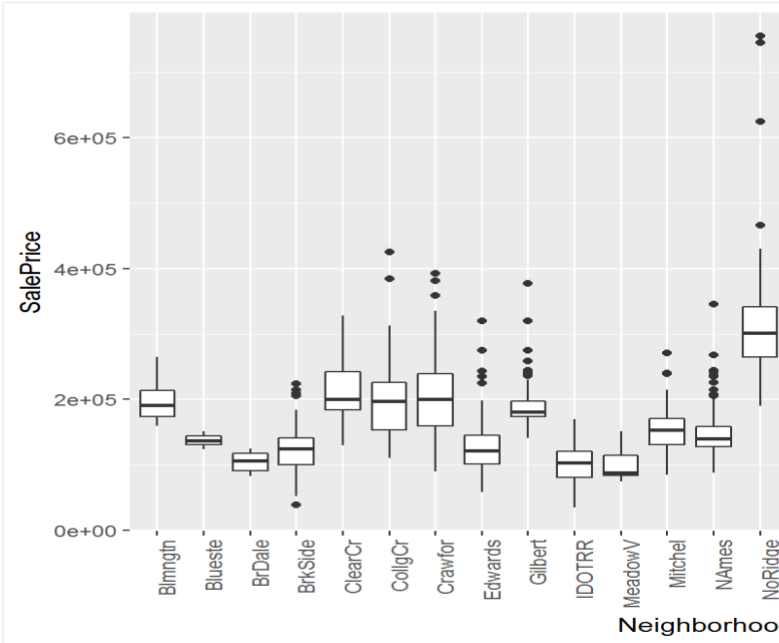
Housing Price Prediction

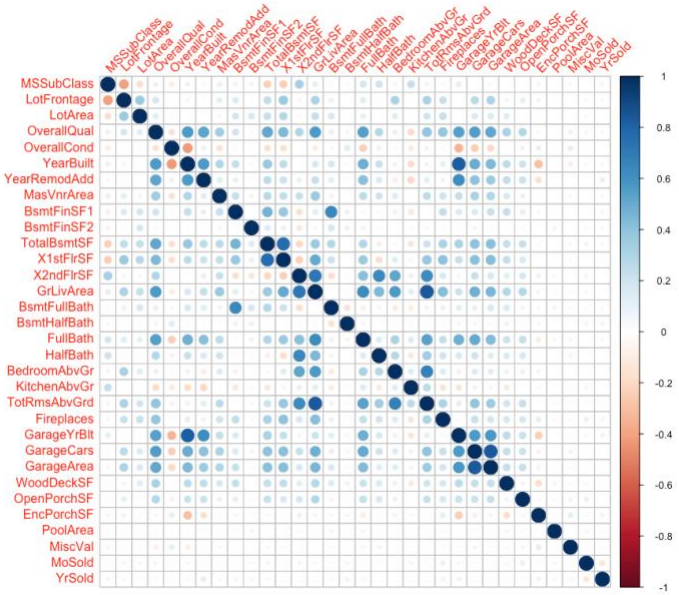
Data Description:

The Train data consists of 1000 records and 76 variables whereas the test data consists of 179 records and 75 variables with the target variable missing.

Plot(1): This Histogram Indicates that House Sales are Increased Gradually from 2006 to 2009 and there is a sudden decrease in 2010 and higher number of homes are sold in 2010 year

Plot(2): boxplots of neighborhood against SalePrice. These helps to confirm our understanding of what may affect a home's sale prices. The distribution of prices make sense here as there should be varying prices with certain neighborhoods clearly valued higher





The correlation matrix shows some variables that are highly correlated such as GrLivArea and TotRmsAbvGrd and GarageYrBlt

and YearBuilt. These variables were not be removed and were kept in the dataset as some of the models are fairly robust and correlated variables will not hinder predictive performance.

Data Pre-Processing:

- The data preprocessing step is most crucial in this project. Prediction on sale price depends on the Data Processing Stage. We combined the train and test datasets to keep the level of factors equal between the two sets so problems were not run into when predicting with unseen factor levels.
- We tried to handle missing values using mice package i.e missing values were imputed using the mice package but we found that it is not so effective so we created our own function to handle NA values i.e we replaced all factor NA's with none because there is no value associated with it and it reflects the performance in prediction and added median to the numerical values as which is best suitable in the given dataset.
- All numeric variables that had a skewness of over 0.75 were transformed using the log function. The categorical variables were one hot encoded into number and each of variable factor level owns the value accordingly.
- We removed unused and un-important variables like X, Id. The missing values indicate that majority of the houses do not have alley access, no pool, no fence and no elevator, 2nd garage and MiscFeatures
- Log transformation has been applied to sale price of home as the distribution is skewed.

```
Ex: Handling SKewness
#determining skew of each numric variable
skew <- sapply(numeric_columns,function(x){
skewness(features[[x]],na.rm = T)})
# Let us determine a threshold skewness and transform all variables above the treshold.
skew <- skew[skew > 0.75]
# transform excessively skewed features with log(x + 1)
for(x in names(skew)) {
  features[[x]] <- log(features[[x]] + 1)
}
```

 # OLS(Ordinary Least Squares:

- Linear Regression technique is used to or the analysis and modelling of linear relationships and predict the house prizes for the given dataset.
- We split the data into two categories namely training and validation. We constructed 900 records to build the model and 100 records to predict the sale prizes based on the model.
- Log transformation applied to the sales prices to handle skewness and we have got good adjusted R-Square(0.94) i.e 94% of the variance is explained by this model. Below are the R-Square, AIC, BIC values.

```
AIC(fit) # -624.6101
BIC(fit) # 186.9947
```

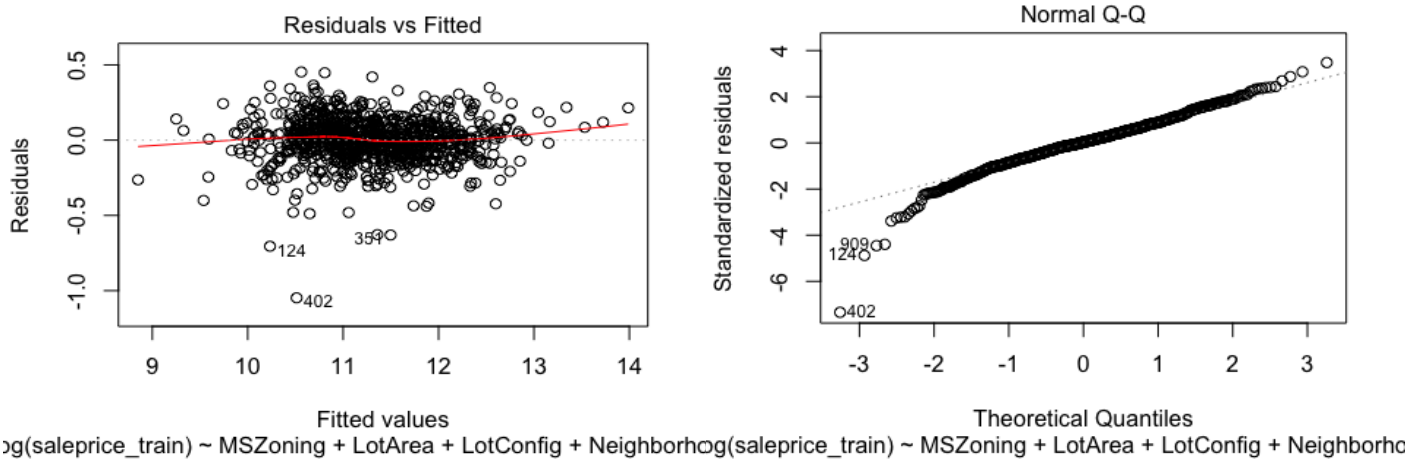
```
## Residual standard error: 0.1572 on 732 degrees of freedom
## Multiple R-squared:  0.9576, Adjusted R-squared:  0.948
## F-statistic: 99.06 on 167 and 732 DF,  p-value: < 2.2e-16
```

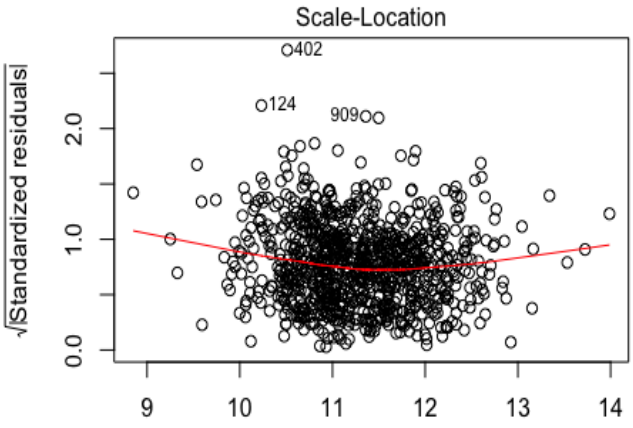
- All are not passed Hypothessis test. Few of the values have >0.05 so, this model is not good. There is potential overfitting could occur if someone insist on using it. The variable selection process should be involved in model construction. I prefer to use stepAIC method.
- Stepwise regression with both the direction is good for this dataset as it has good R-Squre and and all the variables passed the Hypothesis Test. The diagnosis of residuals is also better this gives r- square value as `0.9492` for best fit

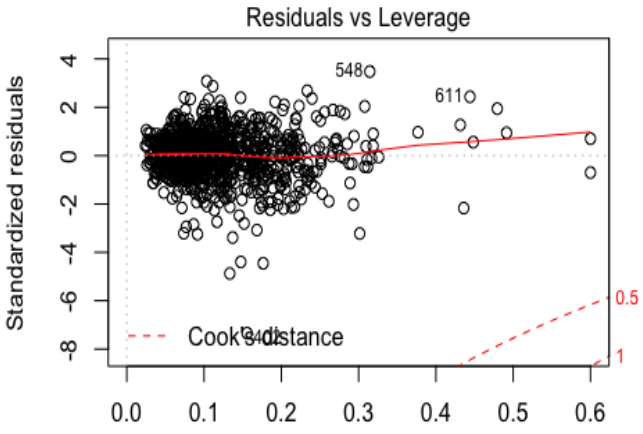
```
## Residual standard error: 0.1553 on 786 degrees of freedom
## Multiple R-squared:  0.9556, Adjusted R-squared:  0.9492
## F-statistic: 149.7 on 113 and 786 DF,  p-value: < 2.2e-16

AIC(fit2) #-690.3082
BIC(fit2) # -138.0328
vif(fit2)

Analysis of the Residuals:
```

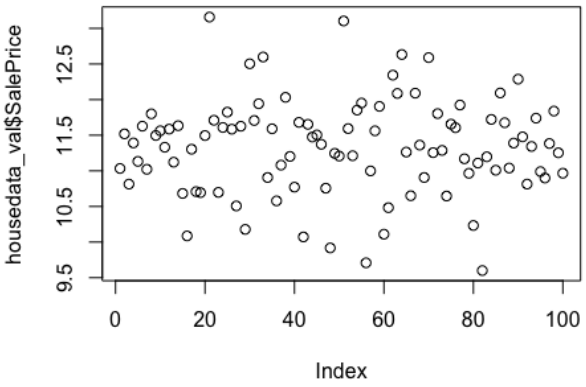
- AIC, BIC values are improved in stepAIC method and on top of that all values passed null hypothesis.
- The First plot clears shows the residuals of our models are spread equally on the horigontal line which indicated our model does not have non-linear relationship.
- The QQ Plot indicates residuals are clearly normally distributed as they have all the values held on the dashed line.
- The Standardized and fitted values plot shows few of the variables are not transformed but many of them are falls I straight line
- Standardized residuals vs legerage indicates that no value appear outside the cooks distance.
- Below is the RMSE for the StepAIC method

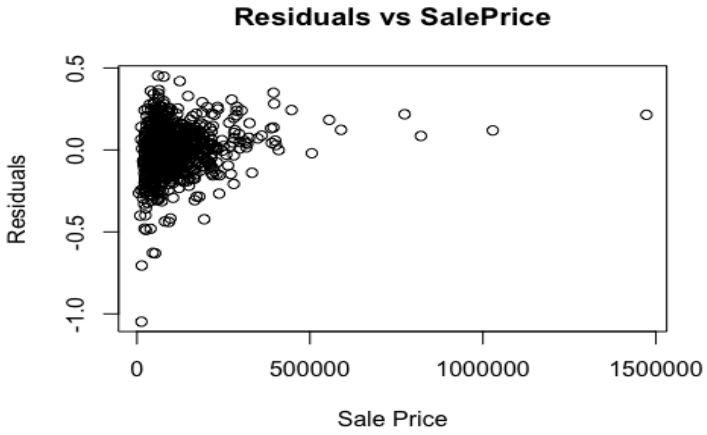
```
RSS <- c(crossprod(fit2$residuals))
MSE <- RSS / length(fit2$residuals) #Mean squared error:
RMSE <- sqrt(MSE) #0.1451161        #Root Mean Squared Error
```





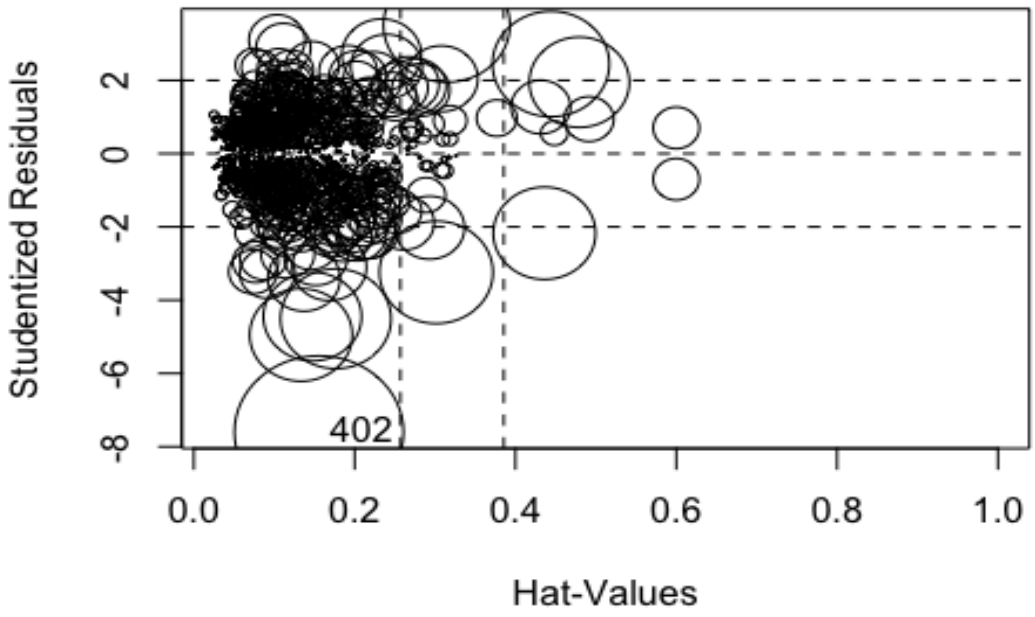






- The plot the residual against the sale price clearly shows most of the sale price predicted.
- Influence plot studentized Residuals vs Hat Values outlyingness, leverage, and influence of each case. The plot shows the residual on the vertical axis, Hat Values on the horizontal axis, and the point size is the square root of Cook's D statistic, a measure of the influence of the point. Some of the most interesting cases may be outliers.

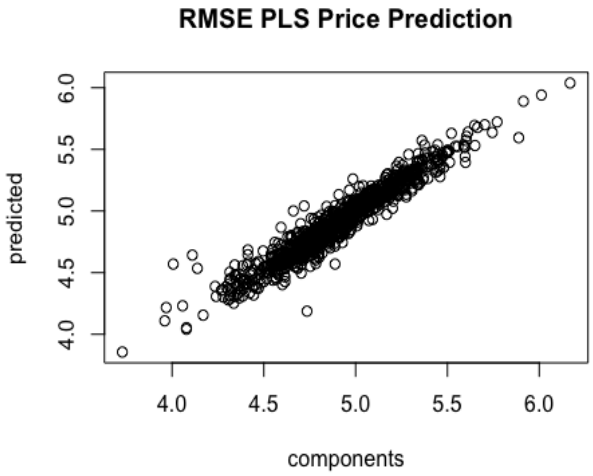
```
##      StudRes       Hat      CookD
## 402 -7.60286 0.1556429 0.08716631
```

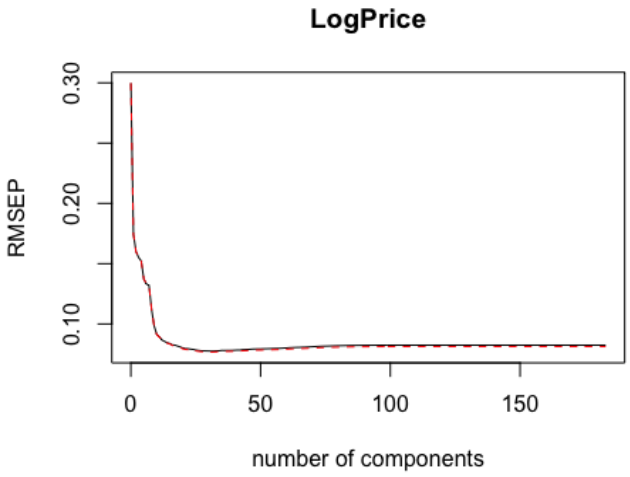



```
#-------------------------------------------------------
#PLS
housedata_pls <- features[1:1000,]
housedata_test <- features[1001:1179,]
plsfit <- plsr(LogPrice~.,data=housedata_pls, validation = "CV")
plspred <- predict(plsfit,housedata_test,ncomp=1:2)
plspredsaleprice <- round(10.0**plspred)
plot(plsfit, main = "RMSE PLS Price Prediction", xlab="components")
```

PLS:

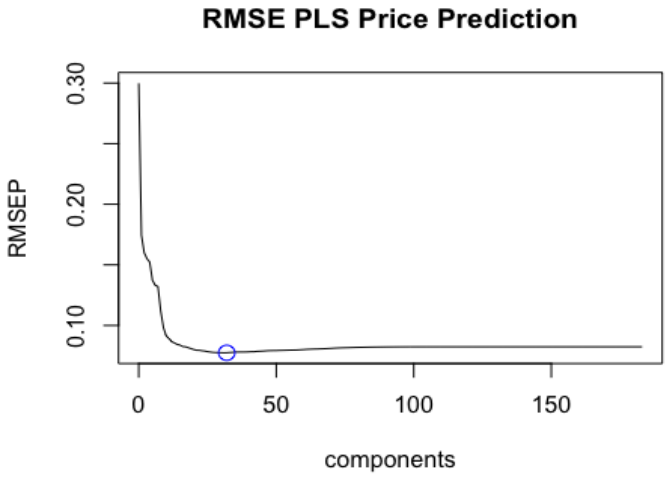
- After we secured very good RMSE values for the OLS model we have chosen the PLS as our next regression model. Partial Least squares regression is a supervised technique that is known to perform better than “OLS” and “PCR” Regression techniques in most cases.
- This method finds components that explain high variability as well as high correlation with the response variable i.e. Price in our case.
- To avoid overfitting the model to the training dataset, we used in-built cross-validation features.





- This RMSE PLS Price Prediction clearly normally distributed as they have all the values held on the same line diagonal
- We fitted the data in the basic "PLS" model. When we looked at the summary of the fit function we found that max variance can be explained around 30 components and then it stabilizes.
- The Validation plot to check the validity of our assumptions and it represents the same.
- We did Hyper Parameter Turning with the PLS Model to understand which component is fitting better. To find the components at which we obtain minimum "RMSE" Value, we used the "RMSEP" plot and found the minimum "RMSEP" value of 30 components from the Plot results.
- We used this value to plot the fit and we obtained “Actual values” VS “Predicted Values”. We Obtained a very good fit with all the points following the diagonal line. We fitted the tuned model with the test data and obtained our final predictions after the hyper parameter tuning improved a lot and we got very good score.

```
#Hypertuning:

pls.RMSEP <- RMSEP(plsfit,estimate = "CV")
plot(pls.RMSEP, main = "RMSE PLS Price Prediction", xlab="components")
min <- which.min(pls.RMSEP$val)
points(min,min(pls.RMSEP$val),pch=1,col="blue",cex=1.5)
min
## [1] 30
plot(plsfit,ncomp=30,asp=1,line=TRUE)
```

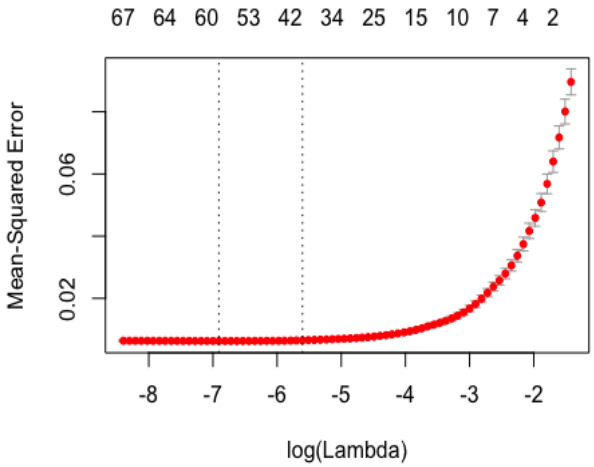


- The above plot the best prediction is at 30 components.
- predicted vs measured plot is also improved with the latest prediction sale price.
- I can conclude that the best prediction with the sale prize for this dataset can be improved with Only hyper parameter tuning**. RMSE Score for this prediction is 20833** in the leaderboard.

We used the default CV. The default **Cross validation value is 5 fold.**

```
#------------------------------------------------------------
#LASSO
```

- The regression model we used here is LASSO (Least Absolute Shrinkage and Selection Operator) model. Variable selection and regularization will be done by LASSO model.
- We observed that in our housing data we have lot of variables and in order to find the variables that matter most we used **5-fold repeated cross validation from** . We reduced the dimensionality as a result.



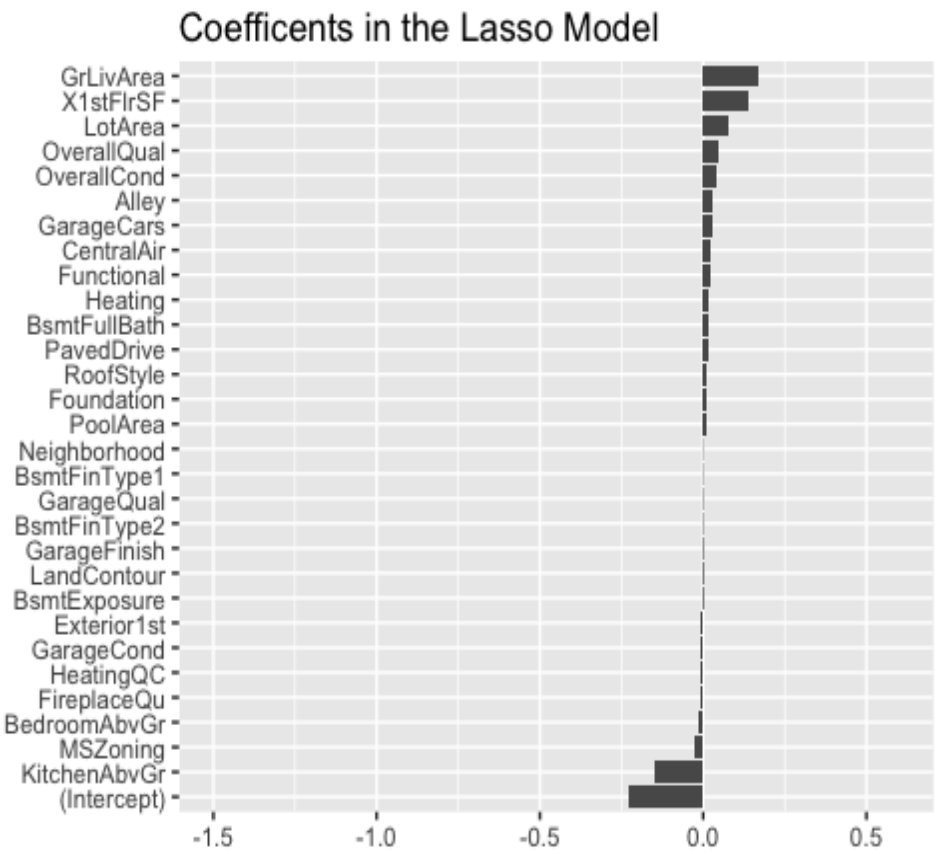
The Mean-Squard Error and Lambda plot states, It includes the cross-validation curve (red dotted line), and upper and lower standard deviation curves along the λλ sequence (error bars). Two selected λλ's are indicated by the vertical dotted lines.

bestlam <- cvLasso$lambda.min… ## [1] 0.001000085
lambda.min is the value of λλ that gives minimum mean cross-validated error.

- The main tuning parameter for the lasso model is alpha - a regularization parameter that measures how flexible our model is. The higher the regularization the less prone our model will be to overfit. However, it will also lose flexibility and might not capture all of the signal in the data.
- One thing to note here however is that the features selected are not necessarily the "correct" ones especially since there are few collinear features in this dataset. One idea to try here is run Lasso a few times on bootstrapped samples and see how stable the feature selection is.
- Here we used our own Grid, repeated cross validation method, used caret package and we are using fivefold cross validation.
- lasso performs even better so we'll just use this one to predict on the test set.
- Lasso model shrinks the coefficients of the regressors to zero that has less importance in the model prediction.
- Lasso picked 62 variables and eliminated the other 9 variables.
- We plotted the Coefficients in the Lasso Model with few selected variables that lasso picked and we noticed that GrLivArea, X1stFlrSf, LotArea predictors contribute to the model.
- This definitely sense. Then a few other location and quality features contributed positively. Some of the negative features make less sense and would be worth looking into more - it seems like they might come from unbalanced categorical variables.

Hyper parameter tuning, best alpha and lambda value:

```
##   alpha  lambda
## 3     1 0.00075
```

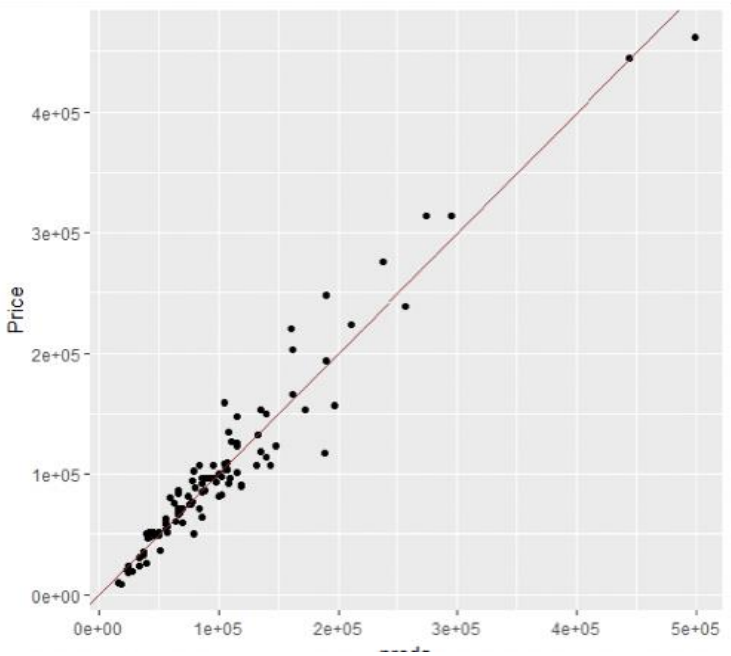


variables with non-zero coefficients

	coef.name	coef.value
1	GrLivArea	0.168779909
2	X1stFlrSF	0.138353208
3	LotArea	0.074488704
4	OverallQual	0.047897029
5	OverallCond	0.038866684
6	Alley	0.029828116
7	GarageCars	0.026252518
8	CentralAir	0.024848636
9	Functional	0.021053519
10	Heating	0.019215945
11	BsmtFullBath	0.017649549
12	PavedDrive	0.013353462
13	RoofStyle	0.011622881
14	Foundation	0.009763865
15	PoolArea	0.009514362

- This plot gives the predicted vs Actual Prices. We can clearly see the best fit in the plot

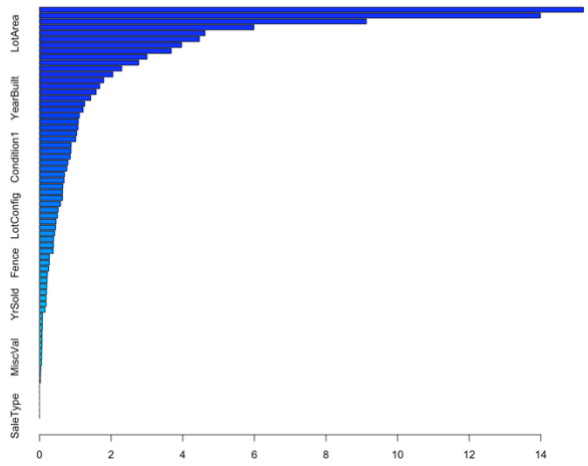
and we obtained a good fit with all the points close to the diagonal line.



We got a RMSE estimate value of 23819.99 for the LASSO model

```
-----------------------------------
#Gradient boosting method
```

- The next machine learning model is we used id **GBM** Gradient boosting models are one of the most popular model.
- Gradient boosting majorly involves three elements namely A loss function to be optimized, A weak learner to make predictions, An additive model to add weak learners to minimize the loss function.
- The Variable Importance plot:
- The summary of the Model gives a feature importance plot. The Summary list is on the top is the most important variable and at last is the least important variable.
  And the 2 most important features are LotArea, X1stFlrSF and SaleType and MiscVal are Least important features

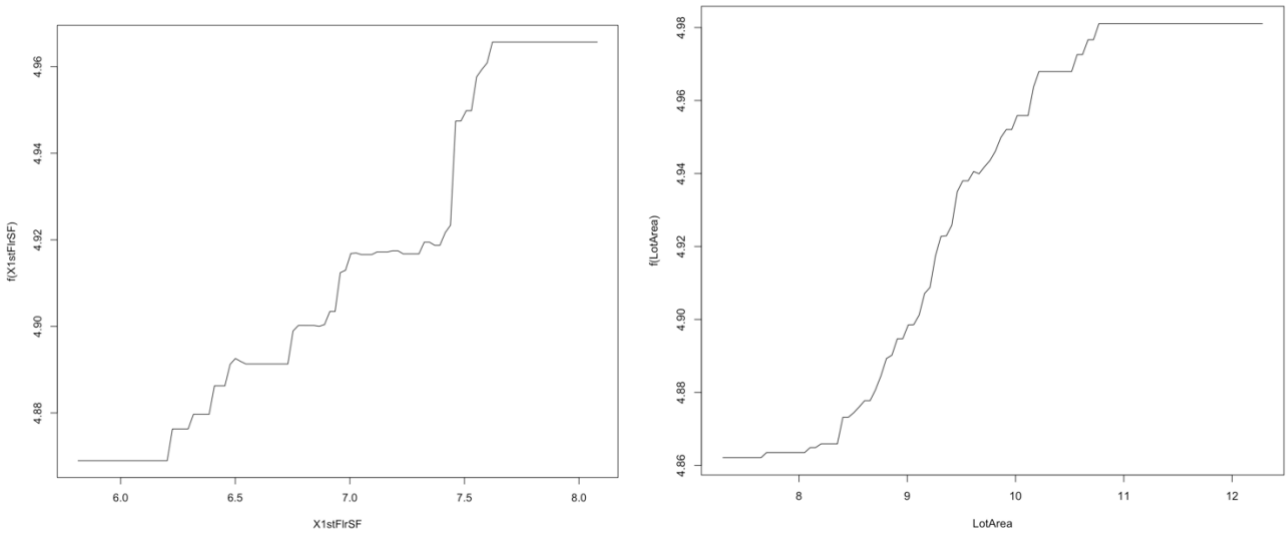


var	rel.inf
Neighborhood	14.15867354
GrLivArea	12.85304809
OverallQual ...	8.49551493
TotalBsmtSF ...	4.89080184
GarageArea	4.73696293
LotArea	4.68334897
PoolArea	0.00000000
PoolQC	0.00000000
MiscFeature	0.00000000

Plotting the Partial Dependence Plot

- The partial Dependence Plots will tell us the relationship and dependence of the variables with the Response variable.
- The LotArea plot clearly stating as the Lot area increases the sales price is also increases.
- The X1stFlrSF response variable plot clearly stating the more square feet have more sale price.

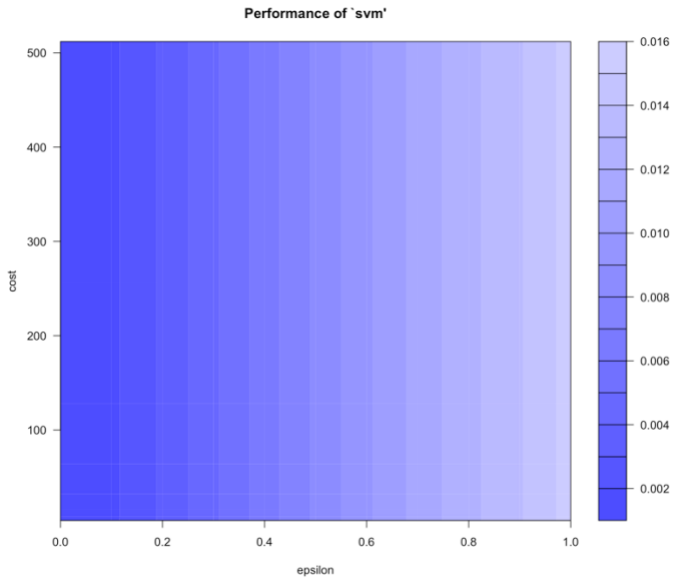
We got RMSE Score **21294.72** in Kaggle.

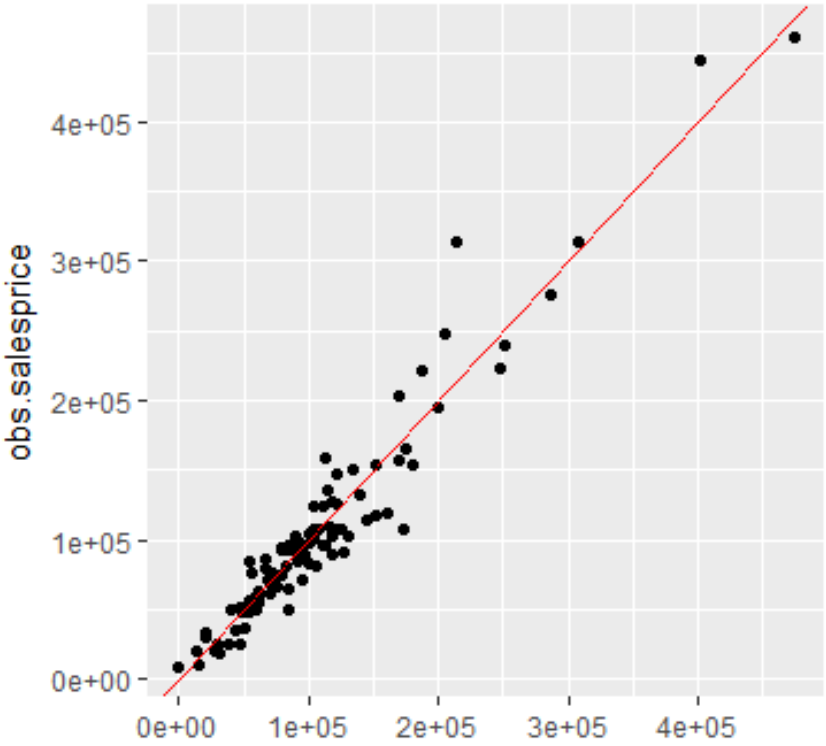



```
#-----------------------------------------------------------------
#SVM
```

- The next machine learnining model we are interested in support vector machine. This model is supervised learning algorithm that analyze data used for classification and regression analysis.
- This model project the low dimensional data into high dimentional data and generates multiple hyper planes such that the data space is divided into segments and each segment contains only one kind of data.

Parameters:
SVM-Type: eps-regression
SVM-Kernel: radial
cost: 3
gamma: 0.005434783
epsilon: 0.1
Number of Support Vectors: 598
Parameter Tuning SVM Regression Model:
plot(tuneResult)





Parameter tuning of 'svm':
- sampling method: 10-fold cross validation
- best parameters:
 epsilon cost
 0 16
- best performance: 0.001141392

- On this graph, we can see that the darker the region is the better our model is (because the RMSE is closer to zero in darker regions). we can narrow down our search for cost and gamma values to and try further tuning if required.
- The plot is showing deviation of actual SalePrice to the Predicted price for all Housing data covered in the validation.

Conclusion & Comments on RMSE Score for models:

- After doing Hyper parameter tuning for PLS we have got RMSE score 20894.50181
- For Lasso Model we have got the RMSE Score 23819.99038
- For Gradient Boosting Method the RMSE score 21294.72658
- For SVM the RMSE score 25125.58653

The Best Model that we have got from all above is PLS after doing hyper parameter tuning with the components 33.