Linear Regression Analysis: Regression Case Study

Neerja Doshi, Sri Santhosh Hari, Ker-Yu Ong, Nicha Ruchirawat

Part I: Explanatory Modelling

Task 0: Exploratory Data Analysis and Data Cleaning

```r
rawDF <- read.csv("/Users/booranium/usf/601_regression/project/housing.txt",
    stringsAsFactors = T)
# rawDF <-
# read.csv('/Users/santhoshhari/Documents/Coursework/LinearRegression/IowaHousing/Data/housing.txt',
# stringsAsFactors = T)
```

The Iowa housing dataset contains 1460 rows and 81 variables, a glimpse of which is as follows:

```r
str(rawDF)
```

```
## 'data.frame':    1460 obs. of  81 variables:
##  $ Id           : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ MSSubClass   : int  60 20 60 70 60 50 20 60 50 190 ...
##  $ MSZoning     : Factor w/ 5 levels "C (all)","FV",..: 4 4 4 4 4 4 4 4 5 4 ...
##  $ LotFrontage  : int  65 80 68 60 84 85 75 NA 51 50 ...
##  $ LotArea      : int  8450 9600 11250 9550 14260 14115 10084 10382 6120 7420 ...
##  $ Street       : Factor w/ 2 levels "Grvl","Pave": 2 2 2 2 2 2 2 2 2 2 ...
##  $ Alley        : Factor w/ 2 levels "Grvl","Pave": NA NA NA NA NA NA NA NA NA NA ...
##  $ LotShape     : Factor w/ 4 levels "IR1","IR2","IR3",..: 4 4 1 1 1 1 4 1 4 4 ...
##  $ LandContour  : Factor w/ 4 levels "Bnk","HLS","Low",..: 4 4 4 4 4 4 4 4 4 4 ...
##  $ Utilities    : Factor w/ 2 levels "AllPub","NoSeWa": 1 1 1 1 1 1 1 1 1 1 ...
##  $ LotConfig    : Factor w/ 5 levels "Corner","CulDSac",..: 5 3 5 1 3 5 5 1 5 1 ...
##  $ LandSlope    : Factor w/ 3 levels "Gtl","Mod","Sev": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Neighborhood : Factor w/ 25 levels "Blmngtn","Blueste",..: 6 25 6 7 14 12 21 17 18 4 ...
##  $ Condition1   : Factor w/ 9 levels "Artery","Feedr",..: 3 2 3 3 3 3 3 5 1 1 ...
##  $ Condition2   : Factor w/ 8 levels "Artery","Feedr",..: 3 3 3 3 3 3 3 3 3 1 ...
##  $ BldgType     : Factor w/ 5 levels "1Fam","2fmCon",..: 1 1 1 1 1 1 1 1 1 2 ...
##  $ HouseStyle   : Factor w/ 8 levels "1.5Fin","1.5Unf",..: 6 3 6 6 6 1 3 6 1 2 ...
##  $ OverallQual  : int  7 6 7 7 8 5 8 7 7 5 ...
##  $ OverallCond  : int  5 8 5 5 5 5 5 6 5 6 ...
##  $ YearBuilt    : int  2003 1976 2001 1915 2000 1993 2004 1973 1931 1939 ...
##  $ YearRemodAdd : int  2003 1976 2002 1970 2000 1995 2005 1973 1950 1950 ...
##  $ RoofStyle    : Factor w/ 6 levels "Flat","Gable",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ RoofMatl     : Factor w/ 8 levels "ClyTile","CompShg",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ Exterior1st  : Factor w/ 15 levels "AsbShng","AsphShn",..: 13 9 13 14 13 13 13 7 4 9 ...
##  $ Exterior2nd  : Factor w/ 16 levels "AsbShng","AsphShn",..: 14 9 14 16 14 14 14 7 16 9 ...
##  $ MasVnrType   : Factor w/ 4 levels "BrkCmn","BrkFace",..: 2 3 2 3 2 3 4 4 3 3 ...
##  $ MasVnrArea   : int  196 0 162 0 350 0 186 240 0 0 ...
##  $ ExterQual    : Factor w/ 4 levels "Ex","Fa","Gd",..: 3 4 3 4 3 4 3 4 4 4 ...
##  $ ExterCond    : Factor w/ 5 levels "Ex","Fa","Gd",..: 5 5 5 5 5 5 5 5 5 5 ...
##  $ Foundation   : Factor w/ 6 levels "BrkTil","CBlock",..: 3 2 3 1 3 6 3 2 1 1 ...
##  $ BsmtQual     : Factor w/ 4 levels "Ex","Fa","Gd",..: 3 3 3 4 3 3 1 3 4 4 ...
##  $ BsmtCond     : Factor w/ 4 levels "Fa","Gd","Po",..: 4 4 4 2 4 4 4 4 4 4 ...
##  $ BsmtExposure : Factor w/ 4 levels "Av","Gd","Mn",..: 4 2 3 4 1 4 1 3 4 4 ...
```

```
##  $ BsmtFinType1 : Factor w/ 6 levels "ALQ","BLQ","GLQ",..: 3 1 3 1 3 3 3 1 6 3 ...
##  $ BsmtFinSF1   : int  706 978 486 216 655 732 1369 859 0 851 ...
##  $ BsmtFinType2 : Factor w/ 6 levels "ALQ","BLQ","GLQ",..: 6 6 6 6 6 6 6 2 6 6 ...
##  $ BsmtFinSF2   : int  0 0 0 0 0 0 0 32 0 0 ...
##  $ BsmtUnfSF    : int  150 284 434 540 490 64 317 216 952 140 ...
##  $ TotalBsmtSF  : int  856 1262 920 756 1145 796 1686 1107 952 991 ...
##  $ Heating      : Factor w/ 6 levels "Floor","GasA",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ HeatingQC    : Factor w/ 5 levels "Ex","Fa","Gd",..: 1 1 1 3 1 1 1 1 3 1 ...
##  $ CentralAir   : Factor w/ 2 levels "N","Y": 2 2 2 2 2 2 2 2 2 2 ...
##  $ Electrical   : Factor w/ 5 levels "FuseA","FuseF",..: 5 5 5 5 5 5 5 5 2 5 ...
##  $ X1stFlrSF    : int  856 1262 920 961 1145 796 1694 1107 1022 1077 ...
##  $ X2ndFlrSF    : int  854 0 866 756 1053 566 0 983 752 0 ...
##  $ LowQualFinSF : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ GrLivArea    : int  1710 1262 1786 1717 2198 1362 1694 2090 1774 1077 ...
##  $ BsmtFullBath : int  1 0 1 1 1 1 1 1 0 1 ...
##  $ BsmtHalfBath : int  0 1 0 0 0 0 0 0 0 0 ...
##  $ FullBath     : int  2 2 2 1 2 1 2 2 2 1 ...
##  $ HalfBath     : int  1 0 1 0 1 1 0 1 0 0 ...
##  $ BedroomAbvGr : int  3 3 3 3 4 1 3 3 2 2 ...
##  $ KitchenAbvGr : int  1 1 1 1 1 1 1 1 2 2 ...
##  $ KitchenQual  : Factor w/ 4 levels "Ex","Fa","Gd",..: 3 4 3 3 3 4 3 4 4 4 ...
##  $ TotRmsAbvGrd : int  8 6 6 7 9 5 7 7 8 5 ...
##  $ Functional   : Factor w/ 7 levels "Maj1","Maj2",..: 7 7 7 7 7 7 7 7 3 7 ...
##  $ Fireplaces   : int  0 1 1 1 1 0 1 2 2 2 ...
##  $ FireplaceQu  : Factor w/ 5 levels "Ex","Fa","Gd",..: NA 5 5 3 5 NA 3 5 5 5 ...
##  $ GarageType   : Factor w/ 6 levels "2Types","Attchd",..: 2 2 2 6 2 2 2 2 6 2 ...
##  $ GarageYrBlt  : int  2003 1976 2001 1998 2000 1993 2004 1973 1931 1939 ...
##  $ GarageFinish : Factor w/ 3 levels "Fin","RFn","Unf": 2 2 2 3 2 3 2 2 3 2 ...
##  $ GarageCars   : int  2 2 2 3 3 2 2 2 2 1 ...
##  $ GarageArea   : int  548 460 608 642 836 480 636 484 468 205 ...
##  $ GarageQual   : Factor w/ 5 levels "Ex","Fa","Gd",..: 5 5 5 5 5 5 5 5 2 3 ...
##  $ GarageCond   : Factor w/ 5 levels "Ex","Fa","Gd",..: 5 5 5 5 5 5 5 5 5 5 ...
##  $ PavedDrive   : Factor w/ 3 levels "N","P","Y": 3 3 3 3 3 3 3 3 3 3 ...
##  $ WoodDeckSF   : int  0 298 0 0 192 40 255 235 90 0 ...
##  $ OpenPorchSF  : int  61 0 42 35 84 30 57 204 0 4 ...
##  $ EnclosedPorch: int  0 0 0 272 0 0 0 228 205 0 ...
##  $ X3SsnPorch   : int  0 0 0 0 0 320 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       : Factor w/ 3 levels "Ex","Fa","Gd": NA NA NA NA NA NA NA NA NA NA ...
##  $ Fence        : Factor w/ 4 levels "GdPrv","GdWo",..: NA NA NA NA NA 3 NA NA NA NA ...
##  $ MiscFeature  : Factor w/ 4 levels "Gar2","Othr",..: NA NA NA NA NA 3 NA 3 NA NA ...
##  $ MiscVal      : int  0 0 0 0 0 700 0 350 0 0 ...
##  $ MoSold       : int  2 5 9 2 12 10 8 11 4 1 ...
##  $ YrSold       : int  2008 2007 2008 2006 2008 2009 2007 2009 2008 2008 ...
##  $ SaleType     : Factor w/ 9 levels "COD","Con","ConLD",..: 9 9 9 9 9 9 9 9 9 9 ...
##  $ SaleCondition: Factor w/ 6 levels "Abnorml","AdjLand",..: 5 5 5 1 5 5 5 5 1 5 ...
##  $ SalePrice    : int  208500 181500 223500 140000 250000 143000 307000 200000 129900 118000 ...
```

At first glance, we see that most of the variables are categorical - both numeric and character types - and only a handful are continuous. The response variable for our analysis is `SalePrice`, and the remaining 79 variables (excluding the record `ID` column) are considered potential predictor variables. Checking the data dictionary, we found the following distribution for the predictor variables:

- 49 categorical

- 19 are continuous, e.g. area, price
- 11 are discrete, e.g. count, year

There are 0 duplicate rows in the dataset.

Handling NA Values

Below, we compute that number and percentage of NAs per variable in the dataset having at least 1 NA.

```
NA_columns <- colnames(rawDF)[unique(which(is.na(rawDF), arr.ind = T)[,
    2])]

NA_count <- rawDF %>% select(NA_columns) %>% summarise_all(funs(sum(is.na(.)))) %>%
    gather(key = "Variable", value = "num_na", everything()) %>% arrange(desc(num_na))

NA_count %<>% mutate(perc_na = paste(round(num_na/nrow(rawDF), 4) *
    100, "%"))
colnames(NA_count) <- c("**Variable**", "**Number of NA**", "**Percentage of NA**")
row.names(NA_count) <- NULL
knitr::kable(NA_count, caption = "\\label{tab:NACount} Variable NA Count and Percentage",
    format.args = list(big.mark = ","))
```

Table 1: Variable NA Count and Percentage

Variable	**Number of NA**	**Percentage of NA**
PoolQC	1,453	99.52 %
MiscFeature	1,406	96.3 %
Alley	1,369	93.77 %
Fence	1,179	80.75 %
FireplaceQu	690	47.26 %
LotFrontage	259	17.74 %
GarageType	81	5.55 %
GarageYrBlt	81	5.55 %
GarageFinish	81	5.55 %
GarageQual	81	5.55 %
GarageCond	81	5.55 %
BsmtExposure	38	2.6 %
BsmtFinType2	38	2.6 %
BsmtQual	37	2.53 %
BsmtCond	37	2.53 %
BsmtFinType1	37	2.53 %
MasVnrType	8	0.55 %
MasVnrArea	8	0.55 %
Electrical	1	0.07 %

The data dictionary tells us that for most of the fields in Table 1, NA is actually meaningful, indicating non-applicability or a lack of the feature rather than missing data. After checking the data dictionary for the meaning of each field, we imputed - for every categorical variable for which NA was meaningful - NAs with 0s.

```
# Create a copy of rawDF to be our working data frame
housingDF <- rawDF

# Update NAs with 0s for applicable fields
levels(housingDF$PoolQC) <- c("0", levels(housingDF$PoolQC))
```

```r
housingDF$PoolQC[is.na(housingDF$PoolQC)] <- "0"
levels(housingDF$MiscFeature) <- c("0", levels(housingDF$MiscFeature))
housingDF$MiscFeature[is.na(housingDF$MiscFeature)] <- "0"
levels(housingDF$Alley) <- c("0", levels(housingDF$Alley))
housingDF$Alley[is.na(housingDF$Alley)] <- "0"
levels(housingDF$Fence) <- c("0", levels(housingDF$Fence))
housingDF$Fence[is.na(housingDF$Fence)] <- "0"
levels(housingDF$FireplaceQu) <- c("0", levels(housingDF$FireplaceQu))
housingDF$FireplaceQu[is.na(housingDF$FireplaceQu)] <- "0"
levels(housingDF$GarageType) <- c("0", levels(housingDF$GarageType))
housingDF$GarageType[is.na(housingDF$GarageType)] <- "0"
levels(housingDF$GarageFinish) <- c("0", levels(housingDF$GarageFinish))
housingDF$GarageFinish[is.na(housingDF$GarageFinish)] <- "0"
levels(housingDF$GarageQual) <- c("0", levels(housingDF$GarageQual))
housingDF$GarageQual[is.na(housingDF$GarageQual)] <- "0"
levels(housingDF$GarageCond) <- c("0", levels(housingDF$GarageCond))
housingDF$GarageCond[is.na(housingDF$GarageCond)] <- "0"
levels(housingDF$BsmtExposure) <- c("0", levels(housingDF$BsmtExposure))
housingDF$BsmtExposure[is.na(housingDF$BsmtExposure)] <- "0"
levels(housingDF$BsmtFinType2) <- c("0", levels(housingDF$BsmtFinType2))
housingDF$BsmtFinType2[is.na(housingDF$BsmtFinType2)] <- "0"
levels(housingDF$BsmtQual) <- c("0", levels(housingDF$BsmtQual))
housingDF$BsmtQual[is.na(housingDF$BsmtQual)] <- "0"
levels(housingDF$BsmtCond) <- c("0", levels(housingDF$BsmtCond))
housingDF$BsmtCond[is.na(housingDF$BsmtCond)] <- "0"
levels(housingDF$BsmtFinType1) <- c("0", levels(housingDF$BsmtFinType1))
housingDF$BsmtFinType1[is.na(housingDF$BsmtFinType1)] <- "0"
```

We then re-check the count and percentage of `NA`s per variable left in the dataset.

```r
NA_columns <- colnames(housingDF)[unique(which(is.na(housingDF), arr.ind = T)[,
    2])]
NA_count <- housingDF %>% select(NA_columns) %>% summarise_all(funs(sum(is.na(.)))) %>%
    gather(key = "Variable", value = "num_na", everything()) %>% arrange(desc(num_na))

NA_count %<>% mutate(perc_na = paste(round(num_na/nrow(housingDF),
    4) * 100, "%"))
colnames(NA_count) <- c("**Variable**", "**Number of NA**", "**Percentage of NA**")
row.names(NA_count) <- NULL
knitr::kable(NA_count, caption = "\\label{tab:NACount1} Variable NA Count and Percentage(after replacing
    format.args = list(big.mark = ","))
```

Table 2: Variable NA Count and Percentage(after replacing NAs with 0s, where appropriate)

Variable	**Number of NA**	**Percentage of NA**
LotFrontage	259	17.74 %
GarageYrBlt	81	5.55 %
MasVnrType	8	0.55 %
MasVnrArea	8	0.55 %
Electrical	1	0.07 %

Table 2 shows the list of remaining variables where `NA` indicates missing data. We impute `NA`s in these variables with

- mean of the data, for continuous variables (LotFrontage)
- median of the data, for discrete variables (GarageYrBlt)
- mode of the data, for categorical variables (MasVnrType, Electrical)

```
# Function to get mode of data
getmode <- function(v) {
    uniqv <- unique(v)
    uniqv[which.max(tabulate(match(v, uniqv)))]
}

# Impute NAs
housingDF$LotFrontage[is.na(housingDF$LotFrontage)] <- mean(housingDF$LotFrontage,
    na.rm = T)
housingDF$GarageYrBlt[is.na(housingDF$GarageYrBlt)] <- median(housingDF$GarageYrBlt,
    na.rm = T)
housingDF$MasVnrType[is.na(housingDF$MasVnrType)] <- getmode(housingDF$MasVnrType)
housingDF$MasVnrArea[is.na(housingDF$MasVnrArea)] <- 0
housingDF$Electrical[is.na(housingDF$Electrical)] <- getmode(housingDF$Electrical)

housingDF$MSSubClass <- factor(housingDF$MSSubClass)
```

Since Masonry veneer area (`MasVnrArea`) is directly related to `MasVnrType`, we impute for area based on the mode of `MasVnrType`, which is None. Our cleaned dataset is named `housingDF`.

Exploratory Data Visualization

With our clean dataset, we perform exploratory data visualization of the distribution of key measures such as volume and sale price of houses by what we hypothesize to be key predictor variables.

To begin with, we check the distribution of sale prices using a histogram and box-plot.

```
# hist(housingDF$SalePrice, main = 'Histogram of Sale Price')
# boxplot(housingDF$SalePricem, main = 'Boxplot of Sale Price')
summary(housingDF$SalePrice)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   34900  129975  163000  180921  214000  755000
```

Intuition suggests the neighborhood is a key determining factor in a house's sale price, hence below, we plot the distribution of sale price by neighborhood.

```
housingDF %>% select(Neighborhood, SalePrice) %>% ggplot(aes(factor(Neighborhood),
    SalePrice)) + geom_boxplot() + theme(axis.text.x = element_text(angle = 90,
    hjust = 1)) + xlab("Neighborhoods")
```

From Figure 1, we can observe that Brookside and Meadow Vista have the lowest median house price while Northridge and Northridge Height have the heighest median house price as well as several outliers.

We then distribution of houses by a number of key features we hypothesize to be important in determining housing price: the property's zoning class (`MSZoning`), type of road access to the property (`Street`), type of alley access to the property (`Alley`), and type of utilies available (`Utilities`)

```
plotHist <- function(data_in, i) {
    data <- data.frame(x = data_in[[i]])
    p <- ggplot(data = data, aes(x = factor(x))) + stat_count() +
        xlab(colnames(data_in)[i]) + theme_light() + theme(axis.text.x = element_text(angle = 90,
```

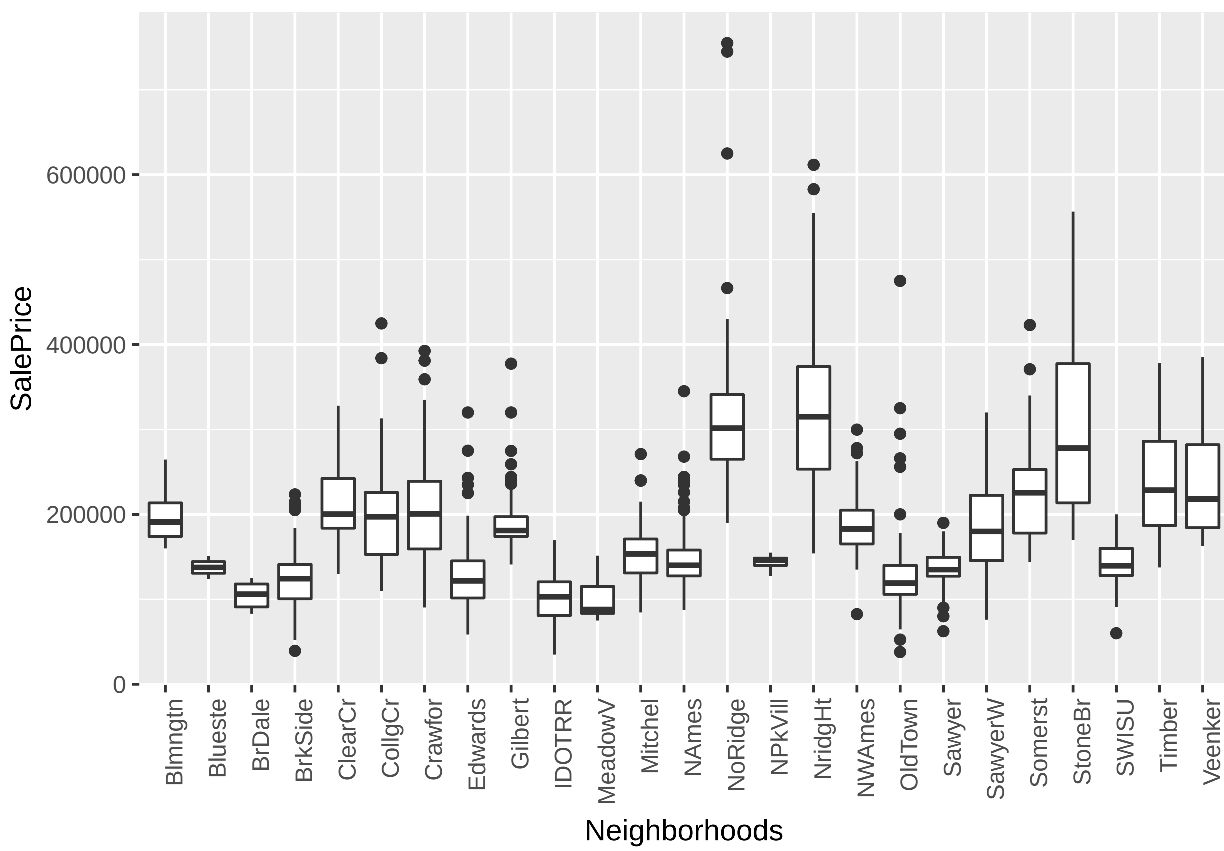


Figure 1: SalePrice distribution per neighborhood

```r
        hjust = 1))
    return(p)
}

doPlots <- function(data_in, fun, ii, ncol = 3) {
    pp <- list()
    for (i in ii) {
        p <- fun(data_in = data_in, i = i)
        pp <- c(pp, list(p))
    }
    do.call("grid.arrange", c(pp, ncol = ncol))
}

plotDen <- function(data_in, i) {
    data <- data.frame(x = data_in[[i]], SalePrice = data_in$SalePrice)
    p <- ggplot(data = data) + geom_line(aes(x = x), stat = "density",
        size = 1, alpha = 1) + xlab(paste0((colnames(data_in)[i]),
        "\n", "Skewness: ", round(skewness(data_in[[i]], na.rm = TRUE),
            2))) + theme_light()
    return(p)
}

plotCorr <- function(data_in, i) {
    data <- data.frame(x = data_in[[i]], SalePrice = data_in$SalePrice)
    p <- ggplot(data, aes(x = x, y = SalePrice)) + geom_point(na.rm = TRUE) +
        geom_smooth(method = lm) + xlab(paste0(colnames(data_in)[i],
        "\n", "R-Squared: ", round(cor(data_in[[i]], data$SalePrice,
            use = "complete.obs"), 2))) + theme_light()
    return(suppressWarnings(p))
}
```

```r
doPlots(housingDF, fun = plotHist, ii = c(3, 6, 7, 10), ncol = 2)
```

Figure 2 suggests that most of the houses are located in `Medium/Low Density` residential areas. We can also observe that most of the houses have paved road access, do not have alleys and have all public utilities(E,G,W,& S). From Figure **??**{fig:hist2}, we can notice that most of the properties are regular or slightly irregular in share, built on level surfaces with gentle slope.

We also plot the distribution of houses against a number of features related to the physical geography of the property:

```r
doPlots(housingDF, fun = plotHist, ii = c(8, 9, 11, 12), ncol = 2)
```

We see that ...

```r
housingDF %>% select(LandSlope, Neighborhood) %>% arrange(Neighborhood) %>%
    group_by(Neighborhood, LandSlope) %>% summarize(Count = n()) %>%
    ggplot(aes(Neighborhood, Count)) + geom_bar(aes(fill = LandSlope),
    position = "dodge", stat = "identity") + theme(axis.text.x = element_text(angle = 90,
    hjust = 1))
```

From Figure **??**{fig:hist3}, we can see that houses with severe slope are located only in Clear Creek and Timberland while more than 10 neighborhoods have properties with moderate slope.

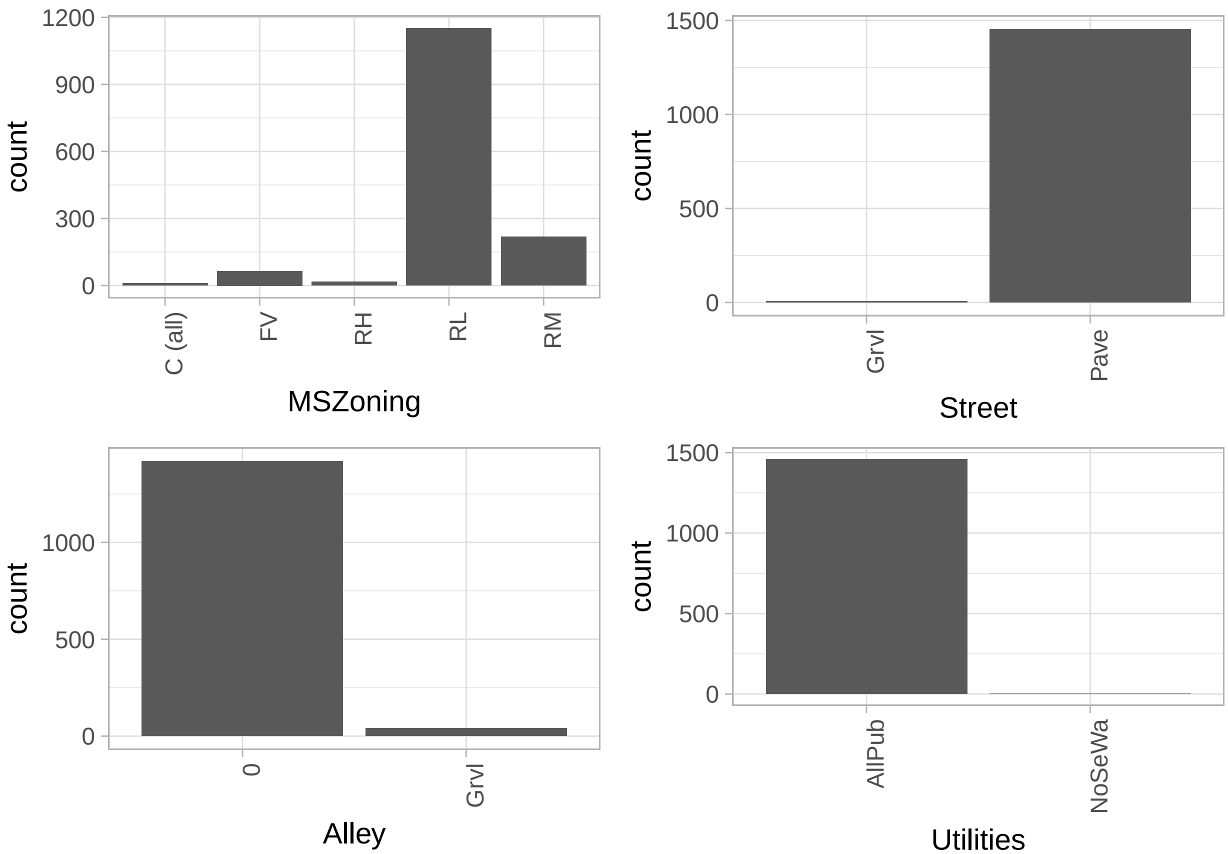


Figure 2: Locality, access, utility features distribution

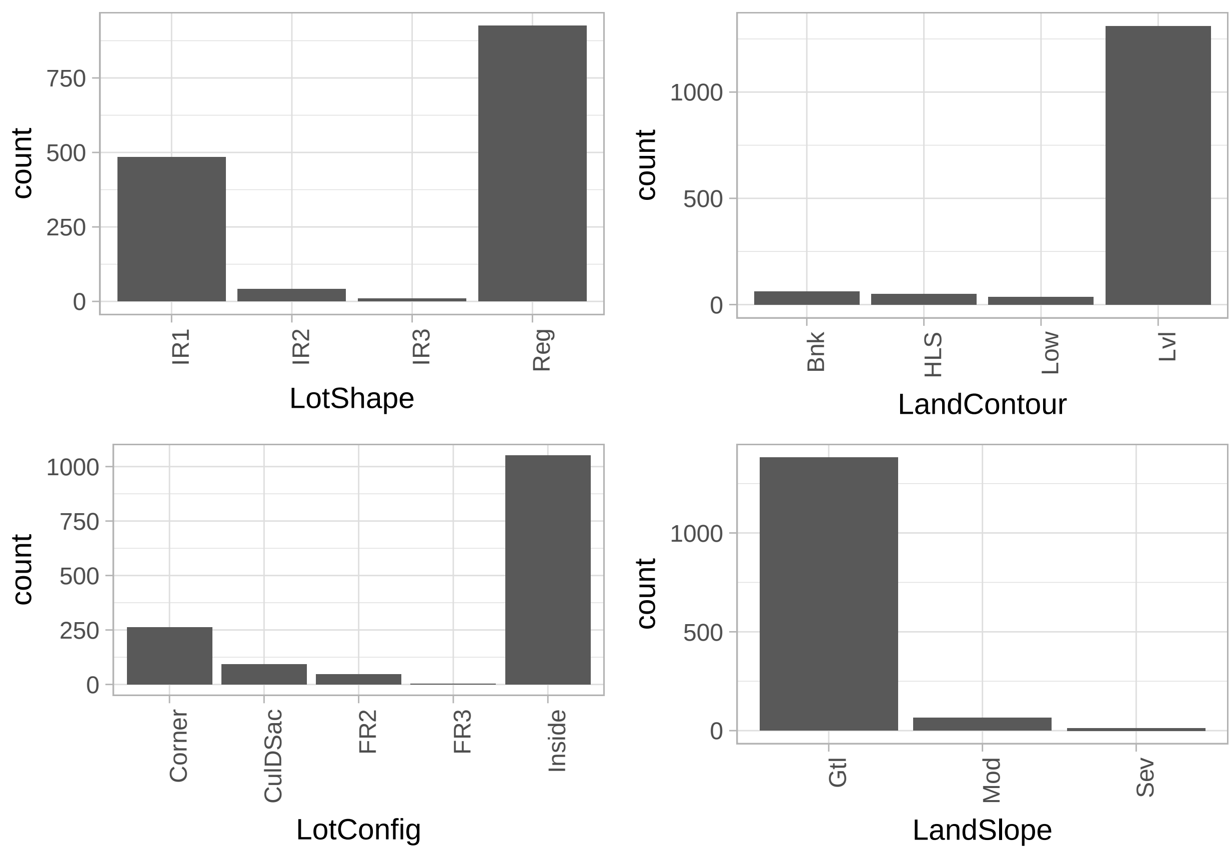


Figure 3: Lot/Land feature distribution

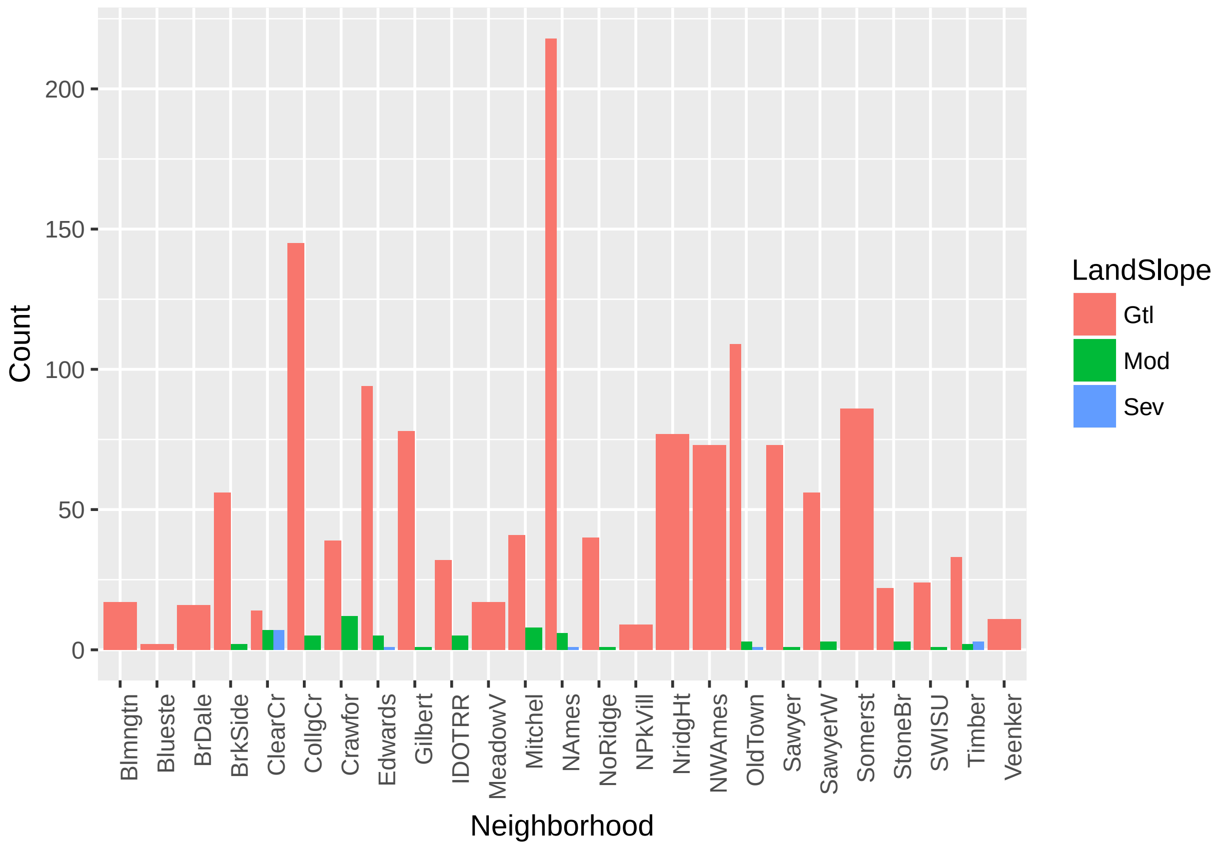


Figure 4: Neighborhood level slope distribution

Task 1. Building the Explanatory Model

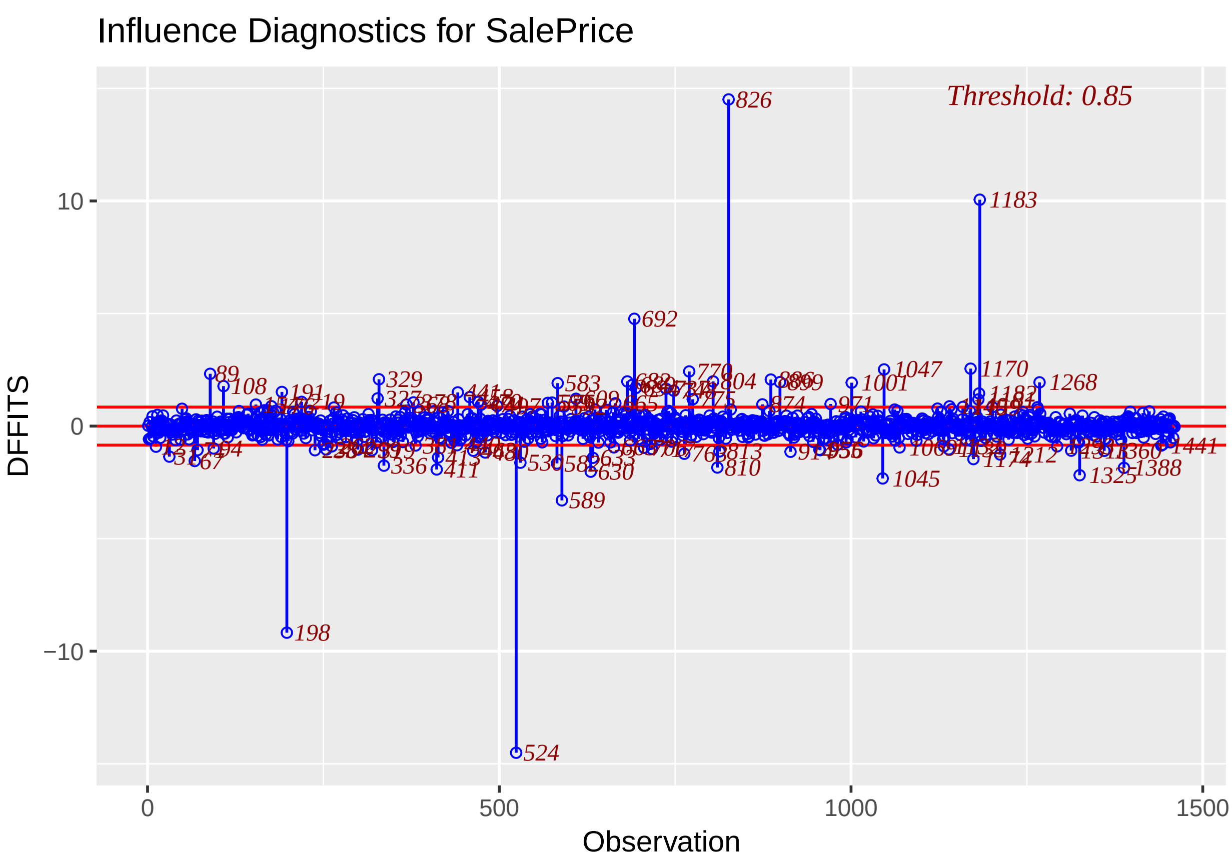
Testing for Influential Points

Having dealt with the `NA`s in our dataset, we use the `model.matrix()` function from the `glmnet` package to convert each categorical variable into an appropriate set of binary indicators: for a categorical variable that takes k levels, `model.matrix()` produces k-1 binary indicators. We then reappend our response vector `SalePrice` to the resulting wide design matrix `designDF` to create `workingDF`, which includes both the converted predictors and response variables.

```
designDF <- model.matrix(SalePrice ~ ., data = housingDF)[, -1]
designDF <- as.data.frame(designDF)
workingDF <- cbind(designDF, SalePrice = housingDF$SalePrice)
```

In looking for influential points, we leverage the `OLSRR` package to test observations for influence according to the DFFITS diagnostic. We do this by first fitting a saturated model on `workingDF` and then calling `ols_dffits_plot()` on it.

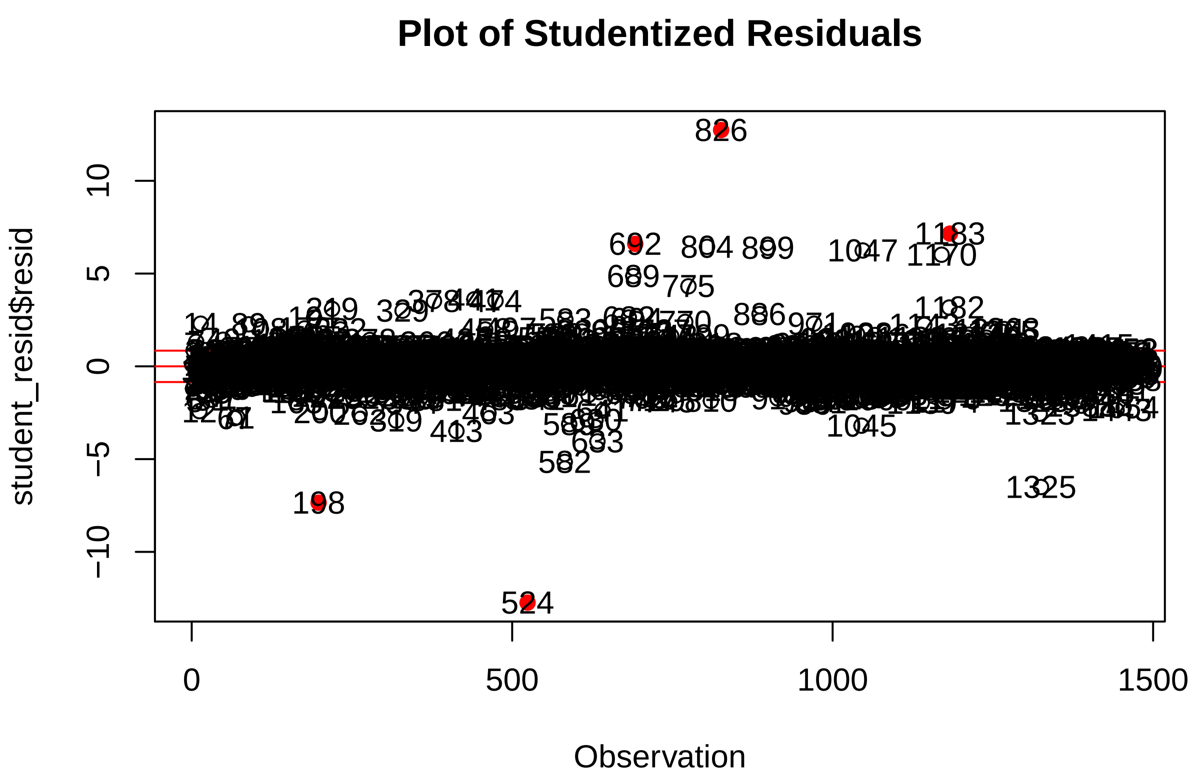
```
ols_model <- lm(SalePrice ~ ., data = workingDF)
ols_dffits_plot(ols_model)
```




```
# identify threshold t for points of influence
n = nrow(workingDF)
p = ncol(workingDF - 1)  # remove response var
t = 2 * sqrt(p/n)
```

Note that according to the criterion of threshold t = 2*sqrt(n/p) = 0.85, the DFFITS plot shows a large number of influential observations. We look specifically at 6 points greater than threshold = abs(5): 198, 524, 692, 826 and 1183. Below we plot the standardized and studentized residuals to check these observations for being outliers and/or points of leverage respectively.

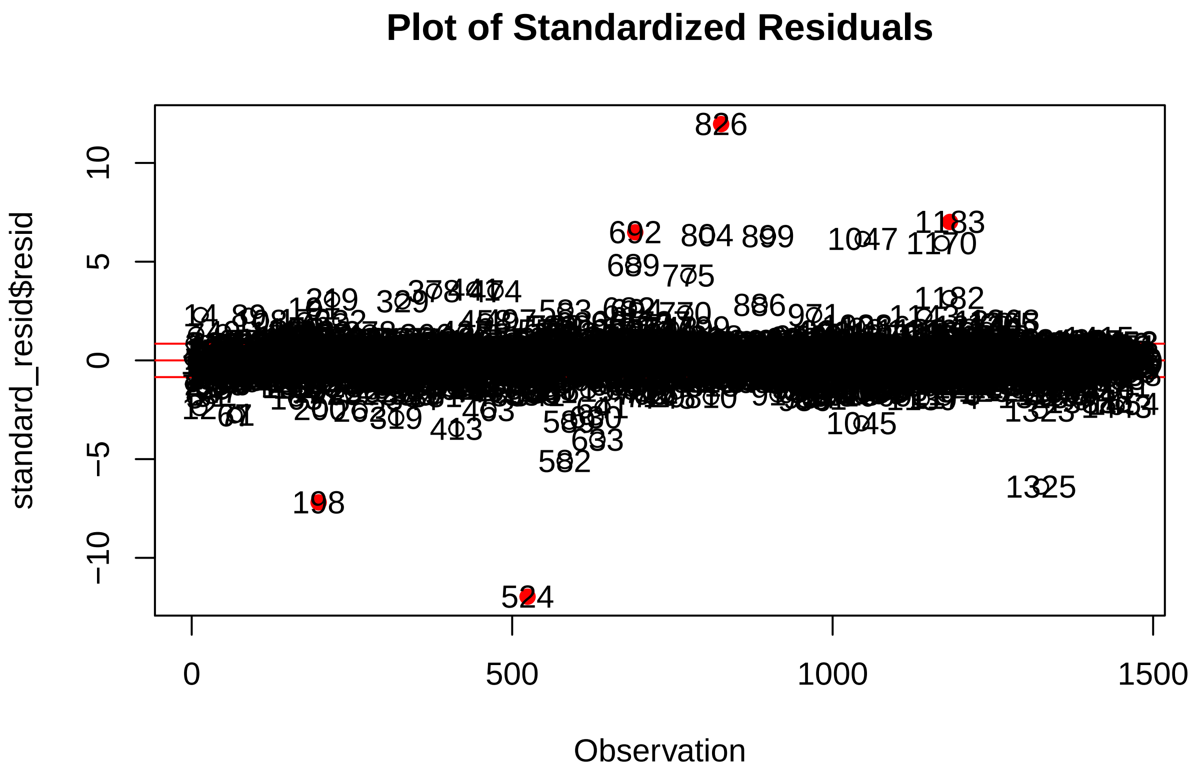
```r
# Creating from scratch because OLSRR ols_srsd_plot() function is
# not working.

# Create df of studentized residuals
student_resid <- as.data.frame(rstudent(ols_model))
student_resid <- setDT(student_resid, keep.rownames = TRUE)[]
colnames(student_resid) <- c("ix", "resid")

# Plot
plot(student_resid$ix, student_resid$resid, col = ifelse(workingDF$Id ==
    198 | workingDF$Id == 524 | workingDF$Id == 692 | workingDF$Id ==
    826 | workingDF$Id == 1183, "red", "black"), pch = ifelse(workingDF$Id ==
    198 | workingDF$Id == 524 | workingDF$Id == 692 | workingDF$Id ==
    826 | workingDF$Id == 1183, 19, 1), main = "Plot of Studentized Residuals",
    xlab = "Observation")
abline(h = t, col = "red")
abline(h = 0, col = "red")
abline(h = -t, col = "red")
text(student_resid$ix, student_resid$resid, labels = student_resid$ix)
```



Our plot of studentized resdiuals indicates that observations all 5 points are leverage points.

```r
# Create df of standardized residuals
standard_resid <- as.data.frame(rstandard(ols_model))
standard_resid <- setDT(standard_resid, keep.rownames = TRUE)[]
colnames(standard_resid) <- c("ix", "resid")

# Plot
plot(standard_resid$ix, standard_resid$resid, col = ifelse(workingDF$Id ==
```

```
    198 | workingDF$Id == 524 | workingDF$Id == 692 | workingDF$Id ==
    826 | workingDF$Id == 1183, "red", "black"), pch = ifelse(workingDF$Id ==
    198 | workingDF$Id == 524 | workingDF$Id == 692 | workingDF$Id ==
    826 | workingDF$Id == 1183, 19, 1), main = "Plot of Standardized Residuals",
    xlab = "Observation")
abline(h = t, col = "red")
abline(h = 0, col = "red")
abline(h = -t, col = "red")
text(standard_resid$ix, standard_resid$resid, labels = standard_resid$ix)
```



Our plots of studentized and standardized resdiuals indicate that all 5 observations are both points of leverage and outliers. We remove them from our dataset and recreate the saturated OLS model below:

```
# remove influential points
workingDF <- filter(workingDF, !Id %in% c(198, 524, 692, 826, 1183))
# recreate model
ols_model <- lm(SalePrice ~ ., data = workingDF)
```

For the purposes of variable selection, we refer to the saturated OLS model created above and perform stepwise model selection according to both AIC and BIC criterions.

```
# Code chunk not run; load saved model for expediency

model_aic <- step(ols_model, direction = "backward", trace = F)
model_bic <- step(ols_model, k = log(nrow(workingDF)), direction = "backward",
    trace = F)

## save the models
save(model_aic, file = "/Users/booranium/usf/601_regression/project/IowaHousing/AIC_model.rda")
save(model_bic, file = "/Users/booranium/usf/601_regression/project/IowaHousing/BIC_model.rda")
```

Per the AIC criterion, the following are the predictor variables signficant at the alpha = 0.05 level.

```
load("/Users/booranium/usf/601_regression/project/IowaHousing/AIC_model.rda")  # model loaded as 'model_

# Find coefficients significant at the alpha = 0.05 level
bool_aic <- summary(model_aic)$coeff[-1, 4] < 0.05
sig_var_aic <- names(bool_aic)[bool_aic == TRUE]
```

Per the BIC criterion, the following are the predictor variables signficant at the alpha = 0.05 level.

```
load("/Users/booranium/usf/601_regression/project/IowaHousing/BIC_model.rda")  # model loaded as 'model_

# find coefficients significant at the alpha = 0.05 level
bool_bic <- summary(model_aic)$coeff[-1, 4] < 0.05
sig_var_bic <- names(bool_bic)[bool_bic == TRUE]
```

Note that both criterions select the same set of variables.

```
setdiff(sig_var_aic, sig_var_bic)
```

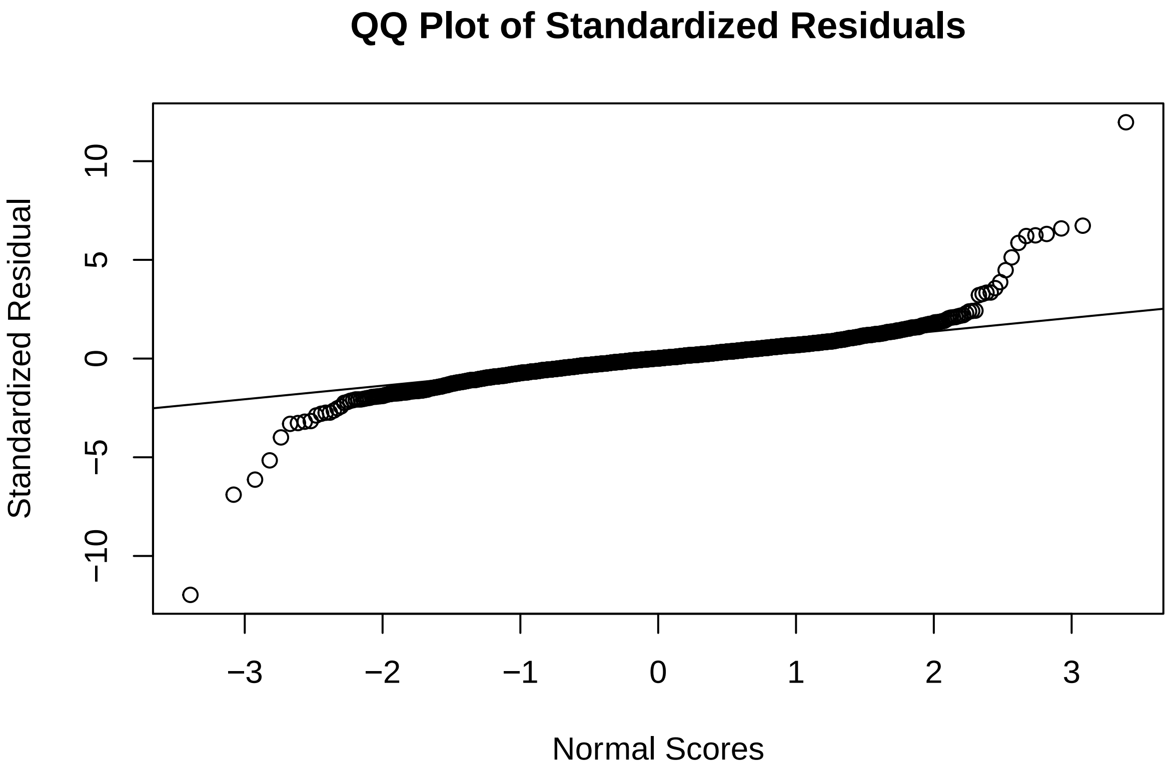
```
## character(0)
```

We can now perform OLS regression with our subset of significant variables. The model summary is as follows:

```
model_sig_formula <- as.formula(paste("SalePrice ~ ", paste(sig_var_bic,
    collapse = "+")))
model_sig <- lm(formula = model_sig_formula, data = workingDF)
model_sig_rsq <- summary(model_sig)$r.squared  # r-squared value
```

At first glance, the multiple R-squared value of 0.9252048 indicates that 91.19% of the variability in SalePrice around its mean is explained by the mode, i.e. by the predictor variables that have been included. This suggests a high-performing explanatory model. Before welcoming this conclusion, we validate the linearity and normality assumptions of our model by checking our residuals as follows:

```
# Residual plots

res = resid(model_sig)  # residuals
stdres = rstandard(model_sig)  # standardized residuals

# QQ Plot of Residuals - for normality
qqnorm(stdres, main = "QQ Plot of Standardized Residuals", xlab = "Normal Scores",
    ylab = "Standardized Residual")
qqline(stdres)
```




```
# Shapiro-Wilks test for normality
shapiro.test(res)
```

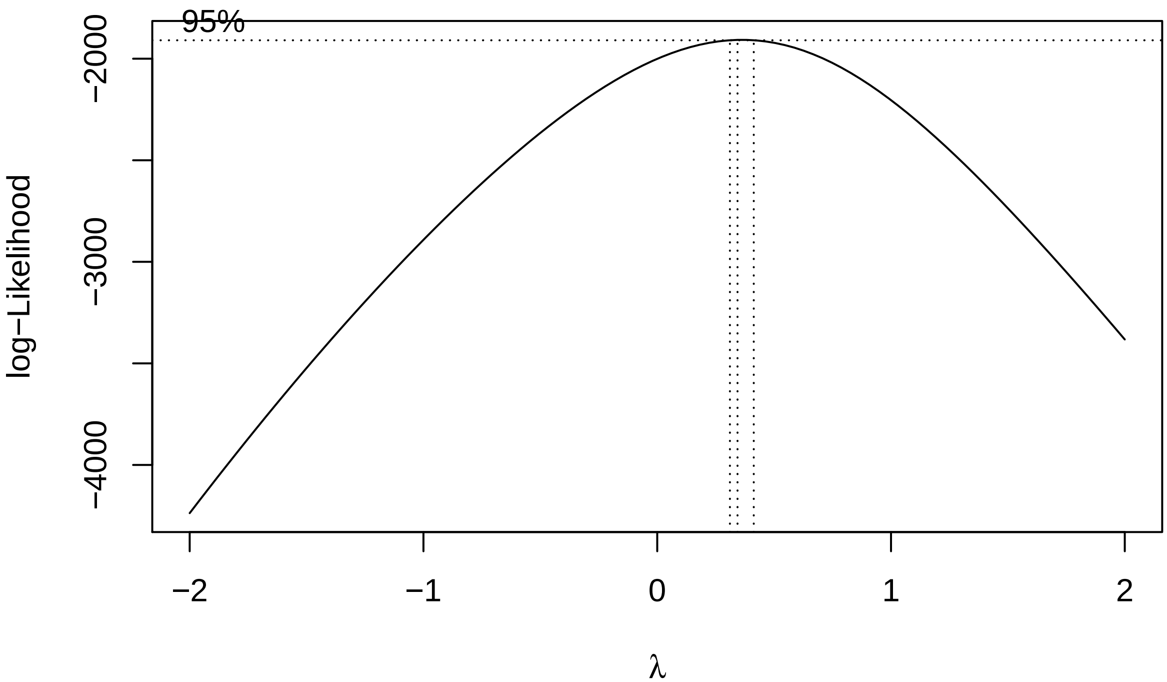
```
##
##  Shapiro-Wilk normality test
##
## data:  res
## W = 0.86375, p-value < 0.00000000000000022
```

```
# Plot of residuals vs. fitted values - for linearity
# plot(workingDF$SalePrice, res, main = 'Plot of Residuals vs.
# Sale Price', xlab = 'Sale Price', ylab = 'Residual')
# abline(h=c(0,0), col = 'red') abline(v=350000, col = 'blue')
```

The plot of residuals against fitted values shows that for the most part, the residuals are evenly distributed across the y = 0 line. However, we see that as Sale Price increases, the residuals start to deviate homoscedasticity. More specifically, we see this deviation happen at approximately Sale Price = $350K, which our earlier summary showed to be between the variable's 3rd quartile and maximum. This suggets that for the last quartile of high-priced houses, the fitted regression model is not as adequate as it is for the reset of the population. The normal probability (QQ) plot corroborates this finding: it shows deviance from linearity at both tail ends of the residual rang, which suggests a heavy tailed distribution. This occurs at both ends of the distribution, i.e. both extremely low-priced houses and extremely high-priced houses are pulling the distribution away from normality. Indeed, the formal Shapiro-Wilks test for normality produces a p-value of ~0, which leads us to reject the null hypothesis, at the alpha = 0.05 level, that the residuals are normally distributed.

As a remedial measure, we consider performing a transformation on Sale Price. We use the `boxcox()` function to determine the transformation under which the maximum likelihood [of ?] is attained.

```r
boxcox(model_sig)
```

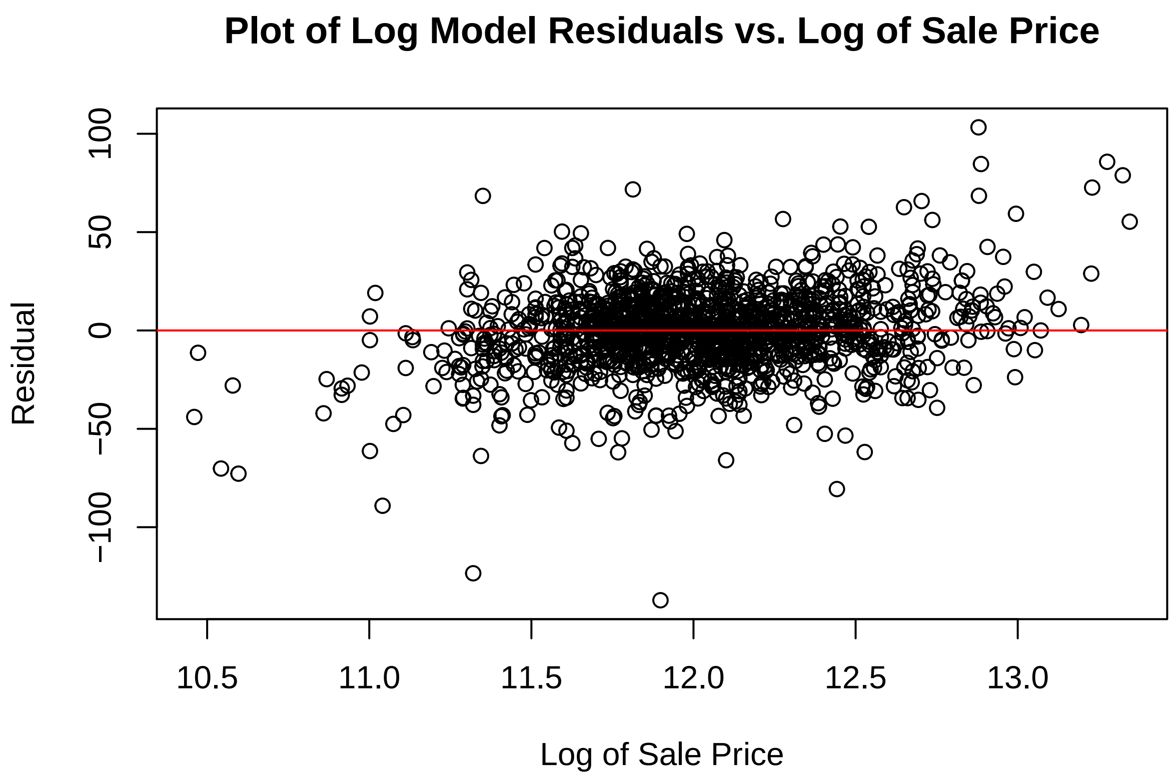


We see that lambda = 1 is captured in the 95% CI of lambdas, which means we are not bound to perform a transformation. However, since the likelihood function appears to take its maximum at lambda = ~ 0.5, we apply a square root transformation to Sale Price, refit the model, and revalidate our model assumptions with residual plots as above.

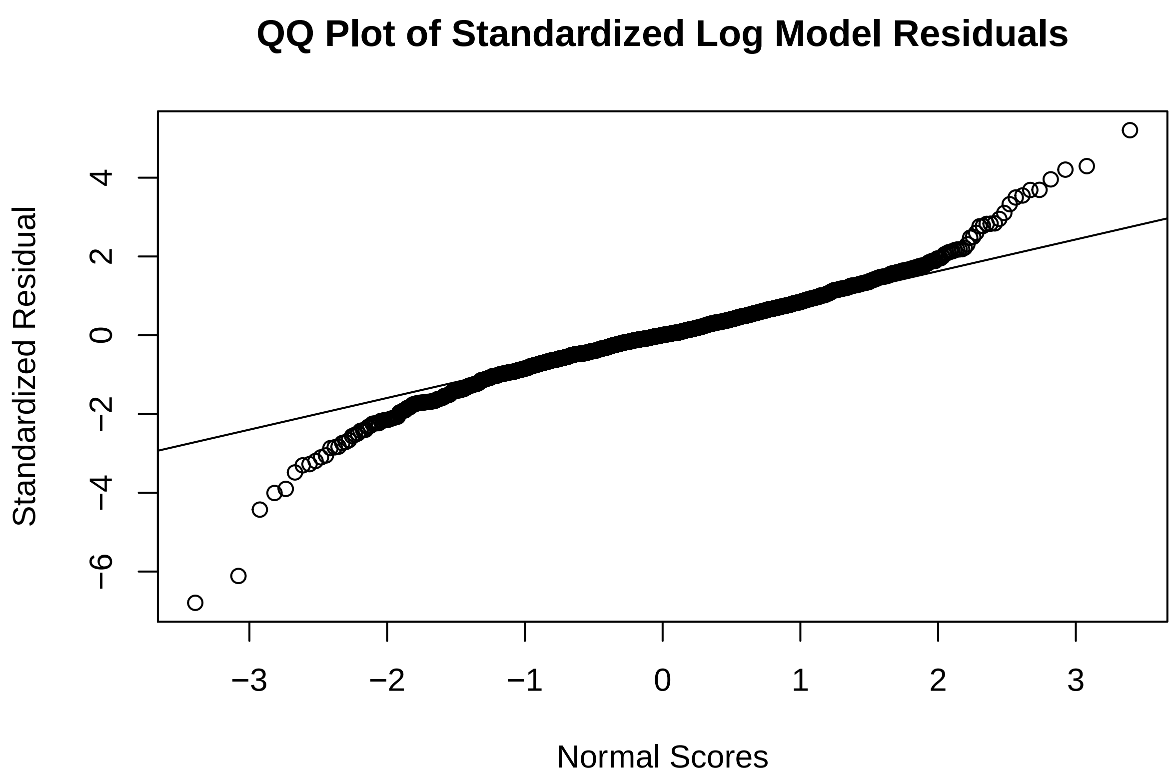
```r
# Log Transformation
log_model_formula <- as.formula(paste("sqrt(SalePrice) ~ ", paste(sig_var_bic,
    collapse = "+")))
log_model <- lm(formula = log_model_formula, data = workingDF)
log_model_rsq <- summary(log_model)$r.squared  # r-squared value

res_log = resid(log_model)  # residuals
stdres_log = rstandard(log_model)  # standardized residuals

shapiro.test(res)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  res
## W = 0.86375, p-value < 0.00000000000000022
```

```r
# Plot of Residuals from Log Model vs. Fitted Values
plot(log(workingDF$SalePrice), res_log, main = "Plot of Log Model Residuals vs. Log of Sale Price",
    xlab = "Log of Sale Price", ylab = "Residual")
abline(h = c(0, 0), col = "red")
abline(v = 350000, col = "blue")
```




```r
# Normal Probability Plot of Residuals from Log Model
qqnorm(stdres_log, main = "QQ Plot of Standardized Log Model Residuals",
    xlab = "Normal Scores", ylab = "Standardized Residual")
qqline(stdres_log)
```



Our new model produces an R-squared value of 0.9436579. Our residuals vs. fitted values plot shows a better pattern of homoscedasticity, which suggests that our linear regression model adequately captures the trend in the log-transformed data. The normal probability plot of the standardized residuals also shows better adherence to a linear pattern, which suggests that our assumption of normality is better. Indeed, the formal Shapiro-Wilks test produces a p-value of . . . , which means we . . .

Accepting our new model, we provide its summary as follows:

```r
summary(log_model)
```

```
##
## Call:
## lm(formula = log_model_formula, data = workingDF)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -137.14  -10.32    0.00   11.20  103.26
##
## Coefficients:
##                          Estimate     Std. Error t value
## (Intercept)          -1716.03085674   114.48640942 -14.989
## MSSubClass              -0.02995939     0.02548945  -1.175
## MSZoningFV              60.47159103     7.94689346   7.609
## MSZoningRH              42.67875830     8.80580694   4.847
## MSZoningRL              47.73173919     7.30581837   6.533
## MSZoningRM              37.98617005     7.34812424   5.170
## LotArea                  0.00075724     0.00008285   9.140
## StreetPave              30.57899811     9.30199786   3.287
## LandContourLow          -7.22698429     4.23095795  -1.708
## LotConfigCulDSac         8.11248080     2.37177321   3.420
## LandSlopeSev           -43.31096995     9.08978589  -4.765
## NeighborhoodCollgCr     -2.50821092     2.38173669  -1.053
## NeighborhoodCrawfor     23.48659030     3.39562760   6.917
## NeighborhoodEdwards    -11.37254495     2.52121054  -4.511
## NeighborhoodGilbert     -3.39137202     3.02577120  -1.121
## NeighborhoodMitchel    -12.47220322     3.35090267  -3.722
## NeighborhoodNAmes       -8.11281283     2.04690360  -3.963
## NeighborhoodNoRidge     16.22853609     4.16170567   3.899
## NeighborhoodNridgHt     14.86051648     3.38230029   4.394
## NeighborhoodNWAmes      -8.72069088     2.93342533  -2.973
## NeighborhoodOldTown     -6.33142638     2.98452770  -2.121
## NeighborhoodStoneBr     31.12471862     4.73336718   6.576
## Condition1Norm           7.98562958     1.74768802   4.569
## Condition1RRAe         -21.20613996     6.64124969  -3.193
## Condition2RRAe        -112.94404828    32.88456740  -3.435
## BldgTypeTwnhs          -22.70046074     4.64067977  -4.892
## BldgTypeTwnhsE         -11.98014470     3.59423883  -3.333
## OverallQual              8.93312559     0.80031560  11.162
## OverallCond              7.17279320     0.67088283  10.692
## YearBuilt                0.40725440     0.04761311   8.553
## YearRemodAdd             0.16527028     0.04407506   3.750
## RoofStyleShed           78.09605324    25.00672053   3.123
## RoofMatlCompShg        701.92466649    27.16356951  25.841
## RoofMatlMembran        781.57309122    36.06544242  21.671
## RoofMatlMetal          763.04056774    35.96483504  21.216
```

```
## RoofMatlRoll          693.75249381    34.49846529  20.110
## `RoofMatlTar&Grv`     695.23658083    27.52332538  25.260
## RoofMatlWdShake       690.79736316    29.09558708  23.742
## RoofMatlWdShngl       708.81488433    28.91135760  24.517
## Exterior1stBrkFace     13.99961876     3.41065487   4.105
## Exterior1stHdBoard     -5.08566400     1.78853493  -2.843
## Exterior1stPlywood     -5.02795841     2.48344853  -2.025
## `Exterior1stWd Sdng`  -10.53744420     3.37661720  -3.121
## `Exterior2ndWd Sdng`    8.21346147     3.31330640   2.479
## MasVnrTypeNone          2.22312336     1.83189917   1.214
## MasVnrTypeStone         7.63140756     2.37626684   3.212
## MasVnrArea              0.01049903     0.00490959   2.138
## ExterQualGd           -10.95570790     3.51363754  -3.118
## ExterQualTA           -13.33118032     3.71053610  -3.593
## FoundationWood        -31.61813698    12.15067926  -2.602
## BsmtQualEx             -8.73768274     4.74840179  -1.840
## BsmtQualFa            -13.74499307     2.52006919  -5.454
## BsmtQualGd            -12.34285929     2.88285118  -4.281
## BsmtCondGd             27.78757233    17.85224356   1.557
## BsmtCondPo              4.11168992     2.05238995   2.003
## BsmtExposureAv         17.28319242     2.27619388   7.593
## BsmtFinType1BLQ         5.03471221     1.73581568   2.900
## BsmtFinSF1              0.03999142     0.00312866  12.782
## BsmtFinType2ALQ        -6.02347318     3.84619410  -1.566
## BsmtFinSF2              0.03093945     0.00453599   6.821
## BsmtUnfSF               0.02216219     0.00301481   7.351
## X1stFlrSF               0.06165042     0.00384214  16.046
## X2ndFlrSF               0.06266840     0.00290489  21.573
## BedroomAbvGr           -3.93226062     1.08469755  -3.625
## KitchenAbvGr          -21.53184264     3.38540632  -6.360
## KitchenQualFa         -16.08887554     4.88414257  -3.294
## KitchenQualGd         -16.93136290     2.93594247  -5.767
## KitchenQualTA         -17.68206320     3.31375673  -5.336
## TotRmsAbvGrd            1.91592841     0.77687007   2.466
## FunctionalTyp          17.37026341     2.43766592   7.126
## GarageTypeBasment       3.28663197     2.69584437   1.219
## GarageCars              9.65939860     1.28598167   7.511
## GarageQualEx         -116.41084947    24.38022742  -4.775
## GarageQualFa         -100.67506523    24.95665310  -4.034
## GarageQualGd         -137.31136461    29.86724820  -4.597
## GarageQualPo         -110.37123131    24.18909630  -4.563
## GarageCondEx          104.92916535    24.81297014   4.229
## GarageCondFa          104.00627277    25.73211734   4.042
## GarageCondGd          113.96226795    26.90941448   4.235
## GarageCondPo          110.23568070    24.49287334   4.501
## WoodDeckSF              0.01820009     0.00490366   3.712
## ScreenPorch             0.04541392     0.01046270   4.341
## PoolArea                0.09644479     0.02418291   3.988
## PoolQCEx              -58.95017483    20.74401579  -2.842
## SaleTypeNew            24.02729040     3.02759964   7.936
## SaleConditionNormal    10.36173379     1.97022408   5.259
##                                  Pr(>|t|)
## (Intercept)         < 0.0000000000000002 ***
## MSSubClass                      0.240053
```

```
## MSZoningFV            0.00000000000005097 ***
## MSZoningRH            0.00000139875996807 ***
## MSZoningRL            0.00000000009044188 ***
## MSZoningRM            0.00000026951867306 ***
## LotArea             < 0.0000000000000002 ***
## StreetPave                      0.001037 **
## LandContourLow                  0.087841 .
## LotConfigCulDSac                0.000644 ***
## LandSlopeSev          0.00000209175914384 ***
## NeighborhoodCollgCr             0.292480
## NeighborhoodCrawfor   0.00000000000707377 ***
## NeighborhoodEdwards   0.00000701273933405 ***
## NeighborhoodGilbert             0.262557
## NeighborhoodMitchel             0.000206 ***
## NeighborhoodNAmes     0.00007768383413144 ***
## NeighborhoodNoRidge             0.000101 ***
## NeighborhoodNridgHt   0.00001200798478136 ***
## NeighborhoodNWAmes              0.003002 **
## NeighborhoodOldTown             0.034066 *
## NeighborhoodStoneBr   0.00000000006875033 ***
## Condition1Norm        0.00000533548055700 ***
## Condition1RRAe                  0.001440 **
## Condition2RRAe                  0.000611 ***
## BldgTypeTwnhs         0.00000111846163960 ***
## BldgTypeTwnhsE                  0.000882 ***
## OverallQual         < 0.0000000000000002 ***
## OverallCond         < 0.0000000000000002 ***
## YearBuilt           < 0.0000000000000002 ***
## YearRemodAdd                    0.000184 ***
## RoofStyleShed                   0.001828 **
## RoofMatlCompShg     < 0.0000000000000002 ***
## RoofMatlMembran     < 0.0000000000000002 ***
## RoofMatlMetal       < 0.0000000000000002 ***
## RoofMatlRoll        < 0.0000000000000002 ***
## `RoofMatlTar&Grv`   < 0.0000000000000002 ***
## RoofMatlWdShake     < 0.0000000000000002 ***
## RoofMatlWdShngl     < 0.0000000000000002 ***
## Exterior1stBrkFace    0.00004288143181888 ***
## Exterior1stHdBoard              0.004529 **
## Exterior1stPlywood              0.043104 *
## `Exterior1stWd Sdng`            0.001842 **
## `Exterior2ndWd Sdng`            0.013297 *
## MasVnrTypeNone                  0.225124
## MasVnrTypeStone                 0.001351 **
## MasVnrArea                      0.032655 *
## ExterQualGd                     0.001858 **
## ExterQualTA                     0.000339 ***
## FoundationWood                  0.009364 **
## BsmtQualEx                      0.065965 .
## BsmtQualFa            0.00000005830029264 ***
## BsmtQualGd            0.00001985913017155 ***
## BsmtCondGd                      0.119813
## BsmtCondPo                      0.045335 *
## BsmtExposureAv        0.00000000000005757 ***
```

```
## BsmtFinType1BLQ                   0.003785 **
## BsmtFinSF1           < 0.0000000000000002 ***
## BsmtFinType2ALQ                   0.117559
## BsmtFinSF2            0.0000000001354066 ***
## BsmtUnfSF             0.0000000000033695 ***
## X1stFlrSF            < 0.0000000000000002 ***
## X2ndFlrSF            < 0.0000000000000002 ***
## BedroomAbvGr                      0.000299 ***
## KitchenAbvGr          0.0000000027403511 ***
## KitchenQualFa                     0.001013 **
## KitchenQualGd         0.0000000996576412 ***
## KitchenQualTA         0.0000011110656332 ***
## TotRmsAbvGrd                      0.013777 *
## FunctionalTyp         0.0000000000166848 ***
## GarageTypeBasment                 0.222998
## GarageCars            0.0000000000010517 ***
## GarageQualEx          0.0000199197684032 ***
## GarageQualFa          0.0000578650530519 ***
## GarageQualGd          0.0000467302215045 ***
## GarageQualPo          0.0000549834749998 ***
## GarageCondEx          0.0000250530435079 ***
## GarageCondFa          0.0000559742829745 ***
## GarageCondGd          0.0000243771875424 ***
## GarageCondPo          0.0000073466410440 ***
## WoodDeckSF                        0.000214 ***
## ScreenPorch           0.0000152573392492 ***
## PoolArea              0.0000701133875159 ***
## PoolQCEx                          0.004553 **
## SaleTypeNew           0.0000000000000431 ***
## SaleConditionNormal   0.0000001677452681 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 20.57 on 1369 degrees of freedom
## Multiple R-squared:  0.9437, Adjusted R-squared:  0.9402
## F-statistic: 269.8 on 85 and 1369 DF,  p-value: < 0.00000000000000022
```

We conclude that the variables included above are most relevant in determining a house's sale price. In particular, those variables that are significant at the 0.01 level, i.e., are most significant.

Task 2: Making recommendations

```r
morty <- read.csv("/Users/booranium/usf/601_regression/project/Morty.txt",
    stringsAsFactors = T)
sig_var_morty <- names(morty)[bool_bic == TRUE]
sig_var_morty
```

```
##  [1] "X"            "Id"           "MSSubClass"   "MSZoning"
##  [5] "LotFrontage"  "Street"       "Alley"        "LotShape"
##  [9] "Utilities"    "LandSlope"    "Condition1"   "Condition2"
## [13] "BldgType"     "HouseStyle"   "OverallQual"  "OverallCond"
## [17] "YearBuilt"    "RoofStyle"    "RoofMatl"     "Exterior1st"
## [21] "MasVnrType"   "ExterQual"    "Foundation"   "BsmtCond"
```

```
## [25] "BsmtFinType1"  "BsmtFinSF1"    "BsmtUnfSF"     "TotalBsmtSF"
## [29] "Heating"       "HeatingQC"     "Electrical"    "X1stFlrSF"
## [33] "X2ndFlrSF"     "LowQualFinSF"  "GrLivArea"     "BsmtFullBath"
## [37] "BsmtHalfBath"  "FullBath"      "HalfBath"      "BedroomAbvGr"
## [41] "KitchenAbvGr"  "TotRmsAbvGrd"  "Fireplaces"    "GarageType"
## [45] "GarageYrBlt"   "GarageFinish"  "GarageCars"    "GarageArea"
## [49] "GarageQual"    "GarageCond"    "PavedDrive"    "WoodDeckSF"
## [53] "OpenPorchSF"   "EnclosedPorch" "X3SsnPorch"    "PoolArea"
## [57] "PoolQC"        "Fence"         "YrSold"        "SaleType"
## [61] "SalePrice"     NA              NA              NA
## [65] NA              NA              NA              NA
## [69] NA              NA              NA              NA
## [73] NA              NA              NA              NA
## [77] NA              NA              NA              NA
## [81] NA              NA              NA              NA
## [85] NA
```

To make a recommendation to Morty regarding the selling price of his house, we leverage the model built above to make a prediction of Sale Price based on select attributes as determined above.

```
morty <- c()
prediction <- predict(log_model, newdata = morty, type = "response",
    interval = "confidence", level = 0.95)
prediction
```

We see that the 95% CI for the predicted Sale Price of Morty's house, based on selected attributes, is … As such, we recommend that Morty sell his house at a maximum of …, which is more/less than the other firm's recommendation of $143K.

In addition, we see that

Part II: Predictive Modelling