Smoothing Model Building

Anusha Raisinghani

- UW ID: 20823986
- Kaggle public score: 0.13236
- Kaggle private score: 0.13801
- Kaggle submission count/times: 30

Summary

To obtain the final model, the following steps were applied:

- Preprocessing: This step involved creating some new variables and converting the existing variables into a more desirable form.
- Data Analysis: This step involved plotting the predictors against the response variable to better guess the relationship between the two. Additionally, some weird data points were identified and fixed in this step.
- Imputing Missing Data: The train and test data had missing values for some predictors such as `ayb`, `yr_rmdl`, `kitchens`, `stories` and `quadrant` which were imputed by intuition as well as by identifying the trends in those predictors.
- Initial Model Building: Then, the initial model was constructed.
- Iterative Model Building: Once the initial model was obtained, based on the residual plots a logarithmic transformation was applied to the response variable. This helped make the residual plots randomly distributed. After transformation, the dimension of the basis used to represent some of the smooth terms (`k`) (such as for `latitude`, `longitude`, `saledate`, `gba`, `bedrm` and `yr_since_rmdl`) was modified, which helped minimise the GCV. Finally, interaction terms were added to further minimise the GCV and obtain the final model.

Preprocessing

The following steps were applied during preprocessing:

- `heat`: This variable was converted to a categorical variable.
- `ac`: This variable was converted to a categorical variable.
- `style`: This variable was converted to a categorical variable.
- `grade`: This variable was converted to a categorical variable.
- `cndtn`: This variable was converted to a categorical variable.
- `saledate`: This variable was converted to a numerical value representing the number of days since 1970/01/01.
- `extwall`: This variable was converted to a categorical variable.
- `roof`: This variable was converted to a categorical variable.
- `intwall`: This variable was converted to a categorical variable.
- `nbhd`: This variable was converted to a categorical variable.
- `ward`: This variable was converted to a categorical variable.
- `quadrant`: This variable was converted to a categorical variable.

Transformation

- `price`: I took the logarithm of the response variate price.

New Variables

- `yr_since_rmdl`: Difference in years between `saledate` and `yr_since_rmdl`
- `yr_since_imprv`: Difference in years between `saledate` and `eyb`
- `yr_since_b`: Difference in years between `saledate` and `ayb`
- `ysb1`: Boolean variable for whether the `yr_since_b` > 0
- `ysb2`: Boolean variable for whether the `yr_since_b` > 125

Other Preprocessing

Upon plotting the `bathrm` vs the `price`, it was found that there was a data point with 0 bathrooms. This data point had "No Data" listed for heat, 0 rooms, 0 bedrooms, etc. Hence, this data point was removed from the dataset. Similarly, for stories, there was an observation with 25 stories. The style for this house was "2.5 Story Fin" and hence the 25 was likely a typo. This was changed to 2.5.

Missing data handling

- `ayb`: Upon plotting the distribution for `eyb - ayb`, it was observed that the difference follows a normal distribution, Hence, the missing values were imputed by taking the difference between `eyb` for those data points and the mean of `eyb-ayb`.
- `yr_since_rmdl`: A missing `yr_rmdl` represents the fact that the house was never remodeled. Hence, the `yr_since_rmdl` was imputed by adding 2 to the maximum of the `yr_since_rmdl`.
- `stories`: This variable was imputed with the mean of the stories for houses with the same style.
- `kitchens`: This variable was imputed with the value of `rooms-(bedrm+bathrm)`.
- `quadrant`: Upon plotting the latitude and longitude grouped by the quadrant, it seemed that the quadrants formed clusters. Hence, quadrant was imputed by using the quadrant of the houses in the same region (by latitude and longitude).

Model Building

Main package used: `mgcv`

Final Model

- The final model is `price ~ ti(saledate, longitude) + ti(saledate, fireplaces) + ti(landarea, longitude) + s(bathrm) + s(hf_bathrm, k=5) + heat + ac + s(rooms) + s(bedrm, k=13) + stories + style + s(saledate, k=25) + s(gba,k=20) + grade + cndtn + roof + intwall + kitchens + s(fireplaces) + s(landarea) + s(latitude, k=20) + s(longitude, k=20) + nbhd + ward + quadrant + ysb1 + ysb2 + s(yr_since_rmdl, k=30) + s(yr_since_imprv)`

1.Preprocessing

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.3.6      v purrr   1.0.1
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.4.1
## v readr   2.1.3      v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
```

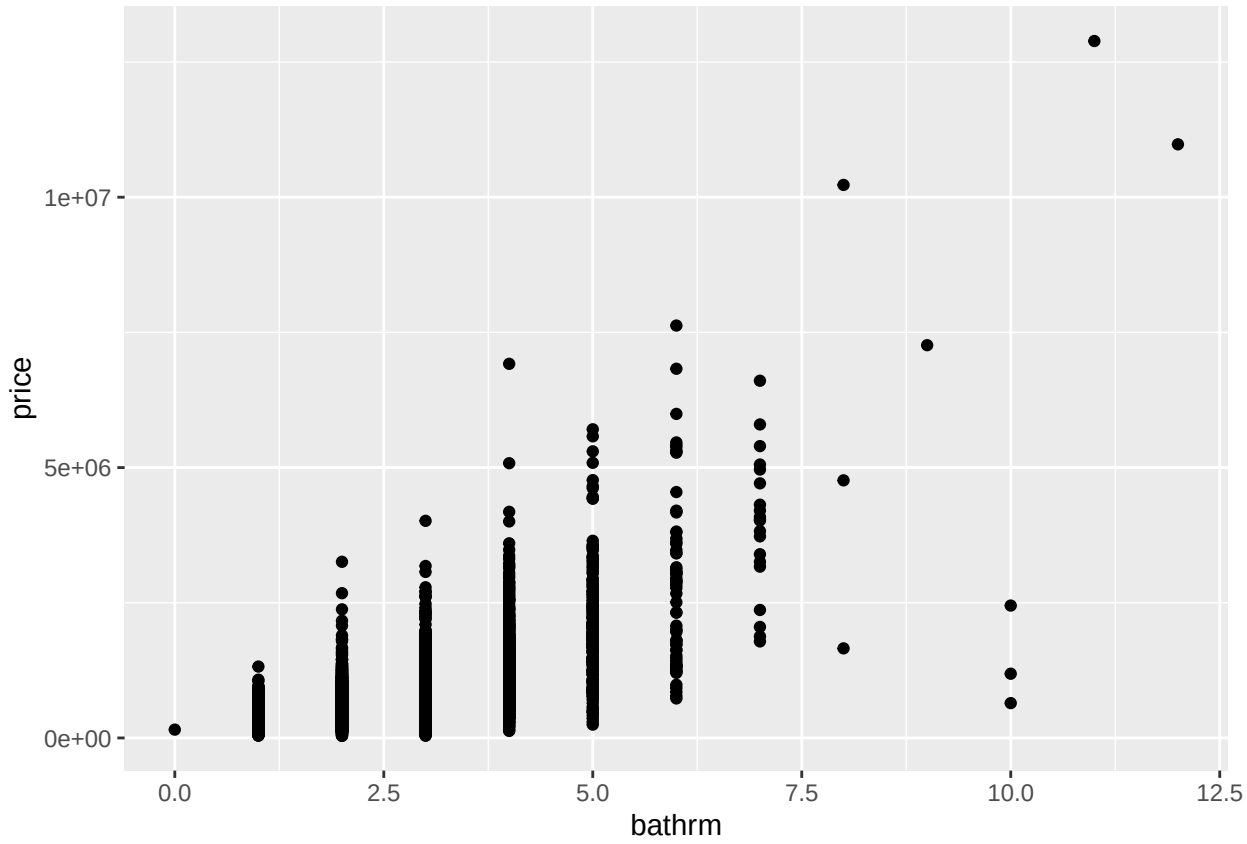
```
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(mgcv)
```

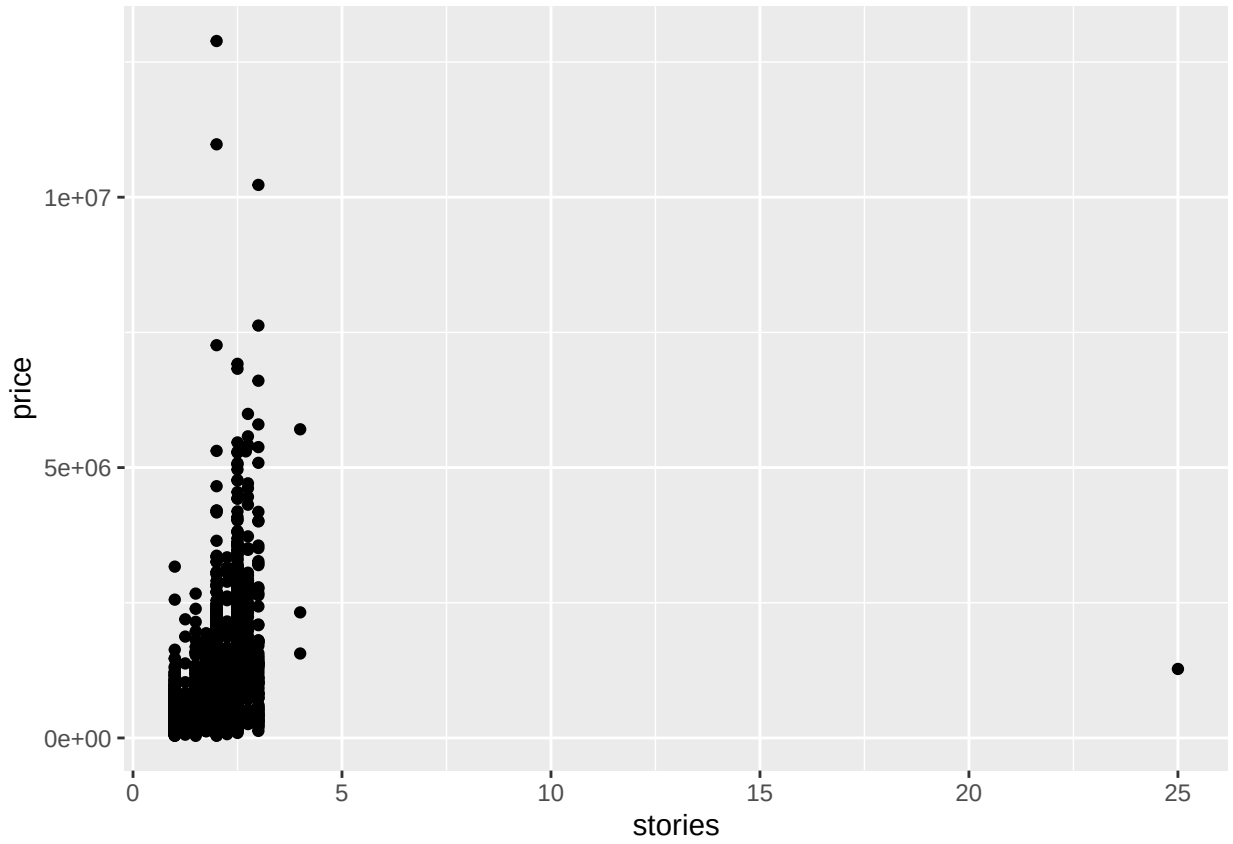
```
## Loading required package: nlme
##
## Attaching package: 'nlme'
##
## The following object is masked from 'package:dplyr':
##
##     collapse
##
## This is mgcv 1.8-40. For overview type 'help("mgcv-package")'.
```

1.1 Loading data

```
load("smooth.Rdata")

dtrain$heat <- as.factor(dtrain$heat)
dtrain$ac <- as.factor(dtrain$ac)
dtrain$saledate <- as.numeric(as.Date(dtrain$saledate))
dtrain$style <- as.factor(dtrain$style)
dtrain$grade <- as.factor(dtrain$grade)
dtrain$cndtn <- as.factor(dtrain$cndtn)
dtrain$extwall <- as.factor(dtrain$extwall)
dtrain$roof <- as.factor(dtrain$roof)
dtrain$intwall <- as.factor(dtrain$intwall)
dtrain$nbhd <- as.factor(dtrain$nbhd)
dtrain$ward <- as.factor(dtrain$ward)
dtrain$quadrant <- as.factor(dtrain$quadrant)

dtrain$yr_since_rmdl <- 1970+dtrain$saledate/365.25-dtrain$yr_rmdl
dtrain$yr_since_imprv <- 1970+dtrain$saledate/365.25-dtrain$eyb

ggplot(dtrain, aes(x = bathrm, y = price)) +
  geom_point()
```




```
# There's a house with 0 bathrooms. Upon close inspection of this data point,
# we see weird data, there's "No Data" for heat, no rooms, and ayb > eyb.
# We get rid of this data point:
dtrain <- dtrain[dtrain$bathrm > 0,]
ggplot(dtrain[!is.na(dtrain$stories),], aes(x = stories, y = price)) +
  geom_point()
```



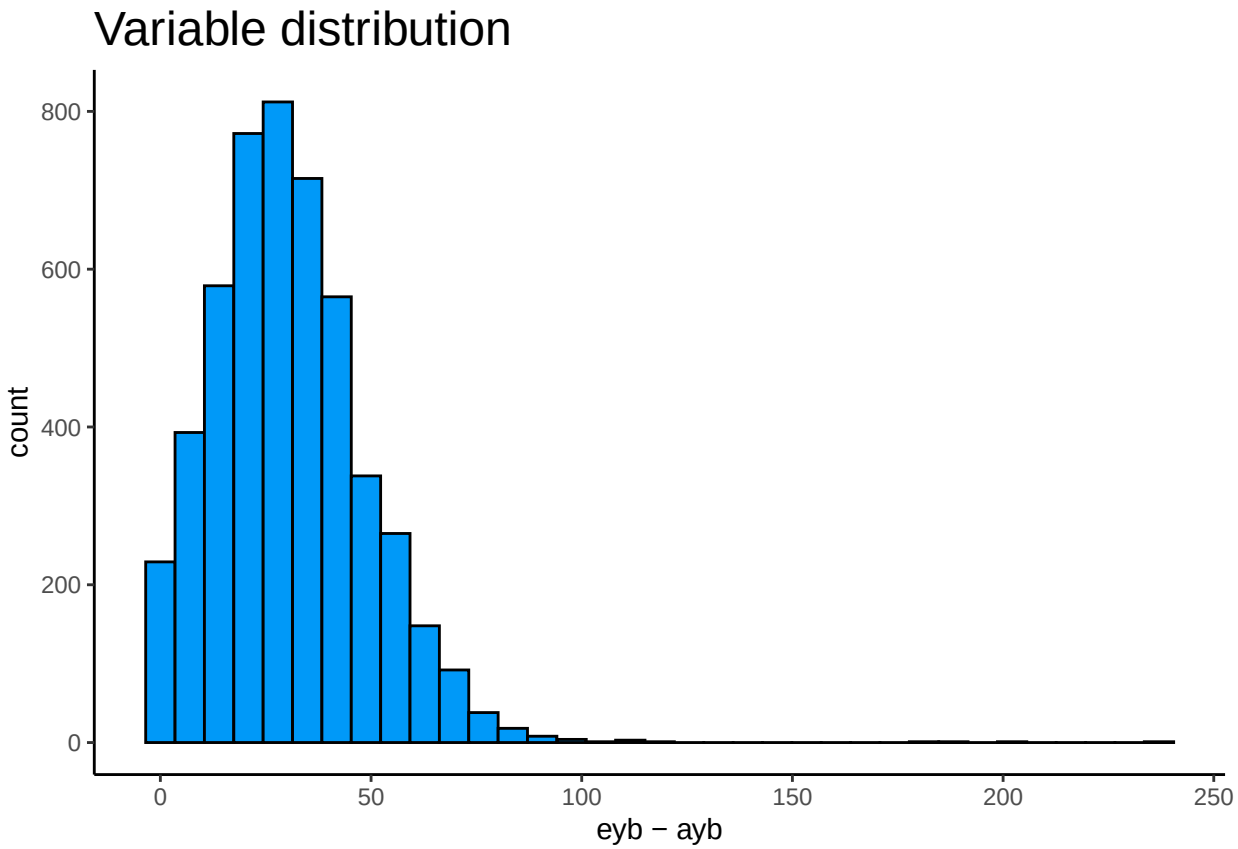

```
# This is likely a typo
dtrain[249, 'stories'] <- 2.5
```

1.2 Missing data handling

For `ayb`:

Upon plotting the distribution of `eyb - ayb`, we get a bell-shaped curve indicating a normal distribution. Hence, the `ayb` is imputed by using the mean of `eyb-ayb`.

```
# This follows a normal distribution
ggplot(dtrain[!is.na(dtrain$ayb),], aes(eyb - ayb)) +
  geom_histogram(color = "#000000", fill = "#0099F8", bins=35) +
  ggtitle("Variable distribution") +
  theme_classic() +
  theme(plot.title = element_text(size = 18))
```




```
# We do mean imputation
mean_ayb_eyb <- round(mean(dtrain$eyb - dtrain$ayb, na.rm=TRUE))
dtrain[is.na(dtrain$ayb),'ayb'] <- dtrain[is.na(dtrain$ayb),'eyb'] - mean_ayb_eyb

# Create yr_since_b variable
dtrain$yr_since_b <- 1970+dtrain$saledate/365.25-dtrain$ayb
```

For `yr_rmdl` (`yr_since_rmdl`)

We are using the `yr_since_rmdl` in our model, and it has missing values due to missing values in `yr_rmdl`. These houses were never remodeled and so for imputing the `yr_since_rmdl`, the 2 plus the maximum of the `yr_since_rmdl` is taken.

```
impute_value_rmdl <- max(dtrain$yr_since_rmdl, na.rm=TRUE) + 2
dtrain[is.na(dtrain$yr_since_rmdl), 'yr_since_rmdl'] <- impute_value_rmdl
```

For `stories`

This was missing for 4 observations. The styles of these houses are listed as "2 Story" and "2.5 Story". Hence, these houses were imputed by taking the mean of the stories of "2 Story" and "2.5 Story" style houses.

```
dtrain[is.na(dtrain$stories), ]
```

```
##      bathrm hf_bathrm       heat ac rooms bedrm  ayb yr_rmdl  eyb stories
## 747       2         1 Forced Air  Y     8     4 1940      NA 1940      NA
## 1233      3         1 Forced Air  Y     7     4 2014      NA 2015      NA
## 2306      5         1  Warm Cool  Y    13     5 2016      NA 2017      NA
## 3857      5         1 Forced Air  Y    12     5 2014      NA 2016      NA
##      saledate   price  gba         style        grade     cndtn      extwall
```

```
## 747     17627  429900 2124       2 Story  Low Quality      Poor Brick/Stucco
## 1233    16317  765100 2816       2 Story Good Quality Very Good Common Brick
## 2306    16936 2411400 4013       2 Story    Excellent Excellent Stone/Siding
## 3857    15229 1078700 7163 2.5 Story Fin Good Quality Very Good Stone/Siding
##              roof       intwall kitchens fireplaces landarea latitude longitude
## 747      Built Up Hardwood/Carp        1          0     1062 40.69924 -74.14419
## 1233 Comp Shingle      Hardwood        1          1     5565 40.77058 -74.14737
## 2306 Comp Shingle      Hardwood        1          1     8878 40.73016 -74.24409
## 3857 Comp Shingle Hardwood/Carp        1          2    11925 40.77587 -74.20430
##      nbhd   ward quadrant yr_since_rmdl yr_since_imprv yr_since_b
## 747    B1 Ward 6       NE      110.3559     78.2600958 78.2600958
## 1233   F3 Ward 4       NE      110.3559     -0.3264887  0.6735113
## 2306   E6 Ward 3       NW      110.3559     -0.6317591  0.3682409
## 3857   B2 Ward 4       NW      110.3559     -4.3052704 -2.3052704
```

```
# Get the mean of stories grouped by style
tapply(dtrain$stories, dtrain$style, mean, na.rm=TRUE)
```

```
##         1 Story   1.5 Story Fin 1.5 Story Unfin         2 Story   2.5 Story Fin
##        1.016762        1.484227        1.442308        1.994413        2.444096
## 2.5 Story Unfin         3 Story         4 Story        Bi-Level     Split Foyer
##        2.369898        2.863287        3.333333        2.000000        1.277778
##     Split Level
##        1.500000
```

```
# Add the mean of stories as a column (avg_stories) to dtrain
dtrain <- dtrain %>%
  group_by(style) %>%
  mutate(avg_stories = mean(stories, na.rm = TRUE))
# Set the missing stories to the mean grouped by style
dtrain <- dtrain %>%
  mutate(stories = ifelse(is.na(stories), avg_stories, stories))
# Drop the avg_stories column
dtrain <- dtrain %>%
  select(-avg_stories)
```

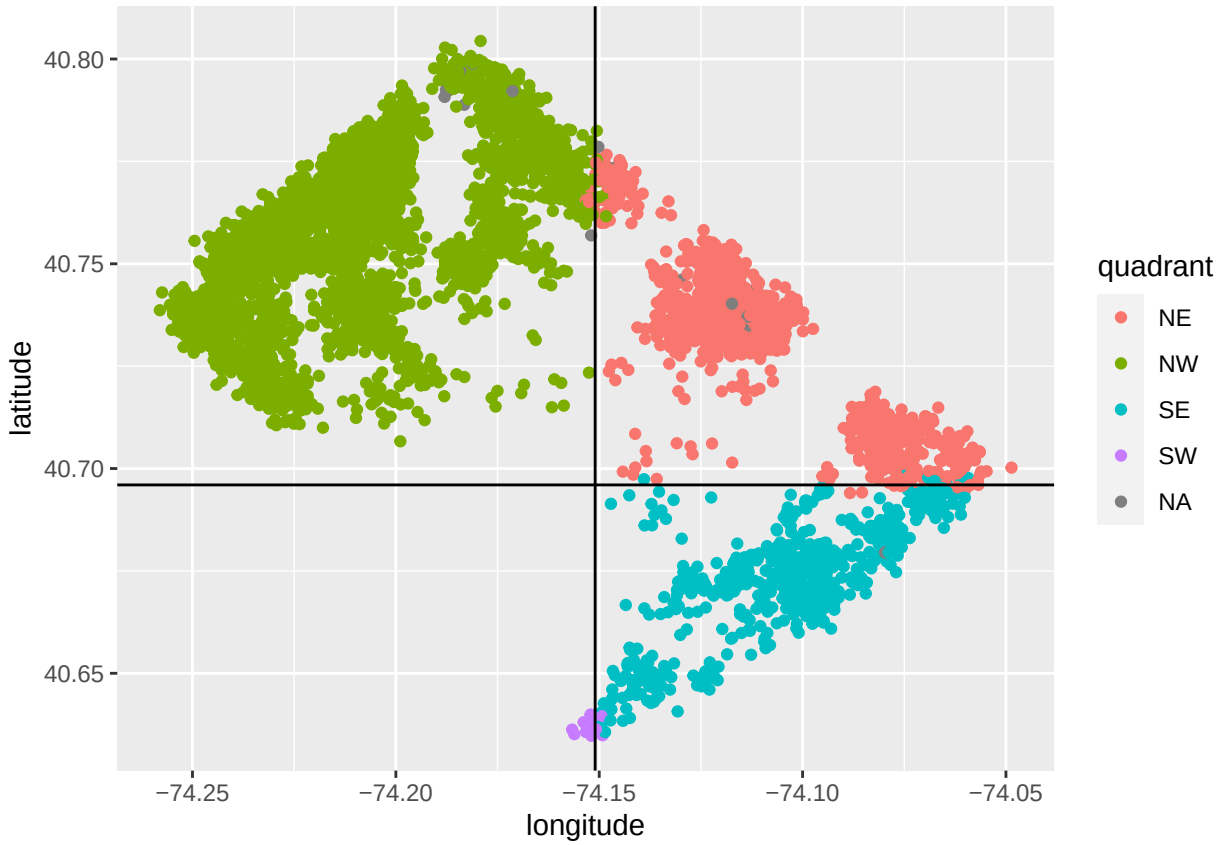
For kitchens

```
dtrain[which(is.na(dtrain$kitchens)), 'kitchens'] <-
  dtrain[which(is.na(dtrain$kitchens)), 'rooms'] - (dtrain[which(is.na(dtrain$kitchens)), 'bedrm']
                                                    + dtrain[which(is.na(dtrain$kitchens)), 'bathrm'])
```

For quadrant

When we plot the graph of latitude and longitude grouped by quadrant, we see clear distinctions between each quadrant. So the quadrant is imputed on the basis of the latitude and longitude of the house.

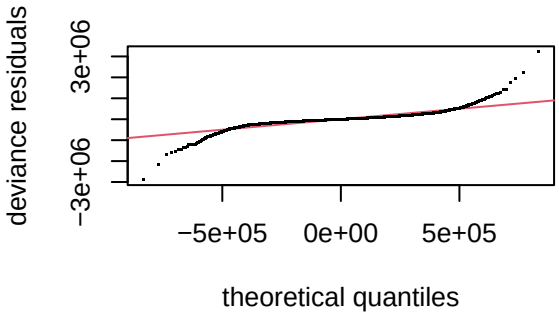
```
ggplot(dtrain, aes(x = longitude, y = latitude, color = quadrant)) +
  geom_point() + geom_vline(xintercept=-74.151) + geom_hline(yintercept = 40.696)
```

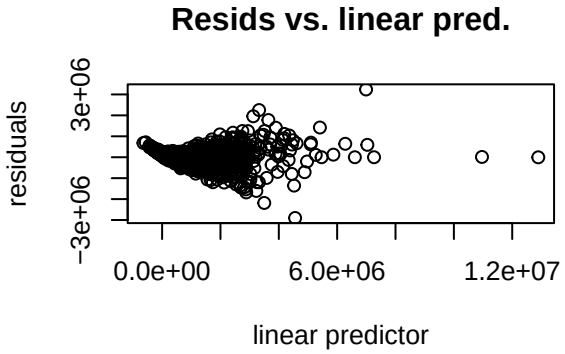


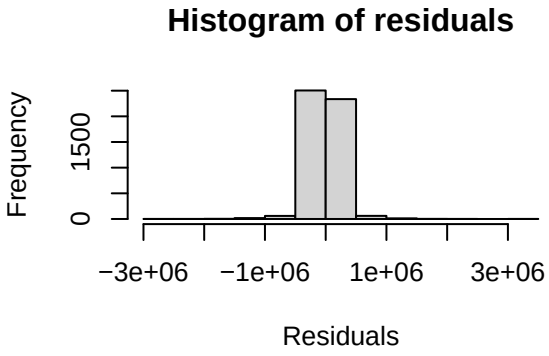

```
dtrain[(dtrain$longitude < -74.151) & (dtrain$latitude > 40.696) & is.na(dtrain$quadrant), 'quadrant'] <
dtrain[(dtrain$longitude > -74.151) & (dtrain$latitude > 40.696) & is.na(dtrain$quadrant), 'quadrant'] <
dtrain[(dtrain$longitude > -74.151) & (dtrain$latitude < 40.696) & is.na(dtrain$quadrant), 'quadrant'] <
```

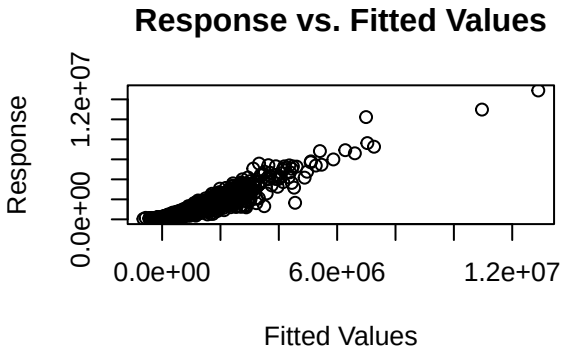
2. Model building

```
gam1 <- gam(price ~ s(bathrm) + s(hf_bathrm, k=5) + heat + ac + s(rooms) + s(bedrm)
            + s(stories) + s(saledate) + s(gba) + style + grade + cndtn + extwall + roof
            + intwall + s(kitchens, k=4) + s(fireplaces) + s(landarea) + s(latitude)
            + s(longitude) + nbhd + ward + quadrant + s(yr_since_b) + s(yr_since_rmdl)
            + s(yr_since_imprv), data=dtrain)
gam.check(gam1)
```







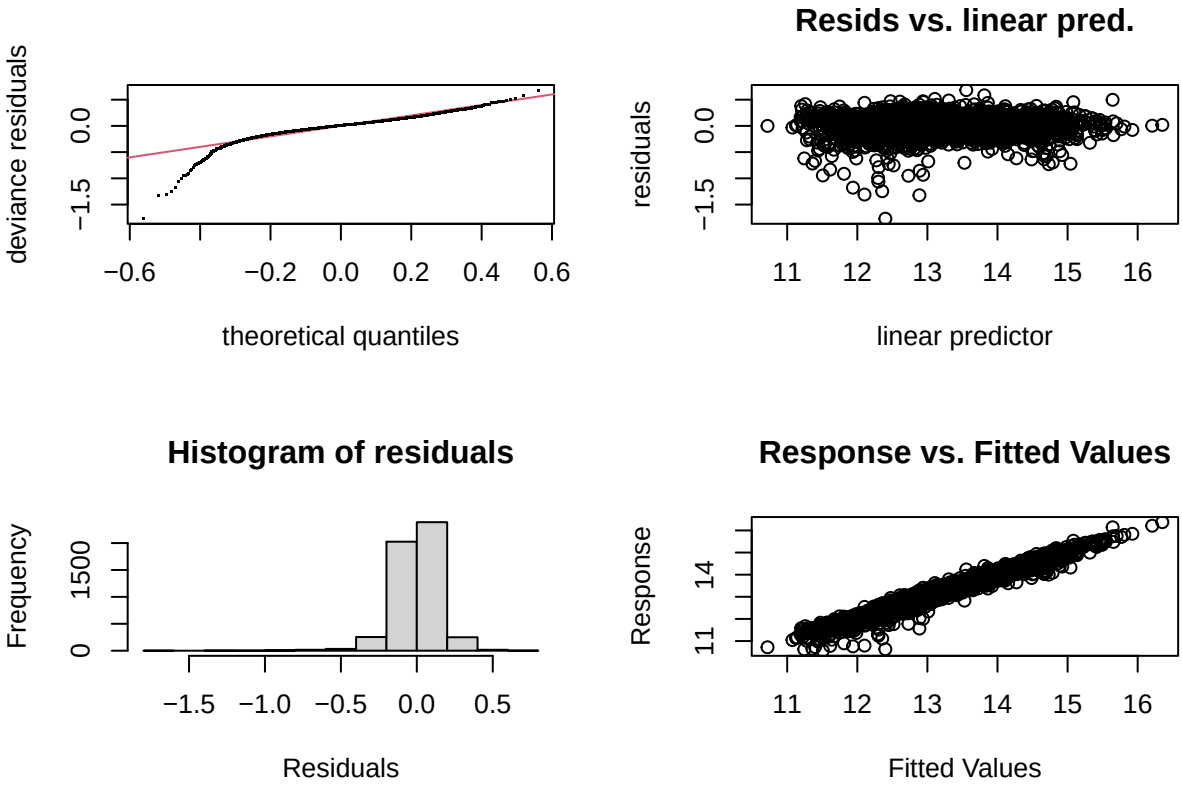



```
##
## Method: GCV   Optimizer: magic
## Smoothing parameter selection converged after 57 iterations.
## The RMS GCV score gradient at convergence was 721.6674 .
## The Hessian was positive definite.
## Model rank =  262 / 264
##
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##                    k'  edf k-index p-value
## s(bathrm)         9.00 8.68    0.97    0.01 **
## s(hf_bathrm)      4.00 3.86    1.02    0.94
## s(rooms)          9.00 8.96    1.01    0.60
## s(bedrm)          9.00 8.79    1.03    0.96
## s(stories)        9.00 2.97    1.00    0.40
## s(saledate)       9.00 8.29    0.99    0.28
## s(gba)            9.00 7.00    1.06    1.00
## s(kitchens)       3.00 1.00    1.01    0.71
## s(fireplaces)     9.00 9.00    1.00    0.40
## s(landarea)       9.00 8.87    0.99    0.12
## s(latitude)       9.00 6.32    1.03    0.96
## s(longitude)      9.00 2.13    0.98    0.11
## s(yr_since_b)     9.00 8.03    0.94  <2e-16 ***
## s(yr_since_rmdl)  9.00 8.54    1.01    0.73
## s(yr_since_imprv) 9.00 6.30    0.99    0.16
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

From the Resids vs linear pred. plot, we can see some pattern and so we transform the response variate to

the logarithm and fit the model again.

```
dtrain$price <- log(dtrain$price)
gam2 <- gam(price ~ s(bathrm) + s(hf_bathrm, k=5) + heat + ac + s(rooms) + s(bedrm)
            + s(stories) + s(saledate) + s(gba) + style + grade + cndtn + extwall + roof
            + intwall + s(kitchens, k=4) + s(fireplaces) + s(landarea) + s(latitude)
            + s(longitude) + nbhd + ward + quadrant + s(yr_since_b) + s(yr_since_rmdl)
            + s(yr_since_imprv), data=dtrain)
gam.check(gam2)
```



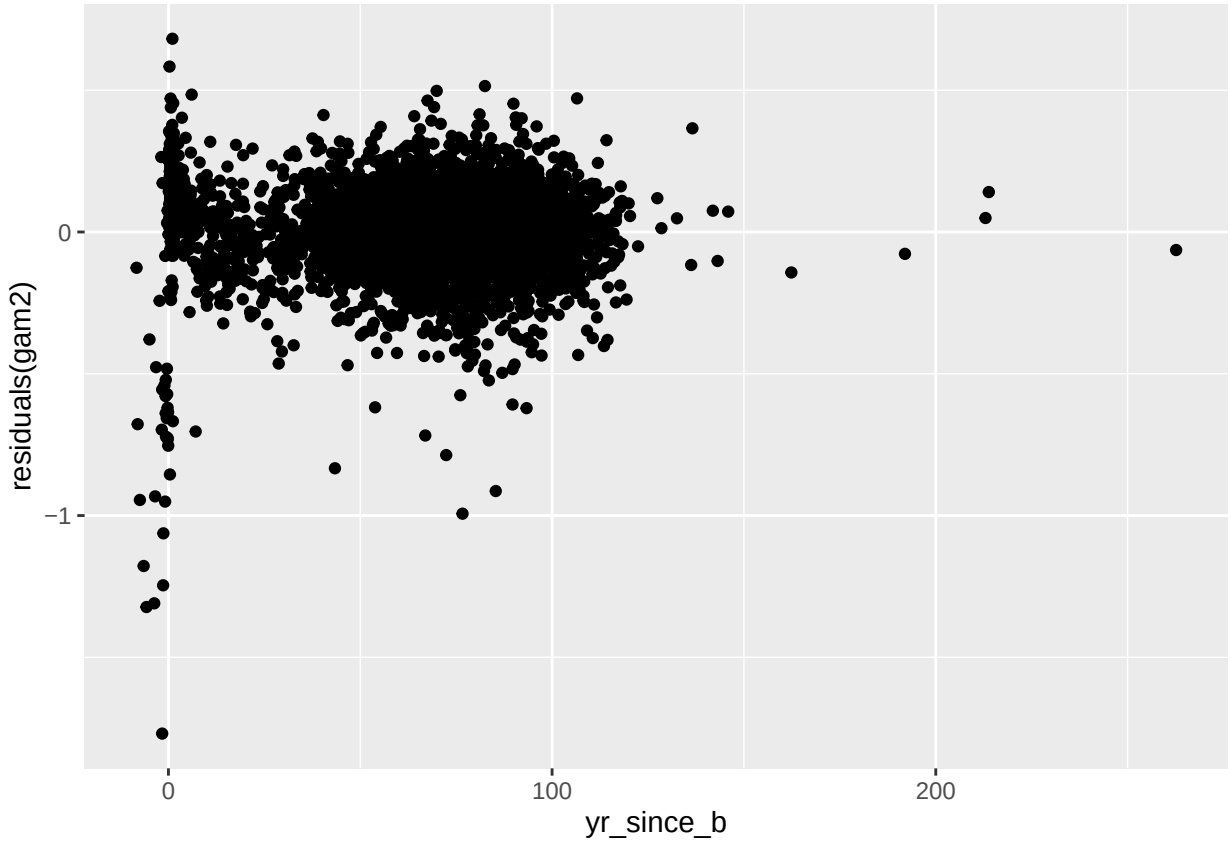

```
##
## Method: GCV   Optimizer: magic
## Smoothing parameter selection converged after 27 iterations.
## The RMS GCV score gradient at convergence was 6.125144e-08 .
## The Hessian was positive definite.
## Model rank =  262 / 264
##
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##                   k'  edf k-index p-value
## s(bathrm)       9.00 3.98    1.00   0.450
## s(hf_bathrm)    4.00 1.78    1.01   0.735
## s(rooms)        9.00 6.80    1.01   0.775
## s(bedrm)        9.00 2.05    0.99   0.305
## s(stories)      9.00 1.35    1.00   0.645
## s(saledate)     9.00 8.71    1.01   0.850
## s(gba)          9.00 8.63    1.00   0.675
## s(kitchens)     3.00 1.00    0.99   0.380
## s(fireplaces)   9.00 4.50    0.99   0.295
```

```
## s(landarea)       9.00 8.14    0.99   0.330
## s(latitude)       9.00 7.98    1.01   0.815
## s(longitude)      9.00 8.23    1.02   0.900
## s(yr_since_b)     9.00 8.64    0.82  <2e-16 ***
## s(yr_since_rmdl)  9.00 8.29    0.97   0.015 *
## s(yr_since_imprv) 9.00 8.92    0.97   0.025 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Check the GCV
gam2$gcv.ubre
```

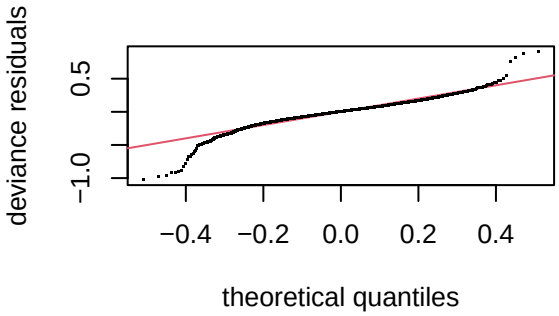
```
##     GCV.Cp
## 0.02384417
```

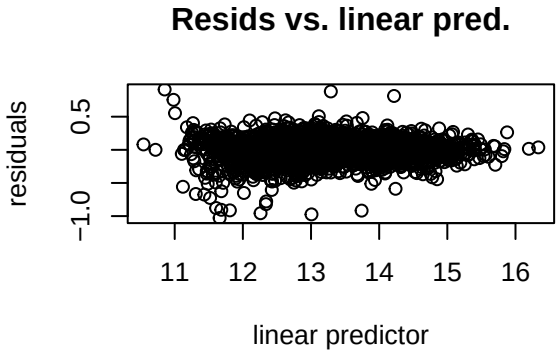
```
ggplot(dtrain, aes(x = yr_since_b, y = residuals(gam2))) +
  geom_point()
```

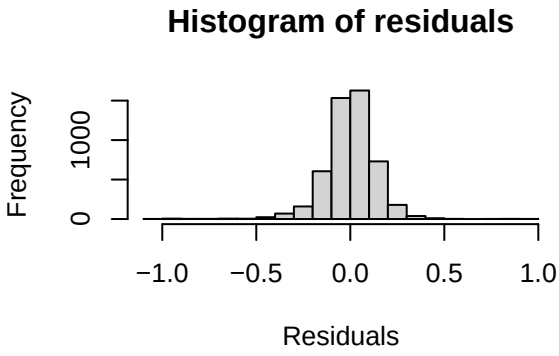


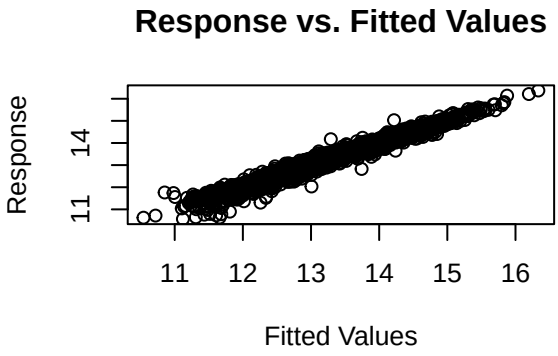
There's a pattern in `yr_since_b` and so we separate that variable into `ysb1` and `ysb2`.

```
dtrain$ysb1 <- dtrain$yr_since_b > 0
dtrain$ysb2 <- dtrain$yr_since_b > 125
gam3 <- gam(price ~ s(bathrm) + s(hf_bathrm, k=5) + heat + ac + s(rooms) + s(bedrm)
            + s(stories) + s(saledate) + s(gba) + style + grade + cndtn + extwall + roof
            + intwall + s(kitchens, k=4) + s(fireplaces) + s(landarea) + s(latitude)
            + s(longitude) + nbhd + ward + quadrant + ysb1 + ysb2 + s(yr_since_rmdl)
            + s(yr_since_imprv), data=dtrain)
gam.check(gam3)
```










```
##
## Method: GCV   Optimizer: magic
## Smoothing parameter selection converged after 29 iterations.
## The RMS GCV score gradient at convergence was 1.510178e-08 .
## The Hessian was positive definite.
## Model rank =  255 / 257
##
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##                     k'  edf k-index p-value
## s(bathrm)         9.00 3.44    1.01   0.760
## s(hf_bathrm)      4.00 1.92    1.02   0.950
## s(rooms)          9.00 7.65    1.00   0.580
## s(bedrm)          9.00 7.98    0.99   0.140
## s(stories)        9.00 1.00    1.01   0.745
## s(saledate)       9.00 8.83    1.02   0.940
## s(gba)            9.00 8.23    1.00   0.460
## s(kitchens)       3.00 1.00    1.00   0.380
## s(fireplaces)     9.00 5.26    0.99   0.205
## s(landarea)       9.00 6.49    1.01   0.735
## s(latitude)       9.00 8.27    1.02   0.910
## s(longitude)      9.00 8.31    1.02   0.865
## s(yr_since_rmdl)  9.00 8.36    0.97   0.035 *
## s(yr_since_imprv) 9.00 6.27    0.99   0.185
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

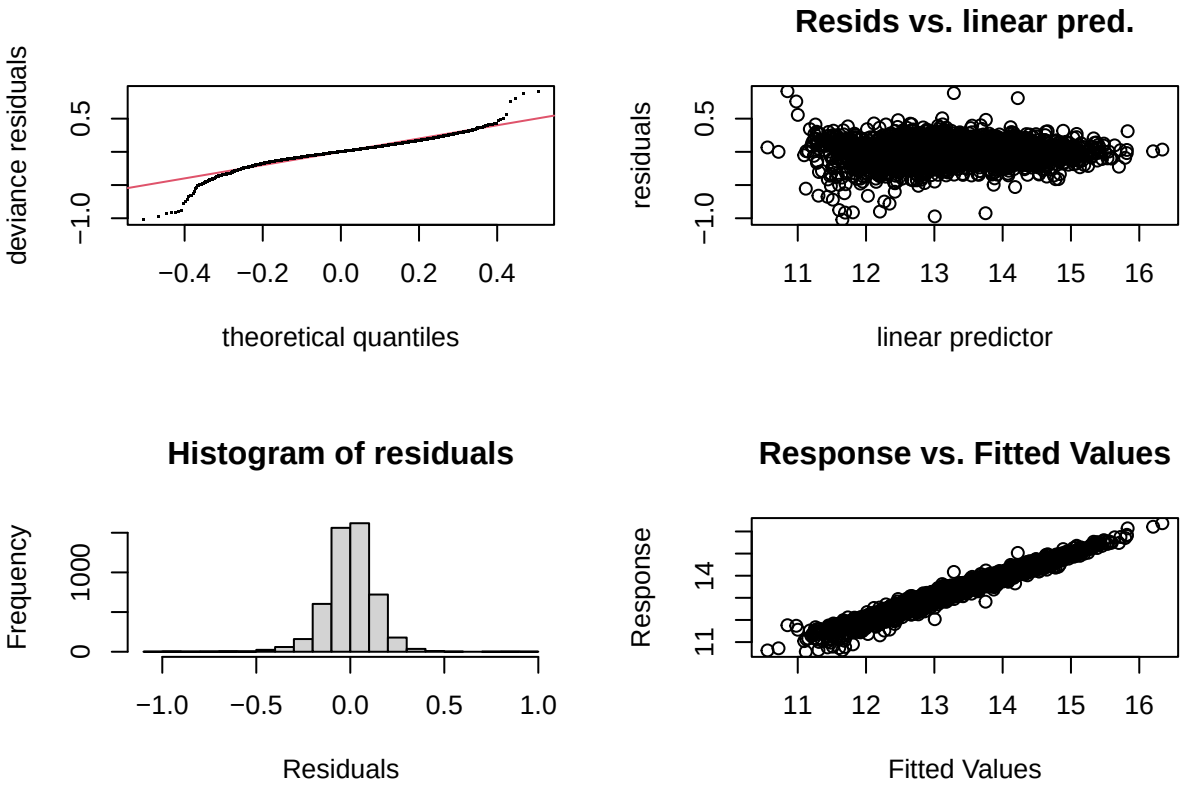
```
# Check the GCV
gam3$gcv.ubre
```

```
##     GCV.Cp
## 0.01963184
```

This helps reduce the GCV from 0.0238442 to 0.0196318

Next, from the `gam.check()` output, we can see that the `k` for `yr_since_rmdl` needs to be increased. This helps reduce the GCV as well.

```
gam4 <- gam(price ~ s(bathrm) + s(hf_bathrm, k=5) + heat + ac + s(rooms) + s(bedrm)
            + s(stories) + s(saledate) + s(gba) + style + grade + cndtn + extwall + roof
            + intwall + s(kitchens, k=4) + s(fireplaces) + s(landarea) + s(latitude)
            + s(longitude) + nbhd + ward + quadrant + ysb1 + ysb2 + s(yr_since_rmdl, k=30)
            + s(yr_since_imprv), data=dtrain)
gam.check(gam4)
```




```
##
## Method: GCV   Optimizer: magic
## Smoothing parameter selection converged after 29 iterations.
## The RMS GCV score gradient at convergence was 2.088461e-08 .
## The Hessian was positive definite.
## Model rank =  275 / 277
##
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##                    k'   edf k-index p-value
## s(bathrm)        9.00  3.36    1.01    0.77
## s(hf_bathrm)     4.00  1.92    1.03    0.97
```

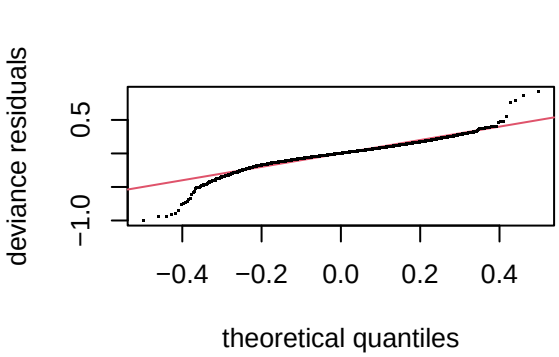
```
## s(rooms)           9.00  7.62    1.00    0.67
## s(bedrm)           9.00  7.93    0.99    0.14
## s(stories)         9.00  1.00    1.02    0.82
## s(saledate)        9.00  8.85    1.02    0.85
## s(gba)             9.00  8.15    1.00    0.47
## s(kitchens)        3.00  1.00    1.00    0.37
## s(fireplaces)      9.00  5.13    0.99    0.30
## s(landarea)        9.00  5.71    1.01    0.72
## s(latitude)        9.00  8.27    1.02    0.86
## s(longitude)       9.00  8.35    1.02    0.85
## s(yr_since_rmdl)  29.00 23.86    0.99    0.27
## s(yr_since_imprv)  9.00  6.18    0.98    0.13
```

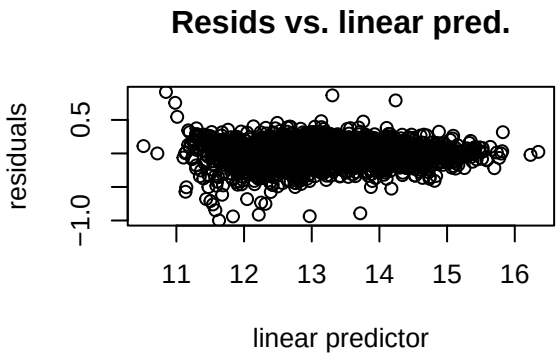
```
# Check the GCV
gam4$gcv.ubre
```

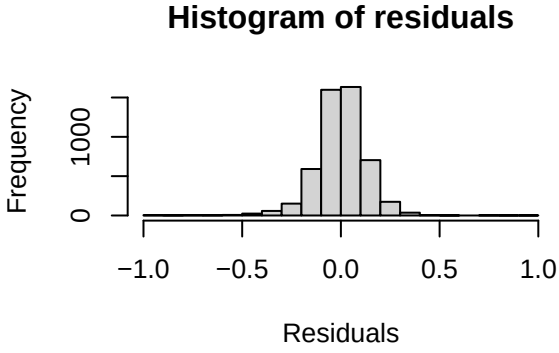
```
##     GCV.Cp
## 0.01937801
```

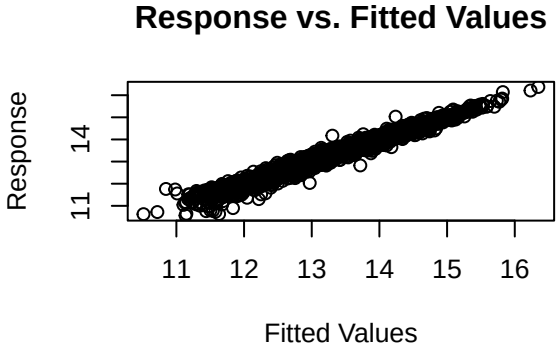
Further the `k'` values are quite close to the edf for `bedrm`, `saledate`, `longitude`, `latitude` and `gba` and so these are increased as well. This helps further reduce the GCV.

```
gam5 <- gam(price ~ s(bathrm) + s(hf_bathrm, k=5) + heat + ac + s(rooms) + s(bedrm, k=13)
            + s(stories) + style + s(saledate, k=25) + s(gba,k=20) + grade + cndtn
            + extwall + roof + intwall + s(kitchens, k=4) + s(fireplaces) + s(landarea)
            + s(latitude, k=20) + s(longitude, k=20) + nbhd + ward + quadrant + ysb1
            + ysb2 + s(yr_since_rmdl, k=30) + s(yr_since_imprv), data=dtrain)
gam.check(gam5)
```










```
##
## Method: GCV   Optimizer: magic
```

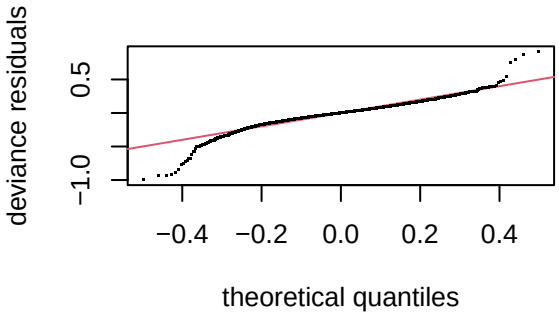
```
## Smoothing parameter selection converged after 22 iterations.
## The RMS GCV score gradient at convergence was 6.959549e-08 .
## The Hessian was positive definite.
## Model rank =  323 / 325
##
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##                       k'   edf k-index p-value
## s(bathrm)           9.00  4.08    1.01    0.69
## s(hf_bathrm)        4.00  1.88    1.03    0.96
## s(rooms)            9.00  7.77    1.01    0.69
## s(bedrm)           12.00 10.50    0.98    0.14
## s(stories)          9.00  1.00    1.01    0.81
## s(saledate)        24.00 20.61    1.03    0.99
## s(gba)             19.00 16.51    1.01    0.68
## s(kitchens)         3.00  1.00    0.99    0.30
## s(fireplaces)       9.00  5.04    0.99    0.20
## s(landarea)         9.00  7.35    1.01    0.65
## s(latitude)        19.00 13.94    1.02    0.90
## s(longitude)       19.00 12.99    1.02    0.89
## s(yr_since_rmdl)   29.00 24.24    0.99    0.19
## s(yr_since_imprv)   9.00  6.65    0.98    0.11
```

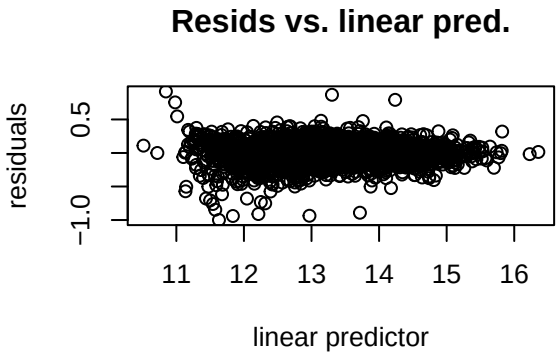
```
# Check the GCV
gam5$gcv.ubre
```

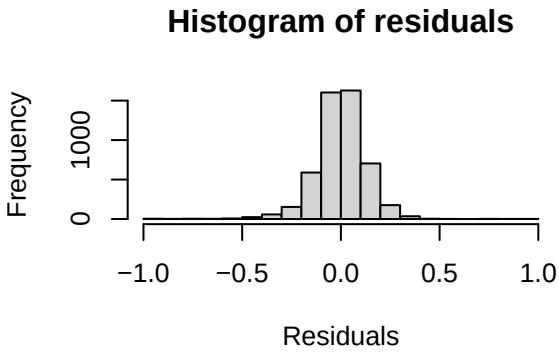
```
##     GCV.Cp
## 0.01895869
```

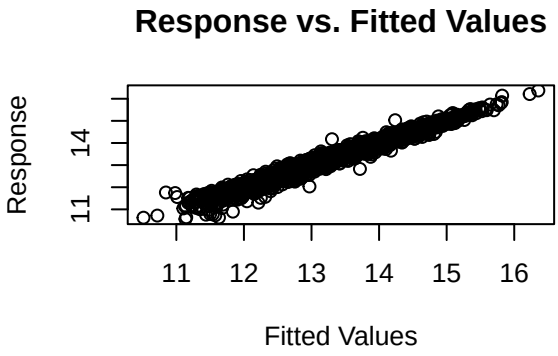
Next, we can see that the edf for kitchens and stories is 1, indicating a linear fit. Hence, these are added as linear terms:

```
gam6 <- gam(price ~ s(bathrm) + s(hf_bathrm, k=5) + heat + ac + s(rooms) + s(bedrm, k=13)
            + stories + style + s(saledate, k=25) + s(gba,k=20) + grade + cndtn + extwall
            + roof + intwall + kitchens + s(fireplaces) + s(landarea) + s(latitude, k=20)
            + s(longitude, k=20) + nbhd + ward + quadrant + ysb1 + ysb2
            + s(yr_since_rmdl, k=30) + s(yr_since_imprv), data=dtrain)
gam.check(gam6)
```










```
##
## Method: GCV   Optimizer: magic
## Smoothing parameter selection converged after 25 iterations.
## The RMS GCV score gradient at convergence was 4.412369e-07 .
## The Hessian was not positive definite.
## Model rank =  313 / 315
##
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##                       k'   edf k-index p-value
## s(bathrm)           9.00  4.17    1.01    0.71
## s(hf_bathrm)        4.00  1.87    1.03    0.95
## s(rooms)            9.00  8.23    1.01    0.76
## s(bedrm)           12.00 10.45    0.98    0.12
## s(saledate)        24.00 20.69    1.03    0.98
## s(gba)             19.00 16.47    1.01    0.72
## s(fireplaces)       9.00  5.03    0.99    0.17
## s(landarea)         9.00  7.39    1.01    0.66
## s(latitude)        19.00 14.15    1.02    0.93
## s(longitude)       19.00 13.00    1.02    0.86
## s(yr_since_rmdl)   29.00 24.76    0.99    0.26
## s(yr_since_imprv)   9.00  6.67    0.98    0.14
```

```
summary(gam6)
```

```
##
## Family: gaussian
## Link function: identity
##
```

```
## Formula:
## price ~ s(bathrm) + s(hf_bathrm, k = 5) + heat + ac + s(rooms) +
##     s(bedrm, k = 13) + stories + style + s(saledate, k = 25) +
##     s(gba, k = 20) + grade + cndtn + extwall + roof + intwall +
##     kitchens + s(fireplaces) + s(landarea) + s(latitude, k = 20) +
##     s(longitude, k = 20) + nbhd + ward + quadrant + ysb1 + ysb2 +
##     s(yr_since_rmdl, k = 30) + s(yr_since_imprv)
##
## Parametric coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)           11.9601673  0.2398358  49.868  < 2e-16 ***
## heatAir-Oil            0.1087280  0.1796469   0.605 0.545054
## heatElec Base Brd      0.2204935  0.1645104   1.340 0.180212
## heatEvp Cool          -0.2498675  0.2047408  -1.220 0.222371
## heatForced Air         0.2787014  0.1510608   1.845 0.065106 .
## heatGravity Furnac     0.3890361  0.1682222   2.313 0.020786 *
## heatHot Water Rad      0.2720208  0.1510277   1.801 0.071746 .
## heatHt Pump            0.2990655  0.1517920   1.970 0.048870 *
## heatNo Data            0.2285711  0.2031429   1.125 0.260573
## heatWall Furnace       0.3284118  0.1732916   1.895 0.058135 .
## heatWarm Cool          0.2732020  0.1510424   1.809 0.070549 .
## heatWater Base Brd     0.2824384  0.1611074   1.753 0.079649 .
## acY                    0.0466621  0.0074246   6.285 3.58e-10 ***
## stories               -0.0158736  0.0149006  -1.065 0.286797
## style1.5 Story Fin     0.0179719  0.0124871   1.439 0.150151
## style1.5 Story Unfin  -0.0502073  0.0389998  -1.287 0.198026
## style2 Story           0.0042278  0.0161874   0.261 0.793965
## style2.5 Story Fin     0.0360727  0.0226256   1.594 0.110929
## style2.5 Story Unfin   0.0370040  0.0254445   1.454 0.145928
## style3 Story           0.0150499  0.0305944   0.492 0.622800
## style4 Story          -0.0331862  0.0875460  -0.379 0.704652
## styleBi-Level          0.0714502  0.1364190   0.524 0.600473
## styleSplit Foyer       0.0680697  0.0247644   2.749 0.006006 **
## styleSplit Level       0.0145973  0.0204632   0.713 0.475670
## gradeAverage          -0.0490535  0.0085986  -5.705 1.24e-08 ***
## gradeExcellent         0.1440378  0.0139368  10.335  < 2e-16 ***
## gradeExceptional-A     0.3095193  0.0246629  12.550  < 2e-16 ***
## gradeExceptional-B     0.4673452  0.0289610  16.137  < 2e-16 ***
## gradeExceptional-C     0.5843739  0.0820075   7.126 1.19e-12 ***
## gradeExceptional-D     0.7254083  0.0613460  11.825  < 2e-16 ***
## gradeFair Quality     -0.0329254  0.0414645  -0.794 0.427198
## gradeGood Quality      0.0205597  0.0089791   2.290 0.022081 *
## gradeLow Quality      -0.5182150  0.1655184  -3.131 0.001754 **
## gradeSuperior          0.2186924  0.0169752  12.883  < 2e-16 ***
## gradeVery Good         0.0563082  0.0106511   5.287 1.30e-07 ***
## cndtnExcellent         0.3043214  0.0211783  14.370  < 2e-16 ***
## cndtnFair             -0.1162971  0.0315201  -3.690 0.000227 ***
## cndtnGood              0.0797154  0.0051351  15.523  < 2e-16 ***
## cndtnPoor             -0.2012433  0.0700422  -2.873 0.004082 **
## cndtnVery Good         0.1994303  0.0082964  24.038  < 2e-16 ***
## extwallBrick Veneer   -0.0084199  0.0248933  -0.338 0.735197
## extwallBrick/Siding   -0.0187311  0.0172503  -1.086 0.277603
## extwallBrick/Stone    -0.0136106  0.0247282  -0.550 0.582067
## extwallBrick/Stucco   -0.0222973  0.0225611  -0.988 0.323052
```

```
## extwallCommon Brick   -0.0008569  0.0162359  -0.053 0.957913
## extwallConcrete        0.0234020  0.0627828   0.373 0.709355
## extwallConcrete Block -0.0684510  0.0711174  -0.963 0.335844
## extwallDefault        -0.0202727  0.0973114  -0.208 0.834982
## extwallFace Brick      0.0147513  0.0511972   0.288 0.773262
## extwallHardboard      -0.0264365  0.0424510  -0.623 0.533477
## extwallMetal Siding    0.0696274  0.0701899   0.992 0.321255
## extwallShingle        -0.0345771  0.0193995  -1.782 0.074752 .
## extwallStone          -0.0146909  0.0235511  -0.624 0.532795
## extwallStone Veneer   -0.0281046  0.0440075  -0.639 0.523094
## extwallStone/Siding   -0.0053056  0.0247756  -0.214 0.830442
## extwallStone/Stucco   -0.0140950  0.0312313  -0.451 0.651787
## extwallStucco          0.0180347  0.0175065   1.030 0.302984
## extwallStucco Block   -0.2049196  0.1379175  -1.486 0.137395
## extwallVinyl Siding   -0.0123437  0.0166744  -0.740 0.459169
## extwallWood Siding     0.0034151  0.0168809   0.202 0.839686
## roofClay Tile          0.0637143  0.0210483   3.027 0.002483 **
## roofComp Shingle       0.0166113  0.0112279   1.479 0.139082
## roofComposition Ro     0.0928298  0.1390783   0.667 0.504508
## roofConcrete Tile      0.0315729  0.1317222   0.240 0.810579
## roofMetal- Cpr         0.2090613  0.0876927   2.384 0.017164 *
## roofMetal- Pre         0.0464950  0.1371774   0.339 0.734669
## roofMetal- Sms        -0.0036144  0.0187444  -0.193 0.847103
## roofNeopren            0.1259232  0.0749479   1.680 0.092996 .
## roofShake             -0.0047479  0.0181397  -0.262 0.793532
## roofShingle            0.0258394  0.0219748   1.176 0.239708
## roofSlate              0.0317115  0.0118479   2.677 0.007464 **
## roofTypical            0.0208671  0.0619697   0.337 0.736334
## intwallCeramic Tile   -0.0179508  0.0946519  -0.190 0.849591
## intwallDefault         0.0281959  0.1113889   0.253 0.800179
## intwallHardwood        0.0490861  0.0139927   3.508 0.000456 ***
## intwallHardwood/Carp   0.0270414  0.0146847   1.841 0.065615 .
## intwallLt Concrete     0.2571115  0.0712874   3.607 0.000313 ***
## intwallParquet         0.0296918  0.1358689   0.219 0.827023
## intwallWood Floor      0.0142876  0.0177247   0.806 0.420234
## kitchens               0.0154799  0.0101446   1.526 0.127097
## nbhdA2                 0.1421805  0.0512071   2.777 0.005515 **
## nbhdA3                -0.1329369  0.0461790  -2.879 0.004011 **
## nbhdA4                -0.3711537  0.0540971  -6.861 7.73e-12 ***
## nbhdA5                 0.2207962  0.0579521   3.810 0.000141 ***
## nbhdA6                -0.1388838  0.0979389  -1.418 0.156238
## nbhdA7                -0.0881878  0.0245975  -3.585 0.000340 ***
## nbhdA8                -0.0769369  0.0790539  -0.973 0.330493
## nbhdA9                -0.0270558  0.1861676  -0.145 0.884456
## nbhdB1                 0.2638032  0.1958434   1.347 0.178041
## nbhdB2                 0.1481559  0.0479391   3.091 0.002010 **
## nbhdB3                -0.1577921  0.0371908  -4.243 2.25e-05 ***
## nbhdB4                 0.3904443  0.0524750   7.441 1.18e-13 ***
## nbhdB5                 0.0656165  0.0384101   1.708 0.087644 .
## nbhdB6                -0.3082981  0.1957276  -1.575 0.115291
## nbhdB7                -0.1252911  0.0516377  -2.426 0.015289 *
## nbhdB8                 0.0920409  0.0255327   3.605 0.000316 ***
## nbhdB9                -0.1105556  0.0410243  -2.695 0.007066 **
## nbhdC1                -0.1531931  0.1024046  -1.496 0.134731
```

```
## nbhdC2                0.1422359  0.2356771   0.604 0.546192
## nbhdC3                0.1815844  0.0497625   3.649 0.000266 ***
## nbhdC4               -0.1353288  0.0405182  -3.340 0.000844 ***
## nbhdC5               -0.2575913  0.1080079  -2.385 0.017122 *
## nbhdC6                0.2396313  0.0742206   3.229 0.001252 **
## nbhdC7                0.4046890  0.0864239   4.683 2.91e-06 ***
## nbhdC8                0.2247034  0.1760048   1.277 0.201775
## nbhdC9                0.2847578  0.1093970   2.603 0.009271 **
## nbhdD1                0.0488107  0.0522157   0.935 0.349945
## nbhdD2               -0.0138694  0.0482677  -0.287 0.773862
## nbhdD3                0.1484431  0.1723890   0.861 0.389230
## nbhdD4                0.1482748  0.0577270   2.569 0.010243 *
## nbhdD5                0.2880894  0.2050776   1.405 0.160152
## nbhdD6               -0.0477642  0.0434881  -1.098 0.272118
## nbhdD7               -0.2004532  0.0433521  -4.624 3.87e-06 ***
## nbhdD8                0.3443190  0.0585153   5.884 4.27e-09 ***
## nbhdD9               -0.1980101  0.0798804  -2.479 0.013216 *
## nbhdE1                0.0806432  0.1833031   0.440 0.659998
## nbhdE2                0.2404549  0.0509932   4.715 2.48e-06 ***
## nbhdE3                0.3333260  0.0568968   5.858 4.99e-09 ***
## nbhdE4               -0.2350543  0.1931627  -1.217 0.223714
## nbhdE5                0.1116075  0.1558574   0.716 0.473973
## nbhdE6                0.1900749  0.0589023   3.227 0.001260 **
## nbhdE7               -0.1266556  0.0344653  -3.675 0.000241 ***
## nbhdE8               -0.0288697  0.0450716  -0.641 0.521859
## nbhdE9               -0.1498705  0.0563206  -2.661 0.007817 **
## nbhdF1                0.0501539  0.0288652   1.738 0.082361 .
## nbhdF2                0.1775709  0.0550533   3.225 0.001266 **
## nbhdF3               -0.1108232  0.0488158  -2.270 0.023238 *
## nbhdF4                0.1738946  0.1616216   1.076 0.282010
## nbhdF5                0.1759797  0.0512755   3.432 0.000604 ***
## nbhdF6                0.2793932  0.0567784   4.921 8.91e-07 ***
## nbhdF7                0.3616658  0.0580387   6.231 5.02e-10 ***
## nbhdF8               -0.2349271  0.0811708  -2.894 0.003818 **
## wardWard 2            0.3627939  0.0963680   3.765 0.000169 ***
## wardWard 3           -0.1186170  0.1762407  -0.673 0.500955
## wardWard 4           -0.1594585  0.1756610  -0.908 0.364050
## wardWard 5           -0.1635512  0.1627020  -1.005 0.314843
## wardWard 6            0.1649624  0.1320624   1.249 0.211681
## wardWard 7           -0.5079712  0.1601510  -3.172 0.001525 **
## wardWard 8           -0.6582514  0.1563709  -4.210 2.61e-05 ***
## quadrantNW            0.0852059  0.0324015   2.630 0.008574 **
## quadrantSE            0.0398919  0.0367402   1.086 0.277631
## quadrantSW            0.0184552  0.0720162   0.256 0.797757
## ysb1TRUE              0.9489049  0.0265311  35.766  < 2e-16 ***
## ysb2TRUE              0.1075328  0.0462268   2.326 0.020050 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##                      edf Ref.df        F  p-value
## s(bathrm)          4.167  4.953   40.921  < 2e-16 ***
## s(hf_bathrm)       1.875  2.194   40.991  < 2e-16 ***
## s(rooms)           8.225  8.705    3.088  0.00164 **
```

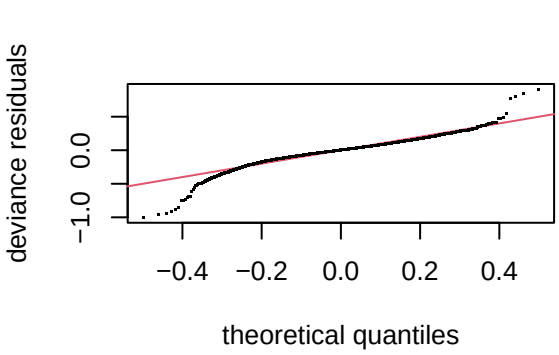
```
## s(bedrm)          10.454 11.259    4.362 1.28e-06 ***
## s(saledate)       20.689 22.923 1001.183  < 2e-16 ***
## s(gba)            16.472 18.138   40.374  < 2e-16 ***
## s(fireplaces)      5.031  5.801   12.759  < 2e-16 ***
## s(landarea)        7.394  8.365   59.298  < 2e-16 ***
## s(latitude)       14.152 16.723    5.481  < 2e-16 ***
## s(longitude)      12.997 15.798    3.903  < 2e-16 ***
## s(yr_since_rmdl)  24.764 27.550   16.667  < 2e-16 ***
## s(yr_since_imprv)  6.669  7.833   13.330  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Rank: 313/315
## R-sq.(adj) =  0.971   Deviance explained = 97.3%
## GCV = 0.01896  Scale est. = 0.017917  n = 4999
```

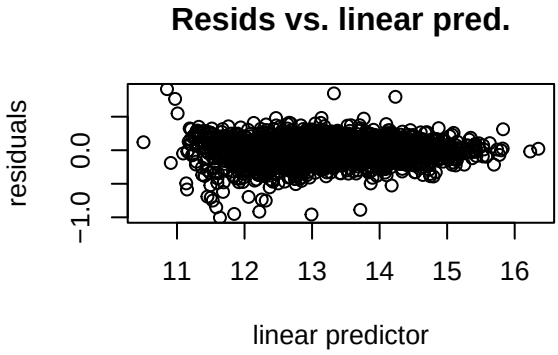
```
gam6$gcv.ubre
```

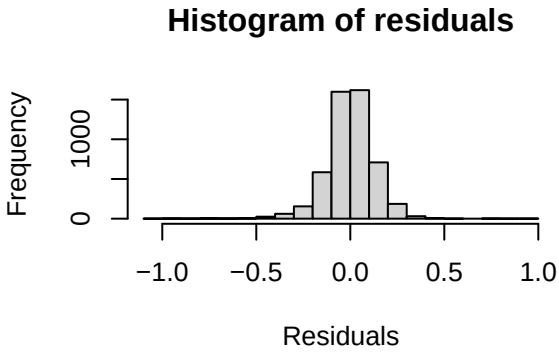
```
##     GCV.Cp
## 0.01895988
```

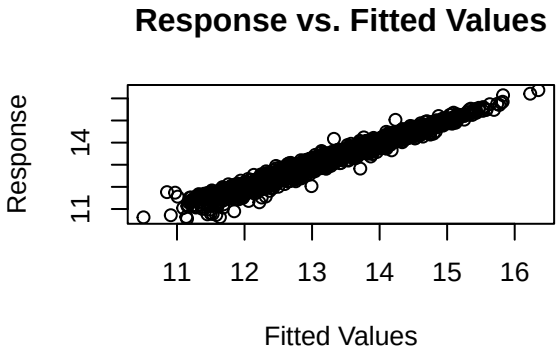
Next, we can see from the summary output that the factors for extwall are not very significant. So, we drop that from the model and notice a drop in the GCV.

```
gam7 <- gam(price ~ s(bathrm) + s(hf_bathrm, k=5) + heat + ac + s(rooms) + s(bedrm, k=13)
            + stories + style + s(saledate, k=25) + s(gba,k=20) + grade + cndtn + roof
            + intwall + kitchens + s(fireplaces) + s(landarea) + s(latitude, k=20)
            + s(longitude, k=20) + nbhd + ward + quadrant + ysb1 + ysb2
            + s(yr_since_rmdl, k=30) + s(yr_since_imprv), data=dtrain)

gam.check(gam7)
```









```
##
## Method: GCV   Optimizer: magic
## Smoothing parameter selection converged after 20 iterations.
## The RMS GCV score gradient at convergence was 1.560437e-08 .
## The Hessian was positive definite.
## Model rank =  293 / 295
##
## Basis dimension (k) checking results. Low p-value (k-index<1) may
## indicate that k is too low, especially if edf is close to k'.
##
##                       k'   edf k-index p-value
## s(bathrm)           9.00  3.94    1.01   0.725
## s(hf_bathrm)        4.00  1.87    1.03   0.975
## s(rooms)            9.00  7.82    1.01   0.755
## s(bedrm)           12.00 10.46    0.99   0.180
## s(saledate)        24.00 20.79    1.03   0.985
## s(gba)             19.00 16.42    1.01   0.670
## s(fireplaces)       9.00  5.13    0.99   0.240
## s(landarea)         9.00  7.07    1.01   0.730
## s(latitude)        19.00 13.77    1.02   0.920
## s(longitude)       19.00 13.51    1.02   0.855
## s(yr_since_rmdl)   29.00 24.27    0.99   0.145
## s(yr_since_imprv)   9.00  6.45    0.98   0.085 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
gam7$gcv.ubre
```

```
##     GCV.Cp
## 0.01891552
```

Finally, we will explore some interaction terms. The interaction between (saledate, longitude), (saledate, fireplaces) and (landarea, longitude) help decrease the GCV by the greatest amount. Hence, these are added to the model and we obtain our final model.

```
fit <- gam(price ~ ti(saledate, longitude) + ti(saledate, fireplaces)
           + ti(landarea, longitude) + s(bathrm) + s(hf_bathrm, k=5) + heat
           + ac + s(rooms) + s(bedrm, k=13) + stories + style + s(saledate, k=25)
           + s(gba,k=20) + grade + cndtn + roof + intwall + kitchens + s(fireplaces)
           + s(landarea) + s(latitude, k=20) + s(longitude, k=20) + nbhd + ward
           + quadrant + ysb1 + ysb2 + s(yr_since_rmdl, k=30)
           + s(yr_since_imprv), data=dtrain)
fit$gcv.ubre
```

```
##     GCV.Cp
## 0.01695481
```