

AMES HOUSING data analysis

Project 2 – STAT 897



November 1, 2017

Daljeet singh

Penn State

Table of Contents

[Introduction 1](#_Toc497749537)

[Data Cleansing / Preparation 1](#_Toc497749538)

[Handling Missing Values 1](#_Toc497749539)

[Zoning Classification 4](#_Toc497749540)

[Lot Area 4](#_Toc497749541)

[MS.Subclass 5](#_Toc497749542)

[Bath Rooms 5](#_Toc497749543)

[Garage Year Built 6](#_Toc497749544)

[Masonry veneer type 6](#_Toc497749545)

[Square footage – Basement and Living areas 6](#_Toc497749546)

[Lot.Frontage 6](#_Toc497749547)

[Neighborhood 7](#_Toc497749548)

[Utilities 8](#_Toc497749549)

[Analysis 8](#_Toc497749550)

[Data Preparation- Test / Train Split 8](#_Toc497749551)

[Test / Train Split 8](#_Toc497749552)

[Penalized Logistic Regression 8](#_Toc497749553)

[Lasso 8](#_Toc497749554)

[Ridge 11](#_Toc497749555)

[KNN 12](#_Toc497749556)

[KNN with all parameters (after data prep) 12](#_Toc497749557)

[KNN with parameters selected from Lasso 13](#_Toc497749558)

[Results 14](#_Toc497749559)

[Conclusion 14](#_Toc497749560)

[Appendix A - Code i](#_Toc497749561)

[Appendix B – Ridge Regression Coefficients xiii](#_Toc497749562)

# Introduction

The Ames data set is an alternative to the well known Boston Housing data set. This study provides one approach of predicting whether the sale price of a house will be less than or greater than 200,000. The data analysis is primarily conducted using the R statistical modeling software.

# Data Cleansing / Preparation

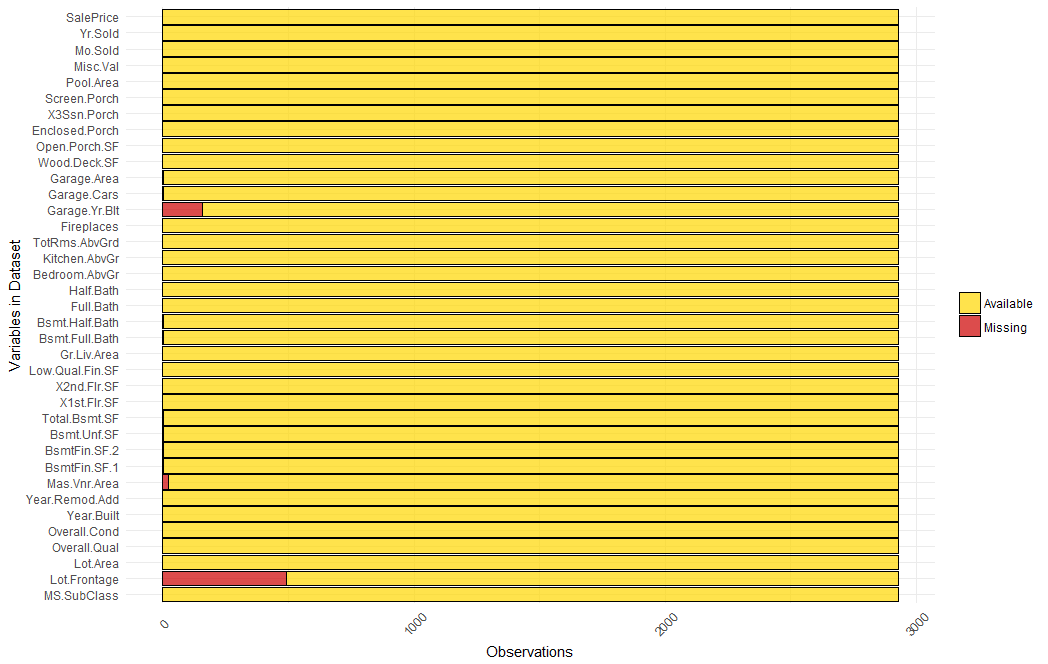
We will be analyzing this dataset with R and the first step was to analyze the data – look for missing values and try to understand the relationships. We see that we have 2930 rows and 82 columns.

As a first step we drop the columns: Order and PID as they serve no purpose in the modeling exercise leaving us with 80 columns.

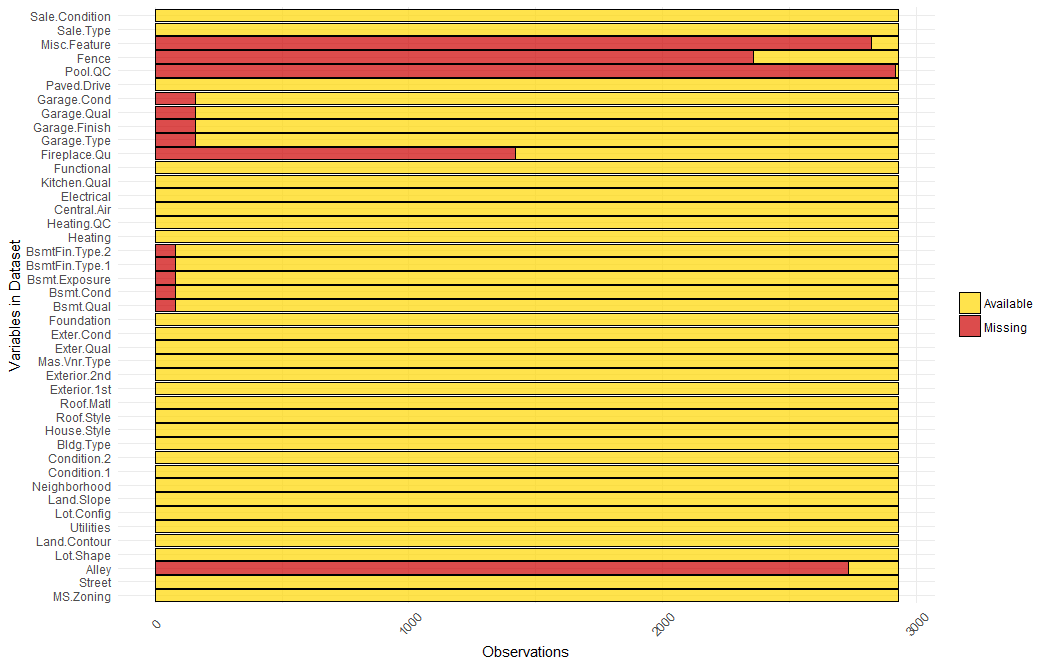
## Handling Missing Values

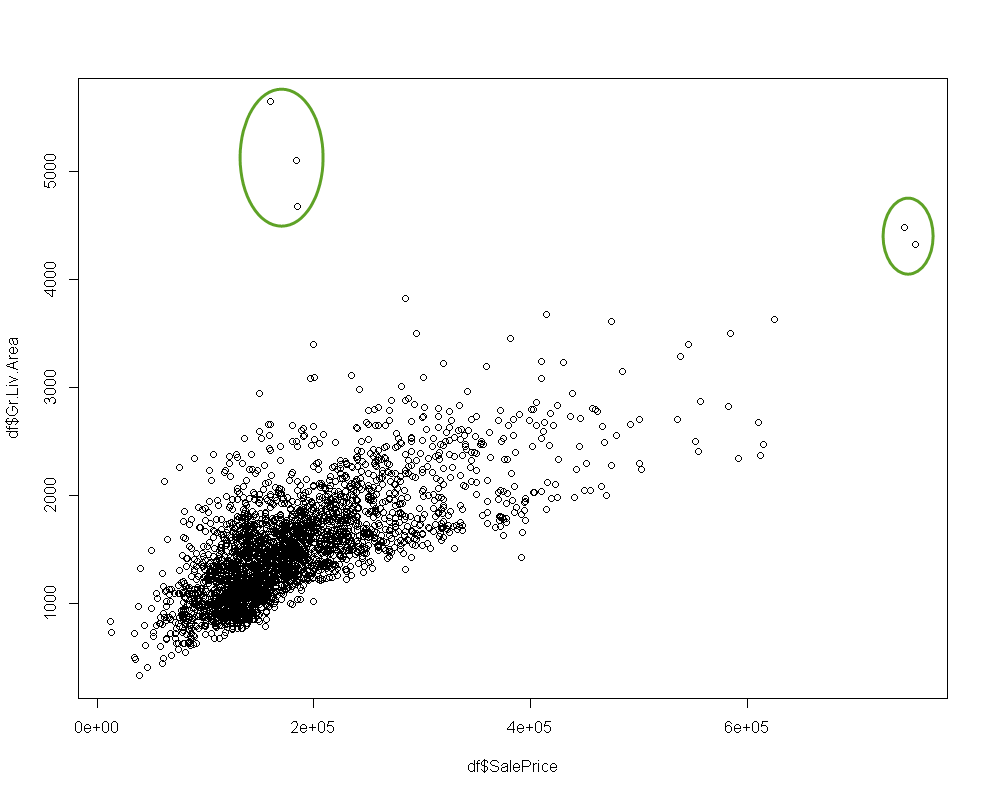
We start to see the breakdown of the missing values in our data set. We find that there are 13960 cells that contain NA (missing values). Let’s see their breakdown within numeric and categorical variables.

Numeric columns – We have 37 numeric columns

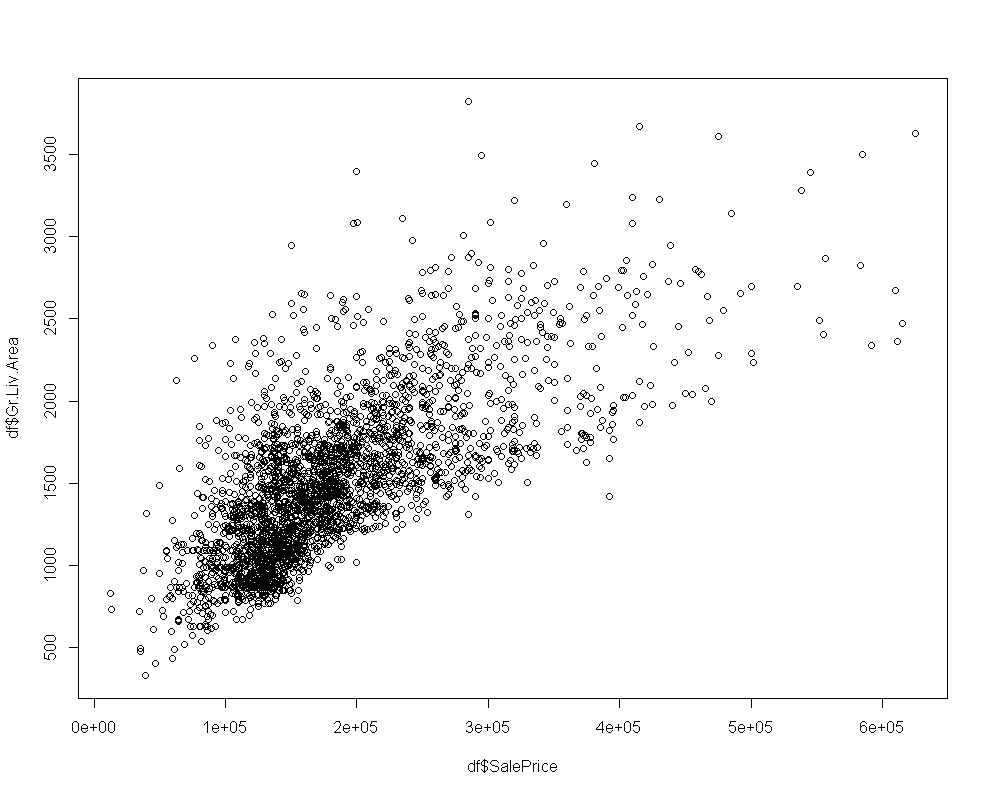


Categorical columns – We have 43 categorical columns



The data documentation mentions that there are 5 outlier rows. Let’s view them on a plot:

We will remove these rows and keep the rows that have Gr.Liv.Area<=4000. After removing we get the plot:



Let’s see the list of columns in the order of the number of missing values:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Pool.QC** | **Misc.Feature** | **Alley** | **Fence** | **Fireplace.Qu** | **Lot.Frontage** |
| 2914 | 2820 | 2737 | 2354 | 1422 | 490 |
| **Garage.Yr.Blt** | **Garage.Qual** | **Garage.Cond** | **Garage.Type** | **Garage.Finish** | **Bsmt.Qual** |
| 159 | 158 | 158 | 157 | 157 | 79 |
| **Bsmt.Cond** | **Bsmt.Exposure** | **BsmtFin.Type.1** | **BsmtFin.Type.2** | **Mas.Vnr.Area** | **Bsmt.Full.Bath** |
| 79 | 79 | 79 | 79 | 23 | 2 |
| **Bsmt.Half.Bath** | **BsmtFin.SF.1** | **BsmtFin.SF.2** | **Bsmt.Unf.SF** | **Total.Bsmt.SF** | **Garage.Cars** |
| 2 | 1 | 1 | 1 | 1 | 1 |
| **Garage.Area** |  |  |  |  |  |
| 1 |  |  |  |  |  |

From the data description we can see that some of the NA's mean that the house doesn't have the feature. They are not really missing values so we relabel those variables as none.

We choose the following variables where the NA is replaced with none:

"Pool.QC", "Misc.Feature", "Alley", "Bsmt.Qual", "Bsmt.Cond", "Bsmt.Exposure", "BsmtFin.Type.1", "BsmtFin.Type.2", "Fireplace.Qu", "Garage.Type", "Garage.Finish", "Garage.Qual", "Garage.Cond", "Fence"

After this we recompute the number of missing values and find them to be substantially less.

We now only have 682 cells with NA.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Lot.Frontage** | **Garage.Yr.Blt** | **Mas.Vnr.Area** | **Bsmt.Full.Bath** | **Bsmt.Half.Bath** |
| 490 | 159 | 23 | 2 | 2 |
| **Total.Bsmt.SF** | **Garage.Cars** | **Garage.Area** | **BsmtFin.SF.2** | **Bsmt.Unf.SF** |
| 1 | 1 | 1 | 1 | 1 |

### Zoning Classification

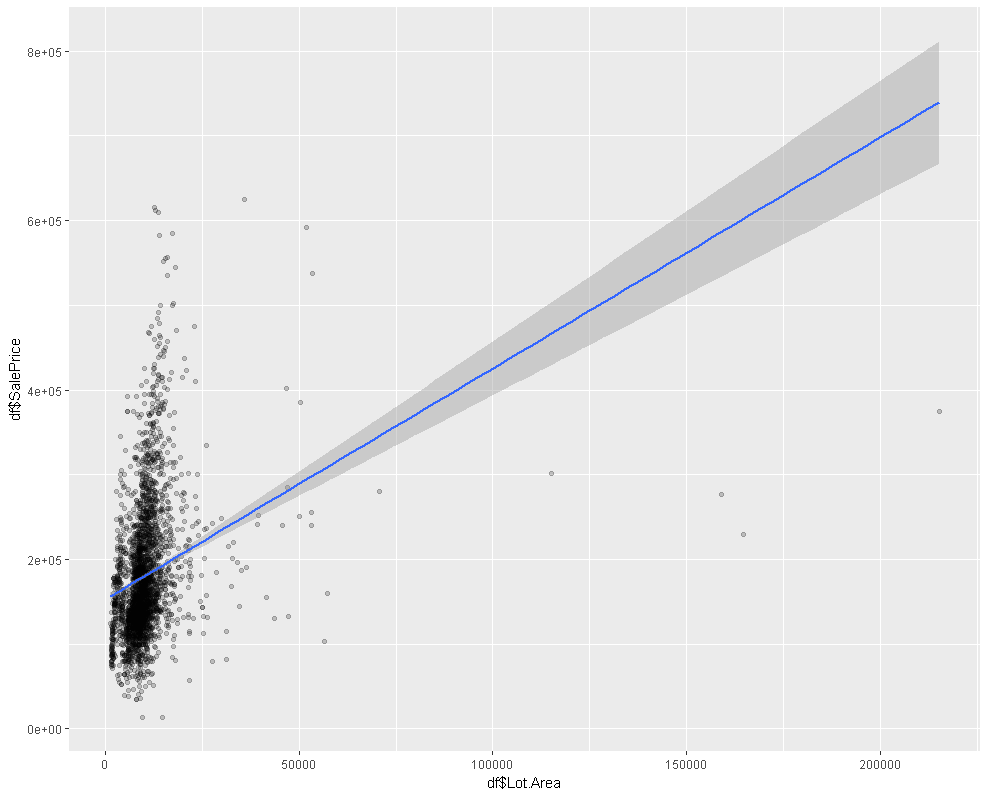
There are 8 different zoning classifications and they range from agricultural to residential. We simplified their names.

A C FV I RH RL RM

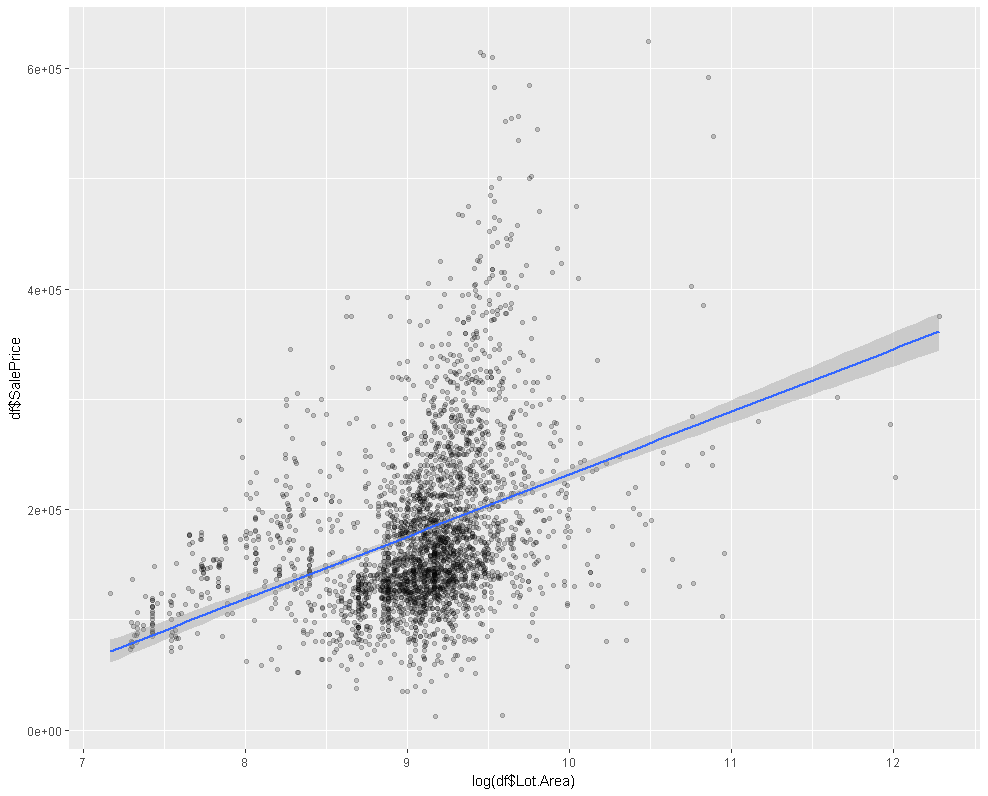
2 25 139 2 27 2268 462

### Lot Area

Here we plot Sale Price against Lot Area and get the following plot:



Clearly, we see that we need some transformation in order to perform linear regression. We try to use logarithm of the lot area and plot against the SalePrice to get:



This looks much better and therefore we add this column to the data frame and drop the original lot area.

## MS.Subclass

We dropped the MS.SubClass variable as it is a combination of bldg.type, house.style, and year.built.

## Bath Rooms

We merge the bath rooms into a single variable and drop the individual columns. We use the following evaluation:

Total.Bath = Basement Full.Bath + Full.Bath + (.5 \* Basement .Half.Bath) + (.5 \* Half.Bath)

We see the following summary of the new Total.Bath variable:

Min. 1st Qu. Median Mean 3rd Qu. Max. NA's

1.000 1.500 2.000 2.214 2.500 7.000 2

## Garage Year Built

Here we start with checking what percentage of houses have the garage year built same as the year when the house was built. We find that the year is same for 76% of the rows left at this point.

The breakdown of garage type is:

2Types Attchd Basment BuiltIn CarPort Detchd none

23 1727 36 185 15 782 157

We verify that all the 157 rows that have garage type as none also have an NA for the garage year built. This seems logical and correct. For all these rows we set the value of garage year built = 0

## Masonry veneer type

Let’s start with looking at a breakdown:

BrkCmn BrkFace CBlock None Stone

23 25 879 1 1751 246

Interestingly we have 23 rows with empty value (not NA) in addition to 1751 with value “None”. We see that for all these rows the masonry veneer area is also NA. So we think it is safe to set the 23 rows in question with the value None for masonry veneer type and value 0 for masonry veneer area.

## Square footage – Basement and Living areas

First we verify that Total.Bsmt.SF = BsmtFin.SF.1 + BsmtFin.SF.2 + Bsmt.Unf.SF and

Gr.Liv.Area = X1st.Flr.SF + X2nd.Flr.SF + Low.Qual.Fin.SF

We add another column: tot.sqft = Total.Bsmt.SF + Gr.Liv.Area

After addition of tot.sqft, we remove Bsmt.Unf.SF and Gr.Liv.Area

Now let’s check the number of remaining NA values after making all the adjustments explained above. We see the following breakdown:

Lot.Frontage Garage.Yr.Blt Total.Bath BsmtFin.SF.1 BsmtFin.SF.2

490 2 2 1 1

Total.Bsmt.SF Garage.Cars Garage.Area tot.sqft

1 1 1 1

## Lot.Frontage

This column has 17% of the rows with missing values – we will drop this column in order to proceed further with our analysis.

At this point we only have 10 cells with missing data and we find that the 10 cells are spread across only 4 rows of data. We take the simplistic approach of dropping these 4 rows leaving us with 2921 rows and 74 columns of data.

At this point we don’t have any missing values in our dataset.

## Neighborhood

Before moving further to analysis part, we take a look at the breakdown of the Neighborhood column and see the following:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Blmngtn | Blueste | BrDale | BrkSide | ClearCr | IDOTRR |
| 28 | 10 | 30 | 107 | 44 | 92 |
| Landmrk | MeadowV | Mitchel | NAmes | NoRidge | Somerst |
| 1 | 37 | 114 | 443 | 69 | 182 |
| StoneBr | SWISU | Timber | Veenker | Greens | GrnHill |
| 51 | 48 | 72 | 24 | 8 | 2 |
| CollgCr | Crawfor | Edwards | Gilbert | Sawyer | SawyerW |
| 267 | 103 | 190 | 165 | 151 | 125 |
| NPkVill | NridgHt | NWAmes | OldTown |  |  |
| 23 | 166 | 131 | 238 |  |  |

Further we check the influence of neighborhoods on saleprice (logical assumption will be there is a strong relationship). We see the following result:

Call:

lm(formula = SalePrice ~ Neighborhood, data = df)

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 196662 9672 20.333 < 2e-16 \*\*\*

NeighborhoodBlueste -53072 18854 -2.815 0.004913 \*\*

NeighborhoodBrDale -91053 13448 -6.771 1.55e-11 \*\*\*

NeighborhoodBrkSide -71478 10864 -6.579 5.59e-11 \*\*\*

NeighborhoodClearCr 12000 12372 0.970 0.332161

NeighborhoodCollgCr 5142 10166 0.506 0.613066

NeighborhoodCrawfor 10889 10908 0.998 0.318216

NeighborhoodEdwards -67344 10360 -6.500 9.41e-11 \*\*\*

NeighborhoodGilbert -6015 10460 -0.575 0.565314

NeighborhoodGreens -3130 20517 -0.153 0.878744

NeighborhoodGrnHill 83338 37459 2.225 0.026174 \*

NeighborhoodIDOTRR -93421 11046 -8.457 < 2e-16 \*\*\*

NeighborhoodLandmrk -59662 52085 -1.145 0.252111

NeighborhoodMeadowV -100905 12820 -7.871 4.92e-15 \*\*\*

NeighborhoodMitchel -34435 10795 -3.190 0.001438 \*\*

NeighborhoodNAmes -51564 9973 -5.170 2.49e-07 \*\*\*

NeighborhoodNoRidge 121493 11468 10.594 < 2e-16 \*\*\*

NeighborhoodNPkVill -55951 14402 -3.885 0.000105 \*\*\*

NeighborhoodNridgHt 125357 10456 11.989 < 2e-16 \*\*\*

NeighborhoodNWAmes -8255 10656 -0.775 0.438587

NeighborhoodOldTown -72821 10225 -7.122 1.34e-12 \*\*\*

NeighborhoodSawyer -59911 10531 -5.689 1.40e-08 \*\*\*

NeighborhoodSawyerW -12592 10700 -1.177 0.239405

NeighborhoodSomerst 33046 10389 3.181 0.001485 \*\*

NeighborhoodStoneBr 127568 12038 10.597 < 2e-16 \*\*\*

NeighborhoodSWISU -61590 12170 -5.061 4.44e-07 \*\*\*

NeighborhoodTimber 49938 11398 4.381 1.22e-05 \*\*\*

NeighborhoodVeenker 51653 14237 3.628 0.000290 \*\*\*

Overall, we see that as expected there is a strong relationship. Since the Landmrk and GrnHill neighborhoods are very poorly represented we drop the rows corresponding to these columns.

## Utilities

Let’s see a breakdown:

AllPub NoSeWa NoSewr

2915 1 2

We see that virtually all the rows have the same value and there is absence of any variance. We drop this column as it is not adding any information.

At this point we are left with 2918 rows and 73 columns.

# Analysis

## Data Preparation- Test / Train Split

The first step is to setup our categorical column. We add a new column sp\_modified and set it to “1” when the SalePrice >= 200,000. Otherwise it is set to “0”

The breakdown is:

sp\_modified

0 1

2047 871

This will be the dataset used for further analysis.

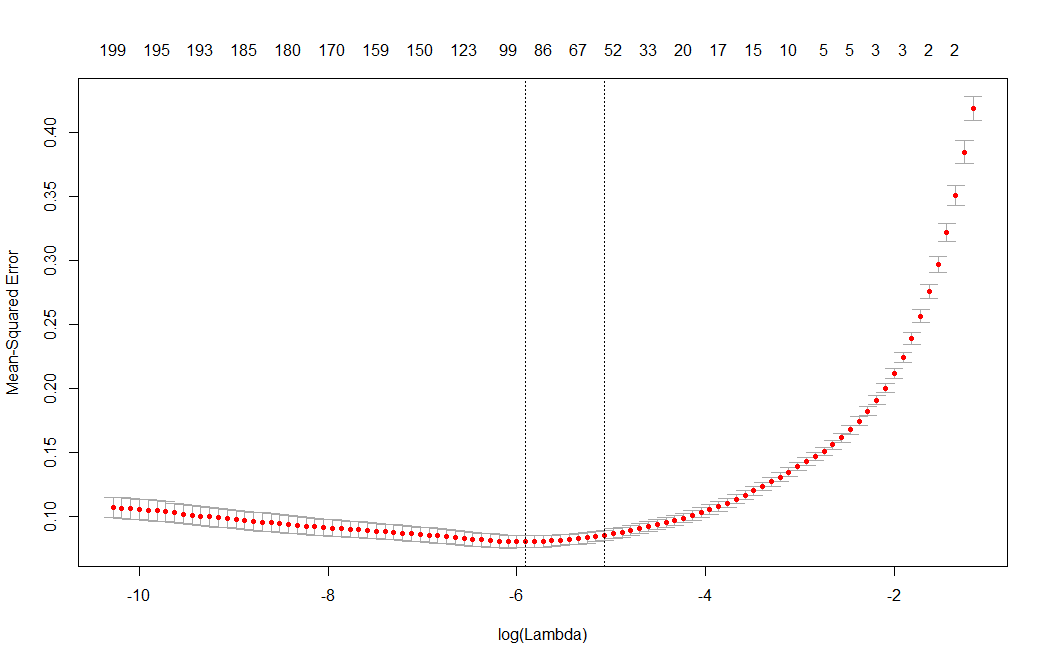
### Test / Train Split

We split the data to use 25% of the data as our test set and the remaining as training data.

## Penalized Logistic Regression

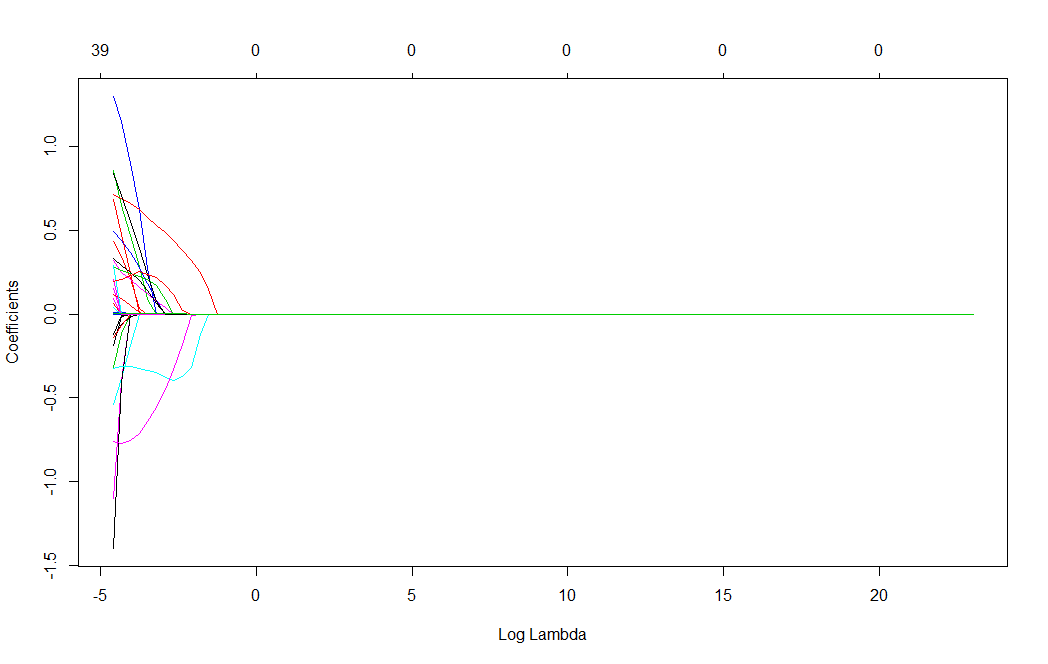
### Lasso

In order to use the lasso procedure, we first perform 10-fold cross validation on the entire data set to compute a desirable lambda:



The desired lambda: 0.002716715

We run lasso on the training data and check its response vis a vis lambda:



The prediction on the test data with the model gives us the following confusion matrix:

pred.glm 0 1

0 493 41

1 8 188

The various performance metrics for the model are:

Sensitivity Specificity Pos Pred Value Neg Pred Value

0.9840319 0.8209607 0.9232210 0.9591837

Precision Recall F1 Prevalence

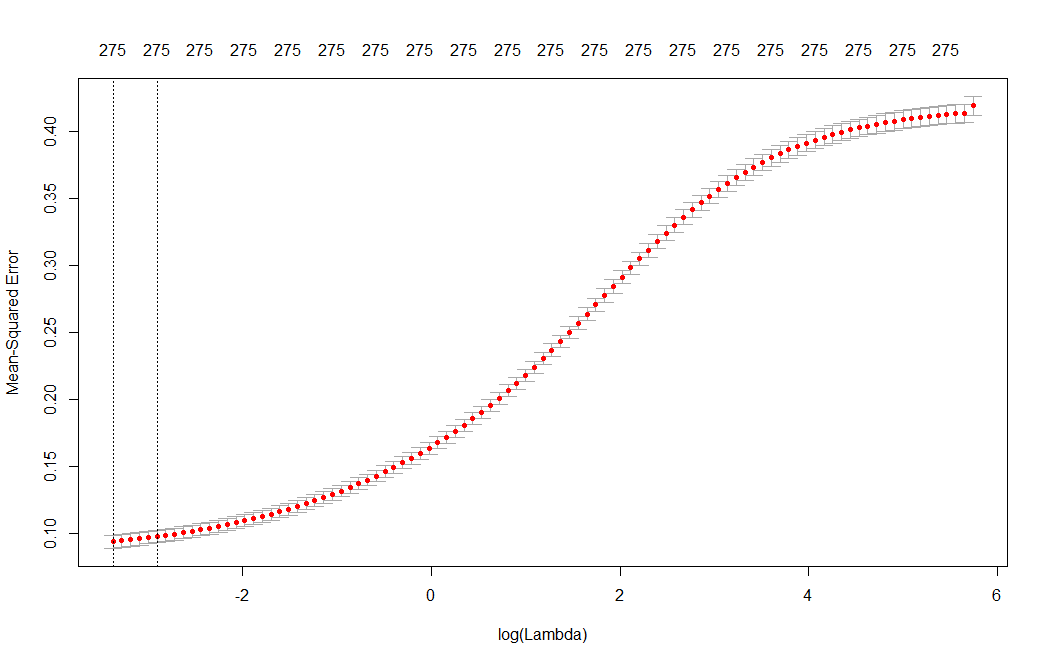
0.9232210 0.9840319 0.9526570 0.6863014

The lasso gives us the following coefficients:

|  |  |  |  |
| --- | --- | --- | --- |
| (Intercept) | Alleynone | Lot.ConfigCulDSac | Lot.ConfigFR2 |
| -3.90E+01 | 1.53E-01 | 4.36E-01 | -3.17E-01 |
| NeighborhoodCrawfor | NeighborhoodGilbert | NeighborhoodGreens | NeighborhoodNAmes |
| 1.30E+00 | -5.39E-01 | 2.06E-01 | -1.20E-01 |
| NeighborhoodNridgHt | NeighborhoodSomerst | Condition.1Norm | Condition.1PosN |
| 6.87E-01 | 8.60E-01 | 6.68E-02 | 4.20E-02 |
| Condition.2RRAe | House.Style2.5Fin | Overall.Qual | Overall.Cond |
| -1.10E+00 | -1.40E+00 | 7.14E-01 | 8.68E-02 |
| Year.Remod.Add | Roof.StyleGambrel | Roof.MatlMembran | Exterior.1stPlywood |
| 8.55E-03 | 4.21E-02 | 1.53E-01 | -1.85E-01 |
| Exterior.1stPreCast | Exterior.2ndPreCast | Exterior.2ndVinylSd | Exter.QualTA |
| 6.26E-02 | 3.13E-03 | 4.96E-01 | -3.26E-01 |
| FoundationPConc | Bsmt.QualTA | Bsmt.ExposureNo | BsmtFin.Type.1GLQ |
| 3.23E-01 | -1.41E-01 | -1.37E-01 | 2.86E-01 |
| BsmtFin.SF.1 | X2nd.Flr.SF | Kitchen.QualTA | Fireplaces |
| 8.19E-04 | 5.54E-04 | -7.63E-01 | 3.32E-01 |
| Fireplace.QuGd | Garage.Area | Open.Porch.SF | Pool.QCTA |
| 1.16E-01 | 5.16E-04 | 9.90E-05 | 2.92E-01 |
| Sale.TypeNew | ln.Lot.Area | Total.Bath | tot.sqft |
| 9.24E-02 | 8.41E-01 | 1.98E-01 | 2.38E-03 |

### Ridge

In order to use the ridge procedure, we first perform 10-fold cross validation on the entire data set to compute a desirable lambda:



The desired lambda: 0.03428106

We run ridge on the training data and the predictions on the test data with the model gives us the following confusion matrix:

pred.glm 0 1

0 490 43

1 11 186

The various performance metrics for the model are:

Sensitivity Specificity Pos Pred Value Neg Pred Value

0.9780439 0.8122271 0.9193246 0.9441624

Precision Recall F1 Prevalence

0.9193246 0.9780439 0.9477756 0.6863014

The ridge regression gives us coefficients for 274 parameters and give 0 coefficient only for Exterior.2ndOther and Sale.TypeVWD

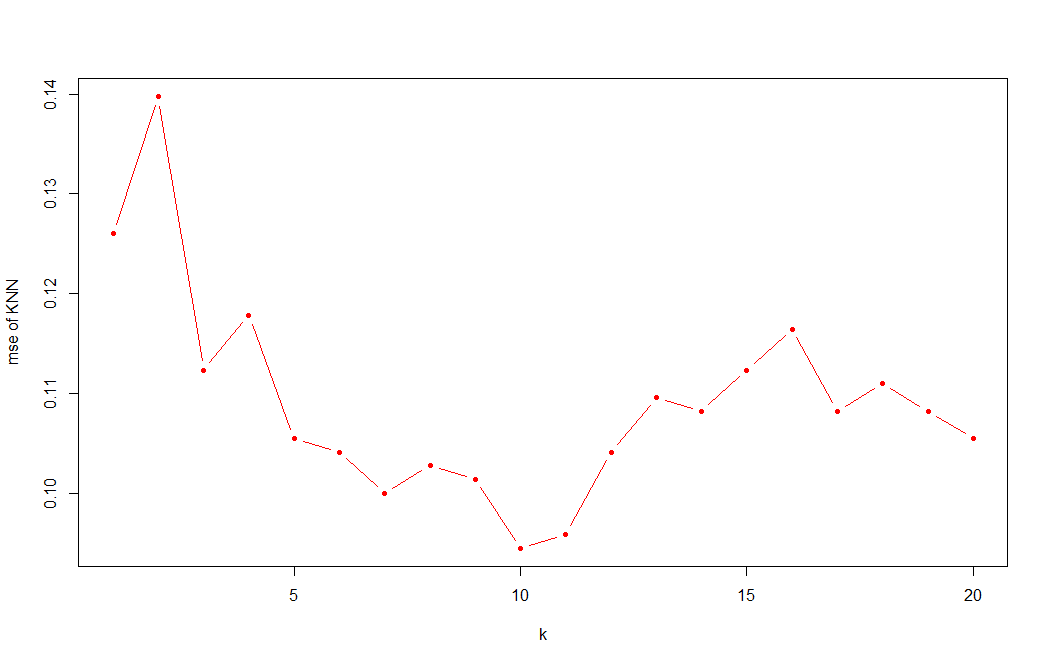
The coefficients are provided in the appendix.

## KNN

We use KNN with k selected through cross validation where we loop through k = 1 : 20

### KNN with all parameters (after data prep)

With all the parameters and k range from 1:20, we see:



We get the best results with k=10. The confusion matrix is:

0 1

0 465 33

1 36 196

The various performance metrics are:

Sensitivity Specificity Pos Pred Value Neg Pred Value

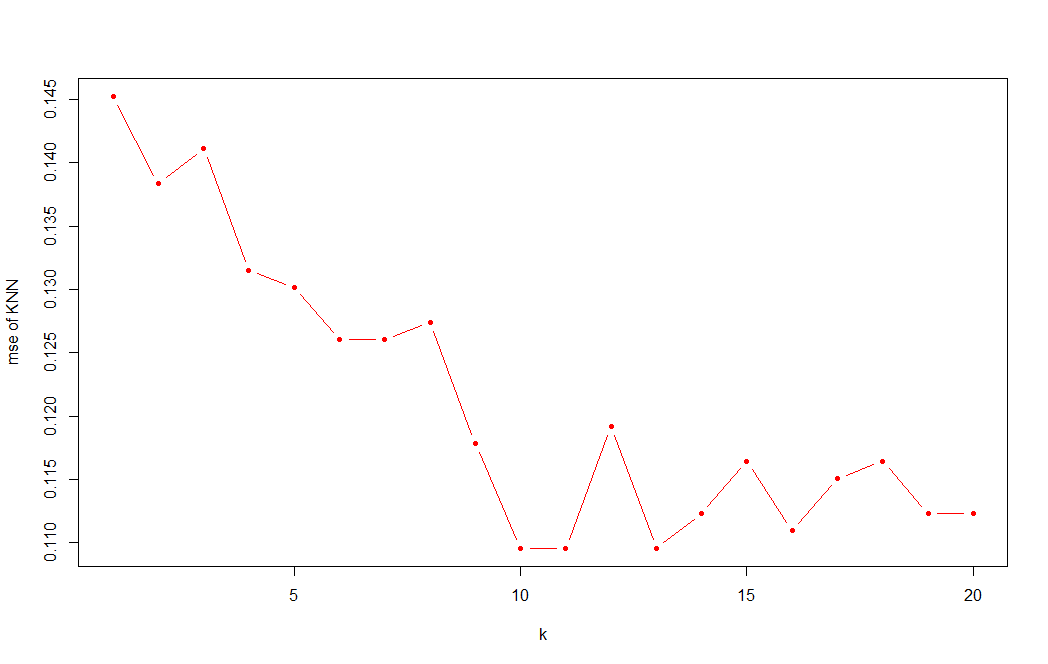
0.9281437 0.8558952 0.9337349 0.8448276

Precision Recall F1 Prevalence

0.9337349 0.9281437 0.9309309 0.6863014

### KNN with parameters selected from Lasso

With the parameters selected from Lasso and k range from 1:20, we see:



We get the best results with k=10. The confusion matrix is:

0 1

0 452 31

1 49 198

The various performance metrics are:

Sensitivity Specificity Pos Pred Value Neg Pred Value

0.9021956 0.8646288 0.9358178 0.8016194

Precision Recall F1 Prevalence

0.9358178 0.9021956 0.9186992 0.6863014

# Results

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model | Sensitivity | Specificity | Precision | F1 |
| **Penalized Logistic Regression – Lasso** | **0.9840319** | **0.8209607** | **0.9232210** | **0.9526570** |
| Penalized Logistic Regression – Ridge | 0.9780439 | 0.8122271 | 0.9193246 | 0.9780439 |
| KNN (all params, k from CV = 10) | 0.9281437 | 0.8558952 | 0.9337349 | 0.9309309 |
| **KNN (lasso params, k from CV = 10)** | **0.9021956** | **0.8646288** | **0.9358178** | **0.9186992** |

# Conclusion

Based on the various performance metrics, we find that the Penalized Logistic Regression – Lasso give the best sensitivity while KNN (lasso params, k from CV = 10) gives the best specificity. The exact approach requires that we have some more analysis of the problem domain to decipher whether sensitivity or specificity is more important for the scenario.

# Appendix A - Code

# ---

# title: "Stat 897 Fall 2017 Project 2"

# author: "Daljeet Singh"

# date: "Due November 8, 2017"

# output: pdf\_document

# ---

#setwd("C:\\study\\897\\hw")

setwd("C:\\study\\psu\\git\\897\\hw")

suppressWarnings(library(MASS))

library(class)

library(glmnet)

library(leaps)

library(caret)

library(ggplot2)

library(plyr)

library(tidyselect)

#+++++++++ FUNCTIONS

naisnone= c("Pool.QC", "Misc.Feature", "Alley", "Bsmt.Qual", "Bsmt.Cond",

"Bsmt.Exposure", "BsmtFin.Type.1", "BsmtFin.Type.2", "Fireplace.Qu",

"Garage.Type", "Garage.Finish", "Garage.Qual", "Garage.Cond", "Fence")

none= function(data, var){

levels(data[, var]) <- c(levels(data[, var]), "none")

data[, var][is.na(data[, var])] <- "none"

return(data[, var])

}

bar\_missing = function(x){

library(dplyr)

library(reshape2)

x %>%

is.na %>%

melt %>%

ggplot(data = .,

aes(x = Var2)) +

geom\_bar(aes(y=(..count..),fill=value),alpha=0.7,color="black")+scale\_fill\_manual(values=c("gold","red3"),name = "",

labels = c("Available","Missing"))+

theme\_minimal()+

theme(axis.text.x = element\_text(angle=45, vjust=0.5)) +

labs(x = "Variables in Dataset",

y = "Observations")+coord\_flip()

}

#-------------------

df = read.csv("proj2\_amesHousing.txt", sep = "\t", header = TRUE)

str(df)

dim(df)

#remove columns we dont need for the model

df = df[ , -which(names(df) %in% c("Order","PID"))]

dim(df)

#for easy looking, let's plot them separately in factor and numeric data set

#numeric data set

num = sapply(df, is.numeric)

numdat= df [, num]

bar\_missing(numdat)

#factor data set

fac= sapply(df, is.factor)

facdat= df [, fac]

bar\_missing(facdat)

# drop the rows that are outliers as explained in the data description

plot(df$SalePrice, df$Gr.Liv.Area)

df <- df[df$Gr.Liv.Area<=4000,]

plot(df$SalePrice, df$Gr.Liv.Area)

dim(df)

# first check and clean the data

# Are there any missing values in the data

any(is.na(df))

# How many are there

sum(is.na(df))

# return index of columns that have missing values

na.cols = which(colSums(is.na(df)) > 0)

# Break down missing values by variable

sort(colSums(sapply(df[na.cols], is.na)), decreasing = TRUE)

#Based on data discription, some of Na's value just mean "house doesn't have it " (not really missing value, just wrong label).So, I will do label those variables in right their categories.

for (i in 1:length(naisnone)){

df[, naisnone[i]]<- none(df, naisnone[i])

}

sum(is.na(df))

# Zoning

table(df$MS.Zoning)

df$MS.Zoning<-as.character(df$MS.Zoning)

# shorten to A

index <- which(df$MS.Zoning == "A (agr)")

df[index, 'MS.Zoning'] <- "A"

# shorten to C

index <- which(df$MS.Zoning == "C (all)")

df[index, 'MS.Zoning'] <- "C"

# Shorten to I

index <- which(df$MS.Zoning == "I (all)")

df[index, "MS.Zoning"] <- "I"

df$MS.Zoning<-factor(df$MS.Zoning)

# results

table(df$MS.Zoning)

# Lot area

# scatter plot vs sale price

ggplot(df, aes(x = df$Lot.Area, y = df$SalePrice)) +

geom\_point(alpha = 0.2) +

geom\_smooth(method = "lm")

# log lot area

ggplot(df, aes(x = log(df$Lot.Area), y = df$SalePrice)) +

geom\_point(alpha = 0.2) +

geom\_smooth(method = "lm")

# add variable

df$ln.Lot.Area <- log(df$Lot.Area)

# Delete the columns Lot.Area - keep the transformed column

df = df[ , -which(names(df) %in% c("Lot.Area"))]

# MS.SubClass

# We dropped the MS.SubClass variable as it is a combination of bldg.type, house.style, and year.built.

df = df[ , -which(names(df) %in% c("MS.SubClass"))]

dim(df)

# Bath rooms - merge into single

df$Total.Bath = df$Bsmt.Full.Bath + df$Full.Bath + (.5 \* df$Bsmt.Half.Bath) + (.5 \* df$Half.Bath)

summary(df$Total.Bath)

df = df[ , -which(names(df) %in% c("Bsmt.Full.Bath", "Full.Bath", "Bsmt.Half.Bath", "Half.Bath"))]

dim(df)

#"GarageYrBlt".

length(which(df$Garage.Yr.Blt == df$Year.Built)) / dim(df)[1]

table(df$Garage.Type)

df[df$Garage.Type == "none" && df$Garage.Yr.Blt < df$Year.Built, c("Garage.Yr.Blt", "Year.Built")]

# where Garage.Yr.Blt=none and Garage type is none, set the year to 0 representing no garage

idx <- which((is.na(df$Garage.Yr.Blt) & df$Garage.Type=="none"))

df[idx, "Garage.Yr.Blt"] <- 0

# do we have condition where garage is before the house - change garage date = year built

df[(df$Garage.Yr.Blt < df$Year.Built && df$Garage.Type != "none"), c("Garage.Yr.Blt", "Year.Built")]

#MasVnrType (Masonry veneer type) & MasVnrArea (Masonry veneer area in square feet) are related to each other

table(df$Mas.Vnr.Type)

df[(is.na(df$Mas.Vnr.Type)) | (is.na(df$Mas.Vnr.Area)), c("Mas.Vnr.Type", "Mas.Vnr.Area")]

count(df[(is.na(df$Mas.Vnr.Type)) | (is.na(df$Mas.Vnr.Area)), c("Mas.Vnr.Type", "Mas.Vnr.Area")])

df$Mas.Vnr.Type<-as.character(df$Mas.Vnr.Type)

df$Mas.Vnr.Type[df$Mas.Vnr.Type==''] = "none"

df$Mas.Vnr.Area[is.na(df$Mas.Vnr.Area)] = 0

df[(is.na(df$Mas.Vnr.Type)) | (is.na(df$Mas.Vnr.Area)), c("Mas.Vnr.Type", "Mas.Vnr.Area")]

df$Mas.Vnr.Type<-factor(df$Mas.Vnr.Type)

# square footage - basement and living areas

count(df[df$Total.Bsmt.SF!=df$BsmtFin.SF.1+df$BsmtFin.SF.2+df$Bsmt.Unf.SF,c("Bsmt.Unf.SF","BsmtFin.SF.1","BsmtFin.SF.2","Total.Bsmt.SF")])

df[df$Gr.Liv.Area!=df$X1st.Flr.SF+df$X2nd.Flr.SF+df$Low.Qual.Fin.SF,c("X1st.Flr.SF","X2nd.Flr.SF","Low.Qual.Fin.SF","Gr.Liv.Area")]

# Merge the square footage (basement, 1st and 2nd floors) and remove columns

df$tot.sqft <- df$Total.Bsmt.SF + df$Gr.Liv.Area

df = df[ , -which(names(df) %in% c("Bsmt.Unf.SF","Gr.Liv.Area"))]

# "BsmtFin.SF.2","BsmtFin.SF.1", "X1st.Flr.SF", "X2nd.Flr.SF","Low.Qual.Fin.SF"

# return index of columns that have missing values

na.cols = which(colSums(is.na(df)) > 0)

# Break down missing values by variable

sort(colSums(sapply(df[na.cols], is.na)), decreasing = TRUE)

# Delete the columns LotFrontage - many missing values

490/nrow(df)

plot(df$SalePrice, df$Lot.Frontage)

df = df[ , -which(names(df) %in% c("Lot.Frontage"))]

sum(is.na(df))

na.rows = which(rowSums(is.na(df)) > 0)

df <- df[-c(na.rows),]

dim(df)

# At this point we dont have any missing values in the data frame

num = sapply(df, is.numeric)

numdat= df [, num]

corr.matrix = cor(numdat)

#near-zero-variance

#nzv.data = nearZeroVar(df, saveMetrics = TRUE)

#drop.cols = rownames(nzv.data)[nzv.data$nzv == TRUE]

#df = df[,!names(df) %in% drop.cols]

#dim(df)

## Neighborhood

lm=lm(SalePrice ~ Neighborhood, data=df)

summary(lm)

table(df$Neighborhood)

# GrnHill and Landmrk neighborhoods have very less representation. removing these rows

df <- df[!(df$Neighborhood == "GrnHill" | df$Neighborhood == "Landmrk"),]

table(df$Neighborhood)

## Utilities

table(df$Utilities)

# Virtually no variance in the column, drop the column

df = df[ , -which(names(df) %in% c("Utilities"))]

dim(df)

## Parameter selection

# First add the classification column

sp\_modified <- rep(0, length(df$SalePrice))

sp\_modified[which(df$SalePrice >= 200000)] <- 1

df = df[ , -which(names(df) %in% c("SalePrice"))]

table(sp\_modified)

df <- data.frame(df, sp\_modified)

dim(df)

n=nrow(df)

set.seed(7736)

test = sample(n, round(n/4)) ## train indices are the rest

train = setdiff(1:n, test)

df.train = df[ train,]

df.test = df[test,]

x <- model.matrix(sp\_modified~ ., data = df)

train.mat <- model.matrix(sp\_modified~ ., data = df.train)

test.mat <- model.matrix(sp\_modified~ ., data = df.test)

#which(sapply(df.test, function(x) (is.character(x) | is.factor(x)) & length(unique(x))<2))

# Forward Selection | BIC

regfit.fwd=regsubsets (sp\_modified~.,data=df.train, nvmax=p, method='forward')

reg.summary = summary (regfit.fwd)

plot(reg.summary$bic, xlab ="Number of Variables",ylab="BIC", type = 'l', main = 'Forward Step - Performance Measure')

which.min (reg.summary$bic )

points (which.min (reg.summary$bic ), reg.summary$bic[which.min (reg.summary$bic )], col ="red",cex =2, pch =20)

coefi=coef(regfit.fwd ,id=which.min (reg.summary$bic ))

pred=test.mat [,names(coefi)] %\*% coefi

mean(( df.test$SalePrice-pred)^2)

names(coefi)

plot(reg.summary$cp, xlab ="Number of Variables",ylab="cp", type = 'l', main = 'Forward Step - Performance Measure')

which.min (reg.summary$cp )

points (which.min (reg.summary$cp ), reg.summary$cp[which.min (reg.summary$cp )], col ="red",cex =2, pch =20)

coefi=coef(regfit.fwd ,id=which.min (reg.summary$cp ))

pred=test.mat [,names(coefi)] %\*% coefi

mean(( df.test$SalePrice-pred)^2)

regfit.bwd=regsubsets (SalePrice~.,data=df.train, nvmax=p, method='backward')

reg.summary = summary(regfit.bwd)

plot(reg.summary$bic, xlab ="Number of Variables",ylab="BIC", type = 'l', main = 'Backward Step - Performance Measure')

which.min (reg.summary$bic )

points (which.min (reg.summary$bic ), reg.summary$bic[which.min (reg.summary$bic )], col ="red",cex =2, pch =20)

plot(reg.summary$cp, xlab ="Number of Variables",ylab="cp", type = 'l', main = 'Backward Step - Performance Measure')

which.min (reg.summary$cp )

points (which.min (reg.summary$cp), reg.summary$cp[which.min (reg.summary$cp)], col ="red",cex =2, pch =20)

# LASSO / Penalized Logistic Regression

grid =10^ seq (10,-2, length =100)

cv.lasso = cv.glmnet(x, y=as.factor(df$sp\_modified), family="binomial",

type.measure = "mse", nfolds=10)

plot(cv.lasso)

bestlam.lasso=cv.lasso$lambda.min #find the best tuning parameter

fit.lasso <- glmnet(train.mat, y=as.factor(df.train$sp\_modified), alpha=1,

family="binomial", lambda = grid, thresh = 1e-12)

plot(fit.lasso, xvar="lambda")

#fit.lasso <- glmnet(train.mat, df.train$SalePrice, alpha = 1, lambda = grid, thresh = 1e-12)

probs=predict (fit.lasso, s=bestlam.lasso, newx=test.mat)

pred.glm <- rep(0, length(probs))

pred.glm[probs > 0.5] <- 1

conf\_matrix = table(pred.glm, df.test$sp\_modified)

conf\_matrix

cm=confusionMatrix(data = pred.glm, reference = df.test$sp\_modified)

cm$byClass

lasso.coef=predict(fit.lasso,type="coefficients",s=bestlam.lasso)

lasso.coef=lasso.coef[1:length(lasso.coef),]

length(lasso.coef[lasso.coef !=0])

lasso.coef[lasso.coef!=0]

final.lasso=glmnet(x,y=as.factor(df$sp\_modified),alpha=1,family = "binomial") #fit on the entire data set to extract coef

lasso.coef=predict(final.lasso,type="coefficients",s=bestlam.lasso)

lasso.coef=lasso.coef[1:length(lasso.coef),]

length(lasso.coef[lasso.coef !=0])

names(lasso.coef[lasso.coef!=0])

#Ridge regression

cv.ridge = cv.glmnet(x, y=as.factor(df$sp\_modified), alpha=0, type.measure="mse",

family="binomial", nfolds=10,grouped=FALSE)

plot(cv.ridge)

bestlam.ridge=cv.ridge$lambda.min #find the best tuning parameter

fit.ridge =glmnet(train.mat, y=as.factor(df.train$sp\_modified), family="binomial",

alpha=0, lambda=grid, thresh=1e-12)

probs = predict (fit.ridge, s=bestlam.ridge, newx=test.mat)

pred.glm <- rep(0, length(probs))

pred.glm[probs > 0.5] <- 1

conf\_matrix = table(pred.glm, df.test$sp\_modified)

conf\_matrix

cm=confusionMatrix(data = pred.glm, reference = df.test$sp\_modified)

cm$byClass

ridge.coef=predict(fit.ridge,type="coefficients",s=bestlam.ridge)

ridge.coef=ridge.coef[1:length(ridge.coef),]

length(ridge.coef[ridge.coef !=0])

ridge.coef[ridge.coef!=0]

final.ridge=glmnet(x,y,alpha=0) #fit on the full data

ridge.coef=predict(final.ridge,type="coefficients",s=bestlam.ridge)

ridge.coef=lasso.coef[1:length(ridge.coef),]

length(ridge.coef[ridge.coef !=0])

names(ridge.coef[ridge.coef!=0])

# KNN with CV - all variables left after cleaning

fit.knn = pred.knn = vector("list", 20)

vmse.knn = rep(NA, 20)

for (k in 1:20){ ## set k from 1 to 20

fit.knn[[k]] = knn(train.mat, test.mat, df.train$sp\_modified, k = k)

pred.knn[[k]] = as.numeric(fit.knn[[k]]) - 1

vmse.knn[k] = mean(abs(df.test$sp\_modified - pred.knn[[k]]))

}

### misclassification rate for each k from 1 to 20

plot(vmse.knn, type = 'b', col = 'red', pch = 20, xlab = 'k', ylab = 'mse of KNN')

table(pred.knn[[which.min(vmse.knn)]], df.test$sp\_modified)

mean(pred.knn[[which.min(vmse.knn)]] == df.test$sp\_modified)

cm=confusionMatrix(data = pred.knn[[which.min(vmse.knn)]], reference = df.test$sp\_modified)

cm$byClass

#Pool.QC + Sale.Type + Roof.Style + Roof.Matl + Fireplace.Qu + Alley + Overall.Qual + Exterior.1st + Foundation + Total.Bath + Overall.Cond + Exterior.2nd + Bsmt.Qual + Kitchen.Qual + tot.sqft + Neighborhood + Condition.1 + Year.Remod.Add + Bsmt.Exposure + Open.Porch.SF + Condition.2 + Exter.Qual + BsmtFin.Type.1 + Fireplaces + Garage.Area + ln.Lot.Area

train.lasso.mat <- model.matrix(sp\_modified~ Pool.QC + Sale.Type + Roof.Style + Roof.Matl + Fireplace.Qu + Alley + Overall.Qual + Exterior.1st + Foundation + Total.Bath + Overall.Cond + Exterior.2nd + Bsmt.Qual + Kitchen.Qual + tot.sqft + Neighborhood + Condition.1 + Year.Remod.Add + Bsmt.Exposure + Open.Porch.SF + Condition.2 + Exter.Qual + BsmtFin.Type.1 + Fireplaces + Garage.Area + ln.Lot.Area, data = df.train)

test.lasso.mat <- model.matrix(sp\_modified~ Pool.QC + Sale.Type + Roof.Style + Roof.Matl + Fireplace.Qu + Alley + Overall.Qual + Exterior.1st + Foundation + Total.Bath + Overall.Cond + Exterior.2nd + Bsmt.Qual + Kitchen.Qual + tot.sqft + Neighborhood + Condition.1 + Year.Remod.Add + Bsmt.Exposure + Open.Porch.SF + Condition.2 + Exter.Qual + BsmtFin.Type.1 + Fireplaces + Garage.Area + ln.Lot.Area, data = df.test)

# KNN with CV - all variables left after cleaning

fit.knn = pred.knn = vector("list", 20)

vmse.knn = rep(NA, 20)

for (k in 1:20){ ## set k from 1 to 20

fit.knn[[k]] = knn(train.lasso.mat, test.lasso.mat, df.train$sp\_modified, k = k)

pred.knn[[k]] = as.numeric(fit.knn[[k]]) - 1

vmse.knn[k] = mean(abs(df.test$sp\_modified - pred.knn[[k]]))

}

### misclassification rate for each k from 1 to 20

plot(vmse.knn, type = 'b', col = 'red', pch = 20, xlab = 'k', ylab = 'mse of KNN')

table(pred.knn[[which.min(vmse.knn)]], df.test$sp\_modified)

mean(pred.knn[[which.min(vmse.knn)]] == df.test$sp\_modified)

cm=confusionMatrix(data = pred.knn[[which.min(vmse.knn)]], reference = df.test$sp\_modified)

cm$byClass

# Appendix B – Ridge Regression Coefficients

(Intercept) MS.ZoningC MS.ZoningFV MS.ZoningI

-4.473075e+01 -4.564987e-01 5.747199e-01 -3.964397e-04

MS.ZoningRH MS.ZoningRL MS.ZoningRM StreetPave

-5.815476e-01 7.439487e-03 -1.359669e-01 1.277518e+00

AlleyPave Alleynone Lot.ShapeIR2 Lot.ShapeIR3

-5.552337e-01 4.666482e-01 1.042024e-01 -5.242932e-01

Lot.ShapeReg Land.ContourHLS Land.ContourLow Land.ContourLvl

-1.383525e-01 3.410755e-02 -4.158164e-01 -3.989572e-01

Lot.ConfigCulDSac Lot.ConfigFR2 Lot.ConfigFR3 Lot.ConfigInside

5.503166e-01 -7.795137e-01 2.159194e-01 -9.283894e-02

Land.SlopeMod Land.SlopeSev NeighborhoodBlueste NeighborhoodBrDale

1.537145e-01 -3.668225e-01 1.106475e+00 -1.493005e-02

NeighborhoodBrkSide NeighborhoodClearCr NeighborhoodCollgCr NeighborhoodCrawfor

7.643183e-01 2.705525e-01 -4.472558e-02 1.154540e+00

NeighborhoodEdwards NeighborhoodGilbert NeighborhoodGreens NeighborhoodIDOTRR

5.677413e-02 -1.024496e+00 1.535894e+00 2.127982e-01

NeighborhoodMeadowV NeighborhoodMitchel NeighborhoodNAmes NeighborhoodNoRidge

-5.561355e-01 -9.220967e-02 -4.104269e-01 4.641549e-01

NeighborhoodNPkVill NeighborhoodNridgHt NeighborhoodNWAmes NeighborhoodOldTown

-1.010770e-01 8.502488e-01 -1.745019e-01 -2.359110e-01

NeighborhoodSawyer NeighborhoodSawyerW NeighborhoodSomerst NeighborhoodStoneBr

-4.740486e-01 -2.762014e-01 7.918841e-01 8.135636e-01

NeighborhoodSWISU NeighborhoodTimber NeighborhoodVeenker Condition.1Feedr

-6.385639e-01 -1.928471e-02 5.804303e-01 -1.206468e-01

Condition.1Norm Condition.1PosA Condition.1PosN Condition.1RRAe

2.821646e-01 -9.957585e-02 7.200395e-01 -8.871150e-01

Condition.1RRAn Condition.1RRNe Condition.1RRNn Condition.2Feedr

-4.267407e-01 -1.197222e+00 4.758265e-01 -5.077571e-01

Condition.2Norm Condition.2PosA Condition.2PosN Condition.2RRAe

2.763201e-01 3.678740e-01 2.028125e-01 -2.267671e+00

Condition.2RRAn Condition.2RRNn Bldg.Type2fmCon Bldg.TypeDuplex

-4.553018e-01 -4.933953e-03 -5.797217e-01 -2.011800e-01

Bldg.TypeTwnhs Bldg.TypeTwnhsE House.Style1.5Unf House.Style1Story

-4.326791e-01 -3.174199e-01 -7.879223e-02 -9.776920e-02

House.Style2.5Fin House.Style2.5Unf House.Style2Story House.StyleSFoyer

-2.126096e+00 8.124810e-01 1.578715e-01 -3.963886e-01

House.StyleSLvl Overall.Qual Overall.Cond Year.Built

-6.046550e-01 2.794347e-01 1.762132e-01 1.802306e-03

Year.Remod.Add Roof.StyleGable Roof.StyleGambrel Roof.StyleHip

7.210531e-03 -5.863128e-02 1.368095e+00 -1.362478e-02

Roof.StyleMansard Roof.StyleShed Roof.MatlCompShg Roof.MatlMembran

1.253898e-01 1.054659e+00 4.510037e-02 2.448643e+00

Roof.MatlMetal Roof.MatlRoll Roof.MatlTar&Grv Roof.MatlWdShake

-6.338028e-01 -2.075795e-01 -2.996472e-01 -4.164350e-01

Roof.MatlWdShngl Exterior.1stAsphShn Exterior.1stBrkComm Exterior.1stBrkFace

9.625981e-01 -6.969258e-03 6.815191e-01 7.453218e-01

Exterior.1stCBlock Exterior.1stCemntBd Exterior.1stHdBoard Exterior.1stImStucc

-3.130426e-02 4.276397e-01 -3.757346e-01 1.511990e+00

Exterior.1stMetalSd Exterior.1stPlywood Exterior.1stPreCast Exterior.1stStone

-6.627008e-02 -4.288988e-01 1.918676e+00 1.734634e+00

Exterior.1stStucco Exterior.1stVinylSd Exterior.1stWd Sdng Exterior.1stWdShing

2.531995e-01 1.949370e-01 -7.818944e-03 1.112191e-01

Exterior.2ndAsphShn Exterior.2ndBrk Cmn Exterior.2ndBrkFace Exterior.2ndCBlock

-3.908308e-01 -4.195584e-01 -5.849666e-02 -1.859628e-02

Exterior.2ndCmentBd Exterior.2ndHdBoard Exterior.2ndImStucc Exterior.2ndMetalSd

3.238495e-01 -2.017607e-01 -1.011876e+00 -2.961331e-02

Exterior.2ndPlywood Exterior.2ndPreCast Exterior.2ndStone Exterior.2ndStucco

-1.561150e-01 1.918672e+00 -2.077054e-01 2.299075e-01

Exterior.2ndVinylSd Exterior.2ndWd Sdng Exterior.2ndWd Shng Mas.Vnr.TypeBrkFace

2.225520e-01 2.870503e-02 -5.506716e-01 1.216286e-01

Mas.Vnr.TypeCBlock Mas.Vnr.Typenone Mas.Vnr.TypeNone Mas.Vnr.TypeStone

-3.633759e-01 2.593037e-01 -1.589368e-01 2.696012e-01

Mas.Vnr.Area Exter.QualFa Exter.QualGd Exter.QualTA

4.718143e-04 -7.826729e-02 2.466809e-01 -2.637356e-01

Exter.CondFa Exter.CondGd Exter.CondPo Exter.CondTA

-3.325552e-01 2.804687e-01 -7.420323e-03 -1.034799e-01

FoundationCBlock FoundationPConc FoundationSlab FoundationStone

-2.177718e-01 2.379263e-01 -1.020447e-01 1.539197e+00

FoundationWood Bsmt.QualEx Bsmt.QualFa Bsmt.QualGd

8.230300e-01 2.526047e-01 2.649028e-02 1.460814e-01

Bsmt.QualPo Bsmt.QualTA Bsmt.Qualnone Bsmt.CondEx

-2.590396e-02 -2.104799e-01 -1.590432e-01 -1.676419e-01

Bsmt.CondFa Bsmt.CondGd Bsmt.CondPo Bsmt.CondTA

-2.244743e-01 1.276062e-01 -1.540370e-02 6.623493e-02

Bsmt.Condnone Bsmt.ExposureAv Bsmt.ExposureGd Bsmt.ExposureMn

-1.590432e-01 -1.109024e-02 3.276991e-01 1.991076e-01

Bsmt.ExposureNo Bsmt.Exposurenone BsmtFin.Type.1ALQ BsmtFin.Type.1BLQ

-1.550337e-01 -1.590431e-01 1.056788e-01 -1.814767e-01

BsmtFin.Type.1GLQ BsmtFin.Type.1LwQ BsmtFin.Type.1Rec BsmtFin.Type.1Unf

3.391665e-01 -1.090846e-01 -2.521193e-01 -1.841230e-01

BsmtFin.Type.1none BsmtFin.SF.1 BsmtFin.Type.2ALQ BsmtFin.Type.2BLQ

-1.590431e-01 6.430554e-04 1.553907e-01 -1.804543e-01

BsmtFin.Type.2GLQ BsmtFin.Type.2LwQ BsmtFin.Type.2Rec BsmtFin.Type.2Unf

7.244074e-01 2.175502e-01 -4.803008e-01 6.918946e-02

BsmtFin.Type.2none BsmtFin.SF.2 Total.Bsmt.SF HeatingGasA

-1.590431e-01 3.412519e-04 6.619186e-04 -8.279005e-02

HeatingGasW HeatingGrav HeatingOthW HeatingWall

1.706284e-01 -3.531994e-02 -4.752292e-01 -5.147425e-03

Heating.QCFa Heating.QCGd Heating.QCPo Heating.QCTA

-1.142853e-01 -1.661395e-01 -1.417069e-02 -1.878012e-01

Central.AirY ElectricalFuseA ElectricalFuseF ElectricalFuseP

-7.812991e-02 1.234918e-01 -1.927224e-01 -1.684330e-01

ElectricalMix ElectricalSBrkr X1st.Flr.SF X2nd.Flr.SF

-2.989281e-03 -4.344772e-02 9.600889e-04 7.660454e-04

Low.Qual.Fin.SF Bedroom.AbvGr Kitchen.AbvGr Kitchen.QualFa

1.354068e-04 1.072028e-01 -2.924249e-01 -6.704059e-02

Kitchen.QualGd Kitchen.QualPo Kitchen.QualTA TotRms.AbvGrd

1.391646e-01 -5.094216e-02 -4.117741e-01 1.492328e-01

FunctionalMaj2 FunctionalMin1 FunctionalMin2 FunctionalMod

-7.661997e-01 -3.957131e-01 3.230046e-02 -1.254312e-01

FunctionalSal FunctionalSev FunctionalTyp Fireplaces

-6.238148e-03 -5.170959e-01 1.778588e-01 3.154035e-01

Fireplace.QuFa Fireplace.QuGd Fireplace.QuPo Fireplace.QuTA

-2.828073e-01 2.531555e-01 -3.821490e-01 8.276442e-02

Fireplace.Qunone Garage.TypeAttchd Garage.TypeBasment Garage.TypeBuiltIn

-2.284256e-01 1.079294e-01 -1.421210e-01 4.627151e-01

Garage.TypeCarPort Garage.TypeDetchd Garage.Typenone Garage.Yr.Blt

-2.254956e-01 -1.730738e-01 8.524196e-05 7.466293e-06

Garage.FinishFin Garage.FinishRFn Garage.FinishUnf Garage.Finishnone

4.032337e-02 1.705061e-02 -4.463528e-02 8.531684e-05

Garage.Cars Garage.Area Garage.QualEx Garage.QualFa

2.249632e-01 1.012650e-03 1.330375e-01 6.955758e-02

Garage.QualGd Garage.QualPo Garage.QualTA Garage.Qualnone

1.261185e-01 -1.946284e-01 -4.027534e-02 8.563769e-05

Garage.CondEx Garage.CondFa Garage.CondGd Garage.CondPo

-1.080630e-01 -1.205938e-01 -2.049298e-01 -1.890503e-01

Garage.CondTA Garage.Condnone Paved.DriveP Paved.DriveY

6.645717e-02 8.601985e-05 -3.986499e-02 -2.907394e-02

Wood.Deck.SF Open.Porch.SF Enclosed.Porch X3Ssn.Porch

5.997815e-04 2.302717e-03 7.168263e-04 1.657646e-03

Screen.Porch Pool.Area Pool.QCFa Pool.QCGd

1.291232e-03 7.908643e-04 -7.645098e-01 8.238487e-01

Pool.QCTA Pool.QCnone FenceGdWo FenceMnPrv

1.884696e+00 -6.678434e-01 -6.167139e-01 -4.325792e-03

FenceMnWw Fencenone Misc.FeatureGar2 Misc.FeatureOthr

-4.694203e-01 4.052676e-02 -7.424189e-01 1.265422e+00

Misc.FeatureShed Misc.FeatureTenC Misc.Featurenone Misc.Val

6.436814e-02 5.516304e-01 -6.570582e-02 3.136372e-05

Mo.Sold Yr.Sold Sale.TypeCon Sale.TypeConLD

-8.890643e-03 4.042577e-03 1.068745e+00 1.610627e-01

Sale.TypeConLI Sale.TypeConLw Sale.TypeCWD Sale.TypeNew

-5.615934e-02 4.488723e-01 1.129521e+00 3.238855e-01

Sale.TypeOth Sale.TypeWD Sale.ConditionAdjLand Sale.ConditionAlloca

-2.680006e-02 -1.165894e-01 -6.416296e-02 5.484619e-01

Sale.ConditionFamily Sale.ConditionNormal Sale.ConditionPartial ln.Lot.Area

2.282512e-01 3.280775e-01 2.805238e-01 6.551777e-01

Total.Bath tot.sqft

3.104881e-01 6.903916e-04