MSDS 6371 Project - Fall 2022

Andrew Yule, Krithika Kondakindi

Introduction

This analysis focuses on applying multi-linear regression techniques to predict the sales price of homes in the Ames area of Iowa. Two specific questions of interest (QOI) were identified:

1. Focusing on 3 specific neighborhoods in Ames, what is the relationship between a home's square footage and it's sales price.
2. Utilizing all available variables and focusing on every neighborhood, what are the best linear regression models that can be constructed? For this QOI, submissions will be judged based on a Kaggle competition, using produced model predictions to achieve the lowest root mean squared error.

Data Description

For this analysis, housing data from the Ames area was supplied between the years of 2006 and 2010 from 1,460 homes. Approximately 80 variables were collected for each home identifying various features that may contribute to a home's final sale price. The variables ranged in nature from quantitative values like square footage of certain parts of the home, to qualitative values like quality rankings and building materials.

Upon analysis of the data, numerous variables were found to have a high number of missing values and were subsequently removed from the analysis.

Many numerical variables in the available data demonstrated high variance. Log transformations were used to reduce the variance and ultimately improve the performance of any linear regression models. The full list of variables that received a log transformation can be found below. It should be noted that sales price (the variable of interest) was among the variables receiving a log transformation.

- LotArea
- FirstFlrSF

- SecondFlrSF
- GrLivArea
- WoodDeckSF
- OpenPorchSF
- EnclosedPorch
- ThirdSsnPorch
- ScreenPorch
- PoolArea
- MiscVal
- SalePrice

Analysis

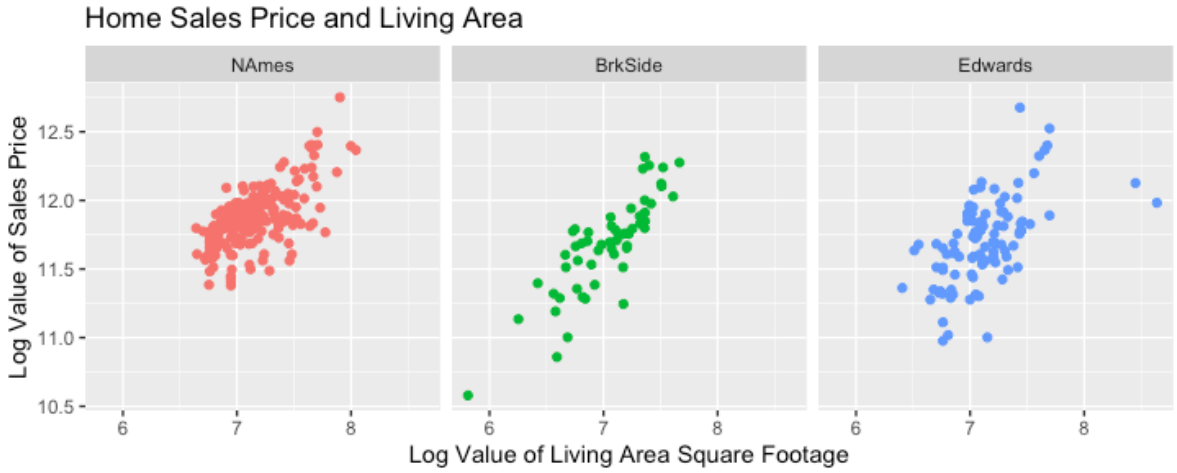
Question 1

Restatement of the Problem

For the first question of interest, the objective was to utilize only a home's square footage to help predict the sales price in the three neighborhoods of North Ames, Brookside, and Edwards.

Limiting the initial dataset to only these 3 neighborhoods results in 383 observations with 225, 58, and 100 for North Ames, Brookside, and Edwards respectively.

Utilizing the log transformations performed in the section above, helps to improve the linear relationships between square footage and sales price for the 3 neighborhoods.



Build and Fit the Model

Three models were fit to the data:

1. Average per neighborhood - A model which fits the average sales price for each neighborhood.
2. Varying y-intercept / same slopes - A model which allows for varying y-intercepts across each neighborhood while maintaining the same relationship in sales price and living room square footage.
3. Varying y-intercept / varying slopes - A model which allows for varying y-intercepts across each neighborhood in addition to varying relationships in sales price and living room square footage.

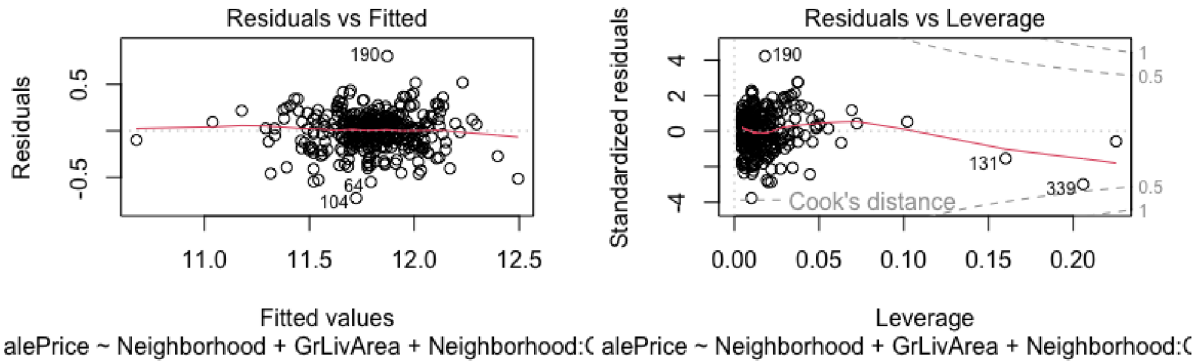
Comparing Competing Models

The table below summarizes the key performance metrics associated with each of the three models produced. Overall the third model, which allows for varying y-intercepts and slopes across each neighborhood, outperforms the other 2 models.

Model	Adjusted R2	CV PRESS
Average per neighborhood	0.0880	26.44
Varying y-intercept / same slope	0.4857	15.00
Varying y-intercepts / varying slopes	0.5056	14.61

Checking the Assumptions

The figure below summarizes the residual and influential point analysis conducted on the final model. Overall, the model appears to meet the basic assumptions required of linear regression as indicated by a random scattering of the residuals. Several high leverage points were found in the influential point analysis, however, no points were removed as they could not confidently be assumed to be erroneous data.



Parameters

The table below summarizes the determined regression coefficients and associated confidence intervals. The reference for y-intercept and slope values is the North Ames neighborhood.

Based on these results, it can be determined that homes in the Brookside and Edwards neighborhoods have lower starting values for homes with low square footage, however, homes in those neighborhoods tend to increase in sales price values more sharply as square footage increases. More specifically, a doubling of square footage in each neighborhood is associated with a multiplicative increase in sales price of 39%, 76%, and 43% for North Ames, Brookside, and Edwards respectively.

	Estimate	Standard Error	Confidence Interval
1	8.49273	0.324417	{7.85483, 9.13062}
Neighborhood[BrkSide]	−2.57981	0.599881	{−3.75934, −1.40027}
Neighborhood[Edwards]	−0.48622	0.517508	{−1.50378, 0.531344}
LivingArea	0.473024	0.0454289	{0.383698, 0.562349}
LivingArea Neighborhood[BrkSide]	0.346624	0.0848201	{0.179845, 0.513404}
LivingArea Neighborhood[Edwards]	0.0466436	0.0724801	{−0.0958723, 0.18916}

R Shiny: Price v. Living Area Chart

To further explore the relationships between home square footage and sales price in these neighborhoods, a web application was created which can be found at the link below:

Shinny App

Question 2

For the second question of interest, the objective was to utilize all available data and variables to create the best linear regression model possible for predicting home sales prices. There are numerous techniques available for creating multi-linear regression models. This QOI focused on applying forward selection, backwards elimination, and stepwise selection to produce the highest performing models.

Model Selection

The olsrr package in R was utilized to perform each of the 3 stepping regression fittings. Minimizing the Akaike Information Criterion (AIC) was used as the model criteria. The following linear forms below were identified to produce the lowest AIC for forward, backward, and step selection respectively. A custom model was also attempted, however, it was unable to achieve higher accuracy than the preceding 3 models and thus was discarded.

Forward selection:

SalePrice = OverallQual + GrLivArea + Neighborhood + OverallCond + HouseStyle + YearBuilt + LotArea + RoofMatl + KitchenAbvGr + SaleCondition + Condition2 + Foundation + Fireplaces + Heating + ExterQual + Condition1 + PoolArea + ScreenPorch + WoodDeckSF + HeatingQC + CentralAir + BedroomAbvGr + FirstFlrSF + `2ndFlrSF` + Street + LandSlope + HalfBath + EnclosedPorch + SecondFlrSF + MiscVal + PavedDrive + `1stFlrSF` + BldgType + ExterCond + YearRemodAdd

Backwards elimination:

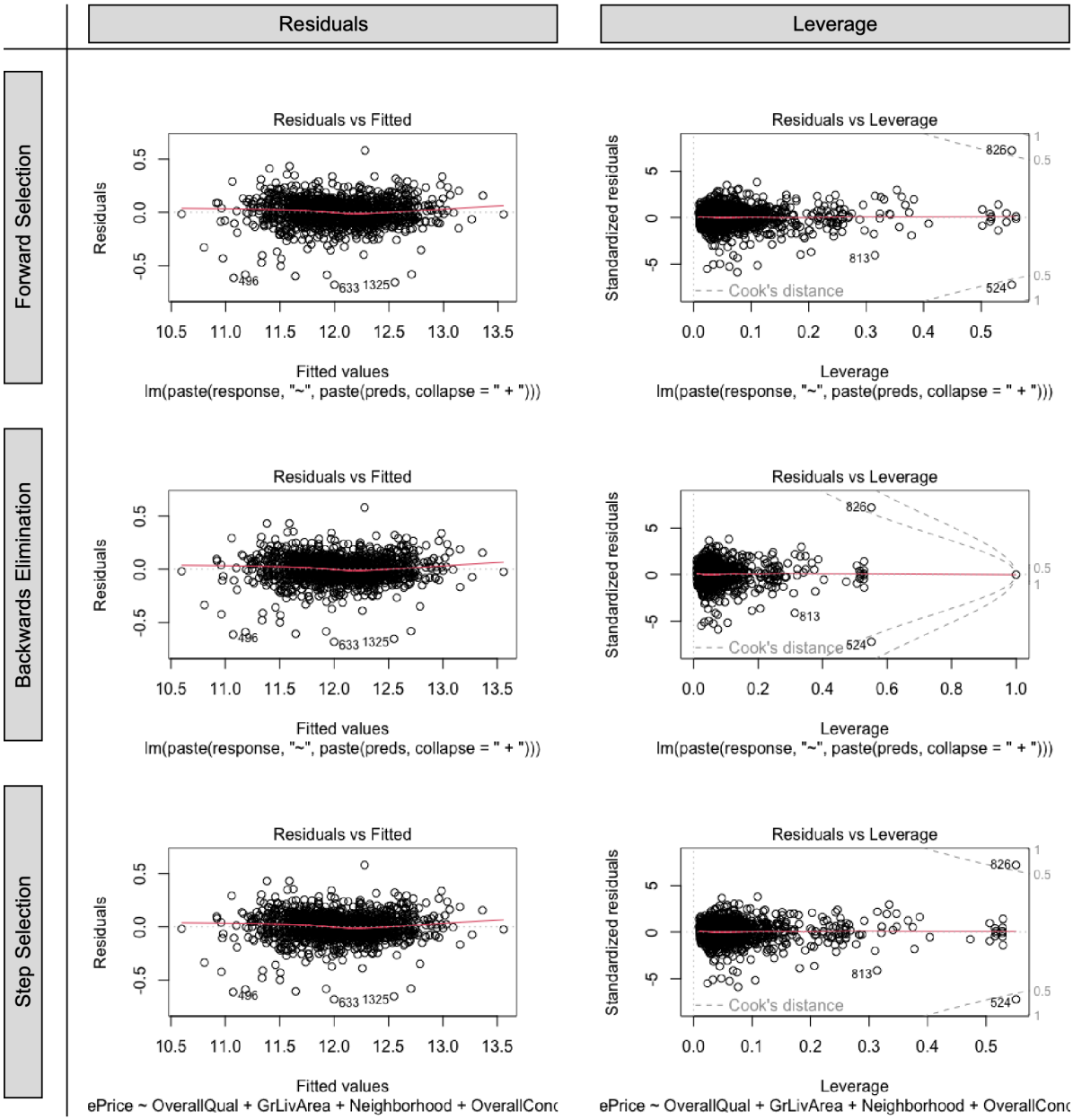
SalePrice = LotArea + Street + LotConfig + LandSlope + Neighborhood + Condition1 + Condition2 + BldgType + OverallQual + OverallCond + YearBuilt + YearRemodAdd + RoofMatl + ExterQual + ExterCond + Foundation + Heating + HeatingQC + CentralAir + `1stFlrSF` + `2ndFlrSF` + GrLivArea + HalfBath + BedroomAbvGr + KitchenAbvGr + Fireplaces + PavedDrive + WoodDeckSF + EnclosedPorch + ScreenPorch + PoolArea + MiscVal + SaleCondition + FirstFlrSF + SecondFlrSF

Stepwise selection:

SalePrice = OverallQual + GrLivArea + Neighborhood + OverallCond + YearBuilt + LotArea + RoofMatl + KitchenAbvGr + SaleCondition + Condition2 + Foundation + Fireplaces + Heating + ExterQual + Condition1 + PoolArea + ScreenPorch + WoodDeckSF + HeatingQC + CentralAir + BedroomAbvGr + FirstFlrSF + `2ndFlrSF` + Street + LandSlope + HalfBath + EnclosedPorch + SecondFlrSF + MiscVal + PavedDrive + BldgType + `1stFlrSF` + ExterCond + LotConfig + YearRemodAdd

Checking Assumptions

The figure below summarizes the residual and influential point analysis conducted on each of the 3 models. Overall, each model appears to meet the assumptions required of linear regression as indicated by a random scattering of the residuals. Several high leverage points were found in the influential point analysis, however, no points were removed as they could not confidently be assumed to be erroneous data.



Comparing Competing Models

The table below summarizes the key performance metrics associated with each of the three models produced. Each model was found to have the same Adjusted R Squared metric within rounding errors. According to the predictive residual sum of squares metric, the stepwise selection model performed the best. When used to predict unknown home sales prices as part of the Kaggle competition, the Forward Selection model was found to achieve the best (lowest) Kaggle score of 0.13496. A custom model was excluded from this table as it was unable to achieve higher performance than the Forward and Stepwise selection methods.

Model	Adjusted R2	CV PRESS	Kaggle Score
Forward Selection	0.9094	26.12	0.13496
Backward Elimination	0.9094	25.95	0.13554
Stepwise Selection	0.9094	25.94	0.13554

Conclusion

In this analysis, a data set of home sales prices from Ames, Iowa was explored using multi-linear regression techniques. A model was produced to predict the sales price of homes specifically in the North Ames, Brookside, and Edwards neighborhoods using only the square footage. That model achieved an RSquared metric of 0.51, meaning 51% of the variance in home prices in those 3 neighborhoods can be explained by the square footage.

Finally, 3 additional models were produced to predict homes sales prices based on all variables available. Various multi-linear regression stepping approaches were used and the forward selection method was found to perform the best overall with an adjusted RSquared metric of 0.91 meaning 91% of the variance in home prices could be explained by the variables and associated model. Additionally, a Kaggle competition score of 0.13496 was achieved by the model on a blind test.

Appendix

The entire code for this analysis can be found below:

Load required libraries

```
library(MASS)
library(olsrr)
library(caret)
library(terra)
library(performance)
```

```
library(tidyverse)
library(broom)
```

Read training and test data, then drop any columns containing missing values

```
housesTrain = read_csv("train.csv")
housesTest = read_csv("test.csv")

# Remove ID column only from the training set
housesTrain = dplyr::select(housesTrain, -Id)

# Find which columns have missing values between the training and test data sets
colsToDrop = housesTrain |>
  select_if(function(x) any(is.na(x))) |>
  colnames()
colsToDrop2 = housesTest |>
  select_if(function(x) any(is.na(x))) |>
  colnames()

# Remove any of the columns that had missing values
housesTrain =  housesTrain |>
  dplyr::select(-starts_with(c(colsToDrop, colsToDrop2)))
housesTest =  housesTest |>
  dplyr::select(-starts_with(c(colsToDrop, colsToDrop2)))
```

```
# Screen for numerical variables that should be log-transformed
housesTrain |>
  dplyr::select(where(is.numeric)) |>
  gather() |>
  ggplot(aes(x = value)) +
    facet_wrap(~key, scales = "free") +
    geom_histogram()
```

The following variables can be log transformed in order to reduce their variance: - LotArea - FirstFlrSF - SecondFlrSF - GrLivArea - WoodDeckSF - OpenPorchSF - EnclosedPorch - ThirdSsnPorch - ScreenPorch - PoolArea - MiscVal - SalePrice

Additionally, MoSold variable was identified as needing to be a factor instead of an integer

```
# MoSold should be a factor instead of integer
housesTrain$MoSold = factor(housesTrain$MoSold, levels = seq(1, 12))
```

```
housesTest$MoSold = factor(housesTest$MoSold, levels = seq(1, 12))

# First ensure that any variables that need log transformations are set
#to a minimum value of 1 instead of 0. Then, take the log of the values.
housesTrain = housesTrain |>
  mutate(
    LotArea = log(clamp(LotArea, lower = 1)),
    FirstFlrSF = log(clamp(`1stFlrSF`, lower = 1)),
    SecondFlrSF = log(clamp(`2ndFlrSF`, lower = 1)),
    GrLivArea = log(clamp(GrLivArea, lower = 1)),
    WoodDeckSF = log(clamp(WoodDeckSF, lower = 1)),
    OpenPorchSF = log(clamp(OpenPorchSF, lower = 1)),
    EnclosedPorch = log(clamp(EnclosedPorch, lower = 1)),
    ThirdSsnPorch = log(clamp(`3SsnPorch`, lower = 1)),
    ScreenPorch = log(clamp(ScreenPorch, lower = 1)),
    PoolArea = log(clamp(PoolArea, lower = 1)),
    MiscVal = log(clamp(MiscVal, lower = 1)),
    SalePrice = log(clamp(SalePrice, lower = 1)),
  )

# Perform any transformations on the test data as well
housesTest = housesTest |>
  mutate(
    LotArea = log(clamp(LotArea, lower = 1)),
    FirstFlrSF = log(clamp(`1stFlrSF`, lower = 1)),
    SecondFlrSF = log(clamp(`2ndFlrSF`, lower = 1)),
    GrLivArea = log(clamp(GrLivArea, lower = 1)),
    WoodDeckSF = log(clamp(WoodDeckSF, lower = 1)),
    OpenPorchSF = log(clamp(OpenPorchSF, lower = 1)),
    EnclosedPorch = log(clamp(EnclosedPorch, lower = 1)),
    ThirdSsnPorch = log(clamp(`3SsnPorch`, lower = 1)),
    ScreenPorch = log(clamp(ScreenPorch, lower = 1)),
    PoolArea = log(clamp(PoolArea, lower = 1)),
    MiscVal = log(clamp(MiscVal, lower = 1))
  )
```

Define a function to determine the PRESS of each model

```
PRESS = function(model) {
    i = residuals(model)/(1 - lm.influence(model)$hat)
    sum(i^2)
}
```

Analysis 1

Create models for Sales Price for the neighborhood Brookside, Edwards, and North Ames neighborhoods.

```
housesTrain |>
  ggplot(aes(x = GrLivArea, y = SalePrice)) +
  geom_point()

housesTrainNeighborhood = housesTrain |>
  filter(Neighborhood %in% c("BrkSide", "Edwards", "NAmes"))

housesTrainNeighborhood$Neighborhood =
  factor(housesTrainNeighborhood$Neighborhood, levels =
           c("NAmes", "BrkSide", "Edwards"))

housesTrainNeighborhood |>
  group_by(Neighborhood) |>
  summarize(n = n())

housesTrainNeighborhood |>
  ggplot(aes(x = GrLivArea, y = SalePrice, color = Neighborhood)) +
  geom_point() +
  facet_wrap(~Neighborhood) +
  theme(legend.position="none") +
  xlab("Log Value of Living Area Square Footage") +
  ylab("Log Value of Sales Price") +
  ggtitle("Home Sales Price and Living Area")

model1 = lm(SalePrice ~ Neighborhood, data = housesTrainNeighborhood)
summary(model1)
PRESS(model1)

model2 = lm(SalePrice ~ Neighborhood + GrLivArea,
            data = housesTrainNeighborhood)
summary(model2)
PRESS(model2)

model3 = lm(SalePrice ~ Neighborhood + GrLivArea + Neighborhood:GrLivArea,
            data = housesTrainNeighborhood)
summary(model3)
PRESS(model3)
```

```
plot(model3)

housesTrainNeighborhood |>
  select(Neighborhood, GrLivArea, SalePrice) |>
  mutate(SalePricePredicted = predict(model3,
                                      newdata = tibble(
                                        Neighborhood = Neighborhood,
                                        GrLivArea = GrLivArea))) |>
  ggplot() +
  geom_point(
    aes(x = exp(GrLivArea), y = exp(SalePrice), color = Neighborhood)) +
  geom_line(
    aes(x = exp(GrLivArea), y = exp(SalePricePredicted),
        color = Neighborhood)) +
  facet_wrap(~Neighborhood) +
  theme(legend.position="none")
```

Analysis 2

Forward step regression using olss

```
forwardResults = ols_step_forward_aic(
  lm(SalePrice ~ ., data = housesTrain), details = TRUE, progress = TRUE)
forwardModel = forwardResults$model
summary(forwardModel)

# Plot the stepping results impact on AIC
plot(forwardResults)

# Plot the residual / leverage functions
plot(forwardModel)

# Calculate the PRESS for the model
tibble(resids = residuals(forwardModel),
       hats = tidy(hatvalues(forwardModel))$x) |>
  filter(hats != 1) |>
  mutate(i = (resids / (1-hats))^2) |>
  select(i) |>
  pull() |>
  sum()
```

```
# Calculate values against the test data set
results = housesTest
results$SalePrice = exp(predict.lm(forwardModel, newdata = housesTest))
results = results |>
  dplyr::select(Id, SalePrice)

# Export and submit to Kaggle
write_csv(results, "Submissions/Forward_Model.csv")
```

Backward step regression

```
backwardResults = ols_step_backward_aic(
  lm(SalePrice ~ ., data = housesTrain), details = TRUE, progress = TRUE)
backwardModel = backwardResults$model
summary(backwardModel)

# Plot the stepping results impact on AIC
plot(backwardResults)

# Plot the residual / leverage functions
plot(backwardModel)

# Calculate the PRESS for the model
tibble(resids = residuals(backwardModel),
       hats = tidy(hatvalues(backwardModel))$x) |>
  filter(hats != 1) |>
  mutate(i = (resids / (1-hats))^2) |>
  select(i) |>
  pull() |>
  sum()

# Calculate values against the test data set
results = housesTest
results$SalePrice = exp(predict.lm(backwardModel, newdata = housesTest))
results = results |>
  dplyr::select(Id, SalePrice)

# Export and submit to Kaggle
write_csv(results, "Submissions/Backward_Model.csv")
```

Stepwise regression

```r
stepResults = ols_step_both_aic(
  lm(SalePrice ~ ., data = housesTrain), details = TRUE)

# ols_step_both_aic does not have a direct output for the model determined
#so we have to write the results as we go and then copy the final model
#produced and paste it below
stepModel = lm(SalePrice ~ OverallQual + GrLivArea + Neighborhood + OverallCond
               + YearBuilt + LotArea + RoofMatl + KitchenAbvGr + SaleCondition
               + Condition2 + Foundation + Fireplaces + Heating + ExterQual
               + Condition1 + PoolArea + ScreenPorch + WoodDeckSF + HeatingQC
               + CentralAir + BedroomAbvGr + FirstFlrSF + `2ndFlrSF` + Street
               + LandSlope + HalfBath + EnclosedPorch + SecondFlrSF + MiscVal
               + PavedDrive + BldgType + `1stFlrSF` + ExterCond + LotConfig
               + YearRemodAdd, data = housesTrain)
# Plot the stepping results impact on AIC
plot(stepResults)

# Plot the residual / leverage functions
plot(stepModel)

# Calculate the PRESS for the model
tibble(resids = residuals(stepModel), hats = tidy(hatvalues(stepModel))$x) |>
  filter(hats != 1) |>
  mutate(i = (resids / (1-hats))^2) |>
  select(i) |>
  pull() |>
  sum()

# Calculate values against the test data set
results = housesTest
results$SalePrice = exp(predict.lm(stepModel, newdata = housesTest))
results = results |>
  select(Id, SalePrice)

# Export and submit to Kaggle
write_csv(results, "Submissions/Step_Model.csv")
```

Custom model using all variables - note, this was unable to achieve higher accuracy than the forward selection results

```r
customModel = lm(SalePrice ~ I(MSSubClass^2) + I(LotArea)^2 + Street + LotShape
                 + LandContour + LotConfig + LandSlope + Neighborhood
                 + Condition1 + Condition2 + BldgType + HouseStyle
                 + I(OverallQual^2) + OverallCond + I(YearBuilt^2)
                 + YearRemodAdd + RoofStyle + RoofMatl + ExterQual + ExterCond
                 + Foundation + Heating + HeatingQC + CentralAir
                 + log(`1stFlrSF`) + `2ndFlrSF` + LowQualFinSF + I(GrLivArea^2)
                 + FullBath + HalfBath + BedroomAbvGr + KitchenAbvGr
                 + I(TotRmsAbvGrd^2) + I(Fireplaces^2) + PavedDrive
                 + WoodDeckSF + OpenPorchSF + EnclosedPorch + `3SsnPorch`
                 + ScreenPorch + PoolArea + MiscVal + MoSold
                 + YrSold + SaleCondition, data = housesTrain)

summary(customModel)

# Calculate values against the test data setresults = housesTest
results = housesTest
results$SalePrice = exp(predict(customModel, newdata = housesTest))
results = results |>
  select(Id, SalePrice)

# Export and submit to Kaggle
write_csv(results, "Submissions/Custom_Model.csv")
```