Housing Price Prediction

April 13th, 2021

Stat 306 Project

Name: Ahmed Abdellatif	**Student Number:** 43067537
Name: Karina Grewal	**Student Number:** 54324488
Name: Chen Yu (Eric) Liu	**Student Number:** 12260162
Name: Shuhan Yang	**Student Number:** 56884745

Contents

1 Introduction

Housing prices is something that everyone will encounter moving into adulthood. As university students it would be beneficial for us to gain some exposure as to what contributes to the varying prices of homes.

1.1 Data Source

The data was collected from 2006-2010 by the Ames City Assessor's Office. This data represents individual households in Ames, Iowa, USA. Dean De Cock, a professor at Truman State University, (the creator of the dataset) received the raw data directly from the Ames City Assessor's Office (a part of the Iowa Government Office). After receiving the raw data, he compiled it into a tidier data set which is what we will be using for our analysis.

1.2 Motivation

The goal of this dataset is to predict the sale price of residences based on 73 variables explanatory features (including 33 quantitative variables, and 40 factor variables). Through performing regressions using the techniques taught in this course, we can understand the importance of each feature from the magnitude of the coefficients and the p-values. In doing so, we hope to obtain an understanding into the composition of housing prices.

Vancouver is one of the most un-affordable cities in North America. Although the structure of housing price composition in Vancouver will likely be different, understanding the price composition of Ames Housing Price will still provide meaningful insights. For example, we may see that Central Air Conditioning could have a large impact on the model, and we could extrapolate that knowledge to benefit our personal housing searches in Vancouver once COVID-19 ends.

1.3 Hypothesis of Interest

In this analysis, since our data has many variables, we want to find out the most influential explanatory variables on the predictor (SalePrice). From anecdotal experience, we expect that the three most important variables in calculating house prices will be:

- LotArea: Lot size in square feet
- YearBuilt: Original construction date
- OverallQual: Overall material and finish quality

Our null hypothesis is as follows: These three explanatory variables above, have no statistically significant effect on the ideal model.

$$\beta_{\text{LotArea}} = \beta_{\text{YearBuilt}} = \beta_{\text{OverallQual}} = 0 \tag{1}$$

Conversely, our alternative hypothesis is that at least one of the three variables does have a statistically significant effect on the ideal model.

$$\beta_{\text{LotArea}} \neq 0 \vee \beta_{\text{YearBuilt}} \neq 0 \vee \beta_{\text{OverallQual}} \neq 0 \tag{2}$$

2 Data Cleaning

In this section, we will explore the raw dataset. From there, we will discuss data cleaning techniques performed on the raw data, and highlight our initial data observation findings.

2.1 Data Overview

The dataset contains 81 variables in total. Of the 81 variables, there is one identifier variable ID, 79 explanatory variables, and 1 response variable Sale Price (reported in USD). Of the 79 explanatory variables, there are 35 quantitative variables, and 44 factor variables. There are a total of 1460 observations in the dataset. However as described below, there were many missing data values and as such we had to remove some variables altogether so as not to create bias in our results.

Detailed Descriptions of the 80 variables are included in the Appendix A.

2.2 Data Cleaning: Handling Missing Data Values

The dataset is not complete, and includes rows with missing data. There are two common approaches to handling missing data values. First, impute the missing data values, often by extrapolating information from its nearest neighbours. Second, simply throwing out observations that contain missing data. Given that data imputation is beyond the scope of Stat 306, we decided to simply omit the entries that contain missing data values.

2.2.1 Omitting Columns with Substantial NA Values

Upon initial inspection of the dataset, we realized that every observation had at least one data field missing. This is because there are multiple columns that have missing data values for all but a few observations.

Our approach to handling missing data values is to omit the observation. Keeping columns that have a large percentage of missing data would shrink the training data size considerably, and will drastically increase the bias of our prediction.

Therefore, we decided to omit columns that have more than 10% (146) missing data values completely to keep our training dataset as large as possible. A total of 6 columns - LotFrontage, Alley, FireplaceQu, PoolQC, Fence, and MiscFeature - were removed. There are now 73 explanatory variables in total, including 33 quantitative variables, and 40 factor variables.

2.2.2 Omitting Observations with NA Values

We were left with 73 explanatory variables after pruning columns with over 10% (146) missing values. However, there were still some observations with missing data values. To address this, we simply omitted every observation that had at least 1 missing value. In total 122 (8%) observations were removed, and we were left with 1338 observations.

3 Data Analysis

We visualized the cleaned data to obtain an understanding of the distribution of variables. To do so, we plotted every explanatory variable against the response variable, Sale Price. In doing so, we hope to obtain a rudimentary understanding of the relationships between each individual explanatory variable and Sale Price.

3.1 Quantitative Variables

Figure 1: Quantitative Variables Plotted Against Sale Price

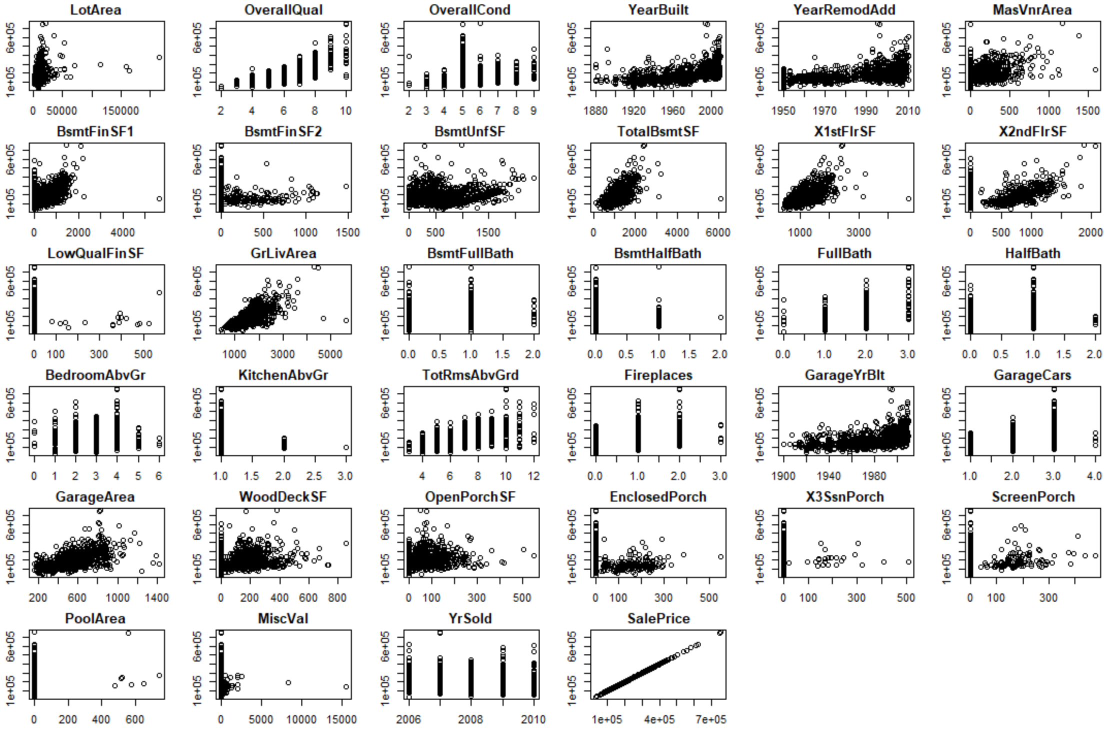
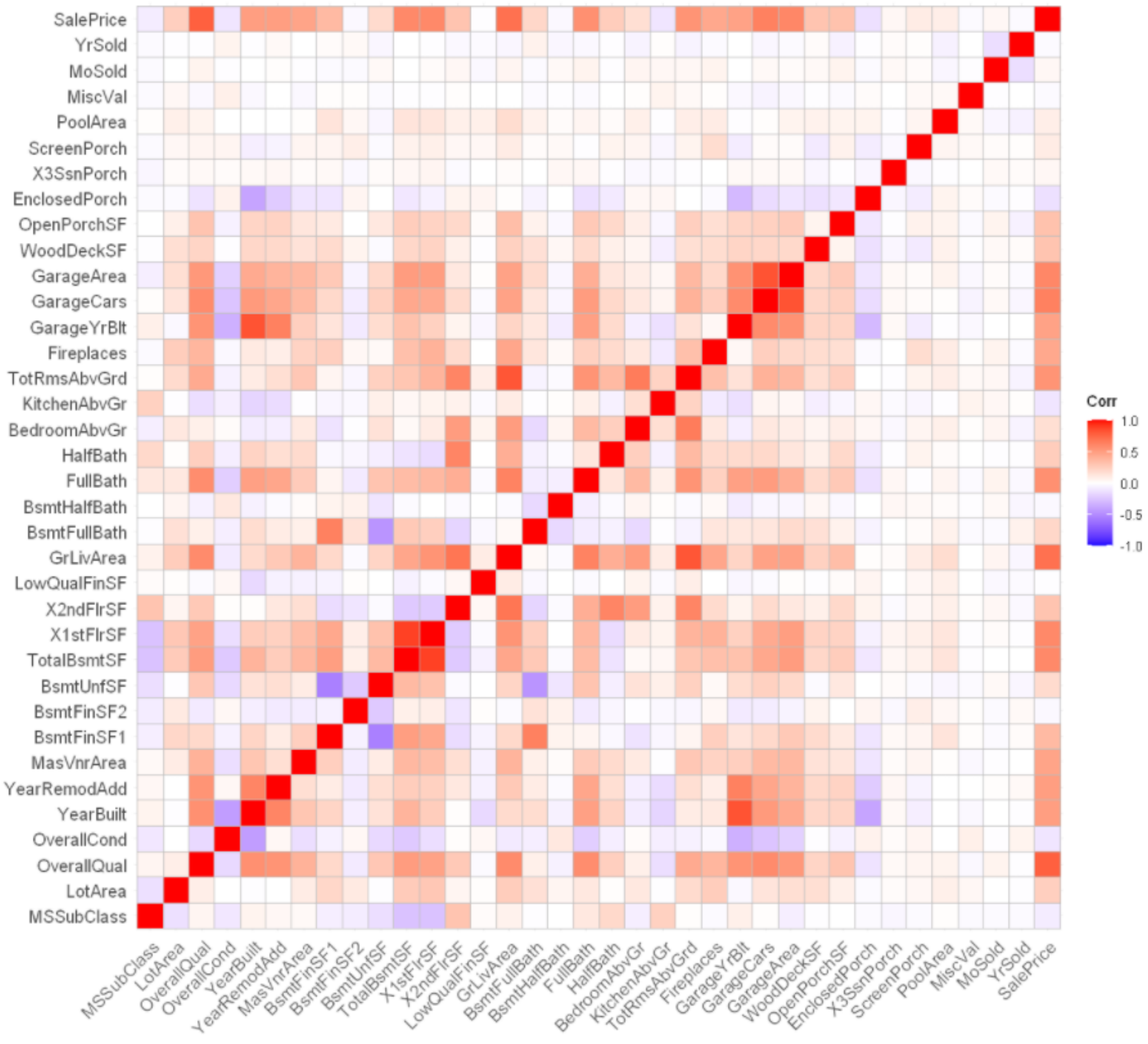


Figure **1** plots each quantitative variable against the response variable, Sale Price. From the plots of the 33 quantitative variables we can observe the following:

- The distribution of many variables pertaining to square footage, e.g. LotArea and General Living Area, are heavily skewed towards observations with lower square footage and have extreme outliers. This could possibly be addressed by through **log transformations**.
- Many quantitative variables have a disproportionate number of observations with a value of 0. For example, not every house have a Wooden Deck, so there are a disproportion number of WoodDeckSF observations with a value of 0. We think this could be addressed later by **adding in interaction terms** that checks to see if a house as a Wooden Deck.
- Not every variable appear relevant upon initial inspection. For example, the majority of PoolArea observations are 0s, and the non-zero observations do not follow a clear trend. The variable YrSold also do not seem to separate by SalePrice very well. Perhaps these variables have strong correlation with other variables, and will become meaningful after **adding in interaction terms**
- Some variables have a **linear relationship** with Sale Price. For example, we can see clearly that as OverQual increases, the Sale Price increases.

Figure 2: Correlation Plot of Quantitative variables



3.1.1 Correlation between quantitative variables

We also created a plot of the correlations between the quantitative variables in figure **2**, and we observed the following.

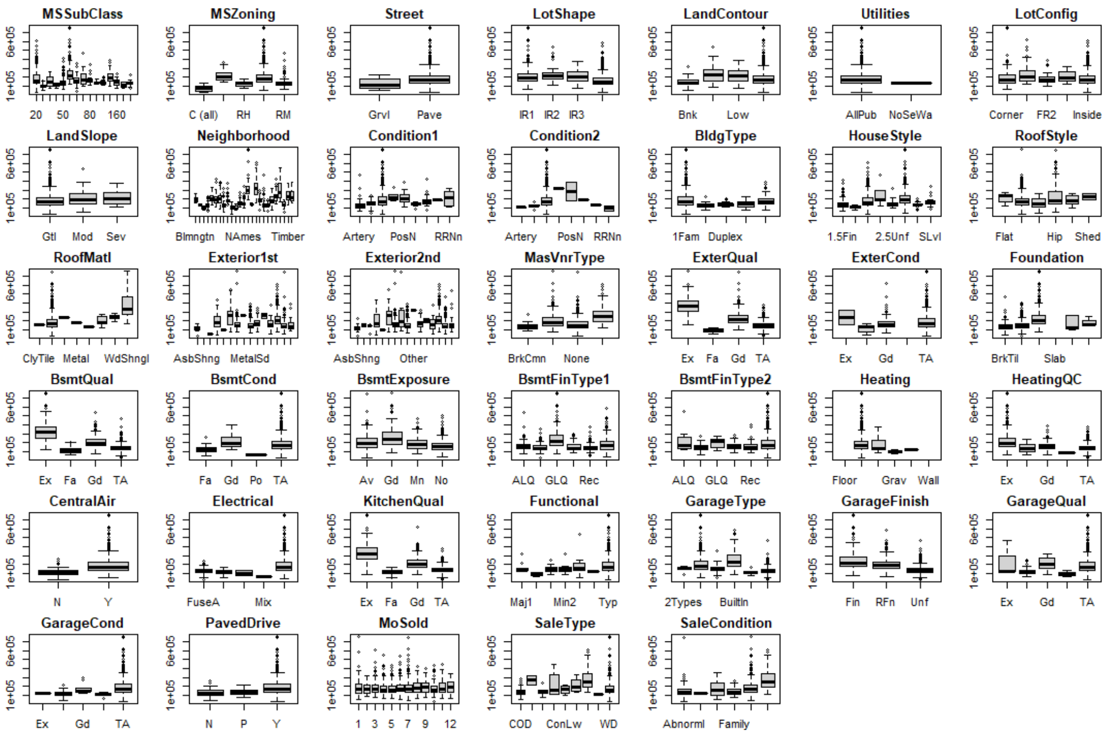
- It's clear there is a lot of collinearity among them. Some of the more prominent examples of such, including the correlations of X1stFlrSF:TotalBsmtSF, BsmtUnfSF:BsmtFinSF1, YrBuilt:GarageYrBlt, and GrLivArea:TotRmsAbvGrd, are very intuitive. For example, the area of the first floor and that of the basement are usually very similar in proportions.
- Conversely, the unfinished square footage of a basement is inversely proportional to the finished square footage. Relationships like these could be addressed through the addition of interaction terms moving forwards with the model.

3.2 Factor Variables

Figure **3** shows a boxplot of the distribution of each factor variables based on sale price. From the plots of the 40 factor variables we can observe the following:

- Some variables appear to have a distinct impact on separating Sale Price. For example, the mean and IQR of houses with CentralAir is significantly higher than houses without Central Air.
- Not every factor variable appear to be meaning by itself. However, perhaps by **adding interaction terms** between a factor variable and its corresponding quantitative variable, we may derive more meaning. For example, perhaps interacting basement condition with basement square footage will provide a better estimate of the basement value, and consequently improve our prediction.

Figure 3: Quantitative Variables Plotted Against Sale Price



4 Feature Engineering

In this section, we will discuss the ideas behind some of the new variables generated and the transformations performed. From there, we will also look at the effects of these changes.

4.1 Generating new variables

From anecdotal experiences, we generated some new features that may improve the model. Namely, we added to main types of new variables.

- **Ratio Variables**. We surmised that many buyers would care about the ratio between some of the variables. For example, the number of bathrooms per bedrooms, or the utilization rate of lot area (General Living Area / Lot Area).
- **Existence Checks**. In our explanatory data analysis, we noted that many quantitative variables have a large proportion of observations with a value of 0. Therefore, we will add in variables that check whether or not a house has a particular feature, such as wooden deck, to prevent the 0s from having a disproportionately large impact on the regression.

  Overall, we added 9 new ratio variables which are quantitative, and 10 new existence check variables which are factors(levels[TRUE, FALSE]). The total number of variables increased from 74 to 93. There are now 43 quantitative variables, and 50 factor variables.

4.2 Log Transformation

From our exploratory data analysis, we noted one important thing about the distribution of many quantitative variables: they are heavily skewed towards smaller values with extreme outliers. Figure **4** shows the variables before the log transformations. There are a total of 21 variables that exhibit this phenomenon. To address this, we performed log transformation on these 21 variables. Since many of these variables have observations with a value of 0, we did $log(df\$v + 1)$ to avoid having negative infinities.

Figure **5** shows the plots of the log transformed variables. We can see that the distribution of the explanatory values are more evenly distributed, and we have successfully removed extreme outliers. Observing the log transformed plots, we notice the following things.

- Many variables have a high concentration of 0s. If we perform regression on the entire explanatory variable domain, the 0s will have a dis proportionally large effect on the slope coefficient. To address this, we should perform regression with **interactions** between a categorical variable that verifies whether or not a feature exist and the explanatory variable of the feature. For example, if we perform an interaction between hasOpenPorchSF[TRUE, FALSE] * OpenPorchSF, we can prevent the 0s from impacting the regression.
- Some variables appear to have a **quadratic, not linear, relationship** with Sale Price. For example, the BsmtFindSF2 variable appears to have an U-shaped curve. To address this, we can include variables of higher order in our regression model.

4.3 Transformation Performance Comparison

4.3.1 Vanilla Model

```
Call:
lm(formula = SalePrice ~ ., data = vTrain)

Residuals:
    Min      1Q  Median      3Q     Max
-181550   -9578     159    9329  181550
```

Figure 4: Before Log Transformation

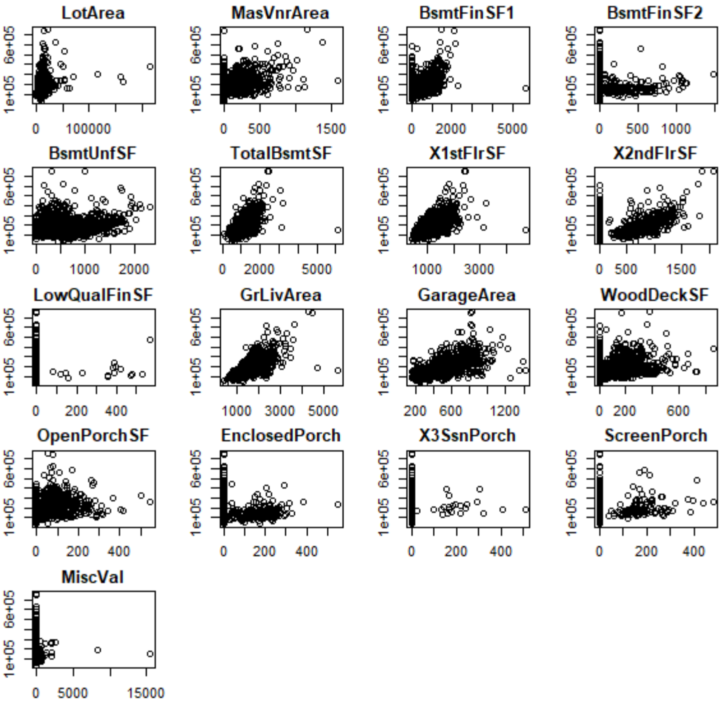
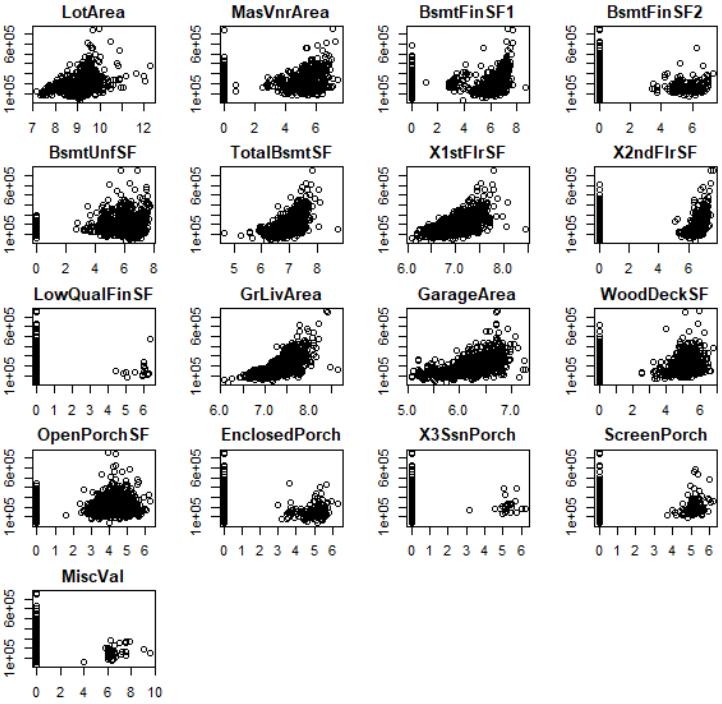


Figure 5: After Log Transformation



```
Residual standard error: 23200 on 1093 degrees of freedom
Multiple R-squared:  0.9293,    Adjusted R-squared:  0.9136
F-statistic: 58.92 on 244 and 1093 DF,  p-value: < 2.2e-16
```

We trained a linear regression model that includes every single explanatory variables in the vanilla training data with 73 covariates. We see that surprising, this model is actually already performing really well. It is explaining approximately 92.9% of the variance, and has an $adjR^2$ value of 0.9136.

4.3.2 Transformed Model

```
Call:
lm(formula = SalePrice ~ ., data = tTrain)

Residuals:
    Min      1Q  Median      3Q     Max
-169743  -10019     355    9954  169743

Residual standard error: 24600 on 1069 degrees of freedom
Multiple R-squared:  0.9218,    Adjusted R-squared:  0.9025
F-statistic: 47.89 on 263 and 1069 DF,  p-value: < 2.2e-16
```

4.3.3 Observations

Looking at the R^2 and $adjR^2$ values, we actually see that the transformed model is performing worse than the vanilla model, which is counter-intuitive. However, this can be explained by the fact that many of the additional variables that we have engineered rely on interactions with another variable. Additionally, by many of the log transformed variables demonstrated a quadratic relationship, which would require additional higher order terms such $I(variable^2)$.

5 Interactions Terms & Higher Order Variables

In this section, we will discuss our approach to adding interaction terms and higher order variables.

5.1 Interaction Terms

Previously, we discussed how many of the quantitative variables could be explained better by interacting them with corresponding categorical variables. For example, intuitively, we thought that tying basement exposure to total basement square footage would provide a better prediction. We also though that the value of a lot area could be highly correlated with the type of zoning, so adding in an interaction term between zoning and lot area could better explain the variances.

5.2 Higher Order Variables

Previously, we discussed how some variables, such as TotalBsmntSF appear to have a quadratic relationship with Sale Price. Obviously, a simple linear regression that is a straight line cannot properly capture a quadratic relationship. Therefore, we have decided to add higher order variables, such as $I(variable^2)$ for features that appear to be quadratic in our regression model.

5.3 Full Transformed Model with Interaction

```
Call:
lm(formula = SalePrice ~ . + BsmtQual * UnfinishedRatio + BsmtCond *
    TotalBsmtSF + hasX2ndFlrSF * X2ndFlrSF + hasLowQualFinSF *
    LowQualFinSF + hasWoodDeckSF * WoodDeckSF + hasOpenPorchSF *
    OpenPorchSF + hasEnclosedPorch * EnclosedPorch + hasX3SsnPorch *
    X3SsnPorch + hasScreenPorch * ScreenPorch + I(TotalBsmtSF^2) +
    I(MasVnrArea^2) + I(BsmtFinSF2^2) + I(GarageArea^2) + I(GrLivArea^2) +
    MasVnrType * I(MasVnrArea^2) + BsmtFinType1 * I(BsmtFinSF1^2) +
    BsmtFinType2 * I(BsmtFinSF2^2) + BsmtQual * I(TotalBsmtSF^2) +
    BsmtExposure * I(TotalBsmtSF^2) + GarageType * I(GarageArea^2) +
    PavedDrive * I(GarageArea^2) + GarageFinish * I(GarageArea^2) +
    MSZoning * LotArea + RoofMatl * I(GrLivArea^2) + RoofMatl *
    X2ndFlrSF + Functional * I(GrLivArea^2) + MSSubClass * MSZoning +
    LotShape * LotArea + LotShape * I(GrLivArea^2) + Street *
    LotArea + LandContour * LotArea + LotConfig * LotArea + LandSlope *
    LotArea + Neighborhood * LotArea + Neighborhood * I(GrLivArea^2) +
    Condition1 * LotArea + Condition1 * I(GrLivArea^2) + Condition2 *
    LotArea, data = tTrain)

Residuals:
    Min      1Q  Median      3Q     Max
-117323   -7049       0    7018  110498
Residual standard error: 17940 on 912 degrees of freedom
  (5 observations deleted due to missingness)
Multiple R-squared:  0.9645,    Adjusted R-squared:  0.9482
F-statistic: 59.03 on 420 and 912 DF,  p-value: < 2.2e-16
```

5.3.1 Observations:

Comparing the model with interaction to the transformed model in section 3.3.2, we see that the R^2 and $adjR^2$ value have both increased significantly. The amount of variance left unexplained (1 - R^2) have been reduced by nearly half! This demonstrates the power of interaction terms and higher order variable. The fit improved because the model explained more variance by accounting for correlations between variables, and captured the quadratic nature of many variables.

6 Model Selection

In this section, we will discuss how we selected our model, and compare the performance differences between the full model and the best selected models. We will also contrast the performance differences between the best model from vanilla training dataset with no interaction terms, and the best model from the transformed dataset with interaction terms added.

6.1 Selection Techniques

```
train <- read_parquet("data/processed/train.parquet")
fullModel_vanilla <- lm(SalePrice ~. , data = train)
finalModel_vanilla <- step(fullModel_vanilla)
summary(finalModel_vanilla)
```

We performed model selection through using the R's step() function. The step() function accepts a linear model as input, and iteratively remove the least significant predictor as long as the AIC score increases.

6.2 Best Selected Vanilla Model

```
Call:
lm(formula = SalePrice ~ MSZoning + LotArea + Street + LandContour +
      LotConfig + LandSlope + Neighborhood + Condition1 + Condition2 +
      BldgType + HouseStyle + OverallQual + OverallCond + YearBuilt +
      YearRemodAdd + RoofStyle + RoofMatl + Exterior1st + MasVnrType +
      MasVnrArea + ExterQual + Foundation + BsmtQual + BsmtExposure +
      BsmtFinSF1 + BsmtFinSF2 + BsmtUnfSF + X1stFlrSF + X2ndFlrSF +
      BedroomAbvGr + KitchenAbvGr + KitchenQual + TotRmsAbvGrd +
      Functional + Fireplaces + GarageCars + GarageArea + GarageQual +
      GarageCond + WoodDeckSF + ScreenPorch + PoolArea + SaleType,
   data = train)
Residual standard error: 22950 on 1187 degrees of freedom
Multiple R-squared:  0.9249,    Adjusted R-squared:  0.9154
F-statistic: 97.44 on 150 and 1187 DF,  p-value: < 2.2e-16
```

This model was selected by using the step() function, and its final AIC score was 27012.16.

```
Step:  AIC=27012.16
SalePrice ~ MSZoning + LotArea + Street + LandContour + LotConfig +
     LandSlope + Neighborhood + Condition1 + Condition2 + BldgType +
     HouseStyle + OverallQual + OverallCond + YearBuilt + YearRemodAdd +
     RoofStyle + RoofMatl + Exterior1st + MasVnrType + MasVnrArea +
     ExterQual + Foundation + BsmtQual + BsmtExposure + BsmtFinSF1 +
     BsmtFinSF2 + BsmtUnfSF + X1stFlrSF + X2ndFlrSF + BedroomAbvGr +
     KitchenAbvGr + KitchenQual + TotRmsAbvGrd + Functional +
     Fireplaces + GarageCars + GarageArea + GarageQual + GarageCond +
     WoodDeckSF + ScreenPorch + PoolArea + SaleType
```

We performed model selection on the vanilla training data with 73 explanatory variables and no interaction terms. Here, we can see the impact of model selection. Relative to the full model, the R^2 value decreased from 0.9293 to 0.9249, but the $adjR^2$ value actually increased from 0.9136 to 0.9154. This signifies that we have slightly reduced overfitting, since the $adjR^2$ value improved.

6.3 Best Selected Transformed Model with Interaction

```
Call:
lm(formula = SalePrice ~ MSSubClass + MSZoning + LotArea + Street +
      LotShape + LandContour + Utilities + LotConfig + LandSlope +
      Neighborhood + Condition1 + Condition2 + OverallQual + OverallCond +
      YearBuilt + YearRemodAdd + RoofMatl + Exterior1st + MasVnrType +
      MasVnrArea + ExterCond + BsmtQual + BsmtExposure + BsmtUnfSF +
      TotalBsmtSF + X2ndFlrSF + GrLivArea + FullBath + BedroomAbvGr +
      KitchenAbvGr + KitchenQual + Functional + Fireplaces + GarageFinish +
      GarageCars + GarageArea + GarageQual + GarageCond + PavedDrive +
      WoodDeckSF + OpenPorchSF + EnclosedPorch + X3SsnPorch + ScreenPorch +
      PoolArea + SaleCondition + UnfinishedRatio + BedPerBath +
      BathsPerLivAbv + hasOpenPorchSF + hasScreenPorch + I(TotalBsmtSF^2) +
      I(MasVnrArea^2) + I(GarageArea^2) + I(GrLivArea^2) + BsmtQual:UnfinishedRatio +
      MasVnrType:I(MasVnrArea^2) + BsmtQual:I(TotalBsmtSF^2) +
      BsmtExposure:I(TotalBsmtSF^2) + GarageFinish:I(GarageArea^2) +
      MSZoning:LotArea + RoofMatl:X2ndFlrSF + MSSubClass:MSZoning +
      LotArea:LotShape + LotArea:LotConfig + LotArea:LandSlope +
      Neighborhood:I(GrLivArea^2) + LotArea:Condition1 + LotArea:Condition2,
    data = tTrain)
Residual standard error: 17840 on 1083 degrees of freedom
(5 observations deleted due to missingness)
Multiple R-squared:  0.9583,    Adjusted R-squared:  0.9488
F-statistic: 100.1 on 249 and 1083 DF,  p-value: < 2.2e-16
```

This model was selected using the step() function, and its final AIC Score was 26320.62.

```
Step:  AIC=26320.63
SalePrice ~ MSSubClass + MSZoning + LotArea + Street + LotShape +
     LandContour + Utilities + LotConfig + LandSlope + Neighborhood +
     Condition1 + Condition2 + OverallQual + OverallCond + YearBuilt +
     YearRemodAdd + RoofMatl + Exterior1st + MasVnrType + MasVnrArea +
     ExterCond + BsmtQual + BsmtExposure + BsmtUnfSF + TotalBsmtSF +
     X2ndFlrSF + GrLivArea + FullBath + BedroomAbvGr + KitchenAbvGr +
     KitchenQual + Functional + Fireplaces + GarageFinish + GarageCars +
     GarageArea + GarageQual + GarageCond + PavedDrive + WoodDeckSF +
     OpenPorchSF + EnclosedPorch + X3SsnPorch + ScreenPorch +
     PoolArea + SaleCondition + UnfinishedRatio + BedPerBath +
     BathsPerLivAbv + hasOpenPorchSF + hasScreenPorch + I(TotalBsmtSF^2) +
     I(MasVnrArea^2) + I(GarageArea^2) + I(GrLivArea^2) + BsmtQual:UnfinishedRatio +
     MasVnrType:I(MasVnrArea^2) + BsmtQual:I(TotalBsmtSF^2) +
     BsmtExposure:I(TotalBsmtSF^2) + GarageFinish:I(GarageArea^2) +
     MSZoning:LotArea + RoofMatl:X2ndFlrSF + MSSubClass:MSZoning +
     LotArea:LotShape + LotArea:LotConfig + LotArea:LandSlope +
     Neighborhood:I(GrLivArea^2) + LotArea:Condition1 + LotArea:Condition2
```

In this model, we performed model selection on the transformed data with 92 explanatory variables and included interaction terms. We can see that relatively, although the R^2 value decreased, the $adjR^2$ value actually increased. Again, this makes sense because by selecting a subset of the variable, we are minimizing overfitting.

Furthermore, relative to the AIC score of best model selected from the not transformed vanilla data, we see that the AIC score of this model is lower. A lower AIC score signifies a better fit. This proves that model selection, adding interaction terms, and including higher order variables can improve fit.

7 Cross Validation

One of the biggest limitation of this dataset was the small number of observations. Therefore, in validating our model, we want to include as many observations in the training set as possible to reduce bias. Consequently, we decided to use Leave-One-Out-Cross-Validation (LOOCV).

7.1 Method

```
kFoldCV <- function(data, k, transformed) {
  # partition data
  folds <- createFolds(1:nrow(data), k = k, list = FALSE, returnTrain = FALSE)
  partedData <- list()
  for(i in 1:k ) {
    partedData <- append(partedData, list(data[folds==i,]))
  }

  # Train and get MSE
  avgMSE <- 0
  for (i in 1:k){
    train <- bind_rows(partedData[-i])
    holdo <- partedData[[i]]
    model <- NULL
    if (transformed) {
      model <- bestModelTransformed(train)
    } else {
      model <- bestModelVanilla(train)
    }
    avgMSE <- avgMSE + getMSE(model, holdo)
    break
  }
  return(avgMSE / k)
}
```

We designed a generic kFoldCV function that accepts a data, the number of folds, and wheth

7.2 LOOCV Model Performances

```
set.seed(888)
vTrain <- read_parquet("data/processed/train.parquet")
vTrainMSE <- kFoldCV(vTrain, nrow(vTrain), FALSE)

set.seed(888)
tTrain <- read_parquet("data/processed/transformed_train.parquet")
tTrainMSE <- kFoldCV(tTrain, nrow(tTrain), TRUE)}
```

Performing LOOCV on the best model selected from the vanilla dataset yielded a MSE of 15080.05. Performing LOOCV on the best model selected from the transformed dataset with interactions yielded a MSE of 336.818.

From the MSE, we can see that the best model is performing surprisingly good! It is has a very small MSE of 336.818, which means that it is almost a perfect fit.

However, we do have to be wary about this result, as the the model could be severely overfitting due to small training and testing data size. However, we cannot know for sure.

8 Limitations

The biggest problem with our prediction is that our training sample size is rather small. Therefore, the model we trained is susceptible to high bias. Furthermore, in the full model with interaction that we trained, we actually ran into the issue of rank-deficiency, which essentially meant that the prediction may not be reliable, since there was not enough observations given the large number of coefficients.

Bootstrapping and re-sampling more data to artificially increase the sample size is beyond the scope of this course. This was a limitation imposed by the small training data, and there wasn't much we could've done.

9 Conclusion

In accordance with our hypothesis stated in the introduction section, we found that the three explanatory variables do have a significance in determining SalePrice.

- LotArea: Lot size in square feet
- YearBuilt: Original construction date
- OverallQual: Overall material and finish quality

Or essentially;

$$\begin{aligned}\beta_{\text{LotArea}} &\neq 0\\ \beta_{\text{YearBuilt}} &\neq 0\\ \beta_{\text{OverallQual}} &\neq 0\end{aligned}$$

Though, we found that the most influential variables in determining SalePrice were;

- RoofMatl Type of Roof
- OverallQual: Overall material and finish quality
- OverallCond: Overall condition rating
- GrLivArea: Above grade (ground) living area square feed
- Unfinished ratio: BsmtUnfSF / TotalBsmtSF
  - BsmtUnfSF: Unfinished square feet of basement area
  - TotalBsmtSF: Total square feet of basement area

The p-values for each of the variables are as follows;

Variable	p-value
RoofMatl	< 2e-16
OverallQual	6.02e-16
OverallCond	6.05e-16
GrLivArea	< 2e-16
Unfinished ratio	< 2e-16

Looking at the p-values we can determine that each of these explanatory variables are significant at any acceptable alpha level (ex. $\alpha = 0.05, 0.01$). Therefore we reject the null hypothesis that the explanatory variables have no influence on the model in favour of the alternative.

10 Learning Outcomes

We applied the method we learned in this course for model selection in the multiple regression context.

We compared R^2, $adjR^2$ and AIC of each model and selected the best one for prediction. Our regression model with feature engineering proved to be the better model, since they have the better AIC and $adjR^2$ values. Therefore, demonstrating the power of model selection.

Furthermore, we transformed some variables and added in some interactions as well to find more appropriate linear model. We demonstrated that the model with interaction terms and higher order variables can explain more variance.

We also confirmed our hypothesis and concluded that LotArea, yearBuilt, and OverallQual were significant variables by applying the concept of p-values.

Lastly, we demonstrated the importance of feature engineering. The engineered value, unfinished ratio, which is the ratio of unfurnished basement to total basement square footage was one of the most important variables. We were able to create a much better prediction model with significantly lower cross-validated MSE partly due to feature engineering.

11 Member Participation

Our group worked really well together, and everyone contributed equally to the project. We peer-programmed most of the R scripts together over Zoom, and compiled the project together.

12 Appendices

12.1 A: Variables Included in Data set

1. SalePrice: the property's sale price in USD. This is the target variable that you're trying to predict.
2. MSSubClass: The building class
3. MSZoning: The general zoning classification
4. LotFrontage: Linear feet of street connected to property
5. LotArea: Lot size in square feet
6. Street: Type of road access
7. Alley: Type of alley access
8. LotShape: General shape of property
9. LandContour: Flatness of the property
10. Utilities: Type of utilities available
11. LotConfig: Lot configuration
12. LandSlope: Slope of property
13. Neighborhood: Physical locations within Ames city limits
14. Condition1: Proximity to main road or railroad
15. Condition2: Proximity to main road or railroad (if a second is present)
16. BldgType: Type of dwelling
17. HouseStyle: Style of dwelling
18. OverallQual: Overall material and finish quality
19. OverallCond: Overall condition rating
20. YearBuilt: Original construction date
21. YearRemodAdd: Remodel date
22. RoofStyle: Type of roof
23. RoofMatl: Roof material
24. Exterior1st: Exterior covering on house
25. Exterior2nd: Exterior covering on house (if more than one material)
26. MasVnrType: Masonry veneer type
27. MasVnrArea: Masonry veneer area in square feet
28. ExterQual: Exterior material quality
29. ExterCond: Present condition of the material on the exterior
30. Foundation: Type of foundation
31. BsmtQual: Height of the basement
32. BsmtCond: General condition of the basement

33. BsmtExposure: Walkout or garden level basement walls
34. BsmtFinType1: Quality of basement finished area
35. BsmtFinSF1: Type 1 finished square feet
36. BsmtFinType2: Quality of second finished area (if present)
37. BsmtFinSF2: Type 2 finished square feet
38. BsmtUnfSF: Unfinished square feet of basement area
39. TotalBsmtSF: Total square feet of basement area
40. Heating: Type of heating
41. HeatingQC: Heating quality and condition
42. CentralAir: Central air conditioning
43. Electrical: Electrical system
44. 1stFlrSF: First Floor square feet
45. 2ndFlrSF: Second floor square feet
46. LowQualFinSF: Low quality finished square feet (all floors)
47. GrLivArea: Above grade (ground) living area square feet
48. BsmtFullBath: Basement full bathrooms
49. BsmtHalfBath: Basement half bathrooms
50. FullBath: Full bathrooms above grade
51. HalfBath: Half baths above grade
52. Bedroom: Number of bedrooms above basement level
53. Kitchen: Number of kitchens
54. KitchenQual: Kitchen quality
55. TotRmsAbvGrd: Total rooms above grade (does not include bathrooms)
56. Functional: Home functionality rating
57. Fireplaces: Number of fireplaces
58. FireplaceQu: Fireplace quality
59. GarageType: Garage location
60. GarageYrBlt: Year garage was built
61. GarageFinish: Interior finish of the garage
62. GarageCars: Size of garage in car capacity
63. GarageArea: Size of garage in square feet
64. GarageQual: Garage quality
65. GarageCond: Garage condition
66. PavedDrive: Paved driveway

67. WoodDeckSF: Wood deck area in square feet
68. OpenPorchSF: Open porch area in square feet
69. EnclosedPorch: Enclosed porch area in square feet
70. 3SsnPorch: Three season porch area in square feet
71. ScreenPorch: Screen porch area in square feet
72. PoolArea: Pool area in square feet
73. PoolQC: Pool quality
74. Fence: Fence quality
75. MiscFeature: Miscellaneous feature not covered in other categories
76. MiscVal: Dollar Value of miscellaneous feature
77. MoSold: Month Sold
78. YrSold: Year Sold
79. SaleType: Type of sale
80. SaleCondition: Condition of sale

12.2 R Codes

Below are the contents of the r scripts we used. We have also included a zip file of the entire project along with our submission.

12.2.1 B: cleaning.R

```
library(dplyr)
library(arrow)

# ------------------------ FUNCTIONS ------------------------
#' @description
#' convert dataset to parquet after cleaning, and save
#'
#' @return list of train and test set, and names of columns dropped
#' with data as factor with NA entries removed:
initDataset <- function(){
  # Load datasets
  train <- read.csv(”./data/raw/train.csv”)
  nTrain <- nrow(train)

  # remove ID from train
  train <- train[,-1]

  # Set character entries as factor
  train <- train %>% mutate_if(is.character, as.factor)

  # Manually set some integer based entries as factors
  # MSSubClass is a code
  train$MSSubClass <- as.factor(train$MSSubClass)
  # Month sold is not cyclic, so set as factor
  train$MoSold <- as.factor(train$MoSold)

  # get NA Columns
  threshold <- ceiling(nTrain/10)
  NACols <- getNACols(train, threshold)

  # select only columns with NA < threshold, and rows without NA
  train <- na.omit(select(train, -NACols))

  # save as parquet
  write_parquet(train, sink = "data/processed/train.parquet")
}

#' @description
#' Return column names of columns that contain NA > threshold
#' @param train the training set
#' @param threshold the threshold for NAs
#'
#' @return data as factor with NA entries removed:
getNACols <- function(train, threshold) {
  NASummary <- sapply(train, function(x) sum(is.na(x)))
  NACols <- c()
  for (i in 1:length(NASummary)){
    naCount <- NASummary[[i]]
    if (naCount > threshold) {
      NACols <- c(NACols, names(NASummary[i]))
    }
  }
  return(NACols)
}

#' @description
#' Print Out the variables names and class
#' @param data the dataset
#'
#' @return Printed statements:
printFeatures <- function(data) {
  types <- sapply(data, class)
  n <- length(types)
  varn <- names(types)

  for (i in 1:n){
    if (types[[i]] == "integer"){
      cat(varn[i], ":", types[[i]], "\n")
    } else {
      cat(varn[i], ":", types[[i]], levels(data[,i]),"\n")
    }
  }
}
```

```
# -------------------- SCRIPTS ----------------------

initDataset()
```

12.2.2 C: feature_engineering.R

```
library(arrow)
library(dplyr)

# -------------------------- FUNCTIONS ------------------------------
addRatios <- function(df) {
  newDf <- df
  # New features
  # UnfinishedRatio: BsmtUnfSF / TotalBsmtSF
  # GaragePerCar : GarageCars / GarageArea
  # X2ndFlrRatio : X2ndFlrSF / GrLivArea
  # GrLivAreaRatio : GrLivArea / LotArea
  newDf$UnfinishedRatio <- newDf$BsmtUnfSF / newDf$TotalBsmtSF
  newDf$GaragePerCar <- newDf$GarageArea / newDf$GarageCars
  newDf$X2ndFlrRatio <- newDf$X2ndFlrSF / newDf$GrLivArea
  newDf$GrLivAreaRatio <- newDf$GrLivArea / newDf$LotArea

  # bedrooms per bath
  newDf$BedPerBath <- newDf$BedroomAbvGr / newDf$FullBath
  newDf$BedPerBath[newDf$BedPerBath == Inf] <- 0

  newDf$BedPerBathAbv <- newDf$BedroomAbvGr / (newDf$FullBath + newDf$HalfBath)
  newDf$BedPerBathAbv[newDf$BedPerBathAbv == Inf] <- 0

  # Total Baths and Beds Per Bath
  newDf$totalBath <- newDf$FullBath + newDf$HalfBath + newDf$BsmtHalfBath + newDf$BsmtFullB

  newDf$BedPerTotalBath <- newDf$BedroomAbvGr / newDf$totalBath
  newDf$BedPerTotalBath[newDf$BedPerTotalBath == Inf] <- 0

  # Sqft Per Bath
  newDf$BathsPerLivAbv <- newDf$GrLivArea / newDf$BedPerTotalBath
  newDf$BathsPerLivAbv[newDf$BathsPerLivAbv == Inf] <- 0

  return(newDf)
}

# Add logVal
addlogVal <- function(df, varn) {
  newDf <- df
  for (name in varn) {
    newDf[name] <- log(newDf[name] + 1)
  }
  return(newDf)
}

# Add Existence Binary of factor True False
```

```
addHasFeat <- function(df, varn) {
  newDf <- df
  for (name in varn) {
    newName <- paste("has", name, sep="")
    newDf[newName] <- as.factor(newDf[name] > 0)
  }
  return(newDf)
}

createTransformedTrain <- function() {
  train <- read_parquet("data/processed/train.parquet")

  # Add new ratio covariates
  train <- addRatios(train)

  # Log SF and MiscVal covariates
  toLog <- c("LotArea", "MasVnrArea", "BsmtFinSF1", "BsmtFinSF2", "BsmtUnfSF",
             "TotalBsmtSF", "X1stFlrSF", "X2ndFlrSF", "LowQualFinSF",
             "GrLivArea", "GarageArea", "WoodDeckSF", "OpenPorchSF",
             "EnclosedPorch", "X3SsnPorch", "ScreenPorch", "MiscVal")
  train <- addlogVal(train, toLog)

  # Add Has Feature covariates
  toExist <- c("X2ndFlrSF", "LowQualFinSF", "WoodDeckSF", "OpenPorchSF",
               "EnclosedPorch", "X3SsnPorch", "ScreenPorch", "MiscVal",
               "PoolArea", "BedroomAbvGr")
  train <- addHasFeat(train, toExist)

  write_parquet(train, sink = "data/processed/transformed_train.parquet")

  return(train)
}

# -------------------------- SCRIPTS ------------------------------

createTransformedTrain()
```

12.2.3 D: model_selection.R

```
library(arrow)
library(dplyr)

# -------------------------- BEST VANILLA ------------------------------

train <- read_parquet("data/processed/train.parquet")

fullModel_vanilla <- lm(SalePrice ~. , data = train)
finalModel_vanilla <- step(fullModel_vanilla)
summary(finalModel_vanilla)
getMSE(finalModel_vanilla, train)

# BEST VANILA
# lm(formula = SalePrice ~ MSZoning + LotArea + Street + LandContour +
#       LotConfig + LandSlope + Neighborhood + Condition1 + Condition2 +
#       BldgType + HouseStyle + OverallQual + OverallCond + YearBuilt +
#       YearRemodAdd + RoofStyle + RoofMatl + Exterior1st + MasVnrType +
#       MasVnrArea + ExterQual + Foundation + BsmtQual + BsmtExposure +
#       BsmtFinSF1 + BsmtFinSF2 + BsmtUnfSF + X1stFlrSF + X2ndFlrSF +
#       BedroomAbvGr + KitchenAbvGr + KitchenQual + TotRmsAbvGrd +
#       Functional + Fireplaces + GarageCars + GarageArea + GarageQual +
#       GarageCond + WoodDeckSF + ScreenPorch + PoolArea + SaleType,
#     data = train)
# Residual standard error: 22950 on 1187 degrees of freedom
# Multiple R-squared:  0.9249,  Adjusted R-squared:  0.9154
# F-statistic: 97.44 on 150 and 1187 DF,  p-value: < 2.2e-16

# ------------------------- BEST TRANSFORMED ------------------------------

tTrain <- read_parquet("data/processed/transformed_train.parquet")
fullModel_transformed <- lm(SalePrice ~. +
                                BsmtQual * UnfinishedRatio +
                                BsmtCond * TotalBsmtSF +
                                hasX2ndFlrSF * X2ndFlrSF +
                                hasLowQualFinSF * LowQualFinSF +
                                hasWoodDeckSF * WoodDeckSF +
                                hasOpenPorchSF * OpenPorchSF +
                                hasEnclosedPorch * EnclosedPorch +
                                hasX3SsnPorch * X3SsnPorch +
                                hasScreenPorch * ScreenPorch +
                                I(TotalBsmtSF^2) +
                                I(MasVnrArea^2) +
                                I(BsmtFinSF2^2) +
                                I(GarageArea^2) +
                                I(GrLivArea^2) +
                                MasVnrType * I(MasVnrArea^2) +
                                BsmtFinType1 * I(BsmtFinSF1^2) +
                                BsmtFinType2 * I(BsmtFinSF2^2) +
                                BsmtQual * I(TotalBsmtSF^2) +
                                BsmtExposure * I(TotalBsmtSF^2) +
                                GarageType * I(GarageArea^2) +
                                PavedDrive * I(GarageArea^2) +
                                GarageFinish * I(GarageArea^2) +
                                MSZoning * LotArea +
                                RoofMatl * I(GrLivArea^2) +
                                RoofMatl * X2ndFlrSF +
                                Functional * I(GrLivArea^2) +
                                MSSubClass * MSZoning +
                                LotShape * LotArea +
                                LotShape * I(GrLivArea^2) +
                                Street * LotArea +
                                LandContour * LotArea +
                                LotConfig * LotArea +
                                LandSlope * LotArea +
                               Neighborhood * LotArea +
                               Neighborhood * I(GrLivArea^2) +
                               Condition1 * LotArea +
                               Condition1 * I(GrLivArea^2) +
                               Condition2 * LotArea, data = tTrain)
summary(fullModel_transformed)

finalModel_transformed <- step(fullModel_transformed)
summary(finalModel_transformed)
getMSE(finalModel_transformed, tTrain)

# Best Model:
# lm(formula = SalePrice ~ MSSubClass + MSZoning + LotArea + Street +
#       LotShape + LandContour + Utilities + LotConfig + LandSlope +
#       Neighborhood + Condition1 + Condition2 + OverallQual + OverallCond +
#       YearBuilt + YearRemodAdd + RoofMatl + Exterior1st + MasVnrType +
#       MasVnrArea + ExterCond + BsmtQual + BsmtExposure + BsmtUnfSF +
#       TotalBsmtSF + X2ndFlrSF + GrLivArea + FullBath + BedroomAbvGr +
#       KitchenAbvGr + KitchenQual + Functional + Fireplaces + GarageFinish +
#       GarageCars + GarageArea + GarageQual + GarageCond + PavedDrive +
#       WoodDeckSF + OpenPorchSF + EnclosedPorch + X3SsnPorch + ScreenPorch +
#       PoolArea + SaleCondition + UnfinishedRatio + BedPerBath +
#       BathsPerLivAbv + hasOpenPorchSF + hasScreenPorch + I(TotalBsmtSF^2) +
#       I(MasVnrArea^2) + I(GarageArea^2) + I(GrLivArea^2) + BsmtQual:UnfinishedRatio +
#       MasVnrType:I(MasVnrArea^2) + BsmtQual:I(TotalBsmtSF^2) +
#       BsmtExposure:I(TotalBsmtSF^2) + GarageFinish:I(GarageArea^2) +
#       MSZoning:LotArea + RoofMatl:X2ndFlrSF + MSSubClass:MSZoning +
#       LotArea:LotShape + LotArea:LotConfig + LotArea:LandSlope +
#       Neighborhood:I(GrLivArea^2) + LotArea:Condition1 + LotArea:Condition2,
#     data = tTrain)
# Residual standard error: 17840 on 1083 degrees of freedom
# (5 observations deleted due to missingness)
# Multiple R-squared:  0.9583,  Adjusted R-squared:  0.9488
# F-statistic: 100.1 on 249 and 1083 DF,  p-value: < 2.2e-16
```

12.2.4 E: training.R

```
library(arrow)
library(dplyr)
library(leaps)
library(caret)

# ------------------------- FUNCTIONS -----------------------------

getMSE <- function(model, data) {
  actual <- data$SalePrice
  pred <- predict(model, newdata = data)
  return (mean((actual - pred)^2))
}

bestModelVanilla <- function(data) {
  model <- lm(formula = SalePrice ~ MSZoning + LotArea + Street + LandContour +
                LotConfig + LandSlope + Neighborhood + Condition1 + Condition2 +
                BldgType + HouseStyle + OverallQual + OverallCond + YearBuilt +
                YearRemodAdd + RoofStyle + RoofMatl + Exterior1st + MasVnrType +
                MasVnrArea + ExterQual + Foundation + BsmtQual + BsmtExposure +
                BsmtFinSF1 + BsmtFinSF2 + BsmtUnfSF + X1stFlrSF + X2ndFlrSF +
                BedroomAbvGr + KitchenAbvGr + KitchenQual + TotRmsAbvGrd +
                Functional + Fireplaces + GarageCars + GarageArea + GarageQual +
                GarageCond + WoodDeckSF + ScreenPorch + PoolArea + SaleType,
              data = data)
  return(model)
}

bestModelTransformed <- function(data) {
  model <- lm(formula = SalePrice ~ MSSubClass + MSZoning + LotArea + Street +
                LotShape + LandContour + Utilities + LotConfig + LandSlope +
                Neighborhood + Condition1 + Condition2 + OverallQual +
                OverallCond + YearBuilt + YearRemodAdd + RoofMatl +
                Exterior1st + MasVnrType + MasVnrArea + ExterCond + BsmtQual +
                BsmtExposure + BsmtUnfSF + TotalBsmtSF + X2ndFlrSF + GrLivArea +
                FullBath + BedroomAbvGr + KitchenAbvGr + KitchenQual +
                Functional + Fireplaces + GarageFinish + GarageCars +
                GarageArea + GarageQual + GarageCond + PavedDrive + WoodDeckSF +
                OpenPorchSF + EnclosedPorch + X3SsnPorch + ScreenPorch +
                PoolArea + SaleCondition + UnfinishedRatio + BedPerBath +
                BathsPerLivAbv + hasOpenPorchSF + hasScreenPorch +
                I(TotalBsmtSF^2) + I(MasVnrArea^2) + I(GarageArea^2) +
                I(GrLivArea^2) + BsmtQual:UnfinishedRatio +
                MasVnrType:I(MasVnrArea^2) + BsmtQual:I(TotalBsmtSF^2) +
                BsmtExposure:I(TotalBsmtSF^2) + GarageFinish:I(GarageArea^2) +
                MSZoning:LotArea + RoofMatl:X2ndFlrSF + MSSubClass:MSZoning +
                LotArea:LotShape + LotArea:LotConfig + LotArea:LandSlope +
                Neighborhood:I(GrLivArea^2) + LotArea:Condition1 +
                LotArea:Condition2, data = data)
  return(model)
}

kFoldCV <- function(data, k, transformed) {
  # partition data
  folds <- createFolds(1:nrow(data), k = k, list = FALSE, returnTrain = FALSE)
  partedData <- list()
  for(i in 1:k ) {
    partedData <- append(partedData, list(data[folds==i,]))
  }

  # Train and get MSE
  avgMSE <- 0
  for (i in 1:k){
    train <- bind_rows(partedData[-i])
    holdo <- partedData[[i]]
    model <- NULL
    if (transformed) {
      model <- bestModelTransformed(train)
    } else {
      model <- bestModelVanilla(train)
    }
    avgMSE <- avgMSE + getMSE(model, holdo)
    break
  }
  return(avgMSE / k)
}

# ------------------------------- TRAIN SCRIPT ------------------------
set.seed(888)
vTrain <- read_parquet(”data/processed/train.parquet”)
vTrainMSE <- kFoldCV(vTrain, nrow(vTrain), FALSE)

set.seed(888)
tTrain <- read_parquet(”data/processed/transformed_train.parquet”)
tTrainMSE <- kFoldCV(tTrain, nrow(tTrain), TRUE)

# Full model
vModel <- lm(SalePrice ˜ ., data = vTrain)
summary(vModel)

# Transformed Model
tModel <- lm(SalePrice ˜ ., data = tTrain)
summary(tModel)
```

12.2.5 F: visualization.R

```
library(arrow)
library(dplyr)

# -------------------------- Original ------------------------
vTrain <- read_parquet(”data/processed/train.parquet”)
nfeature <- dim(vTrain)[[2]]

varn <- names(vTrain)
intCols <- match(names(Filter(is.numeric,vTrain)), varn)
facCols <- match(names(Filter(is.factor,vTrain)), varn)

# Plot  integer features
par(mfrow=c(6,6))
for(ind in intCols) {
  par(mar = c(2,2,2,2))
  plot(vTrain[,ind],vTrain$SalePrice,
       ylab =”Sale Price”,
       main =varn[ind])
}

# Plot factor features
par(mfrow=c(6,7))
for(ind in facCols) {
  par(mar = c(2,2,2,2))
  plot(vTrain[,ind],vTrain$SalePrice,
       ylab =”Sale Price”,
       main =varn[ind])
}

# ------------------------ Engineered -----------------------

tTrain <- read_parquet(”data/processed/transformed_train.parquet”)
nfeature <- dim(tTrain)[[2]]

varn <- names(tTrain)
intCols <- match(names(Filter(is.numeric,tTrain)), varn)
facCols <- match(names(Filter(is.factor,tTrain)), varn)

# Plot  integer features
par(mfrow=c(7,7))
for(ind in intCols) {
  par(mar = c(2,2,2,2))
  plot(tTrain[,ind],tTrain$SalePrice,
       ylab =”Sale Price”,
       main =varn[ind])
}

# Plot factor features
par(mfrow=c(7,8))
for(ind in facCols) {
  par(mar = c(2,2,2,2))
  plot(tTrain[,ind],tTrain$SalePrice,
       ylab =”Sale Price”,
       main =varn[ind])
}

# ---------------- Visualize Log Transformations ---------------------
toLog <- c(”LotArea”, ”MasVnrArea”, ”BsmtFinSF1”, ”BsmtFinSF2”, ”BsmtUnfSF”,
           ”TotalBsmtSF”, ”X1stFlrSF”, ”X2ndFlrSF”, ”LowQualFinSF”,
           ”GrLivArea”, ”GarageArea”, ”WoodDeckSF”, ”OpenPorchSF”,
           ”EnclosedPorch”, ”X3SsnPorch”, ”ScreenPorch”, ”MiscVal”)
# Before
par(mfrow=c(5,4))
for(name in toLog) {
  par(mar = c(2,2,2,2))
  plot(vTrain[[name]],vTrain$SalePrice,
       ylab =”Sale Price”,
       main =name)
}

# after
par(mfrow=c(5,4))
for(name in toLog) {
  par(mar = c(2,2,2,2))
  plot(tTrain[[name]],tTrain$SalePrice,
       ylab ="Sale Price",
       main =name)
}
```

13 Bibliography

References

[1] Dean De Cock: House Prices - Advanced Regression Techniques,
`https://www.kaggle.com/shivam2503/diamonds`

[2] De Cock, D. *Journal of Statistics Education* Volume 19, Number 3 (2011)
`http://jse.amstat.org/v19n3/decock.pdf`