OM60 AP Statistics: Final Project Report

Vishak Srikanth

May 29, 2020

1. Introduction

For many people, buying a home is one of life's major milestones. Not only is it a significant financial commitment, but it is also a symbol of their self-worth. For some, buying a home is purely an investment opportunity from which they plan to earn profits or use as a means to build wealth. The attributes of a home (i.e. location, layout, size) can shape a person's lifestyle and influence how happy they are in their homes. For some of the proud homeowners, the housing booms have been a boon but for others saving up enough money to make a down payment and choosing the right property that matches their needs can be a difficult task. In either case, what people want to know is: Is a particular home worth it? Naturally, home prices vary depending on the size, layout, amenities, among other property characteristics and fluctuate over time. With so many attributes, it can be difficult to get a clear idea of what a fair price range for a home is. Inspired by discussions in the class and what we learned while studying statistics, I want to explore the question of whether it is possible to understand what are the key drivers of home prices (i.e.) what are the different factors that drive the price of a home and whether we can identify and use the attributes to create a statistical model to predict its price.

In this project we will explore how we can create a model of current prices of various types of homes in a location, apply multiple regression modeling techniques we learnt in class to ascertain which factors drive home prices and help potential buyers assess what the homes they are considering buying are worth. For the analysis we will be using a public data sets available such as Ames Housing Data (De Cock 2011). The Ames dataset contains 80 variables describing 2930 property sales that had taken place in Ames, Iowa between 2006 and 2010.

We will perform exploratory analysis to describe various descriptive statistics of the dataset, perform any data cleanup, and then build and tested 3 different regression models and a machine learning model based on random forest algorithm as part of my analysis of the Ames dataset. We will use R notebooks in the R Studio software to generate the reports and data summaries.

In the sections that follow, this project aims to build a predictive model for home prices from the Ames dataset that can help buyers assess whether the homes they are considering buying are worth the price based on property characteristics. In section 2, we summarize the data, along with any initial processing performed, and describe the dataset with some exploratory data analysis. Section 3 covers the various modeling approaches and highlights the results of each approach. Section 4 discusses the conclusions drawn from the results. Finally, Section 5 presents areas of further discussion, as well as the challenges encountered while modeling.

2 Exploratory Data Analysis

The project leverages the Ames Housing Data (De Cock, 2011), a more recent alternative housing dataset, that contains 80 variables describing 2930 property sales that had taken place in Ames, Iowa between 2006

and 2010.[1]

Since the goal of the modeling is to be useful in the predicting the home sale price in a normal market, only the records with a "Normal" Sale Condition are used and any foreclosure, trade, short sale, sale between family members, or an incomplete home sale type are dropped and only those with square footage less than 5000 square feet are used which represents what the home sizes typical buyers are looking for. Only the data that has residential zoning is used and agricultural, commercial, or industrial zoning are all dropped because the model is built to help individuals or families looking to buy the homes as their personal property and not for business or commercial purposes.

```
housedata <- read.csv("AmesHousing.csv", stringsAsFactors=FALSE)
mydata <- subset(housedata, Sale.Condition=="Normal" & Gr.Liv.Area < 5000 & MS.Zoning %in% c("FV",
rawdata <- mydata
housing_data_frame <- mydata
```

2.1 Basic Summary

First we explore the dataset by summarizing to understand the number and types of variables (numeric, categorical) in the dataset and their distributions. There are a total of 80 potential explanatory (predictor variables) and 1 response variable (SalePrice). Of the 80 predictor variables, 23 are nominal, 23 are ordinal, 14 are discrete, and 20 are continuous with sales price being the continuous response variable. The predictor variables capture basic characteristics that anyone wanting to buy a home would be interested in. The 20 continuous variables are related to measurements of area of various parts of the homes such as the sizes of lots, rooms, porches, and garages. The 14 discrete variables mostly have to do with the number of bedrooms, bathrooms, kitchens, etc. that a given property has. Geographic categorical variables that profile properties from individual Parcel ID level to the neighborhood level are included. The rest of the nominal variables identify characteristics of the property and dwelling type/structure. Most of the ordinal variables are rankings of the quality/condition of various aspects of the property such as pool, air conditioning, and lot characteristics.

```
print("Ames housing data set size:")
```

```
## [1] "Ames housing data set size:"
```

```
dim(housing_data_frame)
```

```
## [1] 2397   82
```

```
print("Dataset Summary")
```

```
## [1] "Dataset Summary"
```

```
summary(housing_data_frame)
```

```
##      Order           PID             MS.SubClass      MS.Zoning
##  Min.   :   1   Min.   :526301100   Min.   : 20.00   Length:2397
##  1st Qu.: 686   1st Qu.:531385020   1st Qu.: 20.00   Class :character
##  Median :1389   Median :535457090   Median : 50.00   Mode  :character
##  Mean   :1418   Mean   :716776374   Mean   : 57.93
##  3rd Qu.:2134   3rd Qu.:907187010   3rd Qu.: 70.00
##  Max.   :2930   Max.   :924152030   Max.   :190.00
##
##   Lot.Frontage       Lot.Area         Street             Alley
```

[1]Full description of the variables in the dataset: http://jse.amstat.org/v19n3/decock/DataDocumentation.txt

```
##  Min.   : 21.00   Min.   :  1300   Length:2397        Length:2397
##  1st Qu.: 57.00   1st Qu.:  7399   Class :character   Class :character
##  Median : 68.00   Median :  9373   Mode  :character   Mode  :character
##  Mean   : 68.18   Mean   : 10038
##  3rd Qu.: 80.00   3rd Qu.: 11404
##  Max.   :313.00   Max.   :215245
##  NA's   :449
##   Lot.Shape          Land.Contour        Utilities          Lot.Config
##  Length:2397        Length:2397        Length:2397        Length:2397
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##   Land.Slope         Neighborhood       Condition.1        Condition.2
##  Length:2397        Length:2397        Length:2397        Length:2397
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##   Bldg.Type          House.Style         Overall.Qual     Overall.Cond
##  Length:2397        Length:2397        Min.   : 1.000   Min.   :1.000
##  Class :character   Class :character   1st Qu.: 5.000   1st Qu.:5.000
##  Mode  :character   Mode  :character   Median : 6.000   Median :5.000
##                                        Mean   : 6.023   Mean   :5.654
##                                        3rd Qu.: 7.000   3rd Qu.:6.000
##                                        Max.   :10.000   Max.   :9.000
##
##    Year.Built   Year.Remod.Add  Roof.Style         Roof.Matl
##  Min.   :1872   Min.   :1950   Length:2397        Length:2397
##  1st Qu.:1953   1st Qu.:1965   Class :character   Class :character
##  Median :1971   Median :1991   Mode  :character   Mode  :character
##  Mean   :1970   Mean   :1983
##  3rd Qu.:1998   3rd Qu.:2002
##  Max.   :2010   Max.   :2010
##
##  Exterior.1st       Exterior.2nd       Mas.Vnr.Type        Mas.Vnr.Area
##  Length:2397        Length:2397        Length:2397        Min.   :   0.00
##  Class :character   Class :character   Class :character   1st Qu.:   0.00
##  Mode  :character   Mode  :character   Mode  :character   Median :   0.00
##                                                           Mean   :  96.73
##                                                           3rd Qu.: 150.00
##                                                           Max.   :1600.00
##                                                           NA's   :11
##   Exter.Qual         Exter.Cond         Foundation         Bsmt.Qual
##  Length:2397        Length:2397        Length:2397        Length:2397
##  Class :character   Class :character   Class :character   Class :character
```

```
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##   Bsmt.Cond         Bsmt.Exposure      BsmtFin.Type.1      BsmtFin.SF.1
##  Length:2397        Length:2397        Length:2397        Min.   :   0.0
##  Class :character   Class :character   Class :character   1st Qu.:   0.0
##  Mode  :character   Mode  :character   Mode  :character   Median : 388.0
##                                                           Mean   : 441.1
##                                                           3rd Qu.: 724.0
##                                                           Max.   :2288.0
##
##  BsmtFin.Type.2      BsmtFin.SF.2      Bsmt.Unf.SF     Total.Bsmt.SF
##  Length:2397        Min.   :   0.00   Min.   :   0.0   Min.   :   0
##  Class :character   1st Qu.:   0.00   1st Qu.: 210.0   1st Qu.: 784
##  Mode  :character   Median :   0.00   Median : 439.0   Median : 972
##                     Mean   :  55.49   Mean   : 529.1   Mean   :1026
##                     3rd Qu.:   0.00   3rd Qu.: 777.0   3rd Qu.:1248
##                     Max.   :1526.00   Max.   :2336.0   Max.   :3206
##
##    Heating           Heating.QC        Central.Air         Electrical
##  Length:2397        Length:2397        Length:2397        Length:2397
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##   X1st.Flr.SF    X2nd.Flr.SF     Low.Qual.Fin.SF     Gr.Liv.Area
##  Min.   : 334   Min.   :   0.0   Min.   :   0.000   Min.   : 334
##  1st Qu.: 868   1st Qu.:   0.0   1st Qu.:   0.000   1st Qu.:1104
##  Median :1062   Median :   0.0   Median :   0.000   Median :1432
##  Mean   :1135   Mean   : 339.2   Mean   :   4.339   Mean   :1479
##  3rd Qu.:1353   3rd Qu.: 704.0   3rd Qu.:   0.000   3rd Qu.:1724
##  Max.   :3820   Max.   :1872.0   Max.   :1064.000   Max.   :4316
##
##  Bsmt.Full.Bath   Bsmt.Half.Bath       Full.Bath       Half.Bath
##  Min.   :0.0000   Min.   :0.00000   Min.   :0.000   Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.:0.00000   1st Qu.:1.000   1st Qu.:0.0000
##  Median :0.0000   Median :0.00000   Median :2.000   Median :0.0000
##  Mean   :0.4336   Mean   :0.06219   Mean   :1.542   Mean   :0.3792
##  3rd Qu.:1.0000   3rd Qu.:0.00000   3rd Qu.:2.000   3rd Qu.:1.0000
##  Max.   :2.0000   Max.   :2.00000   Max.   :4.000   Max.   :2.0000
##  NA's   :1        NA's   :1
##  Bedroom.AbvGr   Kitchen.AbvGr  Kitchen.Qual       TotRms.AbvGrd
##  Min.   :0.000   Min.   :0.00   Length:2397        Min.   : 2.00
##  1st Qu.:2.000   1st Qu.:1.00   Class :character   1st Qu.: 5.00
##  Median :3.000   Median :1.00   Mode  :character   Median : 6.00
##  Mean   :2.856   Mean   :1.04                      Mean   : 6.37
```

```
##  3rd Qu.:3.000   3rd Qu.:1.00                      3rd Qu.: 7.00
##  Max.   :6.000   Max.   :3.00                      Max.   :13.00
##
##   Functional          Fireplaces     Fireplace.Qu       Garage.Type
##  Length:2397        Min.   :0.0000   Length:2397        Length:2397
##  Class :character   1st Qu.:0.0000   Class :character   Class :character
##  Mode  :character   Median :1.0000   Mode  :character   Mode  :character
##                     Mean   :0.6062
##                     3rd Qu.:1.0000
##                     Max.   :4.0000
##
##  Garage.Yr.Blt  Garage.Finish       Garage.Cars     Garage.Area
##  Min.   :1900   Length:2397        Min.   :0.000   Min.   :   0.0
##  1st Qu.:1960   Class :character   1st Qu.:1.000   1st Qu.: 318.0
##  Median :1977   Mode  :character   Median :2.000   Median : 472.0
##  Mean   :1976                      Mean   :1.737   Mean   : 462.5
##  3rd Qu.:1999                      3rd Qu.:2.000   3rd Qu.: 576.0
##  Max.   :2009                      Max.   :5.000   Max.   :1488.0
##  NA's   :113
##  Garage.Qual        Garage.Cond        Paved.Drive         Wood.Deck.SF
##  Length:2397        Length:2397        Length:2397        Min.   :   0.0
##  Class :character   Class :character   Class :character   1st Qu.:   0.0
##  Mode  :character   Mode  :character   Mode  :character   Median :   0.0
##                                                           Mean   :  96.4
##                                                           3rd Qu.: 168.0
##                                                           Max.   :1424.0
##
##  Open.Porch.SF    Enclosed.Porch     X3Ssn.Porch       Screen.Porch
##  Min.   :  0.00   Min.   :   0.00   Min.   :  0.000   Min.   :  0.00
##  1st Qu.:  0.00   1st Qu.:   0.00   1st Qu.:  0.000   1st Qu.:  0.00
##  Median : 24.00   Median :   0.00   Median :  0.000   Median :  0.00
##  Mean   : 44.82   Mean   :  23.15   Mean   :  2.476   Mean   : 16.36
##  3rd Qu.: 68.00   3rd Qu.:   0.00   3rd Qu.:  0.000   3rd Qu.:  0.00
##  Max.   :570.00   Max.   :1012.00   Max.   :508.000   Max.   :576.00
##
##    Pool.Area         Pool.QC            Fence           Misc.Feature
##  Min.   :  0.000   Length:2397        Length:2397        Length:2397
##  1st Qu.:  0.000   Class :character   Class :character   Class :character
##  Median :  0.000   Mode  :character   Mode  :character   Mode  :character
##  Mean   :  1.789
##  3rd Qu.:  0.000
##  Max.   :800.000
##
##     Misc.Val           Mo.Sold          Yr.Sold      Sale.Type
##  Min.   :    0.00   Min.   : 1.000   Min.   :2006   Length:2397
##  1st Qu.:    0.00   1st Qu.: 4.000   1st Qu.:2007   Class :character
##  Median :    0.00   Median : 6.000   Median :2008   Mode  :character
##  Mean   :   51.59   Mean   : 6.115   Mean   :2008
##  3rd Qu.:    0.00   3rd Qu.: 7.000   3rd Qu.:2009
##  Max.   :15500.00   Max.   :12.000   Max.   :2010
```

```
##
##  Sale.Condition       SalePrice
##  Length:2397        Min.   : 35000
##  Class :character   1st Qu.:130000
##  Mode  :character   Median :159000
##                     Mean   :176172
##                     3rd Qu.:207000
##                     Max.   :755000
##
```

```
# str(housing_data_frame)
```

Next we explore the other aspects of the dataset such as missing values, categorical variables, and other pre-processing and variable consolidation

2.2 Missing Data

Upon inspecting the dataset, there seem to be a lot of missing data in this dataset and most of the missing values are about the property not having the particular feature being described by the variable, e.g. alley access, basement, fireplace, garage, pool, fence. For example, these variables such as Alley have NA encoded as a level to specify "No Alley Access" so these are not really missing values. To simplify the modeling problem, we added a simple dummy variables (0/1) for whether or not the property had the feature were created.

The remaining missing values were in lot frontage (measure of street length connected to property in feet) and masonry veneer area (in square feet). While there are 490 missing values in the lot frontage variable, this is disproportionately spread across neighborhoods where some have missing lot frontage data for every house listed in that neighborhood, for example GrnHill and Landmark neighborhoods have no frontage data for property. We decided to drop these neighborhoods as they account for only 3 observations in the original dataset.

There is no single variable in the dataset that gives a reason why these values are missing. We make the assumption that the lot frontage for a given house is fairly similar to the other properties in the same neighborhood. So, we can use a median value to fill these missing values and reevaluate whether we want to add this variable to our models later on.

```
print('Are there any missing values in the data?')
```

```
## [1] "Are there any missing values in the data?"
```

```
any(is.na(housedata))
```

```
## [1] TRUE
```

```
print('How many missing values are are there?')
```

```
## [1] "How many missing values are are there?"
```

```
sum(is.na(housedata))
```

```
## [1] 13960
```

```
# return index of columns that have missing values
na.cols = which(colSums(is.na(housedata)) > 0)

print("How are the missing values broken down by variables?")
```

```
## [1] "How are the missing values broken down by variables?"
```

```
sort(colSums(sapply(housedata[na.cols], is.na)), decreasing = TRUE)
```

```
##        Pool.QC   Misc.Feature          Alley          Fence   Fireplace.Qu
##           2917           2824           2732           2358           1422
##   Lot.Frontage  Garage.Yr.Blt    Garage.Qual    Garage.Cond    Garage.Type
##            490            159            158            158            157
##  Garage.Finish      Bsmt.Qual      Bsmt.Cond  Bsmt.Exposure BsmtFin.Type.1
##            157             79             79             79             79
## BsmtFin.Type.2   Mas.Vnr.Area Bsmt.Full.Bath Bsmt.Half.Bath   BsmtFin.SF.1
##             79             23              2              2              1
##   BsmtFin.SF.2    Bsmt.Unf.SF  Total.Bsmt.SF    Garage.Cars    Garage.Area
##              1              1              1              1              1
```

```
# df for the lot frontage imputation data with median
df <- mydata
frontage_grouped_by_hood <- df %>%
  dplyr::select(Neighborhood, Lot.Frontage) %>%
  group_by(Neighborhood) %>%
  summarise(median_frontage = median(Lot.Frontage, na.rm = TRUE))

# Any missing lot frontage data for neighborhood?
# any(is.na(frontage_grouped_by_hood$median_frontage))

#Drop the 3 observations from GreenHill and LandMrk
mydata <- mydata %>%
  filter(Neighborhood != "GrnHill" & Neighborhood != "Landmrk")

# drop from frontage df as well
frontage_grouped_by_hood <- frontage_grouped_by_hood %>%
  filter(Neighborhood != "GrnHill" & Neighborhood != "Landmrk")

# redefine index for missing frontage data
index <- which(is.na(mydata$Lot.Frontage))

# for loop for lot frontage imputation
# first select neighborhood from first column of frontage df above based on the corresponding neig
# in the original df and calculate the median frontage for that neighborhood
for (i in index) {
  mynb = mydata$Neighborhood[i]
  mydf = as.data.frame(frontage_grouped_by_hood)
  med_frontage = mydf[mydf$Neighborhood == mynb , 'median_frontage']
  # # then replace the missing value with the median
  # print(paste0(mynb, " ",str(med_frontage)))
  mydata[i, 'Lot.Frontage'] = med_frontage

}

# check to see lot frontage imputation with median by neighborhood worked
# any(is.na(mydata$Lot.Frontage))
```

```
# # Alley access
mydata$Has.Alley[is.na(mydata$Alley)] <- 0
mydata$Has.Alley[!is.na(mydata$Alley)] <- 1
#
# # Pool
mydata$Has.Pool[is.na(mydata$Pool.QC)] <- 0
mydata$Has.Pool[!is.na(mydata$Pool.QC)] <- 1

# Basement
mydata$Has.Basement[is.na(mydata$Bsmt.Qual) & is.na(mydata$Bsmt.Cond) & is.na(mydata$Bsmt.Exposure
mydata$Has.Basement[!is.na(mydata$Bsmt.Qual) | !is.na(mydata$Bsmt.Cond) | !is.na(mydata$Bsmt.Expos

# Garage
mydata$Has.Garage[is.na(mydata$Garage.Type) & is.na(mydata$Garage.Yr.Blt) & is.na(mydata$Garage.Fi
mydata$Has.Garage[!is.na(mydata$Garage.Type) | !is.na(mydata$Garage.Yr.Blt) | !is.na(mydata$Garage

# # Fence
mydata$Has.Fence[is.na(mydata$Fence)] <- 0
mydata$Has.Fence[!is.na(mydata$Fence)] <- 1
#
# # Misc feature
mydata$Has.Misc[is.na(mydata$Misc.Feature)] <- 0
mydata$Has.Misc[!is.na(mydata$Misc.Feature) | (mydata$Misc.Val > 0)] <- 1

# Fill zeros for Masonry Veneer Area
mydata$Mas.Vnr.Area[is.na(mydata$Mas.Vnr.Area)] <- 0

remove_cols <- c("Order", "PID", "Sale.Condition", "Alley", "Pool.Area", "Pool.QC", "Bsmt.Qual", "

print("Removed Variables: ")
```

```
## [1] "Removed Variables: "
```

```
mydata <- mydata[, !(names(mydata) %in% remove_cols)]
```

2.3 Further Variable Consolidation and Simplification

Since the actual feature of whether a property has a deck or porch might be more relevant than what its square footage is, the square footage variables for the various types of decks and porches (such as open porch, screen porch, wood deck, etc.) were consolidated and a single dummy variable "Has.Deck.Porch" was created to indicate whether or not the property had a deck or porch. While the number of full baths could have an influence directly on the number of people who could shower at the same time, half-baths can be viewed as representing additional convenience. The counts of full baths and half baths were consolidated into a single numeric variable for a total count of the number of bathrooms. Running the regression or other supervised learning models with and without this consolidation did not produce any significantly different predictions and the model fit was approximately the same. (data not shown)

A number of variables were dropped: MS SubClass, because it had redundant (though more detailed) categories as House Style, "Proximity to various conditions" and second "Exterior covering on the house" variables were dropped under the assumption that the more important one is captured by the first variable. Furthermore, buyers would typically want to know about any undesirable characteristics for a home such as

irregular lots for lot shape variable, heavily or moderately sloped lots for lot slope variable, and proximity to various conditions such as railroads while valuing the property. So we combined such negative attributes of a property into single variable so these can be treated consistently in the model. Instead of using many variables that detailed square footage of non-living areas, such as 1st floor deck, 2nd floor balcony etc., it is more reasonable that the overall square footage of the home would be a critical factor in the buyers decision so a single living area square footage variable was created to consolidate all of these non-living areas. In addition for a few variables where almost all of the observations were in a single category such as Utilities (where most of the properties had public utilities), these were dropped as they cannot explain the variation is SalePrice. Sale Type was also dropped because that variable is more about how the sale is financed as opposed to the characteristics of the property. Some variables such as whether the home is on a paved street has wildly unbalanced number of observations in the dataset between the the different categorical values of the dummy variable and were dropped as they cannot explain the observed variation is sale price adequately for this dataset. In general when a level of a categorical variable had less than 2% of observations, these were combined to better explain the observed variability of the response variable.

All variables that represent dates or years such as YearBuilt are in reality categorical variables, so these were transformed into a numeric variables by converting them into ages or intervals.As an example the "Year.Remod.Add" the year when a remodel was done, and "Year.Built" variables were consolidated to age $Year.Age = Yr.Sold - Year.Built$ and age since the last remodel $Year.Since.Remodel = Yr.Sold - Year.Remod.Add$

```
# Deck/Porch
mydata$Has.Deck.Porch[mydata$Wood.Deck.SF==0 & mydata$Open.Porch.SF==0 & mydata$Enclosed.Porch==0
mydata$Has.Deck.Porch[mydata$Wood.Deck.SF!=0 | mydata$Open.Porch.SF!=0 | mydata$Enclosed.Porch!=0

# Baths
mydata$Bath <- mydata$Full.Bath + mydata$Half.Bath*0.5

# Street
mydata$Paved.Street[mydata$Street=="Grvl"] <- 0
mydata$Paved.Street[mydata$Street=="Pave"] <- 1

# Central Air
mydata$Cent.Air[mydata$Central.Air=="N"] <- 0
mydata$Cent.Air[mydata$Central.Air=="Y"] <- 1

# Lot Shape: combine all irregular lot types IR1, IR2 and IR3 into one level
index <- which(mydata$Lot.Shape == "IR1" | mydata$Lot.Shape == "IR2" | mydata$Lot.Shape == "IR3")
mydata[index, 'Lot.Shape'] <- "IR"

# Lot Config: combine FR2 and FR3 into one level
index <- which(mydata$Lot.Config == "FR2"  | mydata$Lot.Config == "FR3")
mydata[index, 'Lot.Config'] <- "FR"

# Bldg Type: combine Townhouse types into one level and Duples/multi-family into single level
index <- which(mydata$Bldg.Type == "Twnhs"  | mydata$Bldg.Type == "TwnhsE")
mydata[index, 'Bldg.Type'] <- "TwnhsComb"
index <- which(mydata$Bldg.Type == "2fmCon"  | mydata$Bldg.Type == "Duplex")
mydata[index, 'Bldg.Type'] <- "Duplex2fmCon"

# Land Slope: combine moderate and severe land slope into 1 level not gentle
```

```r
index <- which(mydata$Land.Slope != "Gtl")
mydata[index, "Land.Slope"] <- "NotGtl"

# Land Contour: combine sloped, banked, and hilly terrain into 1 level Steep
index <- which(mydata$Land.Contour != "Lvl")
mydata[index, "Land.Contour"] <- "Steep"

#Proximity Conditions: Combine all undesirable proximity features of a property to 1 level called
index <- which(mydata$Condition.1 != "Norm")
mydata[index, "Condition.1"] <- "Prox"

# House Style: combine 1.5 story Unfinished and finished into "1.5Story" and 2.5 story finished/u
index1 <- which(mydata$House.Style %in% c("1.5Fin", "1.5Unf"))
mydata[index1, 'House.Style'] <- "1.5Story"

index2 <- which(mydata$House.Style %in% c("2.5Fin", "2.5Unf"))
mydata[index2, 'House.Style'] <- "2.5Story"

#Electrical: Most of the homes have circuit breakers and less than 10% had other electrical types
index <- which(mydata$Electrical != "SBrkr")
mydata[index, "Electrical"] <- "FuseMix"

#Exterior: 4 categories have a single observations (Asphalt Shinges, ImitationStucco, Cinder bloc
index1 <- which(mydata$Exterior.1st == "AsphShn" | mydata$Exterior.1st == "WdShing" | mydata$Exte
mydata[index1, 'Exterior.1st'] <- "Shingle"

index2 <- which(mydata$Exterior.1st == "BrkComm" | mydata$Exterior.1st == "BrkFace")
mydata[index2, 'Exterior.1st'] <- "Brick"

index3 <- which(mydata$Exterior.1st == "CemntBd" | mydata$Exterior.1st == "CBlock" | mydata$Exteri
mydata[index3, 'Exterior.1st'] <- "CemntBd"

index4 <- which(mydata$Exterior.1st == "Stucco" | mydata$Exterior.1st == "ImStucc")
mydata[index4, 'Exterior.1st'] <- "Stucco"

index5 <- which(mydata$Exterior.1st == "Wd Sdng" | mydata$Exterior.1st == "VinylSd" | mydata$Exte
mydata[index5, 'Exterior.1st'] <- "Shingle"

#Foundation: 3 categories (Wood, Slab and Stone) have very few data points and account for less t
index <- which(mydata$Foundation == "Slab" | mydata$Foundation == "Stone" | mydata$Foundation == "
mydata[index, 'Foundation'] <- "Other"

# Remove handled variables
remove_cols <- c("Wood.Deck.SF", "Open.Porch.SF", "Enclosed.Porch", "X3Ssn.Porch", "Screen.Porch",
print("Variables consolidated or removed from original dataset:")
```

```
## [1] "Variables consolidated or removed from original dataset:"
```

```r
print(remove_cols)
```

```
##  [1] "Wood.Deck.SF"    "Open.Porch.SF"   "Enclosed.Porch"  "X3Ssn.Porch"
```

```
##  [5] "Screen.Porch"    "Full.Bath"       "Half.Bath"       "MS.SubClass"
##  [9] "Condition.2"     "Exterior.2nd"    "X1st.Flr.SF"     "X2nd.Flr.SF"
## [13] "Low.Qual.Fin.SF" "Utilities"       "Sale.Type"       "Central.Air"
## [17] "Street"          "Roof.Matl"       "Roof.Style"      "Heating"
## [21] "Mas.Vnr.Type"
```

```
mydata <- mydata[, !(names(mydata) %in% remove_cols)]
```

```
mydata$Year.Age <- mydata$Yr.Sold-mydata$Year.Built
mydata$Year.Since.Remodel <- mydata$Yr.Sold-mydata$Year.Remod.Add

remove_cols <- c("Year.Remod.Add", "Year.Built", "Yr.Sold", "Mo.Sold")
print("Variables converted to intervals or age:")
```

```
## [1] "Variables converted to intervals or age:"
```

```
print(remove_cols)
```

```
## [1] "Year.Remod.Add" "Year.Built"     "Yr.Sold"        "Mo.Sold"
```

```
mydata <- mydata[, !(names(mydata) %in% remove_cols)]
```

```
# Convert the remaining categorial variables to factors
df <- mydata

cols.to.factor <- sapply(df, function(col) ((class(col) %in% c("character")) | (class(col) %in% c

df[cols.to.factor] <- lapply(df[cols.to.factor], factor)

mydata <- df
```

2.4 Categorical Variables

This dataset has a large number of ordinal categorical variables that pose a challenge of whether to treat them as nominal or numeric. On the one hand, treating the ordinal variables as nominal avoids making the assumption that the distances between adjacent categories are equal. However, in doing so, we would also lose the information in the ordering. Many of the ordinal variables in this dataset relate to the quality or condition of different aspects of the property. Since it is important to preserve this information in case the variables have an influence on the SalePrice, these variables were converted into numeric values. However, in a number of qualitative ordinal variables such as kitchen quality, heating quality, exterior quality which were rated Excellent though Poor on a 5 number scale,there were not enough observations in a particular category and we combined them to better reflect the extent to which a home buyer would want to know the quality level whether it is below average by combining poor and fair. Some variables like roof types, roof material, and heating were mostly of one type with less than 1% of observations outside of the dominant category, so these variables cannot explain price variation, so I dropped them. Some of the ordinal condition and quality variables were already numeric, so we converted the qualitative condition and quality values to align with those scales. After this consolidation, re-scaling, and remapping, we created dummy variables for the remaining nominal categorical variables.

```
ordinal_vars_with_5_quality_levels = c("Exter.Qual", "Exter.Cond", "Heating.QC", "Kitchen.Qual")
po_ex_combined_quality_levels = c("PoFa", "TA", "Gd", "Ex")

df <- mydata
```

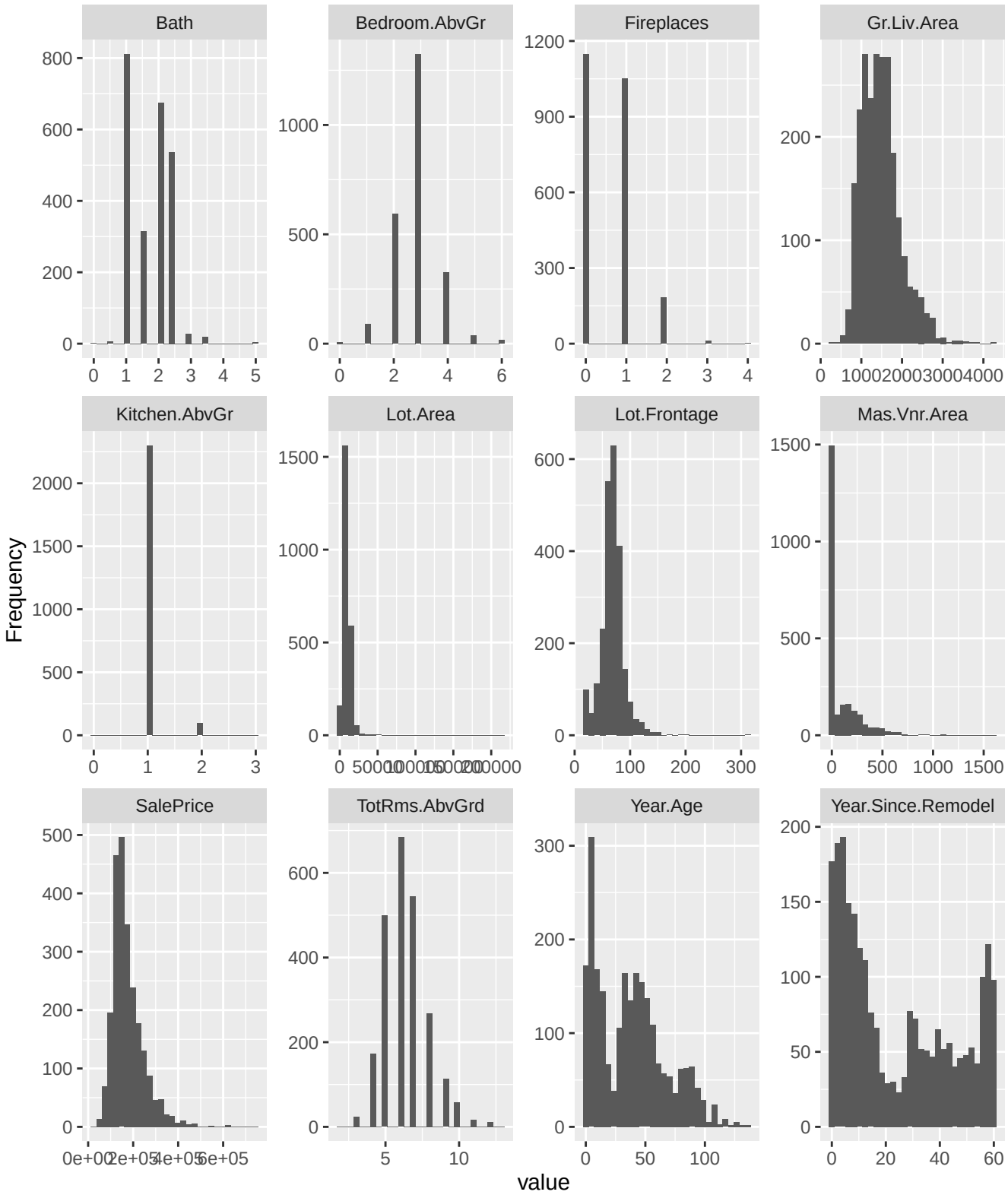
```
# Combine below average quality
for(var in ordinal_vars_with_5_quality_levels){
  colindex <- which(colnames(df) == var)
  index <- which(df[[var]] %in% c("Po", "Fa"))
  levels(df[[var]]) <- c(levels(df[[var]]), "PoFa")
  df[index, colindex] <- "PoFa"
}

df[ordinal_vars_with_5_quality_levels] <- lapply(df[ordinal_vars_with_5_quality_levels] , functior


df$Functional.Type[df$Functional != "Typ"] <- "DedDamgs"
df$Functional.Type[df$Functional == "Typ"] <- "Typical"

df$Functional.Type = factor(df$Functional.Type, order=TRUE, levels=c("DedDamgs", "Typical"))

# Overall Condition
df$Overall.Cond.Type[df$Overall.Cond <= 4] <- "PoorFair"
df$Overall.Cond.Type[df$Overall.Cond %in% c(5,6)] <- "AvgMed"
df$Overall.Cond.Type[df$Overall.Cond > 6] <- "Superior"

# Overall Quality
df$Overall.Qual.Type[df$Overall.Qual <= 4] <- "PoorFair"
df$Overall.Qual.Type[df$Overall.Qual %in% c(5,6)] <- "AvgMed"
df$Overall.Qual.Type[df$Overall.Qual > 6] <- "Superior"

df$Overall.Qual.Type = factor(df$Overall.Qual.Type, order=TRUE, levels=c("PoorFair", "AvgMed", "Su
df$Overall.Cond.Type = factor(df$Overall.Cond.Type, order=TRUE, levels=c("PoorFair", "AvgMed", "Su

mydata <- df

remove_cols <- c("Overall.Cond", "Overall.Qual", "Functional")
mydata <- mydata[, !(names(mydata) %in% remove_cols)]

housing_data_frame <- mydata
print("After data cleaning final cleansed dataset:")
print(dim(housing_data_frame))
print(head(housing_data_frame))
```

After variable exploration, consolidation, simplification, and removal, we end up with 41 potential predictor variables describing the sales of 2393 properties.

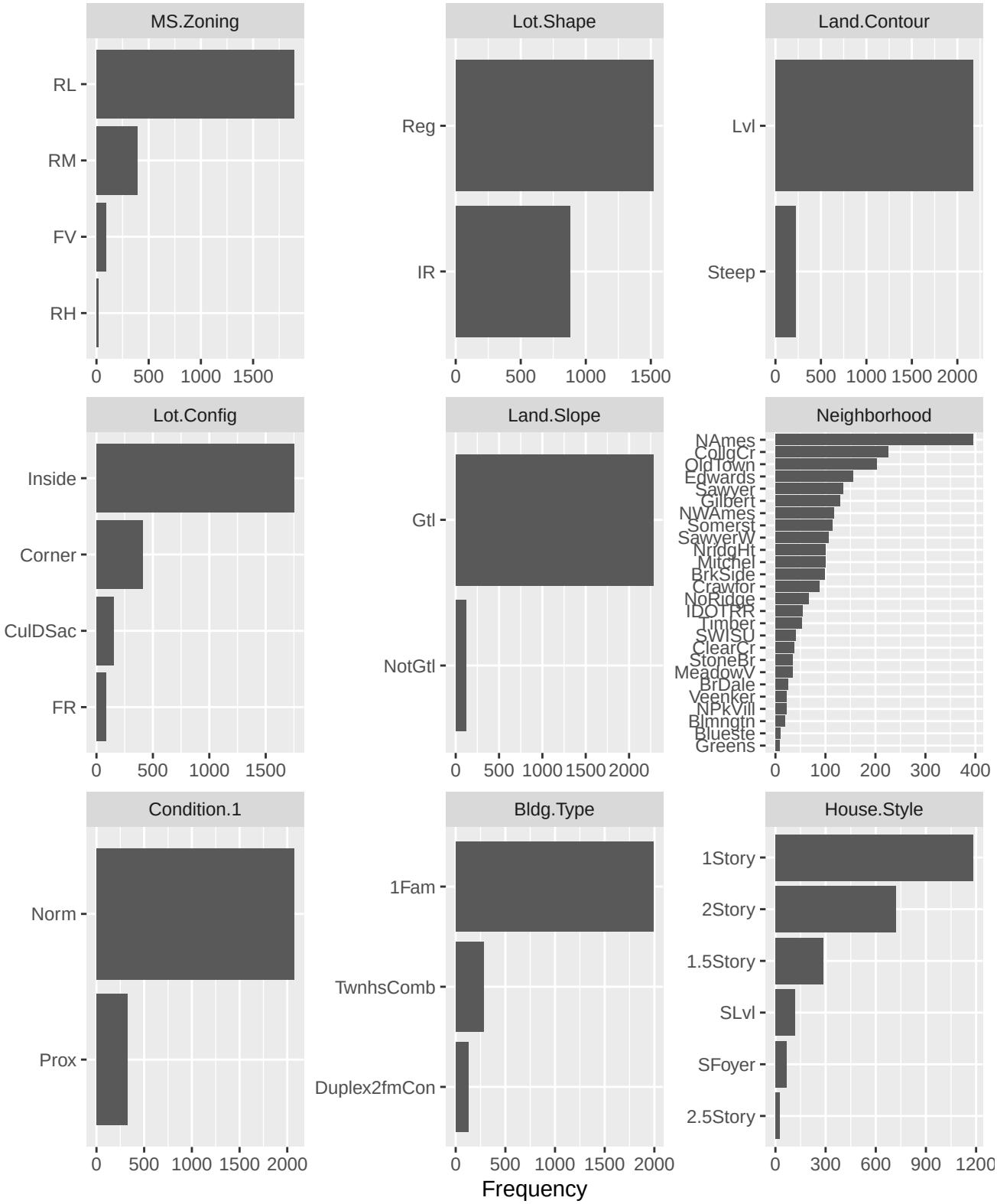
2.5: Data Exploration

We created various charts (histograms, box plots, scatter and bar plots) to explore univariate statistics of categorical and numeric variables.

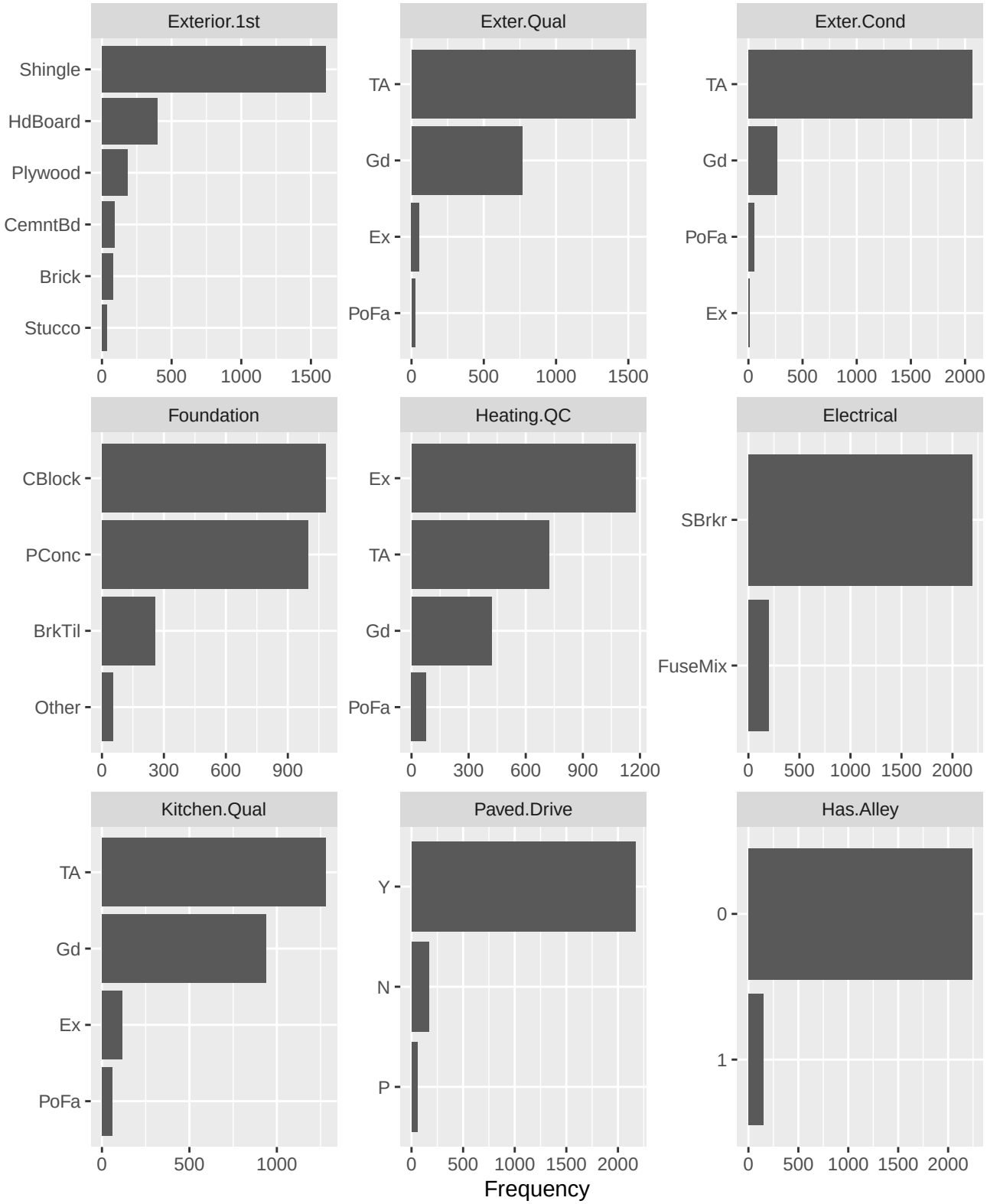
```
library(DataExplorer)

plot_histogram(housing_data_frame)
```



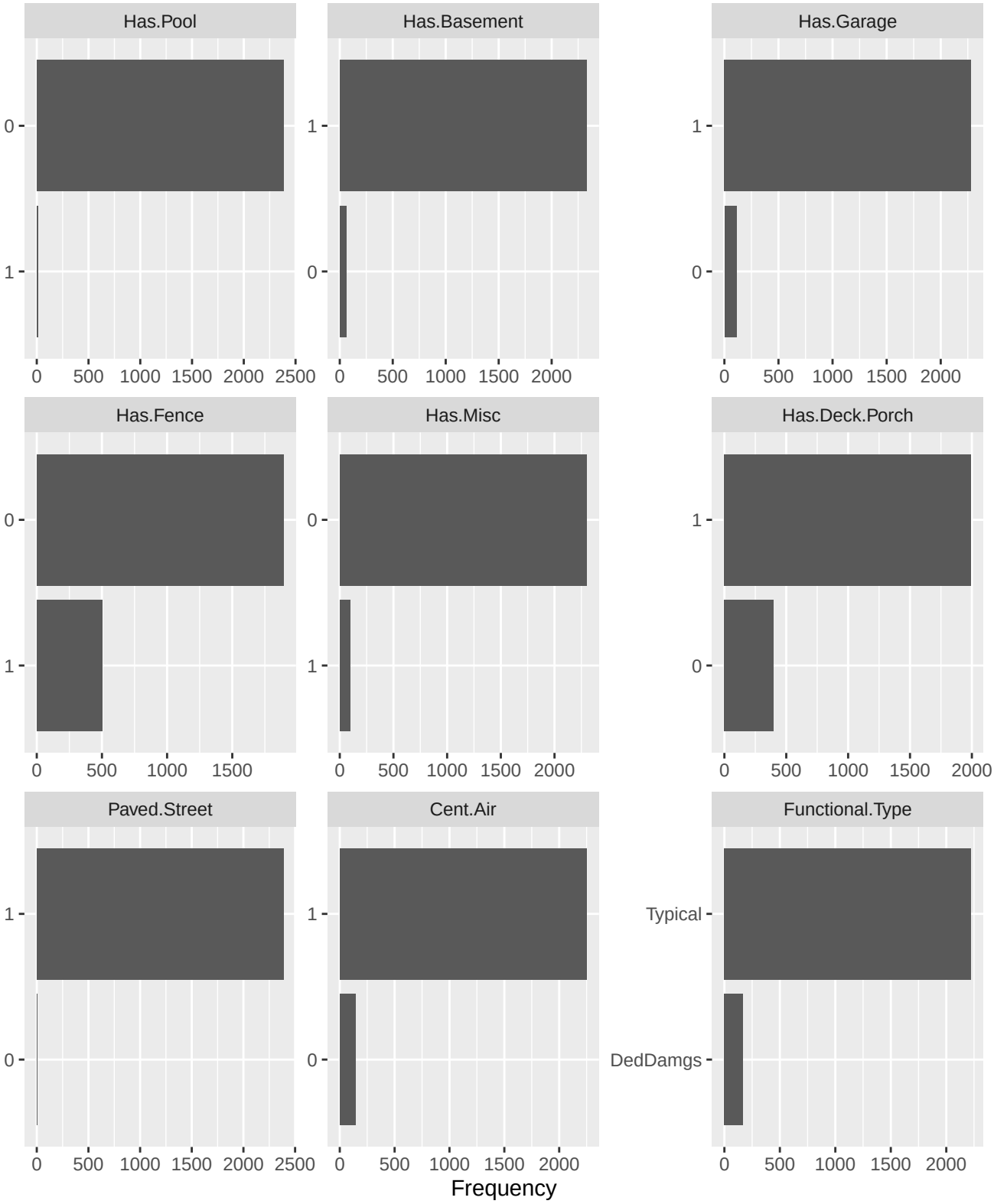

```
plot_bar(housing_data_frame)
```


MS.Zoning
RL
RM
FV
RH
0 500 1000 1500
Lot.Shape
Reg
IR
0 500 1000 1500
Land.Contour
Lvl
Steep
0 500 1000 1500 2000
Lot.Config
Inside
Corner
CulDSac
FR
0 500 1000 1500
Land.Slope
Gtl
NotGtl
0 500 1000 1500 2000
Neighborhood
NAmes
CollgCr
OldTown
Edwards
Sawyer
Gilbert
NWAmes
Somerst
SawyerW
NridgHt
Mitchel
BrkSide
Crawfor
NoRidge
IDOTRR
Timber
SWISU
ClearCr
StoneBr
MeadowV
BrDale
Veenker
NPkVill
Blmngtn
Blueste
Greens
0 100 200 300 400
Condition.1
Norm
Prox
0 500 1000 1500 2000
Bldg.Type
1Fam
TwnhsComb
Duplex2fmCon
0 500 1000 1500 2000
House.Style
1Story
2Story
1.5Story
SLvl
SFoyer
2.5Story
0 300 600 900 1200
Frequency

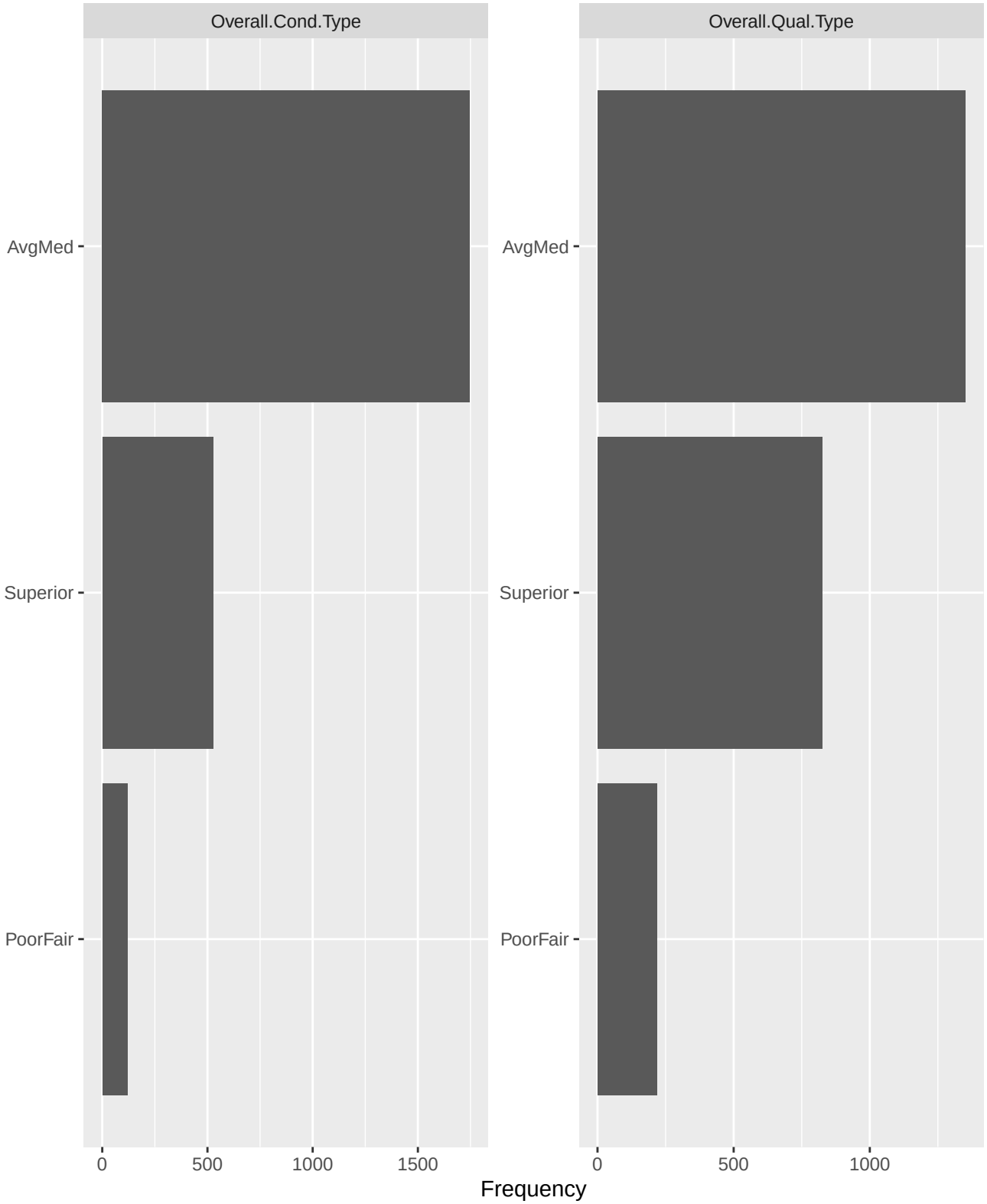
Page 1


Exterior.1st
Shingle
HdBoard
Plywood
CemntBd
Brick
Stucco
0
500
1000
1500
Exter.Qual
TA
Gd
Ex
PoFa
0
500
1000
1500
Exter.Cond
TA
Gd
PoFa
Ex
0
500
1000
1500
2000
Foundation
CBlock
PConc
BrkTil
Other
0
300
600
900
Heating.QC
Ex
TA
Gd
PoFa
0
300
600
900
1200
Electrical
SBrkr
FuseMix
0
500
1000
1500
2000
Kitchen.Qual
TA
Gd
Ex
PoFa
0
500
1000
Paved.Drive
Y
N
P
0
500
1000
1500
2000
Has.Alley
0
1
0
500
1000
1500
2000
Frequency

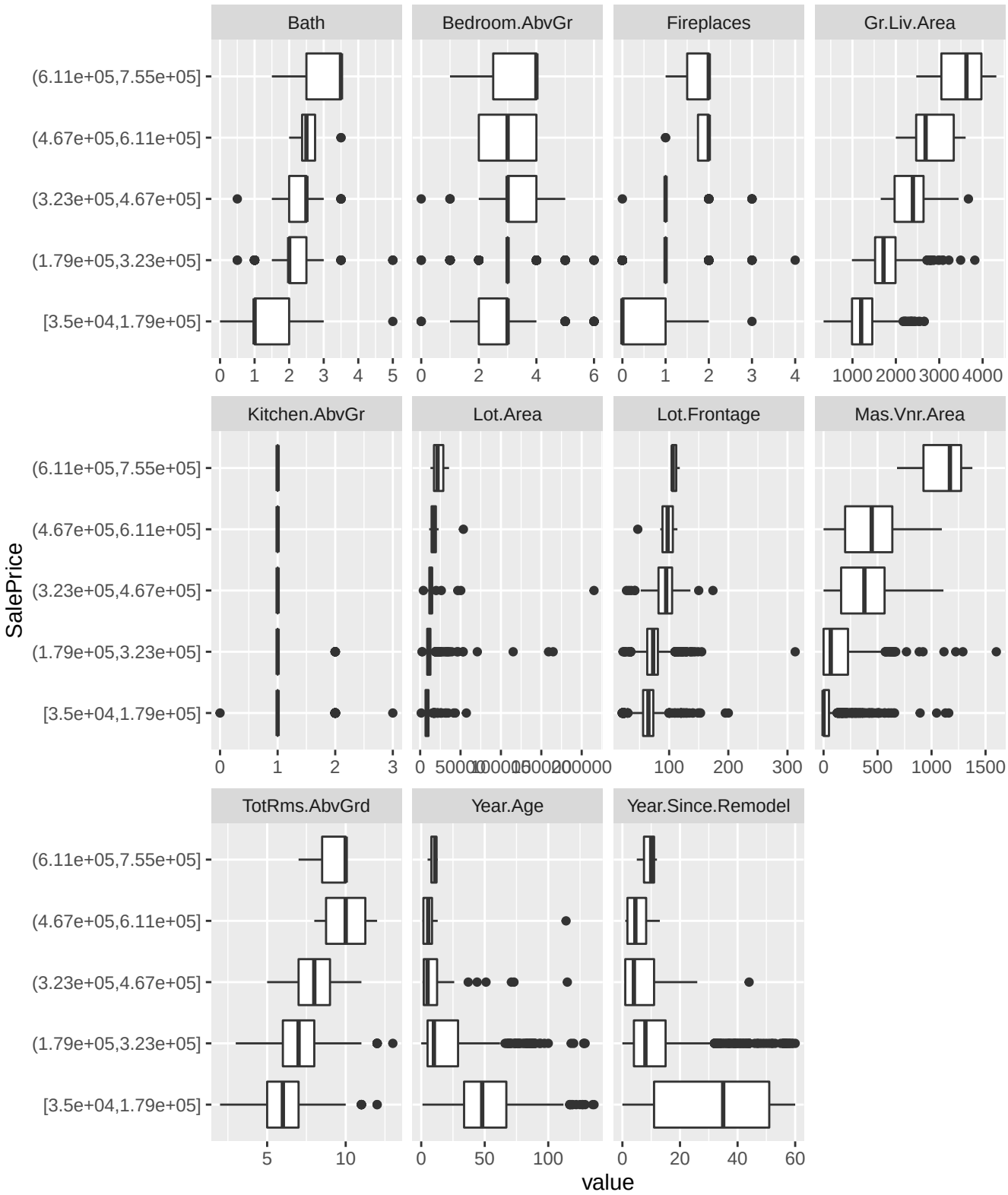
Page 2


Has.Pool
0
1
0 500 1000 1500 2000 2500
Has.Basement
1
0
0 500 1000 1500 2000
Has.Garage
1
0
0 500 1000 1500 2000
Has.Fence
0
1
0 500 1000 1500
Has.Misc
0
1
0 500 1000 1500 2000
Has.Deck.Porch
1
0
0 500 1000 1500 2000
Paved.Street
1
0
0 500 1000 1500 2000 2500
Cent.Air
1
0
0 500 1000 1500 2000
Functional.Type
Typical
DedDamgs
0 500 1000 1500 2000
Frequency

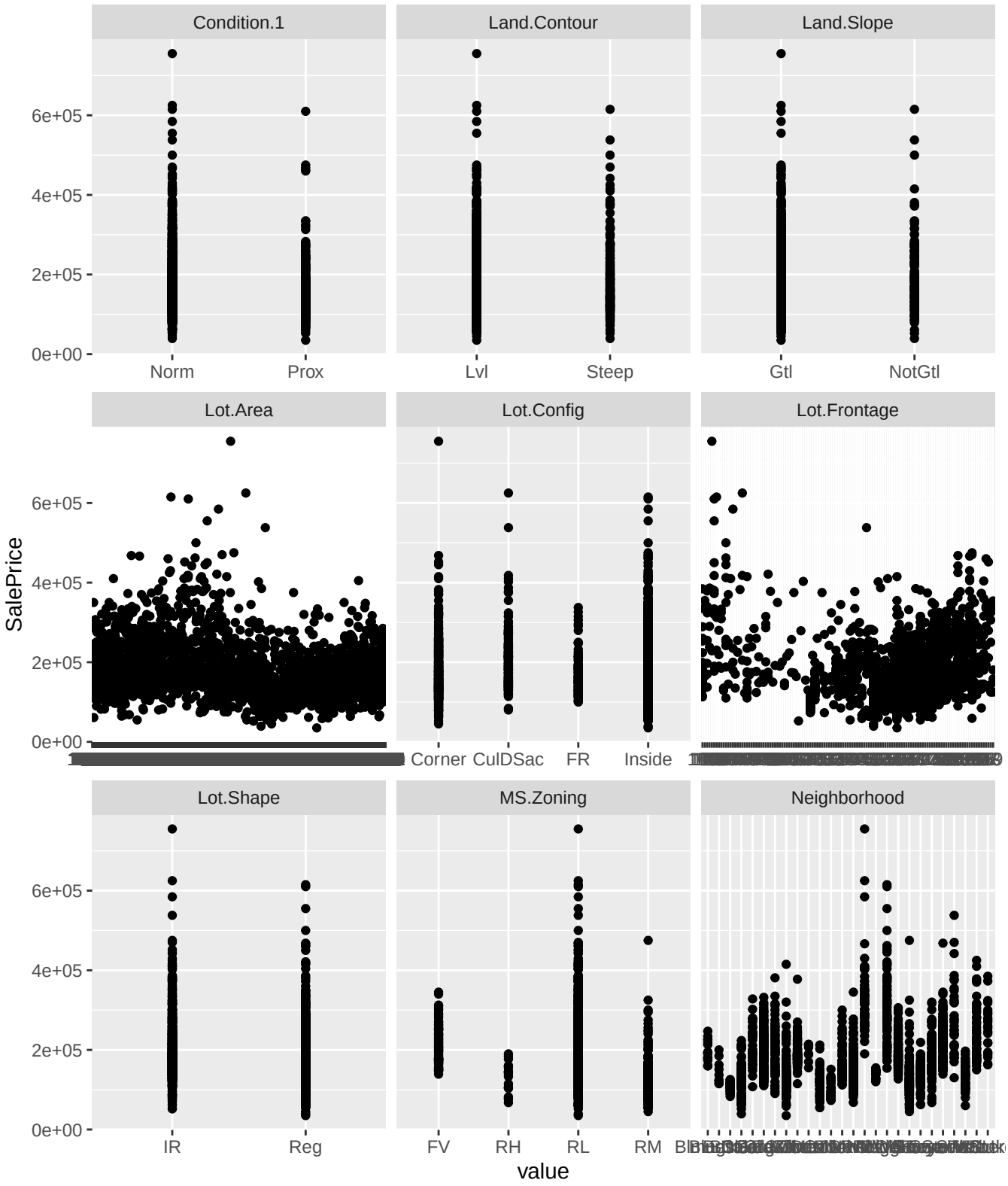
Page 3

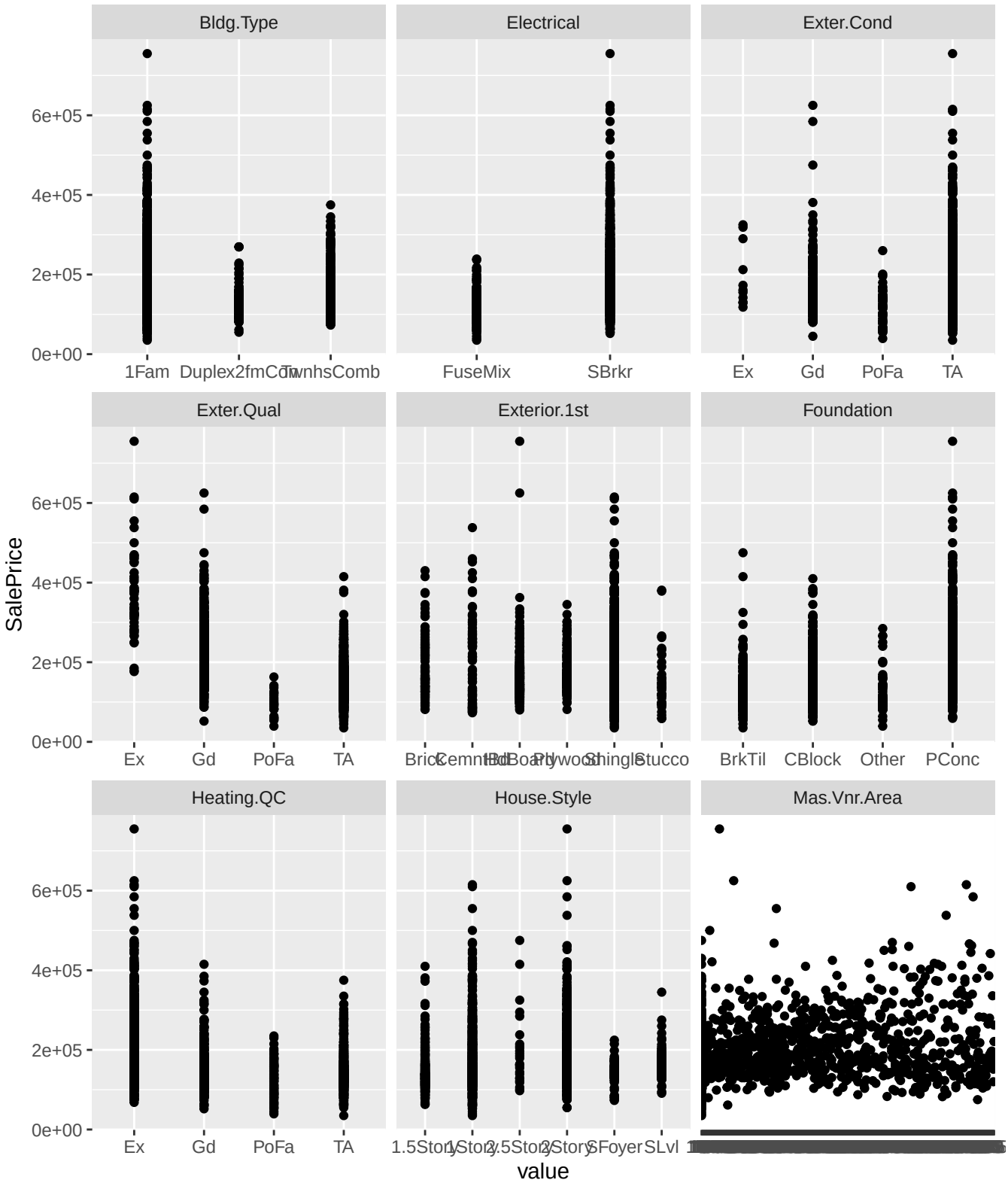


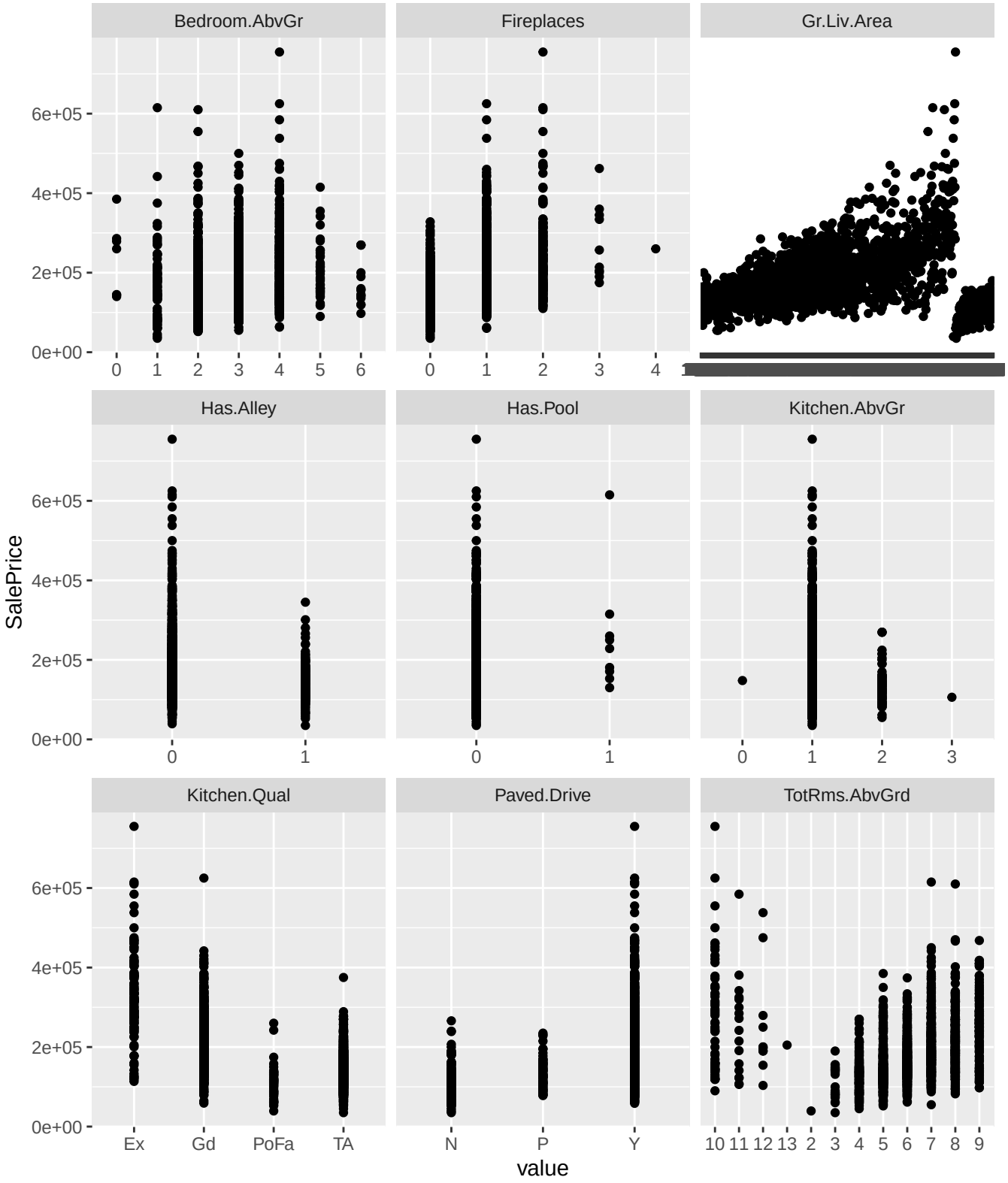

```
plot_boxplot(housing_data_frame, by = "SalePrice")
```



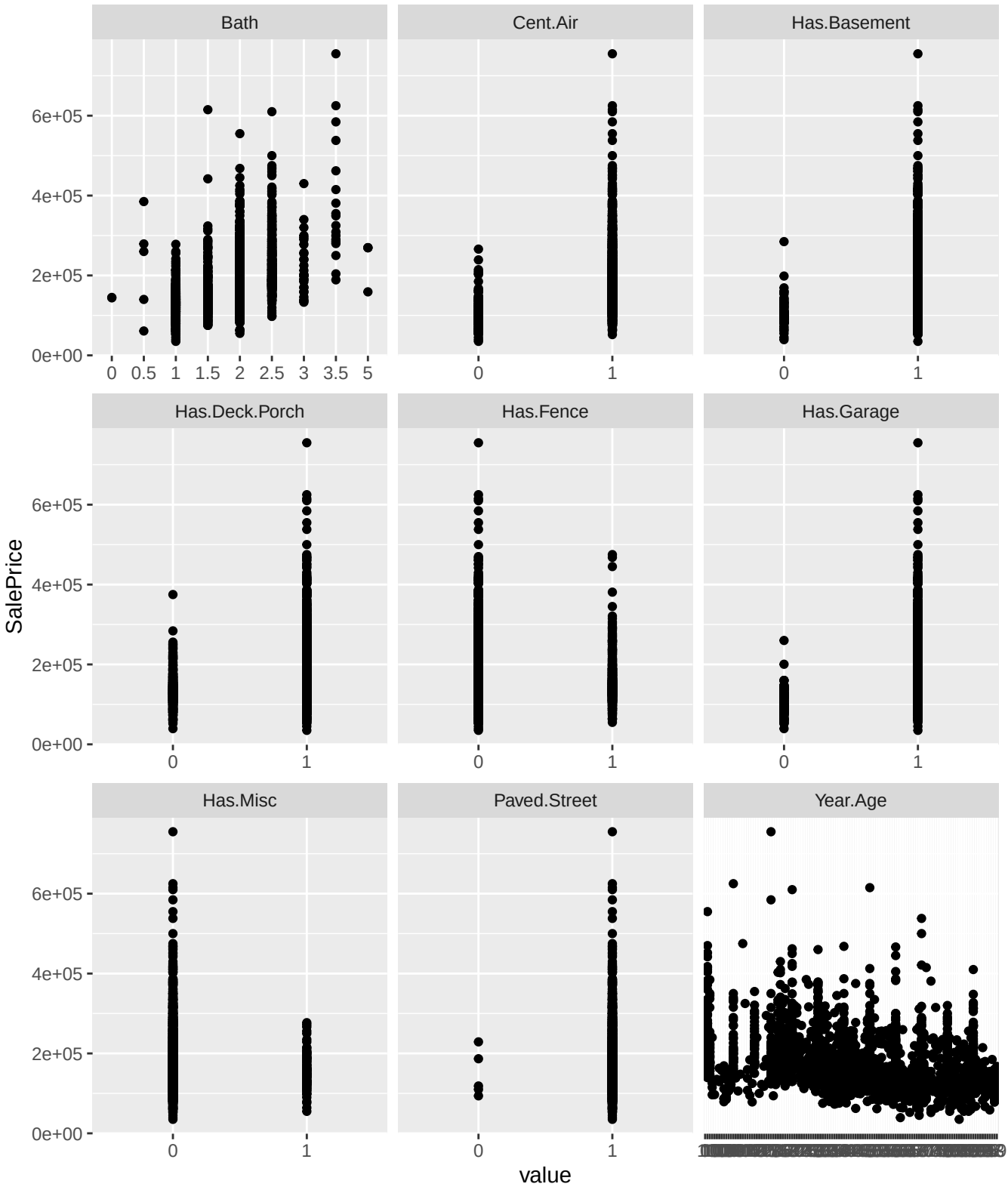

```
plot_scatterplot(housing_data_frame, by = "SalePrice")
```

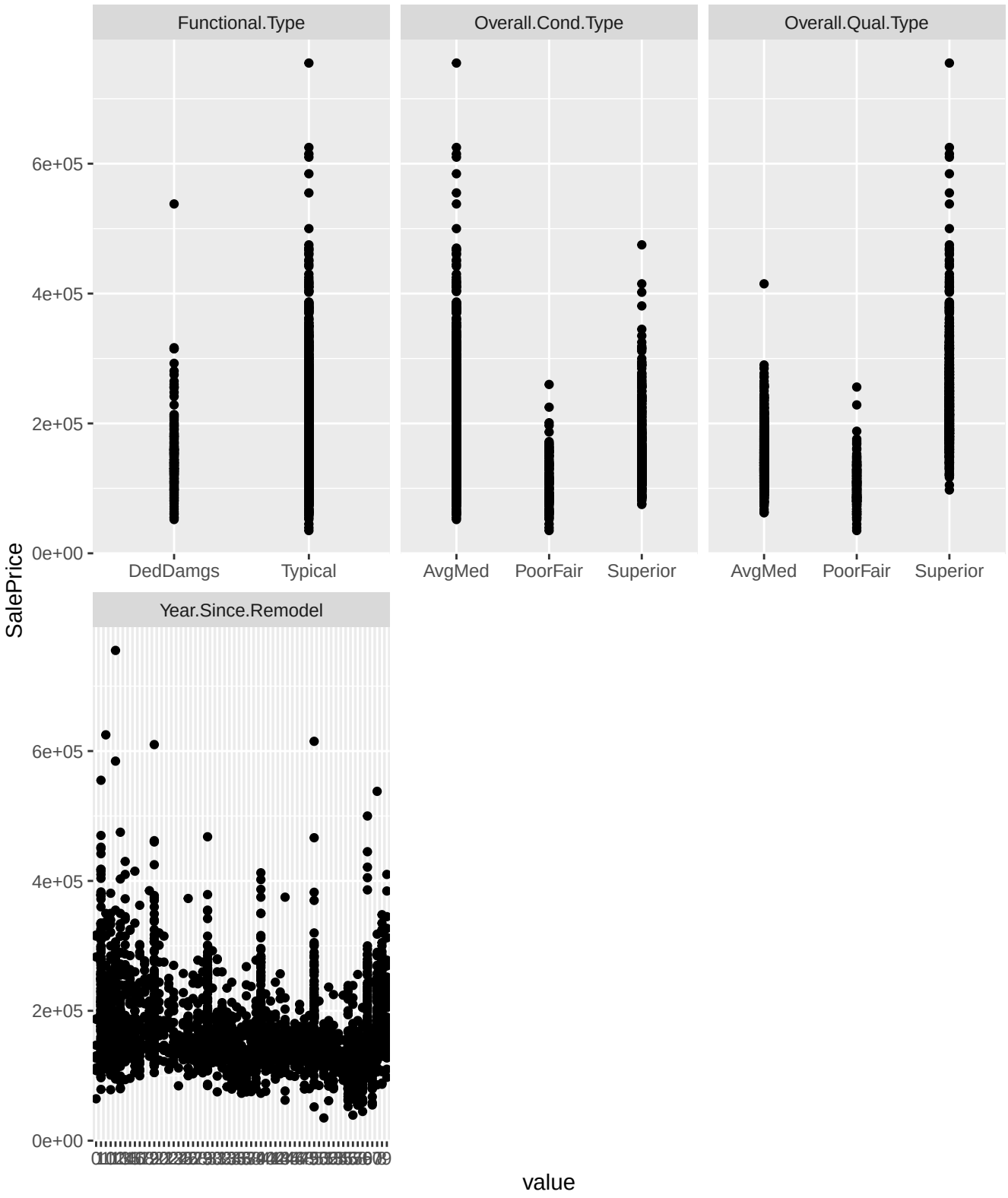

Condition.1
Land.Contour
Land.Slope
Lot.Area
Lot.Config
Lot.Frontage
Lot.Shape
MS.Zoning
Neighborhood
SalePrice
0e+00
2e+05
4e+05
6e+05
Norm
Prox
Lvl
Steep
Gtl
NotGtl
Corner
CulDSac
FR
Inside
IR
Reg
FV
RH
RL
RM
value


Bldg.Type
Electrical
Exter.Cond
Exter.Qual
Exterior.1st
Foundation
Heating.QC
House.Style
Mas.Vnr.Area
SalePrice
value
0e+00
2e+05
4e+05
6e+05
1Fam
Duplex
2fmCon
TwnhsComb
FuseMix
SBrkr
Ex
Gd
PoFa
TA
BrkTil
CBlock
Other
PConc
1.5Story
2Story
SFoyer
SLvl


Bedroom.AbvGr
Fireplaces
Gr.Liv.Area
Has.Alley
Has.Pool
Kitchen.AbvGr
Kitchen.Qual
Paved.Drive
TotRms.AbvGrd
SalePrice
value
0e+00
2e+05
4e+05
6e+05
Ex
Gd
PoFa
TA
N
P
Y

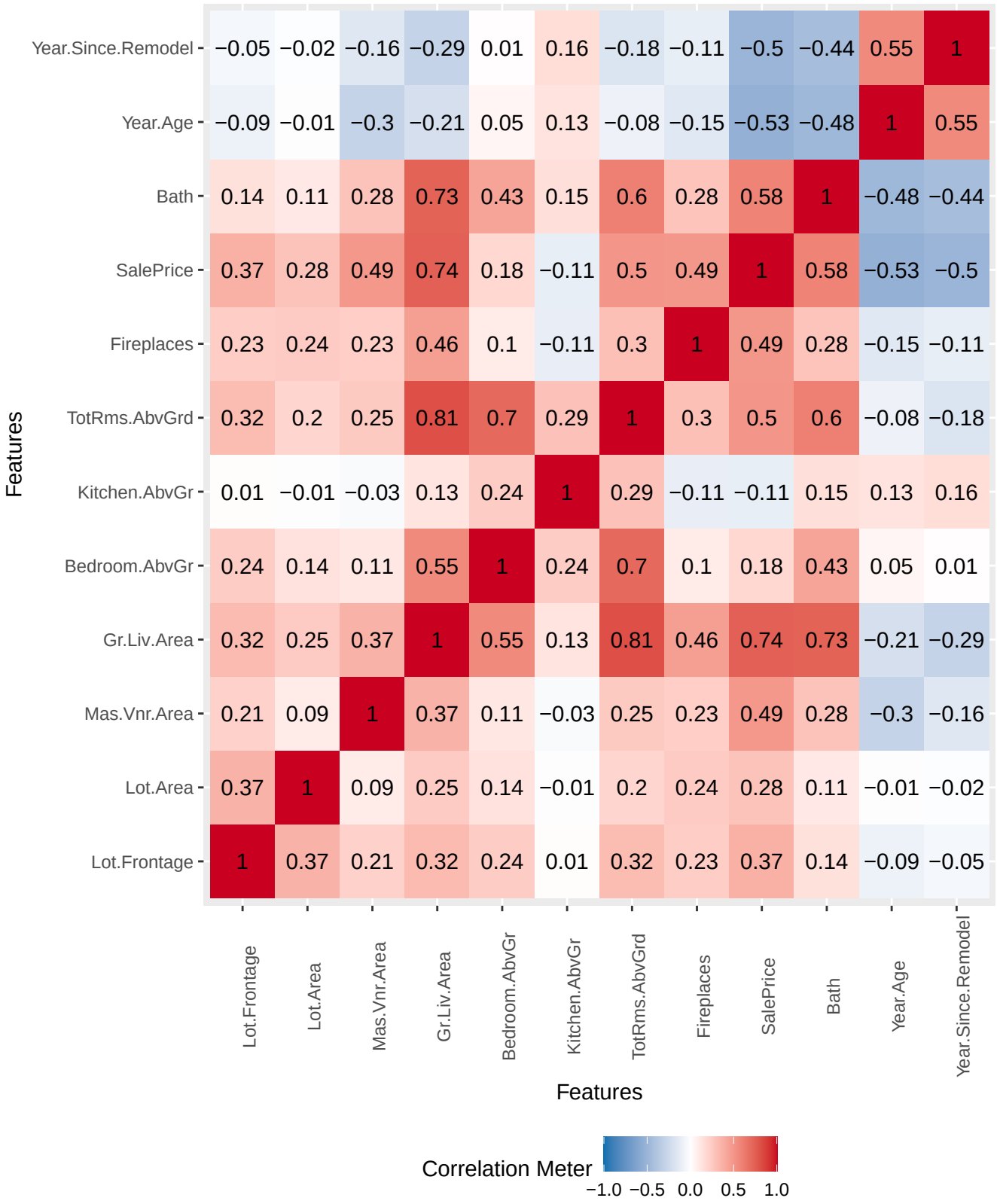
Page 3


Bath
Cent.Air
Has.Basement
Has.Deck.Porch
Has.Fence
Has.Garage
Has.Misc
Paved.Street
Year.Age
SalePrice
value
0e+00
2e+05
4e+05
6e+05
0
0.5
1
1.5
2
2.5
3
3.5
5
0
1



In order to explore which variables are highly correlated to the SalePrice we ran a correlation plot and found that Total Rooms above ground, Total Baths, Gross Living Area and fireplace were the most correlated to the sale price. Additionally, SalePrice, Living Area, Total Rooms variables showed a skewed to the right distribution indicating we might need to transform these variables in the linear regression models.

```
plot_correlation(housing_data_frame, type = 'continuous','SalePrice')
```



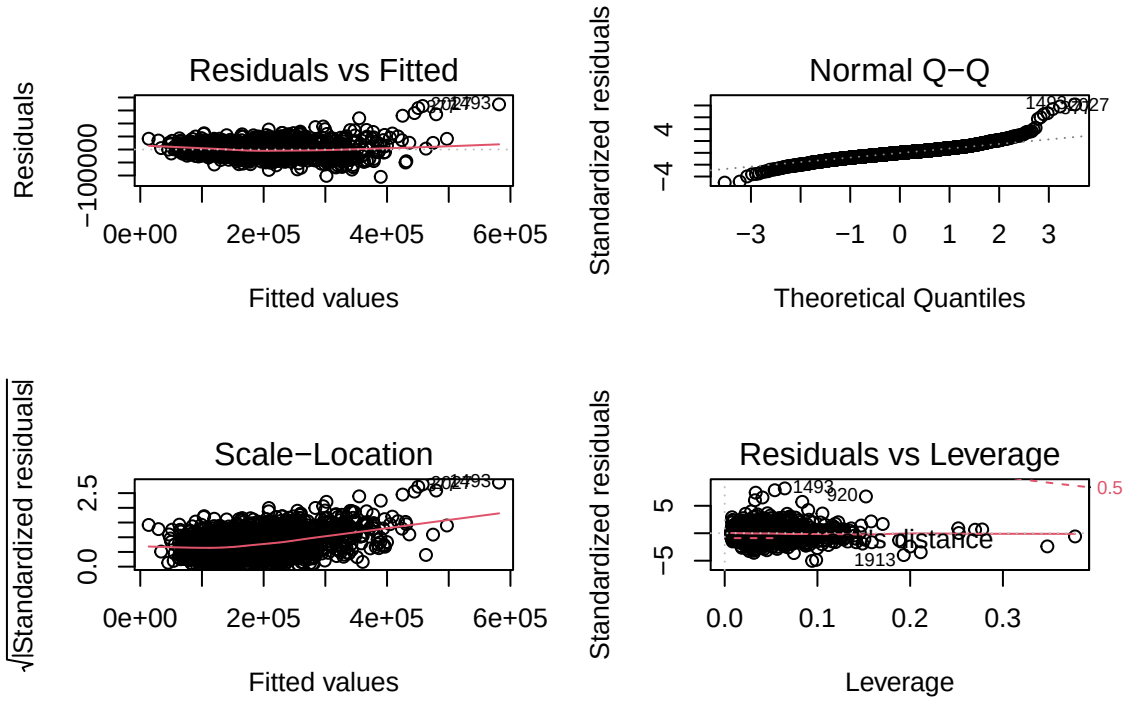
3. Modeling Approaches and Results

3.1 Model 1: Simple Linear Regression

We start by first fitting a simple linear regression to the full set of the untransformed variables, checking the diagnostic plots, and running a couple other diagnostic tests.

```
fit.full <- lm(SalePrice~., data=housing_data_frame)
```

```
par(mfrow=c(2,2))
plot(fit.full)
```



Right away, we notice that there is curvature in the plot of (standardized) residuals vs fitted values which indicates that the errors are not normally distributed. There is an additional plot called the Normal Q-Q Plot which also reveals that the errors are not normally distributed.

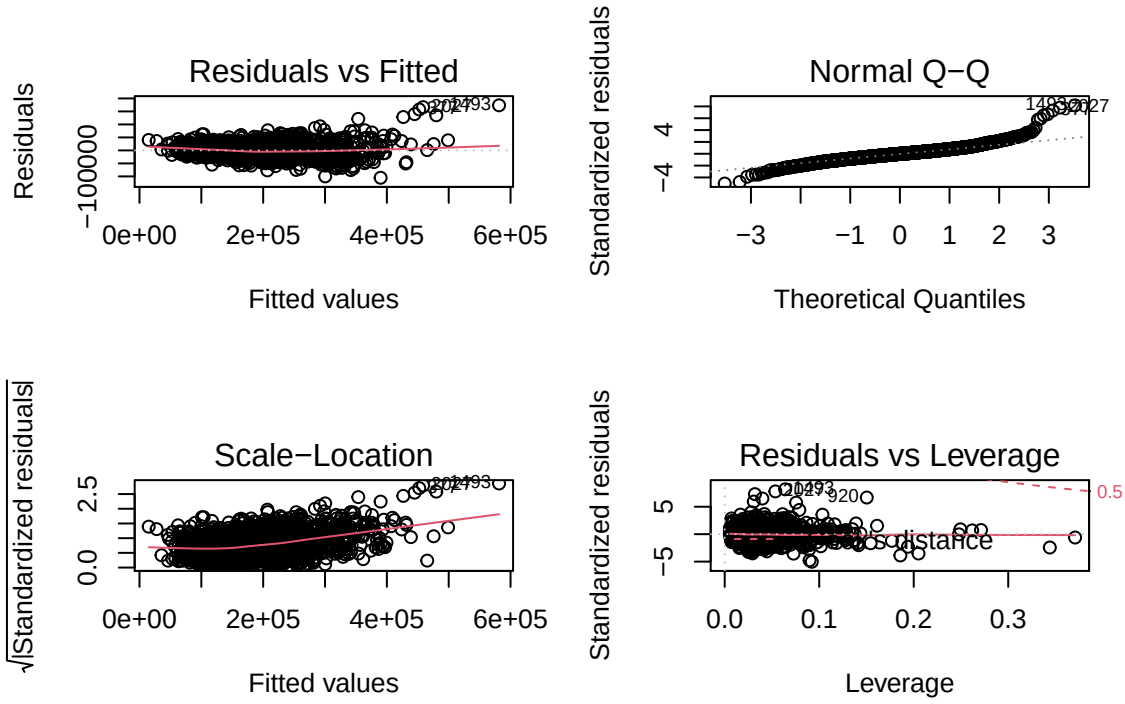
The R-squared suggests that 90.74% of the variation in SalePrice is explained by the variation of the variables in the model. The adjusted R-squared is very close to the unadjusted R-squared, indicating that there are not too many extraneous variables in the model. But there are a number of variables that do not have statistically significant coefficients. We remove these variables in our second model. After removing the variables that did not show significance in the full model, we performed a partial F-test to determine if we could drop the variables without degrading the model fit. With a p-value of 0.5056, we failed to reject the hypothesis that the model fits were significantly different, so it was safe to drop these variables in further modeling. The variables that were dropped with this method were: "MS.Zoning", "Lot.Shape", "Land.Contour", "Electrical", "Paved.Drive", "Has.Alley", "Has.Fence", "Has.Misc","Has.Deck.Porch","Bath", and "Cent.Air".

```
fit.reduced <- lm(SalePrice~.-MS.Zoning-Lot.Shape-Land.Contour-Electrical-Paved.Drive-Has.Alley-Ha
anova(fit.full,fit.reduced)

remove_cols <- c("MS.Zoning","Lot.Shape","Land.Contour","Electrical","Paved.Drive","Has.Alley","Ha
housing_data_frame_reduced <- housing_data_frame[, !(names(housing_data_frame) %in% remove_cols)]
```

In the diagnostic plots, we still observed a small curvature in the residuals plots and saw evidence that the errors were not normally distributed.

```
summary(fit.reduced)
```

```
par(mfrow=c(2,2))
plot(fit.reduced)
```



In our next model, we explored variable transformations to address the heteroskadasticity.

3.3 Reduced Model: Simple Linear Regression with Power-Law Variable Transformation

Next we used the variable transformations suggested by powerTransform, we fitted another model that drastically reduced the heteroskedasticity in the data. In this new model the living area was log transformed and the Lot Areas was raised ti the power 0.14

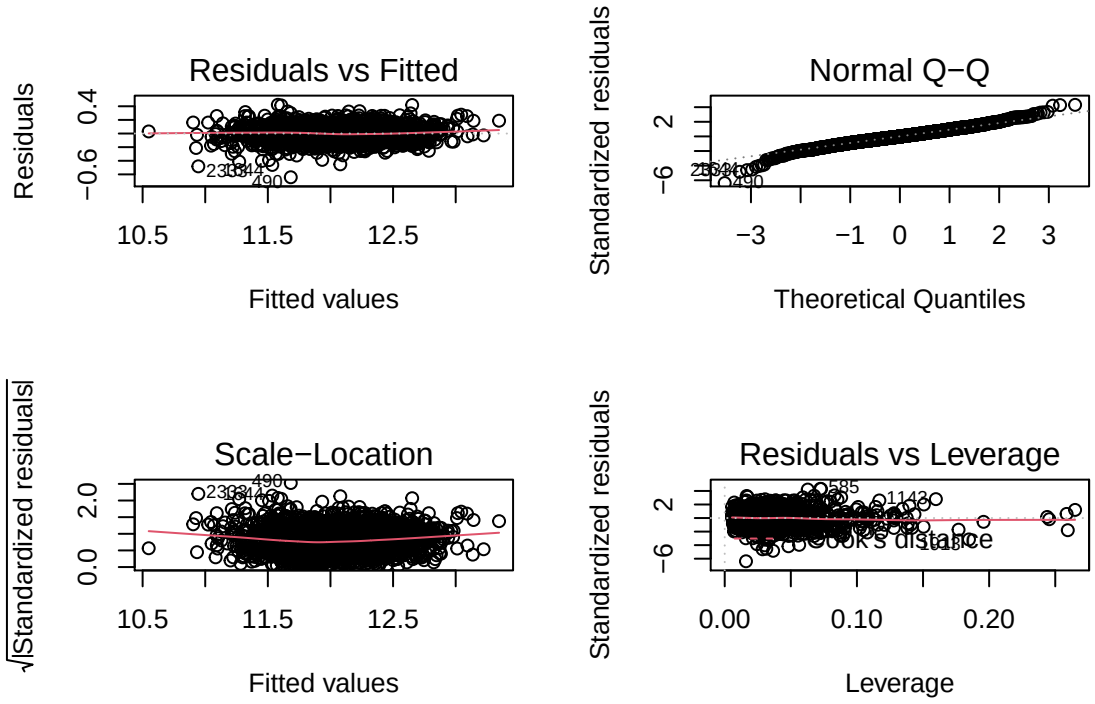
```
summary(powerTransform((with(mydata,cbind(Lot.Area,Gr.Liv.Area)))))
```

```
## bcPower Transformations to Multinormality
##              Est Power Rounded Pwr Wald Lwr Bnd Wald Upr Bnd
## Lot.Area       0.1389        0.14       0.1028       0.1750
## Gr.Liv.Area    0.0760        0.00      -0.0246       0.1766
##
## Likelihood ratio test that transformation parameters are equal to 0
##  (all log transformations)
##                                LRT df      pval
## LR test, lambda = (0 0) 55.41426  2 9.267e-13
##
## Likelihood ratio test that no transformations are needed
##                                LRT df       pval
## LR test, lambda = (1 1) 3393.779  2 < 2.22e-16
```

```
fit.transform <- lm(log(SalePrice)~.-Gr.Liv.Area+log(Gr.Liv.Area)-Lot.Area+I(Lot.Area^0.14), data=
```

The residuals plots flattened out noticeably, and the points in the Normal Q-Q plot are closer to the diagonal, indicating that the error distribution is close to normal. Running the test for Non-Constant Variance, the p-value increased from almost zero in the full model to 0.03889 in the model with transformed variables (data not shown)

```
par(mfrow=c(2,2))
plot(fit.transform)
```



3.4 Stepwise Linear Regression with AIC as stopping criterion

Using the reduced model with transformed variables as the starting point, we performed an automated stepwise regression that went both forward and backward to select or eliminate variables from the model. We used AIC as the criterion to fit this model. Compared to the third model (reduced with transformed variables), only Paved.Street was eliminated. The resulting standard error, R-squared, and adjusted R-squared were identical to those in the third model up to the thousandths place: residual standard error = 0.1015, R-squared = 0.925, adjusted R-squared = 0.923. While we can't compare the residual standard error with the first two models because the units are different, we noted that R-squared was higher than for both of the first two untransformed linear models (R-squared ~0.90).

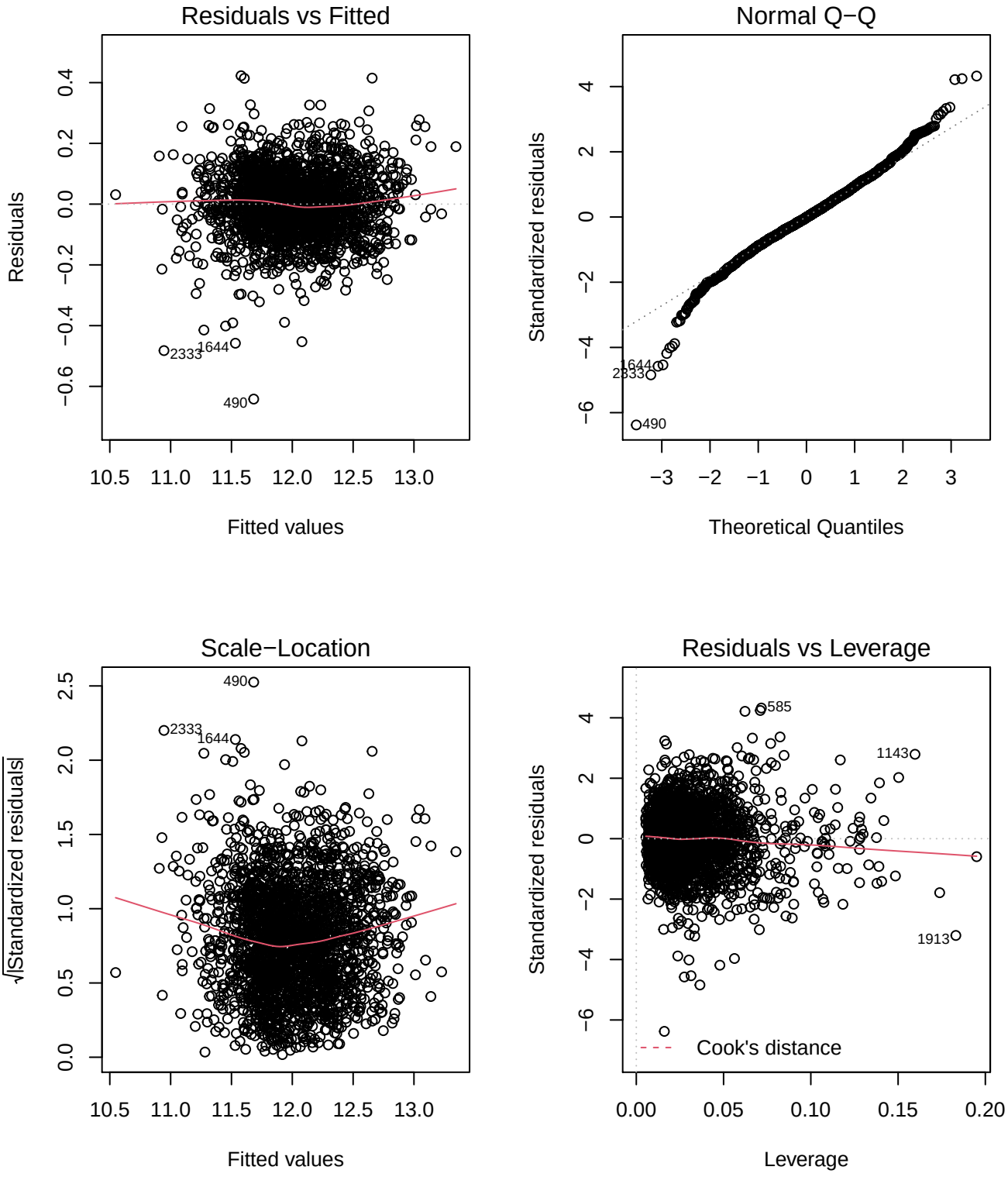
```
fit.stepAIC <- step(fit.transform,direction="both",trace=FALSE)
summary(fit.stepAIC)
```

```
##
## Call:
## lm(formula = log(SalePrice) ~ Lot.Frontage + Lot.Config + Land.Slope +
##     Neighborhood + Condition.1 + Bldg.Type + House.Style + Exterior.1st +
##     Mas.Vnr.Area + Exter.Qual + Exter.Cond + Foundation + Heating.QC +
##     Bedroom.AbvGr + Kitchen.AbvGr + Kitchen.Qual + Fireplaces +
##     Has.Pool + Has.Basement + Has.Garage + Year.Age + Year.Since.Remodel +
##     Functional.Type + Overall.Cond.Type + Overall.Qual.Type +
```

```
##     log(Gr.Liv.Area) + I(Lot.Area^0.14), data = housing_data_frame_reduced)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.64179 -0.06020 -0.00101  0.06243  0.42261
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)             7.095e+00  1.077e-01  65.877  < 2e-16 ***
## Lot.Frontage            2.425e-04  1.531e-04   1.584 0.113245
## Lot.ConfigCulDSac       1.329e-02  1.061e-02   1.252 0.210737
## Lot.ConfigFR           -3.135e-02  1.269e-02  -2.471 0.013554 *
## Lot.ConfigInside        5.866e-03  5.874e-03   0.999 0.318065
## Land.SlopeNotGtl        2.710e-02  1.056e-02   2.566 0.010351 *
## NeighborhoodBlueste     8.288e-02  4.122e-02   2.011 0.044451 *
## NeighborhoodBrDale     -5.204e-02  3.418e-02  -1.522 0.128026
## NeighborhoodBrkSide    -9.907e-03  2.968e-02  -0.334 0.738547
## NeighborhoodClearCr     3.705e-03  3.159e-02   0.117 0.906655
## NeighborhoodCollgCr    -2.793e-02  2.647e-02  -1.055 0.291431
## NeighborhoodCrawfor     5.761e-02  2.892e-02   1.992 0.046523 *
## NeighborhoodEdwards    -9.597e-02  2.780e-02  -3.452 0.000567 ***
## NeighborhoodGilbert    -7.723e-02  2.777e-02  -2.781 0.005457 **
## NeighborhoodGreens      1.993e-01  4.410e-02   4.518 6.54e-06 ***
## NeighborhoodIDOTRR     -6.581e-02  3.107e-02  -2.118 0.034271 *
## NeighborhoodMeadowV    -1.407e-01  3.550e-02  -3.963 7.62e-05 ***
## NeighborhoodMitchel    -3.460e-02  2.831e-02  -1.222 0.221728
## NeighborhoodNAmes      -4.295e-02  2.726e-02  -1.576 0.115251
## NeighborhoodNoRidge     9.276e-02  2.919e-02   3.178 0.001503 **
## NeighborhoodNPkVill     9.367e-02  3.474e-02   2.696 0.007059 **
## NeighborhoodNridgHt     5.555e-02  2.716e-02   2.045 0.040964 *
## NeighborhoodNWAmes     -3.079e-02  2.814e-02  -1.094 0.273989
## NeighborhoodOldTown    -7.887e-02  2.895e-02  -2.725 0.006485 **
## NeighborhoodSawyer     -3.375e-02  2.831e-02  -1.192 0.233269
## NeighborhoodSawyerW    -5.588e-02  2.748e-02  -2.033 0.042133 *
## NeighborhoodSomerst     5.742e-02  2.630e-02   2.183 0.029114 *
## NeighborhoodStoneBr     1.100e-01  3.019e-02   3.644 0.000274 ***
## NeighborhoodSWISU      -5.184e-02  3.201e-02  -1.619 0.105542
## NeighborhoodTimber     -4.144e-03  2.936e-02  -0.141 0.887737
## NeighborhoodVeenker     3.290e-02  3.402e-02   0.967 0.333636
## Condition.1Prox        -4.061e-02  6.483e-03  -6.264 4.45e-10 ***
## Bldg.TypeDuplex2fmCon   1.072e-02  1.602e-02   0.669 0.503509
## Bldg.TypeTwnhsComb     -3.026e-02  1.219e-02  -2.481 0.013154 *
## House.Style1Story       6.717e-02  8.686e-03   7.732 1.56e-14 ***
## House.Style2.5Story     3.389e-02  2.195e-02   1.544 0.122690
## House.Style2Story      -1.378e-02  8.517e-03  -1.618 0.105842
## House.StyleSFoyer       1.415e-01  1.635e-02   8.659  < 2e-16 ***
## House.StyleSLvl         5.319e-02  1.276e-02   4.168 3.18e-05 ***
## Exterior.1stCemntBd    -3.527e-02  1.915e-02  -1.842 0.065575 .
## Exterior.1stHdBoard    -7.705e-02  1.360e-02  -5.667 1.63e-08 ***
## Exterior.1stPlywood    -8.282e-02  1.484e-02  -5.581 2.67e-08 ***
```

```
## Exterior.1stShingle   -7.726e-02  1.261e-02  -6.126 1.06e-09 ***
## Exterior.1stStucco    -4.907e-02  2.118e-02  -2.317 0.020588 *
## Mas.Vnr.Area           8.457e-05  1.548e-05   5.463 5.19e-08 ***
## Exter.Qual.L           1.153e-01  2.080e-02   5.543 3.31e-08 ***
## Exter.Qual.Q           4.027e-02  1.493e-02   2.698 0.007036 **
## Exter.Qual.C           3.038e-02  7.590e-03   4.003 6.46e-05 ***
## Exter.Cond.L           4.074e-02  2.411e-02   1.690 0.091248 .
## Exter.Cond.Q          -9.986e-03  1.800e-02  -0.555 0.579110
## Exter.Cond.C           6.228e-03  9.167e-03   0.679 0.496997
## FoundationCBlock       1.467e-02  9.366e-03   1.566 0.117515
## FoundationOther        3.095e-02  2.101e-02   1.473 0.140914
## FoundationPConc        3.448e-02  1.054e-02   3.270 0.001091 **
## Heating.QC.L           4.503e-02  9.053e-03   4.974 7.04e-07 ***
## Heating.QC.Q          -1.059e-02  7.089e-03  -1.494 0.135386
## Heating.QC.C           6.384e-03  5.182e-03   1.232 0.218099
## Bedroom.AbvGr         -1.912e-02  3.870e-03  -4.941 8.33e-07 ***
## Kitchen.AbvGr         -1.015e-01  1.807e-02  -5.618 2.16e-08 ***
## Kitchen.Qual.L         1.031e-01  1.380e-02   7.470 1.13e-13 ***
## Kitchen.Qual.Q         4.332e-02  9.695e-03   4.468 8.27e-06 ***
## Kitchen.Qual.C         1.971e-02  5.451e-03   3.615 0.000307 ***
## Fireplaces             4.056e-02  4.115e-03   9.856  < 2e-16 ***
## Has.Pool1              5.621e-02  3.486e-02   1.612 0.107026
## Has.Basement1          1.595e-01  1.889e-02   8.443  < 2e-16 ***
## Has.Garage1            6.799e-02  1.108e-02   6.138 9.82e-10 ***
## Year.Age              -2.592e-03  1.955e-04 -13.261  < 2e-16 ***
## Year.Since.Remodel    -8.594e-04  1.655e-04  -5.192 2.26e-07 ***
## Functional.Type.L      6.316e-02  6.405e-03   9.861  < 2e-16 ***
## Overall.Cond.Type.L    1.370e-01  8.511e-03  16.094  < 2e-16 ***
## Overall.Cond.Type.Q   -1.835e-02  5.461e-03  -3.359 0.000794 ***
## Overall.Qual.Type.L    8.302e-02  7.997e-03  10.381  < 2e-16 ***
## Overall.Qual.Type.Q    1.324e-03  4.637e-03   0.286 0.775270
## log(Gr.Liv.Area)       5.765e-01  1.438e-02  40.099  < 2e-16 ***
## I(Lot.Area^0.14)       2.060e-01  1.490e-02  13.824  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1015 on 2319 degrees of freedom
## Multiple R-squared:  0.9258, Adjusted R-squared:  0.9234
## F-statistic: 390.9 on 74 and 2319 DF,  p-value: < 2.2e-16
```

```
par(mfrow=c(2,2))
plot(fit.stepAIC)
```



3.5 Cross-Validation

For all the models we discussed above, we first fitted the model using the full data to do the exploration. However, to evaluate model performance, we performed cross validation using 80-20 train-test split and calculated the root mean squared error (RMSE) of the predictions on the test set. We ran 50 simulations per model and averaged the RMSE across the simulations to get a single metric that we could use to compare performance across models. For the models that had a log transformation on SalePrice, we back-transformed the log on SalePrice before calculating the error. Note: Since the different models had linearizing transforms

such as power and log transforms imposed, RMSE is the best metric that can be used to compare as these as an apples to apples scenario. From the cross validation that the stepwise model with AIC stopping criterion provides the best fit among the ordinary linear regression models.

```
set.seed(2)
nsims <- 50
n <- nrow(housing_data_frame)
rmse.full = rmse.reduced = rmse.transform = rmse.stepAIC = rmse.stepBIC = rmse.ridge = rmse.lasso

for(i in 1:nsims){
  sample <- sample.int(n=n,size=floor(.80*n), replace=FALSE)
  train.data <- housing_data_frame[sample, ]
  test.data <- housing_data_frame[-sample, ]

  #fit1: full model
  fit1 <- glm(formula(fit.full),data=train.data)

  #fit2: reduced model
  fit2 <- glm(formula(fit.reduced),data=train.data)

  #fit3: transformed reduced model
  fit3 <- glm(formula(fit.transform),data=train.data)

  #fit4: stepwise from transformed, reduced model (using AIC as criteria)
  fit4 <- step(fit.transform,direction="both",trace=FALSE)

  rmse.full[i] <- sqrt(mean((predict(fit1,newdata=test.data)-test.data$SalePrice)^2))
  rmse.reduced[i] <- sqrt(mean((predict(fit2,newdata=test.data)-test.data$SalePrice)^2))
  rmse.transform[i] <- sqrt(mean((exp(predict(fit3,newdata=test.data))-test.data$SalePrice)^2))
  rmse.stepAIC[i] <- sqrt(mean((exp(predict(fit4,newdata=test.data))-test.data$SalePrice)^2))


  trainX <- model.matrix(log(SalePrice)~.-Gr.Liv.Area+log(Gr.Liv.Area)-Lot.Area+I(Lot.Area^0.14),t
  trainy <- log(train.data$SalePrice)
  testX <- model.matrix(log(SalePrice)~.-Gr.Liv.Area+log(Gr.Liv.Area)-Lot.Area+I(Lot.Area^0.14),te

}
```

```
results <- data.frame(Model.Name=c("Full Model", "Reduced Model", "Reduced-Transformed Model", "Fo

print(results)
```

```
##                         Model.Name Average.RMSE
## 1                       Full Model     22992.72
## 2                    Reduced Model     22907.63
## 3       Reduced-Transformed Model     20695.39
## 4 Forward/Backward Stepwise (AIC)     19813.91
```

3.6 Using Supervised Learning Models: Random Forest

We also fitted an ensemble learning model on Ames housing dataset based on Random forest algorithm. This algorithm averages output of many decision trees to generate a strong model which performs very well

and does not overfit and also balances the bias-variance trade-off. From the random forest model we also get a variable importance plot which shows which variables are influential in the model predictions. The results from the testing datasets averaged RMSE less than $20K which is comparable to the other models. Also the important variables that were identified include Overall Quality, Neighborhood, and Living Area which match the rest of our best model which is the one obtained with stepwise linear regression with AIC stopping criterion.

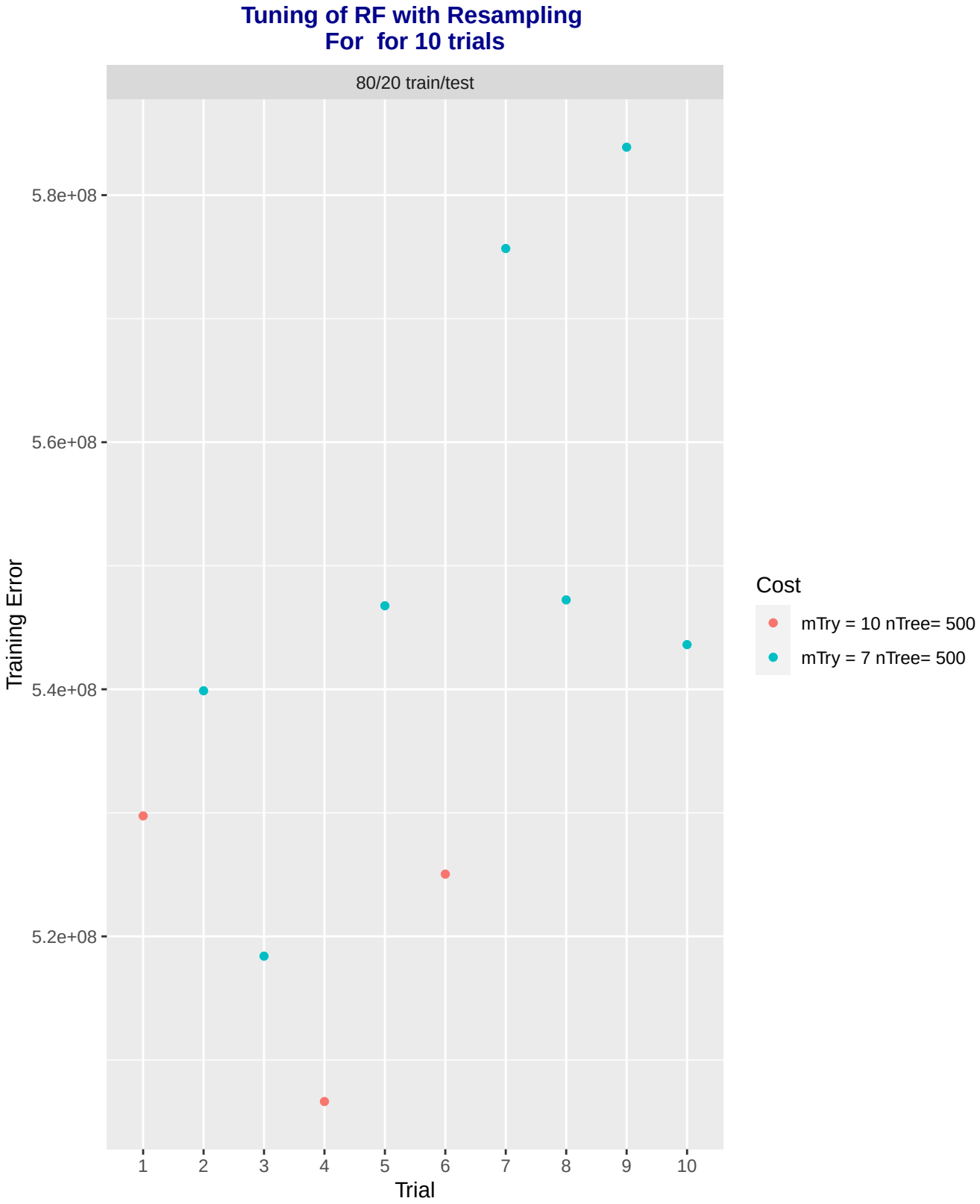
```
summPreds <- function(inpPred,inpTruth,inpMetrNms=c("RMSE","MAE")) {
  retVals <- numeric()
  # Calculate error
  error <- inpTruth - inpPred
  rmse =  sqrt(mean(error^2))
  mae = mean(abs(error))
  retVals["RMSE"] = rmse
  retVals["MAE"] = mae
  retVals
}

train_dataset = housing_data_frame
set.seed(123)
# Running bootstrap and resampling with 80/20 split
rf_resamp_testset_perf_df <- NULL
rf_tune_resamp_out_perf_df <- NULL
rf_var_impt_df <- NULL
num_resamp = 10
rf_best_models_list <- vector("list", num_resamp*2)
rf_best_models_names <- list()
model_list_index = 0
ptr = proc.time()
for ( iResample in c(1) ) {
  for ( iSim in 1:num_resamp ) {
    trainIdx <- sample(nrow(train_dataset),0.8*nrow(train_dataset))
    if ( iResample == 2 ) {
      # break
      trainIdx <- sample(nrow(train_dataset),nrow(train_dataset),replace=TRUE)
    }


    print(paste0(Sys.time(), ": Trial #", iSim, " with ", c("train/test","bootstrap")[iResample]))
    # print("Train Index")
    # print(head(trainIdx))
    model_list_index <- model_list_index +1
    resample_training_df = train_dataset[trainIdx, ]
    resample_testing_df = train_dataset[-trainIdx, ]

    train_x = subset(resample_training_df, select=-c(SalePrice))
    train_y = resample_training_df$SalePrice
    #Running SVM:

    rf_tune_resamp_out <- tuneRF(train_x, train_y, ntreeTry   = 500,
  mtryStart  = 5,  stepFactor = 1.5, improve = 0.05, trace=FALSE, plot= FALSE, doBest=TRUE)
```

```
     # rf_tune_resamp_out <- tuneRF(train_x, train_y, mtryStart=1, ntreeTry=50, stepFactor=1.1, i

#     print(rf_tune_resamp_out)
#   }
# }

    # rf_best_model_fitvec <- serialize(rf_tune_resamp_out,NULL)
    # rf_best_models_list_df <- rbind(rf_best_models_list_df, data.frame(model_id=paste0(c("80/20

    # names(rf_best_models_list[model_list_index]) =
    rf_best_models_list[[model_list_index]] <- rf_tune_resamp_out
    rf_best_models_names[model_list_index] = paste0(c("80/20 train/test","bootstrap")[iResample],'
    rf_imp_matrix = importance(rf_tune_resamp_out)
    imp_df = data.frame(matrix(ncol = nrow(rf_imp_matrix), nrow = 1))

    colnames(imp_df) <- rownames(rf_imp_matrix)


    imp_df[1, ] <- rf_imp_matrix[, 1]
    # print("VIM : ")
    # print(imp_df)

    rf_var_impt_df = rbind(rf_var_impt_df, cbind(trial=iSim, resample=c("80/20 train/test","bootst

    rf_error = min(rf_tune_resamp_out$mse)


    rf_tune_resamp_out_perf_df =
    rbind(rf_tune_resamp_out_perf_df,data.frame(trial=iSim, resample=c("80/20 train/test","bootstr

    # tune_resamp_best_model = tune_resamp_out$best.model

    tune_resamp_TestPred <- predict(rf_tune_resamp_out, resample_testing_df)
    tmpVals <- summPreds(tune_resamp_TestPred, resample_testing_df$SalePrice)


    rf_resamp_testset_perf_df <- rbind(rf_resamp_testset_perf_df,data.frame(resample=c("80/20 trai



  }
}
```

```
## [1] "2020-06-28 15:46:52: Trial #1 with train/test"
## -0.01014306 0.05
## 0.05387544 0.05
## 0.04216378 0.05
## [1] "2020-06-28 15:47:33: Trial #2 with train/test"
## -0.06645127 0.05
```

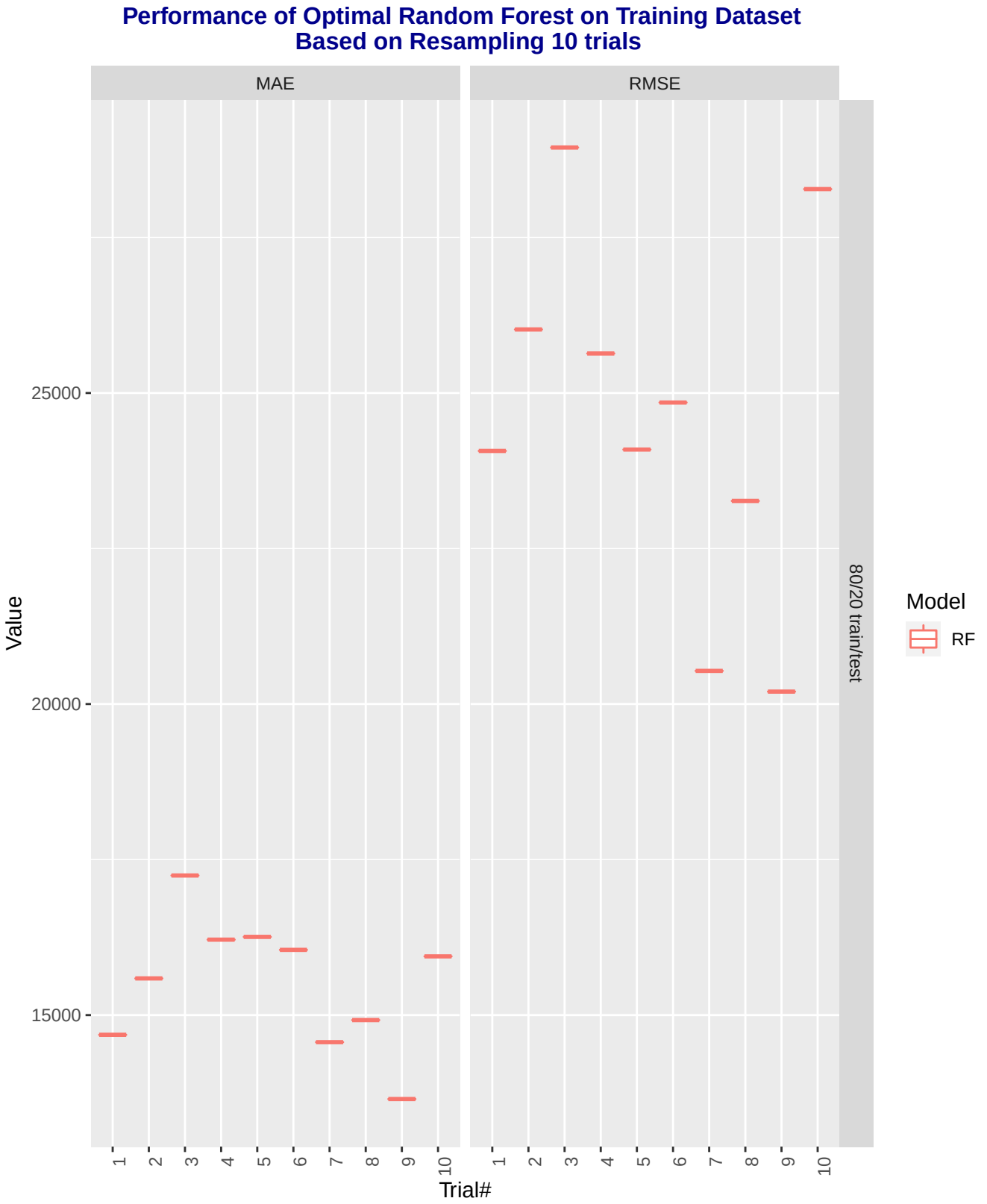
```
## 0.01394785 0.05
## [1] "2020-06-28 15:48:00: Trial #3 with train/test"
## -0.03624255 0.05
## 0.02503544 0.05
## [1] "2020-06-28 15:48:26: Trial #4 with train/test"
## -0.04039266 0.05
## 0.06358745 0.05
## 0.0119319 0.05
## [1] "2020-06-28 15:49:07: Trial #5 with train/test"
## -0.04251934 0.05
## 0.04433978 0.05
## [1] "2020-06-28 15:49:33: Trial #6 with train/test"
## -0.02892662 0.05
## 0.07767534 0.05
## 0.008715364 0.05
## [1] "2020-06-28 15:50:13: Trial #7 with train/test"
## -0.02857239 0.05
## 0.04663503 0.05
## [1] "2020-06-28 15:50:39: Trial #8 with train/test"
## -0.06981571 0.05
## 0.04643967 0.05
## [1] "2020-06-28 15:51:07: Trial #9 with train/test"
## -0.03689671 0.05
## 0.02037786 0.05
## [1] "2020-06-28 15:51:32: Trial #10 with train/test"
## -0.02704064 0.05
## 0.02598088 0.05
```

```
# head(rf_tune_resamp_out_perf_df)
ggplot(rf_tune_resamp_out_perf_df, aes(x=as.factor(trial),y=error, colour=as.factor(model)))+labs(
```

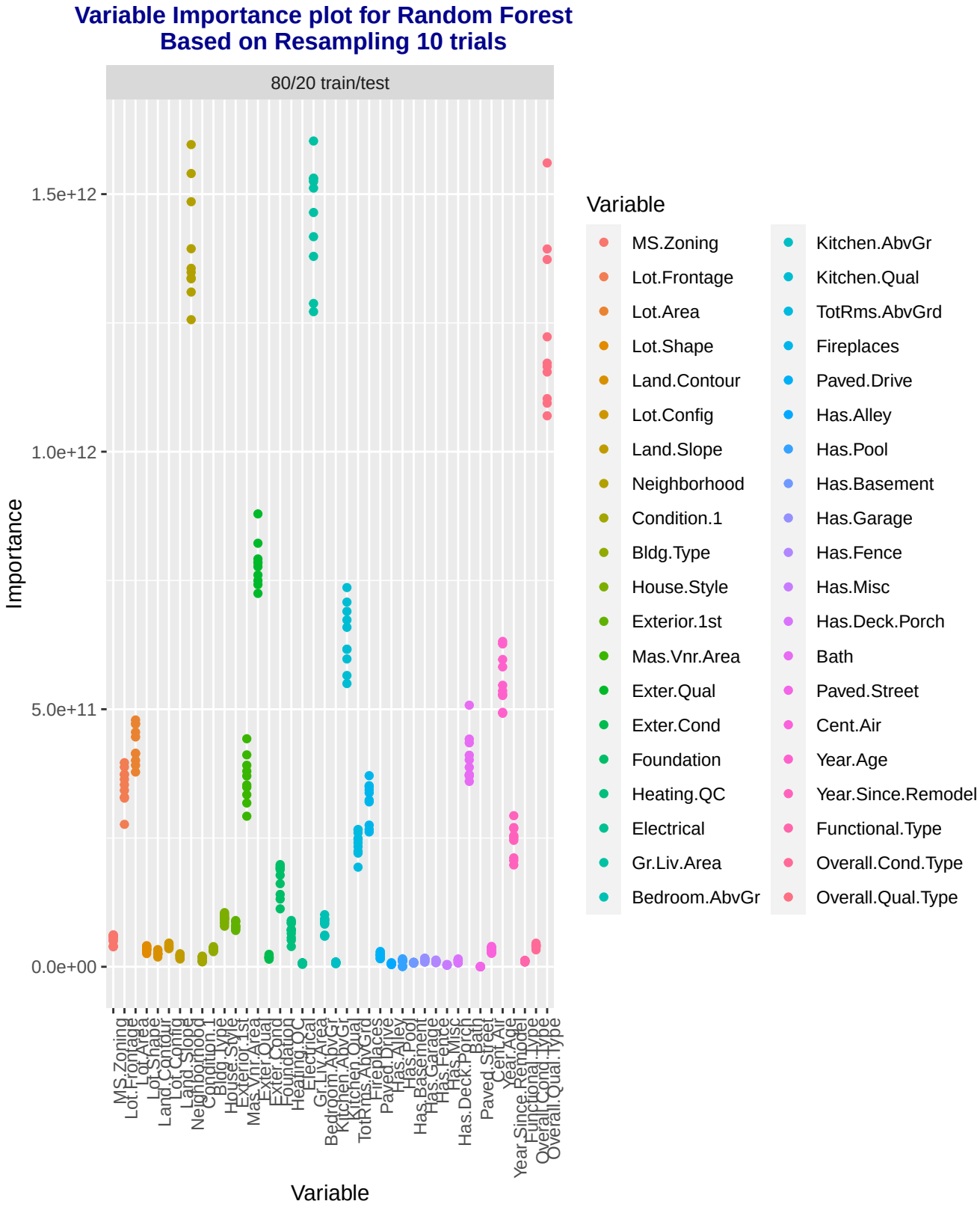



```
# head(rf_resamp_testset_perf_df)

resamp_plotter_df = melt(rf_resamp_testset_perf_df)

## Using resample, model, metric as id variables
```

```
ggplot(rf_resamp_testset_perf_df,aes(x=as.factor(trial),y=value, colour=as.factor(model))) + geom_
```




```
rf_var_impt_plotter_df = melt(rf_var_impt_df, id= c("trial", "resample"))
ggplot(rf_var_impt_plotter_df,aes(x=as.factor(variable),y=value, colour=as.factor(variable))) + ge
```




```
# rf_preds_df_list = write_test_data_predictions(model_obj_list=rf_best_models_list, model_names_
#
# rf_final_test_pred_df = rf_preds_df_list[[1]]
# rf_consensus_pred_df = rf_preds_df_list[[2]]
```

4. Conclusions

Since the model selected by the Forward/Backward Stepwise model (using AIC as the criterion) resulted in the lowest RMSE (~19800) with cross-validation, we took a closer look at the regression output resulting from fitting the full dataset using forward/backward stepwise regression with AIC. This model had the best performance amongst the OLS models with residual standard error = 0.1015, R-squared = 0.925, adjusted R-squared = 0.923.

There were a number of findings were surprising overall:

1. The model suggested that home buyers in Ames placed a lot of emphasis on a basement (finished or unfinished) placing a ~15% premium on the sale price. There seemed to be a higher emphasis on a basement than a garage! The preference for basement might be indicative of the fact that buyers preferred larger and more open plans in their main living areas and to squirrel away all of their other items into a basement or use it for other activities that will reduce the clutter in their main living areas.

2. While we didn't expect most houses to have more than one kitchen, home buyers in Ames really seemed to have a distinct preference against homes with more than one kitchen (above basement), reducing the sale price by 10%!

3. The partial F-test excluding the number of bathrooms had a high p-value, indicating that bathroom count did not have a significant impact on sale price and could be dropped from the model. The results were similar when the models were run with and without consolidating the full and half bathrooms.

4. An increase in the number of bedrooms for comparable other property features (home area, lot area, neighborhood, aesthetics etc.) dropped the sale price by approximately 2%. Controlling for the size of the living area, it appears that home buyers in Ames look for houses with fewer bedrooms and bathrooms. For a given living area, buyers do not like their home layout to be split up too much by having many walled off bedrooms and bathrooms which might indicate their preference for an open plan.

5. We expected newer houses to cost more than older houses. Our final model showed that age of house does had a negative impact but it was less than 0.5% for each additional year in house age. Maybe it's because the mean age of houses in Ames was 38 years, and there were many more older houses than newer houses, rendering home age not as large of a negative factor as we expected. Houses with greater living area have a higher sale price though not much higher. For each 1% increase in living area in square feet, the sale price increased by only 0.57%. We also expected cul-de-sacs and inside lots to have a greater positive impact on sale price but that did not seem to be the case in Ames.

Few other interesting observations: If you owned a 2 story house in Ames, it was tough to find a good deal when you want to sell the house during this period. Single story and single level houses carried a premium of 6% and 5% respectively, but two-story and higher houses didn't seem to have a significant impact on sale price. Being a townhouse decreased the sale price slightly by approximately 3%. Finally, among Neighborhoods, houses in the Greens neighborhood seemed to command the highest premium (approximately 19%) on sale price while houses in Meadow Village had lower sale prices by approximately 14%.

5. Further Discussion and Challenges

For this project, we simplified the modeling by creating dummy variables for many of the property features such as pools, basements, fences. Future work may consider keeping the details e.g. basement finish, pool area, renovation needs of these property features into the model. The age since remodel was not significant in this model since the age of the average home was high, with newer construction this might shift.

Also commute, aesthetics (parks, community amenities), crime rates, median income, and socio-economic disparity (Gini coefficients) in each neighborhood or zip code could be external variables that could have a significant influence on the model. While we excluded Sale.Type on the grounds that the variable was more about how the sale was financed as opposed to capturing the characteristics of the property being purchased, others may want to leave it in the model to control for the impact of sale type, because the way that a buyer finances the home purchase can influence the price that the seller is willing to accept. The dataset had many variables that we were already trying to reduce, so we chose not to introduce interaction and polynomial terms. However, adding those complexities to the model would be simple with the foundation we already have and a worthwhile direction for further exploration.

We faced the biggest challenges in preparing the data for modelling. The dataset started with many variables, and we had to spend a lot of time exploring the data to decide how best to reduce the number of variables we were dealing with. Moreover, many of the variables were categorical, with unbalanced distribution across the categories. This presented a problem during cross-validation, because in any given simulation, a category value for a certain variable could be present in the test set but not in the training set, causing the model to fail. We had to recode many of the categorical variables and group multiple categories together to even out the data distribution between the categories enough such that we wouldn't end up with all the observations of a particular value being in the test set only.

6. References

De Cock, D. (2011). "Ames, Iowa: Alternative to the Boston Housing Data as an End of Semester Regression Project", *Journal of Statistics Education*, Volume 19, Number 3. http://jse.amstat.org/v19n3/decock.pdf