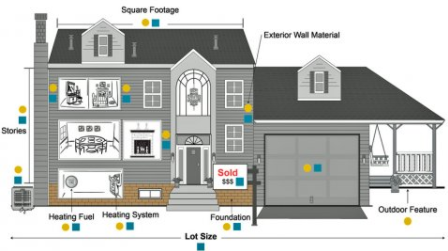
SMU MSDS 6372 - Project 1

Sales Price Prediction Models

Ames, Iowa



Team Members:

Travis Deason

Jeffrey Lancon

James Vasquez

Contents

[Introduction 1](#_Toc506733071)

[Data Description 1](#_Toc506733072)

[Exploratory Analysis 1](#_Toc506733073)

[Missing Data 1](#_Toc506733074)

[Data Transformation 1](#_Toc506733075)

[Team Observation 2](#_Toc506733076)

[Simplified Predictive Model 2](#_Toc506733077)

[Model Selection 2](#_Toc506733078)

[R-Code Model 2](#_Toc506733079)

[Sale Price Predictive Model: 3](#_Toc506733080)

[Assumption Verification 3](#_Toc506733081)

[Kaggle Submission 3](#_Toc506733082)

[Advanced Predictive Model - LASSO 3](#_Toc506733083)

[Model Selection 3](#_Toc506733084)

[Data Transformation 3](#_Toc506733085)

[R-Code Model 4](#_Toc506733086)

[Coefficient Selection - LASSO 4](#_Toc506733087)

[Graphical Coefficient - LASSO 4](#_Toc506733088)

[Kaggle Submission 5](#_Toc506733089)

[Advanced Predictive Model - RIDGE 5](#_Toc506733090)

[Model Selection 5](#_Toc506733091)

[Data Transformation 6](#_Toc506733092)

[R-Code Model 6](#_Toc506733093)

[Coefficient Selection - RIDGE 6](#_Toc506733094)

[Graphical Coefficient - RIDGE 6](#_Toc506733095)

[Kaggle Submission 7](#_Toc506733096)

[Conclusion 7](#_Toc506733097)

[Model Comparisons 8](#_Toc506733098)

[Appendix A-01 9](#_Toc506733099)

[Variable / features not present 9](#_Toc506733100)

[Data set with zero values substituted values 9](#_Toc506733101)

[Appendix A-02 10](#_Toc506733102)

[Observational Plots 10](#_Toc506733103)

[Appendix B-01 11](#_Toc506733104)

[Simplified Model Parameter Interpretations 11](#_Toc506733105)

[Appendix B-02 13](#_Toc506733106)

[Simplified Model Assessment Plots 13](#_Toc506733107)

[Appendix C-01 15](#_Toc506733108)

[Data clean up & Advanced Model 15](#_Toc506733109)

[Appendix D-01 22](#_Toc506733110)

[SAS Stepwise output 22](#_Toc506733111)

# 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 R2 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](http://www.amstat.org/publications/jse/v19n3/decock/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



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)** | β0 | 0.5907007 | 0.6282064 | 0.9402972 | 3.47E-01 |
| **grlivarea** | Β1 | 0.4904695 | 0.0167814 | 29.226954 | 1.22E-147 |
| **overallqual** | Β2 | 0.0865455 | 0.005144 | 16.824434 | 4.24E-58 |
| **overallcond** | Β3 | 0.0580023 | 0.0041253 | 14.060252 | 3.71E-42 |
| **yearbuilt** | Β4 | 0.0035441 | 0.000302 | 11.73672 | 1.95E-30 |
| **Neighborhood** | Β5 | See Chart-B-2: Appendix B-01 |  |  |  |

## Sale Price Predictive Model:

Yhat = β0 + β1\*ln(GrLivArea) + β2\*OverallQual + β3\*OverallCond + β4\*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(Yhat)

Yhat = 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

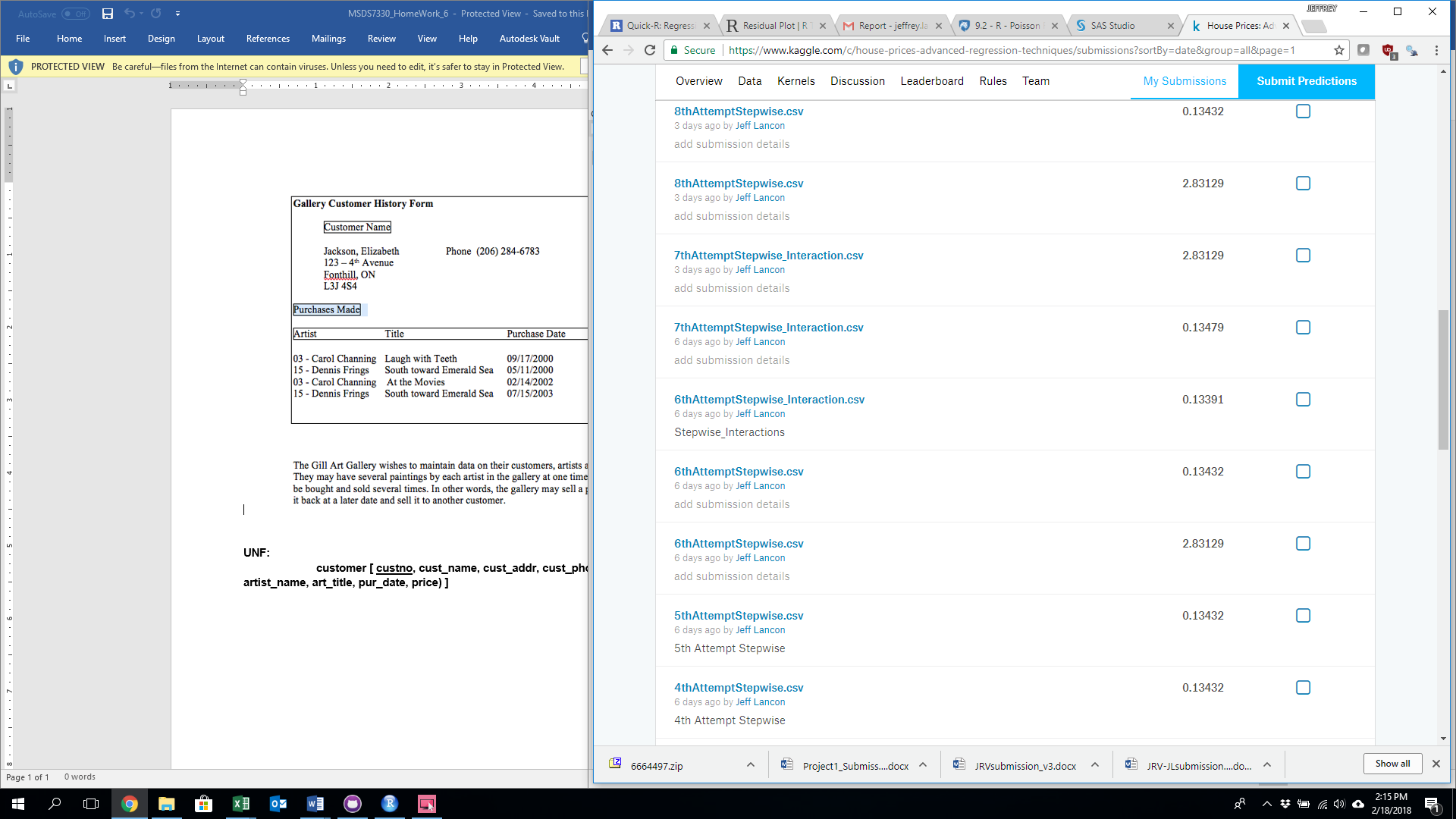
## Assumption Verification

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

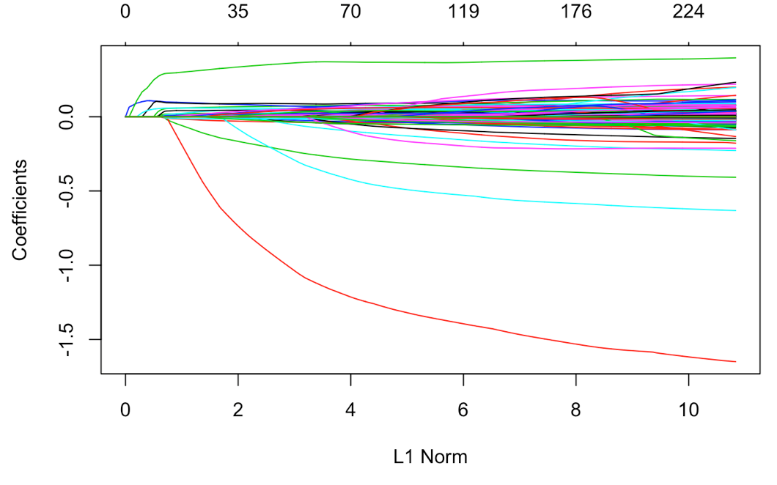
## Coefficient Selection - LASSO

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



## Graphical Coefficient - LASSO

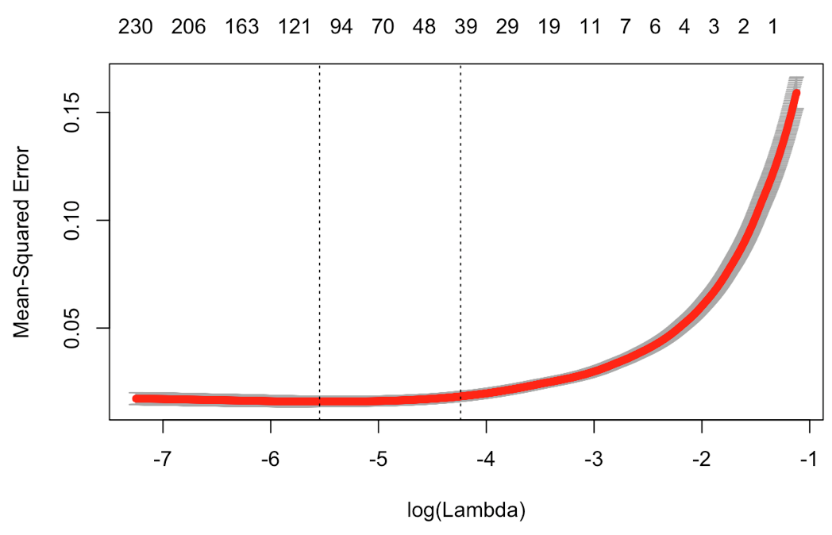
The model of coefficient fallout as lambda increases is also shown below:



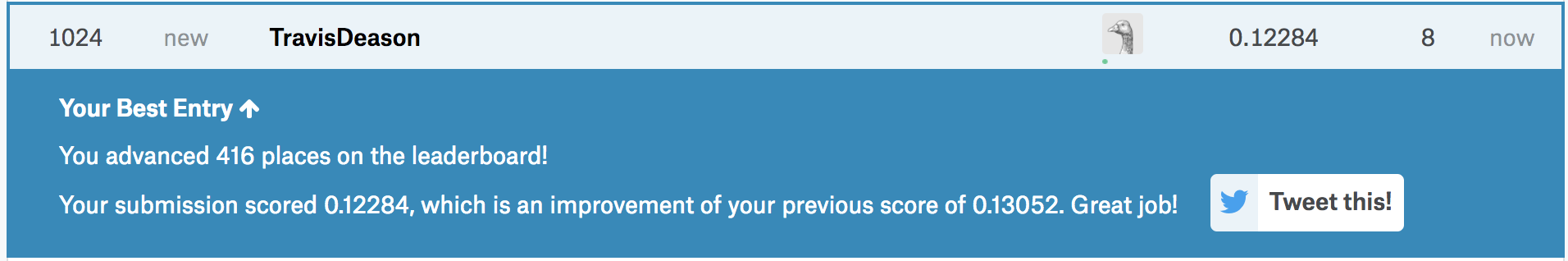
Using a tenfold cross validated version of the same glmnet model as shown (where cross validation was used as the optimization criteria), the team obtained a MSE on the test set of .0371. Overall the coefficient set used is similar in the cross-validated model.



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

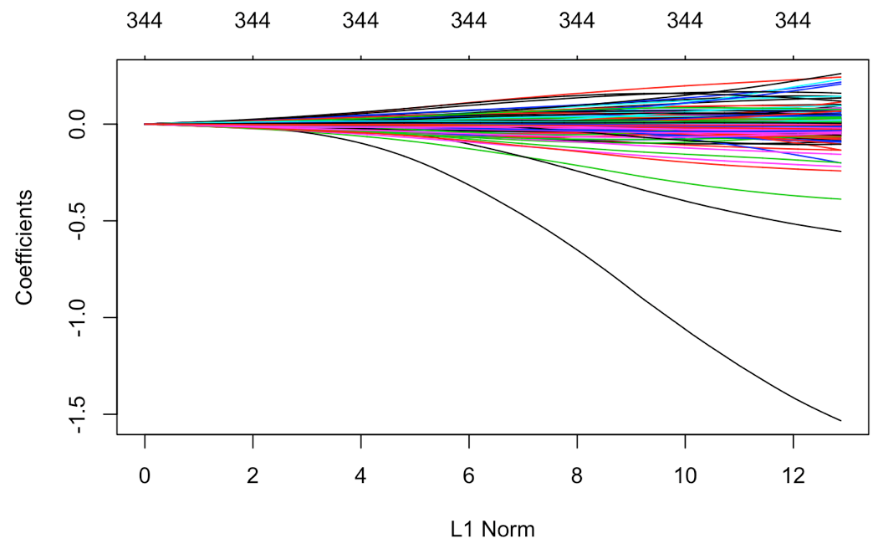
## Coefficient Selection - RIDGE

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



## Graphical Coefficient - RIDGE

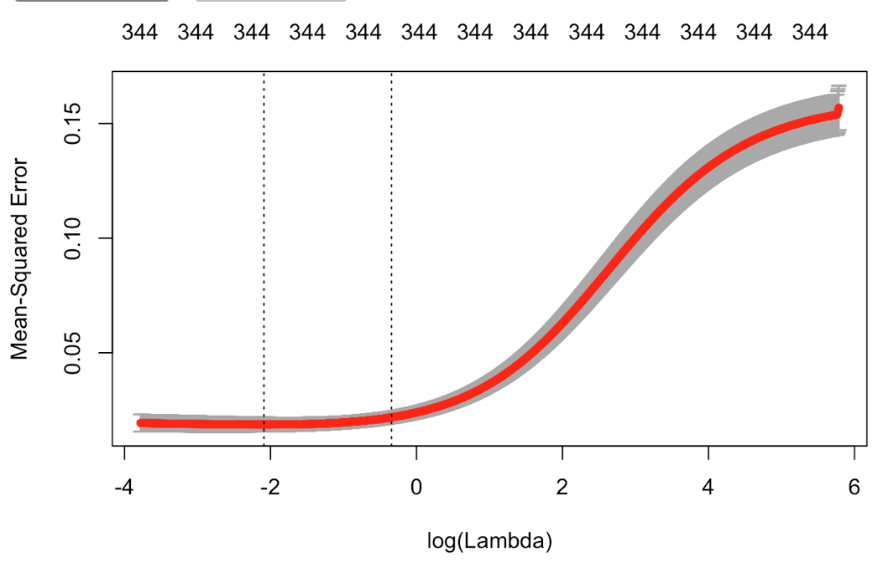
The model of coefficient fallout as lambda increases is also shown below:



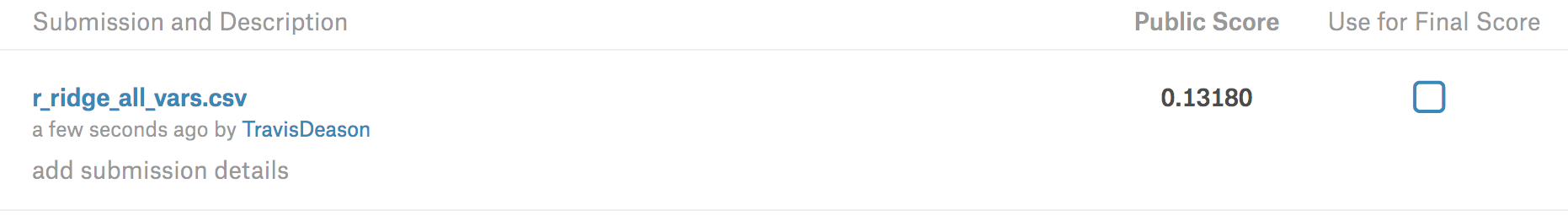
Using a 10 fold cross validated version of the same glmnet model as shown (where cross validation was used as the optimization criteria). The coefficients utilized in the cross validated model were notably different than the standard Ridge model



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

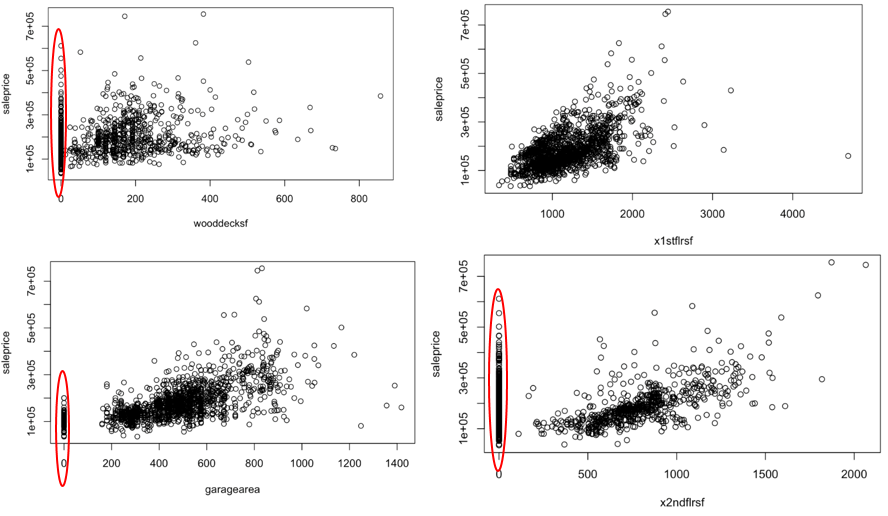
The team was tasked to create three different predictive models using data that collected in Ames Iowa from 2006 to 2010. The first model is a simplistic model using a stepwise analysis, while the second and third model are considered the most predictive models using a lasso and ridge analysis. The three 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

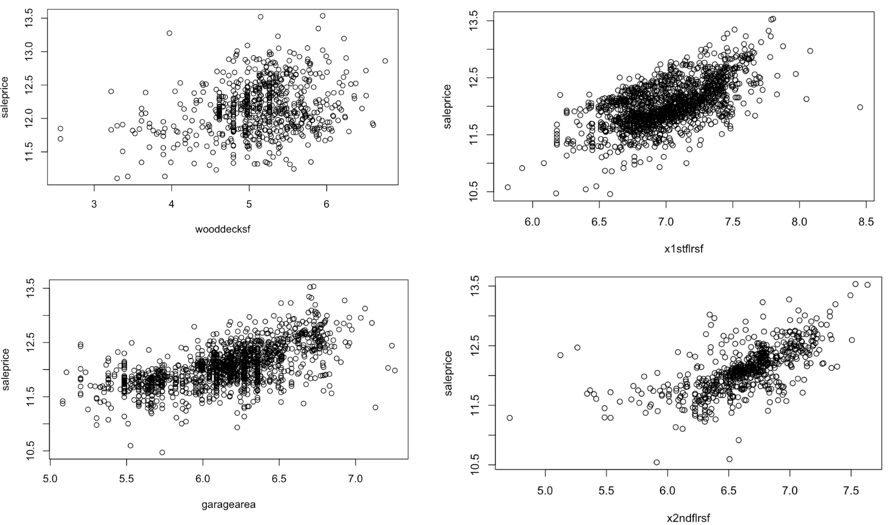


# Appendix A-01

## 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 | |
| --- | --- |
|  | The intercept in this model provides a median home SalePrice estimate of e0.5907 $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 e1.8220 $6.18 and e-0.6406 $-1.90. Note: Intercept parameter is not statistically significant. |
|  | While keeping all other parameters fixed, a doubling of greater living area (GrLivArea) is associated with a 24.05% (20.4905) multiplicative increase in the median Sale Price. A 95% Confidence interval is between 20.93% (20.45758)and 27.4% (20.52336). |
|  | While keeping all other parameters fixed, a one unit increase in overall quality (OverallQual) is associated with a 9.0% (e0.0865) multiplicative increase in median home SalePrice. A 95% Confidence interval is between7.9% (e0.0765) and 10.1% (e0.0966). |
|  | While keeping all other parameters fixed, a one unit increase in overall condition (OverallCond) is associated with a 6.0% (e0.0580) multiplicative increase of in median home SalePrice. A 95% Confidence interval is between 5.1% (e0.04992) and 6.9% (e0.06609). |
|  | While keeping all other parameters fixed, a one unit change in year built (YearBuilt) is associated with a 0.355% (e0.003544) multiplicative increase of in median home SalePrice. A 95% Confidence interval is between 0.296% (e0.002952) and 0.415% (e0.0041360). (See Note Below) |
|  | Neighborhood parameter is a categorical parameter, with multiple values, so a generalization is in order. Neighborhood differences are associated with maximum 26.8% (e0.2379) multiplicative increase to a decrease of 25.0% (e-0.2873). A 95% Confidence interval is between 39.1% (e0.3307) and 34.4% (e-0.4218). See Chart A-1 in Appendix A for a list of parameters for each individual neighborhood. |

Note: (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

