Ames, Iowa House Price Analysis:

Finding a Correlation and Regression of Leading Contributing Factors

Mohammed Al-Rousan

Robert Lane

Southern Methodist University

DS 6371

Company Introduction:

Welcome to Rousan-Lane Consultancy Group, where we provide services and expertise to client organizations to help them improve their business performance. Our work focuses on operations, strategy, management, IT, and finance. At Rousan Consultancy Group we provide top-tier services leveraging data science in improving business solutions and operations leveraging data.

Introduction

Rousan-Lane Consultancy Group has been contracted by real estate company Century 21's branch in Ames, Iowa to estimate the sales price of houses on their market based on square footage of the living area. In particular, we have been hired to look at the homes located in specific neighborhoods, marked in the dataset as NAmes, Edwards, and BrkSide. In conclusion, we are hired to estimate home prices using regression models, prediction models, and confidence intervals.

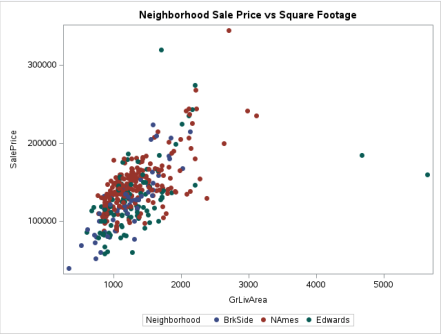
Data Description

The data was compiled by Dean De Cook from the Ames Housing dataset. This data set is a nice set for data scientists to have an observation on how to observe and test the house marking. We were provided a test and train data and each provided different totals and variables. The test dataset contains 1,459 rows and 80 columns, while the train contains 1,460 rows and 81 columns in the dataset.

Analysis 1

Restatement of the Problem

Real estate company Century 21's branch in Ames, Iowa has hired our company to estimate sales prices of the neighborhoods NAmes, Edwards, and BrkSide. We were provided a data set that contained past sales of 1460 homes as well as 80 associated variables.

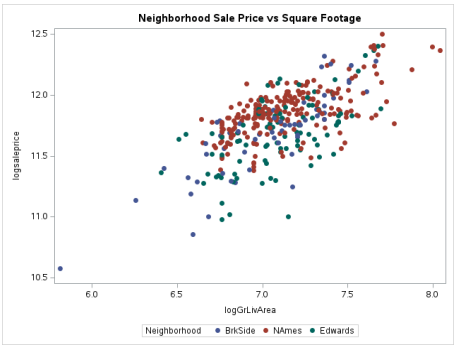


Data Observations

In order to commence, we must filter the data with the specific neighborhoods tasked in NAmes, Edwards, and BrkSide and it can be concluded that there will be 383 points.

Based on the original observation of the scatterplot, it presents a very right-skewed group, and since the graph will be right-skewed, it should be easy to assess that the equal standard deviations will be off. In order to correct this observation, we must log the data, in particular the sale price.

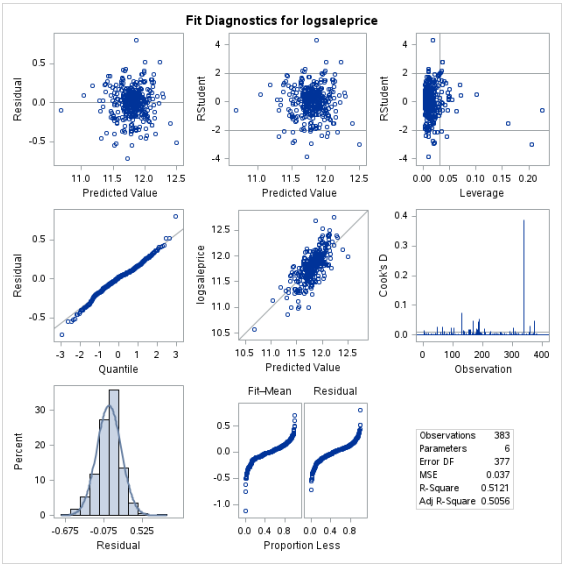
Once we filtered the Sale Price, the scatter plot does present a cloudy normality with the exception of a few outliers.



Build and Fit the Model

Model 1:

$\mu\log((\text{SalePrice})) = \beta_0 + \beta_1 \log((\text{GrLivArea})) + \beta_2 (\log(\text{GrLivArea})*\text{Neighborhood})$



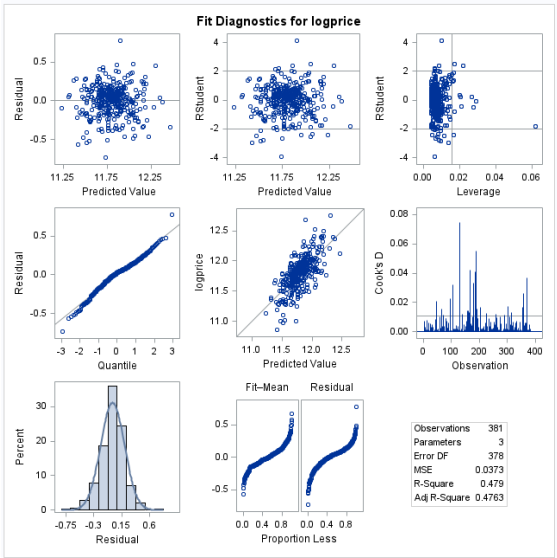
Assumptions

- Linearity
  - By looking at the Q-Q plot, it can be observed that the chart displays a strong sign of normality.
- Multivariate Normality
  - The Q-Q plot and histogram each show strong signs of normality.
- No or low multicollinearity
  - This model only has 1 non-categorical variable.
- No autocorrelation
  - No variable is directly related to the output of SalePrice.
- Homoscedasticity
  - Variance appears to be randomly and evenly spread throughout the scatterplots.

Model 1 results in an R^2 value of 0.5207, with an adjusted R^2 value of 0.5220. The Cross-Validation (CVPRESS) results in a score of 13.27.

Model 2:

$\text{logprice} = \beta_0 + \beta_1 \text{logarea} + \beta_1 \text{NAmes}$
(where the reference neighborhood is Edwards)



Assumptions

- Linearity
  - By looking at the quantile chart, it can be observed that the chart displays a linear tendency.
- Multivariate Normality
  - The Q-Q plot and histogram each show strong signs of normality.
- No or low multicollinearity
  - This model only has 1 non-categorical variable.
- No autocorrelation
  - No variable is directly related to the output of SalePrice.
- Homoscedasticity
  - Variance appears to be randomly and evenly spread throughout the scatterplots.

Model 2 results in an R^2 value of 0.479, with an adjusted R^2 value of 0.476. The Cross-Validation (CVPRESS) results in a score of 14.35.

Comparing the models

Both models do present normal distributions in their residual plots and histogram, linearity in their Q-Q plots, ultimately we decided model one would be a better model due to a lower CV Press and a higher Adjusted R^2.

Parameter Estimates (Model 1)

	Estimate	Standard Error	Confidence Interval
Intercept	8.655	0.31252	8.040933, 9.270024
Neighborhood BrkSide	-2.74255	0.57131	-3.856597, -1.61913
NeighborhoodEdwards	-1.2297	0.57954	-2.36930, -0.090114
logGrLivArea	0.44987	0.04378	0.36377, 0.53597
logGrLiveArea*Neighborhood BrkSide	0.36976	0.080785	0.21091, 0.52862
logGrLiveArea*Neighborhood Edwards	0.15087	0.081519	-0.0094209, 0.311173

Regression Model

$$log(SalePrice) = 8.655 - 0.274BrkSide - 1.230Edwards + (0.0438 + 0.370BrkSide + 0.151Edwards)\log(GrLivArea)$$

Interpretation of Results

- The intercept is 8.655. Assuming all other inputs were negated, this would lead to a SalePrice of $5,738.
- The reference neighborhood is NAmes
    - For every doubling of area within NAmes, the SalePrice is associated with a $2^{0.44987}$ multiplicative change, which translates to a 36.6% increase in value.
    - Homes within BrkSide:
        - Are associated with a multiplicative change in value of $e^{-2.74255}$=0.0644, or 94% decrease in value (if holding all other values constant).
        - Are associated with an increase in the log of the area with a factor of 0.36976. The interpretation of this is that for every doubling of area within this neighborhood is associated with an adjustment (vs. Ames) multiplicative factor of $2^{0.36976}$, or 29.2% increase (65.8% total increase when doubling area).
    - Homes within Edwards:
        - Are associated with a multiplicative change in value of $e^{-1.2297}$=0.2924 , or 71% decrease in value (if holding all other values constant).
        - Are associated with an increase in the log of the area with a factor of 0.15087. The interpretation of this is that for every doubling of area within this neighborhood is associated with an adjustment (vs. Ames) multiplicative factor of $2^{0.15087}$, or 11.0% increase (40.2% total increase when doubling area).

Conclusion

Our analysis has been able to identify approximately 52% of the variation in SalePrice within the dataset neighborhoods of Ames, BrkSide, and Edwards. The primary contributing factor in this model is GrLivArea, which is the living area of the house in square feet.

Scope of Inference

Neither random assignment nor random sampling were mentioned in the dataset description, so generalization to the included neighborhoods and causation are not established in this case.

House Projection App

Robert Lane App

Mohammad Al-Rousan App

Analysis 2

Restatement of the Problem

For the second question of interest, the objective was to utilize all available data and variables to create the best linear regression model possible for predicting home sales prices. There are numerous techniques available for creating multi-linear regression models. This QOI focused on applying forward selection, backwards elimination, and stepwise selection to produce the highest performing models.

To begin, we need to look at the requested data and determine if it is suitable for analysis in its current form, and based on the non-normality of the histogram of sale price, these variables will need to log-transformed to address the normality assumption as well square footage will be log transformed.

Model Build

Forward selection:

$\log SalePrice = \beta_0 + \beta_1 OverallQual + \beta_2 GrLivArea + \beta_3 Neighborhood + \beta_4 BsmtFinSF1 + \beta_5 OveralCond + \beta_6 YearBuilt + \beta_7 GarageArea + \beta_8 BsmtUnfSF + \beta_9 BsmtFinSF2 + \beta_{10} MSZoning + \beta_{11} Fireplaces + \beta_{12} YearRemodAdd + \beta_{13} BldgType + \beta_{14} GarageCars + \beta_{15} CentralAir + \beta_{16} ScreenPorch + \beta_{17} WoodDeckSF + \beta_{18} OpenPorchSF + \beta_{19} EnclosedPorch + \beta_{20} KitchenQual + \beta_{21} HalfBath + \beta_{22} FullBath + \beta_{23} YrSold + \beta_{24} BedroomAbvGr$

Backward elimination:

$\log SalePrice = \beta_0 + \beta_1 OverallQual + \beta_2 OverallCond + \beta_3 YearBuilt + \beta_4 YearRemodAdd + \beta_5 BsmtFinSF1 + \beta_6 BsmtFinSF2 + \beta_7 BsmtUnfSF + \beta_8 GrLivArea + \beta_9 FullBath + \beta_{10} HalfBath + \beta_{11} BedroomAbvGr + \beta_{12} TotRmsAbvGrd + \beta_{13} Fireplaces + \beta_{14} GarageCars + \beta_{15} GarageArea + \beta_{16} WoodDeckSF + \beta_{17} OpenPorchSF + \beta_{18} EnclosedPorch + \beta_{19} ScreenPorch + \beta_{20} PoolArea + \beta_{21} YrSold + \beta_{22} Neighborhood + \beta_{23} MSZoning + \beta_{24} LotShape + \beta_{25} LotConfig + \beta_{26} Condition1 + \beta_{27} BldgType + \beta_{28} BsmtFinType1 + \beta_{29} HeatingQC + \beta_{30} CentralAir + \beta_{31} Electrical + \beta_{32} KitchenQual + \beta_{33} GarageType + \beta_{34} GarageFinish + \beta_{35} SaleType$

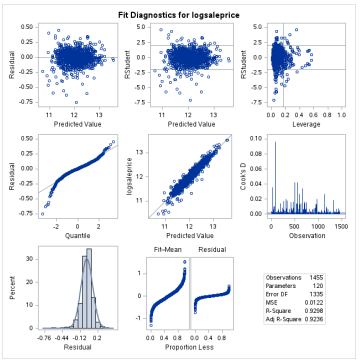
Stepwise selection:

$\log SalePrice = \beta_0 + \beta_1 OverallQual + \beta_2 OverallCond + \beta_3 YearBuilt + \beta_4 YearRemodAdd + \beta_5 BsmtFinSF1 + \beta_6 BsmtFinSF2 + \beta_7 BsmtUnfSF + \beta_8 GrLivArea + \beta_9 Fireplaces + \beta_{10} GarageArea + \beta_{11} Neighborhood + \beta_{12} MSZoning + \beta_{13} BldgType$

Custom selection:

$\log SalePrice = \beta_0 + \beta_1 GrLivArea + \beta_2 Neighborhood + \beta_3 GarageCars + \beta_4 OverallCond + \beta_5 YearBuilt + \beta_6 BsmtFinSF1 + \beta_7 MSZoning + \beta_8 Functional + \beta_9 SaleCondition + \beta_{10} KitchenQual + \beta_{11} LotArea + \beta_{12} ScreenPorch + \beta_{13} WoodDeckSF + \beta_{14} TotalBsmtSF + \beta_{15} CentralAir + \beta_{16} BsmtFullBath + \beta_{17} Fireplaces + \beta_{18} YearRemodAdd + \beta_{19} KitchenAbvGr$

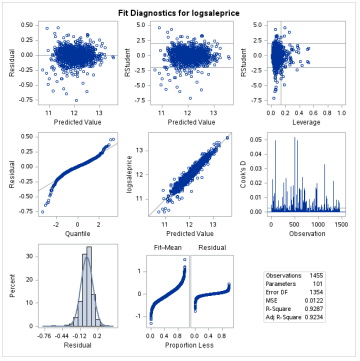
Assumptions



Forward Selection:

- **Multivariate Normality:** After looking at the histogram and Q-Q plot, the graphs look normally distributed
- **Linearity:** By looking at the quantile chart, it can be observed that the chart displays good normality
- **Homoscedasticity:** Based on the the charts, we can assume that there is equal standard deviation
- **Little or No Multicollinearity:** forward selection is helpful at reducing multicollinearity.
- **No Auto-Correlation:** Output variables were excluded from the model.

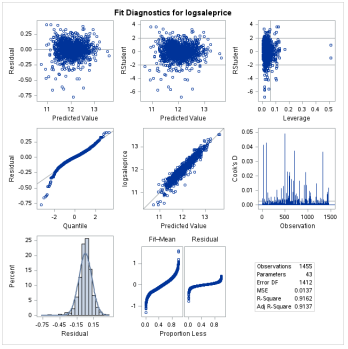
The regression model produced an R^2 of .9187 and a CV press of 19.75.



Backward Elimination:

- **Multivariate Normality:** After looking at the histogram and Q-Q plot, the graphs look normally distributed.
- **Linearity:** By looking at the quantile chart, it can be observed that the chart displays a healthy normality
- **Homoscedasticity:** Based on the the charts, we can assume that there is equal standard deviation
- **Little or No Multicollinearity:** Automatic selection may help reduce multicollinearity, since two variables describing the same thing typically end up becoming less significant individually, thus typically leading to their removal.
- **No Auto-Correlation:** Output variables were excluded from the model.

The regression model produced an adjusted R^2 of .9234 and a CV of 20.45.



Stepwise Selection:

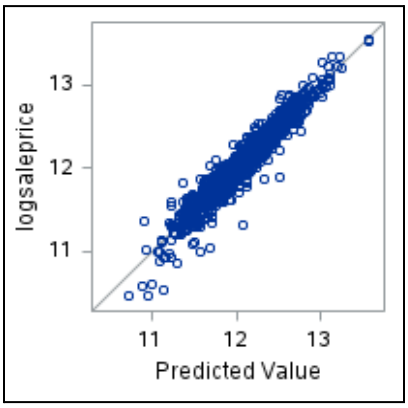
- **Multivariate Normality:** After looking at the histogram and Q-Q plot, the graphs look normally distributed.
- **Linearity:** By looking at the quantile chart, it can be observed that the chart displays a healthy normality
- **Homoscedasticity:** Based on the the charts, we can assume that there is equal standard deviation

- **Little or No Multicollinearity:** Ideally, this method would employ the best parts of Forward and Backward Selection in order to reduce multicollinearity.
- **No Auto-Correlation:** Output variables were excluded from the model.

The regression model produced an adjusted R^2 of .9137 and a CV of 20.90036.

Checking Assumptions

Residual Plots

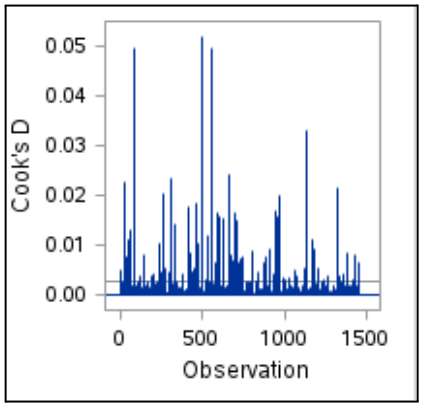


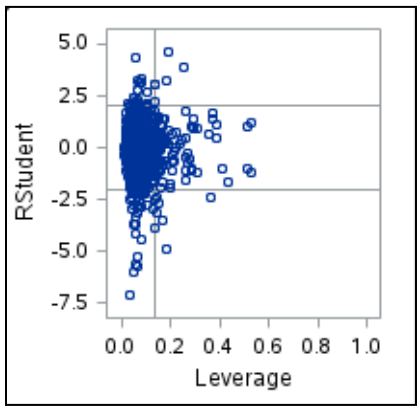
The predicted value shows a strong correlation with the actual values of SalePrice. This is indicative of an overall good fit in the model.

Influential Point Analysis

Because the Backwards Selection process resulted in the best scores, we are focusing the influential point analysis upon that model.

Although there are a handful of points within the high-leverage, high-influence range, they don't seem at the surface to be as abnormal from this data as they had with the previous model. They don't stray particularly far away from the rest of the data, so it seems as though adding more data points and explanatory variables may have helped solidify this model, as well suggesting that these points are explained by randomness.





Comparing Competing Models

Model	Adjusted R2	CV Press	Kaggle Score

Forward Selection	.9187	19.75429	.18707
Backward Elimination	.9234	20.45129	1.19426
Stepwise Selection	.9137	20.90036	.18783
Custom Selection	.9138	26.763	0.15817

Conclusion

Based upon the R^2 and CVPRESS scores, a Backward Selection provides the best fit for this particular application. Many input variables tend to have a linear relationship with the output, and even when they don't, data transformations can make it possible to still establish a form of linearity, which in turn can make linear regression algorithms quite powerful. Frequently, the simplest model which provides a good fit is best.

While there are more powerful methods to employ within the realm of machine learning, the fact that over 90% of the variance can be explained by using linear regression methods proves that it is a powerful statistical tool when properly utilized. Moreover, linear regression algorithms are among the fastest to generate.

Scope of Inference

Because it is not established that the data provided was randomly assigned, no causation is established in this case. Similarly, the lack of notation of random sampling means that the applicability of these results is confined to the dataset itself.

Mohammad Al-Rousan GitHub Project Page

Rob Lane GitHub Project Page

Appendix

```
proc print data = train;
run;

proc print data = trainNeighborhood;
run;

data trainNeighborhood;
set train;
where Neighborhood contains "Edwards"
      or Neighborhood contains "NAmes"
      or Neighborhood contains "BrkSide";
run;

proc sgplot data = trainNeighborhood;
scatter x = GrLivArea y = SalePrice / group = Neighborhood
markerattrs = (symbol=CircleFilled size=7);
title 'Neighborhood Sale Price vs Square Footage';
run;

data trainNeighborhood;
set trainneighborhood;
logsaleprice = log(saleprice);
run;

data trainNeighborhood;
set trainNeighborhood;
where Id ~= 524 AND Id ~= 643 AND Id ~= 725 AND Id ~= 1299 AND Id ~= 1424;
run;
```

```
data trainNeighborhood;
set trainneighborhood;
logGrLivArea = log(GrLivArea);
run;

proc sgplot data = trainNeighborhood;
scatter x = logGrLivArea y = logsaleprice / group = Neighborhood
markerattrs = (symbol=CircleFilled size=7);
title 'Neighborhood Sale Price vs Square Footage';
run;



/*Model 1 with outliers*/
proc glm data = trainNeighborhood2 plots= all;
where Neighborhood;
class Neighborhood;
model SalePrice = Neighborhood|GrLivArea / solution clparm;
run;

proc glmselect data = trainNeighborhood plots= all;
where Neighborhood;
class Neighborhood;
model logsaleprice = Neighborhood|logGrLivArea @2/ selection = Stepwise(stop =
cv)
cvmethod = random(5) stats = adjrsq;;;
run;

proc glm data = trainNeighborhood plots= all;
where Neighborhood;
class Neighborhood;
```

```
model logsaleprice = Neighborhood|logGrLivArea / solution clparm;

run;

data train;

set train;

logsaleprice = log(saleprice);

run;


data train2;

set train;

if _n_ = 524 then delete;

if _n_ = 643 then delete;

if _n_ = 725 then delete;

if _n_ = 1299 then delete;

if _n_ = 1424 then delete;

run;


proc sgscatter data = train2;

matrix logsaleprice verallqual overallcond yearbuilt yearremodadd bsmtfinsf1
bsmtfinsf2

bsmtunfsf grlivarea fullbath halfbath bedroomabvgr totrmsabvgrd fireplaces
garagecars garagearea

wooddecksf openporchsf enclosedporch screenporch poolarea yrsold neighborhood
mszoning lotshape lotconfig condition1 condition2 bldgtype housestyle

roofstyle bsmtfintype1 heatingqc centralair electrical kitchenqual garagetype
garagefinish saletype;

run;


proc glm data = train2 plots= all;

class Neighborhood mszoning lotshape lotconfig condition1 condition2 bldgtype
housestyle

roofstyle bsmtfintype1 heatingqc centralair electrical kitchenqual garagetype
garagefinish saletype;

model logsaleprice = overallqual overallcond yearbuilt yearremodadd bsmtfinsf1
bsmtfinsf2
```

```
bsmtunfsf grlivarea fullbath halfbath bedroomabvgr totrmsabvgrd fireplaces
garagecars garagearea

wooddecksf openporchsf enclosedporch screenporch poolarea yrsold neighborhood
mszoning lotshape lotconfig condition1 condition2 bldgtype housestyle

roofstyle bsmtfintype1 heatingqc centralair electrical kitchenqual garagetype
garagefinish saletype;

run;



proc glmselect data = train2 plots= all;

class Neighborhood mszoning lotshape lotconfig condition1 condition2 bldgtype
housestyle

roofstyle bsmtfintype1 heatingqc centralair electrical kitchenqual garagetype
garagefinish saletype;

model logsaleprice = overallqual overallcond yearbuilt yearremodadd bsmtfinsf1
bsmtfinsf2

bsmtunfsf grlivarea fullbath halfbath bedroomabvgr totrmsabvgrd fireplaces
garagecars garagearea

wooddecksf openporchsf enclosedporch screenporch poolarea yrsold neighborhood
mszoning lotshape lotconfig condition1 condition2 bldgtype housestyle

roofstyle bsmtfintype1 heatingqc centralair electrical kitchenqual garagetype
garagefinish saletype

/ selection= Forward(stop=CV) cvmethod=random(5) stat=adjrsq;

run;



proc glmselect data = train2;

class Neighborhood mszoning lotshape lotconfig condition1 condition2 bldgtype
housestyle

roofstyle bsmtfintype1 heatingqc centralair electrical kitchenqual garagetype
garagefinish saletype;

model logsaleprice = overallqual overallcond yearbuilt yearremodadd bsmtfinsf1
bsmtfinsf2

bsmtunfsf grlivarea fullbath halfbath bedroomabvgr totrmsabvgrd fireplaces
garagecars garagearea

wooddecksf openporchsf enclosedporch screenporch poolarea yrsold neighborhood
mszoning lotshape lotconfig condition1 condition2 bldgtype housestyle

roofstyle bsmtfintype1 heatingqc centralair electrical kitchenqual garagetype
garagefinish saletype
```

```
/ selection= Stepwise(stop=CV) cvmethod=random(5) stat=adjrsq;

run;


proc glmselect data = train2;

class Neighborhood mszoning lotshape lotconfig condition1 condition2 bldgtype
housestyle

roofstyle bsmtfintype1 heatingqc centralair electrical kitchenqual garagetype
garagefinish saletype;

model logsaleprice = overallqual overallcond yearbuilt yearremodadd bsmtfinsf1
bsmtfinsf2

bsmtunfsf grlivarea fullbath halfbath bedroomabvgr totrmsabvgrd fireplaces
garagecars garagearea

wooddecksf openporchsf enclosedporch screenporch poolarea yrsold neighborhood
mszoning lotshape lotconfig condition1 condition2 bldgtype housestyle

roofstyle bsmtfintype1 heatingqc centralair electrical kitchenqual garagetype
garagefinish saletype

/ selection= Backward(stop=CV) cvmethod=random(5) stat=adjrsq;

run;


proc glm data = train2 plots= all;

class Neighborhood mszoning lotshape lotconfig condition1 condition2 bldgtype
housestyle

roofstyle bsmtfintype1 heatingqc centralair electrical kitchenqual garagetype
garagefinish saletype;

model logsaleprice = overallqual overallcond yearbuilt yearremodadd bsmtfinsf1
bsmtfinsf2

bsmtunfsf grlivarea fullbath halfbath bedroomabvgr totrmsabvgrd fireplaces
garagecars garagearea

wooddecksf openporchsf enclosedporch screenporch poolarea yrsold neighborhood
mszoning lotshape lotconfig condition1 condition2 bldgtype housestyle

roofstyle bsmtfintype1 heatingqc centralair electrical kitchenqual garagetype
garagefinish saletype;

run;


proc glm data = train2 plots = all;

class neighborhood MSZoning BldgType CentralAir KitchenQual;
```

```
model logsaleprice = OverallQual OverallCond GrLivArea Neighborhood BsmtFinSF1
YearBuilt

GarageArea BsmtUnfSF BsmtFinSF2 MSZoning Fireplaces YearRemodAdd BldgType
GarageCars

CentralAir ScreenPorch WoodDeckSF OpenPorchSF EnclosedPorch KitchenQual
HalfBath FullBath

YrSold BedroomAbvGr;

run;



proc glm data = train2 plots= all;

class Neighborhood Mszoning LotShape LotConfig Condition1 BldgType BsmtFinType1
HeatingQC CentralAir Electrical

KitchenQual GarageType GarageFinish SaleType;

model logsaleprice = OverallQual OverallCond YearBuilt YearRemodAdd BsmtFinSF1
BsmtFinSF2 BsmtUnfSF

GrLivArea FullBath HalfBath BedroomAbvGr TotRmsAbvGrd Fireplaces GarageCars
GarageArea

WoodDeckSF OpenPorchSF EnclosedPorch ScreenPorch PoolArea YrSold Neighborhood
MSZoning

LotShape LotConfig Condition1 BldgType BsmtFinType1 HeatingQC CentralAir
Electrical

KitchenQual GarageType GarageFinish SaleType;

run;



proc glmselect data = train2 plots= all;

class Neighborhood Mszoning LotShape LotConfig Condition1 BldgType BsmtFinType1
HeatingQC CentralAir Electrical

KitchenQual GarageType GarageFinish SaleType HouseStyle RoofMatl Functional
SaleCondition

Exterior1st Heating LandSlope GarageQual Foundation LotFrontage GarageCond
ExterCond Street;

model logsaleprice = 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 YearRemodAdd GarageArea

Foundation LotFrontage KitchenAbvGr GarageCond SaleType ExterCond Street
HalfBath

/ selection= Forward(stop=CV) cvmethod=random(5) stat=adjrsq;

run;

data results2;

set results;

logprice = Predict;

if Predict >0 then logprice = Predict;

if Predict <= 0 then logprice = 9.21034;

keep id SalePrice logprice;

where id > 1460;


proc print data=results2(obs=5);

run;


data results3;

set results2;

if exp(logprice) > 0 then SalePrice = exp(logprice);

if exp(logprice)<= 0 then SalePrice = 10000;

keep id SalePrice;

where id > 1460;

;


proc print data = results3(obs=5);

run;


PROC EXPORT DATA= WORK.RESULTS3

            OUTFILE= "/home/u62637052/export/backward.csv"
```

```
            DBMS=CSV REPLACE;

     PUTNAMES=YES;

RUN;


proc glmselect data = combine;

class Neighborhood mszoning lotshape lotconfig condition1 condition2 bldgtype
housestyle

roofstyle bsmtfintype1 heatingqc centralair electrical kitchenqual garagetype
garagefinish saletype;

model logsaleprice = overallqual overallcond yearbuilt yearremodadd bsmtfinsf1
bsmtfinsf2

bsmtunfsf grlivarea fullbath halfbath bedroomabvgr totrmsabvgrd fireplaces
garagecars garagearea

wooddecksf openporchsf enclosedporch screenporch poolarea yrsold neighborhood
mszoning lotshape lotconfig condition1 condition2 bldgtype housestyle

roofstyle bsmtfintype1 heatingqc centralair electrical kitchenqual garagetype
garagefinish saletype

/ selection= Stepwise(stop=CV) cvmethod=random(5) stat=adjrsq;

output out = results p = Predict;

run;


proc glmselect data = combine;

class Neighborhood mszoning lotshape lotconfig condition1 condition2 bldgtype
housestyle

roofstyle bsmtfintype1 heatingqc centralair electrical kitchenqual garagetype
garagefinish saletype;

model logsaleprice = overallqual overallcond yearbuilt yearremodadd bsmtfinsf1
bsmtfinsf2

bsmtunfsf grlivarea fullbath halfbath bedroomabvgr totrmsabvgrd fireplaces
garagecars garagearea

wooddecksf openporchsf enclosedporch screenporch poolarea yrsold neighborhood
mszoning lotshape lotconfig condition1 condition2 bldgtype housestyle

roofstyle bsmtfintype1 heatingqc centralair electrical kitchenqual garagetype
garagefinish saletype

/ selection= Backward(stop=CV) cvmethod=random(5) stat=adjrsq;

output out = results p = Predict;
```

```
run;


proc glmselect data = combine;

class Neighborhood mszoning lotshape lotconfig condition1 condition2 bldgtype
housestyle

roofstyle bsmtfintype1 heatingqc centralair electrical kitchenqual garagetype
garagefinish saletype;

model logsaleprice = overallqual overallcond yearbuilt yearremodadd bsmtfinsf1
bsmtfinsf2

bsmtunfsf grlivarea fullbath halfbath bedroomabvgr totrmsabvgrd fireplaces
garagecars garagearea

wooddecksf openporchsf enclosedporch screenporch poolarea yrsold neighborhood
mszoning lotshape lotconfig condition1 condition2 bldgtype housestyle

roofstyle bsmtfintype1 heatingqc centralair electrical kitchenqual garagetype
garagefinish saletype

/ selection= Backward(stop=CV) cvmethod=random(5) stat=adjrsq;

output out = results p = Predict;

run;
```

```
Analysis2.SAS
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 IMPORT OUT= WORK.housetest

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

            DBMS=CSV REPLACE;

      GETNAMES=YES;

      DATAROW=2;

RUN;


data loghouse;

set house;

logprice = log(SalePrice);

logarea = log(GrLivArea);

loglotarea = log(LotArea);

run;


data loghousetest;

set housetest;

logprice = .;

logarea = log(GrLivArea);

loglotarea = log(LotArea);

SalePrice = .;

run;
```

```
data loghousecomb;

set house housetest;

run;


data house_log_dummies;


set loghousecomb;


    if    MSSubClass    =60    then    MSSubClass60    =1; else    MSSubClass60
=0;

    if    MSSubClass    =20    then    MSSubClass20    =1; else    MSSubClass20
=0;

    if    MSSubClass    =70    then    MSSubClass70    =1; else    MSSubClass70
=0;

    if    MSSubClass    =50    then    MSSubClass50    =1; else    MSSubClass50
=0;

    if    MSSubClass    =190    then    MSSubClass190    =1; else
MSSubClass190    =0;

    if    MSSubClass    =45    then    MSSubClass45    =1; else    MSSubClass45
=0;

    if    MSSubClass    =90    then    MSSubClass90    =1; else    MSSubClass90
=0;

    if    MSSubClass    =120    then    MSSubClass120    =1; else
MSSubClass120    =0;

    if    MSSubClass    =30    then    MSSubClass30    =1; else    MSSubClass30
=0;

    if    MSSubClass    =85    then    MSSubClass85    =1; else    MSSubClass85
=0;

    if    MSSubClass    =80    then    MSSubClass80    =1; else    MSSubClass80
=0;

    if    MSSubClass    =160    then    MSSubClass160    =1; else
MSSubClass160    =0;

    if    MSSubClass    =75    then    MSSubClass75    =1; else    MSSubClass75
=0;

    if    MSSubClass    =180    then    MSSubClass180    =1; else
MSSubClass180    =0;
```

```
    if    MSZoning    ="RL"    then    MSZoningRL    =1; else    MSZoningRL
=0;

    if    MSZoning    ="RM"    then    MSZoningRM    =1; else    MSZoningRM
=0;

    if    MSZoning    ="C (all)"    then    MSZoningC (all)    =1; else
MSZoningC (all)    =0;

    if    MSZoning    ="FV"    then    MSZoningFV    =1; else    MSZoningFV
=0;


    if    LotFrontage    ="NA"    then    LotFrontage    =0;


    if    Street    ="Pave"    then    StreetPave    =1; else    StreetPave
=0;


    if    Alley    ="Pave"    then    AlleyPave    =1; else    AlleyPave    =0;

    if    Alley    ="Grvl"    then    AlleyGrvl    =1; else    AlleyGrvl    =0;


    if    LandContour    ="Lvl"    then    LandContourLvl    =1; else
LandContourLvl    =0;

    if    LandContour    ="Bnk"    then    LandContourBnk    =1; else
LandContourBnk    =0;

    if    LandContour    ="Low"    then    LandContourLow    =1; else
LandContourLow    =0;


    if    LotShape    ="Reg"    then    LotShapeReg    =1; else    LotShapeReg
=0;

    if    LotShape    ="IR1"    then    LotShapeIR1    =1; else    LotShapeIR1
=0;

    if    LotShape    ="IR2"    then    LotShapeIR2    =1; else    LotShapeIR2
=0;


    if    LandContour    ="Lvl"    then    LandContourLvl    =1; else
LandContourLvl    =0;

    if    LandContour    ="Bnk"    then    LandContourBnk    =1; else
LandContourBnk    =0;

    if    LandContour    ="Low"    then    LandContourLow    =1; else
LandContourLow    =0;
```

```
    if    Utilities    ="AllPub"    then    UtilitiesAllPub    =1; else
UtilitiesAllPub    =0;


    if    LotConfig    ="Inside"    then    LotConfigInside    =1; else
LotConfigInside    =0;

    if    LotConfig    ="FR2"    then    LotConfigFR2    =1; else
LotConfigFR2    =0;

    if    LotConfig    ="Corner"    then    LotConfigCorner    =1; else
LotConfigCorner    =0;

    if    LotConfig    ="CulDSac"    then    LotConfigCulDSac    =1; else
LotConfigCulDSac    =0;


    if    LandSlope    ="Gtl"    then    LandSlopeGtl    =1; else
LandSlopeGtl    =0;

    if    LandSlope    ="Mod"    then    LandSlopeMod    =1; else
LandSlopeMod    =0;


    if    Neighborhood    ="CollgCr"    then    NeighborhoodCollgCr    =1; else
NeighborhoodCollgCr    =0;

    if    Neighborhood    ="Veenker"    then    NeighborhoodVeenker    =1; else
NeighborhoodVeenker    =0;

    if    Neighborhood    ="Crawfor"    then    NeighborhoodCrawfor    =1; else
NeighborhoodCrawfor    =0;

    if    Neighborhood    ="NoRidge"    then    NeighborhoodNoRidge    =1; else
NeighborhoodNoRidge    =0;

    if    Neighborhood    ="Mitchel"    then    NeighborhoodMitchel    =1; else
NeighborhoodMitchel    =0;

    if    Neighborhood    ="Somerst"    then    NeighborhoodSomerst    =1; else
NeighborhoodSomerst    =0;

    if    Neighborhood    ="NWAmes"    then    NeighborhoodNWAmes    =1; else
NeighborhoodNWAmes    =0;

    if    Neighborhood    ="OldTown"    then    NeighborhoodOldTown    =1; else
NeighborhoodOldTown    =0;

    if    Neighborhood    ="BrkSide"    then    NeighborhoodBrkSide    =1; else
NeighborhoodBrkSide    =0;

    if    Neighborhood    ="Sawyer"    then    NeighborhoodSawyer    =1; else
NeighborhoodSawyer    =0;
```

```
    if    Neighborhood    ="NridgHt"    then    NeighborhoodNridgHt    =1; else
NeighborhoodNridgHt    =0;

    if    Neighborhood    ="NAmes"    then    NeighborhoodNAmes    =1; else
NeighborhoodNAmes    =0;

    if    Neighborhood    ="SawyerW"    then    NeighborhoodSawyerW    =1; else
NeighborhoodSawyerW    =0;

    if    Neighborhood    ="IDOTRR"    then    NeighborhoodIDOTRR    =1; else
NeighborhoodIDOTRR    =0;

    if    Neighborhood    ="MeadowV"    then    NeighborhoodMeadowV    =1; else
NeighborhoodMeadowV    =0;

    if    Neighborhood    ="Edwards"    then    NeighborhoodEdwards    =1; else
NeighborhoodEdwards    =0;

    if    Neighborhood    ="Timber"    then    NeighborhoodTimber    =1; else
NeighborhoodTimber    =0;

    if    Neighborhood    ="Gilbert"    then    NeighborhoodGilbert    =1; else
NeighborhoodGilbert    =0;

    if    Neighborhood    ="StoneBr"    then    NeighborhoodStoneBr    =1; else
NeighborhoodStoneBr    =0;

    if    Neighborhood    ="ClearCr"    then    NeighborhoodClearCr    =1; else
NeighborhoodClearCr    =0;

    if    Neighborhood    ="NPkVill"    then    NeighborhoodNPkVill    =1; else
NeighborhoodNPkVill    =0;

    if    Neighborhood    ="Blmngtn"    then    NeighborhoodBlmngtn    =1; else
NeighborhoodBlmngtn    =0;

    if    Neighborhood    ="BrDale"    then    NeighborhoodBrDale    =1; else
NeighborhoodBrDale    =0;

    if    Neighborhood    ="SWISU"    then    NeighborhoodSWISU    =1; else
NeighborhoodSWISU    =0;


    if    Condition1    ="Norm"    then    Condition1Norm    =1; else
Condition1Norm    =0;

    if    Condition1    ="Feedr"    then    Condition1Feedr    =1; else
Condition1Feedr    =0;

    if    Condition1    ="PosN"    then    Condition1PosN    =1; else
Condition1PosN    =0;

    if    Condition1    ="Artery"    then    Condition1Artery    =1; else
Condition1Artery    =0;

    if    Condition1    ="RRAe"    then    Condition1RRAe    =1; else
Condition1RRAe    =0;
```

```
    if    Condition1    ="RRNn"    then    Condition1RRNn    =1; else
Condition1RRNn    =0;

    if    Condition1    ="RRAn"    then    Condition1RRAn    =1; else
Condition1RRAn    =0;

    if    Condition1    ="PosA"    then    Condition1PosA    =1; else
Condition1PosA    =0;



    if    Condition2    ="Norm"    then    Condition2Norm    =1; else
Condition2Norm    =0;

    if    Condition2    ="Artery"    then    Condition2Artery    =1; else
Condition2Artery    =0;

    if    Condition2    ="RRNn"    then    Condition2RRNn    =1; else
Condition2RRNn    =0;

    if    Condition2    ="Feedr"    then    Condition2Feedr    =1; else
Condition2Feedr    =0;

    if    Condition2    ="PosN"    then    Condition2PosN    =1; else
Condition2PosN    =0;

    if    Condition2    ="PosA"    then    Condition2PosA    =1; else
Condition2PosA    =0;

    if    Condition2    ="RRAn"    then    Condition2RRAn    =1; else
Condition2RRAn    =0;



    if    BldgType    ="1Fam"    then    BldgType1Fam    =1; else
BldgType1Fam    =0;

    if    BldgType    ="2fmCon"    then    BldgType2fmCon    =1; else
BldgType2fmCon    =0;

    if    BldgType    ="Duplex"    then    BldgTypeDuplex    =1; else
BldgTypeDuplex    =0;

    if    BldgType    ="TwnhsE"    then    BldgTypeTwnhsE    =1; else
BldgTypeTwnhsE    =0;



    if    HouseStyle    ="2Story"    then    HouseStyle2Story    =1; else
HouseStyle2Story    =0;

    if    HouseStyle    ="1Story"    then    HouseStyle1Story    =1; else
HouseStyle1Story    =0;

    if    HouseStyle    ="1.5Fin"    then    HouseStyle15Fin    =1; else
HouseStyle15Fin    =0;

    if    HouseStyle    ="1.5Unf"    then    HouseStyle15Unf    =1; else
HouseStyle15Unf    =0;
```

```
    if    HouseStyle    ="SFoyer"    then    HouseStyleSFoyer    =1; else
HouseStyleSFoyer    =0;

    if    HouseStyle    ="SLvl"    then    HouseStyleSLvl    =1; else
HouseStyleSLvl    =0;

    if    HouseStyle    ="2.5Unf"    then    HouseStyle25Unf    =1; else
HouseStyle25Unf    =0;



    if    RoofStyle    ="Gable"    then    RoofStyleGable    =1; else
RoofStyleGable    =0;

    if    RoofStyle    ="Hip"    then    RoofStyleHip    =1; else
RoofStyleHip    =0;

    if    RoofStyle    ="Gambrel"    then    RoofStyleGambrel    =1; else
RoofStyleGambrel    =0;

    if    RoofStyle    ="Mansard"    then    RoofStyleMansard    =1; else
RoofStyleMansard    =0;

    if    RoofStyle    ="Flat"    then    RoofStyleFlat    =1; else
RoofStyleFlat    =0;



    if    RoofMatl    ="CompShg"    then    RoofMatlCompShg    =1; else
RoofMatlCompShg    =0;

    if    RoofMatl    ="WdShngl"    then    RoofMatlWdShngl    =1; else
RoofMatlWdShngl    =0;

    if    RoofMatl    ="Metal"    then    RoofMatlMetal    =1; else
RoofMatlMetal    =0;

    if    RoofMatl    ="WdShake"    then    RoofMatlWdShake    =1; else
RoofMatlWdShake    =0;

    if    RoofMatl    ="Membran"    then    RoofMatlMembran    =1; else
RoofMatlMembran    =0;

    if    RoofMatl    ="ClyTile"    then    RoofMatlClyTile    =1; else
RoofMatlClyTile    =0;

    if    RoofMatl    ="Roll"    then    RoofMatlRoll    =1; else
RoofMatlRoll    =0;



    if    Exterior1st    ="VinylSd"    then    Exterior1stVinylSd    =1; else
Exterior1stVinylSd    =0;

    if    Exterior1st    ="MetalSd"    then    Exterior1stMetalSd    =1; else
Exterior1stMetalSd    =0;

    if    Exterior1st    ="Wd Sdng"    then    Exterior1stWd Sdng    =1; else
Exterior1stWd Sdng    =0;
```

```
    if    Exterior1st    ="HdBoard"    then    Exterior1stHdBoard    =1; else
Exterior1stHdBoard    =0;

    if    Exterior1st    ="BrkFace"    then    Exterior1stBrkFace    =1; else
Exterior1stBrkFace    =0;

    if    Exterior1st    ="WdShing"    then    Exterior1stWdShing    =1; else
Exterior1stWdShing    =0;

    if    Exterior1st    ="CemntBd"    then    Exterior1stCemntBd    =1; else
Exterior1stCemntBd    =0;

    if    Exterior1st    ="Plywood"    then    Exterior1stPlywood    =1; else
Exterior1stPlywood    =0;

    if    Exterior1st    ="AsbShng"    then    Exterior1stAsbShng    =1; else
Exterior1stAsbShng    =0;

    if    Exterior1st    ="Stucco"    then    Exterior1stStucco    =1; else
Exterior1stStucco    =0;

    if    Exterior1st    ="BrkComm"    then    Exterior1stBrkComm    =1; else
Exterior1stBrkComm    =0;

    if    Exterior1st    ="AsphShn"    then    Exterior1stAsphShn    =1; else
Exterior1stAsphShn    =0;

    if    Exterior1st    ="Stone"    then    Exterior1stStone    =1; else
Exterior1stStone    =0;

    if    Exterior1st    ="ImStucc"    then    Exterior1stImStucc    =1; else
Exterior1stImStucc    =0;



    if    Exterior2nd    ="VinylSd"    then    Exterior2ndVinylSd    =1; else
Exterior2ndVinylSd    =0;

    if    Exterior2nd    ="MetalSd"    then    Exterior2ndMetalSd    =1; else
Exterior2ndMetalSd    =0;

    if    Exterior2nd    ="Wd Shng"    then    Exterior2ndWd Shng    =1; else
Exterior2ndWd Shng    =0;

    if    Exterior2nd    ="HdBoard"    then    Exterior2ndHdBoard    =1; else
Exterior2ndHdBoard    =0;

    if    Exterior2nd    ="Plywood"    then    Exterior2ndPlywood    =1; else
Exterior2ndPlywood    =0;

    if    Exterior2nd    ="Wd Sdng"    then    Exterior2ndWd Sdng    =1; else
Exterior2ndWd Sdng    =0;

    if    Exterior2nd    ="CmentBd"    then    Exterior2ndCmentBd    =1; else
Exterior2ndCmentBd    =0;

    if    Exterior2nd    ="BrkFace"    then    Exterior2ndBrkFace    =1; else
Exterior2ndBrkFace    =0;
```

```
    if    Exterior2nd    ="Stucco"    then    Exterior2ndStucco    =1; else
Exterior2ndStucco    =0;

    if    Exterior2nd    ="AsbShng"    then    Exterior2ndAsbShng    =1; else
Exterior2ndAsbShng    =0;

    if    Exterior2nd    ="Brk Cmn"    then    Exterior2ndBrk Cmn    =1; else
Exterior2ndBrk Cmn    =0;

    if    Exterior2nd    ="ImStucc"    then    Exterior2ndImStucc    =1; else
Exterior2ndImStucc    =0;

    if    Exterior2nd    ="AsphShn"    then    Exterior2ndAsphShn    =1; else
Exterior2ndAsphShn    =0;

    if    Exterior2nd    ="Stone"    then    Exterior2ndStone    =1; else
Exterior2ndStone    =0;

    if    Exterior2nd    ="Other"    then    Exterior2ndOther    =1; else
Exterior2ndOther    =0;



    if    MasVnrType    ="BrkFace"    then    MasVnrTypeBrkFace    =1; else
MasVnrTypeBrkFace    =0;

    if    MasVnrType    ="None"    then    MasVnrTypeNone    =1; else
MasVnrTypeNone    =0;

    if    MasVnrType    ="Stone"    then    MasVnrTypeStone    =1; else
MasVnrTypeStone    =0;

    if    MasVnrType    ="BrkCmn"    then    MasVnrTypeBrkCmn    =1; else
MasVnrTypeBrkCmn    =0;



    if    ExterQual    ="Gd"    then    ExterQualGd    =1; else    ExterQualGd
=0;

    if    ExterQual    ="TA"    then    ExterQualTA    =1; else    ExterQualTA
=0;

    if    ExterQual    ="Ex"    then    ExterQualEx    =1; else    ExterQualEx
=0;



    if    ExterCond    ="TA"    then    ExterCondTA    =1; else    ExterCondTA
=0;

    if    ExterCond    ="Gd"    then    ExterCondGd    =1; else    ExterCondGd
=0;

    if    ExterCond    ="Fa"    then    ExterCondFa    =1; else    ExterCondFa
=0;

    if    ExterCond    ="Po"    then    ExterCondPo    =1; else    ExterCondPo
=0;
```

```
    if    Foundation    ="PConc"    then    FoundationPConc    =1; else
FoundationPConc    =0;

    if    Foundation    ="CBlock"    then    FoundationCBlock    =1; else
FoundationCBlock    =0;

    if    Foundation    ="BrkTil"    then    FoundationBrkTil    =1; else
FoundationBrkTil    =0;

    if    Foundation    ="Wood"    then    FoundationWood    =1; else
FoundationWood    =0;

    if    Foundation    ="Slab"    then    FoundationSlab    =1; else
FoundationSlab    =0;


    if    BsmtQual    ="Gd"    then    BsmtQualGd    =1; else    BsmtQualGd
=0;

    if    BsmtQual    ="TA"    then    BsmtQualTA    =1; else    BsmtQualTA
=0;

    if    BsmtQual    ="Ex"    then    BsmtQualEx    =1; else    BsmtQualEx
=0;

    if    BsmtQual    ="NA"    then    BsmtQualNA    =1; else    BsmtQualNA
=0;


    if    BsmtCond    ="TA"    then    BsmtCondTA    =1; else    BsmtCondTA
=0;

    if    BsmtCond    ="Gd"    then    BsmtCondGd    =1; else    BsmtCondGd
=0;

    if    BsmtCond    ="NA"    then    BsmtCondNA    =1; else    BsmtCondNA
=0;

    if    BsmtCond    ="Fa"    then    BsmtCondFa    =1; else    BsmtCondFa
=0;


    if    BsmtExposure    ="No"    then    BsmtExposureNo    =1; else
BsmtExposureNo    =0;

    if    BsmtExposure    ="Gd"    then    BsmtExposureGd    =1; else
BsmtExposureGd    =0;

    if    BsmtExposure    ="Mn"    then    BsmtExposureMn    =1; else
BsmtExposureMn    =0;

    if    BsmtExposure    ="Av"    then    BsmtExposureAv    =1; else
BsmtExposureAv    =0;
```

```
    if    BsmtFinType1    ="GLQ"    then    BsmtFinType1GLQ    =1; else
BsmtFinType1GLQ    =0;

    if    BsmtFinType1    ="ALQ"    then    BsmtFinType1ALQ    =1; else
BsmtFinType1ALQ    =0;

    if    BsmtFinType1    ="Unf"    then    BsmtFinType1Unf    =1; else
BsmtFinType1Unf    =0;

    if    BsmtFinType1    ="Rec"    then    BsmtFinType1Rec    =1; else
BsmtFinType1Rec    =0;

    if    BsmtFinType1    ="BLQ"    then    BsmtFinType1BLQ    =1; else
BsmtFinType1BLQ    =0;

    if    BsmtFinType1    ="NA"    then    BsmtFinType1NA    =1; else
BsmtFinType1NA    =0;


    if    BsmtFinType2    ="Unf"    then    BsmtFinType2Unf    =1; else
BsmtFinType2Unf    =0;

    if    BsmtFinType2    ="BLQ"    then    BsmtFinType2BLQ    =1; else
BsmtFinType2BLQ    =0;

    if    BsmtFinType2    ="NA"    then    BsmtFinType2NA    =1; else
BsmtFinType2NA    =0;

    if    BsmtFinType2    ="ALQ"    then    BsmtFinType2ALQ    =1; else
BsmtFinType2ALQ    =0;

    if    BsmtFinType2    ="Rec"    then    BsmtFinType2Rec    =1; else
BsmtFinType2Rec    =0;

    if    BsmtFinType2    ="LwQ"    then    BsmtFinType2LwQ    =1; else
BsmtFinType2LwQ    =0;


    if    Heating    ="GasA"    then    HeatingGasA    =1; else    HeatingGasA
=0;

    if    Heating    ="GasW"    then    HeatingGasW    =1; else    HeatingGasW
=0;

    if    Heating    ="Grav"    then    HeatingGrav    =1; else    HeatingGrav
=0;

    if    Heating    ="Wall"    then    HeatingWall    =1; else    HeatingWall
=0;

    if    Heating    ="OthW"    then    HeatingOthW    =1; else    HeatingOthW
=0;
```

```
    if    HeatingQC    ="Ex"    then    HeatingQCEx    =1; else    HeatingQCEx
=0;

    if    HeatingQC    ="Gd"    then    HeatingQCGd    =1; else    HeatingQCGd
=0;

    if    HeatingQC    ="TA"    then    HeatingQCTA    =1; else    HeatingQCTA
=0;

    if    HeatingQC    ="Fa"    then    HeatingQCFa    =1; else    HeatingQCFa
=0;


    if    CentralAir    ="Y"    then    CentralAirY    =1; else    CentralAirY
=0;


    if    Electrical    ="SBrkr"    then    ElectricalSBrkr    =1; else
ElectricalSBrkr    =0;

    if    Electrical    ="FuseF"    then    ElectricalFuseF    =1; else
ElectricalFuseF    =0;

    if    Electrical    ="FuseA"    then    ElectricalFuseA    =1; else
ElectricalFuseA    =0;

    if    Electrical    ="FuseP"    then    ElectricalFuseP    =1; else
ElectricalFuseP    =0;

    if    Electrical    ="Mix"    then    ElectricalMix    =1; else
ElectricalMix    =0;


    if    KitchenQual    ="Gd"    then    KitchenQualGd    =1; else
KitchenQualGd    =0;

    if    KitchenQual    ="TA"    then    KitchenQualTA    =1; else
KitchenQualTA    =0;

    if    KitchenQual    ="Ex"    then    KitchenQualEx    =1; else
KitchenQualEx    =0;


    if    Functional    ="Typ"    then    FunctionalTyp    =1; else
FunctionalTyp    =0;

    if    Functional    ="Min1"    then    FunctionalMin1    =1; else
FunctionalMin1    =0;

    if    Functional    ="Maj1"    then    FunctionalMaj1    =1; else
FunctionalMaj1    =0;

    if    Functional    ="Min2"    then    FunctionalMin2    =1; else
FunctionalMin2    =0;
```

```
    if    Functional    ="Mod"    then    FunctionalMod    =1; else
FunctionalMod    =0;

    if    Functional    ="Maj2"    then    FunctionalMaj2    =1; else
FunctionalMaj2    =0;



    if    FireplaceQu    ="NA"    then    FireplaceQuNA    =1; else
FireplaceQuNA    =0;

    if    FireplaceQu    ="TA"    then    FireplaceQuTA    =1; else
FireplaceQuTA    =0;

    if    FireplaceQu    ="Gd"    then    FireplaceQuGd    =1; else
FireplaceQuGd    =0;

    if    FireplaceQu    ="Fa"    then    FireplaceQuFa    =1; else
FireplaceQuFa    =0;

    if    FireplaceQu    ="Ex"    then    FireplaceQuEx    =1; else
FireplaceQuEx    =0;



    if    GarageType    ="Attchd"    then    GarageTypeAttchd    =1; else
GarageTypeAttchd    =0;

    if    GarageType    ="Detchd"    then    GarageTypeDetchd    =1; else
GarageTypeDetchd    =0;

    if    GarageType    ="BuiltIn"    then    GarageTypeBuiltIn    =1; else
GarageTypeBuiltIn    =0;

    if    GarageType    ="CarPort"    then    GarageTypeCarPort    =1; else
GarageTypeCarPort    =0;

    if    GarageType    ="NA"    then    GarageTypeNA    =1; else
GarageTypeNA    =0;

    if    GarageType    ="Basment"    then    GarageTypeBasment    =1; else
GarageTypeBasment    =0;



    if    GarageYrBlt    ="NA"    then    GarageYrBlt    =0;



    if    GarageFinish    ="RFn"    then    GarageFinishRFn    =1; else
GarageFinishRFn    =0;

    if    GarageFinish    ="Unf"    then    GarageFinishUnf    =1; else
GarageFinishUnf    =0;

    if    GarageFinish    ="Fin"    then    GarageFinishFin    =1; else
GarageFinishFin    =0;
```

```
    if    GarageQual    ="TA"    then    GarageQualTA    =1; else
GarageQualTA    =0;

    if    GarageQual    ="Fa"    then    GarageQualFa    =1; else
GarageQualFa    =0;

    if    GarageQual    ="Gd"    then    GarageQualGd    =1; else
GarageQualGd    =0;

    if    GarageQual    ="NA"    then    GarageQualNA    =1; else
GarageQualNA    =0;

    if    GarageQual    ="Ex"    then    GarageQualEx    =1; else
GarageQualEx    =0;


    if    GarageCond    ="TA"    then    GarageCondTA    =1; else
GarageCondTA    =0;

    if    GarageCond    ="Fa"    then    GarageCondFa    =1; else
GarageCondFa    =0;

    if    GarageCond    ="NA"    then    GarageCondNA    =1; else
GarageCondNA    =0;

    if    GarageCond    ="Gd"    then    GarageCondGd    =1; else
GarageCondGd    =0;

    if    GarageCond    ="Po"    then    GarageCondPo    =1; else
GarageCondPo    =0;


    if    PavedDrive    ="Y"    then    PavedDriveY    =1; else    PavedDriveY
=0;

    if    PavedDrive    ="P"    then    PavedDriveP    =1; else    PavedDriveN
=0;


    if    PoolQC    ="NA"    then    PoolQCNA    =1; else    PoolQCNA    =0;

    if    PoolQC    ="Ex"    then    PoolQCEx    =1; else    PoolQCEx    =0;

    if    PoolQC    ="Fa"    then    PoolQCFa    =1; else    PoolQCFa    =0;


    if    Fence    ="MnWw"    then    FenceMnWw    =1; else    FenceMnWw    =0;

    if    Fence    ="MnPrv"    then    FenceMnPrv    =1; else    FenceMnPrv
=0;

    if    Fence    ="GdWo"    then    FenceGdWo    =1; else    FenceGdWo    =0;

    if    Fence    ="GdPrv"    then    FenceGdPrv    =1; else    FenceGdPrv
=0;
```

```
    if    SaleType    ="WD"    then    SaleTypeWD    =1; else    SaleTypeWD
=0;

    if    SaleType    ="New"    then    SaleTypeNew    =1; else    SaleTypeNew
=0;

    if    SaleType    ="COD"    then    SaleTypeCOD    =1; else    SaleTypeCOD
=0;

    if    SaleType    ="ConLD"    then    SaleTypeConLD    =1; else
SaleTypeConLD    =0;

    if    SaleType    ="ConLI"    then    SaleTypeConLI    =1; else
SaleTypeConLI    =0;

    if    SaleType    ="CWD"    then    SaleTypeCWD    =1; else    SaleTypeCWD
=0;

    if    SaleType    ="ConLw"    then    SaleTypeConLw    =1; else
SaleTypeConLw    =0;

    if    SaleType    ="Con"    then    SaleTypeCon    =1; else    SaleTypeCon
=0;



    if    SaleCondition    ="Normal"    then    SaleConditionNormal    =1; else
SaleConditionNormal    =0;

    if    SaleCondition    ="Abnorml"    then    SaleConditionAbnorml    =1;
else    SaleConditionAbnorml    =0;

    if    SaleCondition    ="Partial"    then    SaleConditionPartial    =1;
else    SaleConditionPartial    =0;

    if    SaleCondition    ="AdjLand"    then    SaleConditionAdjLand    =1;
else    SaleConditionAdjLand    =0;

    if    SaleCondition    ="Alloca"    then    SaleConditionAlloca    =1; else
SaleConditionAlloca    =0;



run;




/*drop columns*/

data drop_text_cats;

    set house_log_dummies(drop=MSSubClass    MSZoning    LotFrontage    Street
Alley    LandContour    LotShape    Utilities    LotConfig    LandSlope
Neighborhood    Condition1    Condition2    BldgType    HouseStyle    RoofStyle
RoofMatl    Exterior1st    Exterior2nd    MasVnrType    ExterQual    ExterCond
```

```
Foundation    BsmtQual    BsmtCond    BsmtExposure    BsmtFinType1
BsmtFinType2    Heating    HeatingQC    CentralAir    Electrical    KitchenQual
Functional    FireplaceQu    GarageType    GarageYrBlt    GarageFinish
GarageQual    GarageCond    PavedDrive    PoolQC    Fence    SaleType
SaleCondition

);

run;
```

```
proc glmselect data = drop_text_cats;

      model logprice = logarea loglotarea

OverallQual    OverallCond YearBuilt    YearRemodAdd    BsmtFinSF1
BsmtFinSF2    BsmtUnfSF    TotalBsmtSF

2ndFlrSF    LowQualFinSF    GrLivArea    BsmtFullBath    BsmtHalfBath
FullBath    HalfBath    BedroomAbvGr    KitchenAbvGr

TotRmsAbvGrd    Fireplaces    GarageCars    GarageArea    WoodDeckSF
OpenPorchSF    EnclosedPorch    ScreenPorch    PoolArea

MiscVal    MoSold    YrSold MSSubClass60    MSSubClass20    MSSubClass70
MSSubClass50    MSSubClass190    MSSubClass45    MSSubClass90

MSSubClass120    MSSubClass30    MSSubClass85    MSSubClass80    MSSubClass160
MSSubClass75    MSSubClass180    MSZoningRL

MSZoningRM    MSZoningC (all)    MSZoningFV    LotFrontage    StreetPave
AlleyPave    AlleyGrvl    LandContourLvl    LandContourBnk

LandContourLow    LotShapeReg    LotShapeIR1    LotShapeIR2    UtilitiesAllPub
LotConfigInside    LotConfigFR2    LotConfigCorner

LotConfigCulDSac    LandSlopeGtl    LandSlopeMod    NeighborhoodCollgCr
NeighborhoodVeenker    NeighborhoodCrawfor

NeighborhoodNoRidge    NeighborhoodMitchel    NeighborhoodSomerst
NeighborhoodNWAmes    NeighborhoodOldTown    NeighborhoodBrkSide

NeighborhoodSawyer    NeighborhoodNridgHt    NeighborhoodNAmes
NeighborhoodSawyerW    NeighborhoodIDOTRR    NeighborhoodMeadowV

NeighborhoodEdwards    NeighborhoodTimber    NeighborhoodGilbert
NeighborhoodStoneBr    NeighborhoodClearCr    NeighborhoodNPkVill
```

```
NeighborhoodBlmngtn    NeighborhoodBrDale    NeighborhoodSWISU
Condition1Norm    Condition1Feedr    Condition1PosN    Condition1Artery

Condition1RRAe    Condition1RRNn    Condition1RRAn    Condition1PosA
Condition2Norm    Condition2Artery    Condition2RRNn

Condition2Feedr    Condition2PosN    Condition2PosA    Condition2RRAn
BldgType1Fam    BldgType2fmCon    BldgTypeDuplex    BldgTypeTwnhsE

HouseStyle2Story    HouseStyle1Story    HouseStyle15Fin    HouseStyle15Unf
HouseStyleSFoyer    HouseStyleSLvl    HouseStyle25Unf

RoofStyleGable    RoofStyleHip    RoofStyleGambrel    RoofStyleMansard
RoofStyleFlat    RoofMatlCompShg    RoofMatlWdShngl

RoofMatlMetal    RoofMatlWdShake    RoofMatlMembran    RoofMatlClyTile
RoofMatlRoll    Exterior1stVinylSd    Exterior1stMetalSd

Exterior1stWd Sdng    Exterior1stHdBoard    Exterior1stBrkFace
Exterior1stWdShing    Exterior1stCemntBd    Exterior1stPlywood

Exterior1stAsbShng    Exterior1stStucco    Exterior1stBrkComm
Exterior1stAsphShn    Exterior1stStone    Exterior1stImStucc

Exterior2ndVinylSd    Exterior2ndMetalSd    Exterior2ndWd Shng
Exterior2ndHdBoard    Exterior2ndPlywood    Exterior2ndWd Sdng

Exterior2ndCmentBd    Exterior2ndBrkFace    Exterior2ndStucco
Exterior2ndAsbShng    Exterior2ndBrk Cmn    Exterior2ndImStucc

Exterior2ndAsphShn    Exterior2ndStone    Exterior2ndOther    MasVnrTypeBrkFace
MasVnrTypeNone    MasVnrTypeStone    MasVnrTypeBrkCmn

ExterQualGd    ExterQualTA    ExterQualEx    ExterCondTA    ExterCondGd
ExterCondFa    ExterCondPo    FoundationPConc    FoundationCBlock

FoundationBrkTil    FoundationWood    FoundationSlab    BsmtQualGd
BsmtQualTA    BsmtQualEx    BsmtQualNA    BsmtCondTA    BsmtCondGd

BsmtCondNA    BsmtCondFa    BsmtExposureNo    BsmtExposureGd    BsmtExposureMn
BsmtExposureAv    BsmtFinType1GLQ    BsmtFinType1ALQ

BsmtFinType1Unf    BsmtFinType1Rec    BsmtFinType1BLQ    BsmtFinType1NA
BsmtFinType2Unf    BsmtFinType2BLQ    BsmtFinType2NA    BsmtFinType2ALQ

BsmtFinType2Rec    BsmtFinType2LwQ    HeatingGasA    HeatingGasW    HeatingGrav
HeatingWall    HeatingOthW    HeatingQCEx    HeatingQCGd    HeatingQCTA

HeatingQCFa    CentralAirY    ElectricalSBrkr    ElectricalFuseF
ElectricalFuseA    ElectricalFuseP    ElectricalMix    KitchenQualGd
KitchenQualTA

KitchenQualEx    FunctionalTyp    FunctionalMin1    FunctionalMaj1
FunctionalMin2    FunctionalMod    FunctionalMaj2    FireplaceQuNA

FireplaceQuTA    FireplaceQuGd    FireplaceQuFa    FireplaceQuEx
GarageTypeAttchd    GarageTypeDetchd    GarageTypeBuiltIn
```

```
GarageTypeCarPort    GarageTypeNA    GarageTypeBasment    GarageYrBlt
GarageFinishRFn    GarageFinishUnf    GarageFinishFin    GarageQualTA

GarageQualFa    GarageQualGd    GarageQualNA    GarageQualEx    GarageCondTA
GarageCondFa    GarageCondNA    GarageCondGd

GarageCondPo    PavedDriveY    PavedDriveP    PoolQCNA    PoolQCEx    PoolQCFa
FenceMnWw    FenceMnPrv    FenceGdWo    FenceGdPrv

SaleTypeWD    SaleTypeNew    SaleTypeCOD    SaleTypeConLD    SaleTypeConLI
SaleTypeCWD    SaleTypeConLw    SaleTypeCon    SaleConditionNormal

SaleConditionAbnorml    SaleConditionPartial    SaleConditionAdjLand
SaleConditionAlloca

/ selection = Backward(stop = CV) cvmethod=random(5) stats = adjrsq;

output out = results p = Predict;

run;




/*output the model to a file*/

data results2;

set results;

logprice = Predict;

if Predict >0 then logprice = Predict;

if Predict <= 0 then logprice = 9.21034;

keep id SalePrice logprice;

where id > 1460;


proc print data=results2(obs=5);

run;


data results3;

set results2;

if exp(logprice) > 0 then SalePrice = exp(logprice);

if exp(logprice)<= 0 then SalePrice = 10000;

keep id SalePrice;
```

```
where id > 1460;

;


proc print data = results3(obs=5);

run;


PROC EXPORT DATA= WORK.RESULTS3

            OUTFILE= "C:\Study Files\SMU MSDS\DS 6371 Statistical Foundations
for Data Science\statistical-housing-price-analysis\test-model2.csv"

            DBMS=CSV REPLACE;

      PUTNAMES=YES;

RUN;

/*Success!!!*/
```

```
Analysis1.SAS

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;




/*
train = house
test = housetest
*/
```

```
/*Attempt to output any sort of model to a file*/
data housetest;
set housetest;
SalePrice = .;
logprice = .;
;
/*It is vital to place this column here so it can be predicted.*/

data neighborhoods;
  set house housetest;
  where Neighborhood in ("NAmes", "Edwards", "BrkSide");
run;

proc glm data = neighborhoods;
model log(SalePrice) = log(GrLivArea);
output out = results p = Predict;
run;

data results2;
set results;
if Predict >0 then SalePrice = Predict;
if Predict <= 0 then SalePrice = 10000;
keep id SalePrice;
where id > 1460;

proc print data=results2(obs=5);
run;

PROC EXPORT DATA= WORK.RESULTS2
            OUTFILE= "C:\Study Files\SMU MSDS\DS 6371 Statistical Founda
```

```
tions for Data Science\statistical-housing-price-analysis\test-model.csv"
            DBMS=CSV REPLACE;
      PUTNAMES=YES;
RUN;
/*Success!!!*/
```