House Price Prediction Project - GAM

Yohan Chandrasukmana - 01112190011

Contents

"Saya Yohan Chandrasukmana, menyatakan bahwa saya mengerjakan soal-soal ini secara mandiri tanpa bantuan orang lain ataupun memberikan bantuan kepada orang lain. Jika saya terbukti melakukan kecurangan tersebut (mendapat bantuan dari orang lain ataupun memberi bantuan kepada orang lain) maka saya bersedia untuk tidak lulus dalam mata kuliah ini."

0.1 Loading Library

Library used in this study case are as follows.

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.4     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.0.1     v forcats 0.5.1

## Warning: package 'dplyr' was built under R version 4.1.2

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

```
library(caret) # for splitting train-test set
```

```
## Warning: package 'caret' was built under R version 4.1.2

## Loading required package: lattice

##
## Attaching package: 'caret'

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

```
library(ggpubr) # for arranging ggplot
```

```
## Warning: package 'ggpubr' was built under R version 4.1.2
```

```
library(gam) # for GAM
```

```
## Warning: package 'gam' was built under R version 4.1.2

## Loading required package: splines
```

```
## Loading required package: foreach

## Warning: package 'foreach' was built under R version 4.1.2

##
## Attaching package: 'foreach'

## The following objects are masked from 'package:purrr':
##
##     accumulate, when

## Loaded gam 1.20
```

```
library(car) # for calculating collinearity with vif
```

```
## Warning: package 'car' was built under R version 4.1.2

## Loading required package: carData

## Warning: package 'carData' was built under R version 4.1.2

##
## Attaching package: 'car'

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

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

---

1 Data Description

Data used in this study case are house sales data.

1.1 Data Loading and Prerequisites

Data saved in the CSV files will be loaded using the function `read.csv`. A seed will also be set for the project so that we will obtain the same results in each run.

```
rm(list=ls())   # clean up workspace
set.seed(1)
sales <- read.csv('salesData19.csv')
head(sales)
```

```
##   ID City Province     SALEDT  Price   SQFT RMTOT RMBED BATH FLR1AREA BSMT
## 1  1   54        2 2017-05-12 893000  22100     5     5    4     2130    7
## 2  2   54        2 2013-07-05 209000   3276     2     2    1      738    7
## 3  3   33        3 2016-05-11 165000  10000     3     3    1     1293    7
## 4  4   54        2 2013-06-20 449900  83876     5     5    3     1781    7
## 5  5   54        2 2017-02-22 396764  69019     3     3    2     1788    7
## 6  6   54        2 2014-06-30 533754 105484     4     4    3     1891    5
##   HEATSYS ATTIC SFLA GRADE SALE_YEAR SALE_MONTH EFF_AGE Style
## 1       5     1 4829     B        -1          5      11     U
## 2       3     1  738     C        -5          7      50     U
## 3       3     1 1293     C        -2          5      35     U
## 4       5     1 3470     C        -5          6      11     U
## 5       5     0 2326     C        -1          2       1     U
## 6       5     0 4510     B        -4          6       2     U
```

---

2 Data Cleanup

First, the data set will be cleaned.

```
# Checking for missing values & removing them
table(is.na(sales))
```

```
##
##  FALSE
## 570266
```

```
sales = na.omit(sales)

# Data summary
summary(sales)
```

```
##        ID             City          Province        SALEDT
##  Min.   :    1   Min.   :20.00   Min.   :1.000   Length:30014
##  1st Qu.: 7504   1st Qu.:54.00   1st Qu.:2.000   Class :character
##  Median :15008   Median :54.00   Median :2.000   Mode  :character
##  Mean   :15008   Mean   :56.02   Mean   :2.022
##  3rd Qu.:22511   3rd Qu.:65.00   3rd Qu.:2.000
##  Max.   :30014   Max.   :98.00   Max.   :3.000
##      Price              SQFT              RMTOT            RMBED
##  Min.   :   1200   Min.   :0.000e+00   Min.   : 1.000   Min.   : 1.000
##  1st Qu.: 133000   1st Qu.:6.022e+03   1st Qu.: 3.000   1st Qu.: 3.000
##  Median : 205000   Median :1.145e+04   Median : 3.000   Median : 3.000
##  Mean   : 229836   Mean   :2.308e+05   Mean   : 3.234   Mean   : 3.193
##  3rd Qu.: 291500   3rd Qu.:4.066e+04   3rd Qu.: 4.000   3rd Qu.: 4.000
##  Max.   :5000000   Max.   :2.123e+09   Max.   :18.000   Max.   :13.000
##       BATH           FLR1AREA          BSMT          HEATSYS
##  Min.   : 1.000   Min.   : 192   Min.   :1.000   Min.   :1.000
##  1st Qu.: 1.000   1st Qu.: 750   1st Qu.:6.000   1st Qu.:1.000
##  Median : 1.000   Median :1008   Median :7.000   Median :2.000
```

```
##  Mean   : 1.637   Mean   :1081   Mean   :5.963   Mean   :2.446
##  3rd Qu.: 2.000   3rd Qu.:1286   3rd Qu.:7.000   3rd Qu.:3.000
##  Max.   :12.000   Max.   :5366   Max.   :7.000   Max.   :6.000
##      ATTIC             SFLA          GRADE             SALE_YEAR
##  Min.   :0.0000   Min.   :  192   Length:30014       Min.   :-5.000
##  1st Qu.:1.0000   1st Qu.: 1204   Class :character   1st Qu.:-4.000
##  Median :1.0000   Median : 1610   Mode  :character   Median :-2.000
##  Mean   :0.8773   Mean   : 1711                      Mean   :-2.765
##  3rd Qu.:1.0000   3rd Qu.: 2070                      3rd Qu.:-1.000
##  Max.   :1.0000   Max.   :10784                      Max.   :-1.000
##    SALE_MONTH        EFF_AGE          Style
##  Min.   : 1.000   Min.   :  0.00   Length:30014
##  1st Qu.: 5.000   1st Qu.: 15.00   Class :character
##  Median : 7.000   Median : 29.00   Mode  :character
##  Mean   : 7.029   Mean   : 51.98
##  3rd Qu.: 9.000   3rd Qu.: 40.00
##  Max.   :12.000   Max.   :468.00
```

```
# Dropping ID variable since we have the same values in the data frame's index
sales <- subset(sales, select = -c(ID))

# Changing SALEDT variable into date format in R and renaming the column
sales$SALEDT <- as.Date(sales$SALEDT)
names(sales)[names(sales) == 'SALEDT'] <- 'DATE'

# Formatting categorical variables as factors
categorical <- c('City',
                 'Province',
                 'BSMT',
                 'HEATSYS',
                 'ATTIC',
                 'GRADE',
                 'Style')
sales[, categorical] = lapply(sales[, categorical], factor)
```

---

3 Exploratory Data Analysis

We shall now explore the data set.

In this project, the variables will be divided as follows.

- Dependent Variable: `Price`
- Independent Variables:
  - Numerical Variables: `SQFT`, `FLR1AREA`, `SFLA`, `EFF_AGE`
  - Categorical Variables: `City`, `Province`, `BSMT`, `ATTIC`, `GRADE`, `Style`
  - Numerical/Categorical Variables: `RMTOT`, `RMBED`, `BATH`

Furthermore, the Numerical/Categorical variables will be treated as numerical variables. This is to fit future data that might not be classified into the specified categories, but could be defined as in between of categories.

For example, a house has 2 bathrooms an additional small bathroom. It is unjust to classify the house to have 2 or 3 values in the `BATH` variable. Instead, the house could be given a value of 2.5.

The independent variables will be kept into variables for this project.

```
var.num <- c("RMTOT", "RMBED", "BATH", "SQFT", "FLR1AREA", "SFLA", "EFF_AGE")
var.categ <- c("City", "Province", "BSMT", "ATTIC", "GRADE", "Style")
```

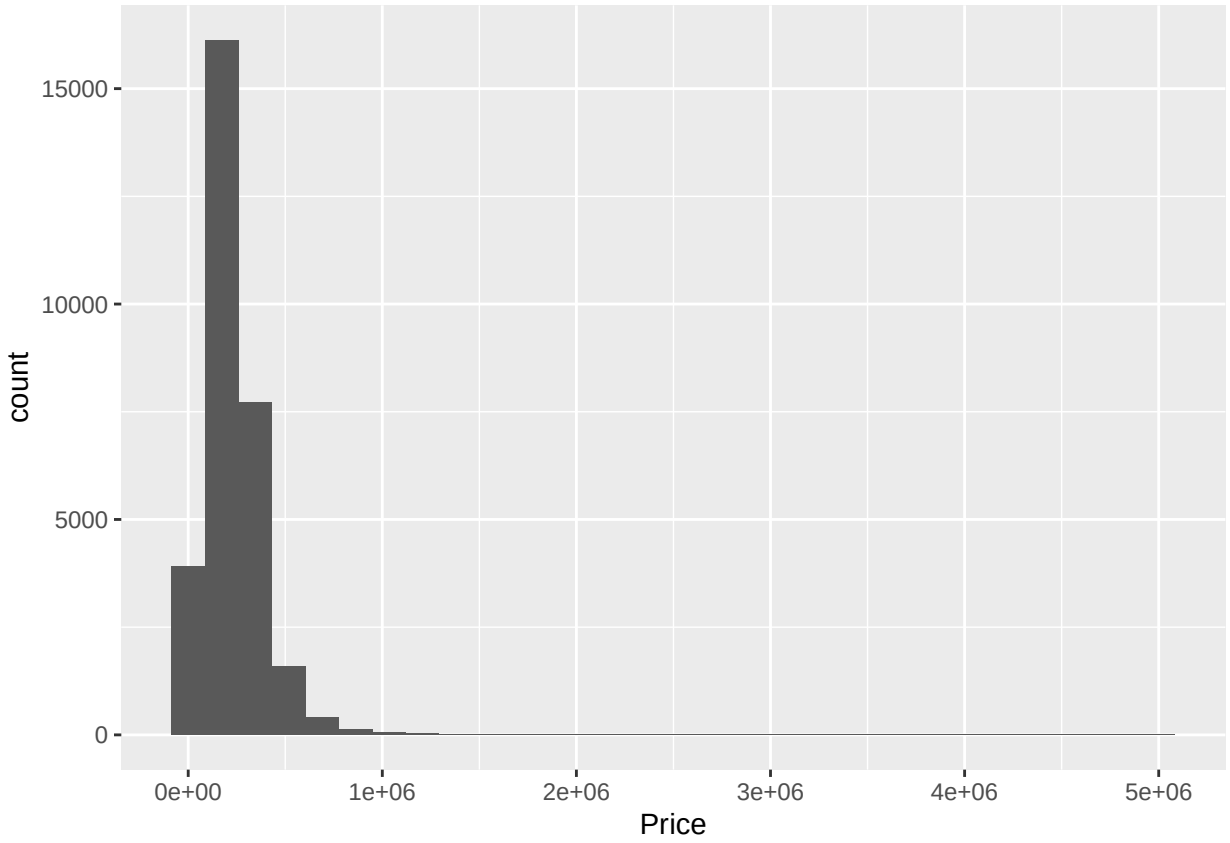
We will analyze the distribution of the `Price` variable and check for outliers in the data.

```
summary(sales$Price)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1200  133000  205000  229836  291500 5000000
```

```
ggplot(sales) + geom_histogram(aes(x = Price))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



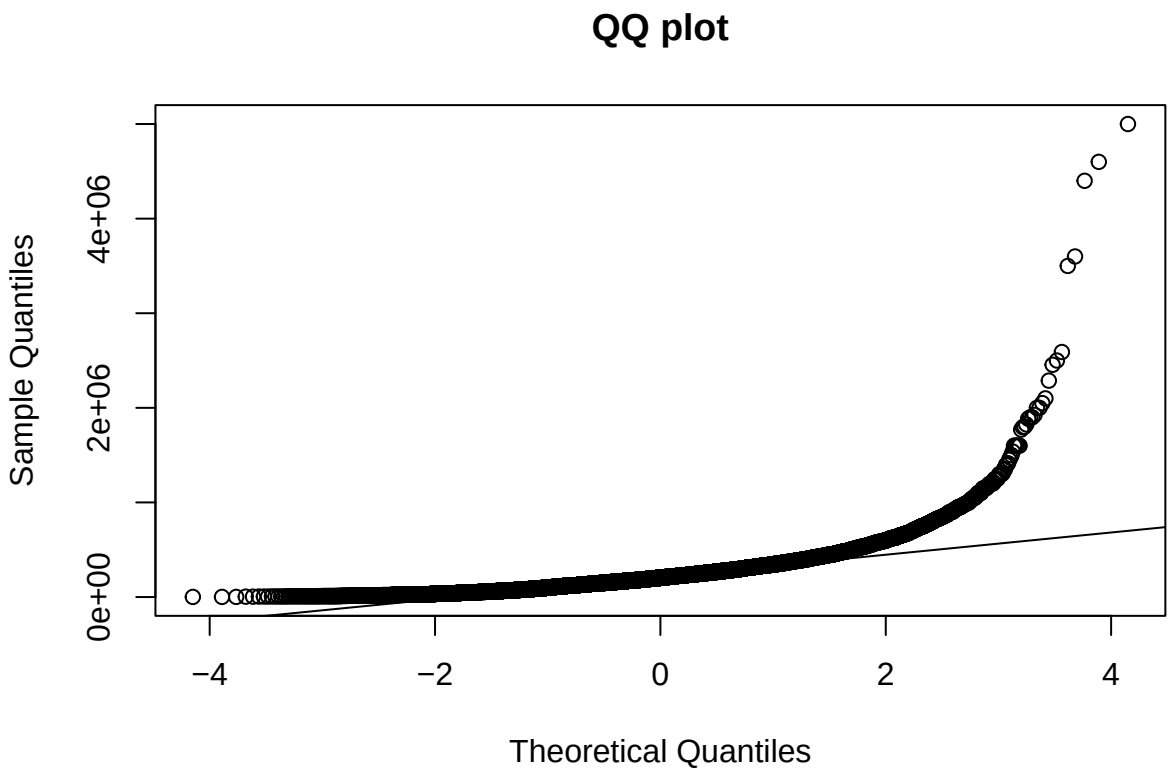

```
ks.test(sales$Price, "pnorm")
```

```
## Warning in ks.test(sales$Price, "pnorm"): ties should not be present for the
## Kolmogorov-Smirnov test
```

```
##
##  One-sample Kolmogorov-Smirnov test
##
## data:  sales$Price
## D = 1, p-value < 2.2e-16
## alternative hypothesis: two-sided
```

From the histogram, we can see that the data is strongly skewed and not normally distributed. With a p-value of 2.2*10^(-6), we also obtain the same conclusion that the data is not from a normal distribution from the Kolmogorov-Smirnov test.

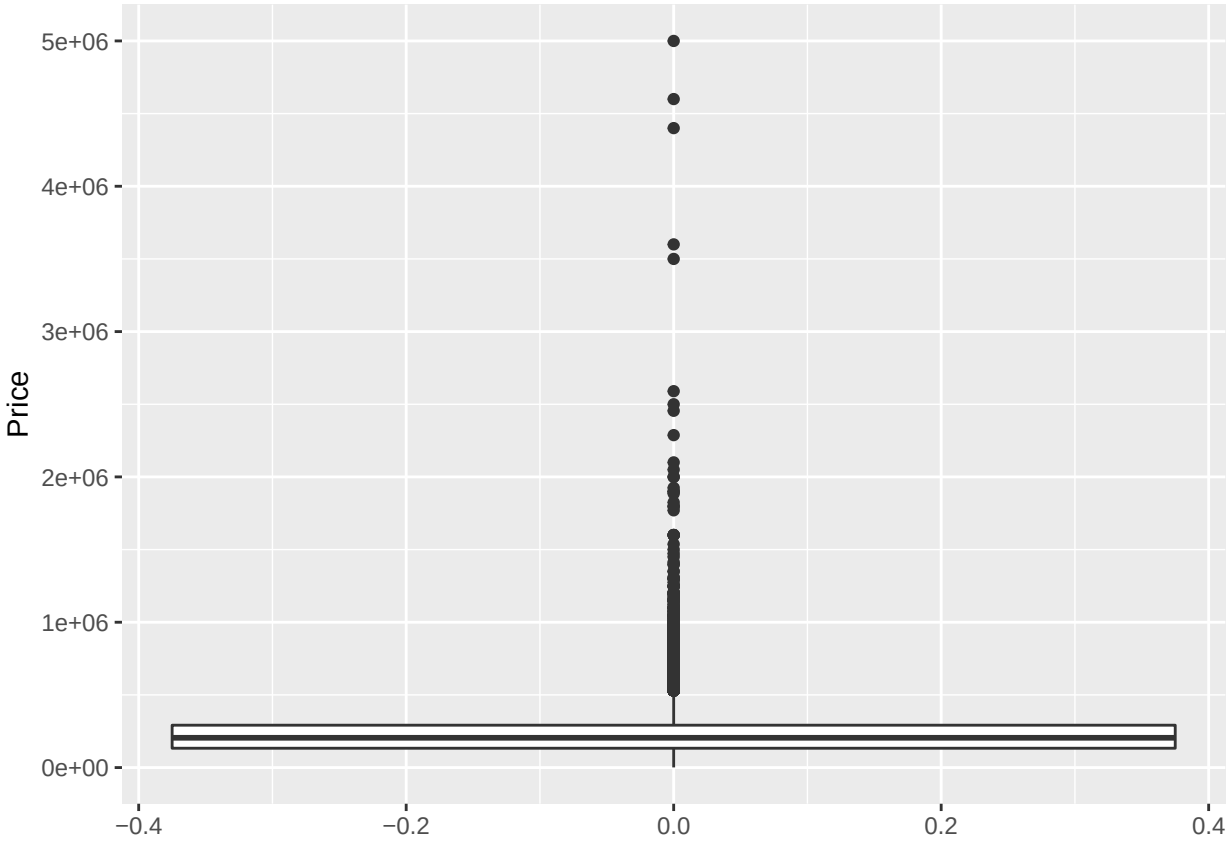
```
# Checking the distribution of the dependent variable Price by plotting its histogram
qqnorm(sales$Price,main="QQ plot")
qqline(sales$Price)
```



In addition to the previous normality test with Kolmogorov-Smirnov, we can see through the QQ plot that the data is skewed and is not normally distributed.

Now, we are going to analyze outliers that might exist in the dependent variable.

```
ggplot(sales) + geom_boxplot(aes(y = Price))
```



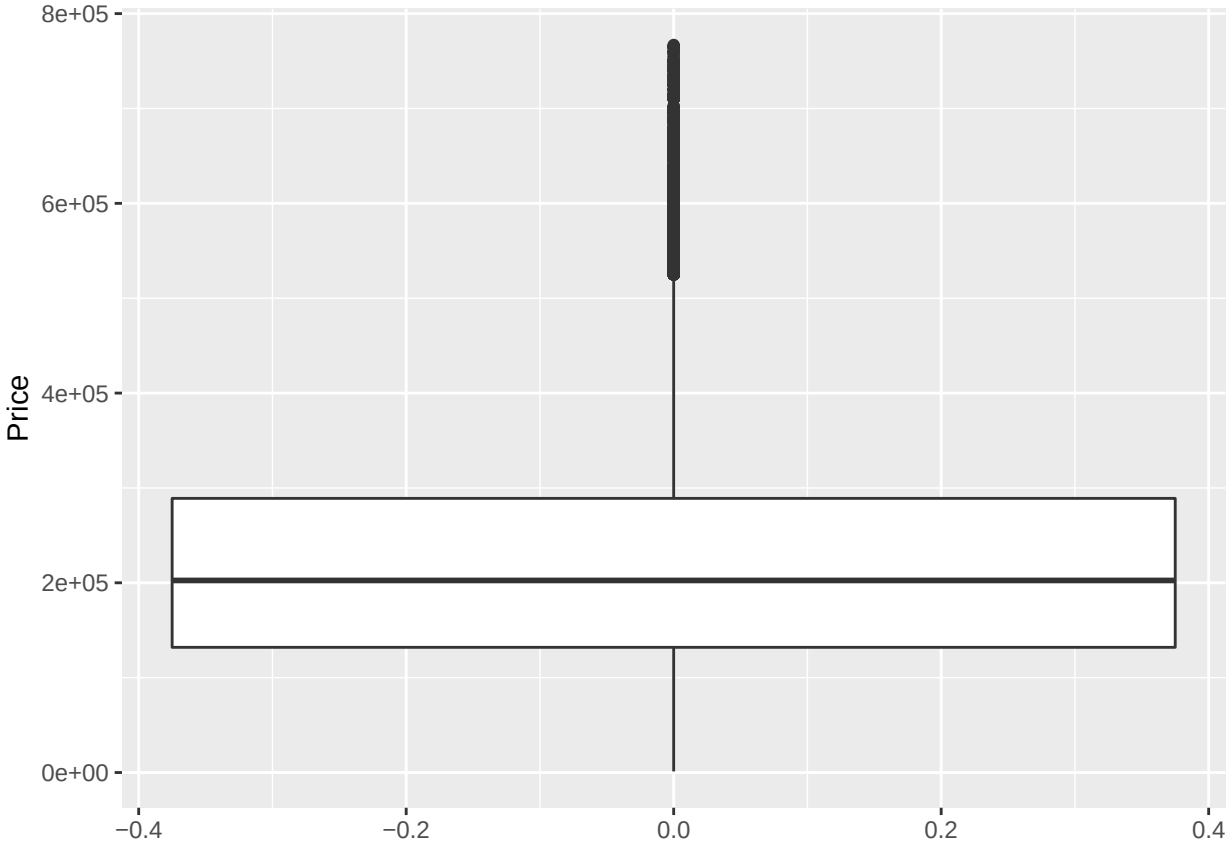
Here, we can conclude that we have outliers present in the response variable. This might be the reason why the distribution of the data is strongly skewed. As these outliers might affect our models, we will remove them. An entry of the data is considered as an outlier with the following formula:

$$outliers = 3 * InterquartileRange/1.5$$

```
# Total number of outliers:
paste("outliers:", length(boxplot.stats(sales$Price, coef = 3)$out))
```

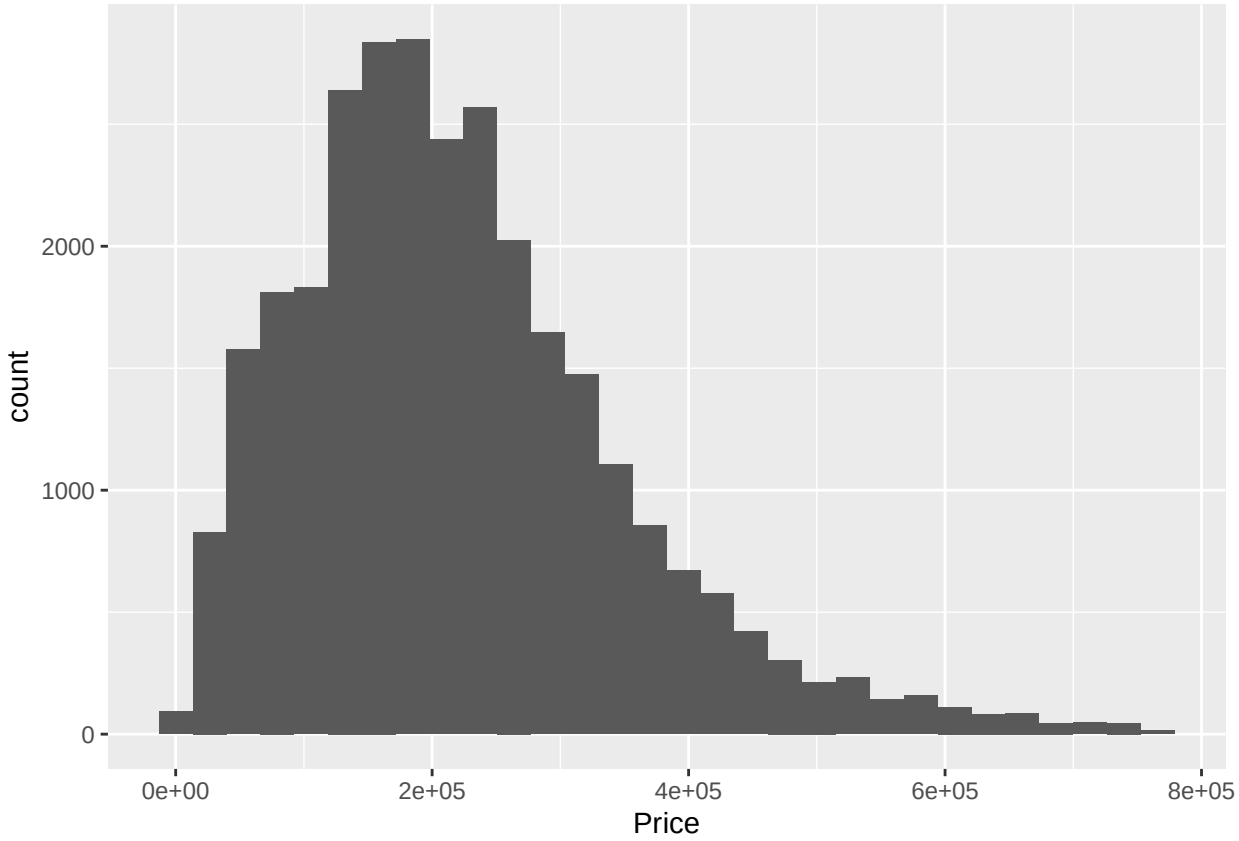
```
## [1] "outliers: 273"
```

```
out = boxplot.stats(sales$Price, coef = 3)$out
out_key = which(sales$Price %in% c(out))
sales = sales[-out_key, ]

# Checking the boxplot of the data with removed outliers
ggplot(sales) + geom_boxplot(aes(y = Price))
```




```
ggplot(sales) + geom_histogram(aes(x = Price))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



We can see that the boxplot is now slightly better and the histogram of the prices is also less skewed.

After analyzing the response variable, we can now analyze the independent variables, which include `City`, `Province`, `SQFT`, `RMTOT`, `RMBED`, `BATH`, `FLR1AREA`, `BSMT`, `ATTIC`, `SFLA`, `GRADE`, `EFF_AGE`, `Style`.

```
summary(sales$City)
```

```
##    20    21    23    24    25    27    28    30    31    32    33    43    44
##   722    32   204    60    71   363   121    83    13    55  2886   851   368
##    45    47    48    49    54    59    60    62    64    65    66    67    69
##    52   540   253    29 13874   756   484   116   304    97   198   184  1667
##    70    71    72    74    75    76    77    78    80    81    82    83    84
##    97   285   196   391   826   339   124    45   534   392   113   145    64
##    85    87    88    89    91    92    93    95    96    97    98
##   121   177    43   308    19   130    74   178   188   334   235
```

```
summary(sales$Province)
```

```
##     1     2     3
##  5020 19048  5673
```

```
summary(sales$SQFT)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## 0.000e+00 6.018e+03 1.140e+04 2.325e+05 4.060e+04 2.123e+09
```

```
summary(sales$RMTOT)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   3.000   3.000   3.224   4.000  18.000
```

```
summary(sales$RMBED)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   3.000   3.000   3.183   4.000  13.000
```

```
summary(sales$BATH)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   1.000   1.000   1.624   2.000   7.000
```

```
summary(sales$FLR1AREA)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     192     750    1008    1074    1279    5366
```

```
summary(sales$BSMT)
```

```
##     1     2     3     4     5     6     7
##  1540  2532   933   233  1917   740 21846
```

```
summary(sales$ATTIC)
```

```
##     0     1
##  3631 26110
```

```
summary(sales$SFLA)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     192    1200    1602    1691    2055    8580
```

```
summary(sales$GRADE)
```

```
##     A     B     C     D     E
##    44  2530 23248  3098   821
```

```
summary(sales$EFF_AGE)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.00   15.00   29.00   52.23   40.00  468.00
```

```
summary(sales$Style)
```

```
##     B     H     R     S     U
##  3607  1898   655  1102 22479
```

From the EDA, it can be concluded that there are some findings that could be highlighted and more data cleaning could be done towards their implications.

```
min(sales$SQFT)
```

```
## [1] 0
```

```
# Since it is impossible to have a house with 0 square feet, we will remove them.
sales = sales[sales$SQFT != 0,]
```

```
head(sort(summary(sales$City)))
```

```
## 31 91 21 49 88 78
## 13 19 28 29 43 45
```

As shown above, we can see that there are some cities with relatively few entries. Therefore, the `City` variable must be taken into context during train-test splitting, especially `Cities` 31, 91, and 21 with the lowest amount of entries. Additionally, since it is unknown what cities these numbers represent, we are unable to combine nor remove these variables.

We will now observe the correlation between the dependent variable and the numeric independent variables.

```
cor(sales$Price, sales$SQFT)
```

```
## [1] -0.01112671
```

```
cor(sales$Price, sales$FLR1AREA)
```

```
## [1] 0.4389079
```

```
cor(sales$Price, sales$SFLA)
```

```
## [1] 0.7160272
```

```
cor(sales$Price, sales$EFF_AGE)
```

```
## [1] -0.3590222
```

We can see that the relationships between `Price` and `SQFT` `Price` and `EFF_AGE` are not significant, while the results also show that there is a moderate correlation between `Price` and `FLR1AREA`. On the other hand, we can see that the correlation of the `SFLA` variable is significant. Therefore, we can assume that `SFLA` will be significant in our models as well.

Now, we will also analyze the correlation between categorical/numerical variables and the outcome variable.

```
cor(sales$Price, sales$RMTOT)
```

```
## [1] 0.2742038
```

```
cor(sales$Price, sales$RMBED)
```

```
## [1] 0.2795559
```

```
cor(sales$Price, sales$BATH)
```

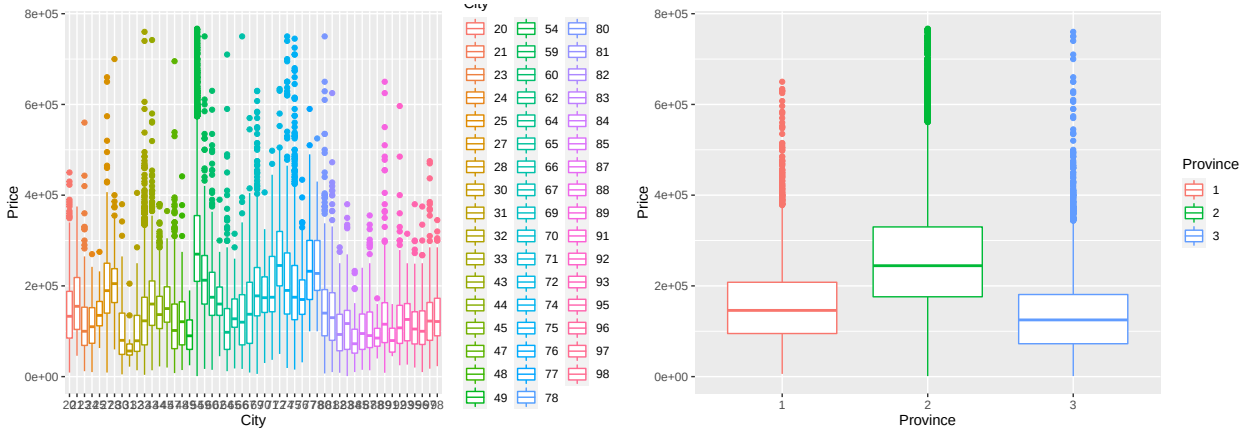
```
## [1] 0.6013735
```

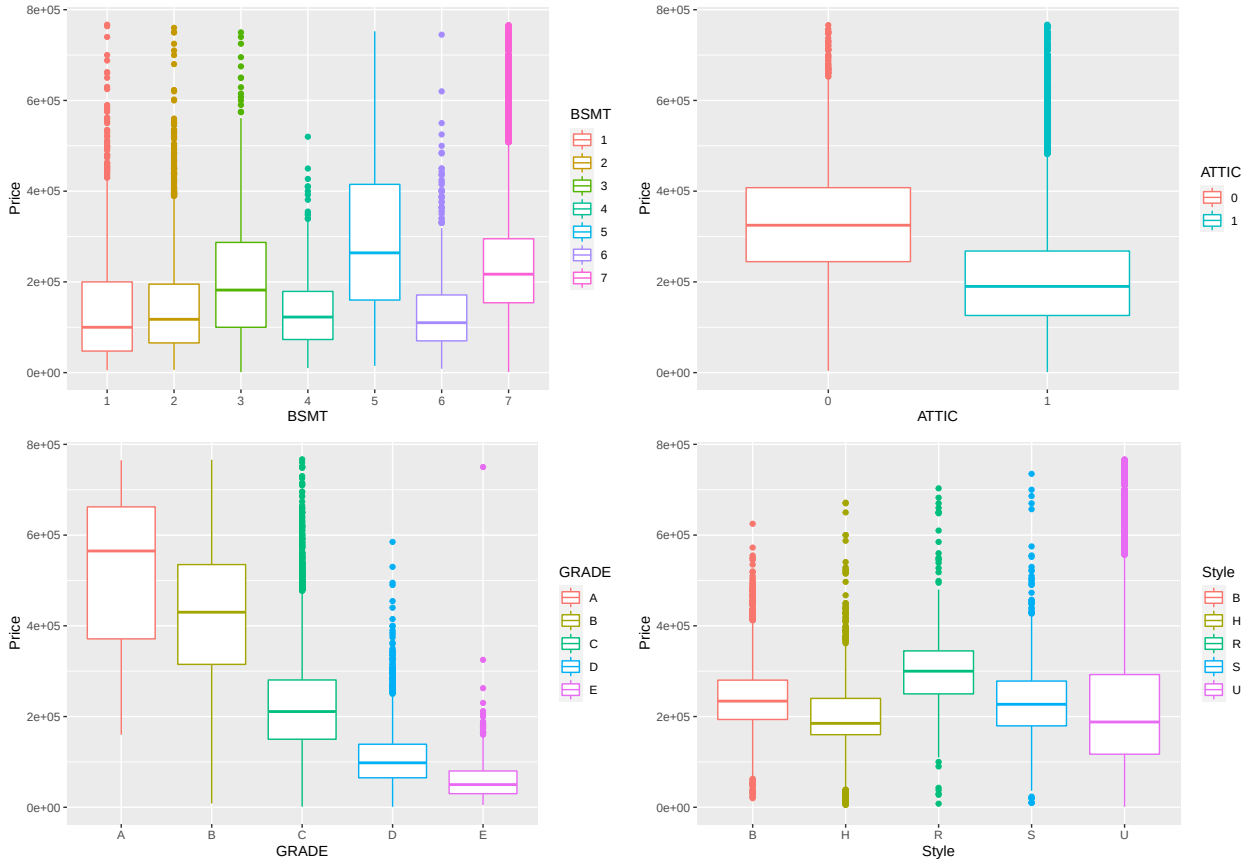
As we can see, the correlation shows that there the relationship between `RMTOT` and `RMBED` towards the dependent variable. On the other hand, `BATH` shows a relatively significant positive correlation towards the prices.

We will now compare the correlation between the variables with box plots and scatter plots for the categorical independent variables.

```
lapply(var.categ, function(x) ggplot(sales, aes_string(x=x, y="Price", color=x)) + geom_boxplot())
```

```
## [[1]]

##
## [[2]]

##
## [[3]]

##
## [[4]]

##
## [[5]]

##
## [[6]]
```





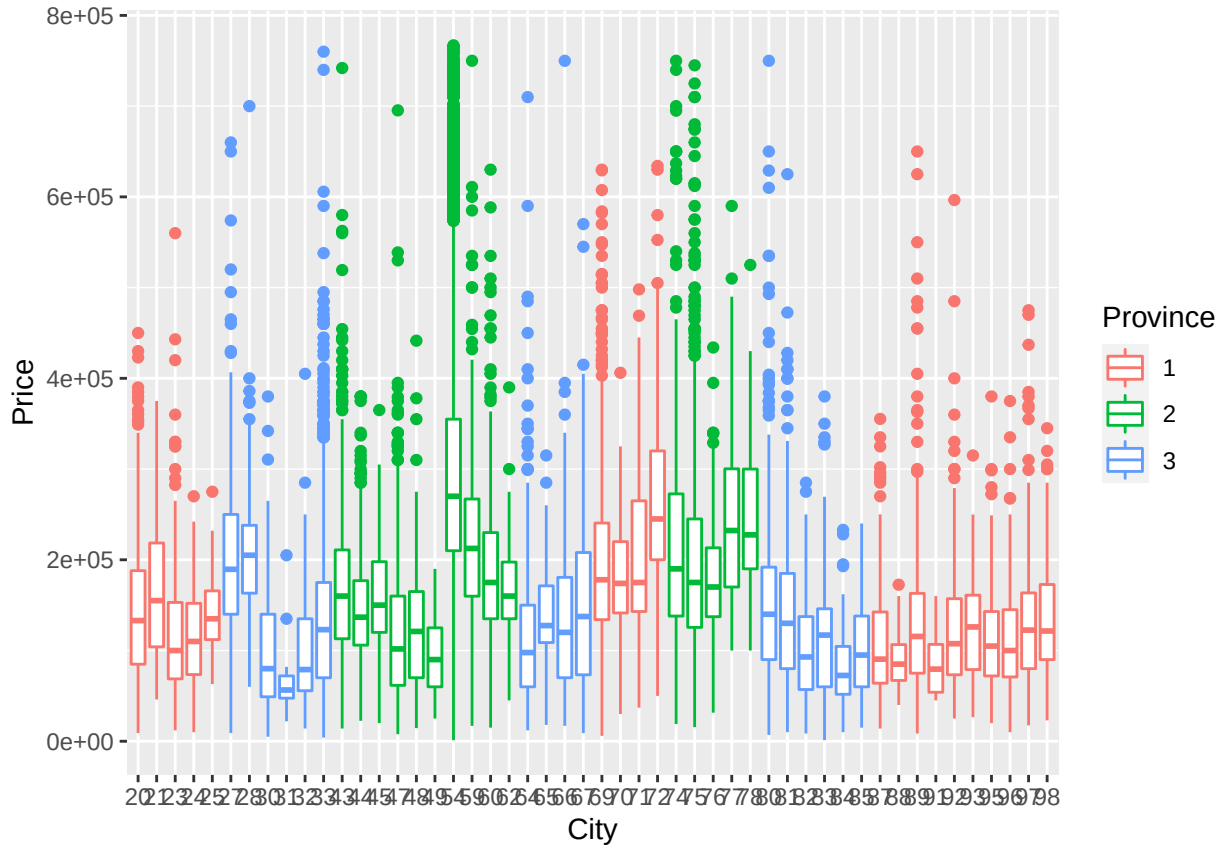
In the box plots above, we can see how each categorical variable relates towards the prices. Overall, there are not a lot of correlation between most categorical variables towards the prices. However, one results that stand out is in the `GRADE` box plot. We can see that the average price of the houses decreases with the `GRADE`. We can also observe the existence of some outliers in each grade category, one of them being a data point that is far on top in Category 'E'.

Other than that, it can be observed that there is a slight decrease of price as a house owns an attic and house prices in Province 2 has the most expensive houses on average, followed by Provinces 1 and 3. and since a city is dependent on a certain province,

Additionally, prices vary in different cities. Furthermore, there may exist a dependency or relationship between `City` and `Province` as both explain geographical location of a certain house: the use of one variable renders the other redundant. However, since the both cities and provinces are only represented in numerical values, the model may not be able to observe their dependency. Therefore, it is crucial to select one of the two variables for modelling.

Since the `City` variable is more detailed in comparison to the provinces, we will exclude the `Province` variable from the model. Additionally, the price of a house is also more likely to be more expensive on the city of the respective house, instead of the province which represents a broader area and too generalized.

```
ggplot(sales) + geom_boxplot(aes(x = City, y = Price, color = Province))
```



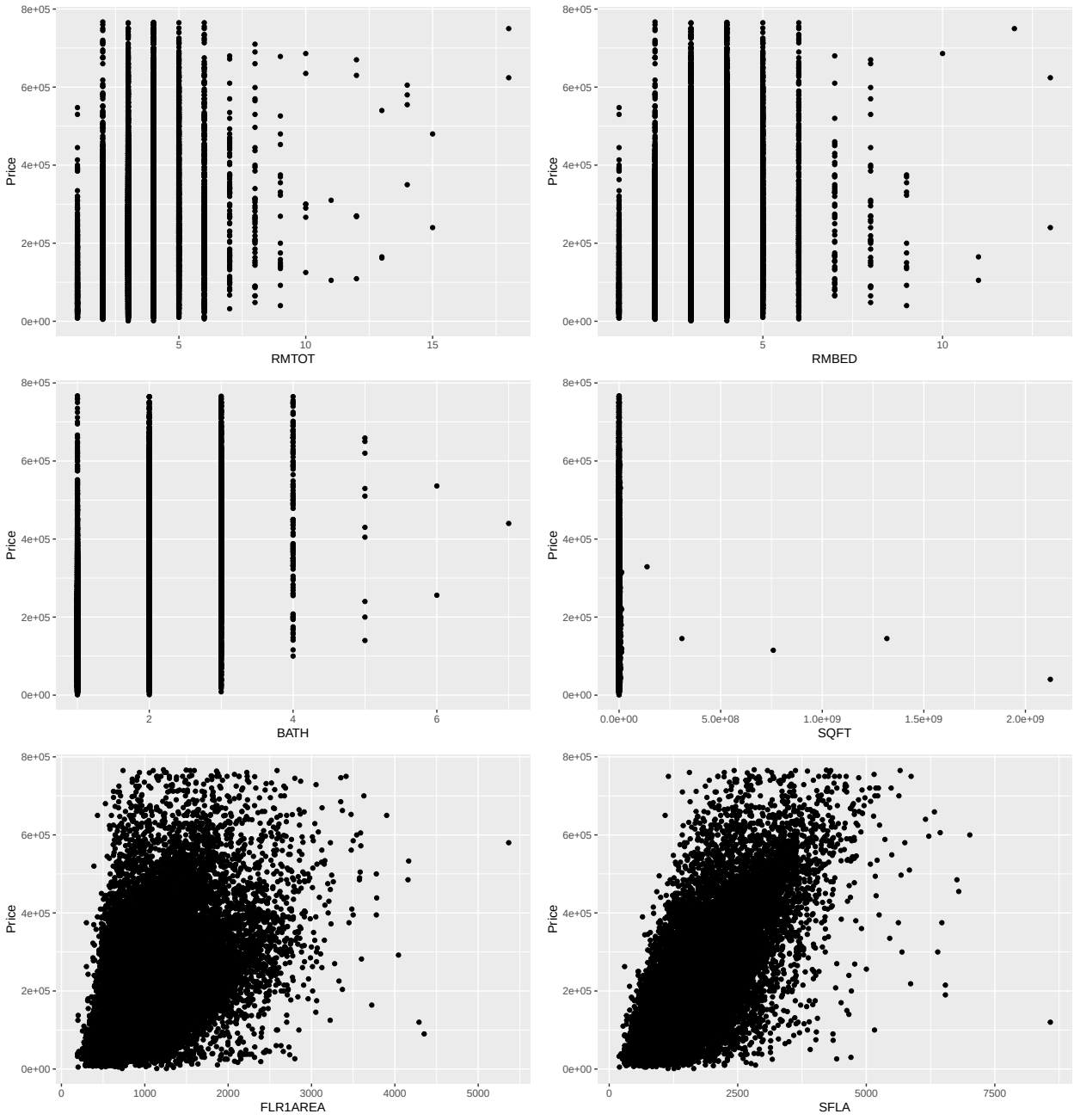
As shown on the figure above, we can see that prices vary based on their cities and somewhat independent of their provinces. Moreover, the cities represent a larger population of the data and is more inclusive rather than the provinces. Consequently, the `City` variable becomes a more suitable candidate in the feature selection, rather than `Province`.

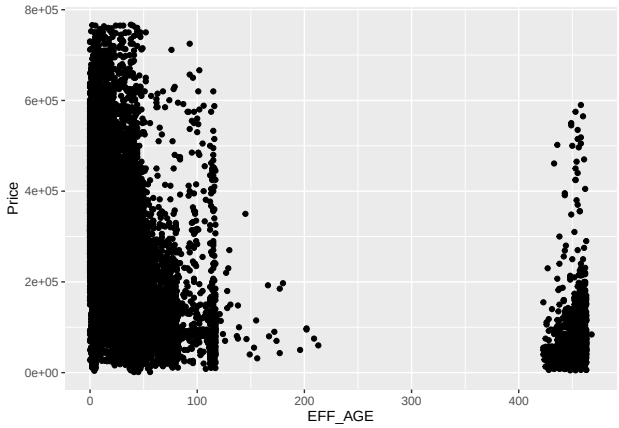
Now, we will proceed to the numerical variables.

```
lapply(var.num, function(x) ggplot(sales, aes_string(x=x, y="Price")) + geom_point())
```

```
## [[1]]

##
## [[2]]

##
## [[3]]

##
## [[4]]

##
## [[5]]

##
## [[6]]
```

```
##
## [[7]]
```





From the plots, we can conclude the same conclusion as before based on the correlation coefficients, such as the relationship between `SQFT` and `Price` which shows almost no correlation between the two. The same can be concluded from the plots for variables `RMTOT`, `RMBED`, `BATH`, and `EFF_AGE`. In contrast, it can be observed that there exist a positive correlation between `SFLA` and `Price` and `FLR1AREA` and `Price`.

---

4 Train Test Splitting

We are now ready to split the data set into train and test set. The splitting will be done using `CreateDataPartition` function from the `caret` package. The train-test split ratio will be 80:20 and stratified splitting will be done based on `GRADE`. This variable selection is due to the strong correlation that exists between a house's grade and its price as previously discussed. Additionally, the train-test set must also include some cities with a few entries.

```
train.idx <- createDataPartition(y = sales$GRADE, p = 0.8, list = FALSE)
train <- sales[train.idx, ]
test <- sales[-train.idx, ]

# Checking the proportion of the GRADE categorical variable
# in the train and test set in comparison to the original data set
rbind("Data Set" = table(sales$GRADE),
      "Train" = table(train$GRADE),
      "Test" = table(test$GRADE))
```

```
##           A    B     C    D   E
## Data Set 44 2474 22832 3041 805
## Train    36 1980 18266 2433 644
## Test      8  494  4566  608 161
```

```
rbind("Data Set" = prop.table(table(sales$GRADE)),
      "Train" = prop.table(table(train$GRADE)),
      "Test" = prop.table(table(test$GRADE)))
```

```
##                    A          B         C         D          E
## Data Set 0.001507056 0.08473764 0.7820249 0.1041581 0.02757227
## Train    0.001541162 0.08476390 0.7819684 0.1041569 0.02756967
## Test     0.001370567 0.08463252 0.7822512 0.1041631 0.02758266
```

We can see that all categories in `GRADE` exist in both train and test set with a similar proportion to the original data. we will now check the proportions of the cities in the train and test set.

```
rbind("Data Set" = table(sales$City),
      "Train" = table(train$City),
      "Test" = table(test$City))
```

```
##           20 21  23 24 25  27  28 30 31 32   33  43  44 45  47  48 49    54  59
## Data Set 720 28 200 57 67 350 121 81 13 54 2844 843 366 50 534 249 29 13694 709
## Train    580 23 172 42 54 279  98 65 10 44 2220 672 307 43 435 197 23 10949 560
## Test     140  5  28 15 13  71  23 16  3 10  624 171  59  7  99  52  6  2745 149
##           60  62  64 65  66  67   69 70  71  72  74  75  76  77 78  80  81  82
## Data Set 441 105 296 88 188 182 1637 97 281 188 380 786 335 124 45 522 392 113
## Train    355  85 235 78 155 148 1313 84 230 152 309 629 277  90 36 430 307  95
## Test      86  20  61 10  33  34  324 13  51  36  71 157  58  34  9  92  85  18
##           83 84  85  87 88  89 91  92 93  95  96  97  98
## Data Set 145 63 117 171 43 300 19 128 71 177 186 333 234
## Train    110 48  90 127 38 244 13 104 54 147 143 274 186
## Test      35 15  27  44  5  56  6  24 17  30  43  59  48
```

```
rbind("Data Set" = prop.table(table(sales$City)),
      "Train" = prop.table(table(train$City)),
      "Test" = prop.table(table(test$City)))
```

```
##                  20           21          23          24          25          27
## Data Set 0.02466091 0.0009590355 0.006850253 0.001952322 0.002294835 0.01198794
## Train    0.02482983 0.0009846312 0.007363329 0.001798022 0.002311743 0.01194400
## Test     0.02398492 0.0008566044 0.004796985 0.002569813 0.002227171 0.01216378
##                  28          30           31          32          33          43
## Data Set 0.004144403 0.002774353 0.0004452665 0.001849568 0.09741060 0.02887382
## Train    0.004195385 0.002782653 0.0004281005 0.001883642 0.09503831 0.02876835
## Test     0.003940380 0.002741134 0.0005139627 0.001713209 0.10690423 0.02929587
##                  44          45         47          48           49        54
## Data Set 0.01253596 0.001712563 0.01829018 0.008528566 0.0009932868 0.4690369
## Train    0.01314269 0.001840832 0.01862237 0.008433580 0.0009846312 0.4687273
## Test     0.01010793 0.001199246 0.01696077 0.008908686 0.0010279253 0.4702758
##                  59          60          62          64          65          66
## Data Set 0.02428415 0.01510481 0.003596383 0.01013838 0.003014112 0.006439238
## Train    0.02397363 0.01519757 0.003638854 0.01006036 0.003339184 0.006635558
## Test     0.02552681 0.01473360 0.003426418 0.01045057 0.001713209 0.005653589
##                  67          69          70          71          72          74
## Data Set 0.006233731 0.05606932 0.003322373 0.009624606 0.006439238 0.01301548
## Train    0.006335888 0.05620960 0.003596044 0.009846312 0.006507128 0.01322831
## Test     0.005824910 0.05550797 0.002227171 0.008737365 0.006167552 0.01216378
##                  75          76          77          78          80          81
## Data Set 0.02692150 0.011474175 0.004247157 0.001541307 0.01787916 0.01342650
## Train    0.02692752 0.011858384 0.003852905 0.001541162 0.01840832 0.01314269
## Test     0.02689738 0.009936611 0.005824910 0.001541888 0.01576152 0.01456228
##                  82          83          84          85          87
## Data Set 0.003870393 0.004966434 0.002157830 0.004007398 0.005856967
## Train    0.004066955 0.004709106 0.002054882 0.003852905 0.005436877
## Test     0.003083776 0.005996231 0.002569813 0.004625664 0.007538119
##                  88          89          91          92          93
```

```
## Data Set 0.0014728045 0.01027538 0.0006507741 0.004384162 0.002431840
## Train    0.0016267820 0.01044565 0.0005565307 0.004452245 0.002311743
## Test     0.0008566044 0.00959397 0.0010279253 0.004111701 0.002912455
##                   95          96         97          98
## Data Set 0.006062474 0.006370736 0.01140567 0.008014797
## Train    0.006293078 0.006121837 0.01172995 0.007962670
## Test     0.005139627 0.007366798 0.01010793 0.008223402
```

It can be observed that the proportions of the cities in both train and test set are close to the proportions in the original data set. Even though there are cities such as cities 21, 31, 49, and 91 that have only a few data points in the test set, we would ignore them as we already have sufficient amount of data with (more than 10 data) to fit in the train set.

---

5 Generalized Additive Model (GAM)

Generalized Additive Model (GAM) is a model that assumes that the explanatory variable may not have a linear relationship with the response variable. For each independent variable, GAM will create a spline instead of only weighting them linearly in Generalized Linear Model (GLM). GAM creates partitions for a certain variable and within each partition, GAM fits a function for that suits the respective variable and continues doing so in the next partition, in which the functions combined as a result is called a spline. GAM does the process of fitting splines automatically. However, splines will not be created for categorical variables.

In the `gam` library in R, we can determine a variable to be fitted with a spline by using the `s()` function during the modelling.

5.1 Modelling Preparation

As previously mentioned, we will assume that all numerical variables do not have a linear relationship with the price and add the `s()` for the modelling.

```
# Preparing outcome and independent variables for the models
# We also transform the outcome with logarithm transformation.
outcome <- "log(Price)"
s.var.num <- paste("s(", var.num, ")", sep = "")
```

We will also choose between `City` or `Province` as the independent variable in our model. We will analyze it based on the aic of the model using the respective variables.

```
f.temp1 <- as.formula(paste(outcome,
                            paste(c(s.var.num, var.categ[!var.categ %in% "Province"]), collapse = "+"),
                            sep = "~"))
f.temp2 <- as.formula(paste(outcome,
                            paste(c(s.var.num, var.categ[!var.categ %in% "City"]),collapse = "+"),
                            sep = "~"))

c("City AIC" = gam(f.temp1, data = train)$aic, "Province AIC" = gam(f.temp2, data = train)$aic)
```

```
##     City AIC Province AIC
##     14221.74     18129.57
```

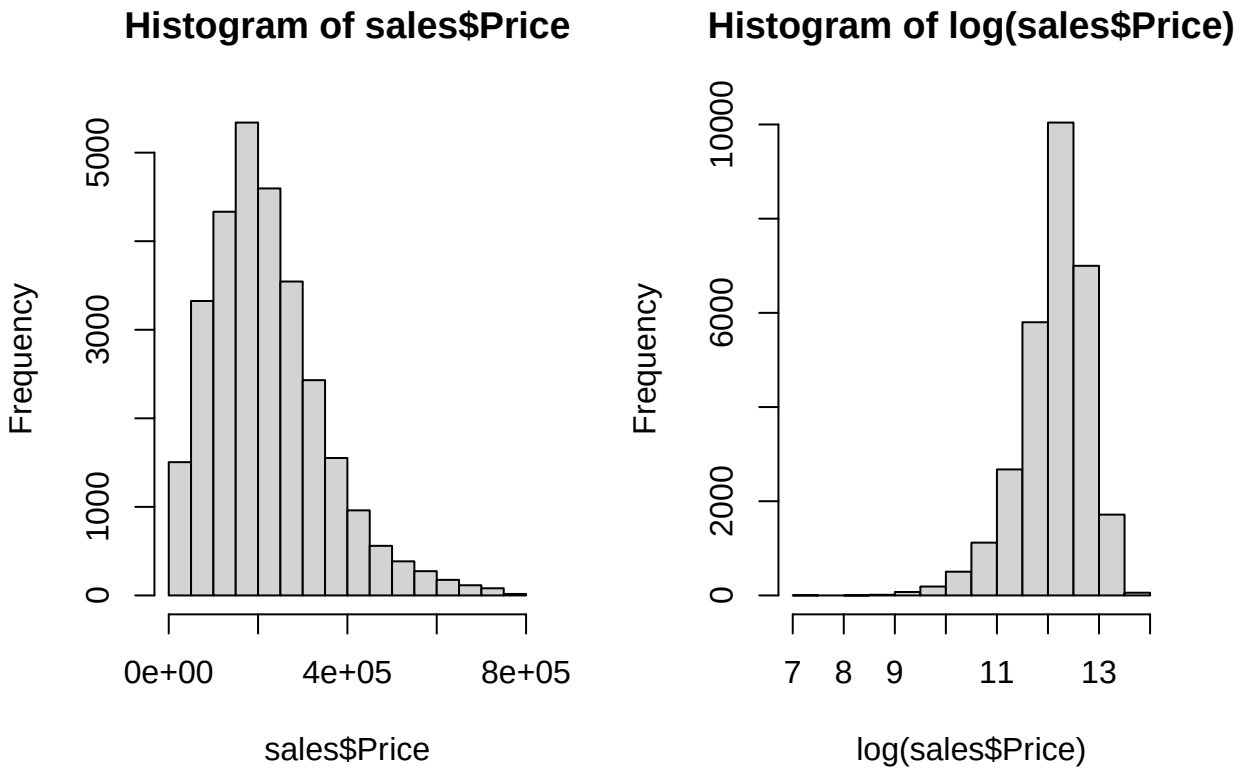
From the results above, the AICs indicate a similar result to the EDA. Model with the `City` variable has a higher AIC in comparison to the `Province`. It can be inferred that there are more residues in the model with province variable which in turn affects the fitted values which increases the AIC value.

Therefore, the models will be created with the `City` variable and exclude the `Province` variable.

```
var.categ <- var.categ[!var.categ %in% "Province"]
(f <- as.formula(paste(outcome,
                       paste(c(s.var.num, var.categ), collapse = "+"),
                       sep = "~")))
```

```
## log(Price) ~ s(RMTOT) + s(RMBED) + s(BATH) + s(SQFT) + s(FLR1AREA) +
##     s(SFLA) + s(EFF_AGE) + City + BSMT + ATTIC + GRADE + Style
```

```
par(mfrow=c(1,2))
hist(sales$Price)
hist(log(sales$Price))
```



Additionally, by transforming the response variable with the logarithm function, we can see that the range of data values become smaller and the distribution of the data becomes less skewed, although it does not completely resemble a normal distribution. This transformation allows a better outcome during model fitting.

5.2 Model Evaluation Functions

To analyze and evaluate our models, we will use the following functions.

```r
options(scipen=999) # Disable Scientific Notation

# Plots relationships between Log Sale Price, Prediction, and Residuals on the train set
# Function also returns a bin which contains indices of extreme residual data
eval.train_init <- function(model, train, outliers){
  zresid = data.frame(x=rstandard(model))
  title = ifelse(outliers == TRUE, "Outliers not Removed", "Outliers Removed")

  print(ggarrange(
    ggplot() + geom_point(aes(x=model$fitted.values, y=log(train$Price))) +
      geom_abline(aes(intercept = 0, slope = 1), colour = "blue") +
      ggtitle(paste("Log SalePrice vs Prediction - Training Set,", title)) +
      theme(plot.title = element_text(hjust = 0.5)) +
      labs(x = "Prediction on Train Data", y ="Log Sales Price"),
    ggplot() + geom_point(aes(x=model$fitted.values, y=zresid$x)) +
      geom_abline(aes(intercept = 0, slope = 0), colour = "blue") +
      ggtitle(paste("Residual vs Prediction - Training Set,", title)) +
      theme(plot.title = element_text(hjust = 0.5)) +
      labs(x = "Prediction on Train Data", y ="Residual"),
    nrow = 2, align = "v"))

  if(outliers == TRUE){
    bin = which(abs(zresid)>3)
    return(bin)
  }
}

# Updates train data by removing residuals in the model.
eval.train_update <- function(model, train, bin){
  if(length(bin)>0) {
    train.outliers = train
    train.outliers$outliers = 0
    train.outliers$outliers[bin] = 1
    train.outliers$pred = model$fitted.values
    train.outliers$pred.dollar = exp(train.outliers$pred)
    train.2 = train[-bin,]
  } else {
    train.2 = train
  }

  return(train.2)
}

# Compare predicted and observed data
# Function returns a list of metrics used for comparison
comp <- function(pred, obs){
  n = length(obs)
  rsq = cor(pred,obs)^2
  mse = sum((pred - obs)^2)/n
  semse = sd((pred - obs)^2) / sqrt(n)
  rmse = sqrt(mse)
  se = sd(pred-obs) / sqrt(n)
  mae = sum(abs(pred-obs))/n
```

```
  mape = sum(abs(pred-obs)/obs)/n*100
  return(list("n"=n,"R2"=rsq,"MSE"=mse,"SEMSE"=semse,"RMSE"=rmse,"SE"=se,"MAE"=mae,"MAPE"=mape))
}

# Plots relationships between Log Sale Price and Prediction, and Residuals in the test set
# Function also returns the comp function for predictions in the test set
# (with respect to normal Price, not log(Price))
eval.test <- function(pred, obs){
  # Residual for the test set (must be scaled)
  zresid.test = scale(pred - log(obs))

  print(ggarrange(
    ggplot() + geom_point(aes(x=pred, y=log(obs))) +
        geom_abline(aes(intercept = 0, slope = 1), colour = "blue") +
        ggtitle("Log SalePrice vs Prediction - Testing Set") +
        theme(plot.title = element_text(hjust = 0.5)) +
        labs(x = "Prediction on Test Data", y ="Log Sales Price"),
    ggplot() + geom_point(aes(x=pred, y=zresid.test)) +
      geom_abline(aes(intercept = 0, slope = 0), colour = "blue") +
      ggtitle("Residual vs Prediction - Testing Set") +
      theme(plot.title = element_text(hjust = 0.5)) +
      labs(x = "Prediction on Test Data", y ="Residual"),
    nrow = 2, align = "v"))

  comp.test = comp(exp(pred), obs)
  print(comp.test)

  return(comp.test)
}


# Saves plot into pdf
plot2pdf <- function(plots, filename){
  pdf = paste("D:/Downloads/", filename, ".pdf", sep = "")
  pdf(file = pdf)
  for (i in 1:length(plots)){
    plot(plots[[i]])
  }
  dev.off
  paste("Saved to ", toString(pdf), sep = "")
}
```

---

6 GAM Model 1

```
mod.gam <- gam(f, data = train)
summary(mod.gam)
```

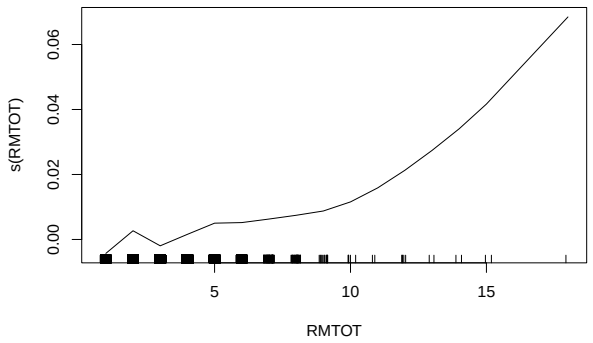
```
##
```

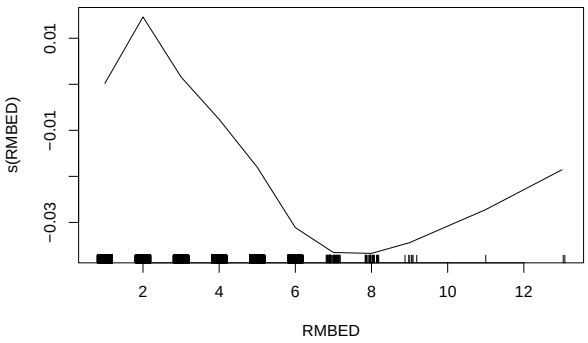
```
## Call: gam(formula = f, data = train)
## Deviance Residuals:
##        Min         1Q     Median         3Q        Max
## -4.9507246 -0.1437972 -0.0004321  0.1611073  2.0538972
##
## (Dispersion Parameter for gaussian family taken to be 0.1072)
##
##     Null Deviance: 10651.77 on 23358 degrees of freedom
## Residual Deviance: 2493.852 on 23265.27 degrees of freedom
## AIC: 14221.74
##
## Number of Local Scoring Iterations: NA
##
## Anova for Parametric Effects
##                Df  Sum Sq Mean Sq    F value                Pr(>F)
## s(RMTOT)        1  566.10  566.10  5281.1361 < 0.00000000000000022 ***
## s(RMBED)        1   47.01   47.01   438.5391 < 0.00000000000000022 ***
## s(BATH)         1 1823.66 1823.66 17013.0418 < 0.00000000000000022 ***
## s(SQFT)         1    0.84    0.84     7.8095              0.005202 **
## s(FLR1AREA)     1  356.37  356.37  3324.6157 < 0.00000000000000022 ***
## s(SFLA)         1 1145.58 1145.58 10687.1888 < 0.00000000000000022 ***
## s(EFF_AGE)      1 1504.33 1504.33 14033.9538 < 0.00000000000000022 ***
## City           49 1090.30   22.25   207.5802 < 0.00000000000000022 ***
## BSMT            6   66.51   11.09   103.4176 < 0.00000000000000022 ***
## ATTIC           1   36.72   36.72   342.6010 < 0.00000000000000022 ***
## GRADE           4  279.35   69.84   651.5291 < 0.00000000000000022 ***
## Style           4   81.79   20.45   190.7554 < 0.00000000000000022 ***
## Residuals   23265 2493.85    0.11
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Anova for Nonparametric Effects
##             Npar Df Npar F                 Pr(F)
## (Intercept)
## s(RMTOT)        3.0   1.57                0.1939
## s(RMBED)        3.0   2.99                0.0298 *
## s(BATH)         3.0   1.81                0.1432
## s(SQFT)         3.7  46.88 < 0.00000000000000022 ***
## s(FLR1AREA)     3.0  12.42         0.00000004091 ***
## s(SFLA)         3.0 291.21 < 0.00000000000000022 ***
## s(EFF_AGE)      3.0 533.33 < 0.00000000000000022 ***
## City
## BSMT
## ATTIC
## GRADE
## Style
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

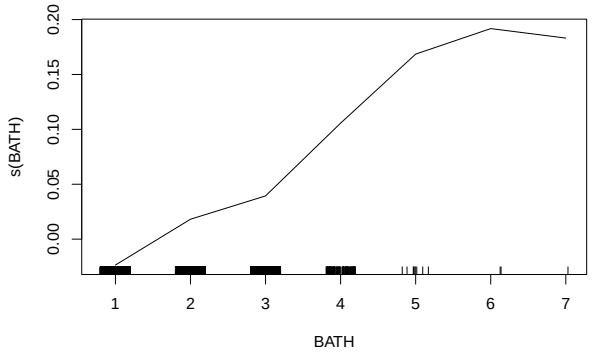
```
# Saving spline plots to pdf
plot2pdf(preplot(mod.gam), "GAM_InteractionPlots_1")
```

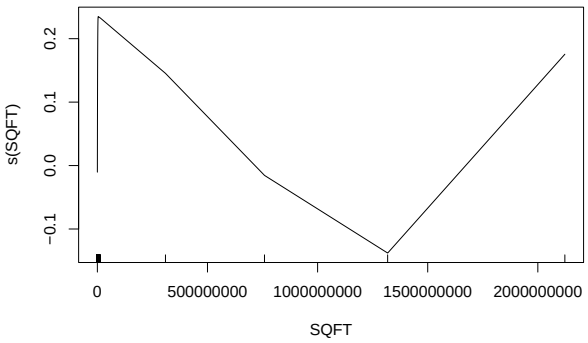
```
## [1] "Saved to D:/Downloads/GAM_InteractionPlots_1.pdf"
```

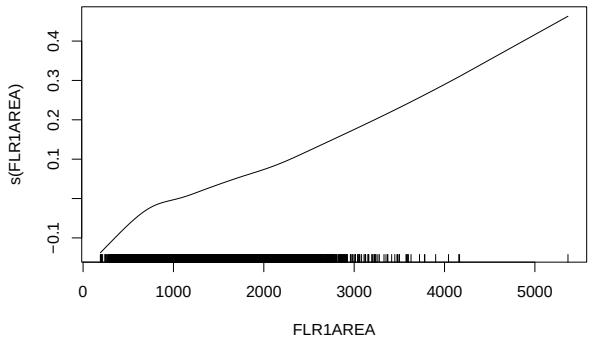
```
plot(mod.gam)
```

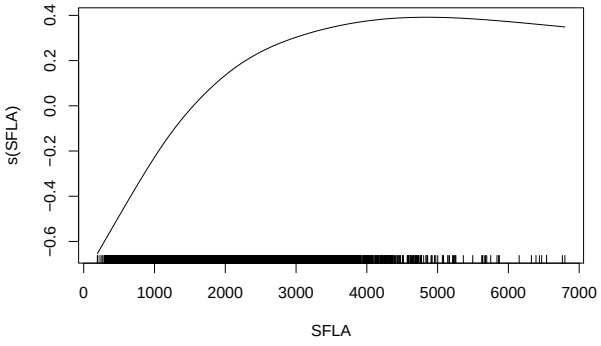

s(RMTOT)
0.00
0.02
0.04
0.06
5
10
15
RMTOT

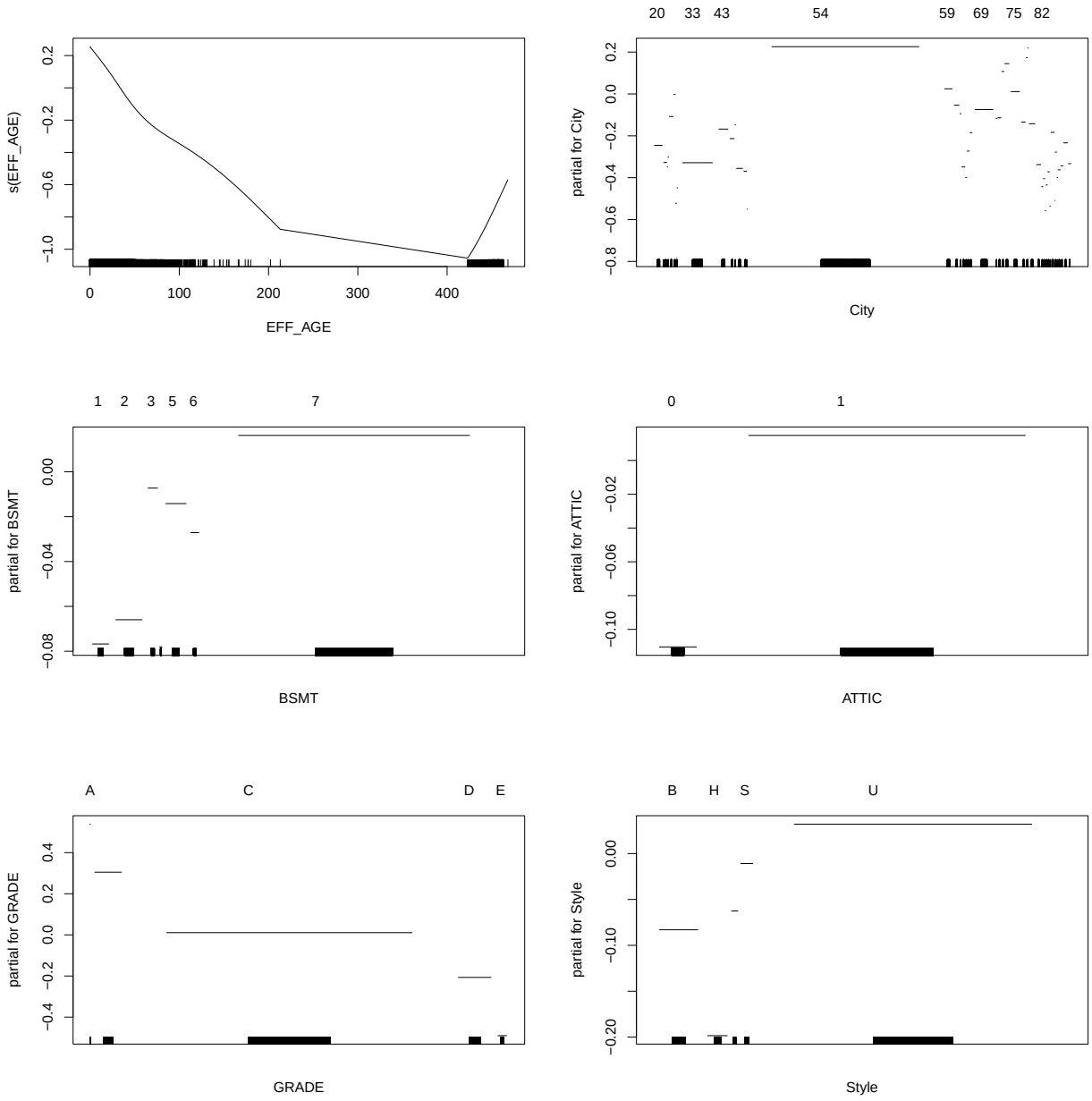

s(RMBED)
−0.03
−0.01
0.01
2
4
6
8
10
12
RMBED


s(BATH)
0.00
0.05
0.10
0.15
0.20
1
2
3
4
5
6
7
BATH


s(SQFT)
−0.1
0.0
0.1
0.2
0
500000000
1000000000
1500000000
2000000000
SQFT


s(FLR1AREA)
−0.1
0.1
0.2
0.3
0.4
0
1000
2000
3000
4000
5000
FLR1AREA


s(SFLA)
−0.6
−0.4
−0.2
0.0
0.2
0.4
0
1000
2000
3000
4000
5000
6000
7000
SFLA



From the GAM interaction plots above, it can be observed that there are some variables that are not linear as the splines do not form a straight line. One of the most distinct plots is the SQFT spline. As observed above, the `SQFT` spline drastically rises at first, then falls, and rises again. This shows that the `SQFT` is not linear. Other results highlight the same conclusion, such as `RMTOT`, `RMBED`, `FLR1AREA`, and `EFF_AGE`. Meanwhile, spline for `SFLA` resembles a quadratic function, while spline for `FLR1AREA` shows an almost linear function.

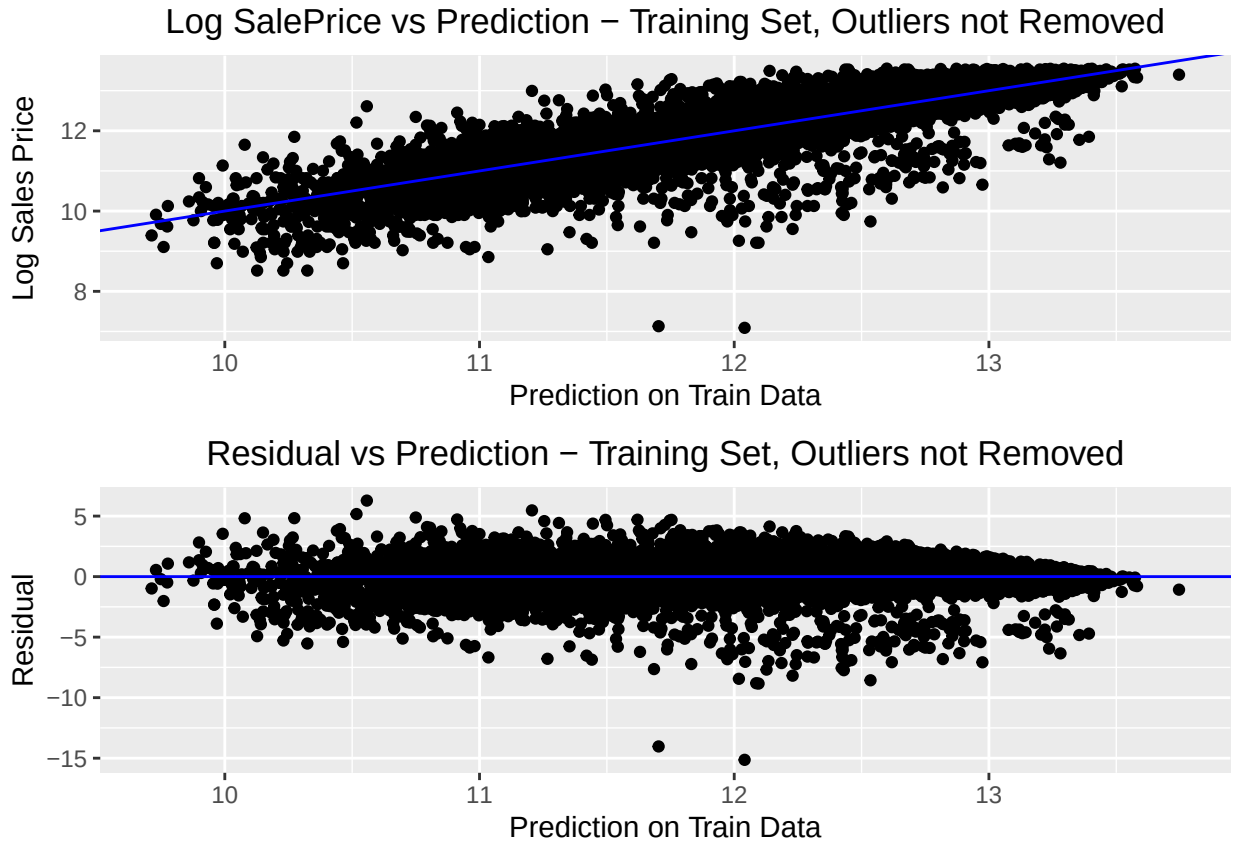
Additionally, conclusions cannot be inferred from the categorical plots as splines were not defined for those variables.

6.1 Model Evaluation 1 - AIC & Residual Analysis

```
mod.gam$aic
```

```
## [1] 14221.74
```

```
bin.gam = eval.train_init(mod.gam, train, outliers = TRUE)
```




```
train.gam = eval.train_update(mod.gam, train, bin.gam)
```

In the model's summary, that the model received an AIC value of 14221.74, which will be used to compare with the next model with removed outliers in the train set.

From the graph above, we can see from the first graph that the predictions made by the model are spread around the y=x line, which shows that the predictions are close to the actual values. However, we can see that there are a couple of wrong predictions. The second graph highlights these wrong outcomes as we can see that there are some residuals that are far from the predictions. Residuals are the difference between each predicted data and the actual value of the data. Data considered as residuals were evaluated by the `rstandard()` function and values that are greater than 3 will be removed in the following step.

6.2 Remodelling With New Train Set

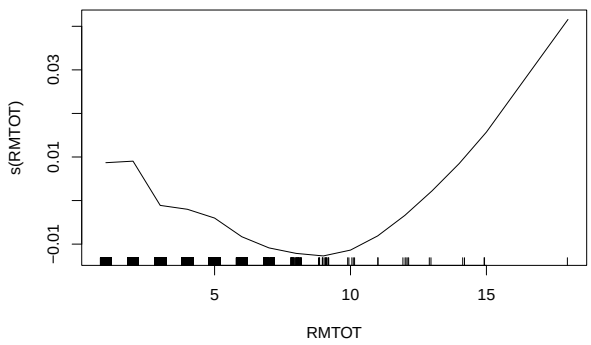
```r
mod.gam.final <- gam(f, data = train.gam)
summary(mod.gam.final)
```

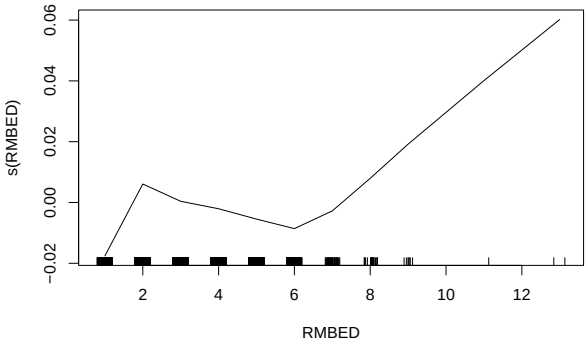
```
##
## Call: gam(formula = f, data = train.gam)
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -1.14756 -0.14597 -0.01293  0.13822  1.01783
##
## (Dispersion Parameter for gaussian family taken to be 0.0677)
##
##     Null Deviance: 9376.269 on 22950 degrees of freedom
## Residual Deviance: 1547.587 on 22857.28 degrees of freedom
## AIC: 3430.407
##
## Number of Local Scoring Iterations: NA
##
## Anova for Parametric Effects
##                Df  Sum Sq Mean Sq   F value                Pr(>F)
## s(RMTOT)        1  538.98  538.98  7960.604 < 0.00000000000000022 ***
## s(RMBED)        1   47.02   47.02   694.527 < 0.00000000000000022 ***
## s(BATH)         1 1779.68 1779.68 26285.168 < 0.00000000000000022 ***
## s(SQFT)         1    0.94    0.94    13.902             0.0001931 ***
## s(FLR1AREA)     1  375.82  375.82  5550.766 < 0.00000000000000022 ***
## s(SFLA)         1 1066.75 1066.75 15755.522 < 0.00000000000000022 ***
## s(EFF_AGE)      1 1455.78 1455.78 21501.304 < 0.00000000000000022 ***
## City           49  990.72   20.22   298.623 < 0.00000000000000022 ***
## BSMT            6   48.19    8.03   118.614 < 0.00000000000000022 ***
## ATTIC           1    4.83    4.83    71.278 < 0.00000000000000022 ***
## GRADE           4  268.75   67.19   992.330 < 0.00000000000000022 ***
## Style           4   83.42   20.85   308.004 < 0.00000000000000022 ***
## Residuals   22857 1547.59    0.07
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Anova for Nonparametric Effects
##             Npar Df Npar F                 Pr(F)
## (Intercept)
## s(RMTOT)        3.0   2.66              0.046218 *
## s(RMBED)        3.0   5.23              0.001319 **
## s(BATH)         3.0   9.61           0.000002453 ***
## s(SQFT)         3.7  55.32 < 0.00000000000000022 ***
## s(FLR1AREA)     3.0   9.02           0.000005794 ***
## s(SFLA)         3.0 369.73 < 0.00000000000000022 ***
## s(EFF_AGE)      3.0 821.81 < 0.00000000000000022 ***
## City
## BSMT
## ATTIC
## GRADE
## Style
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

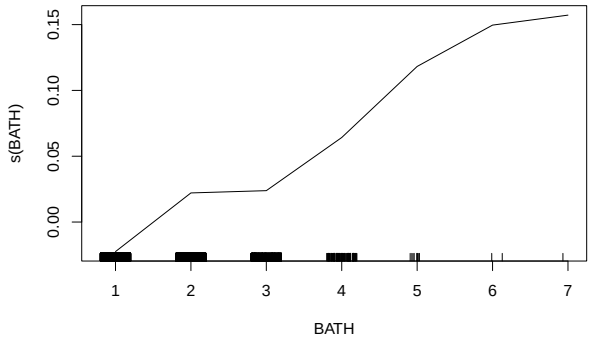
```
# Saving spline plots to pdf
plot2pdf(preplot(mod.gam.final), "GAM_InteractionPlots_1_final")
```

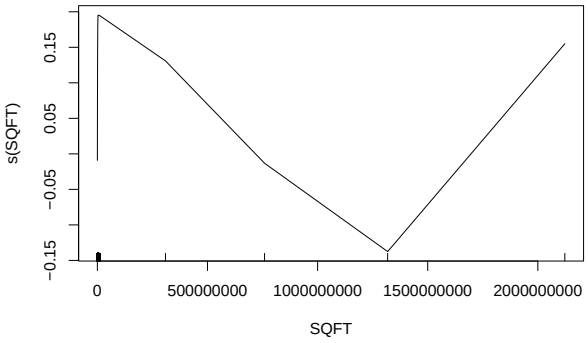
```
## [1] "Saved to D:/Downloads/GAM_InteractionPlots_1_final.pdf"
```

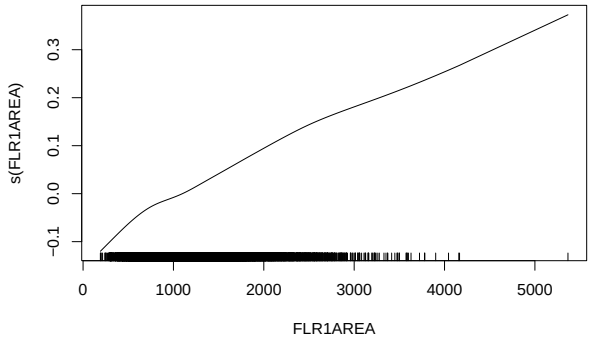
```
plot(mod.gam.final)
```

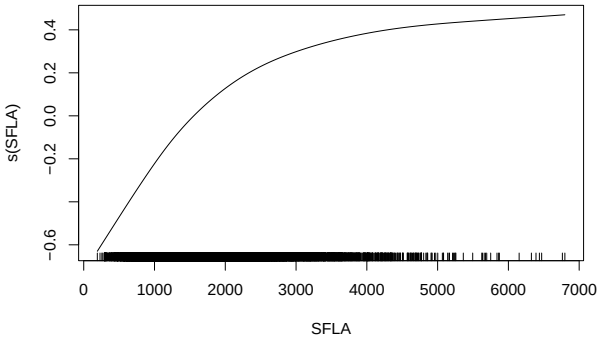


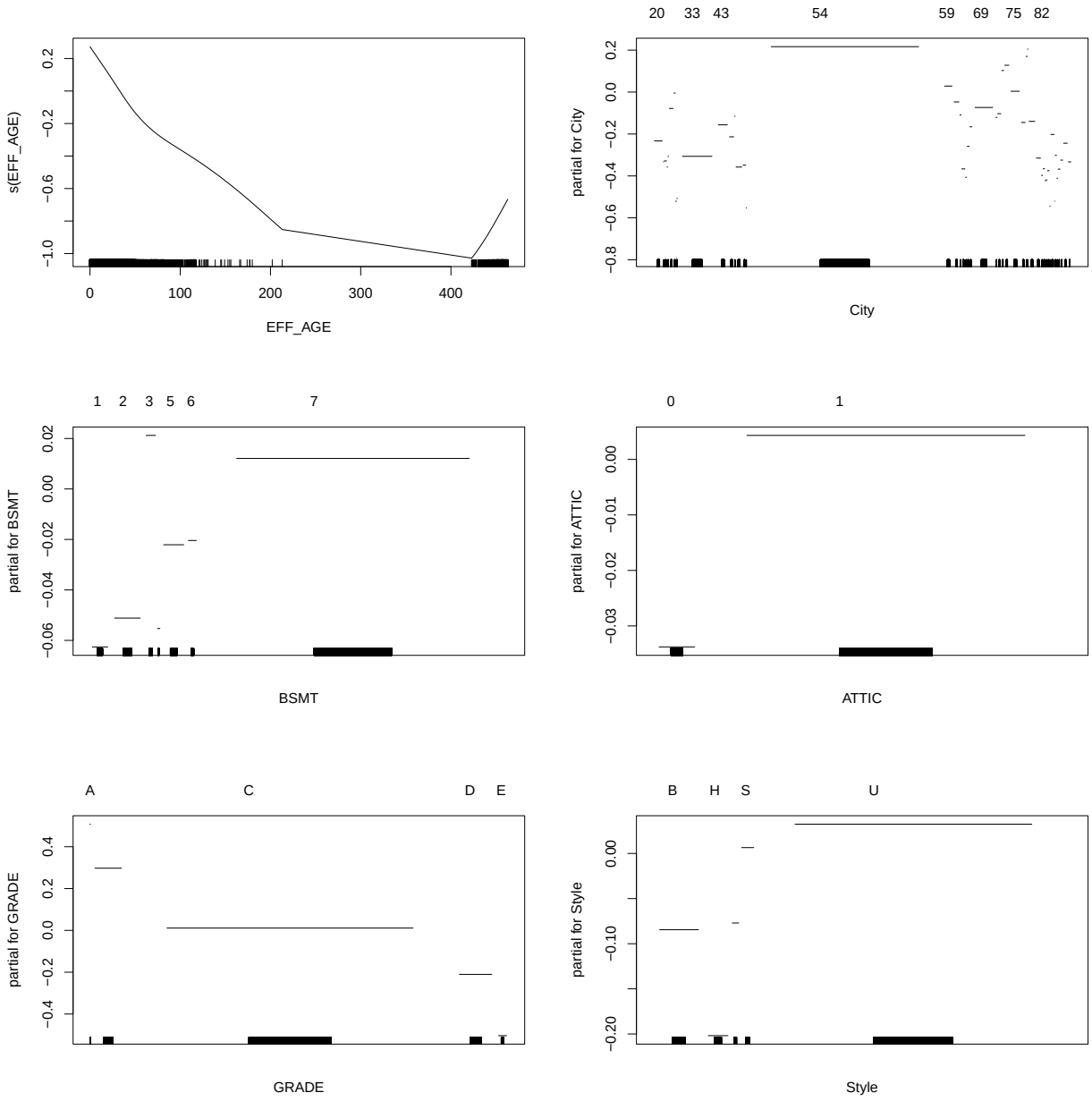












From the splines plot, we can see that there some changes in the spline as a result of fitting with the outliers removed. Two most prominent changes can be seen in the `RMTOT` plot with a more wavy line and `RMBED` which was initially steeply decreasing in the previous splines plot. However, almost all other the plots are consistent with the previous results, such as `FLR1AREA` resembling a linear function.

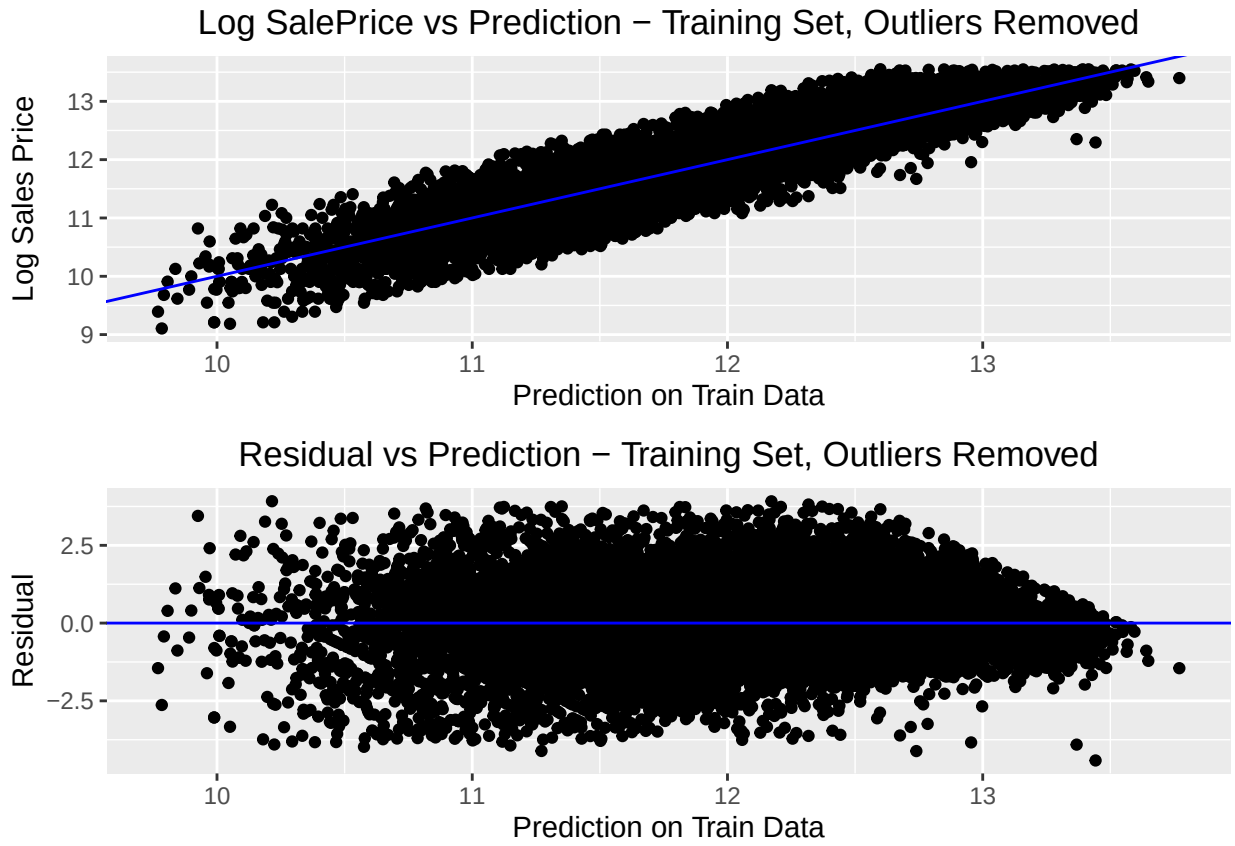
6.3 Model Evaluation 2 - New AIC, Residual Analysis, and Error Analysis

```
data.frame("Outliers Not Removed" = mod.gam$aic,
           "Outliers Removed" = mod.gam.final$aic)
```

```
##   Outliers.Not.Removed Outliers.Removed
```

```
## 1             14221.74         3430.407
```

```
eval.train_init(mod.gam.final, train.gam, outliers = FALSE)
```



After we have removed the outliers in the training set, we can see that there is a highly significant decrease in the AIC of the model. This means that the model is now performing much better with outliers removed.

From the graphs, we can now see that the predictions are now closer the actual values and the residuals are also closer to 0. It is also worth noting that there are no significant patterns present in the residuals vs prediction plot; the residuals are randomly distributed.

Now, we are going to analyze the error in the predictions of the train set. We will also compare the metrics with the previous model with outliers not removed.

```
merge(stack(comp(exp(mod.gam$fitted.values), exp(mod.gam$y))),
      stack(comp(exp(mod.gam.final$fitted.values), exp(mod.gam.final$y))),
      by = "ind", sort = FALSE)
```

```
##     ind           values.x           values.y
## 1     n      23359.0000000      22951.0000000
## 2    R2          0.7426658          0.7887379
## 3   MSE 4251570335.0735035 3403647885.6924467
## 4 SEMSE   88623582.3273855   66266751.8694365
## 5  RMSE      65204.0668599      58340.7909245
## 6    SE        423.7051350        383.6286106
## 7   MAE      42362.6079211      39037.4097674
## 8  MAPE         25.7697061         19.6394097
```

From the results, we can conclude that out of the 22951 data in the train set with outliers removed, the model receives an RMSE of 58340.79 and an average absolute error of 19.64%. These values show an increase in performance from the previous model with outliers intact. This can be concluded from the lower RMSE and MAPE values.

6.4 Multicollinearity

```
vif(mod.gam.final)
```

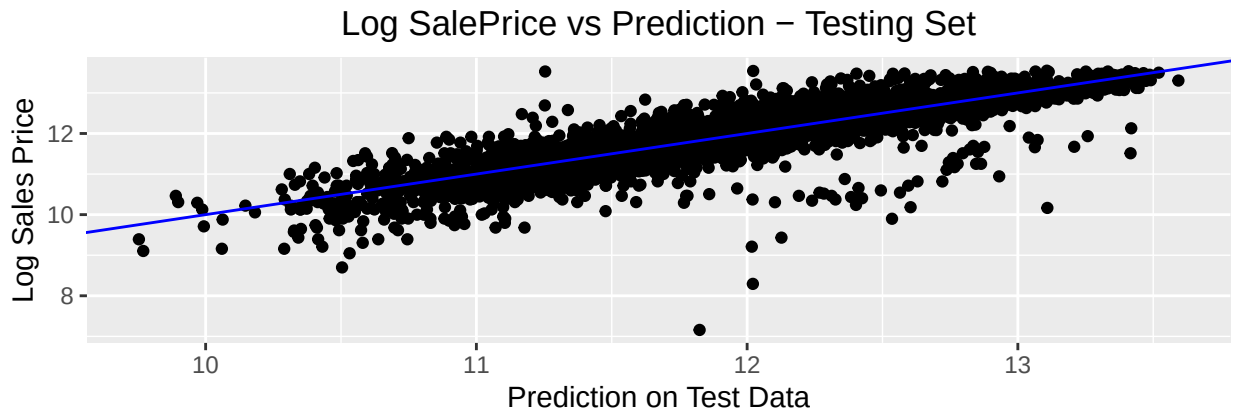
```
##                 GVIF Df GVIF^(1/(2*Df))
## s(RMTOT)    8.033436  1        2.834332
## s(RMBED)    8.343347  1        2.888485
## s(BATH)     2.060585  1        1.435474
## s(SQFT)     1.004034  1        1.002015
## s(FLR1AREA) 1.883395  1        1.372369
## s(SFLA)     3.177493  1        1.782552
## s(EFF_AGE)  1.234876  1        1.111250
## City        1.834711 49        1.006212
## BSMT        1.858569  6        1.053008
## ATTIC       1.221587  1        1.105254
## GRADE       2.070554  4        1.095244
## Style       1.479317  4        1.050165
```

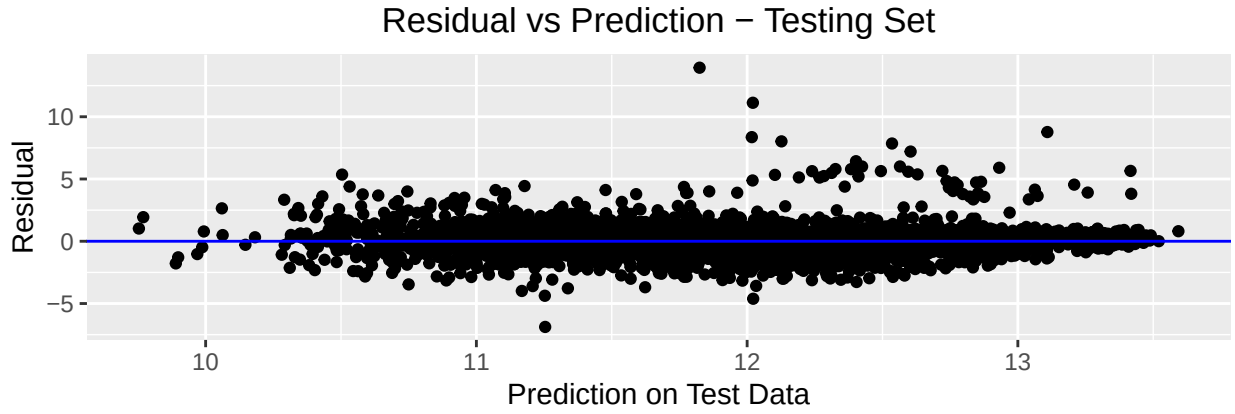
From the VIF scores, we can see that all numerical variables have a relatively small VIF with values below 4. Therefore, we can conclude that each independent variable is not a linear combination of other independent variables. On the other hand, we will ignore the vif values of the categorical and the numerical/categorical variables.

6.5 Prediction and Error Analysis

```
test.gam = test
test.gam$prediction = predict(mod.gam.final, newdata = test.gam, type = "response")
```

```
error.gam = eval.test(test.gam$prediction, test.gam$Price)
```






```
## $n
## [1] 5837
##
## $R2
## [1] 0.7406387
##
## $MSE
## [1] 4362587755
##
## $SEMSE
## [1] 212655500
##
## $RMSE
## [1] 66049.89
##
## $SE
## [1] 863.7374
##
## $MAE
## [1] 41619.89
##
## $MAPE
## [1] 27.6207
```

From the graph, we can see that there are still a lot of predictions that are far away from the y=x line, which

means that the model has some inaccurate predictions in the test set. The second graph further highlights these findings.

From the results, we can conclude that out of the 5837 data in the test set, the model successfully predicted the prices with RMSE of 66049.89 and average absolute error of 27.62%. These metrics and the model's AIC will be kept for further comparison with the results from other models.

---

7 GAM Model 2

From the previous model, we concluded that there is still a number of predictions that are far away from the real values. We will now attempt to obtain a better performance by changing the variables used in the modelling.

First, since `SQFT` has low correlation towards `Price` and an unusual non-linearity, we will exclude `SQFT`. Next, we will also exclude the `RMTOT` variable as `RMBED` also explains the number of rooms, specifically bedrooms, which also has a higher correlation than `RMTOT`. Lastly, we will also attempt to not apply spline to `FLR1AREA` as the splines resemble a linear function.

```
var.num2 <- var.num[!var.num %in% c("SQFT", "RMTOT", "FLR1AREA")]
s.var.num2 <- paste("s(", var.num2, ")", sep = "")

(f2 <- as.formula(paste(outcome,
                        paste(c(s.var.num2, var.categ, "FLR1AREA"), collapse = "+"),
                        sep = "~")))
```

```
## log(Price) ~ s(RMBED) + s(BATH) + s(SFLA) + s(EFF_AGE) + City +
##     BSMT + ATTIC + GRADE + Style + FLR1AREA
```

```
mod.gam2 <- gam(f2, data = train)
summary(mod.gam2)
```

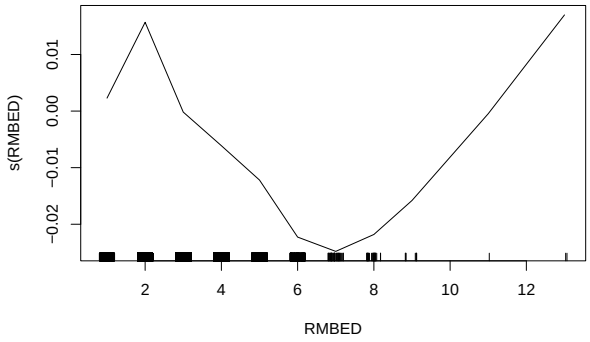
```
##
## Call: gam(formula = f2, data = train)
## Deviance Residuals:
##       Min        1Q    Median        3Q       Max
## -4.951262 -0.144338 -0.002106  0.163359  2.270503
##
## (Dispersion Parameter for gaussian family taken to be 0.1081)
##
##     Null Deviance: 10651.77 on 23358 degrees of freedom
## Residual Deviance: 2516.297 on 23277 degrees of freedom
## AIC: 14407.59
##
## Number of Local Scoring Iterations: NA
##
## Anova for Parametric Effects
##               Df  Sum Sq Mean Sq  F value                Pr(>F)
## s(RMBED)       1  605.23  605.23  5598.71 < 0.00000000000000022 ***
## s(BATH)        1 1799.59 1799.59 16647.14 < 0.00000000000000022 ***
## s(SFLA)        1 1503.55 1503.55 13908.62 < 0.00000000000000022 ***
```

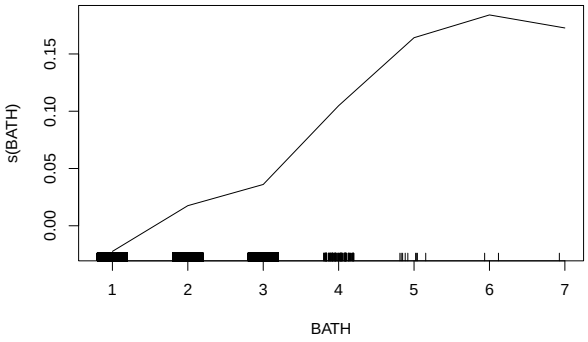
```
## s(EFF_AGE)     1 1486.89 1486.89 13754.44 < 0.00000000000000022 ***
## City          49 1013.94   20.69   191.42 < 0.00000000000000022 ***
## BSMT           6   60.52   10.09    93.31 < 0.00000000000000022 ***
## ATTIC          1   33.08   33.08   306.04 < 0.00000000000000022 ***
## GRADE          4  292.88   73.22   677.31 < 0.00000000000000022 ***
## Style          4  103.49   25.87   239.33 < 0.00000000000000022 ***
## FLR1AREA       1   18.89   18.89   174.74 < 0.00000000000000022 ***
## Residuals  23277 2516.30    0.11
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Anova for Nonparametric Effects
##             Npar Df Npar F                 Pr(F)
## (Intercept)
## s(RMBED)          3   3.90              0.008483 **
## s(BATH)           3   2.02              0.109007
## s(SFLA)           3 303.79 < 0.00000000000000022 ***
## s(EFF_AGE)        3 530.85 < 0.00000000000000022 ***
## City
## BSMT
## ATTIC
## GRADE
## Style
## FLR1AREA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

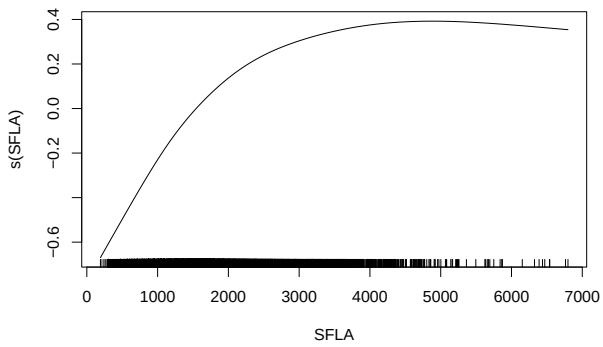
```
# Saving spline plots to pdf
plot2pdf(preplot(mod.gam2), "GAM_InteractionPlots_2")
```

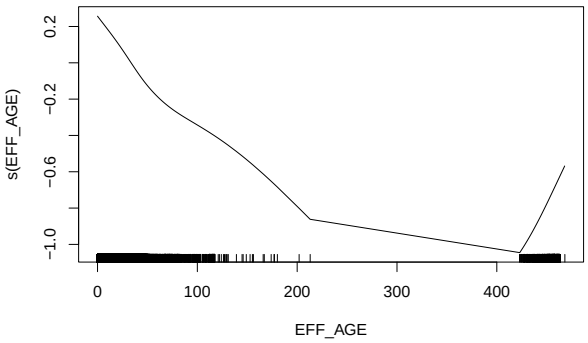
```
## [1] "Saved to D:/Downloads/GAM_InteractionPlots_2.pdf"
```

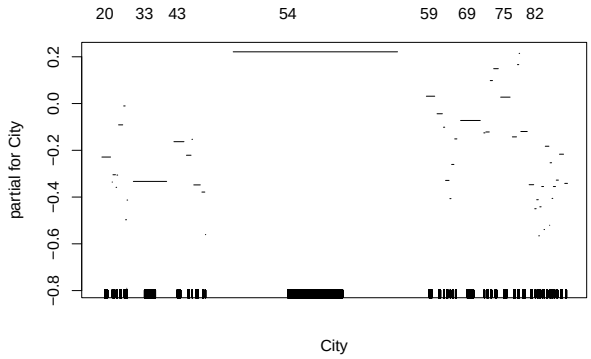
```
plot(mod.gam2)
```

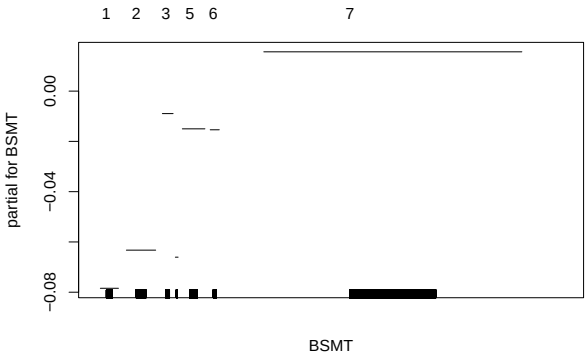


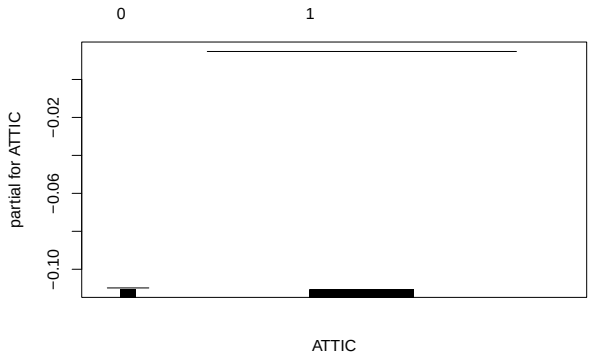


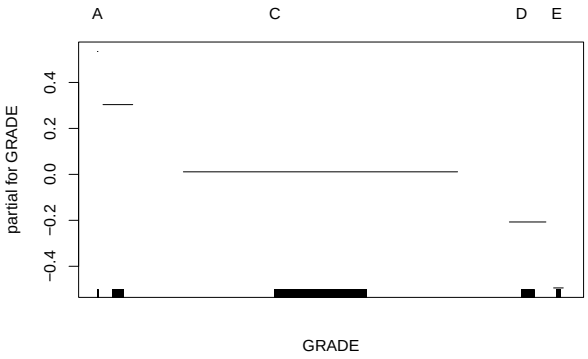

s(SFLA)
−0.6
−0.2
0.0
0.2
0.4
0
1000
2000
3000
4000
5000
6000
7000
SFLA

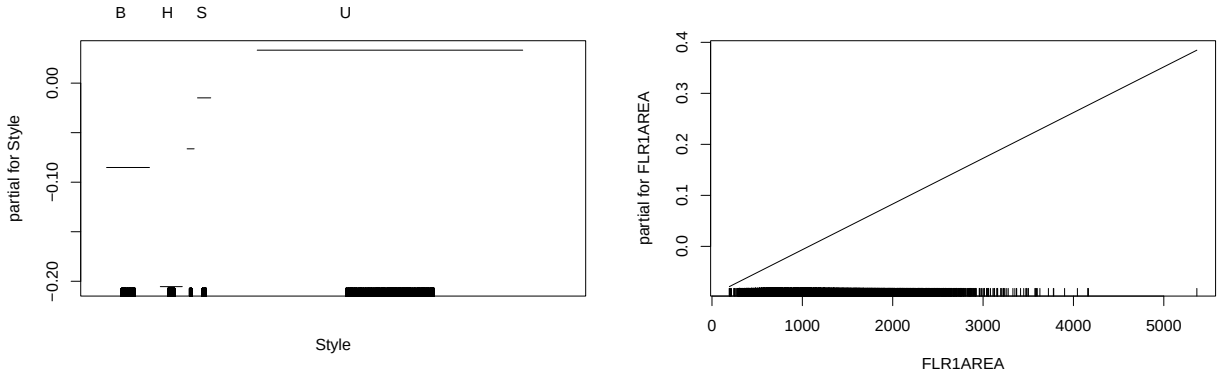

s(EFF_AGE)
−1.0
−0.6
−0.2
0.2
0
100
200
300
400
EFF_AGE


20
33
43
54
59
69
75
82
partial for City
−0.8
−0.6
−0.4
−0.2
0.0
0.2
City


1
2
3
5
6
7
partial for BSMT
−0.08
−0.04
0.00
BSMT


0
1
partial for ATTIC
−0.10
−0.06
−0.02
ATTIC


A
C
D
E
partial for GRADE
−0.4
−0.2
0.0
0.2
0.4
GRADE



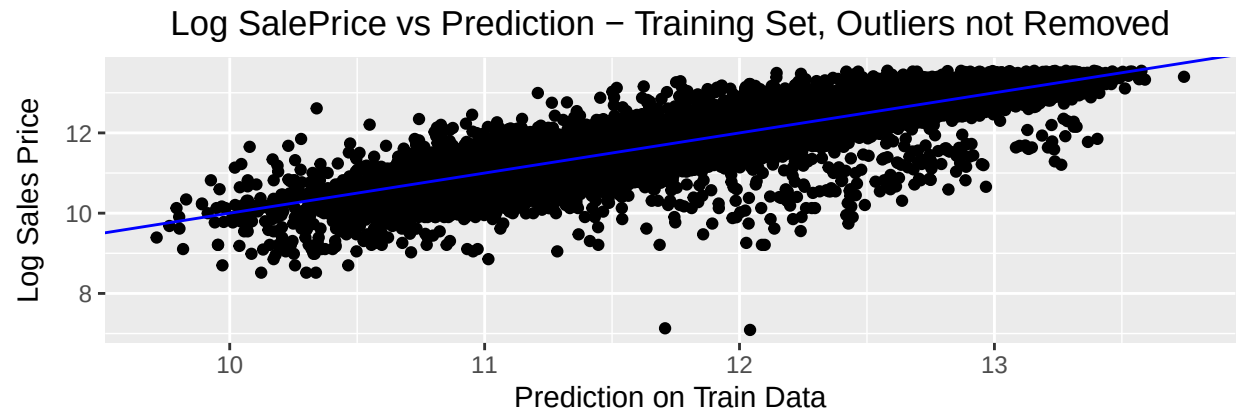
From the new GAM interaction plots above, we now obtain a linear relationship for `FLR1AREA`.

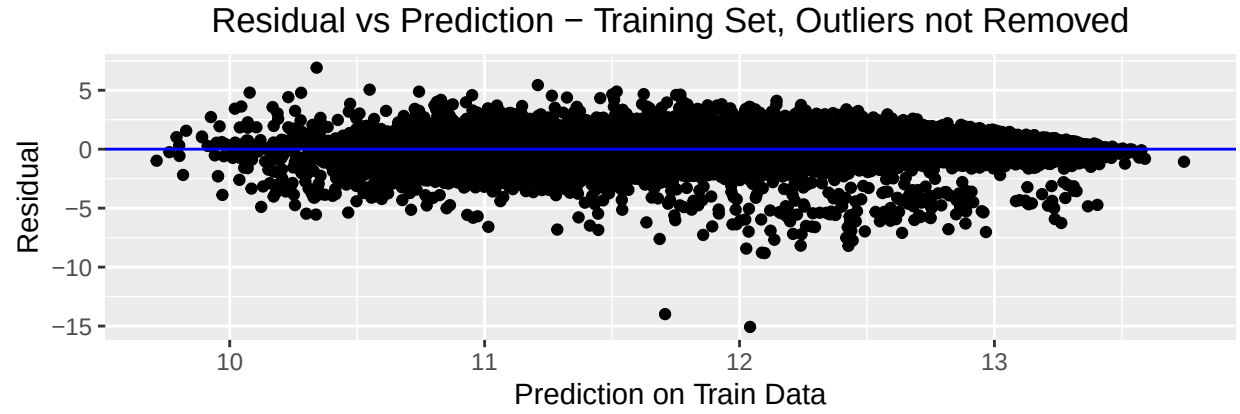
7.1 Model Evaluation 1 - AIC & Residual Analysis

```
mod.gam2$aic
```

```
## [1] 14407.59
```

```
bin.gam2 = eval.train_init(mod.gam2, train, outliers = TRUE)
```





```
train.gam2 = eval.train_update(mod.gam2, train, bin.gam2)
```

In the model's summary, that the model received an AIC value of 14407.59, which will be used to compare with the next model with removed outliers in the train set.

As previously discussed from Model 1, we can see from the first graph that the predictions made by the model are spread around the y=x line, which shows that the predictions are close to the actual values. The second graph highlights the residuals or errors in the fitted values. Residuals are the difference between each predicted data and the actual value of the data. Residuals were then processed the same way as before.

7.2 Remodelling With New Train Set

```
mod.gam2.final <- gam(f2, data = train.gam2)
summary(mod.gam2.final)
```

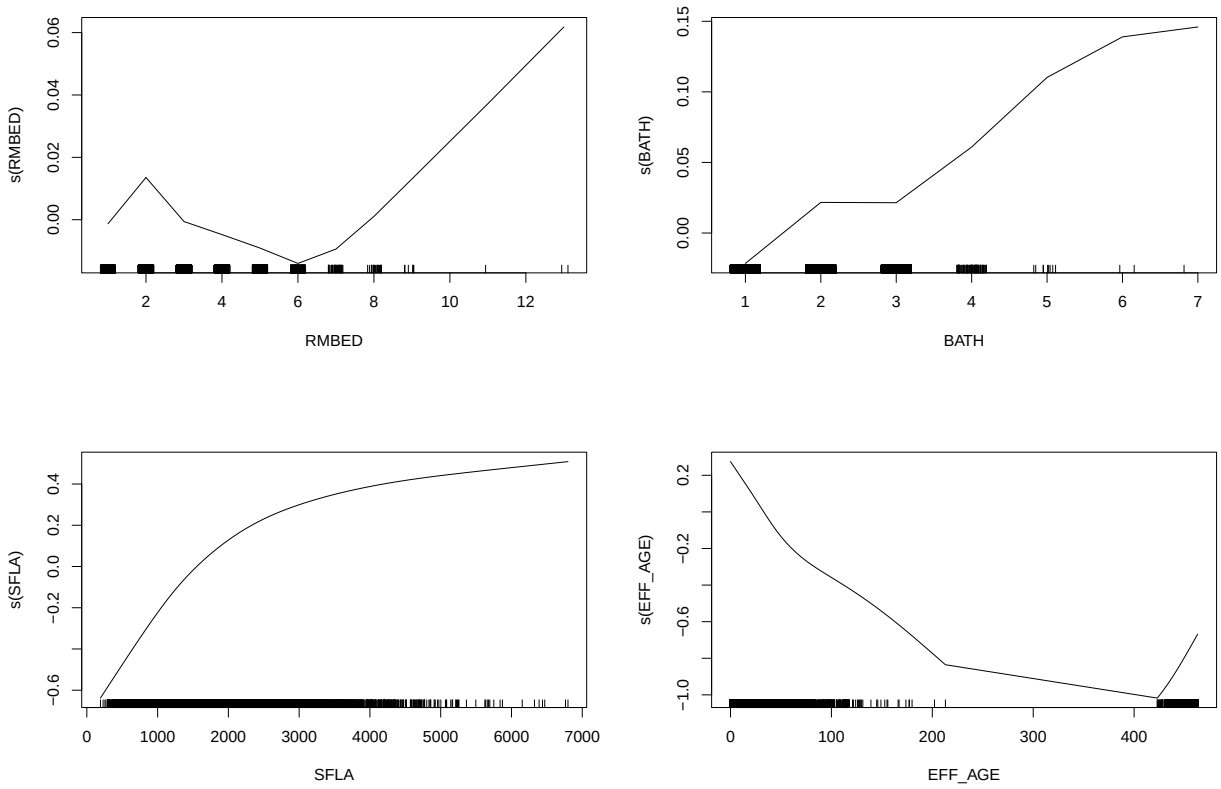
```
##
## Call: gam(formula = f2, data = train.gam2)
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -1.07195 -0.14658 -0.01392  0.13958  1.01283
##
## (Dispersion Parameter for gaussian family taken to be 0.0683)
##
##     Null Deviance: 9379.301 on 22950 degrees of freedom
## Residual Deviance: 1561.844 on 22869 degrees of freedom
## AIC: 3617.433
##
## Number of Local Scoring Iterations: NA
##
## Anova for Parametric Effects
##               Df  Sum Sq Mean Sq   F value                Pr(>F)
## s(RMBED)       1  578.00  578.00  8463.207 < 0.00000000000000022 ***
## s(BATH)        1 1762.38 1762.38 25805.266 < 0.00000000000000022 ***
## s(SFLA)        1 1439.96 1439.96 21084.363 < 0.00000000000000022 ***
## s(EFF_AGE)     1 1448.58 1448.58 21210.605 < 0.00000000000000022 ***
## City          49  908.88   18.55   271.595 < 0.00000000000000022 ***
## BSMT           6   42.50    7.08   103.727 < 0.00000000000000022 ***
## ATTIC          1    3.58    3.58    52.362    0.0000000000004761 ***
## GRADE          4  283.44   70.86  1037.565 < 0.00000000000000022 ***
## Style          4  105.85   26.46   387.462 < 0.00000000000000022 ***
## FLR1AREA       1   23.46   23.46   343.458 < 0.00000000000000022 ***
## Residuals  22869 1561.84    0.07
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Anova for Nonparametric Effects
##             Npar Df Npar F                 Pr(F)
## (Intercept)
## s(RMBED)          3   6.22             0.0003207 ***
## s(BATH)           3   9.78           0.000001904 ***
## s(SFLA)           3 368.39 < 0.00000000000000022 ***
```

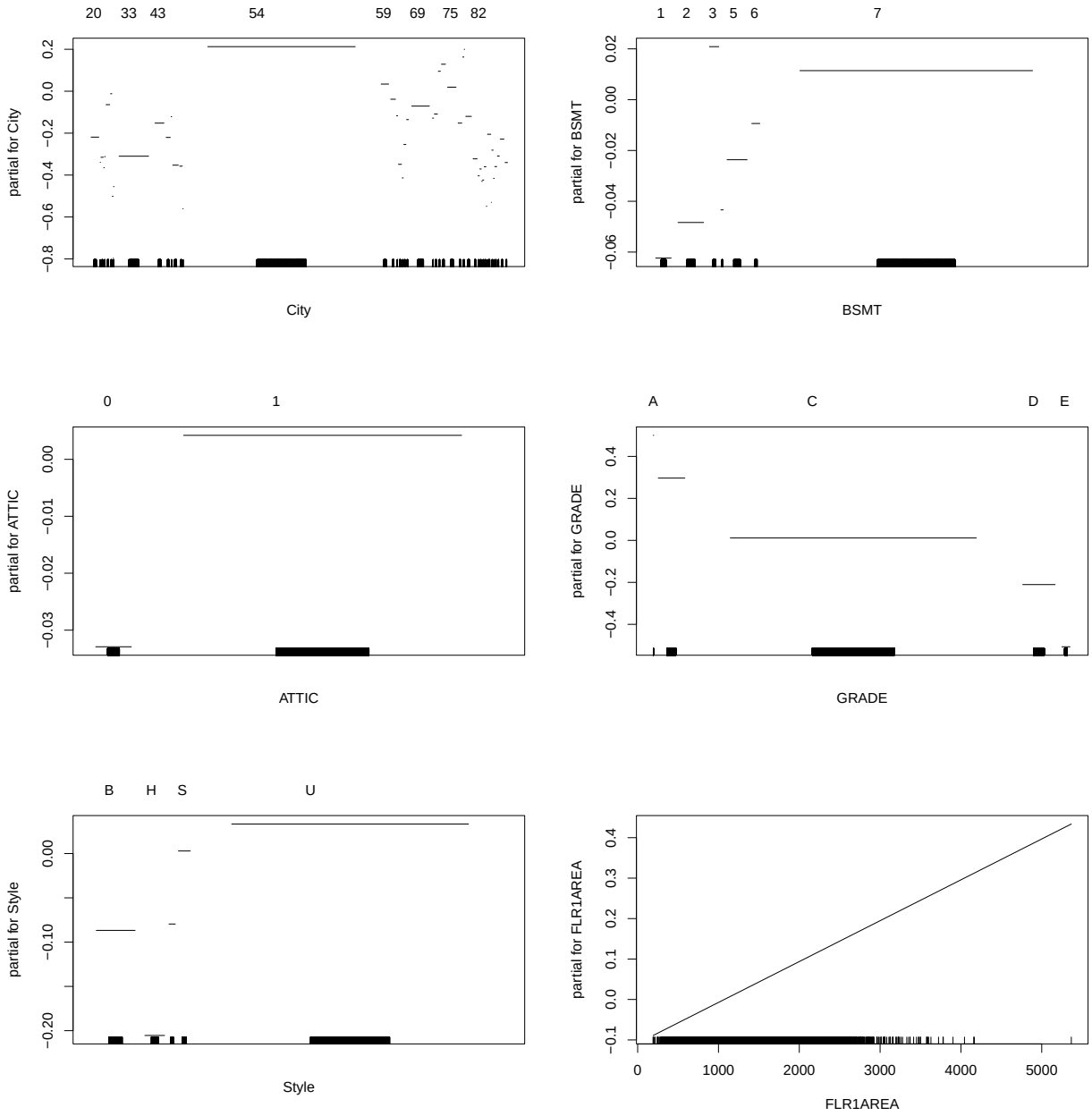
```
## s(EFF_AGE)        3 815.78 < 0.00000000000000022 ***
## City
## BSMT
## ATTIC
## GRADE
## Style
## FLR1AREA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Saving spline plots to pdf
plot2pdf(preplot(mod.gam2.final), "GAM_InteractionPlots_2_final")
```

```
## [1] "Saved to D:/Downloads/GAM_InteractionPlots_2_final.pdf"
```

```
plot(mod.gam2.final)
```





From the splines plot, we can see that there some changes in the spline as a result of fitting with the outliers removed. Just as in Model 1, `RMBED` now has a drastically different fitted spline. This might indicate that the spline fitting is subject to overfitting. However, almost all of the plots are consistent with the previous results, such as `FLR1AREA` resembling a linear function.

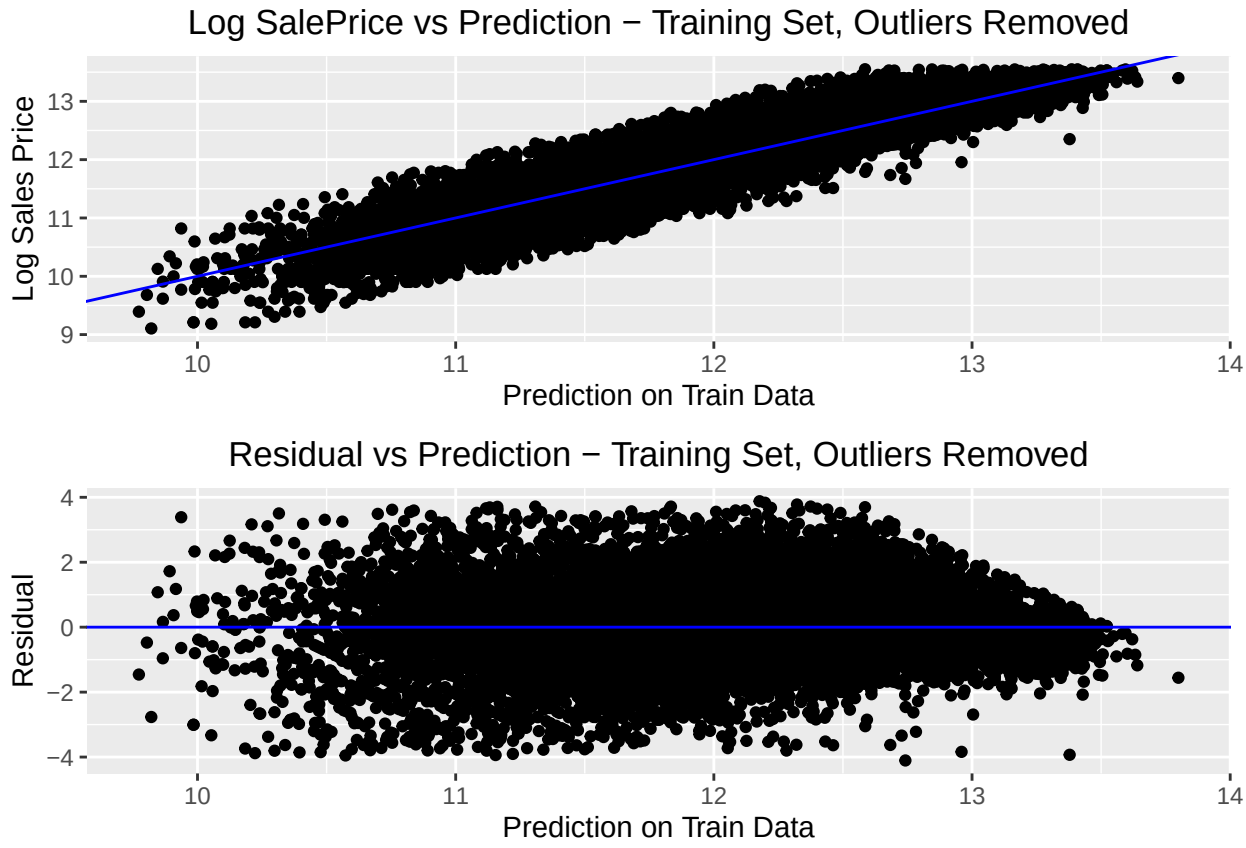
7.3 Model Evaluation 2 - New AIC, Residual Analysis, and Error Analysis

```
data.frame("Outliers Not Removed" = mod.gam2$aic,
           "Outliers Removed" = mod.gam2.final$aic)
```

```
##   Outliers.Not.Removed Outliers.Removed
```

```
## 1             14407.59         3617.433
```

```
eval.train_init(mod.gam2.final, train.gam2, outliers = FALSE)
```



After we have removed the outliers in the training set, we can see that there is a highly significant decrease in the AIC of the model. This means that the model is now performing much better with outliers removed.

From the graphs, we can now see that the predictions are now closer the actual values and the residuals are also closer to 0. It is also worth noting that there are no significant patterns present in the residuals vs prediction plot; the residuals are randomly distributed.

Now, we are going to analyze the error in the predictions of the train set. We will also compare the metrics with the previous model with outliers not removed.

```
merge(stack(comp(exp(mod.gam2$fitted.values), exp(mod.gam2$y))),
      stack(comp(exp(mod.gam2.final$fitted.values), exp(mod.gam2.final$y))),
      by = "ind", sort = FALSE)
```

```
##     ind           values.x           values.y
## 1     n      23359.0000000      22951.0000000
## 2    R2          0.7418046          0.7882518
## 3   MSE 4266420353.4321532 3413781856.0339894
## 4 SEMSE   89013710.6957527   65943542.2599999
## 5  RMSE      65317.8410041      58427.5778724
## 6    SE        424.4210532        384.1917212
## 7   MAE      42460.4564513      39152.4801289
## 8  MAPE         25.8993227         19.7404911
```

From the results, we can conclude that out of the 22951 data in the train set with outliers removed, the model receives an RMSE of 58427.58 and an average absolute error of 19.74%. These values show an increase in performance from the previous model with outliers intact. This can be concluded from the lower RMSE and MAPE values.

7.4 Multicollinearity

```
vif(mod.gam2.final)
```

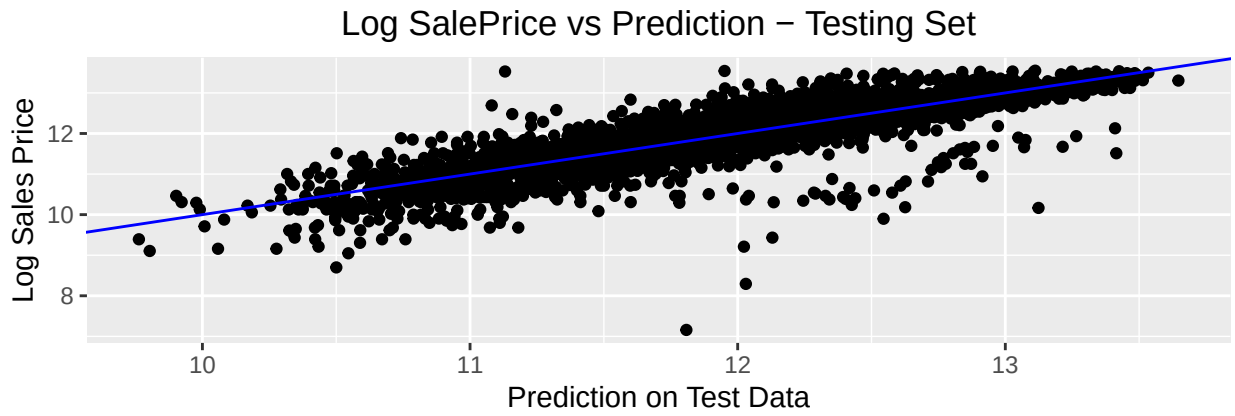
```
##                GVIF Df GVIF^(1/(2*Df))
## s(RMBED)   1.405832  1        1.185678
## s(BATH)    2.059064  1        1.434944
## s(SFLA)    3.177349  1        1.782512
## s(EFF_AGE) 1.233900  1        1.110811
## City       1.813695 49        1.006094
## BSMT       1.857869  6        1.052975
## ATTIC      1.219765  1        1.104430
## GRADE      2.069019  4        1.095142
## Style      1.478643  4        1.050106
## FLR1AREA   1.882855  1        1.372172
```

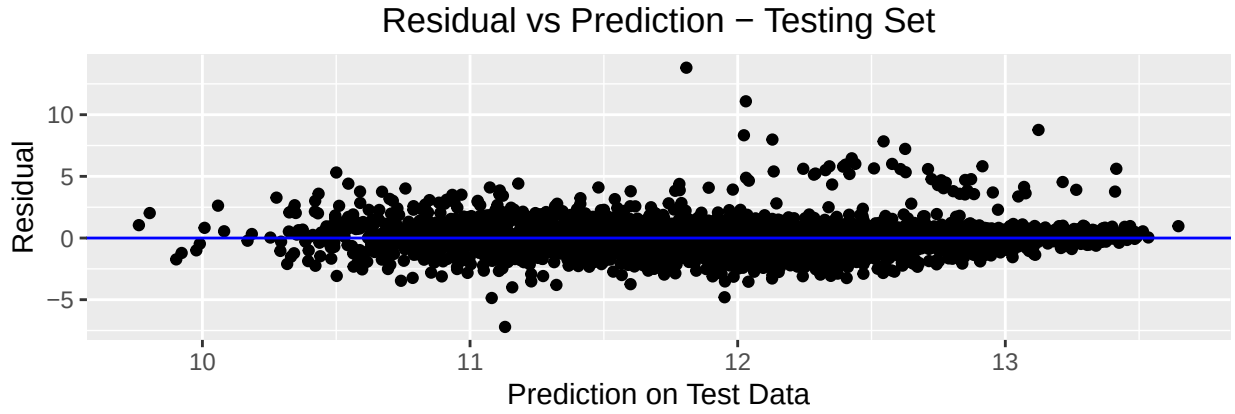
From the VIF scores, we can see that all of them have a relatively small VIF with values below 4. Therefore, we can conclude that each independent variable is not a linear combination of other independent variables. In addition, we can also see that some variables, notably the `RMBED` variable, has lower vif values in comparison to the results in the previous model.

7.5 Prediction and Error Analysis

```
test.gam2 = test
test.gam2$prediction = predict(mod.gam2.final, newdata = test.gam2, type = "response")
```

```
error.gam2 = eval.test(test.gam2$prediction, test.gam2$Price)
```






```
## $n
## [1] 5837
##
## $R2
## [1] 0.7388822
##
## $MSE
## [1] 4394094998
##
## $SEMSE
## [1] 215483304
##
## $RMSE
## [1] 66287.97
##
## $SE
## [1] 866.8233
##
## $MAE
## [1] 41684.66
##
## $MAPE
## [1] 27.70831
```

From the graph, we can see that there are still a lot of predictions that are far away from the y=x line, which means that the model has some inaccurate predictions in the test set. The second graph further highlights

these findings. However, we are unable to conclude a better model between Model 1 and Model 2 from the plots as both result in similar residual plots.

From the results, we can conclude that out of the 5837 data in the test set, the model successfully predicted the prices with RMSE of 66287.97 and average absolute error of 27.71%. These metrics and the model's AIC will be kept for further comparison with the results from Model 1.

---

8 Comparison and Conclusion

After creating 2 models with the gam, we are now going to compare the performance of the models using the computed metrics.

```
comp.aic <- data.frame("Metric" = "AIC",
                       "Model 1" = mod.gam.final$aic,
                       "Model 2" = mod.gam2.final$aic)
comp.error <- merge(stack(error.gam), stack(error.gam2), by="ind", sort = FALSE)
names(comp.error) <- names(comp.aic)
comp.df <- rbind(comp.aic, comp.error)

comp.df
```

```
##   Metric            Model.1            Model.2
## 1    AIC       3430.4065491       3617.4333556
## 2      n       5837.0000000       5837.0000000
## 3     R2          0.7406387          0.7388822
## 4    MSE 4362587754.7767096 4394094998.3381062
## 5  SEMSE  212655499.9861206  215483304.1578088
## 6   RMSE      66049.8883782      66287.9702385
## 7     SE        863.7373809        866.8232692
## 8    MAE      41619.8930012      41684.6550700
## 9   MAPE         27.6207045         27.7083120
```

From the results above, we can see that the AIC of Model 1 is lower than that of Model 2's. This indicates that there are more residues in Model 2 which affects the fitted values of the model which increases the AIC value. Therefore, we can conclude that the Model 1 is a slightly better model in terms of fitted values and its residues.

Similar conclusions can also be taken by evaluating other metrics, such as RMSE and MAPE. Model 1 outperforms Model 2 in both metrics: the RMSE is smaller by approximately 238.082 while the MAPE is smaller by approximately 0.088%. This means that the square root of the average of the squared error and the average absolute percentage of error in the prediction of Model 1 is slightly less than the values obtained from Model 2. Therefore, it can be assumed that Model 1 is more accurate than Model 2.

In conclusion, GAM has managed to fit the independent variables into splines which allows GAM to analyze non-linear relationships between an independent variable and the dependent variable, `Price`, which can not be done in GLM. Splines indicated that there indeed exist some variables that do not have a linear relationship, while others such as `FLR1AREA` indicated differently. However, some spline fitting may be susceptible to overfitting, such as the splines for `RMBED` presented in the models fitted with removed outliers in both Model 1 and Model 2.

Furthermore, removing `Province` and selecting `City` as the selected independent variables the model results in models that with a relatively high performance and accuracy. Although further removal of variables with

low correlation towards price SQFT and RMTOT and the assumption of FLR1AREA having a linear relationship with the response variable resulted in a more inaccurate model, the slight difference in the evaluating metrics indicate that the difference in performance might be insignificant. The removal of the variables also resulted in lower multicollinearity between the independent variables. This further suggests the low importance of the variables and justifies the approach done during feature selection and fitting in Model 2.

These findings may be surprising especially as many might have assumed that the price of a house is highly determined by the total land size (SQFT) of the certain house. On the other hand, the fact that the grade (GRADE) and total square foot living area (SFLA) of a house highly impacts the price of a house has most likely been expected.