HARVARD EXTENSION SCHOOL

EXT CSCI E-106 Model Data Class Group Project Template

Author Kaleo Pudim Author Luciano Carvalho Author Ibrahim Hashim
Author Bethun Bhowmik Author Mohanish Kashiwar Author Seymur Hasanov

02 December 2023

Abstract

This is the location for your abstract. It must consist of two paragraphs.

Contents

House Sales in King County, USA data to be used in the Final Project

Variable	Description
id	**Unique ID for each home sold (it is not a predictor)**
date	*Date of the home sale*
price	*Price of each home sold*
bedrooms	*Number of bedrooms*
bathrooms	*Number of bathrooms, where ".5" accounts for a bathroom with a toilet but no shower*
sqft_living	*Square footage of the apartment interior living space*
sqft_lot	*Square footage of the land space*
floors	*Number of floors*
waterfront	*A dummy variable for whether the apartment was overlooking the waterfront or not*
view	*An index from 0 to 4 of how good the view of the property was*
condition	*An index from 1 to 5 on the condition of the apartment,*
grade	*An index from 1 to 13, where 1-3 falls short of building construction and design, 7 has an average level of construction and design, and 11-13 has a high-quality level of construction and design.*
sqft_above	*The square footage of the interior housing space that is above ground level*
sqft_basement	*The square footage of the interior housing space that is below ground level*
yr_built	*The year the house was initially built*
yr_renovated	*The year of the house's last renovation*
zipcode	*What zipcode area the house is in*
lat	*Latitude*
long	*Longitude*
sqft_living15	*The square footage of interior housing living space for the nearest 15 neighbors*
sqft_lot15	*The square footage of the land lots of the nearest 15 neighbors*

Instructions:

0. Join a team with your fellow students with appropriate size (Four Students total)
1. Load and Review the dataset named "KC_House_Sales'csv
2. Create the train data set which contains 70% of the data and use set.seed (1023). The remaining 30% will be your test data set.
3. Investigate the data and combine the level of categorical variables if needed and drop variables as needed. For example, you can drop id, Latitude, Longitude, etc.
4. Build a regression model to predict price.
5. Create scatter plots and a correlation matrix for the train data set. Interpret the possible relationship between the response.
6. Build the best multiple linear models by using the stepwise selection method. Compare the performance of the best two linear models.
7. Make sure that model assumption(s) are checked for the final model. Apply remedy measures (transformation, etc.) that helps satisfy the assumptions.
8. Investigate unequal variances and multicollinearity. If necessary, apply remedial methods (WLS, Ridge, Elastic Net, Lasso, etc.).
9. Build an alternative model based on one of the following approaches to predict price: regression tree, NN, or SVM. Check the applicable model assumptions. Explore using a logistic regression.
10. Use the test data set to assess the model performances from above.
11. Based on the performances on both train and test data sets, determine your primary (champion) model and the other model which would be your benchmark model.
12. Create a model development document that describes the model following this template, input the name of the authors, Harvard IDs, the name of the Group, all of your code and calculations, etc..:

Due Date: December 18th, 2023 at 11:59 pm EST

Notes No typographical errors, grammar mistakes, or misspelled words, use English language All tables need to be numbered and describe their content in the body of the document All figures/graphs need to be numbered and describe their content All results must be accurate and clearly explained for a casual reviewer to fully understand their purpose and impact Submit both the RMD markdown file and PDF with the sections with appropriate explanations. A more formal document in Word can be used in place of the pdf file but must include all appropriate explanations.

Executive Summary

This section will describe the model usage, your conclusions and any regulatory and internal requirements. In a real world scneario, this section is for senior management who do not need to know the details. They need to know high level (the purpose of the model, limitations of the model and any issues).

I. Introduction (5 points)

This section needs to introduce the reader to the problem to be resolved, the purpose, and the scope of the statistical testing applied. What you are doing with your prediction? What is the purpose of the model? What methods were trained on the data, how large is the test sample, and how did you build the model?

— in-progress —

This project aims to develop a predictive model for house prices in King County, USA. The model's purpose is to provide an analytical tool for understanding the key factors influencing property values in this area, which is significant for various stakeholders in the real estate sector.

Our primary data source is the 'KC_House_Sales' dataset, which includes detailed information on various house attributes. **(Add here the hypothesis of initial exploration on house prices)**.

The methodology adopted for this project combines conventional statistical techniques with modern data analytics methods. Initial models will be based on linear regression, with further exploration into more complex approaches like neural networks and decision trees, depending on their performance in preliminary analyses.

To ensure robust model training and validation, the dataset has been divided into a training set (70%) and a test set (30%). This division is crucial for evaluating the model's effectiveness in making predictions on new, unseen data.

II. Description of the data and quality (15 points)

Here you need to review your data, the statistical test applied to understand the predictors and the response and how are they correlated. Extensive graph analysis is recommended. Is the data continuous, or categorical, do any transformation needed? Do you need dummies?

Data Overview

[Add here initial observation analysis]

```
df_house = read.csv("KC_House_Sales.csv")
cat("Number of NA values in dataframe:",sum(is.na(df_house))) #no NA values
```

```
## Number of NA values in dataframe: 0
```

```
head(df_house)
```

```
##           id            date           price bedrooms bathrooms sqft_living
## 1 7129300520 20141013T000000    $221,900.00         3      1.00        1180
## 2 6414100192 20141209T000000    $538,000.00         3      2.25        2570
## 3 5631500400 20150225T000000    $180,000.00         2      1.00         770
## 4 2487200875 20141209T000000    $604,000.00         4      3.00        1960
## 5 1954400510 20150218T000000    $510,000.00         3      2.00        1680
## 6 7237550310 20140512T000000  $1,225,000.00         4      4.50        5420
##   sqft_lot floors waterfront view condition grade sqft_above sqft_basement
## 1     5650      1          0    0         3     7       1180             0
## 2     7242      2          0    0         3     7       2170           400
## 3    10000      1          0    0         3     6        770             0
## 4     5000      1          0    0         5     7       1050           910
## 5     8080      1          0    0         3     8       1680             0
## 6   101930      1          0    0         3    11       3890          1530
##   yr_built yr_renovated zipcode     lat     long sqft_living15 sqft_lot15
## 1     1955            0   98178 47.5112 -122.257          1340       5650
## 2     1951         1991   98125 47.7210 -122.319          1690       7639
## 3     1933            0   98028 47.7379 -122.233          2720       8062
## 4     1965            0   98136 47.5208 -122.393          1360       5000
## 5     1987            0   98074 47.6168 -122.045          1800       7503
## 6     2001            0   98053 47.6561 -122.005          4760     101930
```

Data Types, Categories and Cleaning

[Add here info about the data and what was performed]

```
# [Kaleo]
#removing `id` column
df_house = subset(df_house, select = -id)

#converting `price` to numeric
df_house$price <- as.numeric(gsub("[\\$,]", "", df_house$price))

##cleaning `date` and adding `year`,`month`,`day` columns
df_house$date <- as.POSIXct(df_house$date, format = "%Y%m%d")
df_house$year <- as.numeric(format(df_house$date, "%Y"))
df_house$month <- as.numeric(format(df_house$date, "%m"))
df_house$day <- as.numeric(format(df_house$date, "%d"))

head(df_house)
```

```
##         date   price bedrooms bathrooms sqft_living sqft_lot floors waterfront
## 1 2014-10-13  221900        3      1.00        1180     5650      1          0
## 2 2014-12-09  538000        3      2.25        2570     7242      2          0
## 3 2015-02-25  180000        2      1.00         770    10000      1          0
## 4 2014-12-09  604000        4      3.00        1960     5000      1          0
## 5 2015-02-18  510000        3      2.00        1680     8080      1          0
## 6 2014-05-12 1225000        4      4.50        5420   101930      1          0
##   view condition grade sqft_above sqft_basement yr_built yr_renovated zipcode
## 1    0         3     7       1180             0     1955            0   98178
## 2    0         3     7       2170           400     1951         1991   98125
## 3    0         3     6        770             0     1933            0   98028
## 4    0         5     7       1050           910     1965            0   98136
## 5    0         3     8       1680             0     1987            0   98074
## 6    0         3    11       3890          1530     2001            0   98053
##       lat     long sqft_living15 sqft_lot15 year month day
## 1 47.5112 -122.257          1340       5650 2014    10  13
## 2 47.7210 -122.319          1690       7639 2014    12   9
## 3 47.7379 -122.233          2720       8062 2015     2  25
## 4 47.5208 -122.393          1360       5000 2014    12   9
## 5 47.6168 -122.045          1800       7503 2015     2  18
## 6 47.6561 -122.005          4760     101930 2014     5  12
```

```
# [Kaleo]
#converting all to numeric...ignores date column
ndf_house = df_house[sapply(df_house, is.numeric)]
# use df_house instead, unless you have a specific reason!
head(ndf_house)
```

```
##     price bedrooms bathrooms sqft_living sqft_lot floors waterfront view
## 1  221900        3      1.00        1180     5650      1          0    0
## 2  538000        3      2.25        2570     7242      2          0    0
## 3  180000        2      1.00         770    10000      1          0    0
## 4  604000        4      3.00        1960     5000      1          0    0
## 5  510000        3      2.00        1680     8080      1          0    0
## 6 1225000        4      4.50        5420   101930      1          0    0
##   condition grade sqft_above sqft_basement yr_built yr_renovated zipcode
## 1         3     7       1180             0     1955            0   98178
## 2         3     7       2170           400     1951         1991   98125
## 3         3     6        770             0     1933            0   98028
## 4         5     7       1050           910     1965            0   98136
## 5         3     8       1680             0     1987            0   98074
## 6         3    11       3890          1530     2001            0   98053
##       lat     long sqft_living15 sqft_lot15 year month day
## 1 47.5112 -122.257          1340       5650 2014    10  13
## 2 47.7210 -122.319          1690       7639 2014    12   9
## 3 47.7379 -122.233          2720       8062 2015     2  25
## 4 47.5208 -122.393          1360       5000 2014    12   9
## 5 47.6168 -122.045          1800       7503 2015     2  18
## 6 47.6561 -122.005          4760     101930 2014     5  12
```

Stat Summary - in progress

[Add here the initial analysis of the summary, if applicable]

```
str(ndf_house)
```

```
## 'data.frame':    21613 obs. of  22 variables:
##  $ price        : num  221900 538000 180000 604000 510000 ...
```

```
##  $ bedrooms     : int  3 3 2 4 3 4 3 3 3 3 ...
##  $ bathrooms    : num  1 2.25 1 3 2 4.5 2.25 1.5 1 2.5 ...
##  $ sqft_living  : int  1180 2570 770 1960 1680 5420 1715 1060 1780 1890 ...
##  $ sqft_lot     : int  5650 7242 10000 5000 8080 101930 6819 9711 7470 6560 ...
##  $ floors       : num  1 2 1 1 1 1 2 1 1 2 ...
##  $ waterfront   : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ view         : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ condition    : int  3 3 3 5 3 3 3 3 3 3 ...
##  $ grade        : int  7 7 6 7 8 11 7 7 7 7 ...
##  $ sqft_above   : int  1180 2170 770 1050 1680 3890 1715 1060 1050 1890 ...
##  $ sqft_basement: int  0 400 0 910 0 1530 0 0 730 0 ...
##  $ yr_built     : int  1955 1951 1933 1965 1987 2001 1995 1963 1960 2003 ...
##  $ yr_renovated : int  0 1991 0 0 0 0 0 0 0 0 ...
##  $ zipcode      : int  98178 98125 98028 98136 98074 98053 98003 98198 98146 98038 ...
##  $ lat          : num  47.5 47.7 47.7 47.5 47.6 ...
##  $ long         : num  -122 -122 -122 -122 -122 ...
##  $ sqft_living15: int  1340 1690 2720 1360 1800 4760 2238 1650 1780 2390 ...
##  $ sqft_lot15   : int  5650 7639 8062 5000 7503 101930 6819 9711 8113 7570 ...
##  $ year         : num  2014 2014 2015 2014 2015 ...
##  $ month        : num  10 12 2 12 2 5 6 1 4 3 ...
##  $ day          : num  13 9 25 9 18 12 27 15 15 12 ...
```

```
summary(ndf_house)
```

```
##      price            bedrooms        bathrooms      sqft_living
##  Min.   :  75000   Min.   : 0.000   Min.   :0.000   Min.   :  290
##  1st Qu.: 321950   1st Qu.: 3.000   1st Qu.:1.750   1st Qu.: 1427
##  Median : 450000   Median : 3.000   Median :2.250   Median : 1910
##  Mean   : 540088   Mean   : 3.371   Mean   :2.115   Mean   : 2080
##  3rd Qu.: 645000   3rd Qu.: 4.000   3rd Qu.:2.500   3rd Qu.: 2550
##  Max.   :7700000   Max.   :33.000   Max.   :8.000   Max.   :13540
##     sqft_lot           floors        waterfront            view
##  Min.   :    520   Min.   :1.000   Min.   :0.000000   Min.   :0.0000
##  1st Qu.:   5040   1st Qu.:1.000   1st Qu.:0.000000   1st Qu.:0.0000
##  Median :   7618   Median :1.500   Median :0.000000   Median :0.0000
##  Mean   :  15107   Mean   :1.494   Mean   :0.007542   Mean   :0.2343
##  3rd Qu.:  10688   3rd Qu.:2.000   3rd Qu.:0.000000   3rd Qu.:0.0000
##  Max.   :1651359   Max.   :3.500   Max.   :1.000000   Max.   :4.0000
##    condition         grade          sqft_above   sqft_basement
##  Min.   :1.000   Min.   : 1.000   Min.   : 290   Min.   :   0.0
##  1st Qu.:3.000   1st Qu.: 7.000   1st Qu.:1190   1st Qu.:   0.0
##  Median :3.000   Median : 7.000   Median :1560   Median :   0.0
##  Mean   :3.409   Mean   : 7.657   Mean   :1788   Mean   : 291.5
##  3rd Qu.:4.000   3rd Qu.: 8.000   3rd Qu.:2210   3rd Qu.: 560.0
##  Max.   :5.000   Max.   :13.000   Max.   :9410   Max.   :4820.0
##     yr_built     yr_renovated       zipcode           lat
##  Min.   :1900   Min.   :   0.0   Min.   :98001   Min.   :47.16
##  1st Qu.:1951   1st Qu.:   0.0   1st Qu.:98033   1st Qu.:47.47
##  Median :1975   Median :   0.0   Median :98065   Median :47.57
##  Mean   :1971   Mean   :  84.4   Mean   :98078   Mean   :47.56
##  3rd Qu.:1997   3rd Qu.:   0.0   3rd Qu.:98118   3rd Qu.:47.68
##  Max.   :2015   Max.   :2015.0   Max.   :98199   Max.   :47.78
##       long        sqft_living15    sqft_lot15          year
##  Min.   :-122.5   Min.   : 399   Min.   :   651   Min.   :2014
##  1st Qu.:-122.3   1st Qu.:1490   1st Qu.:  5100   1st Qu.:2014
##  Median :-122.2   Median :1840   Median :  7620   Median :2014
##  Mean   :-122.2   Mean   :1987   Mean   : 12768   Mean   :2014
##  3rd Qu.:-122.1   3rd Qu.:2360   3rd Qu.: 10083   3rd Qu.:2015
##  Max.   :-121.3   Max.   :6210   Max.   :871200   Max.   :2015
```

```
##      month             day
##  Min.   : 1.000   Min.   : 1.00
##  1st Qu.: 4.000   1st Qu.: 8.00
##  Median : 6.000   Median :16.00
##  Mean   : 6.574   Mean   :15.69
##  3rd Qu.: 9.000   3rd Qu.:23.00
##  Max.   :12.000   Max.   :31.00
```

```
summary(ndf_house$price)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   75000  321950  450000  540088  645000 7700000
```

Graphical Analysis - in progress - Seymur

[Add here the initial analysis]

The pairwise scatter plot shows the correlation between highly correlated variables extracted from heatmap correlation matrix.

- there is a strong correlation between sqft_living and price, indicated that as the living area increases, so does the house price increase.
- A strong positive correlation with grade, implying that higher-quality houses tend to be more expensive.
- Each variable's distribution is shown on the diagonal, with price notably skewed towards lower values, indicating most homes are on the more affordable end of the spectrum with fewer high-priced outliers.

```
library(GGally)
```

```
## Warning: package 'GGally' was built under R version 4.3.2
```

```
## Loading required package: ggplot2
```

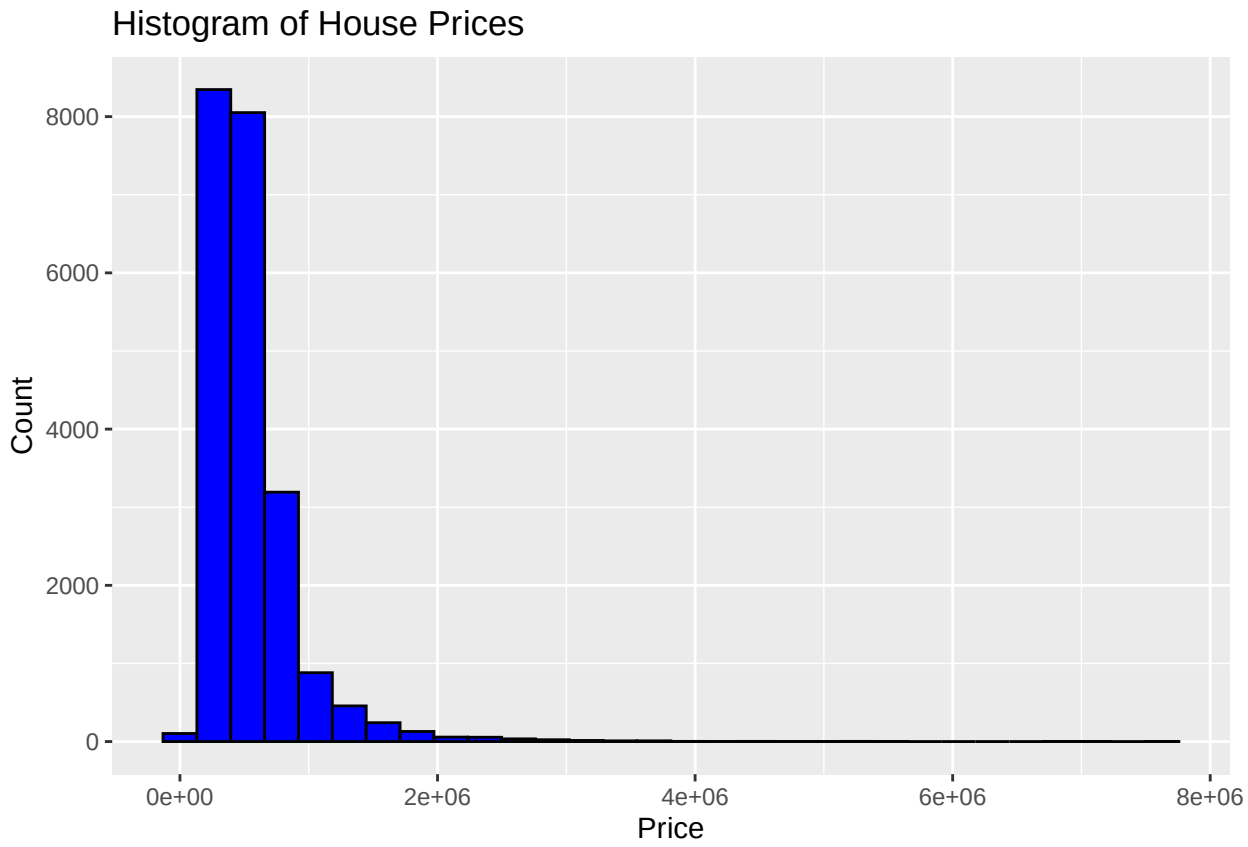
```
## Warning: package 'ggplot2' was built under R version 4.3.2
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
# Pairwise scatter plot with correlation coefficients
ggpairs(ndf_house, columns = c("price", "sqft_living", "bedrooms", "bathrooms", "grade"),
        title = "Pairwise Scatter Plots with House Price")
```

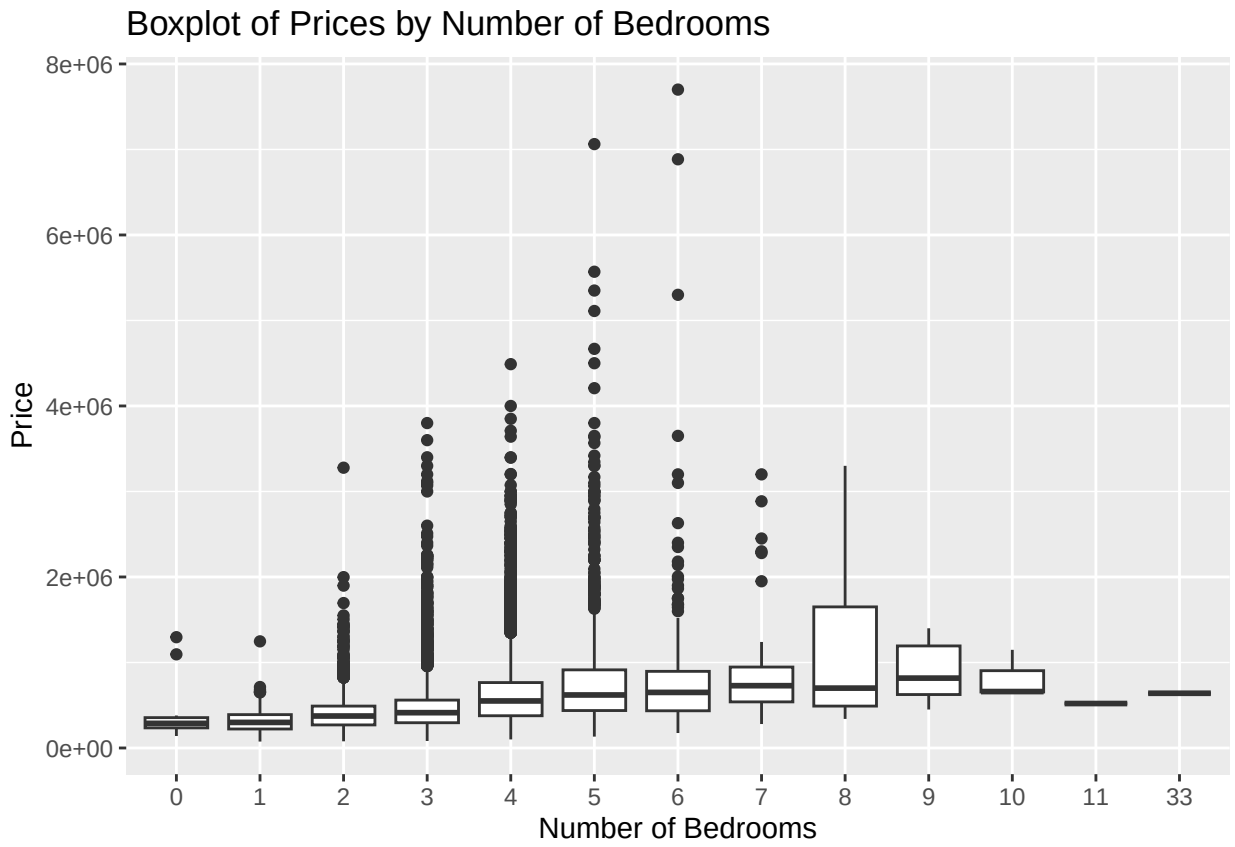
Analysis: The histogram below shows the distribution of house prices, revealing that most houses are in the lower price range with a significant decrease in the number of houses as the price increases, indicating a right-skewed distribution with relatively few high-priced houses.

```
# Histogram of house prices
ggplot(df_house, aes(x = price)) +
  geom_histogram(bins = 30, fill = "blue", color = "black") +
  labs(title = "Histogram of House Prices", x = "Price", y = "Count")
```

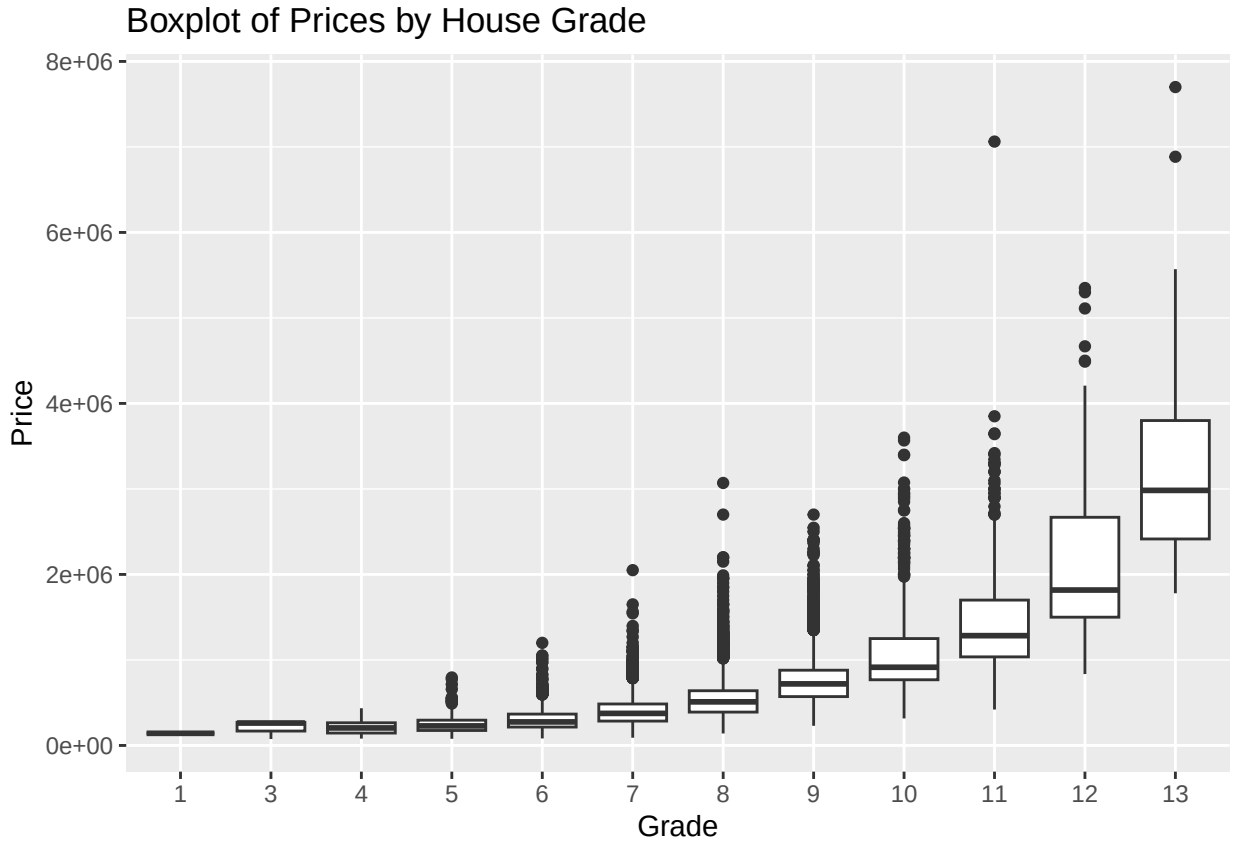


Analysis: The first boxplot shows that house prices generally increase with the number of bedrooms, but vary widely, especially for homes with more bedrooms. The second boxplot indicates that house prices rise with better house grades, with greater price variability at higher grades.Both plots demonstrate that while there is a general trend of increasing price with more bedrooms and higher grades, there is also a considerable variation within these categories. The presence of outliers suggests that factors other than the number of bedrooms and house grade can significantly influence house prices.

```
library(ggplot2)

# Boxplot for price by number of bedrooms
ggplot(df_house, aes(x = factor(bedrooms), y = price)) +
  geom_boxplot() +
  labs(title = "Boxplot of Prices by Number of Bedrooms", x = "Number of Bedrooms", y = "Price")
```

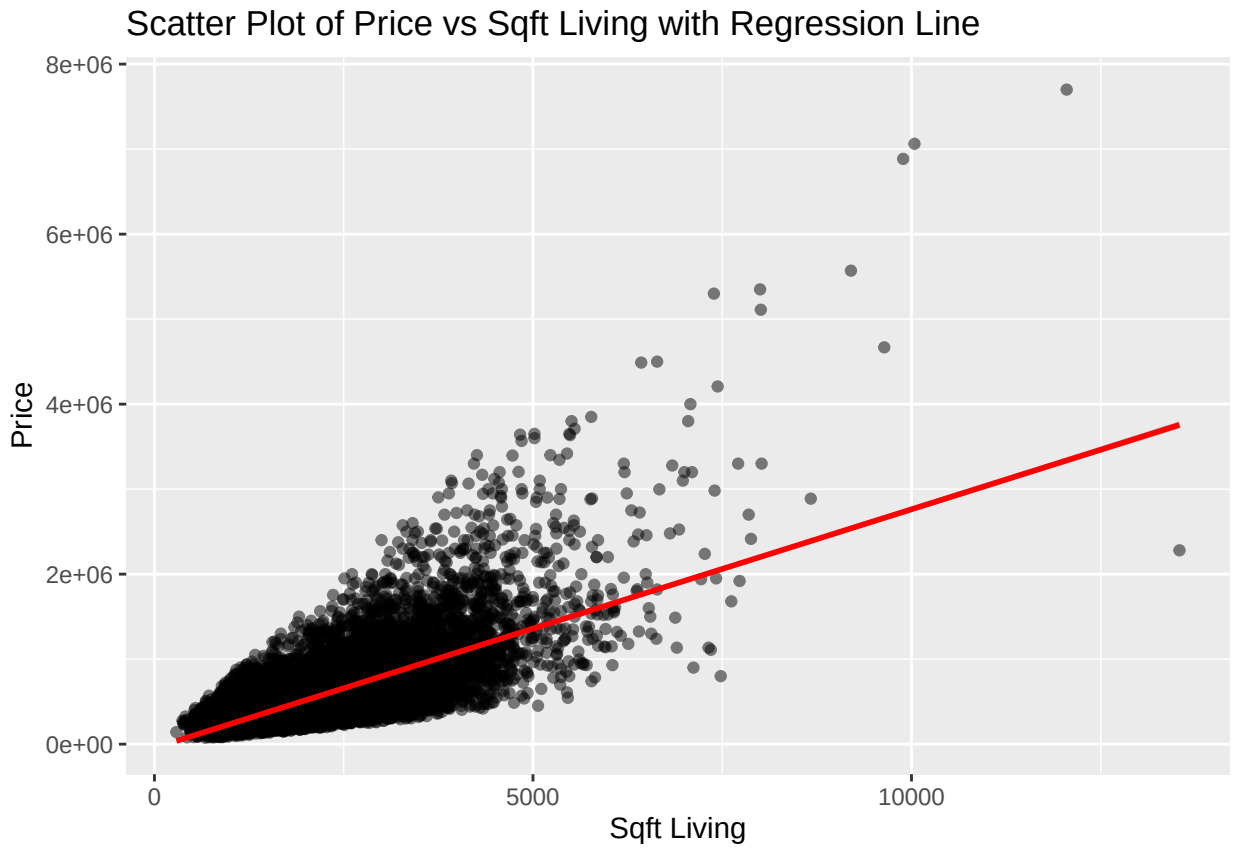



```r
# Boxplot for price by house grade
ggplot(df_house, aes(x = factor(grade), y = price)) +
  geom_boxplot() +
  labs(title = "Boxplot of Prices by House Grade", x = "Grade", y = "Price")
```



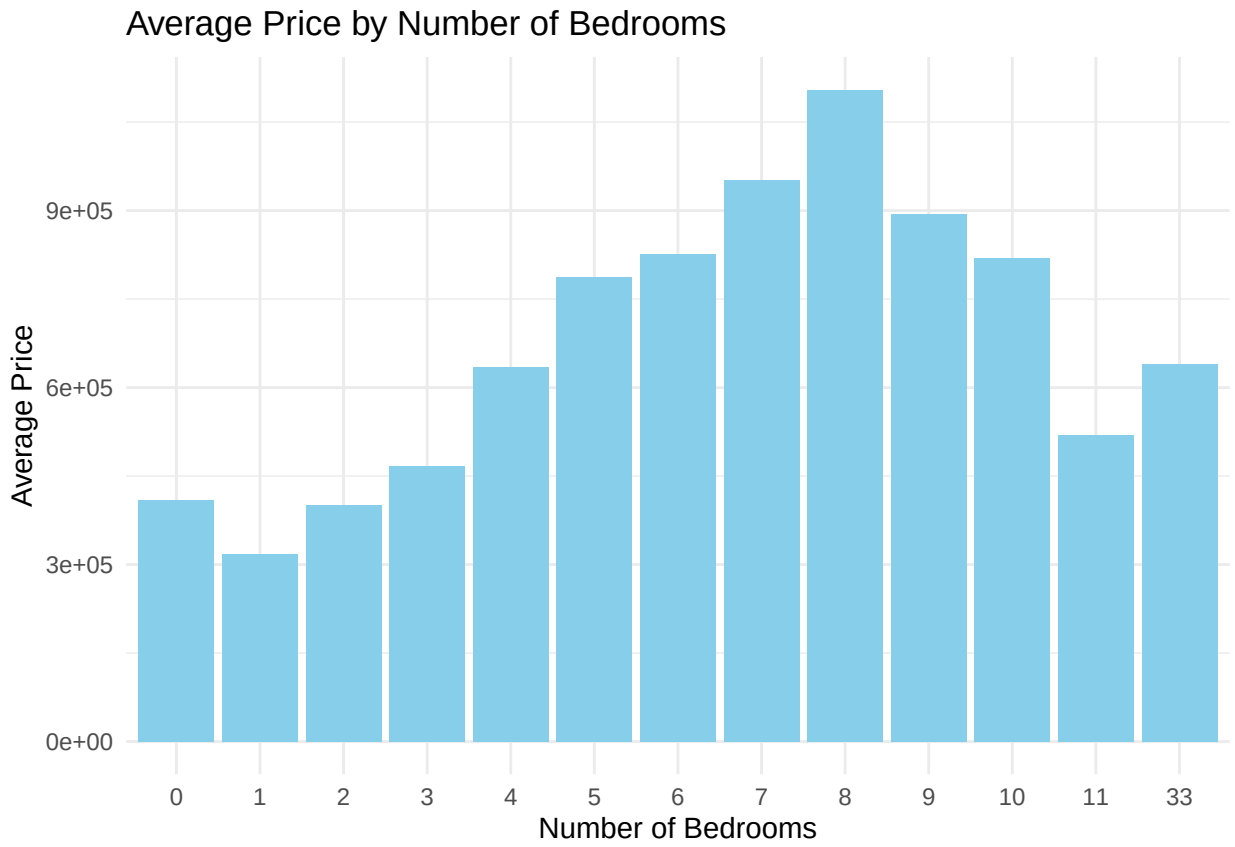
```
ggplot(df_house, aes(x = sqft_living, y = price)) +
  geom_point(alpha = 0.5) +
  geom_smooth(method = "lm", color = "red") +
  labs(title = "Scatter Plot of Price vs Sqft Living with Regression Line", x = "Sqft Living", y = "Price")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



Analysis: the average price tends to increase with the number of bedrooms up to a certain point, but then the trend is less consistent. For instance, homes with 6 bedrooms have a higher average price than those with 7 or 8 bedrooms, and the average price for homes with 11 bedrooms is lower than for those with fewer bedrooms. There is a notable outlier at 33 bedrooms with a relatively low average price, which could indicate an atypical property or data error.

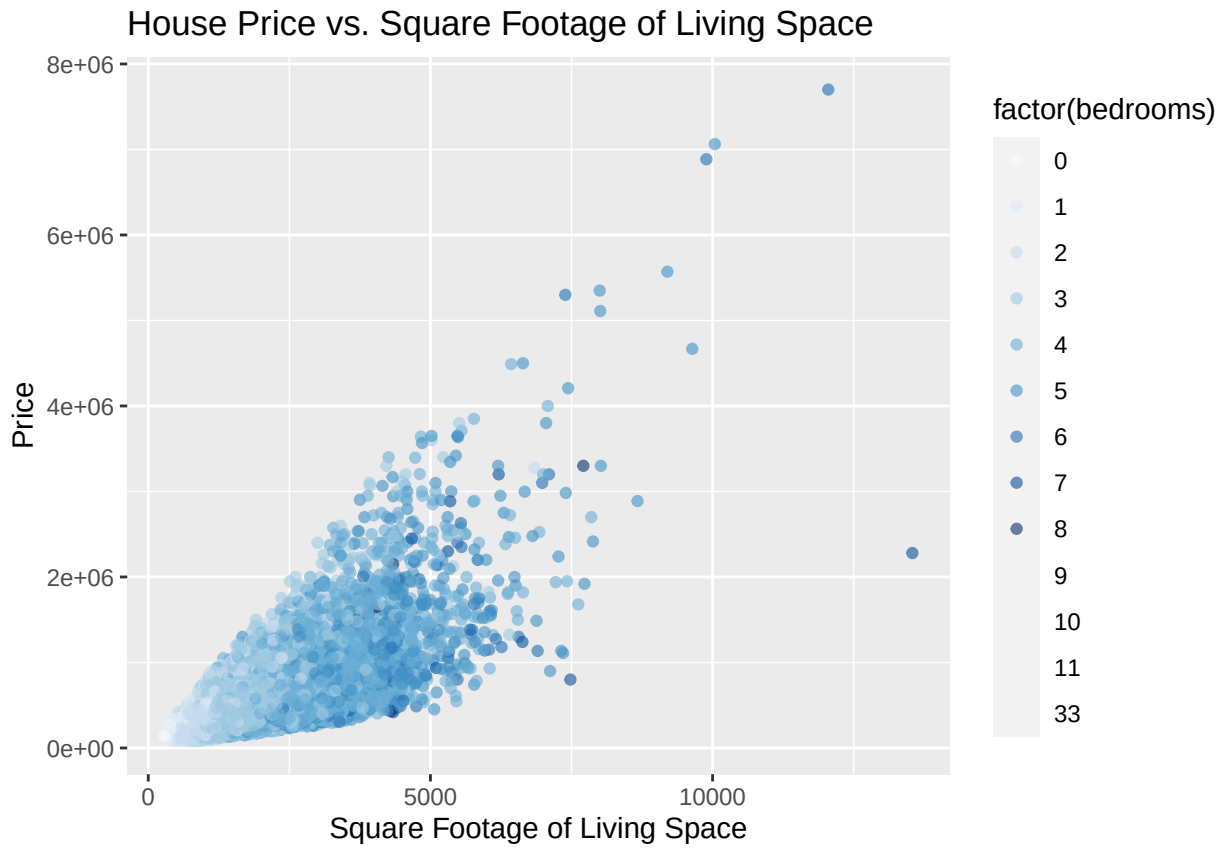
```
library(ggplot2)

average_prices <- aggregate(price ~ bedrooms, data = ndf_house, FUN = mean)
ggplot(average_prices, aes(x = factor(bedrooms), y = price)) +
  geom_bar(stat = "identity", fill = "skyblue") +
  labs(title = "Average Price by Number of Bedrooms",
       x = "Number of Bedrooms",
       y = "Average Price") +
  theme_minimal()
```




```
library(ggplot2)

ggplot(df_house, aes(x = sqft_living, y = price)) +
  geom_point(aes(color = factor(bedrooms)), alpha = 0.6) +
  scale_color_brewer(type = 'seq', palette = 'Blues') +
  labs(title = "House Price vs. Square Footage of Living Space", x = "Square Footage of Living Space", y = "Pr
```

```
## Warning in RColorBrewer::brewer.pal(n, pal): n too large, allowed maximum for palette Blues is 9
## Returning the palette you asked for with that many colors
```

```
## Warning: Removed 11 rows containing missing values (‘geom_point()‘).
```

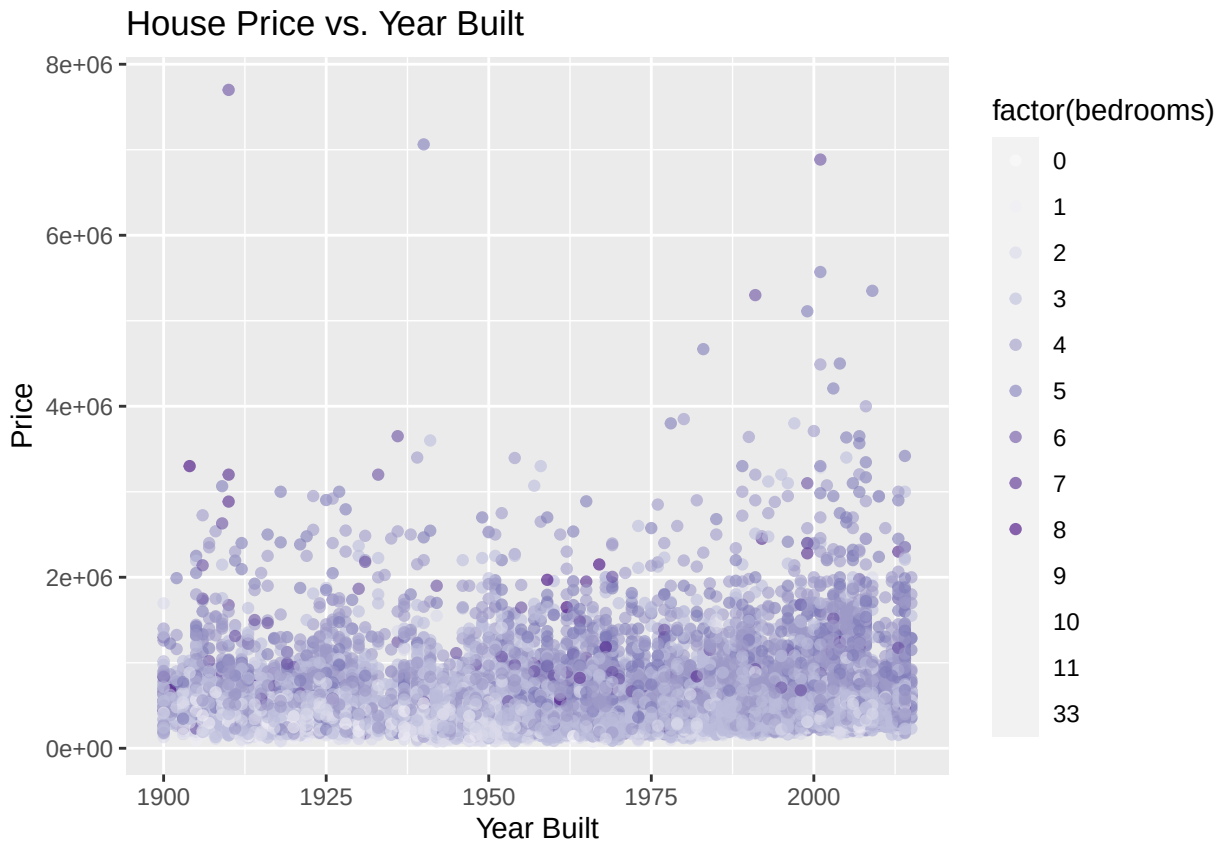



```
ggplot(df_house, aes(x = factor(waterfront), y = price)) +
  geom_boxplot() +
  labs(title = "Boxplot of House Prices by Waterfront", x = "Waterfront", y = "Price")
```

```
ggplot(df_house, aes(x = yr_built, y = price)) +
  geom_point(aes(color = factor(bedrooms)), alpha = 0.6) +
  scale_color_brewer(type = 'seq', palette = 'Purples') +
  labs(title = "House Price vs. Year Built", x = "Year Built", y = "Price")
```

```
## Warning in RColorBrewer::brewer.pal(n, pal): n too large, allowed maximum for palette Purples is 9
## Returning the palette you asked for with that many colors
```

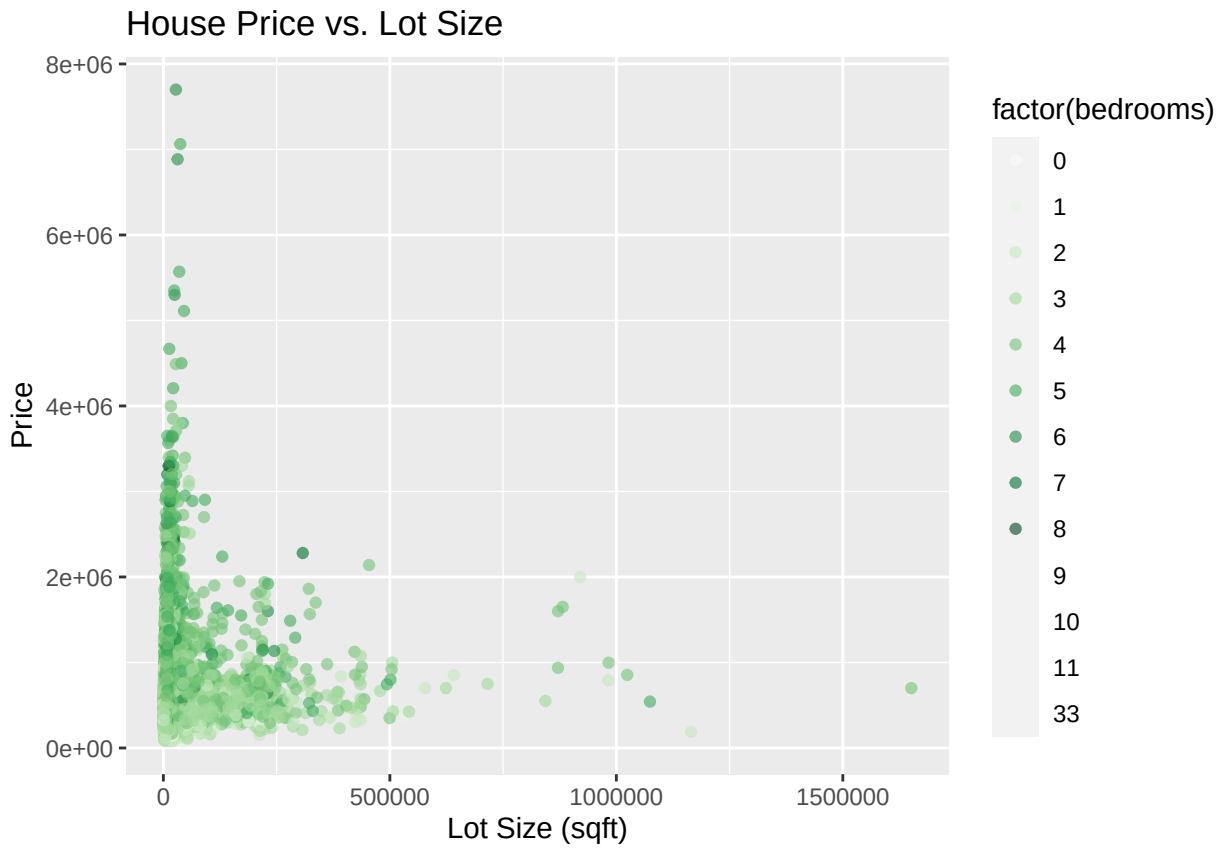
```
## Warning: Removed 11 rows containing missing values (‘geom_point()‘).
```




```
ggplot(df_house, aes(x = sqft_lot, y = price)) +
  geom_point(aes(color = factor(bedrooms)), alpha = 0.6) +
  scale_color_brewer(type = 'seq', palette = 'Greens') +
  labs(title = "House Price vs. Lot Size", x = "Lot Size (sqft)", y = "Price")
```

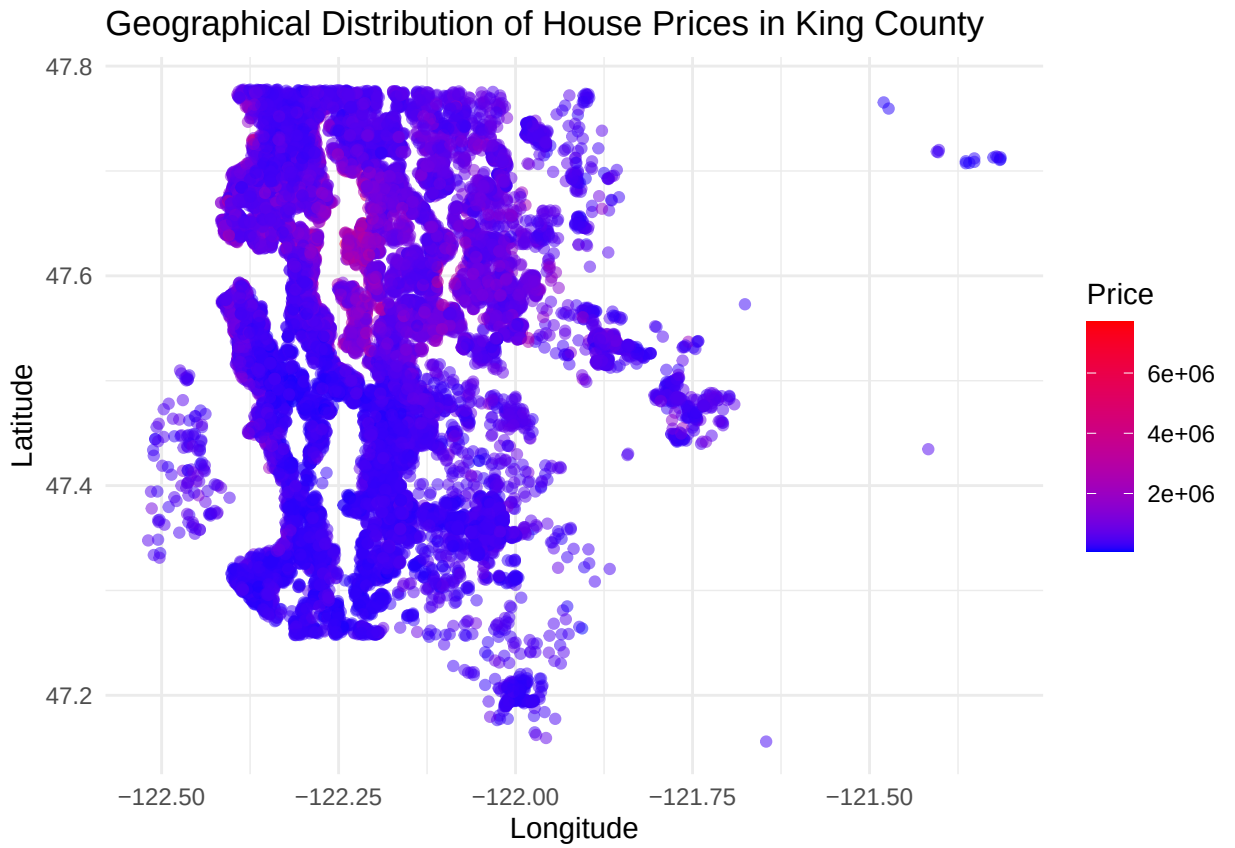
```
## Warning in RColorBrewer::brewer.pal(n, pal): n too large, allowed maximum for palette Greens is 9
## Returning the palette you asked for with that many colors
```

```
## Warning: Removed 11 rows containing missing values (‘geom_point()‘).
```



[Add here the initial analysis]

```
library(ggplot2)

# Scatter plot of properties
ggplot(data = ndf_house, aes(x = long, y = lat, color = price)) +
  geom_point(alpha = 0.5) +
  scale_color_gradient(low = "blue", high = "red") +
  labs(title = "Geographical Distribution of House Prices in King County",
       x = "Longitude", y = "Latitude", color = "Price") +
  theme_minimal()
```



[Analysis section]

From geographical map, it is evident that properties situated around the latitude line of 47.6 and longitude between -122.25 and -122.00, which corresponds to the central and northern parts of Seattle, command higher prices per square foot. The relatively lower-priced properties per square foot, shown in orange, are more dispersed and located primarily south of central Seattle, extending towards Tacoma, as well as in the outlying suburban areas.It is also noticeable that along the latitudinal line around 47.4, there are pockets of high-priced properties per square foot, potentially indicating affluent neighborhoods or areas with high-value real estate.

The map clearly shows a correlation between location and property value per square foot, with central urban areas exhibiting the highest values. This pattern is typical for urban centers where proximity to amenities, employment opportunities, and other socioeconomic factors drive up real estate prices.

```
library(ggmap)
```

```
## Warning: package 'ggmap' was built under R version 4.3.2
```

```
## i Google's Terms of Service: <https://mapsplatform.google.com>
##   Stadia Maps' Terms of Service: <https://stadiamaps.com/terms-of-service/>
##   OpenStreetMap's Tile Usage Policy: <https://operations.osmfoundation.org/policies/tiles/>
## i Please cite ggmap if you use it! Use `citation("ggmap")` for details.
```

```
library(ggplot2)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
##
## The following objects are masked from 'package:stats':
##
```

```
##     filter, lag
##
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

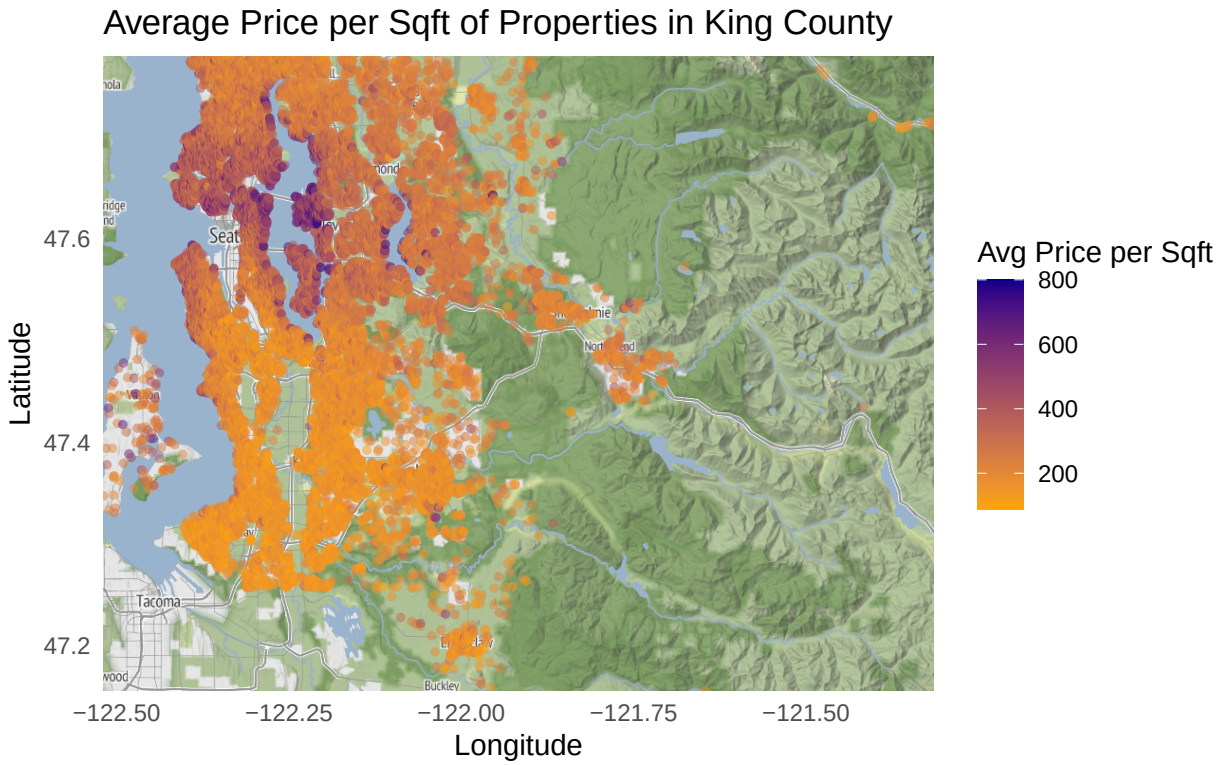
```
library(car)
```

```
## Loading required package: carData
##
## Attaching package: 'car'
##
## The following object is masked from 'package:dplyr':
##
##     recode
```

```
register_stadiamaps(key = '765cbcd1-2ec2-40e3-8a81-98624e616208')
df_summary <- ndf_house %>%
  mutate(price_per_sqft = price / sqft_living) %>%
  group_by(long, lat) %>%
  summarize(avg_price_per_sqft = mean(price_per_sqft, na.rm = TRUE), .groups = 'drop')

king_county_map <- get_stadiamap(bbox = c(left = min(df_house$long),
                                          bottom = min(df_house$lat),
                                          right = max(df_house$long),
                                          top = max(df_house$lat)),
                                 maptype = "stamen_terrain",
                                 zoom = 10)
```

```
## i © Stadia Maps © Stamen Design © OpenMapTiles © OpenStreetMap contributors.
```

```
ggmap(king_county_map) +
  geom_point(data = df_summary, aes(x = long, y = lat, color = avg_price_per_sqft),
             alpha = 0.5, size = 1) +
  scale_color_gradient(low = "orange", high = "darkblue", name = "Avg Price per Sqft") +
  labs(title = "Average Price per Sqft of Properties in King County",
       x = "Longitude", y = "Latitude") +
  theme_minimal()
```



Zipcode

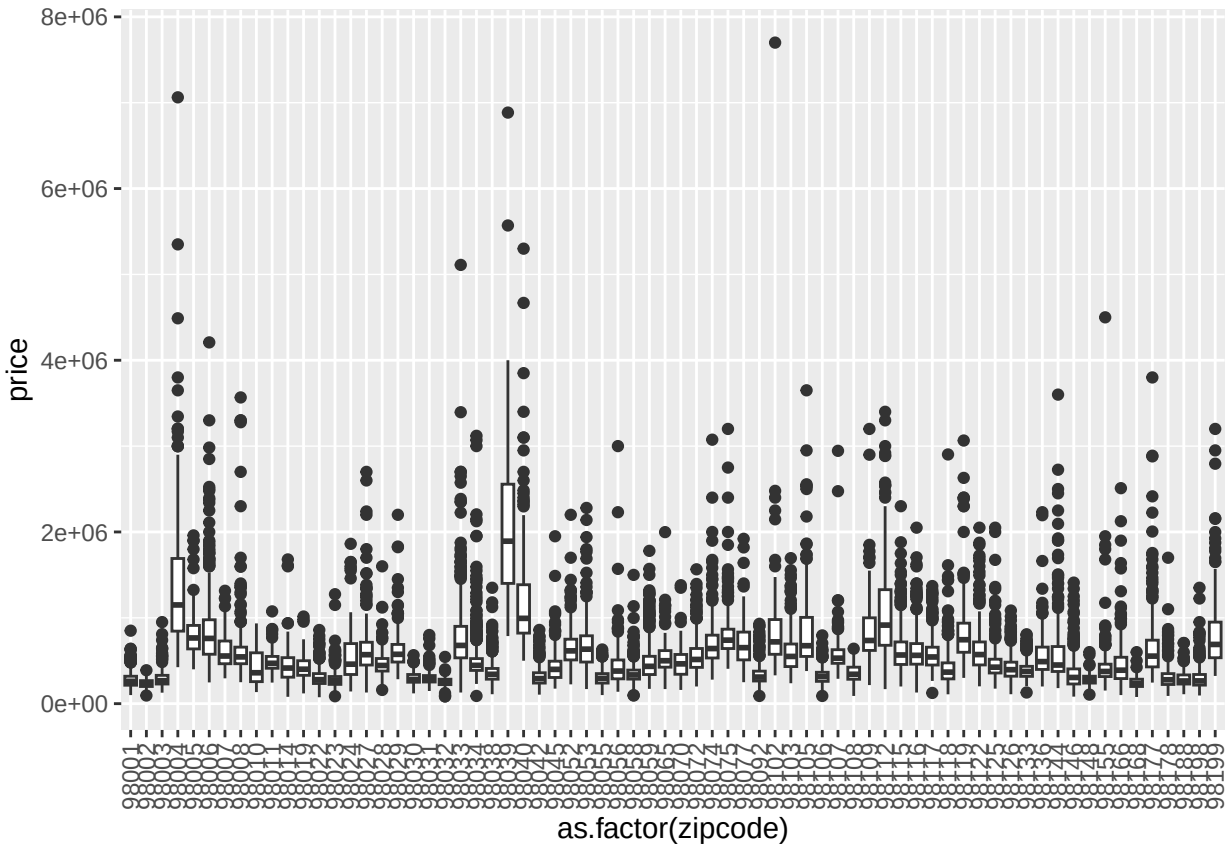
visualizations (can discard) ***Zipcode is a valid predictor***

```
library(dplyr)
library(scales)
```

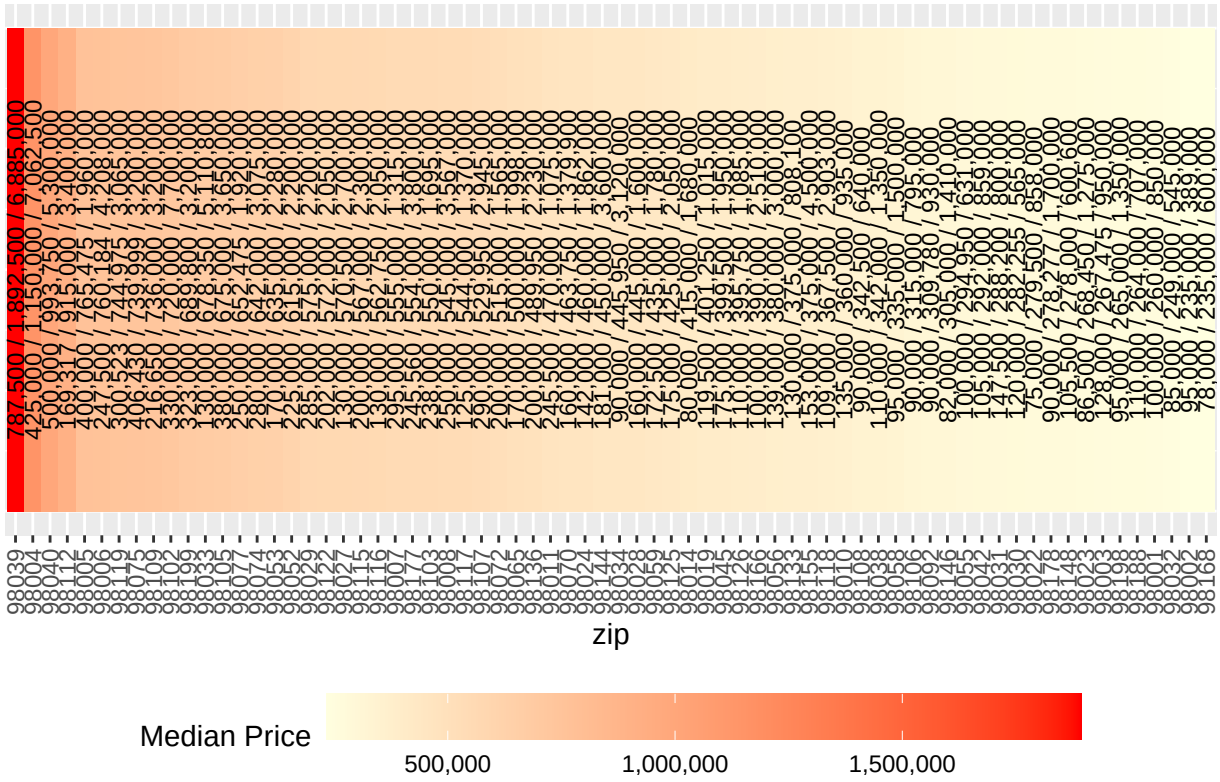
```
##
## Attaching package: 'scales'
```

```
## The following object is masked from 'package:readr':
##
##     col_factor
```

```
# [Kaleo]
ggplot( df_house, aes(x = as.factor(zipcode), y = price)) +
  geom_boxplot()+theme(axis.text.x = element_text(angle = 90,vjust=0.5))
```




```
# [Kaleo]
# Prices by Zipcode heatmap
heatmap_data <- df_house %>%
  group_by(zipcode) %>%
  summarize(med_price = median(price),
            min_price = min(price),
            max_price = max(price))
heatmap_data <- heatmap_data %>%
  arrange(desc(med_price))

ggplot(heatmap_data, aes(x = factor(zipcode, levels = zipcode), y = 1, fill = med_price)) +
  geom_tile() +
  geom_text(aes(label = paste(scales::comma(min_price),"/",
                              scales::comma(med_price),"/",
                              scales::comma(max_price)),
                vjust = 0.5,angle=90),
            color = "black", size = 3)+
  scale_fill_gradient(low = "lightyellow", high = "red", name = "Median Price",
                      labels = scales::comma_format()) +
  labs(title = "Heatmap of median prices by zipcode (min/med/max)",
       x = "zip",
       y = "") +
  theme(axis.text.x = element_text(angle = 90, hjust = 0,vjust=0.5),
        axis.title.y = element_blank(),
        axis.text.y = element_blank(),
        axis.ticks.y = element_blank(),
        legend.position = "bottom",
        legend.key.width = unit(2, "cm"))
```



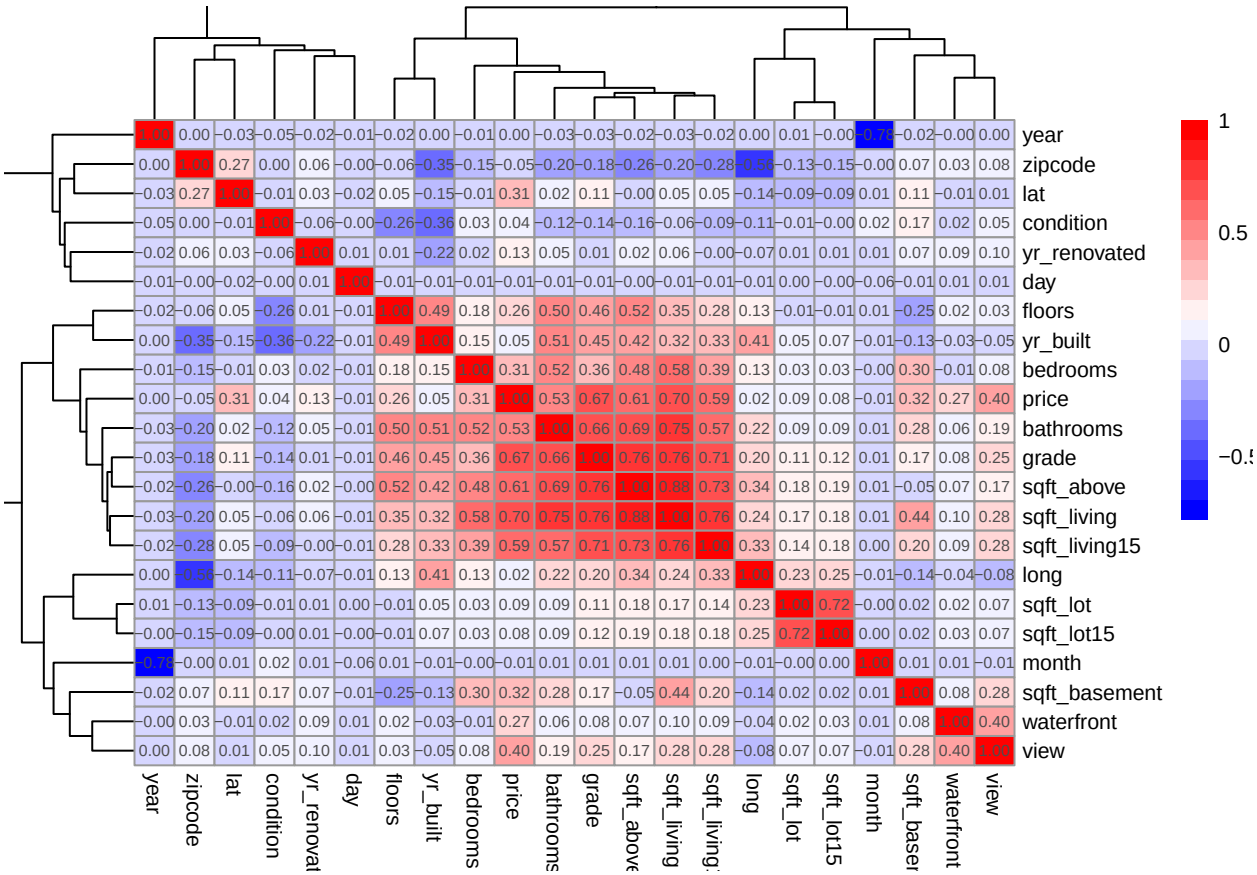
Correlation Analysis - in progress

[Add here the initial analysis of the correlation matrix]

```
library(pheatmap)
```

```
## Warning: package 'pheatmap' was built under R version 4.3.2
```

```
# [Kaleo]
#correlation matrix
cor_matrix = cor(ndf_house)
correlation_df = as.data.frame(cor_matrix)

#new dataframe with variables 'highly' (>0.2) correlated with price
x_high = subset(ndf_house, select = c(view,waterfront,sqft_basement,sqft_living,
                                      sqft_living15,sqft_above,grade,bathrooms,bedrooms,floors,lat))

pheatmap(cor_matrix,
         color = colorRampPalette(c("blue", "white", "red"))(20),
         main = "correlation matrix heatmap",
         fontsize = 8,
         cellwidth = 15,
         cellheight = 11,
         display_numbers = TRUE
)
```



Creating a new dataframe with variables 'highly' (>0.2) correlated with price

```
# [Kaleo]
head(x_high)
```

```
##   view waterfront sqft_basement sqft_living sqft_living15 sqft_above grade
## 1    0          0             0        1180          1340        1180     7
## 2    0          0           400        2570          1690        2170     7
## 3    0          0             0         770          2720         770     6
## 4    0          0           910        1960          1360        1050     7
## 5    0          0             0        1680          1800        1680     8
## 6    0          0          1530        5420          4760        3890    11
##   bathrooms bedrooms floors     lat
## 1      1.00        3      1 47.5112
## 2      2.25        3      2 47.7210
## 3      1.00        2      1 47.7379
## 4      3.00        4      1 47.5208
## 5      2.00        3      1 47.6168
## 6      4.50        4      1 47.6561
```

Data Transformation

Categorical Variables and Age Analysis

Renovation Indicator Variable A binary variable named 'renovated' was introduced to indicate whether a property has undergone renovation. This variable is set to 1 if the 'yr_renovated' field is not zero, signifying that the property has been renovated at least once. Otherwise, it is set to 0, indicating no renovation. This distinction provides a straightforward way to assess the impact of renovations on property values.

```r
# Creating a new variable 'renovated'
# 1 if the property has been renovated (yr_renovated != 0), 0 otherwise
df_house$renovated = ifelse(df_house$yr_renovated != 0, 1, 0)
```

Dummy Variables for yr_built To capture the historical aspect of the properties, the 'yr_built' variable was categorized into six distinct periods: pre-1901, 1901-1925, 1926-1950, 1951-1975, 1976-2000, and post-2000. Dummy variables were then created for each of these categories, allowing for a nuanced analysis of how the construction era affects house prices, while facilitating easier interpretation in regression models.

```r
# Defining breaks for year built
breaks = c(-Inf, 1900, 1925, 1950, 1975, 2000, Inf)
labels = c("<=1900", "1901-1925", "1926-1950", "1951-1975", "1976-2000", "2001+")

# Creating factor variable for year built categories
df_house$yr_built_cat = cut(df_house$yr_built, breaks = breaks, labels = labels, right = FALSE)


# Creating dummy variables
yr_built_dummies = model.matrix(~yr_built_cat-1, data = df_house)
colnames(yr_built_dummies) = labels
df_house = cbind(df_house, yr_built_dummies)
```

Property Age Calculation This is a tentative way to consider "age since las renovation" and see if there's a correlation between the time a house was last built/renovated on its selling price. A new variable called 'age' was calculated to represent the current age of each property. If a property was renovated, its age is the difference between 2023 and the renovation year ('yr_renovated'). If not renovated, the age is the difference between 2023 and the year the house was built ('yr_built'). This variable helps in understanding the effect of property age and recent renovations on house prices.

```r
# Creating 'age' column
df_house$age = ifelse(df_house$renovated == 1, 2023 - df_house$yr_renovated, 2023 - df_house$yr_built)

head(df_house)
```

```
##         date   price bedrooms bathrooms sqft_living sqft_lot floors waterfront
## 1 2014-10-13  221900        3      1.00        1180     5650      1          0
## 2 2014-12-09  538000        3      2.25        2570     7242      2          0
## 3 2015-02-25  180000        2      1.00         770    10000      1          0
## 4 2014-12-09  604000        4      3.00        1960     5000      1          0
## 5 2015-02-18  510000        3      2.00        1680     8080      1          0
## 6 2014-05-12 1225000        4      4.50        5420   101930      1          0
##   view condition grade sqft_above sqft_basement yr_built yr_renovated zipcode
## 1    0         3     7       1180             0     1955            0   98178
## 2    0         3     7       2170           400     1951         1991   98125
## 3    0         3     6        770             0     1933            0   98028
## 4    0         5     7       1050           910     1965            0   98136
## 5    0         3     8       1680             0     1987            0   98074
## 6    0         3    11       3890          1530     2001            0   98053
##       lat     long sqft_living15 sqft_lot15 year month day renovated
## 1 47.5112 -122.257          1340       5650 2014    10  13         0
## 2 47.7210 -122.319          1690       7639 2014    12   9         1
## 3 47.7379 -122.233          2720       8062 2015     2  25         0
## 4 47.5208 -122.393          1360       5000 2014    12   9         0
## 5 47.6168 -122.045          1800       7503 2015     2  18         0
## 6 47.6561 -122.005          4760     101930 2014     5  12         0
##   yr_built_cat <=1900 1901-1925 1926-1950 1951-1975 1976-2000 2001+ age
## 1    1951-1975      0         0         0         1         0     0  68
## 2    1951-1975      0         0         0         1         0     0  32
```

```
## 3    1926-1950      0         0         1         0         0     0  90
## 4    1951-1975      0         0         0         1         0     0  58
## 5    1976-2000      0         0         0         0         1     0  36
## 6        2001+      0         0         0         0         0     1  22
```

Zipcode Dummy Variables

```
# [Kaleo]
#store zip codes
zipcodes = data.frame(df_house$zipcode,df_house$price)
colnames(zipcodes) = c("zip","price")

#create dummies
dummy_zipcodes = model.matrix(~ 0 + as.factor(zipcode), data = df_house)
colnames(dummy_zipcodes) = paste0("zipcode_", levels(as.factor(df_house$zipcode)))
df_house = cbind(df_house, dummy_zipcodes)
#df_house = subset(df_house, select = -c(zipcode))

head(df_house)
```

```
##         date   price bedrooms bathrooms sqft_living sqft_lot floors waterfront
## 1 2014-10-13  221900        3      1.00        1180     5650      1          0
## 2 2014-12-09  538000        3      2.25        2570     7242      2          0
## 3 2015-02-25  180000        2      1.00         770    10000      1          0
## 4 2014-12-09  604000        4      3.00        1960     5000      1          0
## 5 2015-02-18  510000        3      2.00        1680     8080      1          0
## 6 2014-05-12 1225000        4      4.50        5420   101930      1          0
##   view condition grade sqft_above sqft_basement yr_built yr_renovated zipcode
## 1    0         3     7       1180             0     1955            0   98178
## 2    0         3     7       2170           400     1951         1991   98125
## 3    0         3     6        770             0     1933            0   98028
## 4    0         5     7       1050           910     1965            0   98136
## 5    0         3     8       1680             0     1987            0   98074
## 6    0         3    11       3890          1530     2001            0   98053
##       lat     long sqft_living15 sqft_lot15 year month day renovated
## 1 47.5112 -122.257          1340       5650 2014    10  13         0
## 2 47.7210 -122.319          1690       7639 2014    12   9         1
## 3 47.7379 -122.233          2720       8062 2015     2  25         0
## 4 47.5208 -122.393          1360       5000 2014    12   9         0
## 5 47.6168 -122.045          1800       7503 2015     2  18         0
## 6 47.6561 -122.005          4760     101930 2014     5  12         0
##   yr_built_cat <=1900 1901-1925 1926-1950 1951-1975 1976-2000 2001+ age
## 1    1951-1975      0         0         0         1         0     0  68
## 2    1951-1975      0         0         0         1         0     0  32
## 3    1926-1950      0         0         1         0         0     0  90
## 4    1951-1975      0         0         0         1         0     0  58
## 5    1976-2000      0         0         0         0         1     0  36
## 6        2001+      0         0         0         0         0     1  22
##   zipcode_98001 zipcode_98002 zipcode_98003 zipcode_98004 zipcode_98005
## 1             0             0             0             0             0
## 2             0             0             0             0             0
## 3             0             0             0             0             0
## 4             0             0             0             0             0
## 5             0             0             0             0             0
## 6             0             0             0             0             0
##   zipcode_98006 zipcode_98007 zipcode_98008 zipcode_98010 zipcode_98011
## 1             0             0             0             0             0
## 2             0             0             0             0             0
```

```
## 3             0             0             0             0             0
## 4             0             0             0             0             0
## 5             0             0             0             0             0
## 6             0             0             0             0             0
##   zipcode_98014 zipcode_98019 zipcode_98022 zipcode_98023 zipcode_98024
## 1             0             0             0             0             0
## 2             0             0             0             0             0
## 3             0             0             0             0             0
## 4             0             0             0             0             0
## 5             0             0             0             0             0
## 6             0             0             0             0             0
##   zipcode_98027 zipcode_98028 zipcode_98029 zipcode_98030 zipcode_98031
## 1             0             0             0             0             0
## 2             0             0             0             0             0
## 3             0             1             0             0             0
## 4             0             0             0             0             0
## 5             0             0             0             0             0
## 6             0             0             0             0             0
##   zipcode_98032 zipcode_98033 zipcode_98034 zipcode_98038 zipcode_98039
## 1             0             0             0             0             0
## 2             0             0             0             0             0
## 3             0             0             0             0             0
## 4             0             0             0             0             0
## 5             0             0             0             0             0
## 6             0             0             0             0             0
##   zipcode_98040 zipcode_98042 zipcode_98045 zipcode_98052 zipcode_98053
## 1             0             0             0             0             0
## 2             0             0             0             0             0
## 3             0             0             0             0             0
## 4             0             0             0             0             0
## 5             0             0             0             0             0
## 6             0             0             0             0             1
##   zipcode_98055 zipcode_98056 zipcode_98058 zipcode_98059 zipcode_98065
## 1             0             0             0             0             0
## 2             0             0             0             0             0
## 3             0             0             0             0             0
## 4             0             0             0             0             0
## 5             0             0             0             0             0
## 6             0             0             0             0             0
##   zipcode_98070 zipcode_98072 zipcode_98074 zipcode_98075 zipcode_98077
## 1             0             0             0             0             0
## 2             0             0             0             0             0
## 3             0             0             0             0             0
## 4             0             0             0             0             0
## 5             0             0             1             0             0
## 6             0             0             0             0             0
##   zipcode_98092 zipcode_98102 zipcode_98103 zipcode_98105 zipcode_98106
## 1             0             0             0             0             0
## 2             0             0             0             0             0
## 3             0             0             0             0             0
## 4             0             0             0             0             0
## 5             0             0             0             0             0
## 6             0             0             0             0             0
##   zipcode_98107 zipcode_98108 zipcode_98109 zipcode_98112 zipcode_98115
## 1             0             0             0             0             0
## 2             0             0             0             0             0
## 3             0             0             0             0             0
## 4             0             0             0             0             0
## 5             0             0             0             0             0
## 6             0             0             0             0             0
```

```
##   zipcode_98116 zipcode_98117 zipcode_98118 zipcode_98119 zipcode_98122
## 1             0             0             0             0             0
## 2             0             0             0             0             0
## 3             0             0             0             0             0
## 4             0             0             0             0             0
## 5             0             0             0             0             0
## 6             0             0             0             0             0
##   zipcode_98125 zipcode_98126 zipcode_98133 zipcode_98136 zipcode_98144
## 1             0             0             0             0             0
## 2             1             0             0             0             0
## 3             0             0             0             0             0
## 4             0             0             0             1             0
## 5             0             0             0             0             0
## 6             0             0             0             0             0
##   zipcode_98146 zipcode_98148 zipcode_98155 zipcode_98166 zipcode_98168
## 1             0             0             0             0             0
## 2             0             0             0             0             0
## 3             0             0             0             0             0
## 4             0             0             0             0             0
## 5             0             0             0             0             0
## 6             0             0             0             0             0
##   zipcode_98177 zipcode_98178 zipcode_98188 zipcode_98198 zipcode_98199
## 1             0             1             0             0             0
## 2             0             0             0             0             0
## 3             0             0             0             0             0
## 4             0             0             0             0             0
## 5             0             0             0             0             0
## 6             0             0             0             0             0
```

Summary of the section II

III. Model Development Process (15 points) - Bethun Bhowmik (in progress)

Build a regression model to predict price. And of course, create the train data set which contains 70% of the data and use set.seed (1023). The remaining 30% will be your test data set. Investigate the data and combine the level of categorical variables if needed and drop variables. For example, you can drop id, Latitude, Longitude, etc.

```
set.seed(1023)
n<-dim(df_house)[1]
IND<-sample(c(1:n),round(n*0.7))
train.dat<-df_house[IND,]
test.dat<-df_house[-c(IND),]

dim(train.dat)
```

```
## [1] 15129   102
```

```
dim(test.dat)
```

```
## [1] 6484  102
```

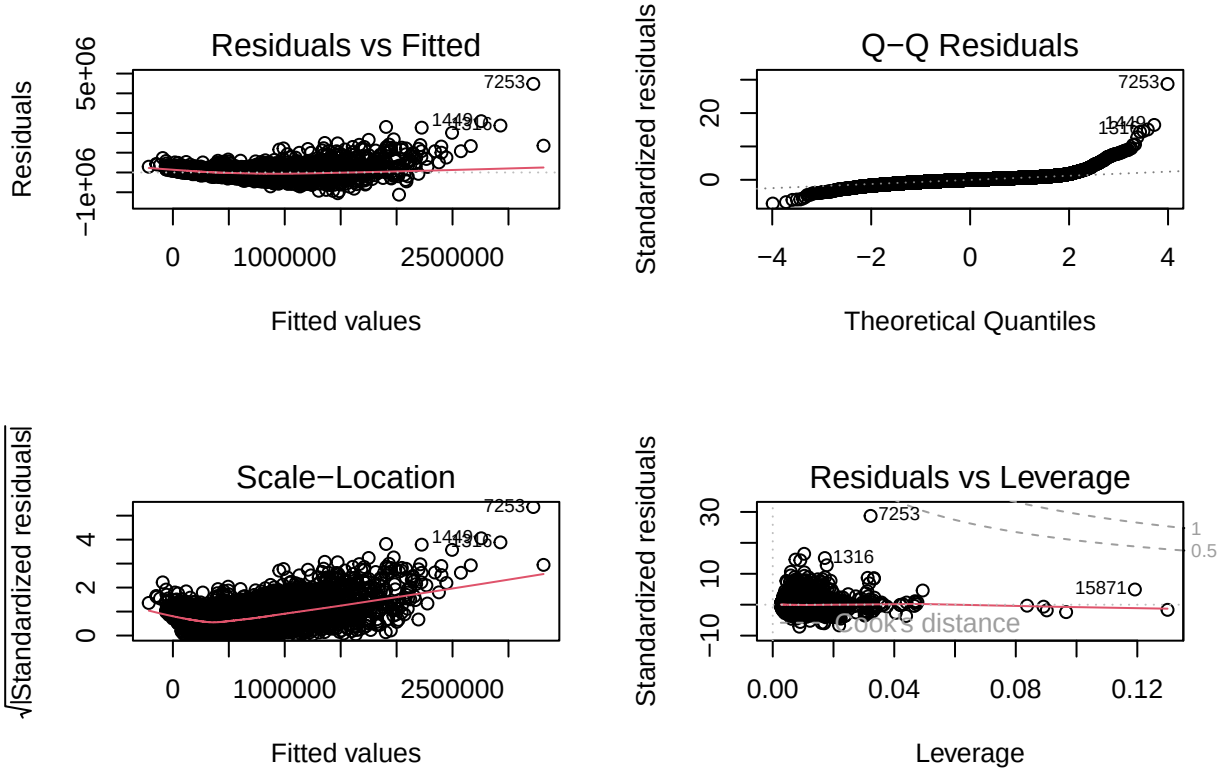
```
house_lm<-lm(price ~.,data=train.dat)
summary(house_lm)
```

```
##
## Call:
## lm(formula = price ~ ., data = train.dat)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1121871   -69702       33    61756  4477878
##
## Coefficients: (9 not defined because of singularities)
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)           -3.764e+10  2.869e+09 -13.121  < 2e-16 ***
## date                  -1.327e-01  2.340e-02  -5.673 1.43e-08 ***
## bedrooms              -2.325e+04  1.796e+03 -12.944  < 2e-16 ***
## bathrooms              2.140e+04  3.104e+03   6.896 5.56e-12 ***
## sqft_living            1.293e+02  4.179e+00  30.939  < 2e-16 ***
## sqft_lot               2.683e-01  4.787e-02   5.605 2.12e-08 ***
## floors                -4.537e+04  3.938e+03 -11.522  < 2e-16 ***
## waterfront             6.222e+05  1.748e+04  35.604  < 2e-16 ***
## view                   5.799e+04  2.093e+03  27.702  < 2e-16 ***
## condition              2.977e+04  2.294e+03  12.978  < 2e-16 ***
## grade                  5.943e+04  2.162e+03  27.484  < 2e-16 ***
## sqft_above             6.984e+01  4.306e+00  16.218  < 2e-16 ***
## sqft_basement                 NA         NA      NA       NA
## yr_built               7.310e+02  3.513e+02   2.081  0.03748 *
## yr_renovated           3.612e+03  4.329e+02   8.344  < 2e-16 ***
## zipcode                2.981e+05  2.515e+04  11.851  < 2e-16 ***
## lat                    1.529e+05  7.498e+04   2.039  0.04145 *
## long                  -1.666e+05  5.369e+04  -3.103  0.00192 **
## sqft_living15          1.538e+01  3.441e+00   4.468 7.96e-06 ***
## sqft_lot15            -7.078e-02  7.114e-02  -0.995  0.31982
## year                   4.234e+06  7.388e+05   5.730 1.02e-08 ***
## month                  3.516e+05  6.168e+04   5.700 1.22e-08 ***
## day                    1.128e+04  2.028e+03   5.560 2.74e-08 ***
## renovated             -7.120e+06  8.572e+05  -8.306  < 2e-16 ***
## yr_built_cat1926-1950  2.775e+03  7.769e+03   0.357  0.72102
```

```
## yr_built_cat1951-1975 -4.989e+04  1.108e+04  -4.501 6.82e-06 ***
## yr_built_cat1976-2000 -6.193e+04  1.520e+04  -4.074 4.64e-05 ***
## yr_built_cat2001+     -5.353e+04  1.890e+04  -2.833  0.00462 **
## `<=1900`                      NA         NA      NA       NA
## `1901-1925`                   NA         NA      NA       NA
## `1926-1950`                   NA         NA      NA       NA
## `1951-1975`                   NA         NA      NA       NA
## `1976-2000`                   NA         NA      NA       NA
## `2001+`                       NA         NA      NA       NA
## age                    8.067e+02  2.992e+02   2.696  0.00702 **
## zipcode_98001          5.874e+07  4.956e+06  11.853  < 2e-16 ***
## zipcode_98002          5.847e+07  4.930e+06  11.860  < 2e-16 ***
## zipcode_98003          5.812e+07  4.906e+06  11.847  < 2e-16 ***
## zipcode_98004          5.858e+07  4.897e+06  11.961  < 2e-16 ***
## zipcode_98005          5.781e+07  4.872e+06  11.867  < 2e-16 ***
## zipcode_98006          5.750e+07  4.843e+06  11.872  < 2e-16 ***
## zipcode_98007          5.717e+07  4.821e+06  11.859  < 2e-16 ***
## zipcode_98008          5.688e+07  4.796e+06  11.860  < 2e-16 ***
## zipcode_98010          5.617e+07  4.729e+06  11.878  < 2e-16 ***
## zipcode_98011          5.582e+07  4.729e+06  11.805  < 2e-16 ***
## zipcode_98014          5.500e+07  4.647e+06  11.836  < 2e-16 ***
## zipcode_98019          5.345e+07  4.525e+06  11.812  < 2e-16 ***
## zipcode_98022          5.252e+07  4.420e+06  11.882  < 2e-16 ***
## zipcode_98023          5.214e+07  4.403e+06  11.841  < 2e-16 ***
## zipcode_98024          5.203e+07  4.389e+06  11.855  < 2e-16 ***
## zipcode_98027          5.117e+07  4.313e+06  11.864  < 2e-16 ***
## zipcode_98028          5.074e+07  4.302e+06  11.796  < 2e-16 ***
## zipcode_98029          5.061e+07  4.264e+06  11.869  < 2e-16 ***
## zipcode_98030          5.010e+07  4.229e+06  11.847  < 2e-16 ***
## zipcode_98031          4.981e+07  4.206e+06  11.843  < 2e-16 ***
## zipcode_98032          4.949e+07  4.180e+06  11.839  < 2e-16 ***
## zipcode_98033          4.951e+07  4.171e+06  11.870  < 2e-16 ***
## zipcode_98034          4.906e+07  4.148e+06  11.825  < 2e-16 ***
## zipcode_98038          4.777e+07  4.027e+06  11.862  < 2e-16 ***
## zipcode_98039          4.850e+07  4.018e+06  12.071  < 2e-16 ***
## zipcode_98040          4.759e+07  3.989e+06  11.931  < 2e-16 ***
## zipcode_98042          4.654e+07  3.927e+06  11.852  < 2e-16 ***
## zipcode_98045          4.578e+07  3.855e+06  11.875  < 2e-16 ***
## zipcode_98052          4.373e+07  3.693e+06  11.843  < 2e-16 ***
## zipcode_98053          4.341e+07  3.667e+06  11.837  < 2e-16 ***
## zipcode_98055          4.268e+07  3.606e+06  11.836  < 2e-16 ***
## zipcode_98056          4.242e+07  3.583e+06  11.840  < 2e-16 ***
## zipcode_98058          4.177e+07  3.529e+06  11.837  < 2e-16 ***
## zipcode_98059          4.152e+07  3.506e+06  11.842  < 2e-16 ***
## zipcode_98065          3.977e+07  3.356e+06  11.851  < 2e-16 ***
## zipcode_98070          3.810e+07  3.228e+06  11.804  < 2e-16 ***
## zipcode_98072          3.768e+07  3.194e+06  11.798  < 2e-16 ***
## zipcode_98074          3.714e+07  3.136e+06  11.844  < 2e-16 ***
## zipcode_98075          3.684e+07  3.109e+06  11.851  < 2e-16 ***
## zipcode_98077          3.617e+07  3.067e+06  11.792  < 2e-16 ***
## zipcode_98092          3.159e+07  2.666e+06  11.849  < 2e-16 ***
## zipcode_98102          2.910e+07  2.434e+06  11.954  < 2e-16 ***
## zipcode_98103          2.860e+07  2.412e+06  11.858  < 2e-16 ***
## zipcode_98105          2.814e+07  2.360e+06  11.921  < 2e-16 ***
## zipcode_98106          2.753e+07  2.328e+06  11.824  < 2e-16 ***
## zipcode_98107          2.741e+07  2.311e+06  11.861  < 2e-16 ***
## zipcode_98108          2.693e+07  2.278e+06  11.819  < 2e-16 ***
## zipcode_98109          2.698e+07  2.259e+06  11.945  < 2e-16 ***
## zipcode_98112          2.623e+07  2.182e+06  12.020  < 2e-16 ***
## zipcode_98115          2.501e+07  2.110e+06  11.853  < 2e-16 ***
```

```
## zipcode_98116          2.468e+07  2.079e+06  11.869  < 2e-16 ***
## zipcode_98117          2.439e+07  2.060e+06  11.840  < 2e-16 ***
## zipcode_98118          2.399e+07  2.026e+06  11.841  < 2e-16 ***
## zipcode_98119          2.398e+07  2.007e+06  11.944  < 2e-16 ***
## zipcode_98122          2.294e+07  1.930e+06  11.887  < 2e-16 ***
## zipcode_98125          2.191e+07  1.860e+06  11.777  < 2e-16 ***
## zipcode_98126          2.161e+07  1.826e+06  11.836  < 2e-16 ***
## zipcode_98133          1.948e+07  1.660e+06  11.737  < 2e-16 ***
## zipcode_98136          1.868e+07  1.574e+06  11.869  < 2e-16 ***
## zipcode_98144          1.633e+07  1.375e+06  11.880  < 2e-16 ***
## zipcode_98146          1.557e+07  1.320e+06  11.795  < 2e-16 ***
## zipcode_98148          1.497e+07  1.266e+06  11.822  < 2e-16 ***
## zipcode_98155          1.291e+07  1.108e+06  11.652  < 2e-16 ***
## zipcode_98166          9.569e+06  8.144e+05  11.750  < 2e-16 ***
## zipcode_98168          8.998e+06  7.658e+05  11.750  < 2e-16 ***
## zipcode_98177          6.412e+06  5.542e+05  11.569  < 2e-16 ***
## zipcode_98178          5.995e+06  5.145e+05  11.652  < 2e-16 ***
## zipcode_98188          3.007e+06  2.606e+05  11.537  < 2e-16 ***
## zipcode_98198                 NA         NA      NA       NA
## zipcode_98199                 NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 158500 on 15033 degrees of freedom
## Multiple R-squared:  0.8092, Adjusted R-squared:  0.808
## F-statistic:   671 on 95 and 15033 DF,  p-value: < 2.2e-16
```

```r
par(mfrow=c(2,2))
plot(house_lm)
```



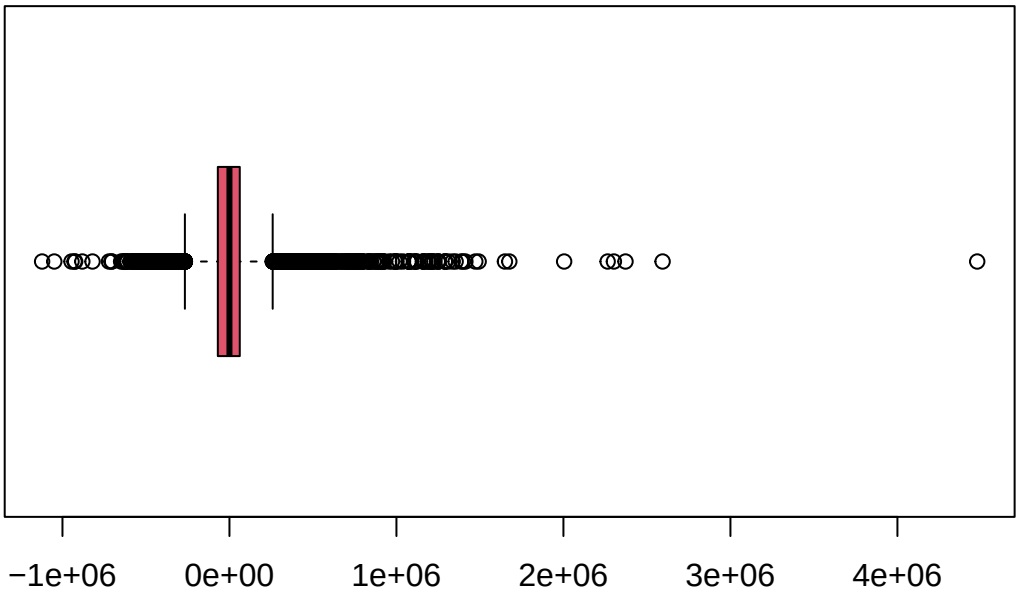
```r
predictors_to_drop <- c("<=1900", "1901-1925", "1926-1950", "1951-1975", "1976-2000", "2001+", "zipcode_98198"

# Update the model by excluding the specified predictors
updated <- as.formula(paste("price ~ .", paste0("- `", predictors_to_drop, "`", collapse = "")))
house_lm <- update(house_lm, formula = updated)
summary(house_lm)
```

```
##
## Call:
## lm(formula = price ~ date + bedrooms + bathrooms + sqft_living +
##     sqft_lot + floors + waterfront + view + condition + grade +
##     sqft_above + yr_built + yr_renovated + zipcode + lat + long +
##     sqft_living15 + sqft_lot15 + year + month + day + renovated +
##     yr_built_cat + age + zipcode_98001 + zipcode_98002 + zipcode_98003 +
##     zipcode_98004 + zipcode_98005 + zipcode_98006 + zipcode_98007 +
##     zipcode_98008 + zipcode_98010 + zipcode_98011 + zipcode_98014 +
##     zipcode_98019 + zipcode_98022 + zipcode_98023 + zipcode_98024 +
##     zipcode_98027 + zipcode_98028 + zipcode_98029 + zipcode_98030 +
##     zipcode_98031 + zipcode_98032 + zipcode_98033 + zipcode_98034 +
##     zipcode_98038 + zipcode_98039 + zipcode_98040 + zipcode_98042 +
##     zipcode_98045 + zipcode_98052 + zipcode_98053 + zipcode_98055 +
##     zipcode_98056 + zipcode_98058 + zipcode_98059 + zipcode_98065 +
##     zipcode_98070 + zipcode_98072 + zipcode_98074 + zipcode_98075 +
##     zipcode_98077 + zipcode_98092 + zipcode_98102 + zipcode_98103 +
##     zipcode_98105 + zipcode_98106 + zipcode_98107 + zipcode_98108 +
##     zipcode_98109 + zipcode_98112 + zipcode_98115 + zipcode_98116 +
##     zipcode_98117 + zipcode_98118 + zipcode_98119 + zipcode_98122 +
##     zipcode_98125 + zipcode_98126 + zipcode_98133 + zipcode_98136 +
##     zipcode_98144 + zipcode_98146 + zipcode_98148 + zipcode_98155 +
##     zipcode_98166 + zipcode_98168 + zipcode_98177 + zipcode_98178 +
##     zipcode_98188, data = train.dat)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1121871   -69702       33    61756  4477878
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)           -3.764e+10  2.869e+09 -13.121  < 2e-16 ***
## date                  -1.327e-01  2.340e-02  -5.673 1.43e-08 ***
## bedrooms              -2.325e+04  1.796e+03 -12.944  < 2e-16 ***
## bathrooms              2.140e+04  3.104e+03   6.896 5.56e-12 ***
## sqft_living            1.293e+02  4.179e+00  30.939  < 2e-16 ***
## sqft_lot               2.683e-01  4.787e-02   5.605 2.12e-08 ***
## floors                -4.537e+04  3.938e+03 -11.522  < 2e-16 ***
## waterfront             6.222e+05  1.748e+04  35.604  < 2e-16 ***
## view                   5.799e+04  2.093e+03  27.702  < 2e-16 ***
## condition              2.977e+04  2.294e+03  12.978  < 2e-16 ***
## grade                  5.943e+04  2.162e+03  27.484  < 2e-16 ***
## sqft_above             6.984e+01  4.306e+00  16.218  < 2e-16 ***
## yr_built               7.310e+02  3.513e+02   2.081  0.03748 *
## yr_renovated           3.612e+03  4.329e+02   8.344  < 2e-16 ***
## zipcode                2.981e+05  2.515e+04  11.851  < 2e-16 ***
## lat                    1.529e+05  7.498e+04   2.039  0.04145 *
## long                  -1.666e+05  5.369e+04  -3.103  0.00192 **
## sqft_living15          1.538e+01  3.441e+00   4.468 7.96e-06 ***
## sqft_lot15            -7.078e-02  7.114e-02  -0.995  0.31982
## year                   4.234e+06  7.388e+05   5.730 1.02e-08 ***
## month                  3.516e+05  6.168e+04   5.700 1.22e-08 ***
```

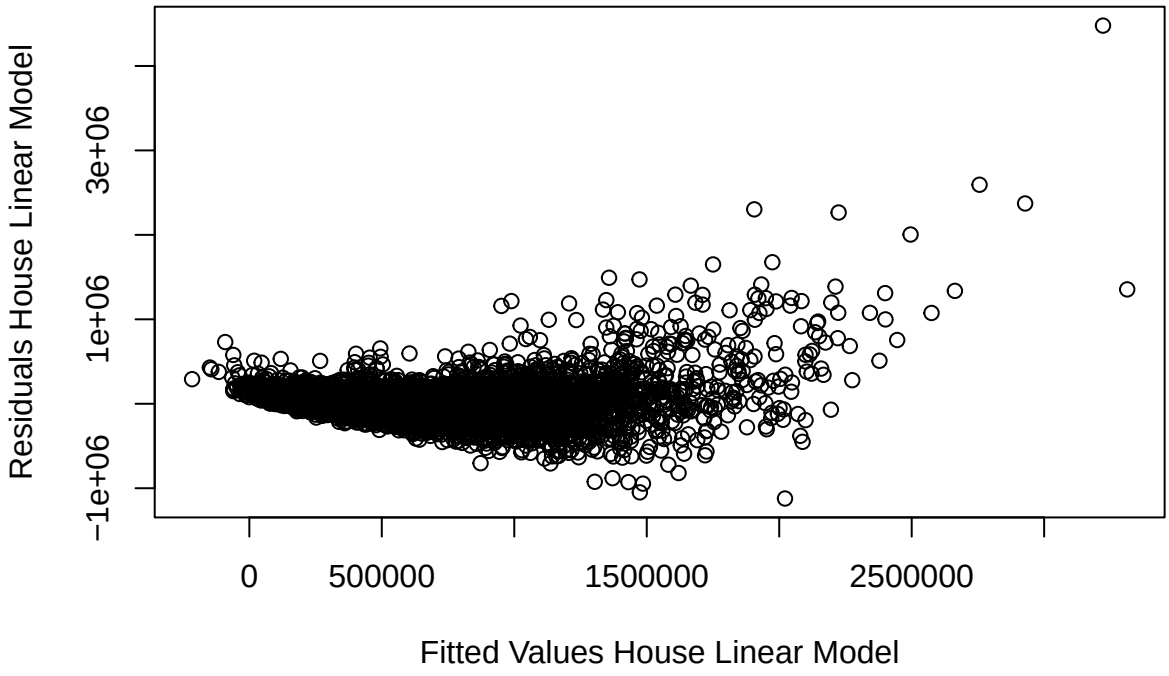
```
## day                    1.128e+04  2.028e+03   5.560 2.74e-08 ***
## renovated             -7.120e+06  8.572e+05  -8.306  < 2e-16 ***
## yr_built_cat1926-1950  2.775e+03  7.769e+03   0.357  0.72102
## yr_built_cat1951-1975 -4.989e+04  1.108e+04  -4.501 6.82e-06 ***
## yr_built_cat1976-2000 -6.193e+04  1.520e+04  -4.074 4.64e-05 ***
## yr_built_cat2001+     -5.353e+04  1.890e+04  -2.833  0.00462 **
## age                    8.067e+02  2.992e+02   2.696  0.00702 **
## zipcode_98001          5.874e+07  4.956e+06  11.853  < 2e-16 ***
## zipcode_98002          5.847e+07  4.930e+06  11.860  < 2e-16 ***
## zipcode_98003          5.812e+07  4.906e+06  11.847  < 2e-16 ***
## zipcode_98004          5.858e+07  4.897e+06  11.961  < 2e-16 ***
## zipcode_98005          5.781e+07  4.872e+06  11.867  < 2e-16 ***
## zipcode_98006          5.750e+07  4.843e+06  11.872  < 2e-16 ***
## zipcode_98007          5.717e+07  4.821e+06  11.859  < 2e-16 ***
## zipcode_98008          5.688e+07  4.796e+06  11.860  < 2e-16 ***
## zipcode_98010          5.617e+07  4.729e+06  11.878  < 2e-16 ***
## zipcode_98011          5.582e+07  4.729e+06  11.805  < 2e-16 ***
## zipcode_98014          5.500e+07  4.647e+06  11.836  < 2e-16 ***
## zipcode_98019          5.345e+07  4.525e+06  11.812  < 2e-16 ***
## zipcode_98022          5.252e+07  4.420e+06  11.882  < 2e-16 ***
## zipcode_98023          5.214e+07  4.403e+06  11.841  < 2e-16 ***
## zipcode_98024          5.203e+07  4.389e+06  11.855  < 2e-16 ***
## zipcode_98027          5.117e+07  4.313e+06  11.864  < 2e-16 ***
## zipcode_98028          5.074e+07  4.302e+06  11.796  < 2e-16 ***
## zipcode_98029          5.061e+07  4.264e+06  11.869  < 2e-16 ***
## zipcode_98030          5.010e+07  4.229e+06  11.847  < 2e-16 ***
## zipcode_98031          4.981e+07  4.206e+06  11.843  < 2e-16 ***
## zipcode_98032          4.949e+07  4.180e+06  11.839  < 2e-16 ***
## zipcode_98033          4.951e+07  4.171e+06  11.870  < 2e-16 ***
## zipcode_98034          4.906e+07  4.148e+06  11.825  < 2e-16 ***
## zipcode_98038          4.777e+07  4.027e+06  11.862  < 2e-16 ***
## zipcode_98039          4.850e+07  4.018e+06  12.071  < 2e-16 ***
## zipcode_98040          4.759e+07  3.989e+06  11.931  < 2e-16 ***
## zipcode_98042          4.654e+07  3.927e+06  11.852  < 2e-16 ***
## zipcode_98045          4.578e+07  3.855e+06  11.875  < 2e-16 ***
## zipcode_98052          4.373e+07  3.693e+06  11.843  < 2e-16 ***
## zipcode_98053          4.341e+07  3.667e+06  11.837  < 2e-16 ***
## zipcode_98055          4.268e+07  3.606e+06  11.836  < 2e-16 ***
## zipcode_98056          4.242e+07  3.583e+06  11.840  < 2e-16 ***
## zipcode_98058          4.177e+07  3.529e+06  11.837  < 2e-16 ***
## zipcode_98059          4.152e+07  3.506e+06  11.842  < 2e-16 ***
## zipcode_98065          3.977e+07  3.356e+06  11.851  < 2e-16 ***
## zipcode_98070          3.810e+07  3.228e+06  11.804  < 2e-16 ***
## zipcode_98072          3.768e+07  3.194e+06  11.798  < 2e-16 ***
## zipcode_98074          3.714e+07  3.136e+06  11.844  < 2e-16 ***
## zipcode_98075          3.684e+07  3.109e+06  11.851  < 2e-16 ***
## zipcode_98077          3.617e+07  3.067e+06  11.792  < 2e-16 ***
## zipcode_98092          3.159e+07  2.666e+06  11.849  < 2e-16 ***
## zipcode_98102          2.910e+07  2.434e+06  11.954  < 2e-16 ***
## zipcode_98103          2.860e+07  2.412e+06  11.858  < 2e-16 ***
## zipcode_98105          2.814e+07  2.360e+06  11.921  < 2e-16 ***
## zipcode_98106          2.753e+07  2.328e+06  11.824  < 2e-16 ***
## zipcode_98107          2.741e+07  2.311e+06  11.861  < 2e-16 ***
## zipcode_98108          2.693e+07  2.278e+06  11.819  < 2e-16 ***
## zipcode_98109          2.698e+07  2.259e+06  11.945  < 2e-16 ***
## zipcode_98112          2.623e+07  2.182e+06  12.020  < 2e-16 ***
## zipcode_98115          2.501e+07  2.110e+06  11.853  < 2e-16 ***
## zipcode_98116          2.468e+07  2.079e+06  11.869  < 2e-16 ***
## zipcode_98117          2.439e+07  2.060e+06  11.840  < 2e-16 ***
## zipcode_98118          2.399e+07  2.026e+06  11.841  < 2e-16 ***
```

```
## zipcode_98119          2.398e+07  2.007e+06  11.944  < 2e-16 ***
## zipcode_98122          2.294e+07  1.930e+06  11.887  < 2e-16 ***
## zipcode_98125          2.191e+07  1.860e+06  11.777  < 2e-16 ***
## zipcode_98126          2.161e+07  1.826e+06  11.836  < 2e-16 ***
## zipcode_98133          1.948e+07  1.660e+06  11.737  < 2e-16 ***
## zipcode_98136          1.868e+07  1.574e+06  11.869  < 2e-16 ***
## zipcode_98144          1.633e+07  1.375e+06  11.880  < 2e-16 ***
## zipcode_98146          1.557e+07  1.320e+06  11.795  < 2e-16 ***
## zipcode_98148          1.497e+07  1.266e+06  11.822  < 2e-16 ***
## zipcode_98155          1.291e+07  1.108e+06  11.652  < 2e-16 ***
## zipcode_98166          9.569e+06  8.144e+05  11.750  < 2e-16 ***
## zipcode_98168          8.998e+06  7.658e+05  11.750  < 2e-16 ***
## zipcode_98177          6.412e+06  5.542e+05  11.569  < 2e-16 ***
## zipcode_98178          5.995e+06  5.145e+05  11.652  < 2e-16 ***
## zipcode_98188          3.007e+06  2.606e+05  11.537  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 158500 on 15033 degrees of freedom
## Multiple R-squared:  0.8092, Adjusted R-squared:  0.808
## F-statistic:   671 on 95 and 15033 DF,  p-value: < 2.2e-16
```

```r
ei<-house_lm$residuals
boxplot(ei,horizontal=TRUE,staplewex=0.5,col=2,xlab="House Price Regression Residuals")
```




```r
plot(house_lm$fitted.values,ei,xlab="Fitted Values House Linear Model",ylab="Residuals House Linear Model")
```



H_o: Error variances are constant H_a: Error variances are not constant Decision Rule is if statistic> critical reject the null or if p-value < alpha (0.01) reject the hull

P value is < 2.2e-16. So we reject H_o, Error variances are not constant.

```
library(lmtest)
```

```
## Loading required package: zoo
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
bptest(house_lm, studentize = FALSE)
```

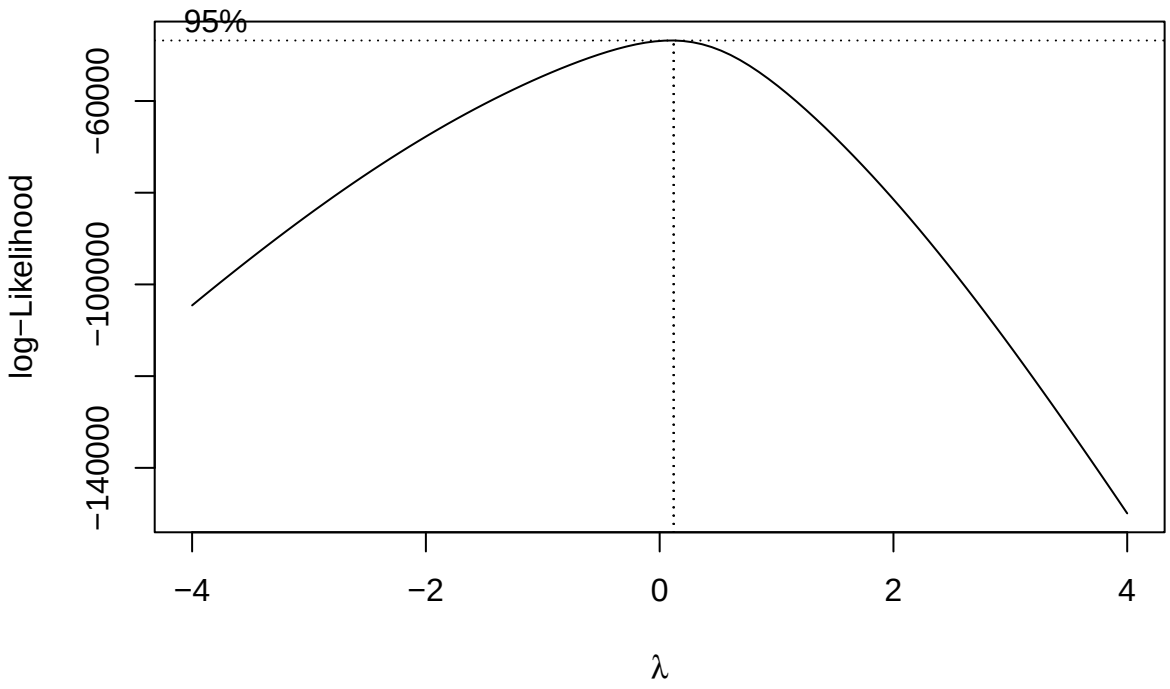
```
##
##  Breusch-Pagan test
##
## data:  house_lm
## BP = 68820, df = 95, p-value < 2.2e-16
```

```
library(MASS)
```

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
par(mfrow=c(1,1))
bc<-boxcox(house_lm,lambda=seq(-4,4,by=0.1))
```




```
lambda <- bc$x[which.max(bc$y)]
lambda  #This is the optimal lambda
```

```
## [1] 0.1212121
```

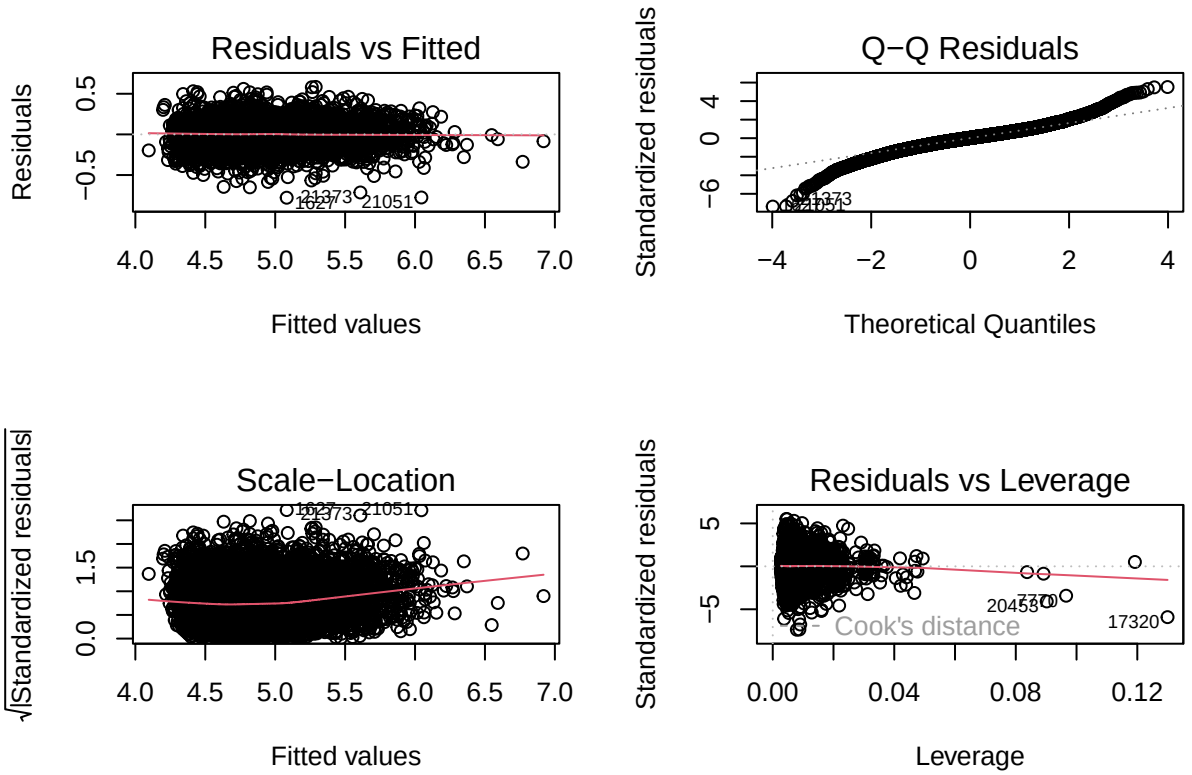
```
house_lm1<-lm(price^lambda~.,data=train.dat)
summary(house_lm1)
```

```
##
## Call:
## lm(formula = price^lambda ~ ., data = train.dat)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.77908 -0.05649  0.00374  0.05913  0.58381
##
## Coefficients: (9 not defined because of singularities)
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)           -4.189e+04  1.919e+03 -21.828  < 2e-16 ***
## date                  -1.213e-07  1.565e-08  -7.751 9.70e-15 ***
## bedrooms               8.424e-04  1.201e-03   0.701 0.483207
## bathrooms              1.999e-02  2.076e-03   9.631  < 2e-16 ***
## sqft_living            8.036e-05  2.795e-06  28.748  < 2e-16 ***
```

```
## sqft_lot               4.138e-07  3.202e-08  12.924  < 2e-16 ***
## floors                -2.664e-02  2.634e-03 -10.115  < 2e-16 ***
## waterfront             2.989e-01  1.169e-02  25.572  < 2e-16 ***
## view                   3.837e-02  1.400e-03  27.404  < 2e-16 ***
## condition              3.829e-02  1.534e-03  24.955  < 2e-16 ***
## grade                  5.461e-02  1.446e-03  37.761  < 2e-16 ***
## sqft_above             4.668e-05  2.881e-06  16.206  < 2e-16 ***
## sqft_basement                 NA         NA      NA       NA
## yr_built               5.809e-04  2.350e-04   2.472 0.013458 *
## yr_renovated           2.844e-03  2.895e-04   9.823  < 2e-16 ***
## zipcode                3.485e-01  1.682e-02  20.715  < 2e-16 ***
## lat                    2.800e-01  5.015e-02   5.582 2.42e-08 ***
## long                  -2.176e-01  3.591e-02  -6.059 1.40e-09 ***
## sqft_living15          4.858e-05  2.302e-06  21.103  < 2e-16 ***
## sqft_lot15             5.993e-08  4.759e-08   1.259 0.207896
## year                   3.871e+00  4.942e-01   7.833 5.07e-15 ***
## month                  3.213e-01  4.126e-02   7.787 7.29e-15 ***
## day                    1.033e-02  1.357e-03   7.613 2.84e-14 ***
## renovated             -5.596e+00  5.733e-01  -9.761  < 2e-16 ***
## yr_built_cat1926-1950 -5.541e-03  5.197e-03  -1.066 0.286311
## yr_built_cat1951-1975 -4.205e-02  7.414e-03  -5.672 1.43e-08 ***
## yr_built_cat1976-2000 -3.586e-02  1.017e-02  -3.527 0.000421 ***
## yr_built_cat2001+     -2.930e-02  1.264e-02  -2.318 0.020469 *
## `<=1900`                      NA         NA      NA       NA
## `1901-1925`                   NA         NA      NA       NA
## `1926-1950`                   NA         NA      NA       NA
## `1951-1975`                   NA         NA      NA       NA
## `1976-2000`                   NA         NA      NA       NA
## `2001+`                       NA         NA      NA       NA
## age                    4.843e-04  2.001e-04   2.420 0.015518 *
## zipcode_98001          6.865e+01  3.315e+00  20.711  < 2e-16 ***
## zipcode_98002          6.830e+01  3.298e+00  20.711  < 2e-16 ***
## zipcode_98003          6.796e+01  3.282e+00  20.709  < 2e-16 ***
## zipcode_98004          6.819e+01  3.276e+00  20.818  < 2e-16 ***
## zipcode_98005          6.761e+01  3.259e+00  20.749  < 2e-16 ***
## zipcode_98006          6.722e+01  3.240e+00  20.750  < 2e-16 ***
## zipcode_98007          6.688e+01  3.225e+00  20.740  < 2e-16 ***
## zipcode_98008          6.654e+01  3.208e+00  20.742  < 2e-16 ***
## zipcode_98010          6.571e+01  3.163e+00  20.772  < 2e-16 ***
## zipcode_98011          6.531e+01  3.163e+00  20.648  < 2e-16 ***
## zipcode_98014          6.430e+01  3.108e+00  20.687  < 2e-16 ***
## zipcode_98019          6.251e+01  3.027e+00  20.652  < 2e-16 ***
## zipcode_98022          6.143e+01  2.957e+00  20.776  < 2e-16 ***
## zipcode_98023          6.095e+01  2.945e+00  20.695  < 2e-16 ***
## zipcode_98024          6.086e+01  2.936e+00  20.729  < 2e-16 ***
## zipcode_98027          5.987e+01  2.885e+00  20.756  < 2e-16 ***
## zipcode_98028          5.936e+01  2.877e+00  20.631  < 2e-16 ***
## zipcode_98029          5.922e+01  2.852e+00  20.763  < 2e-16 ***
## zipcode_98030          5.857e+01  2.829e+00  20.704  < 2e-16 ***
## zipcode_98031          5.823e+01  2.813e+00  20.697  < 2e-16 ***
## zipcode_98032          5.782e+01  2.796e+00  20.681  < 2e-16 ***
## zipcode_98033          5.786e+01  2.790e+00  20.737  < 2e-16 ***
## zipcode_98034          5.736e+01  2.775e+00  20.672  < 2e-16 ***
## zipcode_98038          5.588e+01  2.694e+00  20.745  < 2e-16 ***
## zipcode_98039          5.610e+01  2.687e+00  20.877  < 2e-16 ***
## zipcode_98040          5.550e+01  2.668e+00  20.803  < 2e-16 ***
## zipcode_98042          5.441e+01  2.626e+00  20.715  < 2e-16 ***
## zipcode_98045          5.357e+01  2.579e+00  20.773  < 2e-16 ***
## zipcode_98052          5.117e+01  2.470e+00  20.715  < 2e-16 ***
## zipcode_98053          5.080e+01  2.453e+00  20.709  < 2e-16 ***
```

```
## zipcode_98055          4.988e+01  2.412e+00  20.681  < 2e-16 ***
## zipcode_98056          4.962e+01  2.397e+00  20.703  < 2e-16 ***
## zipcode_98058          4.886e+01  2.360e+00  20.697  < 2e-16 ***
## zipcode_98059          4.859e+01  2.345e+00  20.719  < 2e-16 ***
## zipcode_98065          4.658e+01  2.245e+00  20.751  < 2e-16 ***
## zipcode_98070          4.469e+01  2.159e+00  20.700  < 2e-16 ***
## zipcode_98072          4.409e+01  2.136e+00  20.637  < 2e-16 ***
## zipcode_98074          4.348e+01  2.098e+00  20.730  < 2e-16 ***
## zipcode_98075          4.314e+01  2.079e+00  20.746  < 2e-16 ***
## zipcode_98077          4.233e+01  2.052e+00  20.631  < 2e-16 ***
## zipcode_98092          3.696e+01  1.783e+00  20.727  < 2e-16 ***
## zipcode_98102          3.390e+01  1.628e+00  20.819  < 2e-16 ***
## zipcode_98103          3.344e+01  1.613e+00  20.730  < 2e-16 ***
## zipcode_98105          3.284e+01  1.579e+00  20.798  < 2e-16 ***
## zipcode_98106          3.214e+01  1.557e+00  20.640  < 2e-16 ***
## zipcode_98107          3.205e+01  1.546e+00  20.735  < 2e-16 ***
## zipcode_98108          3.149e+01  1.524e+00  20.661  < 2e-16 ***
## zipcode_98109          3.147e+01  1.511e+00  20.831  < 2e-16 ***
## zipcode_98112          3.049e+01  1.460e+00  20.884  < 2e-16 ***
## zipcode_98115          2.927e+01  1.411e+00  20.736  < 2e-16 ***
## zipcode_98116          2.889e+01  1.391e+00  20.773  < 2e-16 ***
## zipcode_98117          2.854e+01  1.378e+00  20.713  < 2e-16 ***
## zipcode_98118          2.805e+01  1.355e+00  20.699  < 2e-16 ***
## zipcode_98119          2.798e+01  1.343e+00  20.837  < 2e-16 ***
## zipcode_98122          2.683e+01  1.291e+00  20.789  < 2e-16 ***
## zipcode_98125          2.564e+01  1.244e+00  20.604  < 2e-16 ***
## zipcode_98126          2.529e+01  1.221e+00  20.707  < 2e-16 ***
## zipcode_98133          2.277e+01  1.110e+00  20.502  < 2e-16 ***
## zipcode_98136          2.189e+01  1.053e+00  20.788  < 2e-16 ***
## zipcode_98144          1.909e+01  9.197e-01  20.758  < 2e-16 ***
## zipcode_98146          1.819e+01  8.830e-01  20.602  < 2e-16 ***
## zipcode_98148          1.746e+01  8.468e-01  20.619  < 2e-16 ***
## zipcode_98155          1.508e+01  7.409e-01  20.358  < 2e-16 ***
## zipcode_98166          1.125e+01  5.447e-01  20.648  < 2e-16 ***
## zipcode_98168          1.044e+01  5.122e-01  20.382  < 2e-16 ***
## zipcode_98177          7.510e+00  3.707e-01  20.258  < 2e-16 ***
## zipcode_98178          6.989e+00  3.442e-01  20.309  < 2e-16 ***
## zipcode_98188          3.485e+00  1.743e-01  19.993  < 2e-16 ***
## zipcode_98198                 NA         NA      NA       NA
## zipcode_98199                 NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.106 on 15033 degrees of freedom
## Multiple R-squared:  0.887,  Adjusted R-squared:  0.8863
## F-statistic:  1242 on 95 and 15033 DF,  p-value: < 2.2e-16
```

```
par(mfrow=c(2,2))
plot(house_lm1)
```




```
pred<- predict(house_lm1,test.dat)^(1/lambda)
act<-test.dat$price

SST<-var(act)*(length(act)-1)
SSE<-sum((act-pred)^2)
R.Square.Test <- 1- SSE/SST
R.Square.Test
```

```
## [1] 0.8430772
```

```
R.Square.Train=0.887

knitr::kable(cbind(R.Square.Train,R.Square.Test), align = "c",
      caption = "Pseudo $R^2$ values for Train and Test")
```

Table 2: Pseudo R^2 values for Train and Test

R.Square.Train	R.Square.Test
0.887	0.8430772

IV. Model Performance Testing (15 points)

Use the test data set to assess the model performances. Here, build the best multiple linear models by using the stepwise both ways selection method. Compare the performance of the best two linear models. Make sure that model assumption(s) are checked for the final linear model. Apply remedy measures (transformation, etc.) that helps satisfy the assumptions. In particular you must deeply investigate unequal variances and multicollinearity. If necessary, apply remedial methods (WLS, Ridge, Elastic Net, Lasso, etc.).

##Best Subset Regression

```
library(olsrr)
```

```
## Warning: package 'olsrr' was built under R version 4.3.2
```

```
##
## Attaching package: 'olsrr'
```

```
## The following object is masked from 'package:MASS':
##
##     cement
```

```
## The following object is masked from 'package:datasets':
##
##     rivers
```

```
house_lm2<-lm(price^lambda~.,data=test.dat)
# k1<-ols_step_best_subset(house_lm2)
# k1
# plot(k1,guide="none")
```

V. Challenger Models (15 points)

Build an alternative model based on one of the following approaches to predict price: regression tree, NN, or SVM. Explore using a logistic regression. Check the applicable model assumptions. Apply in-sample and out-of-sample testing, backtesting and review the comparative goodness of fit of the candidate models. Describe step by step your procedure to get to the best model and why you believe it is fit for purpose.

##Random Forest

```
# [Kaleo]
# importing library
#install.packages("randomForest")
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.3.2
```

```
## randomForest 4.7-1.1
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

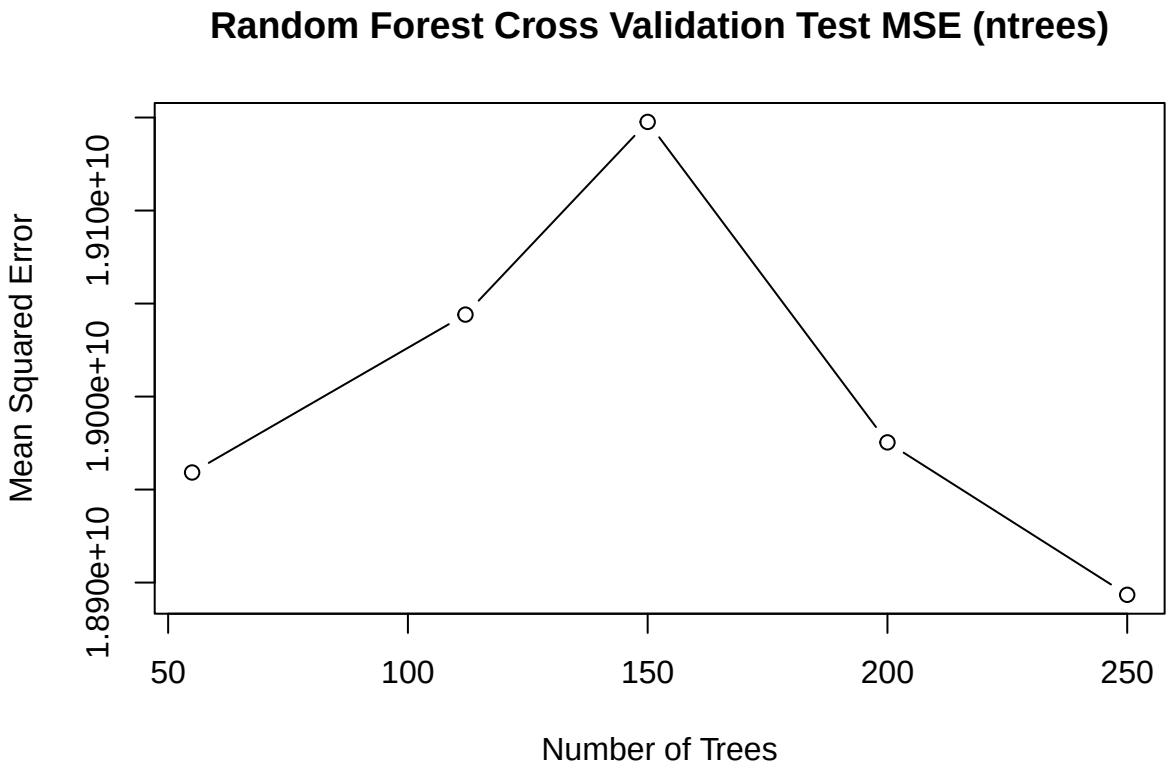
```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

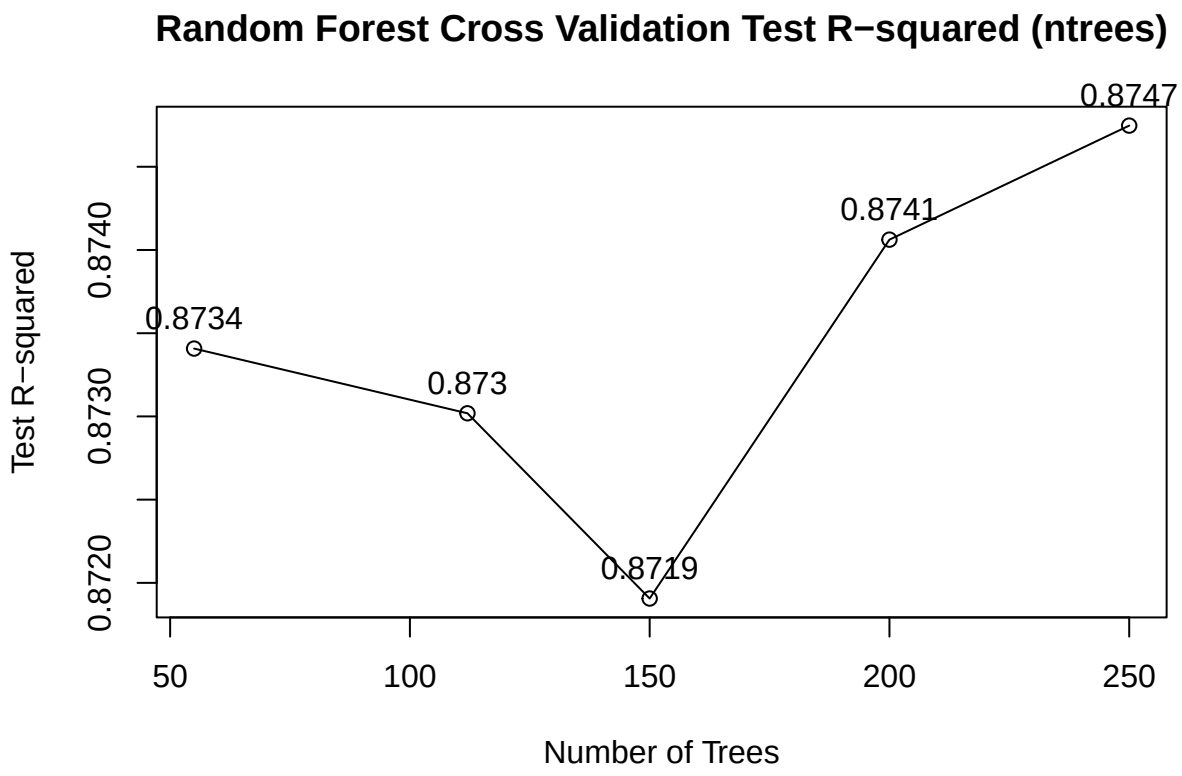
```
# [Kaleo]
# changing column names to fit randomforest
og.names = colnames(train.dat)
col_names_rf = c("until_1900","from.1901_1925","from.1926_1950","from.1951_1975",
                 "from.1976_2000","from.2001_after")
for (i in 26:31){
  colnames(train.dat)[i] = col_names_rf[i-25]
  colnames(test.dat)[i] = col_names_rf[i-25]
}
```

```
ntree_values <- c(55, 112, 150, 200, 250)

mse_values = numeric(length(ntree_values))
test_rsq_values = numeric(length(ntree_values))
train_rsq_values = numeric(length(ntree_values))

# commented out cross validation to save computation time!!

# for (i in seq_along(ntree_values)) {
#   ntree = ntree_values[i]
#
#   model = randomForest(price ~ ., data = train.dat, ntree = ntree)
#   predictions_test <- predict(model, newdata = test.dat)
#   predictions_train <- predict(model, newdata = train.dat)
#
#   mse_values[i] <- mean((predictions_test - test.dat$price)^2)
#   test_rsq_values[i] = cor(predictions_test,test.dat$price)^2
#   train_rsq_values[i] = cor(predictions_train,train.dat$price)^2
# }
```

```r
mse_hard.coded = c(18959180557,19044061189,19147650224,18975362291,18893426330)
test_rsq_values_hard.coded = c(0.8734077,0.8730189,0.8719060,0.8740622,0.8747472)

# Plot the results
plot(ntree_values, mse_hard.coded, type = "b", xlab = "Number of Trees", ylab = "Mean Squared Error", main = "
```




```r
plot(ntree_values, test_rsq_values_hard.coded,type="o", xlab = "Number of Trees", ylab = "Test R-squared", mai
text(ntree_values,test_rsq_values_hard.coded,labels=round(test_rsq_values_hard.coded,4),pos=3,xpd=TRUE)
```




```
# [Kaleo]

#commented out for computation time..

#using best model (250 trees...for now)
# house_rf = randomForest(price ~ ., data = train.dat, ntree = 250)
# rf_test_pred = predict(house_rf,newdata=test.dat)
# rf_train_pred = predict(house_rf,newdata=train.dat)
#
# rf_test_mse = mean((rf_test_pred - test.dat$price)^2)
# rf_test_rsq = cor(rf_test_pred,test.dat$price)^2
# rf_train_rsq = cor(rf_train_pred,train.dat$price)^2
#
# cat("test mse:",rf_test_mse,"train rsq:", rf_train_rsq, "test rsq:",rf_test_rsq)
```

```
# [Kaleo]
# change column names back
colnames(train.dat) = og.names
colnames(test.dat) = og.names
```

VI. Model Limitation and Assumptions (15 points)

Based on the performances on both train and test data sets, determine your primary (champion) model and the other model which would be your benchmark model. Validate your models using the test sample. Do the residuals look normal? Does it matter given your technique? How is the prediction performance using Pseudo R^2, SSE, RMSE? Benchmark the model against alternatives. How good is the relative fit? Are there any serious violations of the model assumptions? Has the model had issues or limitations that the user must know? (Which assumptions are needed to support the Champion model?)

VII. Ongoing Model Monitoring Plan (5 points)

How would you picture the model needing to be monitored, which quantitative thresholds and triggers would you set to decide when the model needs to be replaced? What are the assumptions that the model must comply with for its continuous use?

VIII. Conclusion (5 points)

Summarize your results here. What is the best model for the data and why?

Bibliography (7 points)

Please include all references, articles and papers in this section.

Appendix (3 points)

Please add any additional supporting graphs, plots and data analysis.