Exploratory Data Analysis and Regression Models of the Ames Housing Dataset

Prepared by Amee Amin

Assignment 3

MSPA Course 410, Summer 2017

Professors Chad R. Bhatti and Tom Miller

Introduction

In this report we explore building a predictive model for the sale price of single-family residences in Ames, Iowa, using the Ames housing dataset. We used both simple and multiple regression models, and performed a log transformation of the response variable. The applicability of predicting property sale prices is widespread in today's market. For example, many consumers rely on online estimates of properties, such as Zillow's zestimate, to inform both their buying and selling of real estate. We focused on variables that consumers commonly ask about when looking for a home, such as the square footage. In previous analyses, we completed a data quality check on such variables, and an exploratory data analysis showed that the total living area above grade may be a good predictor of sale price. This analysis found that the total living area above grade, year built, and mean price per square foot by neighborhood have a positive correlation with SalePrice. A multiple linear regression with a log transformation of SalePrice produced the best model for predicting SalePrice. Variability in SalePrice by neighborhood needs further exploration, and the elimination of outliers and influential variables in each neighborhood could improve future models.

Section 1: Sample Definition

Section 1.1: Dataset Overview

The dataset explored in this analysis contains information from the Ames Assessorís Office used in computing assessed values for individual residential properties sold in Ames, IA from 2006 to 2010. This dataset is intended for use by students in learning regression analysis, and therefore "laymen's" variables were chosen for inclusion, meaning variables that could be reasonably understood by those completing the analysis. There are 2930 observations in the dataset, including 23 nominal variables, 23 ordinal variables, 14 discrete variables, and 20 continuous variables involved in assessing home values. In total, the 80 variables represent the quality and quantity of physical attributes that a typical home buyer would want to know about a potential property. For example, attributes address the following questions: When was the home built? How many square feet is the property? How many bathrooms are there?

Because the Ames housing dataset includes properties across a wide range of variables, we will have to define a subset of properties to focus on when predicting sale price. For example, a model to predict the commercial housing market would different than the residential housing market. In this analysis we are interested in the sale price of a typical single family residence sold in Ames, Iowa. We will need to be careful in identifying outliers and collinearity among variables.

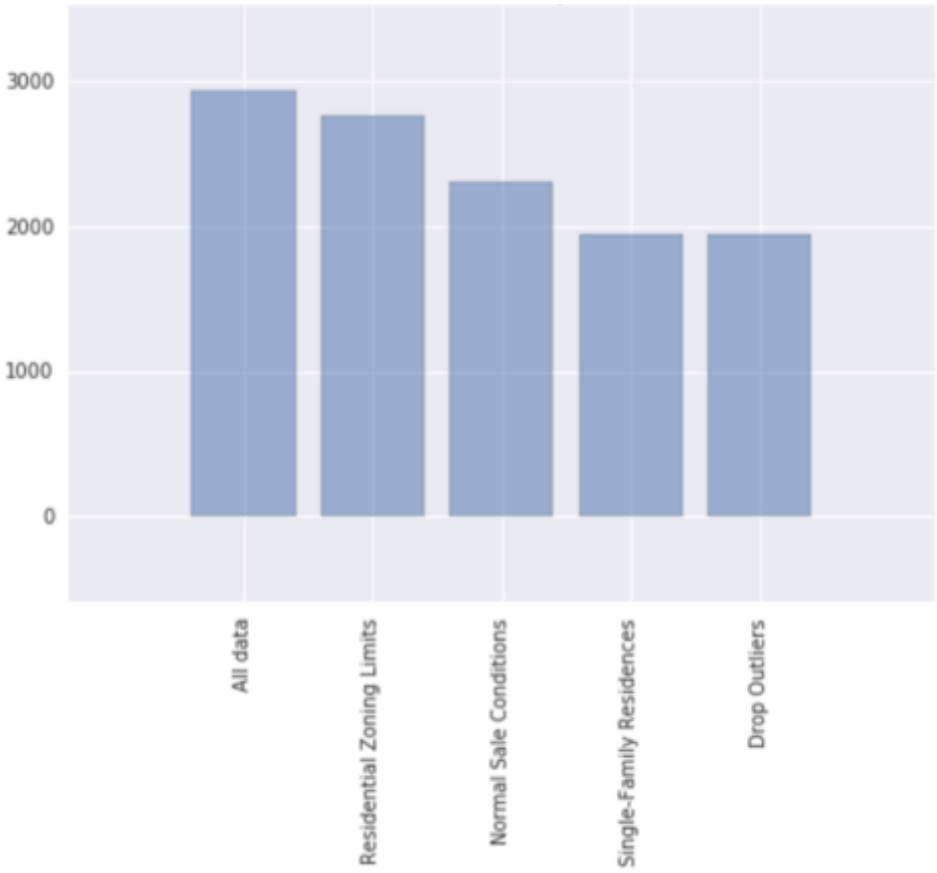
Section 1.2: Sample Overview

This exploratory data analysis will look at SalePrice as the response variable, which indicates the price at which the property was sold on the market. We will focus on predicting the SalePrice of Single-Family detached residential homes.

To properly predict SalePrice, we will limit the model to homes identified within residential zoning limits, to avoid a comparison between commercial and industrial prices to

residential prices, and to homes categorized as "Normal" Sale Conditions. We will then define our sample to focus on single-family residences. We looked at a quick boxplot of SalePrice and found that there appears to be extreme outliers above $700,000. We narrowed the dataset to exclude homes that have a SalePrice of more than $700,000. As noted by the original author of the dataset, these homes are more than 4000 total square feet and should be removed as outliers in order to predict typical housing sale prices. Because there may still be an issue with non-homogenous variance, as home size increases, the model may be more accurate within a range of small to medium-sized homes.

Fig 1: Waterfall chart of drop conditions



Above is a waterfall chart that demonstrates our sample size using the drop conditions described above. We started with 2930 observations in the dataset, narrowed to 2762 observations when looking at properties within residential zoning limits, narrowed to 2305 observations when looking at properties with normal sale conditions, narrowed to 1943 observations when looking at single-family residences, and narrowed to 1942 homes after dropping outliers.

Section 2: Exploratory Data Analysis

Section 2.1: Feature Selection

After checking twenty variables for data quality, we chose ten variables to perform an exploratory data analysis. We chose 8 continuous variables and 2 discrete variables, listed in the table below. We selected these variables based upon what we expect are popular property attributes that customers look for when selecting a house: What is the size of the house? How many bedrooms and bathrooms does it have? How much garage and outdoor space does the property have? How old is the house? What is the building quality? We aim to see whether these variables in particular can be used to predict the sale price of a property.

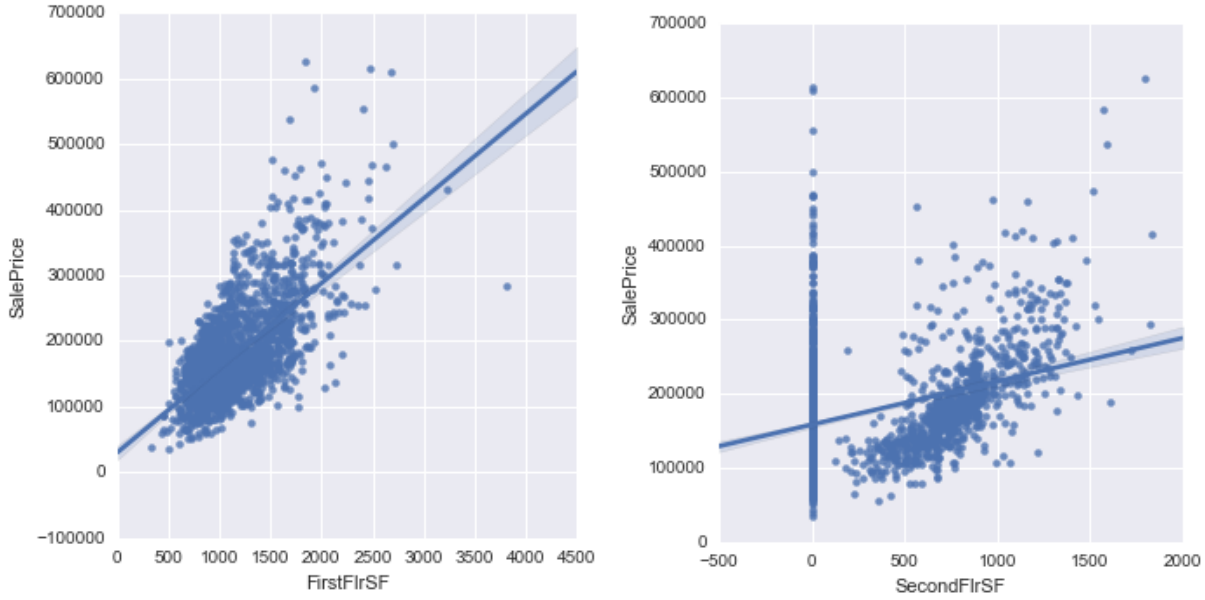
Table 1: Variables selected for EDA

Variable	Description	Type of Variable
1st Flr SF	First Floor square feet	continuous
2nd Flr SF	Second Floor square feet	continuous
GR Liv Area	Above grade (ground) living area square feet	continuous
Bedroom	Bedrooms above grade (not including basement)	discrete
Wood Deck SF	Wood deck area in square feet	continuous
Full Bath	Full bathrooms above grade	discrete
Garage Area	Size of garage in square feet	continuous
Low Qual Fin SF	Low quality finished square feet (all floors)	continuous
Total Bsmt SF	Total square feet of basement area	continuous
Year Built	Original construction date	continuous

Section 2.2: Continuous Variables

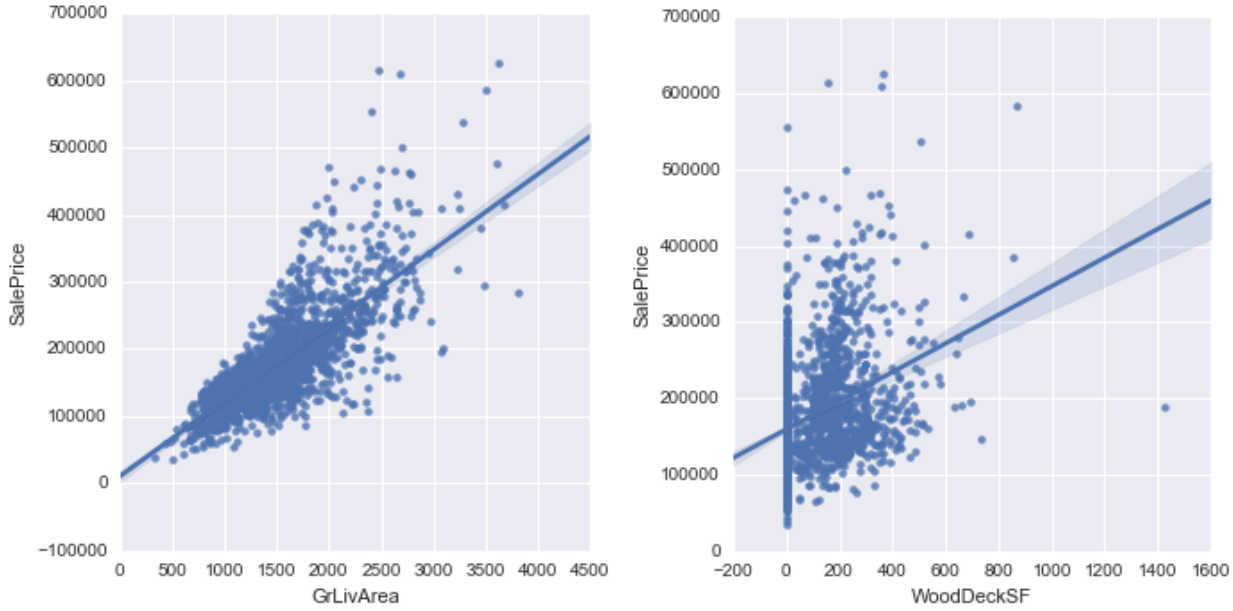
We made scatter plots of the continuous predictor variables to examine their relationship with SalePrice. The FirstFlrSF was clustered between 500 and 2000 square feet, and the SecondFlrSF was clustered between 250 and 1000 square feet. While both are positively related to SalePrice, the SecondFloorSF variable was 0 for many properties, which may make it difficult to predict SalePrice at a low price range, and the FirstFloorSF is clustered around such a small range, it may be difficult to predict SalePrices above 2000 square feet.

Fig 2: Scatterplots of First Floor and Second Floor Square Feet versus. SalePrice



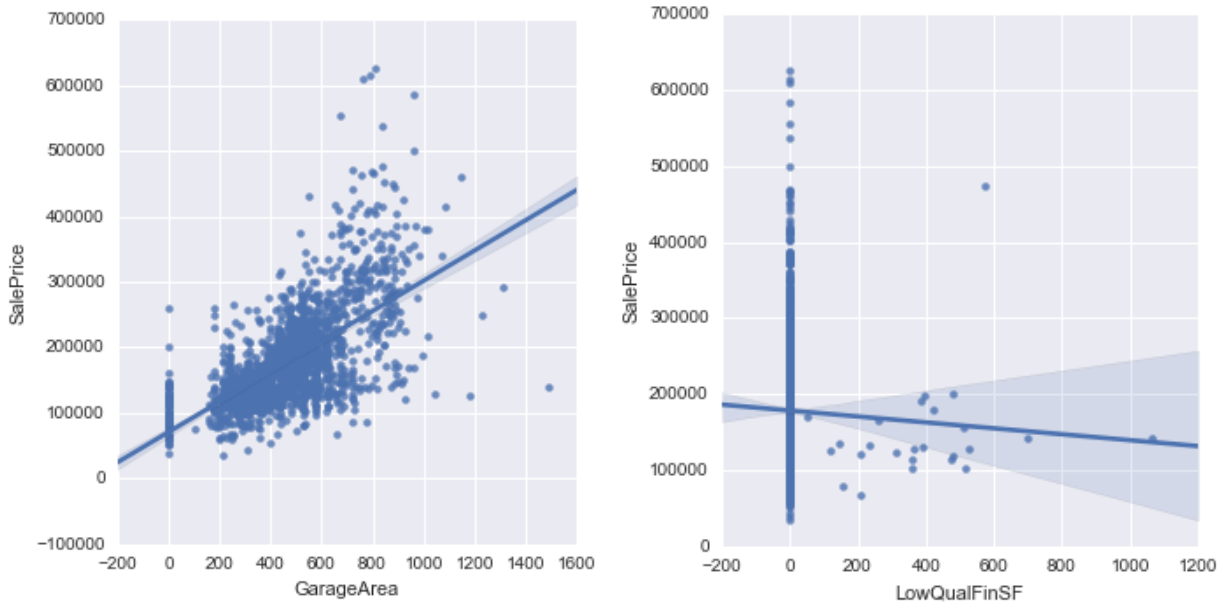
In the scatterplots below, the total square feet above grade, GrLivArea, showed a positive relationship with SalePrice and may be a good predictor for our model. GrLivArea had a more linear relationship with SalePrice before 3000. WoodDeckSF was 0 for many properties up to 300000, which would make it difficult to predict SalePrice.

Fig 3: Scatterplots of GrLivArea and Wood Deck Square Feet versus SalePrice



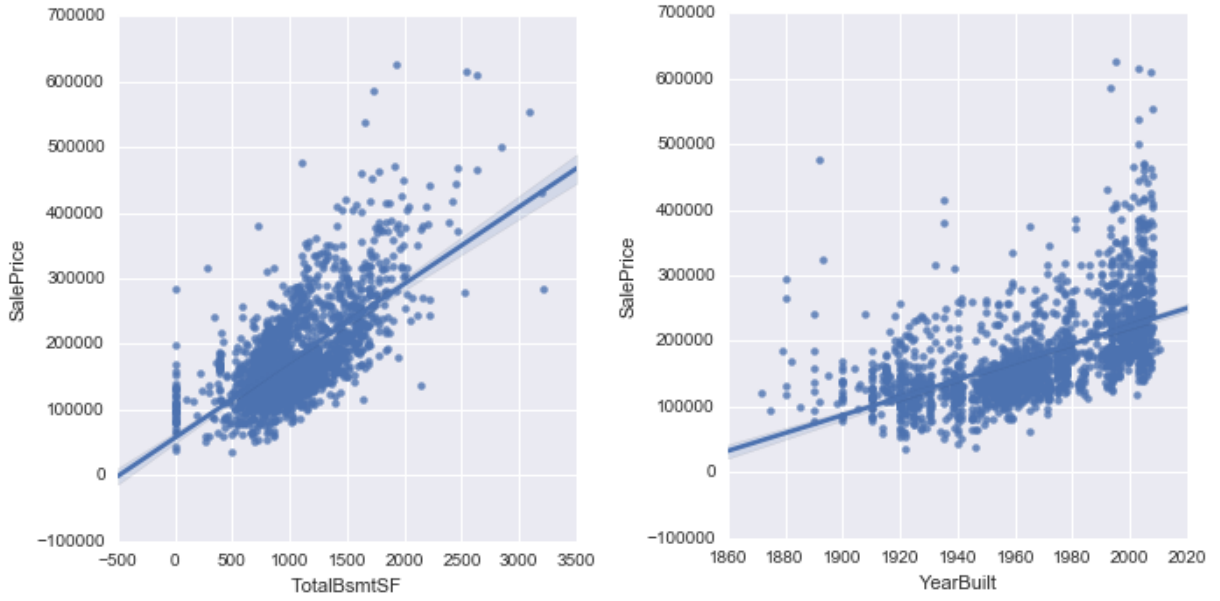
The scatterplots for GarageArea and LowQualFinSF below showed that both may not be good predictors for SalePrice. GarageArea was positively related to SalePrice, but the points were not closely clustered around the linear model with SalePrice. Most properties had 0 LowQualFinSF, which would make it very difficult to produce a predictive model for SalePrice.

Fig 4: Scatterplots of GarageArea and LowQualFinSF versus SalePrice



Both TotalBsmtSF and YearBuilt appeared to be positively related to SalePrice in the scatterplots below. YearBuilt appeared to be a discrete-like variable.

Fig 5: Scatterplots of TotalBsmtSF and YearBuilt versus SalePrice

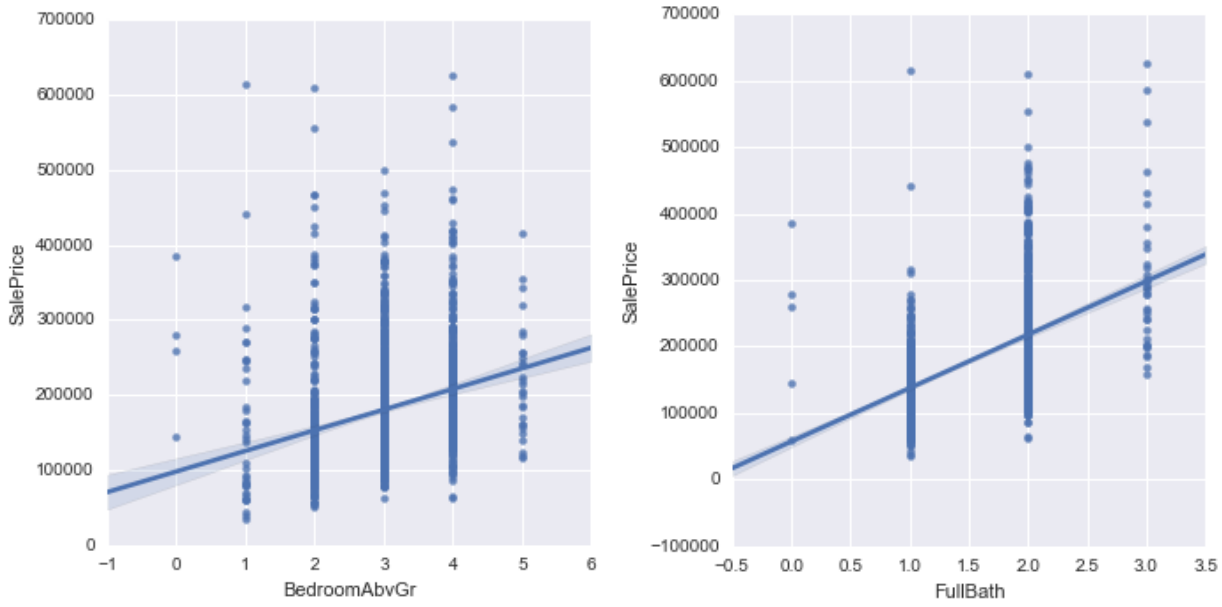


Section 2.3: Discrete Variables

We made scatterplots of the discrete variables BedroomAbvGr and FullBath to explore their relationship with SalePrice. Both appeared to have a positive relationship with SalePrice.

Properties had a larger range of Bedrooms than FullBath, which may make BedroomAbvGr a better predictor for SalePrice. However, both variables had a large SalePrice range for each discrete step, which may make it difficult to accurately predict SalePrice from a discrete variable.

Fig 6: Scatterplots of BedroomAvbGr and FullBath Vs. SalePrice



Section 3: Simple Linear Regression Models

We chose the continuous variables GrLivArea and YearBuilt to futher explore as predictors for SalePrice.

Section 3.1: Model 1 - GrLivArea

We performed an OLS regression on GrLivArea and SalePrice which had the results below.

Table 2: OLS Regression Results GrLivArea and SalePrice

Dep. Variable:	y	**R-squared:**	0.594
Model:	OLS	**Adj. R-squared:**	0.593
Method:	Least Squares	**F-statistic:**	2834.
Date:	Sun, 02 Jul 2017	**Prob (F-statistic):**	0.00
Time:	17:47:35	**Log-Likelihood:**	-23600.
No. Observations:	1942	**AIC:**	4.720e+04
Df Residuals:	1940	**BIC:**	4.722e+04
Df Model:	1		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[95.0% Conf. Int.]**
Intercept	1.097e+04	3309.019	3.316	0.001	4483.749 1.75e+04
X	112.2382	2.108	53.232	0.000	108.103 116.373

Omnibus:	498.190	**Durbin-Watson:**	1.193
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	2817.066
Skew:	1.084	**Prob(JB):**	0.00
Kurtosis:	8.488	**Cond. No.**	4.99e+03

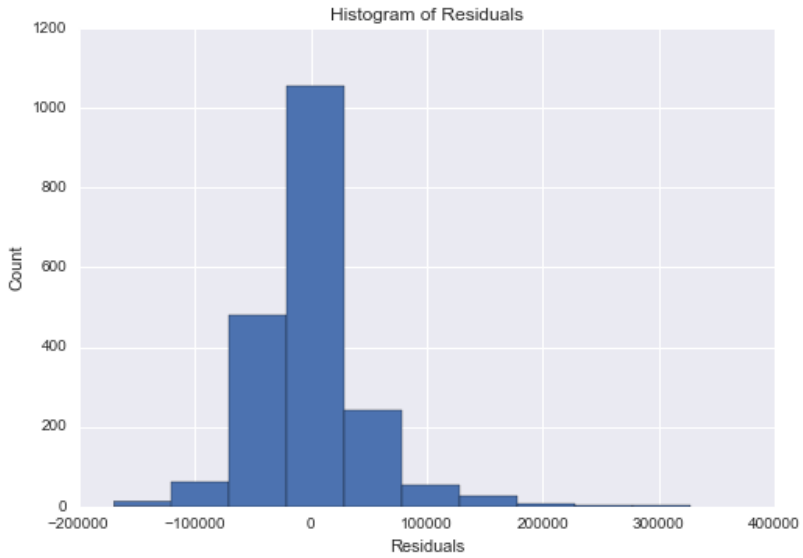
The simple linear regression model has an r-squared coefficient of 0.594, which suggests GrLivArea has a positive correlation with SalePrice. GrLivArea has a coefficient value of 112.2382, which has a p-value less than 0.05. We can reject the null hypothesis that there is no relationship between GrLivArea and SalePrice.

Fig 7: Linear Regression Model of GrLivArea versus SalePrice

The linear model (above) appears to fit better below a SalePrice of 200000, as there is less density around the model after 200000. A histogram of the residuals (below) indicates that

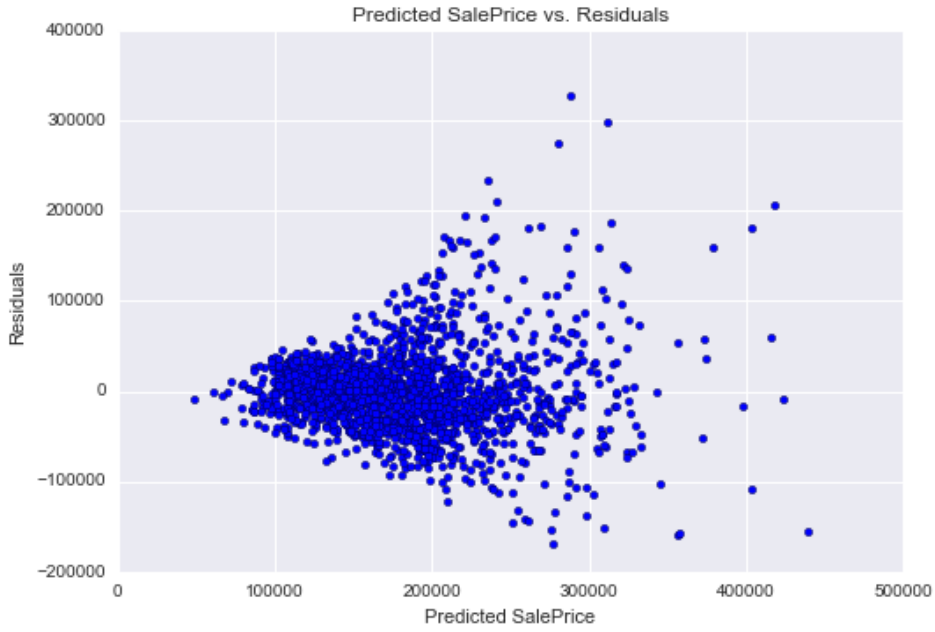
the residual distribution has a positive skew and kurtosis. The residuals may be able to be normalized in future analyses.

Fig 8: Histogram of GrLivArea-SalePrice Simple Linear Regression Residuals



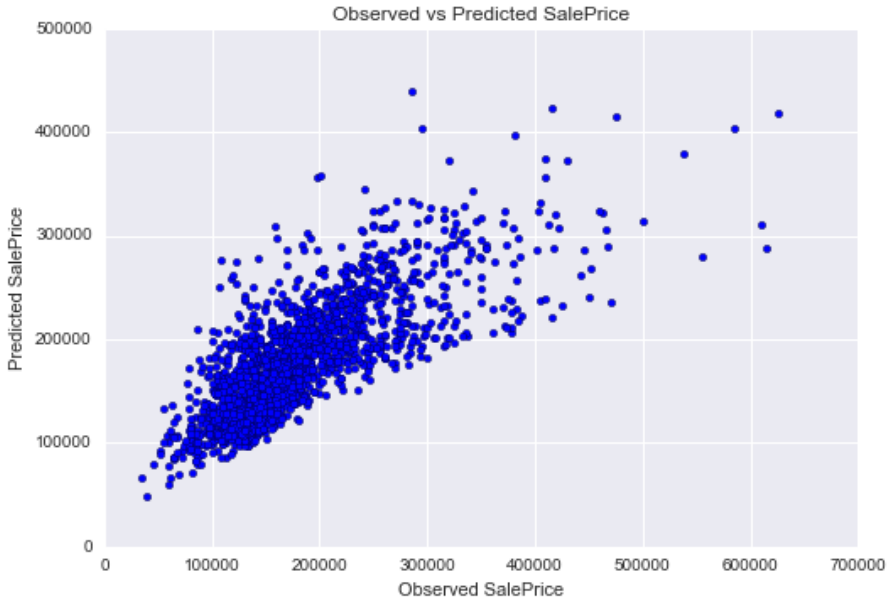
Looking at a plot of the fitted values and residuals (below), Predicted SalePrice appears to be symmetrically distributed above and below zero. However, there is a slight pattern between a predicted SalePrice of 0 and 200000 that suggests that the residuals are not randomly distributed. The pattern resembles heteroscedasticity, which means the residuals get larger as the prediction moves from small to large. Our model could be improved at prediction at higher SalePrices.

Fig 9: Plot of Predicted SalePrice versus Residuals



Last, a graph of the predicted vs. actual values of SalePrice reveals a strong linear relationship (below), although the relationship dissipates after a SalePrice of 300000.

Fig 10: Plot of Actual versus Predicted SalePrice



Overall, GrLivArea appears to be a good predictor for SalePrice and there is room to improve the model.

Section 3.2 Model 2 - YearBuilt

We performed an OLS Regression on YearBuilt and SalePrice which had the results below.

Table 3: OLS Regression Results YearBuilt and SalePrice

Dep. Variable:	y	**R-squared:**	0.309
Model:	OLS	**Adj. R-squared:**	0.308
Method:	Least Squares	**F-statistic:**	866.8
Date:	Sun, 02 Jul 2017	**Prob (F-statistic):**	8.11e-158
Time:	17:47:38	**Log-Likelihood:**	-24116.
No. Observations:	1942	**AIC:**	4.824e+04
Df Residuals:	1940	**BIC:**	4.825e+04
Df Model:	1		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[95.0% Conf. Int.]**
Intercept	-2.497e+06	9.09e+04	-27.478	0.000	-2.68e+06 -2.32e+06
X	1359.9978	46.192	29.442	0.000	1269.406 1450.589

Omnibus:	857.981	**Durbin-Watson:**	1.123
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	5043.867
Skew:	2.010	**Prob(JB):**	0.00
Kurtosis:	9.795	**Cond. No.**	1.32e+05

The simple linear regression model has an r-squared coefficient of 0.309, which suggests YearBuilt has a positive correlation with SalePrice, though not very strong. YearBuilt has a coefficient value of 1359.9978, which has a p-value less than 0.05. We can reject the null hypothesis that there is no relationship between YearBuilt and SalePrice.

The linear model (below) appears to have a strong linear relationship with SalePrice, with most values clustered around the model. There seem to be larger residuals after 1990. A histogram of the residuals (below) indicates that residual distribution has a positive skew and kurtosis. The large number of negative residuals indicates that the predicted value was often larger than the observed value.

Fig 11: Linear Regression Model of YearBuilt versus SalePrice

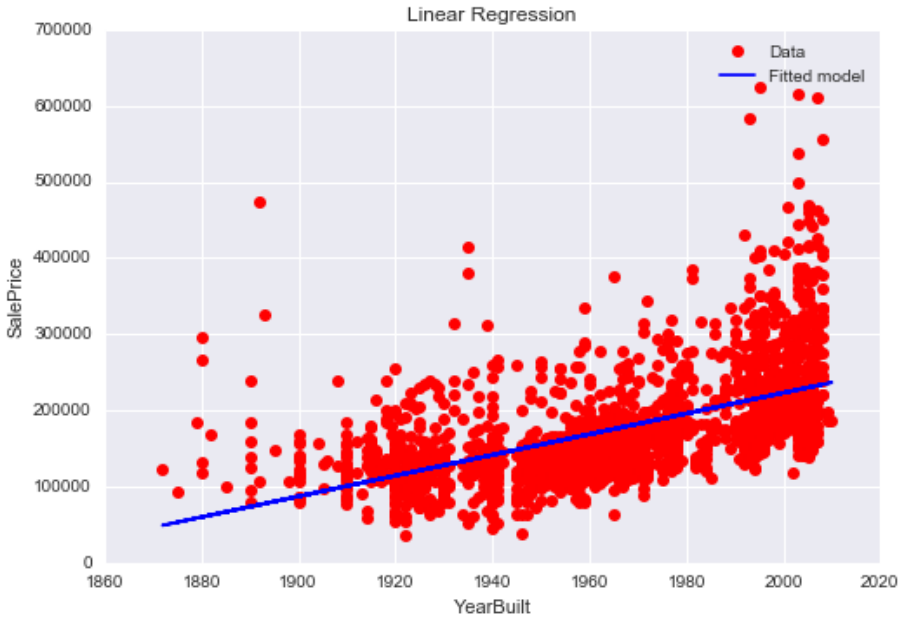
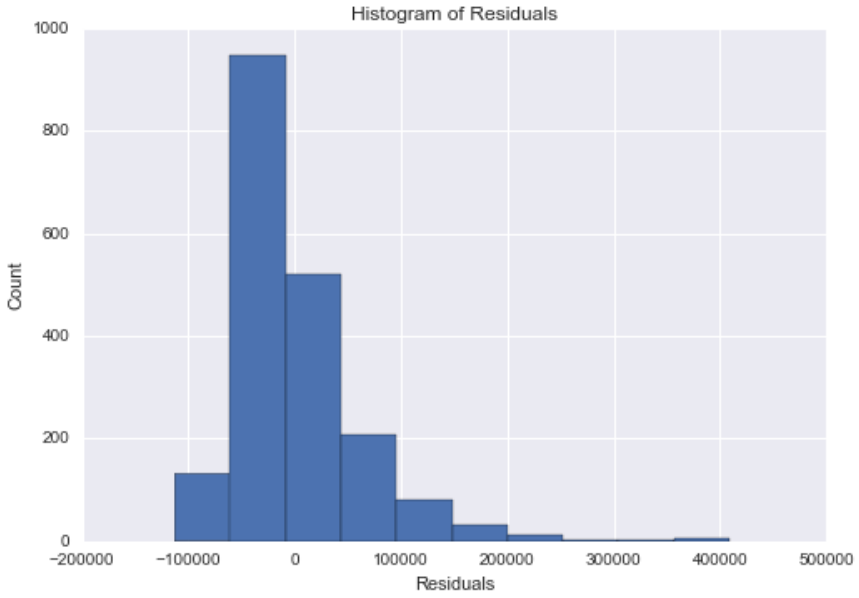
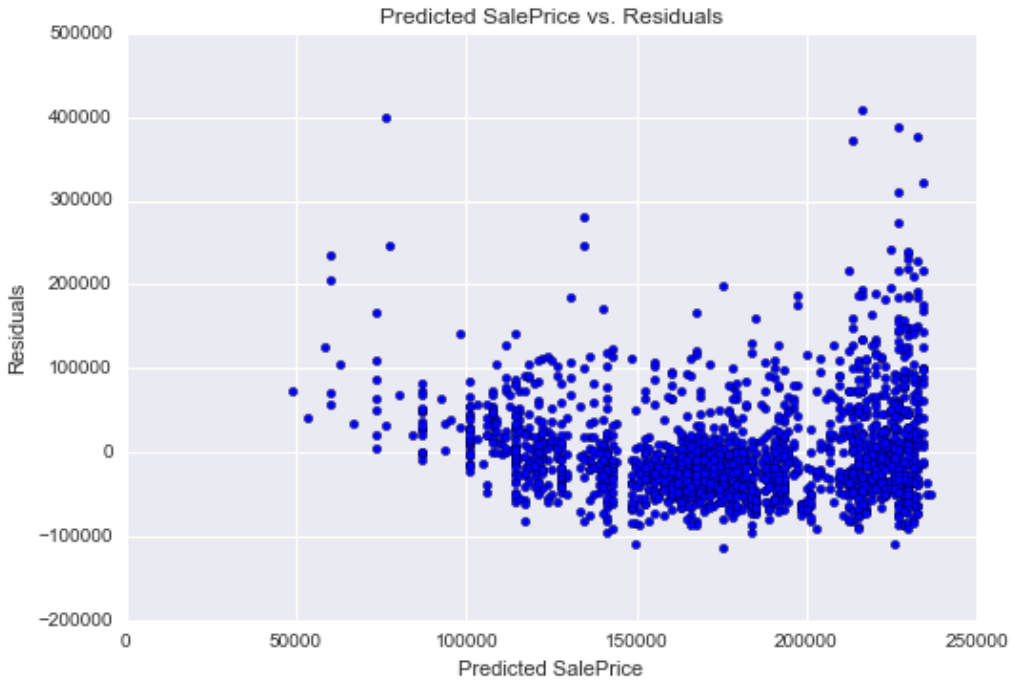


Fig 12: Histogram of YearBuilt-SalePrice Simple Linear Regression Residuals



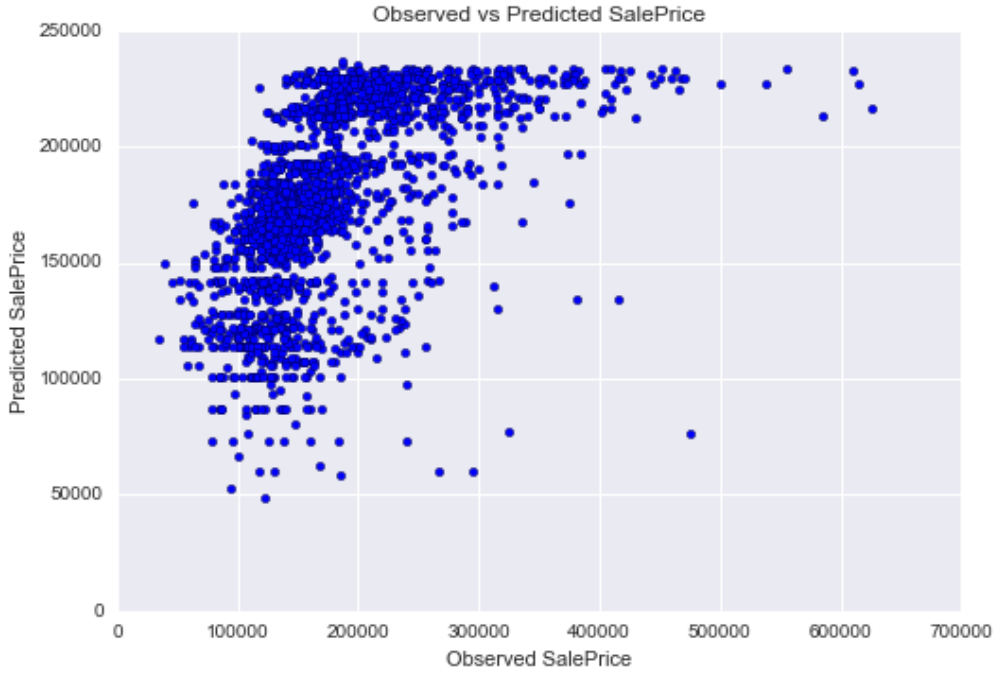
A plot of the predicted SalePrice and residuals show that the residuals lack a symmetrical distribution around 0 and are clustered to the right side of the plot. We should be cautious about using YearBuilt as a predictor. There are also more residuals below 0 than above, which reveals again that our model predicted too high of SalePrice values.

Fig 13: Plot of Predicted SalePrice versus Residuals



A graph of the actual vs. predicted SalePrice values reveals y-axis imbalance and a non-linear pattern. YearBuilt may not be the best predictor of SalePrice, at least own its own. The correlation between YearBuilt and SalePrice may be improved through transformation of SalePrice, or by adding another variable.

Fig 14: Plot of Actual versus Predicted SalePrice



Section 4: Model 3 - Multiple Linear Regression

We performed an OLS regression on GrLivArea and YearBuilt of SalePrice, which had the results below.

Table 4: OLS Regression Results GrLivArea, YearBuilt, and SalePrice

Dep. Variable:	SalePrice	**R-squared:**	0.722
Model:	OLS	**Adj. R-squared:**	0.722
Method:	Least Squares	**F-statistic:**	2523.
Date:	Sun, 02 Jul 2017	**Prob (F-statistic):**	0.00
Time:	17:47:40	**Log-Likelihood:**	-23230.
No. Observations:	1942	**AIC:**	4.647e+04
Df Residuals:	1939	**BIC:**	4.648e+04
Df Model:	2		
Covariance Type:	nonrobust		

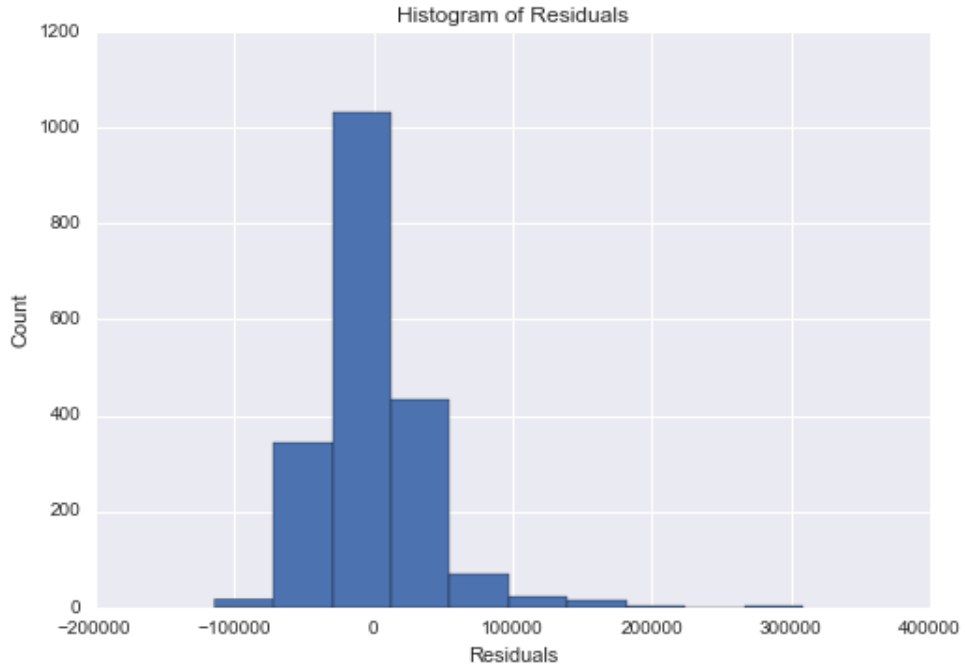
	coef	**std err**	**t**	**P>|t|**	**[95.0% Conf. Int.]**
const	-1.763e+06	5.92e+04	-29.780	0.000	-1.88e+06 -1.65e+06
GrLivArea	97.3908	1.812	53.750	0.000	93.837 100.944
YearBuilt	913.0768	30.438	29.998	0.000	853.382 972.772

Omnibus:	787.437	**Durbin-Watson:**	1.474
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	5948.834
Skew:	1.719	**Prob(JB):**	0.00
Kurtosis:	10.855	**Cond. No.**	1.71e+05

The multiple linear regression model has a correlation coefficient of 0.722, which suggests there is a strong positive correlation between GrLivArea and YearBuilt to SalePrice. This correlation coefficient is higher than that of the simple linear models of GrLivArea and YearBuilt to SalePrice, but this is expected because the correlation coefficient usually gets larger when more variables are included. For this reason, we look at the adjusted r-squared value which adjusts for the number of predictors in the model—however, in this case, both are the same value of 0.722. The coefficient of GrLivArea is 97.3908 and the coefficient for YearBuilt is 913.0768. Both coefficients have a p-value less than 0.05, which means we can reject the null hypothesis that there is no relationship between GrLivArea and YearBuilt and SalePrice.

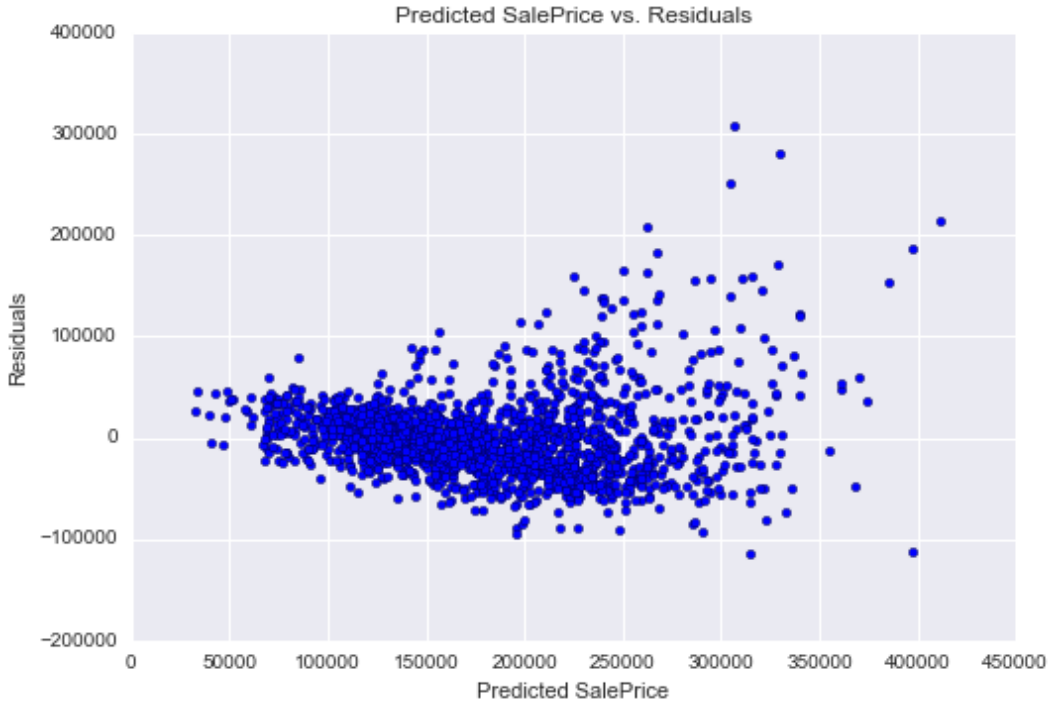
A histogram of the residuals (below) shows positive skewness and kurtosis. There appears to be outliers that may be skewing the model.

Fig 15: Histogram of Residuals



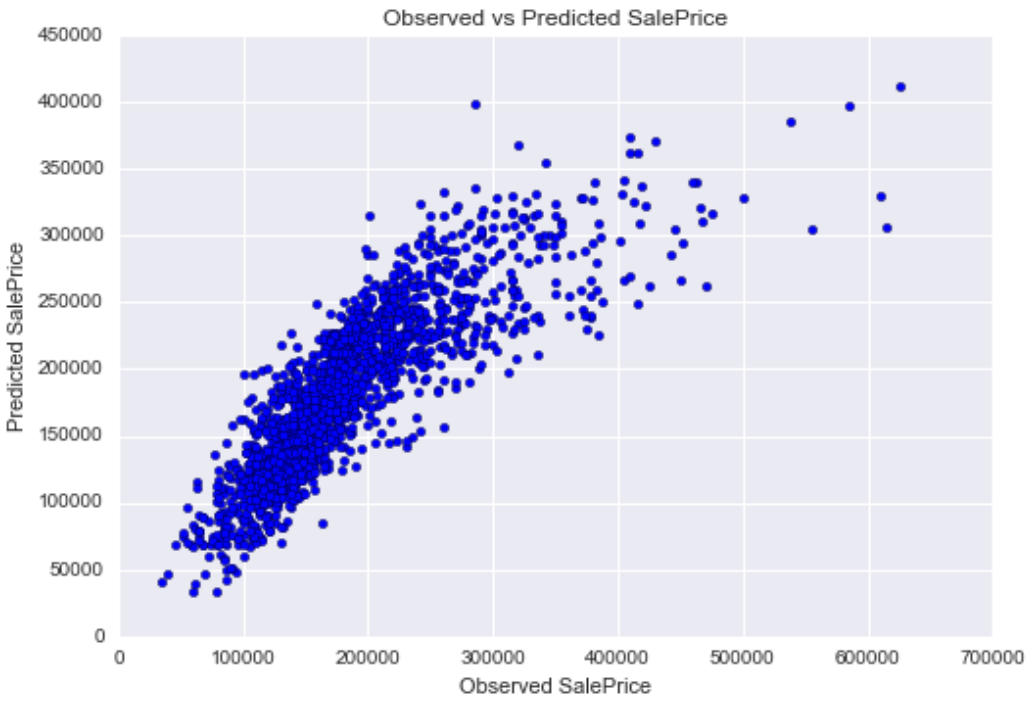
A plot of the predicted SalePrice values vs Residuals (below) shows that the residuals are somewhat symmetrically distributed around 0, with more residuals below 0 than above. The residuals are more clustered towards the middle of the plot than our simple linear models. There is a pattern of heteroscedasticity, which our linear model of GrLivArea and SalePrice showed as well.

Fig 16: Plot of Predicted SalePrice versus Residuals



Lastly, a plot of the observed vs predicted SalePrice values shows a stronger linear relationship than our simple linear models. The model is less accurate after a SalePrice of 300000, which we have consistently observed across our models. This may reveal that we don't have enough observations that have a SalePrice above 300000 to make an accurate prediction at that range.

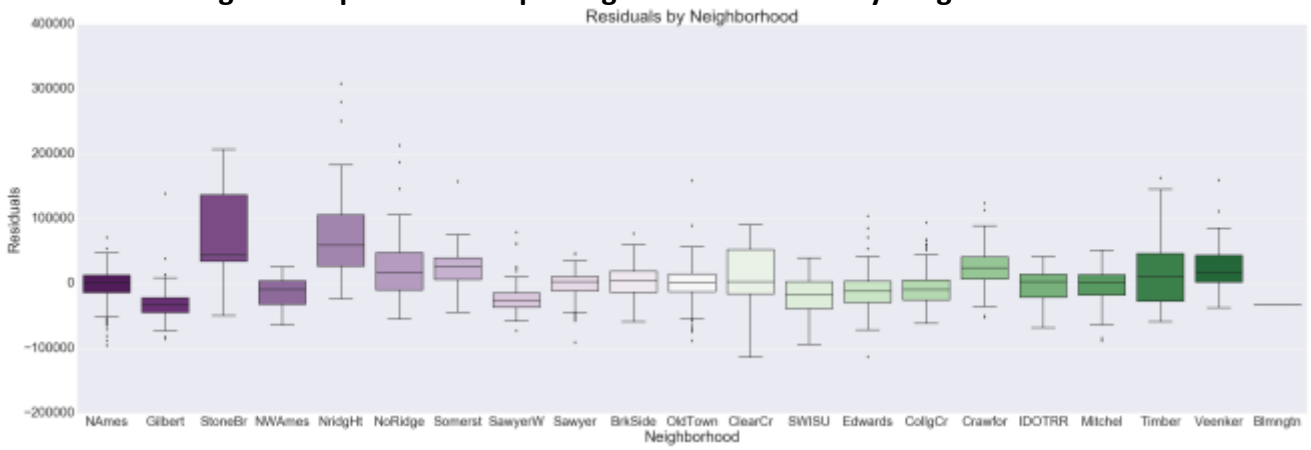
Fig 17: Plot of Observed versus Predicted SalePrice



Section 5: Neighborhood Accuracy

We created a boxplot (below) of the residuals by neighborhood, in order to evaluate whether some neighborhoods are better fit by the model than others.

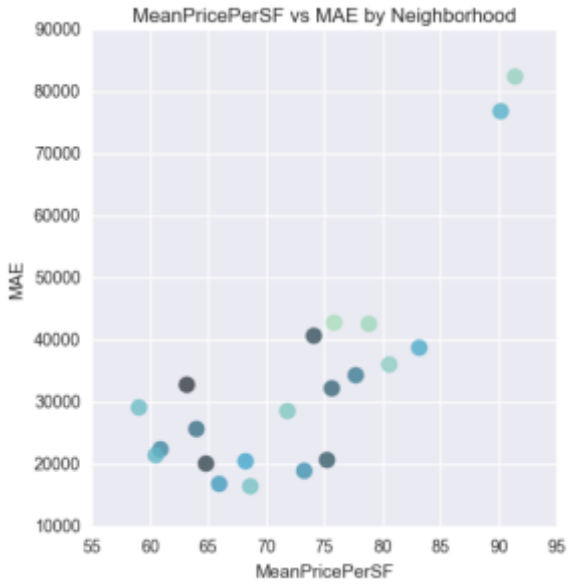
Fig 18: Boxplots of Multiple Regression Residuals by Neighborhood



The neighborhoods of StoneBr and NridgHt are consistently overpredicted with larger residuals than the other neighborhoods. NoRidge, Somerst, Crawfor, Timber and Veenker neighborhoods are also consistently overpredicted by the model, but with smaller residuals. Underpredicted neighborhoods include Gilbert, NWAmes, SawyerW, SWIsu, Edwards, CollgCr, and Blmngtn, although there appears to be only one property in the Blmngtn neighborhood so we may not have enough data to predict SalePrices in that neighborhood. Both over and underprediction could result from either bias in our sample or from a missing variable in our model. We may have left out crucial environmental or demographic factors that vary across neighborhoods and influence SalePrice, such as the amount of green space or food options, or racial demographics.

There also seem to be 1 to 3 extreme outliers for most neighborhoods. These outliers could be influencing the model to overpredict or underpredict SalePrices for a neighborhood. For example, both NridgHt and NoRidge neighborhoods appear to have extreme outliers that may cause overprediction in our model. The StoneBr neighborhood in particular has a very skewed distribution to the right, suggesting there are outliers or influential values that could be removed to improve our model. Long boxplot whiskers in both directions, such as the boxplot for NridgHt neighborhood, could suggest that the neighborhood is being developed due to forces like gentrification or external investments. These forces may result in a large range of SalePrices within the same neighborhood. Additional variables that cause or represent the variation among Neighborhoods could also be used to improve our model.

Fig 19: MeanPricePerSF versus MAE by Neighborhood



We found a positive relationship between the mean price per square foot (MeanPricePerSF) and the mean absolute error (MAE) in predicting SalePrice. The data for each neighborhood tends to be clustered below a mean absolute error of 45000. We see that the lowest MeanPricePerSF has a MAE of 20000 to 35000, and the MAE slightly decreases as the MeanPricePerSF increases from 65 to 70, and then the MAE and MeanPricePerSF increase together. There is a large difference between the MAE for a MeanPricePerSF under 85 and the MAE for a MeanPricePerSF around 90. This suggests to us that, while the MAE in predicting SalePrice generally increases with MeanPricePerSF, there may be more variability in SalePrice as the MeanPricePerSF increases.

In order to further explore the relationship between Neighborhoods and SalePrice, we grouped Neighborhoods together based on their MeanPricePerSF and created a new predictor variable. After removing Neighborhoods that were not in our dataset, we created 5 Neighborhood Groups based on their MeanPricePerSF: properties under 64 were Group 0, properties between 64 and 70 were Group 1, properties between 70 and 74 were Group 2, properties between 74 and 85 were Group 3, and properties between 85 and 95 were Group 4. We then created dummy variables for each Group and selected Group 1 as the base category. We selected the base category based on which Group had the largest number neighborhoods. Table 5 (below) shows the output from a multiple linear regression model of SalePrice that includes GrLivArea, YearBuilt, and Neighborhood Group as the predictors.

Table 5: OLS Regression Results GrLivArea, YearBuilt, Neighborhood, and SalePrice

Dep. Variable:	SalePrice	**R-squared:**	0.786
Model:	OLS	**Adj. R-squared:**	0.785
Method:	Least Squares	**F-statistic:**	1185.
Date:	Sun, 09 Jul 2017	**Prob (F-statistic):**	0.00
Time:	13:15:23	**Log-Likelihood:**	-22977.
No. Observations:	1942	**AIC:**	4.597e+04
Df Residuals:	1935	**BIC:**	4.601e+04
Df Model:	6		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[95.0% Conf. Int.]
const	-1.021e+06	7.4e+04	-13.801	0.000	-1.17e+06 -8.76e+05
GrLivArea	86.9523	1.667	51.531	0.000	83.643 90.262
YearBuilt	539.7119	37.785	14.284	0.000	465.609 613.815
Group_0	-1.173e+04	2310.249	-5.076	0.000	-1.63e+04 -7195.526
Group_2	6228.8430	2796.826	2.227	0.026	743.733 1.17e+04
Group_3	1.85e+04	2284.360	8.097	0.000	1.4e+04 2.3e+04
Group_4	1.032e+05	4478.656	23.054	0.000	9.45e+04 1.12e+05

Omnibus:	669.442	**Durbin-Watson:**	1.660
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	4557.506
Skew:	1.448	**Prob(JB):**	0.00
Kurtosis:	9.923	**Cond. No.**	2.43e+05

The OLS Regression Results from a multiple linear regression model, including Neighborhood groups, has a correlation coefficient of 0.786, which is higher than that of our previous multiple linear regression model (Model #3). This value suggests that there is a strong positive correlation between GrLivArea, YearBuilt, and Neighborhood Group based on MeanPricePerSF and SalePrice. The adjusted correlation coefficient value is almost the same at 0.785. All of the coefficients have a p-value less than 0.05, indicating that we can reject the null hypothesis that there is no relationship between our predictor and response variable.

Both Model #3 and the neighborhood-inclusive multiple regression model had statistically significant F-statistics, suggesting that in both scenarios, we can reject the null hypothesis that all of the regression coefficients are zero. The MAE from our multiple regression model including Neighborhood Groups is approximately 23279. This value is smaller than the MAE of Model #3, which was approximately 26306. The smaller MAE suggests that the multiple linear regression model including Neighborhood Groups has generally lower forecast error and therefore better predictive capabilities.

Section 6: SalePrice versus Log SalePrice as the Response

We compared a multiple linear regression model of SalePrice versus log(SalePrice) to explore whether a log transformation would improve our model. We chose 4 continuous variables as predictors: GrLivArea, YearBuilt, GarageArea, TotalBsmtSF.

Section 6.1: SalePrice Model

The OLS Regression Results from our SalePrice model (below) has a very high correlation coefficient of 0.968. Although the F-statistic and t-statistic for all the coefficients are significant, we should look at the interaction effects between predictors and explore whether there is multicollinearity impacting our model in future analyses.

Table 6: OLS Regression Results GrLivArea, YearBuilt, GarageArea, TotalBsmtSF, and SalePrice

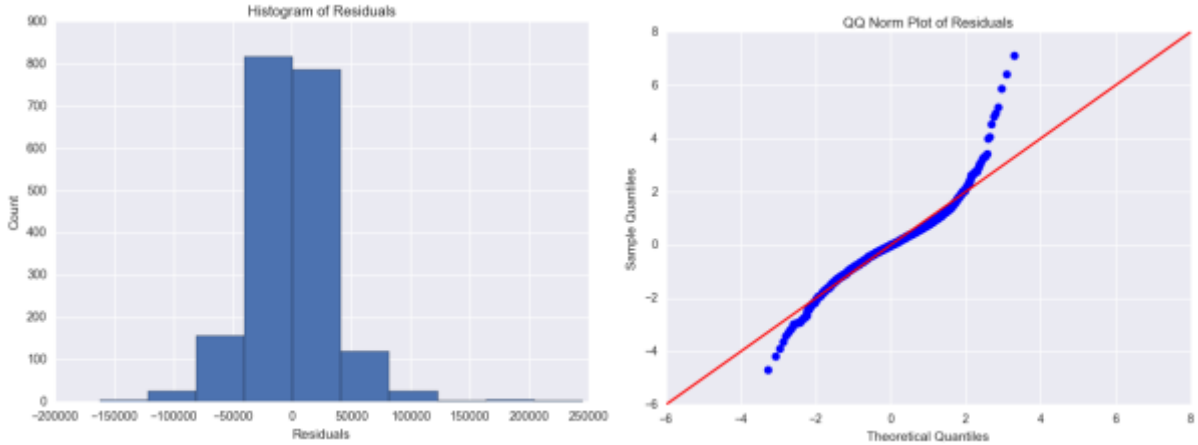
Dep. Variable:	SalePrice	**R-squared:**	0.968
Model:	OLS	**Adj. R-squared:**	0.968
Method:	Least Squares	**F-statistic:**	1.457e+04
Date:	Sun, 09 Jul 2017	**Prob (F-statistic):**	0.00
Time:	13:42:17	**Log-Likelihood:**	-23045.
No. Observations:	1942	**AIC:**	4.610e+04
Df Residuals:	1938	**BIC:**	4.612e+04
Df Model:	4		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[95.0% Conf. Int.]
GrLivArea	75.6943	1.857	40.762	0.000	72.052 79.336
YearBuilt	-18.2626	1.447	-12.621	0.000	-21.100 -15.425
GarageArea	83.6872	4.765	17.561	0.000	74.341 93.033
TotalBsmtSF	60.6986	2.266	26.782	0.000	56.254 65.143

Omnibus:	306.805	**Durbin-Watson:**	1.484
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	2075.229
Skew:	0.556	**Prob(JB):**	0.00
Kurtosis:	7.941	**Cond. No.**	17.1

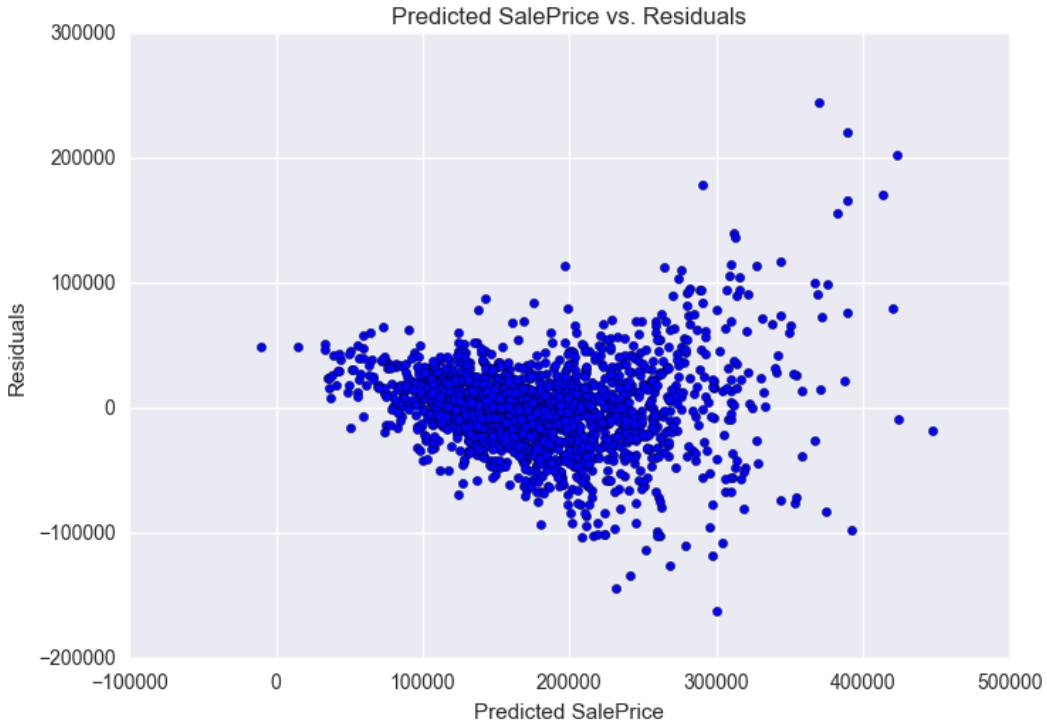
A histogram of our SalePrice model residuals (below, left) reveals a very leptokurtic distribution with slight right skew, with most values clustered around 0. There may be outliers or influential values causing skewness. The QQ Norm Plot of residuals (below, right) also suggests that the residuals may not follow a normal distribution: the residuals have light tails and are skewed on both ends. Because the residuals do not appear to be normal, future analysis should seek to remove outliers or influential values that are skewing the residual distribution.

Fig 20: Histogram and QQ Norm Plot of Residuals of SalePrice Model



Last, a scatterplot of the SalePrice model reveals a pattern of heteroscedasticity that was present in previous models as well. While the residuals are clustered toward the center of the plot and appear symmetrically distributed around 0, we should be cautious about predictions of SalePrices above 250,000.

Fig 21: Scatterplot of Predicted SalePrice versus Residuals



Section 6.2: Log SalePrice Model

The OLS Regression Results from our log(SalePrice) model (below) has a very high correlation coefficient of 1.00, which is higher than the correlation coefficient of the SalePrice model. Although the F-statistic and t-statistic for all the coefficients are significant, similar to the SalePrice model, we should also look at the interaction effects between predictors in future analysis. A correlation coefficient value of 1.00 is very unusual and highly suggestive of multicollinearity. Although our predictors represent different areas of square footage, they may be very strongly correlated with another. For example, the living area above grade may be consistently proportional to the garage area and basement size.

Table 7: OLS Regression Results GrLivArea, YearBuilt, Garage Area, TotalBsmtSF, and log(SalePrice)

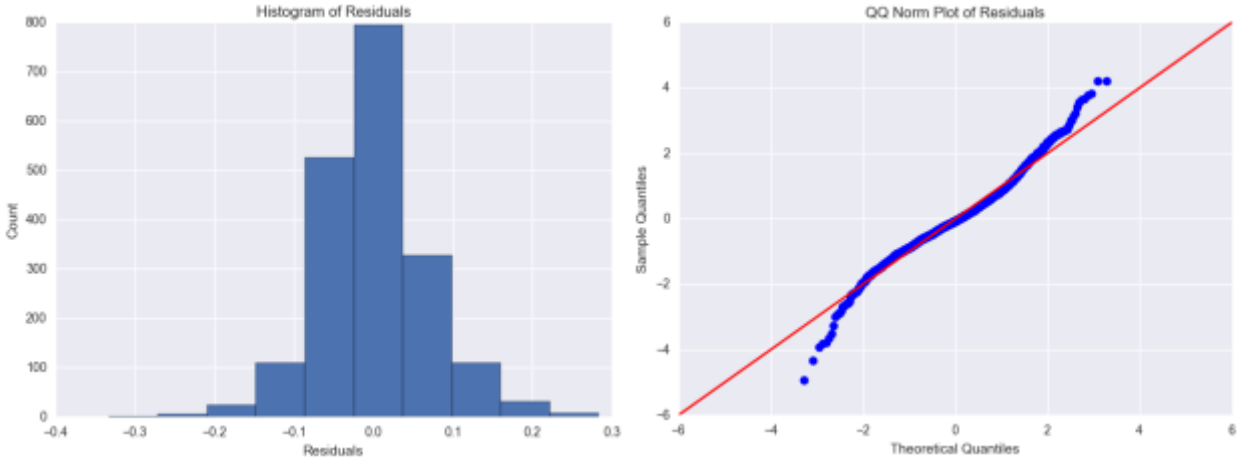
Dep. Variable:	SalePrice	**R-squared:**	1.000
Model:	OLS	**Adj. R-squared:**	1.000
Method:	Least Squares	**F-statistic:**	2.904e+06
Date:	Sun, 09 Jul 2017	**Prob (F-statistic):**	0.00
Time:	13:41:02	**Log-Likelihood:**	2480.7
No. Observations:	1942	**AIC:**	-4953.
Df Residuals:	1938	**BIC:**	-4931.
Df Model:	4		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[95.0% Conf. Int.]
GrLivArea	0.0002	3.63e-06	47.526	0.000	0.000 0.000
YearBuilt	0.0025	2.83e-06	869.886	0.000	0.002 0.002
GarageArea	7.604e-05	9.33e-06	8.154	0.000	5.77e-05 9.43e-05
TotalBsmtSF	7.989e-05	4.44e-06	18.013	0.000	7.12e-05 8.86e-05

Omnibus:	95.461	**Durbin-Watson:**	1.545
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	263.015
Skew:	0.221	**Prob(JB):**	7.71e-58
Kurtosis:	4.748	**Cond. No.**	17.1

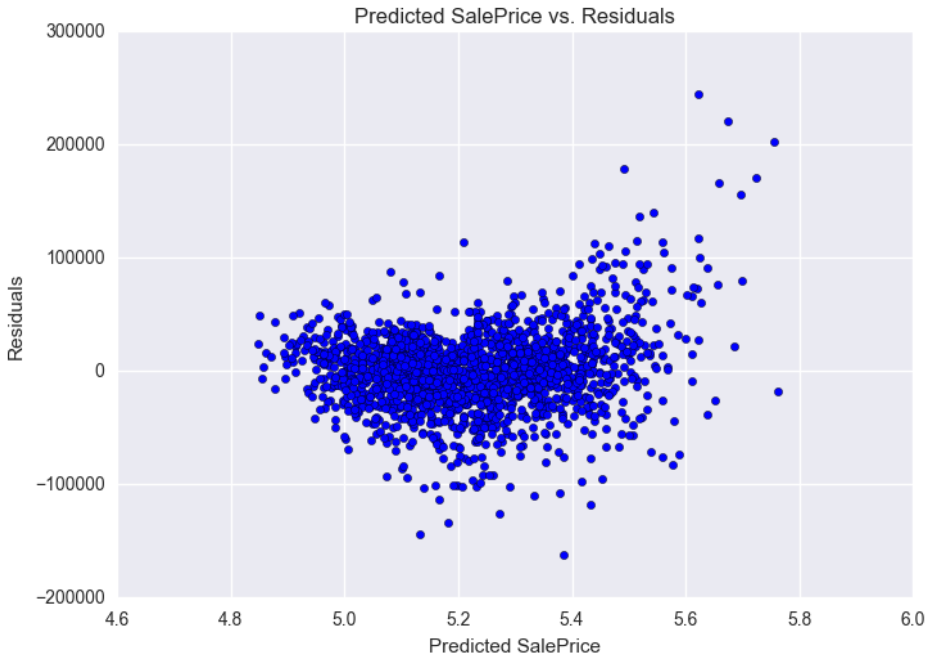
A histogram of our log(SalePrice) model residuals (below, left) reveals a more normal distribution than the SalePrice model. There is less kurtosis and skewness in the residual distribution of the log(SalePrice) model. We also did not observe the outliers or influential values that skewed the SalePrice model residuals. The QQ Norm Plot of log(SalePrice) residuals (below, right) also suggests a more normal distribution: the residuals tails more closely follow the line of normal distribution and are less skewed than those in the SalePrice model.

Fig 22: Histogram and QQ Norm Plot of Residuals of log(SalePrice) Model



Last, a scatterplot of predicted log(SalePrice) and the residuals reveals a distribution more closely clustered in the center of the plot and more symmetrically distributed around zero than the SalePrice model residuals. We also observed less heteroscedasticity in the log(SalePrice) model residuals.

Fig 23: Scatterplot of Predicted log(SalePrice) versus Residuals



In general, a log transformation of the response variable can help eliminate heteroscedasticity and non-normal residual distributions. An assumption of linear regression is that model residuals follow a normal distribution. Techniques like log transforming the predictor variable can reduce the influence of outliers or influential observations and create a more normal residual distribution. A log transformation of both the response and predictor variables can even transform a non-linear model into a linear one. However, techniques to linearize data should be used carefully, because linear regression is often used incorrectly to model non-linear relationships. In this analysis, a log transformation improved the strength of the correlation coefficient, reduced heteroscedasticity, and normally distributed the model residuals, enabling us to validate the assumptions of linear regression.

Conclusion

In conclusion, GrLivArea, YearBuilt, and Neighborhood appear to have a positive relationship with SalePrice, and we rejected the null hypothesis that they have no relation to SalePrice. A log transformation of the response variable ensured our model validated the assumption of normally distributed residuals for linear regression. Neighborhood Groups, determined by MeanPricePerSF, produced a model with smaller MAE than our previous multiple regression models. Future analyses should explore the causes of variability in SalePrice across neighborhoods and seek to incorporate this variability into our model. We should continue to be cautious of very correlation coefficients, such as 1.00, and further explore their causes, such as multicollinearity. Our models consistently performed better at SalePrices below 300,000, and an expanded dataset of high SalePrices could lesson the heterodasticity present in many of our models. In general, there are improvements that can be made to fix slight imbalances or unnormal features in our data, including additional techniques to remove

outliers, and future analysis should explore more regression techniques, including log transformations, to refine the model.

Appendix A: Ipython Notebook Code

```
import os
import matplotlib.pyplot as plt
%matplotlib inline
import numpy as np
import pandas as pd
import sys
sys.path.append('/anaconda/lib/python2.7/site-packages')
import seaborn as sns
sns.set_style("darkgrid")
import statsmodels.api as sm
import statsmodels.formula.api as smf

amesdataset=pd.read_csv('ames_housing_data.csv')
len(amesdataset)
#waterfall counts

#all data
a = amesdataset
lena = len(a)
#Homes w/residential zoning limits
b = amesdataset[(amesdataset['Zoning'] == 'RH') | (amesdataset['Zoning'] == 'RL') |
(amesdataset['Zoning'] == 'RP') | (amesdataset['Zoning'] == 'RM')]
lenb = len(b)
#Homes w/Normal sale conditions
c = b[b['SaleCondition'] == 'Normal']
lenc = len(c)
#Homes single-family residences
d = c[c['BldgType'] == '1Fam']
lend = len(d)
#Homes w/SalePrice < 700,000
e = d[d['SalePrice'] <= 700000]
lene = len(e)
print lena
print lenb
print lenc
print lend
print lene
```

```
dropconditions = ['All data', 'Residential Zoning Limits', 'Normal Sale Conditions', 'Single-Family Residences','Drop Outliers']
y_pos = np.arange(len(dropconditions))

observations = (2930, 2762, 2305, 1943, 1942)

#waterfall = plt.bar(z,y, x_tick_label = x)
plt.bar(y_pos, observations, align='center', alpha=0.5)
plt.xticks(y_pos, dropconditions, rotation='vertical')
plt.margins(0.2)
plt.title('Waterfall chart of drop conditions')
plt.show()

dfcontin = e[['FirstFlrSF', 'SecondFlrSF', 'GrLivArea', 'WoodDeckSF', 'GarageArea', 'LowQualFinSF', 'TotalBsmtSF', 'YearBuilt', 'SalePrice']]

sns.lmplot('FirstFlrSF', 'SalePrice', dfcontin)
sns.lmplot('SecondFlrSF', 'SalePrice', dfcontin)
sns.lmplot('GrLivArea', 'SalePrice', dfcontin)
sns.lmplot('WoodDeckSF', 'SalePrice', dfcontin)
sns.lmplot('GarageArea', 'SalePrice', dfcontin)
sns.lmplot('LowQualFinSF', 'SalePrice', dfcontin)
sns.lmplot('TotalBsmtSF', 'SalePrice', dfcontin)
sns.lmplot('YearBuilt', 'SalePrice', dfcontin)

dfdiscr = e[['FullBath', 'BedroomAbvGr', 'SalePrice']]

sns.lmplot('FullBath', 'SalePrice', dfdiscr)
sns.lmplot('BedroomAbvGr', 'SalePrice', dfdiscr)

X = dfcontin[['GrLivArea']]
y = dfcontin['SalePrice']

# create a fitted model in one line
#formula notiation is the equivalent to writting out our models such that 'outcome = predictor'
#with the follwing syntax formula = 'outcome ~ predictor1 + predictor2 ... predictorN'
lm = smf.ols(formula='y ~ X', data=dfcontin).fit()
#print the full summary
lm.summary()

#Plot data and model
plt.plot(dfcontin['GrLivArea'], dfcontin['SalePrice'], 'ro')
plt.plot(dfcontin['GrLivArea'], lm.fittedvalues, 'b')
plt.legend(['Data', 'Fitted model'])
```

```
plt.xlabel('GrLivArea')
plt.ylabel('SalePrice')
plt.title('Linear Regression')

#Histogram of residuals
plt.hist(lm.resid)
plt.ylabel('Count')
plt.xlabel('Residuals')
plt.title('Histogram of Residuals')

# X vs residuals. Residual Plot
plt.scatter(dfcontin['GrLivArea'], lm.resid)
plt.xlabel('GrLivArea')
plt.ylabel('Residuals')
plt.title('Residual Plot')

# Observed vs Fitted
plt.scatter(dfcontin['SalePrice'], lm.fittedvalues)
plt.ylabel('Predicted SalePrice')
plt.xlabel('Observed SalePrice')
plt.title('Observed vs Predicted SalePrice')

# Fitted  vs Residuals
plt.scatter(lm.fittedvalues, lm.resid)
plt.xlabel('Predicted SalePrice')
plt.ylabel('Residuals')
plt.title('Predicted SalePrice vs. Residuals')

X = dfcontin[['YearBuilt']]
y = dfcontin['SalePrice']

# create a fitted model in one line
#formula notiation is the equivalent to writting out our models such that 'outcome = predictor'
#with the follwing syntax formula = 'outcome ~ predictor1 + predictor2 ... predictorN'
lm2 = smf.ols(formula='y ~ X', data=dfcontin).fit()
#print the full summary
lm2.summary()

#Plot data and model
plt.plot(dfcontin['YearBuilt'], dfcontin['SalePrice'], 'ro')
plt.plot(dfcontin['YearBuilt'], lm2.fittedvalues, 'b')
plt.legend(['Data', 'Fitted model'])
plt.xlabel('YearBuilt')
plt.ylabel('SalePrice')
```

```
plt.title('Linear Regression')

#Histogram of residuals
plt.hist(lm2.resid)
plt.ylabel('Count')
plt.xlabel('Residuals')
plt.title('Histogram of Residuals')

# X vs residuals. Residual Plot
plt.scatter(dfcontin['YearBuilt'], lm2.resid)
plt.ylabel('Residuals')
plt.xlabel('YearBuilt')
plt.title('Residual Plot')

# Observed vs Fitted
plt.scatter(dfcontin['SalePrice'], lm2.fittedvalues)
plt.ylabel('Predicted SalePrice')
plt.xlabel('Observed SalePrice')
plt.title('Observed vs Predicted SalePrice')

# Fitted vs Residuals
plt.scatter(lm2.fittedvalues, lm2.resid)
plt.xlabel('Predicted SalePrice')
plt.ylabel('Residuals')
plt.title('Predicted SalePrice vs. Residuals')

X = dfcontin[['GrLivArea', 'YearBuilt']]
y = dfcontin['SalePrice']

## fit a OLS model with intercept
X = sm.add_constant(X)
lm3 = smf.OLS(y, X).fit()

lm3.summary()

sk.mean_absolute_error(dfcontin["SalePrice"], lm3.fittedvalues, sample_weight=None,
multioutput='uniform_average')

#Histogram of residuals
plt.hist(lm3.resid)
plt.ylabel('Count')
plt.xlabel('Residuals')
plt.title('Histogram of Residuals')
```

```
# GrLivArea vs residuals. Residual Plot
plt.scatter(dfcontin['GrLivArea'], lm3.resid)
plt.ylabel('Residuals')
plt.xlabel('GrLivArea')
plt.title('Residual Plot')

# YearBuilt vs residuals. Residual Plot
plt.scatter(dfcontin['YearBuilt'], lm3.resid)
plt.ylabel('Residuals')
plt.xlabel('YearBuilt')
plt.title('Residual Plot')

# Observed vs Fitted
plt.scatter(dfcontin['SalePrice'], lm3.fittedvalues)
plt.ylabel('Predicted SalePrice')
plt.xlabel('Observed SalePrice')
plt.title('Observed vs Predicted SalePrice')

# Fitted vs Residuals
plt.scatter(lm3.fittedvalues, lm3.resid)
plt.xlabel('Predicted SalePrice')
plt.ylabel('Residuals')
plt.title('Predicted SalePrice vs. Residuals')

dfneigh = pd.concat([dfcontin[['GrLivArea', 'YearBuilt', 'SalePrice']], e[['Neighborhood',
'TotalBsmtSF', 'FirstFlrSF', 'SecondFlrSF']], lm3.resid, lm3.fittedvalues], axis=1)
dfneigh=dfneigh.rename(columns = {0:'Residual', 1:'Fitted Value'})
dfneigh.head()

plt.rcParams["axes.labelsize"] = 35
plt.rc('ytick', labelsize=30)
plt.rc('xtick', labelsize=30)
g = sns.FacetGrid(dfneigh, size=14, aspect=3)
(g.map(sns.boxplot, "Neighborhood", 0, palette="PRGn"))
g.ax.set( ylabel='Residuals')
plt.title('Residuals by Neighborhood', size=40)

#Calculate Price per Square Foot
dfneigh['PricePerSF'] = dfneigh['SalePrice'] / (dfneigh['TotalBsmtSF'] + dfneigh['FirstFlrSF'] +
dfneigh['SecondFlrSF'])

dfneigh.head(2)

#subset by neighborhood
```

```
Blmngtn = dfneigh[dfneigh['Neighborhood'] == 'Blmngtn']
Blueste = dfneigh[dfneigh['Neighborhood'] == 'Blueste']
BrDale = dfneigh[dfneigh['Neighborhood'] == 'BrDale']
BrkSide = dfneigh[dfneigh['Neighborhood'] == 'BrkSide']
ClearCr = dfneigh[dfneigh['Neighborhood'] == 'ClearCr']
CollgCr = dfneigh[dfneigh['Neighborhood'] == 'CollgCr']
Crawfor = dfneigh[dfneigh['Neighborhood'] == 'Crawfor']
Edwards = dfneigh[dfneigh['Neighborhood'] == 'Edwards']
Gilbert = dfneigh[dfneigh['Neighborhood'] == 'Gilbert']
Greens = dfneigh[dfneigh['Neighborhood'] == 'Greens']
GrnHill = dfneigh[dfneigh['Neighborhood'] == 'GrnHill']
IDOTRR = dfneigh[dfneigh['Neighborhood'] == 'IDOTRR']
Landmrk = dfneigh[dfneigh['Neighborhood'] == 'Landmrk']
MeadowV = dfneigh[dfneigh['Neighborhood'] == 'MeadowV']
Mitchel = dfneigh[dfneigh['Neighborhood'] == 'Mitchel']
NAmes = dfneigh[dfneigh['Neighborhood'] == 'NAmes']
NoRidge = dfneigh[dfneigh['Neighborhood'] == 'NoRidge']
NPkVill = dfneigh[dfneigh['Neighborhood'] == 'NPkVill']
NridgHt = dfneigh[dfneigh['Neighborhood'] == 'NridgHt']
NWAmes = dfneigh[dfneigh['Neighborhood'] == 'NWAmes']
OldTown = dfneigh[dfneigh['Neighborhood'] == 'OldTown']
SWISU = dfneigh[dfneigh['Neighborhood'] == 'SWISU']
Sawyer = dfneigh[dfneigh['Neighborhood'] == 'Sawyer']
SawyerW = dfneigh[dfneigh['Neighborhood'] == 'SawyerW']
Somerst = dfneigh[dfneigh['Neighborhood'] == 'Somerst']
StoneBr = dfneigh[dfneigh['Neighborhood'] == 'StoneBr']
Timber = dfneigh[dfneigh['Neighborhood'] == 'Timber']
Veenker = dfneigh[dfneigh['Neighborhood'] == 'Veenker']

dfneigh.groupby('Neighborhood').count()

#check
print len(dfneigh)
#removed neighborhoods that are not present in dataset
print len(Blmngtn) + len(BrkSide) + len(ClearCr) + len(CollgCr) + len(Crawfor) + len(Edwards) +
len(Gilbert) + len(IDOTRR) + len(Mitchel) + len(NAmes) + len(NWAmes) + len(NoRidge) +
len(NridgHt) + len(OldTown) + len(SWISU) + len(Sawyer) + len(SawyerW) + len(Somerst) +
len(StoneBr)+ len(Timber) + len(Veenker)

#list of neighborhoods in data subset
x = ["Blmngtn", "BrkSide", "ClearCr", "CollgCr", "Crawfor", "Edwards", "Gilbert", "IDOTRR",
"Mitchel", "NAmes", "NWAmes", "NoRidge", "NridgHt", "OldTown", "SWISU", "Sawyer",
"SawyerW", "Somerst", "StoneBr", "Timber", "Veenker"]
```

```
#find MAE of each neighborhood
import sklearn.metrics as sk
y = [sk.mean_absolute_error(Blmngtn['SalePrice'], Blmngtn['Fitted Value']),
sk.mean_absolute_error(BrkSide['SalePrice'], BrkSide['Fitted Value']),
sk.mean_absolute_error(ClearCr['SalePrice'], ClearCr['Fitted Value']),
sk.mean_absolute_error(CollgCr['SalePrice'], CollgCr['Fitted Value']),
sk.mean_absolute_error(Crawfor['SalePrice'], Crawfor['Fitted Value']),
sk.mean_absolute_error(Edwards['SalePrice'], Edwards['Fitted Value']),
sk.mean_absolute_error(Gilbert['SalePrice'], Gilbert['Fitted Value']),
sk.mean_absolute_error(IDOTRR['SalePrice'], IDOTRR['Fitted Value']),
sk.mean_absolute_error(Mitchel['SalePrice'], Mitchel['Fitted Value']),
sk.mean_absolute_error(NAmes['SalePrice'], NAmes['Fitted Value']),
sk.mean_absolute_error(NWAmes['SalePrice'], NWAmes['Fitted Value']),
sk.mean_absolute_error(NoRidge['SalePrice'], NoRidge['Fitted Value']),
sk.mean_absolute_error(NridgHt['SalePrice'], NridgHt['Fitted Value']),
sk.mean_absolute_error(OldTown['SalePrice'], OldTown['Fitted Value']),
sk.mean_absolute_error(SWISU['SalePrice'], SWISU['Fitted Value']),
sk.mean_absolute_error(Sawyer['SalePrice'], Sawyer['Fitted Value']),
sk.mean_absolute_error(SawyerW['SalePrice'], SawyerW['Fitted Value']),
sk.mean_absolute_error(Somerst['SalePrice'], Somerst['Fitted Value']),
sk.mean_absolute_error(StoneBr['SalePrice'], StoneBr['Fitted Value']),
sk.mean_absolute_error(Timber['SalePrice'], Timber['Fitted Value']),
sk.mean_absolute_error(Veenker['SalePrice'], Veenker['Fitted Value'])]

#find mean price per square foot for each neighborhood
z = [Blmngtn["PricePerSF"].mean(), BrkSide["PricePerSF"].mean(), ClearCr["PricePerSF"].mean(),
CollgCr["PricePerSF"].mean(), Crawfor["PricePerSF"].mean(), Edwards["PricePerSF"].mean(),
Gilbert["PricePerSF"].mean(), IDOTRR["PricePerSF"].mean(), Mitchel["PricePerSF"].mean(),
NAmes["PricePerSF"].mean(), NWAmes["PricePerSF"].mean(), NoRidge["PricePerSF"].mean(),
NridgHt["PricePerSF"].mean(), OldTown["PricePerSF"].mean(), SWISU["PricePerSF"].mean(),
Sawyer["PricePerSF"].mean(), SawyerW["PricePerSF"].mean(), Somerst["PricePerSF"].mean(),
StoneBr["PricePerSF"].mean(), Timber["PricePerSF"].mean(), Veenker["PricePerSF"].mean()]

#check
print len(x)
print len(y)
print len(z)

#view MAE and MeanPricePerSF by neighborhood
dfneighvalues = pd.DataFrame(list(zip(x,y,z)), columns = ['Neighborhood', 'MAE',
'MeanPricePerSF'])
print dfneighvalues.describe()
print dfneighvalues.sort(columns="MeanPricePerSF")
```

```
#scatterplot of MAW and MeanPricePerSF by neighborhood
sns.set()
sns.lmplot('MeanPricePerSF', 'MAE',
        data=dfneighvalues,
        fit_reg=False,
        hue="Neighborhood",
        legend=False,
        palette=("GnBu_d"),
        scatter_kws={"marker": "D",
                "s": 100})
plt.title('MeanPricePerSF vs MAE by Neighborhood')
plt.xlabel('MeanPricePerSF')
plt.ylabel('MAE')

#group neighborhoods into 5 groups
def grouping(x):
  if x["Neighborhood"] == "SWISU" or x["Neighborhood"] == "OldTown" or x["Neighborhood"]
== "IDOTRR" or x["Neighborhood"] == "Blmngtn" or x["Neighborhood"] == "Edwards":
    return 0
  elif x["Neighborhood"] == "BrkSide" or x["Neighborhood"] == "NAmes" or x["Neighborhood"]
== "NWAmes" or x["Neighborhood"] == "Sawyer":
    return 1
  elif  x["Neighborhood"] == "SawyerW" or x["Neighborhood"] == "Mitchel" or
x["Neighborhood"] == "ClearCr":
    return 2
  elif  x["Neighborhood"] == "CollgCr" or x["Neighborhood"] == "Crawfor" or x["Neighborhood"]
== "Veenker" or x["Neighborhood"] == "Gilbert" or x["Neighborhood"] == "Timber" or
x["Neighborhood"] == "Somerst" or x["Neighborhood"] == "NoRidge":
    return 3
  elif  x["Neighborhood"] == "NridgHt" or x["Neighborhood"] == "StoneBr":
    return 4
  else:
    return 5

dfneigh["Group"] = dfneigh.apply(grouping, axis=1)

#check
dfneigh[dfneigh["Group"] == 5]

# create dummy variables for Group
dummy_group = pd.get_dummies(dfneigh["Group"], prefix="Group")
#check
print dummy_group.head()
```

```
#combine dummy variables w/dataframe
dfneighreg = pd.concat([dfneigh, dummy_group], axis=1)

#check
dfneighreg.head(2)

#see number in each group
dfneighreg.groupby("Group").count()
#choose group 1 as reference group / base category

X = dfneighreg[['GrLivArea', 'YearBuilt', 'Group_0', 'Group_2', 'Group_3', 'Group_4']]
y = dfneighreg['SalePrice']

## fit a OLS model with intercept
X = sm.add_constant(X)
lm4 = smf.OLS(y, X).fit()

lm4.summary()

sk.mean_absolute_error(dfneighreg["SalePrice"], lm7.fittedvalues, sample_weight=None,
multioutput='uniform_average')
#this value is less than MAE of previous multiple regression: 26305

#do a log transformation on SalePrice
log_columns = ['SalePrice']
log_df = dfcontin.copy()
log_df[log_columns] = log_df[log_columns].apply(np.log10)

X = log_df[['GrLivArea', 'YearBuilt', 'GarageArea', 'TotalBsmtSF']]
y = log_df['SalePrice']

## fit a OLS model with intercept
lm5 = smf.OLS(y, X).fit()

lm5.summary()

#Histogram of residuals
plt.hist(lm5.resid)
plt.ylabel('Count')
plt.xlabel('Residuals')
plt.title('Histogram of Residuals')

#### Fitted vs Residuals
plt.scatter(lm5.fittedvalues, lm6.resid)
```

```
plt.xlabel('Predicted SalePrice')
plt.ylabel('Residuals')
plt.title('Predicted SalePrice vs. Residuals')

fig = sm.qqplot(lm5.resid, fit=True, line='45')
plt.show
plt.title('QQ Norm Plot of Residuals')

X = dfcontin[['GrLivArea', 'YearBuilt', 'GarageArea', 'TotalBsmtSF']]
y = dfcontin['SalePrice']

## fit a OLS model with intercept
lm6 = smf.OLS(y, X).fit()

lm6.summary()

#Histogram of residuals
plt.hist(lm6.resid)
plt.ylabel('Count')
plt.xlabel('Residuals')
plt.title('Histogram of Residuals')

#### Fitted vs Residuals
plt.scatter(lm6.fittedvalues, lm6.resid)
plt.xlabel('Predicted SalePrice')
plt.ylabel('Residuals')
plt.title('Predicted SalePrice vs. Residuals')

fig = sm.qqplot(lm6.resid, fit=True, line='45')
plt.show
plt.title('QQ Norm Plot of Residuals')
```