Kaggle House Price Analysis:

Finding a Correlation and Regression of Leading Contributing Factors

Mohammed Al-Rousan

Robert Lane

Southern Methodist University

Contents

[Q1: House Prices in Ames Iowa 3](#_Toc131854507)

[Q1 Assumptions 3](#_Toc131854508)

[Linear Relationship & Normality 3](#_Toc131854509)

[Linear Relationship 3](#_Toc131854510)

[Multivariate Normality 4](#_Toc131854511)

[No Multicollinearity 4](#_Toc131854512)

[No Autocorrelation 4](#_Toc131854513)

[Homoscedasticity 5](#_Toc131854514)

[Fit Analysis 5](#_Toc131854515)

# Q1: House Prices in Ames Iowa

This question focuses on 3 neighborhoods, marked in the dataset at Names, Edwards, and BrkSide.

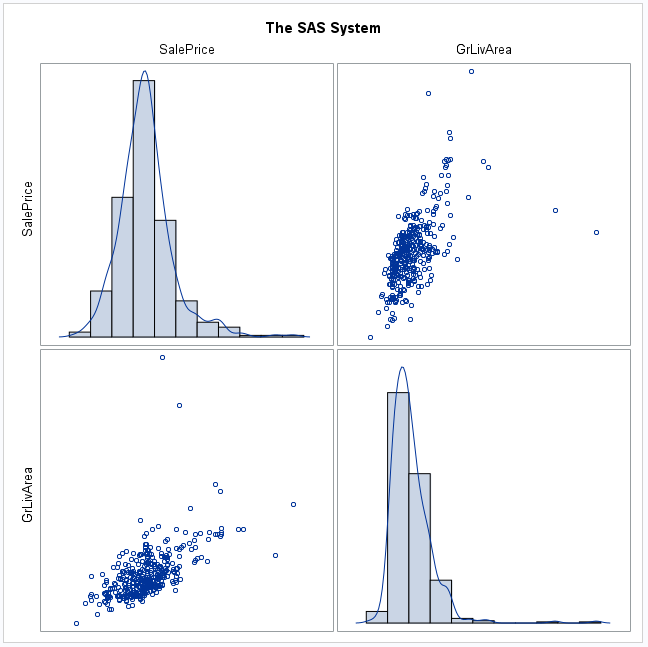
## Q1 Assumptions

### Linear Relationship & Normality

**proc** **sgscatter** data = neighborhoods;

matrix SalePrice GrLivArea / diagonal=(histogram kernel);

**run**;



Fail.

Both the independent and the dependent variables exhibit signs of right-skewness as well as increasing variance. Running a log-log model to attempt to correct for this. Also, coding neighborhoods as dummy variables for later use.

**data** loghood;

set neighborhoods;

logprice = log(SalePrice);

logarea = log(GrLivArea);

BrkSide = (Neighborhood = "BrkSide");

NAmes = (Neighborhood = "NAmes");

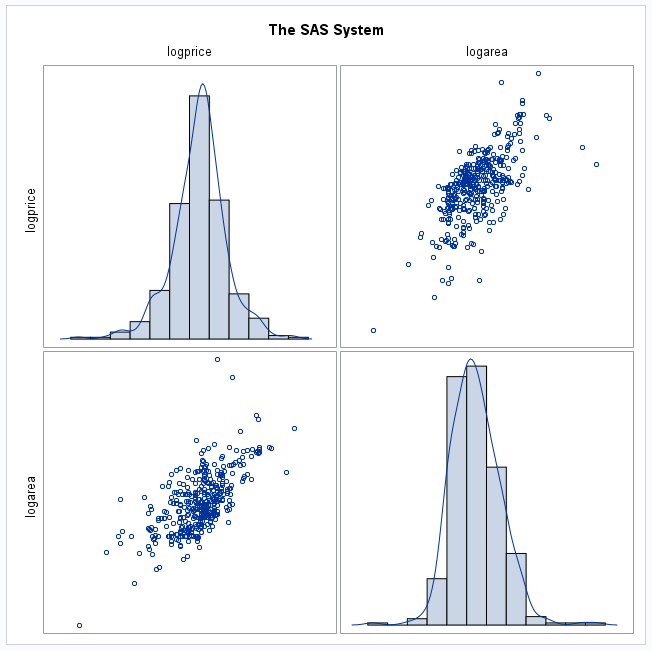
**run**;

### Linear Relationship

**proc** **sgscatter** data = loghood;

matrix logprice logarea / diagonal=(histogram kernel);

**run**;



Pass. The relationship between the log of the area and the log of the price does seem to exhibit a linear relationship.

### Multivariate Normality

Pass. Using the matrix above, it is evident that both inputs have normality.

### No Multicollinearity

Pass. Only one explanatory variable is used here.

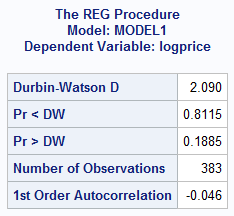
### No Autocorrelation

Running Durbin-Watson test for autocorrelation.

**proc** **reg** data = loghood;

model logprice = logarea BrkSide NAmes / dwProb;

run;



A Durbin-Watson score near 2 indicates that there is almost zero autocorrelation. Pass.

### Homoscedasticity

The variance visually appears to be pretty even at both low and high values, and for each axis. Pass.

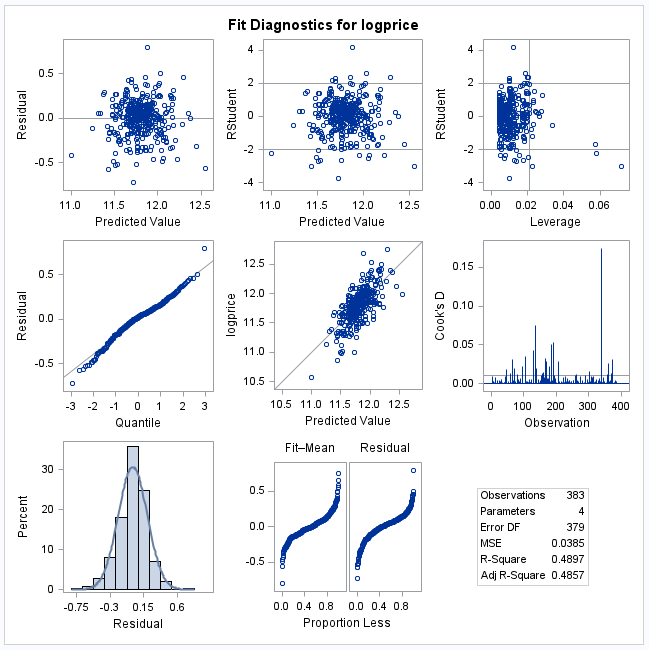
## Fit Analysis

**proc** **reg** data = loghood

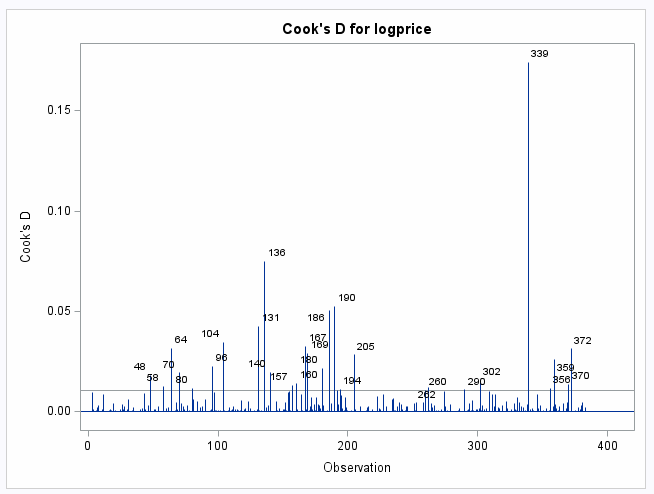
plots = (DiagnosticsPanel ResidualPlot(smooth));

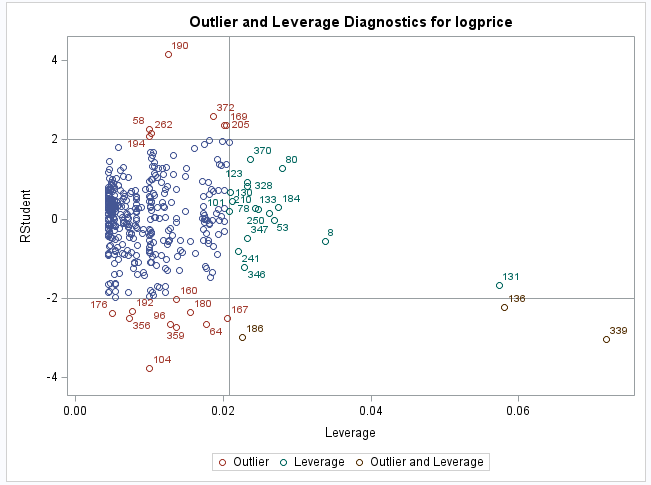
model logprice = logarea BrkSide NAmes;

**quit**;



The histogram and Q-Q plot both indicate that residuals are normally distributed, but the Leverage plot indicates that there are 3 influential outliers which should be checked.





186, 136, and 339 look suspect as influential values.

View just the relevant information for these houses:

**data** temp;

set loghood;

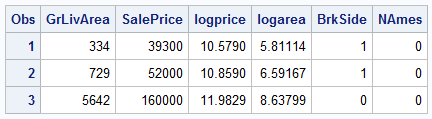
if \_n\_ in (**339**, **186**, **136**);

keep SalePrice GrLivArea logprice logarea BrkSide NAmes;

**run**;

**proc** **print** data=temp(obs=**3**);

**run**;



It seems unreasonable to make much of a prediction for the sale of a livable area of less than 500 square feet. For reference, that would be a living area off less than two typical parking spaces. It further seems unreasonable to expect a house larger than 5,000 square feet to sell at a price of $160,000. Therefore, of these three values, the first and last one (obs = 339 and 136) seem to be unlikely scenarios and may be removed. The middle one, while still an influential outlier, seems at least plausible.

# Appendix

## R Code Used

## library(dplyr)

## library(ggplot2)

## library(car)

## library(caret)

## library(scales)

## library(tidyr)

## library(readr)

## library(purrr)

## library(forcats)

## library(imputeMissings)

## install.packages("imputeMissings")

## library(tidyverse)

## library(leaps)

## install.packages("leaps")

## library(MASS)

## install.packages("olsrr")

## library(olsrr)

## library(asbio)

## install.packages("asbio")

## install.packages("tcltk")

## library(DAAG)

## install.packages("DAAG")

## #Import the data

## train = read.csv(file.choose(), sep = ",")

## test = read.csv(file.choose(), sep = ",")

## #viewing the data

## view(test)

## view(train)

## #Filtering the Data

## trainNeighborhood <- dplyr:: filter(train, Neighborhood == "Edwards" | Neighborhood == "NAmes" | Neighborhood == "BrkSide")

## testNeighborhood <- dplyr:: filter(test, Neighborhood == "Edwards" | Neighborhood == "NAmes" | Neighborhood == "BrkSide")

## #Some EDA

## #Summary of the data

## summary(train)

## summary(test)

## #Basic visualizaton of home prices

## trainHist <- ggplot(trainNeighborhood, aes(x = SalePrice)) + geom\_histogram(color = "black", fill = "red")

## trainHist

## #There is normal distribution for the most part, but it is a bit right leaning

## #Basic visualization of sq footage

## trainBar <- ggplot(trainNeighborhood, aes(x = GrLivArea)) + geom\_histogram(color = "black", fill = "red")

## trainBar

## #This plot is heavily right skewed

## #Basic graphs with log transformations

## trainHitLog <- ggplot(trainNeighborhood, aes(x = log(SalePrice))) + geom\_histogram(color = "black", fill = "red")

## trainHitLog

## # This graph presents normal distribution

## trainBarLog <- ggplot(trainNeighborhood, aes(x = log(GrLivArea))) + geom\_histogram(color = "black", fill = "red")

## trainBarLog

## # This graph presents normal distribution

## #Testing a scatter plot

## trainScatter <- ggplot(trainNeighborhood, aes(x = log(GrLivArea), y = log(SalePrice), color = Neighborhood)) + geom\_point()

## trainScatter

## #Create an LM with outliers

## priceLM <- lm(log(SalePrice) ~ log(GrLivArea) + Neighborhood + Neighborhood + GrLivArea, data = trainNeighborhood)

## summary(priceLM)

## confint(priceLM)

## #Look at ANOVA

## residentANOVA <- aov(log(SalePrice) ~ log(GrLivArea) + Neighborhood + Neighborhood + GrLivArea, data = trainNeighborhood)

## summary(residentANOVA)

## plot(priceLM)

## #There seems to be correlation between home prices and square footage

## #model diagnostics and leverage plots

## leveragePlots(lm(log(SalePrice) ~ log(GrLivArea) + Neighborhood + Neighborhood + GrLivArea, data = trainNeighborhood))

## #Plot Cooks distance

## plot(cooks.distance(leveragePlots(lm(log(SalePrice) ~ log(GrLivArea) + Neighborhood + Neighborhood + GrLivArea, data = trainNeighborhood))

## )

## sort(cooks.distance(leveragePlots(lm(log(SalePrice) ~ log(GrLivArea) + Neighborhood + Neighborhood + GrLivArea, data = trainNeighborhood))

## )

## #Creating the QQ plot

## qqPlot(log(trainNeighborhood$SalePrice))

## #Uneven distribution

## #Histogram of Residuals

## trainHist <- hist(priceLM$residuals, breaks = 10, destiny = 10, col = "lightgray", main = "Residuals")

## xfit <- seq(min(priceLM$residuals), max(priceLM$residuals), length = 40)

## yfit <- dnorm(xfit, mean = mean(priceLM$residuals), sd = sd(priceLM$residuals))

## yfit <- yfit \* diff(h$mids[1:2]) \* length(priceLm$residuals)

## #Create lm without outliers

## cleanData <- trainNeighborhood[-c(169, 190, 339)]

## priceLM2 <- lm(log(SalePrice) ~ log(GrLivArea) + Neighborhood + Neighborhood + GrLivArea, data = cleanData)

## summary(priceLM2)

## confint(priceLM2)

## #Creat an internal CV

## trainIndex <- createDataPartition(trainNeighborhood$SalePrice, p = .8, list = FALSE, times = 1)

## trainSale <- trainNeighborhood[ trainIndex,]

## testSale <- trainNeighborhood[trainIndex, ]

## pred.w.plim <- predict(lm(log(SalePrice)~log(GrLivArea)+Neighborhood+Neighborhood\*GrLivArea,data=trainSale), testSale, interval = "prediction")

## pred.w.clim <- predict(lm(log(SalePrice)~log(GrLivArea)+Neighborhood+Neighborhood\*GrLivArea,data=trainSale), testSale, interval = "confidence")

## matplot(testSale$SalePrice, cbind(pred.w.clim, pred.w.plim[,-1]),

## lty = c(1,2,2,3,3), type = "l", ylab = "predicted y")

## #predict test data set

## pred.w.plim <- predict(priceLM2, testNeighborhood, interval = "prediction")

## view(exp(pred.w.plim))

## view(trainNeighborhood$SalePrice)

## #Look at ANOVA

## res.aov <- aov(log(SalePrice)~log(GrLivArea)+Neighborhood+Neighborhood\*GrLivArea, data = cleanData)

## summary(res.aov)

## #Leverage plots

## leveragePlots(lm(log(SalePrice)~log(GrLivArea)+Neighborhood+Neighborhood\*GrLivArea,data=cleanData))

## #Model diagnostics, Cook's Distance

## plot(cooks.distance(lm(log(SalePrice)~log(GrLivArea)+Neighborhood+Neighborhood\*GrLivArea,data=cleanData)))

sort(cooks.distance(lm(log(SalePrice)~log(GrLivArea)+Neighborhood+Neighborhood\*GrLivArea,data=cleanData)),decreasing = TRUE)

#Model diagnostics, Quantile-Quantile plot

qqPlot(log(cleanData$SalePrice))

#Hist of residuals

h <- hist(priceLM2$residuals, breaks = 10, density = 10,col = "lightgray",main = "Residuals")

xfit <- seq(min(priceLM2$residuals), max(priceLM2$residuals), length = 40)

yfit <- dnorm(xfit, mean = mean(priceLM2$residuals), sd = sd(priceLM2$residuals))

yfit <- yfit \* diff(h$mids[1:2]) \* length(priceLM2$residuals)

CVdat <- CVlm(data = trainNeighborhood, form.lm = formula(log(SalePrice)~log(GrLivArea)+Neighborhood+Neighborhood\*GrLivArea),

m = 3, dots = FALSE, seed = 29, plotit = c("Observed","Residual"),

main="Small symbols show cross-validation predicted values",

legend.pos="topleft", printit = TRUE)

CVdat

(press(priceLm))

### Analysis 2 ####

train2 = read.csv(file.choose(), sep = ",")

test = read.csv(file.choose(), sep = ",")

#Look at outcome variable

ggplot(data = train2 %>% filter(!is.na(log(SalePrice)))) +

geom\_histogram(aes(x = log(SalePrice)), fill = "red", alpha = 1/2, binwidth = 0.01) +

scale\_x\_continuous(labels = dollar\_format()) +

labs(

title = "Outcome Sale Price, right skew"

) +

theme(

plot.title = element\_text(hjust = 0.5, size = 15, face = "bold"),

)

#find missing data

na\_prop <- train2 %>%

dplyr::select(-SalePrice) %>%

map(is.na) %>%

map\_dfr(mean) %>%

pivot\_longer(cols = everything(), names\_to = "variables", values\_to = "prop") %>%

filter(prop > 0) %>%

arrange(desc(prop))

#graphing the data

na\_prop %>%

ggplot(aes(x = fct\_reorder(variables, prop), y = prop, fill = variables)) +

geom\_bar(stat = "identity") +

coord\_flip() +

theme(legend.position = "none") +

labs(

x = "Explanatory variables",

y = "The proportions of NA values per column"

) +

scale\_y\_continuous(breaks = seq(0, 1, by = 0.1)) +

theme(axis.text.y = element\_text(size = 10))

train2$PoolQC[is.na(train2$PoolQC)] <- "None"

train2$MiscFeature[is.na(train2$MiscFeature)] <- "None"

train2$Alley[is.na(train2$Alley)] <- "No"

train2$Fence[is.na(train2$Fence)] <- "No"

train2$FireplaceQu[is.na(train2$FireplaceQu)] <- "No"

train2$GarageType[is.na(train2$GarageType)] <- "No"

train2$GarageFinish[is.na(train2$GarageFinish)] <- "No"

train2$GarageQual[is.na(train2$GarageQual)] <- "No"

train2$GarageCond[is.na(train2$GarageCond)] <- "No"

train2$BsmtExposure[is.na(train2$BsmtExposure)] <- "NoBs"

train2$BsmtCond[is.na(train2$BsmtCond)] <- "NoBs"

train2$BsmtQual[is.na(train2$BsmtQual)] <- "NoBs"

train2$BsmtFinType1[is.na(train2$BsmtFinType1)] <- "NoBs"

train2$BsmtFinType2[is.na(train2$BsmtFinType2)] <- "NoBs"

# To specify the levels of ordered factors

PoolQC\_lev <- c("None", "Fa", "TA", "Gd", "Ex")

Fence\_lev <- c("No", "MnWw", "GdWo", "MnPrv", "GdPrv")

FireplaceQu\_lev <- c("No", "Po", "Fa", "TA", "Gd", "Ex")

GarageFinish\_lev <- c("No", "Unf", "RFn", "Fin")

GarageQual\_lev <- c("No", "Po", "Fa", "TA", "Gd", "Ex")

GarageCond\_lev <- c("No", "Po", "Fa", "TA", "Gd", "Ex")

BsmtExposure\_lev <- c("NoBs", "No", "Mn", "Av", "Gd")

BsmtCond\_lev <- c("NoBs", "Po", "Fa", "TA", "Gd", "Ex")

BsmtQual\_lev <- c("NoBs", "Po", "Fa", "TA", "Gd", "Ex")

BsmtFinType1\_lev <- c("NoBs", "Unf", "LwQ", "Rec", "BLQ", "ALQ", "GLQ")

BsmtFinType2\_lev <- c("NoBs", "Unf", "LwQ", "Rec", "BLQ", "ALQ", "GLQ")

train2analysis <- train2 %>%

mutate(PoolQC = parse\_factor(PoolQC, levels = PoolQC\_lev, ordered = TRUE),

MiscFeature = parse\_factor(MiscFeature),

Alley = parse\_factor(Alley),

Fence = parse\_factor(Fence, levels = Fence\_lev, ordered = TRUE),

FireplaceQu = parse\_factor(FireplaceQu, levels = FireplaceQu\_lev, ordered = TRUE),

GarageType = parse\_factor(GarageType),

GarageFinish = parse\_factor(GarageFinish, levels = GarageFinish\_lev, ordered = TRUE),

GarageQual = parse\_factor(GarageQual, levels = GarageQual\_lev, ordered = TRUE),

GarageCond = parse\_factor(GarageCond, levels = GarageCond\_lev, ordered = TRUE),

BsmtExposure = parse\_factor(BsmtExposure, levels = BsmtExposure\_lev, ordered = TRUE),

BsmtCond = parse\_factor(BsmtCond, levels = BsmtCond\_lev, ordered = TRUE),

BsmtQual = parse\_factor(BsmtQual, levels = BsmtQual\_lev, ordered = TRUE),

BsmtFinType1 = parse\_factor(BsmtFinType1, levels = BsmtFinType1\_lev, ordered = TRUE),

BsmtFinType2 = parse\_factor(BsmtFinType2, levels = BsmtFinType2\_lev, ordered = TRUE))

#lets impute some data

#Col 4 lot frontage

train2analysis[,4][is.na(train2analysis[,4])] <- round(mean(train2analysis[,4], na.rm = TRUE))

#Col 27, massvnr

train2analysis[,27][is.na(train2analysis[,27])] <- round(mean(train2analysis[,27], na.rm = TRUE))

#Col 60, Garage year built

train2analysis[,60][is.na(train2analysis[,60])] <- round(mean(train2analysis[,60], na.rm = TRUE))

#Col 26, MasVnrType

train2analysis$MasVnrType <- train2analysis$MasVnrType %>% tidyr::replace\_na("Stone")

#Col 43, electrical

train2analysis$Electrical <- train2analysis$Electrical %>% tidyr::replace\_na("SBrkr ")

train2analysis$GrLivArea <- log(train2analysis$GrLivArea)

sort(is.na(train2analysis),decreasing = TRUE)

train2analysis[!complete.cases(train2analysis),]

################################################################################

#test dat

#Look at missing data

na\_prop <- test %>%

map(is.na) %>%

map\_dfr(mean) %>%

pivot\_longer(cols = everything(), names\_to = "variables", values\_to = "prop") %>%

filter(prop > 0) %>%

arrange(desc(prop))

na\_prop %>%

ggplot(aes(x = fct\_reorder(variables, prop), y = prop, fill = variables)) +

geom\_bar(stat = "identity") +

coord\_flip() +

theme(legend.position = "none") +

labs(

x = "Explanatory variables",

y = "The proportions of NA values per column"

) +

scale\_y\_continuous(breaks = seq(0, 1, by = 0.1)) +

theme(axis.text.y = element\_text(size = 10))

test$PoolQC[is.na(test$PoolQC)] <- "None"

test$MiscFeature[is.na(test$MiscFeature)] <- "None"

test$Alley[is.na(test$Alley)] <- "No"

test$Fence[is.na(test$Fence)] <- "No"

test$FireplaceQu[is.na(test$FireplaceQu)] <- "No"

test$GarageType[is.na(test$GarageType)] <- "No"

test$GarageFinish[is.na(test$GarageFinish)] <- "No"

test$GarageQual[is.na(test$GarageQual)] <- "No"

test$GarageCond[is.na(test$GarageCond)] <- "No"

test$BsmtExposure[is.na(test$BsmtExposure)] <- "NoBs"

test$BsmtCond[is.na(test$BsmtCond)] <- "NoBs"

test$BsmtQual[is.na(test$BsmtQual)] <- "NoBs"

test$BsmtFinType1[is.na(test$BsmtFinType1)] <- "NoBs"

test$BsmtFinType2[is.na(test$BsmtFinType2)] <- "NoBs"

# To specify the levels of ordered factors

PoolQC\_lev <- c("None", "Fa", "TA", "Gd", "Ex")

Fence\_lev <- c("No", "MnWw", "GdWo", "MnPrv", "GdPrv")

FireplaceQu\_lev <- c("No", "Po", "Fa", "TA", "Gd", "Ex")

GarageFinish\_lev <- c("No", "Unf", "RFn", "Fin")

GarageQual\_lev <- c("No", "Po", "Fa", "TA", "Gd", "Ex")

GarageCond\_lev <- c("No", "Po", "Fa", "TA", "Gd", "Ex")

BsmtExposure\_lev <- c("NoBs", "No", "Mn", "Av", "Gd")

BsmtCond\_lev <- c("NoBs", "Po", "Fa", "TA", "Gd", "Ex")

BsmtQual\_lev <- c("NoBs", "Po", "Fa", "TA", "Gd", "Ex")

BsmtFinType1\_lev <- c("NoBs", "Unf", "LwQ", "Rec", "BLQ", "ALQ", "GLQ")

BsmtFinType2\_lev <- c("NoBs", "Unf", "LwQ", "Rec", "BLQ", "ALQ", "GLQ")

test2 <- test %>%

mutate(PoolQC = parse\_factor(PoolQC, levels = PoolQC\_lev, ordered = TRUE),

MiscFeature = parse\_factor(MiscFeature),

Alley = parse\_factor(Alley),

Fence = parse\_factor(Fence, levels = Fence\_lev, ordered = TRUE),

FireplaceQu = parse\_factor(FireplaceQu, levels = FireplaceQu\_lev, ordered = TRUE),

GarageType = parse\_factor(GarageType),

GarageFinish = parse\_factor(GarageFinish, levels = GarageFinish\_lev, ordered = TRUE),

GarageQual = parse\_factor(GarageQual, levels = GarageQual\_lev, ordered = TRUE),

GarageCond = parse\_factor(GarageCond, levels = GarageCond\_lev, ordered = TRUE),

BsmtExposure = parse\_factor(BsmtExposure, levels = BsmtExposure\_lev, ordered = TRUE),

BsmtCond = parse\_factor(BsmtCond, levels = BsmtCond\_lev, ordered = TRUE),

BsmtQual = parse\_factor(BsmtQual, levels = BsmtQual\_lev, ordered = TRUE),

BsmtFinType1 = parse\_factor(BsmtFinType1, levels = BsmtFinType1\_lev, ordered = TRUE),

BsmtFinType2 = parse\_factor(BsmtFinType2, levels = BsmtFinType2\_lev, ordered = TRUE))

#lets impute some data

#Col 4 lot frontage

test2[,4][is.na(test2[,4])] <- round(mean(test2[,4], na.rm = TRUE))

#Col 27, massvnr

test2[,27][is.na(test2[,27])] <- round(mean(test2[,27], na.rm = TRUE))

#Col 60, Garage year built

test2[,60][is.na(test2[,60])] <- round(mean(test2[,60], na.rm = TRUE))

#Col 26, MasVnrType

test2$MasVnrType <- test2$MasVnrType %>% tidyr::replace\_na("Stone")

#Col 43, electrical

test2$Electrical <- test2$Electrical %>% tidyr::replace\_na("SBrkr ")

test2$GrLivArea <- log(test2$GrLivArea)

#Building Models

#Full model

full.model <- lm(log(SalePrice)~.,data = train2analysis)

#Stepwise model

step.model <- stepAIC(full.model,direction = "both",trace = FALSE)

#Model summary

step.model$pred

models <- regsubsets(log(SalePrice)~., data = train2analysis, nvmax = 1,

method = "seqrep")

summary(models)

set.seed(123)

# Set up repeated k-fold cross-validation

train.control <- trainControl(method = "cv", number = 10)

# Train the model

step.model <- train(log(SalePrice) ~., data = train2analysis,

method = "leapBackward",

tuneGrid = data.frame(nvmax = 1:75),

trControl = train.control

)

step.model$results

step.model$bestTune

stepLm <- lm(log(SalePrice)~OverallQual+OverallCond+YearBuilt+BsmtFinType2+KitchenAbvGr+GarageCond,data = train2analysis)

summary(stepLm)

summary(step.model$finalModel)

coef(step.model$finalModel, 6)

press(stepLm)

plot(stepLm)

#Model diagnostics, leverage plots

leveragePlots(stepLm,data=train2analysis)

#Model diagnostics, Cook's Distance

plot(cooks.distance(stepLm,data=train2analysis))

###############

min.model = lm(log(SalePrice) ~ 1, data=train2analysis)

biggest <- formula(lm(log(SalePrice)~.,train2analysis))

biggest

fwd.model = step(min.model, direction='forward', scope=biggest)

summary(fwd.model)

forwardlm <- lm(log(SalePrice)~GrLivArea+Neighborhood+GarageCars+OverallCond+HouseStyle+YearBuilt+RoofMatl+BsmtFinSF1+MSZoning+Functional+Condition1+SaleCondition+KitchenQual+LotArea+Condition1+Exterior1st+ScreenPorch+Heating+LandSlope+WoodDeckSF+TotalBsmtSF+LotConfig+CentralAir+GarageQual+BsmtFullBath+Fireplaces+X2ndFlrSF+YearRemodAdd+GarageArea+Foundation+LotFrontage+KitchenAbvGr+GarageCond+SaleType+ExterCond+Street+HalfBath,data = train2analysis)

summary(forwardlm)

plot(forwardlm)

#Model diagnostics, leverage plots

leveragePlots(forwardlm,data=train2analysis)

#Model diagnostics, Cook's Distance

plot(cooks.distance(forwardlm,data=train2analysis))

#predict test data set

forward.lm.plim <- predict(forwardlm, test2, interval = "prediction")

####Write forward model

forward.lm.plim <- forward.lm.plim[,1]

forward.lm.plim <- as.data.frame(forward.lm.plim)

forward.lm.plim <- exp(forward.lm.plim[,1])

forward.lm.plim <- forward.lm.plim %>% rename(SalePrice = forward.lm.plim,)

forward.lm.plim[,1][is.na(forward.lm.plim[,1])] <- round(mean(forward.lm.plim[,1], na.rm = TRUE))

out <- write.csv(forward.lm.plim,"forwardModel.csv")

####Write Step model

stepWise.lm.plim <- predict(step.model, test2, interval = "prediction")

stepWise.lm.plim <- exp(stepWise.lm.plim)

stepWise.lm.plim <- as.data.frame(stepWise.lm.plim)

stepWise.lm.plim <- stepWise.lm.plim %>% rename(SalePrice = stepWise.lm.plim,)

stepWise.lm.plim$ID <- seq.int(nrow(stepWise.lm.plim))

outStepwise <- write.csv(stepWise.lm.plim,"stepwiseModel.csv")

View(stepWise.lm.plim)

# Set seed for reproducibility

set.seed(123)

# Set up repeated k-fold cross-validation

train.control <- trainControl(method = "cv", number = 10)

# Train the model

step.model <- train(log(SalePrice) ~., data = train2analysis,

method = "leapBackward",

tuneGrid = data.frame(nvmax = 1:10),

trControl = train.control

)

step.model$results

summary(step.model$finalModel)

coef(step.model$finalModel, 7)

backlm <- lm(log(SalePrice)~OverallQual+OverallCond+YearBuilt+RoofMatl+BsmtFinType2+KitchenAbvGr+GarageCond,data = train2analysis)

summary(backlm)

#predict test data set

back.lm.plim <- predict(backlm, test2, interval = "prediction")

back.lm.plim <- back.lm.plim[,1]

back.lm.plim <- exp(back.lm.plim)

back.lm.plim <- as.data.frame(back.lm.plim)

back.lm.plim <- back.lm.plim %>% rename(SalePrice = back.lm.plim,)

back.lm.plim$ID <- seq.int(nrow(back.lm.plim))

outBack <- write.csv(backModel,"backModel1.csv")

## SAS Code Used

PROC IMPORT OUT= WORK.house

DATAFILE= "C:\Study Files\SMU MSDS\DS 6371 Statistical Foundations for Data Science\statistical-housing-price-analysis\Resources\train.csv"

DBMS=CSV REPLACE;

GETNAMES=YES;

DATAROW=2;

RUN;

proc print data = house;

run;

/\*Select only the 3 neighborhoods of interest\*/

data neighborhoods;

set house;

where Neighborhood in ("NAmes", "Edwards", "BrkSide");

run;

/\*Check for linearity and multivariate normality\*/

proc sgscatter data = neighborhoods;

matrix SalePrice GrLivArea / diagonal=(histogram kernel);

run;

proc print data = neighborhoods;

run;

proc sgscatter data = loghood;

matrix logprice logarea / diagonal=(histogram kernel);

run;

proc glm data = neighborhoods;

model SalePrice = GrLivArea;

run;

proc sgplot data = neighborhoods;

scatter x=GrLivArea y=SalePrice;

run;

data loghood;

set neighborhoods;

logprice = log(SalePrice);

logarea = log(GrLivArea);

BrkSide = (Neighborhood = "BrkSide");

NAmes = (Neighborhood = "NAmes");

run;

proc print data = loghood;

run;

proc sgplot data = loghood;

scatter x=logarea y=logprice;

run;

proc glm data = loghood;

model logprice = logarea;

run;

proc sgscatter data = loghood;

matrix logprice logarea / diagonal=(histogram kernel);

run;

/\*Residual Analysis\*/

proc reg data = loghood

plots = (DiagnosticsPanel ResidualPlot(smooth));

model logprice = logarea BrkSide NAmes;

quit;

/\*

Durbin-Watson test for autocorrelation

http://documentation.sas.com/doc/en/pgmsascdc/9.4\_3.4/statug/statug\_reg\_details33.htm

\*/

proc reg data = loghood;

model logprice = logarea BrkSide NAmes / dwProb;

run;

/\*

Include info on high-leverage nad outlier values

https://blogs.sas.com/content/iml/2021/03/29/influential-obs-regression.html

\*/

proc reg data = loghood plots(only label) = (CooksD RStudentByLeverage);

model logprice = logarea BrkSide NAmes;

run;

/\*View specific observations\*/

data temp;

set loghood;

if \_n\_ in (339, 186, 136);

keep SalePrice GrLivArea logprice logarea BrkSide NAmes;

run;

proc print data=temp;

run;

data loghood2;

set loghood;

if \_n\_ in (339, 136) then delete;

run;

proc glm data=loghood2;

class BrkSide NAmes;

model logprice = logarea BrkSide NAmes logarea\*BrkSide logarea\*NAmes / solution;

means BrkSide NAmes / hovtest=0;

output out=glm\_out p=pred r=resid student=rstudent;

output out=diagnostics residual=residual;

run;