Code Sample: Analysis of Housing Data of Ames, Iowa

Mingsong Chen

I. Data

The aim of this report is to build several linear regression models based on *AmesHousing* data set and select the best model that will be a good tool to predict house prices based on several parameters. *AmesHousing* data set contains information about individual residential properties sold in Ames, IA from 2006 to 2010.

II. Analysis

There are 82 columns in the *AmesHousing* data set, including 23 nominal, 23 ordinal, 14 discrete, and 20 continuous variables. However, this analysis will only focus on the 20 continuous variables.

All the following analyses were completed in RStudio.

Before starting the analysis, the preprocessing of the data set was made. All continuous variables were assigned to a new data frame. Then, all observations with missing values were removed from the data frame. Also, according to the notes from the data set, 5 observations were either outliers or unusual observations. They were checked and then removed. Several exploratory data analyses were performed. For example, correlations between the variables were checked. Also, boxplots and statistic summary were created.

Also, several variables were either removed or combined. The correlation check showed a strong correlation between variable *Total.Bsmt.SF* and *x1st.Flr.SF*. Also, a model fit using *SalePrice* against all other 19 variables was created. It showed severe singularity issues on *Total.Bsmt.SF* and *Gr.Liv.Are*a. And therefore, the variables were removed from the future model fit. Also, *BsmtFin.SF.1* and *BsmtFin.SF.2* were combined into a new variable *BsmtFin.SF* by summing up. Similarly, *X1st.Flr.SF* and *X2nd.Flr.SF* were merged into *Flr.SF* and *Open.Porch.SF* and *Enclosed.Porch* were combined into *Porch*. This was made based on the meanings of the variables, in order to solve the issue of collinearity and reduce the number of predictors. There were 14 predictors after the combination and removal.

The remaining data set was split into a training data set and a testing data set. There were 609 observations in the testing data set after removing observations with missing values.

Then, model analysis using the training data set began. Firstly, a model fit of the training data set using all the predictors was made. Several diagnostic procedures of the model were made. Large leverage points were checked, studentized residuals were calculated and outliers were removed according to Cook's Distance. By this point, there were 1798 observations in the training data set. Then, after another model fit using the training data after the outlier removal, the three assumptions of the linear regression model were tested. Model fits with log and square root transformations of the response were performed. And since square root transformation solved the issue of non-linearity and non-equivariance better, all the following model fits and selections were performed with a square root transformation of the response.

Several model fits were carried out, including a GLS model. However, this analysis focused on comparing between the following 5 models: (i) the full model, (ii) the sub-model chosen by AIC, (iii) the sub-model chosen by BIC, (iv) the sub-model chosen by Lasso, and (v) the sub-model chosen by the Elastic Net. When choosing the best alpha value to fit the Elastic Net model, mean square errors (MSE) across 11 different alpha values, namely, from 0 to 1 at an increment of 0.1, were calculated. And then, the alpha value with the smallest MSE were selected to do the model fit using *glmnet* function. The variables with a non-zero beta were the variables selected by the Elastic Net.

Finally, root mean square errors (RMSE) were calculated for each model using the testing dataset. A smaller RMSE represents smaller prediction errors. However, the final model selection was based on both RMSE and the number of predictors used.

III. Results

Only findings that were essential to draw the conclusion were presented in this section, other major findings and the complete R code are presented in the *Appendix* section.

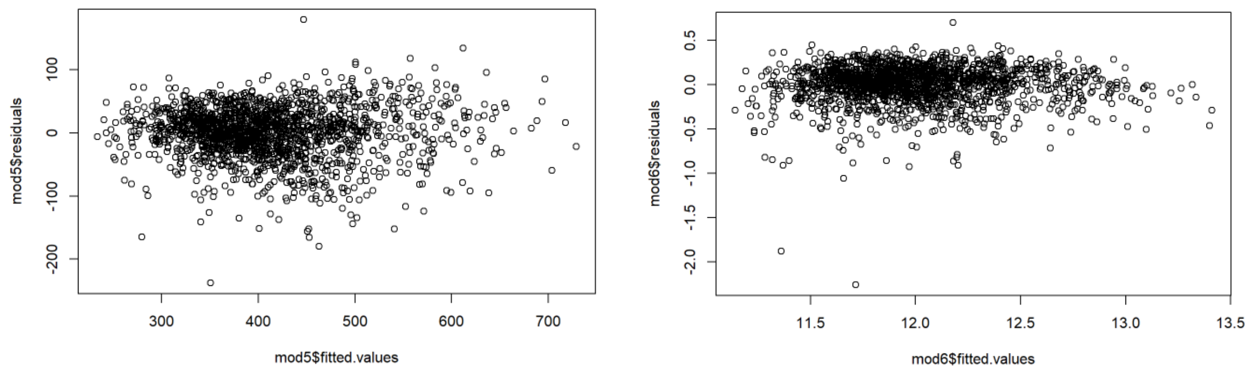

Figure 1 Square root transformation of the response (left) vs. log transformation of the response (right). Both transformations result in non-linear residuals, however, square root transformation results in residuals that are follows the assumption of equivariance better than log transformation.

Residuals vs. fitted values plots for square root transformation and log transformation is presented in Figure 1. Solution path and MSE vs. log(λ) plots for Lasso and Elastic Nets are presented in Figure 2. RMSE of each model and the number of predictors used are presented in Table 1.

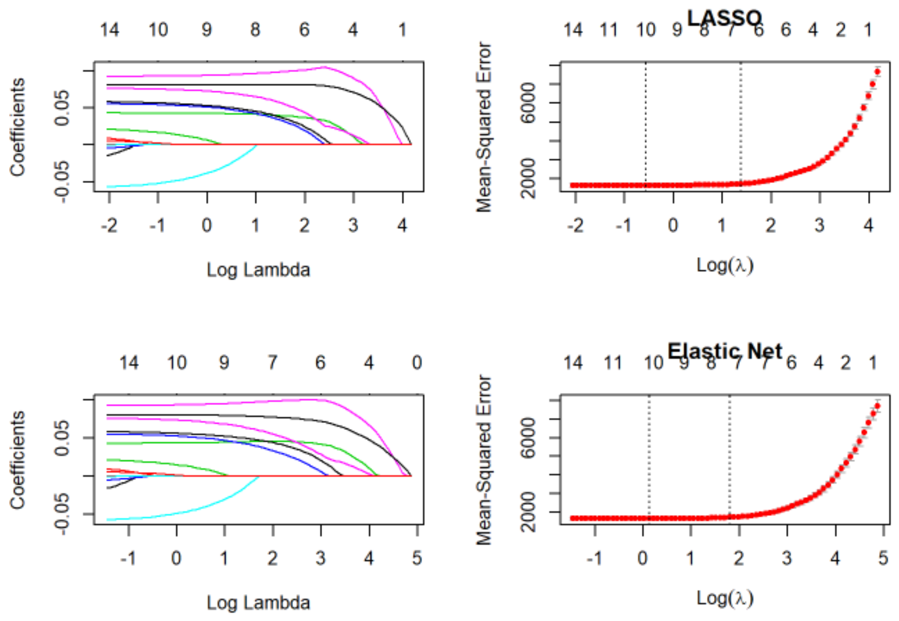

Figure 2 Solution path (left) and MSE vs. log(λ) (right) plots for Lasso (top) and Elastic Nets (bottom). Both Lasso and the Elastic Nets select variables since as log(λ) increases, some predictors shrink to zero while some do not.

Table 1 MSE of each model and the number of predictors used

	Full	AIC	BIC	Lasso	The Elastic Nets (α = 0.8)
Num. of Pred.	14	8	6	6	7
RMSE	35387.92	35518.68	35336.16	35336.16	35143.82

Both BIC and Lasso choose the same model. The Elastic Nests model has the smallest RMSE and uses a fair number of predictors. Thus, it is the model selected.

IV. Conclusion

This report selects a linear model using the predictors selected by the Elastic Nets that can be used to predict the sale price of houses. The model may serve as a reference when people try to measure if the price of houses they want is reasonable as compared to other houses.

A problem with this model is that the assumption of normality is not satisfied as indicated by a small p-value of the Shapiro-Wilks Test even after serval transformation attempts. Therefore, a linear model may not be sufficient. Model fits other than linear models may be applied in future works.

Appendix I. Important Figures

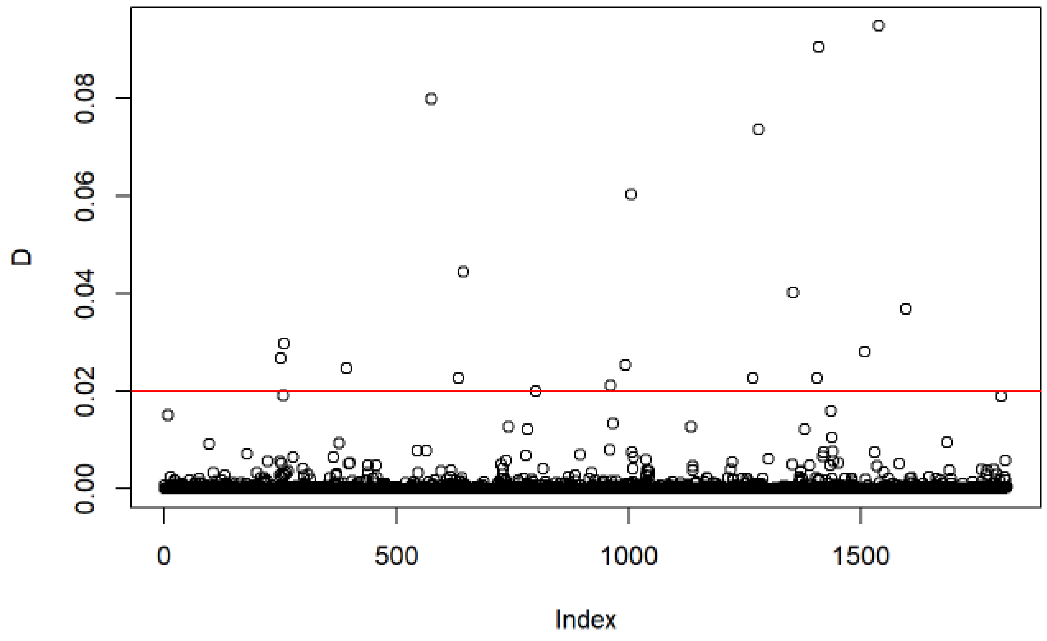

Figure 3 Cook's Distance. Note here 0.02 is used as the cutoff and observations with a Cook's Distance greater than 0.02 are removed.

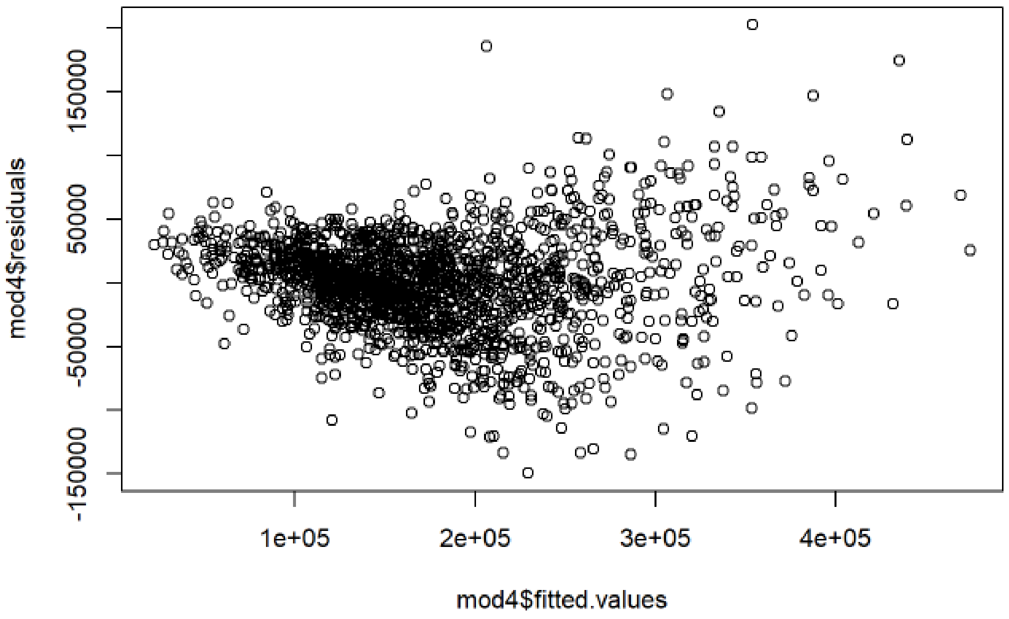

Figure 4 Residual vs. fitted values without transformation. Both linearity and equivariance assumptions are violated.

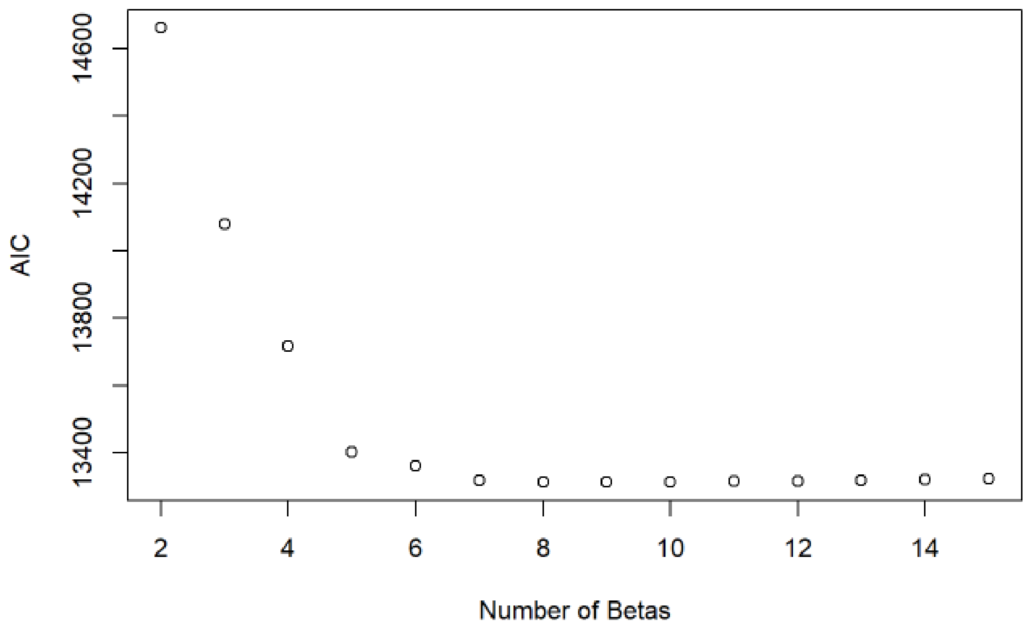

Figure 5 AIC value vs. number of betas. When number of betas is 8, AIC reaches its minimum.

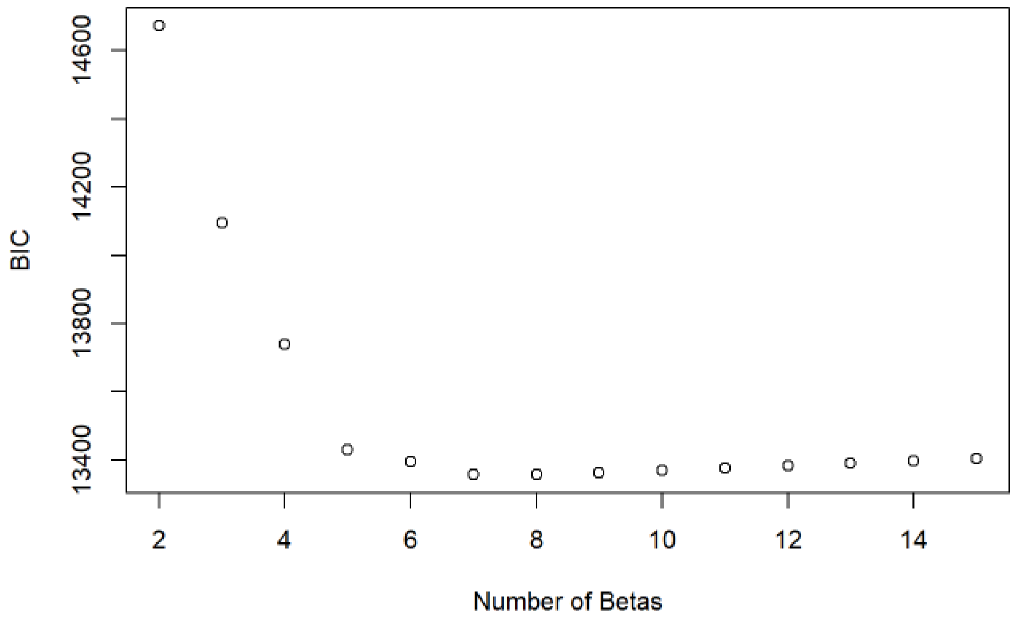

Figure 6 BIC value vs. number of betas. When number of betas is 6, BIC reaches its minimum.

Appendix II. The Complete R Code

~~~
```{r}
setwd('D:/R')
#Read in data set
housing <- read.delim('AmesHousing.txt', header = TRUE)
#20 continuous variables
df_cont <- data.frame(housing$Lot.Frontage, housing$Lot.Area, housing$Mas.Vnr.Area,
housing$BsmtFin.SF.1, housing$BsmtFin.SF.2,
                      housing$Bsmt.Unf.SF, housing$Total.Bsmt.SF, housing$X1st.Flr.SF,
housing$X2nd.Flr.SF, housing$Low.Qual.Fin.SF,
                      housing$Gr.Liv.Area, housing$Garage.Area, housing$Wood.Deck.SF,
housing$Open.Porch.SF, housing$Enclosed.Porch,
                      housing$X3Ssn.Porch, housing$Screen.Porch, housing$Pool.Area,
housing$Misc.Val, housing$SalePrice)
#Check for missing values and remove them
df <- na.omit(df_cont)
plot(x=df$housing.Gr.Liv.Area, y=df$housing.SalePrice)
df_1 <- df[-which(df$housing.Gr.Liv.Area>4000), ]
#Preporocessing
summary(df_1)
boxplot(df_1$housing.SalePrice)
#Check for collineraity issues
cor(df_1)
#Fit LR with all predictors
mod1 <- lm(housing.SalePrice ~ ., data = df_1)
summary(mod1)
#Combine variables
df_1$housing.BsmtFin.SF <- df_1$housing.BsmtFin.SF.1+df_1$housing.BsmtFin.SF.2
df_1$housing.Flr.SF <- df_1$housing.X1st.Flr.SF + df_1$housing.X2nd.Flr.SF
df_1$housing.Porch <- df_1$housing.Open.Porch.SF + df_1$housing.Enclosed.Porch
~~~

```
#Test set
set.seed(2020)
testindices = sample(2930, round(2930/4)) ## train indices are the rest
df.test_1 <- df_1[testindices,]
df.test <- na.omit(df.test_1)
y.test <- df.test$housing.SalePrice
df_1.train <- df_1[-testindices, ]
#Remove Total.Bsmt.SF and Gr.Liv.Area
mod2 <- lm(housing.SalePrice ~ housing.Lot.Frontage + housing.Lot.Area +
housing.Mas.Vnr.Area + housing.BsmtFin.SF.1 + housing.BsmtFin.SF.2 +
        housing.Bsmt.Unf.SF + housing.X1st.Flr.SF + housing.X2nd.Flr.SF +
housing.Low.Qual.Fin.SF +
        housing.Garage.Area + housing.Wood.Deck.SF + housing.Open.Porch.SF +
housing.Enclosed.Porch +
        housing.X3Ssn.Porch + housing.Screen.Porch + housing.Pool.Area + housing.Misc.Val,
data = df_1.train)
summary(mod2)
#Use the new variables
mod3 <- lm(housing.SalePrice ~ housing.Lot.Frontage + housing.Lot.Area +
housing.Mas.Vnr.Area + housing.BsmtFin.SF +
        housing.Bsmt.Unf.SF + housing.Flr.SF + housing.Low.Qual.Fin.SF +
housing.Garage.Area + housing.Wood.Deck.SF +
        housing.Porch +housing.X3Ssn.Porch + housing.Screen.Porch + housing.Pool.Area +
housing.Misc.Val, data = df_1.train)
summary(mod3)
#Re-check collinearity
library(car)
vif(mod2) #All less than 4, no collinearity
vif(mod3)

# Check for large leverage points
```

```
p = length(coefficients(mod3))
n = nrow(df_1.train)
e_hat = mod2$residuals #residual error estimate
sigma2_hat = summary(mod3)$sigma^2 #sigma2 estimate
X = model.matrix(mod3)
H = X %*% solve(t(X) %*% X) %*% t(X) #construct the Hat matrix using design matrix --->
leverage
h = diag(H) #diagonal element
thresh1 = 2*p / n
thresh2 = 3*p / n #thresholds
plot(h) #scatterplot
abline(h = thresh1, col = 'red')
abline(h = thresh2, col = 'blue')
unname(which(h >= thresh2))

# Calculate studentized residuals
r = e_hat / sqrt(sigma2_hat * (1-h)) #studentized residuals
t = r * sqrt((n-p-1) / (n-p-r)) # Studentized deleted residuals
plot(t) #Scatter plot involving the studentized deleted residuals
abline(h = 2, col = "red") #Use cut-off point |t_i| >= 3 here
abline(h = -2, col = "blue")
unname(which(t>2))
unname(which(t< -2))

# Cook's Distance
D = (1/p) * r^2 *(h/(1-h))
plot(D)
abline(h = 0.02, col = 'red')
unname(which(D>0.02))

df_2.train <- df_1.train[-which(D>0.02), ]
```

```
mod4 <- lm(housing.SalePrice ~ housing.Lot.Frontage + housing.Lot.Area +
housing.Mas.Vnr.Area + housing.BsmtFin.SF +
            housing.Bsmt.Unf.SF + housing.Flr.SF + housing.Low.Qual.Fin.SF +
housing.Garage.Area + housing.Wood.Deck.SF +
            housing.Porch +housing.X3Ssn.Porch + housing.Screen.Porch + housing.Pool.Area +
housing.Misc.Val, data = df_2.train)
summary(mod4)

#Linearity
plot(x = mod4$fitted.values, y = mod4$residuals) #residual vs. fitted values plot

#Normality
# QQ Plot:
qqnorm(mod4$residuals)
qqline(mod4$residuals)

# Shapiro-Wilks Test
shapiro.test(mod4$residuals)

#transformation of the response:
#sqrt
mod5 <- lm(sqrt(housing.SalePrice) ~ housing.Lot.Frontage + housing.Lot.Area +
housing.Mas.Vnr.Area + housing.BsmtFin.SF +
            housing.Bsmt.Unf.SF + housing.Flr.SF + housing.Low.Qual.Fin.SF +
housing.Garage.Area + housing.Wood.Deck.SF +
            housing.Porch +housing.X3Ssn.Porch + housing.Screen.Porch + housing.Pool.Area +
housing.Misc.Val, data = df_2.train)
summary(mod5)
#Linearity
plot(x = mod5$fitted.values, y = mod5$residuals) #residual vs. fitted values plot
```

```
#Normality
# QQ Plot:
qqnorm(mod5$residuals)
qqline(mod5$residuals)

# Shapiro-Wilks Test
shapiro.test(mod5$residuals)

#log
mod6 <- lm(log(housing.SalePrice) ~ housing.Lot.Frontage + housing.Lot.Area +
housing.Mas.Vnr.Area + housing.BsmtFin.SF +
        housing.Bsmt.Unf.SF + housing.Flr.SF + housing.Low.Qual.Fin.SF +
housing.Garage.Area + housing.Wood.Deck.SF +
        housing.Porch +housing.X3Ssn.Porch + housing.Screen.Porch + housing.Pool.Area +
housing.Misc.Val, data = df_2.train)
summary(mod6)
#Linearity
plot(x = mod6$fitted.values, y = mod6$residuals) #residual vs. fitted values plot

#Normality
# QQ Plot:
qqnorm(mod6$residuals)
qqline(mod6$residuals)

# Shapiro-Wilks Test
shapiro.test(mod6$residuals)

#GLS model
library(nlme)
```

```
mod7 <- gls(sqrt(housing.SalePrice) ~ housing.Lot.Frontage + housing.Lot.Area +
housing.Mas.Vnr.Area + housing.BsmtFin.SF + housing.Bsmt.Unf.SF + housing.Flr.SF +
housing.Low.Qual.Fin.SF + housing.Garage.Area + housing.Wood.Deck.SF + housing.Porch
+housing.X3Ssn.Porch + housing.Screen.Porch + housing.Pool.Area + housing.Misc.Val, data =
df_2.train)
summary(mod7)

#Full model: 14 predictors (mod5)
# Compute the AIC
library(leaps)
sub_fit <- regsubsets(sqrt(housing.SalePrice) ~ housing.Lot.Frontage + housing.Lot.Area +
housing.Mas.Vnr.Area + housing.BsmtFin.SF + housing.Bsmt.Unf.SF + housing.Flr.SF +
housing.Low.Qual.Fin.SF + housing.Garage.Area + housing.Wood.Deck.SF + housing.Porch
+housing.X3Ssn.Porch + housing.Screen.Porch + housing.Pool.Area + housing.Misc.Val, data =
df_2.train, nvmax=14)
sum_fit <- summary(sub_fit)
sum_fit$which
TSS = sum((sqrt(df_2.train$housing.SalePrice) - mean(sqrt(df_2.train$housing.SalePrice)))^2)
RSS_all = sum_fit$rss
p_full = 15
p_all= 2:15
n = nrow(df_2.train)
AIC_all <- n * log(RSS_all/n) + 2*p_all
plot(x = p_all, y = AIC_all,
     xlab="Number of Betas",
     ylab="AIC")
which(AIC_all == min(AIC_all))
mod_AIC <- step(mod5, direction = 'both', k = 2)
summary(mod_AIC)
#AIC: 8 predictors
```

```
# Compute the BIC
BIC_all <- n * log(RSS_all/n) + log(n)*p_all
plot(x = p_all, y = BIC_all,
     xlab="Number of Betas",
     ylab="BIC")
which(BIC_all == min(BIC_all))
mod_BIC <- step(mod5, direction = 'both', k = log(n))
summary(mod_BIC)
#BIC: 6 predictors

#LASSO:
library(glmnet) # glmnet()
y <- sqrt(df_2.train$housing.SalePrice)
ddf <- data.frame(df_2.train[, 1:3], df_2.train[, 6],df_2.train[, 10],df_2.train[,
12:13],df_2.train[,16:19],df_2.train[,21:23])
names(ddf)[4] <- "housing.Bsmt.Unf.SF"
names(ddf)[5] <- "housing.Low.Qual.Fin.SF"
X <- as.matrix(ddf)
lasso_fit <- glmnet(X, y, alpha = 1)
plot(lasso_fit)
k = 10 # 10-fold validation
cv_fit <- cv.glmnet(X, y, nfolds = k)
# Plot MSE v.s. log (lambda)
plot(cv_fit)
# lasso_fit$lambda
# log(lasso_fit$lambda)
# Plot SD v.s. lambda
plot(x = cv_fit$lambda, y = cv_fit$cvsd)
# Do the same thing, but now using lambda.1se. Compare the estimates with each other as well
as with the least square estimate and the ridge estimate.
lasso_1se = glmnet(X, y, lambda = cv_fit$lambda.1se, alpha = 1)
```

```
lasso_1se$beta
# Take the predictors selected by lasso with "lambda.1se" as the tuning parameter and refit the
model using lm. Comment on any differences with the full model.
mod_L <- lm(sqrt(housing.SalePrice) ~ housing.Mas.Vnr.Area + housing.Bsmt.Unf.SF +
housing.Garage.Area
            + housing.Wood.Deck.SF + housing.BsmtFin.SF + housing.Flr.SF, data = df_2.train)
summary(mod_L)

fit.lasso <- glmnet(X, y, family="gaussian", alpha=1)
fit.elnet <- glmnet(X, y, family="gaussian", alpha=.5)



# 10-fold Cross validation for each alpha = 0, 0.1, ... , 0.9, 1.0
# (For plots on Right)
for (i in 0:10) {
  assign(paste("fit", i, sep=""), cv.glmnet(X, y, type.measure="mse",
                                            alpha=i/10,family="gaussian"))
}

# Plot solution paths:
par(mfrow=c(2,2))
# For plotting options, type '?plot.glmnet' in R console
plot(fit.lasso, xvar="lambda")
plot(fit10, main="LASSO")



plot(fit.elnet, xvar="lambda")
plot(fit5, main="Elastic Net")

x.test1 <- data.frame(df.test[, 1:3], df.test[, 6],df.test[, 10],df.test[,
12:13],df.test[,16:19],df.test[,21:23])
```

```
names(x.test1)[4] <- "housing.Bsmt.Unf.SF"
names(x.test1)[5] <- "housing.Low.Qual.Fin.SF"
x.test <- as.matrix(x.test1)

#Selecting the best alpha for elstic net according to MSE
yhat0 <- predict(fit0, s=fit0$lambda.1se, newx=x.test)
yhat1 <- predict(fit1, s=fit1$lambda.1se, newx=x.test)
yhat2 <- predict(fit2, s=fit2$lambda.1se, newx=x.test)
yhat3 <- predict(fit3, s=fit3$lambda.1se, newx=x.test)
yhat4 <- predict(fit4, s=fit4$lambda.1se, newx=x.test)
yhat5 <- predict(fit5, s=fit5$lambda.1se, newx=x.test)
yhat6 <- predict(fit6, s=fit6$lambda.1se, newx=x.test)
yhat7 <- predict(fit7, s=fit7$lambda.1se, newx=x.test)
yhat8 <- predict(fit8, s=fit8$lambda.1se, newx=x.test)
yhat9 <- predict(fit9, s=fit9$lambda.1se, newx=x.test)
yhat10 <- predict(fit10, s=fit10$lambda.1se, newx=x.test)
mse <- as.vector(1:11)
mse[1] <- mean((y.test - yhat0)^2)
mse[2] <- mean((y.test - yhat1)^2)
mse[3] <- mean((y.test - yhat2)^2)
mse[4] <- mean((y.test - yhat3)^2)
mse[5] <- mean((y.test - yhat4)^2)
mse[6] <- mean((y.test - yhat5)^2)
mse[7] <- mean((y.test - yhat6)^2)
mse[8] <- mean((y.test - yhat7)^2)
mse[9] <- mean((y.test - yhat8)^2)
mse[10] <- mean((y.test - yhat9)^2)
mse[11] <- mean((y.test - yhat10)^2)
which(mse==min(mse))

# Fit Elastic net with alpha = 0.4
```

```
elnet = glmnet(X, y, lambda = cv_fit$lambda.1se, alpha = 0.8)
elnet$beta
# Take the predictors selected by elstic net with "lambda.1se" as the tuning parameter and refit
the model using lm. Comment on any differences with the full model.
mod_E <- lm(sqrt(housing.SalePrice) ~ housing.Lot.Area + housing.Mas.Vnr.Area +
housing.Bsmt.Unf.SF +
            housing.Garage.Area + housing.Wood.Deck.SF + housing.BsmtFin.SF +
housing.Flr.SF, data = df_2.train)
summary(mod_E)

#Mean prediction errors (RMSE):
#1. Full model
sqrt(mean((y.test-(predict(mod5, new = x.test1, interval = 'none'))^2)^2))
#2. AIC model
sqrt(mean((y.test-(predict(mod_AIC, new = x.test1, interval = 'none'))^2)^2))
#3. BIC model
sqrt(mean((y.test-(predict(mod_BIC, new = x.test1, interval = 'none'))^2)^2))
#4. LASSO model
sqrt(mean((y.test-(predict(mod_L, new = x.test1, interval = 'none'))^2)^2))
#5. Elastic net
sqrt(mean((y.test-(predict(mod_E, new = x.test1, interval = 'none'))^2)^2))
```