Assignment 5: Automated Variable selection, multicollinearity and predictive modeling.

Sri Seshadri

7/19/2017

1. Introduction

In this report, Automated Variable selection Methods (AVM) for modeling sales price of typical homes in Ames Iowa is discussed. The models from automated variable selection methods are compared with a known "junk" model with correlated predictors. It is found that the automated variable selection methods converged to a single model, that is better performing than the known junk model. The converged model from AVM is:

SalePrice = -261649.86 + 50.32 * TotalSQFT + 1508.63 * QualityIndex + 49044.18 * KEx+ 42.99 * GarageArea + 10329.91 * TotalBath - 41069.62 * Duplex - 12825.83 * Tier4 +15042.46 * PartialSaleCond - 37577.71 * Tier3 + 0.43 * LotArea + 5455.35 * CulDSac +127.75 * YearRemodel - 21364.26 * Twnhs - 55081.8 * Tier1 - 48981.04 * Tier2 - 17106.01 *TwnhsE - 4022.15* CornerLot - 8585.89 * Frontal2

2. Sample definition

It is assumed that typical home buyers are those that move from apartments to single family or town homes. Also apartments are less likely to be sold to individuals as they remain holdings of owners for rental income. Single family and town homes belong to "Residential Low density" (RL) zoning classification in the city of Ames. Data belonging to only to the RL zone is considered for analysis and model development. Also, it is assumed that typical homes have paved streets for access and above grade living area greater than 800 square feet. Sales data belonging to homes that were sold in abnormal conditions such as trade in, foreclosure or short sale are not included in the analysis. Also, sales between family members, sale of adjoining lot, linked properties are omitted from the data. Homes with no basements are excluded from the analysis at this time. Homes with living area greater than 4000 square feet and garage area greater than 1000 square feet were identified as outliers and are therefore removed from the analysis. Table 1 shows the waterfall of the data not included in the data and the eligible samples.

Table 1: Drop waterfall

DropCondition	counts
01: Not LowDensityZone	657
02: Not Normal/Partial Sale	189
03: Street Not Paved	3
04: Less than 800 SqFt	41
05: No Basement	48
06: Greater 4000 sqft living Area - Influence Points	4
07:Garage area greater than 1000 sqft - Influence points	23
99: Eligible Sample	1965

2.1 Predictor variables of interest for modelling

The following variables in the data were deemed to be of interest for model building. The choice of parameters was based upon intial Exploratory Data Analysis (EDA) and subject matter expertise. The categorical variables are coded into indicator variables for model building purposes, the tables 3, 4, 5 and 6 show the

description of the indicator variables.

Table 2: Predictors of interest

LotArea	BsmtFullBath	MoSold
LotConfig	BsmtHalfBath	YrSold
Neighborhood	FullBath	SaleCondition
BldgType	HalfBath	FirstFlrSF
OverallCond	BedroomAbvGr	SecondFlrSF
YearRemodel	KitchenQual	OverallQual
TotalBsmtSF	TotRmsAbvGrd	
GrLivArea	GarageArea	

The following tables describe the indicator variables:

Table 3: Neighborhood tiers,base category > 90

Tier	Price.per.sq.ft
1	<= 60
2	> 60 and <= 70
3	> 70 and <= 80
4	> 80 and <= 90

Table 4: Lot configuration indicator variables; base category: Inside Lot

Indicator	Description
CornerLot	Corner lot
CulDSac	CulDSac Lot
Frontal2	2 frontal lot
Frontal3	3 frontal lot

Table 5: Building type indicator variables; base category: single family

Indicator	Decription
TwnhsE	Townhouse
Twnhs	Twin house
Duplex	Duplex
fam2	2 family conversion

Table 6: Kitchen Quality indicator variables; base category: poor

Indicator	Decription
KTA	Typical/Average
KGD	Good
KEx	Excellent
KFa	Fair

2.2 Training and validation samples.

From the eligible samples, 70% of the data is randomly sampled to be used as the dataset for model development. This dataset would be refered to as training dataset. The remaining 30% is used as the validation set to evaluate the model performance of predicting sale price on data that is outside the training set. Table 7 shows the split of the total eligible samples.

Table 7: Training and Validation sampling

Data	Samples
Training set	1361
Validation set	604
Total	1965

3. Model identification by Automated Variable Selection

Attribute variables such as "TotalSQFT"; sum of basement square feet and above grade living area and "QualityIndex"; product of overall quality and overall condition has been created for use as predictor variables in model building. With the attribute variables and the indicator variables created, below is a list of predictor variables that will be used for automated variable selection and modelling.

Table 8: Predictors for linear regression models

LotArea	QualityIndex	Tier3	TwnhsE
YearRemodel	TotRmsAbvGrd	Tier4	Twnhs
TotalBsmtSF	GarageArea	PartialSaleCond	Duplex
GrLivArea	YearMonthSold	CornerLot	KTA
TotalBath	Tier1	CulDSac	KGD
TotalSQFT	Tier2	Frontal2	KEx

3.1. Full model

A full model with all the predictors used is fit as an exhaustive search. This will be used as an upper boundry for automated variable selection processes; forward selection and as a starting condition in the backward selection process. The full model and its fit is shown below:

SalePrice = 151325.57 + 0.42 * LotArea + 119.01 * YearRemodel - 20.6 * TotalBsmtSF - 17.16 * GrLivArea + 10197.78 * TotalBath + 70.47 * TotalSQFT + 1483.8 * QualityIndex - 1150.48 * TotRmsAbvGrd + 42.26 * GarageArea - 194.66 * YearMonthSold - 54637.52 * Tier1 - 48465.07 * Tier2 - 37275.13 * Tier3 - 12577.95 * Tier4 + 14991.32 * PartialSaleCond - 4021.03 * CornerLot + 5459.61 * CulDSac - 8401.33 * Frontal2 - 17960.26 * TwnhsE - 22043.3 * Twnhs - 39290.35 * Duplex - 721.05 * KTA + 605.01 * KGD + 50320.13 * KEx

```
##
## Call:
## lm(formula = SalePrice ~ ., data = train.Clean)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -147763  -13411    -410   12988  182406
##
## Coefficients:
```

```
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      1.513e+05  1.092e+06   0.139 0.889835
## LotArea          4.233e-01  1.117e-01   3.788 0.000159 ***
## YearRemodel      1.190e+02  5.419e+01   2.196 0.028260 *
## TotalBsmtSF     -2.060e+01  1.890e+01  -1.090 0.275824
## GrLivArea       -1.716e+01  1.862e+01  -0.922 0.356919
## TotalBath        1.020e+04  1.122e+03   9.089  < 2e-16 ***
## TotalSQFT        7.047e+01  1.871e+01   3.767 0.000173 ***
## QualityIndex     1.484e+03  1.028e+02  14.433  < 2e-16 ***
## TotRmsAbvGrd    -1.150e+03  8.497e+02  -1.354 0.175945
## GarageArea       4.226e+01  4.988e+00   8.473  < 2e-16 ***
## YearMonthSold   -1.947e+02  5.443e+02  -0.358 0.720651
## Tier1           -5.464e+04  5.775e+03  -9.461  < 2e-16 ***
## Tier2           -4.847e+04  5.234e+03  -9.260  < 2e-16 ***
## Tier3           -3.728e+04  5.002e+03  -7.452 1.64e-13 ***
## Tier4           -1.258e+04  4.900e+03  -2.567 0.010375 *
## PartialSaleCond  1.499e+04  2.746e+03   5.459 5.70e-08 ***
## CornerLot       -4.021e+03  1.868e+03  -2.153 0.031502 *
## CulDSac          5.460e+03  2.626e+03   2.079 0.037814 *
## Frontal2        -8.401e+03  4.210e+03  -1.995 0.046200 *
## TwnhsE          -1.796e+04  3.432e+03  -5.233 1.94e-07 ***
## Twnhs           -2.204e+04  5.766e+03  -3.823 0.000138 ***
## Duplex          -3.929e+04  4.674e+03  -8.406  < 2e-16 ***
## KTA             -7.211e+02  5.223e+03  -0.138 0.890211
## KGD              6.050e+02  5.557e+03   0.109 0.913315
## KEx              5.032e+04  6.458e+03   7.792 1.31e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 25440 on 1336 degrees of freedom
## Multiple R-squared:  0.8956, Adjusted R-squared:  0.8938
## F-statistic: 477.8 on 24 and 1336 DF,  p-value: < 2.2e-16
```

3.2. Intercept only model as start for forward variable selection

An intercept only model is used as a lower boundary (no predictors used) condition for forward selection.

$$\text{Sale Price} = 197014.67$$

```
##
## Call:
## lm(formula = SalePrice ~ 1, data = train.Clean)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -138515  -54915  -20015   31485  427985
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   197015       2116    93.1   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 78070 on 1360 degrees of freedom
```

3.3. Simple linear regression as starting condition for stepwise variable selection

A simple linear regression model with total square feet as predictor is used as a starting point for stepwise variable selection.Below is the model and fit:

$$\text{SalePrice} = -46861.39 + 90.56 * \text{TotalSQFT}$$

```
##
## Call:
## lm(formula = SalePrice ~ TotalSQFT, data = train.Clean)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -202123  -20170    1665   21187  233417
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -46861.39    4461.97  -10.50   <2e-16 ***
## TotalSQFT       90.56       1.60   56.59   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 42630 on 1359 degrees of freedom
## Multiple R-squared:  0.702,  Adjusted R-squared:  0.7018
## F-statistic:  3202 on 1 and 1359 DF,  p-value: < 2.2e-16
```

3.4. Forward variable selection.

A forward selection method is used for selecting variables for linear regression fit. The following model was fit by the automated forward selection by AIC.

SalePrice = -261649.86 + 50.32 * TotalSQFT + 1508.63 * QualityIndex + 49044.18 * KEx + 42.99 * GarageArea + 10329.91 * TotalBath - 41069.62 * Duplex - 12825.83 * Tier4 + 15042.46 * PartialSaleCond - 37577.71 * Tier3 + 0.43 * LotArea + 5455.35 * CulDSac + 127.75 * YearRemodel - 21364.26 * Twnhs - 55081.8 * Tier1 - 48981.04 * Tier2 - 17106.01 * TwnhsE - 4022.15 * CornerLot - 8585.89 * Frontal2

```
##
## Call:
## lm(formula = SalePrice ~ TotalSQFT + QualityIndex + KEx + GarageArea +
##     TotalBath + Duplex + Tier4 + PartialSaleCond + Tier3 + LotArea +
##     CulDSac + YearRemodel + Twnhs + Tier1 + Tier2 + TwnhsE +
##     CornerLot + Frontal2, data = train.Clean)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -145734  -13679    -489   12996  182251
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -2.616e+05  9.790e+04  -2.673 0.007615 **
## TotalSQFT        5.032e+01  1.471e+00  34.211  < 2e-16 ***
## QualityIndex     1.509e+03  1.005e+02  15.008  < 2e-16 ***
## KEx              4.904e+04  3.077e+03  15.941  < 2e-16 ***
## GarageArea       4.299e+01  4.910e+00   8.755  < 2e-16 ***
```

```
## TotalBath        1.033e+04  1.013e+03  10.196  < 2e-16 ***
## Duplex          -4.107e+04  4.484e+03  -9.158  < 2e-16 ***
## Tier4           -1.283e+04  4.884e+03  -2.626 0.008739 **
## PartialSaleCond  1.504e+04  2.669e+03   5.637 2.11e-08 ***
## Tier3           -3.758e+04  4.992e+03  -7.527 9.47e-14 ***
## LotArea          4.299e-01  1.114e-01   3.859 0.000119 ***
## CulDSac          5.455e+03  2.615e+03   2.086 0.037184 *
## YearRemodel      1.278e+02  4.985e+01   2.563 0.010490 *
## Twnhs           -2.136e+04  5.722e+03  -3.734 0.000197 ***
## Tier1           -5.508e+04  5.741e+03  -9.595  < 2e-16 ***
## Tier2           -4.898e+04  5.202e+03  -9.416  < 2e-16 ***
## TwnhsE          -1.711e+04  3.320e+03  -5.153 2.95e-07 ***
## CornerLot       -4.022e+03  1.863e+03  -2.159 0.031018 *
## Frontal2        -8.586e+03  4.199e+03  -2.045 0.041098 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 25420 on 1342 degrees of freedom
## Multiple R-squared:  0.8954, Adjusted R-squared:  0.894
## F-statistic:   638 on 18 and 1342 DF,  p-value: < 2.2e-16
```

3.5 Backward elimination selection.

The following model is a result of a backward elimination selection, with full model as its starting model.

SalePrice = -261649.86 + 0.43 * LotArea + 127.75 * YearRemodel + 10329.91 * TotalBath + 50.32 * TotalSQFT + 1508.63 * QualityIndex + 42.99 * GarageArea - 55081.8 * Tier1 - 48981.04 * Tier2 - 37577.71 * Tier3 - 12825.83 * Tier4 + 15042.46 * PartialSaleCond - 4022.15 * CornerLot + 5455.35 * CulDSac - 8585.89 * Frontal2 - 17106.01 * TwnhsE - 21364.26 * Twnhs - 41069.62 * Duplex + 49044.18 * KEx

It is seen that the backward elimination selection has converged to the same model as the forward selection variable selection.

```
##
## Call:
## lm(formula = SalePrice ~ LotArea + YearRemodel + TotalBath +
##     TotalSQFT + QualityIndex + GarageArea + Tier1 + Tier2 + Tier3 +
##     Tier4 + PartialSaleCond + CornerLot + CulDSac + Frontal2 +
##     TwnhsE + Twnhs + Duplex + KEx, data = train.Clean)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -145734  -13679    -489   12996  182251
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -2.616e+05  9.790e+04  -2.673 0.007615 **
## LotArea          4.299e-01  1.114e-01   3.859 0.000119 ***
## YearRemodel      1.278e+02  4.985e+01   2.563 0.010490 *
## TotalBath        1.033e+04  1.013e+03  10.196  < 2e-16 ***
## TotalSQFT        5.032e+01  1.471e+00  34.211  < 2e-16 ***
## QualityIndex     1.509e+03  1.005e+02  15.008  < 2e-16 ***
## GarageArea       4.299e+01  4.910e+00   8.755  < 2e-16 ***
## Tier1           -5.508e+04  5.741e+03  -9.595  < 2e-16 ***
```

```
## Tier2           -4.898e+04  5.202e+03  -9.416  < 2e-16 ***
## Tier3           -3.758e+04  4.992e+03  -7.527 9.47e-14 ***
## Tier4           -1.283e+04  4.884e+03  -2.626 0.008739 **
## PartialSaleCond  1.504e+04  2.669e+03   5.637 2.11e-08 ***
## CornerLot       -4.022e+03  1.863e+03  -2.159 0.031018 *
## CulDSac          5.455e+03  2.615e+03   2.086 0.037184 *
## Frontal2        -8.586e+03  4.199e+03  -2.045 0.041098 *
## TwnhsE          -1.711e+04  3.320e+03  -5.153 2.95e-07 ***
## Twnhs           -2.136e+04  5.722e+03  -3.734 0.000197 ***
## Duplex          -4.107e+04  4.484e+03  -9.158  < 2e-16 ***
## KEx              4.904e+04  3.077e+03  15.941  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 25420 on 1342 degrees of freedom
## Multiple R-squared:  0.8954, Adjusted R-squared:  0.894
## F-statistic:   638 on 18 and 1342 DF,  p-value: < 2.2e-16
```

3.6. Stepwise regression method.

The following model is a result of stepwise variable selection with full model and simple linear regression model with Total square feet as a predictor as boundary conditions.

SalePrice = -261649.86 + 50.32 * TotalSQFT + 1508.63 * QualityIndex + 49044.18 * KEx + 42.99 * GarageArea + 10329.91 * TotalBath - 41069.62 * Duplex - 12825.83 * Tier4 + 15042.46 * PartialSaleCond - 37577.71 * Tier3 + 0.43 * LotArea + 5455.35 * CulDSac + 127.75 * YearRemodel - 21364.26 * Twnhs - 55081.8 * Tier1 - 48981.04 * Tier2 - 17106.01 * TwnhsE - 4022.15 * CornerLot - 8585.89 * Frontal2

Like the forward selection and backward elimination methods, the stepwise regression method has converged to the same model.

```
##
## Call:
## lm(formula = SalePrice ~ TotalSQFT + QualityIndex + KEx + GarageArea +
##     TotalBath + Duplex + Tier4 + PartialSaleCond + Tier3 + LotArea +
##     CulDSac + YearRemodel + Twnhs + Tier1 + Tier2 + TwnhsE +
##     CornerLot + Frontal2, data = train.Clean)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -145734  -13679    -489   12996  182251
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -2.616e+05  9.790e+04  -2.673 0.007615 **
## TotalSQFT        5.032e+01  1.471e+00  34.211  < 2e-16 ***
## QualityIndex     1.509e+03  1.005e+02  15.008  < 2e-16 ***
## KEx              4.904e+04  3.077e+03  15.941  < 2e-16 ***
## GarageArea       4.299e+01  4.910e+00   8.755  < 2e-16 ***
## TotalBath        1.033e+04  1.013e+03  10.196  < 2e-16 ***
## Duplex          -4.107e+04  4.484e+03  -9.158  < 2e-16 ***
## Tier4           -1.283e+04  4.884e+03  -2.626 0.008739 **
## PartialSaleCond  1.504e+04  2.669e+03   5.637 2.11e-08 ***
## Tier3           -3.758e+04  4.992e+03  -7.527 9.47e-14 ***
```

```
## LotArea          4.299e-01  1.114e-01   3.859 0.000119 ***
## CulDSac          5.455e+03  2.615e+03   2.086 0.037184 *
## YearRemodel      1.278e+02  4.985e+01   2.563 0.010490 *
## Twnhs           -2.136e+04  5.722e+03  -3.734 0.000197 ***
## Tier1           -5.508e+04  5.741e+03  -9.595  < 2e-16 ***
## Tier2           -4.898e+04  5.202e+03  -9.416  < 2e-16 ***
## TwnhsE          -1.711e+04  3.320e+03  -5.153 2.95e-07 ***
## CornerLot       -4.022e+03  1.863e+03  -2.159 0.031018 *
## Frontal2        -8.586e+03  4.199e+03  -2.045 0.041098 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 25420 on 1342 degrees of freedom
## Multiple R-squared:  0.8954, Adjusted R-squared:  0.894
## F-statistic:   638 on 18 and 1342 DF,  p-value: < 2.2e-16
```

3.7. Junk Model

The attempt here is to verify the intuition of how correlated predictor variables perform. And how they compare to models from automated variable selection. The predictors chosen are OverallQual,OverallCond, QualityIndex, GrLivArea and TotalSQFT. QualityIndex is a linear combination of OverallQual and OverllCond. TotalSQFT is a result of a linear combination of variables that include GrLivArea. Below is the junk model and its fit.

SalePrice = -271281.8 + 45081.02 * OverallQual + 25737.81 * OverallCond - 3535.76 * QualityIndex + 11.67 * GrLivArea + 53.99 * TotalSQFT

```
##
## Call:
## lm(formula = SalePrice ~ OverallQual + OverallCond + QualityIndex +
##     GrLivArea + TotalSQFT, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -101671  -18670    -835   17134  226610
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -2.713e+05  2.202e+04 -12.317  < 2e-16 ***
## OverallQual   4.508e+04  3.683e+03  12.241  < 2e-16 ***
## OverallCond   2.574e+04  4.105e+03   6.270 4.85e-10 ***
## QualityIndex -3.536e+03  6.961e+02  -5.079 4.32e-07 ***
## GrLivArea     1.168e+01  3.684e+00   3.169  0.00156 **
## TotalSQFT     5.399e+01  2.657e+00  20.319  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 32080 on 1355 degrees of freedom
## Multiple R-squared:  0.8318, Adjusted R-squared:  0.8312
## F-statistic:  1340 on 5 and 1355 DF,  p-value: < 2.2e-16
```

Table 9: VIF - Top 5 by model

Model	Predictors	VIF
Junk	QualityIndex	43.72
	OverallQual	31.72
	OverallCond	23.39
	TotalSQFT	4.87
	GrLivArea	4.14
Forward Selection	Tier2	13.56
	Tier3	11.79
	Tier4	7.10
	Tier1	4.68
Backward Elimination	TotalSQFT	2.38
	Tier2	13.56
	Tier3	11.79
	Tier4	7.10
	Tier1	4.68
Stepwise	TotalSQFT	2.38
	Tier2	13.56
	Tier3	11.79
	Tier4	7.10
	Tier1	4.68
	TotalSQFT	2.38

3.7.1 Variance Inflation Factors (VIF)

Having known that the predictors in the junk model are correlated, it would be interesting to compare correlation amongst predictors in the models from automated variable selection methods against that of the junk model. Table 9 shows the top 5 highest VIFs by models.

It is not surprising that the highest VIFs are from the junk model. However we see neighborhood Tiers 2 and 3 have high VIF (> 10). This is not surprising either as they are indicator(dummy) variables. The indicator variables are correlated amongst themselves and can be modelled as linear combinations of each other. Thereby yielding a high coefficient of determintion when indicator variables are regressed by other predictors, causing an inflated VIF.

4. Model Comparison

The four models (Junk, Forward selection, Backward Elimination ans Stepwise regression) are compared below. It is seen that the model equations for Forward Selection, Backward Elimination and Stepwise regression are the same. The automated variable selections have converged to the same model.

```
##
## ---------------------------------------
## Models   Equation
## -------- ------------------------------
## Junk     SalePrice = -271281.8 +
##          45081.02 * OverallQual +
##          25737.81 * OverallCond -
##          3535.76 * QualityIndex + 11.67
##          * GrLivArea + 53.99 *
##          TotalSQFT
##
```

```
## Forward  SalePrice = -261649.86 + 50.32
##          * TotalSQFT + 1508.63 *
##          QualityIndex + 49044.18 * KEx
##          + 42.99 * GarageArea +
##          10329.91 * TotalBath -
##          41069.62 * Duplex - 12825.83 *
##          Tier4 + 15042.46 *
##          PartialSaleCond - 37577.71 *
##          Tier3 + 0.43 * LotArea +
##          5455.35 * CulDSac + 127.75 *
##          YearRemodel - 21364.26 * Twnhs
##          - 55081.8 * Tier1 - 48981.04 *
##          Tier2 - 17106.01 * TwnhsE -
##          4022.15 * CornerLot - 8585.89
##          * Frontal2
##
## Backward SalePrice = -261649.86 + 0.43
##          * LotArea + 127.75 *
##          YearRemodel + 10329.91 *
##          TotalBath + 50.32 * TotalSQFT
##          + 1508.63 * QualityIndex +
##          42.99 * GarageArea - 55081.8 *
##          Tier1 - 48981.04 * Tier2 -
##          37577.71 * Tier3 - 12825.83 *
##          Tier4 + 15042.46 *
##          PartialSaleCond - 4022.15 *
##          CornerLot + 5455.35 * CulDSac
##          - 8585.89 * Frontal2 -
##          17106.01 * TwnhsE - 21364.26 *
##          Twnhs - 41069.62 * Duplex +
##          49044.18 * KEx
##
## Stepwise SalePrice = -261649.86 + 50.32
##          * TotalSQFT + 1508.63 *
##          QualityIndex + 49044.18 * KEx
##          + 42.99 * GarageArea +
##          10329.91 * TotalBath -
##          41069.62 * Duplex - 12825.83 *
##          Tier4 + 15042.46 *
##          PartialSaleCond - 37577.71 *
##          Tier3 + 0.43 * LotArea +
##          5455.35 * CulDSac + 127.75 *
##          YearRemodel - 21364.26 * Twnhs
##          - 55081.8 * Tier1 - 48981.04 *
##          Tier2 - 17106.01 * TwnhsE -
##          4022.15 * CornerLot - 8585.89
##          * Frontal2
## ---------------------------------------
##
## Table: Model equations
```

Table 10 shows the metrics adjusted R squared, AIC, BIC. MSE and MAE. Table 11 ranks the metrics. We see that the ranks are idetical for every column i.e. rank based on MSE is same as rank based on R squared, AIC, BIC or MAE. This is expected as the "in-sample" MSE, Adjusted R squared, AIC and BIC are

dependent on sum of squared residuals (SS Residuals). The in-sample MAE is dependant on raw residuals, which the SS Residuals is inturn dependendant upon.

Table 10: Model comparison

model	adj.r.squared	AIC	BIC	MSE	MAE
Junk	0.8311830	32113.45	32149.96	1024345098	23439.56
Forward Selection	0.8939616	31493.45	31597.77	637245425	17626.98
Backward Elimination	0.8939616	31493.45	31597.77	637245425	17626.98
Stepwise	0.8939616	31493.45	31597.77	637245425	17626.98

Table 11: Model Ranks

model	Rank.adjR	Rank.AIC	Rank.BIC	Rank.MSE	Rank.MAE
Junk	2	2	2	2	2
Forward Selection	1	1	1	1	1
Backward Elimination	1	1	1	1	1
Stepwise	1	1	1	1	1

4. Predictive Accuracy

Table 12 shows the MAE and MSE of both "in sample" (training dataset) and "out of sample" (validation data set) data. It is seen that the Junk model performs poorly compared to the automated variable selection model (recall that the forward selection, backward elimination and stepwise all converged into the same model) in both in sample and out of sample. The MSE and MAE metrics both convey the same message. The MSE is an inflated measure of the MAE.

$$MAE = \frac{\sum_{i=1}^{n} |y_i - \hat{y_i}|}{n}$$

$$MSE = \frac{\sum_{i=1}^{n} (y_i - \hat{y_i})^2}{n}$$

While there is no statistical implication of choosing MAE over MSE or vice versa, MAE is easily understood and in the same units as of the response variable; in this case US dollars, unlike MSE which is square of US dollars.

It is also seen that the In sample metrics are better than the out of sample metrics in Table 12. When in sample MAE and MSE is much lower than the out sample MAE and MSE, then it is a sign of model being overfitted for the in-sample data or training data. The model was probably over tailored to the in-sample points. However in this case, the in-sample and out of sample MAEs are within $1000.

Table 12: Prediction metrics of models

Model	MAE.Out.of.Sample	MAE.In.Sample	MSE.Out.of.Sample	MSE.In.Sample
Junk	22448.06	23439.56	882406444	1024345098
Forward	18285.26	17626.98	626000090	637245425
Backward	18285.26	17626.98	626000090	637245425
Stepwise	18285.26	17626.98	626000090	637245425

5. Operational Validation.

The models are graded to see if they are fit for practical purposes. The MAE of Automated variable selection model (forward/backward/stepwise) is $18,285. While this may be a high for "typical" homes in Ames, it would be useful to see how many instances does the model predict more close closer to the actual sale price. It'd be useful to grade models depending on how close they predict the actual sale price.

If the model predicts sale price of a house within 10% of its actual sale price, then its graded 1. If the prediction is within 15% (>10% and <= 15%), then graded 2; if the prediction is within 25% of the actual sale price then graded 3. If greater than 25%, then graded 4. Table 13 shows the distribution of grades of junk model and forward selection model. Since all the automated variable selection methods converged to one model, we'd compare only the forward model and junk model.

Table 13: Operational Validation

Grade	forward.train.grade	forward.test.grade	junk.train.grade	junk.test.grade
Grade 1: [0.0, 0.1]	65%	62%	52%	48%
Grade 2: [0.10,0.15]	16%	17%	17%	24%
Grade 3: [0.15, 0.25]	14%	16%	19%	19%
Grade 4: [0.25 +]	5%	5%	12%	9%

It is seen that the Automated Variable selection Model (AVM) performs better than the junk model. The model performs at grade 1 over 60% of the instances of prediction. Also from table 12; it is seen that the in-sample and out of sample prediciton MAEs for AVM and junk models are close within $1000 (out of sample MAE $1000 greater than in sample MAE). The MAE of AVM is better than Junk by approximately $4000. This is consistent with the grading result.

6. Conclusion

It was found that the automated variable selection methods converged to the same model, indicating a much stable model. The choice of preictors played a key role in the stability. Also the performance of the model from automated variable selection was compared with a known junk model with collinear predictors. It was seen that the junk model performed relatively poorly.

The MAE of the automated variable selection model; both in-sample and out of sample were within $1000 dollars of each other. However, the MAE were in the neighborhood of $18,000 dollars, which is more than the desired $10,000. However it is found that the automaterd variable selection model predicts within 10% of the actual sale price over 60% of predition instances. Which makes the model good enough for application.

APPENDIX

A.1 R code

```
knitr::opts_chunk$set(echo = F,tidy.opts=list(width.cutoff=70),tidy=TRUE)
ames <- readr::read_delim(file = 'ames_housing_data.csv', delim = ",")
# chamge from scientic notations, to restore to default options(scipen = 0)
options(scipen = 999)
library(magrittr)
LivingAreaCutoff <- 800
# Adding drop conditions varaible
# insert dummy variable to code SaleCondition being either equal to 'Normal' or 'Partial'
ames$Sale_NrmPar <- ifelse(ames$SaleCondition == 'Normal' | ames$SaleCondition == 'Partial', 1,0)
ames$DropCondition <- ifelse(ames$Zoning!='RL','01: Not LowDensityZone',
    ifelse(ames$Sale_NrmPar == 0,'02: Not Normal/Partial Sale',
    ifelse(ames$Street!='Pave','03: Street Not Paved',
    ifelse(ames$GrLivArea <LivingAreaCutoff,'04: Less than 800 SqFt',
     ifelse(ames$TotalBsmtSF <1,'05: No Basement',
     ifelse(ames$GrLivArea > 4000, '06: Greater 4000 sqft living Area - Influence Points',
ifelse(ames$GarageArea > 1000, '07:Garage area greater than 1000 sqft - Influence points',
    '99: Eligible Sample')
    ))))))

# Waterfall
waterfall <- ames %>%
  dplyr::group_by(DropCondition) %>%
  dplyr::summarise(counts=n())

# Print waterfall table
knitr::kable(waterfall, align = c("l", "r"),caption = "Drop waterfall")
# Define training portion of the data
trainPercent <- round(0.7,1)
# Columns of interest
colsofinterest <- c('SID'
  ,'LotArea'
, 'LotConfig'
, 'Neighborhood'
, 'BldgType'
, 'OverallCond'
, 'YearRemodel'
, 'TotalBsmtSF'
, 'GrLivArea'
, 'BsmtFullBath'
, 'BsmtHalfBath'
, 'FullBath'
, 'HalfBath'
, 'BedroomAbvGr'
, 'KitchenQual'
, 'TotRmsAbvGrd'
, 'GarageArea'
, 'MoSold'
, 'YrSold'
, 'SaleCondition'
```

```
, 'FirstFlrSF'
, 'SecondFlrSF'
, 'OverallQual'
, 'SalePrice')

# Cleanly show the columns of interest in pdf. Making the colsofinterest as matrix for easy printing
# printing on pdf
knitr::kable(matrix(c(colsofinterest[2:23],"",""),ncol = 3,byrow = F), caption = "Predictors of interes
tierdf <- data.frame(Tier = c(1,2,3,4), `Price per sq.ft` = c("<= 60", "> 60 and <= 70", "> 70 and <= 8
Lotconfigdf <- data.frame(Indicator = c("CornerLot", "CulDSac", "Frontal2", "Frontal3"), Description = 
Bldgtypedf <- data.frame(Indicator = c("TwnhsE","Twnhs","Duplex","fam2"),Decription = c("Townhouse", "Tw
KitchenQualdf <- data.frame(Indicator = c("KTA", "KGD", "KEx", "KFa"),Decription = c("Typical/Average",
Predictors <- c('LotArea'
                ,'YearRemodel'
                , 'TotalBsmtST'
                , 'GrLivArea'
                , 'TotalBath'
                , 'TotalSQFT'
                , 'QualityIndex'
                , 'TotRmsAbvGrd'
                , 'GarageArea'
                , 'YearMonthSold'
                , 'Tier1'
                , 'Tier2'
                , 'Tier3'
                , 'Tier4'
                , 'Tier5'
                , 'PartialSaleCond'
                , 'CornerLot'
                , 'CulDSac'
                , 'Frontal2'
                , 'Frontal3'
                , 'TwnhsE'
                , 'Twnhs'
                , 'Duplex'
                , 'fam2'
                , 'KTA'
                , 'KGD'
                , 'KEx'
                , 'KFa'
)
knitr::kable(tierdf, caption = "Neighborhood tiers,base category > 90")
knitr::kable(Lotconfigdf, caption = "Lot configuration indicator variables; base category: Inside Lot")
knitr::kable(Bldgtypedf, caption = "Building type indicator variables; base category: single family")
knitr::kable(KitchenQualdf, caption = "Kitchen Quality indicator variables; base category: poor")
# Get sample frame.
set.seed(500)
SampleFrame <- ames %>%
  dplyr::filter(DropCondition == '99: Eligible Sample') %>%
  dplyr::select_(.dots = colsofinterest)
SampleFrame <- SampleFrame %>%
  dplyr::mutate(TotalBath = BsmtFullBath + BsmtHalfBath + FullBath + HalfBath) %>%
  dplyr::mutate(TotalSQFT = TotalBsmtSF + FirstFlrSF + SecondFlrSF) %>%
```

```
  dplyr::mutate(SQFTNoBsmt = FirstFlrSF + SecondFlrSF) %>%
  dplyr::mutate(QualityIndex = OverallQual*OverallCond) %>%
  dplyr::mutate(YearMonthSold = YrSold + MoSold/100) %>%
  dplyr::mutate(u = runif(nrow(SampleFrame)))
# ----------------------------Grouping Neighborhoods-----------------------------------------------
NeighborhoodGrps <- SampleFrame %>%
              dplyr:: select(Neighborhood,SalePrice,TotalSQFT) %>%
               dplyr::group_by(Neighborhood) %>%
              dplyr::summarise(TotalSalePrice = sum(SalePrice), Total.SQFT = sum(TotalSQFT),PricePerSqF

Less60Neigh <- NeighborhoodGrps$Neighborhood[which(NeighborhoodGrps$PricePerSqFt <= 60)]
Bet60_70 <-  NeighborhoodGrps$Neighborhood[which(NeighborhoodGrps$PricePerSqFt <= 70 & NeighborhoodGrps
Bet70_80 <-  NeighborhoodGrps$Neighborhood[which(NeighborhoodGrps$PricePerSqFt <= 80 & NeighborhoodGrps
Bet80_90 <-   NeighborhoodGrps$Neighborhood[which(NeighborhoodGrps$PricePerSqFt <= 90 & NeighborhoodGrp
Greater_90 <-   NeighborhoodGrps$Neighborhood[which(NeighborhoodGrps$PricePerSqFt > 90)]

SampleFrame$Tier1 <- ifelse(SampleFrame$Neighborhood %in% Less60Neigh,1,0)
SampleFrame$Tier2 <- ifelse(SampleFrame$Neighborhood %in% Bet60_70,1,0)
SampleFrame$Tier3 <- ifelse(SampleFrame$Neighborhood %in% Bet70_80, 1,0)
SampleFrame$Tier4 <- ifelse(SampleFrame$Neighborhood %in% Bet80_90, 1,0)
#SampleFrame$Tier5 <- ifelse(SampleFrame$Neighborhood %in% Greater_90, 1,0)

tierdf <- data.frame(Tier = c(1,2,3,4,5), `Price per sq.ft` = c("<= 60", "> 60 and <= 70", "> 70 and <=
#-------------------------------------------------------------------------------------------------

#-----------------------------Creating indicator variables ----------------------------------------
# IV for LotConfig, base Inside lot
SampleFrame$CornerLot <- ifelse(SampleFrame$LotConfig == "Corner",1,0)
SampleFrame$CulDSac <- ifelse(SampleFrame$LotConfig == "CulDSac",1,0)
SampleFrame$Frontal2 <- ifelse(SampleFrame$LotConfig == "FR2",1,0)
SampleFrame$Frontal3 <- ifelse(SampleFrame$LotConfig == "F3",1,0)
Lotconfigdf <- data.frame(Indicator = c("CornerLot", "CulDSac", "Frontal2", "Frontal3"), Description = c
# IV for BlgType, base "1Fam"
SampleFrame$TwnhsE <- ifelse(SampleFrame$BldgType == "TwnhsE",1,0)
SampleFrame$Twnhs <- ifelse(SampleFrame$BldgType == "Twnhs", 1,0)
SampleFrame$Duplex <- ifelse(SampleFrame$BldgType == "Duplex", 1,0)
SampleFrame$fam2 <- ifelse(SampleFrame$BldgType == "2famCon",1,0)
Bldgtypedf <- data.frame(Indicator = c("TwnhsE","Twnhs","Duplex","fam2"),Decription = c("Townhouse", "Tw
# IV for SaleCondition, bae "Normal"
SampleFrame$PartialSaleCond <- ifelse(SampleFrame$SaleCondition == "Partial",1,0)
# IV for KitchenQual
SampleFrame$KTA <- ifelse(SampleFrame$KitchenQual == "TA", 1,0)
SampleFrame$KGD <- ifelse(SampleFrame$KitchenQual == "Gd", 1,0)
SampleFrame$KEx <- ifelse(SampleFrame$KitchenQual == "Ex", 1,0)
SampleFrame$KFa <- ifelse(SampleFrame$KitchenQual == "Fa", 1,0)
KitchenQualdf <- data.frame(Indicator = c("KTA", "KGD", "KEx", "KFa"),Decription = c("Typical/Average",
# -------------------------------------------------------------------------------------------------
Predictors <- c('LotArea'
                ,'YearRemodel'
                , 'TotalBsmtSF'
                , 'GrLivArea'
                , 'TotalBath'
                , 'TotalSQFT'
```

```
                , 'QualityIndex'
                , 'TotRmsAbvGrd'
                , 'GarageArea'
                , 'YearMonthSold'
                , 'Tier1'
                , 'Tier2'
                , 'Tier3'
                , 'Tier4'
                , 'PartialSaleCond'
                , 'CornerLot'
                , 'CulDSac'
                , 'Frontal2'
                # , 'Frontal3'
                , 'TwnhsE'
                , 'Twnhs'
                , 'Duplex'
            #,'fam2'
                , 'KTA'
                , 'KGD'
                , 'KEx'
                # , 'KFa'
)
# training set
#train <- dplyr::sample_n(SampleFrame,size = trainPercent*nrow(SampleFrame), replace = F,set.seed(2000)

train <- subset(SampleFrame , u < 0.7)
train <- train %>% dplyr::arrange(SID)
train.Clean <- train %>% dplyr::select(c(Predictors,'SalePrice'))
# Validation set
#Validation <- dplyr::sample_n(SampleFrame,size = (1-trainPercent)*nrow(SampleFrame), replace = F, set.
Validation <- subset(SampleFrame, u > 0.7)
Validation <- Validation %>% dplyr::arrange(SID)
# Check row counts
df <- cbind(Data = c("Training set", "Validation set","Total"), Samples = c(nrow(train),nrow(Validation)
knitr::kable(df,align = c("l","r"),caption = "Training and Validation sampling")

knitr::kable(matrix(Predictors,ncol = 4), caption = "Predictors for linear regression models")
# upper model as full model
upper.lm <- lm(data = train.Clean,SalePrice ~ .)
coefs <- round(coefficients(upper.lm),2)
signs <- ifelse(sign(coefs)==1,"+", "-")
Betas <- paste(abs(coefs[2:length(coefs)]),"*",Predictors)
uppereqn <- paste("SalePrice = ",paste(coefs[1],paste(paste(signs[2:24], Betas),collapse = " ")))

options(scipen = 0)
summary(upper.lm)
# lower model as the intercept model
lower.lm <- lm(data = train.Clean, SalePrice ~ 1)
# print summary for InterceptModel
summary(lower.lm)
# model to initate a stepwise regression
sqft.lm <- lm(data = train.Clean, SalePrice ~ TotalSQFT)
coef_IM <- round(coef(sqft.lm),2)
```

```
signs_IM <- ifelse(sign(coef_IM)==1,"+","-")
Betas_IM <- paste(abs(coef_IM[2:length(coef_IM)]),"*","TotalSQFT")
lowereqn <- paste("SalePrice = ",paste(coef_IM[1],paste(paste(signs_IM[2:2], Betas_IM),collapse = " "))
 summary(sqft.lm)
# forward selection
forward.lm <- MASS::stepAIC(object = lower.lm,scope = list(upper = formula(upper.lm), lower = ~1), dire
coef_Fwd <- round(coef(forward.lm),2)
signs_Fwd <- ifelse(sign(coef_Fwd)==1,"+","-")
formula_Fwd <- as.character(formula(forward.lm))[3]
predictors_Fwd <- unlist(strsplit(formula_Fwd, split = "+",fixed = T))
Betas_Fwd <- paste(abs(coef_Fwd[2:length(coef_Fwd)]),"*",predictors_Fwd)
fwdeqn <- paste("SalePrice = ",paste(coef_Fwd[1],paste(paste(signs_Fwd[2:length(signs_Fwd)], Betas_Fwd)
options(scipen = 0)
summary.lm(forward.lm)
# Backward selection
backward.lm <- MASS::stepAIC(object = upper.lm, direction = "backward",trace = F)
coef_Bwd <- round(coef(backward.lm),2)
signs_Bwd <- ifelse(sign(coef_Bwd)==1,"+","-")
formula_Bwd <- as.character(formula(backward.lm))[3]
predictors_Bwd <- unlist(strsplit(formula_Bwd, split = "+",fixed = T))
Betas_Bwd <- paste(abs(coef_Bwd[2:length(coef_Bwd)]),"*",predictors_Bwd)
Bwdeqn <- paste("SalePrice = ",paste(coef_Bwd[1],paste(paste(signs_Bwd[2:length(signs_Bwd)], Betas_Bwd)
summary.lm(backward.lm)
# stepwise
stepwise.lm <- MASS::stepAIC(object = sqft.lm,scope = list(upper = formula(upper.lm), lower = ~1),direc
coef_Stp <- round(coef(stepwise.lm),2)
signs_Stp <- ifelse(sign(coef_Stp)==1,"+","-")
formula_Stp <- as.character(formula(stepwise.lm))[3]
predictors_Stp <- unlist(strsplit(formula_Stp, split = "+",fixed = T))
Betas_Stp <- paste(abs(coef_Stp[2:length(coef_Stp)]),"*",predictors_Stp)
Stpeqn <- paste("SalePrice = ",paste(coef_Stp[1],paste(paste(signs_Stp[2:length(signs_Stp)], Betas_Stp)
summary(stepwise.lm)
#junkmodel
junk.lm <- lm(SalePrice ~ OverallQual + OverallCond + QualityIndex + GrLivArea + TotalSQFT, data = trai
coef_Jnk <- round(coef(junk.lm),2)
signs_Jnk <- ifelse(sign(coef_Jnk)==1,"+","-")
formula_Jnk <- as.character(formula(junk.lm))[3]
predictors_Jnk <- unlist(strsplit(formula_Jnk, split = "+",fixed = T))
Betas_Jnk <- paste(abs(coef_Jnk[2:length(coef_Jnk)]),"*",predictors_Jnk)
Jnkeqn <- paste("SalePrice = ",paste(coef_Jnk[1],paste(paste(signs_Jnk[2:length(signs_Jnk)], Betas_Jnk)

summary(junk.lm)

# VIF

JunkVIF <- as.data.frame(sort(car::vif(junk.lm),decreasing = T))
JunkVIF$Model <- "Junk"
JunkVIF$Predictors <- row.names(JunkVIF)
colnames(JunkVIF) <- c("VIF","Model", "Predictors")

forwardVIF <- as.data.frame(sort(car::vif(forward.lm),decreasing = T))
forwardVIF$Model <- "Forward Selection"
forwardVIF$Predictors <- row.names(forwardVIF)
```

```
colnames(forwardVIF) <- c("VIF","Model", "Predictors")
backwardVIF <- as.data.frame(sort(car::vif(backward.lm),decreasing = T))
backwardVIF$Model <- "Backward Elimination"
backwardVIF$Predictors <- row.names(backwardVIF)
colnames(backwardVIF) <- c("VIF","Model", "Predictors")
stepwiseVIF <- as.data.frame(sort(car::vif(stepwise.lm),decreasing = T))
stepwiseVIF$Model <- "Stepwise"
stepwiseVIF$Predictors <- row.names(stepwiseVIF)
colnames(stepwiseVIF) <- c("VIF","Model", "Predictors")
VIF <- rbind(JunkVIF,forwardVIF,backwardVIF,stepwiseVIF,row.names = F)
VIF <- VIF[,c("Model","Predictors", "VIF")]
VIF$VIF <- round(VIF$VIF,2)

VIFReport <- VIF[1:nrow(VIF)-1,] %>% dplyr::group_by(Model) %>%
  dplyr::top_n(n =5,wt = VIF)
#colnames(VIF) <- c("Predictor","VIF", "Model")
knitr::kable(VIFReport,"latex",caption = "VIF - Top 5 by model") %>%
  kableExtra::kable_styling(latex_options = "striped") %>%
  kableExtra::collapse_rows(columns = 1)

models <- data.frame(Models = c("Junk","Forward","Backward", "Stepwise"), Equation = c(Jnkeqn,fwdeqn,Bwd
pander::pandoc.table(models, "Model equations",justify = "left",emphasixe.cols = 2)
junkmetrics <- broom::glance(junk.lm)
junkmodel <- broom::augment(junk.lm)
junkmetrics$MAE <- mean(abs(junkmodel$.resid))
junkmetrics$MSE <- mean(junkmodel$.resid ^2)

forwardmetrics <- broom::glance(forward.lm)
forwardmodel <- broom::augment(forward.lm)
forwardmetrics$MAE <- mean(abs(forwardmodel$.resid))
forwardmetrics$MSE <- mean(forwardmodel$.resid ^2)

backwardmetrics <- broom::glance(backward.lm)
backwardmodel <- broom::augment(backward.lm)
backwardmetrics$MAE <- mean(abs(backwardmodel$.resid))
backwardmetrics$MSE <- mean(backwardmodel$.resid ^2)

stepwisemetrics <- broom::glance(stepwise.lm)
stepwisemodel <- broom::augment(stepwise.lm)
stepwisemetrics$MAE <- mean(abs(stepwisemodel$.resid))
stepwisemetrics$MSE <- mean(stepwisemodel$.resid ^2)

comparison <- rbind(junkmetrics,forwardmetrics,backwardmetrics,stepwisemetrics)
comparison$model <- c("Junk", "Forward Selection", "Backward Elimination", "Stepwise")
comparison <- comparison %>%
  dplyr::select(model,adj.r.squared,AIC,BIC,MSE, MAE) %>%
  dplyr::mutate(BIC = round(BIC,2))
knitr::kable(comparison,caption = "Model comparison",longtable = T)
Ranktable <- comparison %>%
  dplyr::mutate(Rank.adjR = as.numeric(as.factor(-adj.r.squared)), Rank.AIC = as.numeric(as.factor(AIC))
  dplyr::select(model,Rank.adjR, Rank.AIC, Rank.BIC, Rank.MSE, Rank.MAE)
knitr::kable(Ranktable,caption = "Model Ranks",longtable = T)
junk.test <- predict(junk.lm,newdata = Validation)
```

```r
junktest<- data.frame(Validation$SalePrice, junk.test)
junktest<- junktest %>%
  dplyr::mutate(error = Validation.SalePrice -junk.test)

forward.test <- predict(object = forward.lm, newdata = Validation)
fwdtest<- data.frame(Validation$SalePrice, forward.test)
fwdtest<- fwdtest %>% dplyr::mutate(error = Validation.SalePrice - forward.test)

backward.test <- predict(object = backward.lm, newdata = Validation)
bwdtest<- data.frame(Validation$SalePrice, backward.test)
bwdtest<- bwdtest %>% dplyr::mutate(error = Validation.SalePrice - backward.test)

stepwise.test <- predict(object = stepwise.lm, newdata = Validation)
stptest<- data.frame(Validation$SalePrice, stepwise.test)
stptest<- fwdtest %>% dplyr::mutate(error = Validation.SalePrice - stepwise.test)

PredictCompare <- data.frame(Model = c("Junk", "Forward", "Backward", "Stepwise"), MAE = c(mean(abs(junk

PredictCompare <- cbind(PredictCompare,comparison$MAE, comparison$MSE)
PredictCompare <- PredictCompare[,c(1,2,4,3,5)]
colnames(PredictCompare) <- c("Model", "MAE.Out.of.Sample", "MAE.In.Sample","MSE.Out.of.Sample", "MSE.In

knitr::kable(PredictCompare, caption = "Prediction metrics of models",digits = 2) #%>%
#   kableExtra::add_header_above(c(" " = 1, "MAE" = 2, "MSE" = 2))
library(purrr)
library(scales)
getgrade <- function(x) {
  ifelse(x <= 0.1, 'Grade 1: [0.0, 0.1]',
         ifelse(x <= 0.15, 'Grade 2: [0.10,0.15]',
                ifelse(x <= 0.25, 'Grade 3: [0.15, 0.25]','Grade 4: [0.25 +]')))
}
# in sample prediction grade - forward
forward.error.pct <- abs(forward.lm$residuals)/train.Clean$SalePrice
forward.predict.grade <- getgrade(forward.error.pct)
forward.trainTable <- table(forward.predict.grade)
forward.train.grade <- round(forward.trainTable/sum(forward.trainTable),2)
# out of sample prediction grade - forward
fwd.predictError.pct <- abs(fwdtest$error)/fwdtest$Validation.SalePrice
fwd.predictValid.grade <- getgrade(fwd.predictError.pct)
forward.testTable <- table(fwd.predictValid.grade)
forward.test.grade <- round(forward.testTable/sum(forward.testTable),2)

# in sample prediction grade - Junk
junk.error.pct <- abs(junk.lm$residuals)/train$SalePrice
junk.predict.grade <- getgrade(junk.error.pct)
junk.trainTable <- table(junk.predict.grade)
junk.train.grade <- round(junk.trainTable/sum(junk.trainTable),2)

# out of sample prediction grade - junk
junk.predictError.pct <- abs(junktest$error)/junktest$Validation.SalePrice
junk.predictValid.grade <- getgrade(junk.predictError.pct )
junk.testTable <- table(junk.predictValid.grade)
junk.test.grade <- round(junk.testTable/sum(junk.testTable),2)
```

```
OperationalValidation <- as.data.frame(cbind(forward.train.grade,forward.test.grade,junk.train.grade,ju
name <- row.names(OperationalValidation)
OperationalValidation <- map_df(.x = OperationalValidation, .f = percent)
OperationalValidation$Grade <- name
OperationalValidation <- OperationalValidation[,c(5,1,2,3,4)]

knitr::kable(OperationalValidation, caption = "Operational Validation")
```