SIM Project 2. Model fitting

Adrià Casanova Víctor Garcia Zhengyong Ji

November, 19th 2023

Table of Contents

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.

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   
## -284830 -16703 -822 16153 227181   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -1.729e+06 1.307e+05 -13.231 < 2e-16 \*\*\*  
## LotFrontage 1.060e+02 2.944e+01 3.602 0.000327 \*\*\*  
## LotArea 1.706e+00 3.137e-01 5.437 6.36e-08 \*\*\*  
## YearBuilt 3.881e+02 6.148e+01 6.313 3.64e-10 \*\*\*  
## YearRemodAdd 4.972e+02 6.108e+01 8.139 8.55e-16 \*\*\*  
## MasVnrArea 2.017e+01 7.339e+00 2.749 0.006061 \*\*   
## BsmtFinSF1 3.190e+01 5.585e+00 5.712 1.36e-08 \*\*\*  
## BsmtUnfSF 1.009e+01 5.642e+00 1.788 0.074017 .   
## TotalBsmtSF 2.907e+01 6.713e+00 4.330 1.59e-05 \*\*\*  
## X1stFlrSF 6.815e+01 1.120e+01 6.086 1.48e-09 \*\*\*  
## X2ndFlrSF 8.081e+01 1.043e+01 7.744 1.81e-14 \*\*\*  
## GrLivArea -1.388e+00 1.049e+01 -0.132 0.894698   
## BsmtFullBath -1.966e+03 2.485e+03 -0.791 0.429079   
## FullBath -2.732e+03 2.793e+03 -0.978 0.328218   
## HalfBath -4.013e+03 2.677e+03 -1.499 0.134096   
## BedroomAbvGr -1.367e+04 1.664e+03 -8.212 4.81e-16 \*\*\*  
## TotRmsAbvGrd 1.917e+03 1.193e+03 1.607 0.108168   
## Fireplaces 7.637e+03 1.732e+03 4.410 1.11e-05 \*\*\*  
## GarageYrBlt -8.597e+00 7.634e+01 -0.113 0.910357   
## GarageCars 4.923e+03 2.858e+03 1.723 0.085186 .   
## GarageArea 2.454e+01 1.026e+01 2.393 0.016845 \*   
## WoodDeckSF 2.125e+01 8.173e+00 2.600 0.009413 \*\*   
## OpenPorchSF 4.559e+01 1.812e+01 2.516 0.011978 \*   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 34930 on 1437 degrees of freedom  
## Multiple R-squared: 0.8096, Adjusted R-squared: 0.8067   
## F-statistic: 277.7 on 22 and 1437 DF, p-value: < 2.2e-16

vif(m0)

## LotFrontage LotArea YearBuilt YearRemodAdd MasVnrArea BsmtFinSF1   
## 1.152603 1.497042 4.123365 1.901884 1.320233 7.068653   
## BsmtUnfSF TotalBsmtSF X1stFlrSF X2ndFlrSF GrLivArea BsmtFullBath   
## 7.431487 8.834991 20.368811 24.811318 32.823752 1.988987   
## FullBath HalfBath BedroomAbvGr TotRmsAbvGrd Fireplaces GarageYrBlt   
## 2.832002 2.167450 2.147125 4.433981 1.490638 4.237834   
## GarageCars GarageArea WoodDeckSF OpenPorchSF   
## 5.455675 5.549918 1.178558 1.233308

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.

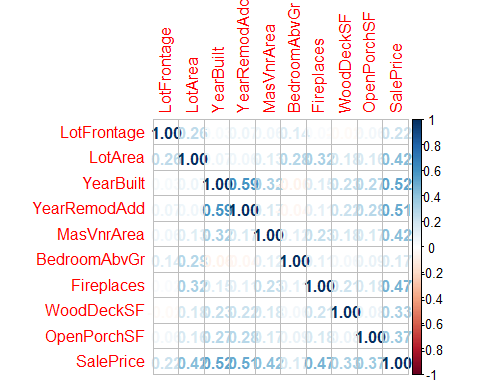
# Let's store the indices of the variables with at least one star in the lm and vif<5  
id\_num\_star1 = c(1:5,15,17,21:23)  
df\_num1 <- df\_num[, id\_num\_star1]  
# And build a new model only with significance features  
m1 = lm(SalePrice ~., data=df\_num1)  
summary(m1)

##   
## Call:  
## lm(formula = SalePrice ~ ., data = df\_num1)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -189053 -27080 -4750 19843 434071   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -2.976e+06 1.355e+05 -21.960 < 2e-16 \*\*\*  
## LotFrontage 2.530e+02 4.024e+01 6.287 4.27e-10 \*\*\*  
## LotArea 4.696e+00 4.103e-01 11.446 < 2e-16 \*\*\*  
## YearBuilt 5.591e+02 5.581e+01 10.018 < 2e-16 \*\*\*  
## YearRemodAdd 9.775e+02 7.879e+01 12.405 < 2e-16 \*\*\*  
## MasVnrArea 9.444e+01 9.795e+00 9.642 < 2e-16 \*\*\*  
## BedroomAbvGr 5.369e+03 1.691e+03 3.175 0.00153 \*\*   
## Fireplaces 3.096e+04 2.183e+03 14.184 < 2e-16 \*\*\*  
## WoodDeckSF 6.130e+01 1.124e+01 5.455 5.75e-08 \*\*\*  
## OpenPorchSF 1.598e+02 2.458e+01 6.499 1.11e-10 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 48990 on 1450 degrees of freedom  
## Multiple R-squared: 0.622, Adjusted R-squared: 0.6197   
## F-statistic: 265.1 on 9 and 1450 DF, p-value: < 2.2e-16

vif(m1)

## LotFrontage LotArea YearBuilt YearRemodAdd MasVnrArea BedroomAbvGr   
## 1.094625 1.301887 1.727062 1.608513 1.195294 1.126313   
## Fireplaces WoodDeckSF OpenPorchSF   
## 1.203511 1.132628 1.153802

# 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 between them, and “YearBuilt” is more correlated to our target SalePrice. Hence, we remove YearRemodAdd in the next model.

# 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   
## -183896 -29373 -5718 21404 429918   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -1.759e+06 9.831e+04 -17.894 < 2e-16 \*\*\*  
## LotFrontage 2.879e+02 4.220e+01 6.823 1.30e-11 \*\*\*  
## LotArea 4.570e+00 4.312e-01 10.598 < 2e-16 \*\*\*  
## YearBuilt 9.247e+02 4.983e+01 18.559 < 2e-16 \*\*\*  
## MasVnrArea 8.807e+01 1.028e+01 8.564 < 2e-16 \*\*\*  
## BedroomAbvGr 4.819e+03 1.777e+03 2.712 0.00677 \*\*   
## Fireplaces 3.121e+04 2.295e+03 13.599 < 2e-16 \*\*\*  
## WoodDeckSF 7.673e+01 1.174e+01 6.534 8.83e-11 \*\*\*  
## OpenPorchSF 2.072e+02 2.553e+01 8.116 1.02e-15 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 51510 on 1451 degrees of freedom  
## Multiple R-squared: 0.5819, Adjusted R-squared: 0.5796   
## F-statistic: 252.4 on 8 and 1451 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/>.

Anova(m2)

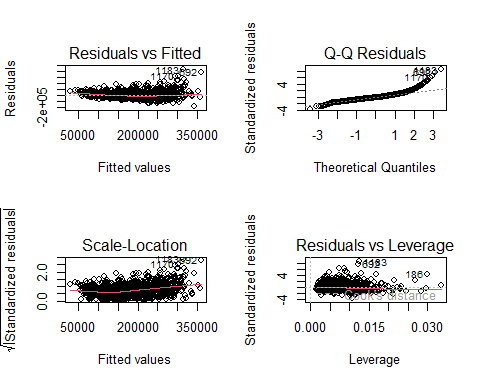
## Anova Table (Type II tests)  
##   
## Response: SalePrice  
## Sum Sq Df F value Pr(>F)   
## LotFrontage 1.2352e+11 1 46.5566 1.303e-11 \*\*\*  
## LotArea 2.9800e+11 1 112.3164 < 2.2e-16 \*\*\*  
## YearBuilt 9.1384e+11 1 344.4318 < 2.2e-16 \*\*\*  
## MasVnrArea 1.9460e+11 1 73.3474 < 2.2e-16 \*\*\*  
## BedroomAbvGr 1.9512e+10 1 7.3542 0.00677 \*\*   
## Fireplaces 4.9067e+11 1 184.9372 < 2.2e-16 \*\*\*  
## WoodDeckSF 1.1328e+11 1 42.6957 8.829e-11 \*\*\*  
## OpenPorchSF 1.7476e+11 1 65.8677 1.019e-15 \*\*\*  
## Residuals 3.8498e+12 1451   
## ---  
## 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

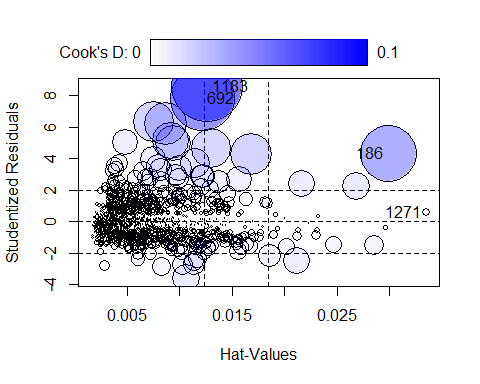
First, let us plot the residuals of m2 to be able to compare them with the next iterations of the model.

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)

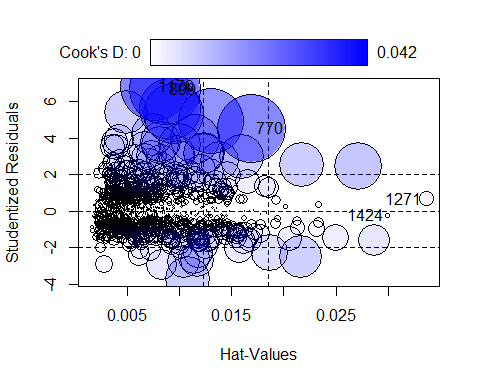


## StudRes Hat CookD  
## 186 4.3264547 0.02991534 0.063362899  
## 692 7.8641613 0.01219078 0.081391863  
## 1183 8.6087259 0.01264601 0.100407718  
## 1271 0.5819775 0.03351270 0.001305513

# Calculate D's threshold  
D\_thresh <- 2/sqrt(dim(df\_num2)[1]); D\_thresh

## [1] 0.05234239

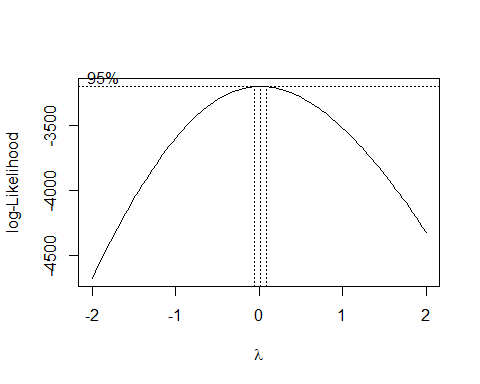
#Remove the points and fit the model again  
influent <- c(1183, 692, 186)  
  
df <- df[-influent,]  
df\_num <- df[, which(sapply(df, is.numeric))]  
df\_num2 <- df\_num[, id\_num\_star2]  
m2 = lm(SalePrice ~., data=df\_num2)  
  
influencePlot(m2)



## StudRes Hat CookD  
## 770 4.5443947 0.016903181 0.0389248857  
## 899 6.6815842 0.008642834 0.0419803351  
## 1170 6.8469664 0.007587455 0.0386020474  
## 1271 0.6859165 0.033627929 0.0018197619  
## 1424 -0.2309732 0.029944826 0.0001831007

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

boxcox(m2)



# 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   
## -1.09825 -0.14427 -0.00563 0.12877 0.94288   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -1.120e-02 4.511e-01 -0.025 0.98   
## LotFrontage 9.145e-04 1.934e-04 4.728 2.48e-06 \*\*\*  
## LotArea 2.117e-05 1.980e-06 10.697 < 2e-16 \*\*\*  
## YearBuilt 5.794e-03 2.287e-04 25.337 < 2e-16 \*\*\*  
## MasVnrArea 3.251e-04 4.705e-05 6.910 7.25e-12 \*\*\*  
## BedroomAbvGr 5.230e-02 8.124e-03 6.438 1.64e-10 \*\*\*  
## Fireplaces 1.657e-01 1.051e-02 15.766 < 2e-16 \*\*\*  
## WoodDeckSF 3.610e-04 5.372e-05 6.720 2.61e-11 \*\*\*  
## OpenPorchSF 9.952e-04 1.174e-04 8.478 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.2354 on 1448 degrees of freedom  
## Multiple R-squared: 0.6468, Adjusted R-squared: 0.6449   
## F-statistic: 331.5 on 8 and 1448 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

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.: 7540 1st Qu.:1954 1st Qu.: 0.00   
## Median : 63.00 Median : 9416 Median :1973 Median : 0.00   
## Mean : 57.16 Mean : 9524 Mean :1971 Mean : 91.67   
## 3rd Qu.: 79.00 3rd Qu.:11345 3rd Qu.:2000 3rd Qu.:162.00   
## Max. :182.00 Max. :23595 Max. :2010 Max. :664.00   
## BedroomAbvGr Fireplaces WoodDeckSF OpenPorchSF   
## Min. :0.000001 Min. :0.000001 Min. : 0.0 Min. : 0.00   
## 1st Qu.:2.000000 1st Qu.:0.000001 1st Qu.: 0.0 1st Qu.: 0.00   
## Median :3.000000 Median :1.000000 Median : 0.0 Median : 24.00   
## Mean :2.861290 Mean :0.610158 Mean : 92.7 Mean : 42.78   
## 3rd Qu.:3.000000 3rd Qu.:1.000000 3rd Qu.:168.0 3rd Qu.: 66.00   
## Max. :6.000000 Max. :3.000000 Max. :670.0 Max. :267.00   
## SalePrice   
## Min. : 34900   
## 1st Qu.:129900   
## Median :163000   
## Mean :179938   
## 3rd Qu.:213500   
## Max. :625000

boxTidwell(log(SalePrice) ~ LotArea+YearBuilt+MasVnrArea, data = df\_num2)

## MLE of lambda Score Statistic (t) Pr(>|t|)   
## LotArea 0.50589 -4.0795 4.759e-05 \*\*\*  
## YearBuilt 69.59853 14.9243 < 2.2e-16 \*\*\*  
## MasVnrArea 0.99009 -0.1764 0.86   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## iterations = 5   
##   
## Score test for null hypothesis that all lambdas = 1:  
## F = 79.82, df = 3 and 1450, Pr(>F) = < 2.2e-16

# 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|)   
## -2.8984 11.274 < 2.2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## iterations = 26

# 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|)  
## 1.528 0.7048 0.481  
##   
## iterations = 26

# Too large p-value  
boxTidwell(log(SalePrice)~Fireplaces, data =df\_num2)

## MLE of lambda Score Statistic (t) Pr(>|t|)   
## 0.24931 -7.6471 3.717e-14 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## iterations = 2

# We apply log() to Fireplaces  
boxTidwell(log(SalePrice)~WoodDeckSF, data = df\_num2)

## MLE of lambda Score Statistic (t) Pr(>|t|)   
## 0.52909 -4.9103 1.012e-06 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## iterations = 7

# 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|)   
## -8.7203 -12 < 2.2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## iterations = 26

# Too small lambda

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

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   
## -1.10570 -0.14470 -0.00927 0.13002 0.87600   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 5.955e-01 4.629e-01 1.286 0.198506   
## LotFrontage 7.427e-04 1.950e-04 3.809 0.000145 \*\*\*  
## sqrt(LotArea) 4.345e-03 3.689e-04 11.777 < 2e-16 \*\*\*  
## YearBuilt 5.484e-03 2.337e-04 23.471 < 2e-16 \*\*\*  
## MasVnrArea 3.457e-04 4.712e-05 7.335 3.68e-13 \*\*\*  
## BedroomAbvGr 4.847e-02 8.193e-03 5.916 4.10e-09 \*\*\*  
## log(Fireplaces) 1.480e-02 9.829e-04 15.058 < 2e-16 \*\*\*  
## sqrt(WoodDeckSF) 6.073e-03 9.215e-04 6.591 6.11e-11 \*\*\*  
## OpenPorchSF 1.004e-03 1.178e-04 8.525 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.2361 on 1448 degrees of freedom  
## Multiple R-squared: 0.6446, Adjusted R-squared: 0.6426   
## F-statistic: 328.3 on 8 and 1448 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 -16.599825  
## m4 10 -7.344876

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)

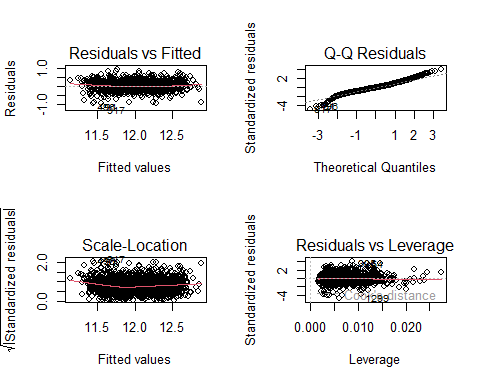
## df BIC  
## m4 10 -7.344876  
## m5 10 1.508573  
## m6 10 -26.417016  
## m7 10 -6.294640  
## m8 10 -24.705618  
## m9 10 2.411503  
## m10 10 -18.163916

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.

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 -16.59982  
## m11 10 -26.41669

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)))

1. 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.

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   
## -0.94527 -0.08542 0.00503 0.09201 0.55128   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 9.065e+00 5.168e-01 17.539 < 2e-16 \*\*\*  
## sqrt(LotArea) 3.614e-03 2.581e-04 14.002 < 2e-16 \*\*\*  
## YearBuilt 1.239e-03 2.558e-04 4.841 1.43e-06 \*\*\*  
## MasVnrArea 1.485e-04 3.440e-05 4.316 1.70e-05 \*\*\*  
## BedroomAbvGr 5.447e-02 5.992e-03 9.090 < 2e-16 \*\*\*  
## Fireplaces 8.099e-02 7.800e-03 10.384 < 2e-16 \*\*\*  
## sqrt(WoodDeckSF) 2.996e-03 6.588e-04 4.548 5.87e-06 \*\*\*  
## OpenPorchSF 3.190e-04 8.559e-05 3.728 0.000201 \*\*\*  
## OverallQualGood 2.749e-01 2.140e-02 12.846 < 2e-16 \*\*\*  
## OverallQualModerate 1.360e-01 1.689e-02 8.056 1.65e-15 \*\*\*  
## OverallQualVBad -4.854e-01 7.886e-02 -6.155 9.70e-10 \*\*\*  
## OverallQualVGood 4.193e-01 3.798e-02 11.039 < 2e-16 \*\*\*  
## BsmtQualFa -1.584e-01 3.962e-02 -3.999 6.68e-05 \*\*\*  
## BsmtQualGd -8.603e-02 2.166e-02 -3.971 7.51e-05 \*\*\*  
## BsmtQualNBsmt -2.649e-01 3.759e-02 -7.049 2.79e-12 \*\*\*  
## BsmtQualTA -1.341e-01 2.568e-02 -5.221 2.05e-07 \*\*\*  
## KitchenQualFa -2.170e-01 3.791e-02 -5.724 1.26e-08 \*\*\*  
## KitchenQualGd -7.900e-02 2.315e-02 -3.412 0.000663 \*\*\*  
## KitchenQualTA -1.746e-01 2.485e-02 -7.027 3.24e-12 \*\*\*  
## NeighborhoodPoor -3.994e-02 1.289e-02 -3.099 0.001981 \*\*   
## NeighborhoodRich 1.354e-01 1.331e-02 10.168 < 2e-16 \*\*\*  
## GarageFinishNGar -1.900e-01 2.437e-02 -7.797 1.21e-14 \*\*\*  
## GarageFinishRFn -1.761e-02 1.257e-02 -1.401 0.161522   
## GarageFinishUnf -6.589e-02 1.439e-02 -4.578 5.09e-06 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.1657 on 1433 degrees of freedom  
## Multiple R-squared: 0.8268, Adjusted R-squared: 0.824   
## F-statistic: 297.4 on 23 and 1433 DF, p-value: < 2.2e-16

BIC(m16.2, m11, m1)

## Warning in BIC.default(m16.2, m11, m1): 模型拟合时没有用同样大小的数据集

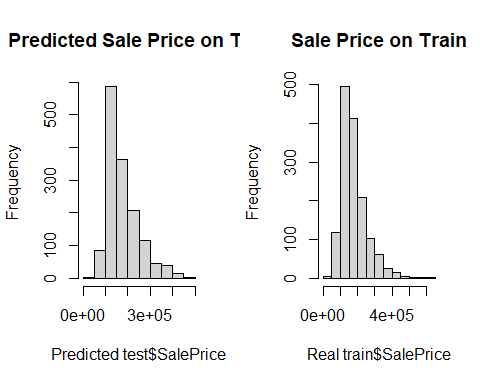
## df BIC  
## m16.2 25 -945.28326  
## m11 10 -26.41669  
## m1 11 35747.73550

# 12. Model validation

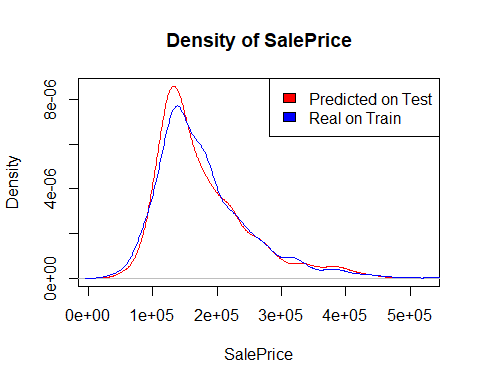
We predict the SalePrice on the test dataset and compare its distribution with the original one in train.

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")



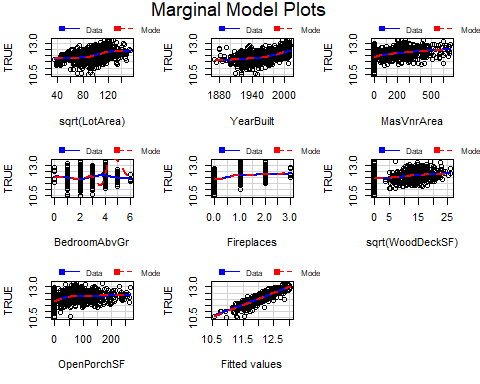
par(mfrow=c(1,1))  
plot(density(test\_price[,1]), col="red", main = "Density of SalePrice",  
 xlab = "SalePrice")  
lines(density(df$SalePrice), col="blue")  
legend("topright",fill = c("red", "blue"), c("Predicted on Test","Real on Train"))



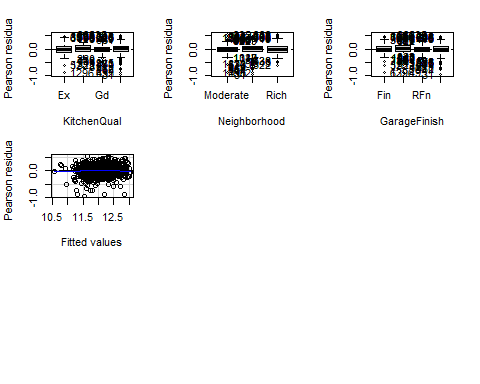
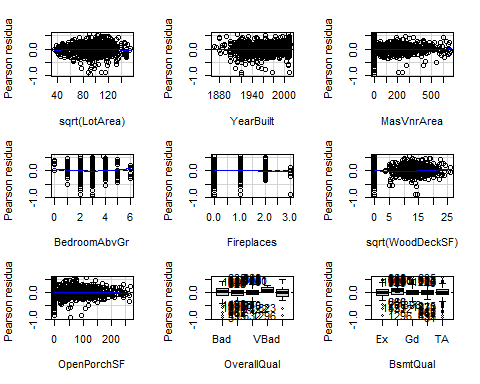
As can be seen in the previous plots, the real and the predicted distributions of SalePrice are similar, but not identical. This was exactly our goal, since both test and train come from the same population and we wanted to avoid overfitting.

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

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



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



## Test stat Pr(>|Test stat|)   
## sqrt(LotArea) -2.2806 0.022718 \*   
## YearBuilt 0.3192 0.749596   
## MasVnrArea 0.3619 0.717489   
## BedroomAbvGr 2.8178 0.004902 \*\*  
## Fireplaces -1.0510 0.293417   
## sqrt(WoodDeckSF) 2.2279 0.026043 \*   
## OpenPorchSF -1.0755 0.282343   
## OverallQual   
## BsmtQual   
## KitchenQual   
## Neighborhood   
## GarageFinish   
## Tukey test -1.0993 0.271638   
## ---  
## 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): 并列的时候P-值将近似

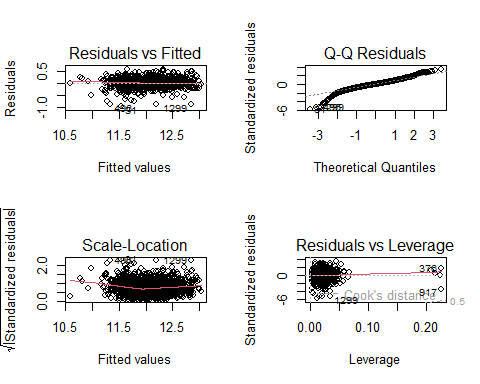
ks\_test\_result

##   
## Asymptotic two-sample Kolmogorov-Smirnov test  
##   
## data: test\_price[, 1] and df$SalePrice  
## D = 0.045956, p-value = 0.09198  
## 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 = 0.95837, 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.