Kaggle: House Prices with Advanced Regression Techniques

Trevor Isaacson, Hawas Alsadeed, Evan Kessler

2/28/2022

Introduction

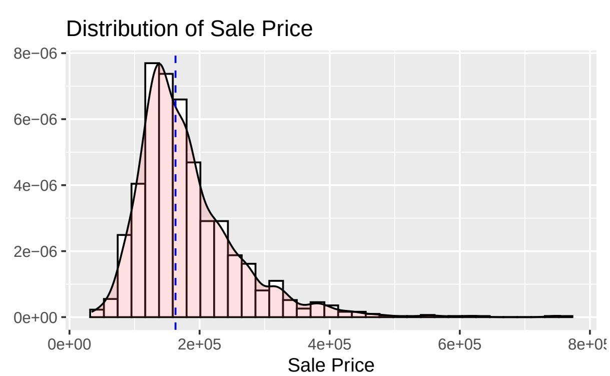
Imagine you're looking to purchase a home in the near future. How do you know the price of that home is accurate and fair. What factors or variables are you looking at to gauge the final price? For many, home prices are a number associated with a few key variables. For example, many buyers are looking at the number of bedrooms, bathrooms and the color of the fence. What other things affect the price of a home? Location, building materials, condition and quality are just a few that could potentially impact the price of a house. In this Kaggle competition, we look at home prices in Ames, Iowa and 79 variables associated with those homes. These 79 variables describe just about everything regarding these homes.

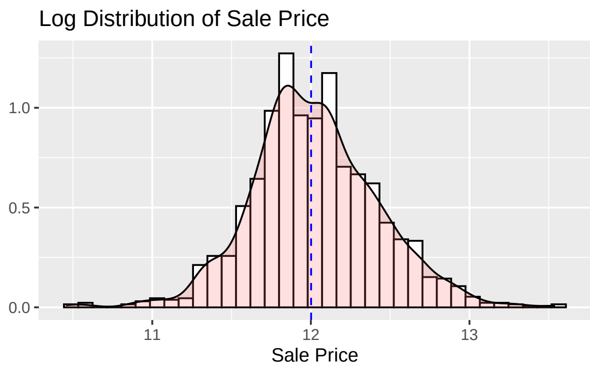
Problem Discription

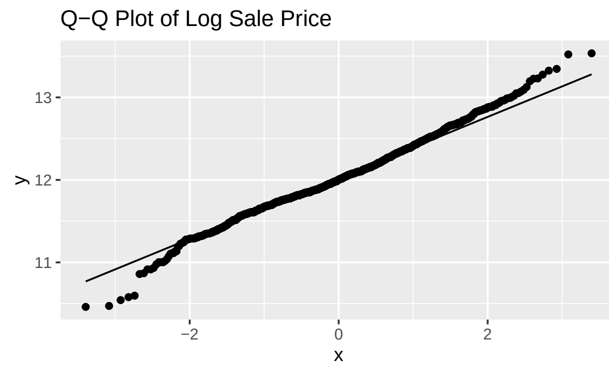
The problem is quite clear, can we accurately predict the price of a home? Given a lengthy set of variables regarding a house, are we able to predict the final sale price of that home? Using several different regression techniques, which method is able to best predict the final sale price and is it accurate enough to potentially use?

Data Cleaning

Because the original sales price data is very heavily skewed, we needed to log transform the prices. As shown in the histogram, we have a very heavy right tail because there are a some listings with very high prices compared to the median price of $180,921 (blue line). This non-normal shape and distribution is clearly evident in the Q-Q plot. By applying a log transformation, we fix this problem.







Additionally, after looking at several different variables we noticed there were some containing upwards of 1400 NA or 0 values in a dataset of 1460 observations. For some of them it made sense; such as fireplaces or half baths, a 0 value in this place just means the house simply doesn't have it in which many houses may not. In others however we noticed an issue could arise in fitting our models. We got rid of some of the values that were character values that had over 1000 of 1460 observations being NA. We figured this many values with NAs proved no relevance to our models, and only clogged up our GAMs and trees with unnecessary fittings. The other changes we made to our dataset was getting rid of NAs and categorizing a value of 'other' for variables with less than 5 values including Neighborhood, Exterior1st and Exterior2nd. Otherwise, we split our training data into a training and testing set 70% to 30% in order to check our models before submitting the final testing data to Kaggle.

Multiple Linear Regression

To begin, we started with a simple multiple linear regression model. We wanted to give ourselves a baseline root mean squared error value and because linear regression is the easiest to apply and interpret, we determined this is the best place to start. The model is fit using 73 variables and the training set. It reported a RMSE of 0.14 and some statistically significant variables include OverallCond, OveralQual, X1stFlrSF, X2ndFlrSF, WoodDeckSF, ScreenPorch and GarageCars.

```
## [1] "RMSE of Testing Set:  0.14"
```

Splines

After finding a baseline using linear regression, the next idea was to try fitting regression splines. When looking at some of the more explanatory variables, it was decided to perform a spline on the variables OverallCond, GarageCars, X1stFlrSF and X2ndFlrSF. The best performing spline was X2ndFlrSF with degrees of freedom of 2. However, none of the splines performed remotely close to our baseline. This was expected because there are over 70 variables in the baseline model and a single spline of a significant variable isn't likely to perform better.

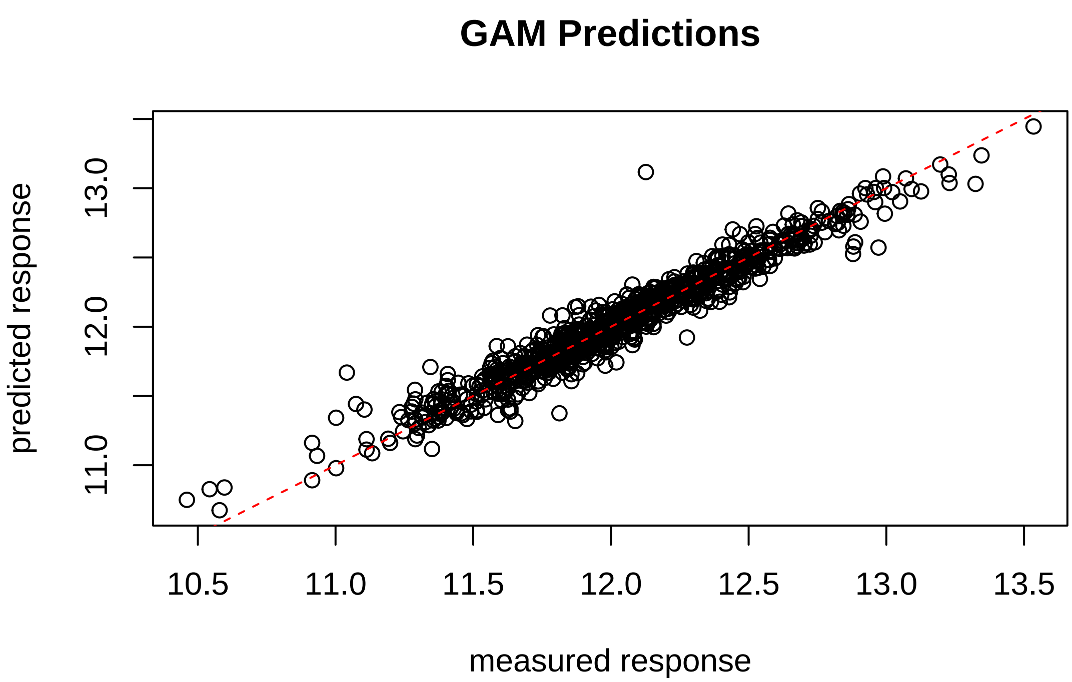
Splines	RMSE	RMSE_Dollars
X2ndFlrSF	0.4253	1.53
OverallCond	0.4456	1.56
X1stFlrSF	0.4695	1.60
GarageCars	0.4767	1.61

GAM Model

After fitting some regression splines, the next model we attempted was a General Additive Model or GAM. In general, this model is used for nonlinear relationships with splines on various variables. All available predictors were used to fit the GAM, along with the splines calculated above. There is definitely some improvement over the baseline as the RMSE = 0.1227 was below the baseline of 0.14. The final RMSE can be shifted depending on the which splines are used but, in general, this RMSE number is an improvement from our baseline model. The GAM combined with the splines is currently our best performing regression method.

```
## [1] "Test RMSE of GAM:  0.1227"
```

Below is a measured versus predicted plot. In general, our predictions follow the direction of the red target line with most points centered around the line. There are no visible splits in the predicted values and the measured responses. This compliments our lower RMSE score.



These are the data sets from our Kaggle Challenge.

We have two data sets

1] Train set (From row #1 to row #1460)

2] Test set (From row #1461 to row #2919)

```
## [1] 1460   81
```

```
## [1] 1459   80
```

```
## [1] 2919   81
```

Sales price column created in test df also has null values

Already the data has Id as row index, so we just remove the first column

```
## [1] 2919   80
```

We have,

1] 80 columns (including target) after droping ID column

2] Mixture of Numerical and Categorical data

```
## 'data.frame':    2919 obs. of  80 variables:
##  $ MSSubClass   : int  60 20 60 70 60 50 20 60 50 190 ...
##  $ MSZoning     : chr  "RL" "RL" "RL" "RL" ...
##  $ LotFrontage  : int  65 80 68 60 84 85 75 NA 51 50 ...
##  $ LotArea      : int  8450 9600 11250 9550 14260 14115 10084 10382 6120 7420 ...
##  $ Street       : chr  "Pave" "Pave" "Pave" "Pave" ...
##  $ Alley        : chr  NA NA NA NA ...
##  $ LotShape     : chr  "Reg" "Reg" "IR1" "IR1" ...
##  $ LandContour  : chr  "Lvl" "Lvl" "Lvl" "Lvl" ...
##  $ Utilities    : chr  "AllPub" "AllPub" "AllPub" "AllPub" ...
##  $ LotConfig    : chr  "Inside" "FR2" "Inside" "Corner" ...
##  $ LandSlope    : chr  "Gtl" "Gtl" "Gtl" "Gtl" ...
##  $ Neighborhood : chr  "CollgCr" "Veenker" "CollgCr" "Crawfor" ...
##  $ Condition1   : chr  "Norm" "Feedr" "Norm" "Norm" ...
##  $ Condition2   : chr  "Norm" "Norm" "Norm" "Norm" ...
##  $ BldgType     : chr  "1Fam" "1Fam" "1Fam" "1Fam" ...
##  $ HouseStyle   : chr  "2Story" "1Story" "2Story" "2Story" ...
##  $ OverallQual  : int  7 6 7 7 8 5 8 7 7 5 ...
##  $ OverallCond  : int  5 8 5 5 5 5 5 6 5 6 ...
##  $ YearBuilt    : int  2003 1976 2001 1915 2000 1993 2004 1973 1931 1939 ...
##  $ YearRemodAdd : int  2003 1976 2002 1970 2000 1995 2005 1973 1950 1950 ...
##  $ RoofStyle    : chr  "Gable" "Gable" "Gable" "Gable" ...
##  $ RoofMatl     : chr  "CompShg" "CompShg" "CompShg" "CompShg" ...
##  $ Exterior1st  : chr  "VinylSd" "MetalSd" "VinylSd" "Wd Sdng" ...
##  $ Exterior2nd  : chr  "VinylSd" "MetalSd" "VinylSd" "Wd Shng" ...
##  $ MasVnrType   : chr  "BrkFace" "None" "BrkFace" "None" ...
##  $ MasVnrArea   : int  196 0 162 0 350 0 186 240 0 0 ...
##  $ ExterQual    : chr  "Gd" "TA" "Gd" "TA" ...
##  $ ExterCond    : chr  "TA" "TA" "TA" "TA" ...
##  $ Foundation   : chr  "PConc" "CBlock" "PConc" "BrkTil" ...
##  $ BsmtQual     : chr  "Gd" "Gd" "Gd" "TA" ...
##  $ BsmtCond     : chr  "TA" "TA" "TA" "Gd" ...
##  $ BsmtExposure : chr  "No" "Gd" "Mn" "No" ...
##  $ BsmtFinType1 : chr  "GLQ" "ALQ" "GLQ" "ALQ" ...
##  $ BsmtFinSF1   : int  706 978 486 216 655 732 1369 859 0 851 ...
##  $ BsmtFinType2 : chr  "Unf" "Unf" "Unf" "Unf" ...
##  $ BsmtFinSF2   : int  0 0 0 0 0 0 0 32 0 0 ...
##  $ BsmtUnfSF    : int  150 284 434 540 490 64 317 216 952 140 ...
##  $ TotalBsmtSF  : int  856 1262 920 756 1145 796 1686 1107 952 991 ...
##  $ Heating      : chr  "GasA" "GasA" "GasA" "GasA" ...
##  $ HeatingQC    : chr  "Ex" "Ex" "Ex" "Gd" ...
##  $ CentralAir   : chr  "Y" "Y" "Y" "Y" ...
##  $ Electrical   : chr  "SBrkr" "SBrkr" "SBrkr" "SBrkr" ...
```

```
##  $ X1stFlrSF    : int  856 1262 920 961 1145 796 1694 1107 1022 1077 ...
##  $ X2ndFlrSF    : int  854 0 866 756 1053 566 0 983 752 0 ...
##  $ LowQualFinSF : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ GrLivArea    : int  1710 1262 1786 1717 2198 1362 1694 2090 1774 1077 ...
##  $ BsmtFullBath : int  1 0 1 1 1 1 1 1 0 1 ...
##  $ BsmtHalfBath : int  0 1 0 0 0 0 0 0 0 0 ...
##  $ FullBath     : int  2 2 2 1 2 1 2 2 2 1 ...
##  $ HalfBath     : int  1 0 1 0 1 1 0 1 0 0 ...
##  $ BedroomAbvGr : int  3 3 3 3 4 1 3 3 2 2 ...
##  $ KitchenAbvGr : int  1 1 1 1 1 1 1 1 2 2 ...
##  $ KitchenQual  : chr  "Gd" "TA" "Gd" "Gd" ...
##  $ TotRmsAbvGrd : int  8 6 6 7 9 5 7 7 8 5 ...
##  $ Functional   : chr  "Typ" "Typ" "Typ" "Typ" ...
##  $ Fireplaces   : int  0 1 1 1 1 0 1 2 2 2 ...
##  $ FireplaceQu  : chr  NA "TA" "TA" "Gd" ...
##  $ GarageType   : chr  "Attchd" "Attchd" "Attchd" "Detchd" ...
##  $ GarageYrBlt  : int  2003 1976 2001 1998 2000 1993 2004 1973 1931 1939 ...
##  $ GarageFinish : chr  "RFn" "RFn" "RFn" "Unf" ...
##  $ GarageCars   : int  2 2 2 3 3 2 2 2 2 1 ...
##  $ GarageArea   : int  548 460 608 642 836 480 636 484 468 205 ...
##  $ GarageQual   : chr  "TA" "TA" "TA" "TA" ...
##  $ GarageCond   : chr  "TA" "TA" "TA" "TA" ...
##  $ PavedDrive   : chr  "Y" "Y" "Y" "Y" ...
##  $ WoodDeckSF   : int  0 298 0 0 192 40 255 235 90 0 ...
##  $ OpenPorchSF  : int  61 0 42 35 84 30 57 204 0 4 ...
##  $ EnclosedPorch: int  0 0 0 272 0 0 0 228 205 0 ...
##  $ X3SsnPorch   : int  0 0 0 0 0 320 0 0 0 0 ...
##  $ ScreenPorch  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ PoolArea     : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ PoolQC       : chr  NA NA NA NA ...
##  $ Fence        : chr  NA NA NA NA ...
##  $ MiscFeature  : chr  NA NA NA NA ...
##  $ MiscVal      : int  0 0 0 0 0 700 0 350 0 0 ...
##  $ MoSold       : int  2 5 9 2 12 10 8 11 4 1 ...
##  $ YrSold       : int  2008 2007 2008 2006 2008 2009 2007 2009 2008 2008 ...
##  $ SaleType     : chr  "WD" "WD" "WD" "WD" ...
##  $ SaleCondition: chr  "Normal" "Normal" "Normal" "Abnorml" ...
##  $ SalePrice    : int  208500 181500 223500 140000 250000 143000 307000 200000 129900 118000 ...
```

Checking for null values and replacing them with mode for categorical features and median for numerical features

Started imputing nulls

For categorical varaible putting mode for missing value

and for numerical varibale putting median

PoolQC : data description says NA means “No Pool”. That make sense, given the huge ratio of missing value (+99%) and majority of houses have no Pool at all in general.

Utilities : For this categorical feature all records are “AllPub”, except for one “NoSeWa” and 2 NA . Since the house with ‘NoSewa’ is in the training set, this feature won’t help in predictive modelling. We can then safely remove it.

Lets check for remaining missing values

sale price will be dropped from training set and then we will have no null values

Separating categorical predictors for one hot encoding

For categorical features we will use one hot encoding Because most of categorical features have 3 to 5 types.

Changing object type data into category type. It will help in one hot endcoding

```
## [1] 1459
```

One Hot Encoding is a process in the data processing that is applied to categorical data, to convert it into a binary vector representation for use in machine learning algorithms

```
## [1] 1460   79
```

```
## [1] 0
```

```
## 'data.frame':    1460 obs. of  79 variables:
##  $ MSSubClass   : int  60 20 60 70 60 50 20 60 50 190 ...
##  $ MSZoning     : Factor w/ 5 levels "C (all)","FV",..: 4 4 4 4 4 4 4 4 5 4 ...
##  $ LotFrontage  : int  65 80 68 60 84 85 75 68 51 50 ...
##  $ LotArea      : int  8450 9600 11250 9550 14260 14115 10084 10382 6120 7420 ...
##  $ Street       : Factor w/ 2 levels "Grvl","Pave": 2 2 2 2 2 2 2 2 2 2 ...
##  $ Alley        : Factor w/ 3 levels "Grvl","None",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ LotShape     : Factor w/ 4 levels "IR1","IR2","IR3",..: 4 4 1 1 1 1 4 1 4 4 ...
##  $ LandContour  : Factor w/ 4 levels "Bnk","HLS","Low",..: 4 4 4 4 4 4 4 4 4 4 ...
##  $ LotConfig    : Factor w/ 5 levels "Corner","CulDSac",..: 5 3 5 1 3 5 5 1 5 1 ...
##  $ LandSlope    : Factor w/ 3 levels "Gtl","Mod","Sev": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Neighborhood : Factor w/ 25 levels "Blmngtn","Blueste",..: 6 25 6 7 14 12 21 17 18 4 ...
##  $ Condition1   : Factor w/ 9 levels "Artery","Feedr",..: 3 2 3 3 3 3 3 5 1 1 ...
##  $ Condition2   : Factor w/ 8 levels "Artery","Feedr",..: 3 3 3 3 3 3 3 3 3 1 ...
##  $ BldgType     : Factor w/ 5 levels "1Fam","2fmCon",..: 1 1 1 1 1 1 1 1 1 2 ...
##  $ HouseStyle   : Factor w/ 8 levels "1.5Fin","1.5Unf",..: 6 3 6 6 6 1 3 6 1 2 ...
##  $ OverallQual  : int  7 6 7 7 8 5 8 7 7 5 ...
##  $ OverallCond  : int  5 8 5 5 5 5 5 6 5 6 ...
##  $ YearBuilt    : int  2003 1976 2001 1915 2000 1993 2004 1973 1931 1939 ...
##  $ YearRemodAdd : int  2003 1976 2002 1970 2000 1995 2005 1973 1950 1950 ...
##  $ RoofStyle    : Factor w/ 6 levels "Flat","Gable",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ RoofMatl     : Factor w/ 8 levels "ClyTile","CompShg",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ Exterior1st  : Factor w/ 15 levels "AsbShng","AsphShn",..: 13 9 13 14 13 13 13 7 4 9 ...
##  $ Exterior2nd  : Factor w/ 16 levels "AsbShng","AsphShn",..: 14 9 14 16 14 14 14 7 16 9 ...
##  $ MasVnrType   : Factor w/ 4 levels "BrkCmn","BrkFace",..: 2 3 2 3 2 3 4 4 3 3 ...
##  $ MasVnrArea   : num  196 0 162 0 350 0 186 240 0 0 ...
##  $ ExterQual    : Factor w/ 4 levels "Ex","Fa","Gd",..: 3 4 3 4 3 4 3 4 4 4 ...
##  $ ExterCond    : Factor w/ 5 levels "Ex","Fa","Gd",..: 5 5 5 5 5 5 5 5 5 5 ...
##  $ Foundation   : Factor w/ 6 levels "BrkTil","CBlock",..: 3 2 3 1 3 6 3 2 1 1 ...
##  $ BsmtQual     : Factor w/ 5 levels "Ex","Fa","Gd",..: 3 3 3 5 3 3 1 3 5 5 ...
##  $ BsmtCond     : Factor w/ 5 levels "Fa","Gd","None",..: 5 5 5 2 5 5 5 5 5 5 ...
##  $ BsmtExposure : Factor w/ 5 levels "Av","Gd","Mn",..: 4 2 3 4 1 4 1 3 4 4 ...
##  $ BsmtFinType1 : Factor w/ 7 levels "ALQ","BLQ","GLQ",..: 3 1 3 1 3 3 3 1 7 3 ...
##  $ BsmtFinSF1   : num  706 978 486 216 655 ...
##  $ BsmtFinType2 : Factor w/ 7 levels "ALQ","BLQ","GLQ",..: 7 7 7 7 7 7 7 2 7 7 ...
##  $ BsmtFinSF2   : num  0 0 0 0 0 0 0 32 0 0 ...
##  $ BsmtUnfSF    : num  150 284 434 540 490 64 317 216 952 140 ...
##  $ TotalBsmtSF  : num  856 1262 920 756 1145 ...
##  $ Heating      : Factor w/ 6 levels "Floor","GasA",..: 2 2 2 2 2 2 2 2 2 2 ...
```

```
##  $ HeatingQC    : Factor w/ 5 levels "Ex","Fa","Gd",..: 1 1 1 3 1 1 1 1 3 1 ...
##  $ CentralAir   : Factor w/ 2 levels "N","Y": 2 2 2 2 2 2 2 2 2 2 ...
##  $ Electrical   : Factor w/ 5 levels "FuseA","FuseF",..: 5 5 5 5 5 5 5 5 2 5 ...
##  $ X1stFlrSF    : int  856 1262 920 961 1145 796 1694 1107 1022 1077 ...
##  $ X2ndFlrSF    : int  854 0 866 756 1053 566 0 983 752 0 ...
##  $ LowQualFinSF : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ GrLivArea    : int  1710 1262 1786 1717 2198 1362 1694 2090 1774 1077 ...
##  $ BsmtFullBath : num  1 0 1 1 1 1 1 1 0 1 ...
##  $ BsmtHalfBath : num  0 1 0 0 0 0 0 0 0 0 ...
##  $ FullBath     : int  2 2 2 1 2 1 2 2 2 1 ...
##  $ HalfBath     : int  1 0 1 0 1 1 0 1 0 0 ...
##  $ BedroomAbvGr : int  3 3 3 3 4 1 3 3 2 2 ...
##  $ KitchenAbvGr : int  1 1 1 1 1 1 1 1 2 2 ...
##  $ KitchenQual  : Factor w/ 4 levels "Ex","Fa","Gd",..: 3 4 3 3 3 4 3 4 4 4 ...
##  $ TotRmsAbvGrd : int  8 6 6 7 9 5 7 7 8 5 ...
##  $ Functional   : Factor w/ 7 levels "Maj1","Maj2",..: 7 7 7 7 7 7 7 7 3 7 ...
##  $ Fireplaces   : int  0 1 1 1 1 0 1 2 2 2 ...
##  $ FireplaceQu  : Factor w/ 6 levels "Ex","Fa","Gd",..: 4 6 6 3 6 4 3 6 6 6 ...
##  $ GarageType   : Factor w/ 7 levels "2Types","Attchd",..: 2 2 2 6 2 2 2 2 6 2 ...
##  $ GarageYrBlt  : num  2003 1976 2001 1998 2000 ...
##  $ GarageFinish : Factor w/ 4 levels "Fin","None","RFn",..: 3 3 3 4 3 4 3 3 4 3 ...
##  $ GarageCars   : num  2 2 2 3 3 2 2 2 2 1 ...
##  $ GarageArea   : num  548 460 608 642 836 480 636 484 468 205 ...
##  $ GarageQual   : Factor w/ 6 levels "Ex","Fa","Gd",..: 6 6 6 6 6 6 6 6 2 3 ...
##  $ GarageCond   : Factor w/ 6 levels "Ex","Fa","Gd",..: 6 6 6 6 6 6 6 6 6 6 ...
##  $ PavedDrive   : Factor w/ 3 levels "N","P","Y": 3 3 3 3 3 3 3 3 3 3 ...
##  $ WoodDeckSF   : int  0 298 0 0 192 40 255 235 90 0 ...
##  $ OpenPorchSF  : int  61 0 42 35 84 30 57 204 0 4 ...
##  $ EnclosedPorch: int  0 0 0 272 0 0 0 228 205 0 ...
##  $ X3SsnPorch   : int  0 0 0 0 0 320 0 0 0 0 ...
##  $ ScreenPorch  : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ PoolArea     : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ PoolQC       : Factor w/ 4 levels "Ex","Fa","Gd",..: 4 4 4 4 4 4 4 4 4 4 ...
##  $ Fence        : Factor w/ 5 levels "GdPrv","GdWo",..: 5 5 5 5 5 3 5 5 5 5 ...
##  $ MiscFeature  : Factor w/ 5 levels "Gar2","None",..: 2 2 2 2 2 4 2 4 2 2 ...
##  $ MiscVal      : int  0 0 0 0 0 700 0 350 0 0 ...
##  $ MoSold       : int  2 5 9 2 12 10 8 11 4 1 ...
##  $ YrSold       : int  2008 2007 2008 2006 2008 2009 2007 2009 2008 2008 ...
##  $ SaleType     : Factor w/ 9 levels "COD","Con","ConLD",..: 9 9 9 9 9 9 9 9 9 9 ...
##  $ SaleCondition: Factor w/ 6 levels "Abnorml","AdjLand",..: 5 5 5 1 5 5 5 5 1 5 ...
##  $ SalePrice    : int  208500 181500 223500 140000 250000 143000 307000 200000 129900 118000 ...

## [1] 1459   79

## [1] 1459

## 'data.frame':    1459 obs. of  79 variables:
##  $ MSSubClass   : int  20 20 60 60 120 60 20 60 20 20 ...
##  $ MSZoning     : Factor w/ 5 levels "C (all)","FV",..: 3 4 4 4 4 4 4 4 4 4 ...
##  $ LotFrontage  : int  80 81 74 78 43 75 68 63 85 70 ...
##  $ LotArea      : int  11622 14267 13830 9978 5005 10000 7980 8402 10176 8400 ...
##  $ Street       : Factor w/ 2 levels "Grvl","Pave": 2 2 2 2 2 2 2 2 2 2 ...
##  $ Alley        : Factor w/ 3 levels "Grvl","None",..: 2 2 2 2 2 2 2 2 2 2 ...
```

```
##  $ LotShape     : Factor w/ 4 levels "IR1","IR2","IR3",..: 4 1 1 1 1 1 1 1 4 4 ...
##  $ LandContour  : Factor w/ 4 levels "Bnk","HLS","Low",..: 4 4 4 4 2 4 4 4 4 4 ...
##  $ LotConfig    : Factor w/ 5 levels "Corner","CulDSac",..: 5 1 5 5 5 1 5 5 5 1 ...
##  $ LandSlope    : Factor w/ 3 levels "Gtl","Mod","Sev": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Neighborhood : Factor w/ 25 levels "Blmngtn","Blueste",..: 13 13 9 9 22 9 9 9 9 13 ...
##  $ Condition1   : Factor w/ 9 levels "Artery","Feedr",..: 2 3 3 3 3 3 3 3 3 3 ...
##  $ Condition2   : Factor w/ 8 levels "Artery","Feedr",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ BldgType     : Factor w/ 5 levels "1Fam","2fmCon",..: 1 1 1 1 5 1 1 1 1 1 ...
##  $ HouseStyle   : Factor w/ 8 levels "1.5Fin","1.5Unf",..: 3 3 6 6 3 6 3 6 3 3 ...
##  $ OverallQual  : int  5 6 5 6 8 6 6 6 7 4 ...
##  $ OverallCond  : int  6 6 5 6 5 5 7 5 5 5 ...
##  $ YearBuilt    : int  1961 1958 1997 1998 1992 1993 1992 1998 1990 1970 ...
##  $ YearRemodAdd : int  1961 1958 1998 1998 1992 1994 2007 1998 1990 1970 ...
##  $ RoofStyle    : Factor w/ 6 levels "Flat","Gable",..: 2 4 2 2 2 2 2 2 2 2 ...
##  $ RoofMatl     : Factor w/ 8 levels "ClyTile","CompShg",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ Exterior1st  : Factor w/ 15 levels "AsbShng","AsphShn",..: 13 14 13 13 7 7 7 13 7 10 ...
##  $ Exterior2nd  : Factor w/ 16 levels "AsbShng","AsphShn",..: 14 15 14 14 7 7 7 14 7 11 ...
##  $ MasVnrType   : Factor w/ 4 levels "BrkCmn","BrkFace",..: 3 2 3 2 3 3 3 3 3 3 ...
##  $ MasVnrArea   : num  0 108 0 20 0 0 0 0 0 0 ...
##  $ ExterQual    : Factor w/ 4 levels "Ex","Fa","Gd",..: 4 4 4 4 3 4 4 4 4 4 ...
##  $ ExterCond    : Factor w/ 5 levels "Ex","Fa","Gd",..: 5 5 5 5 5 5 3 5 5 5 ...
##  $ Foundation   : Factor w/ 6 levels "BrkTil","CBlock",..: 2 2 3 3 3 3 3 3 3 2 ...
##  $ BsmtQual     : Factor w/ 5 levels "Ex","Fa","Gd",..: 5 5 3 5 3 3 3 3 3 5 ...
##  $ BsmtCond     : Factor w/ 5 levels "Fa","Gd","None",..: 5 5 5 5 5 5 5 5 5 5 ...
##  $ BsmtExposure : Factor w/ 5 levels "Av","Gd","Mn",..: 4 4 4 4 4 4 4 4 2 4 ...
##  $ BsmtFinType1 : Factor w/ 7 levels "ALQ","BLQ","GLQ",..: 6 1 3 3 1 7 1 7 3 1 ...
##  $ BsmtFinSF1   : num  468 923 791 602 263 0 935 0 637 804 ...
##  $ BsmtFinType2 : Factor w/ 7 levels "ALQ","BLQ","GLQ",..: 4 7 7 7 7 7 7 7 7 6 ...
##  $ BsmtFinSF2   : num  144 0 0 0 0 0 0 0 0 78 ...
##  $ BsmtUnfSF    : num  270 406 137 324 1017 ...
##  $ TotalBsmtSF  : num  882 1329 928 926 1280 ...
##  $ Heating      : Factor w/ 6 levels "Floor","GasA",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ HeatingQC    : Factor w/ 5 levels "Ex","Fa","Gd",..: 5 5 3 1 1 3 1 3 3 5 ...
##  $ CentralAir   : Factor w/ 2 levels "N","Y": 2 2 2 2 2 2 2 2 2 2 ...
##  $ Electrical   : Factor w/ 5 levels "FuseA","FuseF",..: 5 5 5 5 5 5 5 5 5 5 ...
##  $ X1stFlrSF    : int  896 1329 928 926 1280 763 1187 789 1341 882 ...
##  $ X2ndFlrSF    : int  0 0 701 678 0 892 0 676 0 0 ...
##  $ LowQualFinSF : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ GrLivArea    : int  896 1329 1629 1604 1280 1655 1187 1465 1341 882 ...
##  $ BsmtFullBath : num  0 0 0 0 0 0 1 0 1 1 ...
##  $ BsmtHalfBath : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ FullBath     : int  1 1 2 2 2 2 2 2 1 1 ...
##  $ HalfBath     : int  0 1 1 1 0 1 0 1 1 0 ...
##  $ BedroomAbvGr : int  2 3 3 3 2 3 3 3 2 2 ...
##  $ KitchenAbvGr : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ KitchenQual  : Factor w/ 4 levels "Ex","Fa","Gd",..: 4 3 4 3 3 4 4 4 3 4 ...
##  $ TotRmsAbvGrd : int  5 6 6 7 5 7 6 7 5 4 ...
##  $ Functional   : Factor w/ 7 levels "Maj1","Maj2",..: 7 7 7 7 7 7 7 7 7 7 ...
##  $ Fireplaces   : int  0 0 1 1 0 1 0 1 1 0 ...
##  $ FireplaceQu  : Factor w/ 6 levels "Ex","Fa","Gd",..: 4 4 6 3 4 6 4 3 5 4 ...
##  $ GarageType   : Factor w/ 7 levels "2Types","Attchd",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ GarageYrBlt  : num  1961 1958 1997 1998 1992 ...
##  $ GarageFinish : Factor w/ 4 levels "Fin","None","RFn",..: 4 4 1 1 3 1 1 1 4 1 ...
##  $ GarageCars   : num  1 1 2 2 2 2 2 2 2 2 ...
```

```
##  $ GarageArea   : num  730 312 482 470 506 440 420 393 506 525 ...
##  $ GarageQual   : Factor w/ 6 levels "Ex","Fa","Gd",..: 6 6 6 6 6 6 6 6 6 6 ...
##  $ GarageCond   : Factor w/ 6 levels "Ex","Fa","Gd",..: 6 6 6 6 6 6 6 6 6 6 ...
##  $ PavedDrive   : Factor w/ 3 levels "N","P","Y": 3 3 3 3 3 3 3 3 3 3 ...
##  $ WoodDeckSF   : int  140 393 212 360 0 157 483 0 192 240 ...
##  $ OpenPorchSF  : int  0 36 34 36 82 84 21 75 0 0 ...
##  $ EnclosedPorch: int  0 0 0 0 0 0 0 0 0 0 ...
##  $ X3SsnPorch   : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ ScreenPorch  : int  120 0 0 0 144 0 0 0 0 0 ...
##  $ PoolArea     : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ PoolQC       : Factor w/ 4 levels "Ex","Fa","Gd",..: 4 4 4 4 4 4 4 4 4 4 ...
##  $ Fence        : Factor w/ 5 levels "GdPrv","GdWo",..: 3 5 3 5 5 5 1 5 5 3 ...
##  $ MiscFeature  : Factor w/ 5 levels "Gar2","None",..: 2 1 2 2 2 2 4 2 2 2 ...
##  $ MiscVal      : int  0 12500 0 0 0 0 500 0 0 0 ...
##  $ MoSold       : int  6 6 3 6 1 4 3 5 2 4 ...
##  $ YrSold       : int  2010 2010 2010 2010 2010 2010 2010 2010 2010 2010 ...
##  $ SaleType     : Factor w/ 9 levels "COD","Con","ConLD",..: 9 9 9 9 9 9 9 9 9 9 ...
##  $ SaleCondition: Factor w/ 6 levels "Abnorml","AdjLand",..: 5 5 5 5 5 5 5 5 5 5 ...
##  $ SalePrice    : int  NA NA NA NA NA NA NA NA NA NA ...
```

Spliting the data frame into numerical and categorical predictors for feature engineering

```
## [1] 1460   42
```

```
## [1] 1460   37
```

converting categorical features into binary for train set

ohe_train_total is the training data where categorical predicators are in binary this is final train data for Machine learning

NOw test data cetegorical predictors into binary

```
## [1] 1459   42
```

```
## [1] 1459   37
```

converting categorical features into binary for test set

ohe_test_total is the testing data where categorical predicators are in binary this is final test data for Machine learning

```
## [1] 1459  300
```

test set is ready

PCR model

Is PCA + regression

principle components analysis takes in the predictor data frame and compresses it. It is used to reduce the number of predictors.

We will train our PCR on training data now

Spliting the data into predictor dataframe which is X

and

Response variable which is y

```
## [1] 1460  301
```

Removing response variable

```
## [1] 1460  300
```

Putting response variable in y

```
## [1] 1460   81
```

Log tranformation of Response variable

Train test split

To perform PCR we perfoem regression on the principle components derived from PCA

Creating a Regression intance

```
## Data:    X dimension: 1022 300
##  Y dimension: 1022 1
## Fit method: svdpc
## Number of components considered: 300
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV          0.3973   0.2093   0.2097   0.1929   0.1789   0.1767   0.1751
## adjCV       0.3973   0.2091   0.2097   0.1891   0.1777   0.1759   0.1743
##        7 comps  8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
## CV      0.1748   0.1761   0.1761    0.1756    0.1748    0.1669    0.1659
## adjCV   0.1740   0.1759   0.1762    0.1774    0.1805    0.1662    0.1647
##        14 comps  15 comps  16 comps  17 comps  18 comps  19 comps  20 comps
## CV       0.1654    0.1648    0.1651    0.1649    0.1656    0.1654    0.1657
## adjCV    0.1650    0.1630    0.1647    0.1644    0.1656    0.1654    0.1657
##        21 comps  22 comps  23 comps  24 comps  25 comps  26 comps  27 comps
## CV       0.1655    0.1645    0.1643    0.1633    0.1627    0.1625    0.1622
## adjCV    0.1663    0.1647    0.1653    0.1608    0.1610    0.1608    0.1607
##        28 comps  29 comps  30 comps  31 comps  32 comps  33 comps  34 comps
## CV       0.1619    0.1614    0.1613    0.1614    0.1612    0.1619    0.1618
## adjCV    0.1599    0.1599    0.1600    0.1603    0.1603    0.1614    0.1616
##        35 comps  36 comps  37 comps  38 comps  39 comps  40 comps  41 comps
## CV       0.1613    0.1608    0.1605    0.1609    0.1607    0.1614    0.1604
## adjCV    0.1614    0.1599    0.1596    0.1596    0.1598    0.1605    0.1595
##        42 comps  43 comps  44 comps  45 comps  46 comps  47 comps  48 comps
## CV       0.1595    0.1594    0.1593    0.1588    0.1582    0.1582    0.1578
## adjCV    0.1593    0.1592    0.1590    0.1583    0.1565    0.1568    0.1561
##        49 comps  50 comps  51 comps  52 comps  53 comps  54 comps  55 comps
## CV       0.1577    0.1577    0.1574    0.1569    0.1562    0.1563    0.1557
## adjCV    0.1552    0.1549    0.1549    0.1547    0.1540    0.1541    0.1536
##        56 comps  57 comps  58 comps  59 comps  60 comps  61 comps  62 comps
## CV       0.1557    0.1557    0.1557    0.1553    0.1555    0.1555    0.1553
## adjCV    0.1537    0.1539    0.1540    0.1538    0.1541    0.1542    0.1541
##        63 comps  64 comps  65 comps  66 comps  67 comps  68 comps  69 comps
## CV       0.1554    0.1552    0.1562    0.1562    0.1559    0.1553    0.1551
```

```
## adjCV    0.1536    0.1537    0.1547    0.1551    0.1547    0.1541    0.1539
##        70 comps  71 comps  72 comps  73 comps  74 comps  75 comps  76 comps
## CV       0.1546    0.1547    0.1544    0.1547    0.1547    0.1549    0.1550
## adjCV    0.1529    0.1530    0.1529    0.1533    0.1527    0.1530    0.1532
##        77 comps  78 comps  79 comps  80 comps  81 comps  82 comps  83 comps
## CV       0.1555    0.1553    0.1554    0.1558    0.1556    0.1566    0.1566
## adjCV    0.1539    0.1538    0.1536    0.1542    0.1542    0.1553    0.1547
##        84 comps  85 comps  86 comps  87 comps  88 comps  89 comps  90 comps
## CV       0.1566    0.1576    0.1570    0.1562    0.1560    0.1561    0.1566
## adjCV    0.1550    0.1560    0.1556    0.1551    0.1551    0.1549    0.1543
##        91 comps  92 comps  93 comps  94 comps  95 comps  96 comps  97 comps
## CV       0.1561    0.1561    0.1566    0.1560    0.1557    0.1556    0.1556
## adjCV    0.1539    0.1541    0.1547    0.1539    0.1537    0.1531    0.1532
##        98 comps  99 comps  100 comps  101 comps  102 comps  103 comps
## CV       0.1557    0.1558     0.1561     0.1565     0.1563     0.1562
## adjCV    0.1533    0.1535     0.1538     0.1544     0.1542     0.1543
##        104 comps  105 comps  106 comps  107 comps  108 comps  109 comps
## CV        0.1566     0.1565     0.1567     0.1575     0.1575     0.1579
## adjCV     0.1548     0.1549     0.1550     0.1557     0.1556     0.1560
##        110 comps  111 comps  112 comps  113 comps  114 comps  115 comps
## CV        0.1577     0.1572     0.1574     0.1576     0.1578     0.1574
## adjCV     0.1560     0.1558     0.1554     0.1555     0.1559     0.1554
##        116 comps  117 comps  118 comps  119 comps  120 comps  121 comps
## CV        0.1570     0.1574     0.1569     0.1576     0.1579     0.1583
## adjCV     0.1553     0.1557     0.1551     0.1553     0.1556     0.1558
##        122 comps  123 comps  124 comps  125 comps  126 comps  127 comps
## CV        0.1588     0.1586     0.1584     0.1587     0.1583     0.1584
## adjCV     0.1564     0.1561     0.1557     0.1562     0.1559     0.1561
##        128 comps  129 comps  130 comps  131 comps  132 comps  133 comps
## CV        0.1585     0.1592     0.1591     0.1589     0.1589     0.1590
## adjCV     0.1562     0.1567     0.1566     0.1563     0.1564     0.1561
##        134 comps  135 comps  136 comps  137 comps  138 comps  139 comps
## CV        0.1587     0.1584     0.1587      0.159     0.1580     0.1583
## adjCV     0.1555     0.1554     0.1557      0.156     0.1548     0.1551
##        140 comps  141 comps  142 comps  143 comps  144 comps  145 comps
## CV        0.1575     0.1574     0.1563     0.1556     0.1549     0.1547
## adjCV     0.1546     0.1542     0.1531     0.1527     0.1522     0.1517
##        146 comps  147 comps  148 comps  149 comps  150 comps  151 comps
## CV        0.1548     0.1548     0.1549     0.1550     0.1557     0.1558
## adjCV     0.1518     0.1517     0.1519     0.1518     0.1524     0.1525
##        152 comps  153 comps  154 comps  155 comps  156 comps  157 comps
## CV        0.1559     0.1559     0.1555     0.1554     0.1560     0.1567
## adjCV     0.1526     0.1527     0.1524     0.1524     0.1531     0.1536
##        158 comps  159 comps  160 comps  161 comps  162 comps  163 comps
## CV        0.1564     0.1562     0.1567     0.1561     0.1559     0.1555
## adjCV     0.1536     0.1535     0.1540     0.1535     0.1526     0.1522
##        164 comps  165 comps  166 comps  167 comps  168 comps  169 comps
## CV        0.1563     0.1555     0.1546     0.1558     0.1558     0.1557
## adjCV     0.1531     0.1525     0.1517     0.1530     0.1530     0.1530
##        170 comps  171 comps  172 comps  173 comps  174 comps  175 comps
## CV        0.1555     0.1555     0.1551     0.1532     0.1532     0.1534
## adjCV     0.1527     0.1526     0.1520     0.1502     0.1503     0.1505
##        176 comps  177 comps  178 comps  179 comps  180 comps  181 comps
## CV        0.1530     0.1527     0.1538     0.1526     0.1531     0.1530
```

```
## adjCV     0.1502     0.1500     0.1512     0.1494     0.1501     0.1497
##        182 comps  183 comps  184 comps  185 comps  186 comps  187 comps
## CV        0.1532     0.1527     0.1532     0.1534     0.1529     0.1529
## adjCV     0.1499     0.1494     0.1496     0.1499     0.1496     0.1497
##        188 comps  189 comps  190 comps  191 comps  192 comps  193 comps
## CV        0.1537     0.1535     0.1537     0.1531     0.1532     0.1531
## adjCV     0.1504     0.1501     0.1504     0.1496     0.1497     0.1496
##        194 comps  195 comps  196 comps  197 comps  198 comps  199 comps
## CV        0.1535     0.1538     0.1548     0.1554     0.1558     0.1568
## adjCV     0.1499     0.1502     0.1512     0.1518     0.1521     0.1531
##        200 comps  201 comps  202 comps  203 comps  204 comps  205 comps
## CV        0.1577     0.1591     0.1588     0.1581     0.1583     0.1579
## adjCV     0.1540     0.1553     0.1550     0.1542     0.1544     0.1542
##        206 comps  207 comps  208 comps  209 comps  210 comps  211 comps
## CV        0.1580     0.1579     0.1571     0.1584     0.1589     0.1590
## adjCV     0.1546     0.1543     0.1535     0.1546     0.1550     0.1549
##        212 comps  213 comps  214 comps  215 comps  216 comps  217 comps
## CV        0.1581     0.1584     0.1575     0.1584     0.1583     0.1592
## adjCV     0.1542     0.1546     0.1539     0.1546     0.1542     0.1551
##        218 comps  219 comps  220 comps  221 comps  222 comps  223 comps
## CV        0.1610     0.1610     0.1618     0.1624     0.1630     0.1633
## adjCV     0.1568     0.1568     0.1573     0.1580     0.1585     0.1587
##        224 comps  225 comps  226 comps  227 comps  228 comps  229 comps
## CV        0.1640     0.1631     0.1627     0.1620     0.1642     0.1654
## adjCV     0.1594     0.1585     0.1582     0.1577     0.1600     0.1610
##        230 comps  231 comps  232 comps  233 comps  234 comps  235 comps
## CV        0.1647     0.1645     0.1648     0.1646     0.1641     0.1644
## adjCV     0.1600     0.1597     0.1599     0.1597     0.1594     0.1596
##        236 comps  237 comps  238 comps  239 comps  240 comps  241 comps
## CV        0.1702  4.587e+09  3.158e+10  6.518e+10  8.835e+10  9.138e+10
## adjCV     0.1649  4.352e+09  2.996e+10  6.184e+10  8.383e+10  8.670e+10
##        242 comps  243 comps  244 comps  245 comps  246 comps  247 comps
## CV     1.255e+11  1.257e+11  1.317e+11  1.883e+11  1.886e+11  1.893e+11
## adjCV  1.191e+11  1.193e+11  1.250e+11  1.786e+11  1.790e+11  1.796e+11
##        248 comps  249 comps  250 comps  251 comps  252 comps  253 comps
## CV     2.101e+11  2.368e+11  3.085e+11  2.984e+11  3.248e+11  3.902e+11
## adjCV  1.994e+11  2.246e+11  2.926e+11  2.831e+11  3.081e+11  3.702e+11
##        254 comps  255 comps  256 comps  257 comps  258 comps  259 comps
## CV     4.179e+11  4.709e+11  6.482e+11  7.391e+11  8.230e+11  7.785e+11
## adjCV  3.965e+11  4.467e+11  6.150e+11  7.012e+11  7.808e+11  7.387e+11
##        260 comps  261 comps  262 comps  263 comps  264 comps  265 comps
## CV     9.040e+11  8.875e+11  9.192e+11  9.234e+11  1.475e+12  1.554e+12
## adjCV  8.577e+11  8.420e+11  8.721e+11  8.761e+11  1.400e+12  1.474e+12
##        266 comps  267 comps  268 comps  269 comps  270 comps  271 comps
## CV     1.774e+12  1.882e+12  1.770e+12   2.32e+12  2.622e+12  3.512e+12
## adjCV  1.683e+12  1.785e+12  1.679e+12   2.20e+12  2.488e+12  3.331e+12
##        272 comps  273 comps  274 comps  275 comps  276 comps  277 comps
## CV     3.876e+12  4.064e+12  3.705e+12  3.686e+12  3.254e+12  3.392e+12
## adjCV  3.676e+12  3.855e+12  3.514e+12  3.496e+12  3.087e+12  3.218e+12
##        278 comps  279 comps  280 comps  281 comps  282 comps  283 comps
## CV     3.482e+12  3.612e+12  5.232e+12  5.392e+12  5.704e+12  6.839e+12
## adjCV  3.304e+12  3.427e+12  4.964e+12  5.116e+12  5.412e+12  6.489e+12
##        284 comps  285 comps  286 comps  287 comps  288 comps  289 comps
## CV     1.041e+13  1.681e+13  1.846e+13  1.778e+13  2.069e+13  2.193e+13
```

```
## adjCV  9.872e+12  1.595e+13  1.751e+13  1.687e+13  1.963e+13  2.080e+13
##        290 comps  291 comps  292 comps  293 comps  294 comps  295 comps
## CV     2.059e+13  2.766e+13  3.004e+13  3.177e+13  3.198e+13  3.488e+13
## adjCV  1.953e+13  2.624e+13  2.850e+13  3.014e+13  3.034e+13  3.309e+13
##        296 comps  297 comps  298 comps  299 comps  300 comps
## CV     3.537e+13  3.490e+13  3.715e+13  3.742e+13  9.774e+13
## adjCV  3.356e+13  3.311e+13  3.525e+13  3.550e+13  9.270e+13
##
## TRAINING: % variance explained
##          1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
## X           6.56     9.62    12.24    14.75    16.92    18.82    20.43    21.99
## y_train    72.16    72.23    76.49    79.38    79.81    80.23    80.43    80.44
##          9 comps  10 comps  11 comps  12 comps  13 comps  14 comps  15 comps
## X          23.38     24.66     25.92     27.16     28.32     29.37     30.39
## y_train    81.32     81.35     81.50     83.82     84.39     84.55     84.95
##          16 comps  17 comps  18 comps  19 comps  20 comps  21 comps  22 comps
## X           31.39     32.38     33.35     34.29      35.2     36.10     36.99
## y_train     84.97     85.06     85.09     85.20      85.3     85.42     85.72
##          23 comps  24 comps  25 comps  26 comps  27 comps  28 comps  29 comps
## X           37.86     38.72     39.56     40.40     41.21     42.01     42.82
## y_train     85.74     86.68     86.68     86.84     86.90     87.05     87.06
##          30 comps  31 comps  32 comps  33 comps  34 comps  35 comps  36 comps
## X           43.60     44.37     45.12     45.87      46.6     47.33     48.03
## y_train     87.07     87.07     87.07     87.08      87.1     87.10     87.56
##          37 comps  38 comps  39 comps  40 comps  41 comps  42 comps  43 comps
## X           48.73     49.42     50.10     50.78     51.45      52.1     52.75
## y_train     87.66     87.84     87.86     87.94     88.09      88.1     88.21
##          44 comps  45 comps  46 comps  47 comps  48 comps  49 comps  50 comps
## X           53.38     54.00     54.63     55.24     55.84     56.44     57.03
## y_train     88.30     88.41     88.88     88.88     89.06     89.28     89.45
##          51 comps  52 comps  53 comps  54 comps  55 comps  56 comps  57 comps
## X           57.61     58.18     58.74     59.30     59.85     60.40     60.93
## y_train     89.46     89.46     89.53     89.55     89.59     89.62     89.62
##          58 comps  59 comps  60 comps  61 comps  62 comps  63 comps  64 comps
## X           61.46     61.99     62.51     63.03     63.54     64.05     64.56
## y_train     89.64     89.66     89.69     89.70     89.71     89.88     89.88
##          65 comps  66 comps  67 comps  68 comps  69 comps  70 comps  71 comps
## X           65.06     65.55     66.03     66.51     66.99     67.46     67.92
## y_train     89.91     89.91     89.94     89.99     90.06     90.19     90.22
##          72 comps  73 comps  74 comps  75 comps  76 comps  77 comps  78 comps
## X           68.38     68.84     69.29     69.73     70.16     70.60     71.03
## y_train     90.24     90.25     90.38     90.40     90.42     90.42     90.43
##          79 comps  80 comps  81 comps  82 comps  83 comps  84 comps  85 comps
## X           71.45     71.86     72.28     72.69     73.09     73.49     73.89
## y_train     90.49     90.50     90.52     90.52     90.66     90.67     90.69
##          86 comps  87 comps  88 comps  89 comps  90 comps  91 comps  92 comps
## X           74.28     74.66     75.04     75.42     75.79     76.17     76.54
## y_train     90.69     90.70     90.70     90.77     91.01     91.05     91.05
##          93 comps  94 comps  95 comps  96 comps  97 comps  98 comps  99 comps
## X           76.90     77.26     77.61     77.96     78.31     78.65     78.99
## y_train     91.06     91.11     91.12     91.24     91.25     91.26     91.26
##          100 comps  101 comps  102 comps  103 comps  104 comps  105 comps
## X            79.33      79.66      79.99      80.32      80.64      80.95
## y_train      91.28      91.28      91.31      91.32      91.32      91.33
```

```
##          106 comps  107 comps  108 comps  109 comps  110 comps  111 comps
## X            81.27      81.58      81.89      82.20       82.5       82.8
## y_train      91.36      91.40      91.46      91.47       91.5       91.5
##          112 comps  113 comps  114 comps  115 comps  116 comps  117 comps
## X            83.10      83.39      83.68      83.97      84.25      84.53
## y_train      91.65      91.69      91.70      91.76      91.78      91.79
##          118 comps  119 comps  120 comps  121 comps  122 comps  123 comps
## X            84.81      85.09      85.35      85.62      85.88      86.14
## y_train      91.89      92.03      92.06      92.13      92.14      92.18
##          124 comps  125 comps  126 comps  127 comps  128 comps  129 comps
## X            86.40      86.64      86.89      87.14      87.38      87.62
## y_train      92.25      92.25      92.26      92.27      92.28      92.35
##          130 comps  131 comps  132 comps  133 comps  134 comps  135 comps
## X            87.86      88.10      88.33      88.56      88.79      89.01
## y_train      92.35      92.43      92.44      92.55      92.63      92.63
##          136 comps  137 comps  138 comps  139 comps  140 comps  141 comps
## X            89.24      89.46      89.68      89.89      90.10      90.32
## y_train      92.65      92.65      92.75      92.77      92.77      92.86
##          142 comps  143 comps  144 comps  145 comps  146 comps  147 comps
## X            90.53      90.73      90.93      91.13      91.33      91.52
## y_train      92.90      92.91      92.91      92.97      93.00      93.02
##          148 comps  149 comps  150 comps  151 comps  152 comps  153 comps
## X            91.72      91.91      92.10      92.29      92.47      92.66
## y_train      93.02      93.11      93.15      93.15      93.16      93.16
##          154 comps  155 comps  156 comps  157 comps  158 comps  159 comps
## X            92.84      93.01      93.19      93.36      93.53      93.69
## y_train      93.21      93.22      93.23      93.25      93.25      93.25
##          160 comps  161 comps  162 comps  163 comps  164 comps  165 comps
## X            93.86      94.02      94.18      94.34      94.49      94.64
## y_train      93.25      93.27      93.43      93.44      93.44      93.44
##          166 comps  167 comps  168 comps  169 comps  170 comps  171 comps
## X            94.79      94.94      95.09      95.23      95.37      95.51
## y_train      93.47      93.48      93.50      93.50      93.54      93.59
##          172 comps  173 comps  174 comps  175 comps  176 comps  177 comps
## X            95.65      95.78      95.92      96.05      96.18      96.30
## y_train      93.67      93.70      93.70      93.73      93.75      93.76
##          178 comps  179 comps  180 comps  181 comps  182 comps  183 comps
## X            96.43      96.55      96.67      96.79      96.90      97.02
## y_train      93.77      93.93      93.93      94.02      94.03      94.03
##          184 comps  185 comps  186 comps  187 comps  188 comps  189 comps
## X            97.13      97.24      97.35      97.46      97.56      97.66
## y_train      94.11      94.11      94.13      94.13      94.14      94.18
##          190 comps  191 comps  192 comps  193 comps  194 comps  195 comps
## X            97.75      97.85      97.94      98.03      98.11      98.20
## y_train      94.18      94.26      94.28      94.28      94.33      94.33
##          196 comps  197 comps  198 comps  199 comps  200 comps  201 comps
## X            98.29      98.37      98.44      98.52      98.59      98.66
## y_train      94.33      94.34      94.36      94.36      94.36      94.38
##          202 comps  203 comps  204 comps  205 comps  206 comps  207 comps
## X            98.73      98.80      98.87      98.93      98.99      99.05
## y_train      94.40      94.46      94.47      94.47      94.47      94.53
##          208 comps  209 comps  210 comps  211 comps  212 comps  213 comps
## X            99.11      99.16      99.22      99.27      99.32      99.37
## y_train      94.56      94.61      94.63      94.68      94.69      94.69
```

```
##          214 comps  215 comps  216 comps  217 comps  218 comps  219 comps
## X            99.41      99.46      99.50      99.54      99.58      99.62
## y_train      94.70      94.73      94.81      94.83      94.83      94.86
##          220 comps  221 comps  222 comps  223 comps  224 comps  225 comps
## X            99.65      99.69      99.72      99.75      99.78      99.81
## y_train      94.93      94.93      94.95      94.96      94.97      94.99
##          226 comps  227 comps  228 comps  229 comps  230 comps  231 comps
## X            99.84      99.86      99.88      99.90      99.92      99.94
## y_train      95.01      95.02      95.03      95.09      95.17      95.22
##          232 comps  233 comps  234 comps  235 comps  236 comps  237 comps
## X            99.95      99.96      99.97      99.98      99.98      99.99
## y_train      95.22      95.22      95.22      95.23      95.25      95.25
##          238 comps  239 comps  240 comps  241 comps  242 comps  243 comps
## X           100.00     100.00     100.00     100.00      100.0      100.0
## y_train      95.26      95.27      95.27      95.28       95.3       95.3
##          244 comps  245 comps  246 comps  247 comps  248 comps  249 comps
## X            100.0     100.00     100.00     100.00     100.00     100.00
## y_train       95.3      95.31      95.31      95.33      95.33      95.33
##          250 comps  251 comps  252 comps  253 comps  254 comps  255 comps
## X           100.00     100.00     100.00     100.00     100.00     100.00
## y_train      95.33      95.33      95.33      95.34      95.34      95.34
##          256 comps  257 comps  258 comps  259 comps  260 comps  261 comps
## X           100.00     100.00     100.00     100.00     100.00     100.00
## y_train      95.36      95.36      95.36      95.37      95.38      95.39
##          262 comps  263 comps  264 comps  265 comps  266 comps  267 comps
## X           100.00     100.00     100.00     100.00     100.00     100.00
## y_train      95.41      95.41      95.41      95.43      95.43      95.44
##          268 comps  269 comps  270 comps  271 comps  272 comps  273 comps
## X           100.00     100.00     100.00     100.00     100.00     100.00
## y_train      95.45      95.45      95.45      95.45      95.46      95.47
##          274 comps  275 comps  276 comps  277 comps  278 comps  279 comps
## X           100.00     100.00     100.00     100.00     100.00     100.00
## y_train      95.47      95.47      95.47      95.48      95.49      95.49
##          280 comps  281 comps  282 comps  283 comps  284 comps  285 comps
## X           100.00     100.00     100.00     100.00      100.0      100.0
## y_train      95.49      95.49      95.49      95.49       95.5       95.5
##          286 comps  287 comps  288 comps  289 comps  290 comps  291 comps
## X            100.0      100.0      100.0     100.00     100.00     100.00
## y_train       95.5       95.5       95.5      95.51      95.51      95.51
##          292 comps  293 comps  294 comps  295 comps  296 comps  297 comps
## X           100.00     100.00     100.00     100.00     100.00     100.00
## y_train      95.51      95.51      95.52      95.54      95.54      95.54
##          298 comps  299 comps  300 comps
## X           100.00     100.00     100.00
## y_train      95.54      95.54      95.54
```

From the cumulative variability we note 90% has been achieved for 68 components. For some components the values 1 are so rare the variability is very high divering to infinity.

So we have taken 70 components to predict

Now we fit the observations to train data and obtain the RMSE as below:

```
## [1] 0.1242826
```

So the RMSE for PCR for training dataset is 0.1243

Our regression model is now trained on our pca data

Test Set

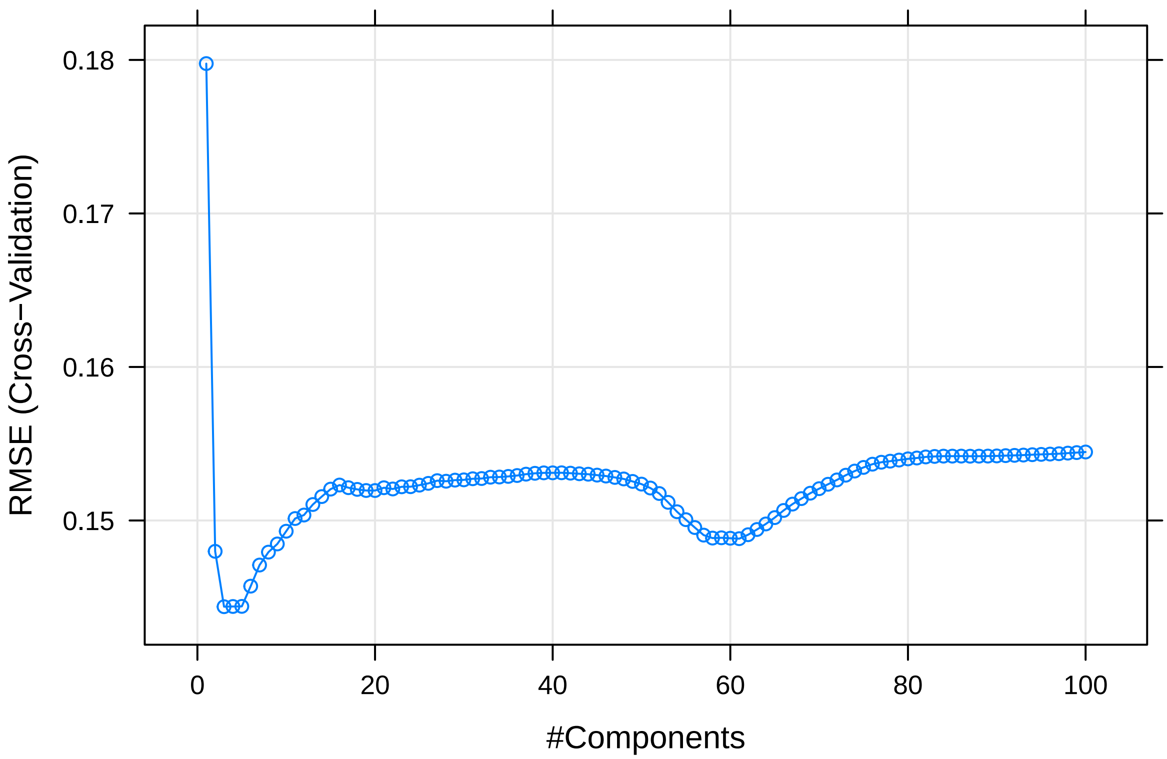
we need to do the same procesure as above that is use standard scaler on test data then use PCA with n_components 45 then use regression

```
## [1] 0.1502027
```

RMSE for unseen the test set is 0.1502.

Partial least square regression

checking of n_components for PLS




```
##   ncomp
## 3     3
```

```
## Data:    X dimension: 1022 300
##  Y dimension: 1022 1
## Fit method: oscorespls
## Number of components considered: 3
## TRAINING: % variance explained
##           1 comps  2 comps  3 comps
```

```
## X           6.477    8.695    10.10
## .outcome   80.352   89.202    92.51
```

```
##        RMSE   Rsquare
## 1 0.1086276 0.9250881
```

RMSE with partial least squares regression on training set is 0.0596

Using PLS on test set (unseen data)

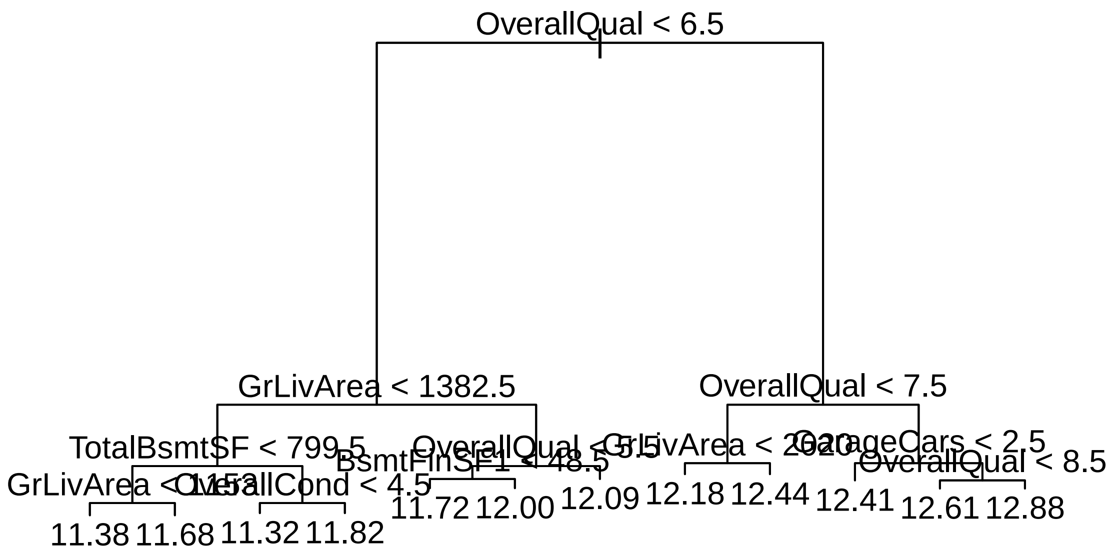
```
##        RMSE   Rsquare
## 1 0.1405641 0.8822649
```

RMSE on test data set is 0.1406

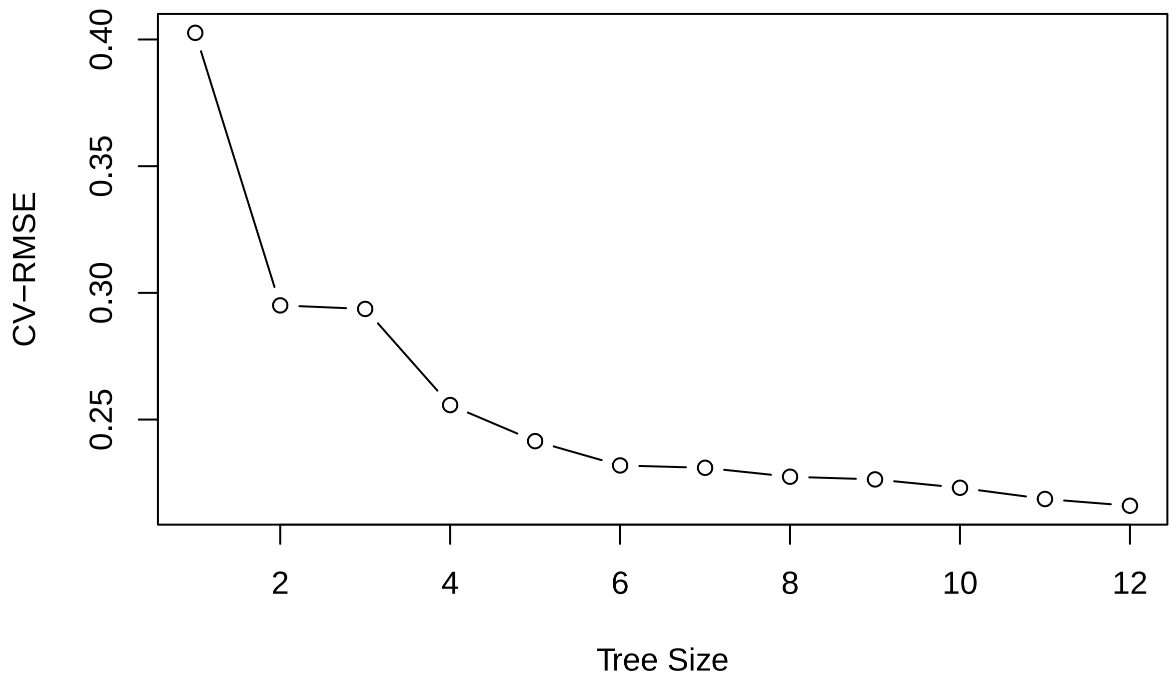
Trees, Bagging, and Random Forests

Finally we are doing to try and do trees, bagging, and random forests. To begin we fit a normal tree without any editing. From this it returned 6 significant variables out of the 80 we are testing. These included: OverallQual, GrLivArea, TotalBsmtSF, OverallCond, BsmtFinSF1, and GarageCars yielding 12 terminal nodes and a root mean squared error of .2135 (not the best). After this we looked to prune the tree to see what the best utilization of terminal nodes. After pruning we graphed the RMSE to terminal nodes to discover that anything above 6 would yield similar results. We noticed that throughout all of our terminal node sizes nothing we changed it to would change our RMSE significantly enough to care. Next we tried our luck at bagging.

```
##
## Regression tree:
## tree(formula = SalePrice ~ ., data = training)
## Variables actually used in tree construction:
## [1] "OverallQual" "GrLivArea"   "TotalBsmtSF" "OverallCond" "BsmtFinSF1"
## [6] "GarageCars"
## Number of terminal nodes:  12
## Residual mean deviance:  0.03784 = 38.18 / 1009
## Distribution of residuals:
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -1.082000 -0.105500 -0.001869  0.000000  0.120000  0.652300
```



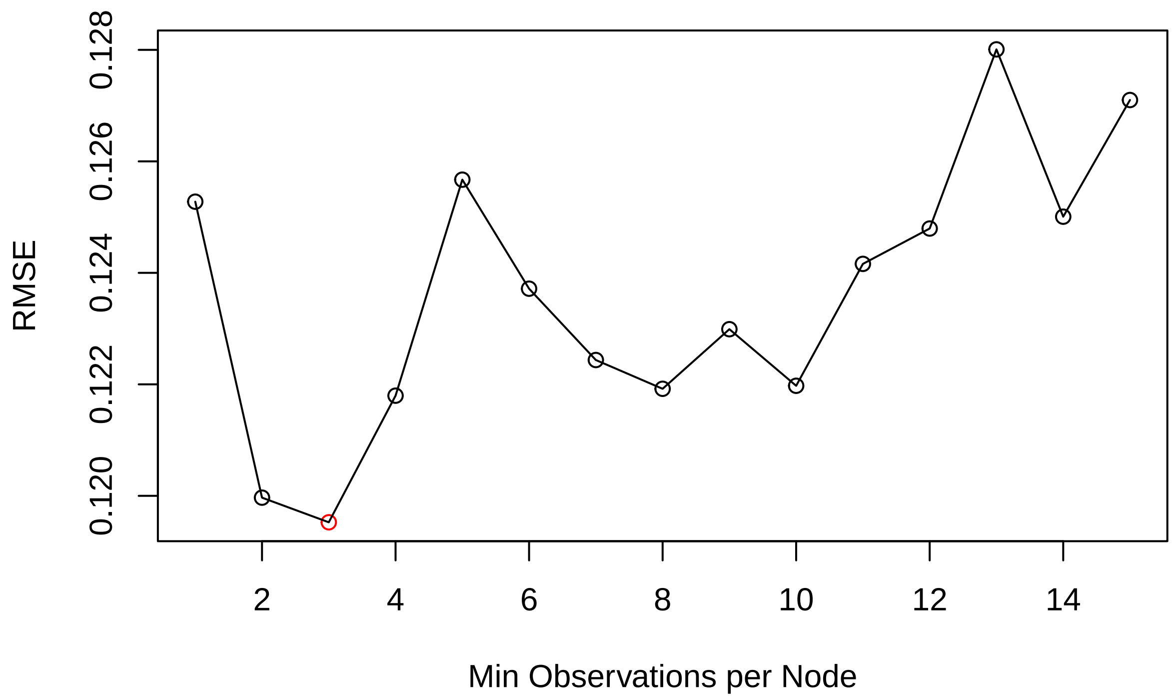

```
## [1] "Test RMSE of Tree:  0.2135"
```



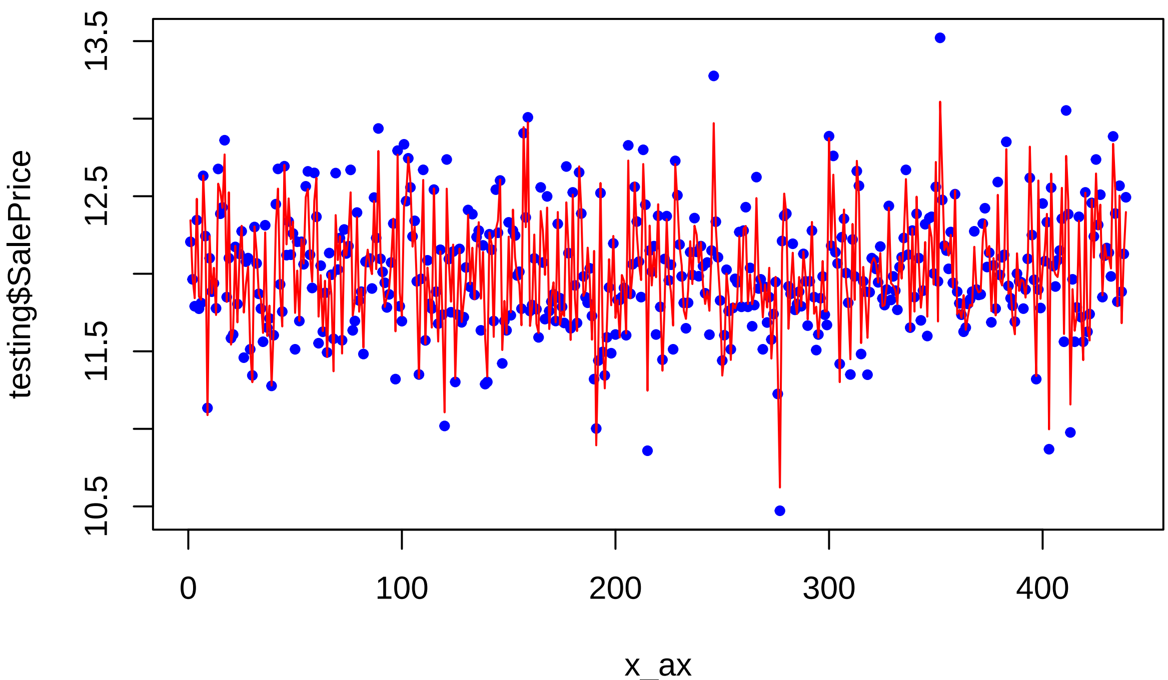
Before we start bagging we need to make sure to change all of our character values to factors. This is because our gbm model stand for gaussian bagging model and it doesn't take characters. In bagging we do it a little bit different than trees. We are not going to start out with a very basic model and prune it, we are instead going to use some of the information we learned from the previous trees. For example, our value for cv.folds in our gaussian bagging model holds the value of 7. This was chosen based off our CV.RMSE values in our graph above as 7 was around when the value of tree sizes begun to fall off and not change too significantly. The value shrinkage we left at the basic value of .1 as we didn't want to unnecessarily use too many trees. We set our trees at a large value of 1460 in order for the bagging model to use as many trees as necessary (most of the time it sets around 400-1000 before not reaching the shrinkage parameter). Finally, the last variable we used in our model was 'n.minobsinnode'; this was used as a constraint on the minimum number of observations in the terminal nodes of the tree. In order to find the best model we looped through n.minobsinnode being 1 through 15 and reported the best RMSE value. Below we will return three different graph. The first being a graph depicting the RMSE values of each bagging model. From this graph we can see that a value of 5 for our n.minobsinnode would return the lowest RMSE. Knowing this, we redid the model with this value in place and graphed the actual values of SalePrice (depicted in blue) and where are model went through predicting.

```
## Distribution not specified, assuming gaussian ...
## Distribution not specified, assuming gaussian ...
## Distribution not specified, assuming gaussian ...
## Distribution not specified, assuming gaussian ...
## Distribution not specified, assuming gaussian ...
## Distribution not specified, assuming gaussian ...
## Distribution not specified, assuming gaussian ...
## Distribution not specified, assuming gaussian ...
## Distribution not specified, assuming gaussian ...
```

```
## Distribution not specified, assuming gaussian ...
## Distribution not specified, assuming gaussian ...
## Distribution not specified, assuming gaussian ...
## Distribution not specified, assuming gaussian ...
## Distribution not specified, assuming gaussian ...
## Distribution not specified, assuming gaussian ...
```




```
## Distribution not specified, assuming gaussian ...
```



Just like with our trees we started with the most basic model we could in random forests, although we did add a few more parameters. One including the mtry, defined as the number of variables sampled as candidates at each split. We chose this to be the number of our columns divided by 3 because as stated in The Elements of Statistical Learning: Data Mining, Inference, and Prediction written by Trevor Hastie, Robert Tibshirani, Jerome Friedman, the text stated, "For regression, the default value for m is [p/3] and the minimum node size is five" (Haste 610). In doing our forest model we got a RMSE value of .14, considering this isn't as small as our bagging was and the only reason to do forests over bagging is because of overfitting we can conclude that our bagging from before holds our best model.

```
## [1] "Test RMSE for Random Forest 0.157027677975789"
```

After fitting our model to the proper testData provided by kaggle, we built our csv file and submitted it to the competition. Based off of our training and testing above we were predicting placing around 100th place with a RMSE of 0.1195259. Because the dataset was different than what we were testing on it could have been anything, but we were hopeful. After submission we ended with a RMSE value of .135 placing around 1000th place.

Conclusion

```
## Distribution not specified, assuming gaussian ...
```

```
## Using 1364 trees...
```

Methods	RMSE	RMSE_Dollars
Linear Regression	0.1400	1.15
PCR	0.0000	1.00
PLS	0.0000	1.00
Splines	0.4253	1.53
GAM	0.1227	1.13
Trees	0.2135	1.24
Random Forest	0.1570	1.17

Conclusion

After running all of our models we thought bagging would have proven to be our best bet for placement in the competition. It yielded our lowest rmse value using our training and testing data at 0.1195259. However, that proved to be misleading as ...

Github

All of our work can be found at: https://github.com/trevorisaacson3/KaggleHomePrices

Sources

Hastie, Trevor, et al. The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer, 2017.