Kaggle House Price Predictions for Ames, Iowa

2025-04-28

The purpose of this document serves to walk readers through our analysis of housing data found on Kaggle

```
library(leaps)
library(ggfortify)
```

```
## Loading required package: ggplot2
```

```
library(ggcorrplot)
library(base)
library(visdat)
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v lubridate 1.9.3     v tibble    3.2.1
## v purrr     1.0.2     v tidyr     1.3.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```
library(ggplot2)
library(olsrr)
```

```
##
## Attaching package: 'olsrr'
##
## The following object is masked from 'package:datasets':
##
##     rivers
```

```
library(caret)
```

```
## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(dplyr)
library(corrplot)
```

```
## corrplot 0.95 loaded
```

```
library(ggpubr)
library(GGally)
```

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

Analysis 1 - Chris Johnson

Read in data and view column names

```
url = "https://raw.githubusercontent.com/cjohnson4510/Housing-Project/main/train.csv"
Htrain = read.csv(url)

nrow(Htrain)
```

```
## [1] 1460
```

```
colnames(Htrain)
```

```
##  [1] "Id"            "MSSubClass"    "MSZoning"      "LotFrontage"
##  [5] "LotArea"       "Street"        "Alley"         "LotShape"
##  [9] "LandContour"   "Utilities"     "LotConfig"     "LandSlope"
## [13] "Neighborhood"  "Condition1"    "Condition2"    "BldgType"
## [17] "HouseStyle"    "OverallQual"   "OverallCond"   "YearBuilt"
## [21] "YearRemodAdd"  "RoofStyle"     "RoofMatl"      "Exterior1st"
## [25] "Exterior2nd"   "MasVnrType"    "MasVnrArea"    "ExterQual"
## [29] "ExterCond"     "Foundation"    "BsmtQual"      "BsmtCond"
## [33] "BsmtExposure"  "BsmtFinType1"  "BsmtFinSF1"    "BsmtFinType2"
## [37] "BsmtFinSF2"    "BsmtUnfSF"     "TotalBsmtSF"   "Heating"
## [41] "HeatingQC"     "CentralAir"    "Electrical"    "X1stFlrSF"
## [45] "X2ndFlrSF"     "LowQualFinSF"  "GrLivArea"     "BsmtFullBath"
## [49] "BsmtHalfBath"  "FullBath"      "HalfBath"      "BedroomAbvGr"
## [53] "KitchenAbvGr"  "KitchenQual"   "TotRmsAbvGrd"  "Functional"
## [57] "Fireplaces"    "FireplaceQu"   "GarageType"    "GarageYrBlt"
## [61] "GarageFinish"  "GarageCars"    "GarageArea"    "GarageQual"
## [65] "GarageCond"    "PavedDrive"    "WoodDeckSF"    "OpenPorchSF"
## [69] "EnclosedPorch" "X3SsnPorch"    "ScreenPorch"   "PoolArea"
## [73] "PoolQC"        "Fence"         "MiscFeature"   "MiscVal"
## [77] "MoSold"        "YrSold"        "SaleType"      "SaleCondition"
## [81] "SalePrice"
```

Change neighborhood to factor

```
class(Htrain$Neighborhood)
```

```
## [1] "character"
```

```
Htrain$Neighborhood=as.factor(Htrain$Neighborhood)
levels(Htrain$Neighborhood)
```

```
##  [1] "Blmngtn" "Blueste" "BrDale"  "BrkSide" "ClearCr" "CollgCr" "Crawfor"
##  [8] "Edwards" "Gilbert" "IDOTRR"  "MeadowV" "Mitchel" "NAmes"   "NoRidge"
## [15] "NPkVill" "NridgHt" "NWAmes"  "OldTown" "Sawyer"  "SawyerW" "Somerst"
## [22] "StoneBr" "SWISU"   "Timber"  "Veenker"
```

Create 'Ames' Dataframe with neighborhoods of interest

```
am=grep("NAmes|Edwards|BrkSide", Htrain$Neighborhood, ignore.case = TRUE)
Ames=Htrain[am,]
Ames$Neighborhood
```

```
##   [1] BrkSide NAmes   BrkSide NAmes   NAmes   NAmes   NAmes   BrkSide NAmes
##  [10] NAmes   NAmes   Edwards NAmes   NAmes   BrkSide NAmes   NAmes   NAmes
##  [19] NAmes   NAmes   NAmes   BrkSide NAmes   NAmes   NAmes   NAmes   Edwards
##  [28] Edwards NAmes   Edwards Edwards NAmes   NAmes   NAmes   NAmes   NAmes
##  [37] NAmes   NAmes   NAmes   Edwards BrkSide BrkSide Edwards NAmes   Edwards
##  [46] NAmes   NAmes   Edwards Edwards BrkSide NAmes   Edwards NAmes   Edwards
##  [55] NAmes   NAmes   Edwards Edwards NAmes   NAmes   NAmes   Edwards NAmes
##  [64] BrkSide NAmes   NAmes   NAmes   Edwards NAmes   BrkSide NAmes   NAmes
##  [73] BrkSide Edwards NAmes   Edwards NAmes   NAmes   NAmes   BrkSide NAmes
##  [82] NAmes   NAmes   NAmes   BrkSide NAmes   NAmes   Edwards BrkSide Edwards
##  [91] NAmes   NAmes   NAmes   NAmes   NAmes   Edwards Edwards Edwards Edwards
## [100] NAmes   BrkSide Edwards NAmes   Edwards Edwards NAmes   NAmes   NAmes
## [109] BrkSide Edwards Edwards BrkSide Edwards NAmes   BrkSide NAmes   NAmes
## [118] Edwards NAmes   NAmes   NAmes   NAmes   BrkSide NAmes   Edwards NAmes
## [127] NAmes   NAmes   NAmes   BrkSide Edwards NAmes   Edwards BrkSide NAmes
## [136] BrkSide Edwards NAmes   NAmes   BrkSide Edwards BrkSide NAmes   Edwards
## [145] Edwards NAmes   NAmes   NAmes   NAmes   NAmes   NAmes   Edwards BrkSide
## [154] BrkSide BrkSide NAmes   Edwards NAmes   NAmes   Edwards NAmes   NAmes
## [163] NAmes   NAmes   NAmes   NAmes   BrkSide Edwards NAmes   NAmes   NAmes
## [172] Edwards NAmes   Edwards NAmes   NAmes   Edwards NAmes   Edwards NAmes
## [181] Edwards NAmes   BrkSide BrkSide Edwards BrkSide BrkSide NAmes   Edwards
## [190] Edwards NAmes   NAmes   NAmes   Edwards Edwards NAmes   NAmes   BrkSide
## [199] Edwards Edwards NAmes   NAmes   NAmes   NAmes   BrkSide NAmes   NAmes
## [208] Edwards NAmes   BrkSide NAmes   NAmes   NAmes   Edwards NAmes   NAmes
## [217] NAmes   NAmes   NAmes   NAmes   NAmes   NAmes   Edwards NAmes   NAmes
## [226] Edwards NAmes   NAmes   NAmes   NAmes   NAmes   NAmes   NAmes   NAmes
## [235] BrkSide Edwards NAmes   Edwards NAmes   NAmes   BrkSide Edwards NAmes
## [244] NAmes   Edwards BrkSide NAmes   NAmes   NAmes   BrkSide BrkSide Edwards
## [253] NAmes   Edwards NAmes   BrkSide NAmes   Edwards NAmes   Edwards NAmes
## [262] Edwards NAmes   NAmes   Edwards Edwards Edwards NAmes   Edwards NAmes
## [271] NAmes   NAmes   NAmes   Edwards Edwards NAmes   NAmes   Edwards NAmes
## [280] NAmes   NAmes   BrkSide NAmes   NAmes   NAmes   NAmes   NAmes   NAmes
## [289] Edwards Edwards NAmes   BrkSide NAmes   BrkSide NAmes   Edwards BrkSide
## [298] NAmes   Edwards NAmes   NAmes   Edwards NAmes   Edwards Edwards NAmes
## [307] BrkSide NAmes   Edwards NAmes   BrkSide NAmes   NAmes   NAmes   NAmes
## [316] NAmes   NAmes   NAmes   NAmes   NAmes   NAmes   NAmes   Edwards BrkSide
## [325] Edwards NAmes   NAmes   BrkSide NAmes   NAmes   Edwards NAmes   BrkSide
## [334] NAmes   Edwards NAmes   NAmes   Edwards Edwards NAmes   Edwards Edwards
## [343] NAmes   Edwards NAmes   BrkSide BrkSide Edwards Edwards NAmes   NAmes
## [352] NAmes   BrkSide NAmes   NAmes   NAmes   NAmes   Edwards Edwards NAmes
## [361] Edwards BrkSide NAmes   NAmes   BrkSide NAmes   BrkSide NAmes   NAmes
## [370] BrkSide NAmes   Edwards NAmes   NAmes   NAmes   NAmes   NAmes   BrkSide
## [379] Edwards NAmes   Edwards NAmes   Edwards
## 25 Levels: Blmngtn Blueste BrDale BrkSide ClearCr CollgCr Crawfor ... Veenker
```

Use Naniar library to find missing values in variables of interest

```
library(naniar)
mv=miss_var_summary(Ames)
print(mv, n=100)
```

```
## # A tibble: 81 x 3
##    variable      n_miss pct_miss
##    <chr>          <int>    <num>
##  1 PoolQC           380    99.2
##  2 Alley            371    96.9
##  3 MiscFeature      361    94.3
##  4 Fence            275    71.8
##  5 FireplaceQu      237    61.9
##  6 LotFrontage       54    14.1
##  7 GarageType        38     9.92
##  8 GarageYrBlt       38     9.92
##  9 GarageFinish      38     9.92
## 10 GarageQual        38     9.92
## 11 GarageCond        38     9.92
## 12 BsmtQual          24     6.27
## 13 BsmtCond          24     6.27
## 14 BsmtExposure      24     6.27
## 15 BsmtFinType1      24     6.27
## 16 BsmtFinType2      24     6.27
## 17 Id                 0     0
## 18 MSSubClass         0     0
## 19 MSZoning           0     0
## 20 LotArea            0     0
## 21 Street             0     0
## 22 LotShape           0     0
## 23 LandContour        0     0
## 24 Utilities          0     0
## 25 LotConfig          0     0
## 26 LandSlope          0     0
## 27 Neighborhood       0     0
## 28 Condition1         0     0
## 29 Condition2         0     0
## 30 BldgType           0     0
## 31 HouseStyle         0     0
## 32 OverallQual        0     0
## 33 OverallCond        0     0
## 34 YearBuilt          0     0
## 35 YearRemodAdd       0     0
## 36 RoofStyle          0     0
## 37 RoofMatl           0     0
## 38 Exterior1st        0     0
## 39 Exterior2nd        0     0
## 40 MasVnrType         0     0
## 41 MasVnrArea         0     0
## 42 ExterQual          0     0
## 43 ExterCond          0     0
## 44 Foundation         0     0
## 45 BsmtFinSF1         0     0
## 46 BsmtFinSF2         0     0
## 47 BsmtUnfSF          0     0
## 48 TotalBsmtSF        0     0
## 49 Heating            0     0
## 50 HeatingQC          0     0
## 51 CentralAir         0     0
```

```
## 52 Electrical         0     0
## 53 X1stFlrSF          0     0
## 54 X2ndFlrSF          0     0
## 55 LowQualFinSF       0     0
## 56 GrLivArea          0     0
## 57 BsmtFullBath       0     0
## 58 BsmtHalfBath       0     0
## 59 FullBath           0     0
## 60 HalfBath           0     0
## 61 BedroomAbvGr       0     0
## 62 KitchenAbvGr       0     0
## 63 KitchenQual        0     0
## 64 TotRmsAbvGrd       0     0
## 65 Functional         0     0
## 66 Fireplaces         0     0
## 67 GarageCars         0     0
## 68 GarageArea         0     0
## 69 PavedDrive         0     0
## 70 WoodDeckSF         0     0
## 71 OpenPorchSF        0     0
## 72 EnclosedPorch      0     0
## 73 X3SsnPorch         0     0
## 74 ScreenPorch        0     0
## 75 PoolArea           0     0
## 76 MiscVal            0     0
## 77 MoSold             0     0
## 78 YrSold             0     0
## 79 SaleType           0     0
## 80 SaleCondition      0     0
## 81 SalePrice          0     0
```

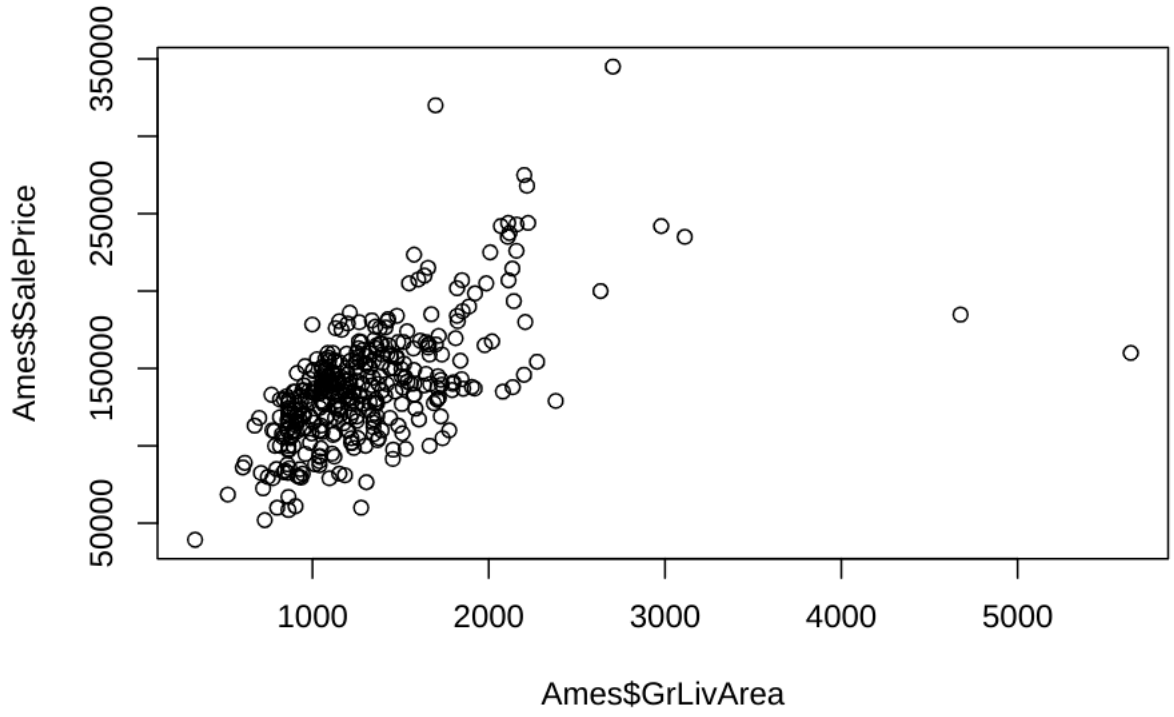
```
Ames$GrLivArea
```

```
##   [1] 1077 1253  854 1004 1339  900 1600  520 1700 1297 1057 1152 1324 1150 1176
##  [16] 1360 1425 2207 2223 1086  952 1285 2142 1065 1040 1235  960  835 1225 1855
##  [31] 1125 1080 1348 1053 2157 1327 1214  864 1385 1709  875 1344 1096 1040 1355
##  [46] 1656 1362 2158 1340 1252 1479 1709 3112 1121 1100 1092  864 1212 1236  864
##  [61] 1194 1487 1375 1306 1302 1314 1382 1113 1632 1548 1734  858 1396 1716 1644
##  [76]  861  972 2978 1383 2134 1728 1056 1215 1040 1577  958 1478 1111 1505 1922
##  [91] 1394 1431 1268 1287 1319  904 1184 1125 1367  882  788 1144 1812 1276 1134
## [106] 1056 1196  907  904 1196 1440 1573 1689 1888 1203 1277 1644 1072 1113 1073
## [121] 1578 1269 1820  912 1214 1041 1363  864 1244 1664 4676  928  605 1362  827
## [136]  334 1347  864  767 1635 1126 1048 1092  996 1674  943 1728  864 1109 1216
## [151] 1429  816 1573  838  935 1986 2008 1054  832  864 1116 1422 1520 2080 1350
## [166] 1056  800  796 2704  981 1048 1094 1839 1510 1053 1458 1486 1392 1181 2380
## [181] 1369 1136 1539  616 1148  729  960  864 1470 1698  864 1776 1040 1200 1529
## [196] 1026  864 1301 1220 1117  912  773 1128  980 1576 1086 1442 1250 1008  784
## [211] 1392 1516 1144 1200 1165 1800 1445 1148 1002  894  910 1268 1090  892 1712
## [226] 1393 2217 1505  924 1796  858 1306 1063 2274 1015 1229 1414 2200  925 2069
## [241]  747 1440 1144  864  980  858 1098 1095 1192 2019  869  894  999 1164 1851
## [256] 1230 1050  944 1657 1664 1082 1132 1264 1376  845 1733  930 1977 1526 1154
## [271] 1611  893 1048 1456 1426 1096 1251 1709 1040 1200  936 1324  950 1134 1194
## [286]  816 1008 1040  960  698 1005  986 1252 1167  952  924 1576  932 1466 1820
## [301] 1265 2108 1261 1124 1221  864 1348 1283  672  999  912  912 1846 2136 1138
## [316]  912 1507 1190 1224 1188  988 2110 1656 1367  864 1054 1050 1824 1337 1524
```

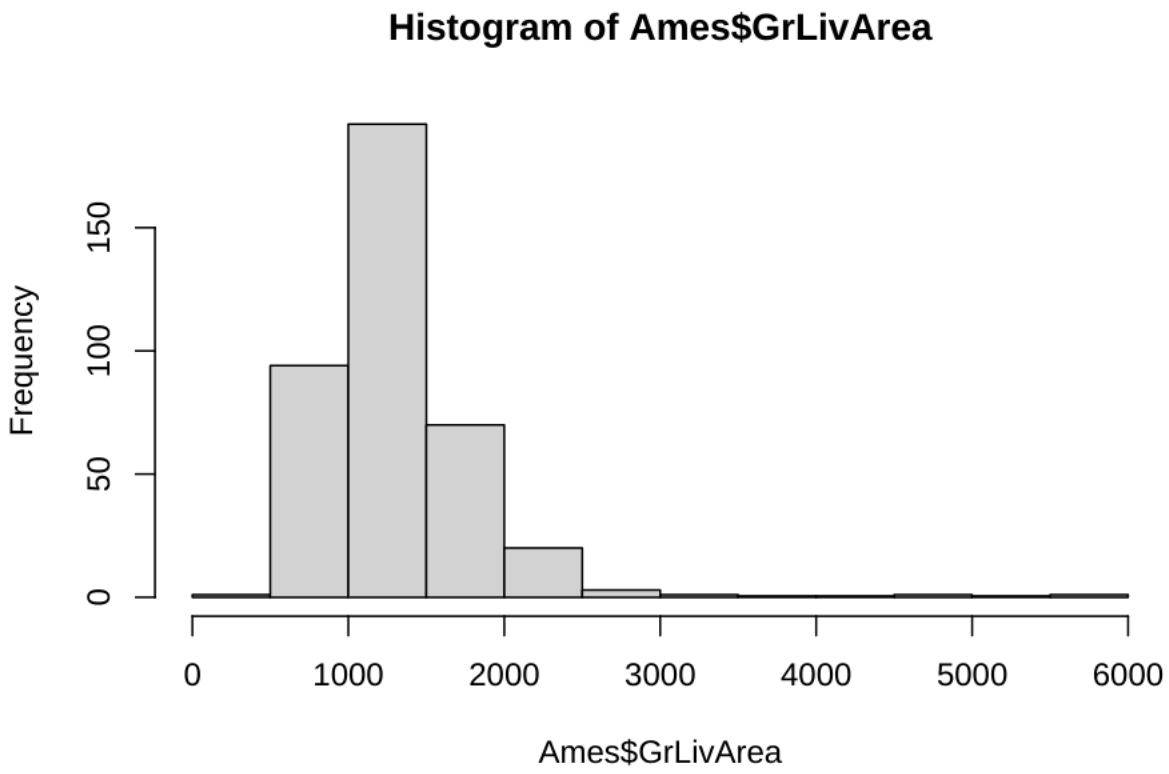
```
## [331] 1357 1920 1412 1152  864 1052 1128 1072 5642 1246 1708  948 2112  948 1478
## [346]  720  708  774  816  872 2634 1716 1176  892 1078 1738 1661 1604  864 2117
## [361] 1258 1218 1584  900 1513 1904 1158 1668 1040 1848 1144 2201 1344 1252 1558
## [376] 1537  864  952 1346 1792 1072 1078 1256
```

No Missing values in variables of interest Create model and Plot to See if Data is normally Distributed

```
plot(Ames$GrLivArea,Ames$SalePrice)
```




```
hist(Ames$GrLivArea)
```



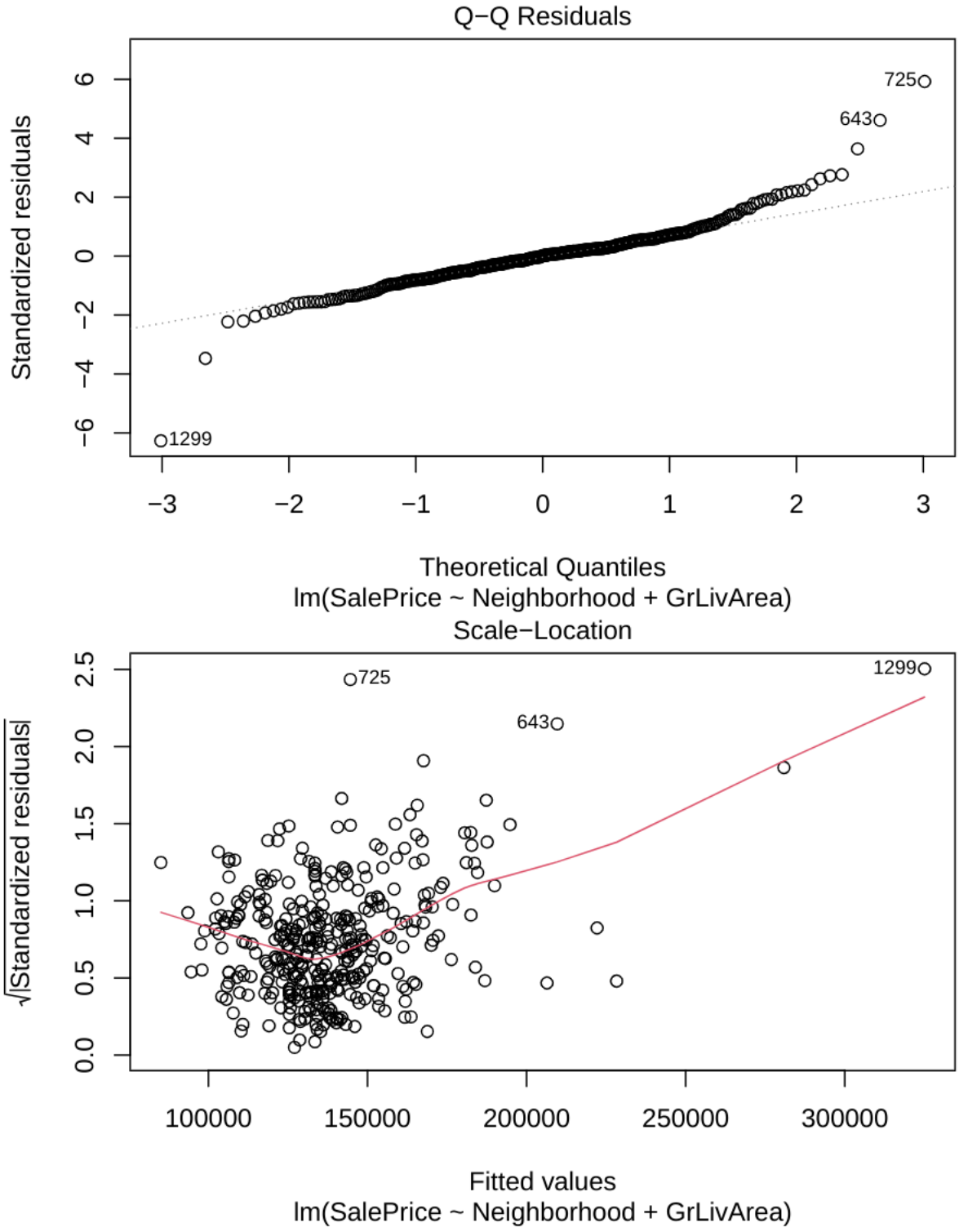

```
hist(Ames$SalePrice)
```

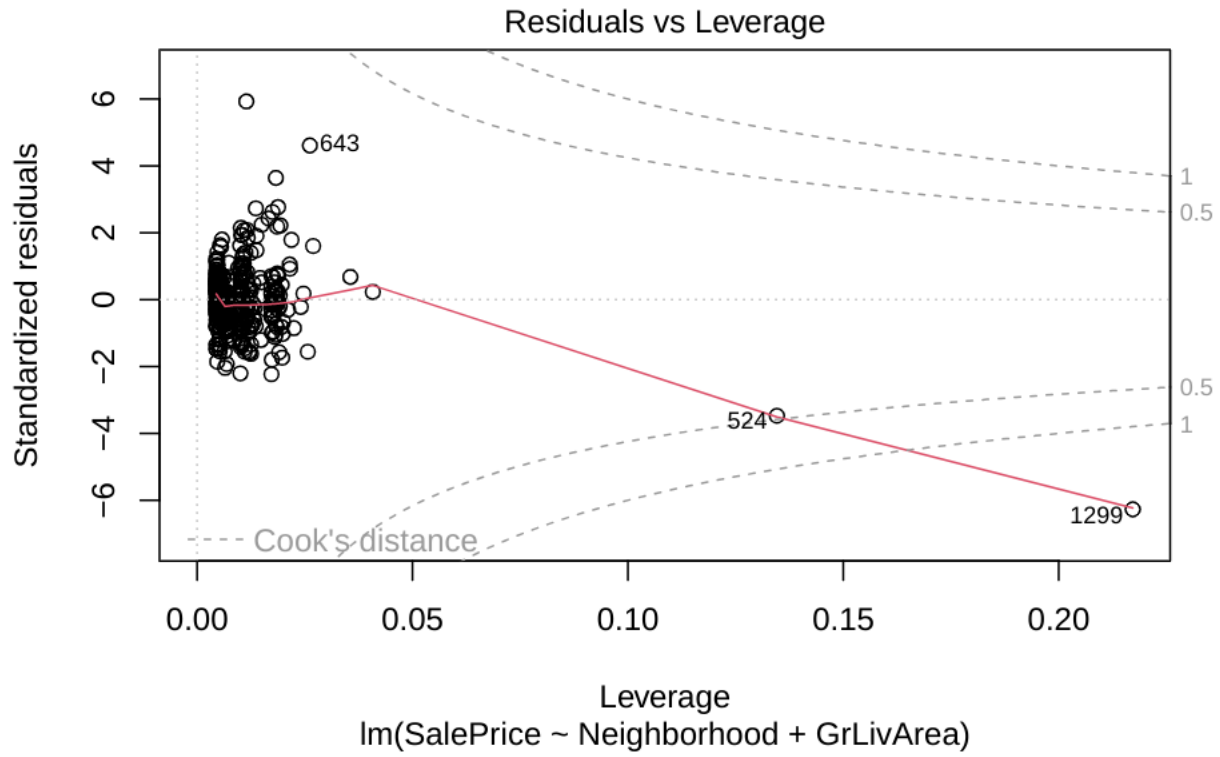

```
model=lm(SalePrice~Neighborhood+GrLivArea, Ames)
summary(model)
```

```
##
```

```
## Call:
## lm(formula = SalePrice ~ Neighborhood + GrLivArea, data = Ames)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -165078  -16215     281   13578  175400
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)          69781.538   5442.400  12.822  < 2e-16 ***
## NeighborhoodEdwards -2882.155   4930.632  -0.585 0.559204
## NeighborhoodNAmes   16105.621   4395.352   3.664 0.000283 ***
## GrLivArea              45.760      3.149  14.533  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 29760 on 379 degrees of freedom
## Multiple R-squared:  0.3965, Adjusted R-squared:  0.3917
## F-statistic:    83 on 3 and 379 DF,  p-value: < 2.2e-16
```

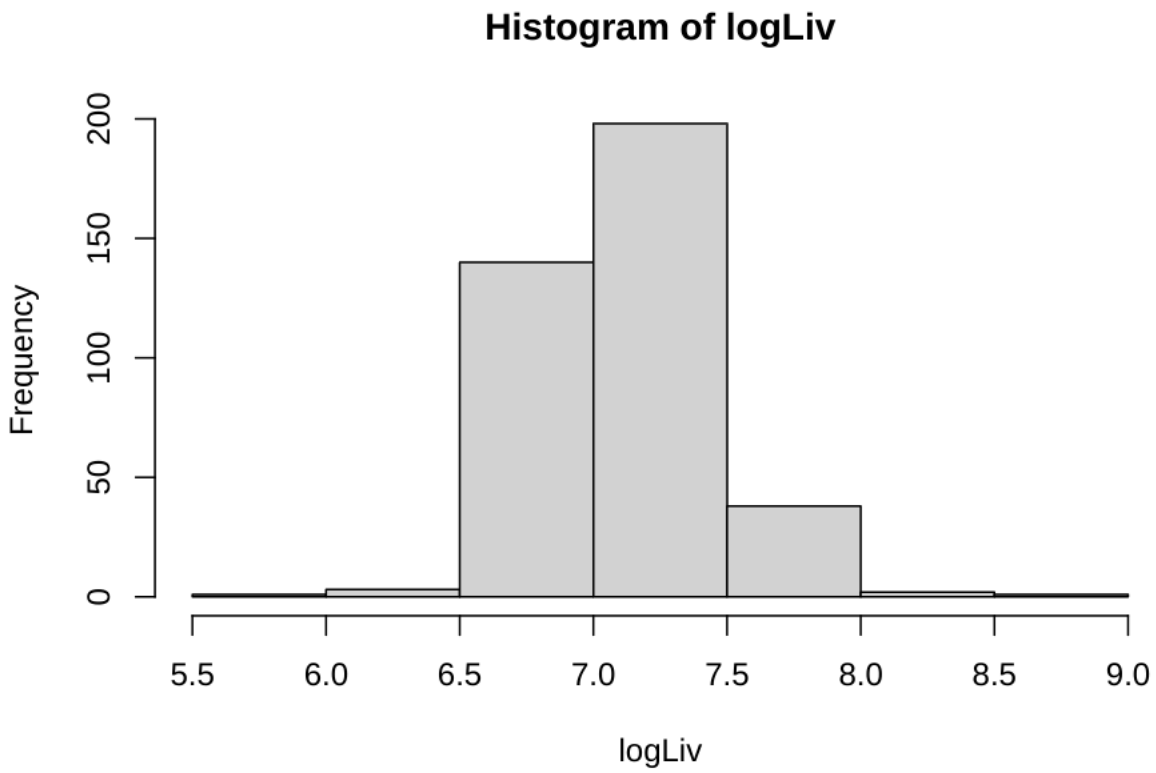
```
plot(model)
```


Q−Q Residuals
Standardized residuals
725
643
1299
−3
−2
−1
0
1
2
3
−6
−4
−2
0
2
4
6
Theoretical Quantiles
lm(SalePrice ~ Neighborhood + GrLivArea)
Scale−Location
√|Standardized residuals|
725
643
1299
0.0
0.5
1.0
1.5
2.0
2.5
100000
150000
200000
250000
300000
Fitted values
lm(SalePrice ~ Neighborhood + GrLivArea)

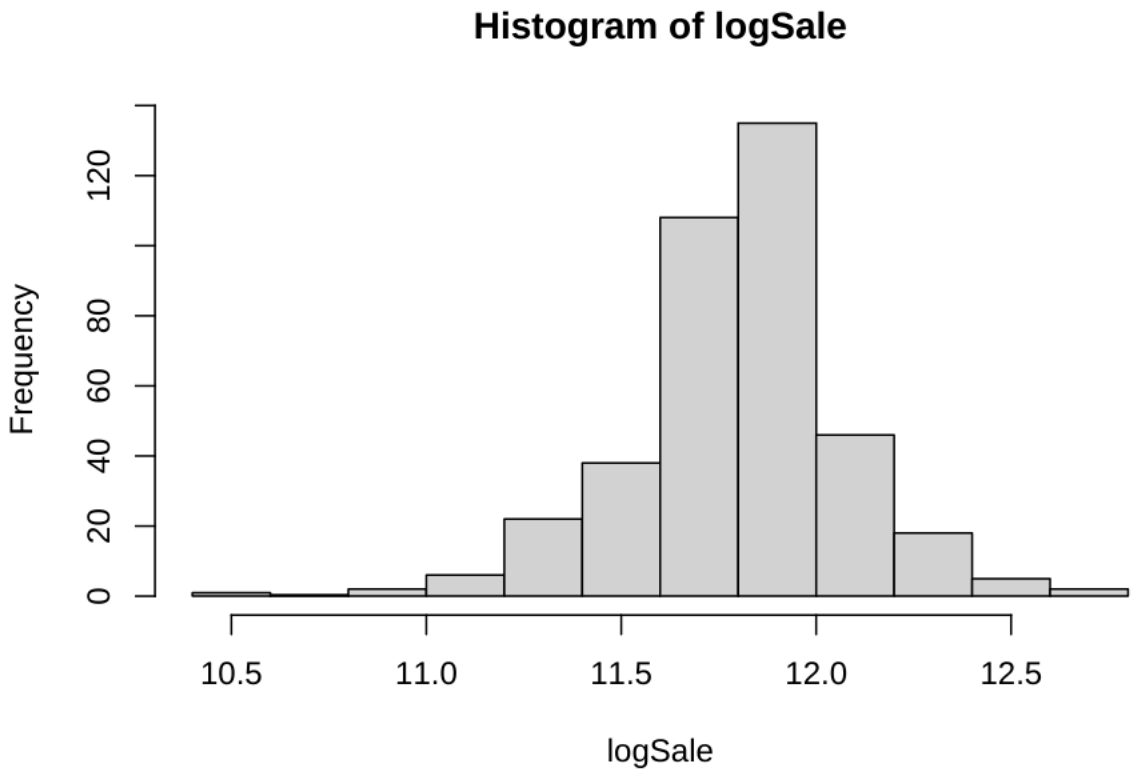


GrlivArea and Sale Price non-normally distributed, log transformation performed and plotted

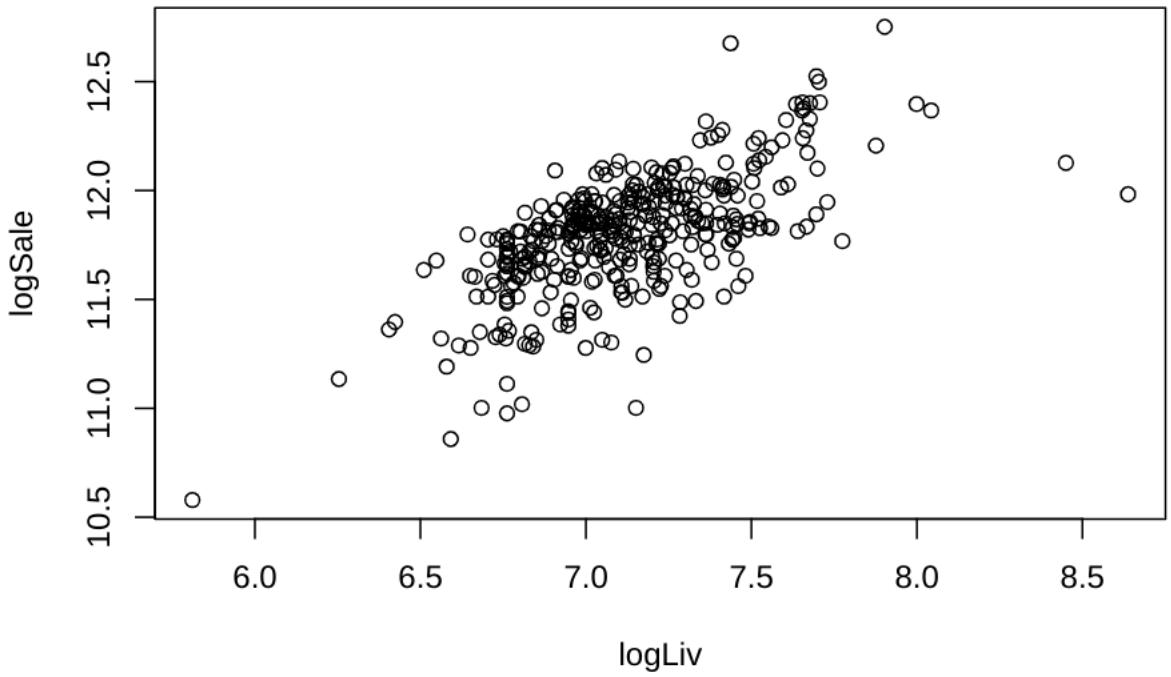
```
logLiv=log(Ames$GrLivArea)
logSale=log(Ames$SalePrice)
hist(logLiv)
```



```
hist(logSale)
```




```
plot(logLiv, logSale)
```



Create new data frame with log transformations. Model coeffecients, confidence intervals and adj r-sqaured. Assumptions: QQ plot looks normal, residuals and leverages address below

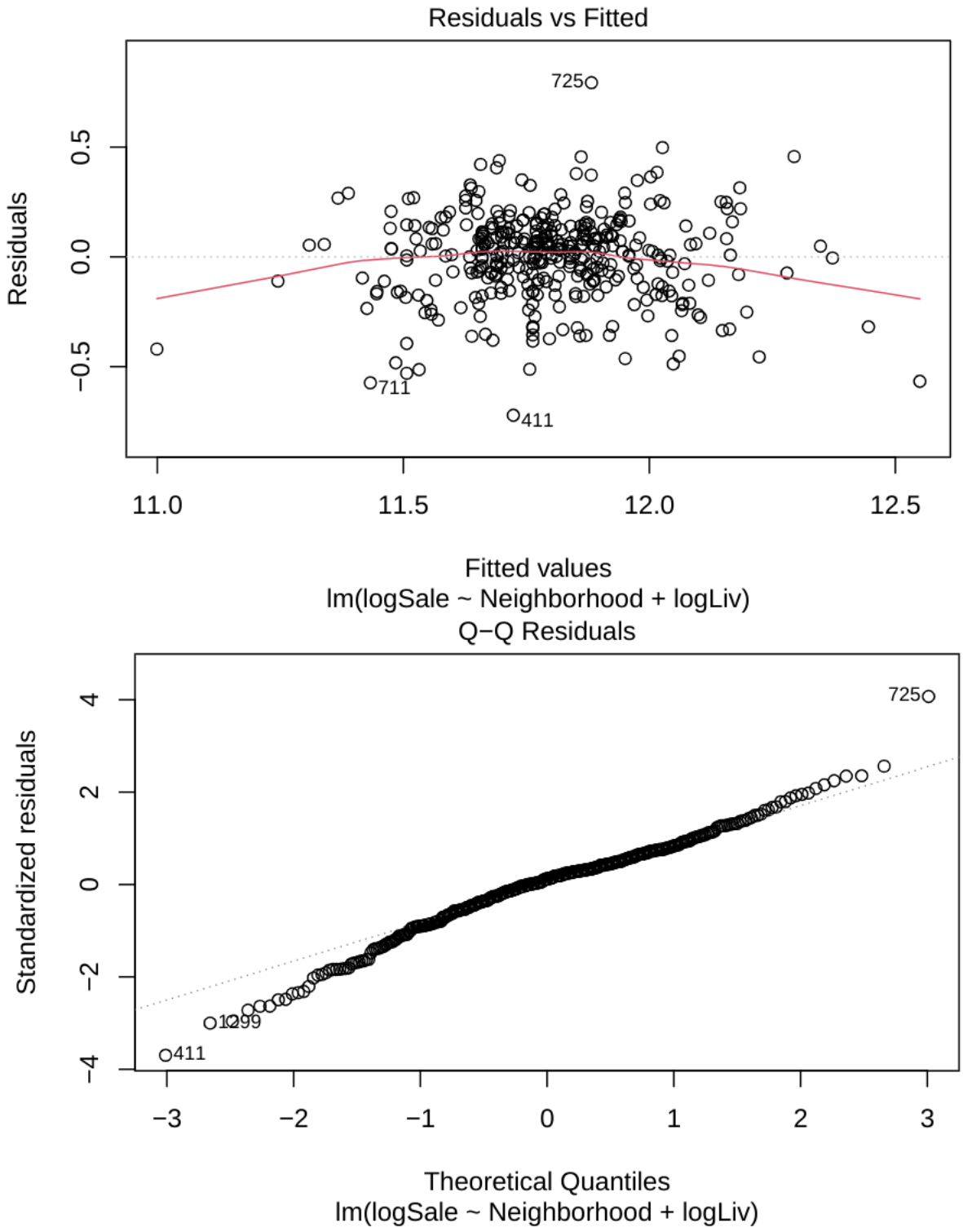
```r
logAmes=cbind(Ames,logSale, logLiv)
logModel=lm(logSale~Neighborhood+logLiv, logAmes)
summary(logModel)
```

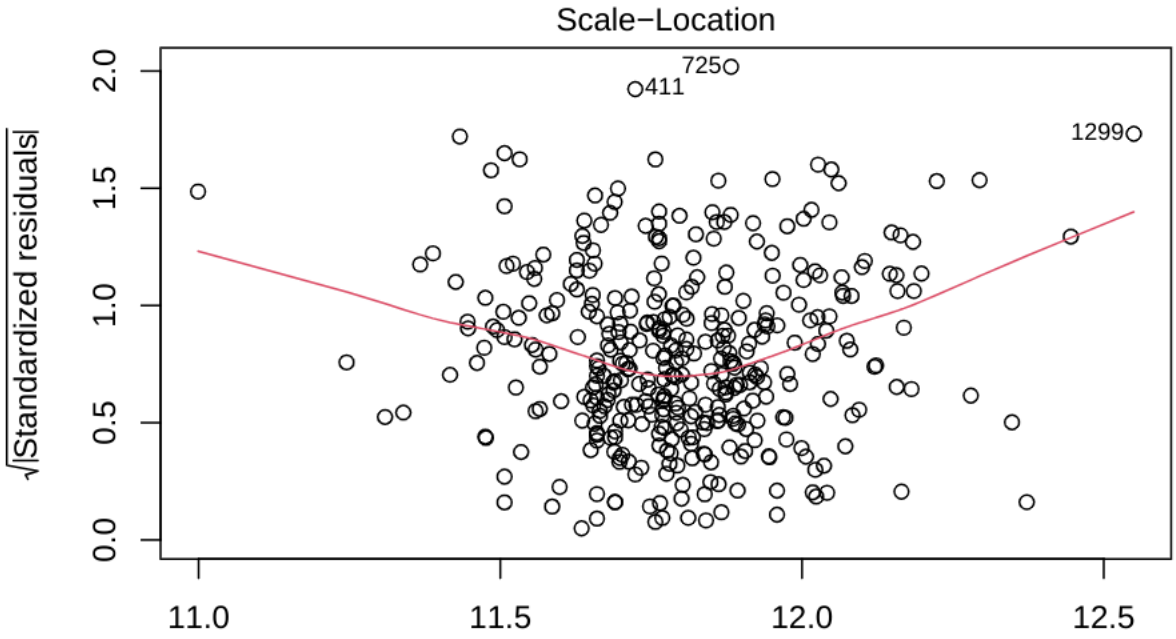
```
##
## Call:
## lm(formula = logSale ~ Neighborhood + logLiv, data = logAmes)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.72154 -0.10592  0.02469  0.11565  0.79364
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)          7.76936    0.22919  33.900  < 2e-16 ***
## NeighborhoodEdwards -0.02044    0.03252  -0.629     0.53
## NeighborhoodNAmes    0.13279    0.02906   4.569 6.63e-06 ***
## logLiv               0.55579    0.03237  17.171  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1961 on 379 degrees of freedom
## Multiple R-squared:  0.4897, Adjusted R-squared:  0.4857
## F-statistic: 121.2 on 3 and 379 DF,  p-value: < 2.2e-16
```

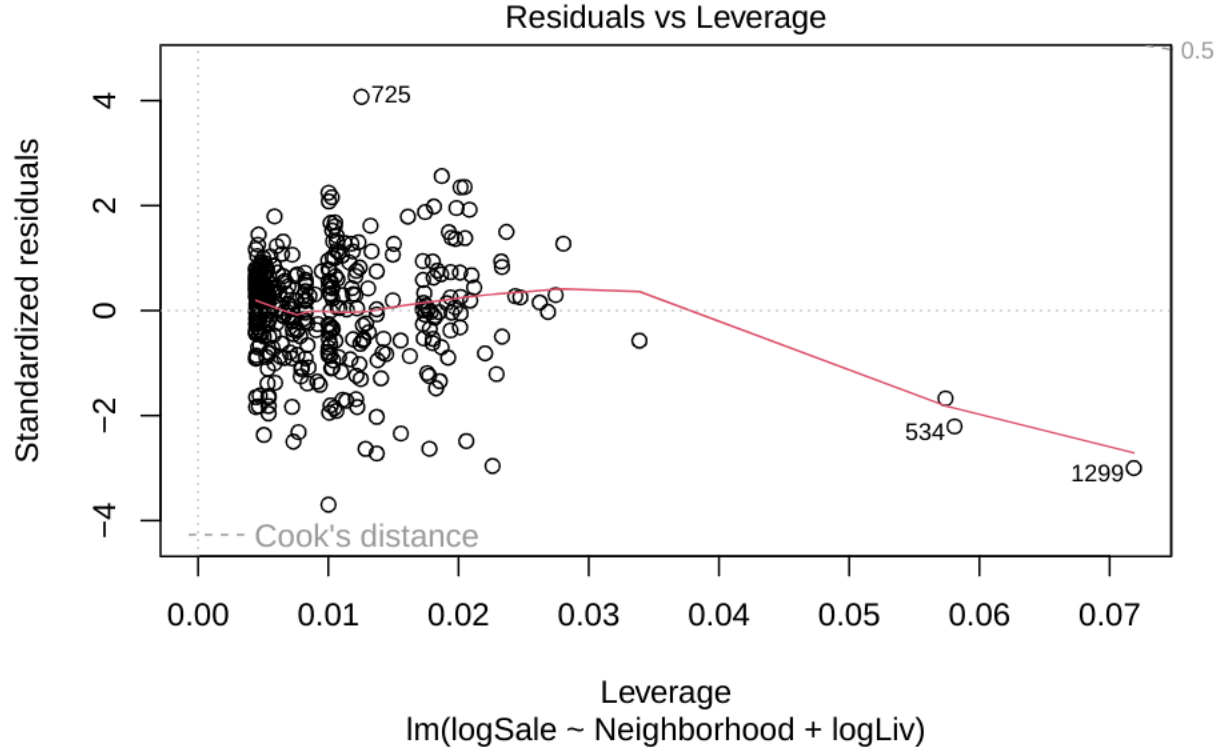
```r
confint(logModel)
```

```
##                          2.5 %     97.5 %
## (Intercept)          7.31872287 8.21999895
## NeighborhoodEdwards -0.08437241 0.04349721
## NeighborhoodNAmes    0.07564743 0.18992983
## logLiv               0.49214387 0.61943290
```

```r
plot(logModel)
```


Residuals vs Fitted
Residuals
Fitted values
lm(logSale ~ Neighborhood + logLiv)
Q−Q Residuals
Standardized residuals
Theoretical Quantiles
lm(logSale ~ Neighborhood + logLiv)
725
711
411
1299





Calculate CV Press of the model

```
#set.seed(123)
#k <- 5
#fold_size <- nrow(logAmes) / k
#cv_press <- 0
#for (i in 1:k) {
```

```
#  test_indices <- ((i-1) * fold_size + 1):(i * fold_size)
#  test_data <- logAmes[test_indices, ]
#  train_data <- logAmes[-test_indices, ]
#  model <- lm(logSale ~ Neighborhood + logLiv, data = train_data)
#  predictions <- predict(model, test_data)
#  cv_press[i] <- cv_press + sum((test_data$logSale - predictions) ^ 2)
# }
#mean(cv_press)
```

To address leverages and residuals we omitted the 3 largest leverage and the 3 largest residual data points Created new model and plot for comparison Leverages and residuals looks normally distributed after datapoints omitted

```
leverages=hatvalues(logModel)
order(leverages, decreasing=TRUE)
```

```
##   [1] 339 136 131   8  80 184  53 250  78 133 370 347 328 123 346 186 241 130
##  [19] 210 101 140 167 309 169  85 205 297 112 153  70 154 183 365   3 246  89
##  [37] 351 290 251  41 372 238 109 311 333  73 155 324 134  48 378 307 187  42
##  [55] 282 292  64 198  22 235  50 302 142 256 362 115   1  15 353 294 367 348
##  [73] 157 180 168  90 152 349  28  30 265 234  76  57 160 325 335 359  19 227
##  [91]  18 252 266  96  74  40  52 278 341 190 296 113  35 267 145 260  23 323
## [109] 258 314 342 344  27 289 360 343 322 358 245 144 164 240 195 269 139 174
## [127] 202  62 177 189 299 118 338 381 274 111 242 223 172  43 275  88  68 200
## [145]  54 226 304  31  98 141 262 105 181  99 102 185 331  12  45 156 137 379
## [163] 254  49 179  97 110 268 194 214  58 125 199 305 104 236 361 383 208 286
## [181] 332 135 366 159 114 255 313 173  72 231  38  60 128 138 148 188 191 197
## [199] 244 306 377 300 103 350 216 230 380 100 192 224 354 272 220   6 364 108
## [217] 356  71 221 124 201 312 316  81 147 352 225 229 239   9 132 298 281 146
## [235] 368 357 283 259  21 295  46  86  75 117  69  77 271 204 170   7 321 363
## [253] 253 310 121 219   4 291 209 287 375 196 376 330 163  25  44  84 193 279
## [271] 288 369 212 126 317 171 273 228 257 327 336  34 175 158 326  82 106 166
## [289]  11 233  24  51  87 345 120 355 382  32 176 261  20 206 217  56 143 207
## [307] 248 276 247  55  92 151  17 149 162 119 237 161 203 337  91 178 211 284
## [325] 182 315  39  79  67 213 243 371 264  63 218  14 334 270 127  47  16 215
## [343] 165  33 373   5 329 320  36 318  13 249  61 285  95 107 280  66 232  65
## [361]  37  83 150  10 319  29  94 308  26 116  59 122  93 222 129 340 301 263
## [379] 303 277 293 374   2
```

```
resid=abs(resid(logModel))
order(resid, decreasing=TRUE)
```

```
##   [1] 190 104 186 339 359  96  64 372 356 167 176 169 180 205 192  58 262 136
##  [19] 194 160  48  84  97  70 140 274 302  43  26  81 260 264 369  12 253 157
##  [37] 164 195 314 223 331 279 319 131 193 178 227 172  31 370 290 155 334 118
##  [55] 332 171  72   3 363 309 154 366 311 381  68  46 296 145 234 240 322  90
##  [73]  80 215 380 198 258   7 346 235 301 174 338 360  19 230 177 317 181 304
##  [91] 216 313  86 316 152 139 329 196 267  54 121 206 275 383  88  76 294 187
## [109] 123 358 161 293 245  60 147 120 368  95 251  62  11 151  69 348 137   4
## [127] 150 199 219  17 224  51  28 328  35  44 241 263  67 265  25 225 326 361
## [145] 125 179 116  34  73 323 237 286 335 337 246 170 300 112 285 375 305  16
## [163] 238 355 272 109 303 210 203 173   6 342 295 242 379 165 270 113 117 378
## [181]  42 175 277 299 128  94 353 111 146 168  40  52   8 341 236 156 244 268
## [199]  27  74 307 102  30  21 115  83 127 330  39 119   2 261  20 347 197 100
## [217] 211  56 106 291 132 214  47 162 371 108  29 204 376 105  13 130 343  93
```

```
## [235]  98 252  65 209 382 340  18 110 122 220 222  99 351 166  92 159 188 315
## [253]  71  82 354 200 336  63 310 292 143  61  77 362 141 254 287 281 153  55
## [271]   5 289 158 114 185 344 269 248 184 129  49 257 255 350 278 243 228 133
## [289] 212  10 208  91 135  15 297  59  78  89 280 276 232 231 321 266 273 226
## [307]  33 318 149 201 138 288 191 377 349 312  14 182 364 101  32 163  79 148
## [325] 213 126 103 308 250   1  24 373 221  41 324 320 249 298 207  36 271 217
## [343] 239 218 283 229  37 134 247   9  50 259 233 256  57 183  85 189 202  66
## [361]  87  23  75 352  38 365 357 333 367 124  53 325 282  22 144 374 345 284
## [379] 327 306 107  45 142
```

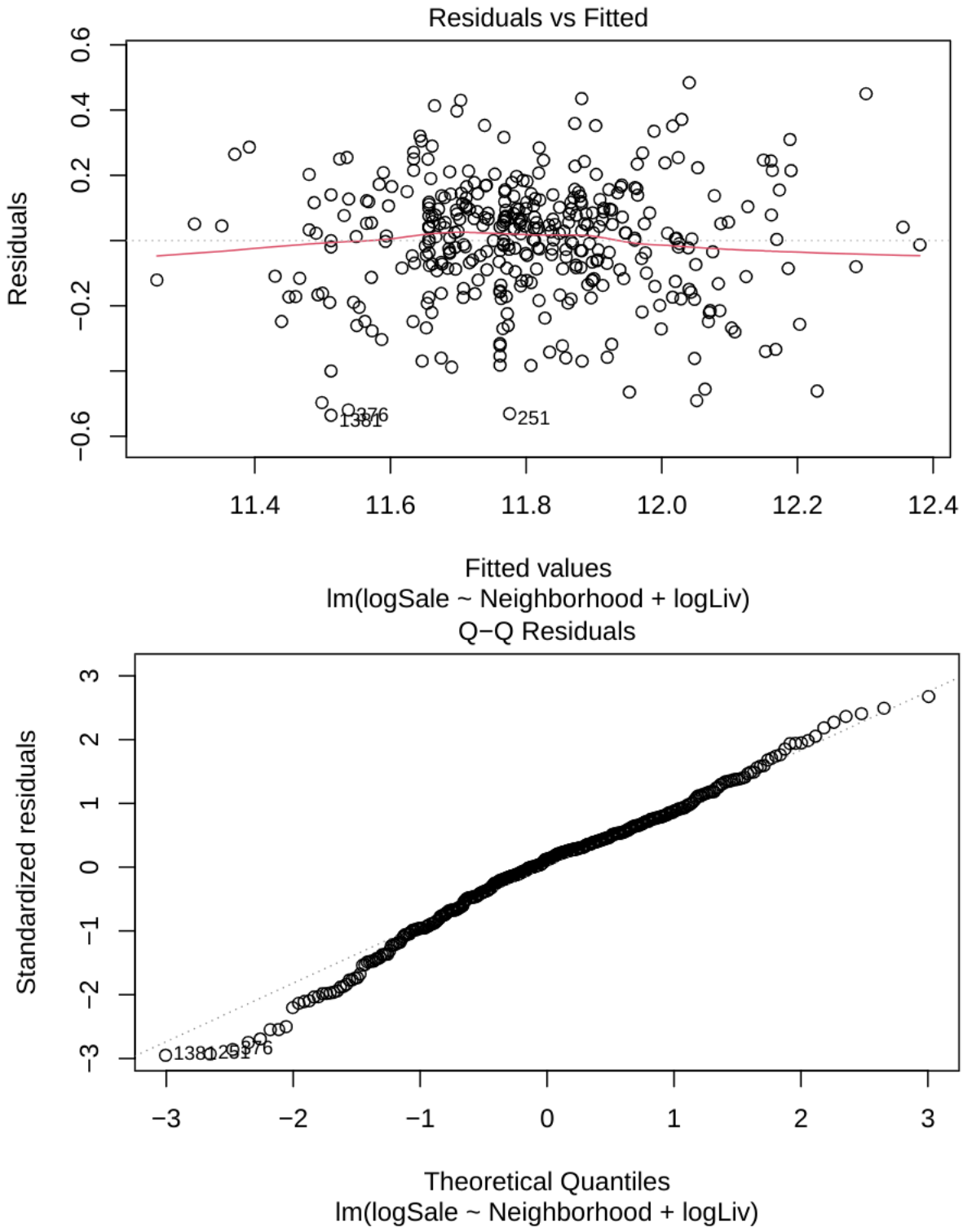
```
outlier_remove_df=logAmes[-c(339, 136, 131, 190, 104, 186  ),]
ordfModel=lm(logSale~Neighborhood+logLiv, outlier_remove_df)
summary(ordfModel)
```

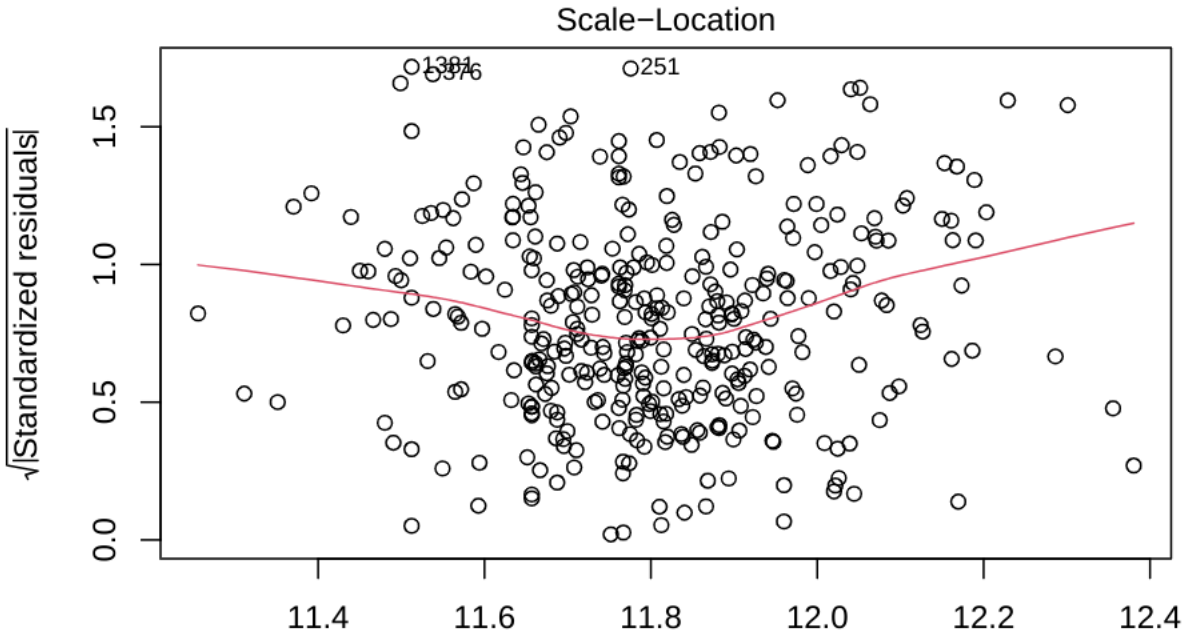
```
##
## Call:
## lm(formula = logSale ~ Neighborhood + logLiv, data = outlier_remove_df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.53562 -0.11056  0.02251  0.11389  0.48397
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)          7.72414    0.23424  32.976  < 2e-16 ***
## NeighborhoodEdwards -0.03001    0.03076  -0.975     0.33
## NeighborhoodNAmes    0.11441    0.02740   4.175 3.71e-05 ***
## logLiv               0.56470    0.03297  17.127  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1829 on 373 degrees of freedom
## Multiple R-squared:  0.4939, Adjusted R-squared:  0.4899
## F-statistic: 121.3 on 3 and 373 DF,  p-value: < 2.2e-16
```

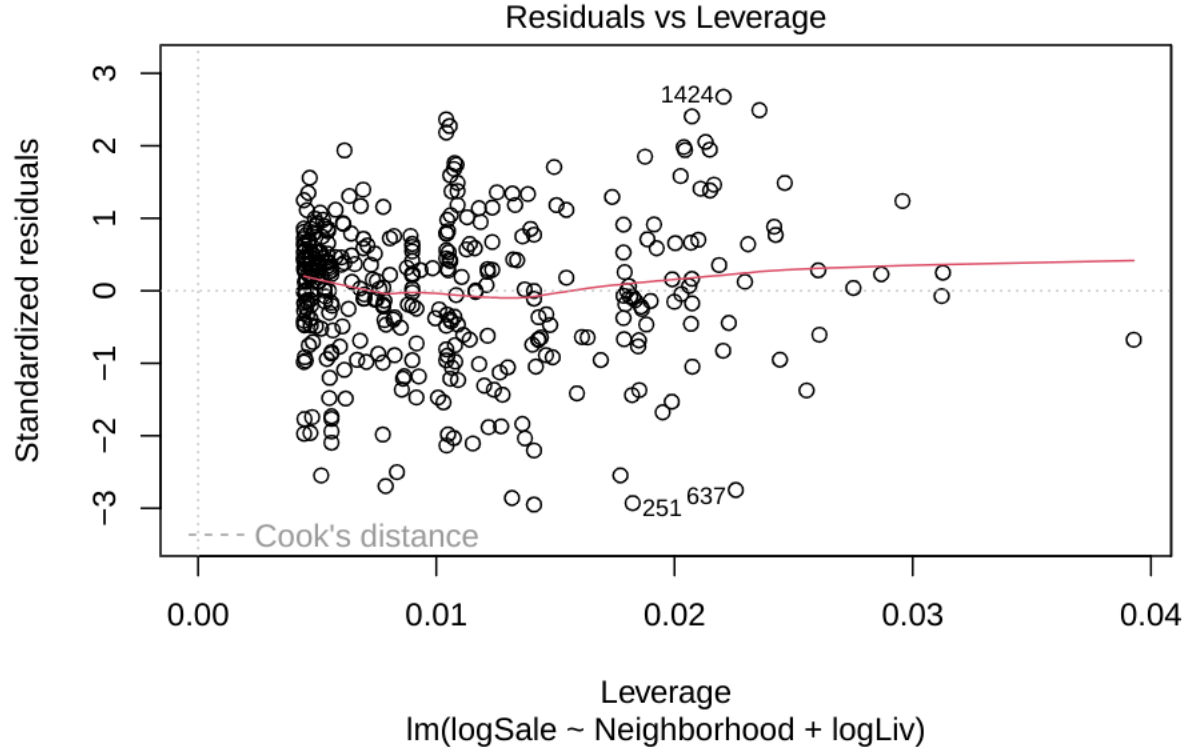
```
confint(ordfModel)
```

```
##                         2.5 %    97.5 %
## (Intercept)          7.26354913 8.1847233
## NeighborhoodEdwards -0.09050183 0.0304869
## NeighborhoodNAmes    0.06052650 0.1682979
## logLiv               0.49986910 0.6295340
```

```
plot(ordfModel)
```


Residuals vs Fitted
Residuals
0.6
0.4
0.2
−0.2
−0.6
1381
376
251
11.4
11.6
11.8
12.0
12.2
12.4
Fitted values
lm(logSale ~ Neighborhood + logLiv)
Q−Q Residuals
Standardized residuals
3
2
1
0
−1
−2
−3
1381
251
376
−3
−2
−1
0
1
2
3
Theoretical Quantiles
lm(logSale ~ Neighborhood + logLiv)



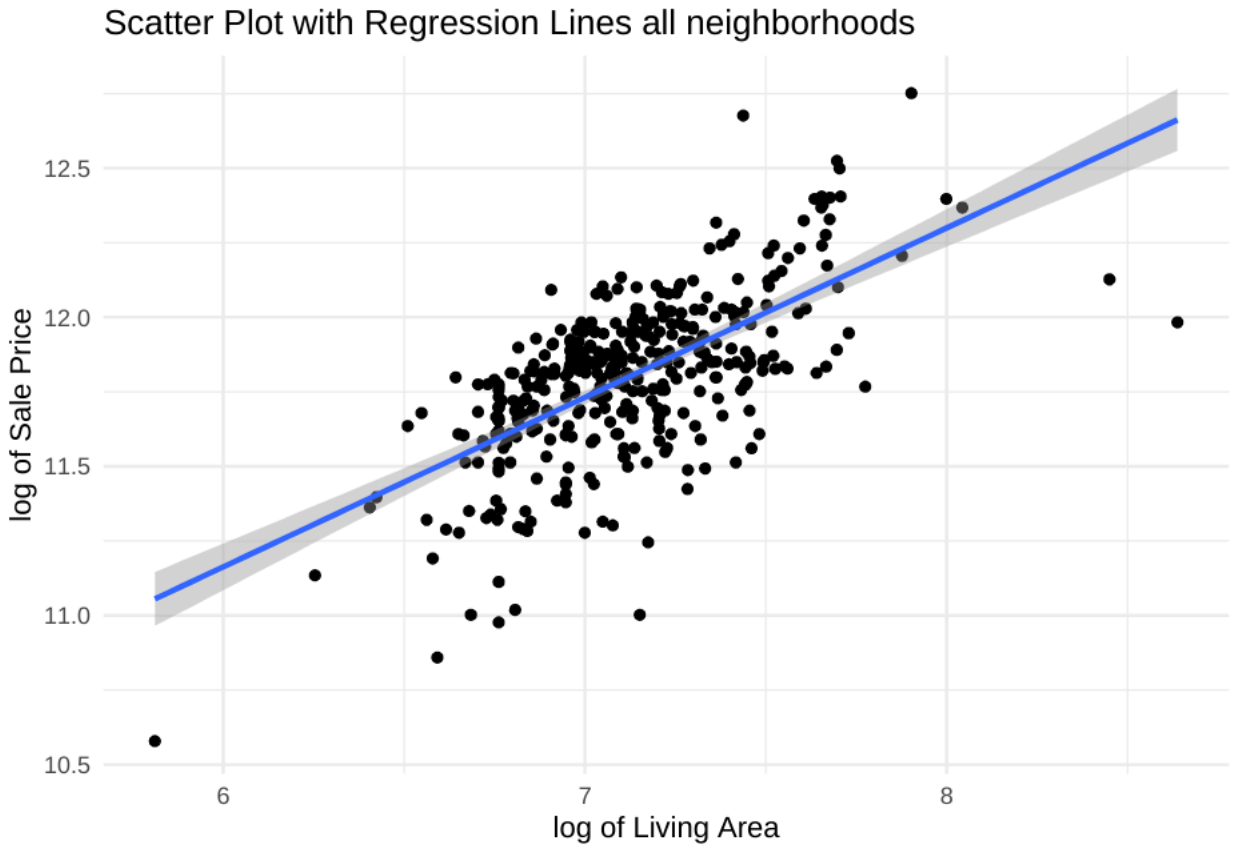


Orginal data, Plot log Model, all neighborhoods

```
library(ggplot2)
ggplot(logAmes, aes(x = logLiv, y = logSale)) +
  geom_point() +
  geom_smooth(method = "lm", se = TRUE) +
  labs(title = "Scatter Plot with Regression Lines all neighborhoods", x = "log of Living Area", y = "l
```

```
  theme_minimal()
```

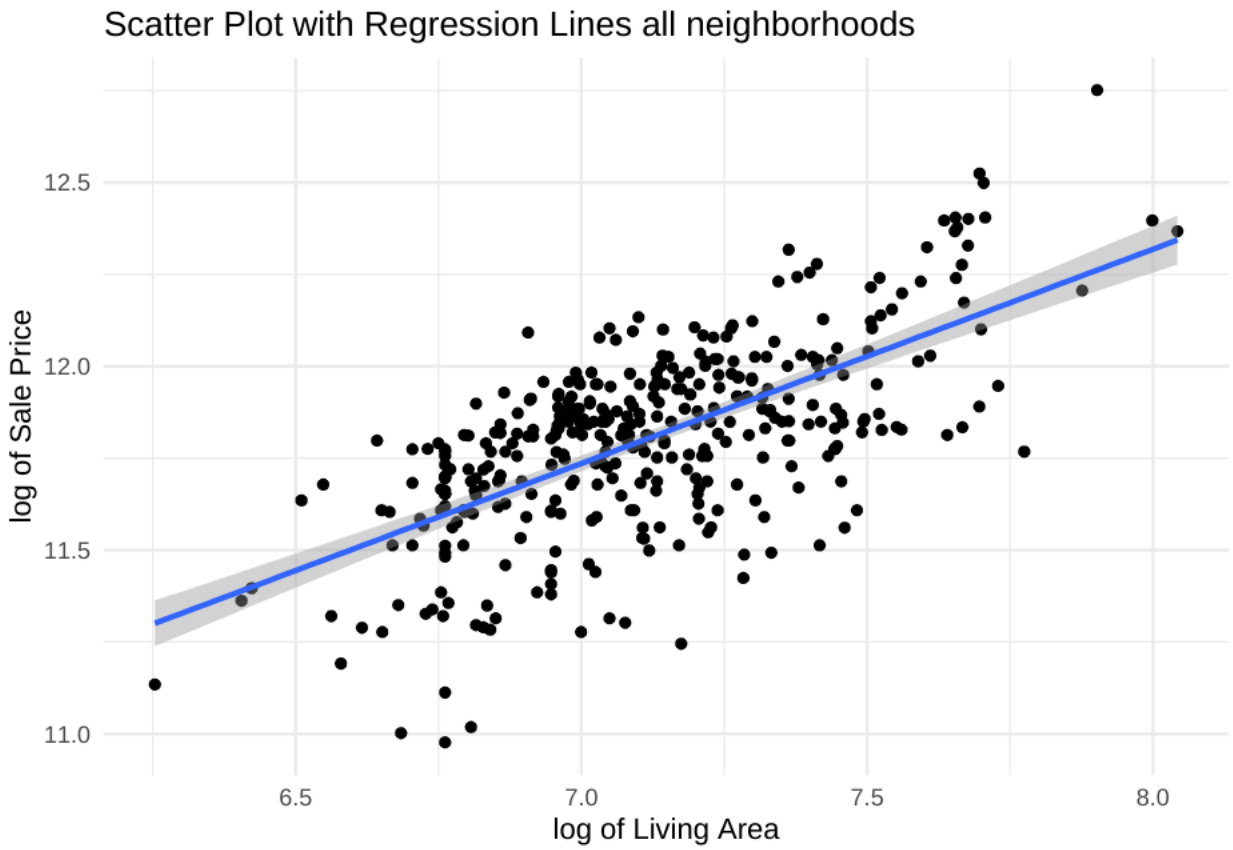
```
## `geom_smooth()` using formula = 'y ~ x'
```



Omitted data, Plot log model, all neighborhoods

```
ggplot(outlier_remove_df, aes(x = logLiv, y = logSale)) +
  geom_point() +
  geom_smooth(method = "lm", se = TRUE) +
  labs(title = "Scatter Plot with Regression Lines all neighborhoods", x = "log of Living Area", y = "l
  theme_minimal()
```

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



Original data, Plot Log Data by Neighboorhood

```
ggplot(logAmes, aes(x = logLiv, y = logSale, color=Neighborhood)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE) +
  labs(title = "Scatter Plot with Regression Lines all neighborhoods", x = "log of Living Area", y = "l
  theme_minimal()
```

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

Omitted data, Plot Log Data by Neighboorhood

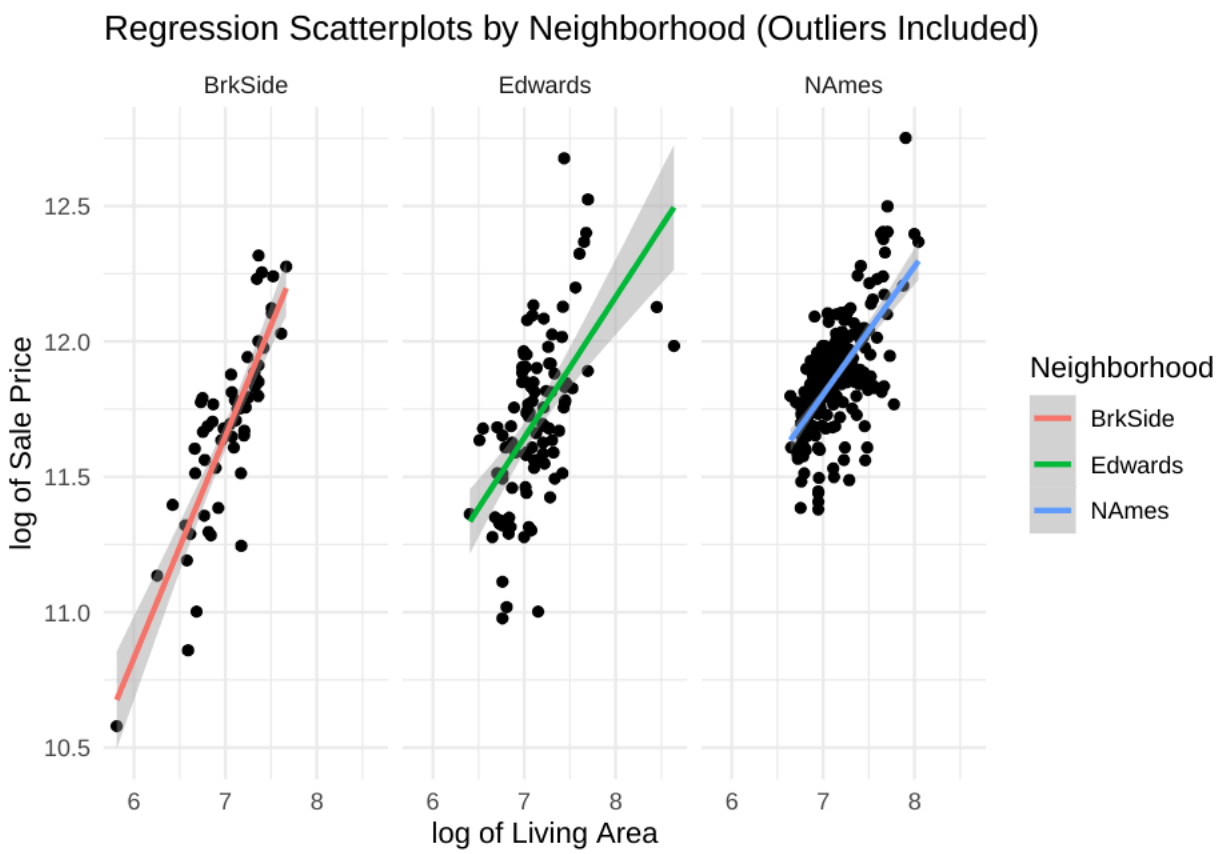
```
ggplot(outlier_remove_df, aes(x = logLiv, y = logSale, color=Neighborhood)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE) +
  labs(title = "Scatter Plot with Regression Lines all neighborhoods", x = "log of Living Area", y = "l
  theme_minimal()
```

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

Original data, Separate plots for each Neighboorhood

```
ggplot(logAmes, aes(x = logLiv, y = logSale)) +
  geom_point() +
  geom_smooth(method = "lm", se = TRUE, aes(color = Neighborhood)) +
  labs(title = "Regression Scatterplots by Neighborhood (Outliers Included)", x = "log of Living Area",
  theme_minimal() +
  facet_wrap(~Neighborhood, scales = "fixed")
```

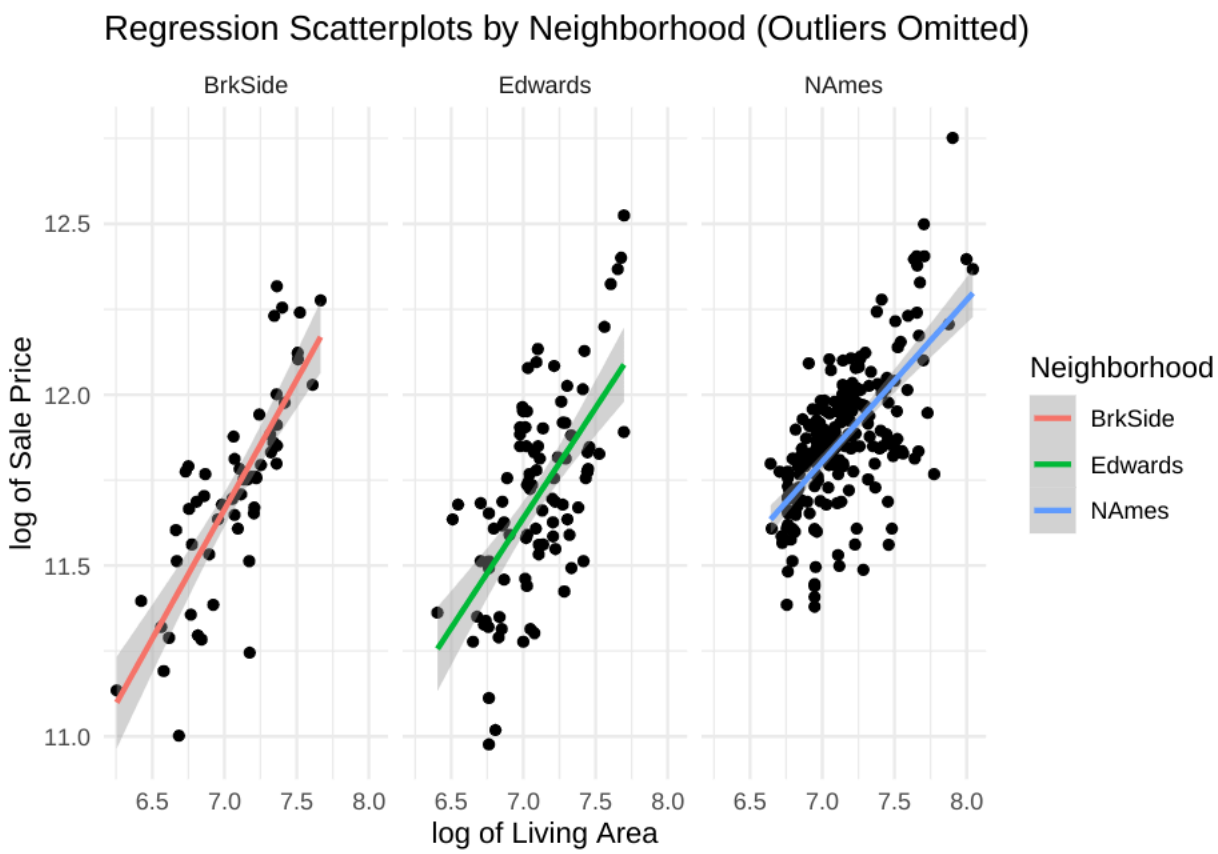
```
## `geom_smooth()` using formula = 'y ~ x'
```



Omitted data, Separate plots for each Neighboorhood

```
ggplot(outlier_remove_df, aes(x = logLiv, y = logSale)) +
  geom_point() +
  geom_smooth(method = "lm", se = TRUE, aes(color = Neighborhood)) +
  labs(title = "Regression Scatterplots by Neighborhood (Outliers Omitted)", x = "log of Living Area",
  theme_minimal() +
  facet_wrap(~Neighborhood, scales = "fixed")
```

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



Transform logmodel back to original scale for final interpretation with confidence intervals

```
summary(logModel)
```

```
##
## Call:
## lm(formula = logSale ~ Neighborhood + logLiv, data = logAmes)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.72154 -0.10592  0.02469  0.11565  0.79364
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)          7.76936    0.22919  33.900  < 2e-16 ***
## NeighborhoodEdwards -0.02044    0.03252  -0.629     0.53
## NeighborhoodNAmes    0.13279    0.02906   4.569 6.63e-06 ***
## logLiv               0.55579    0.03237  17.171  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1961 on 379 degrees of freedom
## Multiple R-squared:  0.4897, Adjusted R-squared:  0.4857
## F-statistic: 121.2 on 3 and 379 DF,  p-value: < 2.2e-16
```

```
exp(confint(logModel))
```

```
##                            2.5 %      97.5 %
```

```
## (Intercept)         1508.2764767 3714.498486
## NeighborhoodEdwards    0.9190889    1.044457
## NeighborhoodNAmes      1.0785822    1.209165
## logLiv                 1.6358194    1.857874
```

Transform omitted model back to original scale for final interpretation with confidence intervals

The data suggest that a doubling of GrLivArea with equates to a multiplicative change of 2^0.55579 in the median of the SalePrice. A 9% confidence interval for the Brookside neighborhood multiplicative increase is (2^1.6358194, 2^1.857874) ; for Edwards, we expect (2^0.9190889, 2^1.044457); and for NAmes we expect (2^1.0785822, 2^1.209165))

```
summary(ordfModel)
```

```
##
## Call:
## lm(formula = logSale ~ Neighborhood + logLiv, data = outlier_remove_df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.53562 -0.11056  0.02251  0.11389  0.48397
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)           7.72414    0.23424  32.976  < 2e-16 ***
## NeighborhoodEdwards -0.03001    0.03076  -0.975     0.33
## NeighborhoodNAmes     0.11441    0.02740   4.175 3.71e-05 ***
## logLiv                0.56470    0.03297  17.127  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1829 on 373 degrees of freedom
## Multiple R-squared:  0.4939, Adjusted R-squared:  0.4899
## F-statistic: 121.3 on 3 and 373 DF,  p-value: < 2.2e-16
```

```
exp(confint(ordfModel))
```

```
##                            2.5 %      97.5 %
## (Intercept)         1427.3132803 3585.751391
## NeighborhoodEdwards    0.9134727    1.030956
## NeighborhoodNAmes      1.0623958    1.183289
## logLiv                 1.6485055    1.876736
```

The outlier-omitted data suggest that a doubling of GrLivArea with equates to a multiplicative change of 2^0.56470 in the median of the SalePrice. A 95% confidence interval for the Brookside neighborhood multiplicative increase is (2^1.6485055, 2^1.876736) ; for Edwards, we expect (2^0.9134727, 2^1.030956); and for NAmes we expect (2^1.0623958, 2^1.183289))

Analysis 2 - Joel Laskow

```
# Train Dataset

train <- data.frame(read.csv("/cloud/project/DS 6371 Housing Project/train.csv", header=TRUE))


# Test Dataset
```

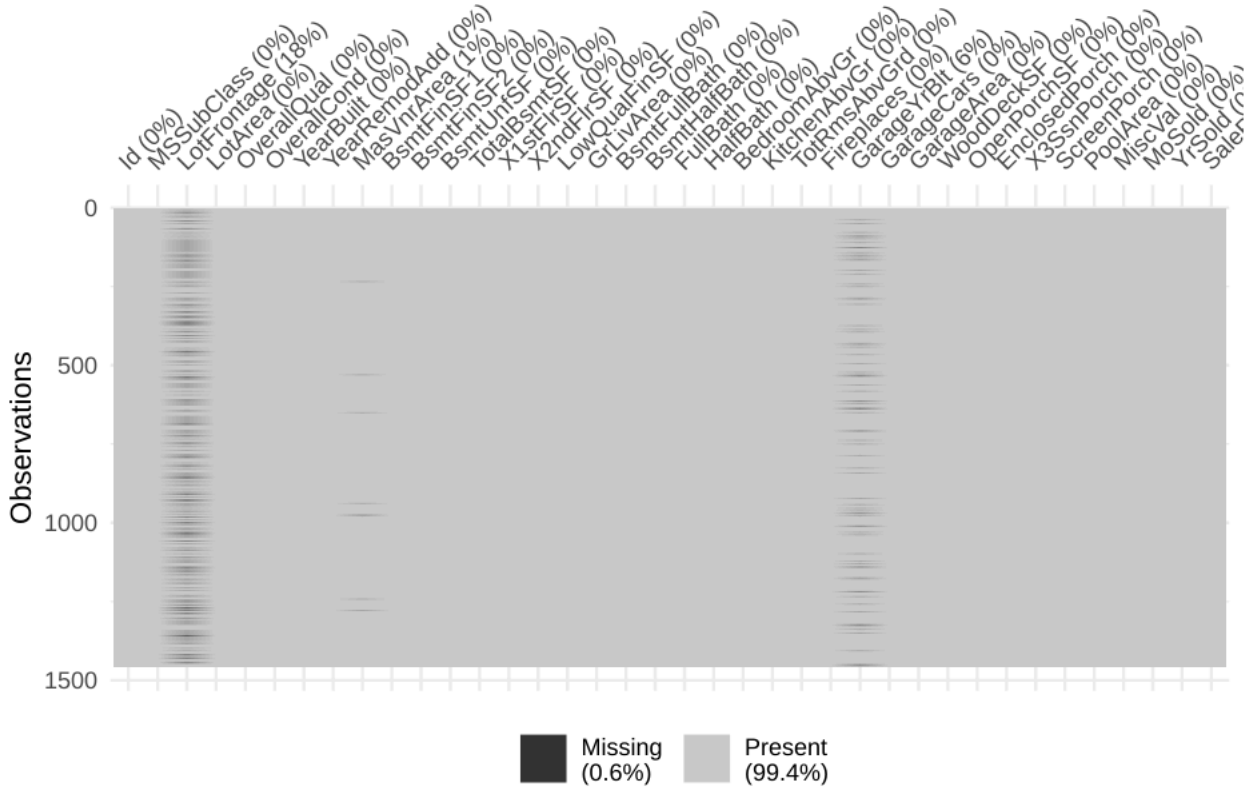
```
test <- read.csv("/cloud/project/DS 6371 Housing Project/test.csv", header=TRUE)

test<-data.frame(test)
```

Replace any instances of "NA" as a class level with "None" to avoid confusion

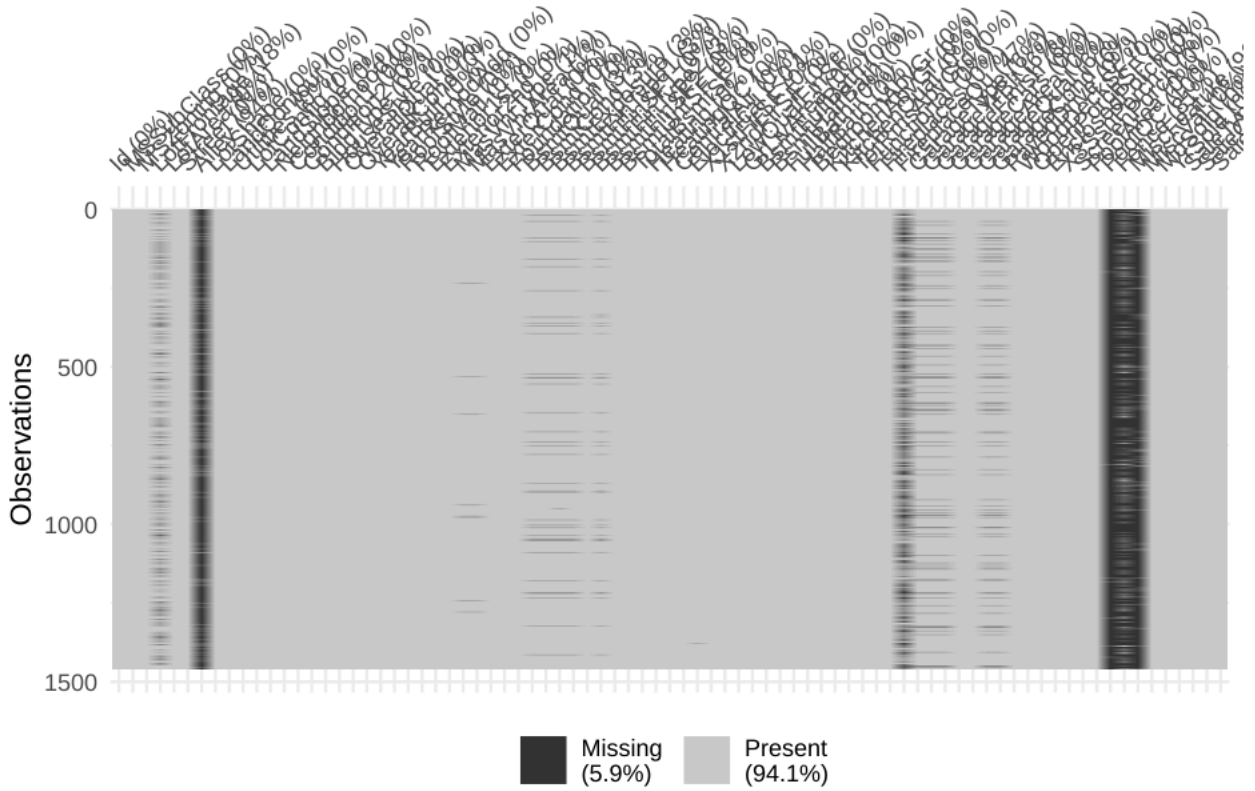
```
# train

train<- train %>%
  mutate_all(~ ifelse(. == "NA", "None", .))

# test

test<- test %>%
  mutate_all(~ ifelse(. == "NA", "None", .))
```

Assessing Numeric NA values within the datasets

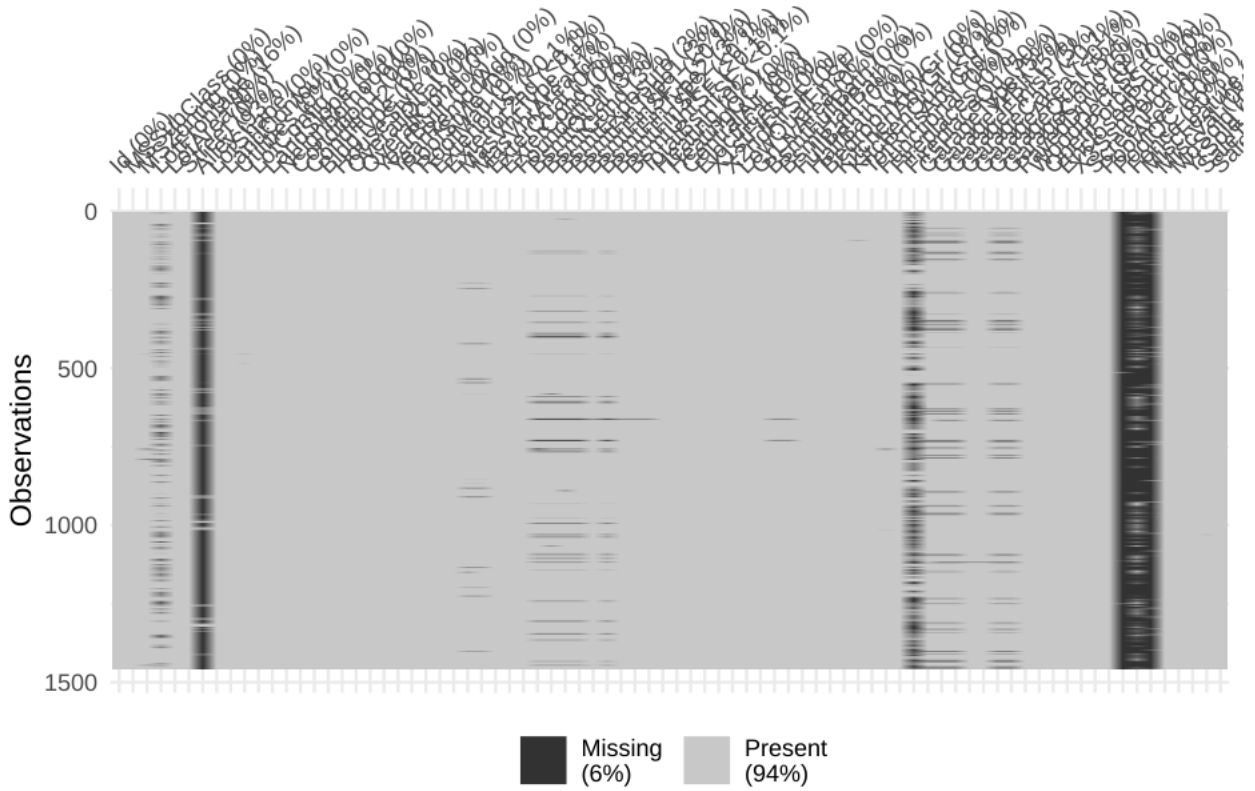
```
# training set

numerictrain<-train[sapply(train,is.numeric)]
vis_miss(numerictrain)
```



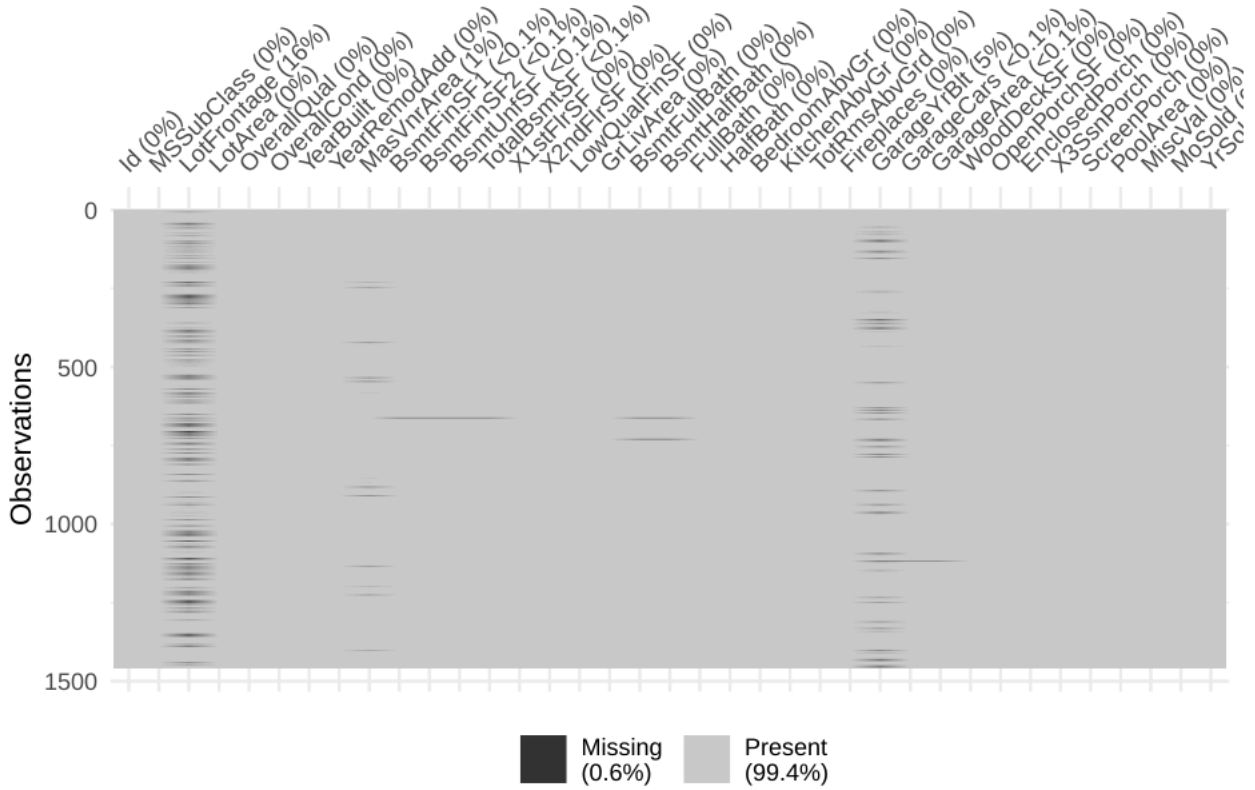
```
vis_miss(train)
```




```
# test set

numerictest<-test[sapply(test,is.numeric)]
vis_miss(test)
```




```
vis_miss(numerictest)
```

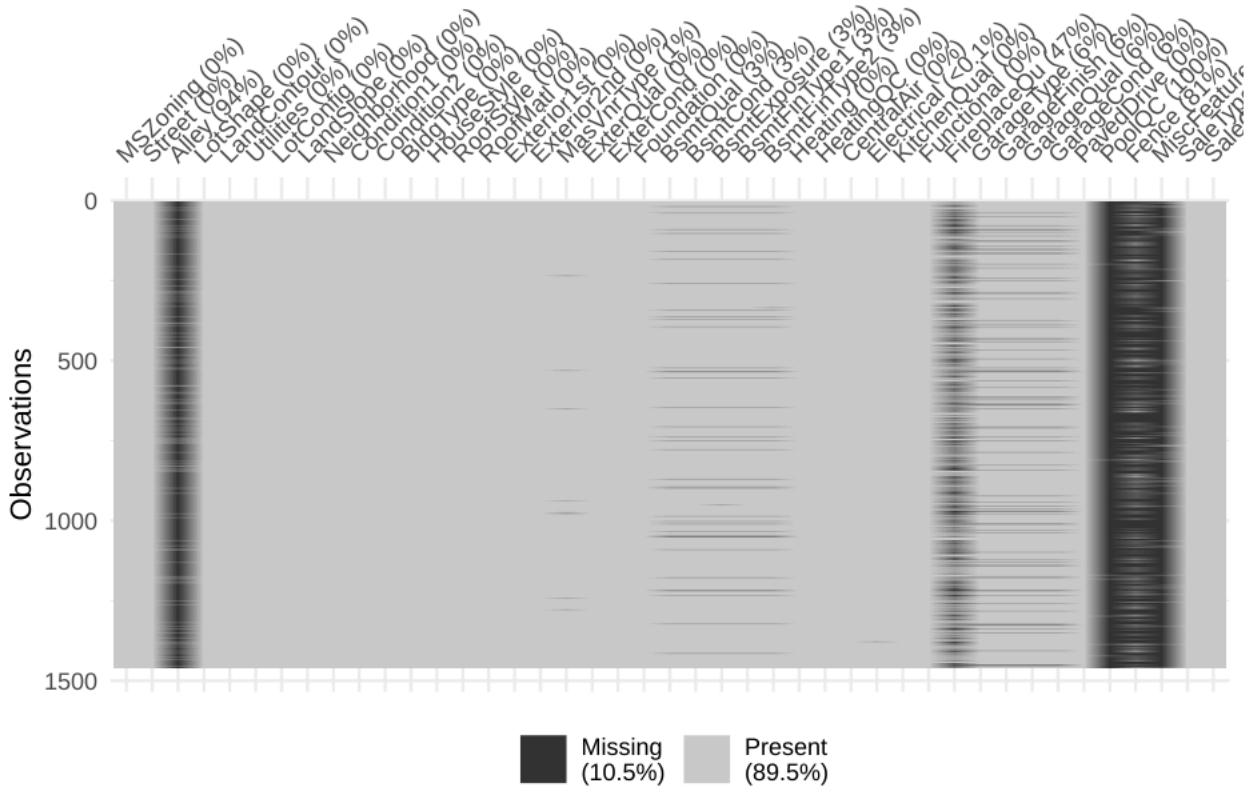


We see from missing value tests that while we're only missing 0.6% of our training set and 0.6% of our test set, we're missing 18% of our LotsFrontage data in training and 16% in testing. We're further missing 6% of

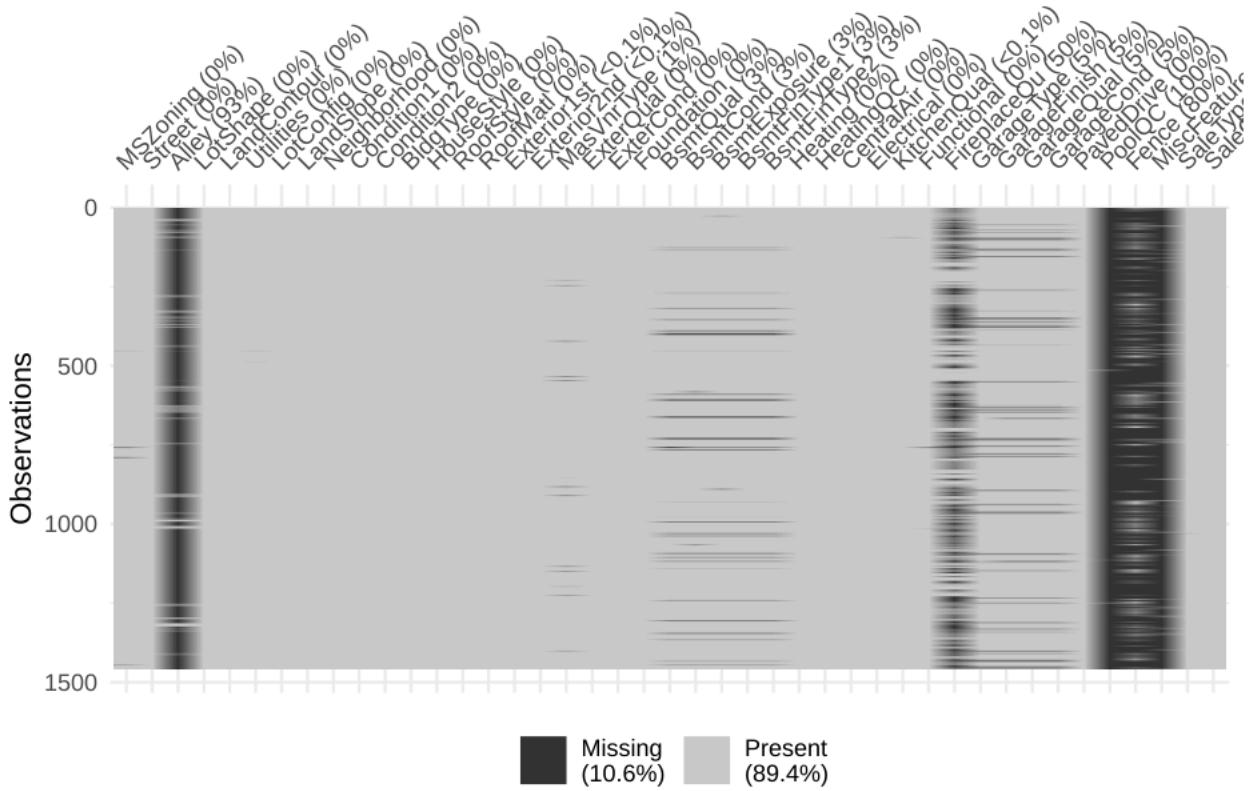
our GarageYrBlt data in the training set and 5% in the testing set. This quantity of missing errors could wildly skew our results.

Assessing Categorical NA values within the datasets

```
# train

nonnumerictrain<-train[sapply(train,is.character)]

vis_miss(nonnumerictrain)
```




```
# test

nonnumerictest<-test[sapply(test,is.character)]

vis_miss(nonnumerictest)
```



10.6% of our training dataset has missing values. Due to the severity of missing values in Alley (93%), FireplaceQu (50%), PoolQC (100%), Fence (80%), and MiscFeat we cannot impute missing values with the most common categorical level. For this reason we will remove these columns from the dataset.

```
# Removing Alley, Fireplace, PookQC, Fence, and MiscFeature from our dataset


# train

trainprime<-subset(train, select= -c(Alley, FireplaceQu, PoolQC, Fence, MiscFeature))


# test

testprime<-subset(test, select= -c(Alley, FireplaceQu, PoolQC, Fence, MiscFeature))
```

Addressing remaining NA values:

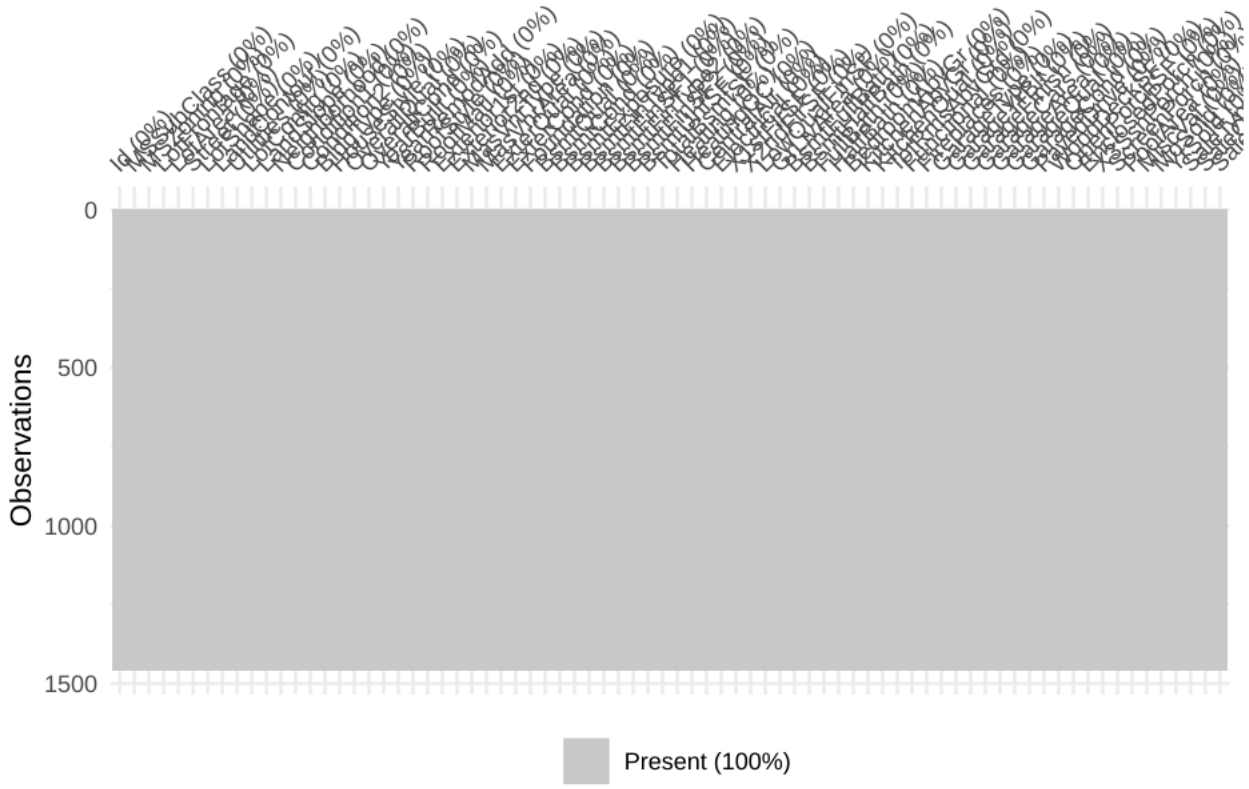
```
##### Numeric Datasets

# train

## Find the mean of columns to impute:

### LotsFrontage

x<-mean(train$LotFrontage, na.rm=TRUE)
```

```
### GarageYrBlt

y<-mean(train$GarageYrBlt, na.rm=TRUE)

### MasVnrArea

z<-mean(train$MasVnrArea, na.rm=TRUE)


## Impute mean for each column

### LotFrontage

trainprime$LotFrontage<-replace(train$LotFrontage, is.na(train$LotFrontage), x)

### GarageYrBlt

trainprime$GarageYrBlt<-replace(train$GarageYrBlt, is.na(train$GarageYrBlt), y)

### MasVnrArea

trainprime$MasVnrArea<-replace(train$MasVnrArea, is.na(train$MasVnrArea), z)

# test

## Find the mean of columns to impute:

### LotsFrontage

x<-mean(testprime$LotFrontage, na.rm=TRUE)

### GarageYrBlt

y<-mean(testprime$GarageYrBlt, na.rm=TRUE)

### MasVnrArea

z<-mean(testprime$MasVnrArea, na.rm=TRUE)

# BsmtFinSF1
d<-mean(testprime$BsmtFinSF1, na.rm=TRUE)

# BsmFinSF2
e<-mean(as.numeric(testprime$BsmFinSF2), na.rm=TRUE)

# BsmtUnfSF
f<-mean(testprime$BsmtUnfSF, na.rm=TRUE)


# TotalBsmtSF
g<-mean(testprime$TotalBsmtSF, na.rm=TRUE)

# BsmtFullBath
```

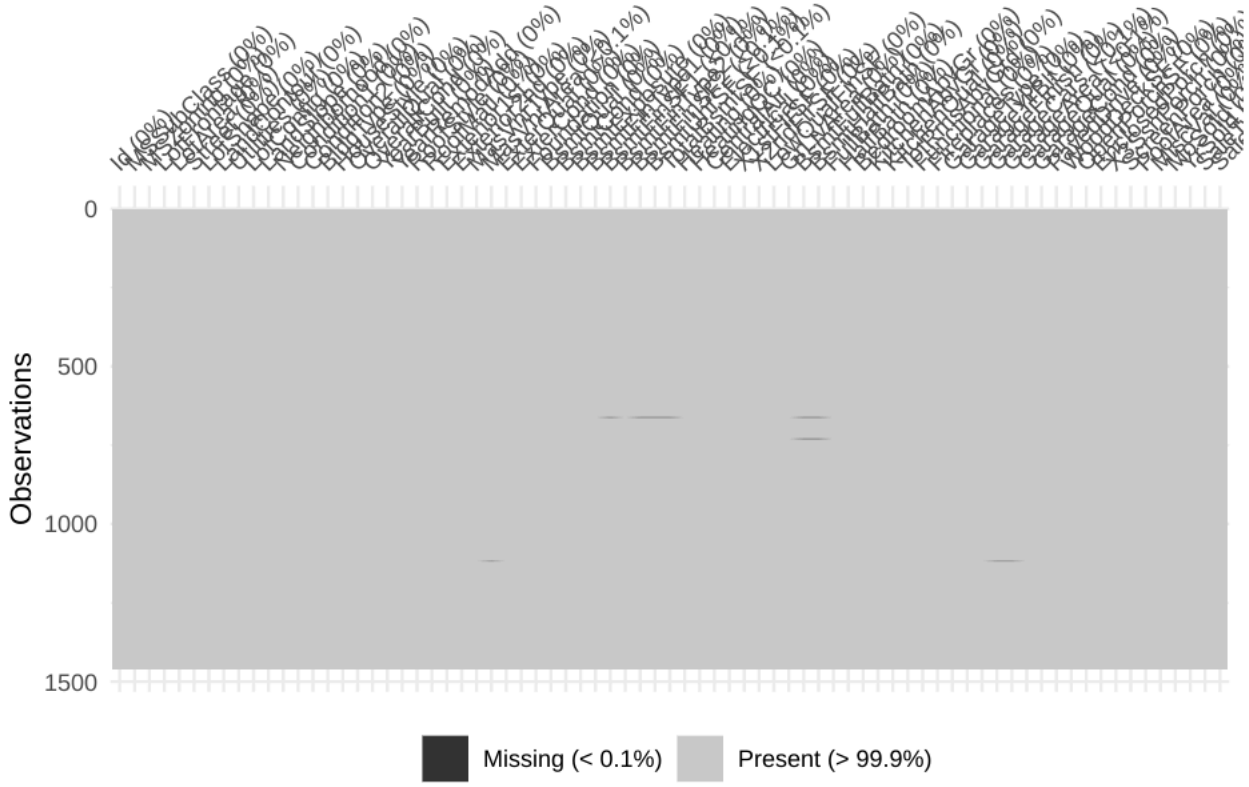
```r
h<-mean(testprime$BsmtFullBath, na.rm=TRUE)

# BsmtHalfBath
i<-mean(testprime$BsmtHalfBath, na.rm=TRUE)

# GarageCars
j<-mean(testprime$GarageCars, na.rm=TRUE)

# GarageArea
k<-mean(testprime$GarageArea, na.rm=TRUE)

## Impute mean for each column

### LotFrontage

testprime$LotFrontage<-replace(test$LotFrontage, is.na(test$LotFrontage), x)

### GarageYrBlt

testprime$GarageYrBlt<-replace(test$GarageYrBlt, is.na(test$GarageYrBlt), y)

### MasVnrArea

testprime$MasVnrArea<-replace(test$MasVnrArea, is.na(test$MasVnrArea), z)

# BsmtFinSF1
testprime$MasVnrArea<-replace(test$BsmtFinSF1, is.na(test$MasVnrArea), d)

# BsmFinSF2
testprime$MasVnrArea<-replace(as.numeric(test$BsmFinSF2), is.na(test$MasVnrArea), e)

# BsmtUnfSF
testprime$MasVnrArea<-replace(test$BsmtUnfSF, is.na(test$MasVnrArea), f)


# TotalBsmtSF
testprime$MasVnrArea<-replace(test$TotalBsmtSF, is.na(test$MasVnrArea), g)


# BsmtFullBath
testprime$MasVnrArea<-replace(test$BsmtFullBath, is.na(test$MasVnrArea), h)


# BsmtHalfBath
testprime$MasVnrArea<-replace(test$BsmtHalfBath, is.na(test$MasVnrArea), i)


# GarageCars

testprime$MasVnrArea<-replace(test$GarageCars, is.na(test$MasVnrArea), j)

# GarageArea
```

```r
testprime$MasVnrArea<-replace(test$GarageArea, is.na(test$MasVnrArea), k)

##### Categorical (Non-numeric)

# trainnew

get_mode <- function(v) {
  uniqv <- unique(v)
  uniqv[which.max(tabulate(match(v, uniqv)))]
}

for (col in names(trainprime)) {
  if (is.character(trainprime[[col]]) && anyNA(trainprime[[col]])) {
    trainprime[[col]][is.na(trainprime[[col]])] <- get_mode(trainprime[[col]])
  }
}

# testnew
get_mode <- function(v) {
  uniqv <- unique(v)
  uniqv[which.max(tabulate(match(v, uniqv)))]
}

for (col in names(testprime)) {
  if (is.character(testprime[[col]]) && anyNA(testprime[[col]])) {
    testprime[[col]][is.na(testprime[[col]])] <- get_mode(testprime[[col]])
  }
}

trainduplicate<-trainprime
testduplicate<-testprime
```

```r
vis_miss(trainprime)
```

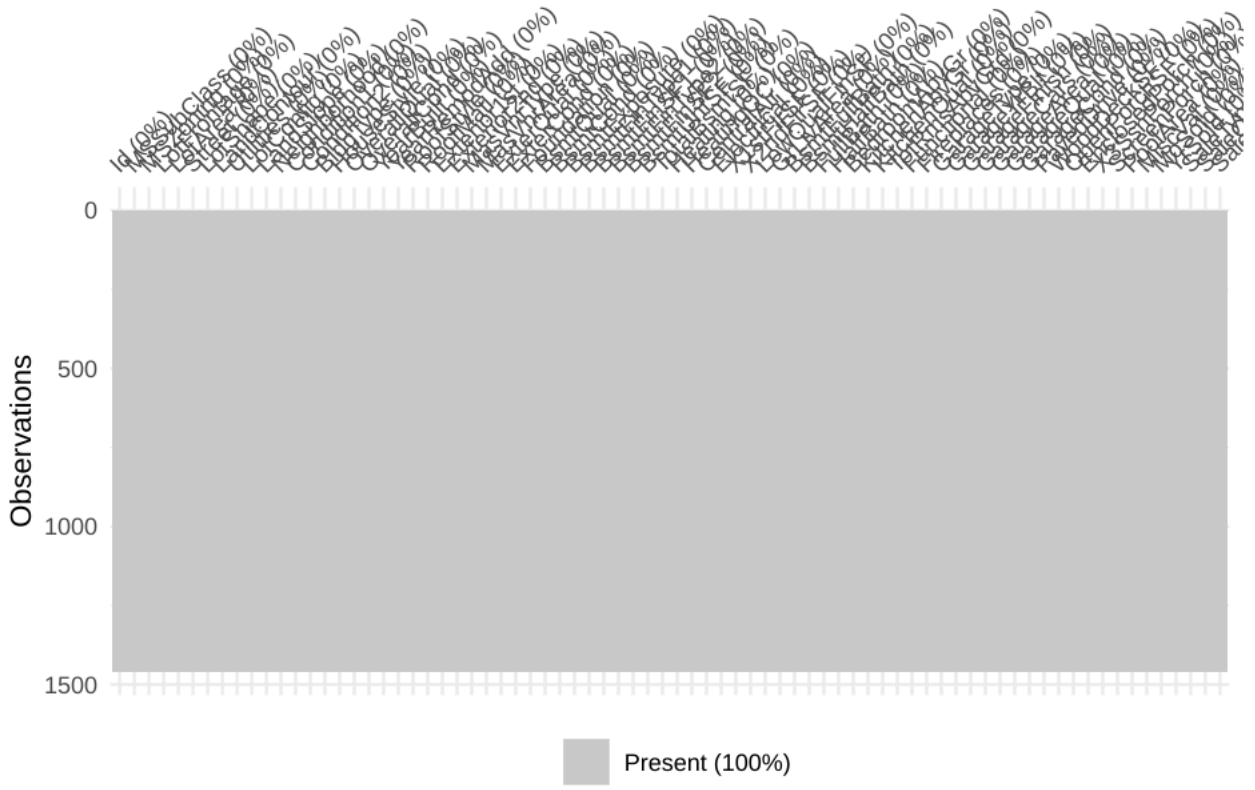



```
vis_miss(testprime)
```




```
# trainnew
```

```r
get_mode <- function(v) {
  uniqv <- unique(v)
  uniqv[which.max(tabulate(match(v, uniqv)))]
}

for (col in names(trainprime)) {
  if (is.character(trainprime[[col]]) && anyNA(trainprime[[col]])) {
    trainprime[[col]][is.na(trainprime[[col]])] <- get_mode(trainprime[[col]])
  }
}

# Mark MSSubClass as factor

trainprime$MSSubClass<-factor(trainprime$MSSubClass)
testprime$MSSubClass<-factor(testprime$MSSubClass)


vis_miss(trainprime)
```



Our datasets are now clear of NA values

Unequal level between train and test data

If we compare the levels of testprime and trainprime within the MSSubClass feature, we see testprime has a level that does not appear in the training set.

```r
levels(testprime$MSSubClass)
```

```
##  [1] "20"  "30"  "40"  "45"  "50"  "60"  "70"  "75"  "80"  "85"  "90"  "120"
```

```
## [13] "150" "160" "180" "190"
```

```
levels(trainprime$MSSubClass)
```

```
##  [1] "20"  "30"  "40"  "45"  "50"  "60"  "70"  "75"  "80"  "85"  "90"  "120"
## [13] "160" "180" "190"
```

Level 150 appears only in the test set

```
subset(testprime, MSSubClass == "150")
```

```
##        Id MSSubClass MSZoning LotFrontage LotArea Street LotShape LandContour
## 1359 2819        150       RL    68.58036    1700   Pave      Reg         HLS
##      Utilities LotConfig LandSlope Neighborhood Condition1 Condition2 BldgType
## 1359    AllPub    Inside       Gtl      ClearCr       Norm       Norm    Twnhs
##      HouseStyle OverallQual OverallCond YearBuilt YearRemodAdd RoofStyle
## 1359     1.5Fin           7           5      1980         1981     Gable
##      RoofMatl Exterior1st Exterior2nd MasVnrType MasVnrArea ExterQual ExterCond
## 1359  CompShg     VinylSd     VinylSd       None        450        Gd        TA
##      Foundation BsmtQual BsmtCond BsmtExposure BsmtFinType1 BsmtFinSF1
## 1359      PConc       Gd       TA           Mn          GLQ        397
##      BsmtFinType2 BsmtFinSF2 BsmtUnfSF TotalBsmtSF Heating HeatingQC CentralAir
## 1359          Unf          0        33         430    GasA        TA          Y
##      Electrical X1stFlrSF X2ndFlrSF LowQualFinSF GrLivArea BsmtFullBath
## 1359      SBrkr       880       680          140      1700            1
##      BsmtHalfBath FullBath HalfBath BedroomAbvGr KitchenAbvGr KitchenQual
## 1359            0        2        1            2           1          Gd
##      TotRmsAbvGrd Functional Fireplaces GarageType GarageYrBlt GarageFinish
## 1359            7        Typ          0    Basment        1980          Fin
##      GarageCars GarageArea GarageQual GarageCond PavedDrive WoodDeckSF
## 1359          1        450         Gd         TA          Y        188
##      OpenPorchSF EnclosedPorch X3SsnPorch ScreenPorch PoolArea MiscVal MoSold
## 1359          36             0          0         200        0       0      4
##      YrSold SaleType SaleCondition
## 1359   2006       WD        Normal
```

Only one row in the test set contains MSSubClass of 150. Descriptor: 150 1-1/2 STORY PUD - ALL AGES

Because of the descriptor similarity to level 50 (50 1-1/2 STORY FINISHED ALL AGES), the level of this row was changed

```
# Before change
table(testprime$MSSubClass)
```

```
##
##  20  30  40  45  50  60  70  75  80  85  90 120 150 160 180 190
## 543  70   2   6 143 276  68   7  60  28  57  95   1  65   7  31
```

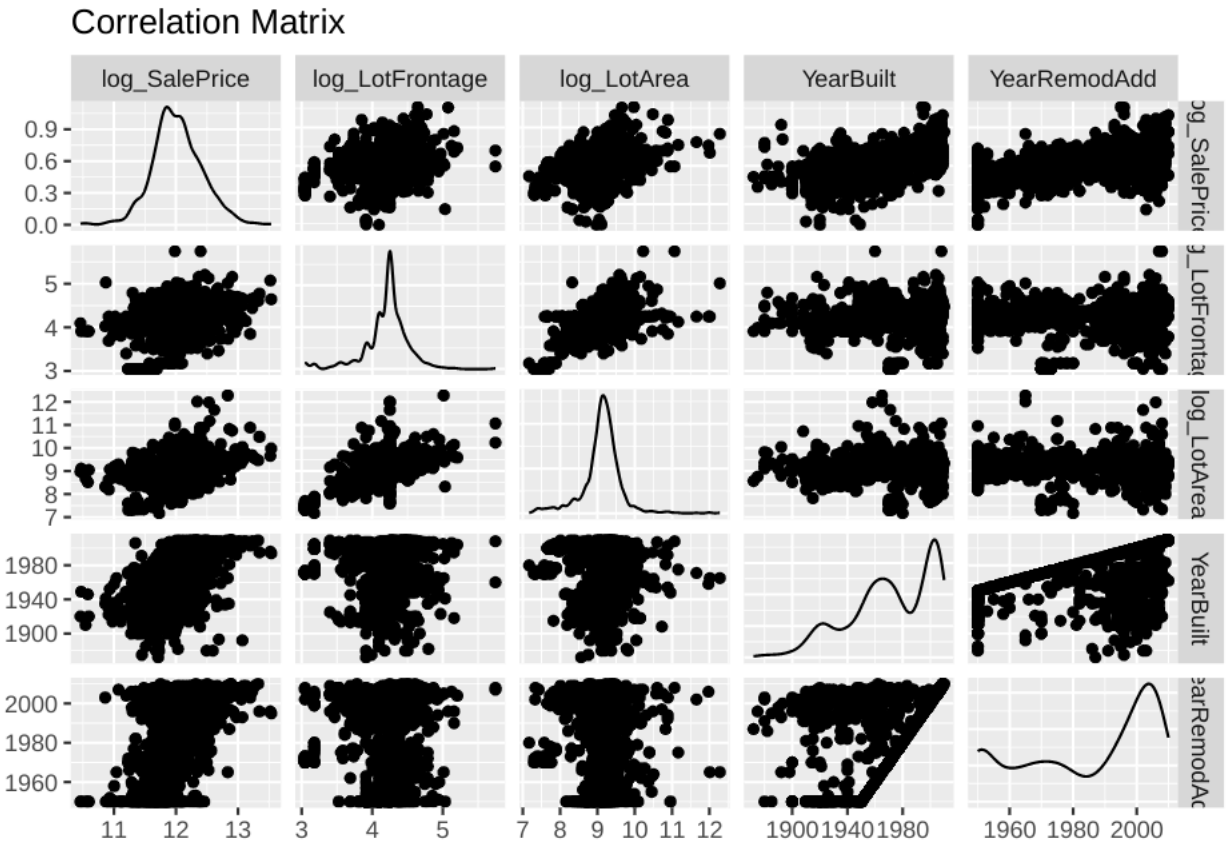
```
row_to_change <- which(testprime$MSSubClass == "150")

# Change the level to 50
testprime$MSSubClass[row_to_change] <- "50"

# Verify the change
table(testprime$MSSubClass)
```

```
##
##  20  30  40  45  50  60  70  75  80  85  90 120 150 160 180 190
```

```
## 543  70   2   6 144 276  68   7  60  28  57  95   0  65   7  31
```

If we are to build an efficient model, we must be selective with our variables. We have 3 methods to select our variables, all of which will be discussed later: Forward Selection, Backward Selection, and Stepwise Selection.

```
trainnew <- trainprime %>%
  mutate(log_SalePrice = log(SalePrice)) %>%
  mutate(log_LotFrontage = log(LotFrontage)) %>%
  mutate(log_LotArea = log(LotArea))

selected_vars <- c("log_SalePrice", "log_LotFrontage", "log_LotArea", "YearBuilt", "YearRemodAdd")

# Calculating correlations

subset_data <- trainnew[, selected_vars]

# Create correlation plot using ggpairs
ggpairs(subset_data, upper=list (continuous="points"), title = "Correlation Matrix")
```



High correlation between LotArea and LotFrontage, YearBuilt and YearRemodAdd.

Variables to remove: LotFrontage, YearBuilt

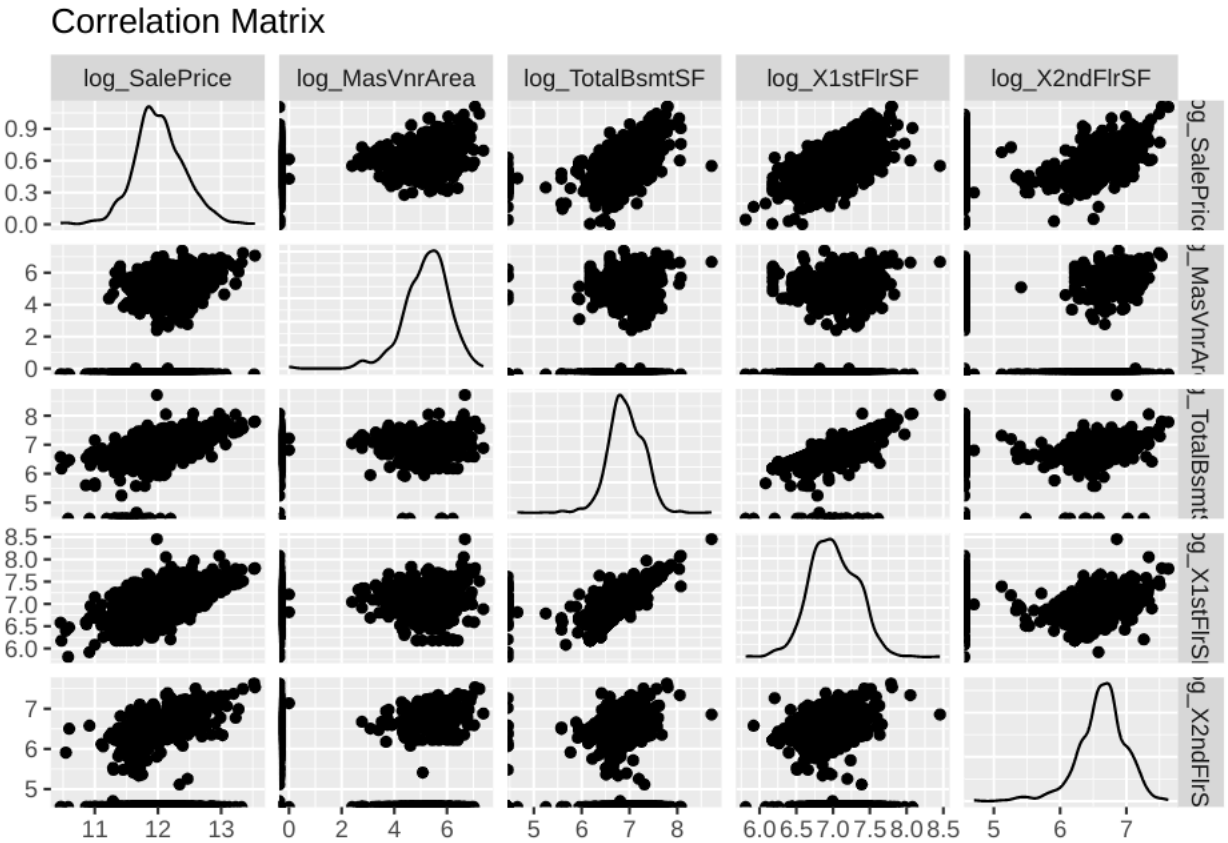
Next Matrix Batch

```
trainnew <- trainprime %>%
  mutate(log_SalePrice = log(SalePrice)) %>%
  mutate(log_MasVnrArea = log(MasVnrArea)) %>%
  mutate(log_TotalBsmtSF = log(TotalBsmtSF))%>%
  mutate(log_X1stFlrSF = log(X1stFlrSF))%>%
  mutate(log_X2ndFlrSF = log(X2ndFlrSF))

selected_vars <- c("log_SalePrice", "log_MasVnrArea", "log_TotalBsmtSF", "log_X1stFlrSF", "log_X2ndFlrS

# Calculating correlations

subset_data <- trainnew[, selected_vars]

# Create correlation plot using ggpairs
ggpairs(subset_data, upper=list (continuous="points"), title = "Correlation Matrix")
```

```
## Warning: Removed 861 rows containing non-finite outside the scale range
## (`stat_density()`).
```

```
## Warning: Removed 37 rows containing non-finite outside the scale range
## (`stat_density()`).
```

```
## Warning: Removed 829 rows containing non-finite outside the scale range
## (`stat_density()`).
```

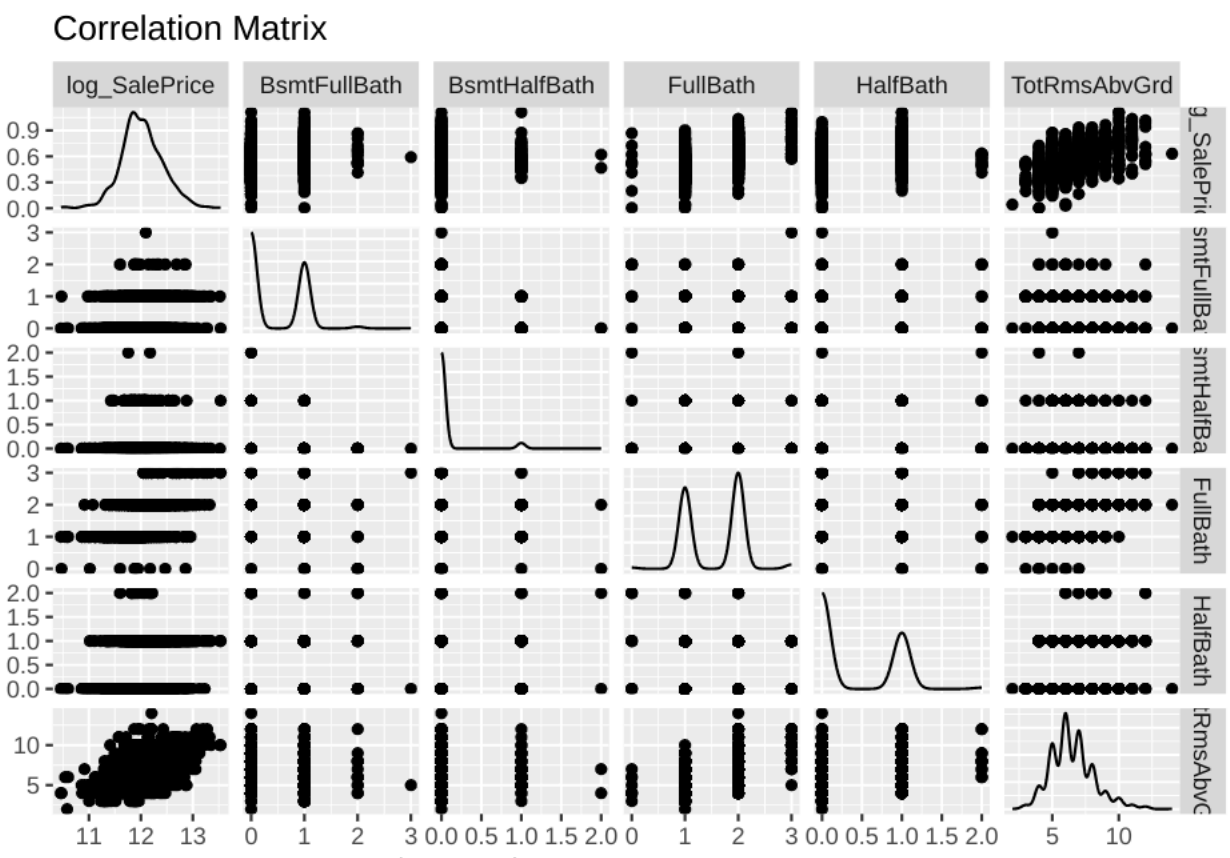


High correlation between total TotalBsmntSF and 1stFlrSF. Minimal correlation between MasVnrArea and Sale Price.

Variables to remove: TotalBsmntSF, MasVnrArea

3rd Matrix Batch

```
# Deal with outliers by logging large-value columns

# NoteL LowQualFinSF ignored

# GrLivArea ignored, similar to 1stFlrSF but only accounts for living area

# BedroomAbvGr and KitchenAbvGr ignored, similar to TotRmsAbvGr


# Log transform 'SalePrice' column
numerictrainnew <- trainprime %>%
  mutate(log_SalePrice = log(SalePrice))

# Select necessary variables
selected_vars <- c("log_SalePrice", "BsmtFullBath", "BsmtHalfBath", "FullBath", "HalfBath", "TotRmsAbvG

# Create a subset of data with selected variables
subset_data <- numerictrainnew[, selected_vars]

# Create correlation plot using ggpairs
ggpairs(subset_data, upper=list (continuous="points"), title = "Correlation Matrix")
```



No correlation seen between log(SalePrice) and BsmtFullBath, BsmtHalfBath, and HalfBath. Possible

positive correlation observed between FullBath and TotRmsAbvGrd

Variables to remove: BsmtFullBath, BsmtHalfBath, and HalfBath

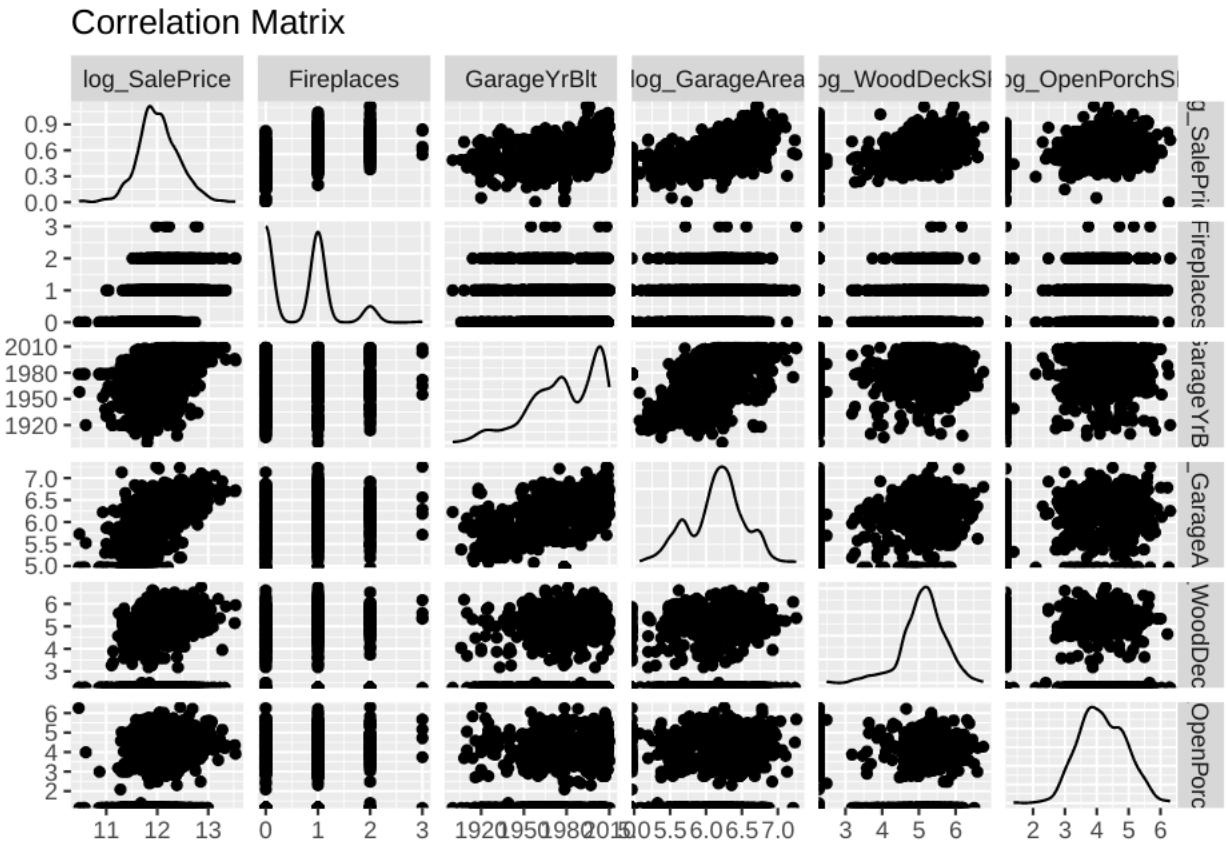
4th Matrix Batch

```
# Log transform 'SalePrice' column
numerictrainnew <- trainprime %>%
  mutate(log_SalePrice = log(SalePrice))%>% mutate(log_GarageArea = log(GarageArea)) %>% mutate(log_Woo

# Select necessary variables

## GarageCars removed, similar to GarageArea

selected_vars <- c("log_SalePrice", "Fireplaces", "GarageYrBlt", "log_GarageArea", "log_WoodDeckSF", "l

# Create a subset of data with selected variables
subset_data <- numerictrainnew[, selected_vars]

# Create correlation plot using ggpairs
ggpairs(subset_data, upper=list (continuous="points"), title = "Correlation Matrix")
```

```
## Warning: Removed 81 rows containing non-finite outside the scale range
## (`stat_density()`).
```

```
## Warning: Removed 761 rows containing non-finite outside the scale range
## (`stat_density()`).
```

```
## Warning: Removed 656 rows containing non-finite outside the scale range
## (`stat_density()`).
```



High correlation between GargeYrBlt and and log_GarageArea; both show high correlation with log_SalePrice. log_GarageArea removed for simplicity.

We also see evidence of correlation between Fireplaces and log_SalePrice

Slight correlation observed between log_WoodDeckSF and log_SalePrice, and log_OpenPorchSF and log_SalePrice. Due to the weak correlation, both WoodDeckSF and OpenPorchSF will be removed.

The following variables were also ignored due to the high prevlance of 0 (i.e., "Not Applicable" or "None") within the column. There does not appear to be a sufficient quantity of values to make an informed interpretation from these variables.

- EnclosedPorch
- X3SsnPorch
- PoolArea
- MiscVal

At this point we have the following numerical variables left:

- FirePlaces
- GarageYrBlt
- FullBath
- TotRmsAbvGrd
- log(1stFlrSF)
- log(LotArea)

- YearRemodAdd

We will move forward with the following categorical variable from the training set:

- MSSubClass

```
nrow(testprime)
```

```
## [1] 1459
```

Subdividing our training set (80:20) to produce a validation and training set

```
# We will split out training dataset to train our model and test its quality
indices <- sample(1:nrow(trainprime), 0.8 * nrow(trainprime))

# Creating the training and validation sets
training_set <- trainprime[indices, ]  # 80% of data for training
validation_set <- trainprime[-indices, ]  # Remaining 20% for validation
```

All variable selection was performed in SAS using the trainprime dataset. Final CV Press and Adjusted R2 Values are shown below for each selection method.

Forward: - No Variables Removed - CV Press: 54.84692 - Adj. R2: 0.7691

Backward: - TotRmsAbvGrd suggested for removal - Candidate CV Press: 55.1482 - Compare CV Press: 55.0320 - Adj. R2: 0.7691

Stepwise: - No variables removed - CV Press: 55.05540 - Adj. R2: 0.7691

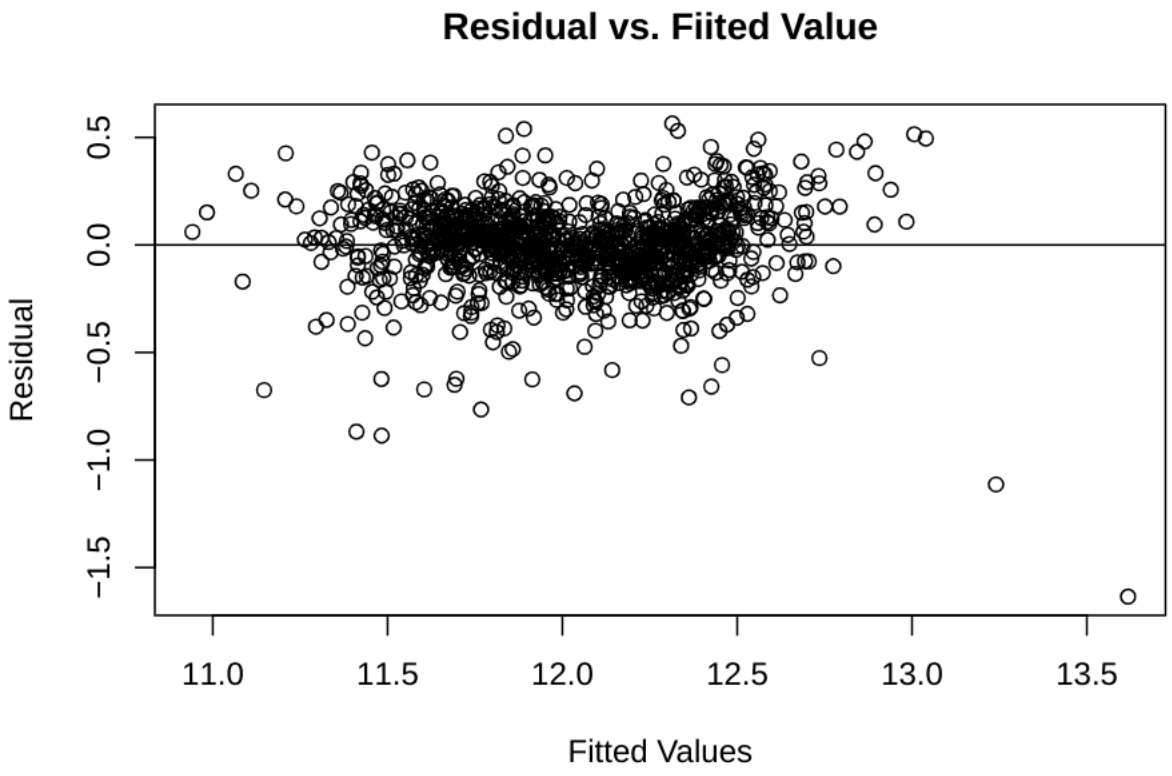
We will move forward with the full model (no variables removed). Removal of TotRmsAbvGrd provides miminimal change in CV Press.

Full Model:

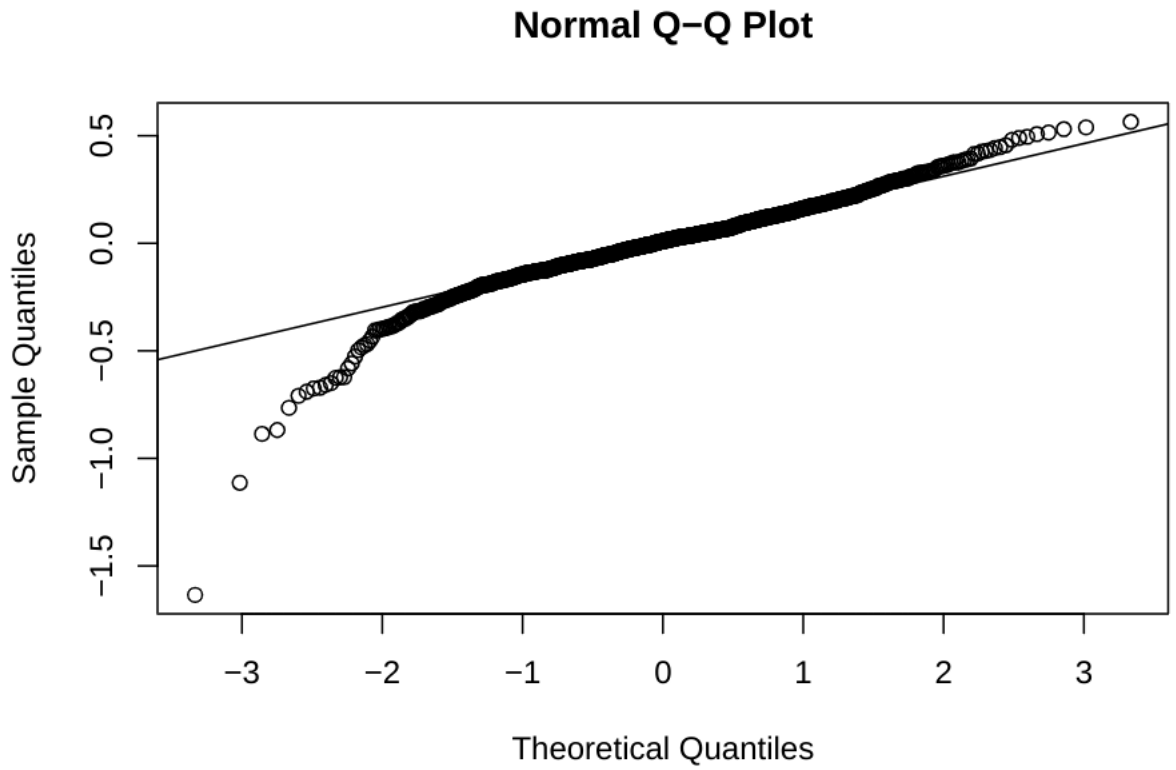
log(SalePrice) = Fireplaces + GarageYrBlt + FullBath + TotRmsAbvGrd + log(X1stFlrSF) + log(LotArea) + YearRemodAdd + MSSubClass

Assumption checks for final model

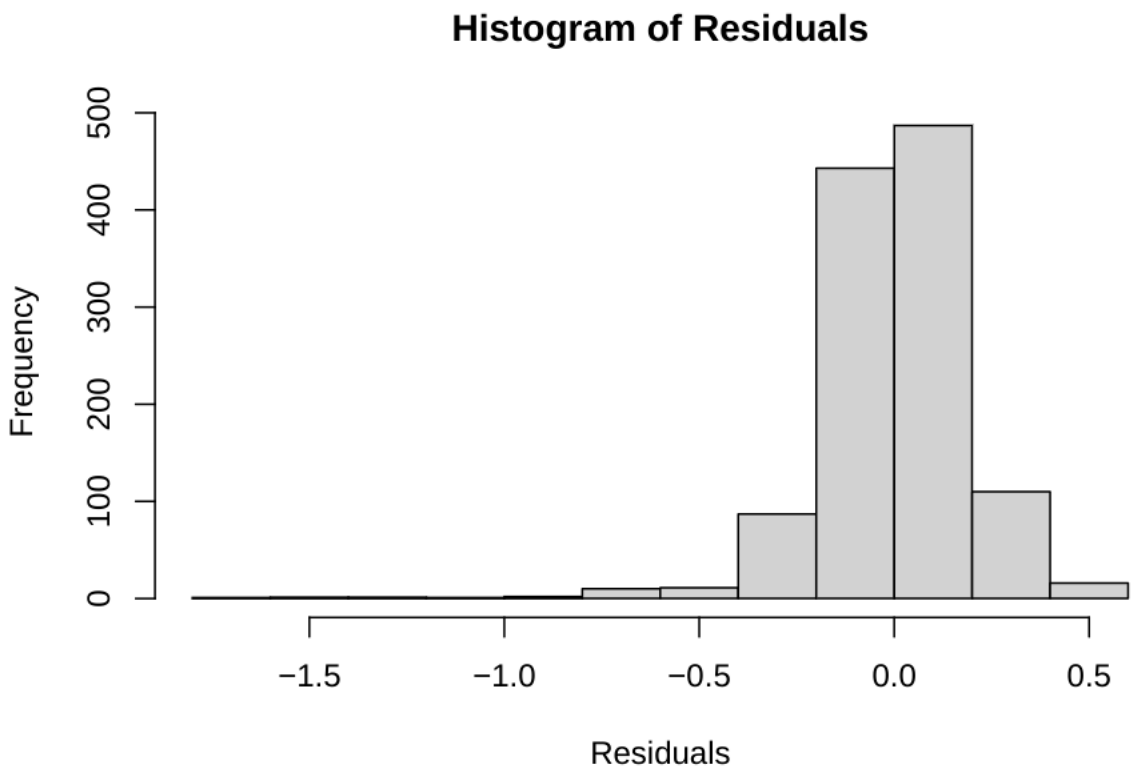
```
fitfull<-lm(log(SalePrice) ~ Fireplaces + GarageYrBlt + FullBath + TotRmsAbvGrd + log(X1stFlrSF) + log(


# Residuals plot

res<-resid(fitfull)

plot(fitted(fitfull), res, xlab="Fitted Values", ylab="Residual", main = "Residual vs. Fiited Value")
abline(0,0)
```




```r
# QQ plot
qqnorm(resid(fitfull))
qqline(resid(fitfull))
```

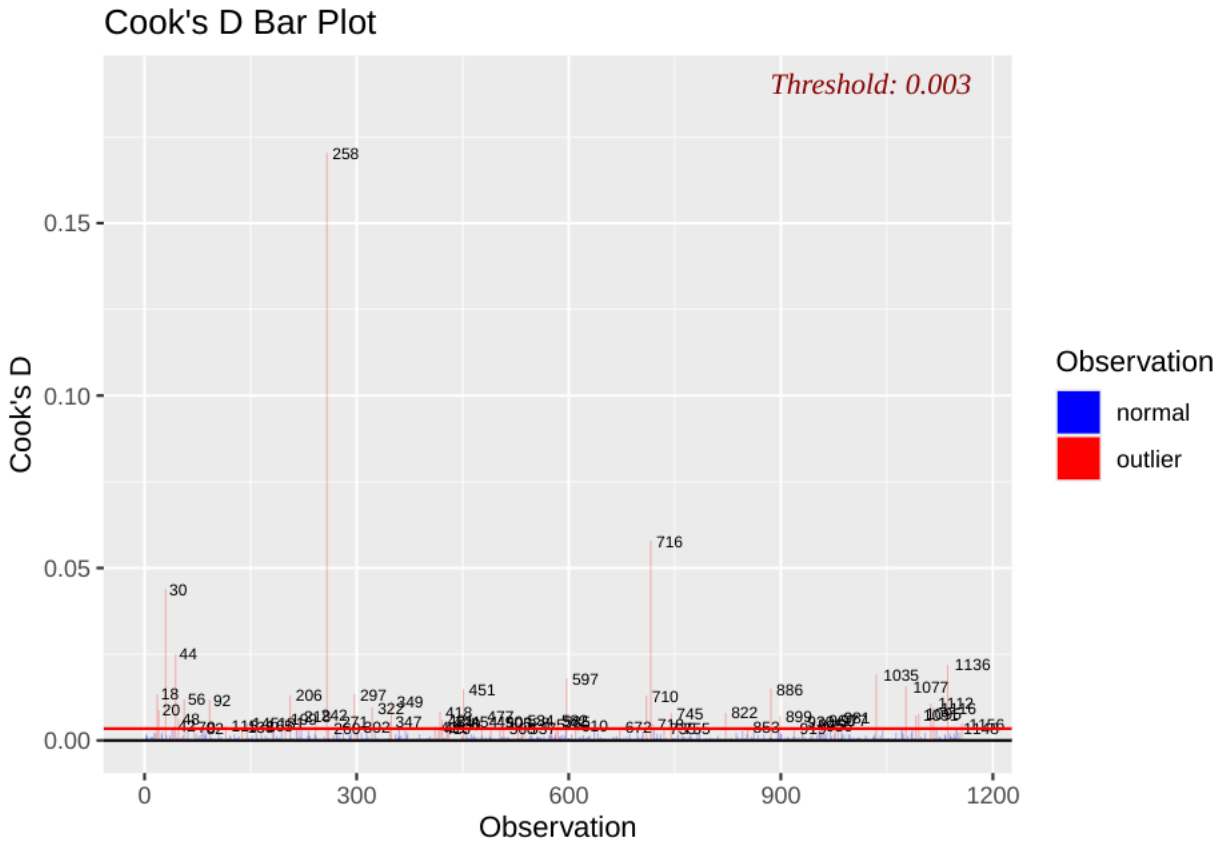


```r
# Histogram of residuals
hist(resid(fitfull), xlab="Residuals", ylab="Frequency", main="Histogram of Residuals")
```

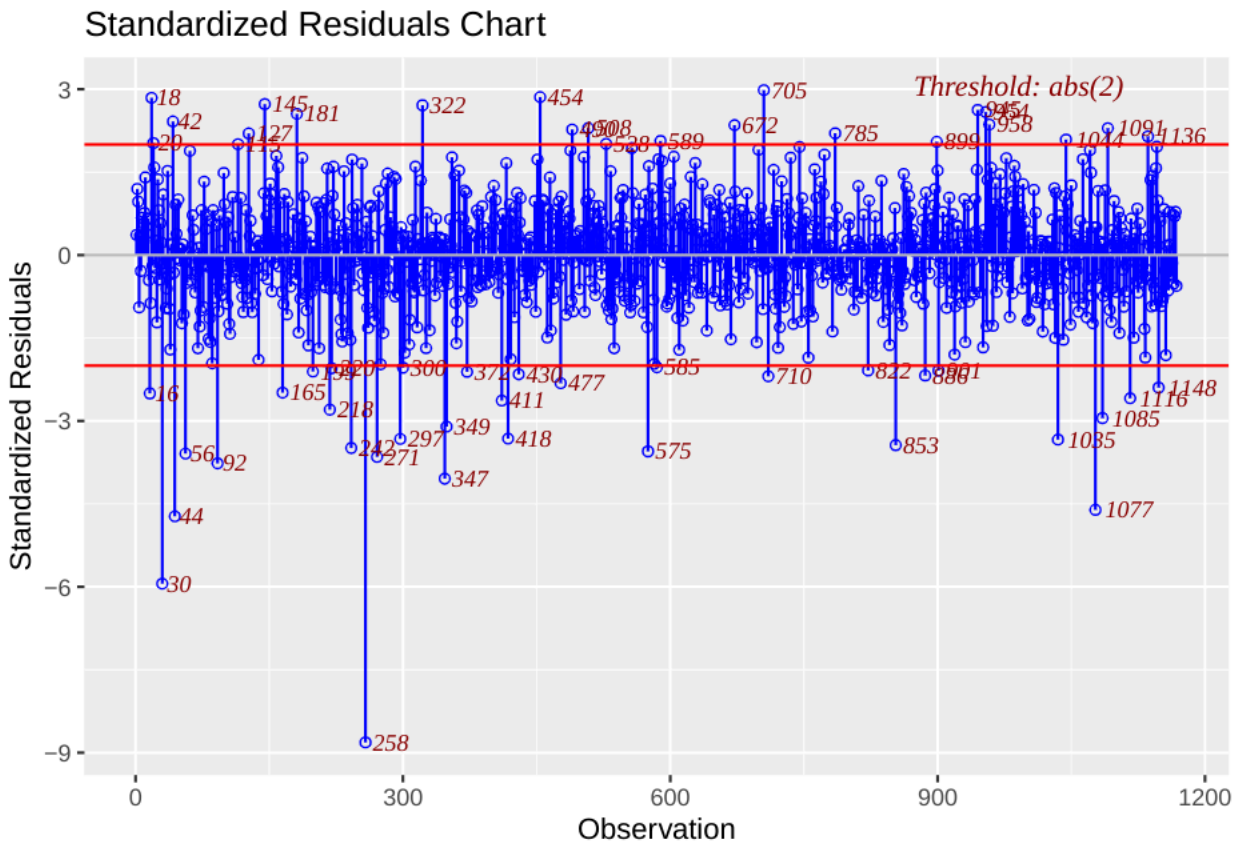


Visualizing Leverage

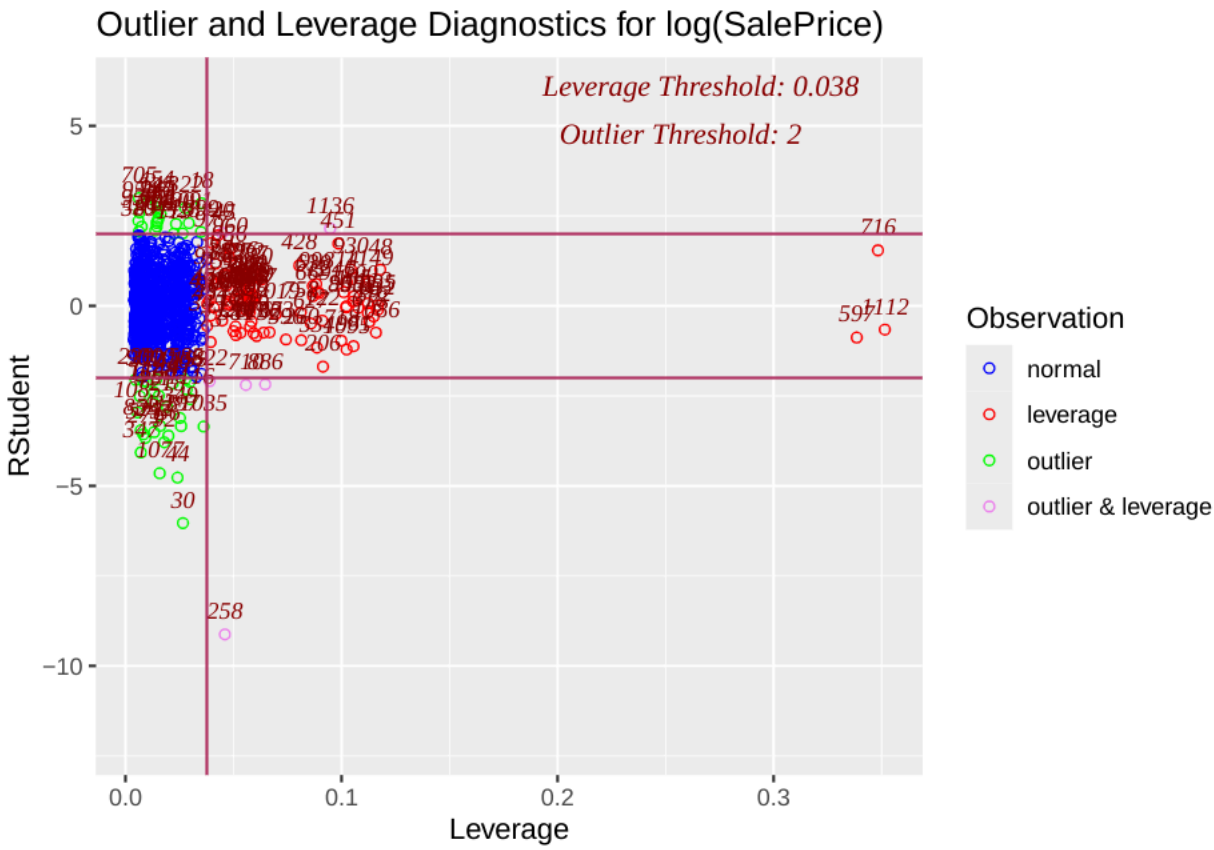
```r
ols_plot_cooksd_bar(fitfull)
```




```
ols_plot_resid_stand(fitfull)
```




```
ols_plot_resid_lev(fitfull)
```



Testing Adjusted R Square Change with Leverage Points Removed

```
# Make new dataet with high-leverage outliers removed

trainnew_no_leverage <- trainprime[-51, -513, -859]


# Perform 80:20 split with edited dataset
train_indices <- sample(nrow(trainnew_no_leverage), 0.8 * nrow(trainnew_no_leverage))  # 80% for traini
train_data_no_outliers <- trainnew_no_leverage[train_indices, ]
test_data_no_outliers <- trainnew_no_leverage[-train_indices, ]  # Remaining 20% for testing


# Refit the model without high leverage points

fit_no_leverage <- lm(fitfull, data = train_data_no_outliers)

fit_no_leverage
```

```
##
## Call:
## lm(formula = fitfull, data = train_data_no_outliers)
##
## Coefficients:
##    (Intercept)      Fireplaces     GarageYrBlt        FullBath    TotRmsAbvGrd
##      -6.379161        0.090199        0.002504        0.054953        0.013770
## log(X1stFlrSF)    log(LotArea)    YearRemodAdd    MSSubClass30    MSSubClass40
##       0.560029        0.108132        0.004148       -0.172604        0.090989
```

```
##   MSSubClass45    MSSubClass50    MSSubClass60    MSSubClass70    MSSubClass75
##       0.068935        0.072181        0.257455        0.217647        0.147455
##   MSSubClass80    MSSubClass85    MSSubClass90   MSSubClass120   MSSubClass160
##       0.040470        0.078155       -0.141590        0.079140        0.251817
##  MSSubClass180   MSSubClass190
##       0.110467       -0.046175
```

```
summary(fit_no_leverage)
```

```
##
## Call:
## lm(formula = fitfull, data = train_data_no_outliers)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.88001 -0.09959  0.00863  0.10808  0.57410
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -6.3791613  0.7698810  -8.286 3.25e-16 ***
## Fireplaces      0.0901986  0.0102934   8.763  < 2e-16 ***
## GarageYrBlt     0.0025037  0.0003484   7.185 1.21e-12 ***
## FullBath        0.0549535  0.0146150   3.760 0.000178 ***
## TotRmsAbvGrd    0.0137698  0.0058592   2.350 0.018936 *
## log(X1stFlrSF)  0.5600294  0.0316312  17.705  < 2e-16 ***
## log(LotArea)    0.1081317  0.0170175   6.354 3.02e-10 ***
## YearRemodAdd    0.0041481  0.0003582  11.581  < 2e-16 ***
## MSSubClass30   -0.1726039  0.0284539  -6.066 1.78e-09 ***
## MSSubClass40    0.0909893  0.1096770   0.830 0.406932
## MSSubClass45    0.0689354  0.0644441   1.070 0.284983
## MSSubClass50    0.0721813  0.0231543   3.117 0.001870 **
## MSSubClass60    0.2574552  0.0235504  10.932  < 2e-16 ***
## MSSubClass70    0.2176466  0.0345347   6.302 4.18e-10 ***
## MSSubClass75    0.1474550  0.0584758   2.522 0.011816 *
## MSSubClass80    0.0404696  0.0291956   1.386 0.165971
## MSSubClass85    0.0781552  0.0472274   1.655 0.098225 .
## MSSubClass90   -0.1415904  0.0337095  -4.200 2.87e-05 ***
## MSSubClass120   0.0791404  0.0288139   2.747 0.006116 **
## MSSubClass160   0.2518174  0.0388247   6.486 1.31e-10 ***
## MSSubClass180   0.1104668  0.0890402   1.241 0.214993
## MSSubClass190  -0.0461746  0.0423077  -1.091 0.275326
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1887 on 1145 degrees of freedom
## Multiple R-squared:  0.7831, Adjusted R-squared:  0.7792
## F-statistic: 196.9 on 21 and 1145 DF,  p-value: < 2.2e-16
```

```
fit_with_leverage <- lm(fitfull, data = training_set)

summary(fit_with_leverage)
```

```
##
## Call:
## lm(formula = fitfull, data = training_set)
```

```
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.63505 -0.09575  0.01004  0.10980  0.56477
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -5.9295024  0.7839313  -7.564 8.00e-14 ***
## Fireplaces      0.0915975  0.0106971   8.563  < 2e-16 ***
## GarageYrBlt     0.0025002  0.0003516   7.112 2.01e-12 ***
## FullBath        0.0743928  0.0144377   5.153 3.02e-07 ***
## TotRmsAbvGrd    0.0110316  0.0061549   1.792  0.07334 .
## log(X1stFlrSF)  0.5067204  0.0325424  15.571  < 2e-16 ***
## log(LotArea)    0.1105127  0.0164516   6.717 2.91e-11 ***
## YearRemodAdd    0.0041037  0.0003627  11.315  < 2e-16 ***
## MSSubClass30   -0.1445345  0.0305530  -4.731 2.52e-06 ***
## MSSubClass40    0.1983086  0.1104754   1.795  0.07291 .
## MSSubClass45   -0.0045592  0.0617886  -0.074  0.94119
## MSSubClass50    0.0493773  0.0237116   2.082  0.03753 *
## MSSubClass60    0.2243949  0.0241688   9.284  < 2e-16 ***
## MSSubClass70    0.1881838  0.0343747   5.474 5.39e-08 ***
## MSSubClass75    0.1652147  0.0577385   2.861  0.00429 **
## MSSubClass80    0.0192625  0.0301931   0.638  0.52362
## MSSubClass85    0.0696186  0.0462468   1.505  0.13250
## MSSubClass90   -0.1946850  0.0329292  -5.912 4.45e-09 ***
## MSSubClass120   0.0898438  0.0278308   3.228  0.00128 **
## MSSubClass160   0.2149725  0.0383021   5.613 2.50e-08 ***
## MSSubClass180   0.0197698  0.0683050   0.289  0.77230
## MSSubClass190  -0.1129488  0.0428627  -2.635  0.00852 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1899 on 1146 degrees of freedom
## Multiple R-squared:  0.7765, Adjusted R-squared:  0.7724
## F-statistic: 189.6 on 21 and 1146 DF,  p-value: < 2.2e-16
```

No significant change in RSE or Adjusted R Square with removal of high-leverage outliers.

Building Predictions

Final MLR model

```
trainprime3<-trainprime
testprime3<-testprime
# Backward

fitfull <- lm(log(SalePrice) ~ Fireplaces + GarageYrBlt + FullBath + TotRmsAbvGrd + log(X1stFlrSF) + lo

prediction<-predict(fitfull, newdata=testprime3)

testprime3$logSalePrice<-prediction

testprime3$SalePrice<-exp(testprime3$logSalePrice)
```

```
houseprices3<-testprime3[c("Id", "SalePrice")]


write.csv(houseprices3,"testhouseprices_final.csv", row.names=TRUE)
```

Final Kaggle Score: 0.20089

Simple Linear Regression Model (YearBuilt)

```
# SLR


slr <- lm(log(SalePrice) ~ YearBuilt, data = trainprime)

predicted <- predict(slr, newdata = testprime)

testprime3$logSalePrice<-predicted

testprime3$SalePrice<-exp(testprime3$logSalePrice)

houseprices3<-testprime3[c("Id", "SalePrice")]


write.csv(houseprices3,"testhouseprices_SLR.csv", row.names=TRUE)
```

Explanatory variables: GrLivArea + FullBath (Custom)

```
testprime$FullBath<-as.numeric(testprime$FullBath)

custom <- lm(log(SalePrice) ~ log(GrLivArea) + FullBath, data = trainprime)

predicted <- predict(custom, newdata = testprime)

testprime3$logSalePrice<-predicted

testprime3$SalePrice<-exp(testprime3$logSalePrice)

houseprices3<-testprime3[c("Id", "SalePrice")]


write.csv(houseprices3,"testhouseprices_Custom.csv", row.names=TRUE)
```