Linear Regression Analysis of Ames Housing Dataset
Kristin Henderson
Summer 2024

Introduction

In this analysis, I aim to address two questions regarding home sale prices in Ames, Iowa using various regression techniques. The first question focuses on understanding the relationship between the sale price and the square footage of the living area in three specific neighborhoods. This estimate is intended to provide valuable market-specific insights for Century 21 Ames. The second question seeks to build a predictive model for home sale prices across all of Ames, identifying which combination of home characteristics produces the most robust model and best predictions.

Data Description

The data used in these analyses are from the Ames Housing Dataset (De Cock, 2011). This dataset was obtained from the "House Prices - Advanced Regression Techniques" competition on Kaggle (Montoya, 2016). The training set has 1460 observations with 81 variables, and the testing set has 1459 observations with 80 variables, missing the 'SalePrice' response variable.

Variables of importance for these analyses:

- SalePrice: The property's sale price in dollars, the target variable.
- BsmtFinSF1: Type 1 finished square feet.
- GarageCars: Size of garage in car capacity.
- GrLivArea: Above ground living area square feet.
- LotArea: Lot size in square feet.
- MSSubClass: The building class.
- Neighborhood: Physical locations within Ames city limits.
- OverallCond: Overall condition rating.
- OverallQual: Overall material and finish quality.
- YearBuilt: Original construction date.

Analysis Question 1

Problem

I aim to use linear regression to estimate the relationship between home sale price and the living area in square feet within the North Ames, Edwards and Brookside neighborhoods. I would also like to determine if this relationship depends on the neighborhood.

Build and Fit the Model

I will start by examining the most simple model, $\mu\{\text{Price} \mid \text{Area}\} = \beta_0 + \beta_1\text{Area}$. I will also include neighborhood as an indicator variable, $\mu\{\text{Price} \mid \text{Area, Neighborhood}\} = \beta_0 + \beta_1\text{Area} + \beta_2\text{Brookside} + \beta_3\text{Edwards}$, and examine interaction terms with Area and Neighborhood, $\mu\{\text{Price} \mid \text{Area, Neighborhood}\} = \beta_0 + \beta_1\text{Area} + \beta_2\text{Brookside} + \beta_3\text{Edwards} + \beta_4\text{Area*Brookside} + \beta_5\text{Area*Edwards}$. Additionally, I will explore if any transformations of the data may be appropriate.

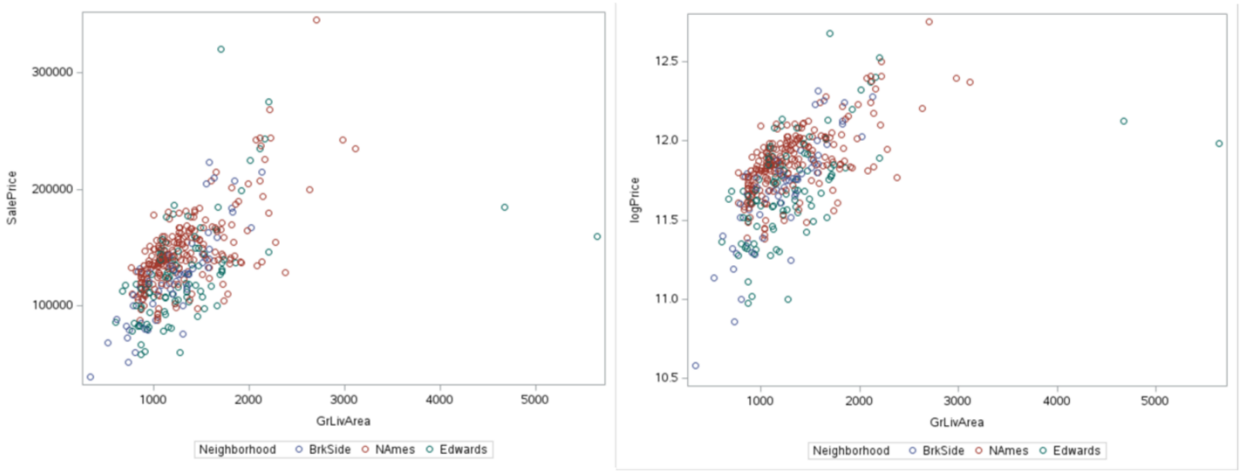


Figure 1: Scatterplot of sale price in dollars and above ground living area in square feet with neighborhoods plotted by color (blue: Brookside, red: North Ames, green: Edwards). Original, untransformed data are on the left, and data with log transformation of sale price are on the right.

After applying log transformations to both the living area explanatory and the sale price response variables and comparing plots of the untransformed data and the log transformed data as in Figure 1, the log transformations seems to provide a better linear fit. I will proceed with candidate models: one with log-transformed price and the untransformed area (log-lin), and the other with log transformations of both price and area (log-log). A comparison of these scatterplots is in Supplementary Figure 1 in the appendix.

Checking Assumptions

Residual Plots

The residual plots of the log-lin transformed data (Figure 2) satisfy the assumptions of linear regression better than that of the untransformed data. The residuals of the log-log transformed data in Supplementary Figure 2 also look quite good.

Influential point analysis (Cook's D and Leverage)

One house in the dataset has a large Cook's D (Figure 3). This house has a large square footage but a low sale price, likely due to its partial sale condition. I found four partially completed houses within the neighborhoods of interest, with at least two being influential points. Those two also are the only houses with square footages greater than 3500. Although I could restrict the data range and exclude partially completed houses above 3500 square feet, I decided to exclude all partially completed houses from this analysis, because there are only four. If analyzing all neighborhoods, this strategy might not be appropriate. In the full dataset, there are many more houses listed in partial condition as there are houses with large square footages.

After restricting the analysis and removing these data, the leverage plots look much better, with little evidence of additional influential points, except for one very small and low-priced house.

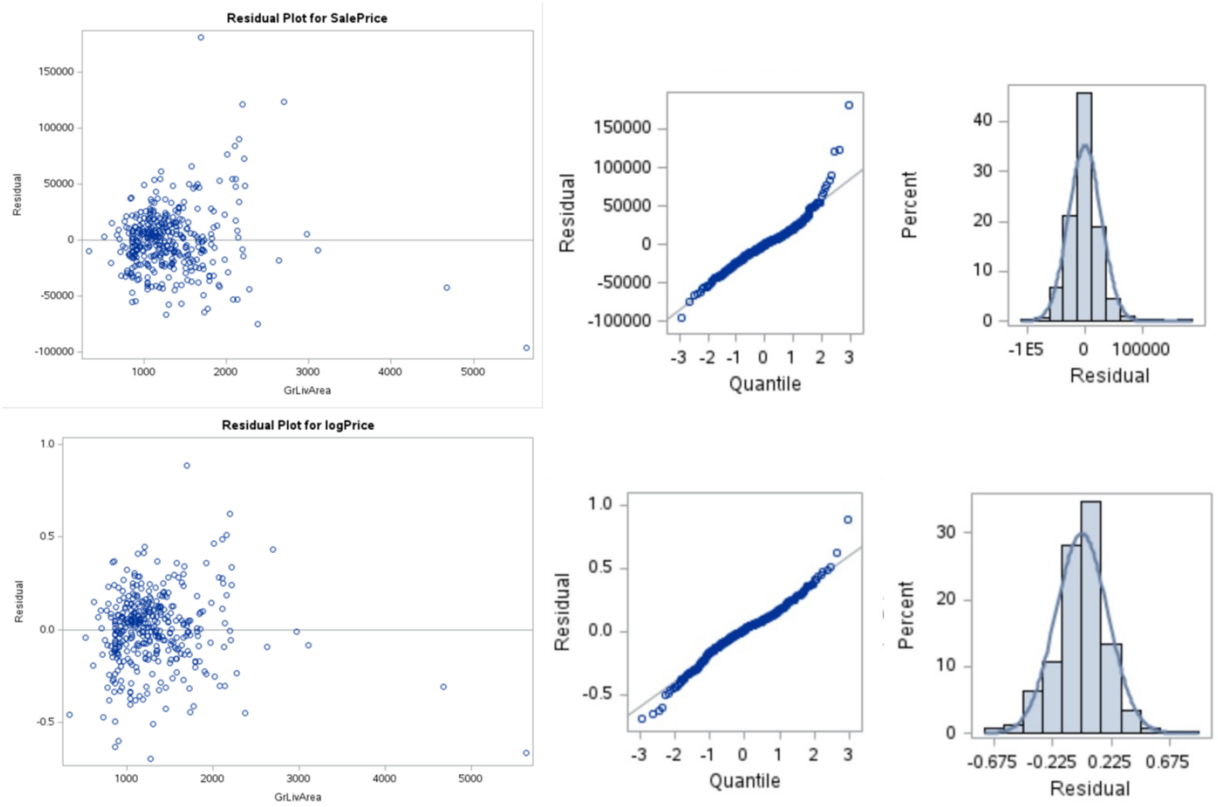


Figure 2: From left to right, residual plots, QQ plots, and histograms of untransformed data (top) and data with log transformation of sale price (bottom).

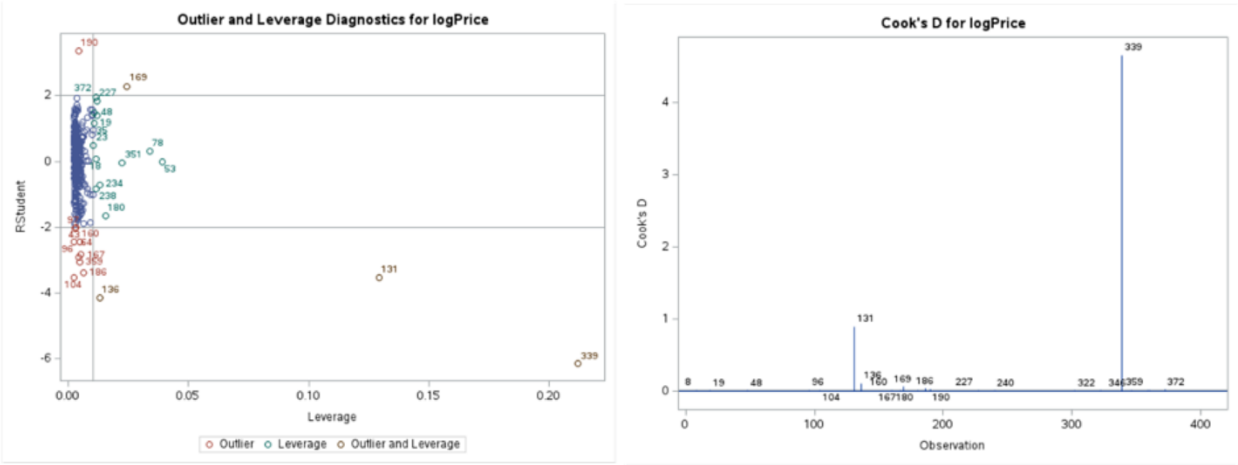


Figure 3: Plots of studentized residuals and leverage (left) and Cook's D (right) for the log-lin transformed data before removing partially completed houses from the analysis.

Address Assumptions

1. Linearity: Examining the scatterplots of the original and log transformed price data in Figure 1, the log-lin transformed data appear to provide a better linear fit.

2. Equal Standard Deviation: The residual plot of the untransformed data in Figure 2 shows some evidence of heteroscedasticity. The log-lin transformation reduces this, showing more random scattering with no evidence against equal standard deviation.
3. Normality: Both the original and log transformed histograms in Figure 2 are consistent with normal distributions. The QQ plot and random scattering of residuals in the log-lin transformed data are more consistent with a normal distribution than those of the untransformed data.
4. Independence: I have no evidence against independence.

Comparing Competing Models: Adj R^2**,** Internal CV PRESS**,** AIC

Effects:	Intercept GrLivArea Neighborhood GrLivArea*Neighborho

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value
Model	5	15.12432	3.02486	85.00
Error	373	13.27352	0.03559	
Corrected Total	378	28.39783		

Root MSE	0.18864
Dependent Mean	11.79698
R-Square	0.5326
Adj R-Sq	0.5263
AIC	-877.31911
AICC	-877.01722
PRESS	13.84633
SBC	-1234.69389

Effects:	Intercept logArea Neighborhood logArea*Neighborhood

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value
Model	5	15.12337	3.02467	84.99
Error	373	13.27446	0.03559	
Corrected Total	378	28.39783		

Root MSE	0.18865
Dependent Mean	11.79698
R-Square	0.5326
Adj R-Sq	0.5263
AIC	-877.29202
AICC	-876.99013
PRESS	13.80478
SBC	-1234.66680

Effects:	Intercept logArea Neighborhood

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value
Model	3	14.43888	4.81296	129.30
Error	375	13.95895	0.03722	
Corrected Total	378	28.39783		

Root MSE	0.19293
Dependent Mean	11.79698
R-Square	0.5085
Adj R-Sq	0.5045
AIC	-862.23634
AICC	-862.07548
PRESS	14.32273
SBC	-1227.48620

Figure 4: Comparison of statistics of competing models. Left: interaction variables and independent slopes when price is log transformed. Center: or when both price and area are log transformed; Right: indicator variables with the same slope when both price and area are log transformed.

All the statistics in Figure 4 indicate that models using log transformation with independent slopes for each neighborhood perform the best among the candidate models. This is consistent when comparing additional models not shown, which did not account for the neighborhood or include data transformation. The model with log-transformed price but untransformed area performed slightly better than when both variables were log-transformed. I opted for this model due to its easier interpretation. The AIC, BIC and internal CV PRESS are the lowest for this model and are lower than the models with log-log transformations with or without interaction variables. The adjusted R^2 is the same for the equivalent model with log-log transformations and is higher than those without interaction variables.

Parameters

Estimates

General model:

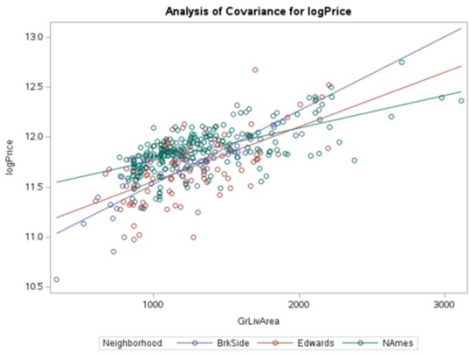
$\mu\{\log(\text{Price}) \mid \text{Area, Neighborhood}\} = 11.4433 + 0.0003*\text{Area} - 0.6517*\text{Brookside} - 0.4334*\text{Edwards} + 0.0004(\text{Area}*\text{Brookside}) + 0.0002(\text{Area}*\text{Edwards})$

Neighborhood specific models:

$\mu\{\log(\text{Price}) \mid \text{Area, Neighborhood = North Ames}\} = 11.0099 + 0.0003*\text{Area}$

$\mu\{\log(\text{Price}) \mid \text{Area, Neighborhood = Edwards}\} = 10.7916 + 0.0005*\text{Area}$

$\mu\{\log(\text{Price}) \mid \text{Area, Neighborhood = Brookside}\} = 11.4433 + 0.0007*\text{Area}$



Parameter	Estimate		Standard Error	t Value	Pr > |t|	95% Confidence Limits	
Intercept	11.44334070	B	0.04187381	273.28	<.0001	11.36100237	11.52567903
GrLivArea	0.00032412	B	0.00003048	10.63	<.0001	0.00026419	0.00038406
Neighborhood BrkSide	-0.65174673	B	0.09171915	-7.11	<.0001	-0.83209815	-0.47139531
Neighborhood Edwards	-0.43335031	B	0.08240274	-5.26	<.0001	-0.59538246	-0.27131815
Neighborhood NAmes	0.00000000	B	.	.	.	.	.
GrLivArea*Neighborho BrkSide	0.00041410	B	0.00007146	5.80	<.0001	0.00027359	0.00055461
GrLivArea*Neighborho Edwards	0.00022211	B	0.00006199	3.58	0.0004	0.00010022	0.00034401
GrLivArea*Neighborho NAmes	0.00000000	B	.	.	.	.	.

Figure 5: Regression lines and parameter estimate table of chosen model: $\mu\{\log(\text{Price}) \mid \text{Area}, \text{Neighborhood}\} = \beta_0 + \beta_1\text{Area} + \beta_2\text{Brookside} + \beta_3\text{Edwards} + \beta_4\text{Area*Brookside} + \beta_5\text{Area*Edwards}$.

Interpretation

Slopes: There is convincing evidence at the $\alpha = 0.05$ confidence level that for every 100 square foot increase in above ground living area, the mean expected sale price increases in each of these neighborhoods. There is overwhelming evidence that in North Ames, with each additional 100 square feet of living area, the mean expected sale price increases $e^{(0.00032*100)} = 1.0325$ times or by 3.25% (p-value < 0.0001). There is strong evidence that in Edwards, the increase is $e^{(0.00054*100)} = 1.0555$ times or by 5.55% (p-value = 0.0004). There is overwhelming evidence that in Brookside the increase is $e^{(0.00073*100)} = 1.0757$ times or by 7.57% (p-value < 0.0001).

Intercepts: There is overwhelming evidence that for houses with zero square feet of above-ground living area, the mean expected sale price in North Ames is $e^{11.4433} = \$93{,}274$ (p-value < 0.0001. In Edwards, the mean expected sale price is 35.2% ($1 - e^{-0.4334} = 1 - 0.648$) less than homes in North Ames or $e^{11.0099} = \$60{,}470$. In Brookside, the mean expected sale price is 47.9% ($1 - e^{-0.6517} = 1 - 0.521$) less than homes in North Ames or $e^{10.7916} = \$48{,}611$ (p-value < 0.0001. As houses typically have more than zero square feet of living area, this estimate should be interpreted cautiously and in the context of other predictors.

Confidence Intervals

Slopes: I am 95% confident at the $\alpha = 0.05$ confidence level that for every 100 square foot increase in above-ground living area, the mean expected sale price in North Ames increases by between 2.63% and 3.87%, in Edwards by between 4.29% and 6.82%, and in Brookside by between 6.07% and 9.09%.

Intercepts: I am 95% confident that for houses with zero square feet of living area, the mean expected sale price in North Ames is between $85,905 and $101,286. In Edwards, it is between $51,426 and $71,111 (22.8% to 44.9% lower than North Ames), and in Brookside, it is between $40,587 and $58,215 (37.6% to 56.5% lower than North Ames). Again, interpreting the sale price for houses with zero square feet of living area should be done cautiously.

Conclusion

In the preceding analysis, I examined the relationship between home sale price and living area within the North Ames, Edwards, and Brookside neighborhoods. The results indicated that this relationship varies by neighborhood. For every 100 square foot increase in above-ground living area, the mean expected sale price increases by 3.25% in North Ames, 5.55% in Edwards, and

7.57% in Brookside. Influential observations were addressed, and model assumptions were verified. A multiple linear regression model with a log transformation of the sale price and interaction terms provided the best fit, enabling generation of estimates and confidence intervals for the relationship between living area and sale price in each neighborhood within the Century 21 Ames market.

R Shiny: Price vs Living Area Chart

https://kdhenderson.shinyapps.io/RegressionOfHousingPricesOnSquareFootage/

Analysis Question 2

Problem

I would like to use linear regression to build a model to predict sale price in all of Ames, Iowa.

Candidate Models

- SLR: $\mu\{\log(\text{Price}) \mid \text{OverallQual}\} = \beta_0 + \beta_1\text{OverallQual}$
- MLR 1: $\mu\{\log(\text{Price}) \mid \text{OverallQual}, \log(\text{GrLivArea}), \text{Neighborhood}, \text{MSSubClass}\} = \beta_0 + \beta_1\text{OverallQual} + \beta_2\log(\text{GrLivArea}) + \beta_*\text{Neighborhood} + \beta_*\text{MSSubClass}$ (* where Neighborhood has 25 categories and MSSubClass has 16 categories)
- MLR 2: $\mu\{\log(\text{Price}) \mid \text{OverallQual}, \log(\text{GrLivArea}), \text{OverallCond}, \text{GarageCars}, \text{BsmtFinSF1}, \text{YearBuilt}, \log(\text{LotArea}), \text{Neighborhood}\} = \beta_0 + \beta_1\text{OverallQual} + \beta_2\log(\text{GrLivArea}) + \beta_3\text{OverallCond} + \beta_4\text{GarageCars} + \beta_5\text{BsmtFinSF1} + \beta_6\text{YearBuilt} + \beta_7\log(\text{LotArea}) + \beta_*\text{Neighborhood}$ (* where Neighborhood has 25 categories)
- MLR 4: $\mu\{\log(\text{Price}) \mid \text{OverallQual}, \log(\text{GrLivArea}), \text{OverallCond}, \text{GarageCars}, \text{BsmtFinSF1}, \text{YearBuilt}, \log(\text{LotArea}), \text{Neighborhood}, \text{MSSubClass}\} = \beta_0 + \beta_1\text{OverallQual} + \beta_2\log(\text{GrLivArea}) + \beta_3\text{OverallCond} + \beta_4\text{GarageCars} + \beta_5\text{BsmtFinSF1} + \beta_6\text{YearBuilt} + \beta_7\log(\text{LotArea}) + \beta_*\text{Neighborhood} + \beta_*\text{MSSubClass}$ (* where Neighborhood has 25 categories and MSSubClass has 16 categories)

Checking Assumptions

Influential point analysis (Cook's D and Leverage)

The studentized residuals and leverage and Cook's D plots for the SLR and MLR1 models (left half of Figure 6) do not show evidence of any influential points. However, MLR2 (center right of Figure 6) has points of concern. There was one observation of basement square footage (BsmtFinSF1) above 5000. I decided to restrict the range of the analysis to houses below this threshold. The residuals for MLR4 benefitted from removing this point. The remaining influential point seemed to result from a basement with over 2000 square feet. I considered 4000 as a cutoff, as only two houses fell above 4000. Additionally, only ten houses in the whole dataset were above 2000. However, the house above 4000 and several of the houses above 2000 were in the test set, and I did not want to eliminate them.

Address Assumptions

1. Linearity: Log transformation of SalePrice, GrLivArea and LotArea resulted in a better linear fit than the untransformed data (Supplementary figure 3).
2. Equal Standard deviation: The residual plots in Figure 6 are randomly scattered with no strong evidence against equal standard deviation.

3. Normality: Prior to log transformation the distributions looked very right-skewed. After log transformation of the above variables, the histograms, QQ plots and random scattering of residuals in Figure 6 are reasonably consistent with a normal distribution. There is some mild left skewness, but I feel comfortable that the sample size will mitigate this.
4. Independence: There is no evidence against independence.

Residual Plots

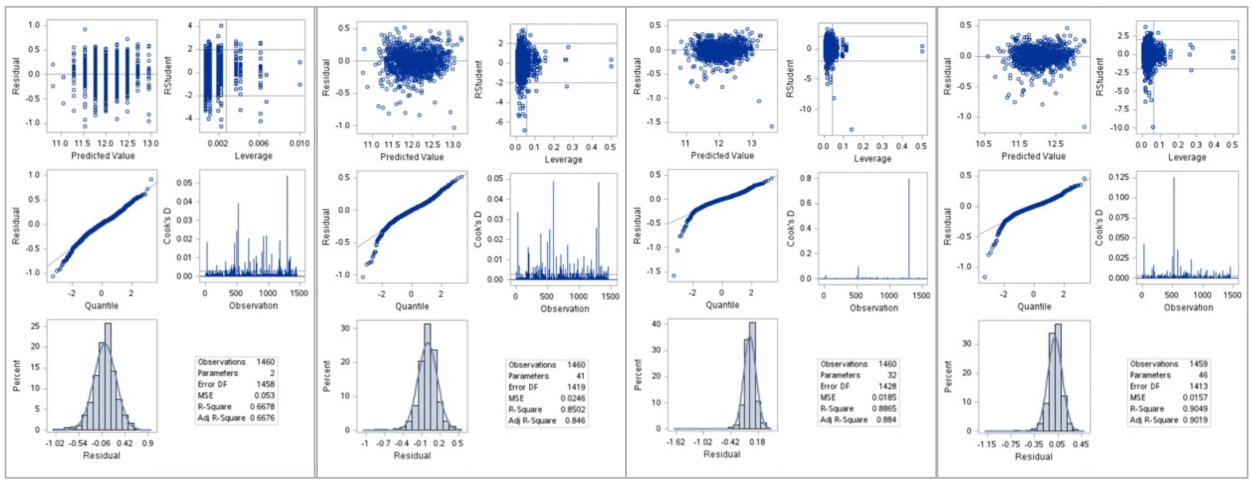


Figure 6: Residual plots for candidate models SLR, MLR1, MLR2, and MLR4 from left to right after log transformation of selected variables. Residuals for MLR4 were plotted after restricting analysis to basement square footages under 5000.

Comparing Competing Models: Adj R^2**,** Internal CV PRESS, AIC, Kaggle Score

Predictive Models	Adjusted R^2	CV PRESS	AIC	Kaggle Score
Simple Linear Regression	.67	76.7	-2838	.229
Multiple Linear Regression 1 (MLR1)	.85	36.5	-3952	.162
MLR2	.90	25.0	-4499	.139
MLR4	.90	24.1	-4558	.136

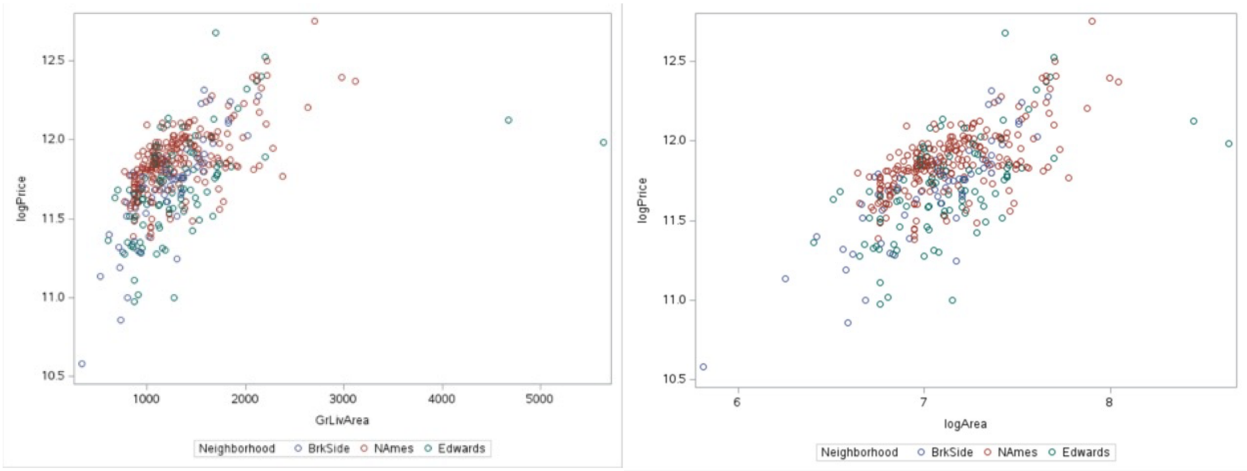
Conclusion

In Analysis 2, I developed several predictive models for home sale prices in Ames, Iowa. Among the candidate models, the Multiple Linear Regression Model 4 (MLR4) performed the best, with a high adjusted R^2 of 0.90, and the lowest CV PRESS, AIC and Kaggle scores. This model included nine predictor variables, two treated categorically and the rest numerically, and employed log transformations of several variables, including the response variable SalePrice. Influential observations and model assumptions were addressed. With one fewer variable (MSSubClass), MLR2 performed almost as well.

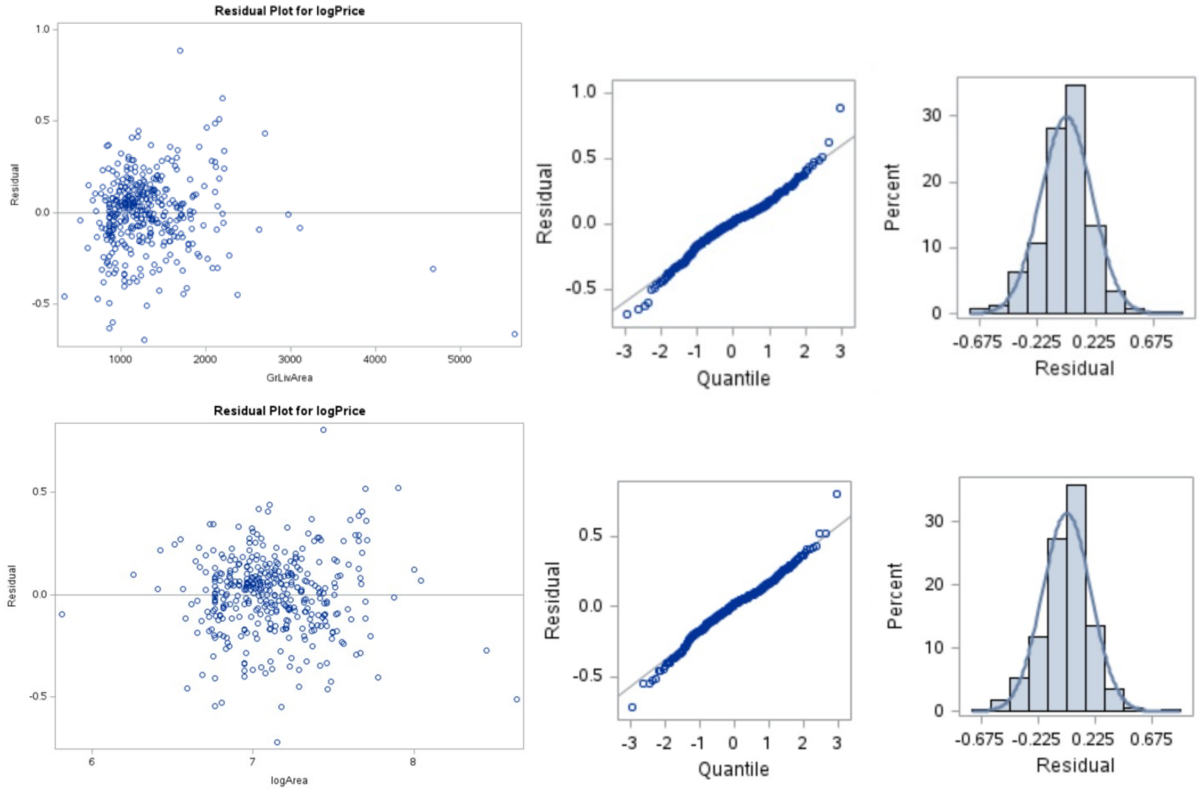
Future improvements could include treating ranked numeric variables as categorical, grouping categories, exploring log transformations on additional variables, and imputing with simpler model predictions rather than median values. Additionally, predicting home prices is highly dependent on the economy and mortgage interest rates, so incorporating these data would benefit future models. MLR4 performed very well and is the best of these candidate models to predict home sale prices in Ames.

Appendix

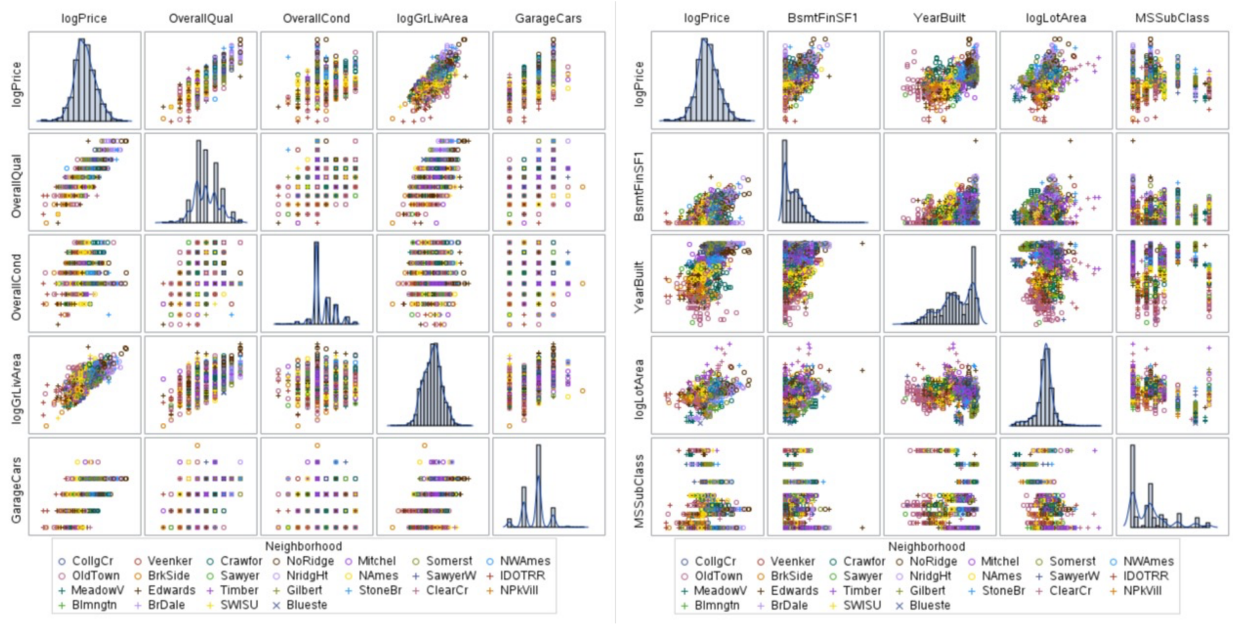
Supplementary Figures for Analysis 1



Supp. Fig. 1: Scatterplot of sale price in dollars and above ground living area in square feet with neighborhoods plotted by color (blue: Brookside, red: North Ames, green: Edwards). Data with log transformation of only sale price are on the left, and data with log transformed price and area are on the right.



Supp. Fig. 2: From left to right, residual plots, QQ plots, and histograms of data with log transformation of sale price (top) and log transformation of sale price and area (bottom).



Supp. Fig. 3: Scatterplots and histograms of variables, some after log transformation, used in candidate models with neighborhoods plotted by color/symbol.

Citations

De Cock, D. 2011. "Ames, Iowa: Alternative to the Boston Housing Data as an End of Semester Regression Project." *Journal of Statistics Education* 19 (3).

Montoya A., DataCanary. 2016. House Prices - Advanced Regression Techniques. Kaggle. https://kaggle.com/competitions/house-prices-advanced-regression-techniques

Code

Analysis 1 (in SAS):

```
/* IMPORT DATA */
/* Import the training dataset, train.csv. */
%web_drop_table(WORK.train);
FILENAME REFFILE '/home/u63864441/sasuser.v94/datasets/train.csv';
PROC IMPORT DATAFILE=REFFILE
        DBMS=CSV
        OUT=WORK.train;
        GETNAMES=YES;
RUN;
PROC CONTENTS DATA=WORK.train; RUN;
%web_open_table(WORK.train);

proc print data = train (obs=50);
run;

/* Import the testing dataset, test.csv. */
%web_drop_table(WORK.test);
```

```
FILENAME REFFILE '/home/u63864441/sasuser.v94/datasets/test.csv';
PROC IMPORT DATAFILE=REFFILE
	DBMS=CSV
	OUT=WORK.test;
	GETNAMES=YES;
RUN;
PROC CONTENTS DATA=WORK.test; RUN;
%web_open_table(WORK.test);

proc print data = test (obs=50);
run;

/* FILTER AND TRANSFORM DATA */
/* Filter for just the 3 neighborhoods with all variables for investigation of unusual observations. */
data neighborhood_lg;
	set train;
	if Neighborhood in ('NAmes', 'Edwards', 'BrkSide');
run;
proc print data = neighborhood_lg;
run;

/* Filter for just the 3 neighborhoods with selected variables of interest and log transformed
variable. */
data neighborhood_sm;
	set train(keep=ID SaleCondition Neighborhood GrLivArea SalePrice);
	if Neighborhood in ('NAmes', 'Edwards', 'BrkSide');
	logPrice = log(SalePrice);
	logArea = log(GrLivArea);
run;
proc print data = neighborhood_sm;
run;

/* SUMMARY STATISTICS AND BASIC PLOTS */
/* Get some summary data. */
proc means data = neighborhood_sm;
run;

/* Plot scatterplots by color. */
proc sgplot data = neighborhood_sm;
	scatter x = GrLivArea y = SalePrice / group = Neighborhood;
run;

proc sgplot data = neighborhood_sm;
	scatter x = GrLivArea y = logPrice / group = Neighborhood;
run;

proc sgplot data = neighborhood_sm;
	scatter x = logArea y = logPrice / group = Neighborhood;
```

```
run;

/* Compare models with and without log transformation and get residual plots. */
* Run the models with all variables;
proc glm data = neighborhood_sm plots = all;
class Neighborhood;
model SalePrice = GrLivArea | Neighborhood / solution;
run;

proc glm data = neighborhood_sm plots = all;
class Neighborhood;
model logPrice = GrLivArea | Neighborhood / solution;
run;

proc glm data = neighborhood_sm plots = all;
class Neighborhood;
model logPrice = logArea | Neighborhood / solution;
run;

/* INFLUENTIAL POINTS */
/* Label influential points in Cook's D and other plots. */
proc reg data = neighborhood_sm plots(label) = (CooksD all);
model SalePrice = GrLivArea;
run;

proc reg data = neighborhood_sm plots(label) = (CooksD all);
model logPrice = GrLivArea;
run;

proc reg data = neighborhood_sm plots(label) = (CooksD all);
model logPrice = logArea;
run;

/* Look at influential points 131 and 339. */
proc sort data = neighborhood_sm;
  by SalePrice;
run;
proc print data = neighborhood_sm;
run;

proc sort data = neighborhood_sm;
  by GrLivArea;
run;
proc print data = neighborhood_sm;
run;

/* How many houses are in Partial SaleCondition or are over 3500 sq.ft? */
proc print data = neighborhood_sm;
```

```
where GrLivArea > 3500
or SaleCondition = 'Partial';
run;

* re-sort by ID;
proc sort data = neighborhood_sm;
   by ID;
run;
proc print data = neighborhood_sm;
run;

/* What about in the full training set? */
proc sort data=train;
   by GrLivArea;
run;
proc print data = train (keep=ID SaleCondition Neighborhood GrLivArea SalePrice);
where GrLivArea > 3500
or SaleCondition = 'Partial';
run;

* re-sort by ID;
proc sort data = train;
   by ID;
run;
proc print data = train (obs=50);
run;

/* Are there common characteristics in influential points other than SaleCondition = Partial. Look
for other variables that might have matching values in obs 131, 339, and others from filtered
dataset. */
data subset;
   set neighborhood_lg;
   if _N_ in (131, 339, 136, 169);
run;
proc print data=subset;
run;

/* Remove SaleCondition = 'Partial' observations. */
data nePartial;
   set neighborhood_sm;
   if SaleCondition ne 'Partial';
run;
proc print data = nePartial;
run;

/* COMPARE MODELS */
/* Get comparison statistics. */
proc glmselect data = nePartial plots = all;
```

```
model logPrice = GrLivArea / selection = stepwise include = 2 stats = press;
run;

proc glmselect data = nePartial plots = all;
class Neighborhood;
model logPrice = GrLivArea Neighborhood / selection = stepwise include = 4 stats = press;
run;

proc glmselect data = nePartial plots = all;
class Neighborhood;
model logPrice = GrLivArea | Neighborhood / selection = stepwise include = 6 stats = press;
run;

proc glmselect data = nePartial plots = all;
model logPrice = logArea / selection = stepwise include = 2 stats = press;
run;

proc glmselect data = nePartial plots = all;
class Neighborhood;
model logPrice = logArea Neighborhood / selection = stepwise include = 4 stats = press;
run;

proc glmselect data = nePartial plots = all;
class Neighborhood;
model logPrice = logArea | Neighborhood / selection = stepwise include = 6 stats = press;
run;

* compare the non-logged model;
proc glmselect data = nePartial plots = all;
class Neighborhood;
model SalePrice = GrLivArea | Neighborhood / selection = stepwise include = 6 stats = press;
run;

/* Fit the simple log-lin model.*/
proc glm data = nePartial plots = all;
model logPrice = GrLivArea / solution clparm;
run;

/* Fit the log-lin model with Neighborhood indicators. */
proc glm data = nePartial plots = all;
class Neighborhood;
model logPrice = GrLivArea Neighborhood / solution clparm;
run;

/* Fit the log-lin model with interaction. */
proc glm data = nePartial plots = all;
class Neighborhood;
model logPrice = GrLivArea | Neighborhood / solution clparm;
```

```
run;

/* Fit the simple log-log model.*/
proc glm data = nePartial plots = all;
model logPrice = logArea / solution clparm;
run;

/* Fit the log-log model with Neighborhood indicators. */
proc glm data = nePartial plots = all;
class Neighborhood;
model logPrice = logArea Neighborhood / solution clparm;
run;

/* Fit the log-log model with interaction. */
proc glm data = nePartial plots = all;
class Neighborhood;
model logPrice = logArea | Neighborhood / solution clparm;
run;
```

Analysis 2 (both in R and SAS):

```
# In R, find variables with lots of missing values.
library(tidyverse)
library(DataExplorer)

# Find missing in training set.
missing = train
str(missing)
#convert empty strings (if any) to NA
missing[missing == ""] = NA

#this finds the number and percent of missing values in each variable
miss_data = missing %>%
 gather(key, value) %>%
 group_by(key) %>%
 count(na = is.na(value)) %>%
 pivot_wider(names_from = na, values_from = n, values_fill = 0) %>%
 mutate(pct_missing = (`TRUE`/sum(`TRUE`, `FALSE`))*100) %>%
 ungroup()
miss_data

#plot the percent missing for PPT
miss_plot = plot_missing(missing)

#visualize what rows have missing values
vis_miss(missing) + theme(axis.text.x = element_text(angle=80))
```

```
# Repeat for testing set
missing = test
str(missing)
#convert empty strings (if any) to NA
missing[missing == ""] = NA

#this finds the number and percent of missing values in each variable
miss_data = missing %>%
 gather(key, value) %>%
 group_by(key) %>%
 count(na = is.na(value)) %>%
 pivot_wider(names_from = na, values_from = n, values_fill = 0) %>%
 mutate(pct_missing = (`TRUE`/sum(`TRUE`, `FALSE`))*100) %>%
 ungroup()
miss_data

#plot the percent missing for PPT
miss_plot = plot_missing(missing)

#visualize what rows have missing values
vis_miss(missing) + theme(axis.text.x = element_text(angle=80))

# Make EDA Plots
# Combine train and test data
test$SalePrice = NA
train_comb = rbind(train, test)

# Select relevant variables and create log transformations
train_comb = train_comb %>%
 select(-Alley, -PoolQC, -Fence, -MiscFeature, -FireplaceQu) %>%
 mutate(logSalePrice = log(SalePrice),
      logGrLivArea = log(GrLivArea))

# Filter numerical variables
numVars = sapply(train_comb, is.numeric)
train_comb_num = train_comb[, numVars]

# Custom function to calculate binwidth
calculate_binwidth = function(data) {
 data = data[!is.na(data)]
 IQR(data) / (2 * (length(data)^(1/3)))
}

# Iterate over numerical variables and create histograms
for (col in names(train_comb_num)) {
 bw = calculate_binwidth(train_comb[[col]])

 hist_plot = train_comb %>%
```

```
   filter(!is.na(.data[[col]])) %>%
   ggplot(aes(x = .data[[col]], y = after_stat(density))) +
   geom_histogram(binwidth = bw, na.rm = TRUE) +
   labs(title = paste("Histogram of", col))

  print(hist_plot)
}

# Filter categorical variables (factors)
train_comb_cat = train_comb %>%
  select(where(is.factor))

# Iterate over categorical variables and create bar plots
for (col in names(train_comb_cat)) {
  bar_plot = train_comb %>%
   ggplot(aes(x = .data[[col]])) +
   geom_bar() +
   labs(title = paste("Bar Chart of", col)) +
   coord_flip()

  print(bar_plot)
}

# Function to create scatter plots for numerical variables
make_scatterplot = function(var1, response) {
  train_comb %>%
   ggplot(aes(x = .data[[var1]], y = .data[[response]])) +
   geom_point(alpha = 0.5, position = "jitter") +
   geom_smooth(method = 'lm') +
   labs(title = paste(var1, "vs", response))
}

# Loop through numerical variables to create scatter plots
for (var1 in names(train_comb_num)) {
  scatter_log = make_scatterplot(var1, "logSalePrice")
  print(scatter_log)
}

# Function to create boxplots for categorical variables
make_boxplot = function(catvar, response) {
  train_comb %>%
   ggplot(aes(x = .data[[catvar]], y = .data[[response]])) +
   geom_boxplot() +
   coord_flip() +
   labs(title = paste(catvar, "vs", response))
}

# Loop through categorical variables to create box plots
```

```
for (catvar in names(train_comb_cat)) {
  boxplt_log = make_boxplot(catvar, "logSalePrice")
  print(boxplt_log)
}

/* Most of the analysis in SAS. */
/*IMPORT, CLEAN AND COMBINE */
/* Standardize the length of LotFrontage in train and drop columns with missing values. */
data train2;
   length LotFrontage $3; /* Set length to 3 */
   set train; * This is the dataset I imported in Analysis 1.;
   drop Alley Fence FireplaceQu MiscFeature PoolQC;
run;

/* Standardize the length of LotFrontage in test and drop columns with missing values. */
data test2;
   length LotFrontage $3; /* Set length to 3 */
   set test; * This is the dataset I imported in Analysis 1.;
   drop Alley Fence FireplaceQu MiscFeature PoolQC;
run;

/* Create a sale price column in the test set. */
data test2;
set test2;
SalePrice = .;

/* Append test set to the training set, which is how Iget the predictions. */
data train_comb;
set train2 test2;
logPrice = log(SalePrice);
run;
proc print data = train_comb (obs=50);
run;

/* USE VARIABLE SELECTION TO NARROW THE LIST OF VARIABLES */
/* Manually create lists of categorical and all variables. I'm sure there is a more efficient way. */

* Categorical variables;
%let class_vars1 = MSSubClass MSZoning LotFrontage Street LotShape LandContour Utilities
LotConfig
 LandSlope Neighborhood Condition1 Condition2 BldgType HouseStyle RoofStyle RoofMatl
Exterior1st
 Exterior2nd MasVnrType ExterQual ExterCond Foundation BsmtQual BsmtCond BsmtExposure
BsmtFinType1
 BsmtFinType2 Heating HeatingQC CentralAir Electrical KitchenQual Functional GarageType
GarageFinish
 GarageQual GarageCond PavedDrive SaleType SaleCondition;
```

```
* All variables;
%let model_vars1 = MSSubClass MSZoning LotFrontage LotArea Street LotShape LandContour
Utilities
 LotConfig LandSlope Neighborhood Condition1 Condition2 BldgType HouseStyle OverallQual
OverallCond
 YearBuilt YearRemodAdd RoofStyle RoofMatl Exterior1st Exterior2nd MasVnrType ExterQual
ExterCond
 Foundation BsmtQual BsmtCond BsmtExposure BsmtFinType1 BsmtFinSF1 BsmtFinType2
BsmtFinSF2 BsmtUnfSF
 TotalBsmtSF Heating HeatingQC CentralAir Electrical LowQualFinSF GrLivArea BsmtFullBath
BsmtHalfBath
 FullBath HalfBath BedroomAbvGr KitchenAbvGr KitchenQual TotRmsAbvGrd Functional Fireplaces
GarageType
 GarageYrBlt GarageFinish GarageCars GarageArea GarageQual GarageCond PavedDrive
WoodDeckSF OpenPorchSF
 EnclosedPorch ScreenPorch PoolArea MiscVal MoSold YrSold SaleType SaleCondition;

/* Use the lists in proc glmselect with several selection methods. */
* Using SalePrice as response;
proc glmselect data=train_comb plots=all;
   class &class_vars1;
   model SalePrice = &model_vars1 / selection = Forward(stop = SL SLE = .2) stats = press;
run;

proc glmselect data=train_comb plots=all;
   class &class_vars1;
   model SalePrice = &model_vars1 / selection = Backward(stop = SL SLS = .2) stats = press;
run;

proc glmselect data=train_comb plots=all;
   class &class_vars1;
   model SalePrice = &model_vars1 / selection = Stepwise(stop = SL SLE = .2 SLS = .2) stats = press;
run;

proc glmselect data=train_comb plots=all;
   class &class_vars1;
   model SalePrice = &model_vars1 / selection = Forward(stop = AIC) stats = press;
run;

proc glmselect data=train_comb plots=all;
   class &class_vars1;
   model SalePrice = &model_vars1 / selection = Backward(stop = AIC) stats = press;
run;

proc glmselect data=train_comb plots=all;
   class &class_vars1;
   model SalePrice = &model_vars1 / selection = Stepwise(stop = AIC) stats = press;
run;
```

```
* Using logPrice as response;
proc glmselect data=train_comb plots=all;
  class &class_vars1;
  model logPrice = &model_vars1 / selection = Forward(stop = SL SLE = .2) stats = press;
run;

proc glmselect data=train_comb plots=all;
  class &class_vars1;
  model logPrice = &model_vars1 / selection = Backward(stop = SL SLS = .2) stats = press;
run;

proc glmselect data=train_comb plots=all;
  class &class_vars1;
  model logPrice = &model_vars1 / selection = Stepwise(stop = SL SLE = .2 SLS = .2) stats = press;
run;

proc glmselect data=train_comb plots=all;
  class &class_vars1;
  model logPrice = &model_vars1 / selection = Forward(stop = AIC) stats = press;
run;

proc glmselect data=train_comb plots=all;
  class &class_vars1;
  model logPrice = &model_vars1 / selection = Backward(stop = AIC) stats = press;
run;

proc glmselect data=train_comb plots=all;
  class &class_vars1;
  model logPrice = &model_vars1 / selection = Stepwise(stop = AIC) stats = press;
run;

/* Reduce list to variables that appear in 6+ of the 12 models above.
(This is where I remove MSSubClass, which ends up in the best performing final model.) */

* Categorical variables;
%let class_vars2 = MSZoning Street LandSlope Neighborhood Condition2 BldgType
 RoofMatl ExterQual BsmtQual BsmtExposure KitchenQual SaleCondition;

* All variables;
%let model_vars2 = LotArea Neighborhood Condition2 BldgType OverallQual OverallCond
 YearBuilt RoofMatl BsmtFinSF1 TotalBsmtSF GrLivArea BedroomAbvGr KitchenAbvGr
 KitchenQual GarageCars GarageArea Street SaleCondition LandSlope ScreenPorch MSZoning
 PoolArea BsmtQual BsmtExposure ExterQual LowQualFinSF FullBath;

* Numerical variables;
%let num_vars2 = LotArea OverallQual OverallCond YearBuilt BsmtFinSF1 TotalBsmtSF
 GrLivArea BedroomAbvGr KitchenAbvGr GarageCars GarageArea ScreenPorch PoolArea
```

```
 LowQualFinSF FullBath;

/* PLOT */
/* Plot the numerical variables with logPrice. */
proc sgscatter data = train_comb;
matrix SalePrice logPrice LotArea OverallQual OverallCond TotalBsmtSF BsmtFinSF1 /
diagonal=(histogram kernel);
run;

proc sgscatter data = train_comb;
matrix SalePrice logPrice YearBuilt KitchenAbvGr BedroomAbvGr GarageCars GarageArea /
diagonal=(histogram kernel);
run;

proc sgscatter data = train_comb;
matrix SalePrice logPrice ScreenPorch PoolArea LowQualFinSF FullBath GrLivArea /
diagonal=(histogram kernel);
run;

/* Look at some correlations. */
proc corr data = train_comb;
var SalePrice logPrice &num_vars2;
run;

/* Plot a correlation heatmap (more useful). */
/* Step 1: Compute the correlation matrix and save the results */
proc corr data=train_comb outp=corr_matrix noprint;
   var SalePrice logPrice &num_vars2; /* List your variables here */
run;

/* Step 2: Prepare the data for the heatmap */
data corr_matrix;
   set corr_matrix;
   if _TYPE_ = 'CORR'; /* Keep only the correlation coefficients */
run;

/* Step 3: Prepare data for heatmap */
data corr_matrix_long;
   set corr_matrix;
   array vars {*} _NUMERIC_;
   do i = 1 to dim(vars);
      if _NAME_ ne vname(vars[i]) then do;
         x = _NAME_;
         y = vname(vars[i]);
         corr_value = vars[i];
         output;
      end;
   end;
```

```
   keep x y corr_value;
run;

/* Step 4: Create the heatmap template */
proc template;
   define statgraph corr_heatmap;
      begingraph;
         entrytitle "Correlation Matrix Heatmap";
         layout overlay / xaxisopts=(type=discrete) yaxisopts=(type=discrete reverse=true);
            heatmapparm x=x y=y colorresponse=corr_value / colormodel=(blue white red)
name='heatmap';
            continuouslegend 'heatmap' / orient=vertical location=outside title="Correlation";
         endlayout;
      endgraph;
   end;
run;

/* Step 5: Create the heatmap with PROC SGPLOT */
proc sgrender data=corr_matrix_long template=corr_heatmap;
run;

/* Here is a smaller list. */
* Categorical variables;
%let class_vars2 = MSZoning Street LandSlope Neighborhood Condition2 BldgType
 RoofMatl ExterQual BsmtQual BsmtExposure KitchenQual SaleCondition;

* All variables;
%let model_vars3 = MSZoning Street LandSlope Neighborhood Condition2 BldgType
 RoofMatl ExterQual BsmtQual BsmtExposure KitchenQual SaleCondition OverallQual GrLivArea
GarageCars TotalBsmtSF BsmtFinSF1 YearBuilt
 LotArea KitchenAbvGr BedroomAbvGr ScreenPorch PoolArea
 LowQualFinSF OverallCond;

/* Numerical variables
remove GarageArea, highly correlated with GarageCars
remove FullBath, highly correlated with GrLivArea */
%let num_vars3 = OverallQual GrLivArea GarageCars TotalBsmtSF BsmtFinSF1 YearBuilt
 LotArea KitchenAbvGr BedroomAbvGr ScreenPorch PoolArea
 LowQualFinSF OverallCond;


/* MORE SELECTION WITH LOGPRICE, FEWER VARIABLES AND CROSS VALIDATION */
proc glmselect data = train_comb plots = all;
partition fraction(test = .2);
   class &class_vars2;
   model logPrice = &model_vars3 / selection = Forward(stop = CV) cvmethod = random(5) stats =
adjrsq CVDETAILS;
run;
```

```
proc glmselect data = train_comb plots = all;
partition fraction(test = .2);
   class &class_vars2;
   model logPrice = &model_vars3 / selection = Backward(stop = CV) cvmethod = random(5) stats =
adjrsq CVDETAILS;
run;

proc glmselect data = train_comb plots = all;
partition fraction(test = .2);
   class &class_vars2;
   model logPrice = &model_vars3 / selection = Stepwise(stop = CV) cvmethod = random(5) stats =
adjrsq CVDETAILS;
run;

/* Go back to the scatterplots and log transform some variables
 to see if Ican achieve similar metrics with a smaller model. */
proc print data = train_comb (obs=20);
var &num_vars3;
run;

data train_comb2;
set train_comb;
logLotArea = log(LotArea);
logGrLivArea = log(GrLivArea);
logTotalBsmtSF = log(TotalBsmtSF);
logBsmtFinSF1 = log(BsmtFinSF1);
run;

* Categorical variables;
%let class_vars2 = MSZoning Street LandSlope Neighborhood Condition2
 RoofMatl BsmtQual BsmtExposure KitchenQual SaleCondition;
%let class_vars3 = MSZoning Neighborhood Condition2
 RoofMatl BsmtQual BsmtExposure KitchenQual SaleCondition;

* All variables;
%let model_vars4 = MSZoning Street LandSlope Neighborhood Condition2
 RoofMatl BsmtQual BsmtExposure KitchenQual SaleCondition OverallQual logGrLivArea
GarageCars TotalBsmtSF logTotalBsmtSF BsmtFinSF1 YearBuilt logLotArea KitchenAbvGr
BedroomAbvGr ScreenPorch PoolArea
 LowQualFinSF OverallCond;
%let model_vars5 = MSZoning Neighborhood Condition2
 RoofMatl BsmtQual BsmtExposure KitchenQual SaleCondition OverallQual logGrLivArea
GarageCars TotalBsmtSF logTotalBsmtSF BsmtFinSF1 YearBuilt logLotArea KitchenAbvGr
BedroomAbvGr ScreenPorch PoolArea
 LowQualFinSF OverallCond;

/* Numerical variables
```

```
replace LotArea and GrLivArea with logLotArea and logGrLivArea
but take out logBsmtFinSF1*/
%let num_vars4 = OverallQual logGrLivArea GarageCars TotalBsmtSF logTotalBsmtSF BsmtFinSF1
YearBuilt
 logLotArea KitchenAbvGr BedroomAbvGr ScreenPorch PoolArea
 LowQualFinSF OverallCond;

* fewer variables;
%let sm_vars = OverallQual OverallCond logGrLivArea GarageCars BsmtFinSF1 YearBuilt
logLotArea Neighborhood;

/* Trying to find a parsimonious model. */
* Test 8 variable list. Maybe try interactions with a few later;
proc glmselect data=train_comb2 plots=all;
  class Neighborhood;
  model logPrice = &sm_vars / selection = Forward(stop = SL SLE = .1) stats = press;
run;

proc glmselect data=train_comb2 plots=all;
  class Neighborhood;
  model logPrice = &sm_vars / selection = Backward(stop = SL SLS = .1) stats = press;
run;

proc glmselect data=train_comb2 plots=all;
  class Neighborhood;
  model logPrice = &sm_vars / selection = Stepwise(stop = SL SLE = .1 SLS = .1) stats = press;
run;

* Test more categorical variables;
proc glmselect data=train_comb2 plots=all;
  class &class_vars2;
  model logPrice = &model_vars4 / selection = Forward(stop = SL SLE = .1) stats = press;
run;

proc glmselect data=train_comb2 plots=all;
  class &class_vars2;
  model logPrice = &model_vars4 / selection = Backward(stop = SL SLS = .1) stats = press;
run;

proc glmselect data=train_comb2 plots=all;
  class &class_vars2;
  model logPrice = &model_vars4 / selection = Stepwise(stop = SL SLE = .1 SLS = .1) stats = press;
run;

* Test fewer categorical variables;
proc glmselect data=train_comb2 plots=all;
  class &class_vars3;
  model logPrice = &model_vars5 / selection = Forward(stop = SL SLE = .1) stats = press;
```

```
run;

proc glmselect data=train_comb2 plots=all;
  class &class_vars3;
  model logPrice = &model_vars5 / selection = Backward(stop = SL SLS = .1) stats = press;
run;

proc glmselect data=train_comb2 plots=all;
  class &class_vars3;
  model logPrice = &model_vars5 / selection = Stepwise(stop = SL SLE = .1 SLS = .1) stats = press;
run;

/* Force number of variables */
proc glmselect data=train_comb2 plots=all;
  class &class_vars1;
  model logPrice = &model_vars1 logLotArea logGrLivArea logTotalBsmtSF / selection =
Stepwise(stop = 2 select = CV) stats = adjrsq;
run;

proc glmselect data=train_comb2 plots=all;
  class &class_vars1;
  model logPrice = &model_vars1 logLotArea logGrLivArea logTotalBsmtSF / selection =
Stepwise(stop = 3 select = CV) stats = adjrsq;
run;

proc glmselect data=train_comb2 plots=all;
  class &class_vars1;
  model logPrice = &model_vars1 logLotArea logGrLivArea logTotalBsmtSF / selection =
Stepwise(stop = 4 select = CV) stats = adjrsq;
run;

proc glmselect data=train_comb2 plots=all;
  class &class_vars1;
  model logPrice = &model_vars1 logLotArea logGrLivArea logTotalBsmtSF / selection =
Stepwise(stop = 5 select = CV) stats = adjrsq;
run;

proc glmselect data=train_comb2 plots=all;
  class &class_vars1;
  model logPrice = &model_vars1 logLotArea logGrLivArea logTotalBsmtSF / selection =
Stepwise(stop = 6 select = CV) stats = adjrsq;
run;

proc glmselect data=train_comb2 plots=all;
  class &class_vars1;
  model logPrice = &model_vars1 logLotArea logGrLivArea logTotalBsmtSF / selection =
Stepwise(stop = 7 select = CV) stats = adjrsq;
run;
```

```
proc glmselect data=train_comb2 plots=all;
   class &class_vars1;
   model logPrice = &model_vars1 logLotArea logGrLivArea logTotalBsmtSF / selection =
Stepwise(stop = 8 select = CV) stats = adjrsq;
run;

proc glmselect data=train_comb2 plots=all;
   class &class_vars1;
   model logPrice = &model_vars1 logLotArea logGrLivArea logTotalBsmtSF / selection =
Stepwise(stop = 9 select = CV) stats = adjrsq;
run;

/* CANDIDATE MODELS */
/* SLR: logPrice = OverallQual
MLR1: logPrice = OverallQual logGrLivArea Neighborhood
MLR2: logPrice = OverallQual | logGrLivArea | Neighborhood
MLR3: logPrice = OverallQual logGrLivArea Neighborhood MSSubClass
MLR4: logPrice = OverallQual OverallCond logGrLivArea GarageCars BsmtFinSF1 YearBuilt
logLotArea Neighborhood
MLR5: logPrice = MSZoning Neighborhood Condition2 RoofMatl BsmtQual BsmtExposure
KitchenQual SaleCondition OverallQual logGrLivArea
GarageCars logTotalBsmtSF BsmtFinSF1 YearBuilt logLotArea KitchenAbvGr BedroomAbvGr
ScreenPorch PoolArea OverallCond */

/* Run the models */
proc glm data = train_comb2 plots = all;
model logPrice = OverallQual / solution clparm;
run;

proc glm data = train_comb2 plots = all;
class Neighborhood;
model logPrice = OverallQual logGrLivArea Neighborhood / solution clparm;
run;

proc glm data = train_comb2 plots = all;
class Neighborhood MSSubClass;
model logPrice = OverallQual logGrLivArea Neighborhood MSSubClass / solution clparm;
run;

proc glm data = train_comb2 plots = all;
class Neighborhood;
model logPrice = OverallQual OverallCond logGrLivArea GarageCars BsmtFinSF1 YearBuilt
logLotArea Neighborhood / solution clparm;
run;

proc glm data = train_comb2 plots = all;
```

```
class Neighborhood MSZoning Condition2 RoofMatl BsmtQual BsmtExposure KitchenQual
SaleCondition;
model logPrice = MSZoning Neighborhood Condition2 RoofMatl BsmtQual BsmtExposure
KitchenQual SaleCondition OverallQual logGrLivArea
GarageCars logTotalBsmtSF BsmtFinSF1 YearBuilt logLotArea KitchenAbvGr BedroomAbvGr
ScreenPorch PoolArea OverallCond
/ solution clparm;
run;

/* Get statistics */
proc glmselect data = train_comb2 plots = all;
model logPrice = OverallQual / Selection = Stepwise(stop = CV) cvmethod = random(5) stats =
adjrsq CVDETAILS include = 1;
run;

proc glmselect data = train_comb2 plots = all;
class Neighborhood;
model logPrice = OverallQual logGrLivArea Neighborhood / Selection = Stepwise(stop = CV)
cvmethod = random(5) stats = adjrsq CVDETAILS include = 3;
run;

proc glmselect data = train_comb2 plots = all;
class Neighborhood MSSubClass;
model logPrice = OverallQual logGrLivArea Neighborhood MSSubClass / Selection = Stepwise(stop
= CV) cvmethod = random(5) stats = adjrsq CVDETAILS include = 4;
run;

proc glmselect data = train_comb2 plots = all;
class Neighborhood;
model logPrice = OverallQual OverallCond logGrLivArea GarageCars BsmtFinSF1 YearBuilt
logLotArea Neighborhood / Selection = Stepwise(stop = CV) cvmethod = random(5) stats = adjrsq
CVDETAILS include = 8;
run;

proc glmselect data = train_comb2 plots = all;
class Neighborhood MSZoning Condition2 RoofMatl BsmtQual BsmtExposure KitchenQual
SaleCondition;
model logPrice = MSZoning Neighborhood Condition2 RoofMatl BsmtQual BsmtExposure
KitchenQual SaleCondition OverallQual logGrLivArea
GarageCars logTotalBsmtSF BsmtFinSF1 YearBuilt logLotArea KitchenAbvGr BedroomAbvGr
ScreenPorch PoolArea OverallCond
/ Selection = Stepwise(stop = CV) cvmethod = random(5) stats = adjrsq CVDETAILS;
run;

/* FIND INFLUENTIAL POINTS */
proc reg data = train_comb2 plots(label) = (CooksD all);
model logPrice = OverallQual OverallCond logGrLivArea GarageCars BsmtFinSF1 YearBuilt
logLotArea;
```

```
run;

data influentialpts;
set train_comb2;
run;
proc sort data = influentialpts;
   by BsmtFinSF1;
run;
proc print data = influentialpts;
run;

/* Restrict model to basement square footages < 5000 sq ft. There is only 1 above (ID 1299).
I would restrict to < 4000, but it is in the testing set.
ID 524 is also influential I suspect for the basement variables, but I don't think it will
have much effect with all the other variables. There are only 10 obs > 2000 sqft but several
are in the test set. */
data train_comb_filt;
set train_comb2;
where BsmtFinSF1 < 5000;
run;

proc glm data = train_comb_filt plots = all;
class Neighborhood;
model logPrice = OverallQual OverallCond logGrLivArea GarageCars BsmtFinSF1 YearBuilt
logLotArea Neighborhood / solution clparm;
run;

proc glm data = train_comb_filt plots = all;
class Neighborhood MSZoning Condition2 RoofMatl BsmtQual BsmtExposure KitchenQual
SaleCondition;
model logPrice = MSZoning Neighborhood Condition2 RoofMatl BsmtQual BsmtExposure
KitchenQual SaleCondition OverallQual logGrLivArea
GarageCars logTotalBsmtSF BsmtFinSF1 YearBuilt logLotArea KitchenAbvGr BedroomAbvGr
ScreenPorch PoolArea OverallCond
/ solution clparm;
run;

proc reg data = train_comb_filt plots(label) = (CooksD all);
model logPrice = OverallQual logGrLivArea
GarageCars logTotalBsmtSF BsmtFinSF1 YearBuilt logLotArea KitchenAbvGr BedroomAbvGr
ScreenPorch PoolArea OverallCond;
run;

/* MAKE SCATTERPLOTS FOR CANDIDATE MODELS */
* SLR;
proc sgscatter data = train_comb_filt;
matrix logPrice OverallQual / diagonal=(histogram kernel);
run;
```

```
* MLR1 with separate plots for categorical variables;
proc sgscatter data = train_comb_filt;
matrix logPrice OverallQual logGrLivArea / diagonal=(histogram kernel) group = Neighborhood;
run;

proc sgscatter data = train_comb_filt;
matrix logPrice OverallQual logGrLivArea / diagonal=(histogram kernel) group = MSSubClass;
run;

* MLR2;
proc sgscatter data = train_comb_filt;
matrix logPrice OverallQual OverallCond logGrLivArea GarageCars BsmtFinSF1 YearBuilt
logLotArea / diagonal=(histogram kernel) group = Neighborhood;
run;
proc sgscatter data = train_comb_filt;
matrix logPrice OverallQual OverallCond logGrLivArea GarageCars / diagonal=(histogram kernel)
group = Neighborhood;
run;
proc sgscatter data = train_comb_filt;
matrix logPrice BsmtFinSF1 YearBuilt logLotArea / diagonal=(histogram kernel) group =
Neighborhood;
run;

* MLR3: numerical variables only;
proc sgscatter data = train_comb_filt;
matrix logPrice OverallQual logGrLivArea GarageCars logTotalBsmtSF BsmtFinSF1 YearBuilt/
diagonal=(histogram kernel);
run;
proc sgscatter data = train_comb_filt;
matrix logPrice logLotArea KitchenAbvGr BedroomAbvGr ScreenPorch PoolArea OverallCond /
diagonal=(histogram kernel);
run;

/* GET MODEL STATISTICS */
proc glmselect data = train_comb_filt plots = all;
model logPrice = OverallQual / Selection = Stepwise(stop = CV) cvmethod = random(5) stats =
adjrsq CVDETAILS include = 1;
run;

proc glmselect data = train_comb_filt plots = all;
class Neighborhood MSSubClass;
model logPrice = OverallQual logGrLivArea Neighborhood MSSubClass / Selection = Stepwise(stop
= CV) cvmethod = random(5) stats = adjrsq CVDETAILS include = 4;
run;

proc glmselect data = train_comb_filt plots = all;
class Neighborhood;
```

```
model logPrice = OverallQual OverallCond logGrLivArea GarageCars BsmtFinSF1 YearBuilt
logLotArea Neighborhood / Selection = Stepwise(stop = CV) cvmethod = random(5) stats = adjrsq
CVDETAILS include = 8;
run;

proc glmselect data = train_comb_filt plots = all;
class Neighborhood MSZoning Condition2 RoofMatl BsmtQual BsmtExposure KitchenQual
SaleCondition;
model logPrice = MSZoning Neighborhood Condition2 RoofMatl BsmtQual BsmtExposure
KitchenQual SaleCondition OverallQual logGrLivArea
GarageCars logTotalBsmtSF BsmtFinSF1 YearBuilt logLotArea KitchenAbvGr BedroomAbvGr
ScreenPorch PoolArea OverallCond
/ Selection = Stepwise(stop = CV) cvmethod = random(5) stats = adjrsq CVDETAILS include = 20;
run;

/* RUN MODELS FOR PREDICTIONS AND BACKTRANSFORM -> TO KAGGLE */
* SLR;
proc glm data=train_comb_filt plots=all;
  model logPrice = OverallQual / solution clparm clm cli;
  output out=resultsSLR p=Predict;
run;

data resultsSLR_preds;
  set resultsSLR;
  PricePreds = exp(Predict);
run;

* MLR1;
proc glm data = train_comb_filt plots = all;
class Neighborhood MSSubClass;
model logPrice = OverallQual logGrLivArea Neighborhood MSSubClass / solution clparm clm cli;
output out = resultsMLR1 p = Predict;
run;

data resultsMLR1_preds;
  set resultsMLR1;
  PricePreds = exp(Predict);
run;

* MLR2;
proc glm data = train_comb_filt plots = all;
class Neighborhood;
model logPrice = OverallQual OverallCond logGrLivArea GarageCars BsmtFinSF1 YearBuilt
logLotArea Neighborhood / solution clparm clm cli;
output out = resultsMLR2 p = Predict;
run;

data resultsMLR2_preds;
```

```
  set resultsMLR2;
  PricePreds = exp(Predict);
run;

* MLR3;
proc glm data = train_comb_filt plots = all;
class Neighborhood MSZoning Condition2 RoofMatl BsmtQual BsmtExposure KitchenQual
SaleCondition;
model logPrice = MSZoning Neighborhood Condition2 RoofMatl BsmtQual BsmtExposure
KitchenQual SaleCondition OverallQual logGrLivArea
GarageCars logTotalBsmtSF BsmtFinSF1 YearBuilt logLotArea KitchenAbvGr BedroomAbvGr
ScreenPorch PoolArea OverallCond
/ solution clparm clm cli;
output out = resultsMLR3 p = Predict;
run;

data resultsMLR3_preds;
  set resultsMLR3;
  PricePreds = exp(Predict);
run;

/* REPLACE NEGATIVE AND MISSING VALUES AND FORMAT WITH 2 COLUMNS FOR KAGGLE */
/* Can't have neg predictions b/c of RMSE, and need to fill in missing values
Also must have only 2 columns with appropriate labels. */
proc means data=resultsSLR_preds noprint;
  var PricePreds;
  output out=median_value median=median_PricePreds;
run;
data resultsSLR_preds;
  if _N_ = 1 then set median_value;  /* Load the median value into a data step variable */
  set resultsSLR_preds;
  if PricePreds > 0 then SalePrice = PricePreds;
  else SalePrice = median_PricePreds;
  keep ID SalePrice; /* Keeps just the id and SalePrice columns */
  where ID > 1460;
run;

proc means data=resultsMLR1_preds noprint;
  var PricePreds;
  output out=median_value median=median_PricePreds;
run;
data resultsMLR1_preds;
  if _N_ = 1 then set median_value;  /* Load the median value into a data step variable */
  set resultsMLR1_preds;
  if PricePreds > 0 then SalePrice = PricePreds;
  else SalePrice = median_PricePreds;
  keep ID SalePrice; /* Keeps just the id and SalePrice columns */
  where ID > 1460;
```

```
run;

proc means data=resultsMLR2_preds noprint;
   var PricePreds;
   output out=median_value median=median_PricePreds;
run;
data resultsMLR2_preds;
   if _N_ = 1 then set median_value;  /* Load the median value into a data step variable */
   set resultsMLR2_preds;
   if PricePreds > 0 then SalePrice = PricePreds;
   else SalePrice = median_PricePreds;
   keep ID SalePrice; /* Keeps just the id and SalePrice columns */
   where ID > 1460;
run;

proc means data=resultsMLR3_preds noprint;
   var PricePreds;
   output out=median_value median=median_PricePreds;
run;
data resultsMLR3_preds;
   if _N_ = 1 then set median_value;  /* Load the median value into a data step variable */
   set resultsMLR3_preds;
   if PricePreds > 0 then SalePrice = PricePreds;
   else SalePrice = median_PricePreds;
   keep ID SalePrice; /* Keeps just the id and SalePrice columns */
   where ID > 1460;
run;

/* EXPORT THE PREDICTION DATASETS TO CSV FILES */
proc export data=resultsSLR_preds
  outfile="~/reports/khenderson_housing_SLR.csv"
  dbms=csv
  replace;
run;

proc export data=resultsMLR1_preds
  outfile="~/reports/khenderson_housing_MLR1.csv"
  dbms=csv
  replace;
run;

proc export data=resultsMLR2_preds
  outfile="~/reports/khenderson_housing_MLR2.csv"
  dbms=csv
  replace;
run;

proc export data=resultsMLR3_preds
```

```
  outfile="~/reports/khenderson_housing_MLR3.csv"
  dbms=csv
  replace;
run;

/* DECIDED TO TRY BEST MODEL AND MSSUBCLASS (BETTER KAGGLE SCORE) */
* MLR4 (MLR2 + MSSubClass);
proc glm data = train_comb_filt plots = all;
class Neighborhood MSSubClass;
model logPrice = OverallQual OverallCond logGrLivArea GarageCars BsmtFinSF1 YearBuilt
logLotArea Neighborhood MSSubClass/ solution clparm clm cli;
output out = resultsMLR4 p = Predict;
run;

data resultsMLR4_preds;
   set resultsMLR4;
   PricePreds = exp(Predict);
run;

proc means data=resultsMLR4_preds noprint;
   var PricePreds;
   output out=median_value median=median_PricePreds;
run;
data resultsMLR4_preds;
   if _N_ = 1 then set median_value;  /* Load the median value into a data step variable */
   set resultsMLR4_preds;
   if PricePreds > 0 then SalePrice = PricePreds;
   else SalePrice = median_PricePreds;
   keep ID SalePrice; /* Keeps just the id and SalePrice columns */
   where ID > 1460;
run;

proc export data=resultsMLR4_preds
  outfile="~/reports/khenderson_housing_MLR4.csv"
  dbms=csv
  replace;
run;

* Scatterplots for this model;
* MLR4;
proc sgscatter data = train_comb_filt;
matrix logPrice OverallQual OverallCond logGrLivArea GarageCars BsmtFinSF1 YearBuilt
logLotArea / diagonal=(histogram kernel) group = Neighborhood;
run;
proc sgscatter data = train_comb_filt;
matrix logPrice OverallQual OverallCond logGrLivArea GarageCars / diagonal=(histogram kernel)
group = Neighborhood;
run;
```

```
proc sgscatter data = train_comb_filt;
matrix logPrice BsmtFinSF1 YearBuilt logLotArea / diagonal=(histogram kernel) group =
Neighborhood;
run;
proc sgscatter data = train_comb_filt;
matrix logPrice OverallQual OverallCond logGrLivArea GarageCars BsmtFinSF1 YearBuilt
logLotArea / diagonal=(histogram kernel) group = MSSubClass;
run;
proc sgscatter data = train_comb_filt;
matrix logPrice OverallQual OverallCond logGrLivArea GarageCars / diagonal=(histogram kernel)
group = MSSubClass;
run;
proc sgscatter data = train_comb_filt;
matrix logPrice BsmtFinSF1 YearBuilt logLotArea / diagonal=(histogram kernel) group =
MSSubClass;
run;

* Statistics for this model;
proc glmselect data = train_comb_filt plots = all;
class Neighborhood MSSubClass;
model logPrice = OverallQual OverallCond logGrLivArea GarageCars BsmtFinSF1 YearBuilt
logLotArea Neighborhood MSSubClass/ Selection = Stepwise(stop = CV) cvmethod = random(5)
stats = adjrsq CVDETAILS include = 9;
run;
/* MLR4b: I realized that MSSubClass is actually numerical.
Model performs better with it treated as categorical.
I should go back and look more closely at categorical number variables,
and potentially group categories. */
proc glmselect data = train_comb_filt plots = all;
class Neighborhood;
model logPrice = OverallQual OverallCond logGrLivArea GarageCars BsmtFinSF1 YearBuilt
logLotArea Neighborhood MSSubClass/ Selection = Stepwise(stop = CV) cvmethod = random(5)
stats = adjrsq CVDETAILS include = 9;
run;
```