

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

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

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

```
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 ...  
## $ 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 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 ...
## $ 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 ...
## $ Electrical   : Factor w/ 5 levels "FuseA","FuseF",...: 5 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 %>% dplyr::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))
```

```

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 NAs per variable left in the dataset.

```

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

NA_count <- housingDF %>% dplyr::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 %

```
colnames(housingDF)
```

```
## [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"    "RoofMat1"      "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"
```

Table 2 shows the list of remaining variables where NA indicates missing data. We impute NAs 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)

# Convert MSSubClass to factor
housingDF$MSSubClass <- factor(housingDF$MSSubClass)
housingDF$MoSold <- factor(housingDF$MoSold)
```

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$SalePrice, main = 'Histogram of Sale Price')
# boxplot(housingDF$SalePrice, 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 %>% dplyr::select(Neighborhood, SalePrice) %>% ggplot(aes(factor(Neighborhood),
  SalePrice)) + geom_boxplot() + theme(axis.text.x = element_text(angle = 90,
  hjust = 1)) + xlab("Neighborhoods")
```

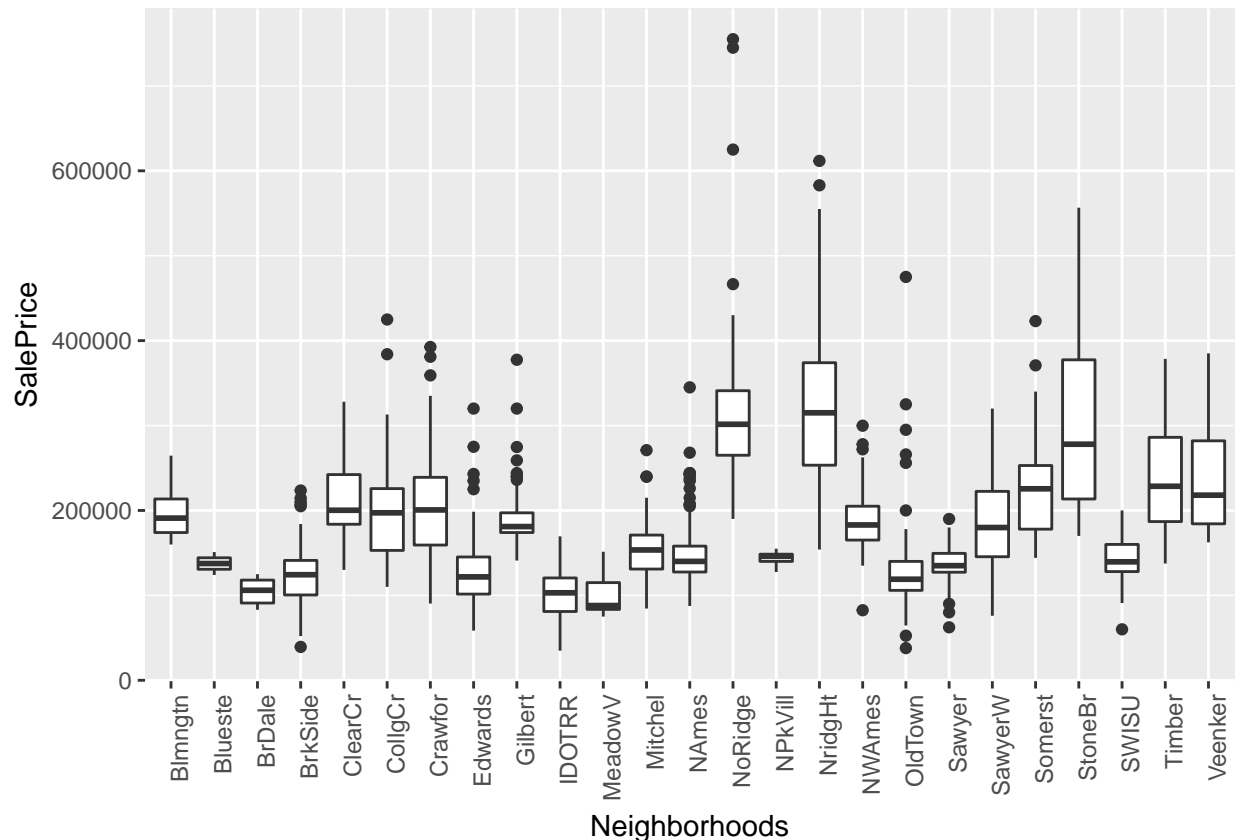


Figure 1: SalePrice distribution per neighborhood

From Figure 1, we can observe that Brookside and Meadow Vista have the lowest median house price while Northridge and Northridge Height have the highest 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 utilities 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,
        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))
}

```

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

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

```
doPlots(housingDF, fun = plotHist, ii = c(8, 9, 11, 12), 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.

```

housingDF %>% dplyr::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.

```

num_var <- names(housingDF)[which(sapply(housingDF, is.numeric))]
housing_numeric <- housingDF[num_var]

```

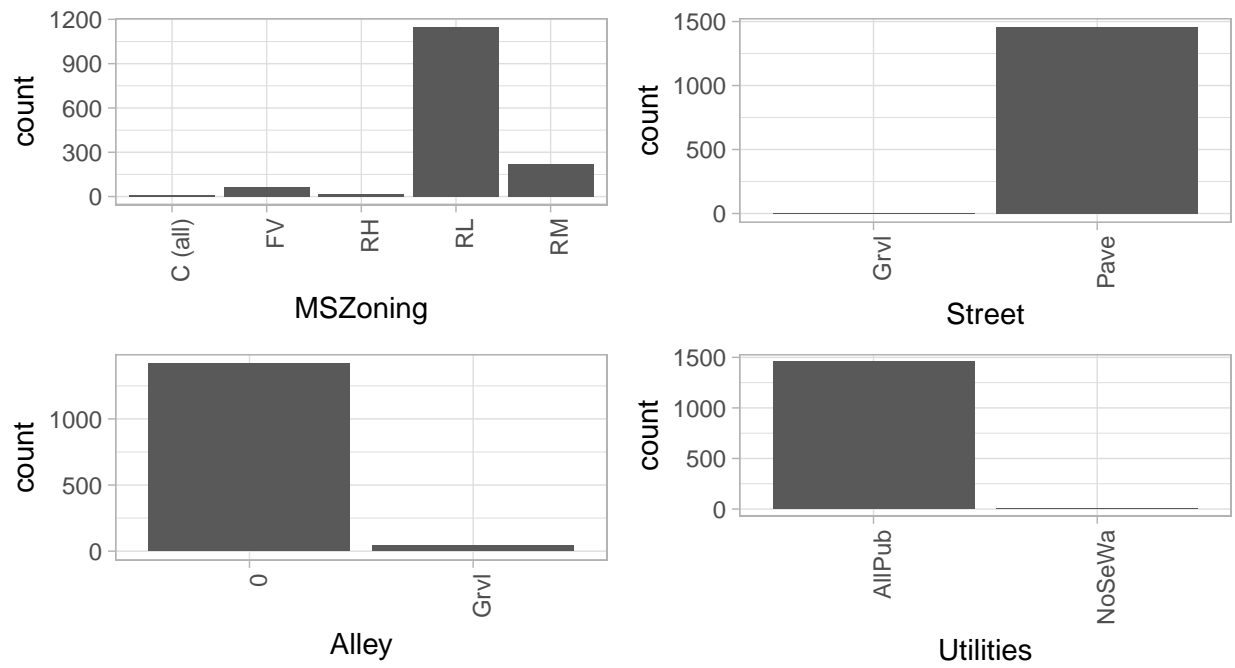


Figure 2: Locality, access, utility features distribution

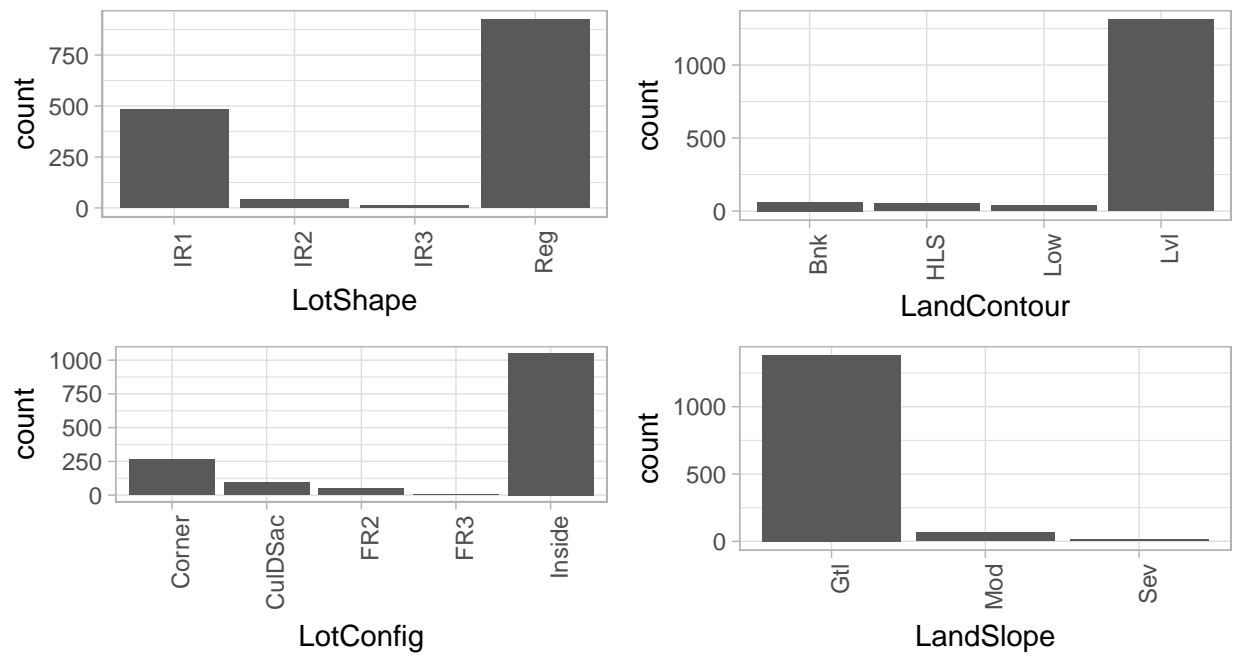


Figure 3: Lot/Land feature distribution

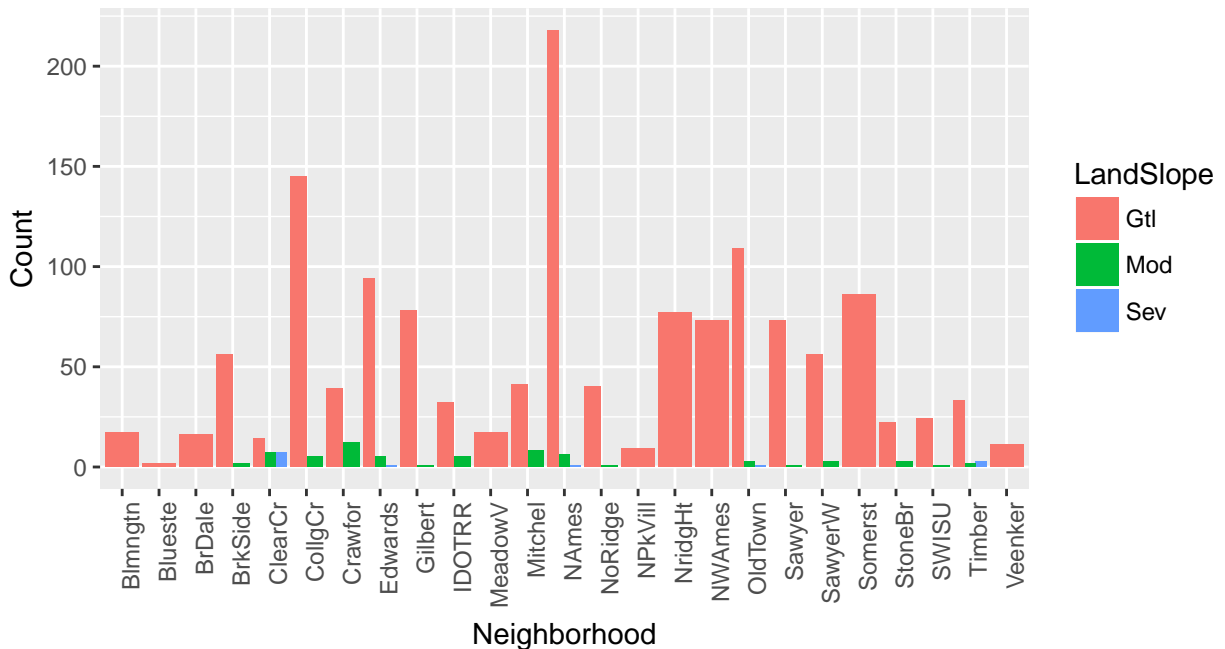


Figure 4: Neighborhood level slope distribution

```

correlations <- cor(housing_numeric[, -1])
highcorr <- c(names(correlations[, "SalePrice"])[which(correlations[,
  "SalePrice"] > 0.5)], names(correlations[, "SalePrice"])[which(correlations[,
  "SalePrice"] < -0.2)])
data_corr <- housingDF[highcorr]
doPlots(data_corr, fun = plotCorr, ii = 1:10)

```

Task 1. Building the Explanatory Model

Testing for Influential Points

Having dealt with the NAs 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)

```

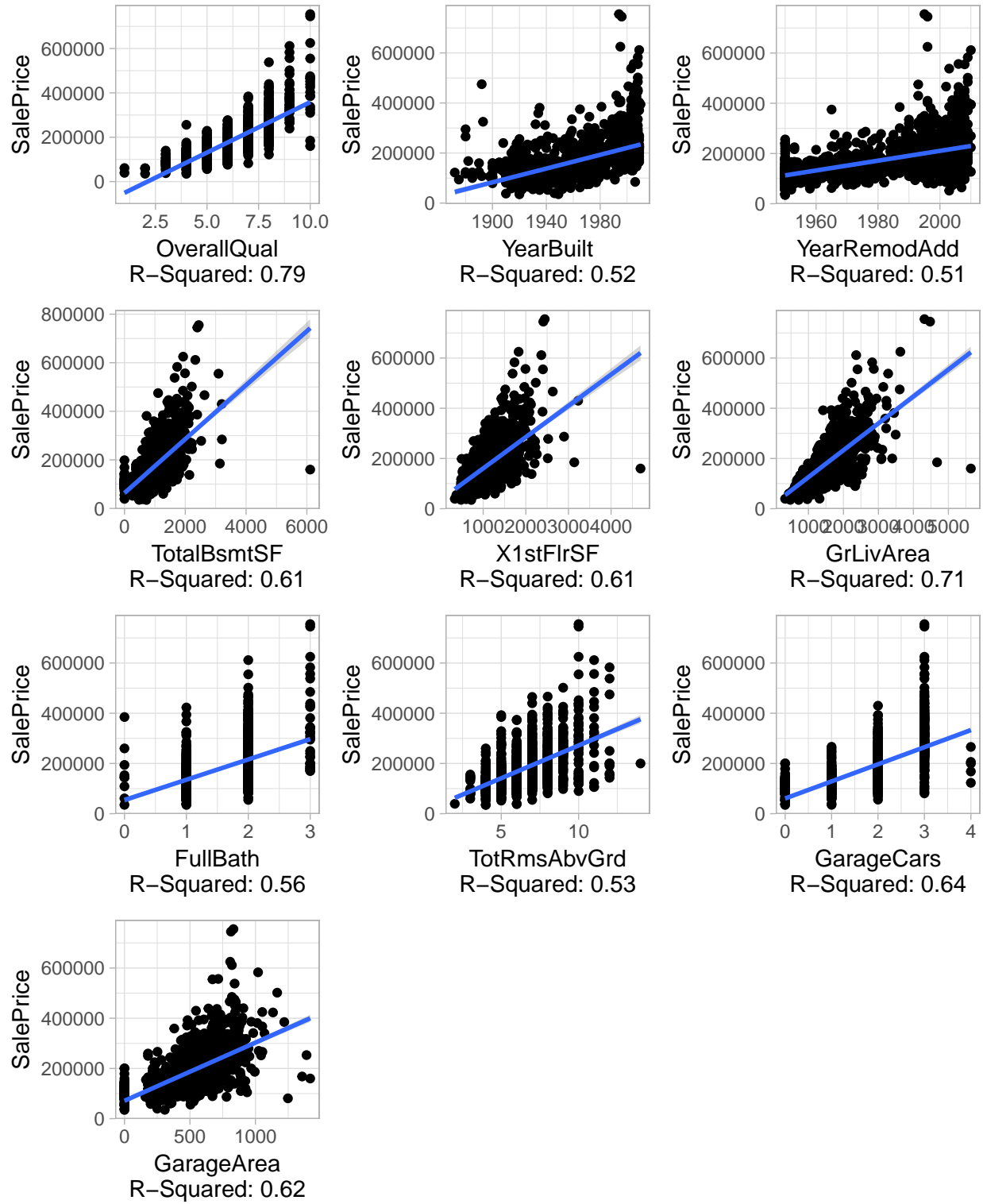
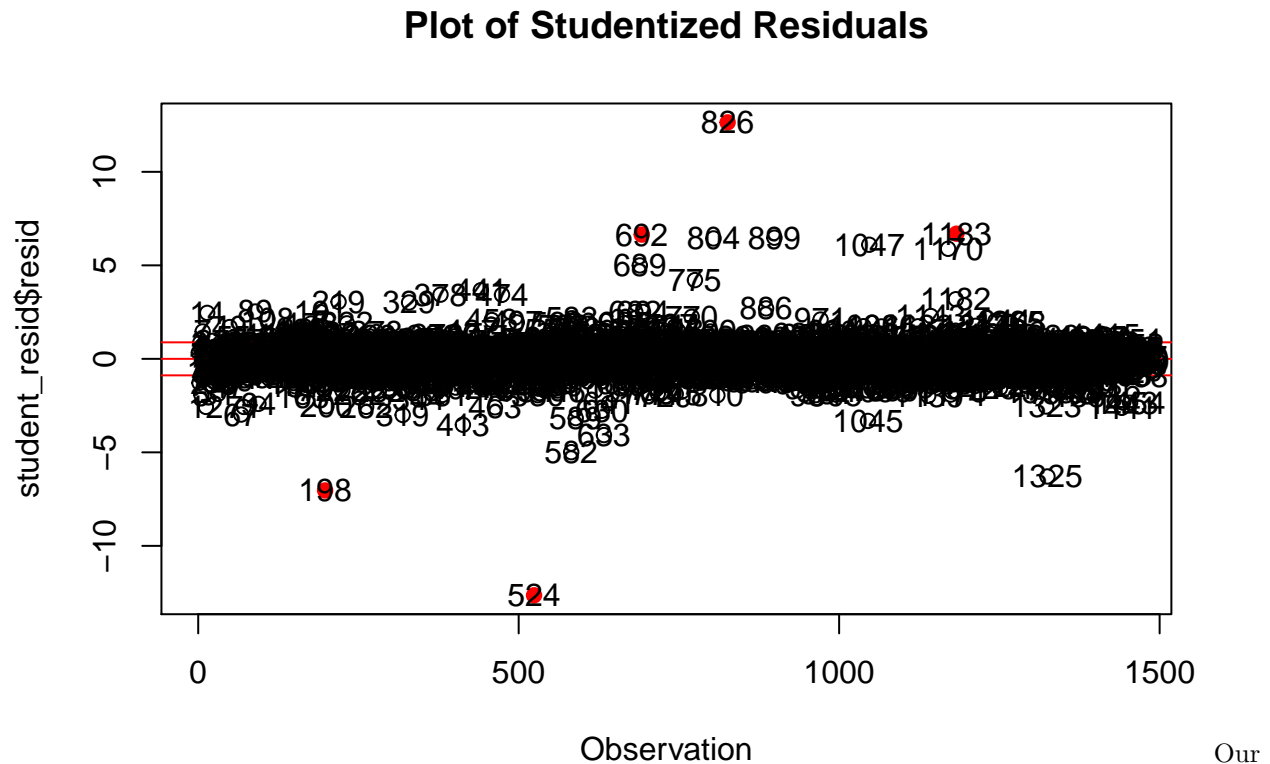


Figure 5: Scatter plot of variables showing high positive linear relationship with SalePrice

Note that according to the criterion of threshold $t = 2 \cdot \sqrt{n/p} = 0.85$, 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.

11

```
text(student_resid$ix, student_resid$resid, labels = student_resid$ix)
```

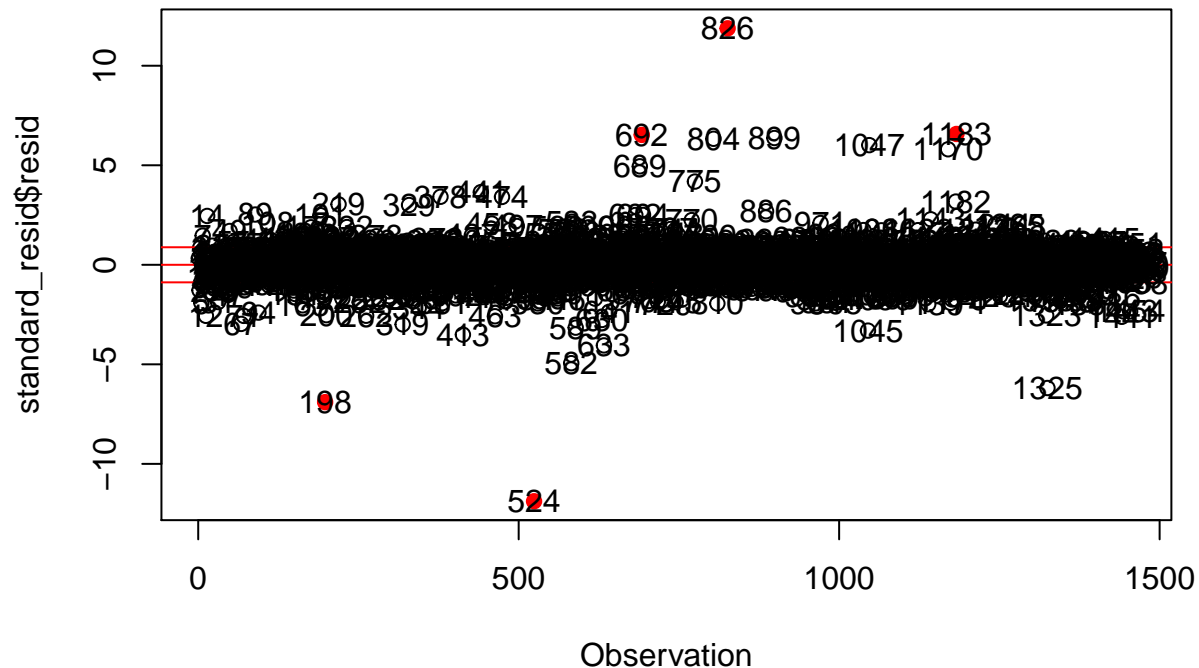


plot of studentized residuals indicates that observations all 5 points are leverage points.

```
# 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)
```

Plot of Standardized Residuals



Our plots of studentized and standardized residuals 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))
# nrow(workingDF) #1455

# 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 significant at the $\alpha = 0.05$ level.

```
# load('/Users/booranium/usf/601_regression/project/IowaHousing/AIC_model.rda')
## model loaded as 'model_aic'
load("/Users/santhoshhari/Documents/Coursework/LinearRegression/IowaHousing/AIC_model.rda") # model lo

# Find coefficients significant at the alpha = 0.01 level
bool_aic <- summary(model_aic)$coeff[-1, 4] < 0.01
sig_var_aic <- names(bool_aic)[bool_aic == TRUE]
```

Per the BIC criterion, the following are the predictor variables significant at the $\alpha = 0.05$ level.

```
# load('/Users/booranium/usf/601_regression/project/IowaHousing/BIC_model.rda')
# # model loaded as 'model_bic'
load("/Users/santhoshhari/Documents/Coursework/LinearRegression/IowaHousing/BIC_model.rda") # model lo

# find coefficients significant at the alpha = 0.01 level
bool_bic <- summary(model_aic)$coeff[-1, 4] < 0.01
sig_var_bic <- names(bool_bic)[bool_bic == TRUE]
```

Note that both criteria 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 75 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)

# create a model with scaled Xs and Y for multicollinearity
# detection
model_sig_scaled <- lm(formula = model_sig_formula, data = as.data.frame(scale(workingDF)))
```

We check for multicollinearity in our model by checking for Variance Inflation Factors:

```
vif(model_sig_scaled)[(sqrt(vif(model_sig_scaled)) > 10) == TRUE]
```

```
## GarageQualPo GarageCondPo
##      184.2926      171.8100
```

We see that there are two variables with VIF values $>$ threshold = 10. This tells us there is multicollinearity present in the dataset. We verify this by checking the Singular Value Criteria for multicollinearity:

```
coll_out = colldiag(model_sig, scale = TRUE, center = FALSE, add.intercept = TRUE)
coll_out$condindx[(coll_out$condindx > 30) == TRUE]
```

```
## [1] 32.49461 36.98012 41.51708 50.60676 56.13318 71.42520
## [7] 93.12395 271.11407 306.50195 915.64842 1232.96408
```

We see that there are several entries $>$ threshold = 30, and hence we conclude that multicollinearity exists. In order to identify which variables are multicollinear:

```
coll_out = colldiag(model_sig, scale = TRUE, center = FALSE, add.intercept = TRUE)
# unlist(coll_out[1], use.names = F) >30
# colnames(coll_out$pi)[unlist(coll_out[1], use.names = F) >30]
# dim(as.matrix(coll_out$pi[unlist(coll_out[1], use.names = F)
# >30, ]))
```

Based on results, we drop the Garage Condition variables since they are collinear with the Garage Quality variables.

```
# Drop GarageCondition variables since they are correlated with
# Garage Quality variables
drop = c("GarageCondEx", "GarageCondFa", "GarageCondGd", "GarageCondPo")
new_sig_var_bic = sig_var_bic[!sig_var_bic %in% drop]
```

We then rerun the model and recheck for multicollinearity using VIFS:

```

new_model_formula <- as.formula(paste("SalePrice ~ ", paste(new_sig_var_bic,
collapse = "+")))
new_model <- lm(formula = new_model_formula, data = workingDF)

# create a model with scaled Xs and Y for multicollinearity
# detection
new_model_scaled <- lm(formula = new_model_formula, data = as.data.frame(scale(workingDF)))

# check for MC
vif(new_model_scaled)[(sqrt(vif(new_model_scaled)) > 10) == TRUE]

```

```
## named numeric(0)
```

Using our rule of thumb, we conclude that there is no more multicollinearity in our model since there are no VIF values > 10. We print the summary of our model below:

```
summary(new_model)
```

```

##
## Call:
## lm(formula = new_model_formula, data = workingDF)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -131485  -10264         0    10154   141543
##
## Coefficients:
##              Estimate      Std. Error t value
## (Intercept)   -1501264.91302    112244.24739  -13.375
## MSSubClass120    -16220.08951     2850.38356   -5.690
## MSSubClass160   -22075.21962     3347.75394   -6.594
## MSZoningFV       38156.25052     8036.88635    4.748
## MSZoningRH       19652.20647     8926.62798    2.202
## MSZoningRL      23371.47826     7386.62559    3.164
## MSZoningRM      19051.08011     7420.13904    2.567
## LotArea          0.75227         0.08234    9.137
## StreetPave      38663.50280     9376.39492    4.123
## LandContourLow  -10195.83610     4254.25975   -2.397
## LotConfigCulDSac  8779.72336     2388.54431    3.676
## LandSlopeSev    -41595.28359     8804.48586   -4.724
## NeighborhoodCollgCr  258.49691     2358.95728    0.110
## NeighborhoodCrawfor 19613.85511     3417.66256    5.739
## NeighborhoodEdwards -8495.28154     2511.56384   -3.382
## NeighborhoodGilbert -2895.07052     3018.05498   -0.959
## NeighborhoodMitchel -13388.23766     3378.24743   -3.963
## NeighborhoodNames  -8194.95954     2039.33105   -4.018
## NeighborhoodNoRidge 27488.21640     4167.77030    6.595
## NeighborhoodNridgHt 19051.93042     3380.96462    5.635
## NeighborhoodNWames -12512.36145     2915.34823   -4.292
## NeighborhoodOldTown -5529.93302     2983.95330   -1.853
## NeighborhoodStoneBr 40921.72927     4719.78982    8.670
## Condition1Norm     6622.88542     1762.88860    3.757
## Condition1RRaE    -17398.07438     6705.93805   -2.594
## Condition2RRaE    -95060.05239    33065.48249   -2.875
## OverallQual       6688.31950      804.96840    8.309

```

## OverallCond	5355.15268	668.09521	8.016
## YearBuilt	312.30718	46.27090	6.750
## YearRemodAdd	108.54151	44.41420	2.444
## RoofStyleShed	64239.54318	25119.21237	2.557
## RoofMatlCompShg	647160.04869	26054.58228	24.839
## RoofMatlMembran	721671.17292	35301.90723	20.443
## RoofMatlMetal	690777.89068	35169.27755	19.642
## RoofMatlRoll	634745.46777	33587.18762	18.898
## `RoofMatlTar&Grv`	637476.78963	26681.46875	23.892
## RoofMatlWdShake	641591.64489	28100.88317	22.832
## RoofMatlWdShngl	683006.14049	27586.87806	24.758
## Exterior1stBrkFace	16337.84226	3365.15389	4.855
## `Exterior1stWd Sdng`	-7630.98849	3370.11676	-2.264
## `Exterior2ndWd Sdng`	7733.43621	3344.56420	2.312
## MasVnrTypeStone	7204.58508	2323.17645	3.101
## MasVnrArea	12.07108	3.97553	3.036
## ExterQualGd	-20087.06918	3524.14146	-5.700
## ExterQualTA	-22829.76580	3712.79583	-6.149
## BsmtQualEx	-12273.92721	4728.68724	-2.596
## BsmtQualFa	-20428.76983	2476.52955	-8.249
## BsmtQualGd	-16048.79772	2840.07243	-5.651
## BsmtExposureAv	19595.59377	2273.19595	8.620
## BsmtFinType1BLQ	4137.92986	1750.48172	2.364
## BsmtFinSF1	37.75769	3.05222	12.371
## BsmtFinSF2	25.80156	4.40669	5.855
## BsmtUnfSF	22.51566	2.93701	7.666
## X1stFlrSF	52.28720	3.81987	13.688
## X2ndFlrSF	56.91686	2.79574	20.358
## BedroomAbvGr	-6118.75256	1094.29142	-5.592
## KitchenAbvGr	-25400.14493	3016.82367	-8.419
## KitchenQualFa	-19644.45816	4918.27287	-3.994
## KitchenQualGd	-22306.88810	2943.12682	-7.579
## KitchenQualTA	-22836.34400	3323.75948	-6.871
## TotRmsAbvGrd	2483.27537	782.09095	3.175
## FunctionalTyp	14814.79638	2469.94324	5.998
## GarageCars	9240.14566	1288.45962	7.171
## GarageQualEx	-15039.17858	4315.98810	-3.485
## GarageQualFa	-1456.50341	6588.04379	-0.221
## GarageQualGd	-16857.98707	12738.07414	-1.323
## GarageQualPo	-10156.77637	3230.28881	-3.144
## WoodDeckSF	15.11486	4.93408	3.063
## ScreenPorch	41.82895	10.38260	4.029
## PoolArea	46.54277	17.56247	2.650
## SaleTypeNew	22639.33785	3056.75435	7.406
## SaleConditionNormal	6635.61415	1988.58730	3.337
##	Pr(> t)		
## (Intercept)	< 0.0000000000000002	***	
## MSSubClass120	0.000000015441952884	***	
## MSSubClass160	0.000000000060744262	***	
## MSZoningFV	0.000002271756534403	***	
## MSZoningRH	0.027863	*	
## MSZoningRL	0.001590	**	
## MSZoningRM	0.010348	*	
## LotArea	< 0.0000000000000002	***	

## StreetPave	0.000039538361591505	***
## LandContourLow	0.016679	*
## LotConfigCulDSac	0.000246	***
## LandSlopeSev	0.000002543849781085	***
## NeighborhoodCollgCr	0.912758	
## NeighborhoodCrawfor	0.000000011688777608	***
## NeighborhoodEdwards	0.000738	***
## NeighborhoodGilbert	0.337600	
## NeighborhoodMitchel	0.000077768287814754	***
## NeighborhoodNAMES	0.000061736078775388	***
## NeighborhoodNoRidge	0.000000000060196556	***
## NeighborhoodNridgHt	0.000000021177336753	***
## NeighborhoodNWames	0.000018948718917523	***
## NeighborhoodOldTown	0.064063	.
## NeighborhoodStoneBr	< 0.0000000000000002	***
## Condition1Norm	0.000179	***
## Condition1RR Ae	0.009575	**
## Condition2RR Ae	0.004103	**
## OverallQual	0.000000000000000228	***
## OverallCond	0.0000000000000002313	***
## YearBuilt	0.000000000021756978	***
## YearRemodAdd	0.014656	*
## RoofStyleShed	0.010652	*
## RoofMatlCompShg	< 0.0000000000000002	***
## RoofMatlMembran	< 0.0000000000000002	***
## RoofMatlMetal	< 0.0000000000000002	***
## RoofMatlRoll	< 0.0000000000000002	***
## `RoofMatlTar&Grv`	< 0.0000000000000002	***
## RoofMatlWdShake	< 0.0000000000000002	***
## RoofMatlWdShngl	< 0.0000000000000002	***
## Exterior1stBrkFace	0.000001340618306495	***
## `Exterior1stWd Sdng`	0.023709	*
## `Exterior2ndWd Sdng`	0.020911	*
## MasVnrTypeStone	0.001966	**
## MasVnrArea	0.002439	**
## ExterQualGd	0.000000014636523040	***
## ExterQualTA	0.000000001018871882	***
## BsmtQualEx	0.009542	**
## BsmtQualFa	0.000000000000000368	***
## BsmtQualGd	0.000000019361569651	***
## BsmtExposureAv	< 0.0000000000000002	***
## BsmtFinType1BLQ	0.018222	*
## BsmtFinSF1	< 0.0000000000000002	***
## BsmtFinSF2	0.000000005947144042	***
## BsmtUnfSF	0.000000000000033196	***
## X1stFlrSF	< 0.0000000000000002	***
## X2ndFlrSF	< 0.0000000000000002	***
## BedroomAbvGr	0.000000027086389464	***
## KitchenAbvGr	< 0.0000000000000002	***
## KitchenQualFa	0.000068334389684203	***
## KitchenQualGd	0.000000000000063342	***
## KitchenQualTA	0.000000000009635832	***
## TotRmsAbvGrd	0.001530	**
## FunctionalTyp	0.000000002545803481	***

```
## GarageCars          0.000000000001204751 ***
## GarageQualEx        0.000508 ***
## GarageQualFa        0.825061
## GarageQualGd        0.185910
## GarageQualPo        0.001701 **
## WoodDeckSF          0.002231 **
## ScreenPorch         0.000059123036763314 ***
## PoolArea            0.008138 **
## SaleTypeNew         0.000000000000224875 ***
## SaleConditionNormal 0.000870 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 20860 on 1383 degrees of freedom
## Multiple R-squared:  0.9293, Adjusted R-squared:  0.9257
## F-statistic: 256.1 on 71 and 1383 DF,  p-value: < 0.000000000000000022
```

At first glance, the multiple R-squared value of 0.9293 indicates that 92.93% of the variability in SalePrice around its mean is explained by the model, i.e. by the predictor variables that have been included. This suggests a high-performing explanatory model.

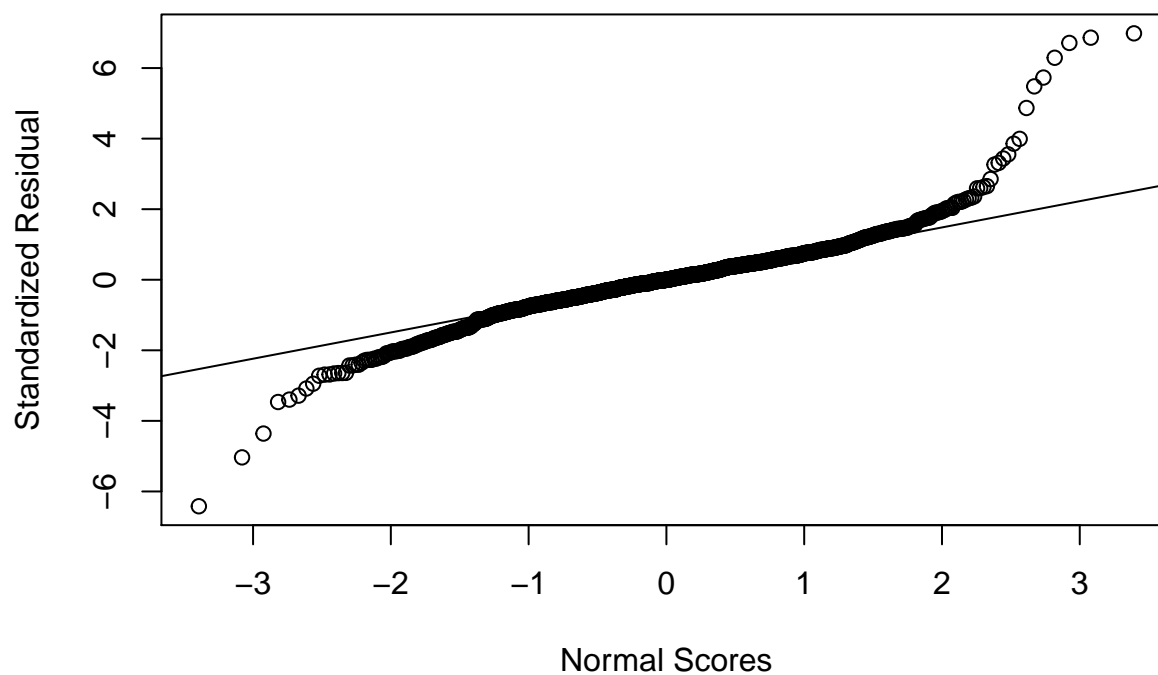
Let Before welcoming this conclusion, we validate the linearity and normality assumptions of our model by checking our residuals as follows:

```
# Residual plots

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

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

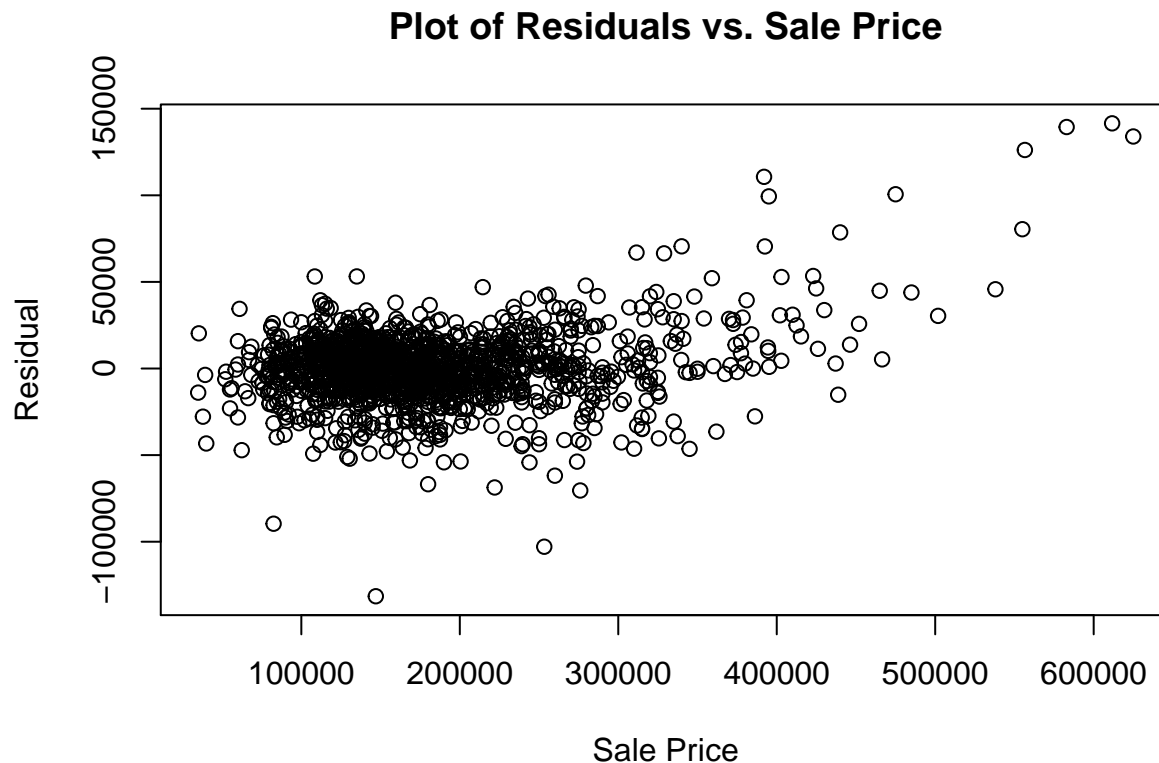
QQ Plot of Standardized Residuals



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

```
##  
##  Shapiro-Wilk normality test  
##  
## data:  res  
## W = 0.91064, 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")
```

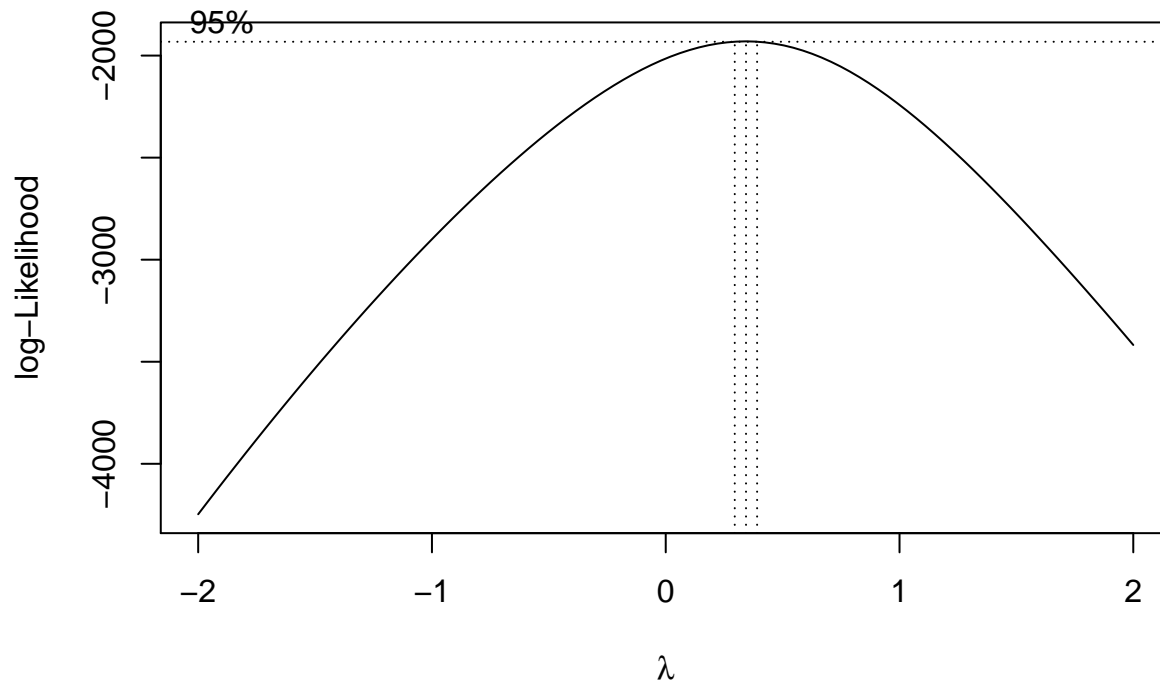


```
# abline(h=c(0,0), col = 'red') abline(v=350000, col = 'blue')
```

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 suggests that for the last quartile of high-priced houses, the fitted regression model is not as adequate as it is for the rest of the population. The normal probability (QQ) plot corroborates this finding: it shows deviance from linearity at both tail ends of the residual range, 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.

```
boxcox(new_model)
```



We see that $\lambda = 1$ is not captured in the 95% CI of lambdas, indicated by the three vertical dashed lines. This means a transformation is necessary. We choose $\lambda = \sim 0.5$ since this is an interpretable transformation value. Below, we apply a square root transformation to Sale Price, refit the model, and revalidate our model assumptions with residual plots as above.

```
# Log Transformation
log_model_formula <- as.formula(paste("sqrt(SalePrice) ~ ", paste(new_sig_var_bic,
  collapse = "+")))
log_model <- lm(formula = log_model_formula, data = workingDF)

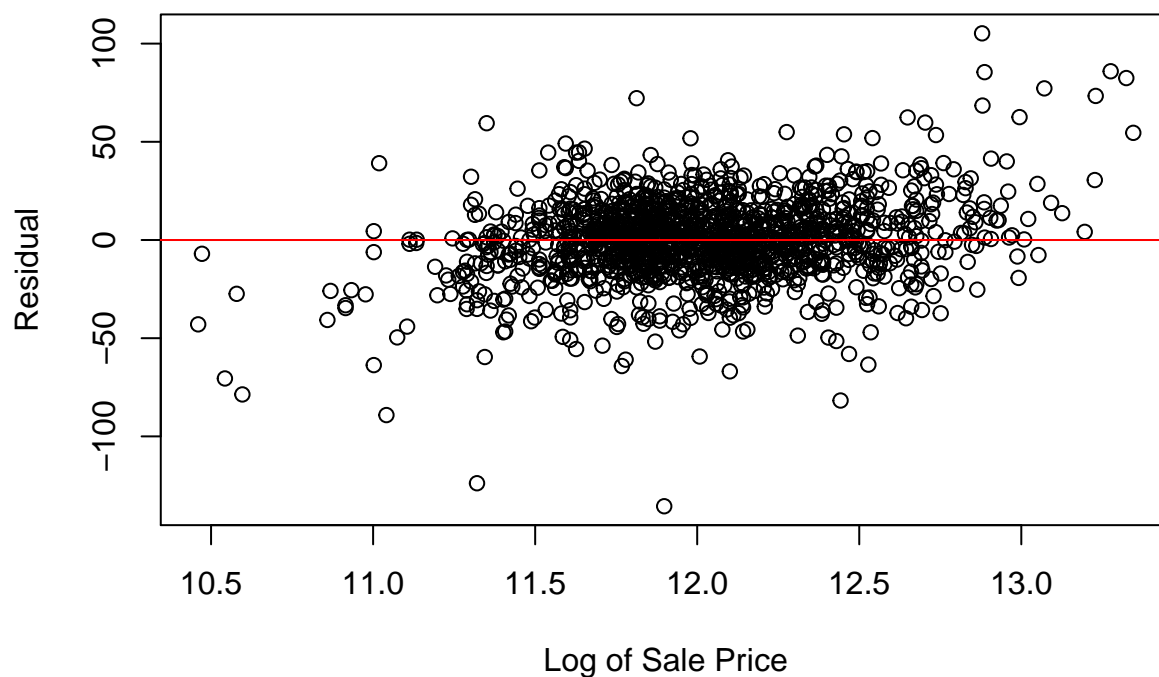
res_log = resid(log_model) # residuals
stdres_log = rstandard(log_model) # standardized residuals

shapiro.test(res)

##
##  Shapiro-Wilk normality test
##
## data:  res
## W = 0.91064, p-value < 0.000000000000000022

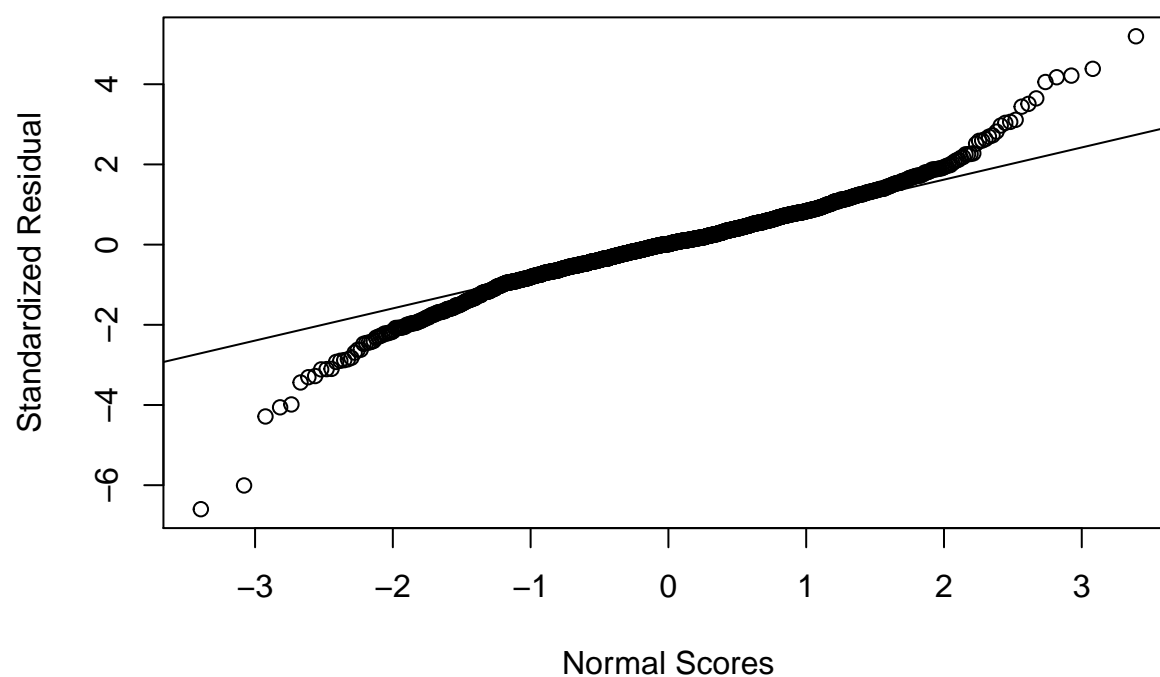
# Plot of Residuals from Log Model vs. Fitted Values
plot(log(workingDF$SalePrice), res_log, main = "Plot of Log Model Residuals vs. Log of Sale Price",
  xlab = "Log of Sale Price", ylab = "Residual")
abline(h = c(0, 0), col = "red")
abline(v = 350000, col = "blue")
```

Plot of Log Model Residuals vs. Log of Sale Price



```
# 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)
```

QQ Plot of Standardized Log Model Residuals



Our new model produces an R-squared value of 0.9411, indicating that 94.11% of the variance in Sale Price is captured by the model. 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.

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

```
summary(log_model)
```

```
##
## Call:
## lm(formula = log_model_formula, data = workingDF)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -135.567  -10.635    0.261   11.306  105.225
##
## Coefficients:
##              Estimate      Std. Error t value
## (Intercept)   -1671.18357282    112.58872806 -14.843
## MSSubClass120    -10.64752125     2.85913147  -3.724
## MSSubClass160   -26.18081164     3.35802829  -7.796
## MSZoningFV       65.46041153     8.06155177   8.120
## MSZoningRH       42.89043937     8.95402404   4.790
## MSZoningRL       48.03525028     7.40929535   6.483
## MSZoningRM       38.45017202     7.44291165   5.166
## LotArea         0.00073517     0.00008259   8.902
## StreetPave      31.53534731     9.40517134   3.353
## LandContourLow   -7.17666556     4.26731619  -1.682
## LotConfigCulDSac  8.15820547     2.39587482   3.405
## LandSlopeSev    -39.91544398     8.83150706  -4.520
## NeighborhoodCollgCr -0.01011682     2.36619698  -0.004
## NeighborhoodCrawfor 24.00687376     3.42815146   7.003
## NeighborhoodEdwards -10.53432796     2.51927190  -4.181
## NeighborhoodGilbert -1.28573035     3.02731747  -0.425
## NeighborhoodMitchel -12.26751912     3.38861536  -3.620
## NeighborhoodNames  -7.48489934     2.04558981  -3.659
## NeighborhoodNoRidge 17.29709181     4.18056130   4.138
## NeighborhoodNridgHt 15.42677210     3.39134089   4.549
## NeighborhoodNWAmes -11.21336222     2.92429552  -3.835
## NeighborhoodOldTown -5.14713279     2.99311114  -1.720
## NeighborhoodStoneBr 32.00644840     4.73427499   6.761
## Condition1Norm     7.48316286     1.76829895   4.232
## Condition1RR Ae    -20.11004958     6.72651876  -2.990
## Condition2RR Ae   -102.75611865    33.16696136  -3.098
## OverallQual       8.93921706     0.80743887  11.071
## OverallCond       7.15741893     0.67014561  10.680
## YearBuilt         0.40385545     0.04641291   8.701
## YearRemodAdd      0.16358008     0.04455051   3.672
## RoofStyleShed     72.49869955    25.19630392   2.877
## RoofMatlCompShg   674.45031929    26.13454451  25.807
## RoofMatlMembran   755.12050273    35.41024975  21.325
## RoofMatlMetal     730.65450605    35.27721303  20.712
## RoofMatlRoll      663.16972068    33.69026761  19.684
```

## `RoofMatlTar&Grv`	668.08268748	26.76335491	24.963
## RoofMatlWdShake	666.66645567	28.18712555	23.651
## RoofMatlWdShngl	703.17079135	27.67154294	25.411
## Exterior1stBrkFace	16.54367684	3.37548164	4.901
## `Exterior1stWd Sdng`	-8.22155502	3.38045974	-2.432
## `Exterior2ndWd Sdng`	7.74299867	3.35482876	2.308
## MasVnrTypeStone	7.35971115	2.33030634	3.158
## MasVnrArea	0.00715501	0.00398773	1.794
## ExterQualGd	-10.85790351	3.53495715	-3.072
## ExterQualTA	-14.04465027	3.72419051	-3.771
## BsmtQualEx	-7.65520773	4.74319971	-1.614
## BsmtQualFa	-13.91804704	2.48413009	-5.603
## BsmtQualGd	-10.71663639	2.84878869	-3.762
## BsmtExposureAv	16.72368119	2.28017245	7.334
## BsmtFinType1BLQ	4.29191348	1.75585399	2.444
## BsmtFinSF1	0.04088735	0.00306158	13.355
## BsmtFinSF2	0.02891711	0.00442021	6.542
## BsmtUnfSF	0.02397144	0.00294602	8.137
## X1stFlrSF	0.06000525	0.00383159	15.661
## X2ndFlrSF	0.06310356	0.00280432	22.502
## BedroomAbvGr	-3.90758206	1.09764983	-3.560
## KitchenAbvGr	-24.61391838	3.02608238	-8.134
## KitchenQualFa	-18.19947694	4.93336719	-3.689
## KitchenQualGd	-18.59497607	2.95215936	-6.299
## KitchenQualTA	-19.88106371	3.33396019	-5.963
## TotRmsAbvGrd	2.06333085	0.78449120	2.630
## FunctionalTyp	16.21671425	2.47752357	6.546
## GarageCars	10.52823232	1.29241394	8.146
## GarageQualEx	-10.17295582	4.32923399	-2.350
## GarageQualFa	8.48332330	6.60826268	1.284
## GarageQualGd	-16.49507217	12.77716764	-1.291
## GarageQualPo	-2.39845798	3.24020265	-0.740
## WoodDeckSF	0.01810918	0.00494922	3.659
## ScreenPorch	0.05080009	0.01041447	4.878
## PoolArea	0.04605799	0.01761637	2.614
## SaleTypeNew	23.80567847	3.06613561	7.764
## SaleConditionNormal	9.87327013	1.99469032	4.950
##	Pr(> t)		
## (Intercept)	< 0.0000000000000002	***	
## MSSubClass120	0.000204	***	
## MSSubClass160	0.00000000000012444	***	
## MSZoningFV	0.00000000000001021	***	
## MSZoningRH	0.000001846634598217	***	
## MSZoningRL	0.000000000124711745	***	
## MSZoningRM	0.000000274125859981	***	
## LotArea	< 0.0000000000000002	***	
## StreetPave	0.000821	***	
## LandContourLow	0.092838	.	
## LotConfigCulDSac	0.000680	***	
## LandSlopeSev	0.000006722390861845	***	
## NeighborhoodCollgCr	0.996589		
## NeighborhoodCrawfor	0.0000000000003901696	***	
## NeighborhoodEdwards	0.000030779955347722	***	
## NeighborhoodGilbert	0.671115		


```

## NeighborhoodMitchel      0.000305 ***
## NeighborhoodNames        0.000263 ***
## NeighborhoodNoRidge      0.000037228199946703 ***
## NeighborhoodNridgHt      0.000005865667023773 ***
## NeighborhoodNWAmes       0.000131 ***
## NeighborhoodOldTown      0.085718 .
## NeighborhoodStoneBr      0.000000000020210524 ***
## Condition1Norm           0.000024705134413399 ***
## Condition1RR Ae          0.002842 **
## Condition2RR Ae          0.001987 **
## OverallQual               < 0.0000000000000002 ***
## OverallCond               < 0.0000000000000002 ***
## YearBuilt                 < 0.0000000000000002 ***
## YearRemodAdd              0.000250 ***
## RoofStyleShed             0.004072 **
## RoofMatlCompShg          < 0.0000000000000002 ***
## RoofMatlMembran          < 0.0000000000000002 ***
## RoofMatlMetal            < 0.0000000000000002 ***
## RoofMatlRoll             < 0.0000000000000002 ***
## `RoofMatlTar&Grv`        < 0.0000000000000002 ***
## RoofMatlWdShake          < 0.0000000000000002 ***
## RoofMatlWdShngl          < 0.0000000000000002 ***
## Exterior1stBrkFace        0.000001065352965006 ***
## `Exterior1stWd Sdng`     0.015139 *
## `Exterior2ndWd Sdng`     0.021145 *
## MasVnrTypeStone           0.001621 **
## MasVnrArea                0.072991 .
## ExterQualGd               0.002171 **
## ExterQualTA               0.000169 ***
## BsmtQualEx                0.106770
## BsmtQualFa                0.000000025419586296 ***
## BsmtQualGd                0.000176 ***
## BsmtExposureAv            0.000000000000377958 ***
## BsmtFinType1BLQ           0.014636 *
## BsmtFinSF1                < 0.0000000000000002 ***
## BsmtFinSF2                0.000000000085228977 ***
## BsmtUnfSF                 0.000000000000000895 ***
## X1stFlrSF                 < 0.0000000000000002 ***
## X2ndFlrSF                 < 0.0000000000000002 ***
## BedroomAbvGr              0.000383 ***
## KitchenAbvGr              0.000000000000000916 ***
## KitchenQualFa             0.000234 ***
## KitchenQualGd             0.000000000402233982 ***
## KitchenQualTA             0.0000000003135879653 ***
## TotRmsAbvGrd              0.008629 **
## FunctionalTyp              0.000000000083308288 ***
## GarageCars                0.000000000000000832 ***
## GarageQualEx              0.018922 *
## GarageQualFa              0.199446
## GarageQualGd              0.196926
## GarageQualPo              0.459293
## WoodDeckSF                0.000263 ***
## ScreenPorch               0.000001196738264173 ***
## PoolArea                  0.009033 **

```

```
## SaleTypeNew          0.000000000000015906 ***
## SaleConditionNormal  0.000000834280350583 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 20.93 on 1383 degrees of freedom
## Multiple R-squared:  0.9411, Adjusted R-squared:  0.9381
## F-statistic: 311.2 on 71 and 1383 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

```
# morty <-
# read.csv('/Users/booranium/usf/601_regression/project/Morty.txt',
# stringsAsFactors = T)
morty <- read.csv('/Users/santhoshhari/Documents/Coursework/LinearRegression/IowaHousing/Data/Morty.txt',
stringsAsFactors = T)
sig_var_morty <- names(morty)[bool_bic == TRUE]
sig_var_morty
```

```
## [1] "MSSubClass"      "MSZoning"        "LotArea"         "Street"
## [5] "Alley"           "LotShape"        "Utilities"        "LotConfig"
## [9] "LandSlope"      "Neighborhood"    "BldgType"         "OverallQual"
## [13] "OverallCond"    "YearBuilt"       "YearRemodAdd"     "RoofStyle"
## [17] "RoofMatl"       "Exterior1st"     "MasVnrType"       "MasVnrArea"
## [21] "ExterQual"      "Foundation"      "BsmtCond"         "BsmtFinType1"
## [25] "BsmtFinType2"   "CentralAir"      "Electrical"       "X1stFlrSF"
## [29] "X2ndFlrSF"     "GrLivArea"       "BsmtFullBath"     "BsmtHalfBath"
## [33] "FullBath"       "HalfBath"        "BedroomAbvGr"     "KitchenAbvGr"
## [37] "KitchenQual"    "TotRmsAbvGrd"    "GarageYrBlt"      "GarageCars"
## [41] "GarageCond"     "PavedDrive"      "WoodDeckSF"       "OpenPorchSF"
## [45] "X3SsnPorch"     "ScreenPorch"     "PoolArea"         "MiscFeature"
## [49] "MoSold"         "YrSold"          "SaleCondition"    "SalePrice"
## [53] NA              NA                NA                 NA
## [57] NA              NA                NA                 NA
## [61] NA              NA                NA                 NA
## [65] NA              NA                NA                 NA
## [69] NA              NA                NA                 NA
## [73] NA              NA                NA                 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