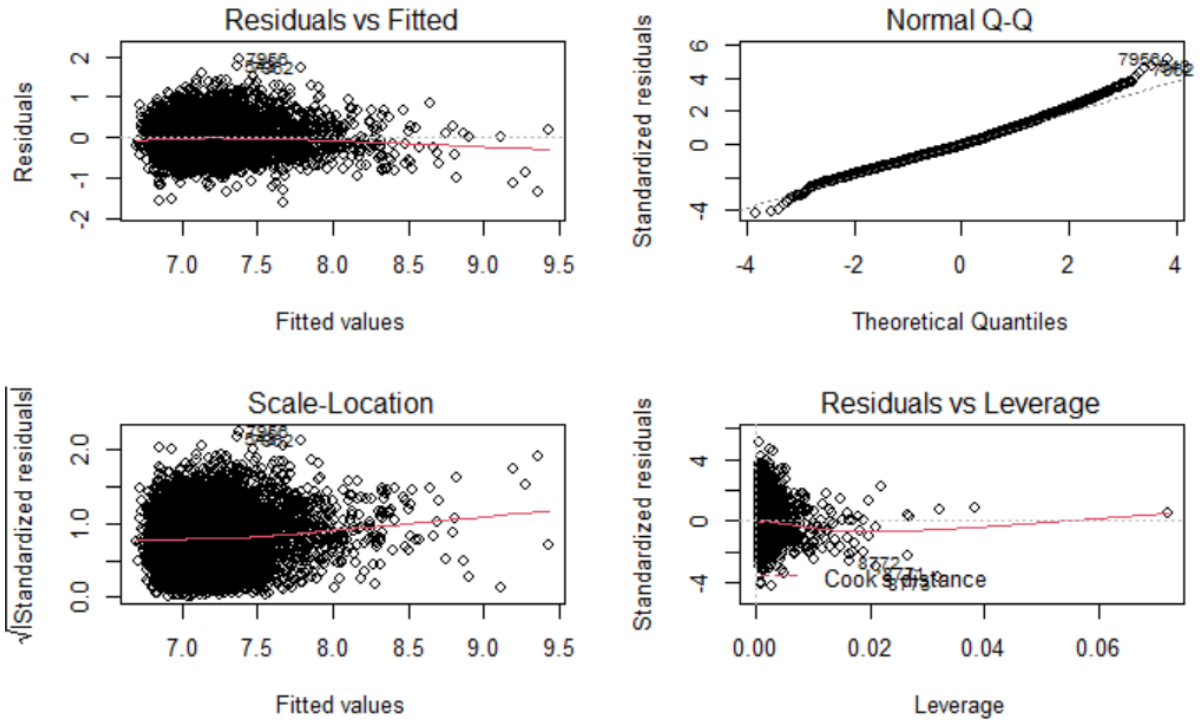


Figure 13. The table indicates no issue with collinearity.

Variable <chr>	VIF <dbl>	Collinearity Issue <chr>
square_feet:pets_allowed	5.76	No
metro_area	5.13	No
square_feet:metro_area	5.05	No
pets_allowed	4.62	No
square_feet	4.28	No
bathrooms	2.77	No
bedrooms	2.51	No
gym	1.17	No
sport	1.15	No
parking	1.13	No
fireplace	1.11	No
elevator	1.08	No

Figure 14. The table shows the log-linear regression estimates and coefficients.

estimate <dbl>	lcl <dbl>	ucl <dbl>	p-value <dbl>	Variable <chr>
6.6388	6.6082	6.6694	0.0000	(Intercept)
0.0004	0.0003	0.0004	0.0000	square_feet
0.0021	-0.0120	0.0163	0.7673	bedrooms
0.1152	0.0919	0.1385	0.0000	bathrooms
-0.0368	-0.0653	-0.0083	0.0115	fireplace1
-0.1351	-0.1623	-0.1080	0.0000	sport1
0.1167	0.0915	0.1420	0.0000	gym1
-0.0437	-0.0620	-0.0255	0.0000	parking1
0.2231	0.1881	0.2581	0.0000	elevator1
0.0301	-0.0062	0.0664	0.1042	pets_allowed1

Reference

1. UCI Machine Learning Repository – Apartment for Rent Classified Dataset – Fredrick Nilsson – URL (https://tinyurl.com/y3zzn2au)
2. Metro Area in the US – Jeff Desjardins – URL (https://tinyurl.com/y2h5o2r3)
3. Sublet.com – 1B1B apartment in Richardson TX – URL (https://tinyurl.com/y59wtrzx)

Attachment

The following pages are the RMD file with code and output in PDF format.

Attachment

Environment & Data

1. Environment Setting

```
if(!require("pacman")) install.packages("pacman")
pacman::p_load(dplyr, caret, GGally, Hmisc, broom, tidyr, car, e1071, rpart, rpart.plot,
               rattle, randomForest, h2o, forecast)
options(digits = 3)
theme_set(theme_minimal())
set.seed(123)
```

2. Data Preprocessing

a. Data Loading

```
df.0 = read.csv("apartments_for_rent.csv")
```

b. Data Type Encoding

```
df.0[, c(5:21)] = lapply(df.0[, c(5:21)], factor)
df.0[, c(22:23)] = lapply(df.0[, c(22:23)], as.character)
```

c. Missing Data Dealing

```
df.1 = na.omit(df.0)
```

d. Variable Selecting

```
nzv = nearZeroVar(df.1)
df.2 = df.1[, -nzv]
```

e. Outlier Removing

```
df.3 = df.2[, -c(16, 17)]
remove = c("319", "3828", "8775")
df.4 = df.3[row.names(df.3) %nin% remove, ]
```

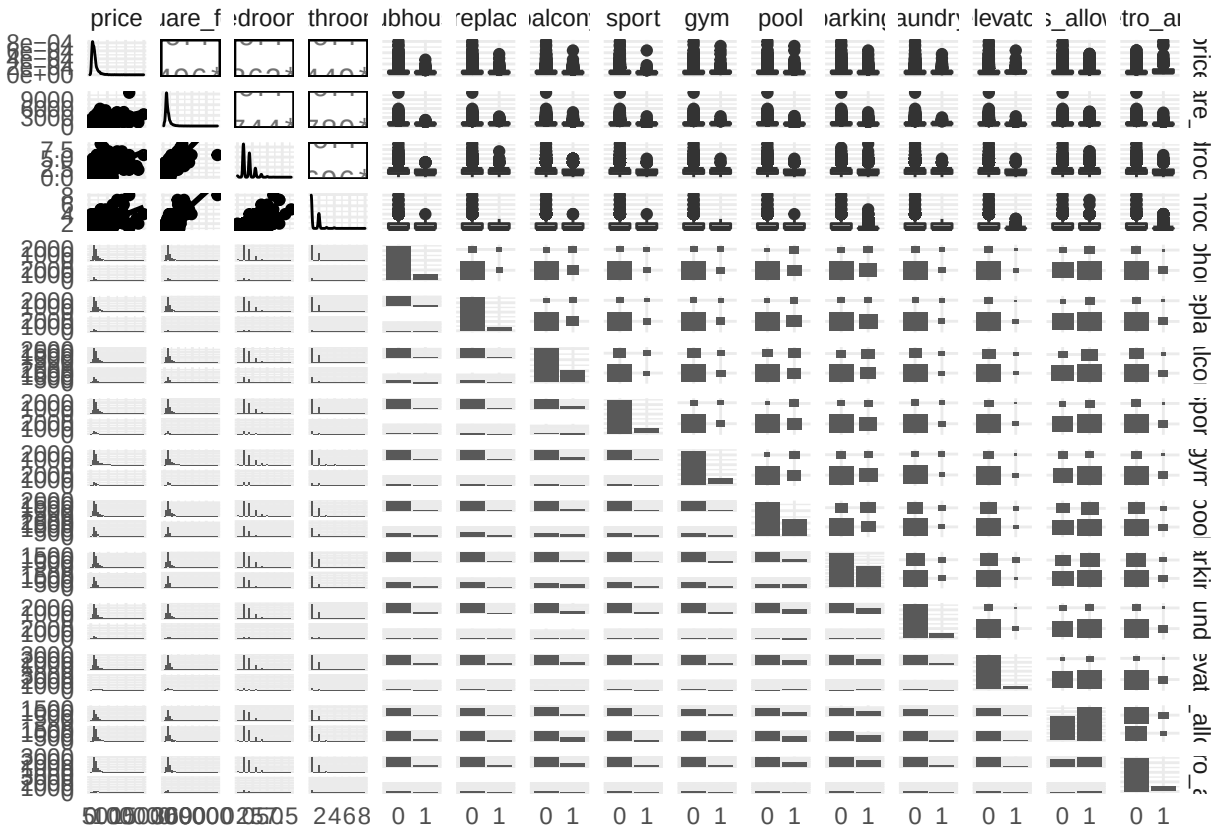
f. Data Partitioning

```
split = createDataPartition(df.4$price, p = 0.8, list = F)
train.set = df.4[split, ]
test.set = df.4[-split, ]
```

The dimension of train.set is 7968 rows and 15 cols. The dimension of test.set is 1990 rows and 15 cols.

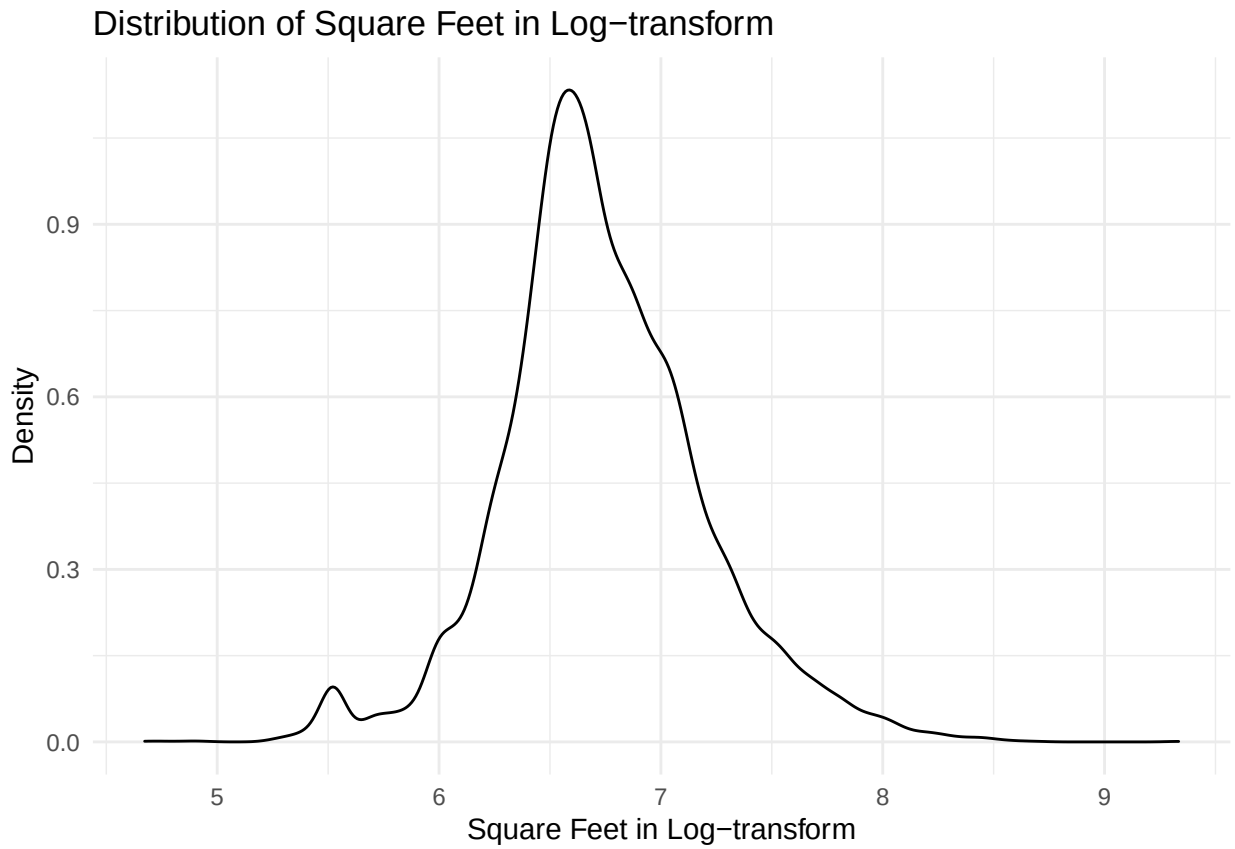
Exploring Data Analysis

```
ggpairs(train.set, lower = list(continuous = wrap("smooth", method = "lm")))
```

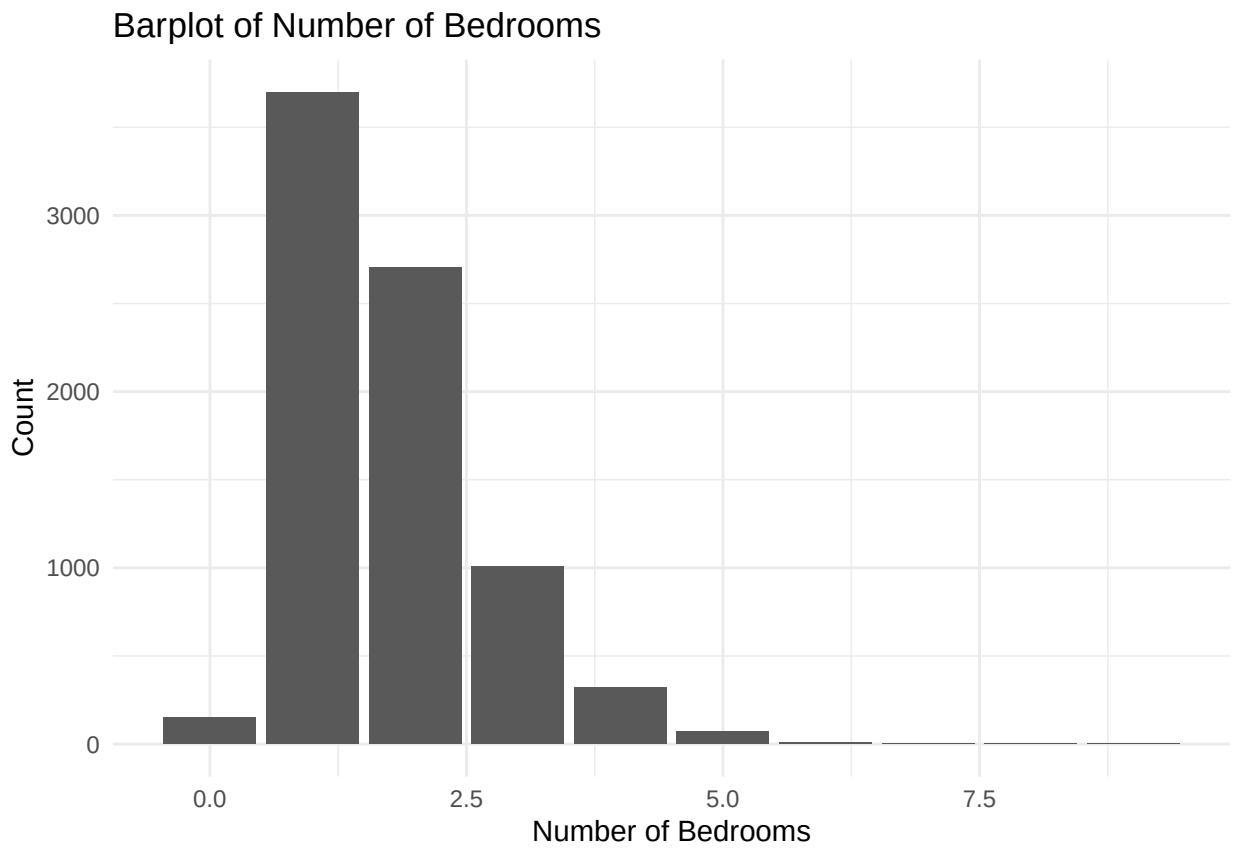


```r
# price in log
qplot(data = train.set, x = log(price), geom = c("density")) +
  labs(title = "Distribution of Price in Log-transform",
       x = "Price in Log-transform", y = "Density")
```

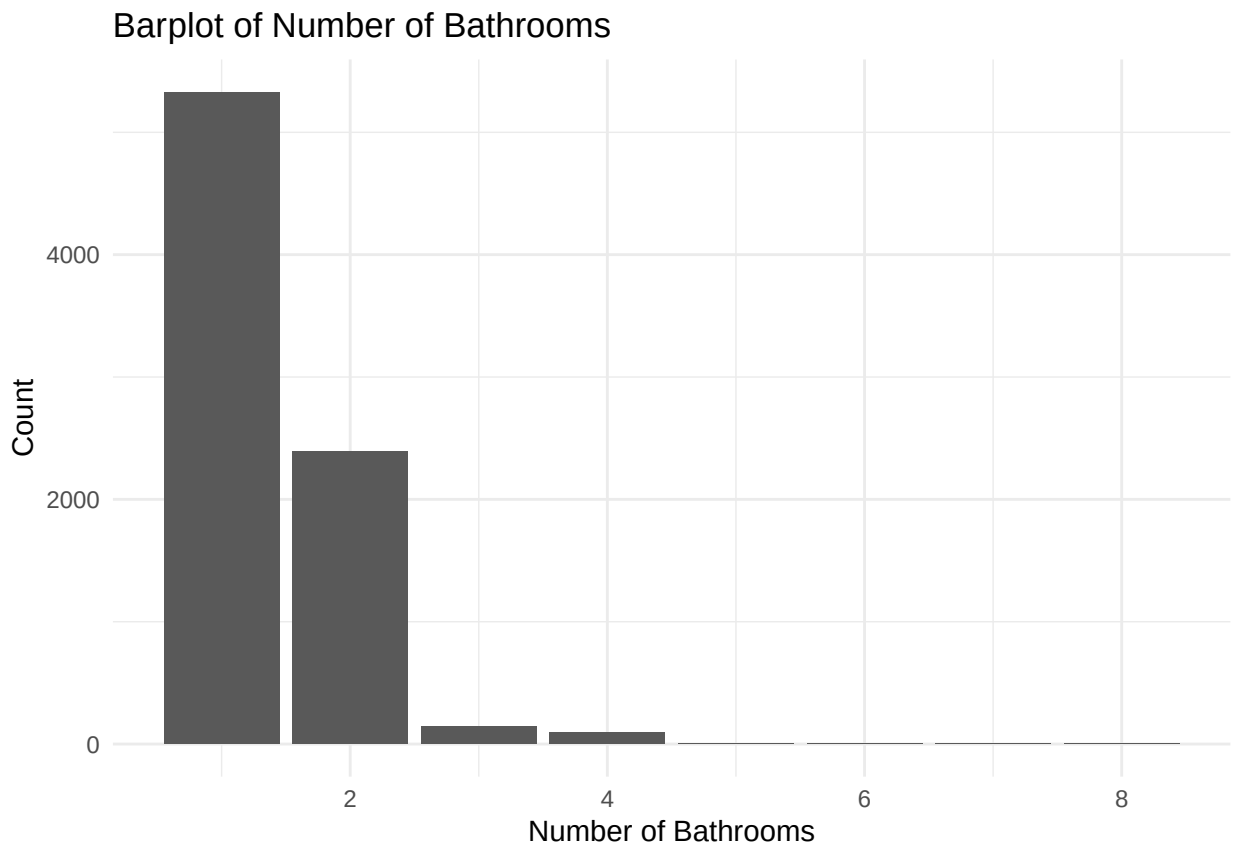
```r
# square_feet in log
qplot(data = train.set, x = log(square_feet), geom = c("density")) +
  labs(title = "Distribution of Square Feet in Log-transform",
       x = "Square Feet in Log-transform", y = "Density")
```



```r
# bedrooms
qplot(data = train.set, x = bedrooms, geom = c("bar")) +
  labs(title = "Barplot of Number of Bedrooms", x = "Number of Bedrooms", y = "Count")
```



```r
# bathrooms
qplot(data = train.set, x = bathrooms, geom = c("bar")) +
  labs(title = "Barplot of Number of Bathrooms", x = "Number of Bathrooms", y = "Count")
```

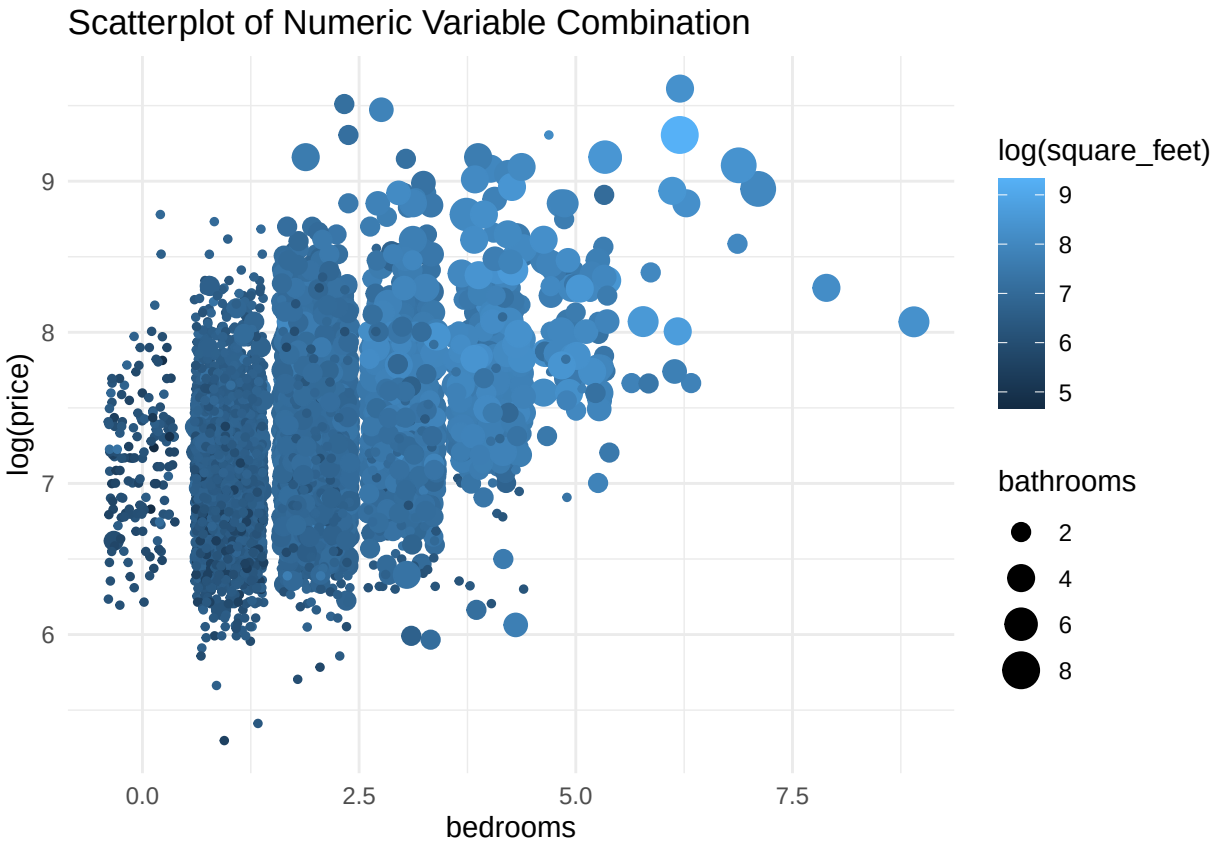


```r
# clubhouse
temp = t.test(price ~ clubhouse, data = train.set)
clubhouse = temp$p.value

# fireplace
temp = t.test(price ~ fireplace, data = train.set)
fireplace = temp$p.value

# balcony
temp = t.test(price ~ balcony, data = train.set)
balcony = temp$p.value

# sport
temp = t.test(price ~ sport, data = train.set)
sport = temp$p.value

# gym
temp = t.test(price ~ gym, data = train.set)
gym = temp$p.value

# pool
temp = t.test(price ~ pool, data = train.set)
pool = temp$p.value

# parking
temp = t.test(price ~ parking, data = train.set)
parking = temp$p.value

# laundry
temp = t.test(price ~ laundry, data = train.set)
laundry = temp$p.value

# elevator
temp = t.test(price ~ elevator, data = train.set)
elevator = temp$p.value

# pets_allowed
temp = t.test(price ~ pets_allowed, data = train.set)
pets_allowed = temp$p.value

# metro_area
temp = t.test(price ~ metro_area, data = train.set)
metro_area = temp$p.value
```

```r
# p-value result
temp = data.frame(clubhouse, fireplace, balcony, sport, gym, pool, parking,
                  laundry, elevator, pets_allowed, metro_area) %>%
  mutate(across(is.numeric, ~ round(., 3))) %>%
  t() %>% data.frame() %>% arrange(desc(.))
colnames(temp) = "p-value"
temp
```

```
##              p-value
## laundry        0.955
## gym            0.231
## elevator       0.002
## pets_allowed   0.002
## clubhouse      0.000
## fireplace      0.000
## balcony        0.000
## sport          0.000
## pool           0.000
## parking        0.000
## metro_area     0.000
```

```
qplot(data = train.set, x = bedrooms, y = log(price), size = bathrooms,
      col = log(square_feet), geom = c("jitter")) +
  labs(title = "Scatterplot of Numeric Variable Combination")
```



Data Analysis

1. Model Fitting

a. Multiple Linear Regression

```
# model fit
# all in
mod.a.1 = lm(price ~ ., data = train.set)
sc.a.1 = glance(mod.a.1)

# backward elimination
mod.all = lm(price ~ ., data = train.set)
mod.a.2 = step(mod.all, direction = "backward", trace = F)
sc.a.2 = glance(mod.a.2)

# foward selection
mod.nul = lm(price ~ 1, data = train.set)
mod.a.3 = step(mod.nul, direction = "forward", trace = F,
               scope = list(lower = mod.nul, upper = mod.all))
sc.a.3 = glance(mod.a.3)

# stepwise combination
mod.a.4 = step(mod.all, direction = "both", trace = F)
sc.a.4 = glance(mod.a.4)

# result compare
temp = data.frame("type" = c("all in", "backward elimination",
                             "foward selection", "stepwise combination"),
                  "adj.r.squared" = c(sc.a.1$adj.r.squared, sc.a.2$adj.r.squared,
                                      sc.a.3$adj.r.squared, sc.a.4$adj.r.squared),
                  "aic" = c(sc.a.1$AIC, sc.a.2$AIC, sc.a.3$AIC, sc.a.4$AIC),
                  "bic" = c(sc.a.1$BIC, sc.a.2$BIC, sc.a.3$BIC, sc.a.4$BIC))
temp
```

```
##                   type adj.r.squared    aic    bic
## 1               all in         0.338 127804 127915
## 2 backward elimination         0.338 127799 127890
## 3     foward selection         0.338 127799 127890
## 4 stepwise combination         0.338 127799 127890
```

```
# final model
mod.a = lm(price ~ square_feet + bedrooms + bathrooms + fireplace + sport + gym +
             parking + elevator + pets_allowed + metro_area, data = train.set)
```

b. Polynomial Non Linear Regression

```
# model fit
# polynomial term
mod.b.1 = lm(price ~ square_feet + I(square_feet^2) + bedrooms + bathrooms +
               fireplace + sport + gym + parking + elevator + pets_allowed + metro_area,
             data = train.set)
sc.b.1 = glance(mod.b.1)

# polynomial & interative term
```

```
mod.b.2 = lm(price ~ square_feet + I(square_feet^2) + bedrooms + bathrooms +
               fireplace + sport + gym + parking + elevator + pets_allowed +
               metro_area + square_feet:bedrooms + square_feet:bathrooms +
               square_feet:pets_allowed + square_feet:metro_area + bedrooms:bathrooms,
             data = train.set)
sc.b.2 = glance(mod.b.2)

# result compare
temp = data.frame("type" = c("polynomial", "polynomial & interative"),
                  "adj.r.squared" = c(sc.b.1$adj.r.squared, sc.b.2$adj.r.squared),
                  "aic" = c(sc.b.1$AIC, sc.b.2$AIC),
                  "bic" = c(sc.b.1$BIC, sc.b.2$BIC))
temp
```

```
##                       type adj.r.squared    aic    bic
## 1               polynomial         0.339 127796 127887
## 2 polynomial & interative         0.363 127504 127630
```

```
# final model
mod.b = mod.b.2
```

c. Support Vector Regression

```
# model fit
mod.c = svm(price ~ ., data = train.set, type = "eps-regression", kernel = "linear")
```

d. Kernel Support Vector Regression

```
# model fit
mod.d = svm(price ~ ., data = train.set, type = "eps-regression", kernel = "radial")
```

e. Decision Tree Regression

```
# model fit
mod.e = train(price ~ ., data = train.set, method = "rpart",
              trControl = trainControl(method = "cv", number = 3, verboseIter = F))
```

f. Random Forest Regression

```
# model fit
set.seed(123)
mod.f = train(price ~ ., data = train.set, method = "rf",
              trControl = trainControl(method = "cv", number = 3, verboseIter = F))
```

g. Artificial Neural Network Regression

```
# server connect
h2o.init(nthreads = -1)
```

```
##  Connection successful!
##
## R is connected to the H2O cluster:
##     H2O cluster uptime:         2 hours 42 minutes
```

```
##     H2O cluster timezone:       America/Chicago
##     H2O data parsing timezone:  UTC
##     H2O cluster version:        3.30.0.1
##     H2O cluster version age:    7 months and 20 days !!!
##     H2O cluster name:           H2O_started_from_R_andy_iry509
##     H2O cluster total nodes:    1
##     H2O cluster total memory:   3.94 GB
##     H2O cluster total cores:    8
##     H2O cluster allowed cores:  8
##     H2O cluster healthy:        TRUE
##     H2O Connection ip:          localhost
##     H2O Connection port:        54321
##     H2O Connection proxy:       NA
##     H2O Internal Security:      FALSE
##     H2O API Extensions:         Amazon S3, Algos, AutoML, Core V3, TargetEncoder, Core V4
##     R Version:                  R version 4.0.2 (2020-06-22)
```

```
# model fit
mod.g = h2o.deeplearning(y = "price", training_frame = as.h2o(train.set),
                         activation = "Rectifier", hidden = c(6, 6),
                         epochs = 100, train_samples_per_iteration = -2)
```

```
##   |                                                                              |
##   |                                                                              |
```

```
# server disconect
pred.g = h2o.predict(mod.g, newdata = as.h2o(train.set)) %>% as.vector() %>% as.numeric()
```

```
##   |                                                                              |
##   |                                                                              |
```

```
h2o.shutdown()
```

```
## Are you sure you want to shutdown the H2O instance running at http://localhost:54321/ (Y/N)?
```

2. Model Evaluating

```
# a. multiple linear regression
pred.a = predict(mod.a, newdata = test.set)
error = accuracy(pred.a, test.set$price)
rmse.a = format(round(error[2], 2), nsmall = 2) %>% as.numeric()

# b. polynomial non linear regression
pred.b = predict(mod.b, newdata = test.set)
error = accuracy(pred.b, test.set$price)
rmse.b = format(round(error[2], 2), nsmall = 2) %>% as.numeric()

# c. support vector regression
pred.c = predict(mod.c, newdata = test.set)
error = accuracy(pred.c, test.set$price)
rmse.c = format(round(error[2], 2), nsmall = 2) %>% as.numeric()

# d. kernel support vector regression
pred.d = predict(mod.d, newdata = test.set)
error = accuracy(pred.d, test.set$price)
rmse.d = format(round(error[2], 2), nsmall = 2) %>% as.numeric()
```

```r
# e. decision tree regression
pred.e = predict(mod.e, newdata = test.set)
error = accuracy(pred.e, test.set$price)
rmse.e = format(round(error[2], 2), nsmall = 2) %>% as.numeric()

# f. random forest regression
pred.f = predict(mod.f, newdata = test.set)
error = accuracy(pred.f, test.set$price)
rmse.f = format(round(error[2], 2), nsmall = 2) %>% as.numeric()

# g. artificial neural network regression
error = accuracy(pred.g, test.set$price)
rmse.g = format(round(error[2], 2), nsmall = 2) %>% as.numeric()

# rmse result
temp = data.frame(rmse.a, rmse.b, rmse.c, rmse.d, rmse.e, rmse.f, rmse.g) %>%
  t() %>% as.data.frame()
rownames(temp) = c("MLR", "PNLR", "SVR", "KSVR", "DT", "RF", "ANN")
colnames(temp) = "RMSE"
temp[order(temp$RMSE), , drop = F]
```

```
##      RMSE
## PNLR  705
## MLR   705
## RF    706
## KSVR  717
## SVR   742
## DT    784
## ANN   995
```

Model Inference

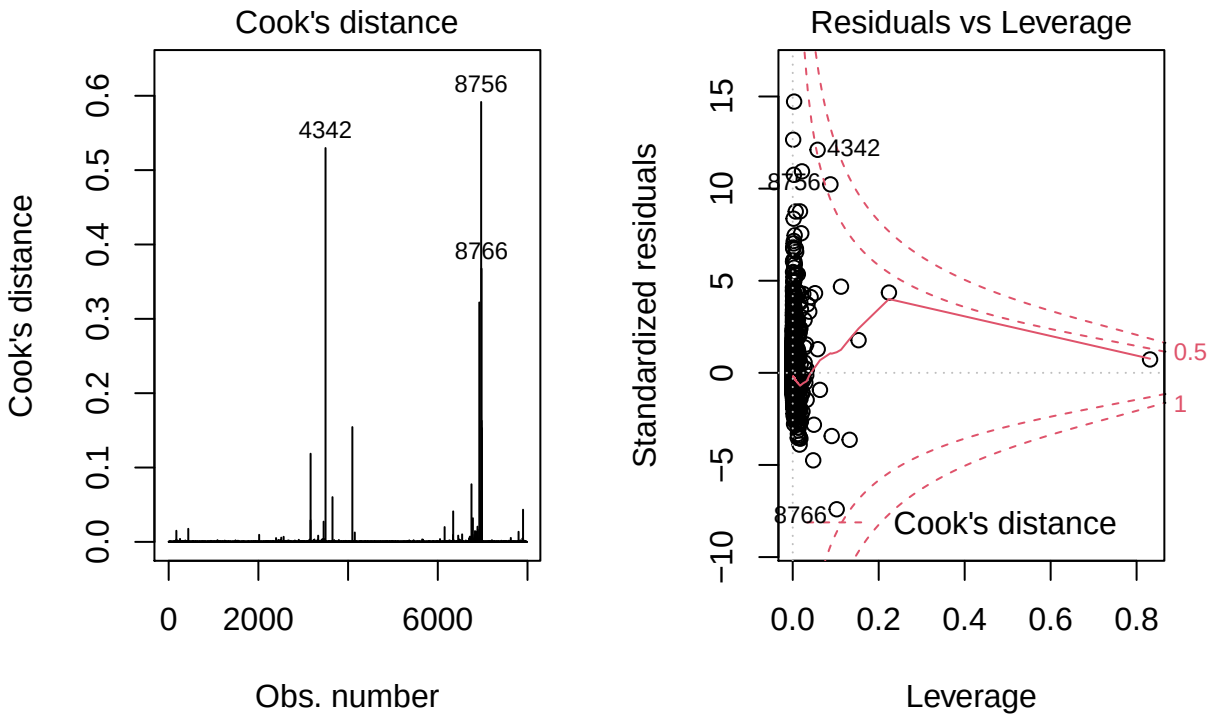
1. Assumption Checking & Remodeling

a. Model Building

```r
# model original
fit.1 = lm(formula = price ~
             square_feet + bedrooms + bathrooms +
             fireplace + sport + gym + parking + elevator + pets_allowed + metro_area +
             I(square_feet^2) +
             square_feet:bedrooms + square_feet:bathrooms +
             square_feet:pets_allowed + square_feet:metro_area + bedrooms:bathrooms,
           data = train.set)
```

b. Assumption Checking

```
# assumption 1 - outlier
par(mfrow = c(1, 2))
plot(fit.1, which = c(4, 5))
```



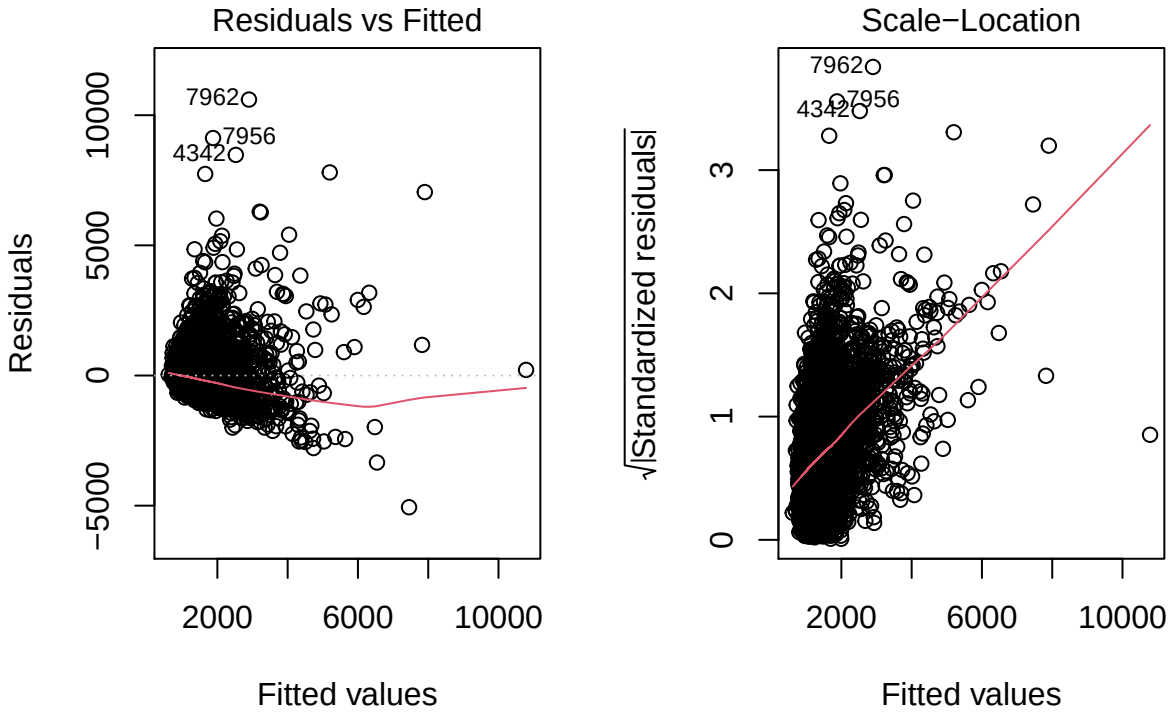

```
# solution
# 1. remove outlier
# 2. transform variable
```

```r
# assumption 2 - collinearity
temp = fit.1 %>% vif() %>% sort(decreasing = T) %>%  data.frame()
colnames(temp) = "VIF"
temp %>%
  mutate("Variable" = row.names(temp)) %>%
  mutate("Collinearity Issue" = ifelse((VIF > 10), "Yes", "No")) %>%
  select(c("Variable", "VIF", "Collinearity Issue"))
```

```
##                    Variable    VIF Collinearity Issue
## 1     square_feet:bathrooms 129.04                Yes
## 2        bedrooms:bathrooms 101.25                Yes
## 3      square_feet:bedrooms  72.19                Yes
## 4          I(square_feet^2)  70.52                Yes
## 5               square_feet  20.04                Yes
## 6                 bathrooms  15.48                Yes
## 7                  bedrooms  10.43                Yes
## 8  square_feet:pets_allowed   5.88                 No
## 9                 metro_area   4.92                 No
## 10   square_feet:metro_area   4.84                 No
## 11             pets_allowed   4.62                 No
## 12                      gym   1.18                 No
## 13                    sport   1.15                 No
## 14                  parking   1.13                 No
## 15                fireplace   1.11                 No
## 16                 elevator   1.08                 No
```

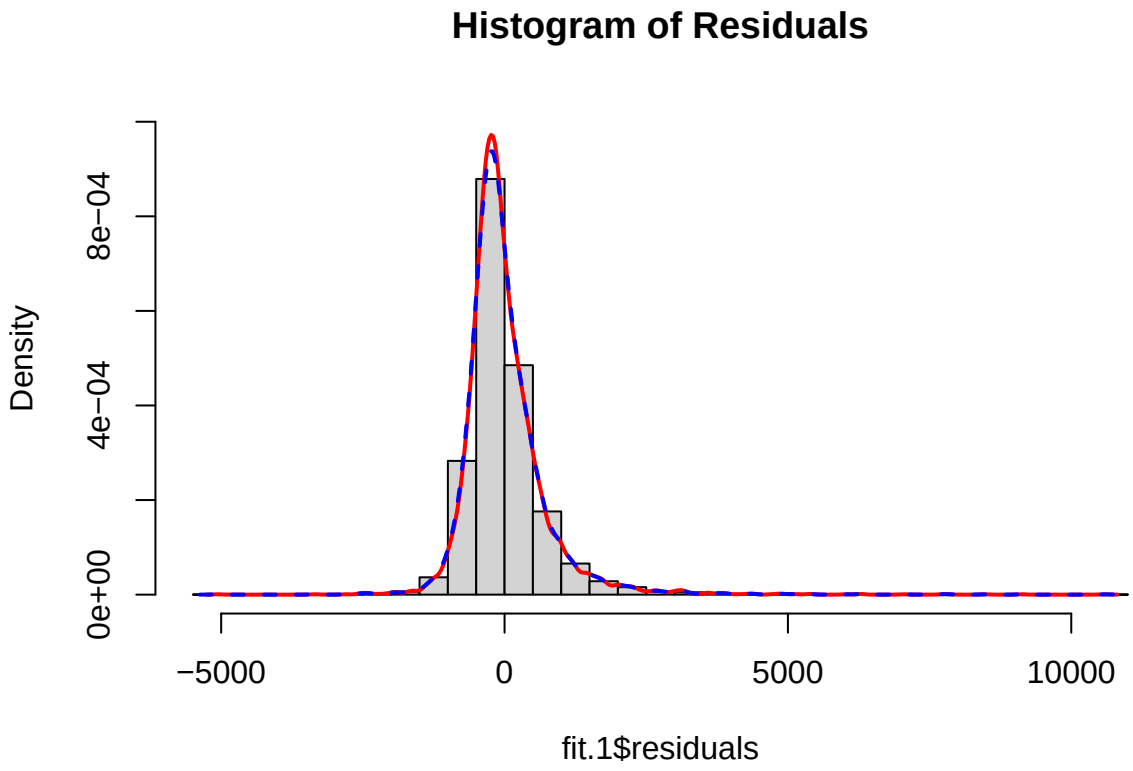
```r
# solution
# 1. combine correlated variable by PCA - no easy to interpretate
# 2. increase data size - impossible now
# 3. remove correlated variable
```

```r
# assumption 3 - heteroscedasticity
par(mfrow = c(1, 2))
plot(fit.1, which = c(1, 3))
```




```r
# solution
# 1. robust SE
# 2. model segmentation
# 3. variable transformation
```

```r
# assumption 4 - normality
par(mfrow = c(1, 1))
hist(fit.1$residuals, probability = T, ylim = c(0, 0.001),
     breaks = 50, main = "Histogram of Residuals")
lines(density(fit.1$residuals), lwd = 2, col = "red") # sample
lines(density(fit.1$residuals, adjust = 1.5), lwd = 2, lty = 2, col = "blue") # normal
```



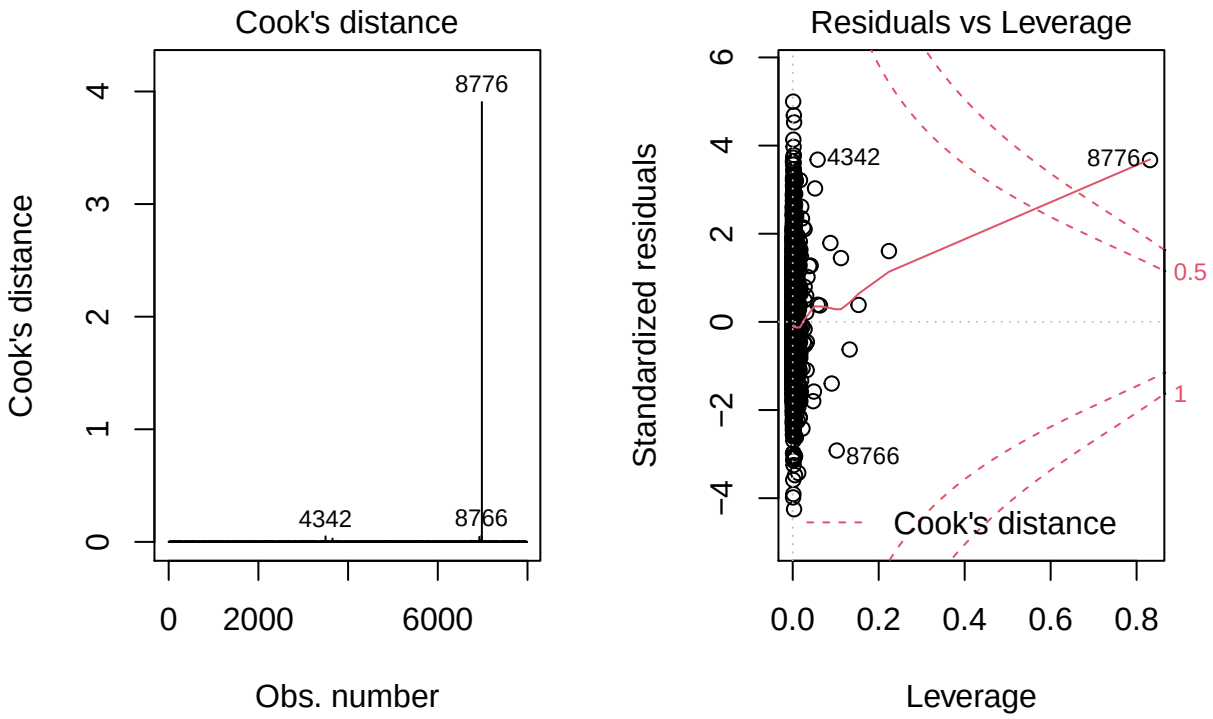

```r
# solution
# 1. remove outlier
# 2. transform variable
```

c. Model Rebuilding

```r
# model transformed
fit.2 = lm(formula = log(price) ~
             square_feet + bedrooms + bathrooms +
             fireplace + sport + gym + parking + elevator + pets_allowed + metro_area +
             I(square_feet^2) +
             square_feet:bedrooms + square_feet:bathrooms +
             square_feet:pets_allowed + square_feet:metro_area + bedrooms:bathrooms,
           data = train.set)
```

d. Assumption Rechecking

```
# assumption 1 - outlier
par(mfrow = c(1, 2))
plot(fit.2, which = c(4, 5))
```



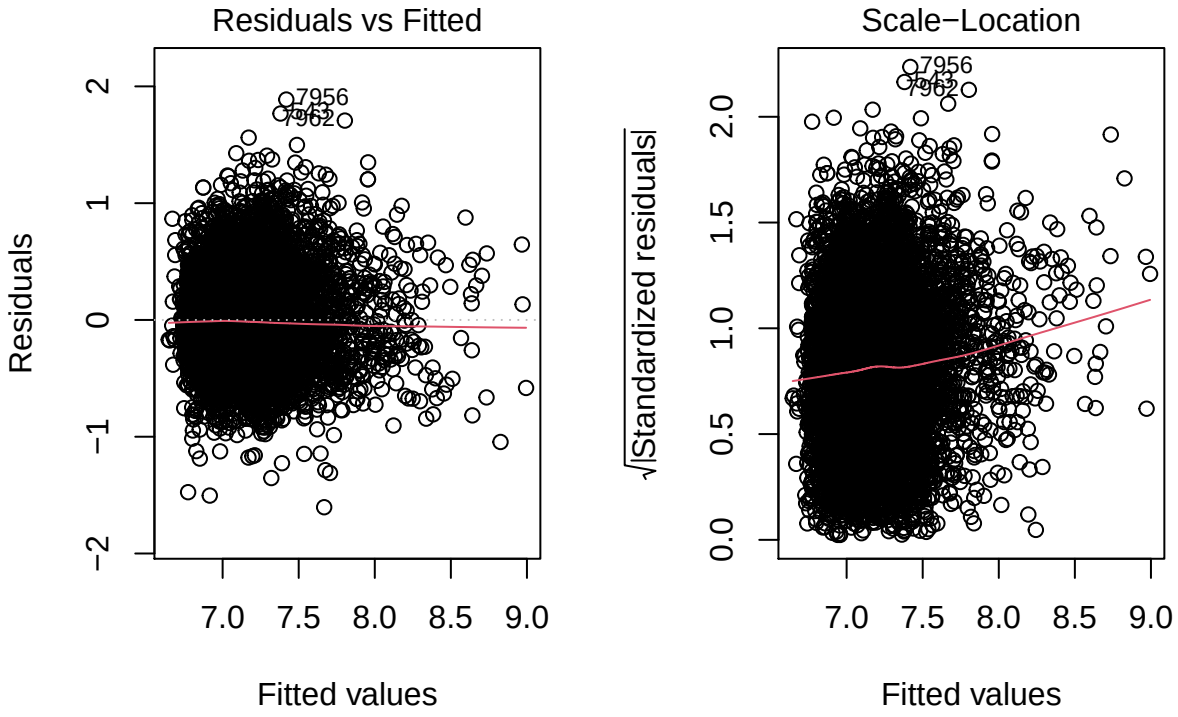

```
# solution
# 1. remove outlier - implement
# 2. transform variable - implement
```

```r
# assumption 2 - collinearity
temp = fit.2 %>% vif() %>% sort(decreasing = T) %>% data.frame()
colnames(temp) = "VIF"
temp %>%
  mutate("Variable" = row.names(temp)) %>%
  mutate("Collinearity Issue" = ifelse((VIF > 10), "Yes", "No")) %>%
  select(c("Variable", "VIF", "Collinearity Issue"))
```

```
##                    Variable    VIF Collinearity Issue
## 1     square_feet:bathrooms 129.04                Yes
## 2        bedrooms:bathrooms 101.25                Yes
## 3      square_feet:bedrooms  72.19                Yes
## 4          I(square_feet^2)  70.52                Yes
## 5               square_feet  20.04                Yes
## 6                 bathrooms  15.48                Yes
## 7                  bedrooms  10.43                Yes
## 8  square_feet:pets_allowed   5.88                 No
## 9                 metro_area   4.92                 No
## 10   square_feet:metro_area   4.84                 No
## 11             pets_allowed   4.62                 No
## 12                      gym   1.18                 No
## 13                    sport   1.15                 No
## 14                  parking   1.13                 No
## 15                fireplace   1.11                 No
## 16                 elevator   1.08                 No
```

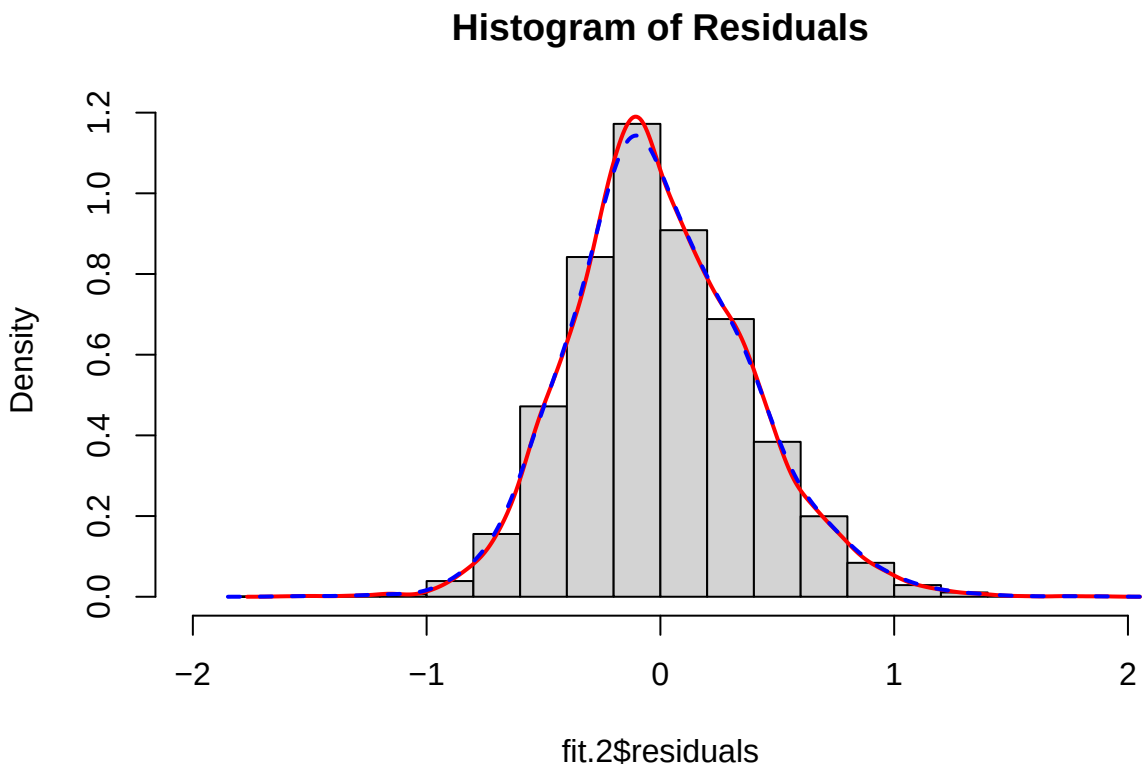
```r
# solution
# 1. combine correlated variable by PCA - no easy to interpretate
# 2. increase data size - impossible now
# 3. remove correlated variable - implement
```

```r
# assumption 3 - heteroscedasticity
par(mfrow = c(1, 2))
plot(fit.2, which = c(1, 3))
```




```r
# solution
# 1. robust SE
# 2. model segmentation
# 3. variable transformation - implement
```

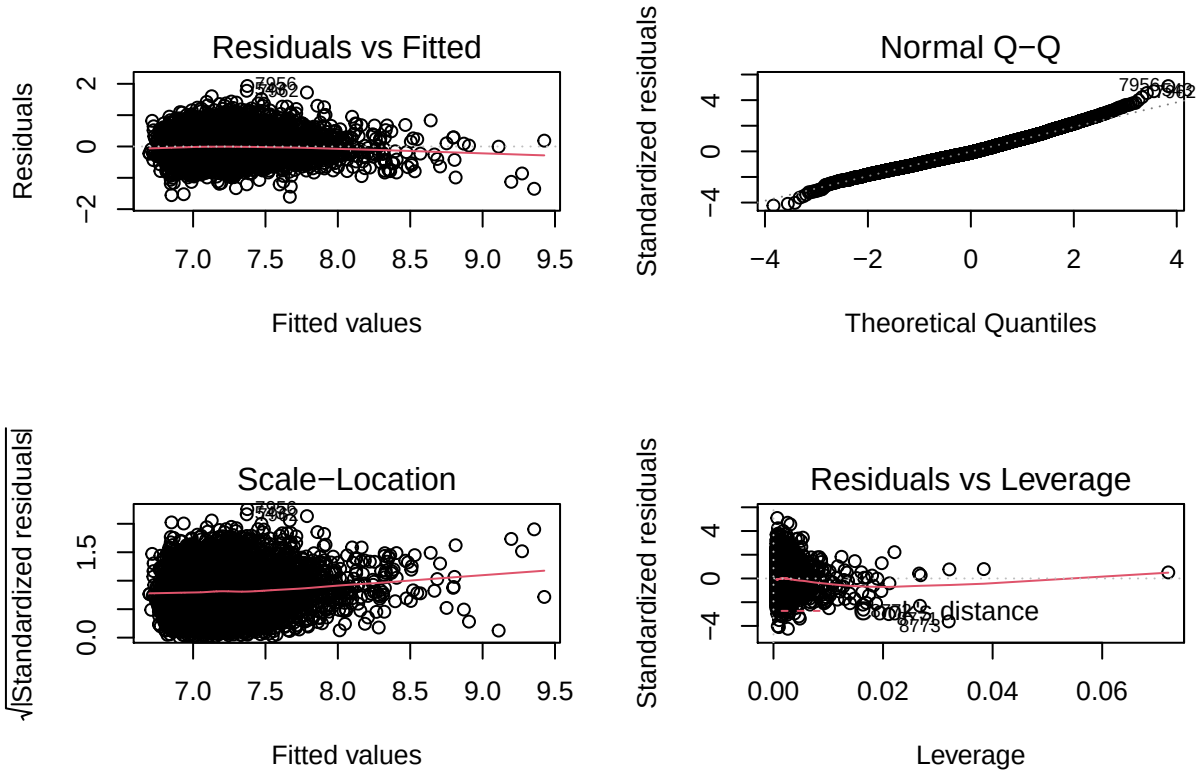
```r
# assumption 4 - normality
par(mfrow = c(1, 1))
hist(fit.2$residuals, probability = T, xlim = c(-2, 2), main = "Histogram of Residuals")
lines(density(fit.2$residuals), lwd = 2, col = "red") # sample
lines(density(fit.2$residuals, adjust = 1.5), lwd = 2, lty = 2, col = "blue") # normal
```




```r
# solution
# 1. remove outlier - implement
# 2. transform variable - implement
```

e. Model Tuning

```
# outlier remove
remove = c("8776", "8766", "4342",
           "4550", "8769", "8701")
train.set = train.set[row.names(train.set) %nin% remove, ]

# model rebuild
fit.3 = lm(formula = log(price) ~
             square_feet + bedrooms + bathrooms +
             fireplace + sport + gym + parking + elevator + pets_allowed + metro_area +
             square_feet:pets_allowed + square_feet:metro_area,
           data = train.set)

# assumption: outlier (fix), heteroscedasticity (fix), normality (fix)
par(mfrow = c(2, 2)); plot(fit.3)
```



```r
# assumption: collinearity (fix)
temp = fit.3 %>% vif() %>% sort(decreasing = T) %>% data.frame()
colnames(temp) = "VIF"
temp %>%
  mutate("Variable" = row.names(temp)) %>%
  mutate("Collinearity Issue" = ifelse((VIF > 10), "Yes", "No")) %>%
  select(c("Variable", "VIF", "Collinearity Issue"))
```

```
##                    Variable  VIF Collinearity Issue
## 1  square_feet:pets_allowed 5.76                 No
## 2                metro_area 5.13                 No
## 3    square_feet:metro_area 5.05                 No
## 4              pets_allowed 4.62                 No
## 5               square_feet 4.28                 No
## 6                 bathrooms 2.77                 No
## 7                  bedrooms 2.51                 No
## 8                       gym 1.17                 No
## 9                     sport 1.15                 No
## 10                  parking 1.13                 No
## 11                fireplace 1.11                 No
## 12                 elevator 1.08                 No
```

2. Causality Establishing

```
temp = summary(fit.3)
coef = temp$coefficients %>%
  data.frame(confint(fit.3, level = 0.95)) %>%
  mutate("Variable" = row.names(temp$coefficients))
colnames(coef) = c("estimate", "SE", "t-value", "p-value", "lcl", "ucl", "Variable")
coef %>%
  mutate(across(is.numeric, ~ round(., 4))) %>%
  select(c("estimate", "lcl", "ucl", "p-value", "Variable"))
```

```
##    estimate     lcl     ucl p-value                 Variable
## 1    6.6388  6.6082  6.6694  0.0000              (Intercept)
## 2    0.0004  0.0003  0.0004  0.0000              square_feet
## 3    0.0021 -0.0120  0.0163  0.7673                 bedrooms
## 4    0.1152  0.0919  0.1385  0.0000                bathrooms
## 5   -0.0368 -0.0653 -0.0083  0.0115               fireplace1
## 6   -0.1351 -0.1623 -0.1080  0.0000                   sport1
## 7    0.1167  0.0915  0.1420  0.0000                     gym1
## 8   -0.0437 -0.0620 -0.0255  0.0000                 parking1
## 9    0.2231  0.1881  0.2581  0.0000                elevator1
## 10   0.0301 -0.0062  0.0664  0.1042            pets_allowed1
## 11   0.2304  0.1742  0.2866  0.0000              metro_area1
## 12  -0.0001 -0.0001  0.0000  0.0001 square_feet:pets_allowed1
## 13   0.0001  0.0001  0.0002  0.0000   square_feet:metro_area1
```

3. Prediction Using

```
# title: 1B1B apartment in Richardson TX $1340/month
# link: https://tinyurl.com/y59wtrzx
# ref: Sublet.com
rentee = data.frame(square_feet = 808, bedrooms = 1, bathrooms = 1,
                    fireplace = factor(0), sport = factor(0), gym = factor(1),
                    parking = factor(1), elevator = factor(0), pets_allowed = factor(1),
                    metro_area = factor(0))
pred.rentee = predict(fit.3, newdata = rentee)
data.frame("Actual" = 1340,
           "Predict" = round(exp(pred.rentee), 0)) %>%
  mutate("Description" = "1B1B apartment in Richardson TX")
```

```
##   Actual Predict                    Description
## 1   1340    1225 1B1B apartment in Richardson TX
```