Aaron Abromowitz (https://aabromowitz.github.io/)
Catherine Ticzon (https://catherineticzon.github.io/)
Statistical Foundations 6371
December 3, 2023

Home Prices in Ames, Iowa

I. Introduction

As millennials, we have lost all hope of home ownership. However, if we were to consider buying a home, many factors would come into play in the decision making process. Similarly, many factors come into play in regard to how much a home is being sold for. In this paper, we analyze home sale prices in Ames, Iowa by investigating the most predictive variables of sale prices and the correlation of those variables to home sale prices. Further, we predict home sale prices through the creation of predictive models based on the relevant variables. The two central questions of interest (QOI) discussed in this paper are:

1. How are home sale prices related to the square footage of the living area? Do sale prices and its relationship to square footage vary based on the neighborhood in which the home is located? This QOI is limited to the neighborhoods of: North Ames, Edwards, and Brookside.
2. What is the most predictive linear model for home sale prices in Ames, Iowa?

All relevant software code and output referenced in this paper can be found in the Appendices.

II. Data Description

All data, which include both a training and test set, were provided to us by the client. Each data set has 1,460 and 1,459 observations, respectively, and 79 explanatory variables. The training set contains the home sale prices while the test set does not. The primary response variable used in the analyses of this paper is home sale price. Not all explanatory variables were used in the analyses. Below are some of the key variables used in our analysis:

- **Home Sale Price** (USD; 'SalePrice'). The property's sale price in dollars.
- **Living Area** (square feet; 'GrLivArea'). The ground living area is square feet.
- **Overall Quality** (range of 1-10; 'OverallQual'). The overall material and finish quality.
- **Number of Full Bathrooms** (range of 0-3; 'FullBath'). The number of bathrooms above grade.

See Appendix A for a full list of variables included in the dataset.

III. Analysis Question 1

Century 21 Ames has commissioned us to investigate the relationship between home sale prices and square footage within the neighborhoods of North Ames, Edwards, and Brookside

(QOI #1). Specifically, how are home sale prices related to the square footage of the living area of the homes? And, do sale prices and its relationship to square footage vary based on the neighborhood in which the home is located?

a. Building and Fitting the Model

Through exploratory data analysis and conducting diagnostic tests on the data, it was determined that the sale prices of homes and its relationships to square footage do indeed vary based on the neighborhood in which it is located (North Ames, Edwards, or Brookside). Therefore, each neighborhood will have its own model.

A regression model will be used for each of the neighborhoods, where square footage of the living area is the explanatory variable and home sale price is the response variable. For the model used for the North Ames neighborhood, both variables will be transformed with a log transformation; the model for North Ames is a log-log model. This is the same for the model for the Edwards neighborhood. The model for the Brookside neighborhood is a linear model with no transformations on the data. Below are the models for each neighborhood[1]:

$$\textit{log(Predicted Sale Price (North Ames))} = 8.492 + 0.473 * \log(\text{Living Area})$$
$$\textit{log(Predicted Sale Price (Edwards))} = 8.0065 + 0.5196 * \log(\text{Living Area})$$
$$\textit{Predicted Sale Price (Brookside)} = 19971.51 + 87.16 * (\text{Living Area})$$

Predicted sale price is in USD and the living area is square feet. In all 3 models, both β_0 (intercept) and β_1 (slope) are statistically significant (at alpha = 0.05 level). The simple linear regression models, with untransformed data, for the North Ames and Edwards neighborhood did not fulfill the assumptions necessary for a regression to be appropriate; therefore, these models were not selected for analyses (see Figures B-10 and B-11 for corresponding diagnostics plots in Appendix B).

b. Assumptions and Influential Points

All necessary assumptions – normal distribution, linearity, equal variance, and independence – for the regression model are met. Additionally, an investigation into additional regression diagnostics – Cook's Distance and leverage – support the use of regression models for analyses. All diagnostic plots for the models can be found in Appendix BII.

Normal distribution. The distribution of residuals in all 3 models are sufficiently spread out based on the visual evidence provided by the residual scatterplots. The histograms of the residuals also support this claim.

[1] See Appendix BI for the full model summaries.

Linearity. Based on the residual scatterplots for all models, the linearity assumption is reasonably fulfilled. We can feel satisfied that the relationship between the response variable, home sale price, and the explanatory variable, living area, is linearly related.

Equal variance. The diagnostic plots for the models do not show evidence against equal standard deviations.

Independence. Based on the provided identification number of each home listing and the nature of the data, each observation is assumed to be independent.

Cook's Distance and leverage. The log-log model for North Ames has a handful of points above the Cook's Distance threshold line and one relatively prominent point with a Cook's Distance value of approximately 0.2. The log-log model for Edwards has around 4 points above the threshold line and one relatively prominent point with a Cook's Distance value of approximately 0.6. The linear model for Brookside also has around 4 points above the threshold line and one relatively prominent point with a value of approximately 0.15. For the purposes of this paper, we will consider Cook's Distance values near or above 1 as extreme. Therefore, though there are some relatively prominent points on the Cook's Distance plots, the overall values are reasonable for continuing with the analyses.

Additionally, while there are prominent points in the Cook's Distance plots for the Edwards and Brookside models, no observations have both high leverage and high residuals in each of the models. In the Edwards and Brookside models, there are each around 3 observations with high leverage and medium residuals (not outliers). In the North Ames model, there are 2 observations with high leverage and high residuals (outliers), and a handful of observations with high leverage and medium residuals (not outliers). See the Cook's Distance and leverage plots in Figures B-1, B-4, and B-7 of Appendix B.

c. Model Parameters

North Ames Model. In the log-log model developed for the North Ames neighborhood, β_0 (intercept) is 8.492 and β_1 (slope) is 0.473. See model below:

$$\textit{log(Predicted Sale Price (North Ames))} = 8.492 + 0.473 * \log(\text{Living Area})$$

Given that this is a log-log model, the equation means that a doubling of the square feet of living area is associated with a 1.39 ($2^{\beta 1} = 2^{0.473}$) increase in the median home sale price. In other words, a doubling of the square feet of living area increases the estimated median of the home sale price by 39%. At 95% confidence, the true increase in the estimated median home sale price with the doubling of square feet of living area is between 32% and 46%.

Edwards Model. In the log-log model developed for the Edwards neighborhood, β_0 is 8.0065 and β_1 (slope) is 0.5196. See model below:

$$\textit{log(Predicted Sale Price (Edwards))} = 8.0065 + 0.5196 * \log(\text{Living Area})$$

Similar to the North Ames neighborhood model, the model for the Edwards neighborhood home sale prices is a log-log model. As such, a doubling of the living area is associated with a 1.43 ($2^{0.5196}$) increase in the median home sale price. In other words, a doubling of the square feet of living area increases the estimated median of the home sale price by 43%. At 95% confidence, the true increase of the estimated median home sale price is between 29% and 59%, as associated with the doubling of the square feet of living area.

Brookside Model. The model for the Brookside neighborhood did not require any data (log) transformations. The simple linear regression model has β_0 = 19,971.51 and β_1 = 87.16. See model below:

Predicted Sale Price (Brookside) = 19971.51 + 87.16 * (Living Area)

As this is a linear model, the interpretation of the slope is that a 1 square foot increase in the living area is associated with a $87.16 change in the home sale price[2]. More practically speaking, a 100 square foot increase in living area is associated with a $8,716 increase in the mean home sale price. At 95% confidence, the true increase in the mean home sale price associated with a 100 square foot increase in living area is between $7,179 and $10,253.

Comparing the Models. The table below lists the corresponding R^2, adjusted R^2, and CV Press values for each of the models. Since the adjusted R^2 adjusts for multiple predictors in a model, we do not see meaningful differences between R^2 and the adjusted R^2 values since the models used only have one predictor (living area). In regard to the CV Press value, it is important to note that the CV Press of the North Ames and Edwards log-log models are not comparable to the Brookside linear model.

Predictive Model	R^2	Adjusted R^2	CV PRESS
North Ames: Log-Log Model	0.4196	0.417	0.1588
Edwards: Log-Log Model	0.3285	0.3217	0.2608
Brookside: Linear Model	0.6975	0.6921	22757.67

See Appendix BIII for the full output and relevant code used in this section.

[2] β_0 indicates that, when the living area is equal to 0 square feet, the estimated mean home sale price is $19,971.51. While this may not have practical significance, it is indeed necessary in order to understand the model.

d. Conclusion of Analysis 1

For QOI #1, three separate models were built for the three different neighborhoods of interest, North Ames, Edwards and Brookside. All the assumptions required of a linear regression model (normal distribution, linearity, equal variance, and independence) were met, making a regression model suitable for these analyses. The models for the North Ames and Edwards neighborhoods use log-log models and the model for the Brookside neighborhood uses a linear model with no transformations to the data. The models for Edwards and Brookside did not have outliers with high leverage, whereas the model for North Ames had 2 outliers with high leverage.

In the North Ames neighborhood, a doubling of the square feet of living area is associated with a 39% increase in the estimated median home sale price. In the Edwards neighborhood, a doubling of the square feet of living area is associated with a 43% increase in the estimated median home sale price. In the Brookside neighborhood, a 100-square feet increase in living area is associated with a $8,716 change in the home sale price.

An RShiny application displaying scatterplots of home sale prices and living area is available at: https://catherineticzon.shinyapps.io/Stat1Project/

IV. Analysis Question 2

The second QOI pertains to creating the most predictive model of home sale prices in any given neighborhood in Ames, Iowa. This involves creating multiple models, and evaluating them by the different metrics – adjusted R^2, CV Press, and Kaggle score. Two models were requested by the client: a single linear regression (SLR), using a single variable only, and a multiple linear regression (MLR), combining the living area variable and the variable for number of full bathrooms. Both models use home sale price as the response variable. Lastly, we were asked to create the most predictive MLR model with no restriction on variables.

First, the correlation was calculated between the sale home price and each of the variables in the data set in order to find a single predictive variable to use in the SLR model. The variable with the highest linear correlation to the home sale price was overall quality. The residual plot of this SLR shows that the residuals increase with the predicted sale price (see Figure C-1, Appendix C). This points to the equal standard deviation assumption of the linear model being violated.

Correlation and residual analysis provided a starting point to establishing useful metrics to develop and refine models. Furthermore, adjusted R^2 values were used since the adjusted R^2 measures linearity, with a penalty for more complex models (i.e., more predictor variables). CV Press is an additional metric to evaluate predictions of the model from the sample data. The Kaggle Score was used to evaluate the developed model against the test set distributed by Kaggle.

Predictive Model	**Adjusted R^2**	**CV PRESS**	**Kaggle Score**
SalePrice vs OverallQual	.6822	45287	.22814

The increasing spread in residual values can sometimes be helped by taking a log transformation. Looking at log of the sale price vs overall quality (Figure C-2 in Appendix C), we can see that there is less visual evidence against the equal standard deviation assumption. When looking at the Cook's Distance plot (Figure C-3 in Appendix C) to see if there are any highly leveraged points, no points are above 1. The Adjusted R^2 and Kaggle metrics are also slightly better with the transformation (CV Press isn't comparable to the previous model because of the transformation).

Predictive Model	**Adjusted R^2**	**CV PRESS**	**Kaggle Score**
logSalePrice vs OverallQual	.6714	.23072	.22613

Adding more variables to the model can be used to increase its predictive power. When looking at variables, the log of the living space variable also had a high correlation coefficient. So we tried a model with the interaction between that variable and the overall quality variable. Even though this model does cause one of the entries to become a point with high Cook's Distance value (Figure C-4 in Appendix C), the overall model does better for the metrics we are looking at.

Predictive Model	**Adjusted R^2**	**CV PRESS**	**Kaggle Score**
logSalePrice vs OverallQual*logGrLivArea	.7635	.21933	.1899

It was specifically requested that we look at a model that uses the combination of living area and number of full bathrooms. This model however did not score as well using the different metrics. In fact, it performed worse than the model that just used the overall quality variable.

Predictive Model	**Adjusted R^2**	**CV PRESS**	**Kaggle Score**
SalePrice vs FullBath + GrLivArea	.5231	55157	.28586

Adding more variables to the model allows for an increase in model performance. One approach to adding many variables to a model is to try adding each variable individually based

on p values. Then continue to add variables until there is no more improvement. This technique is known as forward selection. A model with 12 predictor variables was created using forward selection and performed significantly better than previously used models.

Predictive Model	**Adjusted R^2**	**CV PRESS**	**Kaggle Score**
Forward Selection	.8736	.147	.14811

Another approach to variable selection consists of beginning with all variables and then removing them one at a time based on p-values. This technique is known as Backward Selection. This approach gave us a total of 33 variables, and even better values for the different metrics.

Predictive Model	**Adjusted R^2**	**CV PRESS**	**Kaggle Score**
Backward Selection	.8907	.14185	.14499

Neither the forward nor backward selections model contain variable interactions (e.g., Variable1 * Variable2). To try to add variable interactions, we created new variables by taking products (logOverallQualXlogGrLivArea, for example) and then created a model using these new variables as well. Using a combination of forward and backward selection (called stepwise selection), a new model was created. This model had the best metrics of all of the models we tested.

Predictive Model	**Adjusted R^2**	**CV PRESS**	**Kaggle Score**
Interaction Model	.9223	.11576	.14013

In addition, the residual plot and Cook's Distance plot (Figure C-5 and Figure C-6 in Appendix C) seem reasonable. The residuals seem to be randomly distributed with no major outliers, and there are no points with Cook's Distance much larger than 0.1 (significantly less than 1).

In conclusion, several models were built using different combinations of variables, number of variables, and techniques of variable determination. The model we chose was the one with the highest adjusted R^2, lowest CV Press, and lowest Kaggle Score. See Appendix D for the full code used in this section.

Appendix A: Full Variable List

SalePrice: The property's sale price in dollars.
MSSubClass: The building class
MSZoning: The general zoning classification
LotFrontage: Linear feet of street connected to property
LotArea: Lot size in square feet
Street: Type of road access
Alley: Type of alley access
LotShape: General shape of property
LandContour: Flatness of the property
Utilities: Type of utilities available
LotConfig: Lot configuration
LandSlope: Slope of property
Neighborhood: Physical locations within Ames city limits
Condition1: Proximity to main road or railroad
Condition2: Proximity to main road or railroad (if a second is present)
BldgType: Type of dwelling
HouseStyle: Style of dwelling
OverallQual: Overall material and finish quality
OverallCond: Overall condition rating
YearBuilt: Original construction date
YearRemodAdd: Remodel date
RoofStyle: Type of roof
RoofMatl: Roof material
Exterior1st: Exterior covering on house
Exterior2nd: Exterior covering on house (if more than one material)
MasVnrType: Masonry veneer type
MasVnrArea: Masonry veneer area in square feet
ExterQual: Exterior material quality
ExterCond: Present condition of the material on the exterior
Foundation: Type of foundation
BsmtQual: Height of the basement
BsmtCond: General condition of the basement
BsmtExposure: Walkout or garden level basement walls
BsmtFinType1: Quality of basement finished area
BsmtFinSF1: Type 1 finished square feet
BsmtFinType2: Quality of second finished area (if present)
BsmtFinSF2: Type 2 finished square feet
BsmtUnfSF: Unfinished square feet of basement area
TotalBsmtSF: Total square feet of basement area
Heating: Type of heating
HeatingQC: Heating quality and condition
CentralAir: Central air conditioning
Electrical: Electrical system
1stFlrSF: First Floor square feet
2ndFlrSF: Second floor square feet
LowQualFinSF: Low quality finished square feet (all floors)
GrLivArea: Above grade (ground) living area square feet
BsmtFullBath: Basement full bathrooms
BsmtHalfBath: Basement half bathrooms
FullBath: Full bathrooms above grade
HalfBath: Half baths above grade
Bedroom: Number of bedrooms above basement level
Kitchen: Number of kitchens
KitchenQual: Kitchen quality
TotRmsAbvGrd: Total rooms above grade (does not include bathrooms)

Functional: Home functionality rating
Fireplaces: Number of fireplaces
FireplaceQu: Fireplace quality
GarageType: Garage location
GarageYrBlt: Year garage was built
GarageFinish: Interior finish of the garage
GarageCars: Size of garage in car capacity
GarageArea: Size of garage in square feet
GarageQual: Garage quality
GarageCond: Garage condition
PavedDrive: Paved driveway
WoodDeckSF: Wood deck area in square feet
OpenPorchSF: Open porch area in square feet
EnclosedPorch: Enclosed porch area in square feet
3SsnPorch: Three season porch area in square feet
ScreenPorch: Screen porch area in square feet
PoolArea: Pool area in square feet
PoolQC: Pool quality
Fence: Fence quality
MiscFeature: Miscellaneous feature not covered in other categories
MiscVal: $Value of miscellaneous feature
MoSold: Month Sold
YrSold: Year Sold
SaleType: Type of sale
SaleCondition: Condition of sale

Appendix B: Analysis 1

I. Model Summaries

North Ames Neighborhood

```
Call:
lm(formula = logSalePrice ~ logGrLivArea, data = names)

Residuals:
     Min       1Q   Median       3Q      Max
-0.46095 -0.07958  0.02882  0.09510  0.52051

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)   8.49273    0.26603   31.92   <2e-16 ***
logGrLivArea  0.47302    0.03725   12.70   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.1577 on 223 degrees of freedom
Multiple R-squared:  0.4196,     Adjusted R-squared:  0.417
F-statistic: 161.2 on 1 and 223 DF,  p-value: < 2.2e-16
```

Edwards Neighborhood

```
Call:
lm(formula = logSalePrice ~ logGrLivArea, data = edwards)

Residuals:
     Min       1Q   Median       3Q      Max
-0.72080 -0.16696 -0.00631  0.17432  0.80470

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)   8.00651    0.53582  14.943  < 2e-16 ***
logGrLivArea  0.51967    0.07505   6.924 4.61e-10 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.2555 on 98 degrees of freedom
Multiple R-squared:  0.3285,     Adjusted R-squared:  0.3217
F-statistic: 47.94 on 1 and 98 DF,  p-value: 4.609e-10
```

Brookside Neighborhood

```
Call:
lm(formula = SalePrice ~ GrLivArea, data = brkside)

Residuals:
   Min     1Q Median     3Q    Max
-57306 -12367  -4445  11780  66160

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 19971.51    9684.72   2.062   0.0438 *
GrLivArea      87.16       7.67  11.364 3.63e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 22390 on 56 degrees of freedom
Multiple R-squared:  0.6975,     Adjusted R-squared:  0.6921
F-statistic: 129.1 on 1 and 56 DF,  p-value: 3.635e-16
```

II. Model Diagnostic Plots

A. North Ames Log-Log Model

```
proc glm data = names plots = all;
model logSalePrice = logGrLivArea / solution;
run;
```

Figure B-1: Fit Diagnostics for Log-Log Model - North Ames

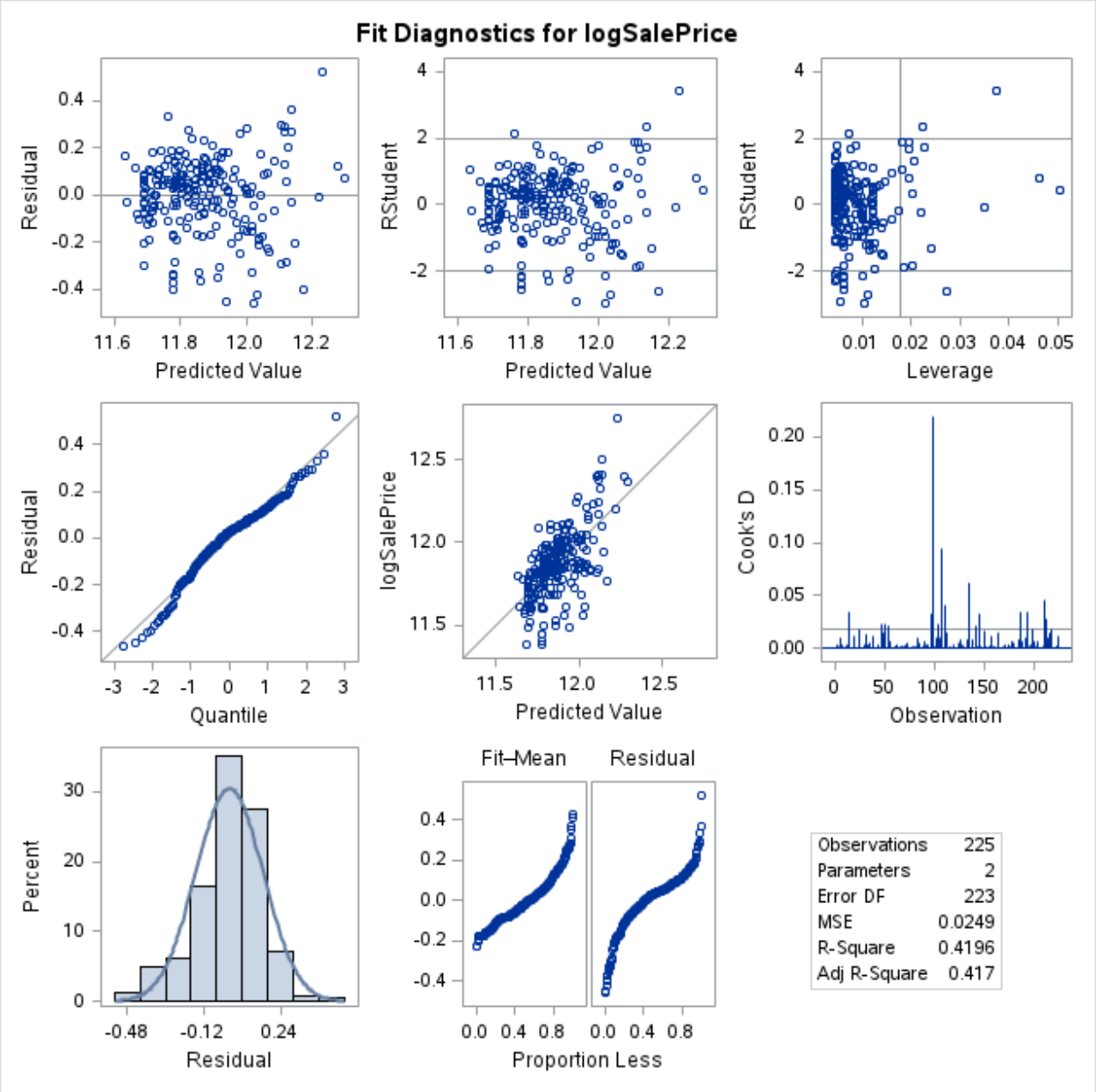


Figure B-2: Scatterplot of Log-Log Model - North Ames

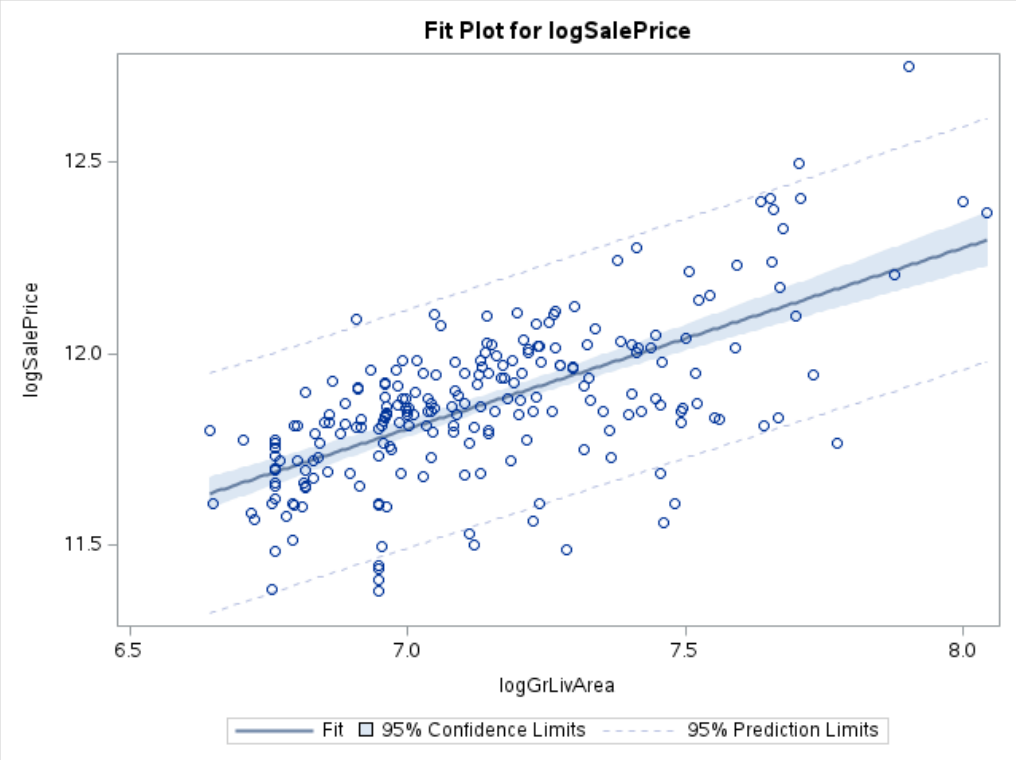
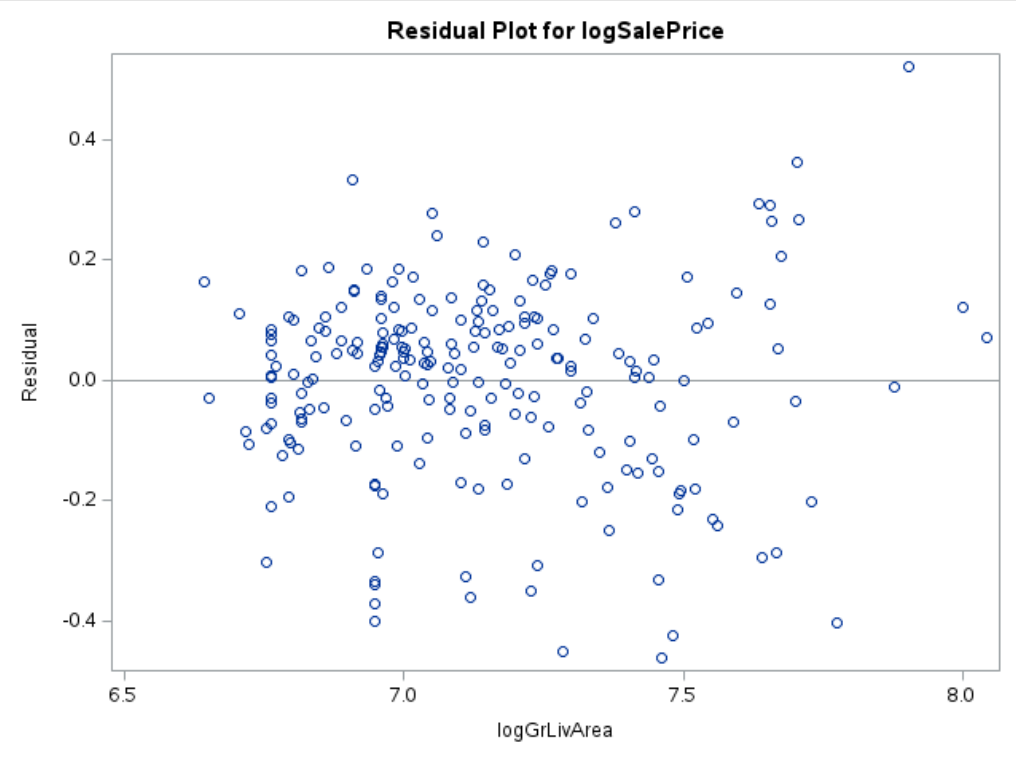


Figure B-3: Scatterplot of Log-Log Model Residuals - North Ames



B. Edwards Log-Log Model

```
proc glm data = edwards plots = all;
model logSalePrice = logGrLivArea / solution;
run;
```

Figure B-4: Fit Diagnostics for Log-Log Model - Edwards

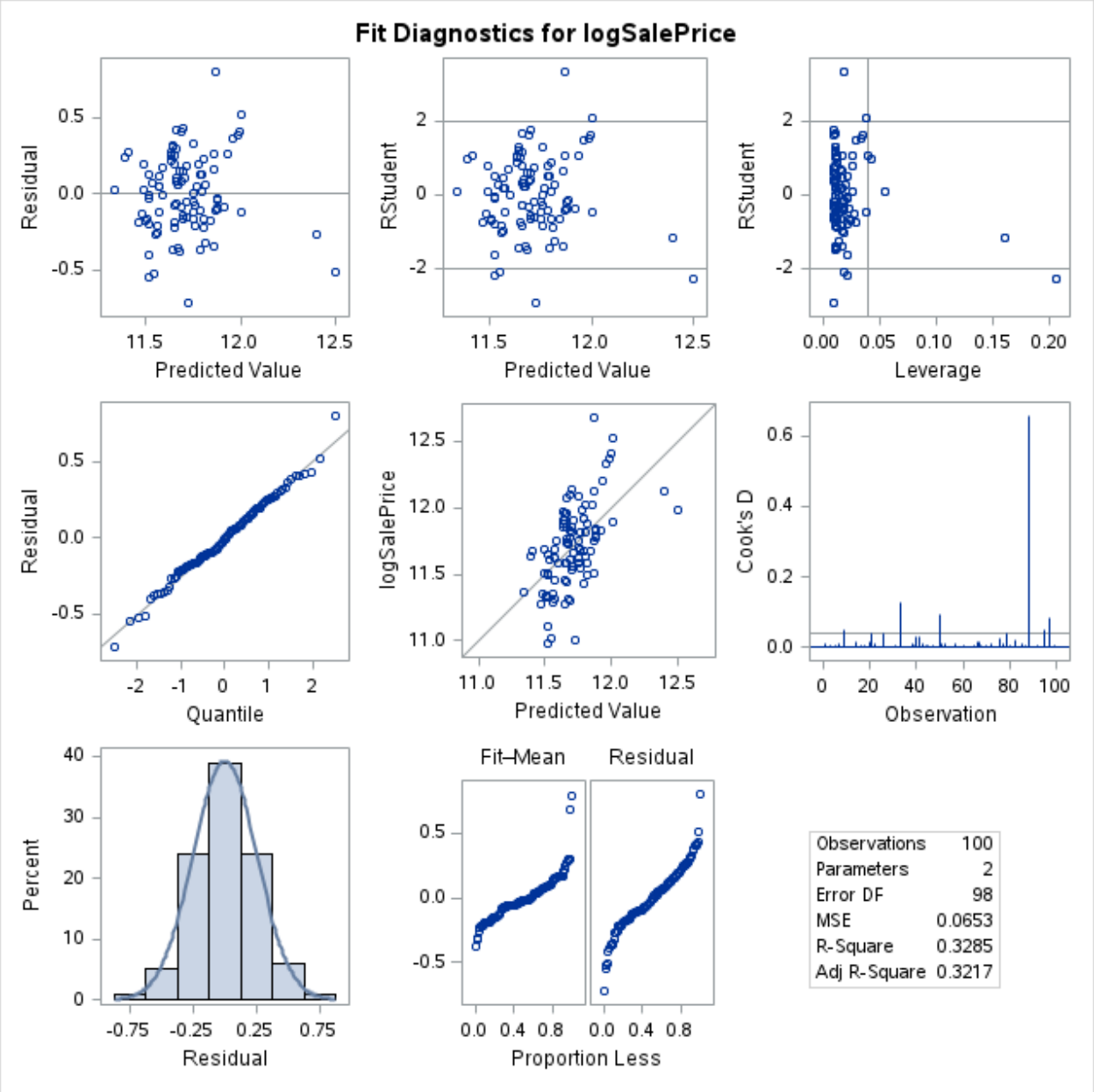


Figure B-5: Scatterplot of Log-Log Model Residuals - Edwards

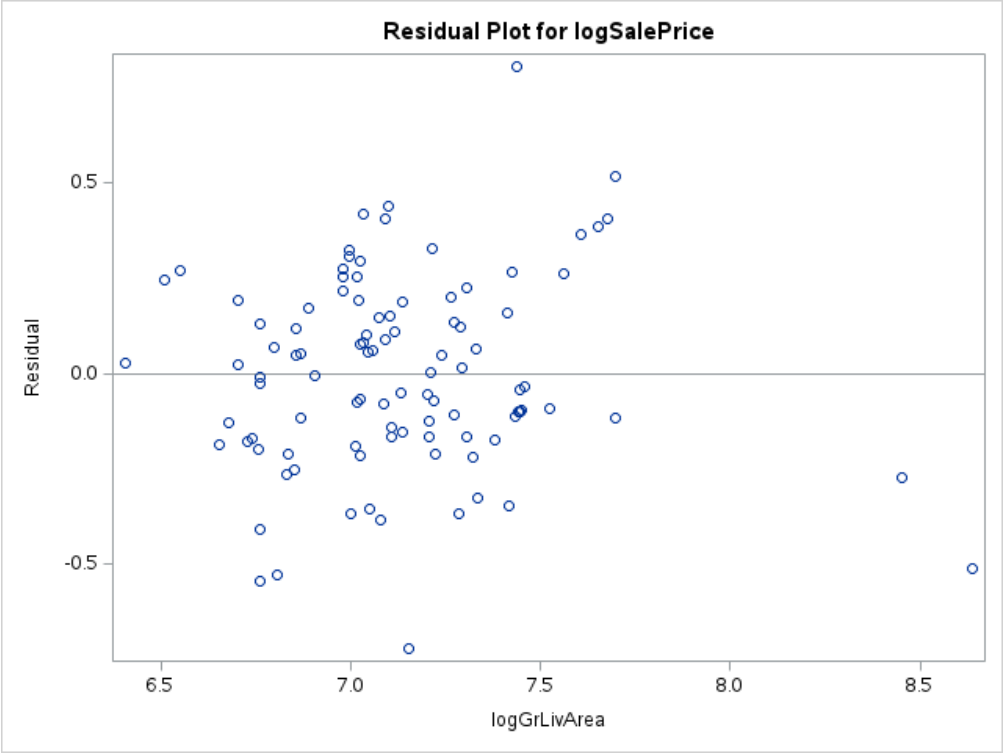
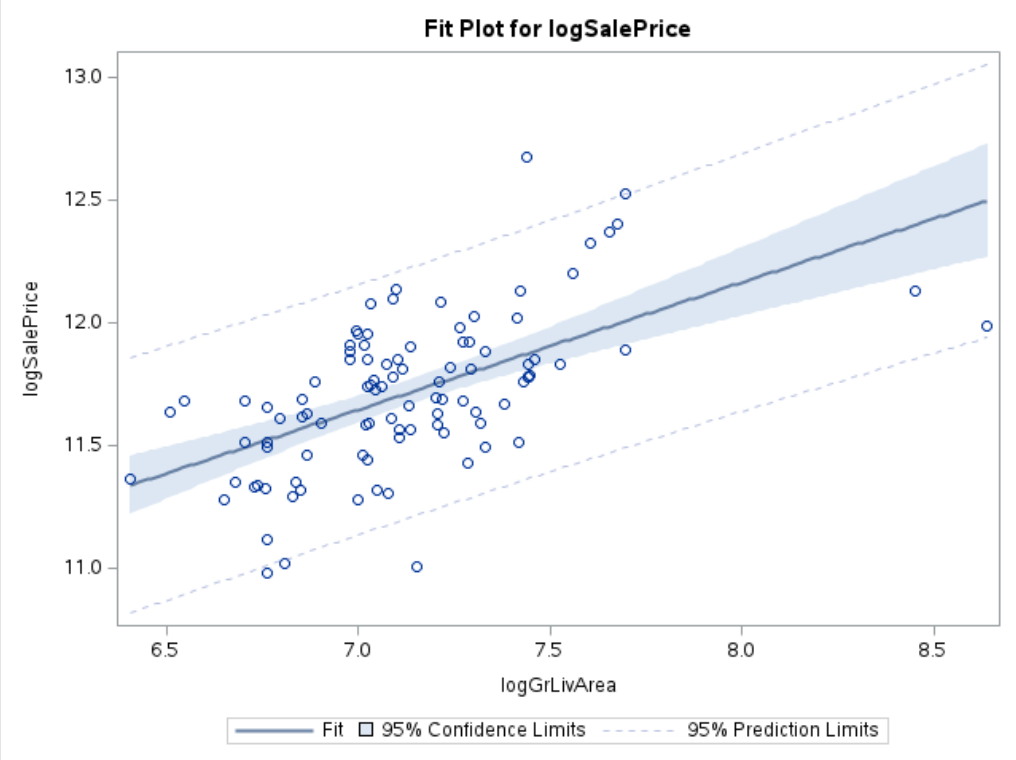


Figure B-6: Scatterplot of Log-Log Model - Edwards



C. Brookside Linear Model

```
proc glm data = brkside plots = all;
model SalePrice = GrLivArea / solution;
run;
```

Figure B-7: Fit Diagnostics for Linear Model - Brookside

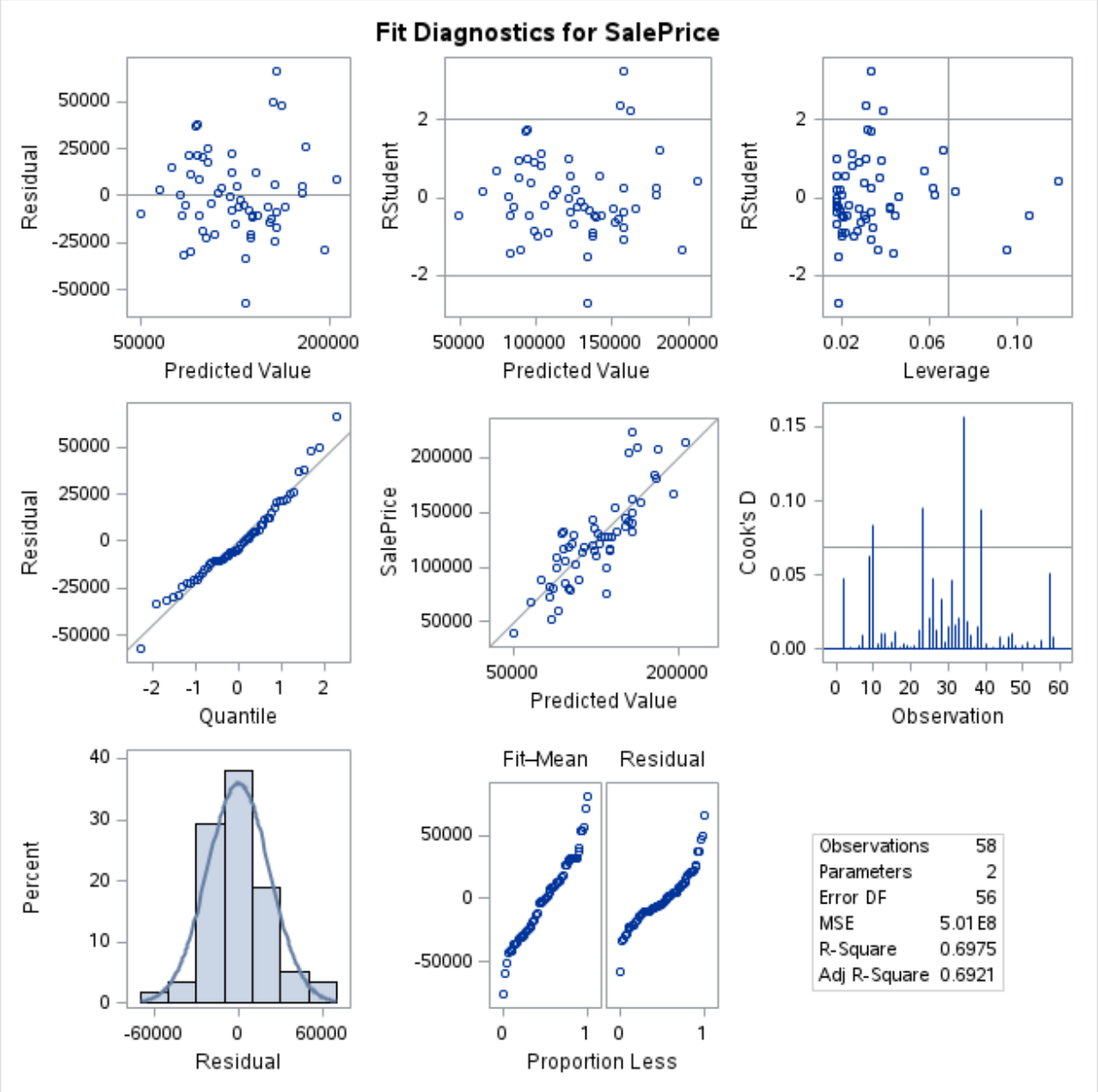


Figure B-8: Scatterplot of Linear Model Residuals - Brookside

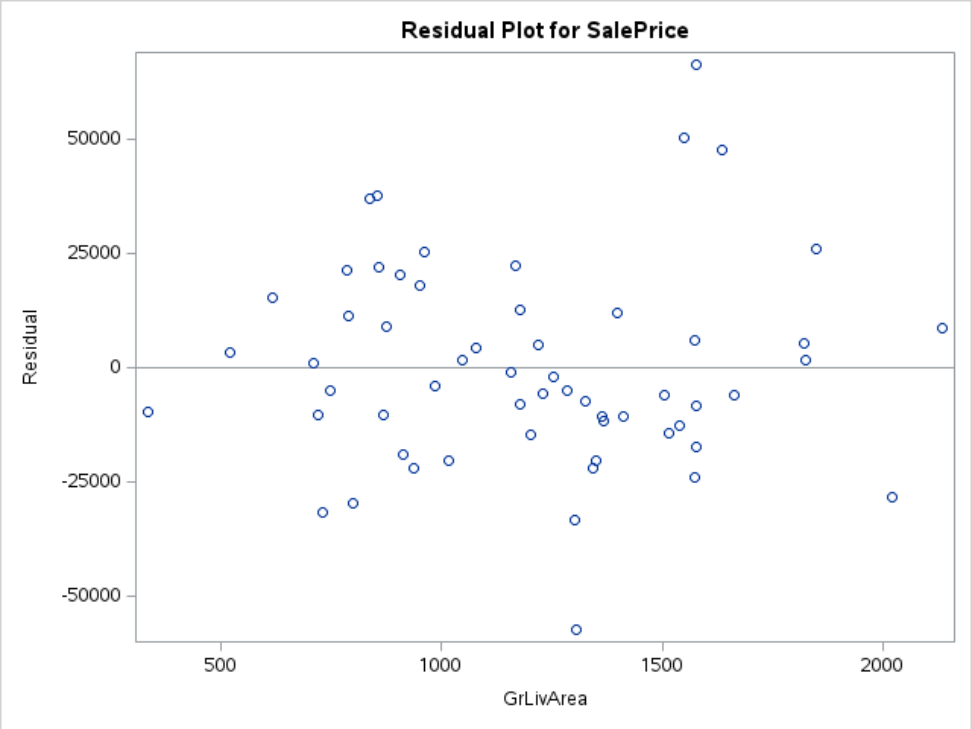
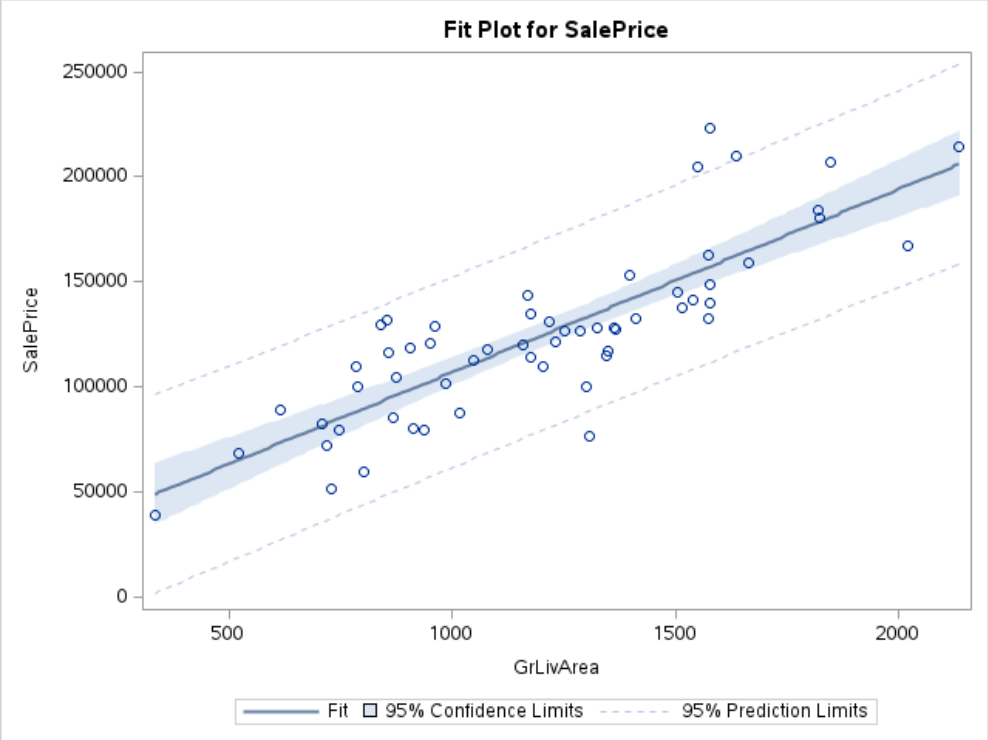


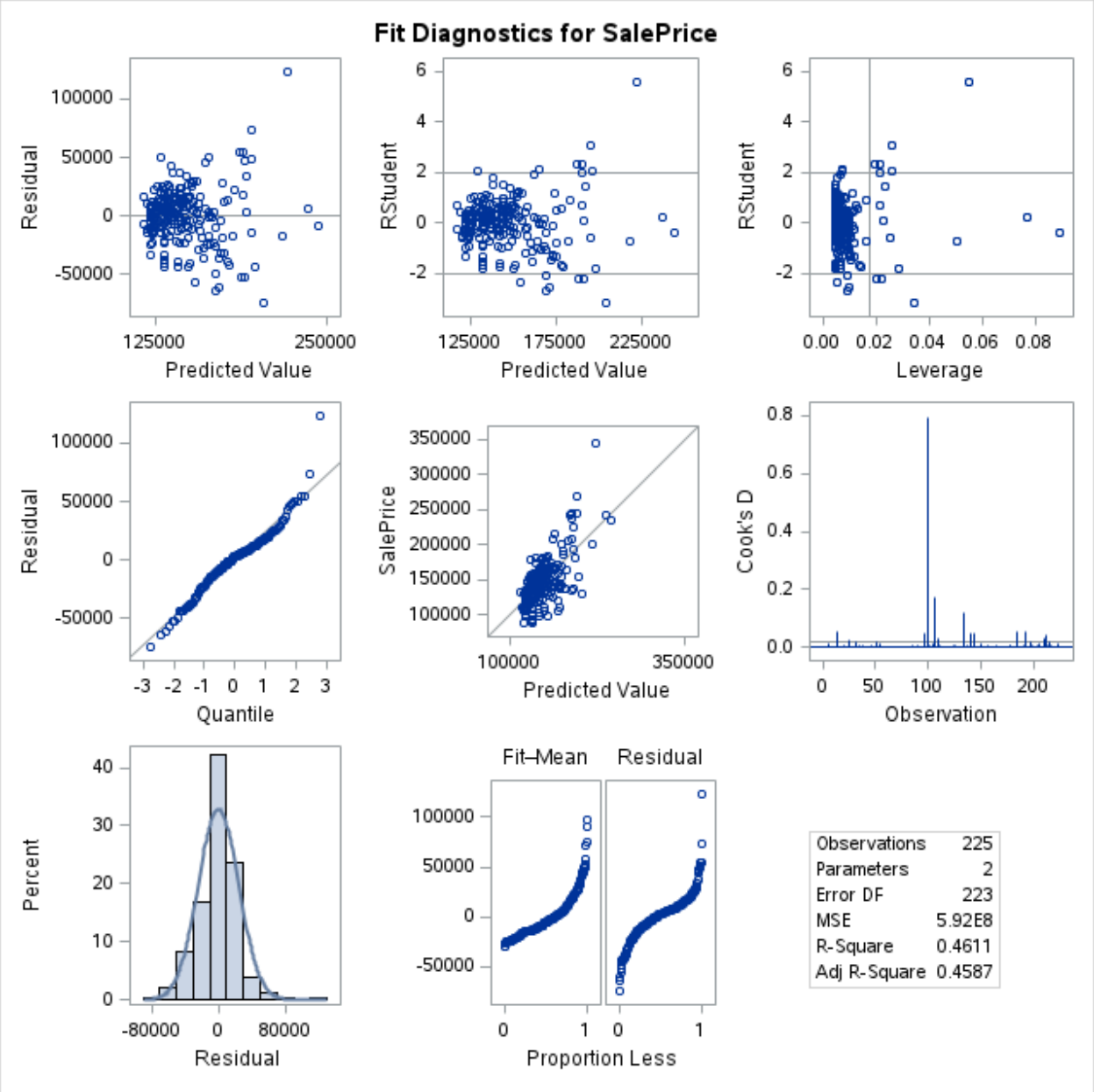
Figure B-9: Scatterplot of Linear Model - Brookside



D. North Ames Linear Model

```
proc glm data = names plots = all;
model SalePrice = GrLivArea / solution;
run;
```

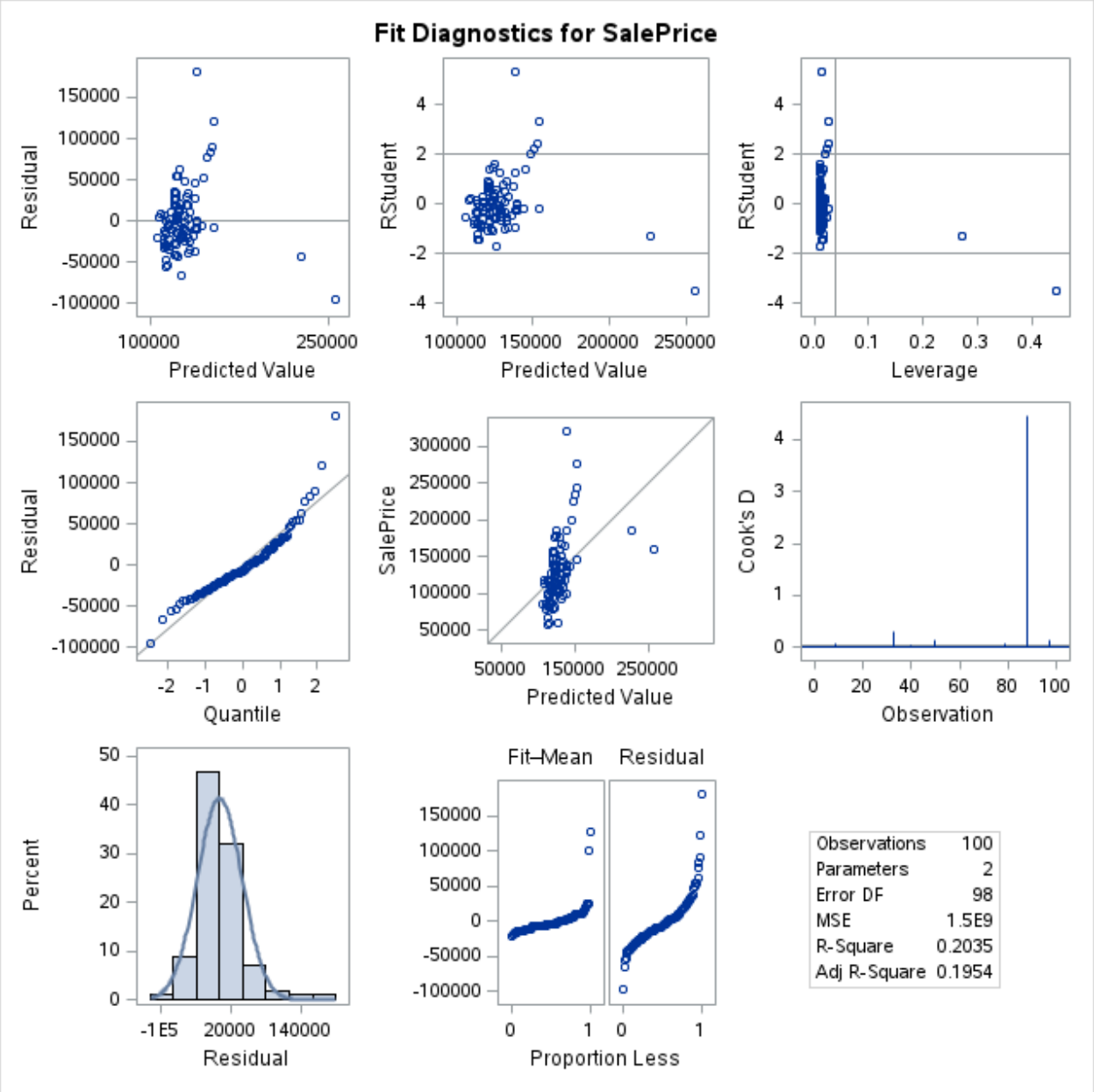
Figure B-10: Fit Diagnostics for Linear Model - North Ames



E. Edwards Linear Model

```
proc glm data = edwards plots = all;
model SalePrice = GrLivArea / solution;
run;
```

Figure B-11: Fit Diagnostics for Linear Model - Edwards



III. Additional Code and Output

A. Confidence Intervals

```
> confint(model.names.loglog)
                 2.5 %    97.5 %
(Intercept)  7.9684756 9.0169796
logGrLivArea 0.3996113 0.5464359

> confint(model.edwards.loglog)
                 2.5 %    97.5 %
(Intercept)  6.9431957 9.0698187
logGrLivArea 0.3707281 0.6686064

> confint(model.brkside)
                2.5 %     97.5 %
(Intercept) 570.69729 39372.3303
GrLivArea    71.79729   102.5278
```

B. CV Press

```
> train_control <- trainControl(method="LOOCV")
>
> # North Ames
> # model.names.loglog = lm(logSalePrice ~ logGrLivArea, data = names)
> train(logSalePrice ~ logGrLivArea, data= names, trControl = train_control,
method="lm")
Linear Regression

225 samples
  1 predictor

No pre-processing
Resampling: Leave-One-Out Cross-Validation
Summary of sample sizes: 224, 224, 224, 224, 224, 224, ...
Resampling results:

  RMSE       Rsquared   MAE
  0.1588008  0.4060751  0.1205799

Tuning parameter 'intercept' was held constant at a value of TRUE
>
> # Edwards
> # model.edwards.loglog = lm(logSalePrice ~ logGrLivArea, data = edwards)
> train(logSalePrice ~ logGrLivArea, data= edwards, trControl = train_control,
method="lm")
```

```
Linear Regression

100 samples
  1 predictor

No pre-processing
Resampling: Leave-One-Out Cross-Validation
Summary of sample sizes: 99, 99, 99, 99, 99, 99, ...
Resampling results:

  RMSE       Rsquared   MAE
  0.2608478  0.2894405  0.2069494

Tuning parameter 'intercept' was held constant at a value of TRUE
>
> # BrookSide
> # model.brkside = lm(SalePrice ~ GrLivArea, data = brkside)
> train(SalePrice ~ GrLivArea, data = brkside, trControl = train_control,
method="lm")
Linear Regression

58 samples
 1 predictor

No pre-processing
Resampling: Leave-One-Out Cross-Validation
Summary of sample sizes: 57, 57, 57, 57, 57, 57, ...
Resampling results:

  RMSE      Rsquared   MAE
  22757.67  0.6764368  17458.14

Tuning parameter 'intercept' was held constant at a value of TRUE
```

Appendix C: Analysis 2 Figures

Figure C-1

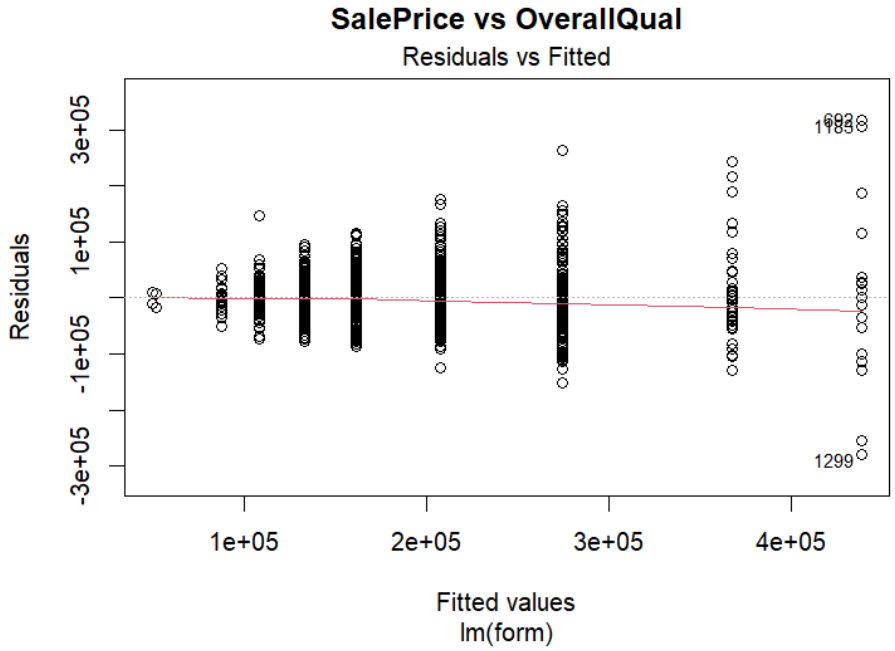


Figure C-2

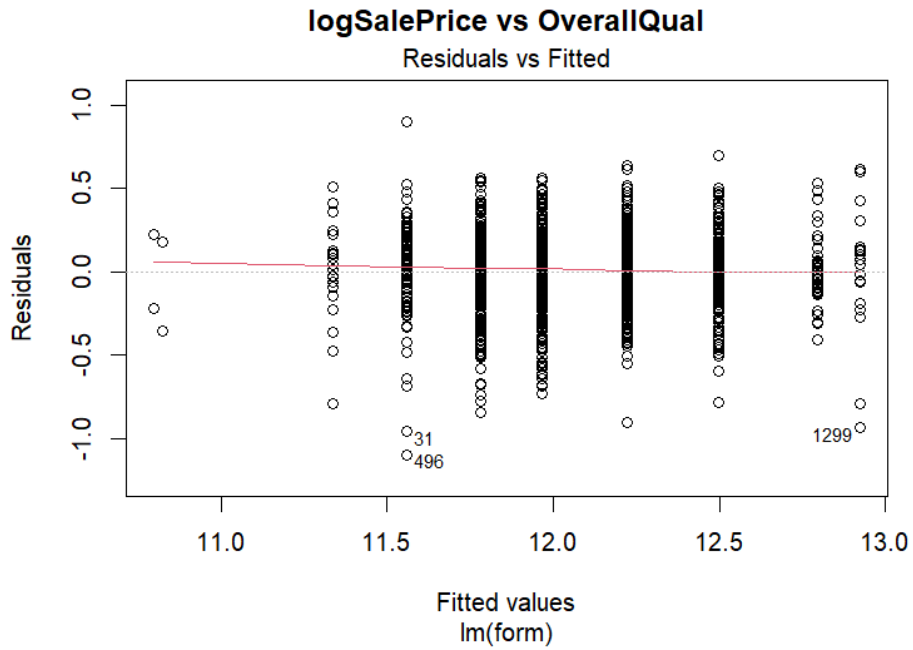


Figure C-3

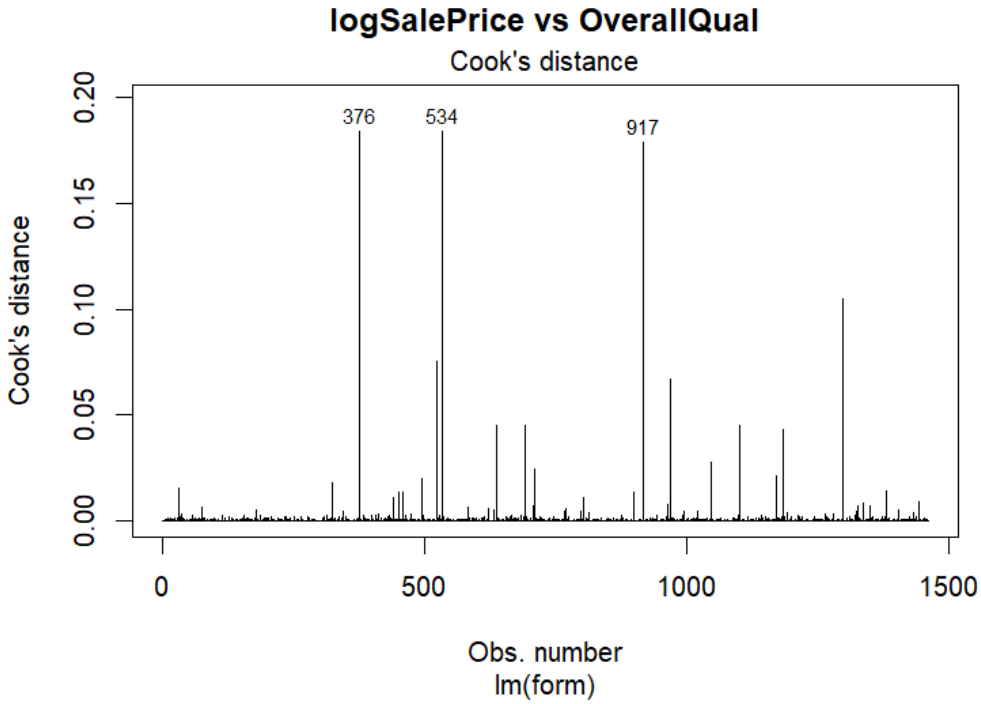


Figure C-4

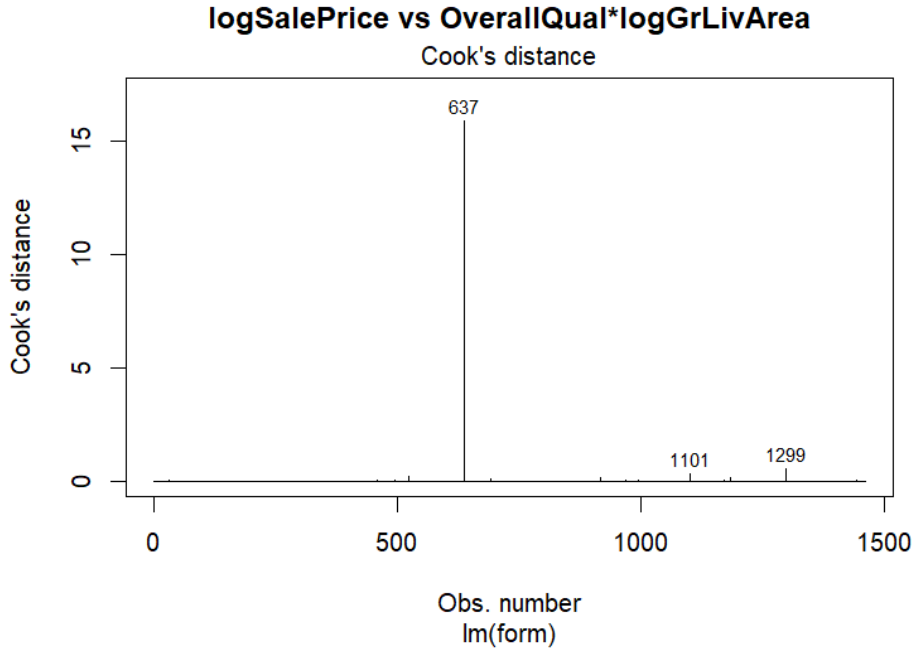


Figure C-5

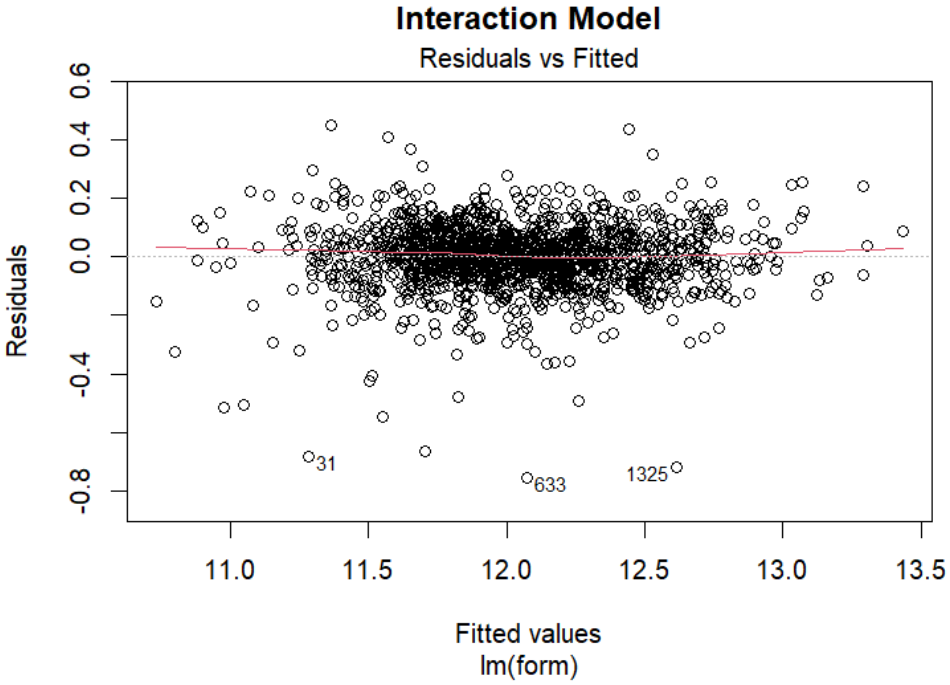
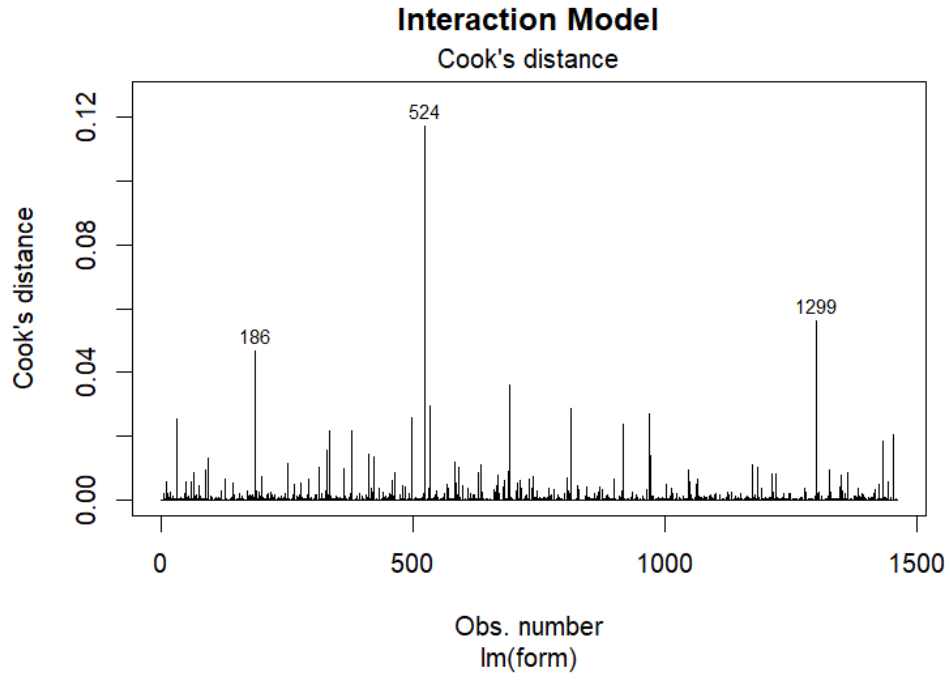


Figure C-6



Appendix D: Analysis 2 R Code

```
# Logs
singleVars <- colnames(data)
singleVars <- singleVars[singleVars != "Id"]
logVars <- c()
nvar <- length(singleVars)
for (ii in 1:nvar){
  var <- singleVars[ii]
  if ((class(data[,var])=="integer")|(class(data[,var])=="numeric")){
    logVar <- paste('log',var,sep='')
    data[,logVar] <- log(data[,var]+1)
    logVars <- c(logVars,logVar)
  }
}

# Squares
singleVars <- singleVars[singleVars != "SalePrice"]
nvar <- length(singleVars)
squareVars <- c()
for (ii in 1:nvar){
  var <- singleVars[ii]
  if ((class(data[,var])=="integer")|(class(data[,var])=="numeric")){
    squareVar <- paste(var,'Squared',sep='')
    data[,squareVar] <- (data[,var])^2
    squareVars <- c(squareVars,squareVar)
  }
}

# One hot encoding
data <- one_hot(as.data.table(data))
data <- as.data.frame(data)

# Make cross products of variables
data$SalePrice <- c() # since we're using logSalePrice
singleVars <- colnames(data)
singleVars <- singleVars[singleVars != "logSalePrice"]
singleVars <- singleVars[singleVars != "Id"]
nvar <- length(singleVars)
for (ii in 1:nvar){
  for(jj in 1:nvar){
    if (ii<jj){
      if(ii %% 10){
        print(paste('var: ', ii))
      }
      var1 <- singleVars[ii]
      var2 <- singleVars[jj]
      new_col <- data[,var1]*data[,var2]
```

```
      if(sum(new_col)!=0){
        new_var <- paste(var1,var2,sep="X")
        data[,new_var]<-new_col
      }
    }
  }
}

# Checking out the correlation values between SalePrice and the other
variables
corr_df <- data.frame(Variables=colnames(data))
corr_df$corr <- 0
corr_df$p <- 0
nvar <- nrow(corr_df)
for(ii in 1:nvar){
  var<-corr_df[ii,'Variables']
  test <- cor.test(unclass(data[,var]),unclass(data[,'SalePrice']))
  corr_df$corr[ii] <- abs(test$estimate)
  corr_df$p[ii] <- test$p.value
}

# Try a basic model with just OverallQual
form1 <- as.formula('SalePrice ~ OverallQual')
mod1 <- lm(form1,data = data)
summary(mod1) # Residual standard error: 44780 Multiple R-squared:  0.6842,
Adjusted R-squared:  0.6822
train(form1, data=data, trControl = train_control, method="lm")
# RMSE      Rsquared  MAE
# 45287.16  0.6748401  31015.72

# Try getting the Kaggle score
data_test <- read.csv('C:/Users/aabro/OneDrive/Desktop/SMU
Program/Classes/Statistics/Final Project/housing_test.csv',header=TRUE)
data_test$OverallQual <- as.factor(data_test$OverallQual)
data_submission <- data.frame(Id = data_test$Id)
data_submission$SalePrice <- predict(mod1, newdata=data_test)
min(data_submission$SalePrice) # 50k, not too low
min(data$SalePrice) # 35k
max(data_submission$SalePrice) # 438k, kinda high but not super high
max(data$SalePrice) # 755k
write.csv(data_submission, 'C:/Users/aabro/OneDrive/Desktop/SMU
Program/Classes/Statistics/Final Project/Submissions/mod1.csv')
# Kaggle Score = .22814

# Try a log variable
form <- as.formula('logSalePrice ~ OverallQual')
mod <- lm(form,data = data)
```

```
summary(mod) # Residual standard error: 0.229 Multiple R-squared:  0.6734,
Adjusted R-squared:  0.6714
train(form, data=data, trControl = train_control, method="lm")
# RMSE      Rsquared  MAE
# 0.2307228  0.6661703  0.1735684, slightly lower Rsquared
data_submission <- data.frame(Id = data_test$Id)
data_submission$logSalePrice <- predict(mod, newdata=data_test)
data_submission$SalePrice <- exp(data_submission$logSalePrice) - 1 # Need to
convert back from the log
min(data_submission$SalePrice) # 43k
max(data_submission$SalePrice) # 548k
write.csv(data_submission, 'C:/Users/aabro/OneDrive/Desktop/SMU
Program/Classes/Statistics/Final Project/Submissions/mod5.csv')
# Kaggle = .22613, interesting, slightly better than without the

# Try the two variable model that Dr Sadler provided
form <- as.formula('SalePrice ~ FullBath + GrLivArea')
mod <- lm(form,data = data)
summary(mod) # Residual standard error: 54860 Multiple R-squared:  0.5238,
Adjusted R-squared:  0.5231
train(form, data=data, trControl = train_control, method="lm")
# RMSE      Rsquared  MAE
# 55156.95  0.5176485  36733.12
data_submission <- data.frame(Id = data_test$Id)
data_submission$SalePrice <- predict(mod, newdata=data_test)
min(data_submission$SalePrice) # 43k
max(data_submission$SalePrice) # 548k
write.csv(data_submission, 'C:/Users/aabro/OneDrive/Desktop/SMU
Program/Classes/Statistics/Final Project/Submissions/mod4.csv')
# Kaggle = .28586, funny worse than just OverallQual

# Looking at assumption plots for the first few models
form <- as.formula('SalePrice ~ OverallQual')
mod <- lm(form,data = data)
plot(mod,which = 1:6)

form <- as.formula('SalePrice~GrLivArea + FullBath')
mod <- lm(form,data = data)
plot(mod,which = 1:6)

form <- as.formula('logSalePrice ~ OverallQual')
mod <- lm(form,data = data)
plot(mod,which = 1:6)

form <- as.formula('logSalePrice ~ OverallQual*logGrLivArea')
mod <- lm(form,data = data)
plot(mod,which = 1:6)
```

```
# Getting the forward model
fit_full <- lm(SalePrice ~ ., data = train_data)
forward_ols_model <- ols_step_forward_p(fit_full,penter=0.0001,details=TRUE)
forward_ols_model$predictors
# [1] "OverallQual"               "GrLivArea"                 "Neighborhood"
# [4] "RoofMatl"                  "BsmtExposure"              "TotalBsmtSF"
# [7] "BsmtUnfSF"                 "YearRemodSince1950Squared" "logLotArea"
# [10] "KitchenAbvGr"              "SaleCondition"
"logYearBuiltSince1870"
# [13] "Functional"                "KitchenQual"               "BsmtQual"
# [16] "GarageCarsSquared"         "PoolQC"

# Try the forward model, without interactions for now
form <- as.formula('logSalePrice ~
OverallQual+logOverallQual+OverallQualSquared+Neighborhood+

logGrLivArea+GrLivArea+GarageYrSince1900+logBsmtFinSF1+logYearRemodAdd+
                   BsmtQual+GarageArea+logTotalBsmtSF')
mod <- lm(form,data = data_no_na)
summary(mod) # Residual standard error: 0.142 Multiple R-squared:  0.8769,
Adjusted R-squared:  0.8736
train(form, data=data_no_na, trControl = train_control, method="lm")
# RMSE      Rsquared  MAE
# 0.1469999  0.8645639  0.09965149
data_submission <- data.frame(Id = data_test_no_na$Id)
data_submission$logSalePrice <- predict(mod, newdata=data_test_no_na)
data_submission$SalePrice <- exp(data_submission$logSalePrice) - 1 # Need to
convert back from the log
min(data_submission$SalePrice) # 43k
max(data_submission$SalePrice) # 548k
write.csv(data_submission, 'C:/Users/aabro/OneDrive/Desktop/SMU
Program/Classes/Statistics/Final Project/Submissions/mod9.csv')
# Kaggle = .14811, oh that's a ton better

# What happens if we try ols_step_backward
fit_full <- lm(logSalePrice ~ ., data = data_no_na)
backward_ols_model <-
ols_step_backward_p(fit_full,penter=0.0001,progress=TRUE)
backward_ols_model$removed
# [1] "BsmtFinSF1"           "logBsmtFinSF2"        "logYearRemodAdd"
"MiscValSquared"
# [5] "logMiscVal"           "EnclosedPorchSquared" "EnclosedPorch"
"GarageArea"
# [9] "GrLivArea"            "ExterQual"            "logMasVnrArea"
"MiscVal"
# [13] "logEnclosedPorch"     "BsmtFinSF2"           "BsmtFinSF2Squared"
"GarageAreaSquared"
# [17] "logGarageYrBlt"       "MasVnrArea"
```

```
colnames(data_no_na)

full_list <- colnames(data_no_na)[!(colnames(data_no_na) %in%
backward_ols_model$removed)]
paste(full_list, collapse = '+')

# Trying the backwards model, which is really long
form <- as.formula('logSalePrice ~
Neighborhood+OverallQual+GarageCars+MSSubClass+YearBuilt+TotRmsAbvGrd+GarageYr
Blt+X1stFlrSF+TotalBsmtSF+YearRemodAdd+BsmtQual+FireplaceQu+GarageYrSince1900+
logGrLivArea+logOverallQual+logGarageArea+logGarageCars+logMSSubClass+logYearB
uilt+logTotRmsAbvGrd+logX1stFlrSF+logTotalBsmtSF+logBsmtFinSF1+logGarageYrSinc
e1900+GrLivAreaSquared+OverallQualSquared+GarageCarsSquared+MSSubClassSquared+
MasVnrAreaSquared+YearBuiltSquared+TotRmsAbvGrdSquared+GarageYrBltSquared+X1st
FlrSFSquared+TotalBsmtSFSquared+YearRemodAddSquared+BsmtFinSF1Squared+GarageYr
Since1900Squared+OverallQualFactor')
mod <- lm(form,data = data_no_na)
summary(mod) # Residual standard error: 0.1296 Multiple R-squared:  0.9003,
Adjusted R-squared:  0.8948
train(form, data=data_no_na, trControl = train_control, method="lm")
# RMSE      Rsquared  MAE
# 0.1414166  0.8748766  0.09541749
data_submission <- data.frame(Id = data_test_no_na$Id)
data_submission$logSalePrice <- predict(mod, newdata=data_test_no_na)
for (ii in 1:nrow(data_submission)) {
  if (data_submission$logSalePrice[ii] < min(data$logSalePrice)){
    data_submission$logSalePrice[ii] = min(data$logSalePrice)
  }
}
for (ii in 1:nrow(data_submission)) {
  if (data_submission$logSalePrice[ii] > max(data$logSalePrice)){
    data_submission$logSalePrice[ii] = max(data$logSalePrice)
  }
}
data_submission$SalePrice <- exp(data_submission$logSalePrice) - 1 # Need to
convert back from the log
min(data_submission$SalePrice)
max(data_submission$SalePrice)
write.csv(data_submission, 'C:/Users/aabro/OneDrive/Desktop/SMU
Program/Classes/Statistics/Final Project/Submissions/mod13.csv')
# Kaggle = .14537, best yet

# Code to select variables for a large model with interactions
# Parameters
start_formula = ''
start_num <- 1
seed <- 1
```

```
txt_file <- 'C:/Users/aabro/OneDrive/Desktop/SMU
Program/Classes/Statistics/Final Project/OldApproachFormula.txt'
start_best <- 1000000000
amount_to_improve <- 0
var_to_predict <- 'logSalePrice'
num_no_change <- 50
train_control <- trainControl(method="LOOCV")

# First get a list of the p-values
corr_df <- corr_df[corr_df$Variables!='logSalePrice',]
var_df <- corr_df
var_df$num <- 0
# var_df$p <- 0
var_df$metric <- 0
var_df$to_use <- 0
var_df$curr_formula <- ""
num_since_last_change <- 0
nvar <- nrow(var_df)
singleVars <- names(data)
singleVars <- singleVars[singleVars != "logSalePrice"]

# Sort the p-values
sort_order <- order(var_df$p,-var_df$corr)

# Loop through all of the columns, add the next variable to the list
if (start_formula==''){
  vars_to_use <- c()
} else {
  vars_to_use <- unlist(strsplit(start_formula, "\\+"))
}
best_passed <- 0
best_metric <- start_best
for (ii in start_num:nvar){
  var<-singleVars[sort_order[ii]]
  vars_to_use_loop <- c(vars_to_use,var)
  formula_str <- paste(var_to_predict," ~ ",
paste(vars_to_use_loop,collapse="+"))
  formula_obj <- as.formula(formula_str)
  curr_p <- var_df$p[sort_order[ii]]
  if(curr_p > 0.001){ # Don't want the variables to not be super predictive
    break
  }
  if(sum(data[,var]!=0)<length(data[,var])*0.01){
    next # Don't want too few numbers
  }

  # Get the CV Press
  model <- lm(formula_obj, data = data)
```

```
  loocv_test <- train(formula_obj, data=data, trControl = train_control,
method="lm")
  metric <- loocv_test$results$RMSE
  var_df$metric[var_df$Variable==var] <- metric
  var_df$num[var_df$Variable==var] <- ii

  # Check if that variable should be added
  if(best_metric-amount_to_improve>metric){
    best_metric <- metric
    vars_to_use <- c(vars_to_use,var)
    var_df$to_use[var_df$Variable==var] <- 1
    var_df$curr_formula[var_df$Variable==var] <-
paste(vars_to_use,collapse="+")
    num_since_last_change <- 0
    file_str <- paste(readLines(txt_file), collapse="\n")
    writeLines(paste(file_str,"\n",'num: ',ii,' var: ',var,
                     ' metric: ', metric,
                     ' formula_str: ',paste(vars_to_use,collapse="+"),sep=""),
               txt_file)
  }

  # Print something to the screen to see general progress
  print(paste('num: ',ii,' var: ',var,' metric: ', metric,
              ' num_since_last_change: ',num_since_last_change, sep=""))

  # If the loop has been going on for too long, just break out of it
  num_since_last_change <- num_since_last_change + 1
  if(num_since_last_change == num_no_change){
    break
  }
}

# Best model combination from the variables that the above process generated
form <- as.formula('logSalePrice ~
logOverallQualXlogGrLivArea+logOverallQualXlogX1stFlrSF+OverallQualXlogLotArea
+logLotAreaXlogOverallQual+logGarageCarsXOverallQualSquared+logTotalBsmtSFXOve
rallQualSquared+logYearRemodAddShiftedXOverallQualSquared+OverallQualSquaredXY
earBuiltSquared+OverallQualSquaredXYearRemodAddSquared+logFullBathXOverallQual
Squared+CentralAir_YXOverallQualSquared+logGrLivAreaXYearRemodAddSquared+logOv
erallCondXOverallQualSquared+Condition2_NormXOverallQualSquared+FullBathXOvera
llQualSquared+logOverallQualXlogTotRmsAbvGrd+Heating_GasAXOverallQualSquared+P
avedDrive_YXOverallQualSquared+OverallQualXlogOverallCond+GarageQual_TAXOveral
lQualSquared+MiscFeature_NAXOverallQualSquared+Functional_TypXOverallQualSquar
ed+OverallQualXCentralAir_Y+OverallQualSquaredXYearRemodAddShiftedSquared+Tota
lBsmtSFXGarageCars+BedroomAbvGrXOverallQualSquared+GarageCarsXlogTotalBsmtSF+l
ogOverallQualXlogYearRemodAddShifted+logFireplacesXOverallQualSquared+logGarag
eCarsXlogYearBuiltShifted+BsmtCond_TAXOverallQualSquared+LandSlope_GtlXOverall
QualSquared+FullBathXlogGarageArea+GrLivAreaXlogTotRmsAbvGrd+logFullBathXlogGa
```

```
rageArea+logYrSoldShiftedXGarageCarsSquared+YearRemodAddShiftedXlogOverallQual
+logLotAreaXlogGrLivArea+GarageCarsXMiscFeature_NA+Foundation_PConcXGrLivArea+
logFireplacesXYearBuiltShiftedSquared+Condition1_NormXGrLivAreaSquared+Electri
cal_SBrkrXGrLivAreaSquared+TotalBsmtSFXElectrical_SBrkr+GarageQual_TAXFullBath
Squared+Heating_GasAXlogGarageCars+logGarageCarsXX1stFlrSFSquared+Condition1_N
ormXGarageCarsSquared+logYrSoldShiftedXYearBuiltShiftedSquared+GarageCond_TAXl
ogFullBath+MSZoning_RLXGrLivArea+Foundation_PConcXTotRmsAbvGrd+OverallQualXYrS
oldShifted+logGarageCarsXlogOpenPorchSF+GarageCarsXlogKitchenAbvGr+logBsmtUnfS
FXYearBuiltShiftedSquared+OverallQualXlogOpenPorchSF+logX1stFlrSFXGrLivAreaSqu
ared+TotalBsmtSFSquaredXGarageCarsSquared+MSZoning_RLXGarageArea+Functional_Ty
pXYearBuiltShiftedSquared+GrLivAreaXlogBedroomAbvGr+CentralAir_YXYearRemodAddS
hifted+Street_PaveXYearBuiltShifted+YearRemodAddXlogTotRmsAbvGrd+LandContour_L
vlXGrLivAreaSquared+PavedDrive_YXTotRmsAbvGrdSquared+X1stFlrSFXlogFireplaces+l
ogGarageAreaXGarageAreaSquared+ExterQual_GdXGrLivAreaSquared')
mod <- lm(form,data = data)
summary(mod) # Residual standard error: 0.1114 Multiple R-squared:  0.9261,
Adjusted R-squared:  0.9223
train(form, data=data, trControl = train_control, method="lm")
# RMSE      Rsquared  MAE
# 0.1157588  0.9160383  0.08214109
data_submission <- data.frame(Id = data_test$Id)
data_submission$logSalePrice <- predict(mod, newdata=data_test)
for (ii in 1:nrow(data_submission)) {
  if (data_submission$logSalePrice[ii] < min(data$logSalePrice)){
    data_submission$logSalePrice[ii] = min(data$logSalePrice)
  }
}
for (ii in 1:nrow(data_submission)) {
  if (data_submission$logSalePrice[ii] > max(data$logSalePrice)){
    data_submission$logSalePrice[ii] = max(data$logSalePrice)
  }
}
data_submission$SalePrice <- exp(data_submission$logSalePrice) - 1 # Need to
convert back from the log
# min(data_submission$SalePrice)
# max(data_submission$SalePrice)
write.csv(data_submission, 'C:/Users/aabro/OneDrive/Desktop/SMU
Program/Classes/Statistics/Final Project/Submissions/mod38.csv')
# Kaggle = .14013, best yet

# Looking at assumptions plots for the interaction model
plot(mod,which = 1:6)
```