SIM Project 2. Model fitting

Adrià Casanova Víctor Garcia Zhengyong Ji

November, 19th 2023

Contents

8. First model building **1**

9. Model analysis and iteration **5**

10. Adding Factors to the numerical model **11**

11. Checking possible Interactions **13**

12. Model validation **15**

13. Model interpretation **21**

To reduce the time of computations, we have split our code in two .Rmd files. In this one, the preprocessed train dataset is found in df, while the preprocessed test database is in df_test.

8. First model building

We create a first model with all the numerical variables that we selected previously.

```r
df_num <- df[, which(sapply(df, is.numeric))]
m0 = lm(SalePrice ~ ., data=df_num)

summary(m0)
```

```
##
## Call:
## lm(formula = SalePrice ~ ., data = df_num)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -160830  -15890   -1092   14377  164012
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.783e+06  1.122e+05 -15.887  < 2e-16 ***
## LotFrontage   8.821e+01  2.535e+01   3.480 0.000516 ***
## LotArea       1.508e+00  2.702e-01   5.581 2.86e-08 ***
## YearBuilt     4.597e+02  5.315e+01   8.650  < 2e-16 ***
## YearRemodAdd  5.492e+02  5.246e+01  10.469  < 2e-16 ***
## MasVnrArea    2.637e+01  6.338e+00   4.160 3.37e-05 ***
## BsmtFinSF1    1.975e+01  4.851e+00   4.072 4.91e-05 ***
```

```
## BsmtUnfSF     2.595e+00  4.872e+00   0.533 0.594302
## TotalBsmtSF   3.133e+01  5.792e+00   5.408 7.45e-08 ***
## X1stFlrSF    -3.742e+01  1.263e+01  -2.964 0.003092 **
## X2ndFlrSF    -2.545e+01  1.219e+01  -2.087 0.037071 *
## GrLivArea     9.523e+01  1.189e+01   8.008 2.40e-15 ***
## BsmtFullBath  5.513e+02  2.138e+03   0.258 0.796537
## FullBath     -3.287e+03  2.399e+03  -1.370 0.170788
## HalfBath     -3.521e+03  2.304e+03  -1.528 0.126693
## BedroomAbvGr -1.013e+04  1.447e+03  -6.998 3.99e-12 ***
## TotRmsAbvGrd  4.475e+02  1.040e+03   0.430 0.667190
## Fireplaces    8.700e+03  1.491e+03   5.836 6.59e-09 ***
## GarageYrBlt  -9.735e+01  6.592e+01  -1.477 0.139962
## GarageCars    6.429e+03  2.464e+03   2.609 0.009169 **
## GarageArea    2.753e+01  8.866e+00   3.105 0.001943 **
## WoodDeckSF    2.326e+01  7.061e+00   3.294 0.001013 **
## OpenPorchSF   4.699e+01  1.561e+01   3.009 0.002664 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 29940 on 1425 degrees of freedom
## Multiple R-squared:  0.8233, Adjusted R-squared:  0.8206
## F-statistic: 301.9 on 22 and 1425 DF,  p-value: < 2.2e-16
```

```r
vif(m0)
```

```
##  LotFrontage      LotArea    YearBuilt YearRemodAdd   MasVnrArea   BsmtFinSF1
##     1.141502     1.463022     4.139242     1.895906     1.313313     6.894967
##    BsmtUnfSF  TotalBsmtSF    X1stFlrSF    X2ndFlrSF    GrLivArea BsmtFullBath
##     7.381445     8.459562    33.083818    44.080849    53.497983     1.987127
##     FullBath     HalfBath BedroomAbvGr TotRmsAbvGrd   Fireplaces  GarageYrBlt
##     2.743664     2.161873     2.172081     4.340452     1.467970     4.270628
##   GarageCars   GarageArea   WoodDeckSF  OpenPorchSF
##     5.396088     5.481694     1.173485     1.223935
```

There are a lot of features with a vif correlation larger than 5. So, in order to reduce the amount of workload, we decided to keep those that are less than 5 and are highly correlated with our target.

```r
# Let's store the indices of the variables with at least one star in the lm and vif<5
id_num_star1 = c(1:5,15,17,21:23)
df_num1 <- df_num[, id_num_star1]
# And build a new model only with significance features
m1 = lm(SalePrice ~., data=df_num1)
summary(m1)
```

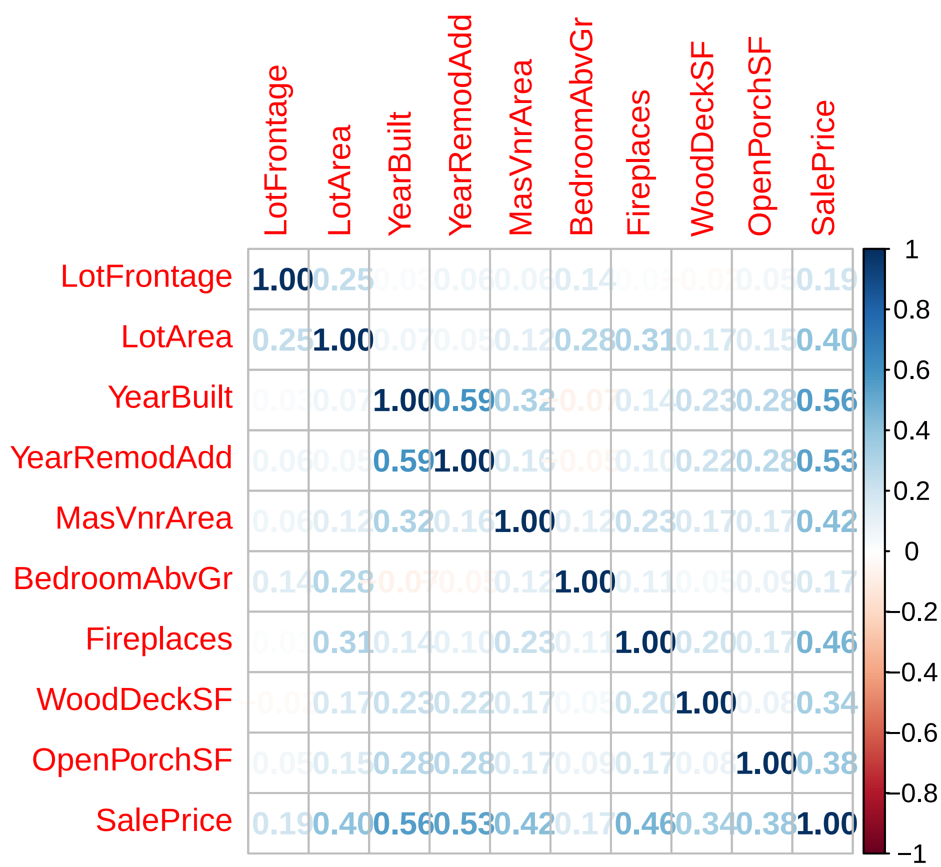
```
##
## Call:
## lm(formula = SalePrice ~ ., data = df_num1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -170973  -25349   -4048   18791  207331
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -2.876e+06  1.159e+05 -24.820  < 2e-16 ***
## LotFrontage   1.951e+02  3.456e+01   5.645 1.99e-08 ***
```

```
## LotArea       3.960e+00  3.532e-01  11.212  < 2e-16 ***
## YearBuilt     5.558e+02  4.789e+01  11.607  < 2e-16 ***
## YearRemodAdd  9.359e+02  6.734e+01  13.897  < 2e-16 ***
## MasVnrArea    8.678e+01  8.411e+00  10.317  < 2e-16 ***
## BedroomAbvGr  5.492e+03  1.453e+03   3.781 0.000163 ***
## Fireplaces    2.779e+04  1.875e+03  14.826  < 2e-16 ***
## WoodDeckSF    5.749e+01  9.669e+00   5.946 3.44e-09 ***
## OpenPorchSF   1.478e+02  2.112e+01   6.997 4.00e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 41800 on 1438 degrees of freedom
## Multiple R-squared:  0.6525, Adjusted R-squared:  0.6503
## F-statistic:   300 on 9 and 1438 DF,  p-value: < 2.2e-16
```

```r
vif(m1)
```

```
##  LotFrontage      LotArea    YearBuilt YearRemodAdd   MasVnrArea BedroomAbvGr
##     1.088404     1.282087     1.723739     1.602800     1.186320     1.122948
##   Fireplaces   WoodDeckSF  OpenPorchSF
##     1.190834     1.128839     1.149369
```

```r
# As we can observe, vif correlations are much better, all values are less than 2.
# So the next step is to check the correlation between predictors.
corr_mat <- cor(df_num1)
corrplot(corr_mat, method = "number")
```



Feature “YearBuilt” and “YearRemodAdd” are highly correlated, and “YearBuilt” is more correlated to our target SalePrice. Hence, we remove YearRemodAdd in the next model.

```r
# Building the model without "YearRemodAdd"
id_num_star2 = c(1:3,5,15,17,21:23)
df_num2 <- df_num[, id_num_star2]
m2 = lm(SalePrice ~., data=df_num2)
summary(m2)
```

```
##
## Call:
## lm(formula = SalePrice ~ ., data = df_num2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -165690  -27970   -5057   19803  205977
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -1.714e+06  8.539e+04 -20.069  < 2e-16 ***
## LotFrontage   2.282e+02  3.670e+01   6.218 6.59e-10 ***
## LotArea       3.810e+00  3.759e-01  10.136  < 2e-16 ***
## YearBuilt     9.075e+02  4.329e+01  20.966  < 2e-16 ***
## MasVnrArea    8.050e+01  8.942e+00   9.002  < 2e-16 ***
## BedroomAbvGr  5.014e+03  1.546e+03   3.243  0.00121 **
## Fireplaces    2.795e+04  1.996e+03  14.006  < 2e-16 ***
## WoodDeckSF    7.267e+01  1.023e+01   7.105 1.88e-12 ***
## OpenPorchSF   1.917e+02  2.224e+01   8.619  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 44500 on 1439 degrees of freedom
## Multiple R-squared:  0.6058, Adjusted R-squared:  0.6036
## F-statistic: 276.4 on 8 and 1439 DF,  p-value: < 2.2e-16
```

Now, the most correlated variables in our model have at most a coefficient of correlation of 0.315, which in the context of real estate it is weak. We have obtained this information from https://37parallel.com/real-estate-correlation/.

```r
Anova(m2)
```

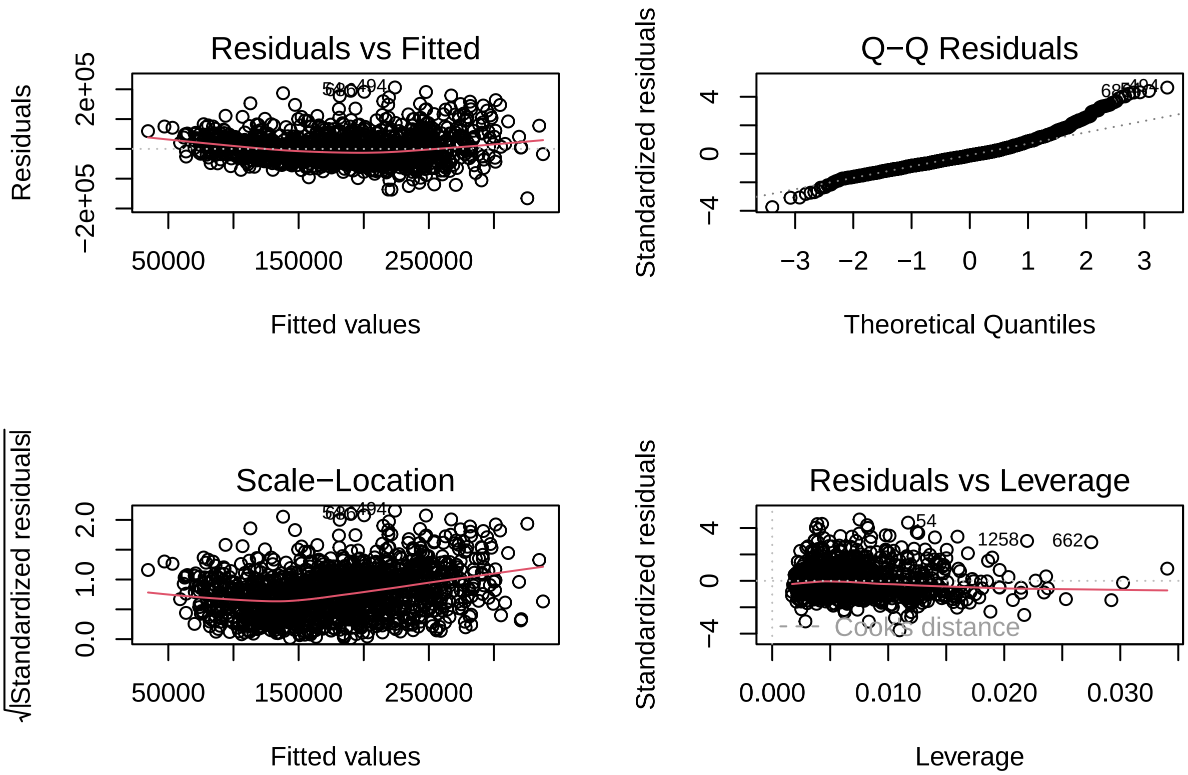
```
## Anova Table (Type II tests)
##
## Response: SalePrice
##                  Sum Sq   Df F value    Pr(>F)
## LotFrontage  7.6561e+10    1  38.662 6.588e-10 ***
## LotArea      2.0346e+11    1 102.744 < 2.2e-16 ***
## YearBuilt    8.7042e+11    1 439.553 < 2.2e-16 ***
## MasVnrArea   1.6047e+11    1  81.034 < 2.2e-16 ***
## BedroomAbvGr 2.0821e+10    1  10.515  0.001212 **
## Fireplaces   3.8845e+11    1 196.163 < 2.2e-16 ***
## WoodDeckSF   9.9972e+10    1  50.485 1.883e-12 ***
## OpenPorchSF  1.4712e+11    1  74.292 < 2.2e-16 ***
## Residuals    2.8496e+12 1439
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Anova shows that all the variables we have kept are relevant.

9. Model analysis and iteration

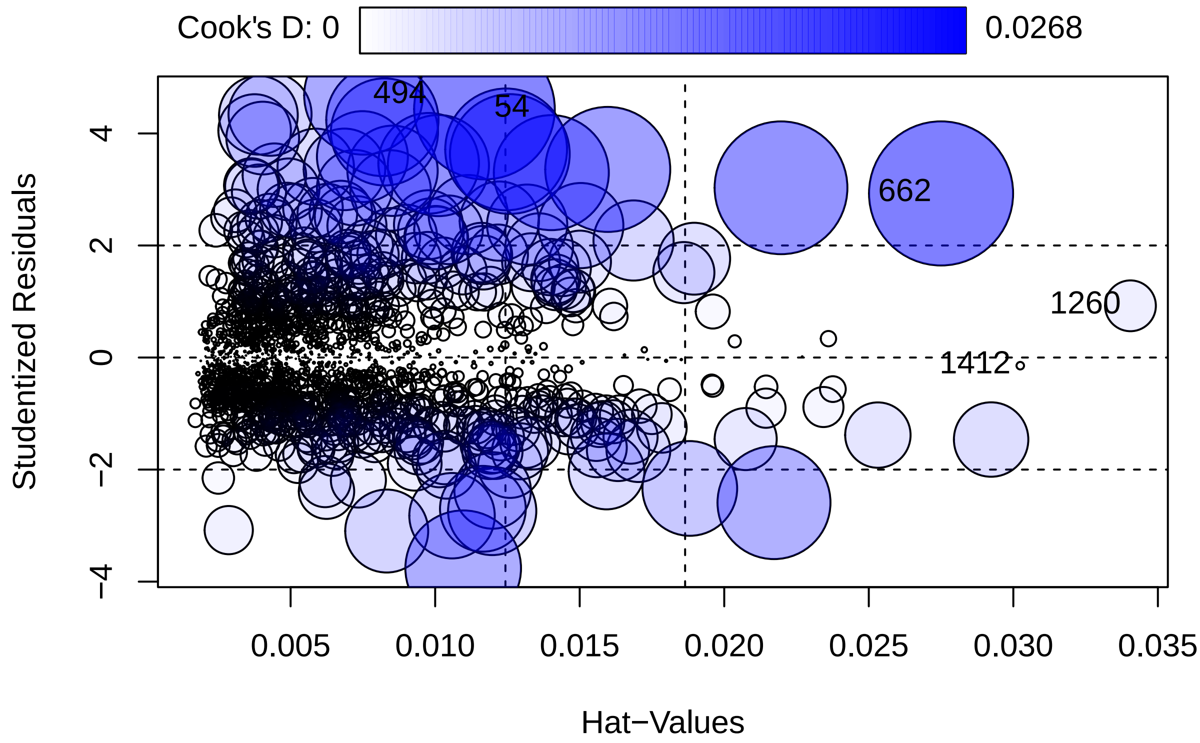
First, let us plot the residuals of m2 to be able to compare them with the next iterations of the model.

```
par(mfrow=c(2,2))
plot(m2)
```



We analysed if there were influential data and found 3 observations with a bigger Cook's distance than the threshold (considered as 2/sqrt(n)). Consequently, we decided to remove those observations.

```
# Check the influential plot before removing the influential observation.
influencePlot(m2)
```




```
##          StudRes         Hat       CookD
## 54    4.4351943 0.011705638 2.555600e-02
## 494   4.6798339 0.007521628 1.817802e-02
## 662   2.9293234 0.027500038 2.681970e-02
## 1260  0.9231784 0.034051251 3.338507e-03
## 1412 -0.1455309 0.030239831 7.343087e-05
```

```
# Calculate D's threshold
D_thresh <- 2/sqrt(dim(df_num2)[1]); D_thresh
```

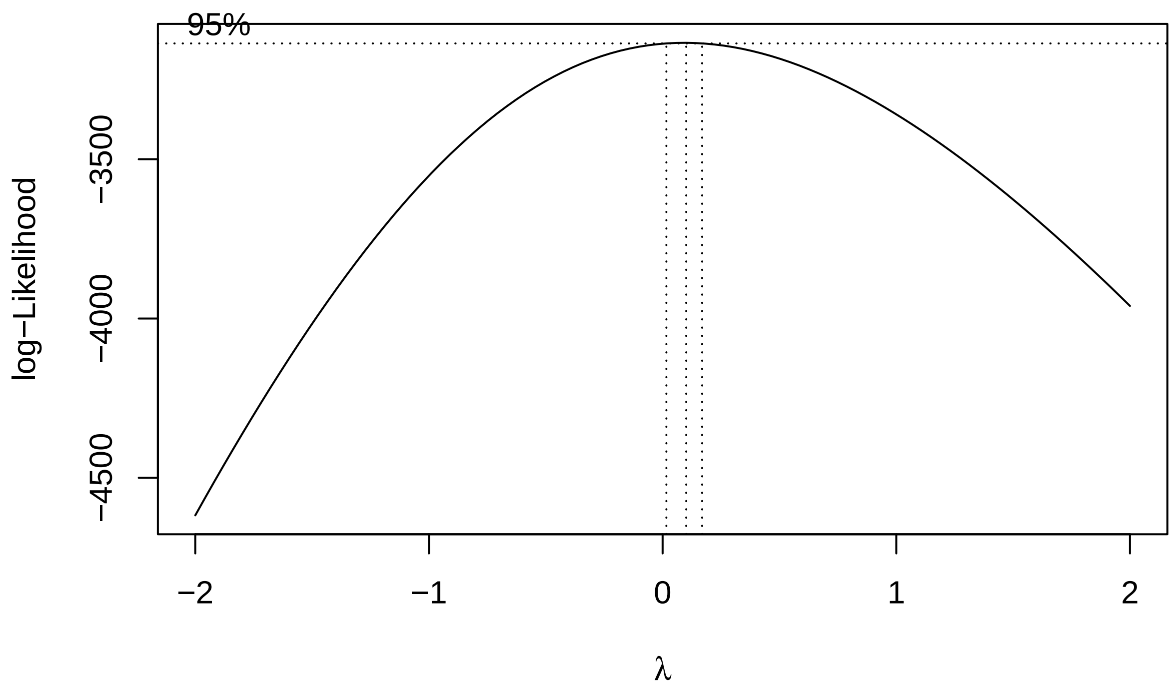
```
## [1] 0.05255883
```

```
#Remove the points and fit the model again
influent <- c(1183, 692, 186)

df <- df[-influent,]
df_num <- df[, which(sapply(df, is.numeric))]
df_num2 <- df_num[, id_num_star2]
m2 = lm(SalePrice ~., data=df_num2)
```

Firstly, we check if there is any needed transformation with boxcox().

```
boxcox(m2)
```




```r
# As the lambda is greater than 0, we should apply a logarithmic transformation
# to SalePrice
m3 = lm(log(SalePrice)~., data=df_num2)
summary(m3)
```

```
##
## Call:
## lm(formula = log(SalePrice) ~ ., data = df_num2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.09727 -0.13827 -0.00372  0.12799  0.94710
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  9.822e-02  4.444e-01   0.221    0.825
## LotFrontage  8.611e-04  1.910e-04   4.507 7.09e-06 ***
## LotArea      2.011e-05  1.956e-06  10.281  < 2e-16 ***
## YearBuilt    5.743e-03  2.253e-04  25.489  < 2e-16 ***
## MasVnrArea   2.988e-04  4.662e-05   6.410 1.97e-10 ***
## BedroomAbvGr 5.453e-02  8.047e-03   6.777 1.78e-11 ***
## Fireplaces   1.625e-01  1.038e-02  15.648  < 2e-16 ***
## WoodDeckSF   3.624e-04  5.321e-05   6.811 1.42e-11 ***
## OpenPorchSF  9.790e-04  1.157e-04   8.462  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 0.2314 on 1436 degrees of freedom
## Multiple R-squared:  0.6412, Adjusted R-squared:  0.6392
## F-statistic: 320.8 on 8 and 1436 DF,  p-value: < 2.2e-16
```

Compared with m2, adjusted R-squared has increased about 4%.

We will proceed now with the study of possible variable transformations. We'll assign 10^(-6) to all cells equal to 0 to be able to use boxTidwell() without altering too much the model

```r
df_num2 = replace(df_num2, df_num2 == 0, 1e-6)
summary(df_num2)
```

```
##   LotFrontage        LotArea        YearBuilt      MasVnrArea
##  Min.   :  0.00   Min.   : 1300   Min.   :1872   Min.   :  0.00
##  1st Qu.: 42.00   1st Qu.: 7500   1st Qu.:1954   1st Qu.:  0.00
##  Median : 63.00   Median : 9375   Median :1972   Median :  0.00
##  Mean   : 57.05   Mean   : 9493   Mean   :1971   Mean   : 90.18
##  3rd Qu.: 78.00   3rd Qu.:11316   3rd Qu.:2000   3rd Qu.:158.99
##  Max.   :182.00   Max.   :23595   Max.   :2010   Max.   :664.00
##   BedroomAbvGr        Fireplaces          WoodDeckSF      OpenPorchSF
##  Min.   :0.000001   Min.   :0.000001   Min.   :  0.00   Min.   :  0.00
##  1st Qu.:2.000000   1st Qu.:0.000001   1st Qu.:  0.00   1st Qu.:  0.00
##  Median :3.000000   Median :1.000000   Median :  0.00   Median : 24.00
##  Mean   :2.861519   Mean   :0.605537   Mean   : 92.28   Mean   : 42.55
##  3rd Qu.:3.000000   3rd Qu.:1.000000   3rd Qu.:168.00   3rd Qu.: 65.00
##  Max.   :6.000000   Max.   :3.000000   Max.   :670.00   Max.   :267.00
##    SalePrice
##  Min.   : 34900
##  1st Qu.:129900
##  Median :162000
##  Mean   :177697
##  3rd Qu.:213000
##  Max.   :465000
```

```r
boxTidwell(log(SalePrice) ~ LotArea+YearBuilt+MasVnrArea, data = df_num2)
```

```
##             MLE of lambda Score Statistic (t)  Pr(>|t|)
## LotArea          0.46268             -4.3123 1.725e-05 ***
## YearBuilt       66.57971             14.5973 < 2.2e-16 ***
## MasVnrArea       1.01690              0.0152    0.9879
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## iterations =  5
##
## Score test for null hypothesis that all lambdas = 1:
## F = 77.374, df = 3 and 1438, Pr(>F) = < 2.2e-16
```

```r
# We should apply sqrt(LotArea). YearBuilt's lambda is too large, so it would be
# difficult to interpret the model using it. MasVnrArea has a too large p-value,
# so we cannot reject the null hypothesis that its lambda = 1.
boxTidwell(log(SalePrice)~LotFrontage, data = df_num2)
```

```
## Warning in boxTidwell.default(y, X1, X2, max.iter = max.iter, tol = tol, :
## maximum iterations exceeded
```

```
##  MLE of lambda Score Statistic (t)  Pr(>|t|)
```

```
##        -3.1109              11.028 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## iterations =  26
```

```r
# Too small lambda
boxTidwell(log(SalePrice)~BedroomAbvGr, data = df_num2)
```

```
## Warning in boxTidwell.default(y, X1, X2, max.iter = max.iter, tol = tol, :
## maximum iterations exceeded
```

```
##  MLE of lambda Score Statistic (t) Pr(>|t|)
##        0.98657              0.3194   0.7494
##
## iterations =  26
```

```r
# Too large p-value
boxTidwell(log(SalePrice)~Fireplaces, data =df_num2)
```

```
##  MLE of lambda Score Statistic (t)  Pr(>|t|)
##        0.17624             -8.0252 2.083e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## iterations =  3
```

```r
# We apply log() to Fireplaces
boxTidwell(log(SalePrice)~WoodDeckSF, data = df_num2)
```

```
##  MLE of lambda Score Statistic (t)  Pr(>|t|)
##        0.50697             -5.2996 1.341e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## iterations =  7
```

```r
# We apply sqrt() to WoodDeckSF
boxTidwell(log(SalePrice)~OpenPorchSF, data = df_num2)
```

```
## Warning in boxTidwell.default(y, X1, X2, max.iter = max.iter, tol = tol, :
## maximum iterations exceeded
```

```
##  MLE of lambda Score Statistic (t)  Pr(>|t|)
##        -7.8358             -11.723 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## iterations =  26
```

```r
# Too small lambda
```

Using the boxTidwell method, the transformation below can be applied to m4.

```r
m4 = lm(log(SalePrice) ~ LotFrontage+sqrt(LotArea)+YearBuilt+MasVnrArea+
          BedroomAbvGr+log(Fireplaces)+sqrt(WoodDeckSF)+OpenPorchSF,
        data=df_num2)
summary(m4)
```

```
##
```

```
## Call:
## lm(formula = log(SalePrice) ~ LotFrontage + sqrt(LotArea) + YearBuilt +
##     MasVnrArea + BedroomAbvGr + log(Fireplaces) + sqrt(WoodDeckSF) +
##     OpenPorchSF, data = df_num2)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.10276 -0.14161 -0.00581  0.13022  0.87128
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      7.403e-01  4.544e-01   1.629 0.103443
## LotFrontage      6.910e-04  1.919e-04   3.601 0.000327 ***
## sqrt(LotArea)    4.130e-03  3.631e-04  11.373  < 2e-16 ***
## YearBuilt        5.418e-03  2.293e-04  23.623  < 2e-16 ***
## MasVnrArea       3.151e-04  4.654e-05   6.770 1.87e-11 ***
## BedroomAbvGr     5.103e-02  8.086e-03   6.311 3.70e-10 ***
## log(Fireplaces)  1.467e-02  9.639e-04  15.218  < 2e-16 ***
## sqrt(WoodDeckSF) 6.185e-03  9.073e-04   6.817 1.37e-11 ***
## OpenPorchSF      9.839e-04  1.157e-04   8.505  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2313 on 1436 degrees of freedom
## Multiple R-squared:  0.6416, Adjusted R-squared:  0.6396
## F-statistic: 321.3 on 8 and 1436 DF,  p-value: < 2.2e-16
```

Adjusted R-squared has increased slightly. Since we cannot find a significant improvement, we will compare m3 and m4 with a more advanced tool, the BIC.

```
BIC(m3, m4)
```

```
##    df       BIC
## m3 10 -65.40847
## m4 10 -66.89307
```

The overall improvement of applying all transformations simultaneously is small, so we decided to check different combinations to find a better result.

```
m5 = lm(log(SalePrice) ~ LotFrontage+LotArea+YearBuilt+MasVnrArea+
          BedroomAbvGr+log(Fireplaces)+sqrt(WoodDeckSF)+OpenPorchSF,data=df_num2)
m6 = lm(log(SalePrice) ~ LotFrontage+sqrt(LotArea)+YearBuilt+MasVnrArea+
          BedroomAbvGr+Fireplaces+sqrt(WoodDeckSF)+OpenPorchSF,data=df_num2)
m7 = lm(log(SalePrice) ~ LotFrontage+sqrt(LotArea)+YearBuilt+MasVnrArea
        +BedroomAbvGr+log(Fireplaces)+WoodDeckSF+OpenPorchSF,data=df_num2)
m8 = lm(log(SalePrice)~LotFrontage+sqrt(LotArea)+YearBuilt+MasVnrArea+
          BedroomAbvGr+Fireplaces+WoodDeckSF+OpenPorchSF, data=df_num2)
m9 = lm(log(SalePrice)~LotFrontage+LotArea+YearBuilt+MasVnrArea+BedroomAbvGr+
          log(Fireplaces)+WoodDeckSF+OpenPorchSF, data=df_num2)
m10 = lm(log(SalePrice)~LotFrontage+LotArea+YearBuilt+MasVnrArea+BedroomAbvGr+
           Fireplaces+sqrt(WoodDeckSF)+OpenPorchSF, data=df_num2)
BIC(m4,m5,m6,m7,m8,m9,m10)
```

```
##     df       BIC
## m4  10 -66.89307
## m5  10 -56.73155
```

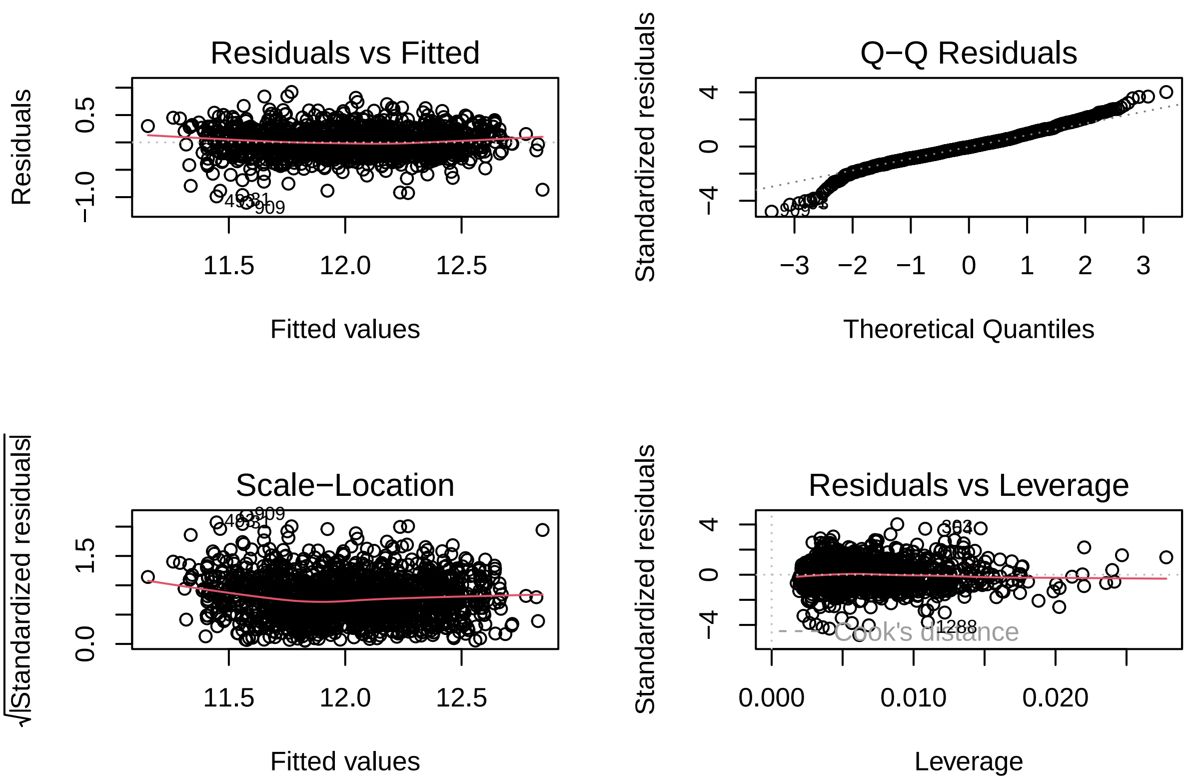
```
## m6   10 -78.08378
## m7   10 -64.27974
## m8   10 -74.75244
## m9   10 -54.29115
## m10  10 -68.56782
```

The best model is m6, that only applies sqrt() to LotArea and WoodDeckSF. For this model we have compared the distribution of residuals and realized that it is very similar to the original model.

```r
par(mfrow=c(2,2))
m11 = lm(log(SalePrice) ~ LotFrontage+sqrt(LotArea)+YearBuilt+MasVnrArea
         +BedroomAbvGr+Fireplaces+sqrt(WoodDeckSF)+OpenPorchSF,data=df_num)
BIC(m3,m11)
```

```
##     df       BIC
## m3  10 -65.40847
## m11 10 -78.08351
```

```r
plot(m11)
```



10. Adding Factors to the numerical model

We followed an heuristic approach when we added factors to the model. As there was an important amount of numeric variables, we tried to add factor variables one by one. We started with the predictor most correlated with the target and continued in decreasing order. To test the improvement of the model's forecasting capability we analysed its BIC and R^2. Moreover, Anova() and step() methods suggest whether some predictors should be removed.

The results of the code of this section are very long and repetitive, so we hide them in the report.

```
m12 = lm(log(SalePrice)~LotFrontage+sqrt(LotArea)+YearBuilt+MasVnrArea
         +BedroomAbvGr+Fireplaces+sqrt(WoodDeckSF)+OpenPorchSF+OverallQual, data=df)
BIC(m11,m12)
Anova(m12)
step(m12, k = log(nrow(df)))
```

Comparing m11 and m12, there was a huge improvement in terms of BIC and Adjusted R-squared, as we expected.

The Anova test indicates that LotFrontage loses its significance once we add OverallQual, and the step method suggests to remove it.

```
m12.1 = lm(log(SalePrice)~sqrt(LotArea)+YearBuilt+MasVnrArea+BedroomAbvGr
           +Fireplaces+sqrt(WoodDeckSF)+OpenPorchSF+OverallQual, data=df)
summary(m12.1)
BIC(m10,m12,m12.1)
```

After removing LotFrontage, although R^2 didn't change, BIC increased because we used less variables and avoided overfitting.

Next, in m13, we have added ExterQual.

```
m13 = lm(log(SalePrice)~sqrt(LotArea)+YearBuilt+MasVnrArea+BedroomAbvGr+
           Fireplaces+sqrt(WoodDeckSF)+OpenPorchSF+OverallQual+ExterQual, data=df)
summary(m13)
BIC(m13,m12.1)
Anova(m13)
step(m13, k = log(nrow(df)))
```

All parameters show that it is correct to add ExterQual, so we continue by adding BsmtQual to the model.

```
m14 = lm(log(SalePrice)~sqrt(LotArea)+YearBuilt+MasVnrArea+BedroomAbvGr+
           Fireplaces+sqrt(WoodDeckSF)+OpenPorchSF+OverallQual+ExterQual+BsmtQual, data=df)
summary(m14)
BIC(m14,m13)
Anova(m14)
step(m14, k = log(nrow(df)))
```

After this, we add KitcheQual.

```
m15 = lm(log(SalePrice)~sqrt(LotArea)+YearBuilt+MasVnrArea+BedroomAbvGr+
           Fireplaces+sqrt(WoodDeckSF)+OpenPorchSF+OverallQual+ExterQual+
           BsmtQual+KitchenQual, data=df); summary(m15)
BIC(m15,m14)
Anova(m15)
step(m15, k = log(nrow(df)))
```

The step method shows that ExterQual, after adding the KitchenQual, has lost significance and suggests to remove it. Indeed, BIC improves afterwards.

```
m15.1 = lm(log(SalePrice)~sqrt(LotArea)+YearBuilt+MasVnrArea+BedroomAbvGr+
             Fireplaces+sqrt(WoodDeckSF)+OpenPorchSF+OverallQual+BsmtQual+
             KitchenQual, data=df); summary(m15.1)
BIC(m15.1,m15)
Anova(m15.1)
step(m15.1, k = log(nrow(df)))
```

Adding Neighbourhood to the model.

```
m16.1 = lm(log(SalePrice)~sqrt(LotArea)+YearBuilt+MasVnrArea+BedroomAbvGr+
             Fireplaces+sqrt(WoodDeckSF)+OpenPorchSF+OverallQual+BsmtQual+
             KitchenQual+Neighborhood, data=df); summary(m16.1)
BIC(m16.1,m15.1)
Anova(m16.1)
step(m16.1, k = log(nrow(df)))
```

Adding GarageFinish.

```
m16.2 = lm(log(SalePrice)~sqrt(LotArea)+YearBuilt+MasVnrArea+BedroomAbvGr+
             Fireplaces+sqrt(WoodDeckSF)+OpenPorchSF+OverallQual+BsmtQual+
             KitchenQual+Neighborhood+GarageFinish, data=df); summary(m16.2)
BIC(m16.2,m16.1)
Anova(m16.2)
step(m16.2, k = log(nrow(df)))
```

Adding FireplaceQu.

```
m16.3 = lm(log(SalePrice)~sqrt(LotArea)+YearBuilt+MasVnrArea+BedroomAbvGr+
             Fireplaces+sqrt(WoodDeckSF)+OpenPorchSF+OverallQual+BsmtQual+
             KitchenQual+Neighborhood+GarageFinish+FireplaceQu, data=df)
summary(m16.3)
BIC(m16.3,m16.2,m16.1)
Anova(m16.3)
step(m16.3, k = log(nrow(df)))
```

In m16.3, FireplaceQu's coefficient has a p-value larger than 0.05 and, indeed, step() suggests to remove it from the model. Hence, we stop adding new categorical variables.

11. Checking possible Interactions

YearBuilt and OverallQual intuitively should interact because of inflation. Indeed, all variables could interact with YearBuilt, but OverallQual summarizes them.

We will also hide the output of this section's chunks to shorten the report.

```
m17 = lm(log(SalePrice)~sqrt(LotArea)+MasVnrArea+
          BedroomAbvGr+Fireplaces+sqrt(WoodDeckSF)+OpenPorchSF+YearBuilt*
           OverallQual+BsmtQual+KitchenQual+Neighborhood+GarageFinish, data=df)
summary(m17)
BIC(m17,m16.2)
Anova(m17)
step(m17, k = log(nrow(df)))
```

2. LotArea and YearBuilt should interact as well because of inflation.

```
m18 = lm(log(SalePrice)~MasVnrArea+
          BedroomAbvGr+Fireplaces+sqrt(WoodDeckSF)+OpenPorchSF+YearBuilt*
           OverallQual+sqrt(LotArea)*YearBuilt+OverallQual+BsmtQual+KitchenQual
         +Neighborhood+GarageFinish, data=df); summary(m18)
BIC(m18,m17,m16.2)
Anova(m18)
step(m18, k = log(nrow(df)))
```

Any of these interactions have improved much the model, so we won't keep them. No other interaction would make sense, so we will not try anymore.

Our final model is m16.2. That is, log(SalePrice) ~ sqrt(LotArea) + YearBuilt + MasVnrArea + BedroomAbvGr + Fireplaces + sqrt(WoodDeckSF) + OpenPorchSF + OverallQual + BsmtQual + KitchenQual + Neighborhood + GarageFinish. Its adjusted R^2 is 0.8195 and its BIC is about -972.

```r
summary(m16.2)
```

```
##
## Call:
## lm(formula = log(SalePrice) ~ sqrt(LotArea) + YearBuilt + MasVnrArea +
##     BedroomAbvGr + Fireplaces + sqrt(WoodDeckSF) + OpenPorchSF +
##     OverallQual + BsmtQual + KitchenQual + Neighborhood + GarageFinish,
##     data = df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.94507 -0.08482  0.00502  0.09269  0.55190
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)          9.0726845  0.5108987  17.758  < 2e-16 ***
## sqrt(LotArea)        0.0035680  0.0002559  13.941  < 2e-16 ***
## YearBuilt            0.0012324  0.0002529   4.873 1.22e-06 ***
## MasVnrArea           0.0001410  0.0000342   4.122 3.98e-05 ***
## BedroomAbvGr         0.0538917  0.0059628   9.038  < 2e-16 ***
## Fireplaces           0.0805645  0.0077364  10.414  < 2e-16 ***
## sqrt(WoodDeckSF)     0.0031694  0.0006553   4.837 1.46e-06 ***
## OpenPorchSF          0.0003211  0.0000847   3.791 0.000156 ***
## OverallQualGood      0.2763502  0.0211637  13.058  < 2e-16 ***
## OverallQualModerate  0.1367402  0.0166894   8.193 5.61e-16 ***
## OverallQualVBad     -0.4854769  0.0779016  -6.232 6.06e-10 ***
## OverallQualVGood     0.3797439  0.0384279   9.882  < 2e-16 ***
## BsmtQualFa          -0.1596736  0.0395162  -4.041 5.61e-05 ***
## BsmtQualGd          -0.0830346  0.0215893  -3.846 0.000125 ***
## BsmtQualNBsmt       -0.2611066  0.0372542  -7.009 3.70e-12 ***
## BsmtQualTA          -0.1305402  0.0255610  -5.107 3.72e-07 ***
## KitchenQualFa       -0.2111850  0.0376490  -5.609 2.44e-08 ***
## KitchenQualGd       -0.0727347  0.0232570  -3.127 0.001799 **
## KitchenQualTA       -0.1692437  0.0248729  -6.804 1.49e-11 ***
## NeighborhoodPoor    -0.0400814  0.0127404  -3.146 0.001689 **
## NeighborhoodRich     0.1339629  0.0131859  10.160  < 2e-16 ***
## GarageFinishNGar    -0.1900531  0.0241084  -7.883 6.30e-15 ***
## GarageFinishRFn     -0.0174421  0.0125007  -1.395 0.163145
## GarageFinishUnf     -0.0653116  0.0142690  -4.577 5.12e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1637 on 1421 degrees of freedom
## Multiple R-squared:  0.8224, Adjusted R-squared:  0.8195
## F-statistic:   286 on 23 and 1421 DF,  p-value: < 2.2e-16
```

```r
BIC(m16.2, m11, m1)
```

```
## Warning in BIC.default(m16.2, m11, m1): models are not all fitted to the same
## number of observations
```

```
##       df         BIC
```

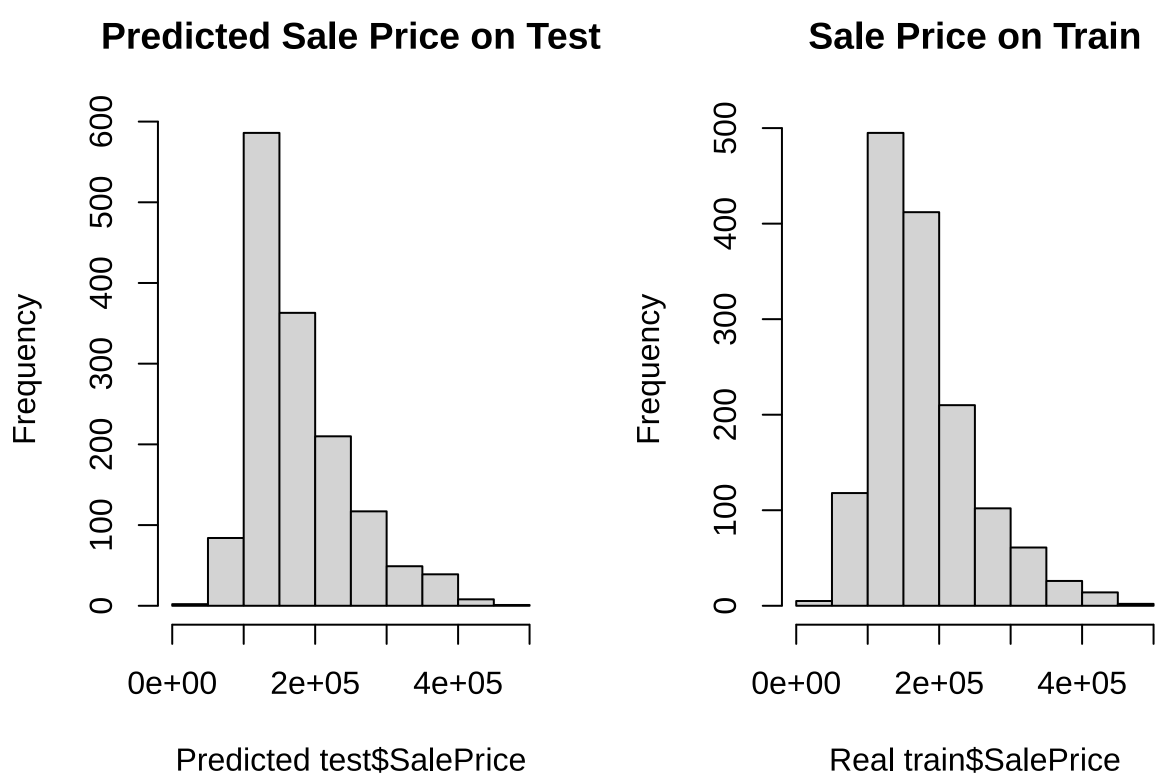
```
## m16.2 25   -972.22665
## m11    10    -78.08351
## m1     11 34994.38604
```

12. Model validation

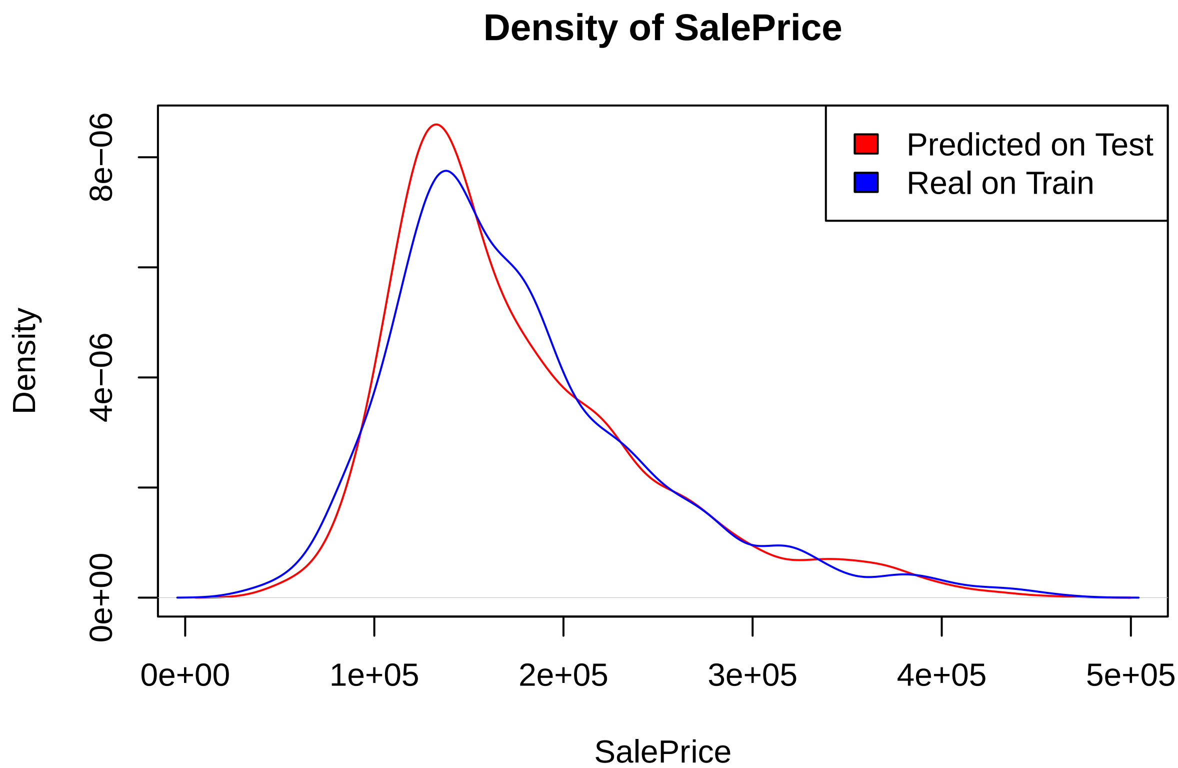
We predict the SalePrice on the test dataset and compare its distribution with the original one in train.

```
predicted_values = predict.lm(m16.2, df_test, se.fit=TRUE,
                              interval="prediction", level=0.95)
test_price = exp(predicted_values$fit)
```

```
par(mfrow=c(1,2))
hist(test_price[,1], main = "Predicted Sale Price on Test",
     xlab =  "Predicted test$SalePrice")
hist(df$SalePrice, main = "Sale Price on Train",
     xlab = "Real train$SalePrice")
```



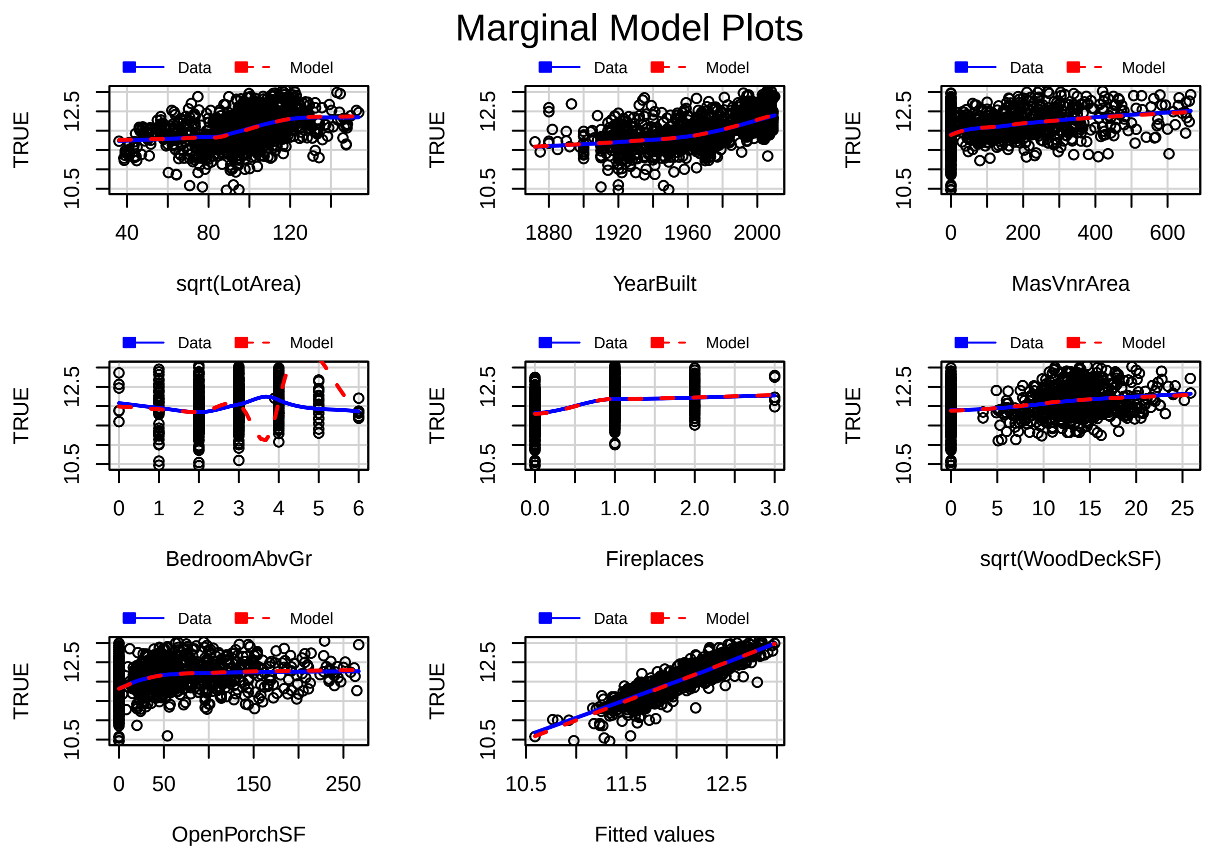

```
par(mfrow=c(1,1))
plot(density(test_price[,1]), col="red", main = "Density of SalePrice",
     xlab = "SalePrice")
lines(density(df$SalePrice), col="blue")
legend("topright",fill = c("red", "blue"), c("Predicted on Test","Real on Train"))
```



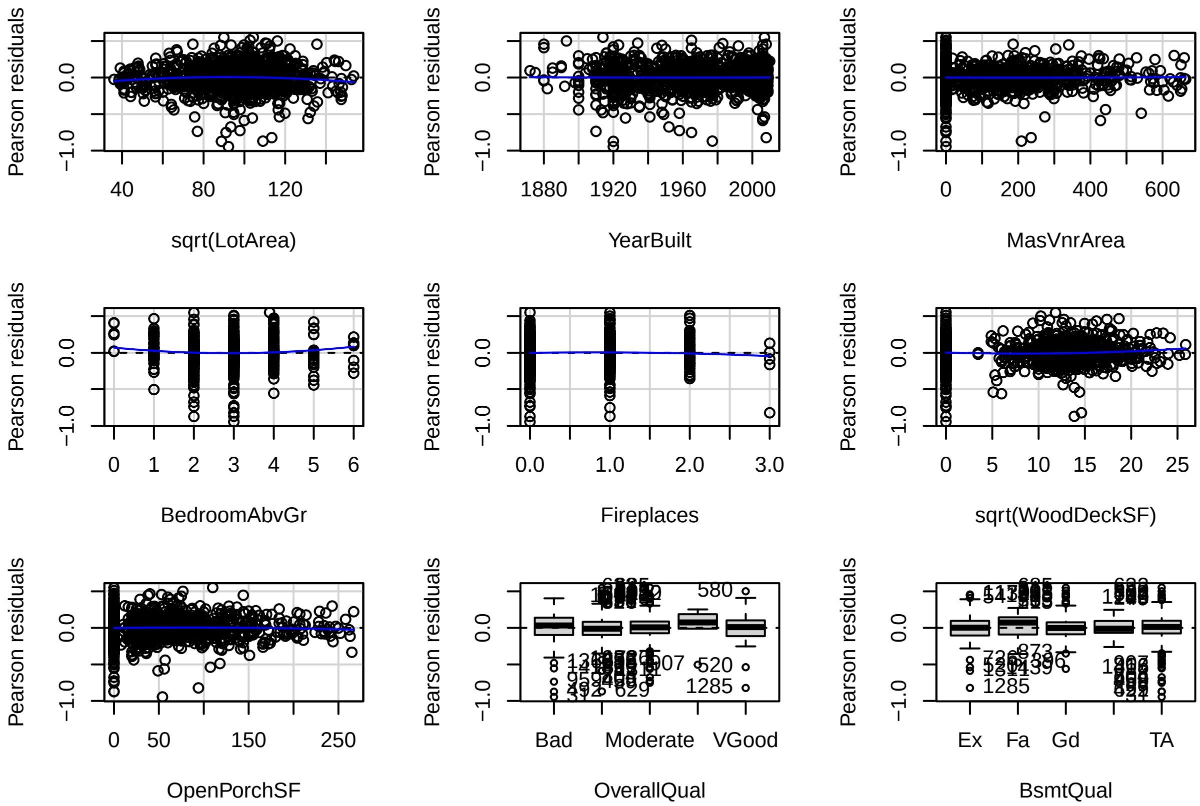
As can be seen in the previous plots, the real and the predicted distributions of SalePrice are similar, but not identical. This was exactly our goal, since both test and train come from the same population and we wanted to avoid overfitting.

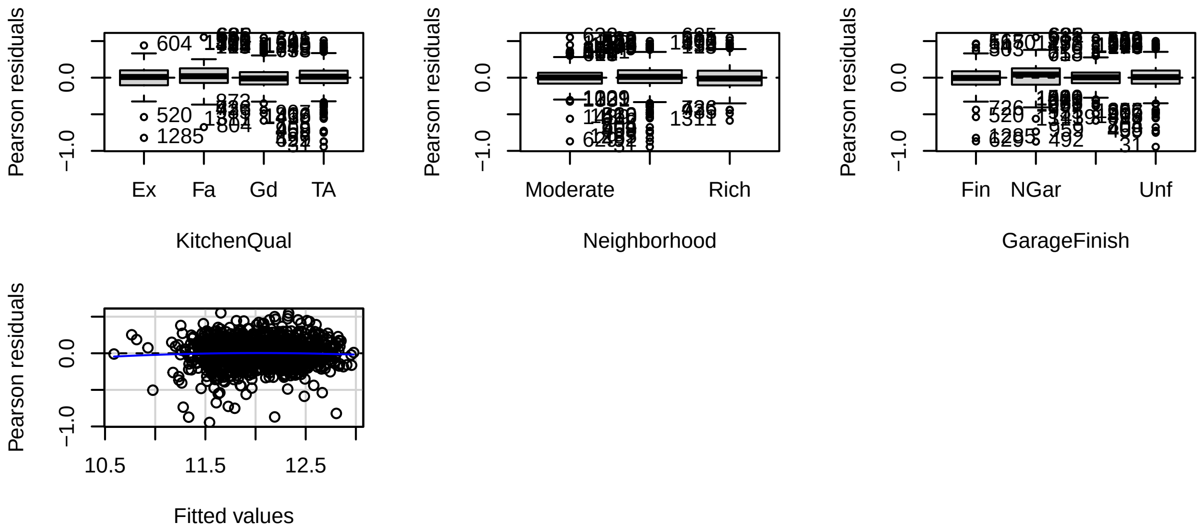
```
marginalModelPlots(m16.2, id=list(n=0))
```

```
## Warning in mmps(...): Interactions and/or factors skipped
```




```r
residualPlots( m16.2, id=list(n=0))
```


Pearson residuals
sqrt(LotArea)
YearBuilt
MasVnrArea
BedroomAbvGr
Fireplaces
sqrt(WoodDeckSF)
OpenPorchSF
OverallQual
BsmtQual
Bad
Moderate
VGood
Ex
Fa
Gd
TA




```
##                    Test stat Pr(>|Test stat|)
## sqrt(LotArea)        -2.5595         0.010585 *
## YearBuilt             0.2685         0.788362
## MasVnrArea            0.3195         0.749388
## BedroomAbvGr          2.6001         0.009417 **
## Fireplaces           -1.2491         0.211843
## sqrt(WoodDeckSF)      1.9216         0.054861 .
## OpenPorchSF          -0.9054         0.365406
## OverallQual
## BsmtQual
## KitchenQual
## Neighborhood
## GarageFinish
## Tukey test           -1.7887         0.073669 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

In general, using the marginal model plots, we can see that the residuals distribution for most variables are close to 0. However, sqrt(LotArea) seems to have bad residuals in marginalModelPlots(), but not in residualPlots(). This could simply mean the first method doesn't properly represent the residuals of this variable. As for categorical variables, all errors are close to 0, except for the level "VBad" of OverallQual, which is due to the fact that it contains few individuals.

```
ks_test_result <- ks.test(test_price[,1], df$SalePrice)
```

```
## Warning in ks.test.default(test_price[, 1], df$SalePrice): p-value will be
## approximate in the presence of ties
```

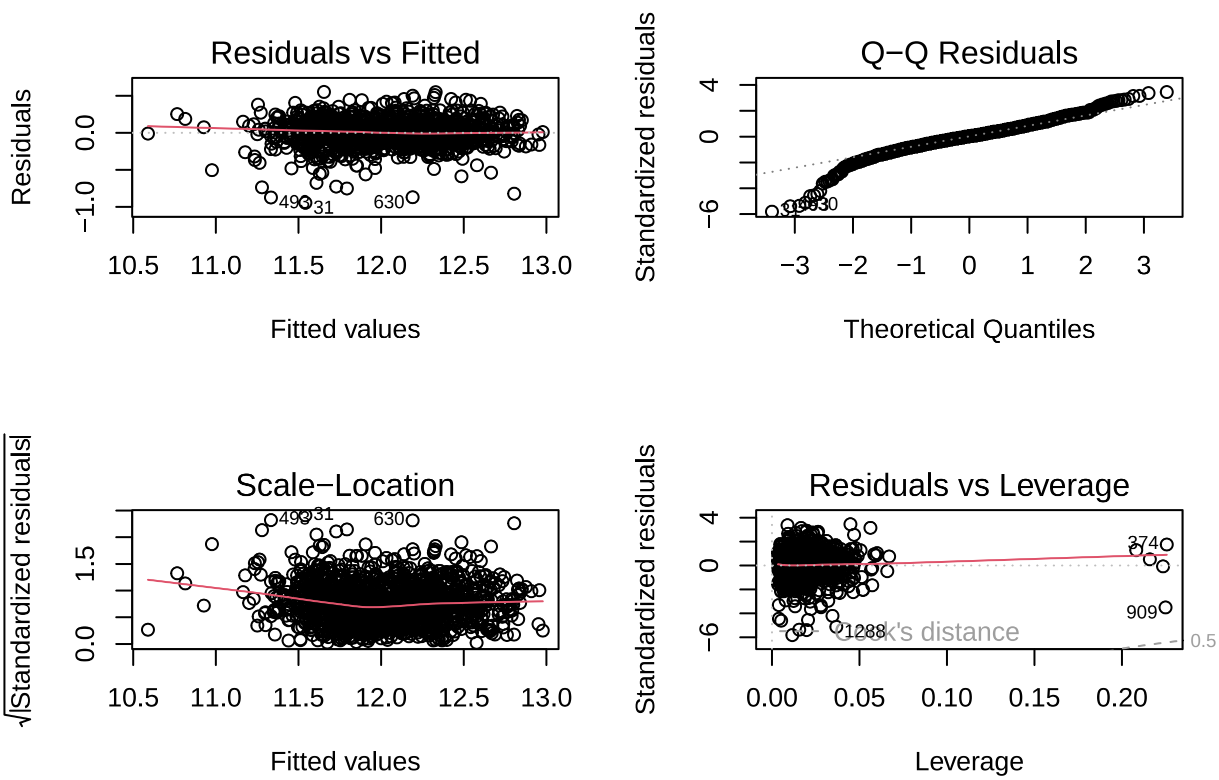
```
ks_test_result
```

```
##
##  Asymptotic two-sample Kolmogorov-Smirnov test
##
## data:  test_price[, 1] and df$SalePrice
## D = 0.042704, p-value = 0.1416
## alternative hypothesis: two-sided
```

The Kolmogorov-Smirnov test shows that predicted and real distributions of SalePrice should be assumed to be different.

Finally, we will check the normality of the residuals.

```
par(mfrow=c(2,2))
plot(m16.2)
```




```
shapiro.test(m16.2$residuals)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  m16.2$residuals
## W = 0.95792, p-value < 2.2e-16
```

Residuals don't follow a normal distribution, so the model won't give very accurate results. Nevertheless, we are happy with our results, so we will not apply any more changes.

13. Model interpretation

First, let us remember the model we have obtained: log(SalePrice) ~ sqrt(LotArea) + YearBuilt + MasVnrArea + BedroomAbvGr + Fireplaces + sqrt(WoodDeckSF) + OpenPorchSF + OverallQual + BsmtQual + KitchenQual + Neighborhood + GarageFinish.

We are modeling the logarithm of SalePrice. That is, an increase of one unit in any of the predictors (except for LotArea and WoodDeckSF) causes the price of the sale to be multiplied by the number e. All the predictors we are using make sense intuitively: the area of the lot, the masonry veneer, the wood deck and the open porch, the amount of bedrooms above ground and fireplaces, the overall quality but also that of the basement and the kitchen, the interior finish of the garage, the dwelling neighborhood's wealth and the year it was built. The area of the lot and the wood deck appear with an exponent of 1/2 in the model, which means that the slope of their contribution to log(SalePrice) is lower than that of the other terms for values larger than 1/4.

In total, our model predictors are composed of 7 numerical features and 5 categorical variables, with three transformations and no interactions.