Assignment #5: Automated Variable Selection, Multicollinearity, and Predictive Modeling

Prabhat Thakur

Data: The data for this assignment is the Ames, Iowa housing data set (ames_housing_data.csv). This data is posted in Canvas.

Introduction:

The goal of this assignment is to extend regression model building process from the previous three assignments. We will set up a predictive modeling framework, explore the use of automated variable selection techniques for model identification, assess the predictive accuracy of our model using cross-validation, and compare and contrast the difference between a statistical model validation and an application (or business) model validation. At the end we will choose the best model.

Assignment Tasks:

(1) Define the Sample Population

The data set used for this assignment is familiar Ames Housing Data. For defining sample population, I will use the EDA analysis performed in Assignment 1 and 2.

I started with the complete set of 2930 observations and applied below drop conditions to remove observations which are way different from majority. I decided to use same drop conditions as used in Assignment #2 to create sample population for this assignment with slight changes to not delete too many observations.

Below is the summary of my waterfall drop conditions:

Waterfall Drop Condition Steps	Records dropped	Total Remaining
0: Original Data Set	0	2930
1: SalePrice > $500000	17	2913
2: GrLivArea > 4000	3	2910
3: Zoning != A or C or I	29	2881
4: TotalBsmtSF > 3000	3	2878
5: LotArea > 50000	10	2868

The remaining sample has 2868 total observations which is approximately 97.88% of the original population. This sample data set will be used in the following sections for model building.

(2) The Predictive Modeling Framework

Before I start define train and test data for model building and validation respectively, based on the EAD analysis, it appeared that creating new variables from existing variables and lumping of categories will be a good idea. This will also help in reducing number of predictors in the model.

New Variables:

I created a few new variables in the sample dataset derived from existing variables. Below is the brief summary of new variables:

1. **TotalSqFtCalc** which is a sum of GrLivArea + BsmtFinSF1 + BsmtFinSF2 and represent the total finished area in square feet.
2. **HouseAge** which is YrSold - YrBuilt and represents the age of the house in years.
3. **QualityIndex** which is OverallQual * OverallCond and represents quality of the property
4. **TotalBathRooms** which is FullBath + BsmtFullBath + 0.5*(BsmtHalfBath + BsmtHalfBath) and represents total number of bathrooms in house.
5. **TotalPorchSF** which is screenPorch + OpenPorchSF + WoodDeckSF in square feet and represents total porch area.
6. **price_sqft** which is given by SalePrice/TotalSqFtCalc (new variable) and represents price /sqft for finished area in $/sqft.

In addition to the numeric variables above I created two new categorical variables by lumping the categories.

7. BldgTypeGrp, regrouped BldgType into three groups, Single Fam, Townhomes and 2Fam/Duplex
8. NbhdGrp, regrouped Neighborhoods into four groups based on average sale price of neighborhoods.

The selection of these categorical variables was based on my current and past EDA analysis of categorical variables available in the dataset.

Train & Test Dataset

We will 'train' each model by estimating the models on the 70% of the data identified as the training data set, and we will 'test' each model by examining the predictive accuracy on the 30% of the data. After including all the new variables in my sample data set, I created a 70/30 train/test set. The total observation count is 2868. The split for the observations contained in each is given below.

Table of observation counts for my train/test data partition from my sample population.

	# of Train Observation	**# of Test Observation**	**Total # of Observation**
70/30 split	2009	859	2868
After dropping NA rows	2007	859	2866

(3) Model Identification by Automated Variable Selection

For this section, based on earlier EDA I selected some good predictors which includes the variables I created along with the existing predictors. We want to include a mix of discrete and continuous variables in our model building. Following is the list of pool of candidate predictor variables I will be using, it includes response variable 'SalePrice' and mix of discrete, categorical and continuous variables.

Pool of candidate predictor variables: "LotArea","TotalBsmtSF","KitchenQual","TotRmsAbvGrd", "GarageArea", "HouseAge", "price_sqft", "QualityIndex", "TotalSqFtCalc", "TotalBathRooms", "TotalPorchSF", "BldgTypeGrp", "NbhdGrp".

Since our focus is on selecting best predictor and build a predictive model, I did not create dummy variables from categorical variables. I will let R do when fitting the models. Since I am using derived variables and categorical variables as is, my list is 13 predictor variables. I created **train.clean** and **test.clean** data sets which contains response variable and above predictor variables.

Model Identification: To begin with, I simply created a model using all the predictor in train.clean dataset. Below is the summary statistics for the same. **I will call it Model1** (full model).

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

Residuals:
   Min     1Q Median     3Q    Max
-82654  -4806    414   5974  78879

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)     -1.223e+05  4.278e+03 -28.578  < 2e-16 ***
LotArea         -1.894e-01  9.836e-02  -1.926 0.054252 .
TotalBsmtSF      5.555e-01  1.156e+00   0.481 0.630869
KitchenQualFa   -2.713e+04  2.837e+03  -9.563  < 2e-16 ***
KitchenQualGd   -2.721e+04  1.535e+03 -17.726  < 2e-16 ***
KitchenQualPo   -1.823e+04  1.486e+04  -1.227 0.220101
KitchenQualTA   -2.744e+04  1.765e+03 -15.552  < 2e-16 ***
TotRmsAbvGrd     7.449e+02  2.808e+02   2.653 0.008052 **
GarageArea       7.861e+00  2.155e+00   3.648 0.000271 ***
HouseAge        -9.938e+01  1.697e+01  -5.855 5.57e-09 ***
QualityIndex     1.799e+02  4.980e+01   3.612 0.000312 ***
TotalSqFtCalc    8.565e+01  1.160e+00  73.864  < 2e-16 ***
TotalBathRooms  -1.925e+03  6.365e+02  -3.025 0.002520 **
TotalPorchSF    -6.231e-01  2.504e+00  -0.249 0.803518
BldgTypeGrpgrp2 -2.799e+03  1.363e+03  -2.054 0.040103 *
BldgTypeGrpgrp3  1.562e+01  1.292e+03   0.012 0.990353
NbhdGrpgrp2      7.351e+03  1.076e+03   6.832 1.11e-11 ***
NbhdGrpgrp3      9.717e+03  1.297e+03   7.491 1.02e-13 ***
NbhdGrpgrp4      5.814e+03  2.004e+03   2.902 0.003748 **
price_sqft       1.594e+03  3.648e+01  43.707  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 14670 on 1987 degrees of freedom
Multiple R-squared:  0.9606,    Adjusted R-squared:  0.9602
F-statistic:  2549 on 19 and 1987 DF,  p-value: < 2.2e-16
```

We can see there are few predictors which are insignificant. That means we don't need all the predictors, to find the 'best' models I am use automated variable selection techniques: forward, backward, and stepwise variable selection using the R function stepAIC() from the MASS library.

My approach was to start with the forward selection, backward and stepwise AIC process and verify resulting models with AIC, BIC and Adjusted R-squared values for each. Reviewing each step of the stepAIC automated results, I verified that AIC values obtained as each variable was added or removed made sense and that the resulting final model did indeed improve on the metrics over the 'unfinished' models obtained. By reviewing the AIC values for each predictor, we can be confident that the right optimal model was obtained without blindly accepting the result.

Following are the good models based on the forward, backward and stepwise tests.

From forward selection technique - Model2

model2 <- lm(formula = SalePrice ~ TotalSqFtCalc + price_sqft + KitchenQual + NbhdGrp + HouseAge + QualityIndex + GarageArea + TotalBathRooms + TotRmsAbvGrd, data = train.clean)

Below is the summary statistics for the same:

```
Call:
lm(formula = SalePrice ~ TotalSqFtCalc + price_sqft + KitchenQual +
    NbhdGrp + HouseAge + QualityIndex + GarageArea + TotalBathRooms +
    TotRmsAbvGrd, data = train.clean)

Residuals:
   Min     1Q Median     3Q    Max
-82662  -4715    451   5794  78930

Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)    -1.230e+05  4.156e+03 -29.592  < 2e-16 ***
TotalSqFtCalc   8.523e+01  9.567e-01  89.082  < 2e-16 ***
price_sqft      1.590e+03  3.435e+01  46.279  < 2e-16 ***
KitchenQualFa  -2.717e+04  2.828e+03  -9.608  < 2e-16 ***
KitchenQualGd  -2.732e+04  1.528e+03 -17.876  < 2e-16 ***
KitchenQualPo  -1.862e+04  1.487e+04  -1.253 0.210443
KitchenQualTA  -2.758e+04  1.750e+03 -15.759  < 2e-16 ***
NbhdGrpgrp2     7.246e+03  1.056e+03   6.860 9.18e-12 ***
NbhdGrpgrp3     9.545e+03  1.272e+03   7.506 9.12e-14 ***
NbhdGrpgrp4     5.623e+03  1.979e+03   2.841 0.004538 **
HouseAge       -9.946e+01  1.650e+01  -6.027 1.99e-09 ***
QualityIndex    1.903e+02  4.889e+01   3.892 0.000103 ***
GarageArea      7.695e+00  2.133e+00   3.607 0.000317 ***
TotalBathRooms -1.980e+03  6.233e+02  -3.176 0.001518 **
TotRmsAbvGrd    8.173e+02  2.616e+02   3.124 0.001810 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 14680 on 1992 degrees of freedom
Multiple R-squared:  0.9605,    Adjusted R-squared:  0.9602
F-statistic:  3457 on 14 and 1992 DF,  p-value: < 2.2e-16
```

This model provided a better AIC, BIC and Adjusted R-squared model over Model 1 (full model) .

From backward elimination technique - Model3

model3 <- lm(formula = SalePrice ~ LotArea + KitchenQual + TotRmsAbvGrd + GarageArea + HouseAge + QualityIndex + TotalSqFtCalc + TotalBathRooms + BldgTypeGrp + NbhdGrp + price_sqft, data = train.clean)

Below is the summary statistics for the same:

```
Call:
lm(formula = SalePrice ~ LotArea + KitchenQual + TotRmsAbvGrd +
    GarageArea + HouseAge + QualityIndex + TotalSqFtCalc + TotalBathRooms +
    BldgTypeGrp + NbhdGrp + price_sqft, data = train.clean)

Residuals:
   Min     1Q Median     3Q    Max
-82125  -4887     402   5893  78812

Coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)      -1.222e+05  4.252e+03 -28.748  < 2e-16 ***
LotArea          -1.860e-01  9.786e-02  -1.900 0.057517 .
KitchenQualFa    -2.714e+04  2.835e+03  -9.574  < 2e-16 ***
KitchenQualGd    -2.725e+04  1.532e+03 -17.787  < 2e-16 ***
KitchenQualPo    -1.825e+04  1.486e+04  -1.229 0.219304
KitchenQualTA    -2.748e+04  1.762e+03 -15.594  < 2e-16 ***
TotRmsAbvGrd      7.173e+02  2.739e+02   2.619 0.008887 **
GarageArea        7.908e+00  2.152e+00   3.674 0.000245 ***
HouseAge         -9.907e+01  1.694e+01  -5.849 5.77e-09 ***
QualityIndex      1.769e+02  4.947e+01   3.576 0.000357 ***
TotalSqFtCalc     8.585e+01  1.020e+00  84.127  < 2e-16 ***
TotalBathRooms   -1.924e+03  6.355e+02  -3.028 0.002492 **
BldgTypeGrpgrp2  -2.780e+03  1.362e+03  -2.042 0.041294 *
BldgTypeGrpgrp3   4.089e+01  1.286e+03   0.032 0.974631
NbhdGrpgrp2       7.284e+03  1.062e+03   6.855 9.45e-12 ***
NbhdGrpgrp3       9.640e+03  1.279e+03   7.536 7.32e-14 ***
NbhdGrpgrp4       5.745e+03  1.984e+03   2.895 0.003833 **
price_sqft        1.598e+03  3.476e+01  45.985  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 14660 on 1989 degrees of freedom
Multiple R-squared:  0.9606,    Adjusted R-squared:  0.9603
F-statistic:  2852 on 17 and 1989 DF,  p-value: < 2.2e-16
```

From stepwise selection technique

Interestingly the best model generated by stepwise selection technique is exactly same as forward selection technique.

I build another model by randomly selecting some predictors, a junk, or bogus model is also a good way to verify the selection process function is working as expected., I call it Model4.

```
Call:
lm(formula = SalePrice ~ OverallQual + OverallCond + QualityIndex +
    GrLivArea + TotalSqFtCalc, data = train.df)

Residuals:
    Min      1Q  Median      3Q     Max
-213804  -18367    -163   16432  148711

Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)   -1.752e+05  1.484e+04 -11.802  < 2e-16 ***
OverallQual    4.145e+04  2.519e+03  16.453  < 2e-16 ***
OverallCond    1.311e+04  2.747e+03   4.773 1.95e-06 ***
QualityIndex  -2.294e+03  4.768e+02  -4.812 1.60e-06 ***
GrLivArea      2.140e+01  2.408e+00   8.888  < 2e-16 ***
TotalSqFtCalc  3.743e+01  1.591e+00  23.517  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 31590 on 2003 degrees of freedom
Multiple R-squared:  0.816,     Adjusted R-squared:  0.8155
F-statistic:  1776 on 5 and 2003 DF,  p-value: < 2.2e-16
```

At this stage 1 have 4 different models, before I select a best model, we want to make sure that our models don't have highly correlated pairs of predictors that we do not like. I computed VIF statistics to validate this. VIF values for first 3 models are below 8 which is a good sign. These models doesn't have highly correlated pairs of predictors. However for the model4 (junk model) below are the VIF values

```
QualityIndex  OverallQual  OverallCond   GrLivArea TotalSqFtCalc
   36.122718    23.964958    18.410471    2.642212    2.476652
```

It is obvious that VIF value for QualityIndex, OverallQual and OverallCond is high because QualityIndex variable is a product of OverallQual and OverallCond. We will discard this junk model . However if our first 3 models had high VIF values, then we should consider removing a variable so that our variable selection models are not junk too.

Below are the VIF values for all 4 models: I have highlighted the highest VIF value for each model.

```
> sort(vif(model1),decreasing=TRUE)
 [1] 7.659282 6.076425 6.072867 4.000000 3.000000 2.767541 2.465041 2.416436 2.351859 2.161590 2.117672 2.000000
[13] 1.863113 1.827601 1.754629 1.684091 1.609430 1.554489 1.470235 1.455222 1.364959 1.351888 1.350721 1.324624
[25] 1.320315 1.268633 1.149049 1.139177 1.112824 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000
[37] 1.000000 1.000000 1.000000
> sort(vif(model2),decreasing=TRUE)
 [1] 6.787727 5.749913 4.133287 4.000000 3.000000 2.605327 2.282857 2.271772 2.071940 2.033049 1.824821 1.760022
[13] 1.521713 1.510913 1.439424 1.350859 1.338475 1.326658 1.233577 1.108015 1.000000 1.000000 1.000000 1.000000
[25] 1.000000 1.000000 1.000000
> sort(vif(model3),decreasing=TRUE)
 [1] 6.960519 5.859543 4.709358 4.000000 3.000000 2.638279 2.408666 2.342074 2.170106 2.156541 2.000000 1.860189
[13] 1.805116 1.670395 1.668540 1.594473 1.551988 1.468517 1.363887 1.343546 1.342695 1.292437 1.262724 1.136539
[25] 1.112244 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000
> sort(vif(model4),decreasing=TRUE)
 QualityIndex   OverallQual   OverallCond     GrLivArea TotalSqFtCalc
    36.122718     23.964958     18.410471      2.642212      2.476652
```

Model Comparison:

To select the best model, we need to compare the in-sample fit and predictive accuracy of our models. For each of these four models I computed the adjusted R-Squared, AIC, BIC, mean squared error, and the mean absolute error for each of these models for the training sample. Each of these metrics represents some concept of 'fit'.

Models	Adj R-sqr	AIC	BIC	MSE	MAE
Model1 (Full model)	0.9602 (#2)	44226.28 (#3)	44312.4 (#1)	213087267 (#1)	9341.263 (#2)
Model2 (Forward and stepwise)	0.9602 (#2)	44222.73 (#2)	44343.98 (#3)	213772625 (#3)	9300.193 (#1)
Model3 (backward)	0.9603 (#1)	44222.59 (#1)	44329.07 (#2)	213119456 (#2)	9344.764 (#3)
Model4 (Junk)	0.8155 (#3)	47338.66 (#4)	47377.89 (#4)	995193086 (#4)	22886.86 (#4)

We can see that models are ranked differently in different statistics. For example model3 is ranked 1 as per AIC but its ranked 2 in BIC.

Out of these 4 models, **Model3 best fits in-sample (train) dataset. Model2 also has near identical results.**

(4) Predictive Accuracy

In predictive modeling we are interested in how well our model performs (predicts) out-of-sample. For each of the four models we will compute the Mean Squared Error (MSE) and the Mean Absolute Error (MAE) for the test sample.

Models	MSE	MAE
Model1 (Full model)	189542403 (#3)	9293.027 (#3)
Model2 (Forward and stepwise)	188248943 (#1)	9251.175 (#1)
Model3 (backward)	188895301 (#2)	9279.05 (#2)
Model4 (Junk)	1032189669 (#4)	23769.75 (#4)

MSE and MAE are much better in test set compared to train that means for first 3 models that means these models are not over fitted.
When using MSE and MAE, Because of the square, large errors have relatively greater influence on MSE than do the smaller error. Therefore, MAE is more robust to outliers since it does not make use of square.
When a model has better predictive accuracy in-sample then it does out-of-sample, it implies that model is over fitted. I this situation, removing predictors can resolve this problem.
Out of these 4 models, based on different in-sample and out-of-sample fit metrics , **Model2 is the best model.**

(5) Operational Validation

We have validated these models in the statistical sense, lets validation these metrics in business sense. we need a grading system that we can use to determine the performance of the models relative to the business goals. In this case our goal is to come up with a model that can be used to predict future sale prices with some level confidence. Because we want to establish the confidence in the predictive ability, I'll use a grading system lists the performance of our models with between 0 and 10%, 10 to 15%, 15 to 25%, and more than 25% error respectively. To do this we compare the absolute values of the residuals for each divided by the SalePrice. We are interested in two aspects of this metric. We want to see 'most' of our observations fall within the lowest margin of error possible and we want to make sure our test set results do not deviate too much from our training set results. The table below give the complete comparison for each percentage accuracy grade obtained.

	Grade 1: [0 to 10%]		Grade 2: [10% to15%]		Grade 3: [15% to 25%]		Grade 4: [25% +]	
Models	Train	Test	Train	Test	Train	Test	Train	Test
Model1 (Full model)	83.81%	82.19%	7.22%	9.08%	5.63%	4.89%	3.34%	3.84%
Model2 (Forward and stepw	83.36%	81.72%	7.42%	9.78%	5.83%	4.66%	3.39%	3.84%
Model3 (backward)	83.91%	82.07%	7.08%	9.08%	5.68%	4.89%	3.34%	3.96%
Model4 (Junk)	49.43%	47.73%	17.67%	17.58%	18.02%	20.72%	14.88%	13.97%

Each listing above shows the training set value followed by the test set value for each prediction grade. For first 3 models these values are quite close. In these models more than 83% predictions are with less than 10% error. The other promising thing to note is that training and test results are close. The actual values between train and test will vary depending on the randomization of the split function. However, we should not see major deviations between train and test. Based on these results, my two best models

Model2 and Model3 appear to perform with strong accuracy in both train and test data. Also, these numbers appear to be good enough for 'underwriting quality' designation.

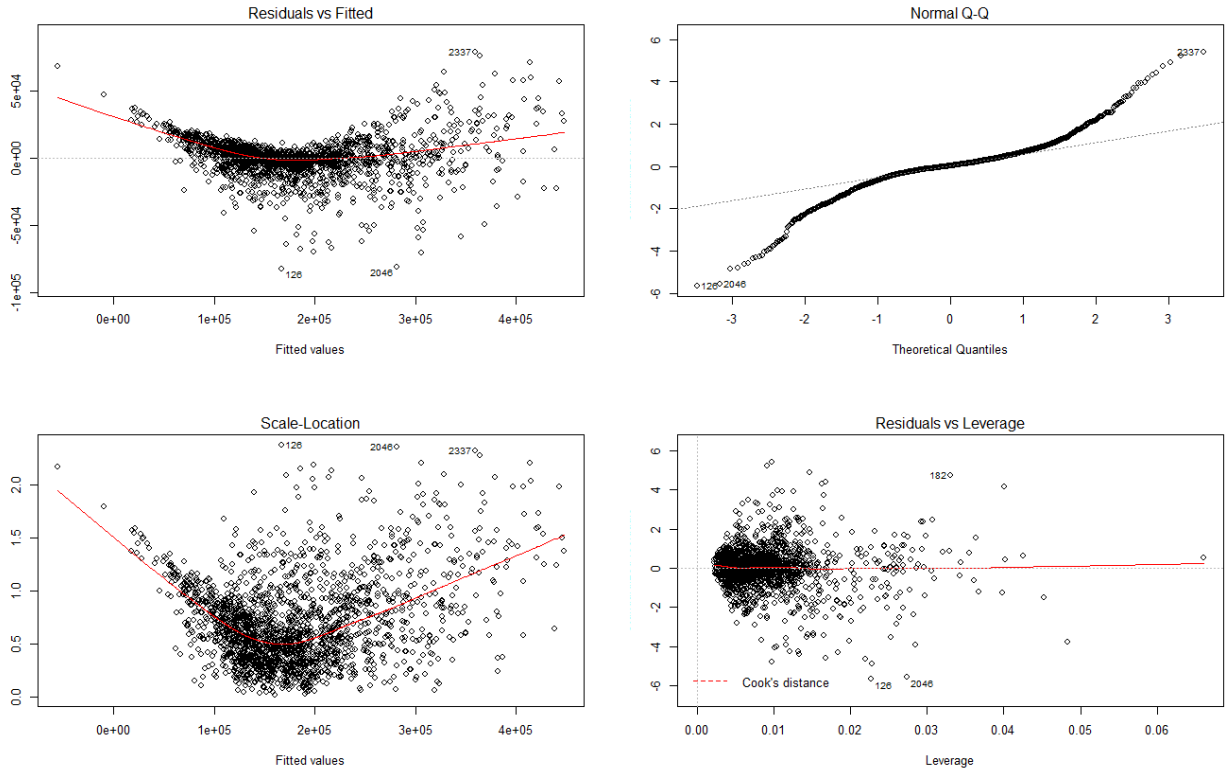
(6) Best Selected Model

Based on above automated variables selection techniques, predictive accuracy evaluation and operation validation, my best model is Model3. Model2 is very close but I will go with Model3 as is obtained using backward elimination process and is better able to handle collinearity than
the forward selection procedure.

Final selected model

lm(formula = SalePrice ~ LotArea + KitchenQual + TotRmsAbvGrd + GarageArea + HouseAge + QualityIndex + TotalSqFtCalc + TotalBathRooms + BldgTypeGrp + NbhdGrp + price_sqft, data = train.clean)

This model was selected on the basis of being the best fit for both training data and test data. I feel it strikes the right balance of number of predictors and ideal prediction accuracy.
I revisited the assumptions for the multiple linear model and conducted one final review. The plot of the result when running the model on the full sample set is shown below.



In residuals vs Fitted values scatter plot, under the standard assumptions, the standardized residuals are uncorrelated with the fitted values therefore we expect a random pattern but it seems like a funnel

shape distribution. Also the red line seems to be quadratic which suggest that our model violates the linearity and normality assumptions for residuals.
From the QQ plot as well, we can see that residuals are deviating from the mean line which again shows that our model is violating the normality assumption. Points in bottom left corner suggest that there are some outliers as well. From Scale-Location plot as well can see that the red line going upwards suggesting constant residual variance assumption violated.
We encounter these issues in previous assignments and transformation of SalePrice helped, however I am restricting my final model to non-transformed interpretations only.

Residual vs Leverage plot gives more insight into outliers and leverage data points in the sample dataset. Data points beyond the accepted limit of cooks distance can be treated as outliers. From the plot we can see few outliers in the dataset. I intentionally limited the number of drop conditions at the start to maintain as many observations as possible. As such, I was willing to let the assumptions lax a bit to focus on predictive robustness for future out-of-sample performance of the final model

Conclusion & Reflections:
I enjoyed this week's assignment. We extend our model building process to use automated variable selection process to arrive at a best model. My best final model gives a decent prediction grade while utilizing a range of predictor variables. It was interesting to see how changes to variables (and observations) included in the various models changed the significance and predictive power of other variables. Without a disciplined process and automated tools within R packages, this would be very time consuming.
The overall predictive ability of this final measure can likely be improved some but with a final estimated R-squared value of 96% and a Residual Standard Error of 14,660 without transformation, I'm happy with the results. Perhaps a really good model is never complete. As more observations become available we can continue to modify as needed. The important measure of any model we intend to use is if it meets the needs of the original business problem we started out to solve.