ADA2: Class 14, Ch 07b, Analysis of Covariance

[Advanced Data Analysis 2](https://StatAcumen.com/teach/ada12, Stat 428/528, Spring 2023, Prof. Erik Erhardt, UNM

AUTHOR
sina mokhtar

PUBLISHED
March 9, 2023

This is a challenging dataset, in part because it's real and messy. I will guide you through a simplified sensible analysis, but other models are possible.

Note that I needed to set `cache=FALSE` to assure all output was updated.

ANCOVA model: Albuquerque NM 87108, House and Apartment listing prices

Prof Erhardt constructed a dataset of listing prices for dwellings (homes and apartments) for sale from Zillow.com on Feb 26, 2016 at 1 PM for Albuquerque NM 87108. In this assignment we'll develop a model to help understand which qualities that contribute to a **typical dwelling's listing price**. We will then also predict the listing prices of new listings posted on the following day, Feb 27, 2016 by 2 PM.

Because we want to model a *typical dwelling*, it is completely reasonable to remove "unusual" dwellings from the dataset. Dwellings have a distribution with a long tail!

Unusual assignment, not top-down, but up-down-up-down

This is an unusual assignment because the workflow of this assignment isn't top-down; instead, you'll be scrolling up and down as you make decisions about the data and model you're fitting. Yes, I have much of the code worked out for you. However, there are data decisions to make early in the code (such as excluding observations, transforming variables, etc.) that depend on the analysis (model checking) later. Think of it as a "choose your own adventure" that I've written for you.

Keep a record of your decisions

It is always desirable to make your work reproducible, either by someone else or by your future self. For each step you take, keep a diary of (a) what the next minor goal is, (b) what evidence/information you have, (c) what decision you make, and (d) what the outcome was.

For example, here's the first couple steps of your diary:

1. Include only "typical dwellings". Based on scatterplot, remove extreme observations. Keep only HOUSE and APARTMENT.
2. Exclude a few variables to reduce multicollinearity between predictor variables. Exclude `Baths` and `LotSize`.
3. etc.

(2 p) (Step 1) Restrict data to "typical" dwellings

Step 1: After looking at the scatterplot below, identify what you consider to be a "typical dwelling" and exclude observations far from that range. For example, there are only a couple `TypeSale` that are common enough to model; remember to run `factor()` again to remove factor levels that no longer appear.

```r
library(erikmisc)
```

```
── Attaching packages ─────────────────────────────────────── erikmisc 0.1.18 ──

✔ tibble 3.1.7      ✔ dplyr  1.0.10

── Conflicts ──────────────────────────────────────────── erikmisc_conflicts() ──
✖ dplyr::filter() masks stats::filter()
✖ dplyr::lag()    masks stats::lag()

erikmisc, solving common complex data analysis workflows
  by Dr. Erik Barry Erhardt <erik@StatAcumen.com>
```

```r
library(tidyverse)
```

```
── Attaching packages
───────────────────────────────────────

tidyverse 1.3.2 ──

✔ ggplot2 3.4.0      ✔ purrr   1.0.1
✔ tidyr   1.2.1      ✔ stringr 1.5.0
✔ readr   2.1.2      ✔ forcats 0.5.2
── Conflicts ──────────────────────────────────────────── tidyverse_conflicts() ──
✖ dplyr::filter() masks stats::filter()
✖ dplyr::lag()    masks stats::lag()
```

```r
library(ggplot2)
library(dplyr)
# First, download the data to your computer,
#   save in the same folder as this Rmd file.

# read the data, skip the first two comment lines of the data file
dat_abq <-
  read_csv("ADA2_CL_14_HomePricesZillow_Abq87108.csv", skip=2) %>%
  mutate(
    id = 1:n()
  , TypeSale = factor(TypeSale)
    # To help scale the intercept to a more reasonable value
    #   Scaling the x-variables are sometimes done to the mean of each x.
    # center year at 1900 (negative values are older, -10 is built in 1890)
  , YearBuilt_1900 = YearBuilt - 1900
  ) %>%
  select(
    id, everything()
    , -Address, -YearBuilt
  )
```

```
Rows: 143 Columns: 9
── Column specification ────────────────────────────────────────────────────────
Delimiter: ","
chr (2): Address, TypeSale
dbl (7): PriceList, Beds, Baths, Size_sqft, LotSize, YearBuilt, DaysListed

ℹ Use `spec()` to retrieve the full column specification for this data.
ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
head(dat_abq)
```

```
# A tibble: 6 × 9
     id TypeSale  PriceList  Beds Baths Size_sqft LotSize DaysListed
  <int> <fct>         <dbl> <dbl> <dbl>     <dbl>   <dbl>      <dbl>
1     1 HOUSE        186900     3     2      1305    6969          0
2     2 APARTMENT    305000     1     1      2523    6098          0
3     3 APARTMENT    244000     1     1      2816    6098          0
4     4 CONDO        108000     3     2      1137      NA          0
5     5 CONDO         64900     2     1      1000      NA          1
6     6 HOUSE        275000     3     3      2022    6098          1
# … with 1 more variable: YearBuilt_1900 <dbl>
```

```r
## RETURN HERE TO SUBSET THE DATA

dat_abq <-
  dat_abq %>%
  filter(
    TypeSale %in% c("APARTMENT" , "HOUSE")
    #, !id %in% c(120, 130, 50)# (X <= z)  # keep observations where variable X <= value z
  ) %>%
  mutate(
    TypeSale = factor(TypeSale)
  )
# note, if you remove a level from a categorical variable, then run factor() again

  # SOLUTION
  # these deletions are based only on the scatter plot in order to have
  #  "typical" dwellings




summary(dat_abq)
```

```
       id              TypeSale     PriceList            Beds
 Min.   :  1.00   APARTMENT:41   Min.   :  65000   Min.   :1.00
 1st Qu.: 39.75   HOUSE    :91   1st Qu.: 139250   1st Qu.:1.00
 Median : 72.50                  Median : 169250   Median :3.00
 Mean   : 73.32                  Mean   : 226403   Mean   :2.28
 3rd Qu.:108.25                  3rd Qu.: 249250   3rd Qu.:3.00
 Max.   :143.00                  Max.   :3110000   Max.   :5.00
```

```
    Baths          Size_sqft        LotSize          DaysListed
 Min.   :1.000   Min.   :  783   Min.   :  3049   Min.   :   0.0
 1st Qu.:1.000   1st Qu.: 1310   1st Qu.:  6534   1st Qu.:  33.5
 Median :1.000   Median : 1748   Median :  6969   Median :  88.0
 Mean   :1.542   Mean   : 2272   Mean   : 13571   Mean   : 122.4
 3rd Qu.:2.000   3rd Qu.: 2559   3rd Qu.:  8712   3rd Qu.: 174.0
 Max.   :5.000   Max.   :33000   Max.   :609840   Max.   :1867.0
 NA's   :1                       NA's   :18
 YearBuilt_1900
 Min.   : 30.00
 1st Qu.: 50.00
 Median : 52.00
 Mean   : 56.72
 3rd Qu.: 60.00
 Max.   :106.00
 NA's   :3
```

```
#filter(dat_abq, id == 21)
```

```
library(ggplot2)
library(GGally)
```

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

```
#ggpairs(dat_abq[,c("TypeSale", "PriceList", "Beds", "Baths", "Size_sqft")])
ggpairs(dat_abq %>% dplyr::select(everything(), -id, -Baths), mapping = ggplot2::aes(color=TypeSale,
```

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

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


```
dat_abq_reduced = dat_abq %>% dplyr::select(everything(), -Baths, -LotSize) %>% na.omit()

ggpairs(dat_abq_reduced %>% dplyr::select(everything(), -id), mapping = ggplot2::aes(color=TypeSale,
```

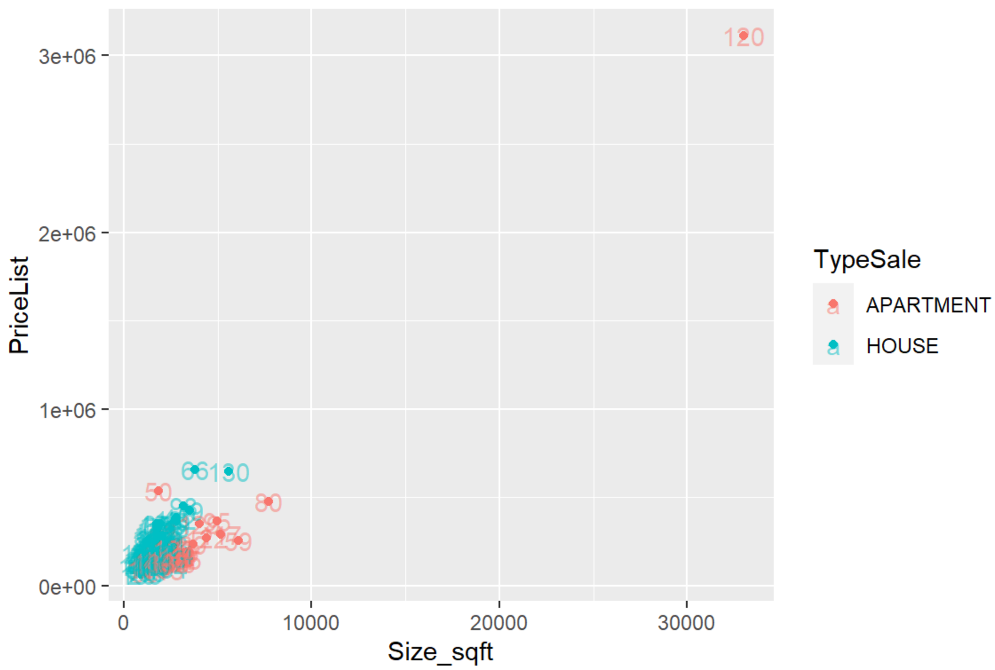
```
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`.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


```r
cor(dat_abq_reduced %>% dplyr::select(everything(), -id, -TypeSale))
```

```
               PriceList        Beds   Size_sqft  DaysListed YearBuilt_1900
PriceList     1.00000000  0.01050194  0.93729170  0.01589238     0.04649946
Beds          0.01050194  1.00000000 -0.17577972 -0.12561607    -0.21475246
Size_sqft     0.93729170 -0.17577972  1.00000000  0.04687242     0.16943291
DaysListed    0.01589238 -0.12561607  0.04687242  1.00000000     0.07070996
YearBuilt_1900 0.04649946 -0.21475246  0.16943291  0.07070996     1.00000000
```

```r
ggplot(data=dat_abq,
       aes(x=Size_sqft,
           y=PriceList,
```

```
          color=TypeSale,
          label=id))+
  geom_point() +
  geom_text(alpha = .5,
           nudge_x = 0.3)
```



(2 p) (Step 3) Transform response, if necessary.

Step 3: Does the response variable require a transformation? If so, what transformation is recommended from the model diagnostic plots (Box-Cox)?

Solution

Yes we need transformation based of COX-BOX plot (it contain zero) we do log transformation. [answer]

```
library(car)
```

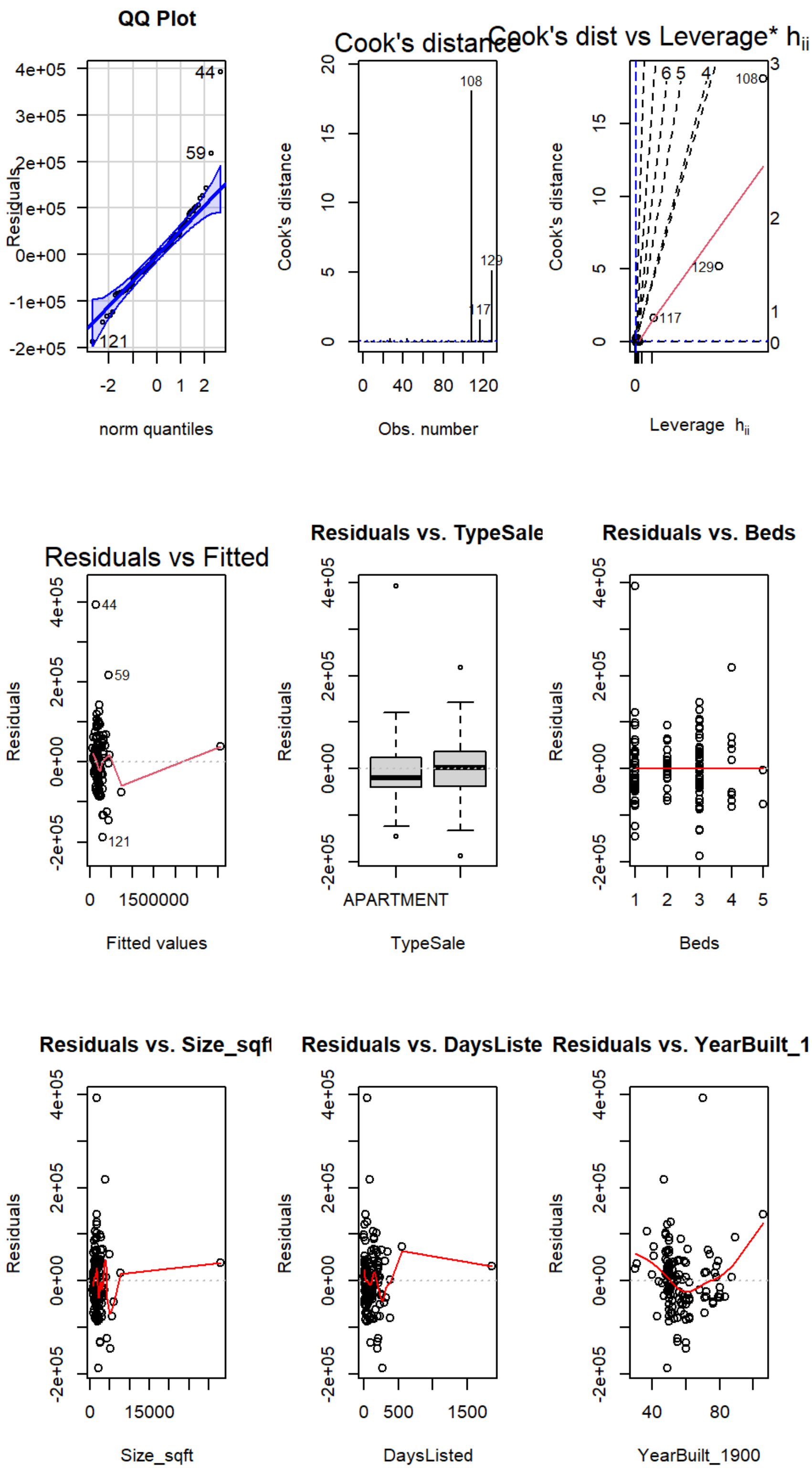
```
Loading required package: carData


Attaching package: 'car'

The following object is masked from 'package:purrr':

    some

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

    recode
```

```r
full.model.lm = lm(
  PriceList ~ (TypeSale + Beds + Size_sqft + DaysListed + YearBuilt_1900)^2,
  data = dat_abq_reduced)
summary(full.model.lm)
```

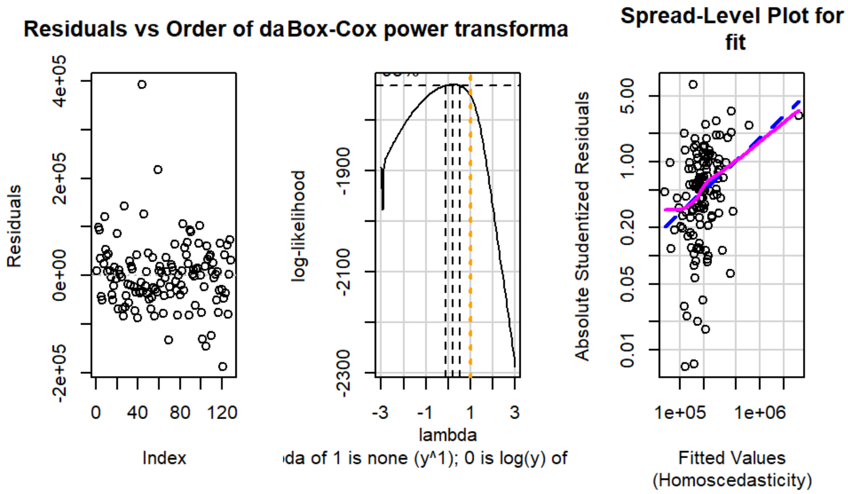
```

Call:
lm(formula = PriceList ~ (TypeSale + Beds + Size_sqft + DaysListed + 
    YearBuilt_1900)^2, data = dat_abq_reduced)

Residuals:
    Min      1Q  Median      3Q     Max 
-187238  -39119     521   33801  392702 

Coefficients: (1 not defined because of singularities)
                               Estimate Std. Error t value Pr(>|t|)  
(Intercept)                   1.932e+04  1.473e+05   0.131   0.8959  
TypeSaleHOUSE                 3.098e+05  1.970e+05   1.573   0.1185  
Beds                         -1.404e+05  8.320e+04  -1.688   0.0942 .
Size_sqft                     1.538e+02  7.229e+01   2.127   0.0356 *
DaysListed                   -3.851e+02  4.083e+02  -0.943   0.3476  
YearBuilt_1900                1.936e+02  2.276e+03   0.085   0.9324  
TypeSaleHOUSE:Beds                   NA         NA      NA       NA  
TypeSaleHOUSE:Size_sqft       1.714e+01  3.703e+01   0.463   0.6443  
TypeSaleHOUSE:DaysListed      3.850e+02  2.529e+02   1.522   0.1307  
TypeSaleHOUSE:YearBuilt_1900 -5.723e+03  3.110e+03  -1.841   0.0683 .
Beds:Size_sqft                2.946e+00  1.036e+01   0.284   0.7766  
Beds:DaysListed              -1.298e+02  1.056e+02  -1.230   0.2214  
Beds:YearBuilt_1900           2.817e+03  1.352e+03   2.084   0.0394 *
Size_sqft:DaysListed          2.118e-01  1.063e-01   1.992   0.0487 *
Size_sqft:YearBuilt_1900     -1.748e+00  9.800e-01  -1.784   0.0771 .
DaysListed:YearBuilt_1900    -5.985e-02  5.324e+00  -0.011   0.9911  
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 73770 on 114 degrees of freedom
Multiple R-squared:  0.9367,	Adjusted R-squared:  0.9289 
F-statistic: 120.5 on 14 and 114 DF,  p-value: < 2.2e-16
```

```r
#car::Anova(full.model.lm, type=3)
```

```r
e_plot_lm_diagostics(full.model.lm)
```




```
Non-constant Variance Score Test
Variance formula: ~ fitted.values
Chisquare = 0.1788738, Df = 1, p = 0.67234

there are higher-order terms (interactions) in this model
consider setting type = 'predictor'; see ?vif

Error in vif.default(fit): there are aliased coefficients in the model
```

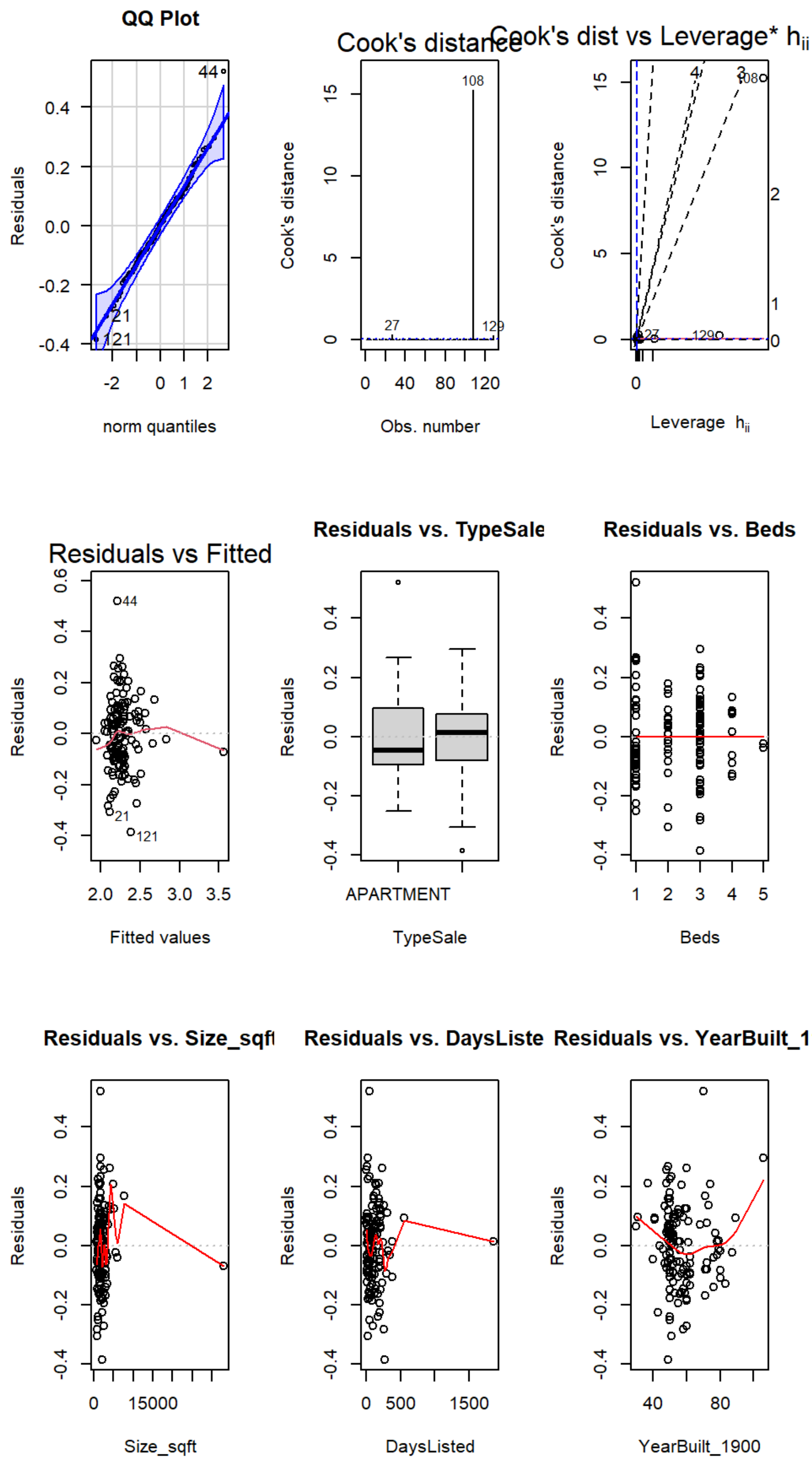



```
dat_abq_trans <-
  dat_abq_reduced %>%
  mutate(
    # Price in units of $1000
    PriceListK = (PriceList / 1000)

    # SOLUTION
  ) %>%
  select(
    -PriceList
  )
str(dat_abq_trans)
```

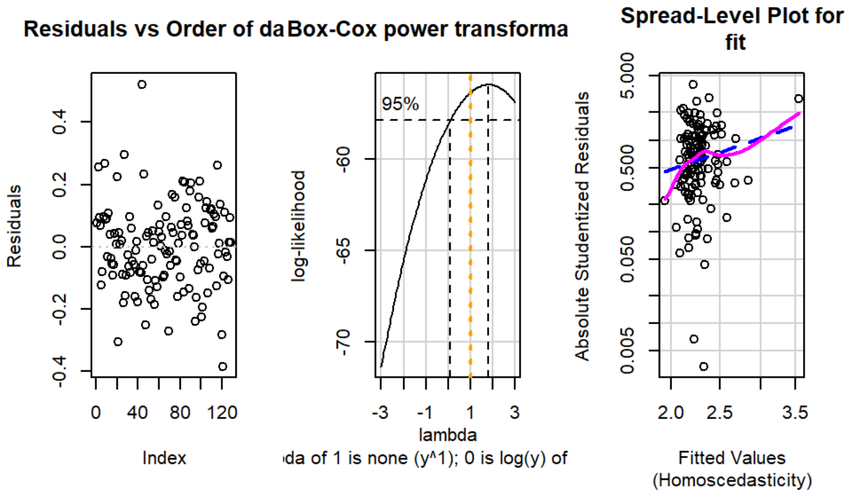
```
tibble [129 × 7] (S3: tbl_df/tbl/data.frame)
 $ id            : int [1:129] 1 2 3 6 7 9 10 12 13 14 ...
 $ TypeSale      : Factor w/ 2 levels "APARTMENT","HOUSE": 2 1 1 2 2 2 2 1 2 2 ...
 $ Beds          : num [1:129] 3 1 1 3 2 3 3 1 4 2 ...
 $ Size_sqft     : num [1:129] 1305 2523 2816 2022 1440 ...
 $ DaysListed    : num [1:129] 0 0 0 1 1 1 2 2 6 6 ...
 $ YearBuilt_1900: num [1:129] 54 48 89 52 52 58 52 49 41 53 ...
 $ PriceListK    : num [1:129] 187 305 244 275 133 ...
 - attr(*, "na.action")= 'omit' Named int [1:3] 57 83 98
  ..- attr(*, "names")= chr [1:3] "57" "83" "98"
```

```
full.model.log = lm(
  log10(PriceListK) ~ (TypeSale + Beds + Size_sqft + DaysListed + YearBuilt_1900)^2,
  data = dat_abq_trans)
```

```
e_plot_lm_diagostics(full.model.log)
```




```
Non-constant Variance Score Test
Variance formula: ~ fitted.values
Chisquare = 0.3182669, Df = 1, p = 0.57265

there are higher-order terms (interactions) in this model
consider setting type = 'predictor'; see ?vif

Error in vif.default(fit): there are aliased coefficients in the model
```



(2 p) (Step 4) Remove extremely influential observations.

Step 4: The goal is to develop a model that will work well for the typical dwellings. If an observation is highly influential, then it's unusual.

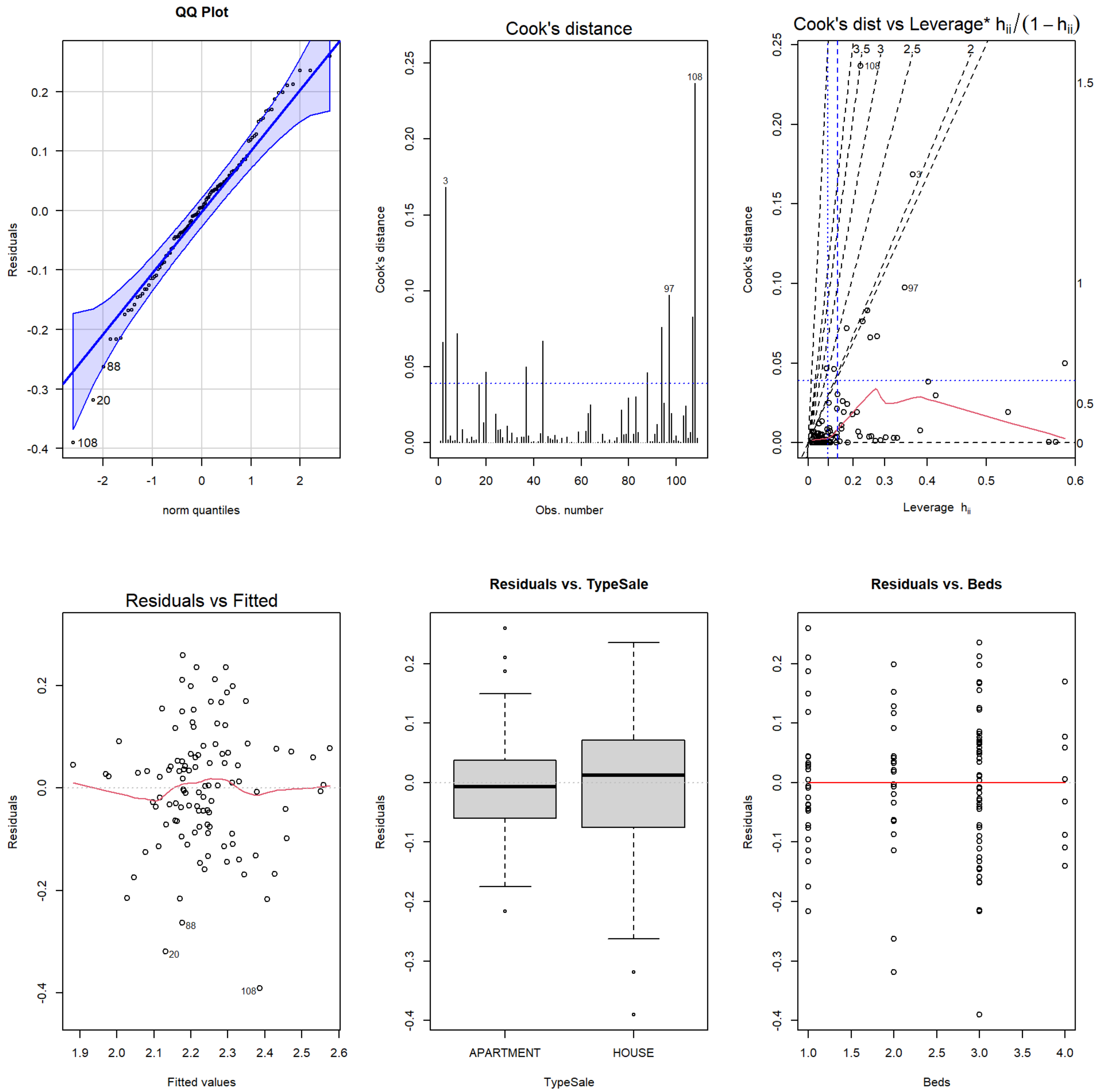
based on Cooks dist vs Leverage plot we noticed observations with id 132 are highly influential and also we remove PriceListK < 500, Size_sqft < 4000, Beds < 5, and DaysListed < 300 and YearBuilt_1900 < 100 to develop a model that will work well for the typical dwellings.
After transformation and removing influential observation it seems all assumptions are met

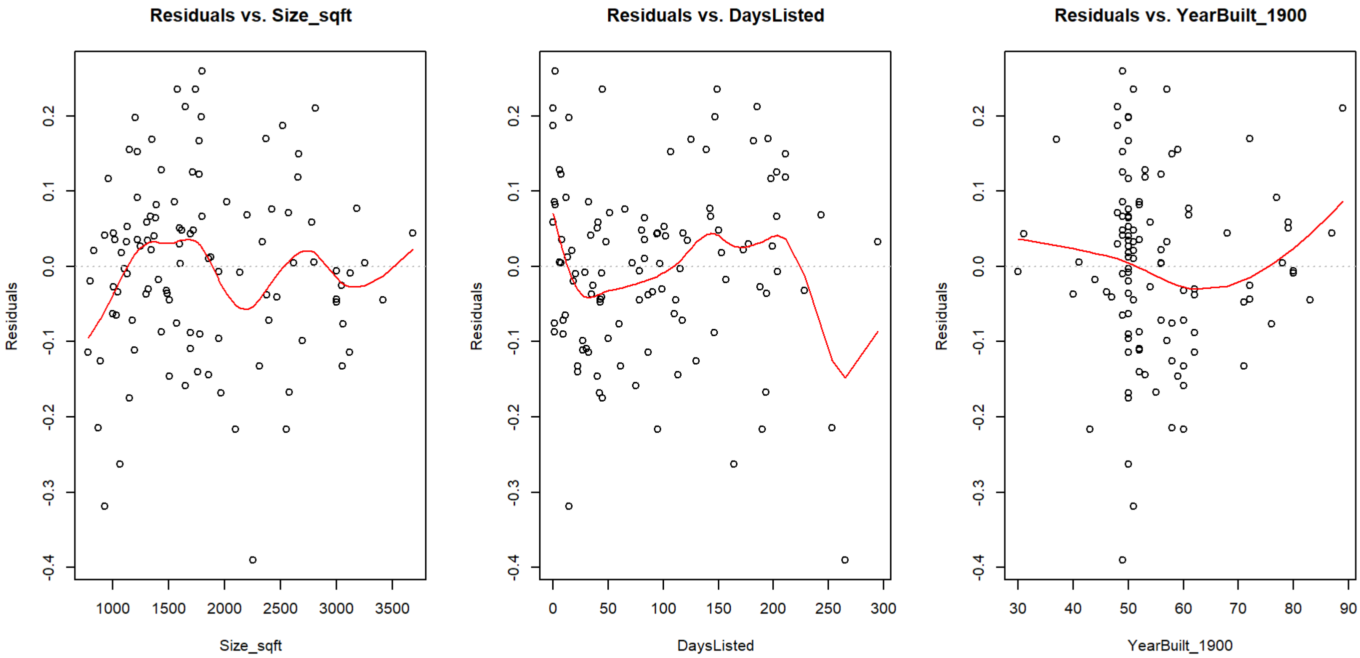
```
names(dat_abq_trans)
```

```
[1] "id"             "TypeSale"       "Beds"           "Size_sqft"
[5] "DaysListed"     "YearBuilt_1900" "PriceListK"
```

```
## Remove influential observation
#dat_abq_rem_influen[107,]
  dat_abq_rem_influen <-
    dat_abq_trans %>%
    dplyr::filter(
     # !(id  %in% c(120, 143, 130, 32, 50, 134, 140)),
      !(id  %in% c(132)),
      PriceListK < 500,
      Size_sqft < 4000,
      Beds < 5,
      DaysListed < 300,
      YearBuilt_1900 < 100
    )

  # SOLUTION
full.model.log = lm(
  log10(PriceListK) ~ (TypeSale + Beds + Size_sqft + DaysListed + YearBuilt_1900)^2,
  data = dat_abq_rem_influen)
```

```
e_plot_lm_diagostics(full.model.log)
```

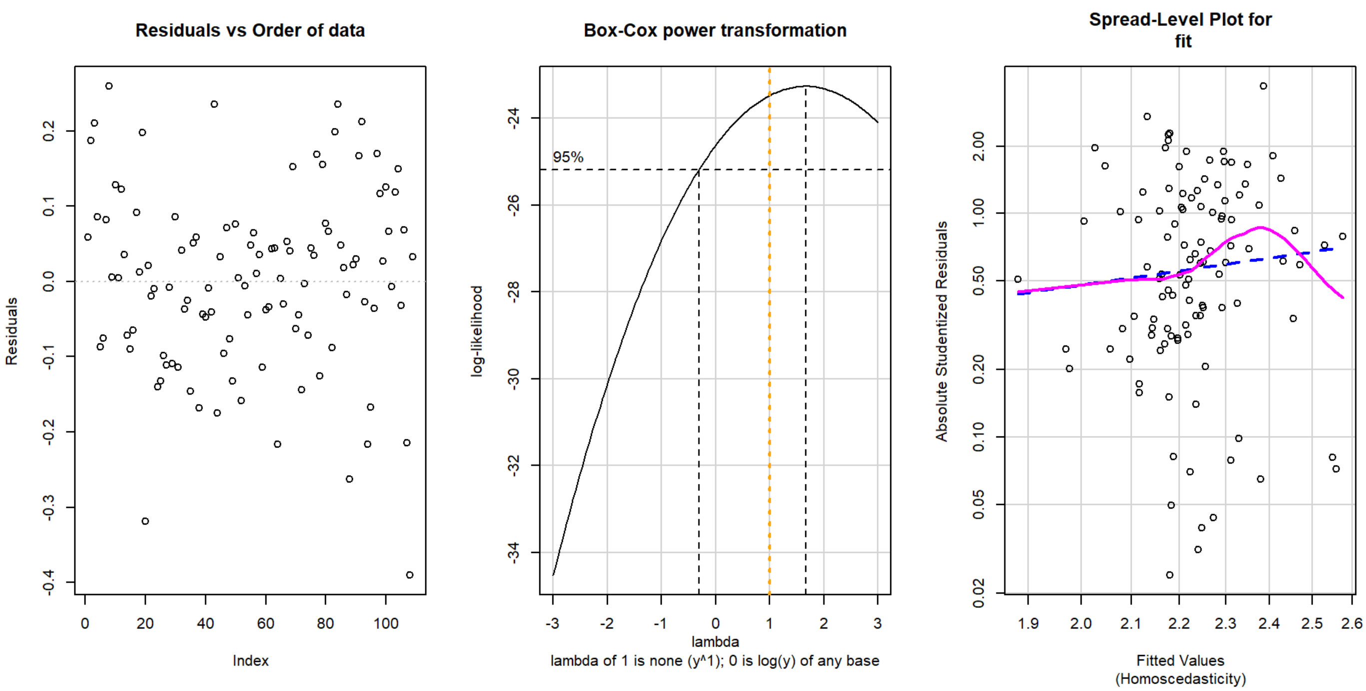





```
Non-constant Variance Score Test
Variance formula: ~ fitted.values
Chisquare = 0.4715461, Df = 1, p = 0.49228
```

```
there are higher-order terms (interactions) in this model
consider setting type = 'predictor'; see ?vif
```

```
Error in vif.default(fit): there are aliased coefficients in the model
```



Subset data for model building and prediction

Create a subset of the data for building the model, and another subset for prediction later on.

```
# remove observations with NAs
dat_abq_rem_influen <-
  dat_abq_rem_influen %>%
  na.omit()

# the data subset we will use to build our model
dat_sub <-
  dat_abq_rem_influen %>%
  filter(
    DaysListed > 0
  )

# the data subset we will predict from our model
dat_pred <-
  dat_abq_rem_influen %>%
  filter(
    DaysListed == 0
  ) %>%
  mutate(
    # the prices we hope to predict closely from our model
    PriceListK_true = PriceListK
    # set them to NA to predict them later
  , PriceListK = NA
  )
```

Scatterplot of the model-building subset.

```
# NOTE, this plot takes a long time if you're repeadly recompiling the document.
# comment the "print(p)" line so save some time when you're not evaluating this plot.
library(GGally)
library(ggplot2)
p <-
  ggpairs(
    dat_sub
  , mapping = ggplot2::aes(colour = TypeSale, alpha = 0.5)
  , lower = list(continuous = "points")
  , upper = list(continuous = "cor")
  , progress = FALSE
  )
print(p)
```

```
Warning in cor(x, y): the standard deviation is zero

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

Warning in cor(x, y): the standard deviation is zero

Warning in cor(x, y): the standard deviation is zero
```

```
Warning in cor(x, y): the standard deviation is zero

Warning in cor(x, y): the standard deviation is zero
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
`stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

There are clearly some unusual observations. Go back to the first code chunk and remove some observations that don't represent a "typical" dwelling.

For example, remove these dwellings (in code above):

- Super expensive dwelling
- Dwellings with huge lots

- Dwellings that were listed for years
- Because most dwellings were APARTMENTs and HOUSEs, remove the others (there are only 1 or so of each).

Discuss the observed correlations or other outstanding features in the data.

Solution

[answer]

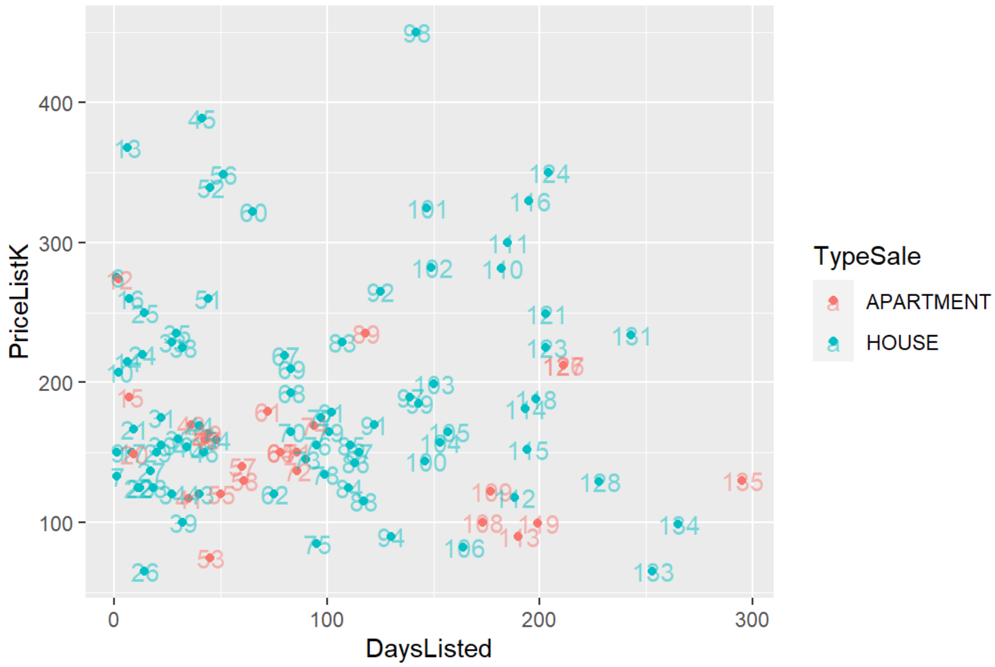
Features of data: 1. "TypeSale"
2. "Beds"
3. "Size_sqft"
4. "DaysListed"
5. "YearBuilt_1900"

There was high correletion between beds and baths and also ypeSale:Beds which cause coliniarity problem in or model. in addition we did transformation and removed influential observations.

```
names(dat_sub)
```

```
[1] "id"             "TypeSale"       "Beds"           "Size_sqft"
[5] "DaysListed"     "YearBuilt_1900" "PriceListK"
```

```
ggplot(data=dat_sub,
       aes(x=DaysListed,
           y=PriceListK,
           color=TypeSale,
           label=id))+
  geom_point() +
  geom_text(alpha = .5,
            nudge_x = 0.3)
```



(2 p) (Step 2) Fit full two-way interaction model.

You'll revisit this section after each modification of the data above.

Step 2: Let's fit the full two-way interaction model and assess the assumptions. However, some of the predictor variables are highly correlated. Recall that the interpretation of a beta coefficient is "the expected increase in the response for a 1-unit increase in x with all other predictors held constant". It's hard to hold one variable constant if it's correlated with another variable you're increasing. Therefore, we'll make a decision to retain some variables but not others depending on their correlation values. (In the PCA chapter, we'll see another strategy.)

Somewhat arbitrarily, let's exclude `Baths` (since highly correlated with `Beds` and `Size_sqft`). Let's also exclude `LotSize` (since highly correlated with `Size_sqft`). Modify the code below. Notice that because APARTMENTs don't have more than 1 Beds or Baths, those interaction terms need to be excluded from the model; I show you how to do this manually using the `update()` function.

Note that the formula below `y ~ (x1 + x2 + x3)^2` expands into all main effects and two-way interactions.

```
## SOLUTION
lm_full <-
  lm(
    log10(PriceListK) ~ (TypeSale + Beds + Size_sqft + DaysListed + YearBuilt_1900)^2
  , data = dat_sub
  )
#lm_full <-
#  lm(
#    PriceListK ~ (Beds + Baths + Size_sqft + LotSize + DaysListed + YearBuilt_1900)^2
#  , data = dat_sub
#  )
summary(lm_full)
```

```
Call:
lm(formula = log10(PriceListK) ~ (TypeSale + Beds + Size_sqft +
    DaysListed + YearBuilt_1900)^2, data = dat_sub)

Residuals:
     Min       1Q   Median       3Q      Max
-0.38166 -0.06724  0.01320  0.06718  0.29322

Coefficients: (1 not defined because of singularities)
                               Estimate Std. Error t value Pr(>|t|)
(Intercept)                   2.347e+00  3.609e-01   6.504 4.15e-09 ***
TypeSaleHOUSE                 3.714e-01  4.415e-01   0.841    0.402
Beds                         -1.697e-01  1.697e-01  -1.000    0.320
Size_sqft                     1.350e-04  1.777e-04   0.760    0.449
DaysListed                   -2.822e-04  1.418e-03  -0.199    0.843
YearBuilt_1900               -1.017e-02  7.562e-03  -1.345    0.182
TypeSaleHOUSE:Beds                   NA         NA      NA       NA
TypeSaleHOUSE:Size_sqft       4.803e-06  1.190e-04   0.040    0.968
TypeSaleHOUSE:DaysListed      9.868e-04  9.567e-04   1.032    0.305
TypeSaleHOUSE:YearBuilt_1900 -5.155e-03  7.955e-03  -0.648    0.519
Beds:Size_sqft                1.177e-05  4.637e-05   0.254    0.800
Beds:DaysListed              -4.293e-04  4.047e-04  -1.061    0.292
Beds:YearBuilt_1900           3.556e-03  2.890e-03   1.230    0.222
Size_sqft:DaysListed          1.390e-07  3.598e-07   0.386    0.700
Size_sqft:YearBuilt_1900      3.126e-07  2.926e-06   0.107    0.915
DaysListed:YearBuilt_1900     2.876e-06  2.527e-05   0.114    0.910
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1253 on 91 degrees of freedom
Multiple R-squared:  0.5245,    Adjusted R-squared:  0.4513
F-statistic: 7.168 on 14 and 91 DF,  p-value: 9.238e-10
```

```r
try(Anova(lm_full, type=3))
```

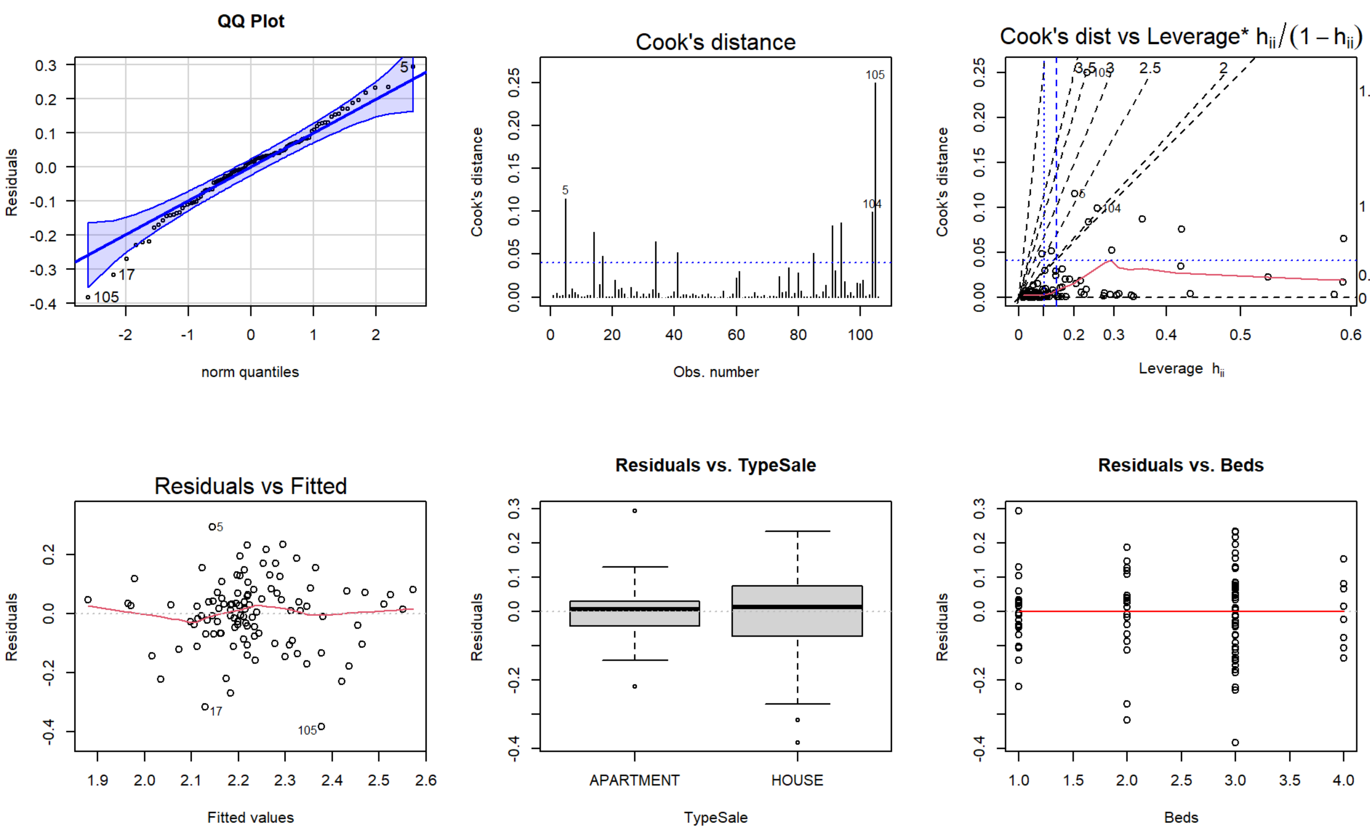
```
Error in Anova.III.lm(mod, error, singular.ok = singular.ok, ...) :
  there are aliased coefficients in the model
```

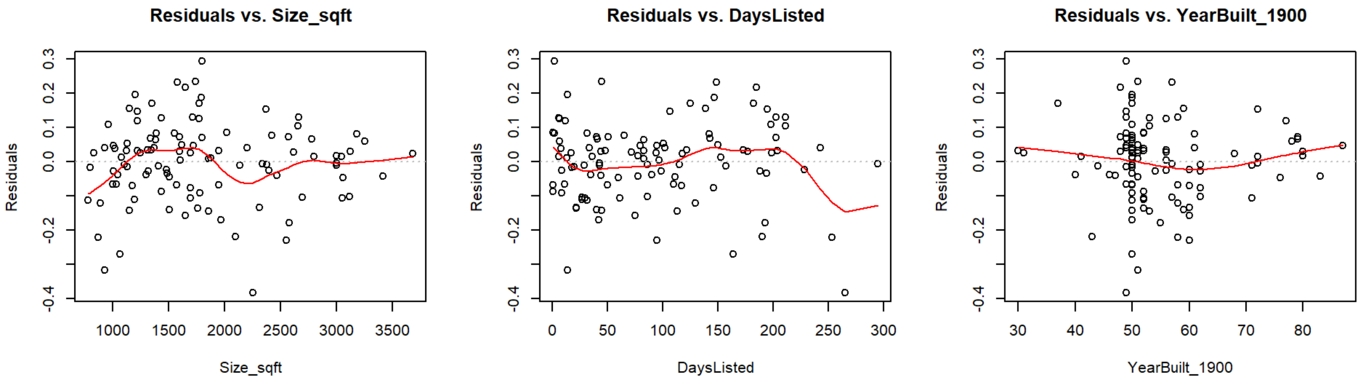
```r
## Note that this doesn't work because APARTMENTs only have 1 bed and 1 bath.
## There isn't a second level of bed or bath to estimate the interaction.
## Therefore, remove those two terms
lm_full <-
  update(
    lm_full
  , . ~ . - TypeSale:Beds
  )
try(Anova(lm_full, type=3))
```

```
Anova Table (Type III tests)
```

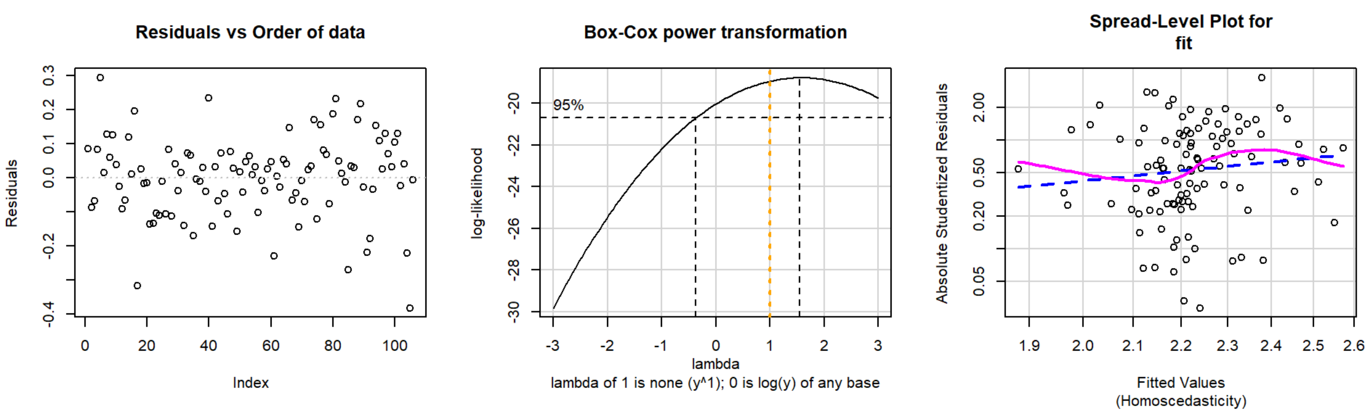
```
Response: log10(PriceListK)
                             Sum Sq Df F value    Pr(>F)    
(Intercept)                 0.66456  1 42.2963 4.151e-09 ***
TypeSale                    0.01112  1  0.7078    0.4024    
Beds                        0.01571  1  0.9997    0.3200    
Size_sqft                   0.00907  1  0.5775    0.4492    
DaysListed                  0.00062  1  0.0396    0.8428    
YearBuilt_1900              0.02842  1  1.8085    0.1820    
TypeSale:Size_sqft          0.00003  1  0.0016    0.9679    
TypeSale:DaysListed         0.01672  1  1.0640    0.3050    
TypeSale:YearBuilt_1900     0.00660  1  0.4200    0.5186    
Beds:Size_sqft              0.00101  1  0.0644    0.8003    
Beds:DaysListed             0.01768  1  1.1255    0.2915    
Beds:YearBuilt_1900         0.02379  1  1.5141    0.2217    
Size_sqft:DaysListed        0.00234  1  0.1492    0.7002    
Size_sqft:YearBuilt_1900    0.00018  1  0.0114    0.9152    
DaysListed:YearBuilt_1900   0.00020  1  0.0130    0.9096    
Residuals                   1.42980 91                      
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
## Uncomment this line when you're ready to assess the model assumptions
# plot diagnostics
e_plot_lm_diagostics(lm_full)
```

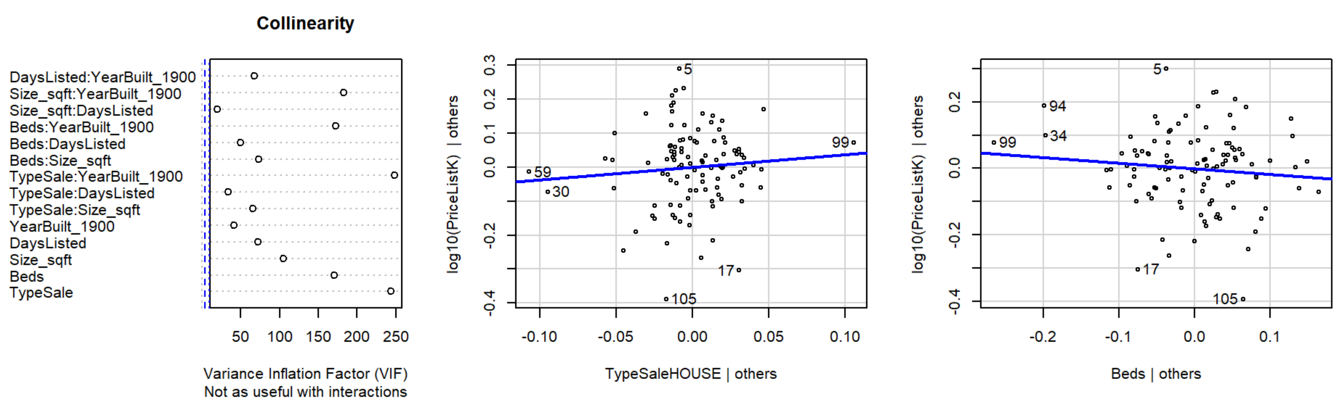


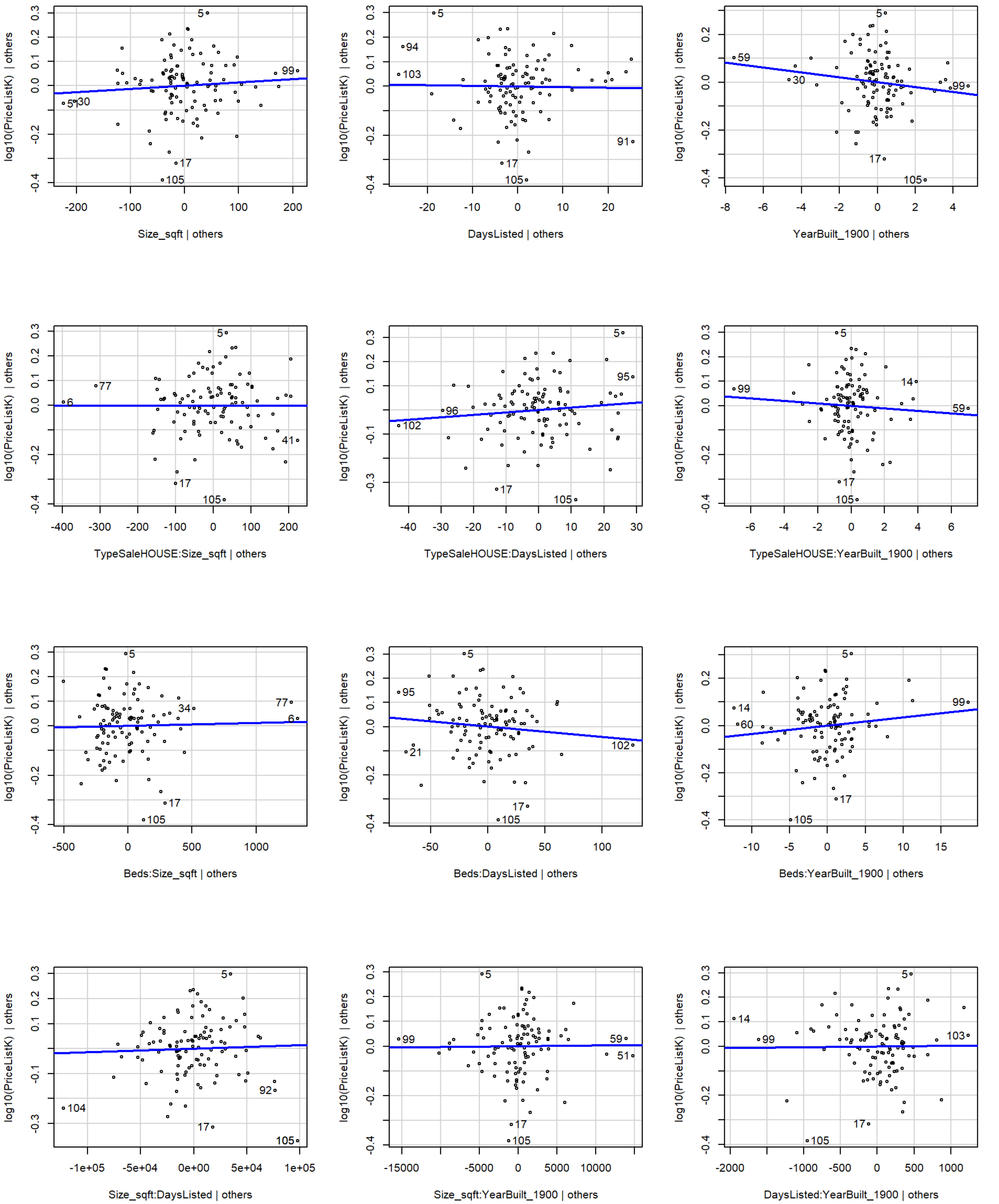



```
Non-constant Variance Score Test
Variance formula: ~ fitted.values
Chisquare = 0.6971503, Df = 1, p = 0.40374

there are higher-order terms (interactions) in this model
consider setting type = 'predictor'; see ?vif
```




```
Warning in e_plot_lm_diagostics(lm_full): Note: Collinearity plot unreliable for
predictors that also have interactions in the model.
```






```
# List the row numbers with id numbers
#   The row numbers appear in the residual plots.
```

```
#   The id number can be used to exclude values in code above.
dat_sub %>% select(id) %>% print(n = Inf)
```

```
# A tibble: 106 × 1
      id
   <int>
 1     6
 2     7
 3     9
 4    10
 5    12
 6    13
 7    14
 8    15
 9    16
10    17
11    20
12    21
13    22
14    23
15    24
16    25
17    26
18    27
19    28
20    29
21    30
22    31
23    33
24    34
25    35
26    36
27    38
28    39
29    40
30    41
31    42
32    43
33    44
34    45
35    46
36    47
37    48
38    49
39    51
40    52
41    53
42    54
43    55
44    56
45    57
```

```
46   58
47   60
48   61
49   62
50   64
51   65
52   67
53   68
54   69
55   70
56   71
57   72
58   73
59   74
60   75
61   76
62   77
63   78
64   79
65   81
66   83
67   84
68   85
69   86
70   87
71   88
72   89
73   91
74   92
75   94
76   97
77   98
78   99
79  100
80  101
81  102
82  103
83  104
84  105
85  106
86  108
87  109
88  110
89  111
90  112
91  113
92  114
93  115
94  116
95  118
96  119
97  121
```

```
 98   123
 99   124
100   126
101   127
102   128
103   131
104   133
105   134
106   135
```

```
shapiro.test(lm_full$residuals)
```

```

    Shapiro-Wilk normality test

data:  lm_full$residuals
W = 0.98268, p-value = 0.183
```

After Step 2, interpret the residual plots. What are the primary issues in the original model?

Solution

[answer] The Residual plot is roughly normal however the tail and head is skewed. by Shapiro test the residual p-value is greater than .05 which means we have not enough evidence that say residuals is not normal. so we met normality assumption. the residuals vs fitted value look acceptable. the plots for residuals vs other features are also acceptable

because there was high correlation between bath and beds and also lotsize we remove baths and lotsize from our model and also APARTMENTs don't have more than 1 Beds or Baths,so those interaction terms need to be excluded from the model. so we fixed the collinearity problem in the model.

(2 p) (Step 5) Model selection, check model assumptions.

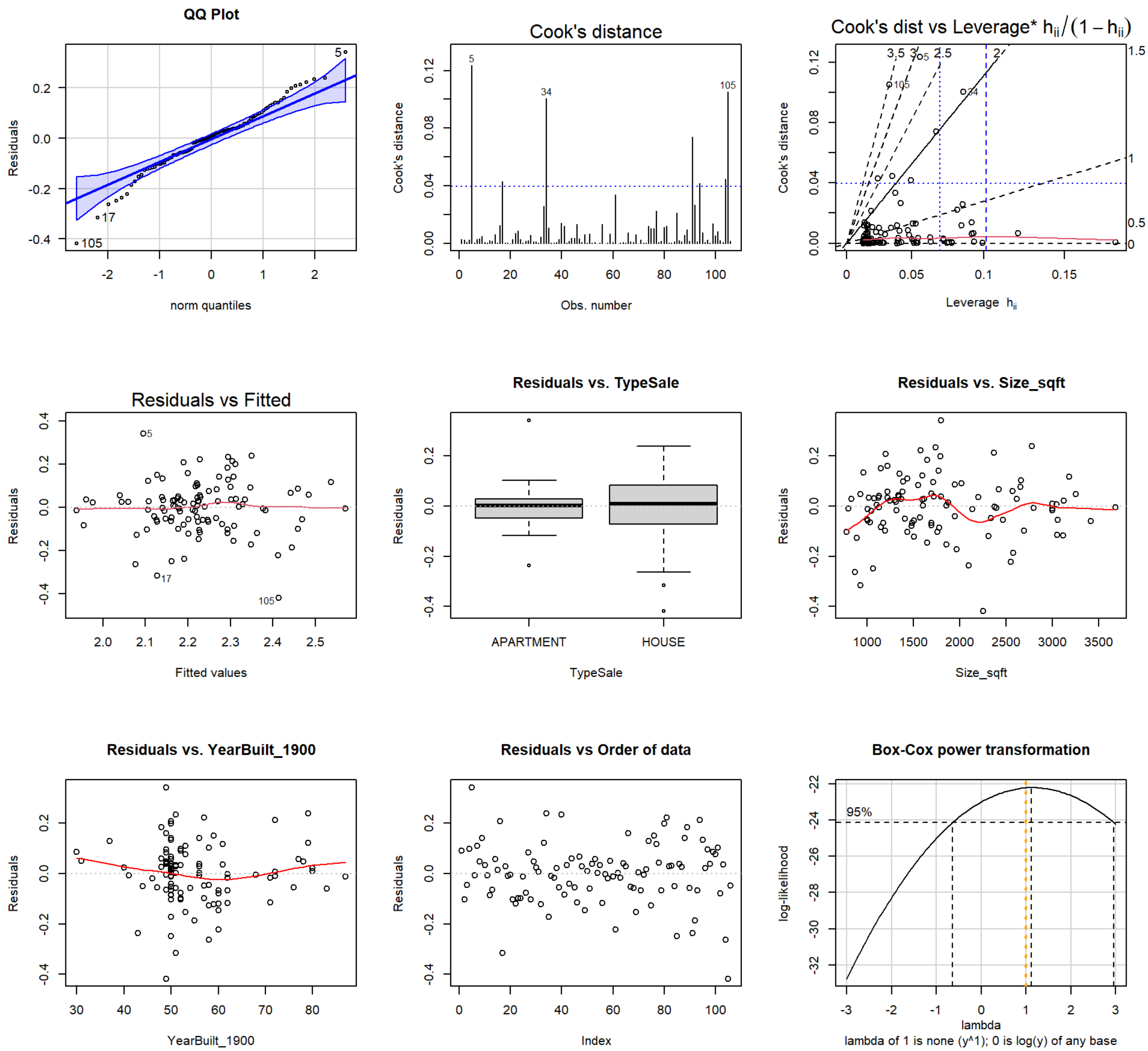
Using `step(..., direction="both")` with the BIC criterion, perform model selection.

Solution

```
## BIC
# option: test="F" includes additional information
#           for parameter estimate tests that we're familiar with
# option: for BIC, include k=log(nrow( [data.frame name] ))
lm_red_BIC <-
  step(
    lm_full
  , direction = "both"
  , test = "F"
  , trace = 0
  , k = log(nrow(dat_sub))
  )
```

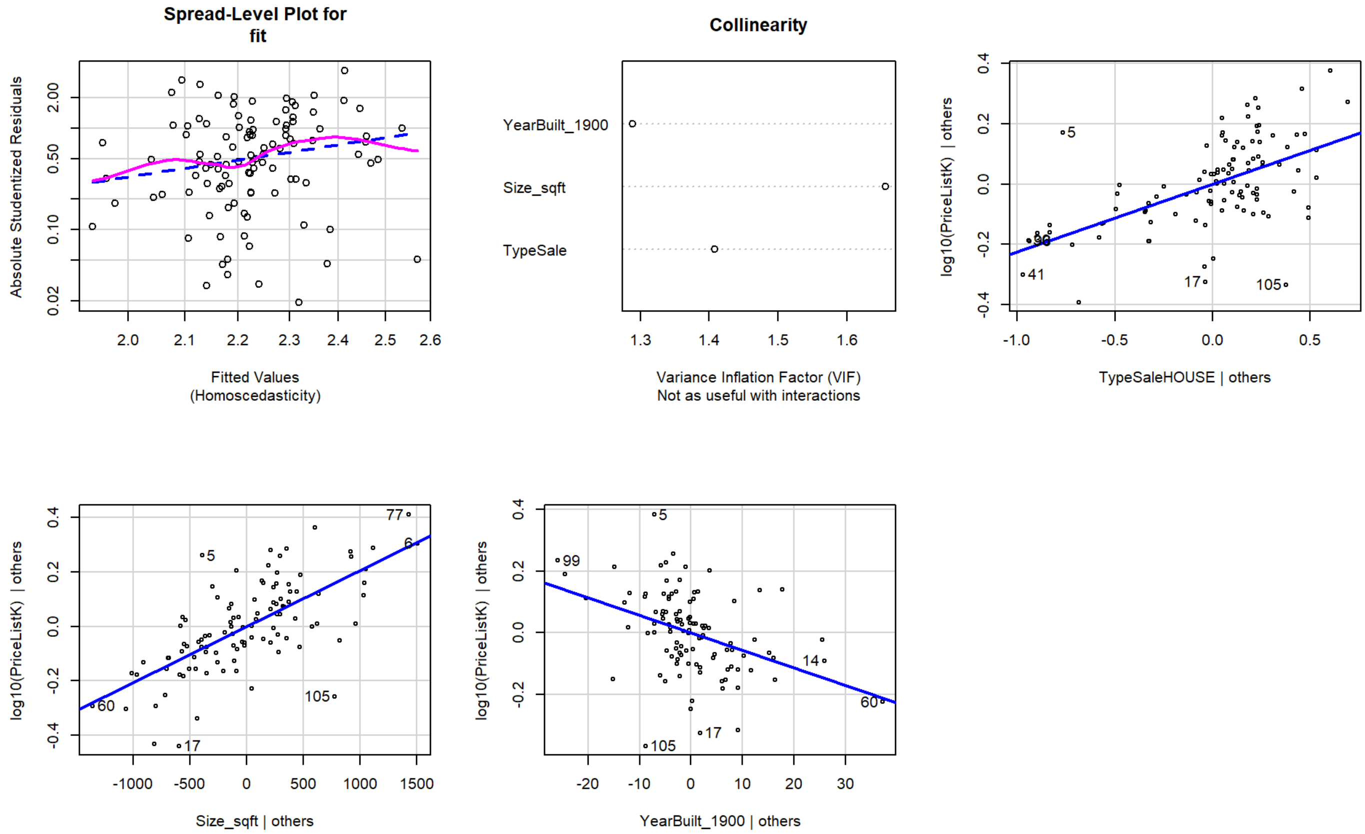
```
lm_final <- lm_red_BIC
lm.final = lm_red_BIC
```

```
## Uncomment this line when you're ready to assess the model assumptions
# plot diagnostics
e_plot_lm_diagostics(lm_final)
```




```
Non-constant Variance Score Test
Variance formula: ~ fitted.values
Chisquare = 1.789649, Df = 1, p = 0.18097
```

```
Warning in e_plot_lm_diagostics(lm_final): Note: Collinearity plot unreliable
for predictors that also have interactions in the model.
```



Model assumptions appear to be reasonably met. A few influential observations exist. The residuals are roughly distributed normal based on QQplot (there is a little bit skewness, but it is not that much severe). A few influential observations exist. The variances looks constant. based on box-cox plot we do not need transformation. residuals look acceptable

(4 p) (Step 6) Plot final model, interpret coefficients.

If you arrived at the same model I did, then the code below will plot it. Eventually (after Step 7), the fitted model equations will describe the each dwelling `TypeSale` and interpret the coefficients.

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


```
library(car)
Anova(lm.final, type=3)
```

```
Anova Table (Type III tests)

Response: log10(PriceListK)
               Sum Sq  Df F value    Pr(>F)
(Intercept)    9.4443   1 633.368 < 2.2e-16 ***
TypeSale       0.6950   1  46.611 6.363e-10 ***
Size_sqft      1.3438   1  90.118 1.079e-15 ***
YearBuilt_1900 0.2879   1  19.309 2.729e-05 ***
Residuals      1.5210 102
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(lm.final)
```

```
Call:
lm(formula = log10(PriceListK) ~ TypeSale + Size_sqft + YearBuilt_1900,
    data = dat_sub)

Residuals:
     Min       1Q   Median       3Q      Max
-0.41779 -0.06529  0.00684  0.05714  0.34219
```

```
Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)     2.004e+00  7.964e-02  25.167  < 2e-16 ***
TypeSaleHOUSE   2.233e-01  3.271e-02   6.827 6.36e-10 ***
Size_sqft       2.064e-04  2.174e-05   9.493 1.08e-15 ***
YearBuilt_1900 -5.718e-03  1.301e-03  -4.394 2.73e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1221 on 102 degrees of freedom
Multiple R-squared:  0.4941,    Adjusted R-squared:  0.4793
F-statistic: 33.21 on 3 and 102 DF,  p-value: 4.639e-15
```

Fitted model equation is

$$\log 10(\widehat{\text{PriceList}}) = 2 + 0.223 I(TypeSale = \text{HOUSE}) + 2.06 \times 10^{-4}(\text{Size.sqft}) + -0.00572(\text{YearBuilt})$$

Solution

After Step 7, return and interpret the model coefficients above.

[answer] the log10 of Price will increase if
on average, by one sqft increase in size we expect 2.06^{-4} increase in log(PriceListK) assuming other variables constant.
on average, by one year increase in yearBuild we expect -0.00572 increase in log(PriceListK) assuming other variables constant.
for Apartment, on average the log(PriceListK) would be 2 if all other variable would be zero(which is not usefull for interpretation).
for House, on average the log(PriceListK) would be (2 + 0.223) if all other variable would be zero(which is not usefull for interpretation).

(2 p) (Step 7) Transform predictors.

We now have enough information to see that a transformation of a predictor can be useful. See the curvature with `Size_sqft`? This is one of the headaches of regression modelling, *everything depends on everything else* and you learn as you go. Return to the top and transform `Size_sqft` and `LotSize`.

A nice feature of this transformation is that the model interaction goes away. Our interpretation is now on the log scale, but it’s a simpler model.

```
## SOLUTION
lm_full_logSize <-
  lm(
    log10(PriceListK) ~ (TypeSale + Beds + log10(Size_sqft) + DaysListed + YearBuilt_1900)^2
  , data = dat_sub
  )
#lm_full <-
#  lm(
#    PriceListK ~ (Beds + Baths + Size_sqft + LotSize + DaysListed + YearBuilt_1900)^2
#  , data = dat_sub
```

```
  #  )
  summary(lm_full_logSize)
```

```
Call:
lm(formula = log10(PriceListK) ~ (TypeSale + Beds + log10(Size_sqft) +
    DaysListed + YearBuilt_1900)^2, data = dat_sub)

Residuals:
     Min       1Q   Median       3Q      Max
-0.37568 -0.06478  0.01052  0.06831  0.27809

Coefficients: (1 not defined because of singularities)
                                  Estimate Std. Error t value Pr(>|t|)
(Intercept)                      1.086e+00  2.381e+00   0.456    0.649
TypeSaleHOUSE                    1.177e+00  1.421e+00   0.829    0.409
Beds                            -5.668e-01  5.710e-01  -0.993    0.323
log10(Size_sqft)                 4.733e-01  7.519e-01   0.629    0.531
DaysListed                      -1.487e-03  4.480e-03  -0.332    0.741
YearBuilt_1900                  -2.247e-02  4.348e-02  -0.517    0.607
TypeSaleHOUSE:Beds                      NA         NA      NA       NA
TypeSaleHOUSE:log10(Size_sqft)  -2.403e-01  4.396e-01  -0.546    0.586
TypeSaleHOUSE:DaysListed         9.654e-04  9.530e-04   1.013    0.314
TypeSaleHOUSE:YearBuilt_1900    -4.997e-03  8.291e-03  -0.603    0.548
Beds:log10(Size_sqft)            1.268e-01  1.759e-01   0.721    0.473
Beds:DaysListed                 -4.139e-04  4.056e-04  -1.020    0.310
Beds:YearBuilt_1900              3.439e-03  2.997e-03   1.147    0.254
log10(Size_sqft):DaysListed      2.995e-04  1.356e-03   0.221    0.826
log10(Size_sqft):YearBuilt_1900  3.890e-03  1.325e-02   0.294    0.770
DaysListed:YearBuilt_1900        1.184e-05  2.455e-05   0.482    0.631

Residual standard error: 0.1221 on 91 degrees of freedom
Multiple R-squared:  0.5485,	Adjusted R-squared:  0.479
F-statistic: 7.895 on 14 and 91 DF,  p-value: 1.132e-10
```

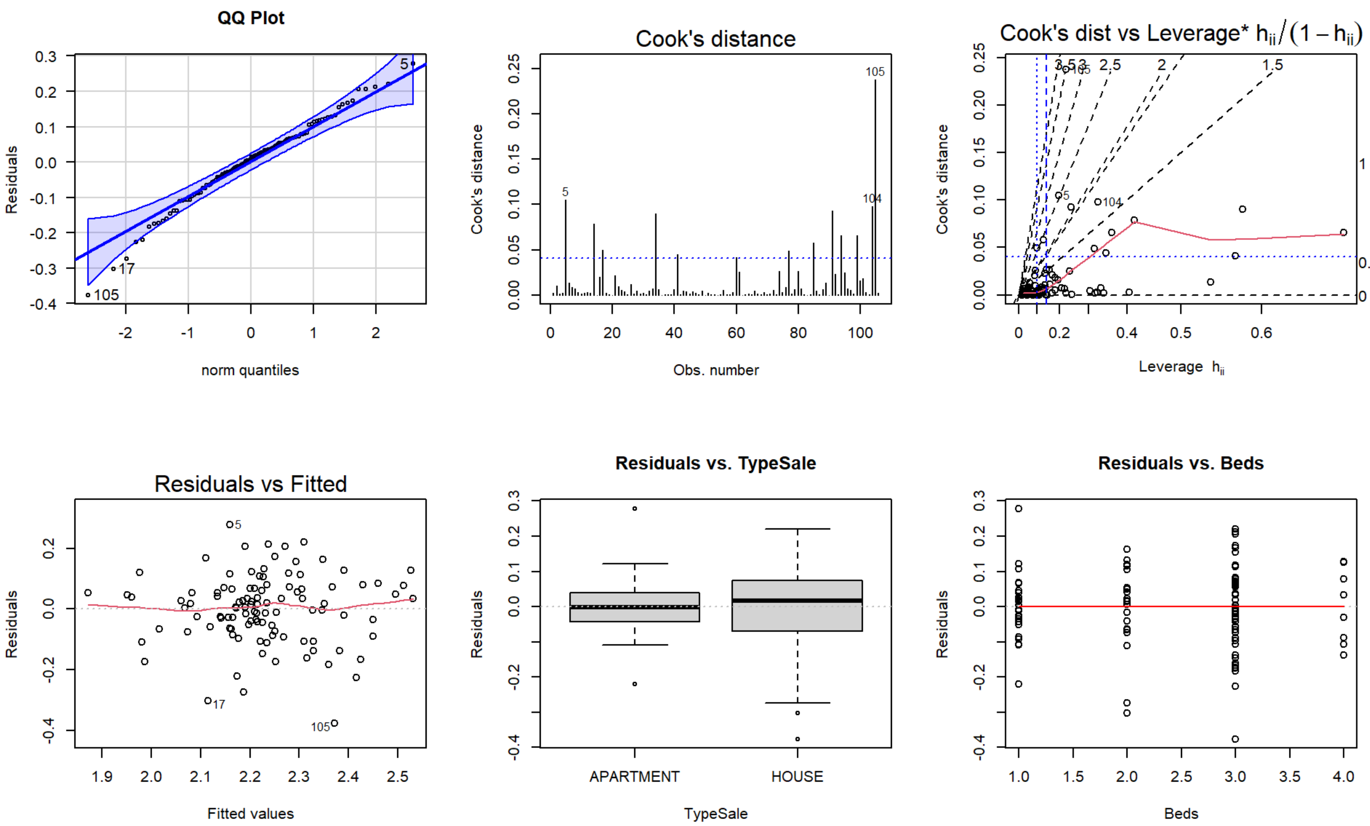
```
  try(Anova(lm_full_logSize, type=3))
```

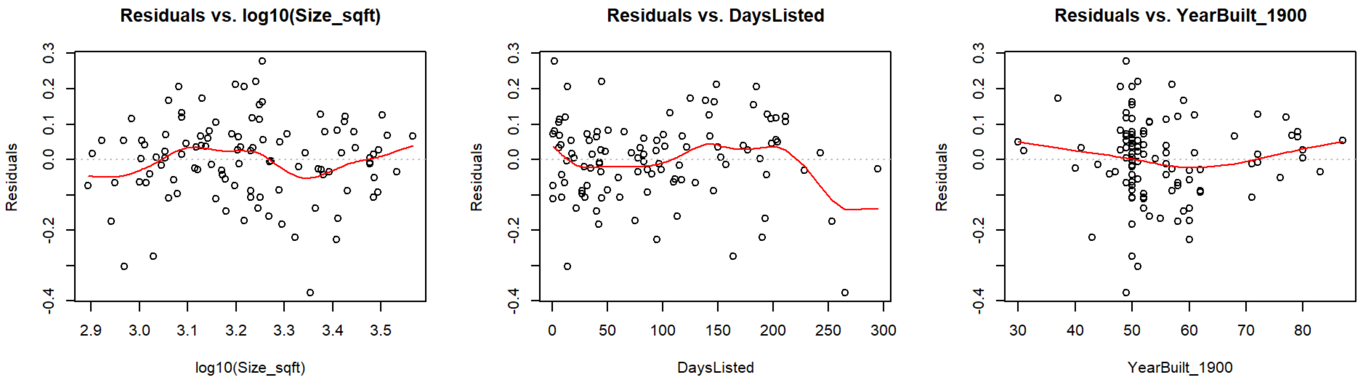
```
Error in Anova.III.lm(mod, error, singular.ok = singular.ok, ...) :
  there are aliased coefficients in the model
```

```
  ## Note that this doesn't work because APARTMENTs only have 1 bed and 1 bath.
  ## There isn't a second level of bed or bath to estimate the interaction.
  ## Therefore, remove those two terms
  lm_full_logSize <-
    update(
      lm_full_logSize
    , . ~ . - TypeSale:Beds
    )
  try(Anova(lm_full_logSize, type=3))
```

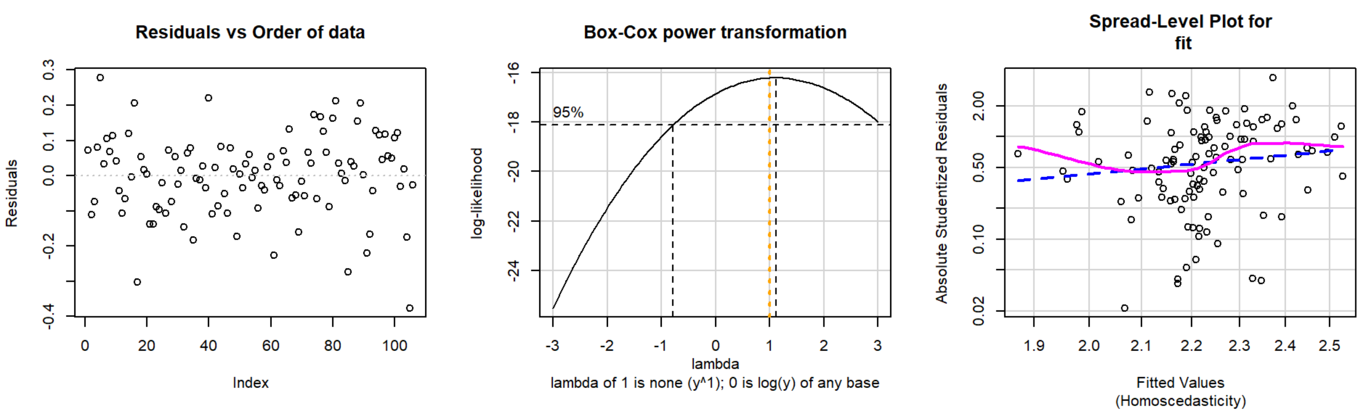
```
Anova Table (Type III tests)

Response: log10(PriceListK)
                                   Sum Sq Df F value Pr(>F)
(Intercept)                       0.00310  1  0.2080 0.6494
TypeSale                          0.01024  1  0.6866 0.4095
Beds                              0.01470  1  0.9856 0.3235
log10(Size_sqft)                  0.00591  1  0.3962 0.5306
DaysListed                        0.00164  1  0.1102 0.7407
YearBuilt_1900                    0.00398  1  0.2671 0.6065
TypeSale:log10(Size_sqft)         0.00446  1  0.2986 0.5861
TypeSale:DaysListed               0.01531  1  1.0263 0.3137
TypeSale:YearBuilt_1900           0.00542  1  0.3633 0.5482
Beds:log10(Size_sqft)             0.00775  1  0.5196 0.4729
Beds:DaysListed                   0.01554  1  1.0414 0.3102
Beds:YearBuilt_1900               0.01964  1  1.3167 0.2542
log10(Size_sqft):DaysListed       0.00073  1  0.0488 0.8257
log10(Size_sqft):YearBuilt_1900   0.00129  1  0.0862 0.7698
DaysListed:YearBuilt_1900         0.00347  1  0.2324 0.6309
Residuals                         1.35764 91
```

```
## Uncomment this line when you're ready to assess the model assumptions
# plot diagnostics
e_plot_lm_diagostics(lm_full_logSize)
```

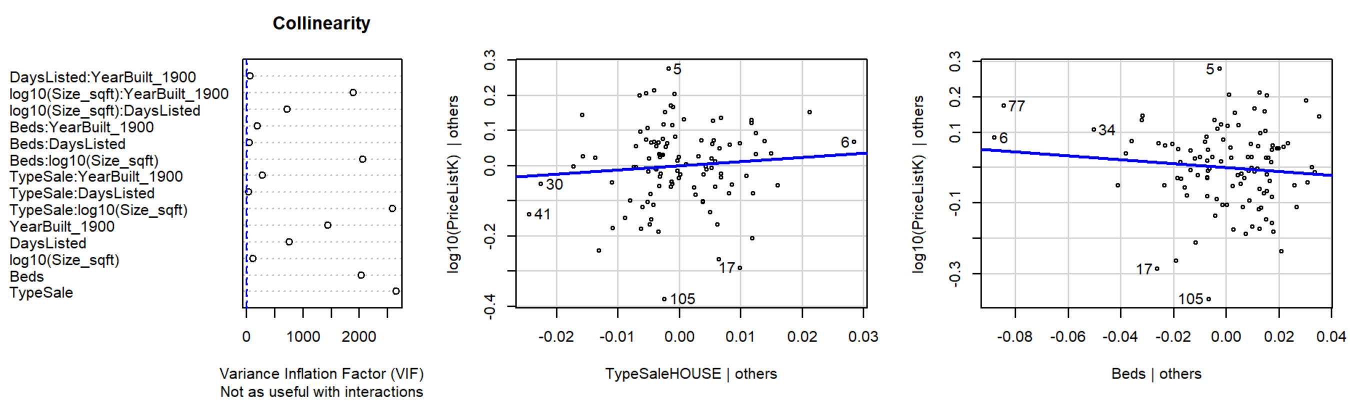


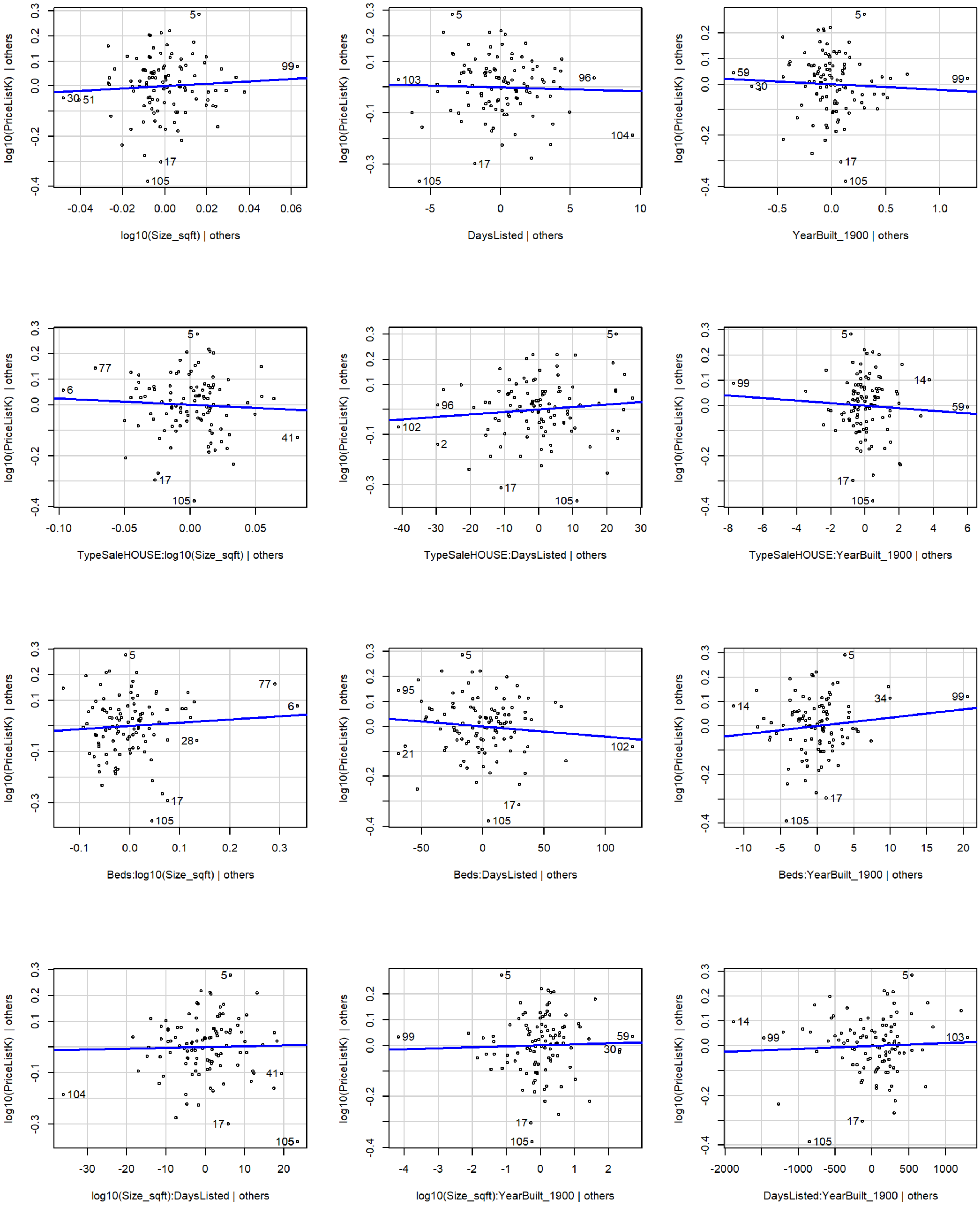



```
Non-constant Variance Score Test
Variance formula: ~ fitted.values
Chisquare = 1.591387, Df = 1, p = 0.20713

there are higher-order terms (interactions) in this model
consider setting type = 'predictor'; see ?vif
```




```
Warning in e_plot_lm_diagostics(lm_full_logSize): Note: Collinearity plot
unreliable for predictors that also have interactions in the model.
```



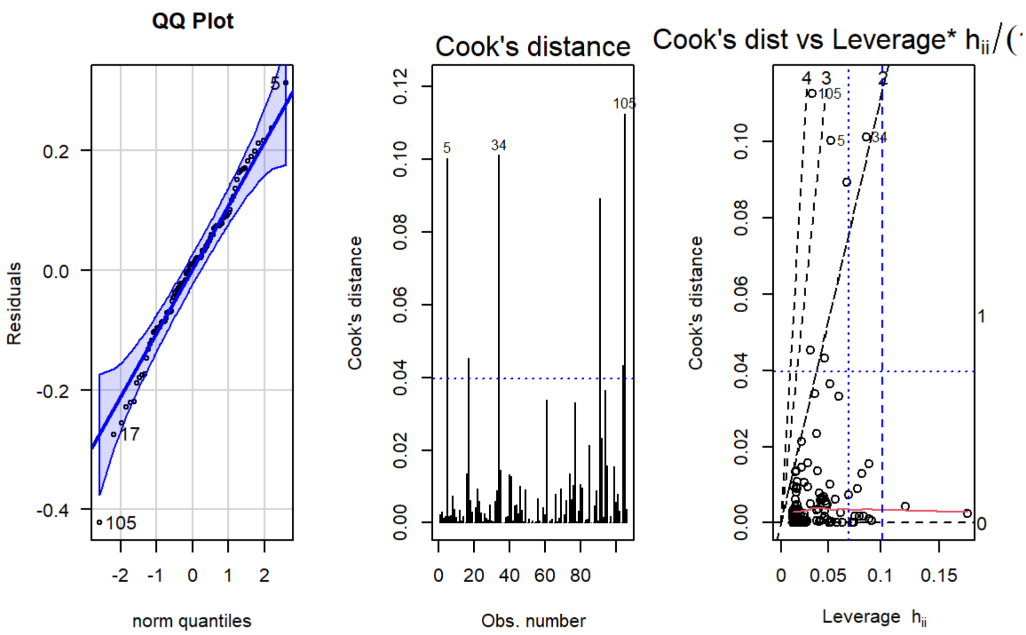


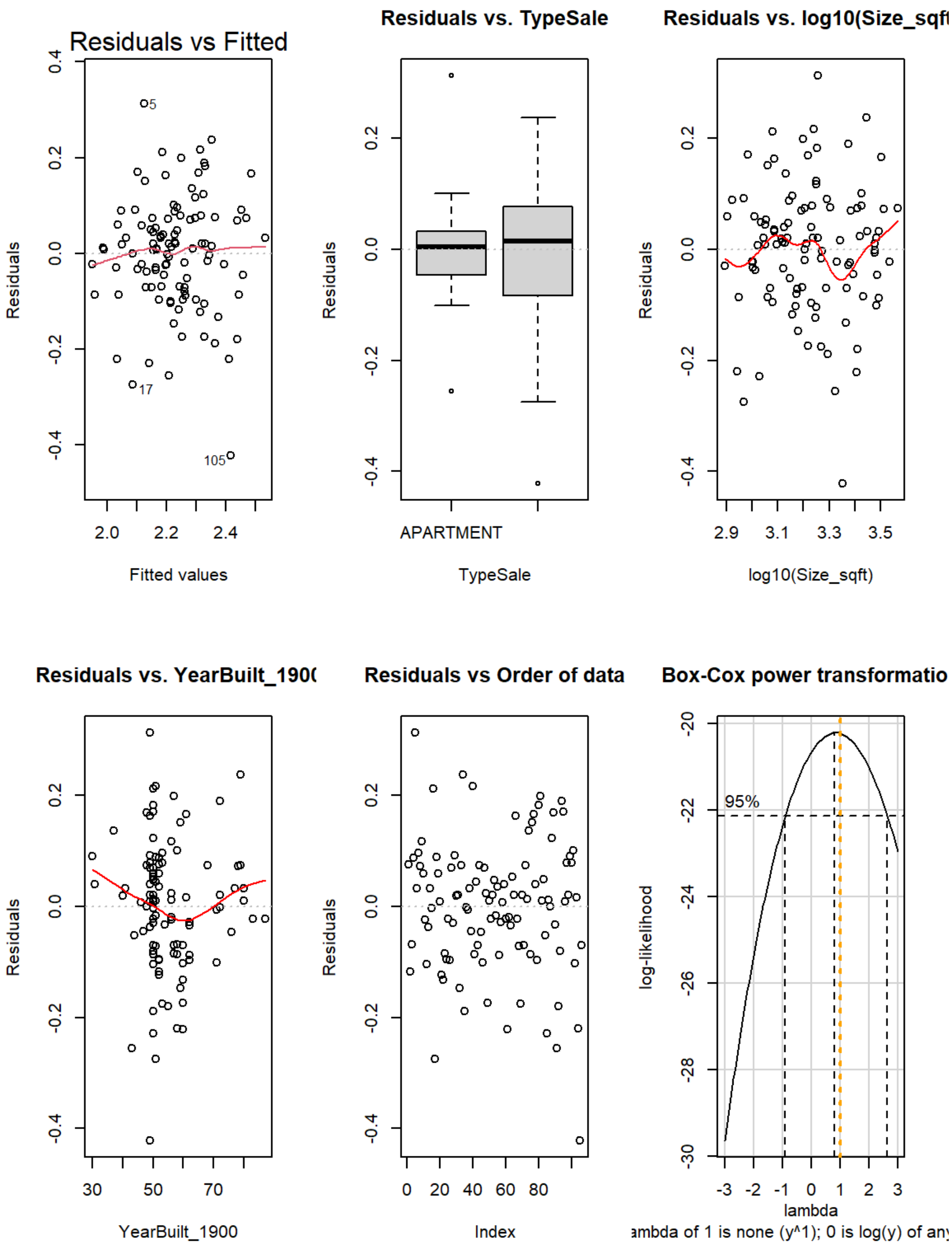

```
# List the row numbers with id numbers
#   The row numbers appear in the residual plots.
```

```
#   The id number can be used to exclude values in code above.
shapiro.test(lm_full_logSize$residuals)
```

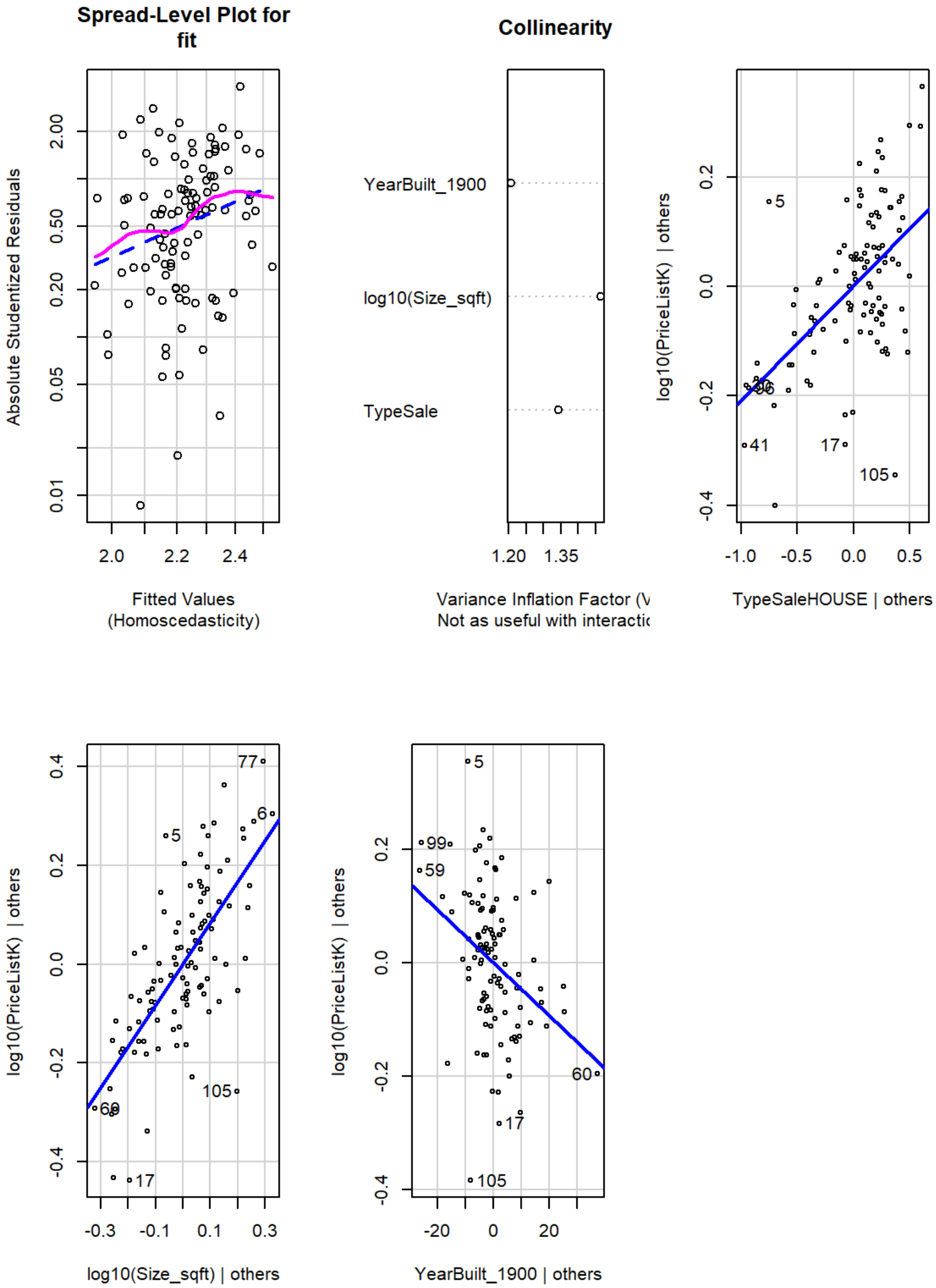
```

	Shapiro-Wilk normality test

data:  lm_full_logSize$residuals
W = 0.98387, p-value = 0.2277
```

```
lm_red_BIC_logSize <-
  step(
    lm_full_logSize
  , direction = "both"
  , test = "F"
  , trace = 0
  , k = log(nrow(dat_sub))
  )

lm.final.logSize = lm_red_BIC_logSize

e_plot_lm_diagostics(lm.final.logSize)
```






```
Non-constant Variance Score Test
Variance formula: ~ fitted.values
Chisquare = 4.091626, Df = 1, p = 0.043096

Warning in e_plot_lm_diagostics(lm.final.logSize): Note: Collinearity plot
unreliable for predictors that also have interactions in the model.
```




```r
summary(lm.final.logSize)
```

```

Call:
lm(formula = log10(PriceListK) ~ TypeSale + log10(Size_sqft) +
    YearBuilt_1900, data = dat_sub)

Residuals:
     Min       1Q   Median       3Q      Max
-0.42236 -0.07062  0.00990  0.07286  0.31216

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
```

```
(Intercept)      -0.360661   0.271947  -1.326 0.187729
TypeSaleHOUSE     0.210319   0.031363   6.706 1.13e-09 ***
log10(Size_sqft)  0.833836   0.084437   9.875  < 2e-16 ***
YearBuilt_1900   -0.004655   0.001236  -3.768 0.000276 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.1198 on 102 degrees of freedom
Multiple R-squared:  0.5129,    Adjusted R-squared:  0.4986
F-statistic:  35.8 on 3 and 102 DF,  p-value: 6.868e-16
```

$$\log 10(\widehat{\text{PriceList}}) = -0.361 + 0.21 I(TypeSale = \text{HOUSE}) + 0.834(\text{log10}(\text{Size.sqft})) + -0.00466(\text{YearBuilt})$$

on average, by one unit log(size_sqft) increase we expect 0.834 increase in log(PriceListK) assuming other variables constant.
on average, by one year decrease in yearBuild we expect -0.00466 increase in log(PriceListK) assuming other variables constant.
for Apartment, on average the log(PriceListK) would be -0.361 if all other variable would be zero(which is not usefull for interpretation).
for House, on average the log(PriceListK) would be (-0.361+0.21) if all other variable would be zero(which is not usefull for interpretation).

(4 p) (Step 8) Predict new observations, interpret model's predictive ability.

Using the `predict()` function, we'll input the data we held out to predict earlier, and use our final model to predict the `PriceListK` response. Note that `10^lm_pred` is the table of values on the scale of "thousands of dollars".

Interpret the predictions below the output.

How well do you expect this model to predict? Justify your answer.

```
# predict new observations, convert to data frame
lm_pred <-
  as.data.frame(
    predict(
      lm.final
    , newdata = dat_pred
    , interval = "prediction"
    )
  ) %>%
  mutate(
    # add column of actual list prices
    PriceListK = dat_pred$PriceListK_true
  )
lm_pred
```

```
       fit      lwr      upr PriceListK
1 2.188175 1.944126 2.432224      186.9
2 2.250483 2.001429 2.499536      305.0
3 2.076528 1.820203 2.332853      244.0
```

```
# on "thousands of dollars" scale
10^lm_pred
```

```
       fit       lwr      upr    PriceListK
1 154.2321  87.92769 270.5354 7.943282e+186
2 178.0256 100.32957 315.8902 1.000000e+305
3 119.2691  66.10022 215.2053 1.000000e+244
```

```
# attributes of the three predicted observations
dat_pred %>% print(n = Inf, width = Inf)
```

```
# A tibble: 3 × 8
     id TypeSale   Beds Size_sqft DaysListed YearBuilt_1900 PriceListK
  <int> <fct>     <dbl>     <dbl>      <dbl>          <dbl> <lgl>
1     1 HOUSE         3      1305          0             54 NA
2     2 APARTMENT     1      2523          0             48 NA
3     3 APARTMENT     1      2816          0             89 NA
  PriceListK_true
            <dbl>
1            187.
2            305
3            244
```

Solution

```
# predict new observations, convert to data frame
lm_pred <-
  as.data.frame(
    predict(
      lm.final
    , newdata = dat_pred
    , interval = "prediction"
    )
  ) %>%
  mutate(
    # add column of actual list prices
    PriceListK = dat_pred$PriceListK_true
  )
lm_pred
```

```
       fit      lwr      upr PriceListK
1 2.188175 1.944126 2.432224      186.9
2 2.250483 2.001429 2.499536      305.0
3 2.076528 1.820203 2.332853      244.0
```

```
# on "thousands of dollars" scale
#10^lm_pred
pre.df = lm_pred %>%
  mutate(fit = 10^fit,
         lwr = 10^lwr,
         upr = 10^upr)
dat_pred$PriceListK = pre.df$fit
dat_predfinal = pre.df
# attributes of the three predicted observations
```

[answer] for a with beds and size_sqft and yearBuild we predict the 154.2321451 PriceListK with interaval (87.9276911, 270.5354172).
for a with beds and size_sqft and yearBuild we predict the 178.0256314 PriceListK with interaval (100.3295663, 315.8901868). for a with beds and size_sqft and yearBuild we predict the 119.2691104 PriceListK with interaval (87.9276911, 215.2053479).

the model did a good job on prediction the first observation (apartment) price with just 32.67 error. for the second observation it predict price 178 with true price of 305. however the prediction is inside the interval but its close to upper interval. for the third observation the model could not predict well. it predict the price 119 but true price is almost two time bigger and is not in the 95% interval. overall it seems the model can not predict the price precisely.

```
# predict new observations, convert to data frame
lm_pred <-
  as.data.frame(
    predict(
      lm.final.logSize
    , newdata = dat_pred
    , interval = "prediction"
    )
  ) %>%
  mutate(
    # add column of actual list prices
    PriceListK = dat_pred$PriceListK_true
  )
lm_pred
```

```
       fit      lwr      upr PriceListK
1 2.196173 1.956783 2.435563      186.9
2 2.252520 2.008123 2.496917      305.0
3 2.101434 1.850131 2.352737      244.0
```

```
# on "thousands of dollars" scale
#10^lm_pred
pre.df = lm_pred %>%
  mutate(fit = 10^fit,
         lwr = 10^lwr,
         upr = 10^upr)
dat_pred$PriceListK = pre.df$fit
```

```
dat_predfinal  = pre.df
# attributes of the three predicted observations
```

The model with log(Size_sqft) in prediction did slightly a better job however overall it did not predict precisely either.