MSDS 6372 Project 2

Additional Analysis of Housing Prices

Introduction

With big data on the rise, there is a greater interest in wanting to use statistical tools to predict home values and find out what factors are the most significant for a home buyer when buying a home. The obvious factors such as square feet and number of bathrooms may not actually be the most important factors when determining a sale price. Using a sample of home sales from Ames, Iowa, between 2006 and 2010, we explore the relationships between the various factors that affect sale price to determine both an intuitively simple model as well as a more complex and predictive model.

This report is a second version of the report handed in for Project 1. Changes were made based on comments from Dr. Biven Sadler and two extra sections were added to address LDA and PCA. Specific changes based on comments are detailed in the "Addressing the Comments" section directly before the appendix.

Data Description

The data comes from a Kaggle tournament taken from Ames Iowa. Dean De Cock, who compiled the dataset obtained the data directly from Ames City Assessor's Office. While there were over 100 variables in the initial dataset, the dataset used for this analysis features 79 different explanatory variables that required no special knowledge or previous calculations. The variables range from including information on "condition" and "roof type" to "size of living area".

A breakdown of just a few of the many notable variables:

- **`LotArea`** - Lot size in square feet
- **`SaleCondition`** - The type of sale (normal, foreclosure, etc)
- **`ScreenPorch`** - Screen porch area in square feet
- **`MasVnrArea`** - Masonry veneer area in square feet
- **`Condition1`** - Proximity to main road or railroad
- **`BldgType`** - Type of dwelling
- **`BsmtFinSF1`** - Type 1 finished square feet
- **`BsmtExposure`** - Walkout or garden level basement walls
- **`2ndFlrSF`** - Square feet of second floor
- **`GrLivArea`** - Above grade (ground) living area square feet

- **`RoofMatl`** - Roof material
- **`MSSubClass`**- Type of dwelling involved in the sale.

A complete list and explanation of all the variables looked at for this analysis are available in the appendix and and Kaggle.com.

Exploratory Analysis

Before we begin with data exploration and look for correlation, we performed a deep dive into the data. Our objectives were to look for missing values, identify outliers, and design strategies to clean and address these issues. Using SAS scripts, we found over 80 missing values within the dataset. Our first order was to fill in the values based on the variable explanations provided by the Kaggle project. Wherever applicable, we used "NA", 0, or "None" where appropriate. In situations where "NA" has a specific meaning, we used "NT" as the identifier. In cases where the value is not obvious, we perform record-level reconciliation to evaluate the appropriate response. For example, the observation with `ID` *#1916* has a missing value in `Utilities`. Examining other observations with the same `Neighborhood` and similar characteristics such as `SalePrice, Street`, and `Alley`, we determined the most appropriate value for that record should be "AllPub".

Through this process, we also identified other errors in the dataset. The observation with `ID` *#1299* has 4692 SqFt on the first level and 950 SqFt on the second level. However, it only has a total above ground area of 1426.9 SqFt. Due to this inconsistency between the `SalePrice`, `Neighborhood,` and `LotArea` explanatory variables, we were not able to determine which of the variables has the incorrect value thus we removed it from the train set.

In addition, we made sure that levels for factors in both the train and test set aligned. We identified that the observation with `ID` *#1556* (test set) has a `KitchenQual` value of "NA", which is not a recognized level in the train set. Using the same technique mentioned above, we assigned the specific value of "TA", a more appropriate identifier. A detailed list of changes is provided with the SAS code in the appendix and includes things like changing the spelling of the values that were different from the description like "Duplx" vs. "Duplex" for the `BldgType` variable. In all, heavy emphasis was placed on data wrangling prior to the start of the full statistical analysis.

After we had mostly exhausted all our data cleansing examinations, we began the actual data exploration to help construct a good model. The first step was reviewing the scatter plots

(Figures 1 and 2). From the plots, we were able to immediately identify some correlations among different variables. There are also signs of large outliers and possibly covariation. As expected, the size of the living area above ground (GrLivArea) has a strong correlation with sale price (Pearson's "r" correlation ≈ 0.73). You can see in the scatter plots below that there appears to be a high correlation to SalePrice between many variables.

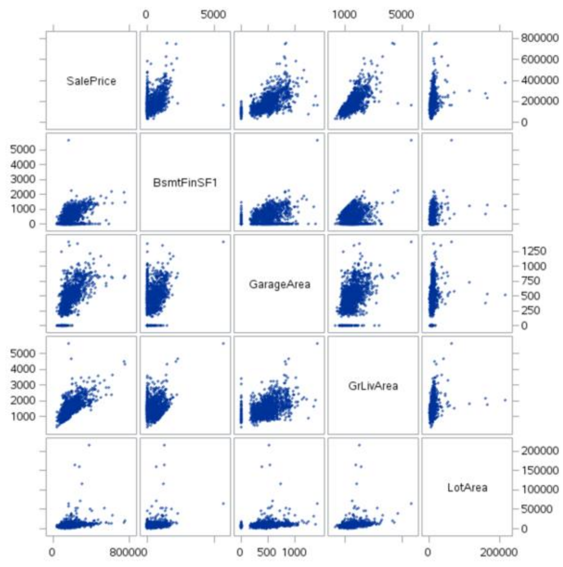


Figure 1 - Matrix of scatterplots - SalePrice vs continuous variables

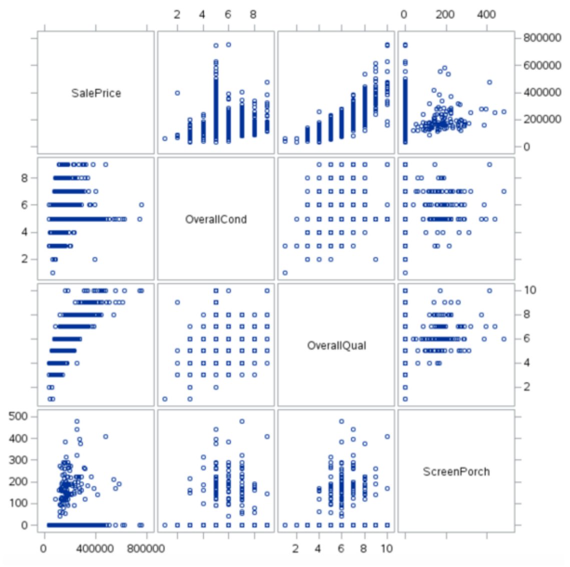


Figure 2 - Matrix of scatterplots - SalePrice versus ordinal variables

With Neighborhood, we theorized that there would be interesting interactions with other variables. We spent time analyzing its effect on SalePrice (Figure 1). As suspected, the location of the house has a large effect on the price of the home. One way to see this correlation is to simply look at the SalePrice vs. Neighborhood. The box-plots below show the variation in the price for each neighborhood.

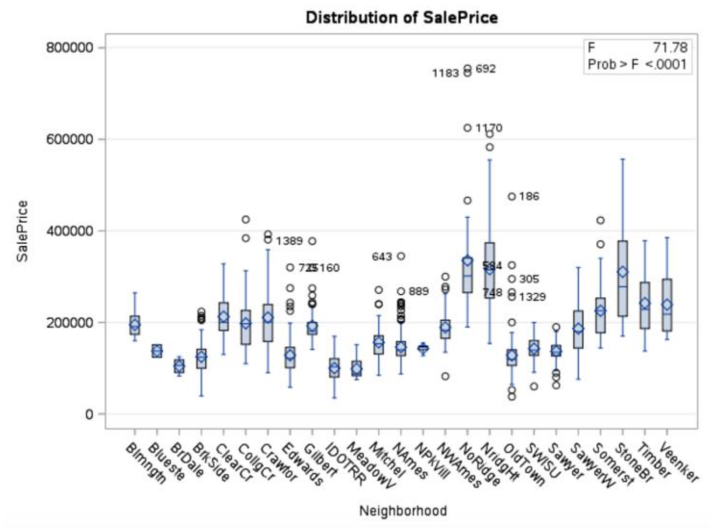


Figure 3 - Boxplots of SalePrice versus Neighborhood

Serial Correlation

An analysis was done to determine if a serial correlation exists in our dataset using the month and year of the sale. A plot of the data (Figure 4) shows no evidence of a serial correlation. Because we have 1460 observations in the train set, we are able to use a standard autocorrelation analysis to determine if this is just white noise.

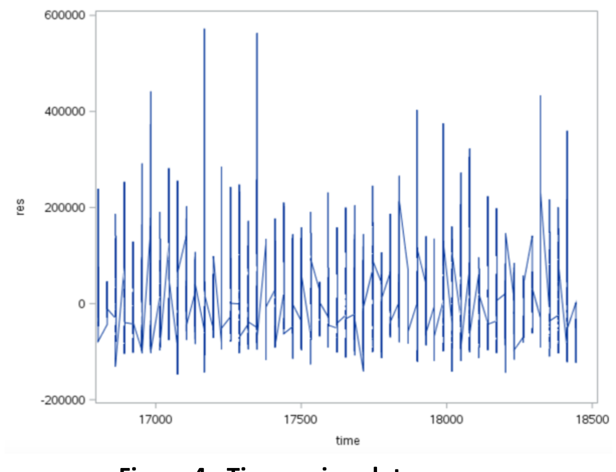


Figure 4 - Time series plot

Looking at the ACF/PACF output (Figure 5), in the autocorrelation analysis, we see no evidence of a serial correlation. A Durbin-Watson statistic of 1.9994 supports this claim.

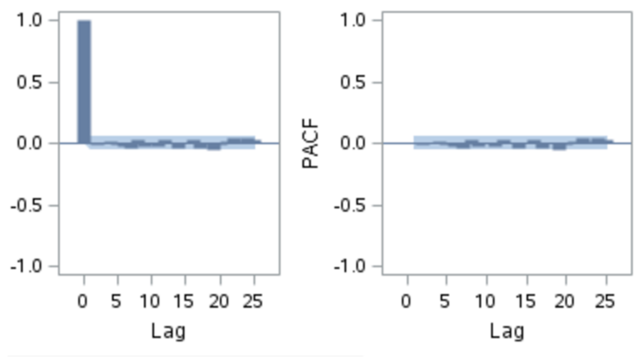


Figure 5 - ACF/PACF versus lag

Further exploration will require us to consider if grouping the observations by the month and year sold in combination will show evidence of a serial correlation between the means of each of these groups.

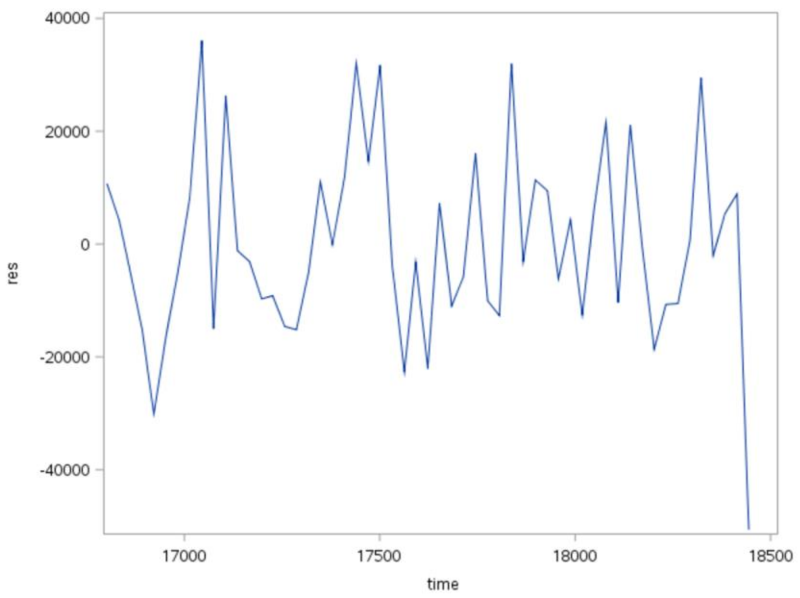


Figure 6 - Time series plot

There is not overwhelming evidence for a positive correlation from a plot of residual vs. time (Figure 6), but we will use autocorrelation to make sure (Figure 7). Although much more evidence exists here, than above, it is difficult to make a convincing argument for a first order autocorrelation.

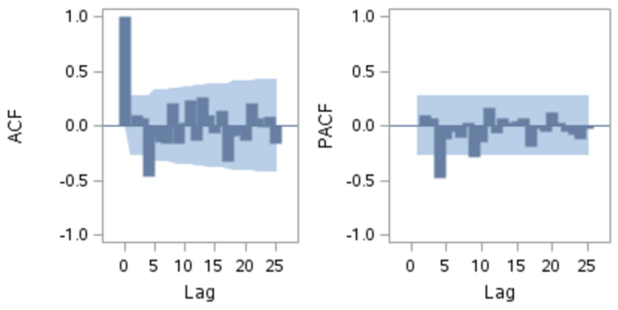


Figure 7 - ACF/PACF versus lag

A Durbin-Watson statistic of 1.81 and a Yule-Walker' Total R^2 of 0.09 seems to strongly suggest there is no measurable serial correlation.

Analysis: Question 1

When searching for variables that most simply and powerfully model the sale price of a home, we compare various selection techniques; specifically forward, backward, stepwise, and LASSO using cross validation. Because we are looking for a parsimonious model we chose to focus on variables produced by forward and stepwise selection methods. Using intuition and criterion compares, we then carefully examined the variables presented to us and removed variables that either overly-complicated the model or did not contribute to the model in a significant enough way. This is a subjective process, but ultimately does leave us with one "full model" which seems to do a good job of explaining a large portion of the variation in sale price with just 6 explanatory variables (4 categorical) and a total of 58 parameters. This gives us Model 1.1:

$$\text{SalePrice1} = \beta_0 + \beta_1\text{GrLivArea} + \beta_2\text{LotArea} + \beta_{3j}\text{MSSubClass} + \beta_{4j}\text{Neighborhood} + \beta_{5j}\text{OverallCond} + \beta_{6j}\text{OverallQual}$$

Where the parameter estimates are β_0, the intercept, with β_i and β_{ij} as corresponding parameter estimates for the variables and their levels. For instance, i corresponds to the variable; while j corresponds to the variable level. Each variable and variable level will have a different parameter estimate β_i or β_{ij}. Examples and tables of the values are provided below.

Exploring that perhaps the size of the living area of the house is more expensive depending on the neighborhood, we include an interaction term for

`Neighborhood*GrLivArea`. This increases the number of parameters to 82 and gives us Model 1.2:

$$\begin{aligned}\texttt{SalePrice2} = {} & \beta_0 + \beta_1\texttt{GrLivArea} + \beta_2\texttt{LotArea} + \beta_{3j}\texttt{MSSubClass} \\ & + \beta_{4j}\texttt{Neighborhood} + \beta_{5j}\texttt{OverallCond} + \beta_{6j}\texttt{OverallQual} \\ & + \beta_{7j}\texttt{Neighborhood*GrLivArea}\end{aligned}$$

Taking a different approach, we look at an oversimplified model which includes no categorical variables, but only continuous variables. With only 5 parameters, this seems by far the easiest model to interpret but, as we will see, performs poorly compared to the other models. This gives us Model 1.3:

$$\begin{aligned}\texttt{SalePrice3} = {} & \beta_0 + \beta_1\texttt{FullBath} + \beta_2\texttt{GrLivArea} + \beta_3\texttt{LotArea} \\ & + \beta_4\texttt{TotalBsmtSF}\end{aligned}$$

Table 1 below provides a statistical comparison of the 3 models. The residual plots show similar patterns among the 3 model and provide little indication of which is better. All models seem to have a few outliers. One could argue that the first two models have the most “well formed” cluster around the 0 residual line.

Looking at the Cook’s D and Studentized Residual plots from SAS, we can see that the first model appears to have the least number of influential outliers. The last model, in particular, has one observation which seems to be very influential and may need to be reviewed.

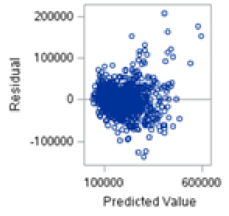
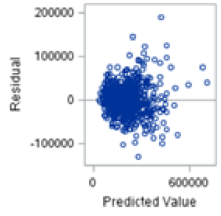
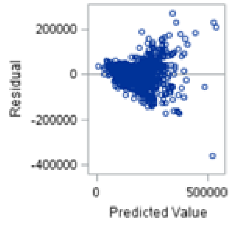
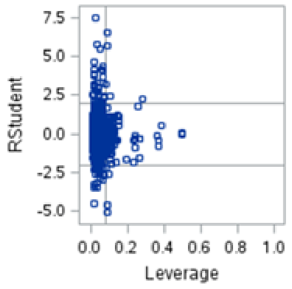
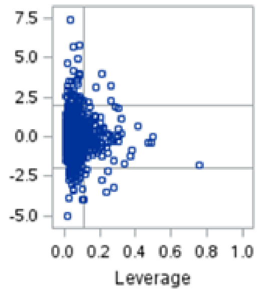
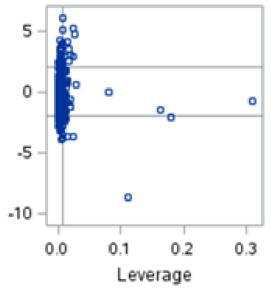
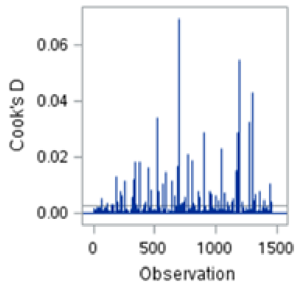
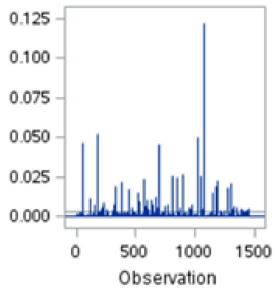
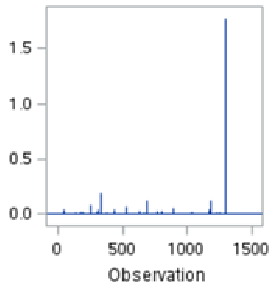
Model Statistical comparison			
	Model 1.1	Model 1.2	Model 1.3
R-Squared	0.8747	0.8935	0.6782
Adj. R-Squared	0.8697	0.8873	0.6773
Coeff. of Variance	15.853	14.7406	24.9425
			
Studentized Residual			
Cooks'D			
AIC	31490	31350	32761
BIC	30035	29889	31301
CV PRESS	1.275728E12	1.180003E12	3.035844E12

Table 1 - This table includes the fit and selection statistics for models 1.1-1.3.

To explore whether multicollinearity is an issue for these models, we look at the tolerance values from SAS' PROC GLM output. Since tolerance = 1 / VIF, and VIF > 10 is generally considered to be an issue, we instead look for tolerance < 0.1. There were no multicollinearity issues detected in Model 1.1 and 1.3. In Model 1.2, we found isolated collinearity issues in the interaction term `Neighborhood*GrLivArea`. This is not surprising since Model 1.2 has Neighborhood now in two parts of the model. This sort of collinearity is expected and generally not a large issue.[1]

Comparing our three models, Model 1.2 provides the best fit and makes the most sense. Although there was some concern about multicollinearity, the impact is small (24 interaction terms with average tolerance of 0.5). Model 1.2 provides a straightforward and easily interpretable picture of the significant factors that influence the final sale price of a home in Ames, Iowa.

In regards to our assumptions, the histogram of residuals (figure 8) shows no evidence against equal spread, additionally we can assume independence of observations and a linear relationship between the explanatory variables and sale price.

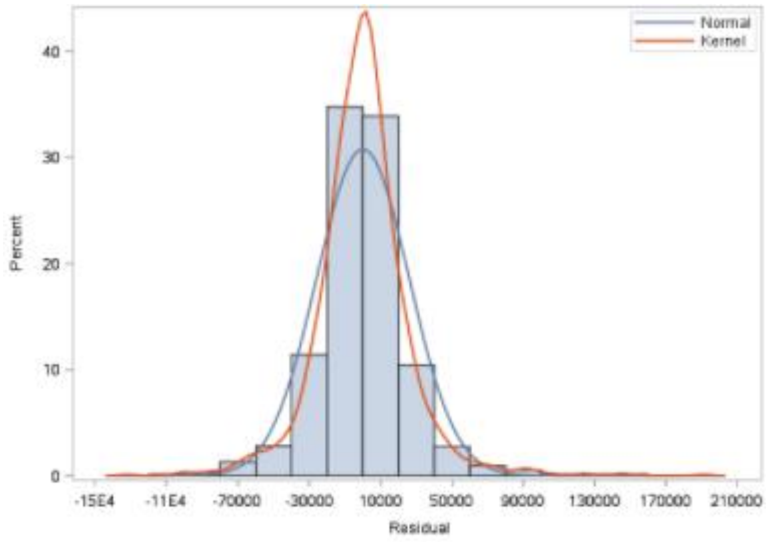


Figure 8 - Distribution of residuals for SalePrice

It should be noted that LASSO and external cross validation techniques were used to help choose our models. However, LASSO proved difficult and unwieldy while a manual external cross validation turned out to be a very powerful technique that we used to fine tune our predictive model in question 2.

Interpretation of Model 1.2:

$$\begin{aligned} \text{SalePrice} = {} & 65170.95 + (54.58)\text{GrLivArea} + (0.68)\text{LotArea} \\ & + \beta_{3j}\text{Neighborhood} + \beta_{4j}\text{OverallCond} + \beta_{5j}\text{OverallQual} \\ & + \beta_{6j}\text{Neighborhood*GrLivArea} \end{aligned}$$

In order to use this model, the user must have the information on the above variables. It is straightforward for continuous variables such as `GrLivArea` and `LotArea`. However, for the categorical variables, specific coefficients must be used (see Appendix I for a full list of parameter estimates, p-values, and confidence intervals). For example, assuming a house for sale with the following criteria:

```
GrLivArea = 2500 Sqft
LotArea = 5000 Sqft
MSSubClass = "50"
OverallCond = "7"
OverallQual = "6"
Neighborhood = "Gilbert"
```

$$\begin{aligned}\text{SalePrice2} = {} & \beta_0 + \beta_1\text{GrLivArea} + \beta_2\text{LotArea} + \beta_{3j}\text{MSSubClass} \\ & + \beta_{4j}\text{Neighborhood} + \beta_{5j}\text{OverallCond} + \beta_{6j}\text{OverallQual} \\ & + \beta_{7j}\text{Neighborhood*GrLivArea}\end{aligned}$$

The estimated `SalePrice`:

```
= 65170.95 + 54.58*(2500) + 0.6753*(5000)
+ (-9428.53) + (-31049.29) + (10926.21) + (12281.08) +
22.33*(2500)
= $243,551.92
```

with a 95% confidence interval is ($100,727.90, $383,328.66)

The specific references for each categorical variable that were used for this specific example are MSSubClass (ref = "20"), Neighborhood (ref = "NAmes"), OverallCond (ref = "5"), OverallQual (ref = "5"). These references were chosen because the median sale price for these levels fell roughly in the middle of their respective categories.

Interpreting these values is simple. The intercept for this case $65,170.95 would be the base cost of a home for the reference values. A β_1 of 54.58 means that every increase of `GrLivArea` (square foot of living space) is worth $54.58 for a 2500 square foot home. A β_2 of 0.6753 means that every square foot of area on the lot around the house is worth $0.6753 for a 5000 square foot lot. A β_3 of -9428.53 with `MSSubClass` (Building class) = `"50"` means that a “1-1/2 story finished all ages” home is worth $9428.53 less than the reference “1-story 1946 & newer all styles” home; all other things being held equal. A β_4 of -31049.29 in the “Gilbert” neighborhood is worth $31049.29 less than the reference home in the “NAmes” `Neighborhood` all other things being held equal. A β_5 of 10926.21 with `OverallCond = "7"` or `"Good"` home condition is worth $10926.21 more than the reference home of “5” or “Average”; all other things being held equal. A β_6 of 12281.08 with `OverallQual = "6"` or `"Above Average"` home quality is worth $12281.08 more than the reference home of “5” or “Average”; all other things being held equal. And finally a β_7 of 22.33 in the “Gilbert”

[redacted]

`Neighborhood` is worth \$22.33 more a square foot than the reference home in the "NAmes" `Neighborhood` all other things being held equal.

The most important things about this model are its simplicity and parsimony. Out of the original 79 variables it was determined that only 6 were necessary to obtain an adjusted R^2 of almost 0.89. By itself `GrLivArea` produces an adjusted R^2 of 0.54, while the interaction of `GrLivArea` and `Neighborhood` alone in the model produce an adjusted R^2 of 0.78. In the final analysis, these variables were chosen because they form a model that fits the data nearly as well as the final model in question 2 with half the number of variables. This model helps to show to interested parties (such as real estate agents and prospective home buyers/sellers) the most significant factors that impact the sale price of a home. The coolest part is this model makes sense. Location and size of the house have always been thought to be the key price indicators when selling or buying a home. Add in the size of the house lot, the type of building (`MSSubClass)`, and the overall condition and quality of the home, and you can account for nearly 95% of the variance in the housing prices.

Analysis: Question 2

For prediction, our first goal was to find a model that would minimize adjusted R^2 and provide best predictability. We were driven by the motto: "if you're not first, you're last". This led us to use a backward elimination of parameters, giving us a model with an adjusted R^2 of 0.95 (which includes 78 explanatory variables and 573 parameters). The complete model will not be included for brevity, but this will be considered to be Model 2.1 for prediction.

Unfortunately, Model 2.1 performed terribly when submitted in Kaggle. In fact, we were not even able to submit to Kaggle by itself as there were so many missing observations so we had to trim it down to get a score (thus the NA in the Kaggle submission summary for Model 2.1 below). We found that Model 2.1 was grossly overfitting the data. We continued to try different selection processes; LASSO, stepwise, forward, etc. It turns out that forward selection seemed to give us an initial model with the highest adjusted R^2 and overall best Kaggle score. We continued through this process but we were finding that the criterion from our own data (Adjusted R^2, AIC, etc.) seem to always overvalue our model when compared to what we got on Kaggle.

To compensate for the first model deficiency, we experimented with grouping within variables. The idea is to minimize the influence of some of the outliers while preserving the correlation effect with the sale price. Reducing the number of levels for a category will also increase the degrees of freedom. Special care was taken to preserve the significance of each level by looking carefully at how significant the difference was between each level within a category. For example, a comparison using a *pdiff* or Bonferroni adjustment was run on `Neighborhood` and similar categories to look at the significant differences between levels with respect to `SalePrice`. These noted differences, along with side-by-side box-plot comparisons helped to regroup most of the categorical variables into new variables with fewer levels. This was done to both numerical and character categorical variables. After the levels within a category were combined, the new groupings were tested using an external cross validation on the original train dataset. The most notable variables that were regrouped are `Neighborhood, Condition1, BldgType, GarageType, BsmtExposure, BsmtQual, RoofMatl, SaleCondition`, and `Functional.`

The `Neighborhood` variable, which has 25 levels, was particularly difficult to regroup. It was found through trial and error using our external cross validation technique that `Neighborhood` could be grouped into a new variable that had only 12 levels and increase our prediction metrics significantly! In fact, this particular regrouping decreased our Kaggle score by nearly 10%. Through this trial and error process, we were able to identify the other new variables that helped to improve the model for best prediction.

The manual external cross validation technique, mentioned above, was chosen because it used the same metric that the Kaggle website uses to check submitted predictions: Root Mean Squared Logarithmic Error or RMSLE.[1] The cross validation set was created using the original train set. A random number was assigned to each observation in the train set and then the dataset was split in half using these random numbers to create a new train and test set. The data was then fit using proc glm and the new train set to make predictions for the new test set. These predictions were then used to calculate the RMSLE described on the Kaggle website, which in essence is a method for calculating the average total logarithmic difference between the predicted value and actual values. This technique was also used to refine our final model.

Model 2.2 was the first model we selected using this external cross validation technique and RMSLE. It produced a model with 18 variables. Model 2.2 includes the grouping of several new categorical variables (`OverallCondGroup`, `OverallQualGroup`, `NeighborhoodGroup`), which seemed to greatly help with overfitting. Also, an interaction term `GrLivArea*Neighborhood` was seen to help as well.

After several more iterations of adding, removing, and grouping variables by hand, the final model was uncovered which gave us our best to date Kaggle score. The only difference between this new Model 2.3 and the previous Model 2.2 is that the new model includes the newly grouped variable `Condition1Group` as well as the interaction term `GrLivArea*RoofMatl`. The following is the final model used:

$$\begin{aligned}\texttt{SalePrice} = {} & \beta_0 + \beta_1\texttt{2ndFlrSF} + \beta_2\texttt{MasVnrArea} + \beta_3\texttt{3SsnPorch} + \beta_4\texttt{BsmtFinSF1} \\ & + \beta_5\texttt{GarageArea} + \beta_6\texttt{LotArea} + \beta_7\texttt{ScreenPorch} + \beta_{8j}\texttt{BldgType} \\ & + \beta_{9j}\texttt{BsmtExposure} + \beta_{10j}\texttt{BsmtQual} + \beta_{11j}\texttt{Condition1Group} \\ & + \beta_{12j}\texttt{Condition2} + \beta_{13j}\texttt{KitchenQual} + \beta_{14j}\texttt{RoofMatl} + \beta_{15j}\texttt{SaleCondition} \\ & + \beta_{16j}\texttt{NeighborhoodGroup} + \beta_{17j}\texttt{OverallCondGroup} \\ & + \beta_{18j}\texttt{OverallQualGroup} + \beta_{19j}\texttt{GrLivArea*NeighborhoodGroup} \\ & + \beta_{20j}\texttt{GrLivArea*RoofMatl}\end{aligned}$$

Table 2 below provides a statistical comparison of the 3 models. The residual plots show similar patterns among the 3 models. All models seem to have a few outliers. One could argue that the last two models have the most "well formed" cluster around the 0 residual line. Looking at the Cook's D and RStudent plots from SAS, we can see that there may be a few issues with outliers. However, since the objective is prediction some of these outliers may be remove later and then the model checked with cross validation to determine if the prediction metrics are better.

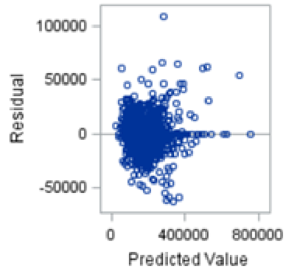
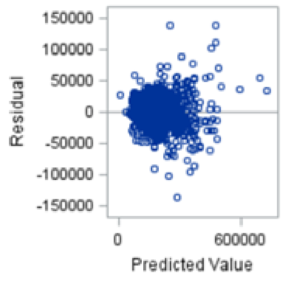
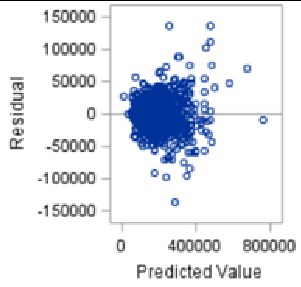
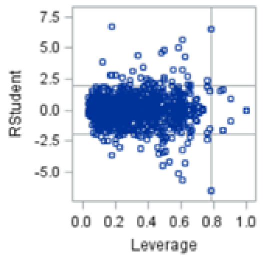
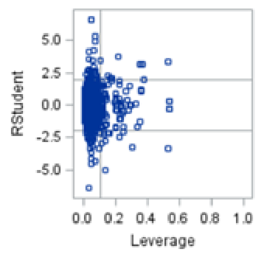
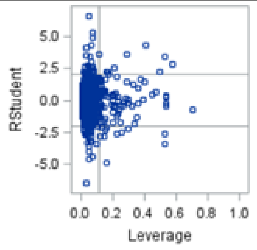
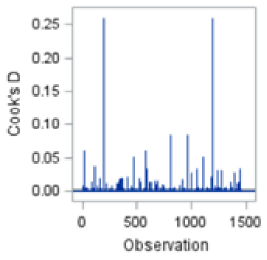
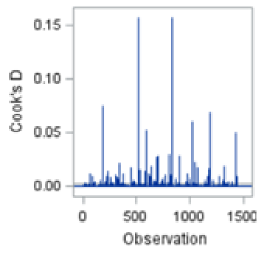
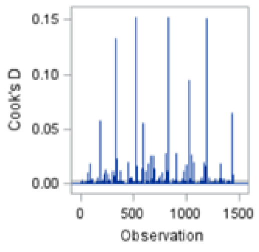
Model Statistical comparison			
	Model 2.1	Model 2.2	Model 2.3
R-Squared	0.9206	0.9229	0.9272
Adj. R-Squared	0.9157	0.9176	0.9236
Coeff. of Variance	10.02468	12.06299	11.94465
Residual			
Studentized Residual			
Cooks'D			
AIC	30879	30787	30721
BIC	29360	29324	29263
CV PRESS	1.025297E12	9.047722E11	9.054827E11
RMSLE (from our external CV)	NA	0.13297	0.13297

Table 2 - This table includes the fit and selection statistics for models 2.1-2.3.

Similar to Question 1, the histogram of residuals (below) shows no evidence against equal spread, additionally we can assume independence of observations and a linear relationship between the explanatory variables and sale price.

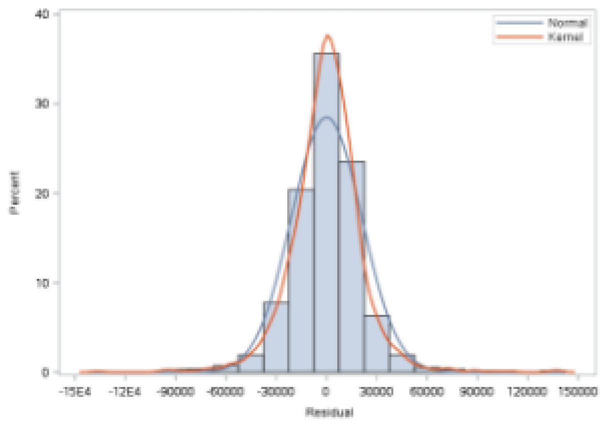


Figure 9 - Distribution of residuals for SalePrice

The Kaggle submission summary:

Model 2.1: Kaggle score = NA. No Kaggle score was produced for this model. Because there were over 50 variables in this model there were simply too many mismatches between the levels of the training and test data sets. However, our first submission to Kaggle using Model 2 from Question 1 produced a Kaggle score of 0.16010.

Model 2.2: Kaggle score = 0.13722. This model was our first big improvement of our Kaggle score after we started using external cross validation.

Model 2.3: Kaggle score = 0.13474. This model was the last big improvement using the new level regrouping technique along with external cross validation.

For several days, Model 3 was our best Kaggle submission and nothing else we tried seemed to improve our Kaggle score. The very last thing that we did was to pull out all the stops to get the best Kaggle score we could. We were ruthless in this regard and have tried several last ditch and some perhaps highly questionable techniques. If we had more time for this project we would rewrite certain parts of this report with updated models.

In what we call our final Kaggle score minimization project, we were finally able to see some further reduction in our Kaggle score by using a log transformation on several of the continuous variables as well as on `SalePrice`. Using a log transformation on the prediction variable was a bit tricky but an exponential transformation was done after the prediction to transform it back to a form that could be submitted to Kaggle. After that the only thing left to do was to clean up any observations that had high Cook’s D values. In this last effort, we simply started deleting the observations with the largest Cook’s D values until our cross validation prediction metrics bottomed out. In the end this dirty trick improved our final Kaggle score from 0.13474 to 0.12611.

The Kaggle submission summary continued:

Final best Model from project 1: Kaggle score = 0.12611. As discussed directly above, this score was obtained using a log transformation of most of the continuous variables as well as removing most of highest Cook's D observations. Our goal was to try anything that could reduce the score. And as you can see this final push illustrates that complicated models with a large diversity of data and variables will have many outliers that can play a significant role in prediction accuracy.

If we spent more time on this, we would do a couple of other things. The first would be to use several different seeds for our external cross validation RMSLE checks. We would write some code to use at least 10 different seeds one after the other to have a metric that was not overly dependent on one random train and test set. This would allow us to fine tune our model on a more diverse set. Right now, we are basically squeezing a damp cloth for that one final drip of water. And finally, we would run more proc glmselect methods on the log transformed model.

It should be noted that in project 2, detailed below, all of these things were tried and more.

The following two sections, LDA and PCA, were added specifically for project 2.

Linear Discriminant Analysis

A new task for project 2 requires solving for missing values within our dataset. The categorical variable "`Foundation`" was looked at and includes the following levels:

- `Slab`
- `Stone`
- `Wood`
- `BrkTil` - "Brick & Tile"
- `CBlock` - "Cinder Block"
- `PConc` - "Poured Concrete"

Using `proc discrim` we were able to determine, with roughly 85% accuracy, the foundation of each house. To do this, we first used `proc stepdisc` to narrow down the most essential explanatory variables that might help us predict the foundation. We narrowed down the number of variables a bit more by using a combination of intuition and by monitoring the accuracy provided by the `proc discrim` result. We used mostly continuous variables and a few categorical and ordinal variables such as `OverallCond` (Overall Condition - ranked from 1-10). To specify the prior probabilities of group membership, we leveraged the priors statement and used the actual distribution from the test dataset. (See code in appendix with title “LDA Script”.)

One way we added numeric variables was to find some categorical variables which we could make ordinal. One such example is `KitchenQual` (Kitchen Quality), which is ranked from “Poor” to “Excellent”. We made “Poor” equal 0 and “Excellent” equal to 5, with ascending values in between (1, 2, 3, 4, 5). We found that the only ordinal variable which seemed to significantly help determine `Foundation` was `BsmtQual` (Basement Quality). This is understandable since different foundations are generally associated with different quality basements.

All together 21 explanatory variables were used to help determine `Foundation` for each observation. We used the `train` data set as the “input” for `proc discrim` and we used the `test` data set as the “output” data set, which received the predicted value of `Foundation`. An internal cross validation was used in `proc discrim`.

To check if our assumptions were met, we considered several questions. The assumptions needed for a MANOVA are difficult to meet as we cannot really say that a regression line exists, in the traditional sense, as the response variable is qualitative in nature instead of quantitative. We should continue with caution with this assumption.

Additionally, we considered a matrix of scatterplots versus values selected separated by each level of `Foundation`. Figure 10 below shows four of the six levels of `Foundation` and how they correlate with other variables.

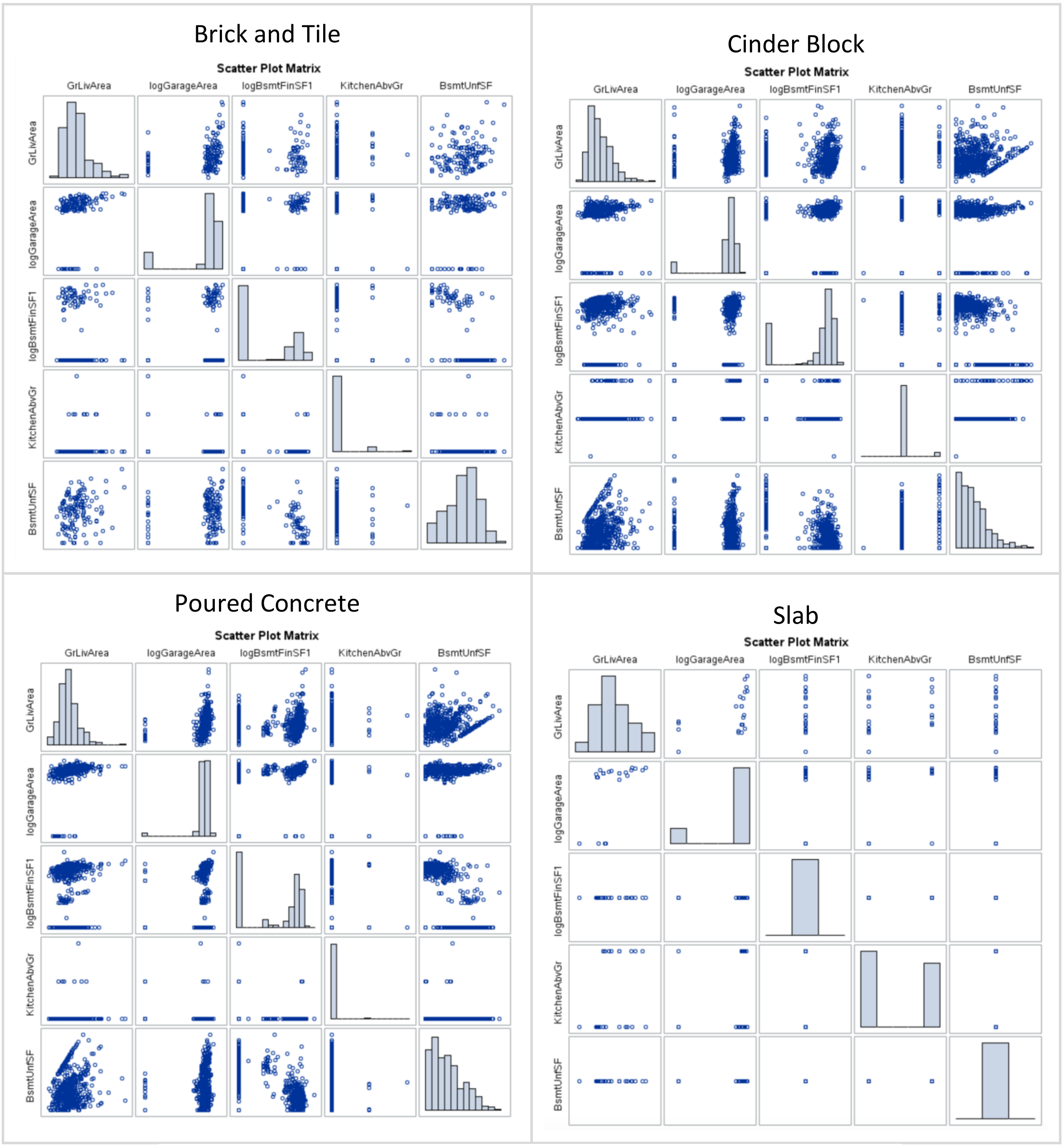
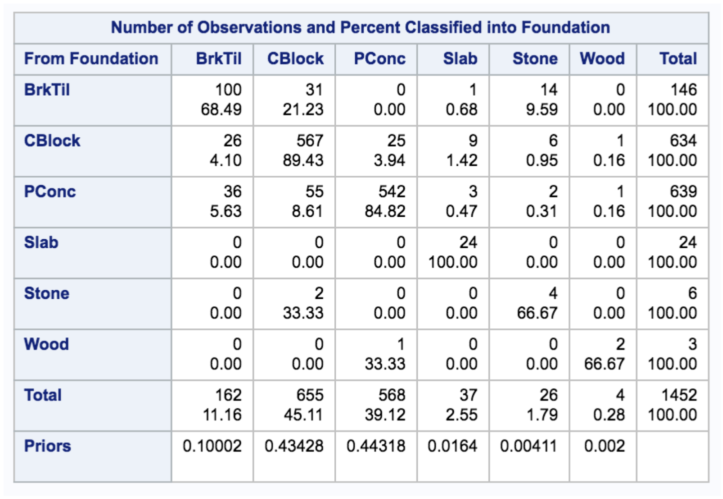


Figure 10 - Matrix scatterplots of explanatory variables for levels of Foundation

Wood has a very small n, (3 observations), therefore it is difficult to confirm that the assumptions are met. The same goes for `Stone` (6 observations) and we should be cautious with `Slab` (24 observations) . It should be noted that the three variables in the center (`logGarageArea`, `logBsmtFinSF1`, and `KitchenAbvGr`) are left skewed due to some houses not having that given feature (such as pool, kitchen, basement) at all. We will attempt to infer from the first 3 figures a general linearity of the data although we should proceed with caution.

Ultimately we used `proc discrim`'s `test="pool"` parameter to determine if Linear Discriminant Analysis would be an adequate representation of the data. From the output of `proc discrim`, it appears to be adequate (p-value = 1 with chi-square distribution using Bartlett's test). Looking at the confusion table from `proc discrim` (Table 3 below), we are able to see that `BrkTil` (Brick & Tile) was often confused (about 21% of the time) to be `CBlock` (Cinder Block) and then again with `Stone` (almost 10% of the time). It would be a worthwhile question to consider, in another study, why the level `BrkTil` seems to be so easily misclassified as other materials while `Slab` was never chosen to be any other material. One would guess that some of these could very well be mislabeled. After all, different realtors simply checked boxes on a form. It must be that `Slab` and `Wood` are very hard to mistake for other materials. This makes sense. And that combined with so few houses with those materials it is not too surprising to see those successfully classified. However, `BrkTil` presents a more difficult problem. Brick and Tile (`BrkTil`) typically corresponds to large clay bricks and can be confused with concrete blocks.[2] This could account for the confusion and large number of misclassified values of `BrkTil` as `CBlock`.



Number of Observations and Percent Classified into Foundation

From Foundation	BrkTil	CBlock	PConc	Slab	Stone	Wood	Total
BrkTil	100 68.49	31 21.23	0 0.00	1 0.68	14 9.59	0 0.00	146 100.00
CBlock	26 4.10	567 89.43	25 3.94	9 1.42	6 0.95	1 0.16	634 100.00
PConc	36 5.63	55 8.61	542 84.82	3 0.47	2 0.31	1 0.16	639 100.00
Slab	0 0.00	0 0.00	0 0.00	24 100.00	0 0.00	0 0.00	24 100.00
Stone	0 0.00	2 33.33	0 0.00	0 0.00	4 66.67	0 0.00	6 100.00
Wood	0 0.00	0 0.00	1 33.33	0 0.00	0 0.00	2 66.67	3 100.00
Total	162 11.16	655 45.11	568 39.12	37 2.55	26 1.79	4 0.28	1452 100.00
Priors	0.10002	0.43428	0.44318	0.0164	0.00411	0.002	

Table 3 - Confusion table of all levels of Foundation

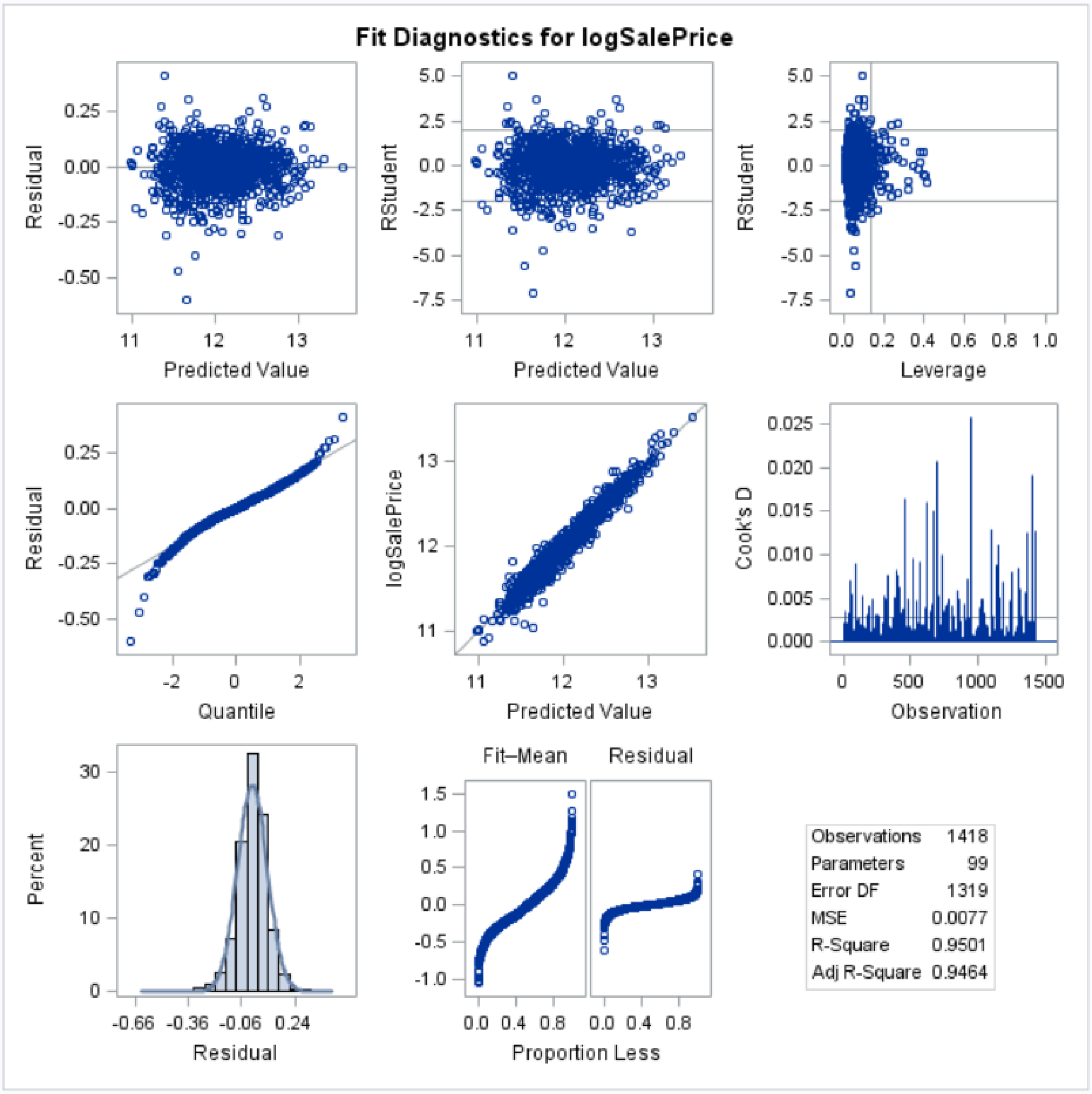
Principal Components Analysis

Principal Components Analysis (PCA) was used to improve our model and better predict `SalesPrice`. PCA is a statistical procedure using orthogonal transformations that convert observations of potentially correlated values into linearly uncorrelated ones. Four principal components out of 24 were found to improve the final model. These components were fashioned by using `proc prinqual` and `proc princomp` and were taken from a mixture of continuous and ordinal variables that had predominantly not been used in the best model from question 2. In a sense, we took the variables that we found tended to overfit the data individually and used PCA to combine them in a way that would make it easier to select only the components of those variables that would help the prediction metrics.

Additional ordinal variables were first created using certain nominal variables that used a ranking for their levels. We then took all the ordinal variables that were not part of our best model and used `proc prinqual` to transform them with the monotone transformation. This ensured the model would not be over-fitted by using the same variables that have already been identified as high value predictors. We then combined this with the continuous variables that were also predominantly excluded in our previous best model. The combined dataset was then transformed with the `princomp` procedure.

It is useful to note that PCA is a non-parametric analysis that gives unique and independent results. There is also no need for a specific distribution of the data and hence nominal, ordinal, discrete, and continuous variables can be used. PCA assumes linearity in the sense that the data is transformed into linear combinations. PCA also assumes that large variances have importance and places the components with large variances first while those with lower variances are placed towards the end. PCA gives an importance to the mean and covariance and there is no guarantee that the directions of maximum variance will help more or contain more features for better discrimination.

The variables used in this PCA are ones that are mainly a mix of ordinal and continuous variables that were not used in our base model. Many of them do not have normal distributions. In the end, the assumptions of linear regression are the most important. Since we intend to use the PCs to improve the regression, we are hoping that PCA will remove the correlation among the chosen variables. That way a few PCs could, perhaps, explain part of the variance that has been left unexplained by the current best model. Looking at fitness plots (Figure 11) of Model 3.3, we see that the assumptions for linear regression hold up nicely. The residuals show a great deal of normal behavior and the Cooks D plots show minor fluctuations.



Observations	1418
Parameters	99
Error DF	1319
MSE	0.0077
R-Square	0.9501
Adj R-Square	0.9464

Figure 11 - This figure shows the fit diagnostics for Model 3.3.

The result of this Principal Component Analysis yielded 24 principal components. Using visual inspection of the scree plot (Figure 12), we noticed the "elbow" of the curve to be at around 4 to 5. However, the first four principal components did not all improve the prediction metrics or the Kaggle score. It was found that the best prediction metrics were obtained when principal components 1, 12, 18, and 20 were added to the best model. This is shown more clearly in Table 4 below. Adding the other PCs increased R^2 but only worsened the prediction metrics. We took that to mean that the bulk of the PCs were contributing to overfitting and we then did careful variable selection with the PCs to choose the ones that together improved the prediction metrics and ultimately the Kaggle score.

[redacted]

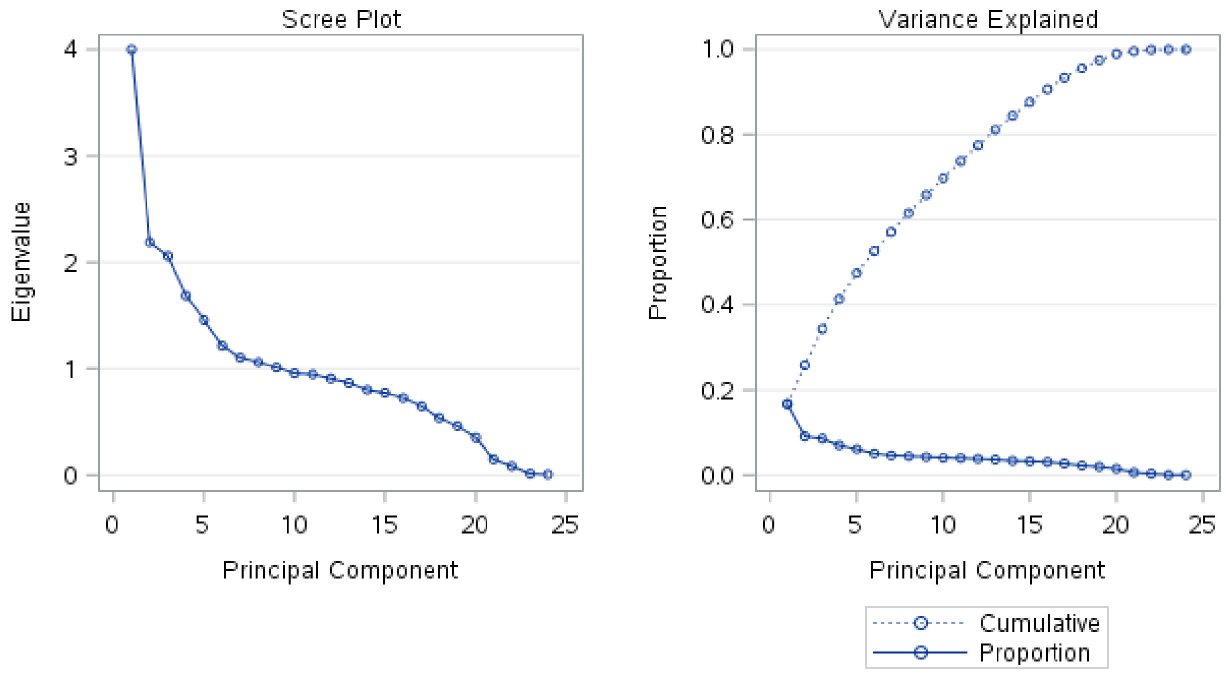


Figure 12 - Scree plot

Each of these principal components contain a portion of a set of variables that account for variation in the data. Because only three of the eighteen continuous variables were in both the best model and the PCs, we felt confident that some of these principal components would be able to provide increased prediction to our original model. Using iterative modeling from `proc glmselect`, we added different principal component variables individually into the model and obtained the statistical results. We were able to finally identify a model that provided a better fit by leveraging the Kaggle score. The goal of the iterative process is to get the best prediction possible. Throughout this lengthy process, square and cubic terms of other variables were also tried. Hundreds of models were generated, and we finally settled on this model,

Model 3.3:

$$\begin{aligned}\text{LogSalePrice} = {} & \beta_0 + \beta_1\text{log2ndFlrSF} + \beta_2\text{OpenPorchSF} + \beta_3\text{BsmtFinSF1} \\ & + \beta_4\text{GarageArea} + \beta_5\text{logLotArea} + \beta_6\text{ScreenPorch} \\ & + \beta_{7j}\text{BldgType} + \beta_{8j}\text{BsmtExposure} + \beta_{9j}\text{BsmtExposure} \\ & + \beta_{10j}\text{BsmtQual} \\ & + \beta_{11j}\text{Condition1Group} + \beta_{12j}\text{KitchenQual} \\ & + \beta_{13j}\text{SaleCondition} + \beta_{14j}\text{NeighborhoodGroup} \\ & + \beta_{15j}\text{RoofMatl} + \beta_{16j}\text{MSZoning} \\ & + \beta_{17j}\text{Exterior1st} + \beta_{18j}\text{CentralAir} + \beta_{19j}\text{PoolQC}\end{aligned}$$

[redacted]

$$
\begin{aligned}
&+ \beta_{20}\texttt{OverallCond} + \beta_{21}\texttt{OverallQual} \\
&+ \beta_{22}\texttt{OverallCond*OverallCond} \\
&+ \beta_{23}\texttt{OverallQual*OverallQual} + \beta_{24}\texttt{YearBuilt} \\
&+ \beta_{25}\texttt{YearRemodAdd} + \beta_{26}\texttt{FullBath} + \beta_{27}\texttt{GarageCars} \\
&+ \beta_{28}\texttt{FirePlaces} + \beta_{29}\texttt{FullBath*FullBath} \\
&+ \beta_{30}\texttt{BsmtFinType1Ord} + \beta_{31}\texttt{FunctionalOrd} \\
&+ \beta_{32}\texttt{logGrLivArea*NeighborhoodGroup} \\
&+ \beta_{33}\texttt{logGrLivArea*PoolQC} \\
&+ \beta_{34}\texttt{log2ndFlrSF*logLotArea} \\
&+ \beta_{35}\texttt{Prin1}+ \beta_{36}\texttt{Prin12} + \beta_{37}\texttt{Prin18} + \beta_{38}\texttt{Prin20}
\end{aligned}
$$

Selected model with selected principal components

Model 3.1 is the same as Model 3.3 above except for the exclusion of the PC terms. Model 3.2 is the same as Model 3.1 except the inclusion of the first 15 PCs. Model 3.3 detailed directly above illustrates the moderate effectiveness of adding a selected small portion of the principal components taken from the variables that did not seem to contribute significantly to prediction. Evidently small portions of the unused variables explained part of the variance that the other variables could not. Table 4 below shows a comparison of the statistical metrics for the 3 models.

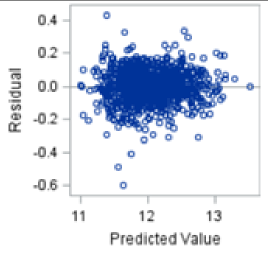
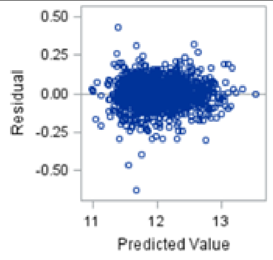
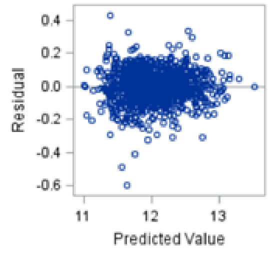
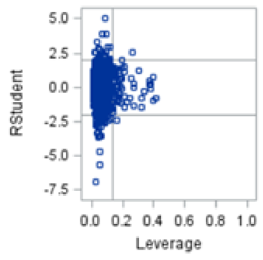
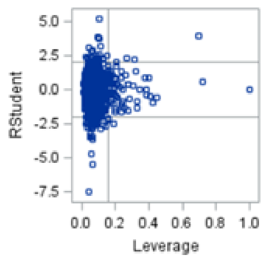
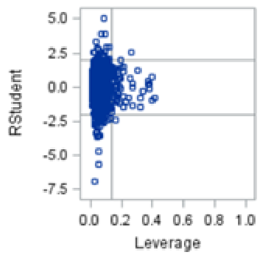
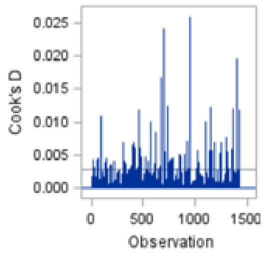
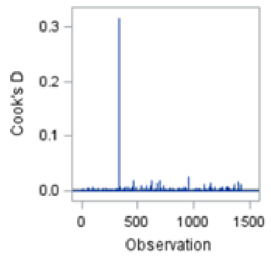
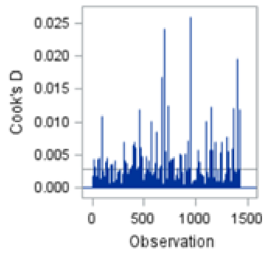
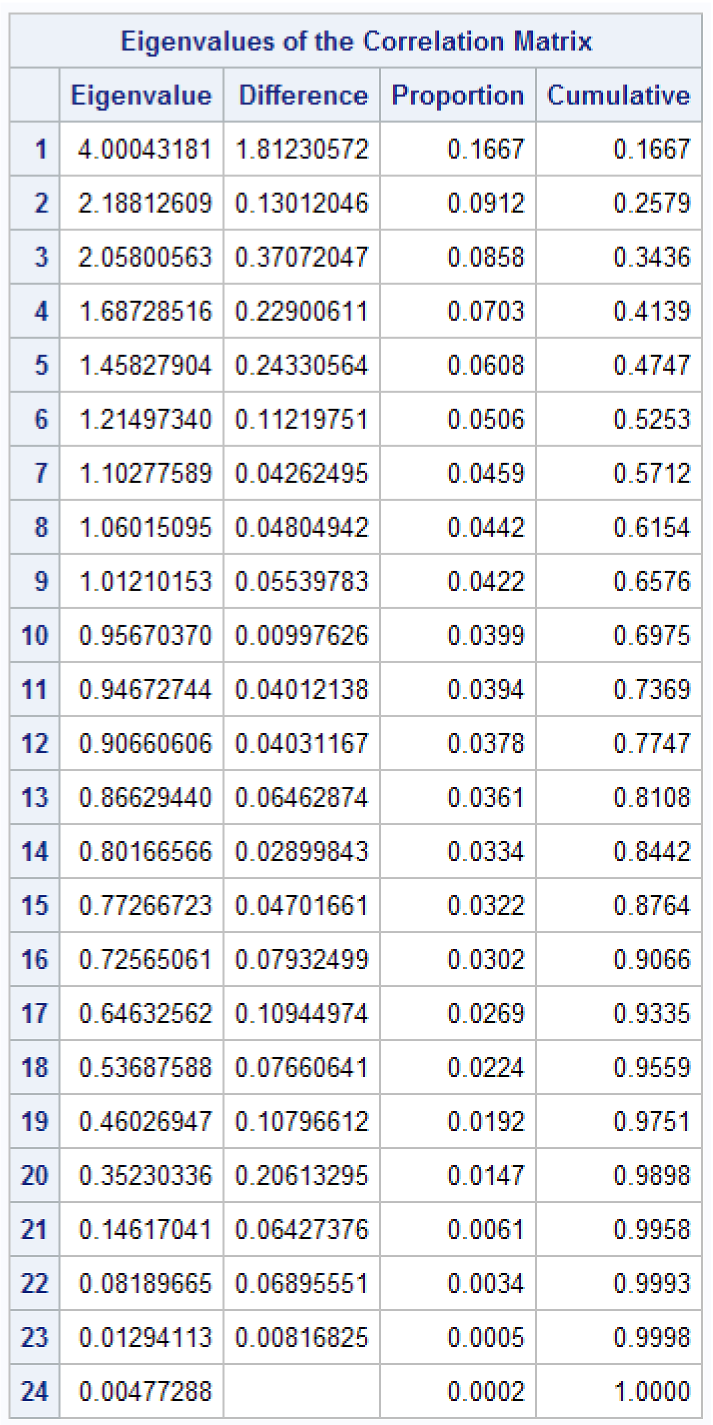
Model Statistical comparison			
	Model 3.1 Without PCs	Model 3.2 With 16 PCs	Model 3.3 With 4 Selected PCs (Final model)
R-Squared	0.9478	0.9503	0.9501
Adj. R-Squared	0.9441	0.9462	0.9464
Coeff. of Variance	0.745435	0.731443	0.729820
Residual			
Studentized Residual			
Cooks'D			
AIC	-5357.74334	-5367.47556	-5383.89989
BIC	-6285.79363	-6209.20526	-6283.45663
CV PRESS	14.56042	21.45117	14.66273
Kaggle Score	0.12065	0.22027	0.11991
RMSLE (from our external CV)	0.095459	0.12397	0.093887

Table 4 - The statistical metrics for the three models is shown. The final model outperforms in RMSLE score and Kaggle score.

This was a difficult set of data to apply principal component analysis. As stated previously the large number of variables and the wide variety of categorical and continuous variables with interactions made it difficult to simply apply PCA to all the variables at once. In fact, we tried a great many different approaches. We first tried doing PCA on all the variables in our best model. The categorical and ordinal variables were transformed and then run through `princomp` with the continuous variables added in. Interaction variables were made to facilitate this and were run through `princomp`. It became very difficult to do variable selection and the essence of some of the interaction terms, especially between continuous and categorical, were lost. It was difficult to look at the eigenvalue and eigenvector tables and gain any substantial information about variable selection. Even after clever applications of glmselect it seemed like some of the key information to help explain the variance in `SalePrice` was lost. The prediction metrics and Kaggle scores for those attempts were very poor and we moved on. Next, we tried looking at only the continuous variables, then ordinals by themselves, and even categorical variables by themselves. In each attempt variable selection was difficult and key information seemed to be lost during the transformation. This finally lead us to only using PCA on the variables that did not seem to help the prediction. We then focused on ordinal and continuous variables left out of the best model. As explained above, Model 3.3 contains only 4 of the PCs from the full set of 24 ordinal and continuous variables used in our PCA. Again, variable selection was very difficult and a combination of glmselect and external cross validation was used to select the 4 PCs ultimately used. The Kaggle score was the ultimate decider but the 4 PCs used did individually reduce the cross validation RMSLE score.

Picking the principal components was difficult as very little variation is explained by each iteration of component selection. Looking at the table of eigenvalues below (table 5), we see that the first PC (`Prin1`) explains only roughly 16% of the cumulative variance. Indeed, we need to include 18 (of the possible 24) components to explain 95% of the cumulative variance.



Eigenvalues of the Correlation Matrix				
	Eigenvalue	Difference	Proportion	Cumulative
1	4.00043181	1.81230572	0.1667	0.1667
2	2.18812609	0.13012046	0.0912	0.2579
3	2.05800563	0.37072047	0.0858	0.3436
4	1.68728516	0.22900611	0.0703	0.4139
5	1.45827904	0.24330564	0.0608	0.4747
6	1.21497340	0.11219751	0.0506	0.5253
7	1.10277589	0.04262495	0.0459	0.5712
8	1.06015095	0.04804942	0.0442	0.6154
9	1.01210153	0.05539783	0.0422	0.6576
10	0.95670370	0.00997626	0.0399	0.6975
11	0.94672744	0.04012138	0.0394	0.7369
12	0.90660606	0.04031167	0.0378	0.7747
13	0.86629440	0.06462874	0.0361	0.8108
14	0.80166566	0.02899843	0.0334	0.8442
15	0.77266723	0.04701661	0.0322	0.8764
16	0.72565061	0.07932499	0.0302	0.9066
17	0.64632562	0.10944974	0.0269	0.9335
18	0.53687588	0.07660641	0.0224	0.9559
19	0.46026947	0.10796612	0.0192	0.9751
20	0.35230336	0.20613295	0.0147	0.9898
21	0.14617041	0.06427376	0.0061	0.9958
22	0.08189665	0.06895551	0.0034	0.9993
23	0.01294113	0.00816825	0.0005	0.9998
24	0.00477288		0.0002	1.0000

Table 5 - Eigenvalues for each Principal Component Iteration

Looking specifically at the PCs generated for this final model we can see how convoluted things get. The Component Pattern Profile figure below (Figure 13) shows the relative correlation between all the variables. What a mess! It was extremely difficult to use this or the eigenvalue (or eigenvector) matrix for variable selection. In the end, we started with one PC and then ran cross validation. If the RMSLE went down, we kept it. This method gave us 6-8 different PCs then we went to the Kaggle website for the ultimate metric. The 4 PCs together ultimately contributed to the best Kaggle score.

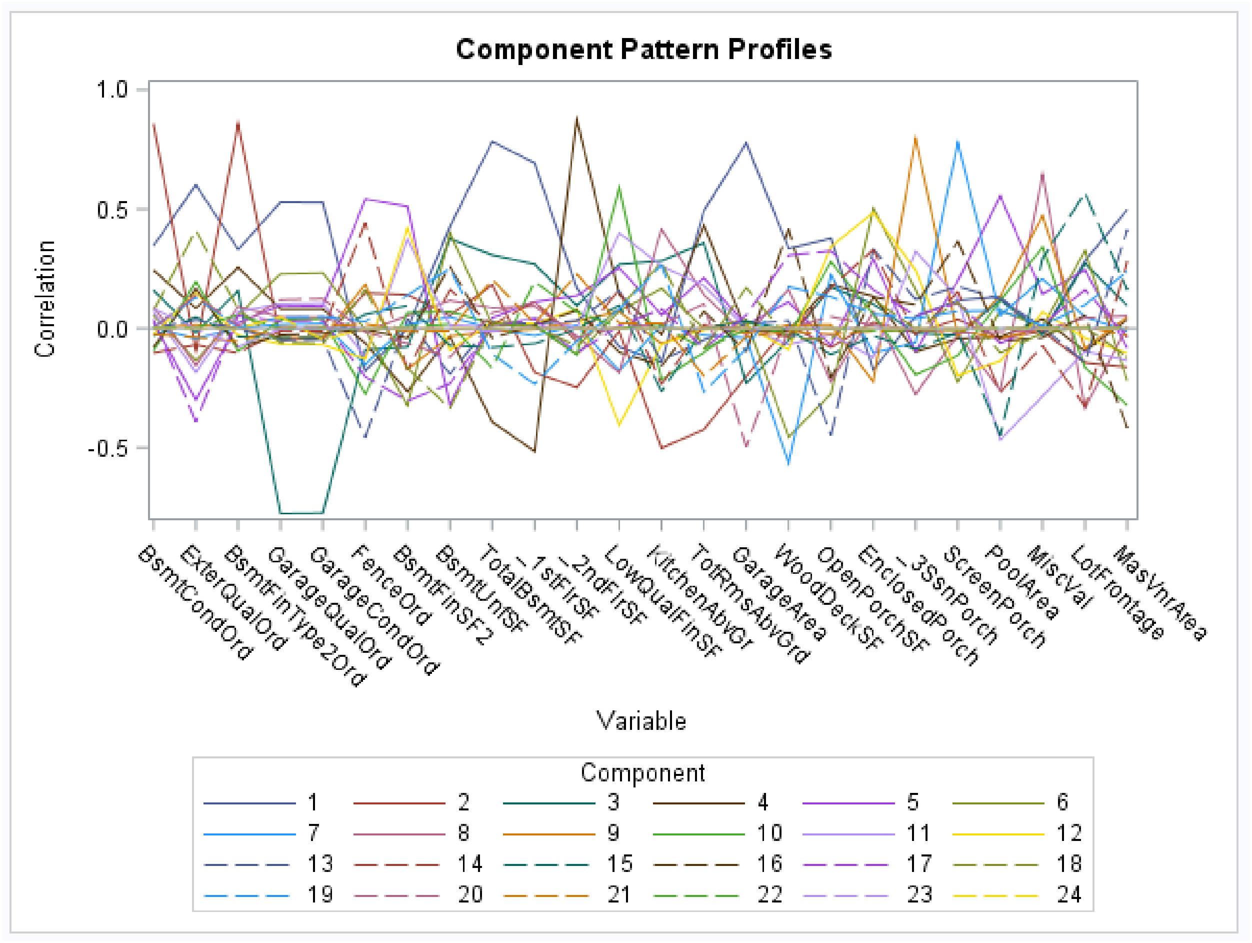


Figure 13 - This figure shows the component pattern profiles for the 24 PCs obtained creating Models 3.1-3.3.

Leaving all other variables unchanged, we can interpret the principle components in the following ways:

For the first principal component (`Prin1`), we can see in Figure 14 the different loading for each variable. With five negative coefficients `FenceOrd` (Fence whose levels are transformed as ordinal integers), `EnclosedPorch`, `KitchenAbvGr` (number of Kitchens above grade), etc. and the rest largely positive. The parameter estimate for `Prin1` is 0.039861 while the response is logSalePrice. This means that 1 unit of `Prin1` would result in ($e^{0.039861}$ -

1)*100 percent change in SalePrice. In this case that is a 4% increase in SalePrice for every 1 unit of Prin1. Further details are obtained using Figure 14 below which shows the loading of each variable in Prin1.

For the twelfth principal component (Prin12), we see a sizeable, negative value for LowQualFinSF; and we see sizeable, positive values for OpenPorchSF, BsmtFinSF2, and EnclosedPorch. Other variables, in between, have little significant difference. The parameter estimate for Prin12 is 0.013661 while the response is logSalePrice. This means that 1 unit of Prin12 would result in ($e^{0.013661}$ -1)*100 percent change in SalePrice. In this case that is a 1.4% increase in SalePrice for every 1 unit of Prin12.

Very similar to Prin12, Prin18 provides a sizeable, negative value for BsmtUnfSF; and we see sizeable, positive values for KitchenAbvGr and ExterQualOrd. Other variables, in between, have little significant difference. The parameter estimate for Prin18 is -0.00618 while the response is logSalePrice. This means that 1 unit of Prin18 would result in ($e^{-0.00618}$ -1)*100 percent change in SalePrice. In this case that is a 0.6% decrease in SalePrice for every 1 unit of Prin18.

Finally, for the Prin20, we see a very large, negative coefficient provided to GarageArea; while other parameter coefficients are not significantly different. This at first seems particularly odd in that we would interpret this to mean that, the larger the garage is, the lower the sale price of the house will be. This is even more interesting because GarageArea is one of the three variables (2ndFloorSF, GarageArea, ScreenPorch) also included in the model and not just in the PCs. The parameter estimate for Prin20 is -0.046861 while the response is logSalePrice. This means that 1 unit of Prin20 would result in ($e^{-0.046861}$ -1)*100 percent change in SalePrice. In this case that is a 4.6% decrease in SalePrice for every 1 unit of Prin20. The parameter estimate for GarageArea in the model however is -0.000175 which means that 1 unit of GarageArea corresponds to a 0.05% decrease in SalePrice which doesn't make a lot of sense. However, seeing as how highly loaded Prin20 is with GarageArea this could mean that GarageArea is ultimately positively correlated with SalePrice like we would expect.

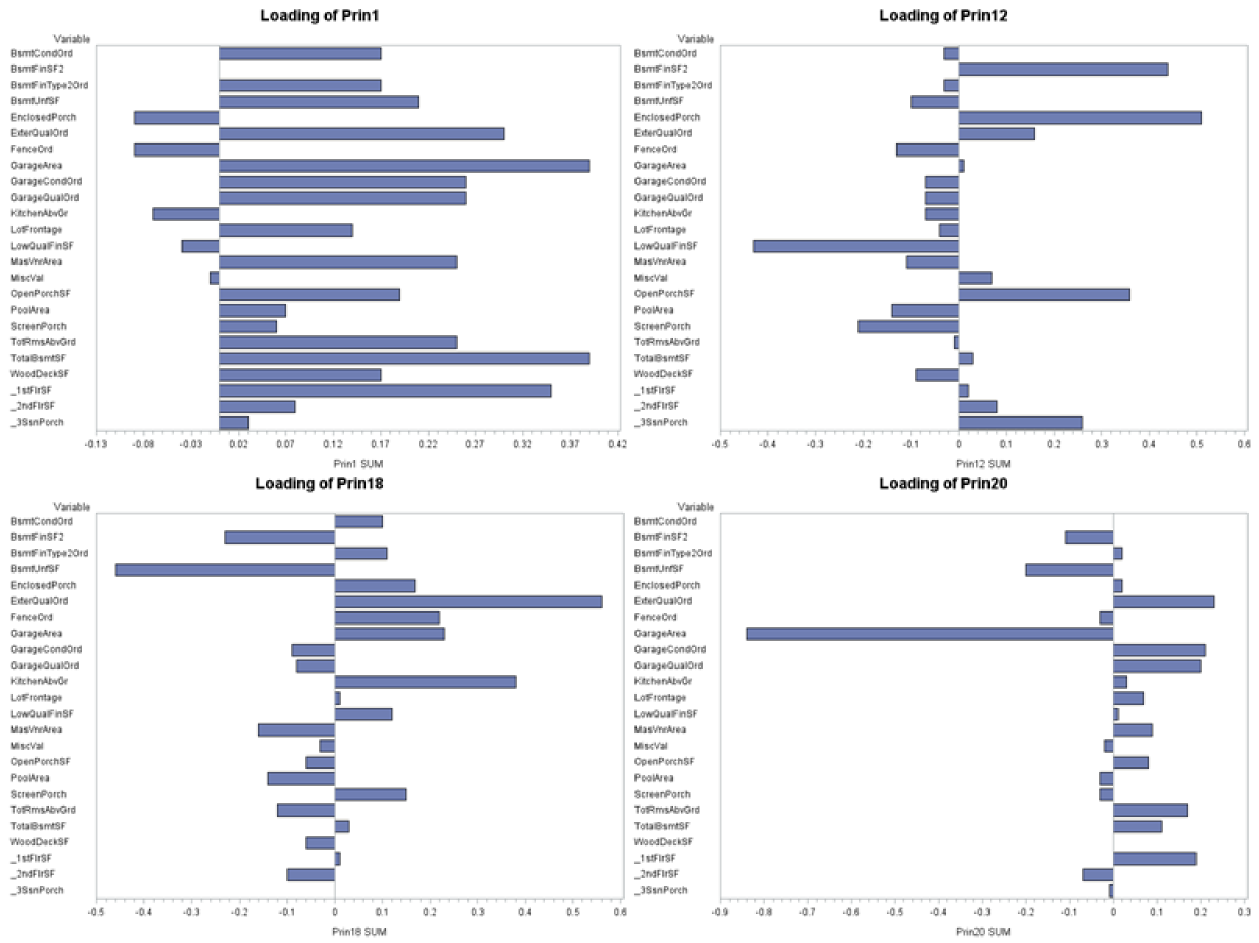


Figure 14 - This figure shows the variable loadings for the four PCs used in the final Model 3.3. It interesting to note the huge negative loading of GarageArea in Prin20. This is particularly interesting because GarageArea is one of the three variables (2ndFloorSF, GarageArea, ScreenPorch) that are also in the model by themselves.

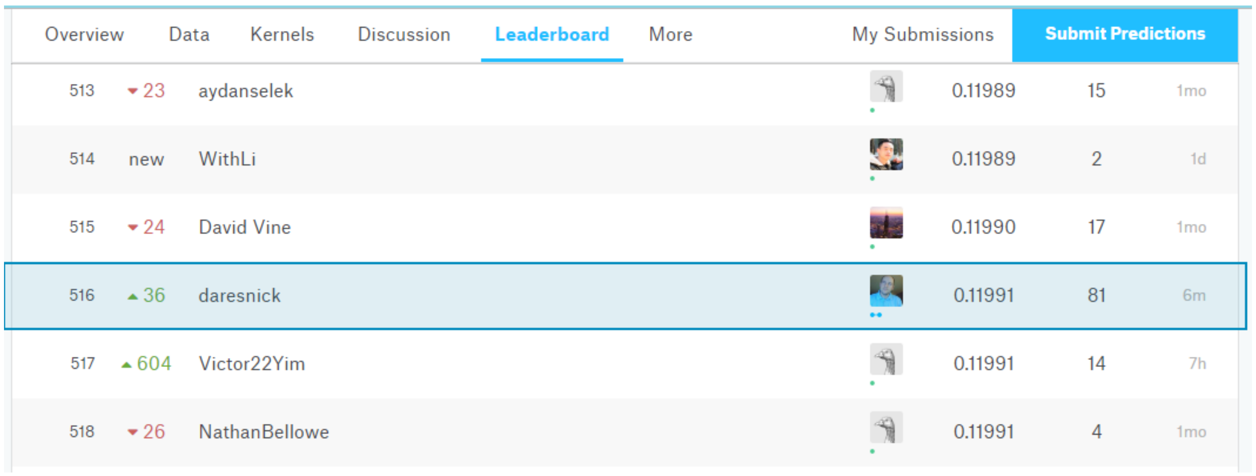
Conclusion/Discussion

In project 1, we built two models using a specific design and technique. For Model 1.1 in Question 1, our goal was parsimony and to create a model that made sense and could be easily interpreted and understood by a realtor or home buyer/seller. Instead of just using a variable selection technique, we looked for variables that were intuitive and that common knowledge told us should impact any given sale price of a home. However, we went one step further and decided to introduce an interaction term to enhance the predictability without adding much complexity. With an adjusted R^2 of 0.8873, it suggested that 89% of the sale price variation has been captured by the model. This demonstrates a strong correlation, and allows anyone to predict an individual house sale price with only 8 easy-to-obtain parameters, from the table in appendix I, and as shown in the example in the interpretation section of Question 1. The shortcoming however is the confidence interval. With 95% confidence, the range of the possible values is much larger than desired. This is largely driven by the range of home price in the city Ames, IA.

For question two, we explored many of the techniques that we have learned. Multiple selection techniques were used and compared. Interaction terms were tested based on assumptions. We tested out transformation techniques such as applying a logarithm to certain variables. We also performed grouping of variables to narrow the band, thus possibly reducing the influence of some outliers. At the very end of project 1, we settled on a model that gave us the best Kaggle score = 0.12611.

In project 2 we used even more tricks to reduce our Kaggle score. First, we tried ordinalizing some of the nominal variables. Basically, just assigning numbers to the ranking levels. Placing a few of these in the model improved the prediction metrics. Next, we tried a few more interaction terms. Specifically, we tried squared and cubed interactions of the ordinal and continuous variables. A few ordinal squared terms helped. We also found another interaction term between `log2ndFlrSF` and `logLotArea` that helped and together with the four PCs brought our final Kaggle score to 0.11991! Remember "If you can dream it. You can do it!"[3]

Kaggle submission proof below. I have also attached the csv file of the submission as well as a SAS script file to be run to generate the below score.



513	▼ 23	aydanselek	0.11989	15	1mo
514	new	WithLi	0.11989	2	1d
515	▼ 24	David Vine	0.11990	17	1mo
516	▲ 36	daresnick	0.11991	81	6m
517	▲ 604	Victor22Yim	0.11991	14	7h
518	▼ 26	NathanBellowe	0.11991	4	1mo

[redacted]

Addressing the Comments

We originally put that we wanted to "prove a serial correlation", but this is not possible. What we should have said is "show evidence of a serial correlation".

The subscripts for model parameters estimates have been adjusted to make it more clear that each parameter estimates is different.

Weird sentence "*Although much more evidence exists here, than above,* ***we it is*** *difficult to make a convincing argument for a first order autocorrelation*" should have been "*Although much more evidence exists here, than above,* ***it is*** *difficult to make a convincing argument for a first order autocorrelation*"

RStudent plots should be *Studentized Residual* plots. This was fixed.

A question arose as to whether or not "CV was used in Question 1". Indeed, CV was used. We used an internal cross validation technique via `proc glmselect`.

Labels have been added to all graphics as well as captions to some.

An ASE plot is now included for the model selection in Question 2 table.

RMSLE (our own) score is included in Question table.

More detailed interpretation was provided for the model in question 1. Very proud of this addition. https://www.kaggle.com/wiki/RootMeanSquaredLogarithmicError

References

1:https://www.kaggle.com/wiki/RootMeanSquaredLogarithmicError
2:https://www.britannica.com/technology/brick-building-material#toc76631
3:https://www.google.com/search?q=if+you+can+dream+it+you+can+do+it++blades+of+glory+pic&espv=2&tbm=isch&tbo=u&source=univ&sa=X&ved=0ahUKEwiQrqCjk_HSAhVU7GMKHQVmDjsQsAQIGQ&biw=1536&bih=760#imgrc=1HvaIJMKM46v3M:

Appendix specifically for Project 2 (Project 1 Appendices follows)

Project 2 Scripts: LDA and PCA scripts are provided. These scripts can only be run after the data has been loaded and cleaned. That script is also below.

LDA Script

```
proc discrim data=train pool=test crossvalidate
testdata=test testout=foundation noprint;
class Foundation;
var
      BedroomAbvGr BsmtFullBath BsmtHalfBath Fireplaces FullBath GarageYrBlt
      MSSubClass  OverallCond YearBuilt
      logYearBuilt BsmtQualOrd logBsmtFinSF1 KitchenAbvGr BsmtUnfSF
logMasVnrArea
      OverallQualGroup GrLivArea logGarageArea logGarageCars ;
 priors "BrkTil"=0.113091158 "CBlock"=0.411925977
"PConc"=0.453050034 "Slab"=0.017135024
"Stone" = 0.003427005 "Wood" = 0.001370802;
run;
```

PCA Script

```
/*
1. Use prinqual only on ordinal variables.
2. Use princomp on continous and ordinal variables transfromed by prinqual
not use in interaction terms.
3. Use glmselect to winnow the prins.
4. Use categorical variables and continous variables not in princomp in the
final model.
Summary:
1. Use prinqual on: YearBuilt YearRemodAdd YrSold FullBath GarageCars
FirePlaces BsmtFinType1Ord FunctionalOrd.
2. Use princomp on: variables in 1 above and log_2ndFlrSF OpenPorchSF
BsmtFinSF1 GarageArea logLotArea ScreenPorch.
3. Use glmselect to now winnow those prins.
4. Use glm and check with CV with above prins and BldgType BsmtExposure
BsmtQual Condition1Group KitchenQual SaleCondition NeighborhoodGroup RoofMatl
MSZoning
      Exterior1st CentralAir PoolQC OverallCond OverallQual FullBath and
logGrLivArea
*/

proc prinqual data=train out=prinq1a replace converge=1e-10 maxiter=200;
      transform
                  monotone(BsmtCondOrd ExterQualOrd BsmtFinType2Ord
                                    GarageQualOrd GarageCondOrd FenceOrd);
      run; quit;

*proc print data=prinq1a; run;

data prinq2b;
set train;
keep BsmtFinSF2 BsmtUnfSF EnclosedPorch GarageArea KitchenAbvGr
      LotFrontage LowQualFinSF MasVnrArea MiscVal OpenPorchSF
      PoolArea ScreenPorch TotRmsAbvGrd TotalBsmtSF WoodDeckSF
      _1stFlrSF _2ndFlrSF _3SsnPorch;
run;

data prinq1a;
set prinq1a;
set prinq2b;
run;

proc princomp data=prinq1a out=pca1a OUTSTAT= pcaoutstat plots=all ;
run; quit;
```

The following are the Appendices from Project 1 and 2.

Appendix I Kaggle Submission SAS code

Copy and paste this code directly into SAS. Make sure to change the two datafile= lines at the beginning of the code as well as the outfile= line at the very end of the code. Look for two proc import statements and at the end a proc export statement. We have also attached a .sas file that has this code. (Kagglescorerunfinal1a.sas) If you prefer, you can just run it in SAS after you have made the directory file changes.

```
/*
** Data files taken from www.kaggle.com/c/house-prices-advanced-regression-
techniques/data
** Running this will likely say that the imports have failed...
** ...Be sure you update the path with your own path...
** ...Otherwise, there will be some errors will SAS trying to turn "NA" into
a number
*/

/* Replace with your own file path */
proc import datafile='C:\Users\hp\Desktop\SMU\Exp Stats II\Homework and
projects\Project 1\test.csv' dbms=csv out=test replace;
delimiter = ",";
getnames=yes;
guessingrows=1460;
run;

*proc print data=test; run;

/* Replace with your own file path */
proc import datafile='C:\Users\hp\Desktop\SMU\Exp Stats II\Homework and
projects\Project 1\train.csv' dbms=csv out=train replace;
delimiter = ",";
getnames=yes;
guessingrows=1461;
run;

*proc print data=train; run;


/* Below is code to clean the data */

data test;
  set test;
  if MasVnrType = "NA" then MasVnrType = "None";
  if MasVnrArea = "NA" then MasVnrArea = 0;
  if MSZoning = "NA" then MSZoning = .;
  if Functional = "NA" then Functional = .;
  if BsmtHalfBath = "NA" then BsmtHalfBath = .;
  if BsmtFullBath = "NA" then BsmtFullBath = .;
  if Utilities = "NA" then Utilities = .;
  if SaleType = "NA" then SaleType = .;
  if GarageArea = "NA" then GarageArea = 0;
  if GarageCars = "NA" then GarageCars = 0;
  if TotalBsmtSF = "NA" then TotalBsmtSF = 0;
  if BsmtUnfSF = "NA" then BsmtUnfSF = 0;
  if BsmtFinSF2 = "NA" then BsmtFinSF2 = 0;
  if BsmtFinSF1 = "NA" then BsmtFinSF1 = 0;
  if Exterior2nd = "NA" then Exterior2nd = .;
  if Exterior1st = "NA" then Exterior1st = .;
  if LotFrontage = "NA" then LotFrontage = "0";
  if GarageYrBlt = "NA" then GarageYrBlt = "0";
  /* Fill missing data */
  if ID=1556 then KitchenQual = 'TA';
  /*if ID=1620 then TBD*/
  if ID=1692 then MasVnrType= 'None';
  if ID=1692 then MasVnrArea = 0;
  if ID=1707 then MasVnrType= 'None';
  if ID=1707 then MasVnrArea = 0;
  /*if ID=1829 then TBD*/
  /*if ID=1862 then TBD*/
  /*if ID=1863 then TBD*/
  /*if ID=1864 then TBD*/
  if ID=1883 then MasVnrType= 'None';
  if ID=1883 then MasVnrArea = 0;
  if ID=1916 then MSZoning='RL';
  if ID=1916 then Utilities='AllPub';
  if ID=1946 then Utilities='AllPub';
  if ID=1993 then MasVnrType= 'None';
  if ID=1993 then MasVnrArea = 0;
  if ID=2005 then MasVnrType= 'None';
  if ID=2005 then MasVnrArea = 0;
  if ID=2042 then MasVnrType= 'None';
  if ID=2042 then MasVnrArea = 0;
  if ID=2121 then BsmtFinSF2= 0;
  if ID=2121 then BsmtUnfSF = 0;
  if ID=2121 then TotalBsmtSF = 0;
  if ID=2121 then BsmtFinSF1 = 0;
  if ID=2121 then BsmtFullBath = 0;
  if ID=2121 then BsmtHalfBath = 0;
  if ID=2189 then BsmtFullBath = 0;
  if ID=2189 then BsmtHalfBath = 0;
  if ID=2217 then MSZoning='RL';
  if ID=2217 then Functional ='Typ';
  if ID=2251 then MSZoning='RL';
  if ID=2312 then MasVnrType= 'None';
  if ID=2312 then MasVnrArea = 0;
  if ID=2326 then MasVnrType= 'None';
  if ID=2326 then MasVnrArea = 0;
  if ID=2341 then MasVnrType= 'None';
  if ID=2341 then MasVnrArea = 0;
  if ID=2350 then MasVnrType= 'None';
  if ID=2350 then MasVnrArea = 0;
  if ID=2369 then MasVnrType= 'None';
  if ID=2369 then MasVnrArea = 0;
  if ID=2474 then Functional ='Typ';
```

```
  if ID=2577 then GarageCars =0;
  if ID=2577 then GarageArea =0;
  if ID=2593 then MasVnrType= 'None';
  if ID=2593 then MasVnrArea = 0;
  if ID=2611 then MasVnrType= 'BrkFace';
  if ID=2658 then MasVnrType= 'None';
  if ID=2658 then MasVnrArea = 0;
  if ID=2687 then MasVnrType= 'None';
  if ID=2687 then MasVnrArea = 0;
  /*If ID=2711 then TBD*/
  if ID=2863 then MasVnrType= 'None';
  if ID=2863 then MasVnrArea = 0;
  if ID=2905 then MSZoning='RL';
run;

data train;
  set train;
  if MasVnrType = "NA" then MasVnrType = ".";
  if MasVnrArea = "NA" then MasVnrArea = ".";
  if Electrical = "NA" then Electrical = ".";
  if LotFrontage = "NA" then LotFrontage = "0";
  if GarageYrBlt = "NA" then GarageYrBlt = "0";

  /* Fill missing data */
  if ID=1299 then GrLivArea = 1426.9;
  if ID=1299 then LotArea= 9833.2;
  if ID=1183 then LotArea= 23404.7;
  if ID=524 then GrLivArea = 1574.536;
  if ID=524 then LotArea = 10107.7;
  if ID=1424 then MSSubClass= '60';
  if ID=1424 then OverallCond = '5';
  if ID=1424 then OverallQual = '8';
  if ID=692 then OverallCond = '5';
run;

/***** Turn all 'NA' values into the SAS-Safe '.' *****/

data test;
  set test;
  array change _character_;
  do over change;
  if change='NA' then change='NT';
  end;
run;


data train;
  set train;
  array change _character_;
  do over change;
  if change='NA' then change='NT';
  end;
run;


/******** cast columns to numbers (that were lost in proc import) ********/
data test;
```

```
   set test;

   LotFrontage1 = input(LotFrontage, 8.);
       attrib LotFrontage1  format= BEST12. informat=BEST32.;
   MasVnrArea1 = input(MasVnrArea, 8.);
       attrib MasVnrArea1  format= BEST12. informat=BEST32.;
   GarageYrBlt1 = input(GarageYrBlt, 8.);
       attrib GarageYrBlt1  format= BEST12. informat=BEST32.;
   BsmtFinSF11 = input(BsmtFinSF1, 8.);
       attrib BsmtFinSF11  format= BEST12. informat=BEST32.;
   BsmtFinSF21 = input(BsmtFinSF2, 8.);
     attrib BsmtFinSF21 format= BEST12. informat=BEST32.;
   BsmtUnfSF1 = input(BsmtUnfSF, 8.);
     attrib BsmtUnfSF1 format= BEST12. informat=BEST32.;
   TotalBsmtSF1 = input(TotalBsmtSF, 8.);
     attrib TotalBsmtSF1 format= BEST12. informat=BEST32.;
   BsmtFullBath1 = input(BsmtFullBath, 8.);
     attrib BsmtFullBath1 format= BEST12. informat=BEST32.;
   BsmtHalfBath1 = input(BsmtHalfBath, 8.);
     attrib BsmtHalfBath1 format= BEST12. informat=BEST32.;
   GarageCars1 = input(GarageCars, 8.);
     attrib GarageCars1 format= BEST12. informat=BEST32.;
   GarageArea1 = input(GarageArea, 8.);
     attrib GarageArea1 format= BEST12. informat=BEST32.;

   drop BsmtFinSF1;
   drop BsmtFinSF2;
   drop BsmtUnfSF;
   drop TotalBsmtSF;
   drop BsmtFullBath;
   drop BsmtHalfBath;
   drop GarageCars;
   drop GarageArea;
   drop LotFrontage;
   drop MasVnrArea;
   drop GarageYrBlt;

   rename BsmtFinSF11 = BsmtFinSF1;
   rename BsmtFinSF21 = BsmtFinSF2;
   rename BsmtUnfSF1 = BsmtUnfSF;
   rename TotalBsmtSF1 = TotalBsmtSF;
   rename BsmtFullBath1 = BsmtFullBath;
   rename BsmtHalfBath1 = BsmtHalfBath;
   rename GarageCars1 = GarageCars;
   rename GarageArea1 = GarageArea;
   rename LotFrontage1= LotFrontage;
   rename MasVnrArea1 = MasVnrArea;
   rename GarageYrBlt1 = GarageYrBlt;
run;


data train;
   set train;

   LotFrontage1 = input(LotFrontage, 8.);
        attrib LotFrontage1  format= BEST12. informat=BEST32.;
   MasVnrArea1 = input(MasVnrArea, 8.);
```

```
      attrib MasVnrArea1  format= BEST12. informat=BEST32.;
   GarageYrBlt1 = input(GarageYrBlt, 8.);
      attrib GarageYrBlt1  format= BEST12. informat=BEST32.;

   drop LotFrontage;
   drop MasVnrArea;
   drop GarageYrBlt;

   rename LotFrontage1= LotFrontage;
   rename MasVnrArea1 = MasVnrArea;
   rename GarageYrBlt1 = GarageYrBlt;
run;


/* I created 3 new categorical variables based on existig values.   */

Data Train;
Set train;
TotalBath = BsmtFullBath+BsmtHalfBath+FullBath;
if YearRemodAdd<2007 then YearRemodAddGROUP = "OLD";
if YearRemodAdd>=2007 then YearRemodAddGROUP = "NEW";
if GrLivArea<1000 then GrLivAreaGroup = "Small1";
If GrLivArea>=1000 and GrLivArea< 1500 then GrLivAreaGroup = "Small2";
If GrLivArea>=1500 and GrLivArea< 2000 then GrLivAreaGroup = "Med1";
If GrLivArea>=2000 and GrLivArea< 2500 then GrLivAreaGroup = "Med2";
If GrLivArea>=2500 and GrLivArea< 3000 then GrLivAreaGroup = "Large1";
If GrLivArea>=3000then GrLivAreaGroup = "Large2";
If Neighborhood ="Blmngtn" then NeighborhoodGroup =  "G01";
If Neighborhood ="Blueste" then NeighborhoodGroup =  "G02";
If Neighborhood ="BrDale" then NeighborhoodGroup =  "G03";
If Neighborhood ="BrkSide" then NeighborhoodGroup =  "G04";
If Neighborhood ="ClearCr" then NeighborhoodGroup =  "G01";
If Neighborhood ="CollgCr" then NeighborhoodGroup =  "G01";
If Neighborhood ="Crawfor" then NeighborhoodGroup =  "G01";
If Neighborhood ="Edwards" then NeighborhoodGroup =  "G02";
If Neighborhood ="Gilbert" then NeighborhoodGroup =  "G05";
If Neighborhood ="IDOTRR" then NeighborhoodGroup =  "G03";
If Neighborhood ="MeadowV" then NeighborhoodGroup =  "G03";
If Neighborhood ="Mitchel" then NeighborhoodGroup =  "G06";
If Neighborhood ="NAmes" then NeighborhoodGroup =  "G06";
If Neighborhood ="NPkVill" then NeighborhoodGroup =  "G02";
If Neighborhood ="NWAmes" then NeighborhoodGroup =  "G07";
If Neighborhood ="NoRidge" then NeighborhoodGroup =  "G12";
If Neighborhood ="NridgHt" then NeighborhoodGroup =  "G08";
If Neighborhood ="OldTown" then NeighborhoodGroup =  "G09";
If Neighborhood ="SWISU" then NeighborhoodGroup =  "G02";
If Neighborhood ="Sawyer" then NeighborhoodGroup =  "G02";
If Neighborhood ="SawyerW" then NeighborhoodGroup =  "G10";
If Neighborhood ="Somerst" then NeighborhoodGroup =  "G11";
If Neighborhood ="StoneBr" then NeighborhoodGroup =  "G08";
If Neighborhood ="Timber" then NeighborhoodGroup =  "G11";
If Neighborhood ="Veenker" then NeighborhoodGroup =  "G11";
run;

Data test;
Set test;
TotalBath = BsmtFullBath+BsmtHalfBath+FullBath;
```

[redacted]

```
if YearRemodAdd<2007 then YearRemodAddGROUP = "OLD";
if YearRemodAdd>=2007 then YearRemodAddGROUP = "NEW";
if GrLivArea<1000 then GrLivAreaGroup = "Small1";
If GrLivArea>=1000 and GrLivArea< 1500 then GrLivAreaGroup = "Small2";
If GrLivArea>=1500 and GrLivArea< 2000 then GrLivAreaGroup = "Med1";
If GrLivArea>=2000 and GrLivArea< 2500 then GrLivAreaGroup = "Med2";
If GrLivArea>=2500 and GrLivArea< 3000 then GrLivAreaGroup = "Large1";
If GrLivArea>=3000then GrLivAreaGroup = "Large2";
If Neighborhood ="Blmngtn" then NeighborhoodGroup =  "G01";
If Neighborhood ="Blueste" then NeighborhoodGroup =  "G02";
If Neighborhood ="BrDale" then NeighborhoodGroup =  "G03";
If Neighborhood ="BrkSide" then NeighborhoodGroup =  "G04";
If Neighborhood ="ClearCr" then NeighborhoodGroup =  "G01";
If Neighborhood ="CollgCr" then NeighborhoodGroup =  "G01";
If Neighborhood ="Crawfor" then NeighborhoodGroup =  "G01";
If Neighborhood ="Edwards" then NeighborhoodGroup =  "G02";
If Neighborhood ="Gilbert" then NeighborhoodGroup =  "G05";
If Neighborhood ="IDOTRR" then NeighborhoodGroup =  "G03";
If Neighborhood ="MeadowV" then NeighborhoodGroup =  "G03";
If Neighborhood ="Mitchel" then NeighborhoodGroup =  "G06";
If Neighborhood ="NAmes" then NeighborhoodGroup =  "G06";
If Neighborhood ="NPkVill" then NeighborhoodGroup =  "G02";
If Neighborhood ="NWAmes" then NeighborhoodGroup =  "G07";
If Neighborhood ="NoRidge" then NeighborhoodGroup =  "G12";
If Neighborhood ="NridgHt" then NeighborhoodGroup =  "G08";
If Neighborhood ="OldTown" then NeighborhoodGroup =  "G09";
If Neighborhood ="SWISU" then NeighborhoodGroup =  "G02";
If Neighborhood ="Sawyer" then NeighborhoodGroup =  "G02";
If Neighborhood ="SawyerW" then NeighborhoodGroup =  "G10";
If Neighborhood ="Somerst" then NeighborhoodGroup =  "G11";
If Neighborhood ="StoneBr" then NeighborhoodGroup =  "G08";
If Neighborhood ="Timber" then NeighborhoodGroup =  "G11";
If Neighborhood ="Veenker" then NeighborhoodGroup =  "G11";
run;

/***** Include boolean values *****/
data train;
set train;
if EnclosedPorch > 0 or ScreenPorch > 0 or OpenPorchSF > 0 or _3SsnPorch > 0
      then porch = 1;
      else porch = 0;
run;

data test;
set test;
if EnclosedPorch > 0 or ScreenPorch > 0 or OpenPorchSF > 0 or _3SsnPorch > 0
      then porch = 1;
      else porch = 0;
/* Blueste neighborhood does not a house with a porch in train data */
if Neighborhood = 'Blueste' then porch = 0;
run;


data train;
set train;
if OverallQual = 1 then OverallQualGroup = 12;
if OverallQual = 2 then OverallQualGroup = 12;
if OverallQual = 3 then OverallQualGroup = 3;
if OverallQual = 4 then OverallQualGroup = 4;
if OverallQual = 5 then OverallQualGroup = 5;
if OverallQual = 6 then OverallQualGroup = 6;
if OverallQual = 7 then OverallQualGroup = 7;
if OverallQual = 8 then OverallQualGroup = 8;
if OverallQual = 9 then OverallQualGroup = 9;
if OverallQual = 10 then OverallQualGroup = 10;
run;


data test;
set test;
if OverallQual = 1 then OverallQualGroup = 12;
if OverallQual = 2 then OverallQualGroup = 12;
if OverallQual = 3 then OverallQualGroup = 3;
if OverallQual = 4 then OverallQualGroup = 4;
if OverallQual = 5 then OverallQualGroup = 5;
if OverallQual = 6 then OverallQualGroup = 6;
if OverallQual = 7 then OverallQualGroup = 7;
if OverallQual = 8 then OverallQualGroup = 8;
if OverallQual = 9 then OverallQualGroup = 9;
if OverallQual = 10 then OverallQualGroup = 10;
run;


data train;
set train;
if OverallCond = 1 then OverallCondGroup = 12;
if OverallCond = 2 then OverallCondGroup = 12;
if OverallCond = 3 then OverallCondGroup = 34;
if OverallCond = 4 then OverallCondGroup = 34;
if OverallCond = 5 then OverallCondGroup = 5;
if OverallCond = 6 then OverallCondGroup = 6;
if OverallCond = 7 then OverallCondGroup = 78;
if OverallCond = 8 then OverallCondGroup = 78;
if OverallCond = 9 then OverallCondGroup = 9;
if OverallCond = 10 then OverallCondGroup = 9;
run;

data test;
set test;
if OverallCond = 1 then OverallCondGroup = 12;
if OverallCond = 2 then OverallCondGroup = 12;
if OverallCond = 3 then OverallCondGroup = 34;
if OverallCond = 4 then OverallCondGroup = 34;
if OverallCond = 5 then OverallCondGroup = 5;
if OverallCond = 6 then OverallCondGroup = 6;
if OverallCond = 7 then OverallCondGroup = 78;
if OverallCond = 8 then OverallCondGroup = 78;
if OverallCond = 9 then OverallCondGroup = 9;
if OverallCond = 10 then OverallCondGroup = 9;
run;


/* group condition1 values by similar mean|observations|variation */
data train;
```

```
set train;
if Condition1 = 'Artery' then Condition1Group = 'Small';
if Condition1 = 'Feedr' then Condition1Group = 'Small';
if Condition1 = 'RRAe' then Condition1Group = 'Small';

if Condition1 = 'PosA' then Condition1Group = 'Large';
if Condition1 = 'PosN' then Condition1Group = 'Large';
if Condition1 = 'RRNe' then Condition1Group = 'Large';
if Condition1 = 'RRAn' then Condition1Group = 'Large';
if Condition1 = 'RRNn' then Condition1Group = 'Large';

if Condition1 = 'Norm' then Condition1Group = 'Norm';
run;

data test;
set test;
if Condition1 = 'Artery' then Condition1Group = 'Small';
if Condition1 = 'Feedr' then Condition1Group = 'Small';
if Condition1 = 'RRAe' then Condition1Group = 'Small';

if Condition1 = 'PosA' then Condition1Group = 'Large';
if Condition1 = 'PosN' then Condition1Group = 'Large';
if Condition1 = 'RRNe' then Condition1Group = 'Large';
if Condition1 = 'RRAn' then Condition1Group = 'Large';
if Condition1 = 'RRNn' then Condition1Group = 'Large';

if Condition1 = 'Norm' then Condition1Group = 'Norm';
run;


data train;
set train;
If BldgType in ("1Fam","TwnhsI") then BldgTypeGroup="Single";
If BldgType in ("2FmCon","Duplx","TwnhsE","Twnhs","Duplex","2fmCon") then
BldgTypeGroup="Multi";
run;

data test;
set test;
If BldgType in ("1Fam","TwnhsI") then BldgTypeGroup="Single";
If BldgType in ("2FmCon","Duplx","TwnhsE","Twnhs","Duplex","2fmCon") then
BldgTypeGroup="Multi";
run;


data train;
set train;
If GarageType in ("BuiltIn","Attchd") then GarageTypeGroup = "Good";
else GarageTypeGroup = "Bad";
Run;

data test;
set test;
If GarageType in ("BuiltIn","Attchd") then GarageTypeGroup = "Good";
else GarageTypeGroup = "Bad";
Run;
```

```
data train;
set train;
if BsmtExposure = "Av" then BsmtExposureGroup = "Av";
if BsmtExposure = "Gd" then BsmtExposureGroup = "Gd";
if BsmtExposure = "Mn" then BsmtExposureGroup = "Gd";
if BsmtExposure = "NT" then BsmtExposureGroup = "Av";
if BsmtExposure = "No" then BsmtExposureGroup = "Av";
run;

data test;
set test;
if BsmtExposure = "Av" then BsmtExposureGroup = "Av";
if BsmtExposure = "Gd" then BsmtExposureGroup = "Gd";
if BsmtExposure = "Mn" then BsmtExposureGroup = "Gd";
if BsmtExposure = "NT" then BsmtExposureGroup = "Av";
if BsmtExposure = "No" then BsmtExposureGroup = "Av";
run;


data train;
set train;
if BsmtQual = "Ex" then BsmtQualGroup = "Ex";
if BsmtQual = "Fa" then BsmtQualGroup = "FN";
if BsmtQual = "Gd" then BsmtQualGroup = "Gd";
if BsmtQual = "NT" then BsmtQualGroup = "FN";
if BsmtQual = "TA" then BsmtQualGroup = "TA";
run;

data test;
set test;
if BsmtQual = "Ex" then BsmtQualGroup = "Ex";
if BsmtQual = "Fa" then BsmtQualGroup = "FN";
if BsmtQual = "Gd" then BsmtQualGroup = "Gd";
if BsmtQual = "NT" then BsmtQualGroup = "FN";
if BsmtQual = "TA" then BsmtQualGroup = "TA";
run;


data train;
set train;
if KitchenQual = "Ex" then KitchenQualGroup = "Ex";
if KitchenQual = "Fa" then KitchenQualGroup = "Gd";
if KitchenQual = "Gd" then KitchenQualGroup = "Gd";
if KitchenQual = "TA" then KitchenQualGroup = "TA";
run;

data test;
set test;
if KitchenQual = "Ex" then KitchenQualGroup = "Ex";
if KitchenQual = "Fa" then KitchenQualGroup = "Gd";
if KitchenQual = "Gd" then KitchenQualGroup = "Gd";
if KitchenQual = "TA" then KitchenQualGroup = "TA";
run;
```

```
data train;
set train;
if RoofMatl = "ClyTile" then RoofMatlGroup = "ClyTile";
if RoofMatl = "CompShg" then RoofMatlGroup = "CompShg";
if RoofMatl = "Membran" then RoofMatlGroup = "WdShake";
if RoofMatl = "Metal" then RoofMatlGroup = "Metal";
if RoofMatl = "Roll" then RoofMatlGroup = "Roll";
if RoofMatl = 'Tar&Grv' then RoofMatlGroup = 'Tar&Grv';
if RoofMatl = "WdShake" then RoofMatlGroup = "WdShake";
if RoofMatl = "WdShngl" then RoofMatlGroup = "WdShngl";
run;

data test;
set test;
if RoofMatl = "ClyTile" then RoofMatlGroup = "ClyTile";
if RoofMatl = "CompShg" then RoofMatlGroup = "CompShg";
if RoofMatl = "Membran" then RoofMatlGroup = "WdShake";
if RoofMatl = "Metal" then RoofMatlGroup = "Metal";
if RoofMatl = "Roll" then RoofMatlGroup = "Roll";
if RoofMatl = 'Tar&Grv' then RoofMatlGroup = 'Tar&Grv';
if RoofMatl = "WdShake" then RoofMatlGroup = "WdShake";
if RoofMatl = "WdShngl" then RoofMatlGroup = "WdShngl";
run;


data train;
set train;
if SaleCondition = "Abnorml" then SaleConditionGroup = "Abnorml";
if SaleCondition = "AdjLand" then SaleConditionGroup = "AdjLand";
if SaleCondition = "Alloca" then SaleConditionGroup = "Family";
if SaleCondition = "Family" then SaleConditionGroup = "Family";
if SaleCondition = "Normal" then SaleConditionGroup = "Normal";
if SaleCondition = "Partial" then SaleConditionGroup = "Partial";
run;

data test;
set test;
if SaleCondition = "Abnorml" then SaleConditionGroup = "Abnorml";
if SaleCondition = "AdjLand" then SaleConditionGroup = "AdjLand";
if SaleCondition = "Alloca" then SaleConditionGroup = "Family";
if SaleCondition = "Family" then SaleConditionGroup = "Family";
if SaleCondition = "Normal" then SaleConditionGroup = "Normal";
if SaleCondition = "Partial" then SaleConditionGroup = "Partial";
run;


data train;
set train;
if ExterCond = "Ex" then ExterCondGroup = "Gd";
if ExterCond = "Fa" then ExterCondGroup = "Fa";
if ExterCond = "Gd" then ExterCondGroup = "Gd";
if ExterCond = "Po" then ExterCondGroup = "Po";
if ExterCond = "TA" then ExterCondGroup = "TA";
run;

data test;
set test;
```

```
if ExterCond = "Ex" then ExterCondGroup = "Gd";
if ExterCond = "Fa" then ExterCondGroup = "Fa";
if ExterCond = "Gd" then ExterCondGroup = "Gd";
if ExterCond = "Po" then ExterCondGroup = "Po";
if ExterCond = "TA" then ExterCondGroup = "TA";
run;


data train;
set train;
if Functional = "Maj1" then FunctionalGroup = "Mod";
if Functional = "Maj2" then FunctionalGroup = "Maj2";
if Functional = "Min1" then FunctionalGroup = "Min1";
if Functional = "Min2" then FunctionalGroup = "Min1";
if Functional = "Mod" then FunctionalGroup = "Mod";
if Functional = "Sev" then FunctionalGroup = "Sev";
if Functional = "Typ" then FunctionalGroup = "Typ";
run;

data test;
set test;
if Functional = "Maj1" then FunctionalGroup = "Mod";
if Functional = "Maj2" then FunctionalGroup = "Maj2";
if Functional = "Min1" then FunctionalGroup = "Min1";
if Functional = "Min2" then FunctionalGroup = "Min1";
if Functional = "Mod" then FunctionalGroup = "Mod";
if Functional = "Sev" then FunctionalGroup = "Sev";
if Functional = "Typ" then FunctionalGroup = "Typ";
run;


data train;
Set train;
If ID=301 then MasVnrType = "Brkface";
If ID=1335 then MasVnrType = "Brkface";
If ID=625 then MasVnrType = "Brkface";
If ID=1301 then MasVnrType = "Brkface";
If ID=1670 then MasVnrType = "Brkface";
If MasVnrArea = 1 then  MasVnrArea = 0;
If MasVnrArea = 0 then MasVnrType ="None";
*if MasVnrType = "."  then MasVnrArea = 0;
if MasVnrType = "." then MasVnrType = "None";
if Electrical = "." then Electrical =  "SBrkr";
*If ID=524 then _2ndFlrSF = 769;
*If ID=1299 then _2ndFlrSF = 475;
*if Id=530 then LotArea = 11650;
Run;

data test;
Set test;
If ID=301 then MasVnrType = "Brkface";
If ID=1335 then MasVnrType = "Brkface";
If ID=625 then MasVnrType = "Brkface";
If ID=1301 then MasVnrType = "Brkface";
If ID=1670 then MasVnrType = "Brkface";
If MasVnrArea = 1 then  MasVnrArea = 0;
If MasVnrArea = 0 then MasVnrType ="None";
```

```
*if MasVnrType = "."  then MasVnrArea = 0;
if MasVnrType = "." then MasVnrType = "None";
if Electrical = "." then Electrical = "SBrkr";
*If ID=524 then _2ndFlrSF = 769;
*If ID=1299 then _2ndFlrSF = 475;
*if Id=530 then LotArea = 11650;
Run;

data train;
Set train;
logSalePrice = log(SalePrice + 1);
log_1stFlrSF = log(_1stFlrSF + 1);
log_2ndFlrSF = log(_2ndFlrSF + 1);
logMasVnrArea = log(MasVnrArea + 1);
log_3SsnPorch = log(_3SsnPorch + 1);
logBsmtFinSF1 = log(BsmtFinSF1 + 1);
logGarageArea = log(GarageArea + 1);
logLotArea = log(LotArea + 1);
logScreenPorch = log(ScreenPorch + 1);
logGrLivArea = log(GrLivArea + 1);
logYearBuilt = log(YearBuilt + 1);
logYearRemodAdd = log(YearRemodAdd + 1);
logYrSold = log(YrSold + 1);
logFullBath = log(FullBath + 1);
logGarageCars = log(GarageCars + 1);
logFirePlaces = log(FirePlaces + 1);
logOpenPorchSF = log(OpenPorchSF + 1);
run;

data test;
Set test;
log_1stFlrSF = log(_1stFlrSF + 1);
log_2ndFlrSF = log(_2ndFlrSF + 1);
logMasVnrArea = log(MasVnrArea + 1);
log_3SsnPorch = log(_3SsnPorch + 1);
logBsmtFinSF1 = log(BsmtFinSF1 + 1);
logGarageArea = log(GarageArea + 1);
logLotArea = log(LotArea + 1);
logScreenPorch = log(ScreenPorch + 1);
logGrLivArea = log(GrLivArea + 1);
logYearBuilt = log(YearBuilt + 1);
logYearRemodAdd = log(YearRemodAdd + 1);
logYrSold = log(YrSold + 1);
logFullBath = log(FullBath + 1);
logGarageCars = log(GarageCars + 1);
logFirePlaces = log(FirePlaces + 1);
logOpenPorchSF = log(OpenPorchSF + 1);
run;


data train;
Set train;
If ID=89 then delete;
If ID=813 then delete;
If ID=524 then delete;
If ID=826 then delete;
If ID=589 then delete;
If ID=1424 then delete;
If ID=969 then delete;
If ID=186 then delete;
If ID=633 then delete;
If ID=325 then delete;
If ID=534 then delete;
If ID=496 then delete;
If ID=31 then delete;
If ID=1025 then delete;
If ID=667 then delete;
If ID=917 then delete;
If ID=1001 then delete;
If ID=711 then delete;
If ID=1325 then delete;
If ID=590 then delete;
run;



data test;
Set test;
If ID=1706 then Condition2 = "Norm";
If ID=1947 then Condition2 = "Norm";
*If ID=2111 then Condition2 = "Norm";
*If ID=2239 then Condition2 = "Norm";
*If ID=2456 then Condition2 = "Norm";
If ID=1832 then Functional = "Typ";
If ID=2152 then Exterior1st = "BrkFace";
run;


data train;
Set train;
If ID=278 then delete;
If ID=706 then delete;
If ID=411 then delete;
If ID=739 then delete;
If ID=689 then delete;
If ID=1063 then delete;
If ID=1381 then delete;
If ID=975 then delete;
If ID=480 then delete;
If ID=379 then delete;
If ID=692 then delete;
If ID=1062 then delete;
If ID=609 then delete;
If ID=1045 then delete;
*If ID=1187 then delete;
*If ID=10 then delete;
*If ID=630 then delete;
*If ID=54 then delete;
*If ID=629 then delete;
run;

data train;
Set train;
if Exterior1st = "BrkComm" then Exterior1st = "BrkFace";
```

```
run;

data test;
Set test;
if Exterior1st = "BrkComm" then Exterior1st = "BrkFace";
run;


data train;
set train;
if BsmtQual = "Ex" then BsmtQualOrd = 5;
if BsmtQual = "Gd" then BsmtQualOrd = 4;
if BsmtQual = "TA" then BsmtQualOrd = 3;
if BsmtQual = "Fa" then BsmtQualOrd = 2;
if BsmtQual = "Po" then BsmtQualOrd = 1;
if BsmtQual = "NT" then BsmtQualOrd = 0;
run;

data test;
set test;
if BsmtQual = "Ex" then BsmtQualOrd = 5;
if BsmtQual = "Gd" then BsmtQualOrd = 4;
if BsmtQual = "TA" then BsmtQualOrd = 3;
if BsmtQual = "Fa" then BsmtQualOrd = 2;
if BsmtQual = "Po" then BsmtQualOrd = 1;
if BsmtQual = "NT" then BsmtQualOrd = 0;
run;


data train;
set train;
if BsmtCond = "Ex" then BsmtCondOrd = 5;
if BsmtCond = "Gd" then BsmtCondOrd = 4;
if BsmtCond = "TA" then BsmtCondOrd = 3;
if BsmtCond = "Fa" then BsmtCondOrd = 2;
if BsmtCond = "Po" then BsmtCondOrd = 1;
if BsmtCond = "NT" then BsmtCondOrd = 0;
run;

data test;
set test;
if BsmtCond = "Ex" then BsmtCondOrd = 5;
if BsmtCond = "Gd" then BsmtCondOrd = 4;
if BsmtCond = "TA" then BsmtCondOrd = 3;
if BsmtCond = "Fa" then BsmtCondOrd = 2;
if BsmtCond = "Po" then BsmtCondOrd = 1;
if BsmtCond = "NT" then BsmtCondOrd = 0;
run;


data train;
set train;
if BsmtExposure = "Gd" then BsmtExposureOrd = 4;
if BsmtExposure = "Av" then BsmtExposureOrd = 3;
if BsmtExposure = "Mn" then BsmtExposureOrd = 2;
if BsmtExposure = "No" then BsmtExposureOrd = 1;
if BsmtExposure = "NT" then BsmtExposureOrd = 0;
```

```
run;

data test;
set test;
if BsmtExposure = "Gd" then BsmtExposureOrd = 4;
if BsmtExposure = "Av" then BsmtExposureOrd = 3;
if BsmtExposure = "Mn" then BsmtExposureOrd = 2;
if BsmtExposure = "No" then BsmtExposureOrd = 1;
if BsmtExposure = "NT" then BsmtExposureOrd = 0;
run;


data train;
set train;
if ExterQual = "Ex" then ExterQualOrd = 5;
if ExterQual = "Gd" then ExterQualOrd = 4;
if ExterQual = "TA" then ExterQualOrd = 3;
if ExterQual = "Fa" then ExterQualOrd = 2;
if ExterQual = "Po" then ExterQualOrd = 1;
if ExterQual = "NT" then ExterQualOrd = 0;
run;

data test;
set test;
if ExterQual = "Ex" then ExterQualOrd = 5;
if ExterQual = "Gd" then ExterQualOrd = 4;
if ExterQual = "TA" then ExterQualOrd = 3;
if ExterQual = "Fa" then ExterQualOrd = 2;
if ExterQual = "Po" then ExterQualOrd = 1;
if ExterQual = "NT" then ExterQualOrd = 0;
run;


data train;
set train;
if BsmtFinType1 = "GLQ" then BsmtFinType1Ord = 7;
if BsmtFinType1 = "ALQ" then BsmtFinType1Ord = 6;
if BsmtFinType1 = "BLQ" then BsmtFinType1Ord = 5;
if BsmtFinType1 = "Rec" then BsmtFinType1Ord = 4;
if BsmtFinType1 = "LwQ" then BsmtFinType1Ord = 3;
if BsmtFinType1 = "Unf" then BsmtFinType1Ord = 2;
if BsmtFinType1 = "NT" then BsmtFinType1Ord = 0;
run;

data test;
set test;
if BsmtFinType1 = "GLQ" then BsmtFinType1Ord = 7;
if BsmtFinType1 = "ALQ" then BsmtFinType1Ord = 6;
if BsmtFinType1 = "BLQ" then BsmtFinType1Ord = 5;
if BsmtFinType1 = "Rec" then BsmtFinType1Ord = 4;
if BsmtFinType1 = "LwQ" then BsmtFinType1Ord = 3;
if BsmtFinType1 = "Unf" then BsmtFinType1Ord = 2;
if BsmtFinType1 = "NT" then BsmtFinType1Ord = 0;
run;


data train;
```

```
set train;
if BsmtFinType2 = "GLQ" then BsmtFinType2Ord = 6;
if BsmtFinType2 = "ALQ" then BsmtFinType2Ord = 5;
if BsmtFinType2 = "BLQ" then BsmtFinType2Ord = 4;
if BsmtFinType2 = "Rec" then BsmtFinType2Ord = 3;
if BsmtFinType2 = "LwQ" then BsmtFinType2Ord = 2;
if BsmtFinType2 = "Unf" then BsmtFinType2Ord = 1;
if BsmtFinType2 = "NT" then BsmtFinType2Ord = 0;
run;

data test;
set test;
if BsmtFinType2 = "GLQ" then BsmtFinType2Ord = 6;
if BsmtFinType2 = "ALQ" then BsmtFinType2Ord = 5;
if BsmtFinType2 = "BLQ" then BsmtFinType2Ord = 4;
if BsmtFinType2 = "Rec" then BsmtFinType2Ord = 3;
if BsmtFinType2 = "LwQ" then BsmtFinType2Ord = 2;
if BsmtFinType2 = "Unf" then BsmtFinType2Ord = 1;
if BsmtFinType2 = "NT" then BsmtFinType2Ord = 0;
run;


data train;
set train;
if KitchenQual = "Ex" then KitchenQualOrd = 5;
if KitchenQual = "GD" then KitchenQualOrd = 4;
if KitchenQual = "TA" then KitchenQualOrd = 3;
if KitchenQual = "Fa" then KitchenQualOrd = 2;
if KitchenQual = "Po" then KitchenQualOrd = 1;
if KitchenQual = "NT" then KitchenQualOrd = 0;
run;

data test;
set test;
if KitchenQual = "Ex" then KitchenQualOrd = 5;
if KitchenQual = "GD" then KitchenQualOrd = 4;
if KitchenQual = "TA" then KitchenQualOrd = 3;
if KitchenQual = "Fa" then KitchenQualOrd = 2;
if KitchenQual = "Po" then KitchenQualOrd = 1;
if KitchenQual = "NT" then KitchenQualOrd = 0;
run;


data train;
set train;
if GarageQual = "Ex" then GarageQualOrd = 5;
if GarageQual = "GD" then GarageQualOrd = 4;
if GarageQual = "TA" then GarageQualOrd = 3;
if GarageQual = "Fa" then GarageQualOrd = 2;
if GarageQual = "Po" then GarageQualOrd = 1;
if GarageQual = "NT" then GarageQualOrd = 0;
run;

data test;
set test;
if GarageQual = "Ex" then GarageQualOrd = 5;
if GarageQual = "GD" then GarageQualOrd = 4;
if GarageQual = "TA" then GarageQualOrd = 3;
if GarageQual = "Fa" then GarageQualOrd = 2;
if GarageQual = "Po" then GarageQualOrd = 1;
if GarageQual = "NT" then GarageQualOrd = 0;
run;


data train;
set train;
if GarageCond = "Ex" then GarageCondOrd = 5;
if GarageCond = "GD" then GarageCondOrd = 4;
if GarageCond = "TA" then GarageCondOrd = 3;
if GarageCond = "Fa" then GarageCondOrd = 2;
if GarageCond = "Po" then GarageCondOrd = 1;
if GarageCond = "NT" then GarageCondOrd = 0;
run;

data test;
set test;
if GarageCond = "Ex" then GarageCondOrd = 5;
if GarageCond = "GD" then GarageCondOrd = 4;
if GarageCond = "TA" then GarageCondOrd = 3;
if GarageCond = "Fa" then GarageCondOrd = 2;
if GarageCond = "Po" then GarageCondOrd = 1;
if GarageCond = "NT" then GarageCondOrd = 0;
run;


data train;
set train;
if PoolQC = "Ex" then PoolQCOrd = 5;
if PoolQC = "GD" then PoolQCOrd = 4;
if PoolQC = "TA" then PoolQCOrd = 3;
if PoolQC = "Fa" then PoolQCOrd = 2;
if PoolQC = "NT" then PoolQCOrd = 0;
run;

data test;
set test;
if PoolQC = "Ex" then PoolQCOrd = 5;
if PoolQC = "GD" then PoolQCOrd = 4;
if PoolQC = "TA" then PoolQCOrd = 3;
if PoolQC = "Fa" then PoolQCOrd = 2;
if PoolQC = "NT" then PoolQCOrd = 0;
run;


data train;
set train;
if Fence = "GdPrv" then FenceOrd = 4;
if Fence = "MnPrv" then FenceOrd = 3;
if Fence = "GdWo" then FenceOrd = 2;
if Fence = "MnWw" then FenceOrd = 1;
if Fence = "NT" then FenceOrd = 0;
run;

data test;
```

```
set test;
if Fence = "GdPrv" then FenceOrd = 4;
if Fence = "MnPrv" then FenceOrd = 3;
if Fence = "GdWo" then FenceOrd = 2;
if Fence = "MnWw" then FenceOrd = 1;
if Fence = "NT" then FenceOrd = 0;
run;


data train;
set train;
if Functional = "Typ" then FunctionalOrd = 8;
if Functional = "Min1" then FunctionalOrd = 7;
if Functional = "Min2" then FunctionalOrd = 6;
if Functional = "Mod" then FunctionalOrd = 5;
if Functional = "Maj1" then FunctionalOrd = 4;
if Functional = "Maj2" then FunctionalOrd = 3;
if Functional = "Sev" then FunctionalOrd = 2;
if Functional = "Sal" then FunctionalOrd = 1;
if Functional = "NT" then FunctionalOrd = 0;
run;

data test;
set test;
if Functional = "Typ" then FunctionalOrd = 8;
if Functional = "Min1" then FunctionalOrd = 7;
if Functional = "Min2" then FunctionalOrd = 6;
if Functional = "Mod" then FunctionalOrd = 5;
if Functional = "Maj1" then FunctionalOrd = 4;
if Functional = "Maj2" then FunctionalOrd = 3;
if Functional = "Sev" then FunctionalOrd = 2;
if Functional = "Sal" then FunctionalOrd = 1;
if Functional = "NT" then FunctionalOrd = 0;
run;



/* The code below creates the submission file to Kaggle */ It can only be run
after the load and clean script has been run and after the PCA script. A file
was made that has all the SAS scripts in order to facilitate generating the
Kaggle score submission table. It is attached with this report.


proc prinqual data=test out=prinqtest1a replace converge=1e-10 maxiter=200;
      transform
                  monotone(BsmtCondOrd ExterQualOrd BsmtFinType2Ord
                                    GarageQualOrd GarageCondOrd FenceOrd);
      run; quit;

data prinqtest2b;
set test;
keep BsmtFinSF2 BsmtUnfSF EnclosedPorch GarageArea KitchenAbvGr
      LotFrontage LowQualFinSF MasVnrArea MiscVal OpenPorchSF
      PoolArea ScreenPorch TotRmsAbvGrd TotalBsmtSF WoodDeckSF
      _1stFlrSF _2ndFlrSF _3SsnPorch;
run;
```

```
data prinqtest1a;
set prinqtest1a;
set prinqtest2b;
run;

proc princomp data=prinqtest1a out=pcatest1a;
run; quit;

data pcatest2b;
set test;
keep Id logSalePrice logGrLivArea log_2ndFlrSF OpenPorchSF BsmtFinSF1
GarageArea logLotArea ScreenPorch
            BldgType BsmtExposure BsmtQual Condition1Group KitchenQual
SaleCondition NeighborhoodGroup RoofMatl MSZoning Exterior1st CentralAir
PoolQC
            OverallCond OverallQual YearBuilt YearRemodAdd YrSold FullBath
GarageCars FirePlaces BsmtFinType1Ord FunctionalOrd;
run;

data pcatest1a;
set pcatest1a;
set pcatest2b;
run;

data loghousing;
set pca1a pcatest1a;
run;

*proc print data=loghousing; run;

/* kaggle = 0.11991, adjrsquared = 0.9464, e_average = 0.093887  */
proc glm data = loghousing plots=all;
            class

                  BldgType BsmtExposure BsmtQual Condition1Group KitchenQual
SaleCondition NeighborhoodGroup RoofMatl
                  MSZoning Exterior1st CentralAir PoolQC;

            model logSalePrice = log_2ndFlrSF OpenPorchSF BsmtFinSF1
GarageArea logLotArea ScreenPorch

                  BldgType BsmtExposure BsmtQual Condition1Group KitchenQual
SaleCondition NeighborhoodGroup RoofMatl
                  MSZoning Exterior1st CentralAir PoolQC

                  OverallCond OverallCond*OverallCond OverallQual
OverallQual*OverallQual
                  YearBuilt YearRemodAdd FullBath GarageCars FirePlaces
                  FullBath*FullBath
                  BsmtFinType1Ord
                  FunctionalOrd

                  logGrLivArea*NeighborhoodGroup
                  logGrLivArea*PoolQC
                  logLotArea*log_2ndFlrSF
                  Prin1 prin12 prin18 Prin20;
```

```
      output out = predictions predicted = prediction;

run; quit;

data predictions;
set predictions;
prediction = exp(prediction) - 1;
run;


/* the predictions for test data */
data finalprediction;
set predictions;
if Id > 1460;
SalePrice = prediction;
keep Id SalePrice;
run;



/* Be sure to change the outfile to your own file location */
/* The outfile "submission.csv" will be used in submission to kaggle */
proc export data=finalprediction outfile='C:\Users\hp\Desktop\SMU\Exp Stats
II\Homework and projects\Project 1\Submissions\submission20a.csv' replace
dbms=csv; run;


/* If this comes out with nothing then you should be able to submit cleanly.
*/
title 'Observations with Missing Values';
data finalpredictionMissing;
set finalprediction;
if SalePrice = .;
run;
proc print data=finalpredictionMissing;
run;

data finalprediction;
set predictions;
if Id > 1460;
SalePrice = prediction;
      run;






/* This code will be used to test models with another training and test set
*/
/* It calculates the Root Mean Squared Logarithmic Error for the test2 set */
/* Run this code after you have loaded and cleaned the data */


/* Try to make this into a loop with 20 seeds 11-30 */
/* This code directly below create principal components for the six numerical
variables */
/*
proc princomp plots=all data=train out=pca1a;
      var logGrLivArea log_2ndFlrSF BsmtFinSF1 GarageArea logLotArea
ScreenPorch;
run; quit;
*/

data pca1a;
set pca1a;
run;


%macro mcv(seednum);

    data trainrandom;
    set pca1a;
    RandNumber = ranuni(&seednum);
    run;

    data train2;
    set trainrandom;
    if RandNumber <= 1/2 then delete;
    run;

    data test2;
    set trainrandom;
    if RandNumber > 1/2 then delete;
    logSalePriceReal = logSalePrice;
    run;

    data test2;
    set test2;
    drop logSalePrice;
    run;

    data housing2;
    set train2 test2;
    run;


    ods exclude all;
    proc glm data = housing2 plots(unpack)=all;
            class

                  BldgType BsmtExposure BsmtQual Condition1Group KitchenQual
SaleCondition NeighborhoodGroup RoofMatl
                  MSZoning Exterior1st CentralAir PoolQC;

            model logSalePrice = log_2ndFlrSF OpenPorchSF BsmtFinSF1
GarageArea logLotArea ScreenPorch

                  BldgType BsmtExposure BsmtQual Condition1Group KitchenQual
SaleCondition NeighborhoodGroup RoofMatl
                  MSZoning Exterior1st CentralAir PoolQC
                  OverallCond OverallCond*OverallCond OverallQual
OverallQual*OverallQual
                  YearBuilt YearRemodAdd FullBath GarageCars FirePlaces
                  FullBath*FullBath
                  BsmtFinType1Ord
                  FunctionalOrd

                  logGrLivArea*NeighborhoodGroup
                  logGrLivArea*PoolQC
            logLotArea*log_2ndFlrSF
                  Prin1 prin12 prin18 Prin20;

        output out = predictions predicted = prediction cookd = cookd;
    run; quit;


    /*
    proc sort data=predictions;
    by descending cookd;
    run;
    */

    *proc print data=predictions (obs=20); run;


    data predictions;
    set predictions;
    SalePriceReal = exp(logSalePriceReal) - 1;
    prediction = exp(prediction) - 1;
    run;

    /* the predictions for test2 data */
    data finalprediction2;
    set predictions;
    if RandNumber <= 1/2;
    difference = (prediction - SalePriceReal)*(prediction - SalePriceReal);
    logdiff = (log(prediction + 1) - log(SalePriceReal + 1))*(log(prediction
+ 1) - log(SalePriceReal + 1));
    keep RandNumber Id difference logdiff;
    run;

    data finalpredictionMissing2;
    set finalprediction2;
    if difference = .;
    run;

    title 'Rows with missing predictions';
    *proc print data=finalpredictionMissing2; run;

    proc summary data = finalprediction2;
    var logdiff;
    output out = diffsummary mean=mean sum=sum n=n;
    run; quit;

    data sqr&seednum;
    set diffsummary;
    e = sqrt(sum/(n));
    e&seednum = sqrt(sum/(n));
    keep e&seednum;
    run;


    ods exclude none;


    *title 'Root Mean Squared Logarithmic Error';
    *proc print data=sqr&seednum; run; quit;

%mend mcv;


%mcv(11)
%mcv(12)
%mcv(13)
%mcv(14)
%mcv(15)
%mcv(16)
%mcv(17)
%mcv(18)
%mcv(19)
%mcv(20)
%mcv(21)
%mcv(22)
%mcv(23)
%mcv(24)
%mcv(25)
%mcv(26)
%mcv(27)
%mcv(28)
%mcv(29)
%mcv(30)


data eaverage;
set sqr11;
set sqr12;
set sqr13;
set sqr14;
set sqr15;
set sqr16;
set sqr17;
set sqr18;
set sqr19;
set sqr20;
set sqr21;
set sqr22;
set sqr23;
set sqr24;
set sqr25;
set sqr26;
set sqr27;
set sqr28;
set sqr29;
set sqr30;
```

```
eaverage = (e11 + e12 + e13 + e14 + e15 + e16 + e17 + e18 + e19 + e20 + e21 +
e22 + e23 + e24 + e25 + e26 + e27 + e28 + e29 + e30)/20;
keep eaverage;
run;


title 'Average Root Mean Squared Logarithmic Error for 20 seeds';
proc print data=eaverage; run; quit;

/*
proc sort data=predictions;
by cookd;
run;
      proc print data=predictions; run;
```

Appendix II Parameter table estimates for Question 1 Model 2.

Below is a table with all the parameter estimates for Model 2 of Question 1.

Parameter	Estimate		Standard	t Value	Pr > |t|	95% Confidence Limits	
			Error				
Intercept	65170.952	B	6300.7675	10.34	<.0001	52810.818	77531.0854
GrLivArea	54.5817	B	4.6756	11.67	<.0001	45.4097	63.7537
LotArea	0.6753		0.0826	8.18	<.0001	0.5133	0.8373
MSSubClass 30	-21609.83	B	4610.1809	-4.69	<.0001	-30653.565	-12566.101
MSSubClass 40	-13168.65	B	13638.196	-0.97	0.3344	-39922.517	13585.2261
MSSubClass 45	-19121.37	B	8364.8658	-2.29	0.0224	-35530.613	-2712.1165
MSSubClass 50	-31049.29	B	3394.1512	-9.15	<.0001	-37707.55	-24391.025
MSSubClass 60	-25442.42	B	2729.7374	-9.32	<.0001	-30797.306	-20087.526
MSSubClass 70	-39664.04	B	4790.0548	-8.28	<.0001	-49060.627	-30267.45
MSSubClass 75	-37615.87	B	8252.6464	-4.56	<.0001	-53804.982	-21426.764
MSSubClass 80	-7998.147	B	3864.3994	-2.07	0.0387	-15578.89	-417.4054
MSSubClass 85	5325.6338	B	6217.7771	0.86	0.3919	-6871.6987	17522.9664
MSSubClass 90	-29566.88	B	4273.951	-6.92	<.0001	-37951.035	-21182.727
MSSubClass 120	-91.9547	B	4187.7209	-0.02	0.9825	-8306.9523	8123.0429
MSSubClass 160	-44026.37	B	5659.1982	-7.78	<.0001	-55127.947	-32924.796
MSSubClass 180	-6729.019	B	12167.52	-0.55	0.5803	-30597.885	17139.8467
MSSubClass 190	-33706.59	B	5777.3211	-5.83	<.0001	-45039.887	-22373.295
MSSubClass 20	0	B	.	.	.	.	.
Neighborhood Blmngtn	-84770.98	B	69561.851	-1.22	0.2232	-221229.56	51687.5938
Neighborhood Blueste	-15710.74	B	161950.67	-0.1	0.9227	-333407.26	301985.771
Neighborhood BrDale	-20015.18	B	52366.646	-0.38	0.7024	-122742.15	82711.7905
Neighborhood BrkSide	-29205.99	B	14879.215	-1.96	0.0499	-58394.352	-17.6255
Neighborhood ClearCr	42614.131	B	21606.843	1.97	0.0488	228.2676	84999.9941
Neighborhood CollgCr	-26037.87	B	10189.732	-2.56	0.0107	-46026.939	-6048.8084
Neighborhood Crawfor	8153.6641	B	14734.771	0.55	0.5801	-20751.345	37058.6737
Neighborhood Edwards	-11539.27	B	12354.078	-0.93	0.3504	-35774.101	12695.5686
Neighborhood Gilbert	-9428.527	B	18245.214	-0.52	0.6054	-45219.927	26362.8723
Neighborhood IDOTRR	-21962.1	B	20838.126	-1.05	0.2921	-62839.979	18915.783
Neighborhood MeadowV	-11038.72	B	22838.394	-0.48	0.6289	-55840.504	33763.0571
Neighborhood Mitchel	-11122.67	B	15159.6	-0.73	0.4633	-40861.056	18615.7248
Neighborhood NPkVill	-72713.51	B	53263.466	-1.37	0.1724	-177199.76	31772.743
Neighborhood NWAmes	-14445.45	B	14970.772	-0.96	0.3348	-43813.414	14922.5237
Neighborhood NoRidge	-142926.2	B	20697.395	-6.91	<.0001	-183527.99	-102324.37

[redacted]

Neighborhood NridgHt	-75304.85	B	17204.163	-4.38	<.0001	-109054.03	-41555.666
Neighborhood OldTown	-2360.728	B	9931.5102	-0.24	0.8121	-21843.243	17121.7865
Neighborhood SWISU	13801.377	B	19001.648	0.73	0.4678	-23473.909	51076.6633
Neighborhood Sawyer	3305.6301	B	13029.693	0.25	0.7998	-22254.55	28865.8103
Neighborhood SawyerW	-31082.9	B	13343.466	-2.33	0.02	-57258.601	-4907.192
Neighborhood Somerst	-19747.08	B	19102.401	-1.03	0.3014	-57220.013	17725.8505
Neighborhood StoneBr	-87620.23	B	21206.69	-4.13	<.0001	-129221.11	-46019.339
Neighborhood Timber	-16032.56	B	21143.979	-0.76	0.4484	-57510.434	25445.3047
Neighborhood Veenker	-53478.91	B	42941.411	-1.25	0.2132	-137716.52	30758.6944
Neighborhood NAmes	0	B	.	.	.	.	.
OverallCond 1	-29128.16	B	40414.852	-0.72	0.4712	-108409.45	50153.1278
OverallCond 2	-16001.2	B	12610.425	-1.27	0.2047	-40738.906	8736.5083
OverallCond 3	-23895.5	B	6123.4269	-3.9	<.0001	-35907.75	-11883.257
OverallCond 4	-8726.718	B	4090.9764	-2.13	0.0331	-16751.933	-701.5025
OverallCond 6	5579.9104	B	2252.6285	2.48	0.0134	1160.9584	9998.8624
OverallCond 7	10926.208	B	2451.4075	4.46	<.0001	6117.314	15735.1024
OverallCond 8	15262.858	B	3608.1331	4.23	<.0001	8184.8303	22340.8857
OverallCond 9	29565.69	B	6290.0729	4.7	<.0001	17226.536	41904.8439
OverallCond 5	0	B	.	.	.	.	.
OverallQual 1	-4146.427	B	29356.39	-0.14	0.8877	-61734.475	53441.6204
OverallQual 2	-15241.22	B	17446.017	-0.87	0.3825	-49464.846	18982.4023
OverallQual 3	-11958.42	B	6571.2345	-1.82	0.069	-24849.12	932.2903
OverallQual 4	-4465.712	B	3068.4785	-1.46	0.1458	-10485.106	1553.6827
OverallQual 6	12281.078	B	2234.4662	5.5	<.0001	7897.7551	16664.4017
OverallQual 7	29562.069	B	2834.0354	10.43	<.0001	24002.579	35121.5597
OverallQual 8	54546.033	B	3773.9671	14.45	<.0001	47142.69	61949.3746
OverallQual 9	107741.02	B	5850.0971	18.42	<.0001	96264.964	119217.083
OverallQual 10	137103.53	B	7896.5339	17.36	<.0001	121613	152594.055
OverallQual 5	0	B	.	.	.	.	.
GrLivArea*Neighborho Blmngtn	70.2775	B	48.5317	1.45	0.1478	-24.9264	165.4814
GrLivArea*Neighborho Blueste	26.6447	B	115.4316	0.23	0.8175	-199.796	253.0855
GrLivArea*Neighborho BrDale	26.0535	B	45.0018	0.58	0.5627	-62.2259	114.3328
GrLivArea*Neighborho BrkSide	29.2515	B	11.2736	2.59	0.0096	7.1362	51.3668
GrLivArea*Neighborho ClearCr	-8.1447	B	12.072	-0.67	0.5	-31.8262	15.5367
GrLivArea*Neighborho CollgCr	35.6866	B	7.1016	5.03	<.0001	21.7555	49.6177
GrLivArea*Neighborho Crawfor	15.5523	B	8.4528	1.84	0.066	-1.0296	32.1341
GrLivArea*Neighborho Edwards	6.7965	B	9.1182	0.75	0.4562	-11.0906	24.6836
GrLivArea*Neighborho Gilbert	22.33	B	11.2236	1.99	0.0468	0.3128	44.3472
GrLivArea*Neighborho IDOTRR	11.1825	B	17.3461	0.64	0.5192	-22.8451	45.2102
GrLivArea*Neighborho MeadowV	8.4602	B	17.5999	0.48	0.6308	-26.0653	42.9857
GrLivArea*Neighborho Mitchel	17.0544	B	11.2441	1.52	0.1296	-5.0029	39.1118

GrLivArea*Neighborho NPkVill	70.8368	B	42.1404	1.68	0.093	-11.8295	153.5031
GrLivArea*Neighborho NWAmes	13.9915	B	8.8859	1.57	0.1156	-3.4398	31.4229
GrLivArea*Neighborho NoRidge	91.9615	B	8.8859	10.35	<.0001	74.5301	109.3929
GrLivArea*Neighborho NridgHt	78.4985	B	9.4401	8.32	<.0001	59.98	97.017
GrLivArea*Neighborho OldTown	-6.131	B	6.6734	-0.92	0.3584	-19.2221	6.96
GrLivArea*Neighborho SWISU	-10.3308	B	10.3802	-1	0.3198	-30.6934	10.0319
GrLivArea*Neighborho Sawyer	-3.1257	B	10.155	-0.31	0.7583	-23.0467	16.7952
GrLivArea*Neighborho SawyerW	32.8645	B	8.3873	3.92	<.0001	16.4113	49.3177
GrLivArea*Neighborho Somerst	41.2805	B	11.7235	3.52	0.0004	18.2827	64.2784
GrLivArea*Neighborho StoneBr	88.5111	B	11.0406	8.02	<.0001	66.8529	110.1692
GrLivArea*Neighborho Timber	28.4211	B	12.1917	2.33	0.0199	4.5048	52.3373
GrLivArea*Neighborho Veenker	63.4592	B	27.4776	2.31	0.0211	9.5568	117.3616
GrLivArea*Neighborho NAmes	0	B	.	.	.	.	.

Appendix III

Model SAS scripts

Since these models can be very large we simply copy the SAS code for proc glm for each of the models discussed in Question 2.

Q2
Model1

```
/******* Model 1 (keep) - Backward elemination (adjrsquared = 0.9479, 573 parameters) *******/
proc glm data=housing plots=all;
Class
          MSSubClass MSZoning Alley LotShape LandContour LotConfig Neighborhood Condition1
          BldgType HouseStyle  OverallQual OverallCond RoofStyle RoofMatl Exterior1st
          Exterior2nd MasVnrType MasVnrArea ExterQual ExterCond Foundation BsmtQual
          BsmtCond BsmtExposure BsmtFinType1 BsmtFinType2 HeatingQC HeatingQC Electrical
          KitchenQual FireplaceQu GarageType GarageFinish GarageQual GarageCond PavedDrive
          PoolQC  Fence SaleType SaleCondition ;
model saleprice =
          BedroomAbvGr BsmtFinSF1 BsmtFinSF2 BsmtFullBath BsmtHalfBath BsmtUnfSF EnclosedPorch
          Fireplaces FullBath GarageArea GarageCars GarageYrBlt GrLivArea HalfBath Id KitchenAbvGr
          LotArea LotFrontage LowQualFinSF MSSubClass MasVnrArea MiscVal MoSold OpenPorchSF
          OverallCond OverallQual PoolArea ScreenPorch TotRmsAbvGrd WoodDeckSF YearBuilt
          YearRemodAdd YrSold _1stFlrSF _3SsnPorch MSZoning Alley LotShape LandContour
          LotConfig Neighborhood Condition1 BldgType HouseStyle RoofStyle RoofMatl Exterior1st
          Exterior2nd MasVnrType ExterQual ExterCond Foundation BsmtQual BsmtCond BsmtExposure
          BsmtFinType1 BsmtFinType2 HeatingQC Electrical KitchenQual FireplaceQu GarageType
          GarageFinish GarageQual GarageCond PavedDrive PoolQC Fence SaleType SaleCondition;
run; quit;
```

Q2
Model2

```
/******* Model 2 (keep) - Our own cross validation (kaggle = 0.13722, adjrsquared = 0.9245, e = 0.13297)
*******
proc glm data = housing plots=all;
class
          BldgType BsmtExposure BsmtQual Condition2 KitchenQual RoofMatl SaleCondition
          NeighborhoodGroup OverallCondGroup OverallQualGroup ;
model saleprice =
                    _2ndFlrSF MasVnrArea _3SsnPorch BsmtFinSF1 GarageArea LotArea ScreenPorch
                    BldgType BsmtExposure BsmtQual Condition2 KitchenQual RoofMatl SaleCondition
                    NeighborhoodGroup OverallCondGroup OverallQualGroup GrLivArea*NeighborhoodGroup;
run; quit;
```

Q2
Model3

```
/******* Model 3 (keep) - (kaggle= 0.13474, adjrsquared = 0.9259, e = 0.13025) *******/
proc glm data = housing plots=all;
class
          BldgType BsmtExposure BsmtQual Condition1Group Condition2 KitchenQual RoofMatl
          SaleCondition NeighborhoodGroup OverallCondGroup OverallQualGroup;
model saleprice =
          _2ndFlrSF MasVnrArea _3SsnPorch BsmtFinSF1 GarageArea LotArea ScreenPorch
                    BldgType BsmtExposure BsmtQual Condition1Group Condition2 KitchenQual RoofMatl
                    SaleCondition NeighborhoodGroup OverallCondGroup OverallQualGroup
                    GrLivArea*NeighborhoodGroup GrLivArea*RoofMatl;
run; quit;
```

Appendix IV

Other SAS scripts used.

Question 1

```
/* Model 1 (keep) - normal (without interaction) adj-r-squared=.8697, 58
params */
proc glm data = train plots = all;
class MSSubClass (ref = "20") OverallCond (ref = "5") OverallQual (ref = "5")
Neighborhood (ref = "NAmes");
model SalePrice = GrLivArea LotArea MSSubClass OverallCond OverallQual
Neighborhood  / solution tolerance clparm;
run; quit;

/* Model 2 (keep) - Full (with interaction) adj-r-squared=.8873, 82 params */
proc glm data = train plots = all;
class MSSubClass (ref = "20") OverallCond (ref = "5") OverallQual (ref = "5")
Neighborhood (ref = "NAmes");
model SalePrice = GrLivArea LotArea MSSubClass OverallCond OverallQual
Neighborhood GrLivArea*Neighborhood / solution tolerance clparm;
run; quit;

/* Model 3 (keep) - Simple (only continuous) adj-r-squared .6773, 5 params */
proc glm data = train plots=all;
model SalePrice = TotalBsmtSF FullBath GrLivArea LotArea / solution tolerance
clparm;
run; quit;

/* Model 4 (reject) - Just GrLivArea, Neighborhood and interaction adj-r-
squared=.7955, 50 params */
proc glm data = train plots = all;
class MSSubClass (ref = "20") OverallCond (ref = "5") OverallQual (ref = "5")
Neighborhood (ref = "NAmes");
model SalePrice = GrLivArea  GrLivArea*Neighborhood Neighborhood / solution
tolerance clparm;
run; quit;

/* Model 5 (reject) - Just GrLivArea, and Neighborhood interaction adj-r-
squared=.7828, 26 params */
proc glm data = train plots = all;
class MSSubClass (ref = "20") OverallCond (ref = "5") OverallQual (ref = "5")
Neighborhood (ref = "NAmes");
model SalePrice = GrLivArea  GrLivArea*Neighborhood / solution tolerance
clparm;
run; quit;

/* Model 6 (reject) - Full (with groups) adj-r-squared= .885 , 52 params */
proc glm data = train plots = all;
class MSSubClass (ref = "20") OverallCond (ref = "5") OverallQual (ref = "5")
Neighborhood (ref = "NAmes") NeighborhoodGroup OverallCondGroup
OverallQualGroup;
model SalePrice = GrLivArea LotArea MSSubClass OverallCondGroup
OverallQualGroup NeighborhoodGroup   GrLivArea*NeighborhoodGroup / solution
tolerance clparm;
run; quit;
```

Question 2

```
data housing;
set train test;
run;

/******* Model 3 (keep) - (kaggle= 0.13474, adjrsquared = 0.9259, e =
0.13025) *******/
/* Add GrLivArea*RoofMatl interaction */
proc glm data = housing plots=all;
class
      /* Character Categorical Variables */
      BldgType BsmtExposure BsmtQual Condition1Group Condition2 KitchenQual
RoofMatl SaleCondition NeighborhoodGroup

      /* Numerical Categorical Variables */
      OverallCondGroup OverallQualGroup;

      model saleprice = _2ndFlrSF MasVnrArea _3SsnPorch
            BsmtFinSF1 GarageArea LotArea ScreenPorch
            BldgType BsmtExposure BsmtQual Condition1Group Condition2
KitchenQual RoofMatl
            SaleCondition NeighborhoodGroup
            OverallCondGroup OverallQualGroup
            GrLivArea*NeighborhoodGroup
            GrLivArea*RoofMatl;

      output out = predictions predicted = prediction;

run; quit;

/* the predictions for test data */
data finalprediction;
set predictions;
if Id > 1460;
SalePrice = prediction;

keep Id SalePrice;
run;

/* Be sure to change the outfile to your own file location */
/* The outfile "submission.csv" will be used in submission to kaggle */
proc export data=finalprediction
outfile='/home/kjprice120/sasuser.v94/Project1/data/submission.csv' dbms=csv;
run;

title 'Observations with Missing Values';
data finalpredictionMissing;
set finalprediction;
if SalePrice = .;
run;
proc print data=finalpredictionMissing;
run;
```

Manual External Cross validation code

```
/* This code will be used to test models with another training and test set
*/
/* It calculates the Root Mean Squared Logarithmic Error for the test2 set */
/* Run this code after you have loaded and cleaned the data */

/* If you beat this score replace it with the better one kaggle= 0.13474,
adjrsquared = 0.9259, e = 0.13025 */

data trainrandom;
set train;
RandNumber = ranuni(11);
run;

data train2;
set trainrandom;
if RandNumber <= 1/2 then delete;
run;

data test2real;
set trainrandom;
SalePriceReal = saleprice;
if RandNumber > 1/2 then delete;
keep ID RandNumber SalePriceReal;
run;

data test2;
set trainrandom;
if RandNumber > 1/2 then delete;
SalePriceReal = SalePrice;
run;

data test2;
set test2;
drop SalePrice;
run;


data housing2;
set train2 test2;
run;


/* Place the model you want to use here */

/* kaggle= 0.13474, adjrsquared = 0.9259, e = 0.13025  */
/* Add RoofMatl*GrLivArea interaction */
proc glm data = housing2 plots=all;
class
      /* Character Categorical Variables */
      BldgType BsmtExposure BsmtQual Condition1Group Condition2 KitchenQual
RoofMatl SaleCondition NeighborhoodGroup

      /* Numerical Categorical Variables */
      OverallCondGroup OverallQualGroup;
      model saleprice = _2ndFlrSF MasVnrArea _3SsnPorch
            BsmtFinSF1 GarageArea LotArea ScreenPorch
            BldgType BsmtExposure BsmtQual Condition1Group Condition2
KitchenQual RoofMatl
            SaleCondition NeighborhoodGroup
            OverallCondGroup OverallQualGroup
            GrLivArea*NeighborhoodGroup
            RoofMatl*GrLivArea;

      output out = predictions predicted = prediction;
run; quit;


/* the predictions for test2 data */
data finalprediction2;
set predictions;
if RandNumber <= 1/2;
difference = (prediction - SalePriceReal)*(prediction - SalePriceReal);
logdiff = (log(prediction + 1) - log(SalePriceReal + 1))*(log(prediction + 1)
- log(SalePriceReal + 1));
keep RandNumber Id difference logdiff;
run;

data finalpredictionMissing2;
set finalprediction2;
if difference = .;
run;

title 'Rows with missing predictions';
proc print data=finalpredictionMissing2; run;

proc summary data = finalprediction2;
var logdiff;
output out = diffsummary mean=mean sum=sum n=n;
run;

data sqr;
set diffsummary;
e = sqrt(sum/(n));
keep e;
run;

title 'Root Mean Squared Logarithmic Error';
proc print data=sqr; run;
```

Automatic External Cross validation code

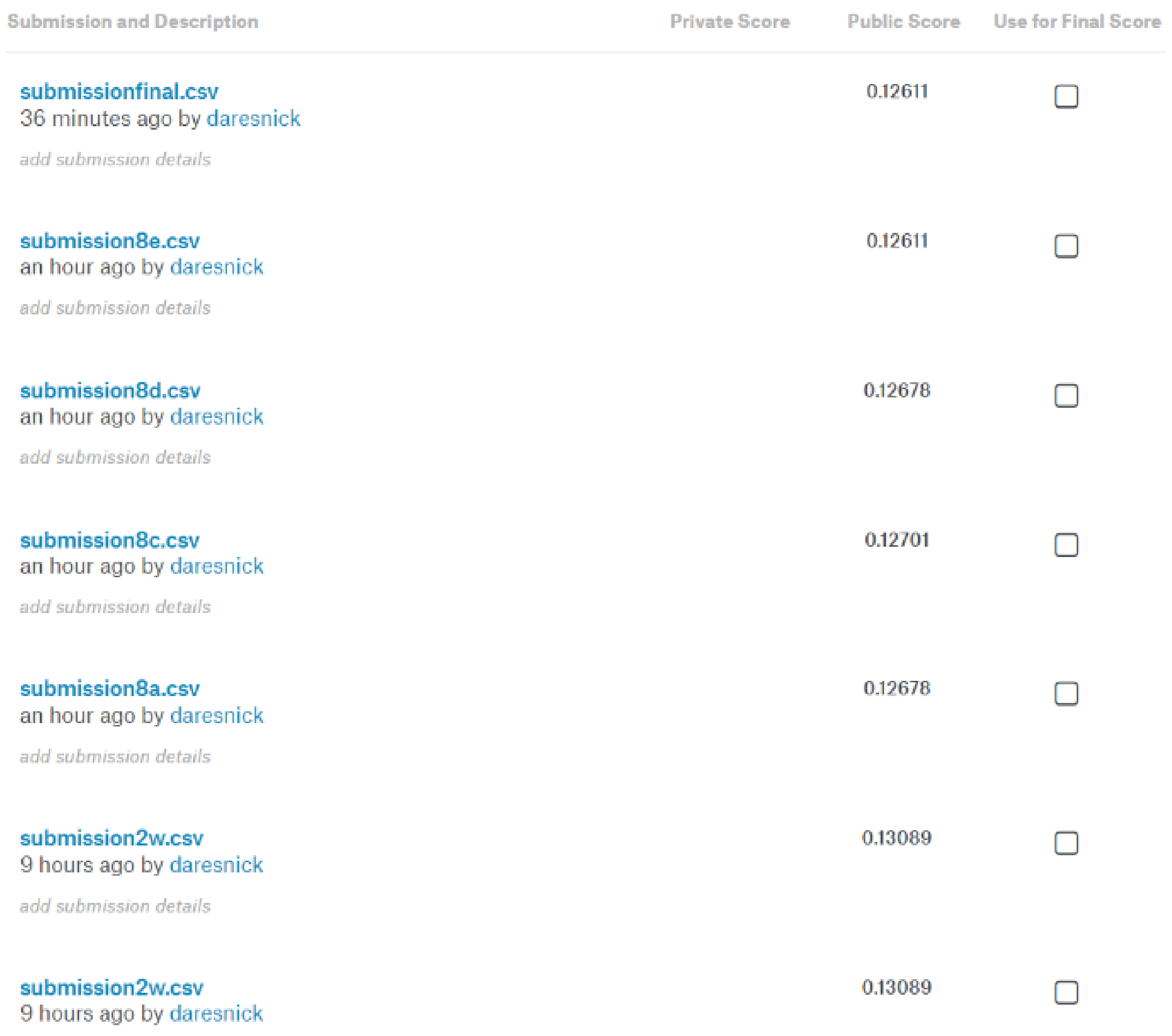
```
/* This code is designed to be run after data is loaded and cleaned. */

data trainrandom;
set train;
```

```
RandNumber = ranuni(11);
run;

data train3;
set trainrandom;
if RandNumber <= 1/2 then delete;
run;

data test3;
set trainrandom;
if RandNumber > 1/2 then delete;
run;

/* With Stepwise */
proc glmselect data = train3 testdata=test3
                        seed = 1 plots(stepAxis=number)=(criterionPanel
ASEPlot CRITERIONPANEL);
class
      /* Character Categorical Variables */
      BldgType BsmtExposure BsmtQual Condition2 KitchenQual RoofMatl
SaleCondition NeighborhoodGroup

      /* Numerical Categorical Variables */
      OverallCond OverallQual;

      /* model taken by using proc glmselect with stepwise selection  on KJ
model (kaggle = 0.14060, adjrsquared = 0.9162) */
      /* there are slight tolerance/VIF issues with OverallQual 4-8 and
OverallCond 5-6  */
      model saleprice =
            BsmtFinSF1 GarageArea LotArea ScreenPorch
            BldgType BsmtExposure BsmtQual Condition2 KitchenQual RoofMatl
SaleCondition NeighborhoodGroup
            OverallCond OverallQual
            GrLivArea*NeighborhoodGroup / selection= stepwise(choose=CV
stop=AIC) CVdetails;
run; quit;


/* With LASSO */
proc glmselect data = train3 testdata=test3
                        seed = 1 plots(stepAxis=number)=(criterionPanel
ASEPlot CRITERIONPANEL);
class
      /* Character Categorical Variables */
      BldgType BsmtExposure BsmtQual Condition2 KitchenQual RoofMatl
SaleCondition NeighborhoodGroup

      /* Numerical Categorical Variables */
      OverallCond OverallQual;

      /* model taken by using proc glmselect with stepwise selection  on KJ
model (kaggle = 0.14060, adjrsquared = 0.9162) */
      /* there are slight tolerance/VIF issues with OverallQual 4-8 and
OverallCond 5-6  */
      model saleprice =
            BsmtFinSF1 GarageArea LotArea ScreenPorch
            BldgType BsmtExposure BsmtQual Condition2 KitchenQual RoofMatl
SaleCondition NeighborhoodGroup
            OverallCond OverallQual
            GrLivArea*NeighborhoodGroup / selection= LASSO(choose=CV
stop=AIC) CVdetails;
run; quit;
```

[redacted]

Appendix V

Screen shot of final Kaggle submissions



Submission and Description	Private Score	Public Score	Use for Final Score
submissionfinal.csv 36 minutes ago by daresnick *add submission details*		0.12611	☐
submission8e.csv an hour ago by daresnick *add submission details*		0.12611	☐
submission8d.csv an hour ago by daresnick *add submission details*		0.12678	☐
submission8c.csv an hour ago by daresnick *add submission details*		0.12701	☐
submission8a.csv an hour ago by daresnick *add submission details*		0.12678	☐
submission2w.csv 9 hours ago by daresnick *add submission details*		0.13089	☐
submission2w.csv 9 hours ago by daresnick		0.13089	☐

Appendix VI

Variable Description
The document below is just a copy of the file Kaggle has on there website.

https://www.kaggle.com/c/house-prices-advanced-regression-techniques/data

MSSubClass: Identifies the type of dwelling involved in the sale.

20	1-STORY 1946 & NEWER ALL STYLES
30	1-STORY 1945 & OLDER
40	1-STORY W/FINISHED ATTIC ALL AGES
45	1-1/2 STORY - UNFINISHED ALL AGES
50	1-1/2 STORY FINISHED ALL AGES
60	2-STORY 1946 & NEWER
70	2-STORY 1945 & OLDER
75	2-1/2 STORY ALL AGES
80	SPLIT OR MULTI-LEVEL
85	SPLIT FOYER
90	DUPLEX - ALL STYLES AND AGES
120	1-STORY PUD (Planned Unit Development) - 1946 & NEWER
150	1-1/2 STORY PUD - ALL AGES
160	2-STORY PUD - 1946 & NEWER
180	PUD - MULTILEVEL - INCL SPLIT LEV/FOYER
190	2 FAMILY CONVERSION - ALL STYLES AND AGES

MSZoning: Identifies the general zoning classification of the sale.

A	Agriculture
C	Commercial
FV	Floating Village Residential
I	Industrial
RH	Residential High Density
RL	Residential Low Density
RP	Residential Low Density Park
RM	Residential Medium Density

LotFrontage: Linear feet of street connected to property

LotArea: Lot size in square feet

Street: Type of road access to property

Grvl	Gravel

Pave	Paved

Alley: Type of alley access to property

Grvl	Gravel
Pave	Paved
NA	No alley access

LotShape: General shape of property

Reg	Regular
IR1	Slightly irregular
IR2	Moderately Irregular
IR3	Irregular

LandContour: Flatness of the property

Lvl	Near Flat/Level
Bnk	Banked - Quick and significant rise from street grade to building
HLS	Hillside - Significant slope from side to side
Low	Depression

Utilities: Type of utilities available

AllPub	All public Utilities (E,G,W,& S)
NoSewr	Electricity, Gas, and Water (Septic Tank)
NoSeWa	Electricity and Gas Only
ELO	Electricity only

LotConfig: Lot configuration

Inside	Inside lot
Corner	Corner lot
CulDSac	Cul-de-sac
FR2	Frontage on 2 sides of property
FR3	Frontage on 3 sides of property

LandSlope: Slope of property

Gtl	Gentle slope
Mod	Moderate Slope
Sev	Severe Slope

Neighborhood: Physical locations within Ames city limits

Blmngtn	Bloomington Heights
Blueste	Bluestem
BrDale	Briardale
BrkSide	Brookside
ClearCr	Clear Creek
CollgCr	College Creek
Crawfor	Crawford
Edwards	Edwards
Gilbert	Gilbert
IDOTRR	Iowa DOT and Rail Road
MeadowV	Meadow Village
Mitchel	Mitchell
Names	North Ames
NoRidge	Northridge
NPkVill	Northpark Villa
NridgHt	Northridge Heights
NWAmes	Northwest Ames
OldTown	Old Town
SWISU	South & West of Iowa State University
Sawyer	Sawyer
SawyerW	Sawyer West
Somerst	Somerset
StoneBr	Stone Brook
Timber	Timberland
Veenker	Veenker

Condition1: Proximity to various conditions

Artery	Adjacent to arterial street
Feedr	Adjacent to feeder street
Norm	Normal
RRNn	Within 200' of North-South Railroad
RRAn	Adjacent to North-South Railroad
PosN	Near positive off-site feature--park, greenbelt, etc.
PosA	Adjacent to postive off-site feature
RRNe	Within 200' of East-West Railroad
RRAe	Adjacent to East-West Railroad

Condition2: Proximity to various conditions (if more than one is present)

Artery	Adjacent to arterial street
Feedr	Adjacent to feeder street
Norm	Normal
RRNn	Within 200' of North-South Railroad
RRAn	Adjacent to North-South Railroad
PosN	Near positive off-site feature--park, greenbelt, etc.
PosA	Adjacent to postive off-site feature
RRNe	Within 200' of East-West Railroad

RRAe	Adjacent to East-West Railroad

BldgType: Type of dwelling

1Fam	Single-family Detached
2FmCon	Two-family Conversion; originally built as one-family dwelling
Duplx	Duplex
TwnhsE	Townhouse End Unit
TwnhsI	Townhouse Inside Unit

HouseStyle: Style of dwelling

1Story	One story
1.5Fin	One and one-half story: 2nd level finished
1.5Unf	One and one-half story: 2nd level unfinished
2Story	Two story
2.5Fin	Two and one-half story: 2nd level finished
2.5Unf	Two and one-half story: 2nd level unfinished
SFoyer	Split Foyer
SLvl	Split Level

OverallQual: Rates the overall material and finish of the house

10	Very Excellent
9	Excellent
8	Very Good
7	Good
6	Above Average
5	Average
4	Below Average
3	Fair
2	Poor
1	Very Poor

OverallCond: Rates the overall condition of the house

10	Very Excellent
9	Excellent
8	Very Good
7	Good
6	Above Average
5	Average
4	Below Average
3	Fair
2	Poor
1	Very Poor

[redacted]

YearBuilt: Original construction date

YearRemodAdd: Remodel date (same as construction date if no remodeling or additions)

RoofStyle: Type of roof

Flat	Flat
Gable	Gable
Gambrel	Gabrel (Barn)
Hip	Hip
Mansard	Mansard
Shed	Shed

RoofMatl: Roof material

ClyTile	Clay or Tile
CompShg	Standard (Composite) Shingle
Membran	Membrane
Metal	Metal
Roll	Roll
Tar&Grv	Gravel & Tar
WdShake	Wood Shakes
WdShngl	Wood Shingles

Exterior1st: Exterior covering on house

AsbShng	Asbestos Shingles
AsphShn	Asphalt Shingles
BrkComm	Brick Common
BrkFace	Brick Face
CBlock	Cinder Block
CemntBd	Cement Board
HdBoard	Hard Board
ImStucc	Imitation Stucco
MetalSd	Metal Siding
Other	Other
Plywood	Plywood
PreCast	PreCast
Stone	Stone
Stucco	Stucco
VinylSd	Vinyl Siding
Wd Sdng	Wood Siding
WdShing	Wood Shingles

Exterior2nd: Exterior covering on house (if more than one material)

AsbShng	Asbestos Shingles
AsphShn	Asphalt Shingles
BrkComm	Brick Common
BrkFace	Brick Face
CBlock	Cinder Block
CemntBd	Cement Board
HdBoard	Hard Board
ImStucc	Imitation Stucco
MetalSd	Metal Siding
Other	Other
Plywood	Plywood
PreCast	PreCast
Stone	Stone
Stucco	Stucco
VinylSd	Vinyl Siding
Wd Sdng	Wood Siding
WdShing	Wood Shingles

MasVnrType: Masonry veneer type

BrkCmn	Brick Common
BrkFace	Brick Face
CBlock	Cinder Block
None	None
Stone	Stone

MasVnrArea: Masonry veneer area in square feet

ExterQual: Evaluates the quality of the material on the exterior

Ex	Excellent
TA	Average/Typical
Fa	Fair
Po	Poor

ExterCond: Evaluates the present condition of the material on the exterior

Ex	Excellent
Gd	Good
TA	Average/Typical
Fa	Fair
Po	Poor

Foundation: Type of foundation

BrkTil	Brick & Tile
CBlock	Cinder Block

PConc	Poured Contrete
Slab	Slab
Stone	Stone
Wood	Wood

BsmtQual: Evaluates the height of the basement

Ex	Excellent (100+ inches)
Gd	Good (90-99 inches)
TA	Typical (80-89 inches)
Fa	Fair (70-79 inches)
Po	Poor (<70 inches
NA	No Basement

BsmtCond: Evaluates the general condition of the basement

Ex	Excellent
Gd	Good
TA	Typical - slight dampness allowed
Fa	Fair - dampness or some cracking or settling
Po	Poor - Severe cracking, settling, or wetness
NA	No Basement

BsmtExposure: Refers to walkout or garden level walls

Gd	Good Exposure
Av	Average Exposure (split levels or foyers typically score average or above)
Mn	Mimimum Exposure
No	No Exposure
NA	No Basement

BsmtFinType1: Rating of basement finished area

GLQ	Good Living Quarters
ALQ	Average Living Quarters
BLQ	Below Average Living Quarters
Rec	Average Rec Room
LwQ	Low Quality
Unf	Unfinshed
NA	No Basement

BsmtFinSF1: Type 1 finished square feet

BsmtFinType2: Rating of basement finished area (if multiple types)

GLQ	Good Living Quarters
ALQ	Average Living Quarters
BLQ	Below Average Living Quarters
Rec	Average Rec Room
LwQ	Low Quality
Unf	Unfinshed
NA	No Basement

BsmtFinSF2: Type 2 finished square feet

BsmtUnfSF: Unfinished square feet of basement area

TotalBsmtSF: Total square feet of basement area

Heating: Type of heating

Floor	Floor Furnace
GasA	Gas forced warm air furnace
GasW	Gas hot water or steam heat
Grav	Gravity furnace
OthW	Hot water or steam heat other than gas
Wall	Wall furnace

HeatingQC: Heating quality and condition

Ex	Excellent
Gd	Good
TA	Average/Typical
Fa	Fair
Po	Poor

HeatingQC: Central air conditioning

N	No
Y	Yes

Electrical: Electrical system

SBrkr	Standard Circuit Breakers & Romex
FuseA	Fuse Box over 60 AMP and all Romex wiring (Average)
FuseF	60 AMP Fuse Box and mostly Romex wiring (Fair)
FuseP	60 AMP Fuse Box and mostly knob & tube wiring (poor)
Mix	Mixed

1stFlrSF: First Floor square feet

2ndFlrSF: Second floor square feet

LowQualFinSF: Low quality finished square feet (all floors)

GrLivArea: Above grade (ground) living area square feet

BsmtFullBath: Basement full bathrooms

BsmtHalfBath: Basement half bathrooms

FullBath: Full bathrooms above grade

HalfBath: Half baths above grade

Bedroom: Bedrooms above grade (does NOT include basement bedrooms)

Kitchen: Kitchens above grade

KitchenQual: Kitchen quality

Ex	Excellent
Gd	Good
TA	Typical/Average
Fa	Fair
Po	Poor

TotRmsAbvGrd: Total rooms above grade (does not include bathrooms)

Functional: Home functionality (Assume typical unless deductions are warranted)

Typ	Typical Functionality
Min1	Minor Deductions 1
Min2	Minor Deductions 2
Mod	Moderate Deductions
Maj1	Major Deductions 1
Maj2	Major Deductions 2
Sev	Severely Damaged
Sal	Salvage only

Fireplaces: Number of fireplaces

FireplaceQu: Fireplace quality

Ex	Excellent - Exceptional Masonry Fireplace

Gd	Good - Masonry Fireplace in main level
TA	Average - Prefabricated Fireplace in main living area or Masonry Fireplace in basement
Fa	Fair - Prefabricated Fireplace in basement
Po	Poor - Ben Franklin Stove
NA	No Fireplace

GarageType: Garage location

2Types	More than one type of garage
Attchd	Attached to home
Basment	Basement Garage
BuiltIn	Built-In (Garage part of house - typically has room above garage)
CarPort	Car Port
Detchd	Detached from home
NA	No Garage

GarageYrBlt: Year garage was built

GarageFinish: Interior finish of the garage

Fin	Finished
RFn	Rough Finished
Unf	Unfinished
NA	No Garage

GarageCars: Size of garage in car capacity

GarageArea: Size of garage in square feet

GarageQual: Garage quality

Ex	Excellent
Gd	Good
TA	Typical/Average
Fa	Fair
Po	Poor
NA	No Garage

GarageCond: Garage condition

Ex	Excellent
Gd	Good
TA	Typical/Average
Fa	Fair

Po	Poor
NA	No Garage

PavedDrive: Paved driveway

Y	Paved
P	Partial Pavement
N	Dirt/Gravel

WoodDeckSF: Wood deck area in square feet

OpenPorchSF: Open porch area in square feet

EnclosedPorch: Enclosed porch area in square feet

3SsnPorch: Three season porch area in square feet

ScreenPorch: Screen porch area in square feet

PoolArea: Pool area in square feet

PoolQC: Pool quality

Ex	Excellent
Gd	Good
TA	Average/Typical
Fa	Fair
NA	No Pool

Fence: Fence quality

GdPrv	Good Privacy
MnPrv	Minimum Privacy
GdWo	Good Wood
MnWw	Minimum Wood/Wire
NA	No Fence

MiscFeature: Miscellaneous feature not covered in other categories

Elev	Elevator
Gar2	2nd Garage (if not described in garage section)
Othr	Other
Shed	Shed (over 100 SF)
TenC	Tennis Court
NA	None

MiscVal: $Value of miscellaneous feature

MoSold: Month Sold (MM)

YrSold: Year Sold (YYYY)

SaleType: Type of sale

WD	Warranty Deed - Conventional
CWD	Warranty Deed - Cash
VWD	Warranty Deed - VA Loan
New	Home just constructed and sold
COD	Court Officer Deed/Estate
Con	Contract 15% Down payment regular terms
ConLw	Contract Low Down payment and low interest
ConLI	Contract Low Interest
ConLD	Contract Low Down
Oth	Other

SaleCondition: Condition of sale

Normal	Normal Sale
Abnorml	Abnormal Sale - trade, foreclosure, short sale
AdjLand	Adjoining Land Purchase
Alloca	Allocation - two linked properties with separate deeds, typically condo with a garage unit
Family	Sale between family members
Partial	Home was not completed when last assessed (associated with New Homes)

[redacted]