SMU MSDS 6372 - Project 1
Sales Price Prediction Models
Ames, Iowa

Team Members:
Travis Deason
Jeffrey Lancon
James Vasquez

Group Project 1 – Iowa Home Predictions
Travis Deason, Jeffrey Lancon, James Vasquez
MSDS 6372 Applied Statistics: Inference and Modeling
Monday evenings 8:30pm to 10pm Central

Contents

Introduction 1
Data Description 1
Exploratory Analysis 1
- Missing Data 1
- Data Transformation 1
- Team Observation 2

Simplified Predictive Model 2
- Model Selection 2
- R-Code Model 2
- Sale Price Predictive Model: 3
- Assumption Verification 3
- Kaggle Submission 3

Advanced Predictive Model - LASSO 3
- Model Selection 3
- Data Transformation 4
- R-Code Model 4
- Coefficient Selection - LASSO 4
- Graphical Coefficient - LASSO 4
- Kaggle Submission 5

Advanced Predictive Model - RIDGE 5
- Model Selection 5
- Data Transformation 6
- R-Code Model 6
- Coefficient Selection - RIDGE 6
- Graphical Coefficient - RIDGE 6
- Kaggle Submission 7

Conclusion 7
Model Comparisons 8
Appendix A-01 9
- Variable / features not present 9
- Data set with zero values substituted values 9

Appendix A-02 10
- Observational Plots 10

Appendix B-01 11
Simplified Model Parameter Interpretations 11
Appendix B-02 13
Simplified Model Assessment Plots 13
Appendix C-01 15
Data clean up & Advanced Model 15
Appendix D-01 22
SAS Stepwise output 22

Group Project 1 – Iowa Home Predictions
Travis Deason, Jeffrey Lancon, James Vasquez
MSDS 6372 Applied Statistics: Inference and Modeling
Monday evenings 8:30pm to 10pm Central

Introduction

Realtors in Ames Iowa are interested in building a predictive model for sales prices of homes within the city, given typically collected MLS data. To accomplish this, the team investigates existing data that has been collected on homes sales between 2006 and 2010, creating two separate predictive models using multiple regression analysis methodologies.

The first model will be a 'simplified' model that can be easily interpreted and utilized by local realtors, contactors and prospective buyers, gain insight into important factors affecting housing prices, utilizing fewer parameters while still yielding an accurate prediction.

The second model will be a full model, utilizing many more predictive parameters and covariates, increasing the prediction accuracy of home sale prices within Ames. The second model is not as intuitively interpreted as the 'simplified' model, so use of this model will require more technical knowledge and requires specialized software.

The team's simplistic model though, be it the least sophisticated model does appear to predict very well with variables the team has included into the model. The adjusted R^2 value of 0.8523 is only slightly lower than the advanced model of 0.8722. Individuals/teams shall will obtain satisfactory results with either model.

Data Description

Ames, Iowa housing data set is available on the AmStat.org website. The data set contains 2,919 observations and 81 variables (23 normal, 23 ordinal, 14 discrete, 20 continuous, and unique Id). Most of the variables are typical information home buyers would seek to know about properties they were potentially interested in (e.g. Living Area, Year Built, Lot size, Neighborhood, Overall Condition, bedrooms, bathrooms, etc...). To view the data set please see AMSTAT.org_AmesHousing.xls

Exploratory Analysis

Plotting several continuous data attributes, several relationships become clear. Most data is clustered near a centroid with a large scattered tail on either the high end, the low end, or both. Additionally, most of the continuous data seems to be used to describe features that is not present in all houses. In the chart below of sales price vs wooddecksf, garagearea, and 2ndflrsf a large vertical line is present at zero (see graphs below). This value is effectively a N/A value, we will treat the presence of a feature, such as a garage or a 2nd floor, as categorical, and insert a dummy value to tell the model the feature is there. A similar place where we can use this same methodology, is with the yearremodadd variable, in that case an additional variable called 'has_remodadd' is added where the variable will have a value of TRUE whenever the year built is not equal to the year remodeled.
See Appendix A-01

Missing Data

- If the column is numeric, a category called "_isNA" is added which is filled in for all null_values
- If the column is continuous, median value is substituted for the N/A values
    - Median value is selected due to most of the columns seem to have a long tail
    - The median should better protect our model from extreme data points.
- Zero values substituted with dummy values

Data Transformation

Applying a log function to these attributes provides a higher linear model rather than not logging these attributes.

SalePrice	GrLivArea	X1stFlrSF	X2ndFlrSF	BsmtUnfSF
GarageArea	WoodDeckSF	OpenPorchSF	LotArea	BsmtFinSF1

With all variables transformed into a more palatable form, the data is now ready to be modeled.

Team Observation

- Highest predictive power attributes are categorical
- Second most predictive variable is first floor square footage (X1stFlrSF) which is a continuous value

To better understand these categorical values, the team plotted variables against saleprice. See Appendix A-02

Simplified Predictive Model

Realtors in Ames Iowa are interested in building a predictive model for sales prices of homes within the city, given typically collected MLS data. The model is constructed in a way that it can easily interpreted and utilized by local realtors, contactors and prospective buyers, to gain insight into important factors affecting housing prices.

Model Selection

The team took an informal poll to determine variables that would influences their decisions in making an offer on homes. From the poll these variables were selected:

Neighborhood	GrLivArea	OverallQual	OverallCond	YearBuilt

As many potential home buyers and real estate agents say…location, location, and location is a key driver in many potential home buyers decision making. A Log transform was performed on variable Greater Living Area (GrLivArea), to better adhere to the regression analysis assumptions and reduce variance. To simplify the model, for ease of interpretation, no covariance was assumed for the analysis.

R-Code Model

lm(saleprice ~ neighborhood + grlivarea + overallqual + overallcond + yearbuilt, data=sub_frame)

The summary overview is shown in the below screenshot taken from R

```
Residual standard error: 0.1535 on 1431 degrees of freedom
Multiple R-squared:  0.8552,    Adjusted R-squared:  0.8523
F-statistic: 301.8 on 28 and 1431 DF,  p-value: < 2.2e-16
```

With the 'Simplified' Regression model (p-value <.001), containing only (1) numeric, (3) ordinal, and (1) categorical variable, is statistically significant and is able explain 85.23% of the variation in the sale price of a home in Ames, Iowa.

	Beta	Estimate	Std. Error	t-value	Pr(>|t|)
(Intercept)	β_0	0.5907007	0.6282064	0.9402972	3.47E-01
grlivarea	B_1	0.4904695	0.0167814	29.226954	1.22E-147
overallqual	B_2	0.0865455	0.005144	16.824434	4.24E-58
overallcond	B_3	0.0580023	0.0041253	14.060252	3.71E-42
yearbuilt	B_4	0.0035441	0.000302	11.73672	1.95E-30
Neighborhood	B_5	See Chart-B-2: Appendix B-01			

Sale Price Predictive Model:

$Y_{hat} = \beta_0 + \beta_1{*}\ln(GrLivArea) + \beta_2{*}OverallQual + \beta_3{*}OverallCond + \beta_4{*}YearBuilt + Neighborhood$

Example:

ID	GrLivArea	OverallQual	OverallCond	YearBuilt	Neighborhood	Sale Price	Pred Sale Price	% Diff
1	1710	7	5	2003	CollgCr	\$208,500	\$215,954	3.58%
434	1604	6	5	1997	MeadowV	\$181,000	\$146,634	23.4%

Predicted Sale Price: ID1

Predicted SalePrice = $\exp(Y_{hat})$

$$Y_{hat} = 0.5907 + 0.4905{*}\ln(1710) + 0.0865{*}7 + 0.0580{*}5 + 0.00354{*}2003 + 0.0546$$

Predicted SalePrice = exp(12.283) = \$215,954.70

Parameter Interpretation (Simplified Model): See Table B-1: Appendix B-01

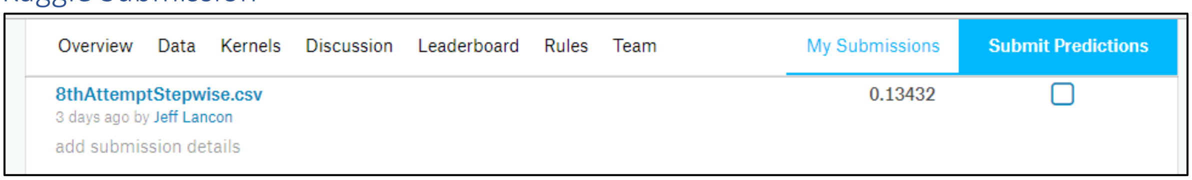
Assumption Verification

Outliers: Three observations were considered outliers (31, 525, and 1299) See Figure B-4 & Table B-6 (Appendix B-02). While preliminary investigation into these values did reveal some extreme values, too little information was available to fully analyze the observations, so they were included in the model.

Residuals: Residuals are normally and uniformly distributed See Figure B-5 (Appendix B-02), indicating that a linear model will produce a valid predicted output.

Additional descriptive graphics: Q-Q, Leverage Plots are included in Appendix B-02

Kaggle Submission



See Appendix D-01 for SAS output

Advanced Predictive Model - LASSO

Realtors in Ames Iowa are interested in building a predictive model for sales prices of homes within the city, given typically collected MLS data. The Advanced Predictive model takes into account many additional variables and covariates that the simplified model does not consider. This model is used to predict the most accurate sale price of a house in the data set.

Model Selection

The team utilized the glmnet library with a min lambda ratio of .00005 and 2500 different lambda iterations. Applying max in sample accuracy as the stopping criteria increased the model performance.

Data Transformation

The same data transformation techniques were used on the 'Advanced' model as the 'Simplified' model.

R-Code Model

See Appendix C-01

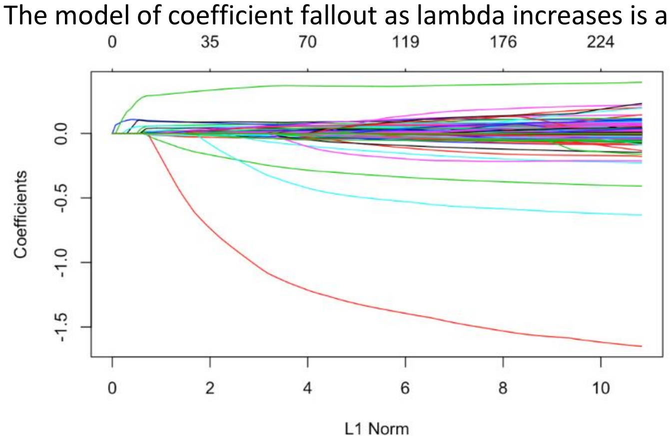
Coefficient Selection - LASSO

Without cross validating, the model had a training set average MSE of .00085 on the log scale data. On the 1500th lambda iteration.

Top 25 Coefficients					
Intercept	2.36883747	functional_Typ	0.05844194	heatingqc_Ex	0.0229939
neighborhood_StoneBr	0.07971896	exterior1sr_BrkFace	0.04729371	neighborhood_Crawfo	0.08309497
overallqual	0.05990326	centralair_Y	0.03530952	kitchenqual_Ex	0.06080678
roofmatl_WdShngl	0.04910594	neighborhood_Somerst	0.0241038	garagecars	0.05131706
bsmtexposure_Gd	0.03877982	x1stflrsf	0.08970574	saletype_New	0.04113073
bsmtfullbath	0.02569472	neighborhood_NoRidge	0.06273614	condition1_Norm	0.02681592
fireplaces	0.01989025	bsmtqual_Ex	0.05384154	garagequal_Ex	0.02087361
grlivarea	0.36698249	neighborhood_NridgHt	0.04613688		
lotarea	0.07680804	overallcond	0.03180884		

Graphical Coefficient - LASSO

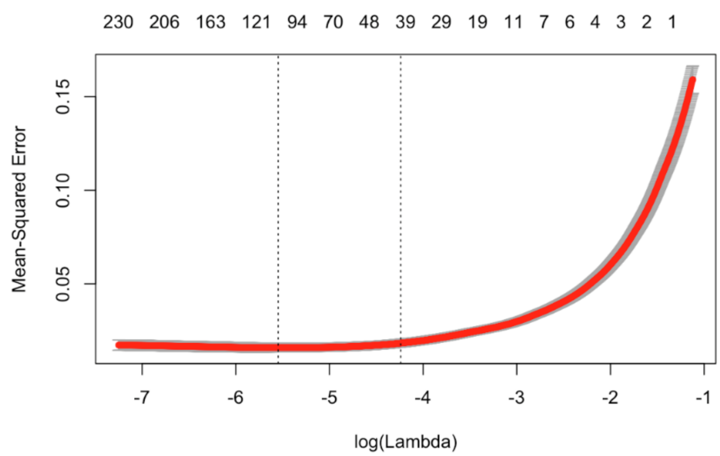
The model of coefficient fallout as lambda increases is also shown below:



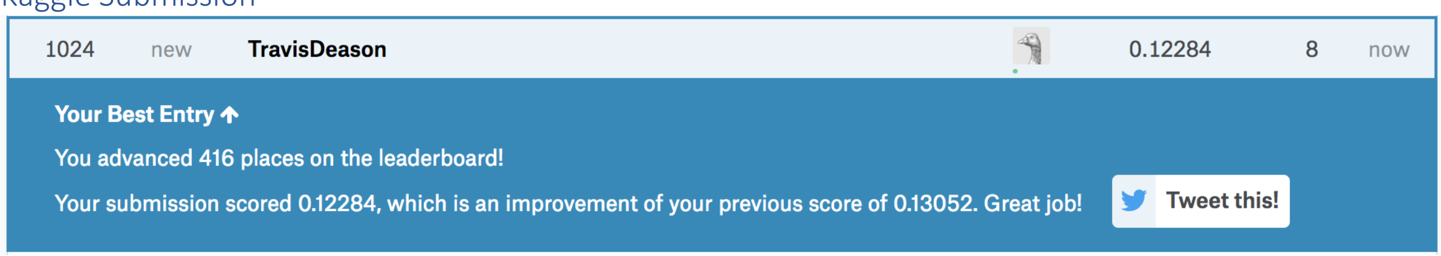
Using a tenfold cross validated version of the same glmnet model as shown (where cross validation was used as the optimization criteria), the team obtained a MSE on the test set of .0371. Overall the coefficient set used is similar in the cross-validated model.

Cross-Validated Model (Parameter Estimates)					
Intercept	2.44267383	poolqc_Ex	0.07498096	overallcond	0.03351619
neighborhood_Crawfor	0.10091946	neighborhood_NridgHt	0.06034027	neighborhood_StoneBr	0.10436945
neighborhood_NoRidge	0.07516175	garagecars	0.0488819	roofmatl_WdShngl	0.07797323
functional_Typ	0.06169893	centralair_Y	0.03458144	kitchenequal_Ex	0.06694208
overallqual	0.05305656	condition2_PosA	0.11318847	bsmtqual_Ex	0.05480261
foundation_Stone	0.03563052	garageequal_Ex	0.07972055	saletype_New	0.03897751
condition1_Norm	0.02916543	lotarea	0.07497926	neighborhood_Somerst	0.0325333
grlivarea	0.36455274	exterior1st_BrkFace	0.05935236		
x1stflrsf	0.09100952	bsmtexposure_Gd	0.04251501		

The MSE/Lambda curve shows that 2500 iterations may have been slightly overkill with the curve starting to rebound at ln(lambda) = -6



Kaggle Submission



Advanced Predictive Model - RIDGE

Realtors in Ames Iowa are interested in building a predictive model for sales prices of homes within the city, given typically collected MLS data. The Advanced Predictive model takes into account many additional variables and covariates that the simplified model does not consider. This model is used to predict the most accurate sale price of a house in the data set by using all variables available.

Model Selection

The team utilized the glmnet library with a min lambda ratio of .00005 and 2,500 different lambda iterations. Applying max in sample accuracy as the stopping criteria increased the model performance.

Data Transformation

The same data transformation techniques were used on the 'Advanced' model as the 'Simplified' model.

R-Code Model

See Appendix C-01

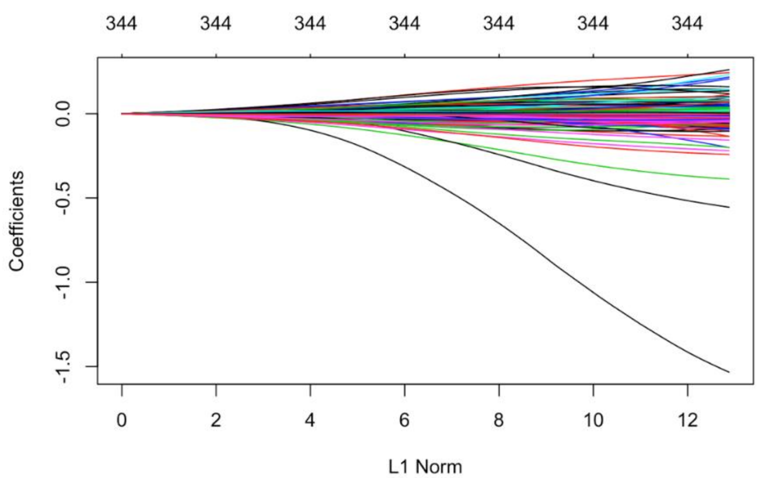
Coefficient Selection - RIDGE

Without cross validating, the model had a training set average MSE of .036 on the log scale data. On the 2300th lambda iteration.

Top 25 Coefficients (RIDGE)					
Intercept	1.92769E+01	poolqc_Ex	1.64036E-03	grlivarea	1.54467E-03
x1stflrsf	1.35381E-03	exterqual_Ex	1.35309E-03	roofmatl_WdShngl	1.32379E-03
condition2_PosA	1.20423E-03	neighborhood_NoRidge	1.18319E-03	bsmtqual_Ex	1.18301E-03
fireplacequ_Ex	1.16041E-03	kitchenqual_Ex	1.15516E-03	exterior2nd_Other	1.14178E-03
neighborhood_NridgHt	1.10524E-03	neighborhood_StoneBr	1.00864E-03	centralair_Y	9.93624E-04
garageyrblt_isNA_FALSE	9.80044E-04	garagecond_TA	8.80272E-04	saletype_New	8.30008E-04
saletype_Con	8.13522E-04	salecondition_Partial	8.08978E-04	condition2_PosN	8.04264E-04
exterior1st_ImStucc	8.00727E-04	exterior1st_Stone	7.71793E-04	masvnrtype_Stone	7.70207E-04
paveddrive_y	7.58371E-04				

Graphical Coefficient - RIDGE

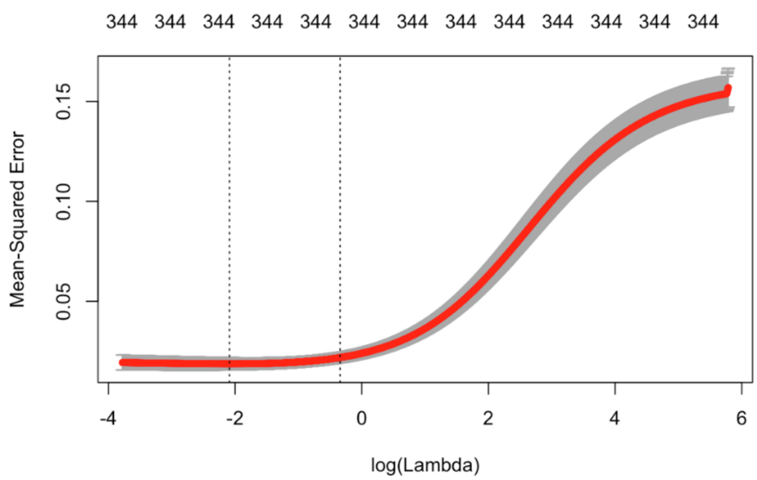
The model of coefficient fallout as lambda increases is also shown below:



Using a 10 fold cross validated version of the same glmnet model as shown (where cross validation was used as the optimization criteria). The coefficients utilized in the cross validated model were notably different than the standard Ridge model

Cross-Validated Model (Parameter Estimates)					
Intercept	10.31696883	neighborhood_StoneBr	0.12752855	exterior1st_BrkFace	0.07189627
poolqc_Ex	0.1520473	saletype_Con	0.08401221	lotarea	0.054783
roofmatl_Membran	0.1289378	heating_GasW	0.07340327	roofmatl_WdShngl	0.16597152
neighborhood_Crawfor	0.09327937	housestyle_2.5Unf	0.05571738	roofstyle_Shed	0.13483298
saletype_ConLD	0.07359137	garageequal_Ex	0.17237228	neighborhood_NoRidge	0.09429034
saletype_CWD	0.05734064	x1stflrsf	0.13598925	neighborhood_NridgHt	0.07373735
roofmatl_Metal	0.05103053	utilities_AllPub	0.11634679	kitchenqual_Ex	0.06892711
condition2 PosA	0.20939363	extercond_Ex	0.07416425	bsmtqual_Ex	0.05381502
grlivarea	0.15101724				

The MSE/Lambda curve shows that Ridge regression required much more iterations to converge on a more reliable model then LASSO, however the min MSE with a cross validated LASSO was .079 which was significantly higher than the LASSO model.



Kaggle Submission

Conclusion

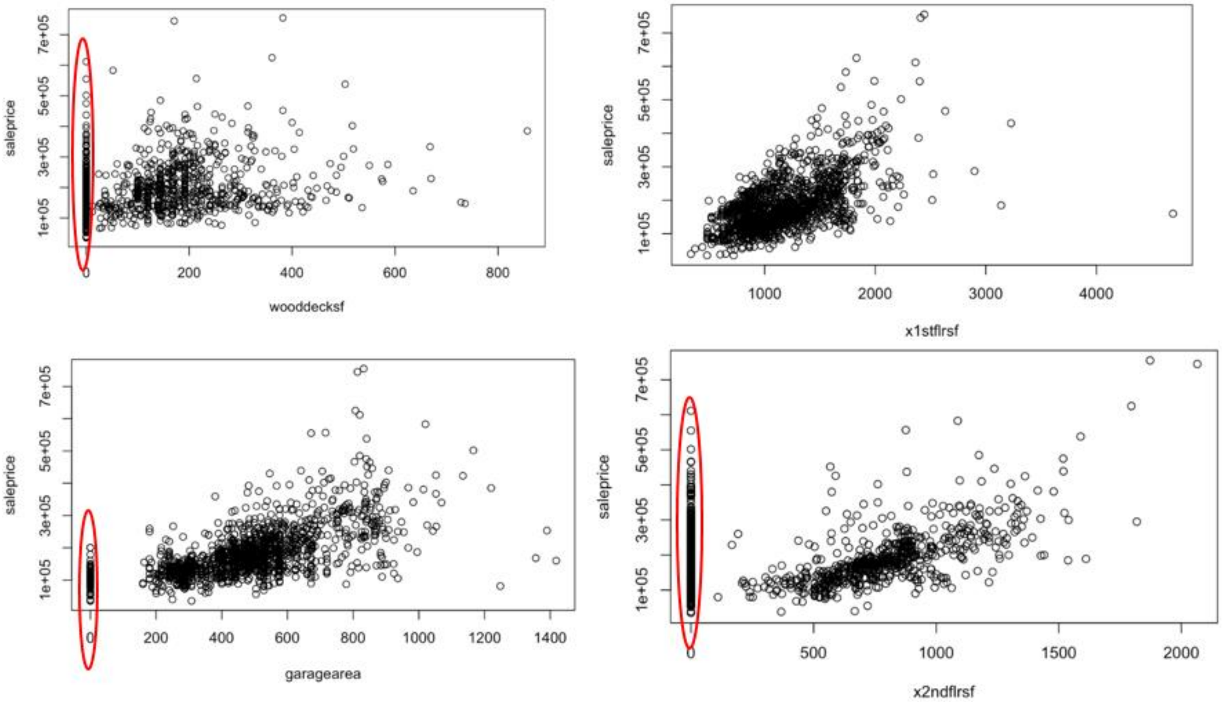
The team was tasked to create three different predictive models using data that collected in Ames Iowa from 2006 to 2010. The first model is a simplistic model using a stepwise analysis, while the second and third model are considered the most predictive models using a lasso and ridge analysis. The three different models assessed and created provide very similar results, however it is the teams' decision that the simplistic model delivers the best results for the effort. Though the two advanced models do provide a slightly higher correlation the amount of effort and explanation to teams that can use the data model would not yield additional value. Limiting the prediction variables to known drivers (neighborhood, general living area, etc…) the team has concluded that a simplified model is best used to predict and describe the sales price of homes in Ames Iowa.

Model Comparisons

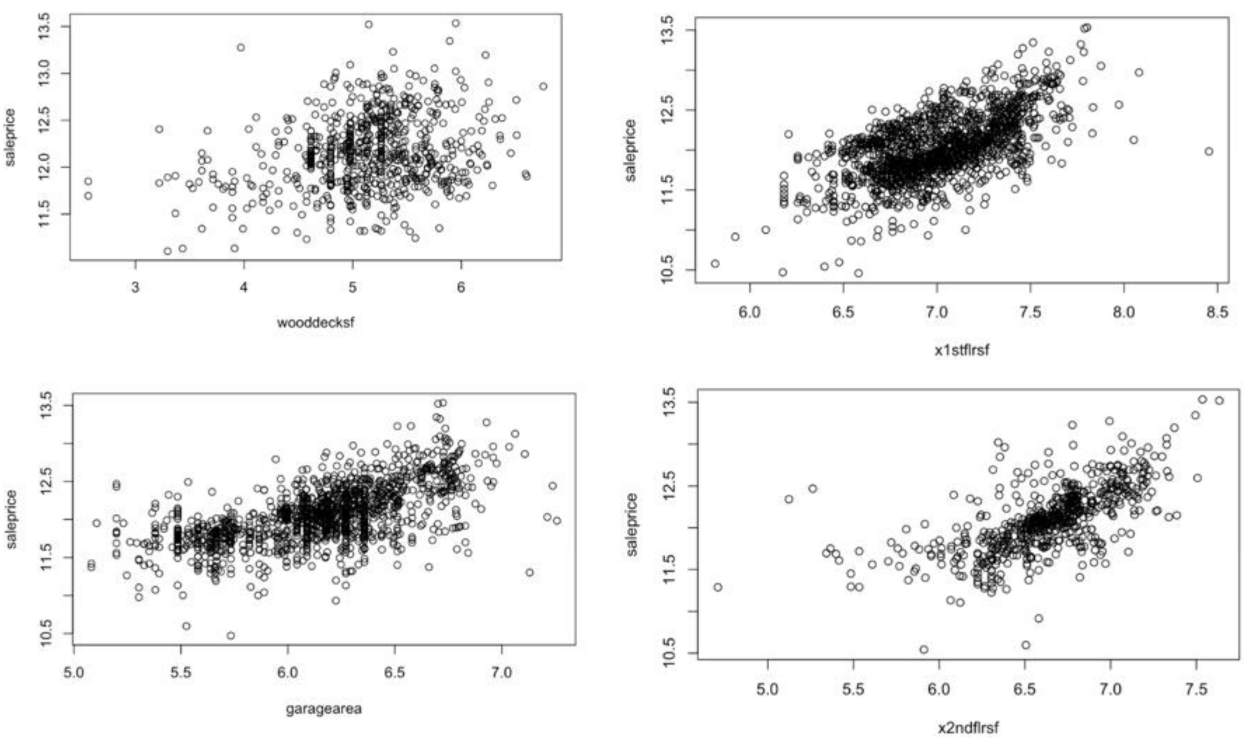
Kaggle Scoring Model	Adjusted R2	AIC	Kaggle Score	Selection Method
Model 1	0.9355605	0.041265	0.12284	LASSO
Model 2	0.9341201	6.545977	0.1318	RIDGE
Model 3	0.9066	-1763	0.13432	STEPWISE (SAS)

Appendix A-01

Variable / features not present

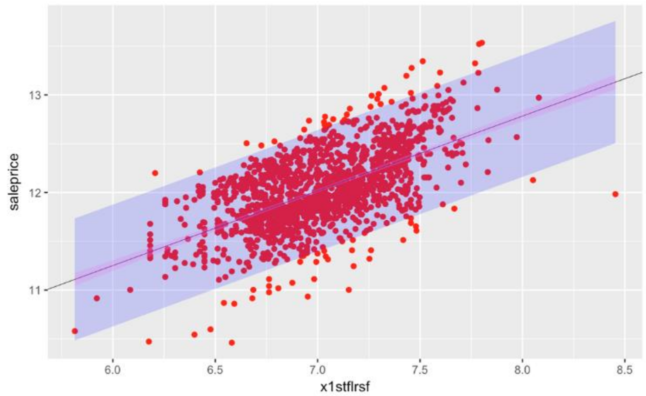


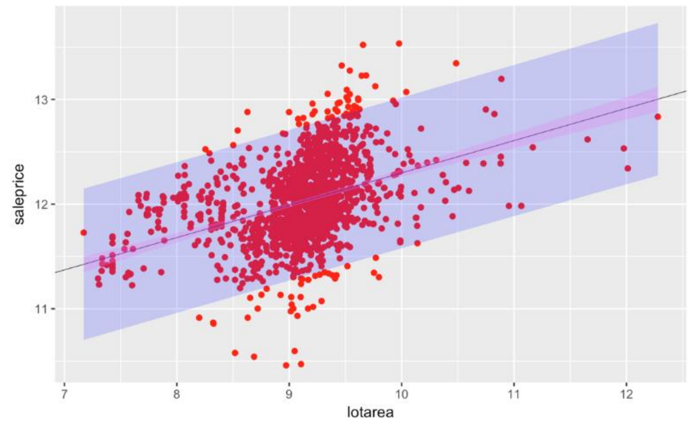
Data set with zero values substituted values

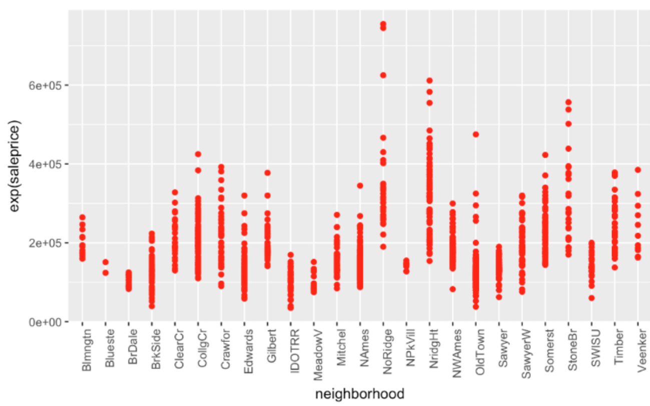


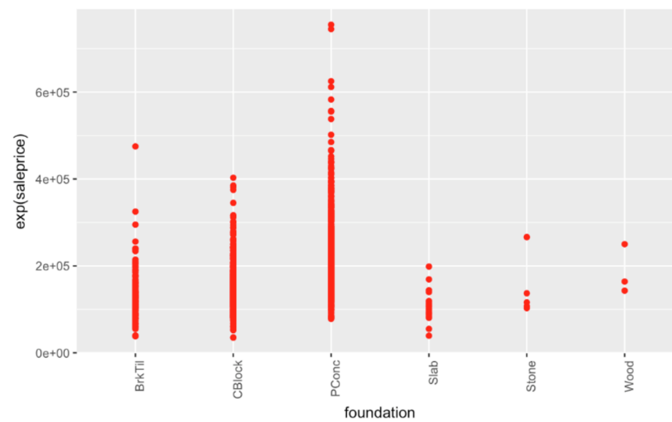
Appendix A-02

Observational Plots









Appendix B-01

Simplified Model Parameter Interpretations

Table B-1 Parameter Interpretation	
β_0:	The intercept in this model provides a median home SalePrice estimate of $e^{0.5907}$ \$1.80, with all other parameters being 0. This of course is extrapolation and does not have a clear practical meaning. A 95% Confidence interval is between $e^{1.8220}$ \$6.18 and $e^{-0.6406}$ \$-1.90. Note: Intercept parameter is not statistically significant.
β_1:	While keeping all other parameters fixed, a doubling of greater living area (GrLivArea) is associated with a 24.05% ($2^{0.4905}$) multiplicative increase in the median Sale Price. A 95% Confidence interval is between 20.93% ($2^{0.45758}$) and 27.4% ($2^{0.52336}$).
β_2:	While keeping all other parameters fixed, a one unit increase in overall quality (OverallQual) is associated with a 9.0% ($e^{0.0865}$) multiplicative increase in median home SalePrice. A 95% Confidence interval is between7.9% ($e^{0.0765}$) and 10.1% ($e^{0.0966}$).
β_3:	While keeping all other parameters fixed, a one unit increase in overall condition (OverallCond) is associated with a 6.0% ($e^{0.0580}$) multiplicative increase of in median home SalePrice. A 95% Confidence interval is between 5.1% ($e^{0.04992}$) and 6.9% ($e^{0.06609}$).
β_4:	While keeping all other parameters fixed, a one unit change in year built (YearBuilt) is associated with a 0.355% ($e^{0.003544}$) multiplicative increase of in median home SalePrice. A 95% Confidence interval is between 0.296% ($e^{0.002952}$) and 0.415% ($e^{0.0041360}$). (See Note Below)
β_5:	Neighborhood parameter is a categorical parameter, with multiple values, so a generalization is in order. Neighborhood differences are associated with maximum 26.8% ($e^{0.2379}$) multiplicative increase to a decrease of 25.0% ($e^{-0.2873}$). A 95% Confidence interval is between 39.1% ($e^{0.3307}$) and 34.4% ($e^{-0.4218}$). See Chart A-1 in Appendix A for a list of parameters for each individual neighborhood.

Note: β_4(YearBuilt) parameter should have been normalized in this model. To keep interpretation simple and improve usability of the prediction equation, the YearBuilt parameter was left as-is.

Chart B-2						
NeighborHood Parameter β5 Chart						
	Estimate	**Std. Error**	**t-value**	**Pr(>|t|)**	**95% CI**	
neighborhoodBlueste	-0.196180318	0.115088105	-1.70460986	8.85E-02	0.0293924	-0.421753
neighborhoodBrDale	-0.287291888	0.054490618	-5.27231836	1.55E-07	-0.1804903	-0.3940935
neighborhoodBrkSide	-0.000657548	0.047467427	-0.01385262	9.89E-01	0.0923786	-0.09369371
neighborhoodClearCr	0.18282426	0.048872816	3.74081701	1.91E-04	0.278615	0.08703354
neighborhoodCollgCr	0.054623925	0.039407111	1.38614384	1.66E-01	0.1318619	-0.02261401
neighborhoodCrawfor	0.158120561	0.04691866	3.37009966	7.71E-04	0.2500811	0.06615999
neighborhoodEdwards	-0.063749348	0.043037988	-1.48123437	1.39E-01	0.0206051	-0.1481038
neighborhoodGilbert	-0.005219287	0.041303749	-0.12636352	8.99E-01	0.0757361	-0.08617464
neighborhoodIDOTRR	-0.147109718	0.05032497	-2.92319536	3.52E-03	-0.0484728	-0.24574666
neighborhoodMeadowV	-0.193393441	0.054329084	-3.55966689	3.83E-04	-0.0869084	-0.29987845
neighborhoodMitchel	0.029503546	0.044063607	0.66956719	5.03E-01	0.1158682	-0.05686112
neighborhoodNAmes	0.030912594	0.041034618	0.75332965	4.51E-01	0.1113404	-0.04951526
neighborhoodNoRidge	0.196595859	0.04505293	4.36366424	1.37E-05	0.2848996	0.10829212
neighborhoodNPkVill	-0.061395317	0.063892523	-0.96091552	3.37E-01	0.063834	-0.18662466
neighborhoodNridgHt	0.220006506	0.041433811	5.30983038	1.27E-07	0.3012168	0.13879624
neighborhoodNWAmes	0.002215607	0.04253327	0.05209115	9.58E-01	0.0855808	-0.0811496
neighborhoodOldTown	-0.083689858	0.046404493	-1.80348611	7.15E-02	0.0072629	-0.17464266
neighborhoodSawyer	0.022263028	0.043505613	0.51172772	6.09E-01	0.107534	-0.06300797
neighborhoodSawyerW	0.013980283	0.042680498	0.3275567	7.43E-01	0.0976341	-0.06967349
neighborhoodSomerst	0.062888122	0.040765882	1.54266555	1.23E-01	0.1427893	-0.01701301
neighborhoodStoneBr	0.237923374	0.048542176	4.90137433	1.06E-06	0.333066	0.14278071
neighborhoodSWISU	-0.032790956	0.05357335	-0.6120759	5.41E-01	0.0722128	-0.13779472
neighborhoodTimber	0.141278389	0.04501138	3.138726	1.73E-03	0.2295007	0.05305608
neighborhoodVeenker	0.1904469	0.059930963	3.17777141	1.52E-03	0.3079116	0.07298221
Maximum	0.237923374			**Maximum**	0.333066	
Min	-0.287291888			**Minimum**		-0.421753

Appendix B-02

Simplified Model Assessment Plots

Table – B1 *VIF (Simplistic Model)*

Variable	GVIF	Df	GVIF^(1/(2*Df))
Neighborhood	8.171933	24	1.044736
Grlivarea	1.937459	1	1.391926
Overallqual	3.134352	1	1.770410
Overallcond	1.305075	1	1.142399
Yearbuilt	5.151294	1	2.269646

Figure - B1 Log(SalePrice) vs Log(GrLivArea)

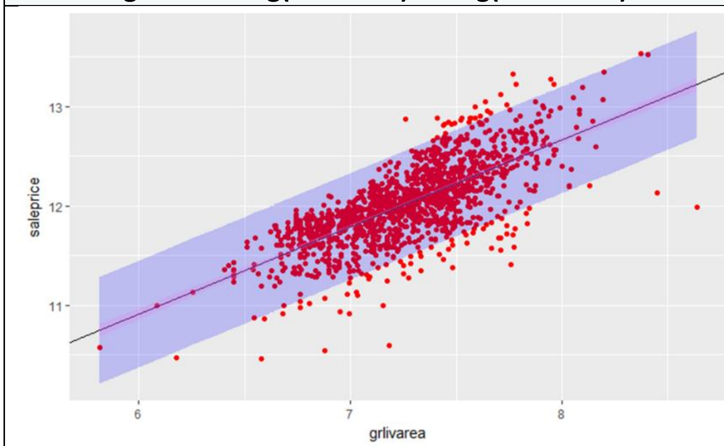


Figure - B2 (Q-Q Plot Studentized Residuals)

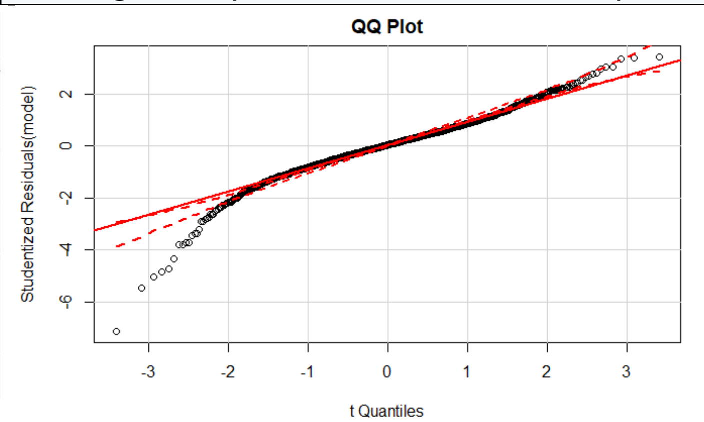


Figure – B3 (Leverage Plots)

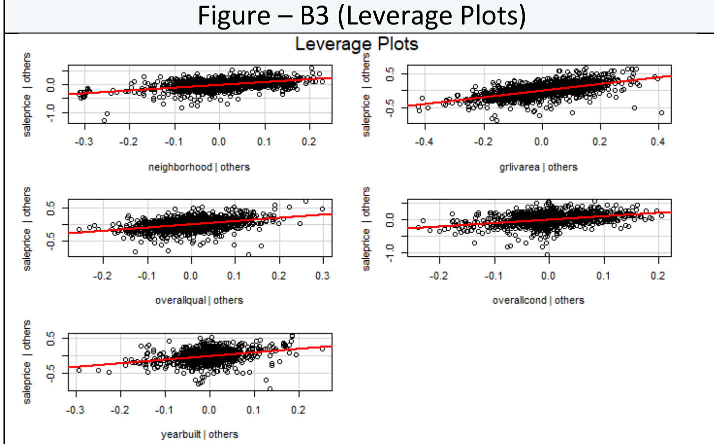


Figure – B4 (CooksD Plot)

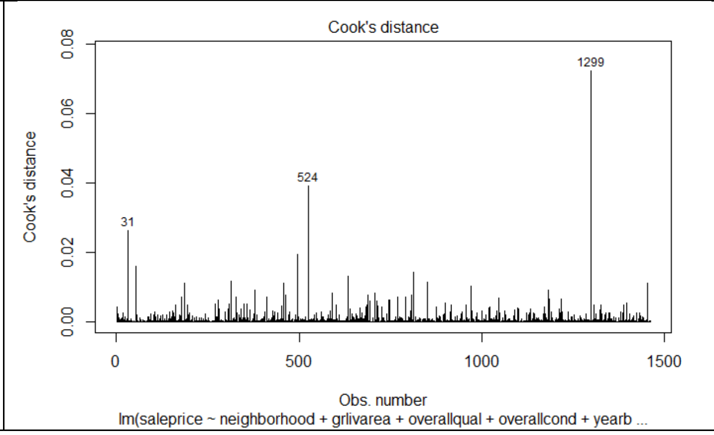


Figure – B5 (Residuals Plot)

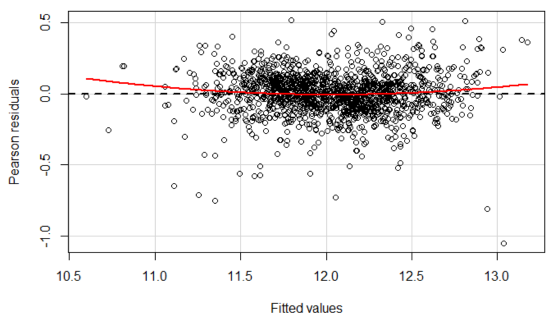


Table – B6 (Outliers Test)

Obser.	RStudent	UnAdjust. p-Value	Bonferroni p-value
1299	-7.125175	1.6425e-1	2.3981e-09
524	-5.455135	5.7591e-08	8.4083e-05
31	-5.024356	5.6881e-07	8.3046e-04
633	-4.846374	1.3944e-06	2.0358e-03
969	-4.715909	2.6413e-06	3.8563e-03
496	-4.336742	1.5474e-05	2.2592e-02

Appendix C-01

Data clean up & Advanced Model

```
library(Hmisc)
library(tidyr)
library(dplyr)
library(stringr)

find_percent <- function(df, label_col, num_obs, sep='&'){
  ##find the ratio of a certian value which includes the label
  ##--------
  ##INPUTS
  ##df: data.frame
  ##    - dataframe with all catagorical data
  ##label_col
  ##    -   binary column of interest in df
  ##num_obs: named.vector
  ##     -   contains all possible values in df with the  correlated number of observations
  ## --------
  ## RETURNS
  ## percent_pos: named.vector
  ##     -   contains the percentage of each value within the dataframe which is associated with the
label_column.
  percent_pos <- c()
  sub <- df[,label_col] == TRUE
  sub_df <- df[sub,]
  for(col_val in names(num_obs)){
    colval <- unlist(strsplit(col_val, sep))
    percent_pos[col_val] = ((sum(sub_df[,colval[1]] == colval[2]) + .0001))  / (num_obs[col_val] + .0001)
    }
  return(percent_pos)
  }
find_number_observations <- function(df, sep='&', check_na=FALSE){
  num_obs= c()
  for (col in names(data_binned)){
    if(check_na){
      num_obs[paste(col, 'isna', sep=sep)] = sum(is.na(data[,col])) / dim(df)[1]
      }
    for (value in unique(data_binned[,col]))
      {
      num_obs[paste(col, value, sep=sep)] = sum(data_binned[,col] == value)
      }
  }
```

```
  return(num_obs)
}
make_dummy <- function(df, sep='&', cat_but_keep=c(), known_cats=c(), drop_others=FALSE){
  # takes original dataframe and converts all values with less then 8 unique sets, and
  # non-numeric data to dummy variables
  # --------
  # INPUTS
  # df: data.frame
  #    -  Data should all be catagorical.
  # sep: str
  #    -  Character to use as seperator between column and value
  # subcols: bool or array
  #  -  subset of all columns which contain columns to ignore
  # --------
  # RETURNS
  # dfo: data.frame
  #    -  all data is bool
  dfo <- df
  #for(col in names(select(df,-one_of(cat_but_keep)))){
  for(col in names(df)){
    if( col %in% known_cats |
      is(df[,col])[1] == 'factor' |
      length(unique(dfo[,col])) < 4){
      print(col)
      vals <- unique(dfo[,col])#[-1]
      for(val in vals){#[1:length(vals)+1]){
        dfo[, paste(col, val, sep=sep)] = dfo[, col] == val
      }
      dfo <- dfo[, names(dfo) != col]
    }
  }
  return(dfo)
}
fill_nulls <- function(df, null_sep='_isNA'){
  for(col in names(df)){
    nans <- is.na(df[,col])
    if(sum(nans) > 0){
      if('factor' %in% is(df[,col])){
        ncol <- as.character(addNA(df[,col]))
        ncol[nans] <- null_sep
        df[col] <- factor(ncol)
      }
```

```
        else{
          print(col)
          df[nans, col] <- median(df[!nans, col])
          df[, paste(col, null_sep, sep='')] <- nans
        }
      }
  }
  return(df)
}
find_covariance <- function(df, items, sep='+'){
  # Take a subset of columns in df and find the covariance between them
  # ---------
  # INPUTS
  # df: data.frame
  #    -  All data inputs must of type bool
  # items: data.frame
  #    -  columns in the dataframe to compare to each other
  # sep: str
  #    -  character to use to seperate columns being compared
  # --------
  # RETURNS
  # covar: named vector
  #    -  sets of column names seperated by sep with duplicates and self correalations removed
  covar = c()
  for(col1 in items){
    for(col2 in items){
      if(col1 != col2){
        covar[paste(col1, col2, sep='+')] = (sum(df[,col1] == TRUE & df[,col2] == TRUE)) / (max(c(sum(df[,col1]
== TRUE), sum(df[,col2] == TRUE))) + .00001)
        }
      }
    items = items[items != col1]
    }
    covar <- covar[order(-covar)]
    return(covar[c(TRUE, FALSE)])
  }

bin_columns <- function(data, min_size=100, num_splits=6){
  # Convert continous data into discrete catagorical data
  # by splitting continous data into equal sized (by number of members) groups.
  # --------
  # data: data.frame
```

```
    #     -   data frame which contains continous data
    # min_size: integer
    #     -   min number of members in a group (determine split pts). columns with less discrete points then this value will be ignored.
    # num_splits
    #     -   number of times to split continous dataset
    # --------
    types <- sapply(data, class)
    data_binned <- data
    for(col in names(types)){
        if(types[[col]] == 'integer' & length(unique(data[,col])) > num_splits){
            data_binned[col] <- cut2(data[,col], m=min_size, g=num_splits)
            data_binned[,col] = sapply(data_binned[,col], toString)
            }
        else{
            data_binned[,col] = sapply(data[,col], toString)}
        }
    names(data_binned) <- sapply(names(data_binned), str_trim)
    return(data_binned)
    }
make_balanced_df <- function(df, label){
    train_1 <- sample(c(TRUE, FALSE), dim(df)[1], replace=TRUE, prob=c(.8, .2))
    tdf <- subset(df, label != 0)
    sub_data <- subset(df, label == 0)
    df_bal <-rbind(tdf, sample_n(sub_data, dim(tdf)[1], replace=FALSE))
    return(df_bal <-rbind(tdf, sample_n(sub_data, dim(tdf)[1], replace=FALSE)))
}
train_bool_arrays <- function(df, label, test_percent=.2, num_frames=5, rand_seed=42){
    tp <- test_percent
    set.seed(rand_seed)
    df_bal <- make_balanced_df(df, label)
    df_bal['train'] = sample(c(TRUE, FALSE), dim(df_bal)[1], replace=TRUE, prob=c(1-tp, tp))
return(df_bal)
}
check_label_corelation <- function(df, label, dsep='&', sd_ratio=1){
## function to generate top contributing variables to a specific label
##--------
##INPUTS
##df: data.frame
##  -   contains all catagorical variables, label col must be T/F
##label: string
##  -   name of label column
```

```
##sep: str
## -   character to use in seperating dummy values from col name
##--------
##RETURNS
## coors: named_vector
## -   contains coorelation rate for each value in the df
  all_pos <- sum(df[,label] == TRUE) / dim(df)[1]
  num_obs <- find_number_observations(df, sep=dsep)
  percent_pos <- find_percent(df, label, num_obs, sep=dsep)
  label_frame <- data.frame(percent_pos, num_obs)
  label_frame[,'ratio_delta'] <- label_frame$percent_pos - all_pos
  not_label <- (rownames(label_frame) != paste(label, 'TRUE', sep=dsep) & rownames(label_frame) !=
paste(label, 'FALSE', sep=dsep))
  label_frame <- label_frame[not_label,]
  one_dev <- sd(label_frame[,'ratio_delta']) * sd_ratio
  label_infl <- label_frame[abs(label_frame[,'ratio_delta']) > one_dev,]
  return(label_infl[order(-label_infl$ratio_delta),])
}
but_I_regress <-function(df, model, label, thres=50, dsep='&'){
  ##
  ##
  ##
  ##
  df.te <- df[df[,'train'] == FALSE,]
  df.te[,'predicted'] <- predict(fit, df.te[,names(dfd) != label])
  df.te['posi'] <- df.te['predicted'] > thres
  df.te['correct'] <- df.te['posi'] == df.te[label]
  return(df.te)
  #error = sum((dfd[, label] - dfd[,'predicted'])**2)
  # sqrt(error / dim(dfd[,names(dfd) != label])[1])
}
featurize_frame <- function(df, label, csep='&'){
  snam <- names(df)
  no_labs <- df[,snam[(snam != label & snam != 'train')]]
  types <- sapply(no_labs, class)
  cat_cols <- names(types)
  idx = 1
  for(col in names(types)){
    if(types[[col]] == 'integer'){
      cat_cols = cat_cols[cat_cols != col]
      df[,col] = df[,col] / max(df[,col])
      }
```

```
    }
  dfd <- make_dummy(df, subcols=cat_cols)
  #dfd[,label] <- df[,label]
  #dfd[,'train'] <- df[,'train']
  return(dfd)
  }
gen_train_frame <- function(df, label, ptest=.2){
  # df[,label]  <- df[,label] == TRUE
  df[,label]  <- df[,label] * 100
  tdf <- df[df[,label] != 0,]
  sub_data <- df[df[,label] == 0,]
  df_bal <- rbind(tdf, sample_n(sub_data, dim(tdf)[1], replace=FALSE))
  df_bal$train <- sample(c(TRUE, FALSE),
                dim(df_bal)[1],
                replace=TRUE,
                prob=c(1-ptest, ptest))
  return(df_bal)
}
subset_use_cols <- function(df, train_frame, label, min_qt=.75, csep= '&'){
  infl <- check_label_corelation(data_binned, label, '&')
  infl$percent_sq <- infl$percent_pos ** 2
  min_inf <- quantile(infl$percent_sq, min_qt)
  alls <- subset(infl, percent_sq >= min_inf)
  tups <- str_split(row.names(alls), pattern=csep)
  first_cell <- function(x){x[1]}
  use_cols <- unique(sapply(tups, first_cell))
  return(train_frame[,append(use_cols, c(label, 'train'))])
}
Attrition_prop_table <- function(variable_name, data.f){
  # Generates a table containing proportion of responses for both Attrition values. This should allow us to
examine values in the context of whether they attrified.

  # Generate a table containing variable/Attrition rates
  prop <-prop.table(xtabs(as.formula(paste( '~ ',paste(variable_name, 'attrition ', sep = ' + '))) , data=data.f))

  #Normalize each column to sum to 1.
  prop.app <-apply(prop,2,sum)
  return(melt(sweep(prop, MARGIN=2,prop.app,'/')))
}
plot_ci <- function(x, y, data){
  model <- lm(y~x, data)
  pred.int =  predict(model, interval="prediction")
```

```
  conf.int =  predict(model, interval="confidence")
  sub_frame$pred.lower <- pred.int[,2]
  sub_frame$pred.upper <- pred.int[,3]
  sub_frame$ci.upper <- conf.int[,2]
  sub_frame$ci.lower <- conf.int[,3]

  slope <- model$coefficients[2]
  intercept <- model$coefficients[1]

  plt <-ggplot(data=sub_frame, aes(x=x, y=y, main=paste(paste('Scatterplot of', x), paste('footage vs', y)))) +
    geom_point(color= 'red') +
    geom_abline(intercept=intercept, slope=slope, color='black', size=.2) +
    geom_ribbon(data=sub_frame, aes(ymin= pred.lower, ymax= pred.upper), fill = "blue", alpha = 0.2) +
    geom_ribbon(data=sub_frame, aes(ymin= ci.lower, ymax= ci.upper), fill = "violet", alpha = 0.3)
  return(plt)
}
```

Appendix D-01

SAS Stepwise output

Model Regression For Stepwise (no Catagorical)

The REG Procedure
Model: MODEL1
Dependent Variable: LogSalePrice

Number of Observations Read	1457
Number of Observations Used	1457

Analysis of Variance					
Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	38	211.41359	5.56352	372.89	<.0001
Error	1418	21.15643	0.01492		
Corrected Total	1456	232.57002			

Root MSE	0.12215	R-Square	0.9090
Dependent Mean	12.02369	Adj R-Sq	0.9066
Coeff Var	1.01589		

Residual Statistics	
Observations	1457
Minimum	-0.783
Mean	16E-15
Maximum	0.4281
Std Dev	0.1205
Fit Statistics	
Objective	-1765
AIC	-1763
AICC	-1763
BIC	-1758