final-project

Pragyat Agrawal Ruxuan Ji Meng-Chuan Chang

Contents

1 Executive Summary

1.1 Background

The US rental market has been growing rapidly growing over time, making it one of the most sought after areas for investment. Be it from small home owners to big private equity firms, everyone seems to be after housing, expecting the values of these houses to rise and supplement their income by renting these places. Many people consider that location is the "only important" factor responsible for a house's value and the rent that can be expected of it, but this is far from the truth. There are a lot of other factors that need to be considered for determining housing valuations as we see a huge variation in prices in houses located in the same vicinity. There must be something about these houses which is causing such a big price change. Hence we will be analyzing the data related to US Rental Listings in Summer of 2021, to find which of these factors, which consist of many in-house amenity components, impacts housing values the most.

These amenities range from simple aka micro-factors like the availability of Pools and Dishwashers in the house to major aka macro-factors i.e. cities. The data will also give us the opportunity to find in which cities are these factors playing the most impact. With over 27,000 values for each predictor in our data, we have a sufficient sample size to make a reasonable conclusion regarding the price of these summer rentals.

Graph: Rental vacancy rates in the United States from 2000 to 2021, by region (Source: Statista.com)

This shows how the US housing market has been more sought after year by year, making housing a form of valuable investment. This increase in demand has already pushed up the prices.

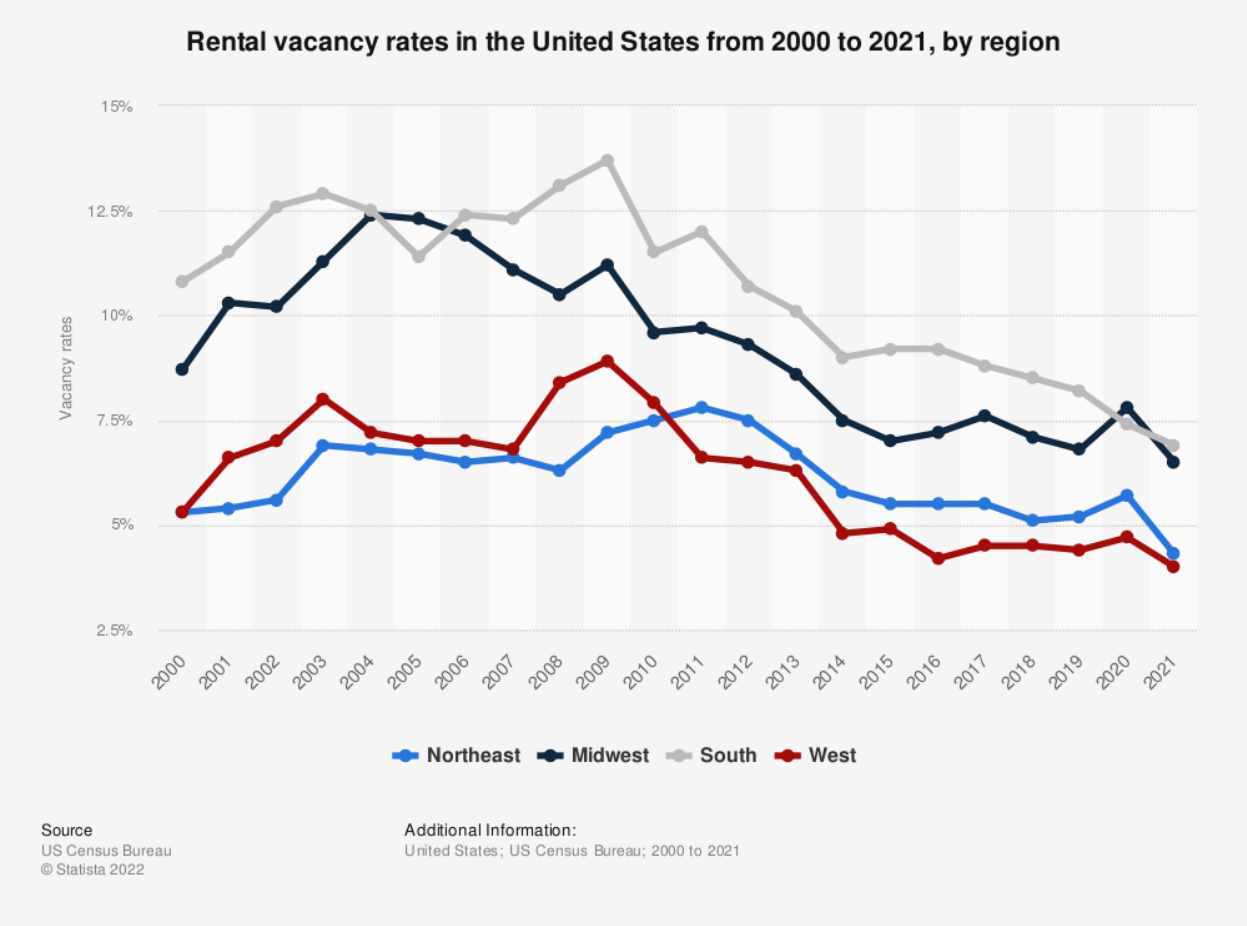


Figure 1: A caption

1.2 Description of Data

The data is gathered from Kaggle, a huge repository of community published code and data. This data was pulled from Rentler.com on 7/12/2021, 8/12/2021, and 9/6/2021, and population density data was scraped by zip code from mapszipcode.com on 7/12/2021. The pull from Rentler.com resulted in 4 CSV files which included the main rental listing, the list of amenities, the list of lease terms, and a list of who was responsible to pay each utility. Many of the variables that were sparsely populated were dropped before denormalizing the dataset. The rental listing information was joined with the population and population density information from mapszipcode.com (Source: Kaggle). Many of the data columns in the dataset are embedded in binary format with 0 representing the absence of the predictor attribute while 1, shows that the attribute is present. The data file is massive at around half a Gigabyte. Working with this size of data, we would have to use EDA or take a subset of the dataset for R to run effectively and not crash over the large size of the data. We will explore this idea in later sections. For now, the data is sufficient for analysis.

Our response variable is PRICE which represents the monthly price for the particular summer listing on rental.com.

Table 1: Variables and their descriptions

Variable	Description
V1	Numbering for the house number we are looking at (form of house identification)
pool	Binary data to show if pool exists in this house (1) or not (0)
dishwasher	Binary data to show if dishwasher exists in the house (1) or not (0)
washer-dryer	Binary data to show if washer-dryer exists in the house(1) or not (0)
ac	Binary data to show if air conditioning exists in the house (1) or not (0)
parking	Binary data to show if Parking exists in the house (1) or not (0)
zip	Zip code of the property
price	Monthly rent price for the property
city	City where the property is located
num_beds	Number of beds in the property
num_baths	Number of baths in the property
house_type	Type of house we are looking at
sqft	Square Feet in the property
smoking_ind	Does the rental allow smoking (Yes/No)
pets_ind	Does the rental allow pets (Yes/No)
acres	Number of acres rental includes
description	4000 character listing description of the rental
ZipCity	Primary city for the zip code
Population	Population in the zip code
PopulationDensity	Population density per square mile for zip code
security_deposit	Security deposit required

1.3 Goal

The goal of this study is to analyze the most important factors affecting the housing prices in the US. We will be using the variables in the dataset to do so. In our preliminary market analysis, we found that housing prices are determined by many factors and hence we will try to ascertain factors which have a significant impact on housing prices. Our goal is to build a model that will give us the value added or subtracted from a house with/without the presence of a variable factor. This observation will benefit people who are looking to rent properties in the US and can help them get a better value for the kind of place they may be looking for.

1.4 Summary of Findings

1.5 Issues and Limitations

The biggest issue we initially faced was with respect to the file size which turned out to be quiet massive even for R Studio. The raw data we started with was half a gigabyte big which turned out to be very massive for any for of extrapolation. Hence we had to shorten the data out.....

2 Exploratory Data Analysis

2.1 Data Preprocessing/Cleaning

2.1.1 Read the data

Firstly, we read the data from Kaggle - US Rental Listings Summer 2021

Then we read SOI Tax statistics in 2019 from IRS

Before joining those two dataset, we would use groupby to find the income level based on zip code

Here, `AGI_STUB` shows the level of adjusted gross income, and `N1` shows the number of returns of each level.

The following shows the level and corresponding income range

AGI_STUB	Range
1	$1~25,000
2	$25,000~$50,000
3	$50,000~$75,000
4	$75,000~$100,000
5	$100,000~$200,000
6	$200,000~

Based on the above two columns, we generate a new column `avg_level` from 1 to 6, showing the average level of gross income in each zip code region

2.1.2 Filter the data

The original dataset contains 276757 data, but we just need partial data. Before we randomly pick 20000 for further analysis, we can remove rows that is lack of important factors. The criteria is as follows

- sqft (squart feet) must be non-zero
- population and the density must be non-zero
- price must be non-zero

Then we drop several columns which is clearly not helpful for predicting the rental price

- link
- street_address
- full_address
- acres
- description

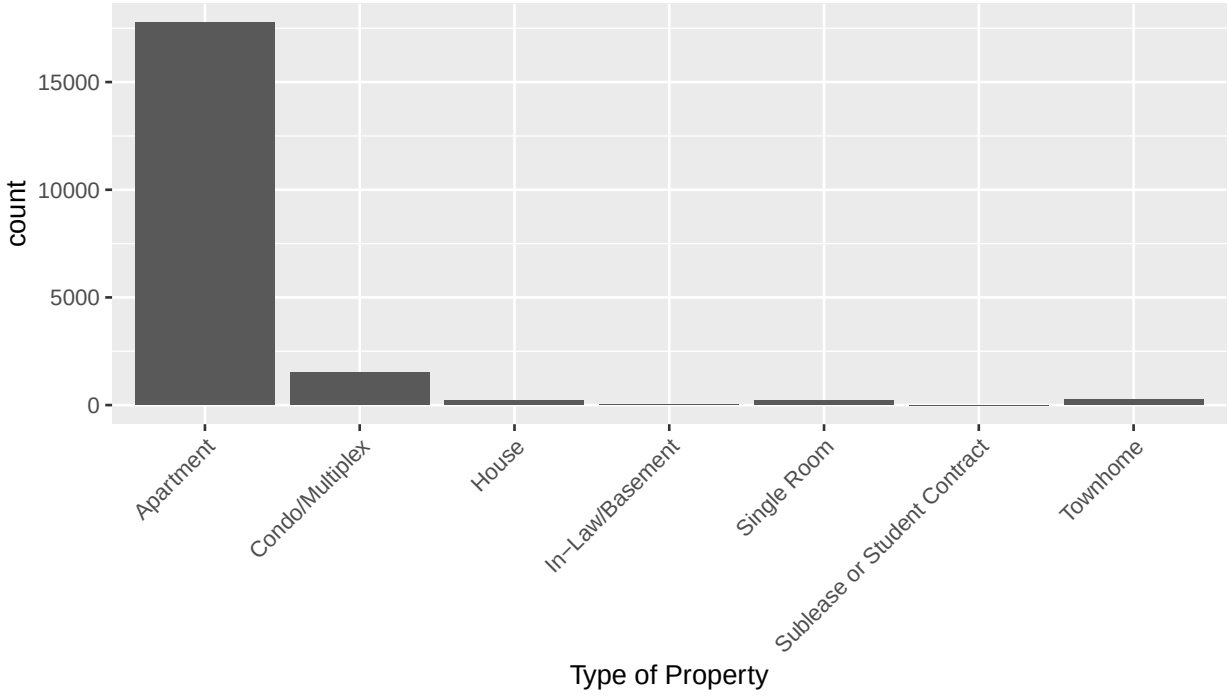
- zip city (duplicated data)

Finally, we fill all NA with 0. The columns having NA is as follows

- pool
- dishwasher
- washer-dryer
- ac
- parking

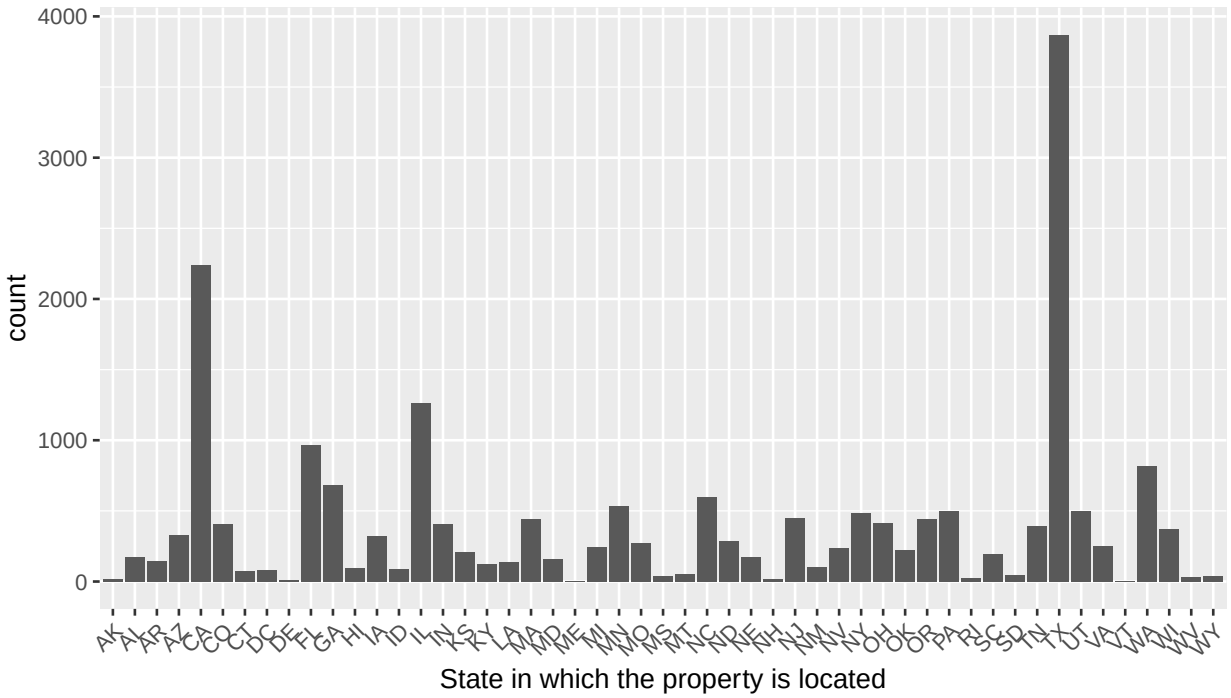
Then we export the cleaned dataframe to csv

2.2 Data Transformations and Plots

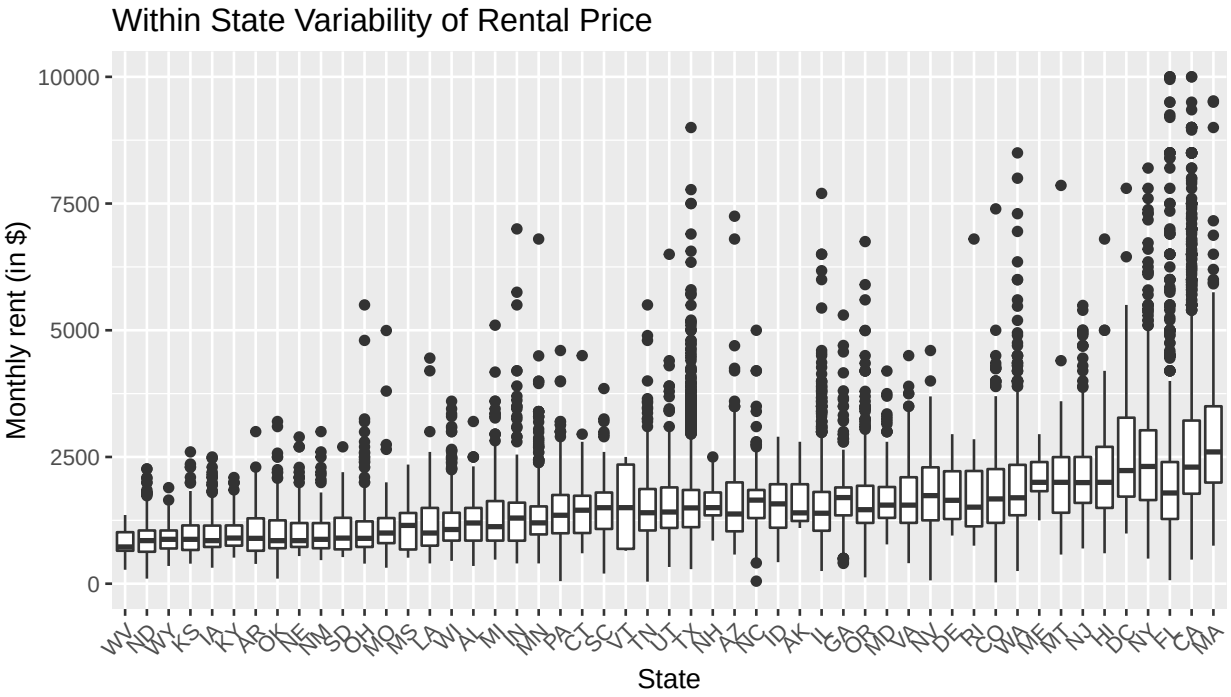


house_type n 1: Apartment 17749 2: Condo/Multiplex 1519 3: House 219 4: In-Law/Basement 26 5: Single Room 211 6: Sublease or Student Contract 1 7: Townhome 275

Looking at the data we see that the properties we will most be evaluating will be Apartment style places with 17749 observations. Other places are lesser in number but still there. The second largest group is the Condo/Multiplex group that we are looking at with 1519 observations. Other than that the smallest group we see is the sublease or student contract group which only has 1 observation.



The data represents all states, some more than others. Texas is the most represented state with the least being Vermont at 4 listings. The sample is representative of all states in the US. We are randomly choosing 20,000 data points so this will fluctuate if we change the data seed.



We see the prices remaining fairly same for the start with the prices rising with states like New York, California, Florida and Massachusetts. This is evidently true for these "more expensive" places as states on

the right end of the spectrum like California, and Massachusetts are known for their high per capita incomes which tends to push housing prices up. There is a lot of variation in data as well as we move to states with higher housing prices, indicated through the presence of excessive outliers on the right end of the boxplot.

3 Model Training

To do the further model analysis, we transform some of the data. Firstly, convert the following columns from characters to binary data

- house_type
- smoking_ind
- pets_ind
- state

Also, we need to predict the rental price which is continuous. In order to implement random forest, we transform the price to binary outcome, with the threshold of median of rental price, which is (1500).

Unlike state, the amount of unique city is far more than the state. In our sample data, we find there are 2395 unique cities, more than 1/10 of the observation. Therefore, we choose to discard this factor for our further model analysis.

4 Model Training

Firstly, we split the data for choosing and validating and model

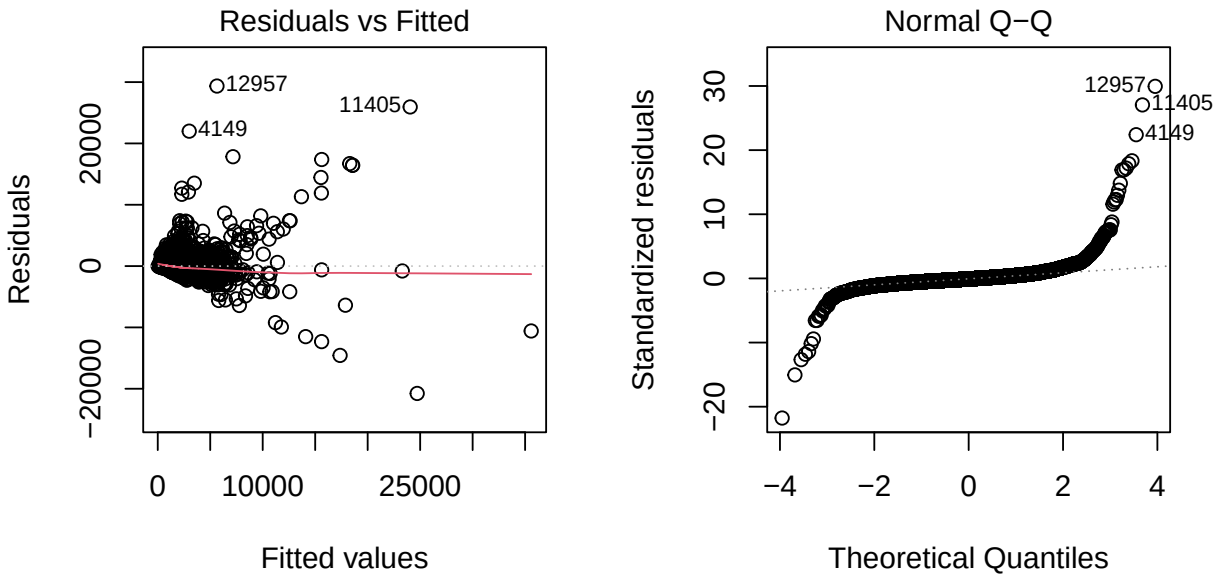
- train: 13000
- test: 5000
- validation: the rest

4.1 Linear Regression

4.1.1 Normal Linear Regression

In this part, we fit the linear regression model `price` vs. other variables as `fit1` directly, then select significant variables to fit the final fit, `fit2`.

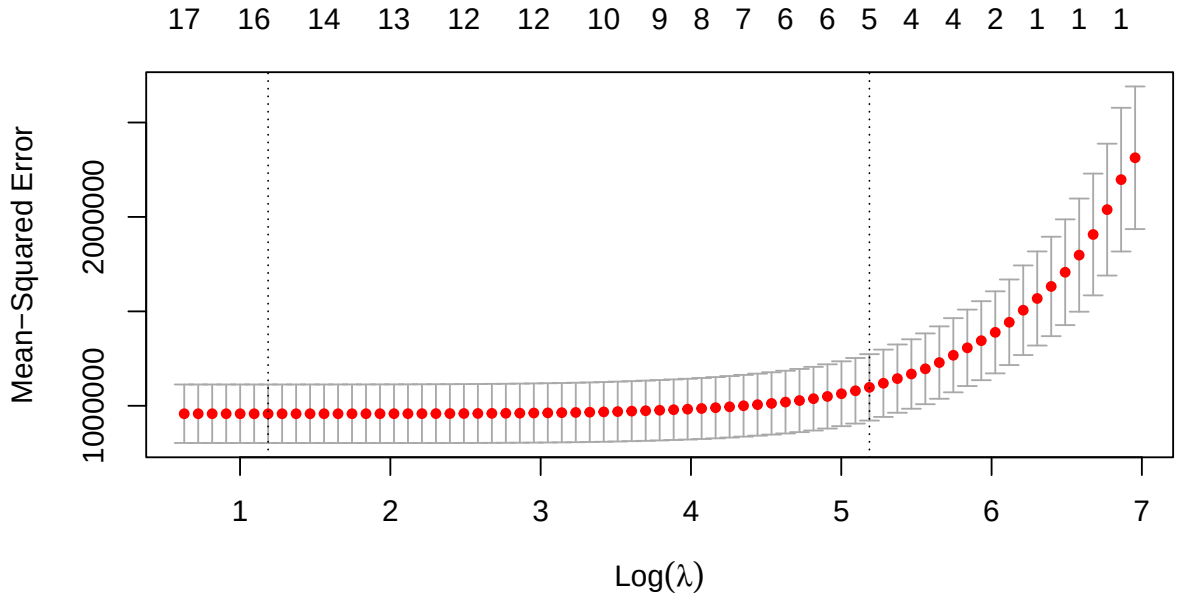
1. Since the `description` and `price` features are useless for linear model, we drop them.
2. From the summary of fit1, we select pool, dishwasher, washer-dryer, ac, parking, state, num_beds, num_baths, house_type, sqft, Population, population density and security deposit as the variables of our final model. All the variables are significant at 0.001. The result of this model is showed as follow.
3. Training MSE: 965083.3, Testing MSE: 1304949



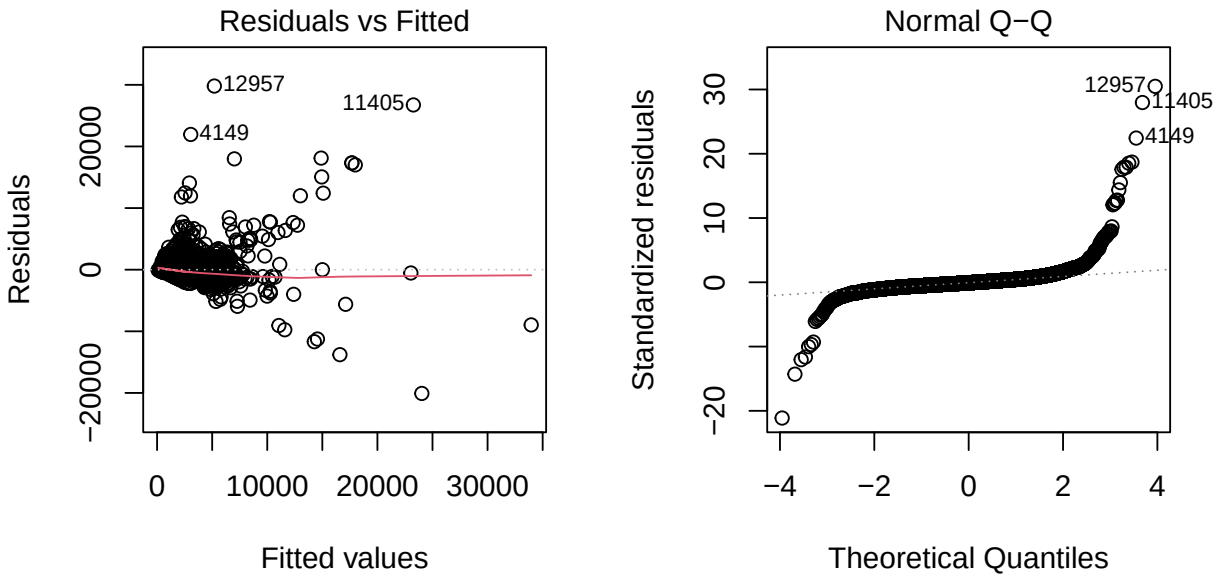
4.1.2 LASSO Model

In this part, we introduce LASSO for a linear regression model with fewer variable since in many case not all the variables are useful for a model. So using lasso can help to reduce computation load significantly with maintaining a good result. We build the model with lasso following 3 steps.

1. Description and price_binary column are drop since they make no sense for this model.
2. `price` is set to be X and the other column of training data are set to be Y, and they are used to fit the first model, `model1`, with alpha = 1 and nfolds = 10.
3. We use LASSO to select several useful variables and refit the linear regression model, named `model2_lasso`, finding that all the vaiables are significant at 0.001, including dishwasher, number of baths, sqft, population density and security deposit.



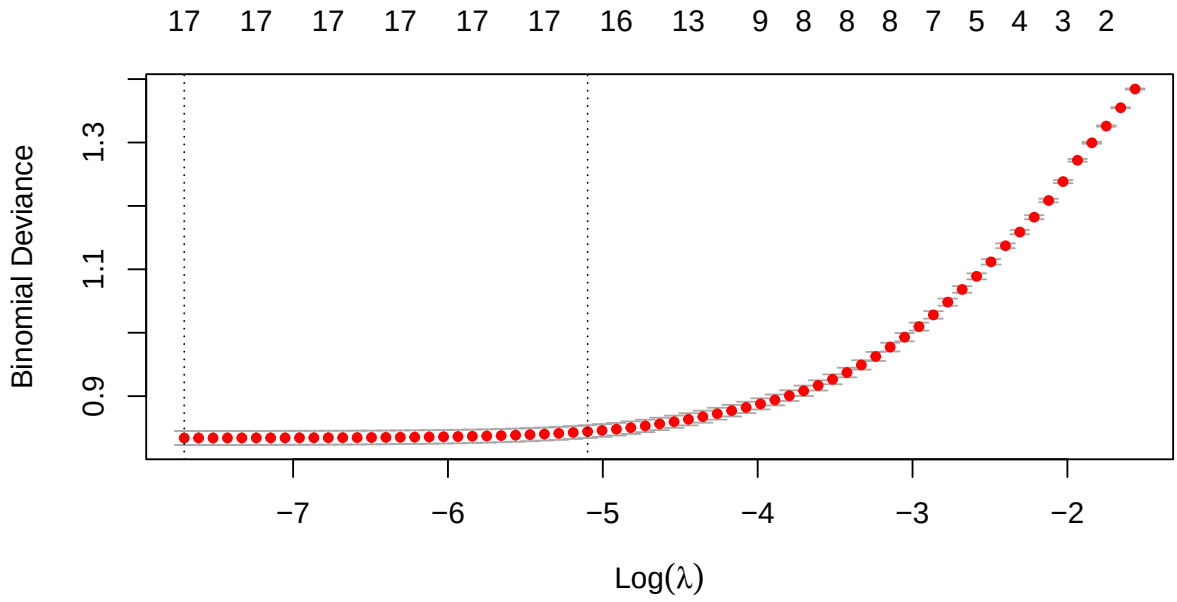

```
##
## Call:
## lm(formula = price ~ ., data = lasso_sub)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -20083.0   -376.4    -78.6    261.9  29802.6
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)       -9.588e+02  4.625e+01  -20.73   <2e-16 ***
## num_baths          2.285e+02  1.515e+01   15.08   <2e-16 ***
## sqft               5.328e-01  2.021e-02   26.36   <2e-16 ***
## PopulationDensity  1.949e-02  7.999e-04   24.37   <2e-16 ***
## security_deposit   3.987e-01  4.514e-03   88.31   <2e-16 ***
## avg_level          4.490e+02  1.584e+01   28.35   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 978.7 on 12994 degrees of freedom
## Multiple R-squared:  0.5895, Adjusted R-squared:  0.5894
## F-statistic:  3733 on 5 and 12994 DF,  p-value: < 2.2e-16
```



4.2 Logistic Regression

After using linear regression models to create the predictors for specific price of rental, we use logistic regression model to predict whether one rental house or department's price is higher or lower than the median. In this part, price_binary that we create early is used as the result for prediction, and we build the model following 4 steps.

1. Description and price column are drop since they make no sense for this model.
2. Price_binary is set to be X and the other column of training data are set to be Y, and they are used to fit the first model, `fit.log.cv`, with alpha = 1 and nfolds = 10.
3. We use LASSO to select several useful variables and refit the logistic regression model, named `fit.logit2`, finding that `num_beds` is not significant in this model. So we drop this column and refit the model again, named `fit.logit.final`. 4.Finally, the logistic regression model includes pool, dishwasher, washer_dryer, ac, parking, state, num_baths, house_type, sqft, smoking_ind, pets_ind, Population, PopulationDensity and security_deposit. All of the variables are significant at 0.01. Among those, ac, dishwasher and Pool play relatively important roles when the prices are setting.




```
##
## Call:
## glm(formula = Y ~ pool + dishwasher + washer_dryer + ac + parking +
##     state + num_beds + num_baths + house_type + sqft + smoking_ind +
##     pets_ind + Population + PopulationDensity + security_deposit,
##     family = binomial, data = data.log.train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -6.1703  -0.7245   0.0126   0.7288   2.5192
##
## Coefficients:
##                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)       -1.525e+00  7.610e-01  -2.004 0.045066 *
## pool               6.183e-01  7.499e-02   8.245  < 2e-16 ***
## dishwasher         8.944e-01  6.058e-02  14.765  < 2e-16 ***
## washer_dryer      -2.139e-01  6.138e-02  -3.485 0.000491 ***
## ac                -5.287e-01  7.541e-02  -7.011 2.37e-12 ***
## parking            4.752e-01  8.290e-02   5.732 9.92e-09 ***
## state             -1.423e-02  1.619e-03  -8.791  < 2e-16 ***
## num_beds           4.286e-02  3.979e-02   1.077 0.281448
## num_baths          5.964e-01  5.212e-02  11.444  < 2e-16 ***
## house_type        -1.413e-01  3.197e-02  -4.421 9.81e-06 ***
## sqft               2.234e-03  1.109e-04  20.152  < 2e-16 ***
## smoking_ind       -2.206e+00  7.519e-01  -2.933 0.003353 **
## pets_ind          -2.317e-01  5.155e-02  -4.496 6.93e-06 ***
## Population        -4.244e-06  1.359e-06  -3.122 0.001798 **
## PopulationDensity  9.671e-05  3.845e-06  25.152  < 2e-16 ***
## security_deposit   7.949e-04  3.089e-05  25.732  < 2e-16 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 18010  on 12999  degrees of freedom
## Residual deviance: 11853  on 12984  degrees of freedom
## AIC: 11885
##
## Number of Fisher Scoring iterations: 6

##
## Call:
## glm(formula = Y ~ pool + dishwasher + washer_dryer + ac + parking +
##     state + num_baths + house_type + sqft + smoking_ind + pets_ind +
##     Population + PopulationDensity + security_deposit, family = binomial,
##     data = data.log.train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -6.2670  -0.7247   0.0125   0.7310   2.5167
##
## Coefficients:
##                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)       -1.501e+00  7.648e-01  -1.963 0.049657 *
## pool               6.095e-01  7.454e-02   8.177 2.91e-16 ***
## dishwasher         8.934e-01  6.055e-02  14.754  < 2e-16 ***
## washer_dryer      -2.161e-01  6.133e-02  -3.523 0.000426 ***
## ac                -5.273e-01  7.538e-02  -6.994 2.67e-12 ***
## parking            4.722e-01  8.285e-02   5.699 1.20e-08 ***
## state             -1.425e-02  1.619e-03  -8.807  < 2e-16 ***
## num_baths          6.064e-01  5.137e-02  11.805  < 2e-16 ***
## house_type        -1.415e-01  3.201e-02  -4.421 9.81e-06 ***
## sqft               2.305e-03  8.957e-05  25.741  < 2e-16 ***
## smoking_ind       -2.228e+00  7.560e-01  -2.947 0.003212 **
## pets_ind          -2.335e-01  5.152e-02  -4.533 5.82e-06 ***
## Population        -4.137e-06  1.355e-06  -3.053 0.002268 **
## PopulationDensity  9.600e-05  3.781e-06  25.390  < 2e-16 ***
## security_deposit   7.967e-04  3.086e-05  25.818  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 18010  on 12999  degrees of freedom
## Residual deviance: 11854  on 12985  degrees of freedom
## AIC: 11884
##
## Number of Fisher Scoring iterations: 6

## Analysis of Deviance Table (Type II tests)
##
## Response: Y
##                   LR Chisq Df Pr(>Chisq)
## pool                 66.84  1  2.941e-16 ***
```

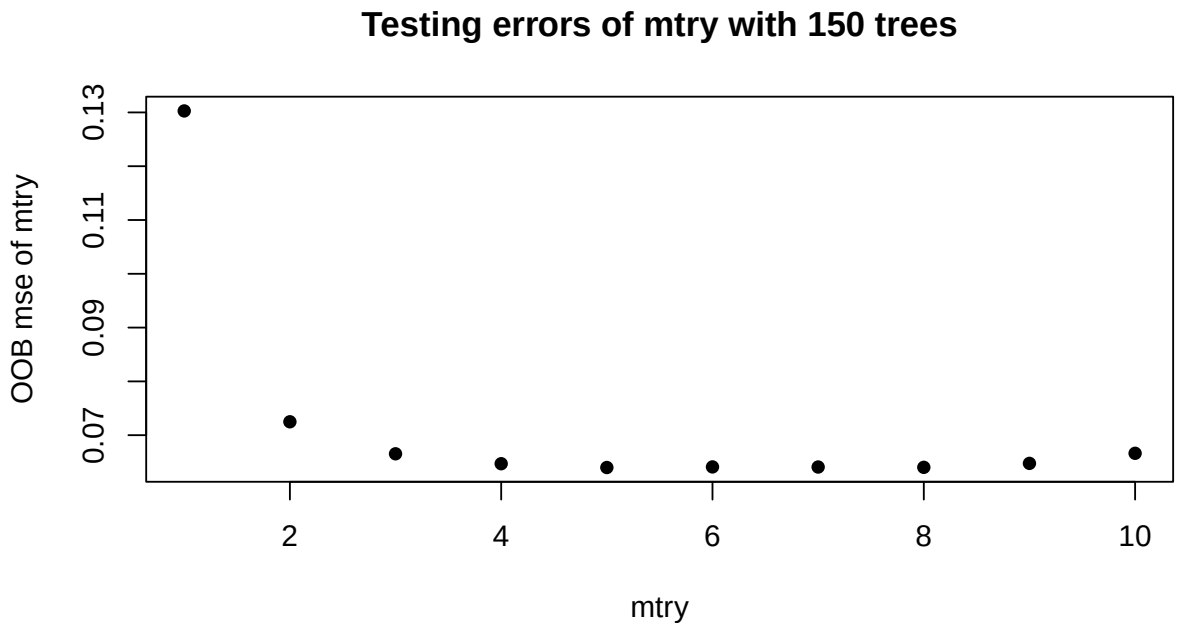
```
## dishwasher          224.10  1  < 2.2e-16 ***
## washer_dryer         12.48  1  0.0004108 ***
## ac                   49.74  1  1.755e-12 ***
## parking              32.79  1  1.025e-08 ***
## state                78.25  1  < 2.2e-16 ***
## num_baths           142.87  1  < 2.2e-16 ***
## house_type           19.23  1  1.160e-05 ***
## sqft                831.98  1  < 2.2e-16 ***
## smoking_ind          12.04  1  0.0005206 ***
## pets_ind             20.51  1  5.919e-06 ***
## Population            9.34  1  0.0022421 **
## PopulationDensity   900.00  1  < 2.2e-16 ***
## security_deposit    814.34  1  < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

4.3 Random Forest

In random forest model, we just need the binary data of the price, so we remove `price` column, and we need to rename the column `washer-dryer` for the random forest package

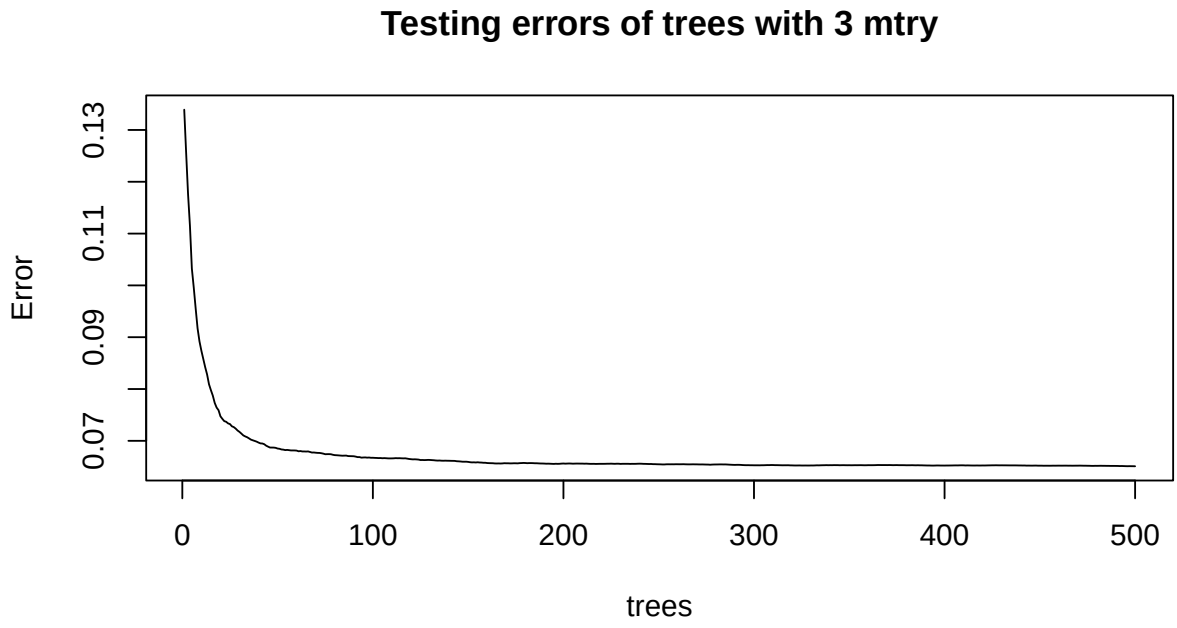
4.3.1 Tune Hyperparameter

Firstly, we use OOB to find the testing error for given parameter, and we choose mtry from 1 to 10, with 150 tree.



According to the above plot, we choose 3 as mtry based on elbow rule.

Then we set ntree to be 500, to find the optimal number of tree.



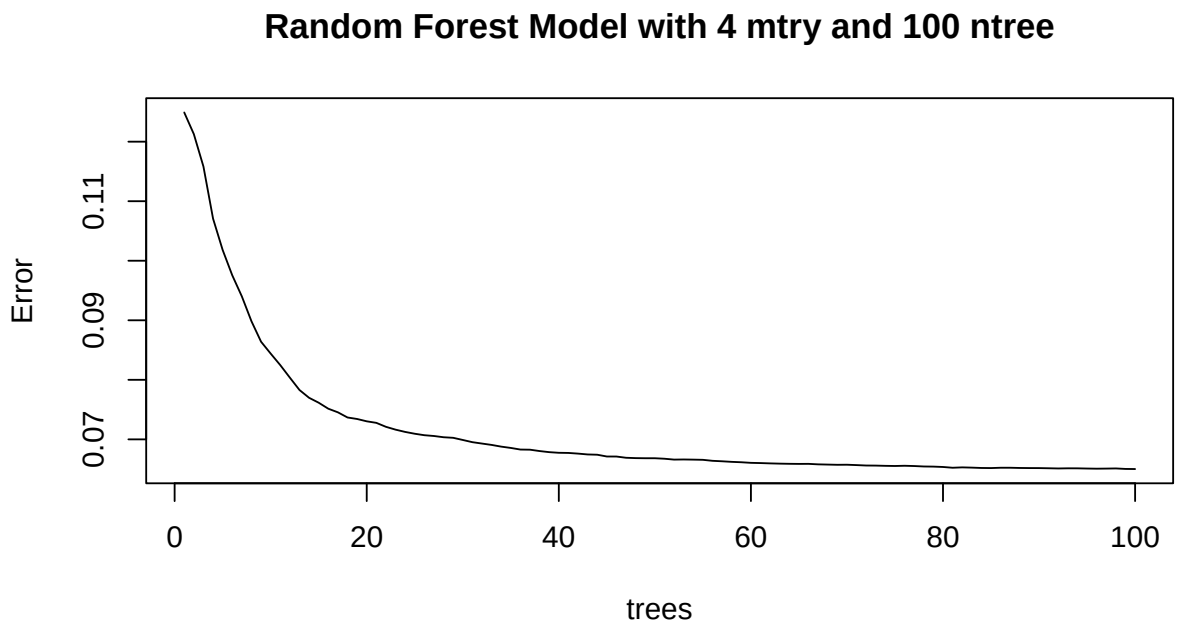
Also based on the above plot, we choose 100 as ntree

4.3.2 PCA

Apply PCA to dataset would help us to remove correlated data and speed up the runtime of the program. However, in this analysis, the runtime to train 20000 data is fast enough (about several minutes). Therefore, we decide not to implement PCA here to keep the information in the dataset.

4.3.3 Performance of Random Forest Model

Finally, given fixed mtry and ntree, we can compute the testing error of our random forest model. After we do prediction on the testing dataset, we set the threshold as 0.5. If the result is greater than 0.5, we would categorize the result as 1. On the other hand, result less than 0.5 is 0. Finally we compte the error to be 0.082



5 Performance Analysis

6 Conclusion