ASSIGNMENT 5

Predict 410 Fall 2017

Zeeshan Latifi

Northwestern University

Introduction:

This is an exploratory data analysis of housing data for Ames, Iowa. In this analysis, we will be using determining factors that can help predict the sales prices for a typical home in Ames, Iowa. The data has been provided by DeCock (2011). We will be looking for several predictor variables that will help determine our response variable: Sales Price.

To do this, we'll work through many aspects of data analysis. Initially, we'll evaluate our data, define the sample population, set up a predictive modeling framework, explore the use of automated variable selection techniques for model validation, assess the predictive accuracy of our model using cross-validation, and compare and contrast the difference between a statistical model validation and an application model validation. From our analyses, we'll be able to assess whether our response variable, sale price, can be predicted accurately for new observations of the predictor variable.

Sample Population:

There are 82 variables and 2,930 observations in the Ames, Iowa data set. We begin by conducting a waterfall in R to clean our data set. By evaluation of each variable, we've identified what constitutes a 'typical' home in Ames.

We began with filtering on the 'SubClass' field. This field identifies the class of the home. The decision was to keep only homes that are:

- 1-STORY 1946 & NEWER ALL STYLES
- 2-STORY 1946 & NEWER
- SPLIT OR MULTI-LEVEL

We then removed all non-residential zoning, keeping only residential high, medium, and low density. Next, removed all homes that were not on a paved street and did not have all public utilities included. A 'typical' home should have all standard utilities available.

To keep with our assumptions, the decision was made to only include homes that are in overall condition and quality of a 5 or higher. This means homes quality and condition ranked 'average' or higher. The same decision was made when filtering on the homes' exterior quality and condition. To eliminate homes that may skew our data set, only houses that were built in

1950 or later were included. Per our definition of the 'typical' home, we decided to only include homes that have square footage of 800 ft.^{2} or higher for the first level. With that in mind we also eliminated homes without a paved driveway or central air. Also, disregarded townhomes and homes with lot areas less than 5,000 ft^2 and above 20,000 ft^2. Additionally, we removed homes that had above 2,000 ft.^{2} of above ground living area, as this is atypical in Ames, Iowa. Finally, our sample will not include homes without a garage. With these transformations, the observations were reduced down to 1,082. Figure 1 displays the count for each of the reductions.

Figure 1:

	[,1]
01: Not SFR	1158
02: Non-Residential Zoning	82
03: Street Not Paved	2
04: Not All Utilities Included	2
05: Overall Quality Under 5	90
06: Overall Condition Under 5	32
07: Homes Built Pre-1950	17
08: Below Good Exterior Quality	2
09: Below Good Exterior Condition	5
10: First Floor Under 800 SqFt	113
11: No Central Air	4
12: No Paved Driveway	16
13: Not a Single Family Home	1
14: Not a Normal Lot Area	48
15: Abnormal Ground Living Area	261
99: Eligible Sample	1082

Predictive Modeling Framework:

In order to have a model that will be able to predict sale price of a home in Ames, we'll need to be able to assess it out-of-sample. We will follow the 70/30 train/test split for our analyses. With a train/test split we now have two data sets: one for in-sample model development and one for out-of-sample model assessment. This is the most basic form of model cross-validation. We will estimate the models on the 70% of the data (training data) identified as the training data set, and then examine the predictive accuracy on the remaining 30% of the data (test data). Figure 2 below outlines the breakdown of the sample population partition for both data sets.

Figure 2:

	Training Set	Test Set	Total Set
Count	766	316	1082

Model Identification by Automated Variable Selection and In-Sample Model Fit:

For our automated variable selection, we've decided to pull several variables that are believed to be good predictors of sales price based on our previous assignments. Figure 3 below contains the variables that are going to be a part of our pool of candidate predictor variables. When beginning the selection of variables, we calculated some additional variables:

- Quality Index: Overall Quality * Overall Condition
- Total Square Footage: Basement Finish 1 + Basement Finish 2 + Above Ground Living Area
- Total Baths: Basement Full and Half Baths + Full and Half Baths

To conduct our automated variable selection, we will utilize R to create a drop list of variables. The remaining variables will be used to for selection. Figure 3 below contains the variables that are going to be a part of our pool of candidate predictor variables.

Figure 3:

Variable	**Description**
SubClass	(Nominal): Identifies the type of dwelling involved in the sale.
LotArea	(Continuous): Lot size in square feet
LotShape	(Ordinal): General shape of property
Condition1	(Nominal): Proximity to various conditions
HouseStyle	(Nominal): Style of dwelling
YearBuilt	(Discrete): Original construction date
YearRemodel	(Discrete): Remodel date (same as construction date if no remodeling or additions)
MasVnrType	(Nominal): Masonry veneer type
ExterQual	(Ordinal): Evaluates the quality of the material on the exterior
ExterCond	(Ordinal): Evaluates the present condition of the material on the exterior
Foundation	(Nominal): Type of foundation
BsmtUnfSF	(Continuous): Unfinished square feet of basement area
TotalBsmtSF	(Continuous): Total square feet of basement area

FirstFlrSF	(Continuous): First Floor square feet
SecondFlrSF	(Continuous): Second floor square feet
GrLivArea	(Continuous): Above grade (ground) living area square feet
BedroomAbvGr	(Discrete): Bedrooms above grade (does NOT include basement bedrooms)
KitchenQual	(Ordinal): Kitchen quality
TotRmsAbvGrd	(Discrete): Total rooms above grade (does not include bathrooms)
Fireplaces	(Discrete): Number of fireplaces
GarageYrBlt	(Discrete): Year garage was built
GarageFinish	(Ordinal): Interior finish of the garage
GarageCars	(Discrete): Size of garage in car capacity
GarageArea	(Continuous): Size of garage in square feet
WoodDeckSF	(Continuous): Wood deck area in square feet
OpenPorchSF	(Continuous): Open porch area in square feet
MiscVal	(Continuous): $Value of miscellaneous feature
SaleType	(Nominal): Type of sale
OverallCond	(Ordinal): Rates the overall condition of the house
OverallQual	(Ordinal): Rates the overall material and finish of the house
SalePrice	(Continuous): Sale price $$
TotalSqftCalc	Basement Finish 1 + Basement Finish 2 + Above Ground Living Area
TotalBaths	Basement Full and Half Baths + Full and Half Baths
QualityIndex	Overall Quality * Overall Condition

This new data frame in R will be called 'train.clean'. This will be our data set going forward with the automated variable selection. Forward, backward, and stepwise model identification techniques will be used in order to find the best possible model. To accomplish this, we will use the stepAIC function in R.

Forward Selection:

Figures 4 and 5 below illustrate the stepAIC output for the lowest value of AIC as well as the summary of the model for forward selection.

Figure 4:

```
Step:  AIC=15186.92
SalePrice ~ ExterQual + GrLivArea + TotalBsmtSF + GarageCars +
    BsmtUnfSF + YearBuilt + KitchenQual + MasVnrType + SaleType +
    LotArea + BedroomAbvGr + Fireplaces + YearRemodel + MiscVal +
    Condition1 + TotRmsAbvGrd + OpenPorchSF

               Df  Sum of Sq          RSS   AIC
<none>                       282127465807 15187
+ GarageArea    1  476463688 281651002120 15188
+ SecondFlrSF   1  384078633 281743387175 15188
+ FirstFlrSF    1  336279230 281791186577 15188
+ WoodDeckSF    1  332284525 281795181282 15188
+ GarageFinish  2 1060682249 281066783558 15188
+ TotalBaths    1  224205188 281903260619 15188
+ SubClass      1   90594606 282036871201 15189
+ GarageYrBlt   1   35511582 282091954225 15189
+ Foundation    4 1970468586 280156997221 15190
+ HouseStyle    2  498953451 281628512357 15190
+ LotShape      3 1206747455 280920718352 15190
+ ExterCond     2  117808339 282009657468 15191
```

The output above provides the name of the variable, that is dropped, the change in degrees of freedom, the sum of squares explained by the dropped variable, the residual sum of squares for each subset model, and the value of the AIC statistics to be used to compare models. This is the final step in the output. Adding any more predictors will then increase the AIC value. As shown in the figure above, the model's lowest AIC value is at 15186.92. Figure 5 provides the summary statistics for the forward selection model. Note the adjusted R^2 at 0.8864.

Figure 5:

```
Call:
lm(formula = SalePrice ~ ExterQual + GrLivArea + TotalBsmtSF +
    GarageCars + BsmtUnfSF + YearBuilt + KitchenQual + MasVnrType +
    SaleType + LotArea + BedroomAbvGr + Fireplaces + YearRemodel +
    MiscVal + Condition1 + TotRmsAbvGrd + OpenPorchSF, data = train.clean)

Residuals:
    Min      1Q  Median      3Q     Max
-113448   -9974    -337   10236  153120

Coefficients:
                      Estimate   Std. Error t value             Pr(>|t|)
(Intercept)      -1255879.1477  148143.8608  -8.477 < 0.0000000000000002 ***
ExterQualGd        -62503.1637    5739.7256 -10.890 < 0.0000000000000002 ***
ExterQualTA        -72187.3533    6262.7493 -11.526 < 0.0000000000000002 ***
GrLivArea              45.5402       4.3879  10.378 < 0.0000000000000002 ***
TotalBsmtSF            40.0397       2.7152  14.746 < 0.0000000000000002 ***
GarageCars          11489.4184    1818.9890   6.316     0.0000000004662 ***
BsmtUnfSF             -19.0028       1.9434  -9.778 < 0.0000000000000002 ***
YearBuilt             536.7157      79.1120   6.784     0.0000000000242 ***
KitchenQualFa       -3901.6076   20448.0659  -0.191            0.848731
KitchenQualGd      -23057.3916    4193.4779  -5.498     0.0000000530959 ***
KitchenQualTA      -28072.0368    4659.5925  -6.025     0.0000000026928 ***
MasVnrTypeBrkCmn    -9935.4461   12785.5982  -0.777            0.437364
MasVnrTypeBrkFace   -8943.9815   10242.6887  -0.873            0.382839
MasVnrTypeNone     -11043.8600   10232.0313  -1.079            0.280793
MasVnrTypeStone      5621.6628   10390.9192   0.541            0.588662
SaleTypeCon         -7003.4027   20209.8699  -0.347            0.729042
SaleTypeConLD       13322.4642   12144.9518   1.097            0.273024
SaleTypeConLI      -18219.4520   20747.2166  -0.878            0.380145
SaleTypeConLw        1021.7271   20958.3987   0.049            0.961132
SaleTypeCWD        11460.7875    9739.3277   1.177            0.239679
SaleTypeNew         20879.0062    4965.0434   4.205     0.0000293404095 ***
SaleTypeOth         43863.1981   14518.2683   3.021            0.002606 **
SaleTypeVWD        -12281.8806   20608.1625  -0.596            0.551380
SaleTypeWD           7872.5432    3987.8576   1.974            0.048745 *
LotArea                 1.1811       0.3401   3.472            0.000546 ***
BedroomAbvGr        -6273.8657    1755.3699  -3.574            0.000375 ***
Fireplaces           4786.3683    1454.2942   3.291            0.001046 **
YearRemodel           164.5116      67.2214   2.447            0.014628 *
MiscVal                 2.1901       1.1026   1.986            0.047386 *
Condition1Feedr      3312.6551    7713.3111   0.429            0.667707
Condition1Norm       6497.6895    7187.4339   0.904            0.366276
Condition1PosA      14044.8695   10388.0857   1.352            0.176790
Condition1PosN      19501.0935    8922.7123   2.186            0.029166 *
Condition1RRAe      -9120.6730    9109.7022  -1.001            0.317061
Condition1RRAn      -5919.2728   10998.8258  -0.538            0.590622
Condition1RRNe      -4204.7212   15756.9535  -0.267            0.789661
Condition1RRNn      -3849.5955   15991.4989  -0.241            0.809834
TotRmsAbvGrd         2474.5513    1224.8435   2.020            0.043719 *
OpenPorchSF            26.5092      13.8199   1.918            0.055478 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

```
Residual standard error: 19700 on 727 degrees of freedom
Multiple R-squared:  0.892,     Adjusted R-squared:  0.8864
F-statistic:   158 on 38 and 727 DF,  p-value: < 0.00000000000000022
```

Backward Selection:

Figures 6 and 7 below illustrate the stepAIC output for the lowest value of AIC as well as the summary of the model for backward selection.

Figure 6:

```
Step:  AIC=15186.92
SalePrice ~ LotArea + Condition1 + YearBuilt + YearRemodel +
    MasVnrType + ExterQual + BsmtUnfSF + TotalBsmtSF + GrLivArea +
    BedroomAbvGr + KitchenQual + TotRmsAbvGrd + Fireplaces +
    GarageCars + OpenPorchSF + MiscVal + SaleType

               Df   Sum of Sq          RSS   AIC
<none>                        282127465807 15187
- OpenPorchSF   1  1427886210 283555352018 15189
- MiscVal       1  1530944117 283658409924 15189
- TotRmsAbvGrd  1  1583954371 283711420178 15189
- YearRemodel   1  2324284675 284451750482 15191
- Condition1    8  7600133330 289727599138 15191
- Fireplaces    1  4203573502 286331039309 15196
- LotArea       1  4679413881 286806879688 15198
- BedroomAbvGr  1  4957287659 287084753466 15198
- SaleType      9 13321129738 295448595545 15204
- MasVnrType    4 13313655974 295441121782 15214
- KitchenQual   3 14795005393 296922471200 15220
- GarageCars    1 15482707291 297610173098 15226
- YearBuilt     1 17861370535 299988836342 15232
- BsmtUnfSF     1 37105375197 319232841005 15280
- GrLivArea     1 41800120246 323927586053 15291
- ExterQual     2 52075942206 334203408013 15313
- TotalBsmtSF   1 84387248156 366514713963 15385
```

This is the final step in the output. Subtracting any more predictors will then increase the AIC value. As shown in the figure above, the model's lowest AIC value is at 15186.92. Figure 7 provides the summary statistics for the forward selection model. Note the adjusted R^2 at 0.8864 similar to the forward selection model.

Figure 7:

```
Call:
lm(formula = SalePrice ~ LotArea + Condition1 + YearBuilt + YearRemodel +
    MasVnrType + ExterQual + BsmtUnfSF + TotalBsmtSF + GrLivArea +
    BedroomAbvGr + KitchenQual + TotRmsAbvGrd + Fireplaces +
    GarageCars + OpenPorchSF + MiscVal + SaleType, data = train.clean)

Residuals:
    Min      1Q  Median      3Q     Max
-113448   -9974    -337   10236  153120

Coefficients:
                      Estimate    Std. Error t value            Pr(>|t|)
(Intercept)      -1255879.1477   148143.8608  -8.477 < 0.0000000000000002 ***
LotArea                 1.1811        0.3401   3.472            0.000546 ***
Condition1Feedr      3312.6551     7713.3111   0.429            0.667707
Condition1Norm       6497.6895     7187.4339   0.904            0.366276
Condition1PosA      14044.8695    10388.0857   1.352            0.176790
Condition1PosN      19501.0935     8922.7123   2.186            0.029166 *
Condition1RRAe      -9120.6730     9109.7022  -1.001            0.317061
Condition1RRAn      -5919.2728    10998.8258  -0.538            0.590622
Condition1RRNe      -4204.7212    15756.9535  -0.267            0.789661
Condition1RRNn      -3849.5955    15991.4989  -0.241            0.809834
YearBuilt             536.7157       79.1120   6.784      0.0000000000242 ***
YearRemodel           164.5116       67.2214   2.447            0.014628 *
MasVnrTypeBrkCmn    -9935.4461    12785.5982  -0.777            0.437364
MasVnrTypeBrkFace   -8943.9815    10242.6887  -0.873            0.382839
MasVnrTypeNone     -11043.8600    10232.0313  -1.079            0.280793
MasVnrTypeStone      5621.6628    10390.9192   0.541            0.588662
ExterQualGd        -62503.1637     5739.7256 -10.890 < 0.0000000000000002 ***
ExterQualTA        -72187.3533     6262.7493 -11.526 < 0.0000000000000002 ***
BsmtUnfSF             -19.0028        1.9434  -9.778 < 0.0000000000000002 ***
TotalBsmtSF            40.0397        2.7152  14.746 < 0.0000000000000002 ***
GrLivArea              45.5402        4.3879  10.378 < 0.0000000000000002 ***
BedroomAbvGr        -6273.8657     1755.3699  -3.574            0.000375 ***
KitchenQualFa       -3901.6076    20448.0659  -0.191            0.848731
KitchenQualGd      -23057.3916     4193.4779  -5.498      0.0000000530959 ***
KitchenQualTA      -28072.0368     4659.5925  -6.025      0.0000000026928 ***
TotRmsAbvGrd         2474.5513     1224.8435   2.020            0.043719 *
Fireplaces           4786.3683     1454.2942   3.291            0.001046 **
GarageCars          11489.4184     1818.9890   6.316      0.0000000004662 ***
OpenPorchSF            26.5092       13.8199   1.918            0.055478 .
MiscVal                 2.1901        1.1026   1.986            0.047386 *
SaleTypeCon         -7003.4027    20209.8699  -0.347            0.729042
SaleTypeConLD       13322.4642    12144.9518   1.097            0.273024
SaleTypeConLI      -18219.4520    20747.2166  -0.878            0.380145
SaleTypeConLw        1021.7271    20958.3987   0.049            0.961132
SaleTypeCWD         11460.7875     9739.3277   1.177            0.239679
SaleTypeNew         20879.0062     4965.0434   4.205      0.0000293404095 ***
SaleTypeOth         43863.1981    14518.2683   3.021            0.002606 **
SaleTypeVWD        -12281.8806    20608.1625  -0.596            0.551380
SaleTypeWD           7872.5432     3987.8576   1.974            0.048745 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

```
Residual standard error: 19700 on 727 degrees of freedom
Multiple R-squared:  0.892,     Adjusted R-squared:  0.8864
F-statistic:   158 on 38 and 727 DF,  p-value: < 0.00000000000000022
```

Stepwise Regression:

Figures 8 and 9 below illustrate the stepAIC output for the lowest value of AIC as well as the summary of the model for stepwise regression.

Figure 8:

```
Step:  AIC=15186.07
SalePrice ~ TotalSqftCalc + ExterQual + BsmtUnfSF + YearBuilt +
    GarageCars + KitchenQual + MasVnrType + Fireplaces + SaleType +
    LotArea + YearRemodel + Condition1 + OpenPorchSF + BedroomAbvGr +
    TotRmsAbvGrd + MiscVal

                Df    Sum of Sq          RSS   AIC
<none>                          282551848563 15186
+ GarageFinish   2   1174222575 281377625988 15187
+ TotalBsmtSF    1    424382755 282127465807 15187
+ GrLivArea      1    424382755 282127465807 15187
+ GarageArea     1    410446257 282141402305 15187
+ WoodDeckSF     1    333792145 282218056417 15187
+ TotalBaths     1    323372775 282228475788 15187
+ FirstFlrSF     1    108778451 282443070111 15188
+ GarageYrBlt    1     34254503 282517594059 15188
+ SecondFlrSF    1     17261761 282534586802 15188
+ SubClass       1      1819801 282550028762 15188
+ Foundation     4   2189000362 280362848200 15188
- MiscVal        1   1540072967 284091921529 15188
+ LotShape       3   1306242689 281245605874 15188
- OpenPorchSF    1   1677940662 284229789224 15189
+ ExterCond      2     97041588 282454806975 15190
+ HouseStyle     2     22599239 282529249323 15190
- YearRemodel    1   2414746853 284966595415 15191
- Condition1     8   8109045018 290660893580 15192
- TotRmsAbvGrd   1   3755148948 286306997511 15194
- LotArea        1   4763649268 287315497830 15197
- BedroomAbvGr   1   4884266861 287436115424 15197
- Fireplaces     1   4885417590 287437266152 15197
- SaleType       9  13241511762 295793360325 15203
- MasVnrType     4  12947635888 295499484451 15212
- KitchenQual    3  14710275967 297262124530 15219
- GarageCars     1  15835318346 298387166908 15226
- YearBuilt      1  18364624015 300916472578 15232
- BsmtUnfSF      1  31534272860 314086121422 15265
- ExterQual      2  51825317702 334377166265 15311
- TotalSqftCalc  1 131197634956 413749483519 15476
```

This is the final step in the output. Further modifications of any predictors will then increase the AIC value. As shown in the figure above, the model's lowest AIC value is at 15186.07, slightly lower than the forward and backward selection methods. Figure 9 provides the summary statistics for the forward selection model. Note the adjusted R^2 at 0.8864 similar to the previous two models.

Figure 9:

```
Call:
lm(formula = SalePrice ~ TotalSqftCalc + ExterQual + BsmtUnfSF +
    YearBuilt + GarageCars + KitchenQual + MasVnrType + Fireplaces +
    SaleType + LotArea + YearRemodel + Condition1 + OpenPorchSF +
    BedroomAbvGr + TotRmsAbvGrd + MiscVal, data = train.clean)

Residuals:
    Min      1Q  Median      3Q     Max
-113226  -10005    -312    9877  152073

Coefficients:
                    Estimate   Std. Error t value            Pr(>|t|)
(Intercept)      -1275560.762  146952.909  -8.680 < 0.0000000000000002 ***
TotalSqftCalc          41.609       2.263  18.386 < 0.0000000000000002 ***
ExterQualGd        -62090.373    5726.504 -10.843 < 0.0000000000000002 ***
ExterQualTA        -72045.390    6261.680 -11.506 < 0.0000000000000002 ***
BsmtUnfSF              22.152       2.458   9.014 < 0.0000000000000002 ***
YearBuilt             542.767      78.905   6.879     0.000000000013 ***
GarageCars          11599.899    1816.035   6.387     0.000000000301 ***
KitchenQualFa       -2378.919   20397.467  -0.117           0.907187
KitchenQualGd      -22927.393    4191.904  -5.469     0.000000062108 ***
KitchenQualTA      -27918.024    4657.564  -5.994     0.000000003218 ***
MasVnrTypeBrkCmn    -9487.007   12779.226  -0.742           0.458098
MasVnrTypeBrkFace   -8922.653   10243.327  -0.871           0.384003
MasVnrTypeNone     -10858.102   10231.147  -1.061           0.288915
MasVnrTypeStone      5435.771   10390.066   0.523           0.601015
Fireplaces           5069.661    1428.932   3.548           0.000413 ***
SaleTypeCon         -7811.348   20196.394  -0.387           0.699040
SaleTypeConLD       12981.970   12141.367   1.069           0.285318
SaleTypeConLI      -18801.451   20741.083  -0.906           0.364980
SaleTypeConLw        1929.842   20941.746   0.092           0.926602
SaleTypeCWD        11390.603    9739.722   1.169           0.242585
SaleTypeNew        20693.257    4962.184   4.170     0.000034102664 ***
SaleTypeOth        44018.360   14518.443   3.032           0.002517 **
SaleTypeVWD       -13186.922   20591.305  -0.640           0.522106
SaleTypeWD           7832.470    3987.930   1.964           0.049905 *
LotArea                 1.191       0.340   3.503           0.000487 ***
YearRemodel           167.528      67.164   2.494           0.012840 *
Condition1Feedr      3361.849    7713.663   0.436           0.663089
Condition1Norm       6594.815    7187.296   0.918           0.359150
Condition1PosA      15058.731   10343.409   1.456           0.145858
Condition1PosN      19645.155    8922.222   2.202           0.027991 *
Condition1RRAe      -9750.460    9090.358  -1.073           0.283799
Condition1RRAn      -5912.722   10999.531  -0.538           0.591057
Condition1RRNe      -3170.621   15726.904  -0.202           0.840282
Condition1RRNn      -4282.611   15987.164  -0.268           0.788869
OpenPorchSF            28.471      13.693   2.079           0.037945 *
BedroomAbvGr        -6225.308    1754.868  -3.547           0.000414 ***
TotRmsAbvGrd         3180.058    1022.362   3.110           0.001941 **
MiscVal                 2.197       1.103   1.992           0.046745 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

```
Residual standard error: 19700 on 728 degrees of freedom
Multiple R-squared:  0.8918,    Adjusted R-squared:  0.8864
F-statistic: 162.3 on 37 and 728 DF,  p-value: < 0.00000000000000022
```

Junk Model:

Figures 10 below illustrates the junk model.

Figure 10:

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

Residuals:
    Min      1Q  Median      3Q     Max
-134006  -14076    -338   12769  146511

Coefficients:
                Estimate Std. Error t value            Pr(>|t|)
(Intercept)   -299785.62   38030.44  -7.883   0.0000000000000111 ***
OverallQual     62958.12    6283.17  10.020 < 0.0000000000000002 ***
OverallCond     34945.25    6935.16   5.039   0.0000005857067015 ***
QualityIndex    -5603.00    1184.32  -4.731   0.0000026640777830 ***
GrLivArea          23.66       4.45   5.318   0.0000001382863281 ***
TotalSqftCalc      28.47       2.16  13.178 < 0.0000000000000002 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 25460 on 760 degrees of freedom
Multiple R-squared:  0.8114,    Adjusted R-squared:  0.8102
F-statistic:   654 on 5 and 760 DF,  p-value: < 0.00000000000000022
```

Figure 10 is the summary statistics for our junk model. The junk model is used as our baseline model. We've incorporated variables that we feel may be good predictors for sale price. Our data set for the junk model is also the train set. This is the data set that contains all variables prior to our drops. However, looking at the summary statistics, it seems some of variables such as OverallQual and OverallCond are highly correlated.

Notice that in our forward, backward, stepwise models there is no OverallQual, OverallCond, or QualityIndex variables present. This is because after calculating the variance inflation factor (VIF), these variables had severely high values and were very much highly correlated. Figure 11 below provides the VIF values of each variable. Once we realized the VIF values were too high, the decision was to remove it from our models, as represented by the models above. Although we did not use any indicator variables, variables with high collinearity can be disregarded when evaluating VIF.

Figure 11:

VIF: Forward

	GVIF
OverallQual	77.951614
TotalSqftCalc	3.191541
ExterQual	6.449569
BsmtUnfSF	2.718855
GarageCars	2.525061
YearRemodel	3.832442
SaleType	1.957600
KitchenQual	4.795287
LotArea	1.352774
MasVnrType	1.767683
YearBuilt	7.050117
OverallCond	47.522388
Condition1	1.675616
Fireplaces	1.543091
QualityIndex	71.567754
OpenPorchSF	1.194862
BedroomAbvGr	1.458017
TotRmsAbvGrd	2.243724

VIF: Backward

	GVIF
LotArea	1.352375
Condition1	1.714862
OverallQual	78.662374
OverallCond	47.634156
YearBuilt	4.598977
MasVnrType	1.781958
ExterQual	6.546086
BsmtUnfSF	1.601295
TotalBsmtSF	1.881203
GrLivArea	4.480643
BedroomAbvGr	1.457023
KitchenQual	4.449285
TotRmsAbvGrd	3.223083
Fireplaces	1.569109
GarageCars	2.518310
OpenPorchSF	1.207126
SaleType	1.934703
QualityIndex	72.735528

VIF: Stepwise

	GVIF
TotalSqftCalc	3.188512
OverallQual	77.853520
ExterQual	6.448593
BsmtUnfSF	2.716050
GarageCars	2.514126
SaleType	1.923490
KitchenQual	4.421700
LotArea	1.351774
MasVnrType	1.740217
YearBuilt	4.589043
OverallCond	46.639440
Condition1	1.664911
Fireplaces	1.528116
QualityIndex	71.403906
OpenPorchSF	1.183256
BedroomAbvGr	1.455313
TotRmsAbvGrd	2.238787

VIF: Junk

OverallQual	OverallCond	QualityIndex	GrLivArea	TotalSqftCalc
62.708966	43.164800	65.372870	2.563643	1.651407

After removal of QualityIndex, OverallQual, & OverallCond

VIF: Forward

	GVIF
ExterQual	5.994884
GrLivArea	4.164253
TotalBsmtSF	1.857603
GarageCars	2.463613
BsmtUnfSF	1.583491
YearBuilt	4.704295
KitchenQual	4.562864
MasVnrType	1.700181
SaleType	1.914301
LotArea	1.346203
BedroomAbvGr	1.449919
Fireplaces	1.539515
YearRemodel	3.095374
MiscVal	1.034960
Condition1	1.639853
TotRmsAbvGrd	3.202032
OpenPorchSF	1.213060

VIF: Backward

	GVIF
LotArea	1.346203
Condition1	1.639853
YearBuilt	4.704295
YearRemodel	3.095374
MasVnrType	1.700181
ExterQual	5.994884
BsmtUnfSF	1.583491
TotalBsmtSF	1.857603
GrLivArea	4.164253
BedroomAbvGr	1.449919
KitchenQual	4.562864
TotRmsAbvGrd	3.202032
Fireplaces	1.539515
GarageCars	2.463613
OpenPorchSF	1.213060
MiscVal	1.034960
SaleType	1.914301

VIF: Stepwise

	GVIF
TotalSqftCalc	3.027141
ExterQual	5.914573
BsmtUnfSF	2.531937
YearBuilt	4.679129
GarageCars	2.455302
KitchenQual	4.538506
MasVnrType	1.666531
Fireplaces	1.486095
SaleType	1.898191
LotArea	1.345117
YearRemodel	3.089675
Condition1	1.585183
OpenPorchSF	1.190711
BedroomAbvGr	1.448905
TotRmsAbvGrd	2.230583
MiscVal	1.034927

VIF in Descending Order

VIF Forward:

```
> sort(vif(forward.lm),decreasing=TRUE)
 [1] 9.000000 8.000000 5.994884 4.704295 4.562864 4.164253 4.000000 3.202032 3.095374 3.000000 2.463613 2.168939
[13] 2.040650 2.000000 1.914301 1.857603 1.789422 1.759368 1.700181 1.639853 1.583491 1.569590 1.564751 1.539515
[25] 1.449919 1.362939 1.346203 1.287873 1.258368 1.240772 1.213060 1.204126 1.160260 1.101390 1.068592 1.036734
[37] 1.034960 1.031396 1.017330 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000
[49] 1.000000 1.000000 1.000000
```

VIF Backward:

```
> sort(vif(backward.lm),decreasing=TRUE)
 [1] 9.000000 8.000000 5.994884 4.704295 4.562864 4.164253 4.000000 3.202032 3.095374 3.000000 2.463613 2.168939
[13] 2.040650 2.000000 1.914301 1.857603 1.789422 1.759368 1.700181 1.639853 1.583491 1.569590 1.564751 1.539515
[25] 1.449919 1.362939 1.346203 1.287873 1.258368 1.240772 1.213060 1.204126 1.160260 1.101390 1.068592 1.036734
[37] 1.034960 1.031396 1.017330 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000
[49] 1.000000 1.000000 1.000000
```

VIF Stepwise:

```
> sort(vif(stepwise.lm),decreasing=TRUE)
 [1] 9.000000 8.000000 5.914573 4.679129 4.538506 4.000000 3.089675 3.027141 3.000000 2.531937 2.455302 2.230583
[13] 2.163129 2.000000 1.898191 1.757747 1.739868 1.666531 1.591206 1.585183 1.566940 1.559484 1.493514 1.486095
[25] 1.448905 1.345117 1.286724 1.219055 1.203705 1.190711 1.159792 1.091197 1.065925 1.036247 1.034927 1.029212
[37] 1.017314 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000 1.000000
```

According to the values listed above in Figure 11, the values of VIF that were lowest were found in the stepwise regression model. This corresponds to our models above where stepwise was determined to be the best model for predictive accuracy. We will examine these models further next.

Model Comparison:

We will now compare the four models with metrics that represent some concept of 'fit'. We'll be ranking the metric for each model in order to determine the best model. The figure below breaks down each metric for each model:

Figure 12:

	AIC	**BIC**	**MAE**	**MSE**	**Adj R^2**	**Overall Rank**
Forward	17362.7368970	17548.3841838	13419.4821163	368312618.5472388	0.8863657	2
Backward	17362.7368970	17548.3841838	13419.4821163	368312618.5472390	0.8863657	3
Stepwise	17361.8882664	17542.8943710	13447.5382824	368866643.0319906	0.8863511	1
Junk	17723.8363596	17756.3246347	17975.1422339	643230531.2298237	0.8101632	4

The stepwise model is ranked number 1 in AIC, BIC, and Adjusted R^2. However, the MAE and MSE are lower for the forward and backward models. MAE and MSE for the forward and backward models are lower than the stepwise model. Therefore, they rank number 1 in that category. The junk model scored last in all categories, as this is expected. Based on our findings, having the lowest AIC or BIC will not always translate into having the best MAE or MSE.

Predictive Accuracy:

We will now evaluate how our model performs out-of-sample. Figure 13 below outlines the MAE and MSE for each model based on our test sample data (30% split).

Figure 13:

	MAE	**MSE**
Forward	14084.69	415431232
Backward	14084.69	415431232
Stepwise	14126.98	417760163
Junk	17174.77	593459770

Based on our findings, the model that fits best for the test data (out-of-sample) is either the forward selection or backward elimination models. This is in contrast to the in-sample models. According to those, the stepwise regression worked best with the lowest AIC values. However, the stepwise method did not have the lowest MAE even within the in-sample data set. Ideally there shouldn't a preference between MAE and MSE, however, we prefer the values of MAE. This is because since MSE squared penalizes large errors more so. Interpretation of a model for predicting out of sample, MAE will be more forgiving. Generally, a better fitting model will be better at predicting in-sample data. This in turn will reflect to the out of sample data. Therefore, in our analyses, we can infer that since the model is a better predictor of in-sample, it can be considered a good predictive model. Since we based our model with the training data (in-sample) this has better predictive accuracy.

Operational Validation:

Within our analyses, we need to ensure that the models are within a threshold that is satisfactory to the business policies. We've assigned cut-off points to provide statistics on the accuracy of our predicted value. To this, in R we've established a variable, 'PredictionGrade'. If the predicted value falls within 10% of the actual value, it'll be assigned a value of Grade 1. Anything between 10%-15% will be given a Grade 2. 15%-25% will be assigned Grade 3 and anything above 25% will be assigned Grade 4.

We've done this for each of our models for both in-sample and out-of-sample. Figure 14 below represents the breakdown for each:

Figure 14:

Forward In-Sample

```
forward.PredictionGrade
   Grade 1: [0.0.10] Grade 2: (0.10,0.15] Grade 3: (0.15,0.25]     Grade 4: (0.25+]
          0.75195822          0.14229765          0.08616188           0.01958225
```

Backward In-Sample

```
backward.PredictionGrade
   Grade 1: [0.0.10] Grade 2: (0.10,0.15] Grade 3: (0.15,0.25]     Grade 4: (0.25+]
          0.75195822          0.14229765          0.08616188           0.01958225
```

Stepwise In-Sample

```
stepwise.PredictionGrade
   Grade 1: [0.0.10] Grade 2: (0.10,0.15] Grade 3: (0.15,0.25]     Grade 4: (0.25+]
          0.75456919          0.13707572          0.08877285           0.01958225
```

Junk In-Sample

```
junk.PredictionGrade
   Grade 1: [0.0.10] Grade 2: (0.10,0.15] Grade 3: (0.15,0.25]     Grade 4: (0.25+]
          0.61879896          0.19060052          0.12793734           0.06266319
```

Forward Out-of-Sample

```
forward.testPredictionGrade
   Grade 1: [0.0.10] Grade 2: (0.10,0.15] Grade 3: (0.15,0.25]     Grade 4: (0.25+]
          0.74050633          0.16139241          0.08227848           0.01582278
```

Backward Out-of-Sample

```
backward.testPredictionGrade
   Grade 1: [0.0.10] Grade 2: (0.10,0.15] Grade 3: (0.15,0.25]     Grade 4: (0.25+]
          0.74050633          0.16139241          0.08227848           0.01582278
```

Stepwise Out-of-Sample

```
stepwise.testPredictionGrade
  Grade 1: [0.0.10] Grade 2: (0.10,0.15] Grade 3: (0.15,0.25]     Grade 4: (0.25+]
         0.74367089           0.16772152           0.07278481           0.01582278
```

Junk Out-of-Sample

```
junk.testPredictionGrade
  Grade 1: [0.0.10] Grade 2: (0.10,0.15] Grade 3: (0.15,0.25]     Grade 4: (0.25+]
         0.64556962           0.18987342           0.12658228           0.03797468
```

Staying with the pattern of the analyses above, the stepwise regression model had the greatest predictive accuracy with the in-sample data with approximately 75% of the values were within 10% of the actual value. When evaluating out-of-sample data, forward selection and backward elimination methods were best in predictive accuracy. The model rankings look to remain the same based on the assessment of prediction grades.

Conclusion:

According to our findings with the several analyses above, the best model for predicting out-of-sample data for housing sale price in Ames, Iowa is the stepwise regression method. Although, the MAE was not the lowest when assessing the model both in-sample and out-of-sample (Figure 12), we felt that due to the little discrepancy between the values this model is the most accurate. The AIC and BIC values for this model were also the lowest. In addition, 75.45% of the predicted values were within the actual values, giving us the highest percentage in Grade 1 than any other model, as shown in Figure 14.

Code Appendix:

```
# Zeeshan Latifi
# 10.21.2017
# ames_waterfall.R

# Read in csv file for Ames housing data;

# Note that back slash is an escape character in R so we use \\ when we want \;
path.name <- '/Users/Zeeshan/Desktop/PREDICT 410/Week 1/';
file.name <- paste(path.name,'ames_housing_data.csv',sep='');

# Read in the csv file into an R data frame;
amesiowa.df <- read.csv(file.name,header=TRUE,stringsAsFactors=FALSE);

# Single ifelse() statement
# ifelse(condition, value if condition is TRUE, value if the condition is FALSE)

# Nested ifelse() statement
# ifelse(condition1, value if condition1 is TRUE,
#       ifelse(condition2, value if condition2 is TRUE,
#       value if neither condition1 nor condition2 is TRUE
#       )
# )


# Create a waterfall of drop conditions;
# Work the data frame as a 'table' like you would in SAS or SQL;
amesiowa.df$dropConditions <- ifelse(amesiowa.df$SubClass!= 020 & amesiowa.df$SubClass != 060 &
amesiowa.df$SubClass != 080,'01: Not SFR',
  ifelse(amesiowa.df$Zoning!='RH' & amesiowa.df$Zoning!='RL' & amesiowa.df$Zoning!='RM','02: Non-Residential
Zoning',
  ifelse(amesiowa.df$Street!='Pave','03: Street Not Paved',
  ifelse(amesiowa.df$Utilities!='AllPub', '04: Not All Utilities Included',
  ifelse(amesiowa.df$OverallQual<5, '05: Overall Quality Under 5',
  ifelse(amesiowa.df$OverallCond<5, '06: Overall Condition Under 5',
```

```
  ifelse(amesiowa.df$YearBuilt<1950, '07: Homes Built Pre-1950',
  ifelse(amesiowa.df$ExterQual!='TA' & amesiowa.df$ExterQual!='Gd'& amesiowa.df$ExterQual!='Ex', '08: Below Good Exterior Quality',
  ifelse(amesiowa.df$ExterCond!='TA' & amesiowa.df$ExterCond!='Gd'& amesiowa.df$ExterCond!='Ex', '09: Below Good Exterior Condition',
  ifelse(amesiowa.df$FirstFlrSF<800, '10: First Floor Under 800 SqFt',
  ifelse(amesiowa.df$CentralAir!='Y', '11: No Central Air',
  ifelse(amesiowa.df$PavedDrive!='Y', '12: No Paved Driveway',
  ifelse(amesiowa.df$BldgType!='1Fam', '13: Not a Single Family Home',
  ifelse(amesiowa.df$LotArea<5000 | amesiowa.df$LotArea>20000, '14: Not a Normal Lot Area',
  ifelse(amesiowa.df$GrLivArea>2000, '15: Abnormal Ground Living Area',
  ifelse(amesiowa.df$GarageFinish=='NA', '16: No Garage',
  '99: Eligible Sample')
  )))))))))))))));


table(amesiowa.df$dropConditions)

# Save the table
waterfalls <- table(amesiowa.df$dropConditions);

# Format the table as a column matrix for presentation;
as.matrix(waterfalls,15,1)



# Eliminate all observations that are not part of the eligible sample population;
myeligible.population <- subset(amesiowa.df,dropConditions=='99: Eligible Sample');

# Check that all remaining observations are eligible;
table(myeligible.population$dropConditions);

head(myeligible.population)

#############################################################################################
#Assignment 5
```

```
#Part 2 predictive modeling framework

set.seed(123)
myeligible.population$u <- runif(n=dim(myeligible.population)[1],min=0,max=1);

myeligible.population$QualityIndex <- myeligible.population$OverallQual*myeligible.population$OverallCond;
myeligible.population$TotalSqftCalc <- myeligible.population$BsmtFinSF1 + myeligible.population$BsmtFinSF2 +
myeligible.population$GrLivArea;
myeligible.population$TotalBaths <- myeligible.population$BsmtFullBath +
myeligible.population$BsmtHalfBath*0.5 +
  myeligible.population$FullBath + myeligible.population$HalfBath*0.5



# Create train/test split;
train.df <- subset(myeligible.population, u<0.70);
test.df <- subset(myeligible.population, u>=0.70);
# Check your data split. The sum of the parts should equal the whole. # Do your totals add up?
dim(myeligible.population)[1]
dim(train.df)[1]
dim(test.df)[1]
dim(train.df)[1]+dim(test.df)[1]

framework.table <- matrix(c(dim(train.df)[1],dim(test.df)[1], dim(train.df)[1]+dim(test.df)[1]),ncol=3,byrow=TRUE)
colnames(framework.table) <- c("Training Set","Test Set","Total Set")
rownames(framework.table)<-c("Count")
fm.table <- as.table(framework.table)
fm.table
##########################################################################################
#Assignment 5
#Part 3 Model Identification by Automated Variable Selection

drop.list <- c('SID','PID','LotConfig','dropConditions','Utilities','Zoning','LotFrontage','Street', 'Fence',
        'Exterior1','Exterior2','BsmtFinSF1','BsmtFinSF2','CentralAir','YrSold','MoSold','SaleCondition',
        'u','train','I2010','BsmtFullBath','BsmtHalfBath','FullBath','HalfBath', 'FireplaceInd1',
        'FireplaceInd2','RoofStyle','RoofFlat','PoolArea','LandContour','LandSlope','HeatingQC', 'PoolQC',
```

```
        'Alley','FireplaceQu','MiscFeature','KitchenAbvGr','LowQualFinSF','Functional','EnclosedPorch',
        'ThreeSsnPorch','PavedDrive','BldgType','RoofMat','Condition2','BsmtCond', 'Electrical','GarageQual',

'GarageCond','ScreenPorch','MasVnrArea','BsmtQual','BsmtExposure','BsmtFinType1','BsmtFinType2','Heating',
        'GarageType','Neighborhood','OverallQual','OverallCond','QualityIndex');

train.clean <-train.df[,!(names(myeligible.population) %in% drop.list)];
head(train.clean)

colnames(train.clean)



#Model Identification
library(MASS)

# Define the upper model as the FULL model
upper.lm <- lm(SalePrice ~ .,data=train.clean);
summary(upper.lm)

# Define the lower model as the Intercept model
lower.lm <- lm(SalePrice ~ 1,data=train.clean);
summary(lower.lm)

# Need a SLR to initialize stepwise selection
sqft.lm <- lm(SalePrice ~ TotalSqftCalc,data=train.clean);
summary(sqft.lm)

#unlist(lapply(train.clean, function(x) any(is.na(x))))

# Call stepAIC() for variable selection
forward.lm <- stepAIC(object=lower.lm,scope=list(upper=formula(upper.lm),lower=~1),direction=c('forward'));
summary(forward.lm)

backward.lm <- stepAIC(object=upper.lm,direction=c('backward'));
summary(backward.lm)
```

```
stepwise.lm <- stepAIC(object=sqft.lm,scope=list(upper=formula(upper.lm),lower=~1), direction=c('both'));
summary(stepwise.lm)

junk.lm <- lm(SalePrice ~ OverallQual + OverallCond + QualityIndex + GrLivArea + TotalSqftCalc, data=train.df)
summary(junk.lm)

# Compute the VIF values
library(car)
sort(vif(forward.lm),decreasing=TRUE)
sort(vif(backward.lm),decreasing=TRUE)
sort(vif(stepwise.lm),decreasing=TRUE)
sort(vif(junk.lm),decreasing=TRUE)



vif(forward.lm)
vif(backward.lm)
vif(stepwise.lm)
vif(junk.lm)

forward.info <- c(AIC(forward.lm),BIC(forward.lm), mean(abs(forward.lm$residuals)),
mean(forward.lm$residuals^2),
        summary(forward.lm)$adj.r.squared, 1)

backward.info <- c(AIC(backward.lm),BIC(backward.lm), mean(abs(backward.lm$residuals)),
            mean(backward.lm$residuals^2), summary(backward.lm)$adj.r.squared, 2)

stepwise.info <- c(AIC(stepwise.lm),BIC(stepwise.lm), mean(abs(stepwise.lm$residuals)),
            mean(stepwise.lm$residuals^2), summary(stepwise.lm)$adj.r.squared, 3)

junk.info <- c(AIC(junk.lm),BIC(junk.lm), mean(abs(junk.lm$residuals)), mean(junk.lm$residuals^2),
          summary(junk.lm)$adj.r.squared, 4)

options(scipen = 9999)
```

```
models.info <- matrix(c(AIC(forward.lm),BIC(forward.lm), mean(abs(forward.lm$residuals)),
mean(forward.lm$residuals^2),
        summary(forward.lm)$adj.r.squared, 1, AIC(backward.lm),BIC(backward.lm),
mean(abs(backward.lm$residuals)),
        mean(backward.lm$residuals^2), summary(backward.lm)$adj.r.squared, 2,
AIC(stepwise.lm),BIC(stepwise.lm), mean(abs(stepwise.lm$residuals)),
        mean(stepwise.lm$residuals^2), summary(stepwise.lm)$adj.r.squared, 3, AIC(junk.lm),BIC(junk.lm),
mean(abs(junk.lm$residuals)), mean(junk.lm$residuals^2),
        summary(junk.lm)$adj.r.squared, 4),ncol=6,byrow=TRUE)

#models.table <- matrix(c(forward.info,backward.info,stepwise.info,junk.info, nrow=6,byrow=TRUE))
colnames(models.info) <- c('AIC','BIC','MAE','MSE','Adj R2','Rank')
rownames(models.info)<-c('Forward','Backward','Stepwise','Junk')

model.tbl <- as.table(models.info)
model.tbl

###############################################################################################
#Assignment 5
#Part 4 Predictive Accuracy
forward.test <- predict(forward.lm,newdata=test.df);
backward.test <- predict(backward.lm,newdata=test.df);
stepwise.test <- predict(stepwise.lm,newdata=test.df)
junk.test <- predict(junk.lm,newdata=test.df)

forward.pred.mae <- mean(abs(forward.test-test.df$SalePrice))
forward.pred.mse <- mean((forward.test-test.df$SalePrice)^2)

backward.pred.mae <- mean(abs(backward.test-test.df$SalePrice))
backward.pred.mse <- mean((backward.test-test.df$SalePrice)^2)

stepwise.pred.mae <- mean(abs(stepwise.test-test.df$SalePrice))
stepwise.pred.mse <- mean((stepwise.test-test.df$SalePrice)^2)

junk.pred.mae <- mean(abs(junk.test-test.df$SalePrice))
```

```
junk.pred.mse <- mean((junk.test-test.df$SalePrice)^2)



##########################################################################################
#Assignment 5
#Part 5 Operational Validation

# Training Data
# Abs Pct Error
forward.pct <- abs(forward.lm$residuals)/train.clean$SalePrice;
# Assign Prediction Grades;
forward.PredictionGrade <- ifelse(forward.pct<=0.10,'Grade 1: [0.0.10]',
                ifelse(forward.pct<=0.15,'Grade 2: (0.10,0.15]',
                ifelse(forward.pct<=0.25,'Grade 3: (0.15,0.25]', 'Grade 4: (0.25+]')
                ))
forward.trainTable <- table(forward.PredictionGrade)
forward.trainTable/sum(forward.trainTable)

#------------------------------------------------------------------------------
backward.pct <- abs(backward.lm$residuals)/train.clean$SalePrice;
# Assign Prediction Grades;
backward.PredictionGrade <- ifelse(backward.pct<=0.10,'Grade 1: [0.0.10]',
                ifelse(backward.pct<=0.15,'Grade 2: (0.10,0.15]',
                  ifelse(backward.pct<=0.25,'Grade 3: (0.15,0.25]', 'Grade 4: (0.25+]')
                ))
backward.trainTable <- table(backward.PredictionGrade)
backward.trainTable/sum(backward.trainTable)

#------------------------------------------------------------------------------
stepwise.pct <- abs(stepwise.lm$residuals)/train.clean$SalePrice;
# Assign Prediction Grades;
stepwise.PredictionGrade <- ifelse(stepwise.pct<=0.10,'Grade 1: [0.0.10]',
                ifelse(stepwise.pct<=0.15,'Grade 2: (0.10,0.15]',
                  ifelse(stepwise.pct<=0.25,'Grade 3: (0.15,0.25]', 'Grade 4: (0.25+]')
```

```
                          ))
stepwise.trainTable <- table(stepwise.PredictionGrade)
stepwise.trainTable/sum(stepwise.trainTable)

#--------------------------------------------------------------------------------
junk.pct <- abs(junk.lm$residuals)/train.clean$SalePrice;
# Assign Prediction Grades;
junk.PredictionGrade <- ifelse(junk.pct<=0.10,'Grade 1: [0.0.10]',
                          ifelse(junk.pct<=0.15,'Grade 2: (0.10,0.15]',
                               ifelse(junk.pct<=0.25,'Grade 3: (0.15,0.25]', 'Grade 4: (0.25+]')
                          ))
junk.trainTable <- table(junk.PredictionGrade)
junk.trainTable/sum(junk.trainTable)

#--------------------------------------------------------------------------------
# Test Data

# Abs Pct Error
forward.testPCT <- abs(test.df$SalePrice-forward.test)/test.df$SalePrice;
backward.testPCT <- abs(test.df$SalePrice-backward.test)/test.df$SalePrice;
stepwise.testPCT <- abs(test.df$SalePrice-stepwise.test)/test.df$SalePrice;
junk.testPCT <- abs(test.df$SalePrice-junk.test)/test.df$SalePrice;

# Assign Prediction Grades;
forward.testPredictionGrade <- ifelse(forward.testPCT<=0.10,'Grade 1: [0.0.10]',
                             ifelse(forward.testPCT<=0.15,'Grade 2: (0.10,0.15]',
                             ifelse(forward.testPCT<=0.25,'Grade 3: (0.15,0.25]', 'Grade 4: (0.25+]')
                             ))
forward.testTable <-table(forward.testPredictionGrade)
forward.testTable/sum(forward.testTable)

#--------------------------------------------------------------------------------
backward.testPredictionGrade <- ifelse(backward.testPCT<=0.10,'Grade 1: [0.0.10]',
                              ifelse(backward.testPCT<=0.15,'Grade 2: (0.10,0.15]',
                                   ifelse(backward.testPCT<=0.25,'Grade 3: (0.15,0.25]', 'Grade 4: (0.25+]')
```

```
                    ))
backward.testTable <-table(backward.testPredictionGrade)
backward.testTable/sum(backward.testTable)


#--------------------------------------------------------------------------------
stepwise.testPredictionGrade <- ifelse(stepwise.testPCT<=0.10,'Grade 1: [0.0.10]',
                    ifelse(stepwise.testPCT<=0.15,'Grade 2: (0.10,0.15]',
                        ifelse(stepwise.testPCT<=0.25,'Grade 3: (0.15,0.25]', 'Grade 4: (0.25+]')
                    ))
stepwise.testTable <-table(stepwise.testPredictionGrade)
stepwise.testTable/sum(stepwise.testTable)

#--------------------------------------------------------------------------------
junk.testPredictionGrade <- ifelse(junk.testPCT<=0.10,'Grade 1: [0.0.10]',
                    ifelse(junk.testPCT<=0.15,'Grade 2: (0.10,0.15]',
                        ifelse(junk.testPCT<=0.25,'Grade 3: (0.15,0.25]', 'Grade 4: (0.25+]')
                    ))
junk.testTable <-table(junk.testPredictionGrade)
junk.testTable/sum(junk.testTable)
```