What determines the value of a house?

```
knitr::opts_chunk$set(echo = TRUE)
```

```
####Libaries####

library(tidyverse)
library(ggplot2)
library(caret)     #createDataPartition
library(DAAG)      #CVlm
library(car)       #leverage.plots
library(lindia)    #gg_cooksd
library(gridExtra) #grid.arrange
library(kableExtra)
library(olsrr)     #ols_step_forward_aic
```

```
####Import Data####

getwd()
```

```
## [1] "C:/Users/dnguy/OneDrive/Desktop/epicduy.github.io/house-prediction"
```

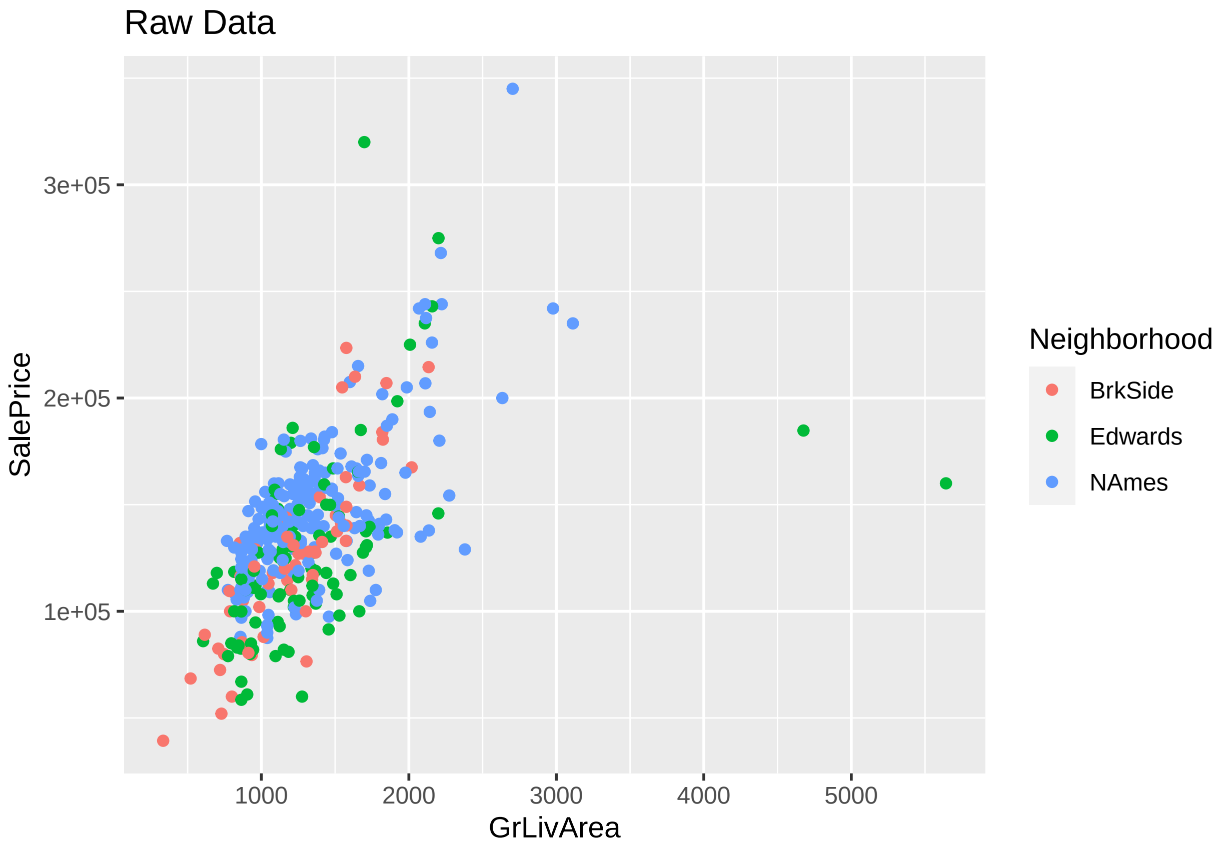
```
train <- read.csv("train.csv")
test <- read.csv("test.csv")
```

```
####Analysis 1 Wrangle Data####
df <- dplyr::filter(train, Neighborhood =="Edwards" | Neighborhood =="NAmes" |
                      Neighborhood == "BrkSide")
df$Neighborhood <- as.factor(df$Neighborhood)
str(df)
```

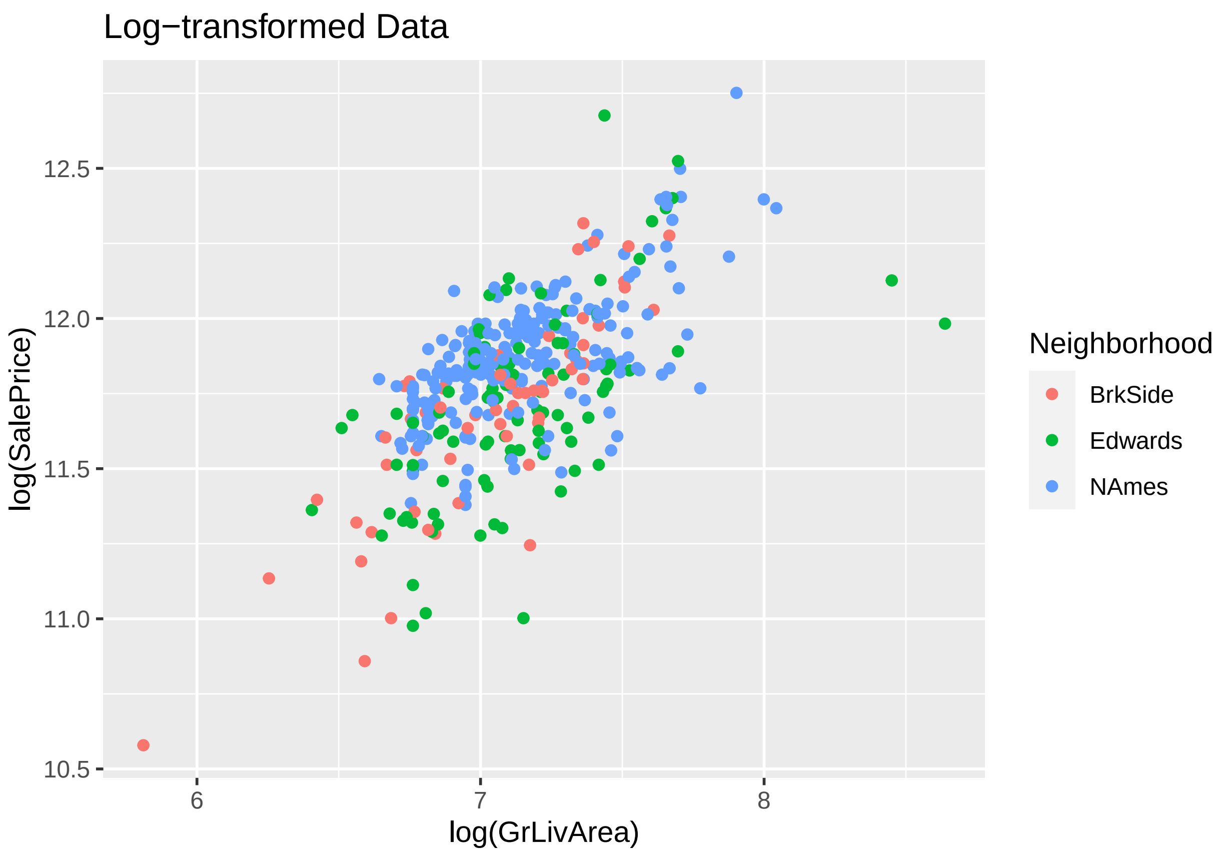
```
## ’data.frame':    383 obs. of  81 variables:
##  $ Id           : int  10 15 16 17 20 27 29 30 34 38 ...
##  $ MSSubClass   : int  190 20 45 20 20 20 20 30 20 20 ...
##  $ MSZoning     : chr  "RL" "RL" "RM" "RL" ...
##  $ LotFrontage  : int  50 NA 51 NA 70 60 47 60 70 74 ...
##  $ LotArea      : int  7420 10920 6120 11241 7560 7200 16321 6324 10552 8532 ...
##  $ Street       : chr  "Pave" "Pave" "Pave" "Pave" ...
##  $ Alley        : chr  NA NA NA NA ...
##  $ LotShape     : chr  "Reg" "IR1" "Reg" "IR1" ...
##  $ LandContour  : chr  "Lvl" "Lvl" "Lvl" "Lvl" ...
##  $ Utilities    : chr  "AllPub" "AllPub" "AllPub" "AllPub" ...
##  $ LotConfig    : chr  "Corner" "Corner" "Corner" "CulDSac" ...
##  $ LandSlope    : chr  "Gtl" "Gtl" "Gtl" "Gtl" ...
##  $ Neighborhood : Factor w/ 3 levels "BrkSide","Edwards",..: 1 3 1 3 3 3 3 1 3 3 ...
##  $ Condition1   : chr  "Artery" "Norm" "Norm" "Norm" ...
##  $ Condition2   : chr  "Artery" "Norm" "Norm" "Norm" ...
##  $ BldgType     : chr  "2fmCon" "1Fam" "1Fam" "1Fam" ...
##  $ HouseStyle   : chr  "1.5Unf" "1Story" "1.5Unf" "1Story" ...
##  $ OverallQual  : int  5 6 7 6 5 5 5 4 5 5 ...
##  $ OverallCond  : chr  "6" "5" "8" "7" ...
##  $ YearBuilt    : int  1939 1960 1929 1970 1958 1951 1957 1927 1959 1954 ...
##  $ YearRemodAdd : int  1950 1960 2001 1970 1965 2000 1997 1950 1959 1990 ...
##  $ RoofStyle    : chr  "Gable" "Hip" "Gable" "Gable" ...
##  $ RoofMatl     : chr  "CompShg" "CompShg" "CompShg" "CompShg" ...
##  $ Exterior1st  : chr  "MetalSd" "MetalSd" "Wd Sdng" "Wd Sdng" ...
##  $ Exterior2nd  : chr  "MetalSd" "MetalSd" "Wd Sdng" "Wd Sdng" ...
##  $ MasVnrType   : chr  "None" "BrkFace" "None" "BrkFace" ...
##  $ MasVnrArea   : int  0 212 0 180 0 0 0 0 0 650 ...
##  $ ExterQual    : chr  "TA" "TA" "TA" "TA" ...
##  $ ExterCond    : chr  "TA" "TA" "TA" "TA" ...
##  $ Foundation   : chr  "BrkTil" "CBlock" "BrkTil" "CBlock" ...
##  $ BsmtQual     : chr  "TA" "TA" "TA" "TA" ...
##  $ BsmtCond     : chr  "TA" "TA" "TA" "TA" ...
##  $ BsmtExposure : chr  "No" "No" "No" "No" ...
##  $ BsmtFinType1 : chr  "GLQ" "BLQ" "Unf" "ALQ" ...
##  $ BsmtFinSF1   : int  851 733 0 578 504 234 1277 0 1018 1213 ...
##  $ BsmtFinType2 : chr  "Unf" "Unf" "Unf" "Unf" ...
##  $ BsmtFinSF2   : int  0 0 0 0 0 486 0 0 0 0 ...
##  $ BsmtUnfSF    : int  140 520 832 426 525 180 207 520 380 84 ...
##  $ TotalBsmtSF  : int  991 1253 832 1004 1029 900 1484 520 1398 1297 ...
##  $ Heating      : chr  "GasA" "GasA" "GasA" "GasA" ...
##  $ HeatingQC    : chr  "Ex" "TA" "Ex" "Ex" ...
##  $ CentralAir   : chr  "Y" "Y" "Y" "Y" ...
##  $ Electrical   : chr  "SBrkr" "SBrkr" "FuseA" "SBrkr" ...
##  $ X1stFlrSF    : int  1077 1253 854 1004 1339 900 1600 520 1700 1297 ...
##  $ X2ndFlrSF    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ LowQualFinSF : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ GrLivArea    : int  1077 1253 854 1004 1339 900 1600 520 1700 1297 ...
##  $ BsmtFullBath : int  1 1 0 1 0 0 1 0 0 0 ...
##  $ BsmtHalfBath : int  0 0 0 0 0 1 0 0 1 1 ...
##  $ FullBath     : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ HalfBath     : int  0 1 0 0 0 0 0 0 1 0 ...
##  $ BedroomAbvGr : int  2 2 2 2 3 3 2 1 4 3 ...
##  $ KitchenAbvGr : int  2 1 1 1 1 1 1 1 1 1 ...
##  $ KitchenQual  : chr  "TA" "TA" "TA" "TA" ...
##  $ TotRmsAbvGrd : int  5 5 5 5 6 5 6 4 6 5 ...
##  $ Functional   : chr  "Typ" "Typ" "Typ" "Typ" ...
##  $ Fireplaces   : int  2 1 0 1 0 0 2 0 1 1 ...
##  $ FireplaceQu  : chr  "TA" "Fa" NA "TA" ...
##  $ GarageType   : chr  "Attchd" "Attchd" "Detchd" "Attchd" ...
##  $ GarageYrBlt  : int  1939 1960 1991 1970 1958 2005 1957 1920 1959 1954 ...
##  $ GarageFinish : chr  "RFn" "RFn" "Unf" "Fin" ...
##  $ GarageCars   : int  1 1 2 2 1 2 1 1 2 2 ...
##  $ GarageArea   : int  205 352 576 480 294 576 319 240 447 498 ...
##  $ GarageQual   : chr  "Gd" "TA" "TA" "TA" ...
##  $ GarageCond   : chr  "TA" "TA" "TA" "TA" ...
##  $ PavedDrive   : chr  "Y" "Y" "Y" "Y" ...
##  $ WoodDeckSF   : int  0 0 48 0 0 222 288 49 0 0 ...
##  $ OpenPorchSF  : int  4 213 112 0 0 32 258 0 38 0 ...
##  $ EnclosedPorch: int  0 176 0 0 0 0 0 87 0 0 ...
##  $ X3SsnPorch   : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ ScreenPorch  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ PoolArea     : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ PoolQC       : chr  NA NA NA NA ...
##  $ Fence        : chr  NA "GdWo" "GdPrv" NA ...
##  $ MiscFeature  : chr  NA NA NA "Shed" ...
##  $ MiscVal      : int  0 0 0 700 0 0 0 0 0 0 ...
##  $ MoSold       : int  1 5 7 3 5 5 12 5 4 10 ...
##  $ YrSold       : int  2008 2008 2007 2010 2009 2010 2006 2008 2010 2009 ...
##  $ SaleType     : chr  "WD" "WD" "WD" "WD" ...
##  $ SaleCondition: chr  "Normal" "Normal" "Normal" "Normal" ...
##  $ SalePrice    : int  118000 157000 132000 149000 139000 134800 207500 68500 165500 153000 ...
```

Raw vs Log-transformed Scatterplots

```r
####Analysis 1 EDA####

#Normal Data Scatterplot
ggplot(df, aes(x = GrLivArea, y = SalePrice, color = Neighborhood)) +
  geom_point() +
  ggtitle("Raw Data")
```



```r
#Log-transformed Data Scatterplot
ggplot(df, aes(x = log(GrLivArea), y = log(SalePrice), color = Neighborhood)) +
  geom_point() +
  ggtitle("Log-transformed Data")
```



How does grade living area relate to sales price?

From an initial model, we can deduce from the R-Squared values below that, the log-transformed grade living area is about 42% effective at explaining log-transformed sales price of a house.

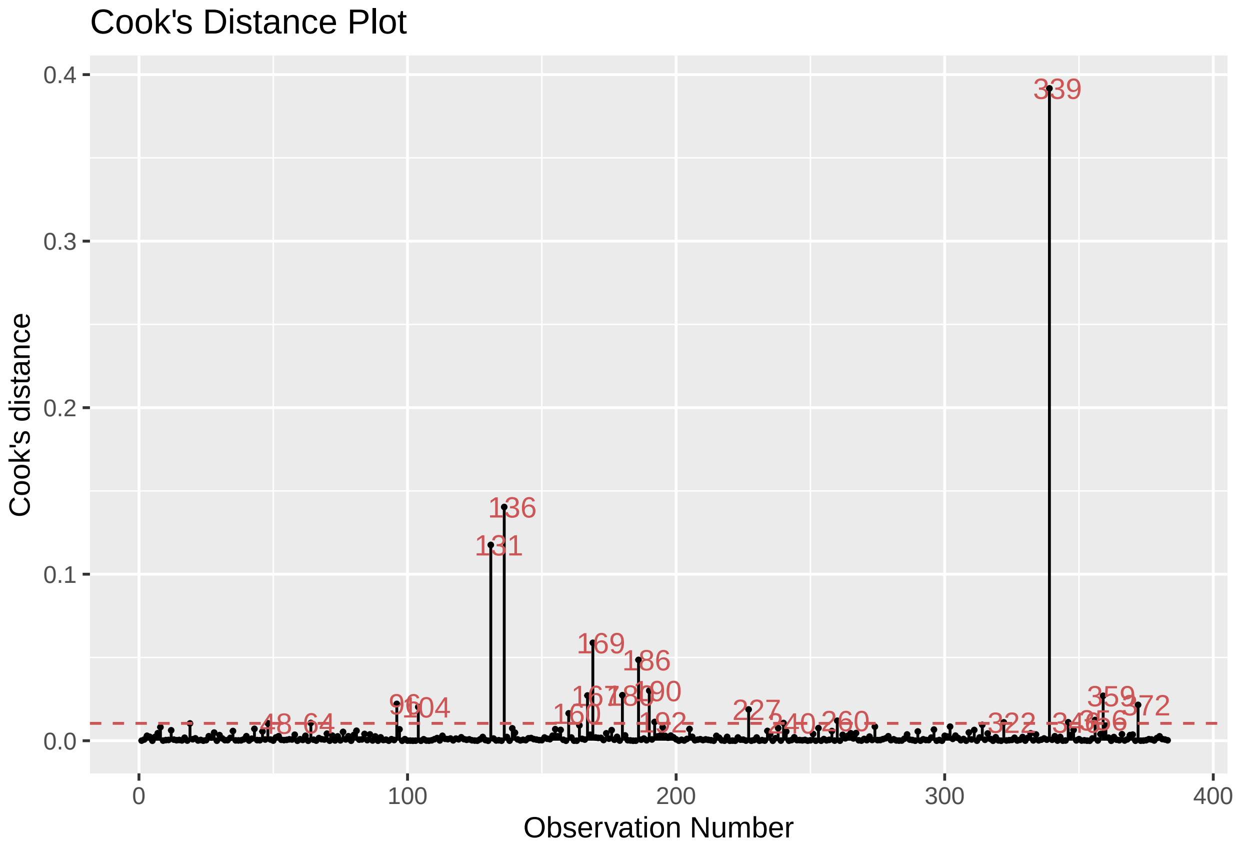
$$log(SalePrice) = B0 + B1log(GrLivArea)$$

```
model0 <- lm(log(SalePrice) ~ log(GrLivArea), data = df)
paste(summary(model0)$r.squared, "  | ", summary(model0)$adj.r.squared)
```

```
## [1] "0.420370141901713   |  0.418848803691481"
```

We can further hone our dataset with influential points analysis.

```
#Cook's Distance Plot
gg_cooksd(model0)
```



```
#Identify influential points
#From the above graph, those points are greater than 0.1 Cook's D
as.numeric(names(cooks.distance(model0))[(cooks.distance(model0) > 0.1)])
```

```
## [1] 131 136 339
```

```
#And they are 131, 136, 339

#Start removing outliers one by one and a combo of outliers
#And find the most desired R-Squared and Adj R-Squared
A1Data01 = df[-131,]
model01 <- lm(log(SalePrice) ~ log(GrLivArea), data = A1Data01)
paste(summary(model01)$r.squared, "  |  ", summary(model01)$adj.r.squared)
```

```
## [1] "0.424963702994452   |  0.423450449581279"
```

```
A1Data02 = df[-136,]
model02 <- lm(log(SalePrice) ~ log(GrLivArea), data = A1Data02)
paste(summary(model02)$r.squared, "  |  ", summary(model02)$adj.r.squared)
```

```
## [1] "0.397243732877509   |  0.395657532174555"
```

```r
A1Data03 = df[-339,]
model03 <- lm(log(SalePrice) ~ log(GrLivArea), data = A1Data03)
paste(summary(model03)$r.squared, "  | ", summary(model03)$adj.r.squared)
```

```
## [1] "0.436950647337104   |  0.435468938514307"
```

```r
A1Data04 = df[-c(131, 136, 339),]
model04 <- lm(log(SalePrice) ~ log(GrLivArea), data = A1Data04)
paste(summary(model04)$r.squared, "  | ", summary(model04)$adj.r.squared)
```

```
## [1] "0.419021718033358   |  0.417484738451436"
```

```r
A1Data05 = df[-c(131, 136),]
model05 <- lm(log(SalePrice) ~ log(GrLivArea), data = A1Data05)
paste(summary(model05)$r.squared, "  | ", summary(model05)$adj.r.squared)
```

```
## [1] "0.401337834721517   |  0.39975825117197"
```

```r
A1Data06 = df[-c(131, 339),]
model06 <- lm(log(SalePrice) ~ log(GrLivArea), data = A1Data06)

#The most desired is A1Data06 and model06
A1Data = A1Data06
model1 = model06

#str(A1Data)
#str(model1)
```

Observations 131 and 136 are then removed from the dataset.

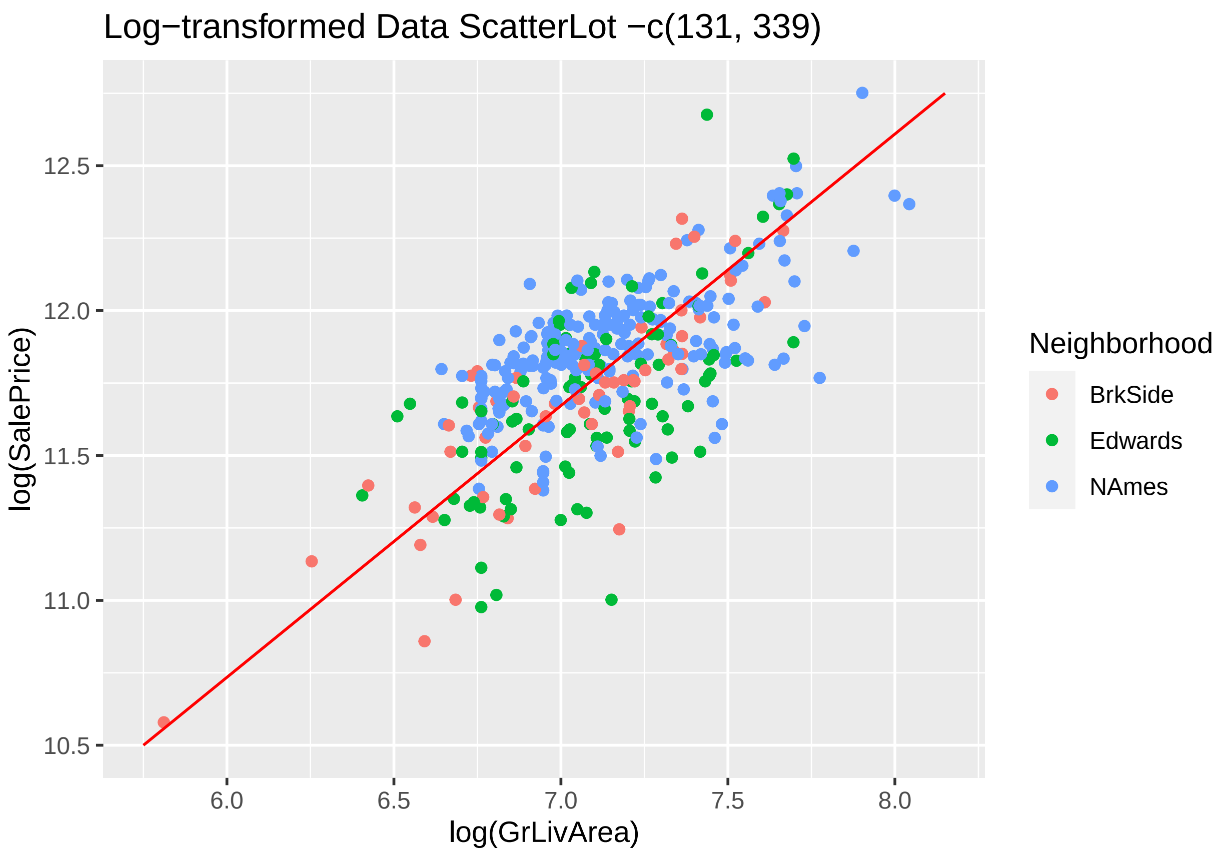
The R-Square values of our new model proves that it performs slighty better, with grade living area effectively explaining 44% of sales prices of houses.

```r
paste(summary(model06)$r.squared, "  | ", summary(model06)$adj.r.squared)
```

```
## [1] "0.443011621633085   |  0.44154199530494"
```

The scatters on our new plot now appears more normalized, judging by the arbitrary diagonal red line.

```r
#Log-transformed Data Scatterplot without indexes 131 and 339
ggplot(A1Data, aes(x = log(GrLivArea), y = log(SalePrice), color = Neighborhood)) +
  geom_point() +
  annotate(geom = "segment", x = 5.75, y = 10.5, xend = 8.15, yend = 12.75, color = "red") +
  ggtitle("Log-transformed Data ScatterLot -c(131, 339)")
```



What if neighborhoods are included?

The R-Squared values below proves that our full model performs the best, with Neighborhood added as an additive.

$$log(SalePrice) = B0 + B1log(GrLivArea) + B2Edwards +$$
$$B3NAmes + B4Edwards * log(GrLivArea) + B5NAmes * log(GrLivArea)$$

```
####Model 2 Modeling####
model21 <- lm(log(SalePrice) ~ log(GrLivArea) + as.factor(Neighborhood), data = A1Data)
paste(summary(model21)$r.squared, "   | ", summary(model21)$adj.r.squared)
```

```
## [1] "0.504137828818283    |  0.500191975997209"
```

```
#"BrkSide" is read first so it was used as reference

model22 <- lm(log(SalePrice) ~ log(GrLivArea) + as.factor(Neighborhood) +
                as.factor(Neighborhood)*log(GrLivArea), data = A1Data)
#model22 <- lm(log(SalePrice) ~ log(GrLivArea) * as.factor(Neighborhood), data = A1Data)
paste(summary(model22)$r.squared, "   | ", summary(model22)$adj.r.squared)
```

```
## [1] "0.527926654806879    |  0.521632343537637"
```

```
press(model22)
```

```
## [1] 13.94807
```

```
#The most desired is the full model
model2 = model22
paste(summary(model2)$r.squared, "  | ", summary(model2)$adj.r.squared)
```

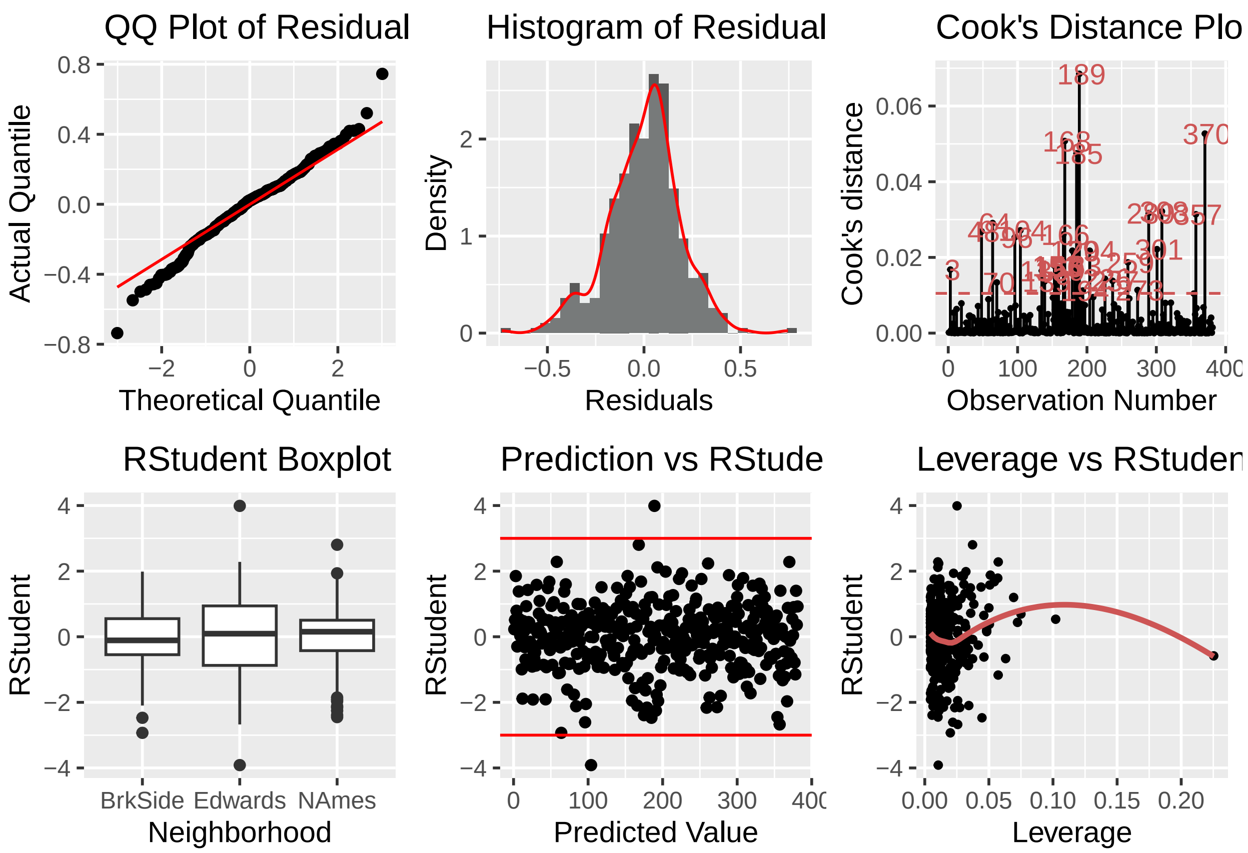
```
## [1] "0.527926654806879   |  0.521632343537637"
```

```
#press(model2)
```

Final Model Assumptions

```
####Model 2 EDA####

#Residuals QQ Plot
residuals = resid(model2)
p1 = ggplot(A1Data, aes(sample = residuals)) +
  geom_qq() +
  geom_qq_line(color = "red") +
  labs(title = "QQ Plot of Residuals", x = "Theoretical Quantile", y = "Actual Quantile")

#Residuals Histogram
p2 = ggplot(A1Data, aes(residuals)) +
  geom_histogram(aes(y = ..density..)) +
  geom_density(alpha = .2, color = "red", fill = "azure") +
  labs(title = "Histogram of Residuals", x = "Residuals", y = "Density")

#Cook's Distance Plot
library(lindia)
p3 = gg_cooksd(model2)

stdres2 <- rstandard(model2)
#Neighborhood vs RStudent
p4 = ggplot(A1Data, aes(as.factor(Neighborhood), stdres2)) +
  geom_boxplot() +
  labs(title = "    RStudent Boxplot", x = "Neighborhood", y = "RStudent")

#Standardized Residuals Plot
p5 = ggplot(A1Data, aes(x = seq(stdres2), y = stdres2)) +
  geom_point() +
  geom_hline(yintercept = 3, color = "red") +
  geom_hline(yintercept = -3, color = "red") +
  labs(title = "Prediction vs RStudent", x = "Predicted Value", y = "RStudent")

#Standardized Residuals vs Leverage
p6 = gg_resleverage(model2, method = "loess", se = FALSE, scale.factor = 1) +
  labs(title = "Leverage vs RStudent", x = "Leverage", y = "RStudent")

grid.arrange(p1, p2, p3, p4, p5, p6, ncol = 3)
```

```
## Warning: The dot-dot notation (`..density..`) was deprecated in ggplot2 3.4.0.
## i Please use `after_stat(density)` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



Sanity Check of Above 2 Models

Extra Sums of Square Test

There is enough evidence below to conclude that the latter model is superior (p-value < 0.05).

```
####Last defense between competing models####

#Extra sums of square test
anova(model1, model2)
```

```
## Analysis of Variance Table
##
## Model 1: log(SalePrice) ~ log(GrLivArea)
## Model 2: log(SalePrice) ~ log(GrLivArea) + as.factor(Neighborhood) + as.factor(Neighborhood) *
##     log(GrLivArea)
##   Res.Df    RSS Df Sum of Sq      F    Pr(>F)
## 1    379 15.832
## 2    375 13.418  4    2.4137 16.863 1.004e-12 ***
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
```

Repeated Cross-Validation

Both models are evaluated 5 times using RMSE, R-Squared, and PRESS, respectively. We desire the lower RMSE, higher R-Squared, and lower PRESS. The latter model qualifies all three categories.

```
#Repeated Cross-validation
set.seed(760397)
train.control <- trainControl(method = "repeatedcv", number = 10, repeats = 5)

#Train CV model1
CVModel1 <- train(log(SalePrice) ~ log(GrLivArea),
                  data = A1Data, method = 'lm',
                  trControl = train.control)
paste(CVModel1$results$RMSE, "  | ", CVModel1$results$Rsquared, " | ", press(model1))
```

```
## [1] "0.203387941136873   |  0.445487375655191  |  16.028917038051"
```

```
#Train CV model1
CVModel2 <- train(log(SalePrice) ~ log(GrLivArea) + as.factor(Neighborhood) + as.factor(Neighborhood)*l
                  data = A1Data, method = 'lm',
                  trControl = train.control)
paste(CVModel2$results$RMSE, "  | ", CVModel2$results$Rsquared, " | ", press(model2))
```

```
## [1] "0.189419154092284   |  0.515579561289575  |  13.9480700164513"
```

```
#Parameters
summary(model2)
```

```
##
## Call:
## lm(formula = log(SalePrice) ~ log(GrLivArea) + as.factor(Neighborhood) +
##     as.factor(Neighborhood) * log(GrLivArea), data = A1Data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.73636 -0.10679  0.02187  0.10524  0.74523
##
## Coefficients:
##                                              Estimate Std. Error t value
## (Intercept)                                   5.91292    0.49642  11.911
## log(GrLivArea)                                0.81965    0.07047  11.631
## as.factor(Neighborhood)Edwards                1.01017    0.69821   1.447
## as.factor(Neighborhood)NAmes                  2.57981    0.59016   4.371
## log(GrLivArea):as.factor(Neighborhood)Edwards -0.14631    0.09868  -1.483
## log(GrLivArea):as.factor(Neighborhood)NAmes   -0.34662    0.08345  -4.154
##                                              Pr(>|t|)
## (Intercept)                                   < 2e-16 ***
## log(GrLivArea)                                < 2e-16 ***
## as.factor(Neighborhood)Edwards                   0.149
## as.factor(Neighborhood)NAmes                   1.60e-05 ***
## log(GrLivArea):as.factor(Neighborhood)Edwards    0.139
## log(GrLivArea):as.factor(Neighborhood)NAmes   4.05e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1892 on 375 degrees of freedom
## Multiple R-squared:  0.5279, Adjusted R-squared:  0.5216
## F-statistic: 83.87 on 5 and 375 DF,  p-value: < 2.2e-16
```

```
confint(model2)
```

```
##                                                    2.5 %      97.5 %
## (Intercept)                                    4.9368110  6.88903046
## log(GrLivArea)                                 0.6810853  0.95821077
## as.factor(Neighborhood)Edwards                -0.3627196  2.38306395
## as.factor(Neighborhood)NAmes                   1.4193600  3.74025379
## log(GrLivArea):as.factor(Neighborhood)Edwards -0.3403440  0.04772312
## log(GrLivArea):as.factor(Neighborhood)NAmes   -0.5107056 -0.18254334
```