DSC 424 Advanced Data Analysis
By Dr. John McDonald

Predicting Sales Price of Houses

Analysis Using Multivariate Techniques

Prepared By:
Christian Craig | Nina Eskandari | Umair Chaanda | Vineet Dcunha

Introduction and Data Cleaning

Housing prices are an important reflection of the economy, and housing price ranges are of great interest for both buyers and sellers. The aim of our group project is the deep and thorough analysis of a very large Housing data set using several different multivariate techniques such as Regularized Regression, Principal Components Analysis (PCA), Correspondence Analysis (CA), Multiple Correspondence Analysis (MCA), and Linear Discriminant Analysis (LDA). Our main parameter of interest in this data set is “SalePrice”. We are going to analyze some of the important numerical and categorical variables in relation with “SalePrice”.

The dataset consists of features in various formats. It has numerical data such as prices and numbers of bathrooms/bedrooms/living rooms, as well as categorical features such as zone classifications for sale, which can be ‘Agricultural’, ‘Residential High Density’, ‘Residential Low Density’, ‘Residential Low Density Park’, etc.

The data set has 1460 rows and 81 columns and they consist of several different types such as numerical, ordinal, true categorical, month, year etc.

```
> nrow(housingTrain)          # Report number of rows in dataset
[1] 1460
> ncol(housingTrain)          # Report number of columns in dataset
[1] 81
```

Also shown below, are the class / type of variables. There are 38 numeric variables, and 42 categorical variables.

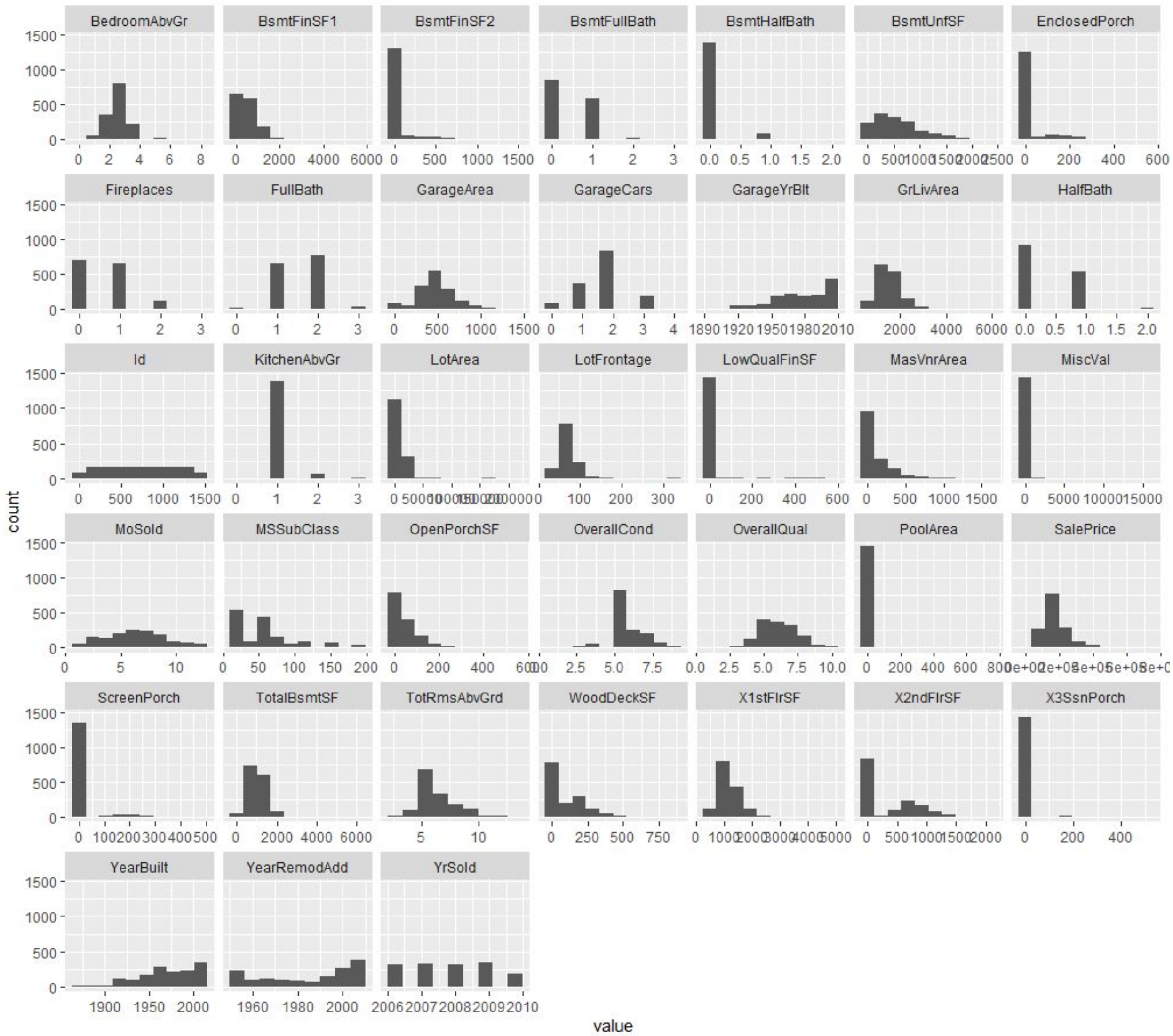
```
> sapply(housingTrain, class) # check type of variables"
           Id    MSSubClass      MSZoning   LotFrontage       LotArea        Street
    "integer"     "integer"      "factor"     "integer"     "integer"      "factor"
    LandSlope  Neighborhood    Condition1    Condition2      BldgType    HouseStyle
     "factor"      "factor"      "factor"      "factor"      "factor"      "factor"
     RoofMatl   Exterior1st   Exterior2nd    MasVnrType    MasVnrArea     ExterQual
     "factor"      "factor"      "factor"      "factor"     "integer"      "factor"
 BsmtFinType1    BsmtFinSF1  BsmtFinType2    BsmtFinSF2     BsmtUnfSF   TotalBsmtSF
     "factor"     "integer"      "factor"     "integer"     "integer"     "integer"
    X2ndFlrSF  LowQualFinSF     GrLivArea  BsmtFullBath  BsmtHalfBath      FullBath
    "integer"     "integer"     "integer"     "integer"     "integer"     "integer"
   Functional    Fireplaces   FireplaceQu    GarageType   GarageYrBlt  GarageFinish
     "factor"     "integer"      "factor"      "factor"     "integer"      "factor"
   WoodDeckSF   OpenPorchSF EnclosedPorch    X3SsnPorch   ScreenPorch      PoolArea
    "integer"     "integer"     "integer"     "integer"     "integer"     "integer"
       YrSold      SaleType SaleCondition     SalePrice
    "integer"      "factor"      "factor"     "integer"
```

```
      Alley      LotShape   LandContour     Utilities     LotConfig
   "factor"      "factor"      "factor"      "factor"      "factor"
OverallQual   OverallCond     YearBuilt  YearRemodAdd     RoofStyle
  "integer"     "integer"     "integer"     "integer"      "factor"
  ExterCond    Foundation      BsmtQual      BsmtCond  BsmtExposure
   "factor"      "factor"      "factor"      "factor"      "factor"
    Heating     HeatingQC    CentralAir    Electrical     X1stFlrSF
   "factor"      "factor"      "factor"      "factor"     "integer"
   HalfBath  BedroomAbvGr  KitchenAbvGr   KitchenQual  TotRmsAbvGrd
  "integer"     "integer"     "integer"      "factor"     "integer"
 GarageCars    GarageArea    GarageQual    GarageCond    PavedDrive
  "integer"     "integer"      "factor"      "factor"      "factor"
     PoolQC         Fence   MiscFeature       MiscVal        MoSold
   "factor"      "factor"      "factor"     "integer"     "integer"
```

Histogram Matrix - Numerical Variables

For numerical variables, we produced a histogram of each, looking for normality. If a variable is highly skewed, you should look at what kind of transformation (e.g. log, sqrt, etc.) can be used. First, we will select our numeric variables and display histograms, shown below:

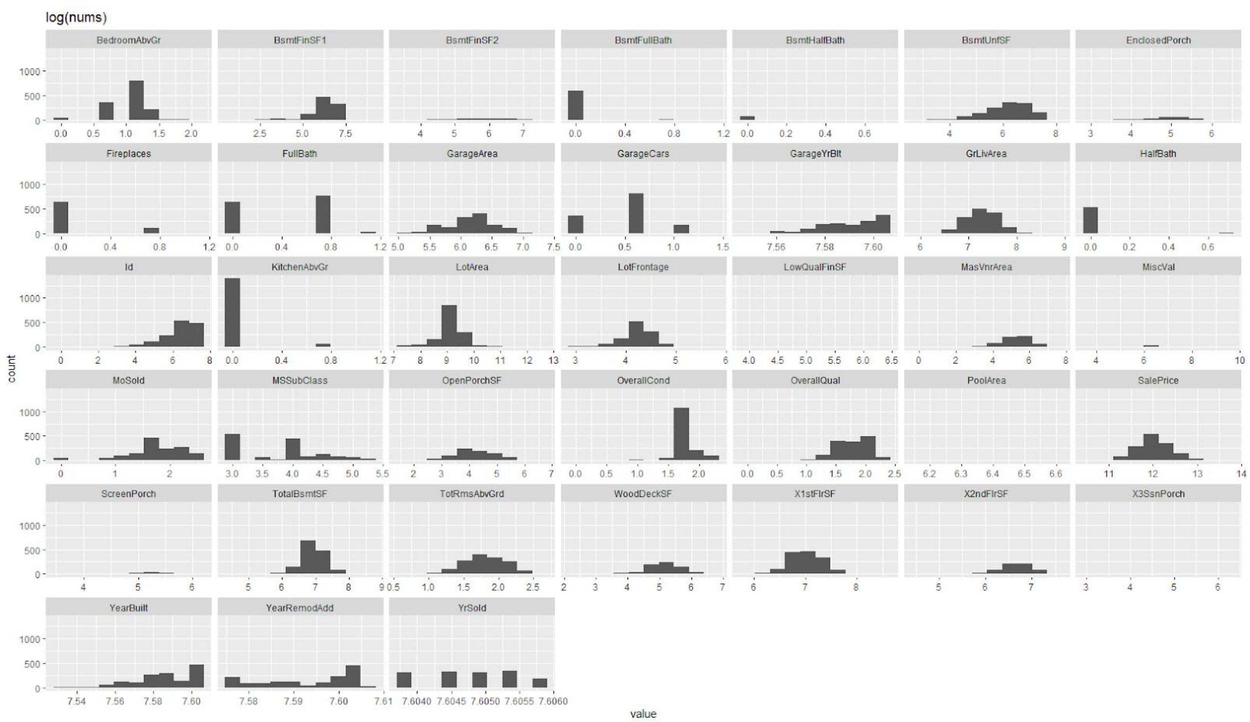
```
nums <- select_if(housing,is.numeric)
nums%>%gather()%>%head()

ggplot(gather(nums), aes(value)) +
  geom_histogram(bins = 10) +
  facet_wrap(~key, scales = 'free_x')
```



Looking at the Histogram Matrix above, there is a mix of variables which are heavily right skewed e.g. BedroomAbvGr, BsmtFinSf1, BsmtFinSf2, GrLivArea etc. Then there are few variables which are heavily left skewed e.g. GarageYrBlt, OverallCond. Finally, some of the variables do not look either heavily left or heavily right skewed e.g. GarageArea, MoSold, OverallQual etc. One thing to notice about our parameter of interest "SalePrice" is that it is slightly left skewed and might have to apply a log transformation to it. For the other variables which are skewed and if they are included in our final list of variables, we can apply either log or other type of transformation to it.

Log Transformation - Numerical Variables

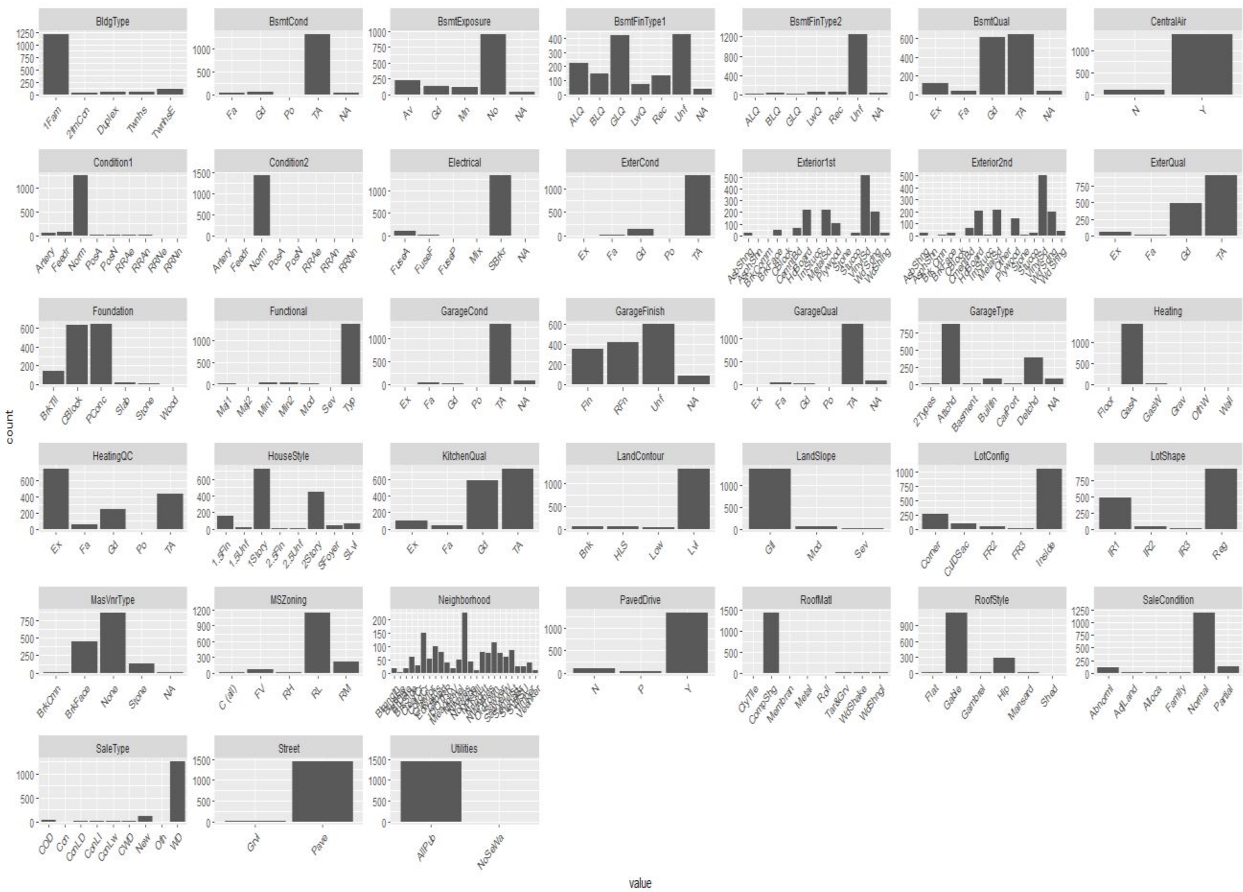
```
log.nums<-log(nums)

ggplot(gather(log.nums), aes(value)) +
  geom_histogram(bins = 10) +
  facet_wrap(~key, scales = 'free_x')+ggtitle("log(nums)")
```



Since we can create a compact histogram matrix, we decided to see what would happen if we took the log of all of our numerical attributes. As shown above, this significantly improved our SalePrice, which is our parameter of interest. It also evened out the distribution of a lot of our variables such as: GarageArea, LotFrontage, OpenPorchSF, GrlivArea, LotArea, etc. The log transformations somewhat helped the distribution of BsmtFinSF1, but instead skewed right distribution it is now skewed left. BsmtFinSF2's distribution is more normal, but also flattened. It is also worth noting that BedroomAbvGr when from skewed right to somewhat skewed left. As somewhat expected, the log transformation did not improve the distribution of variables with equally heavy tails. An example of this is YearRemodAdd.

Histogram Matrix - Categorical Variables

For categorical variables, we produced a histogram matrix to look for imbalance, so that we can start thinking about how we might deal with it when we get to apply multivariate techniques using categorical data.

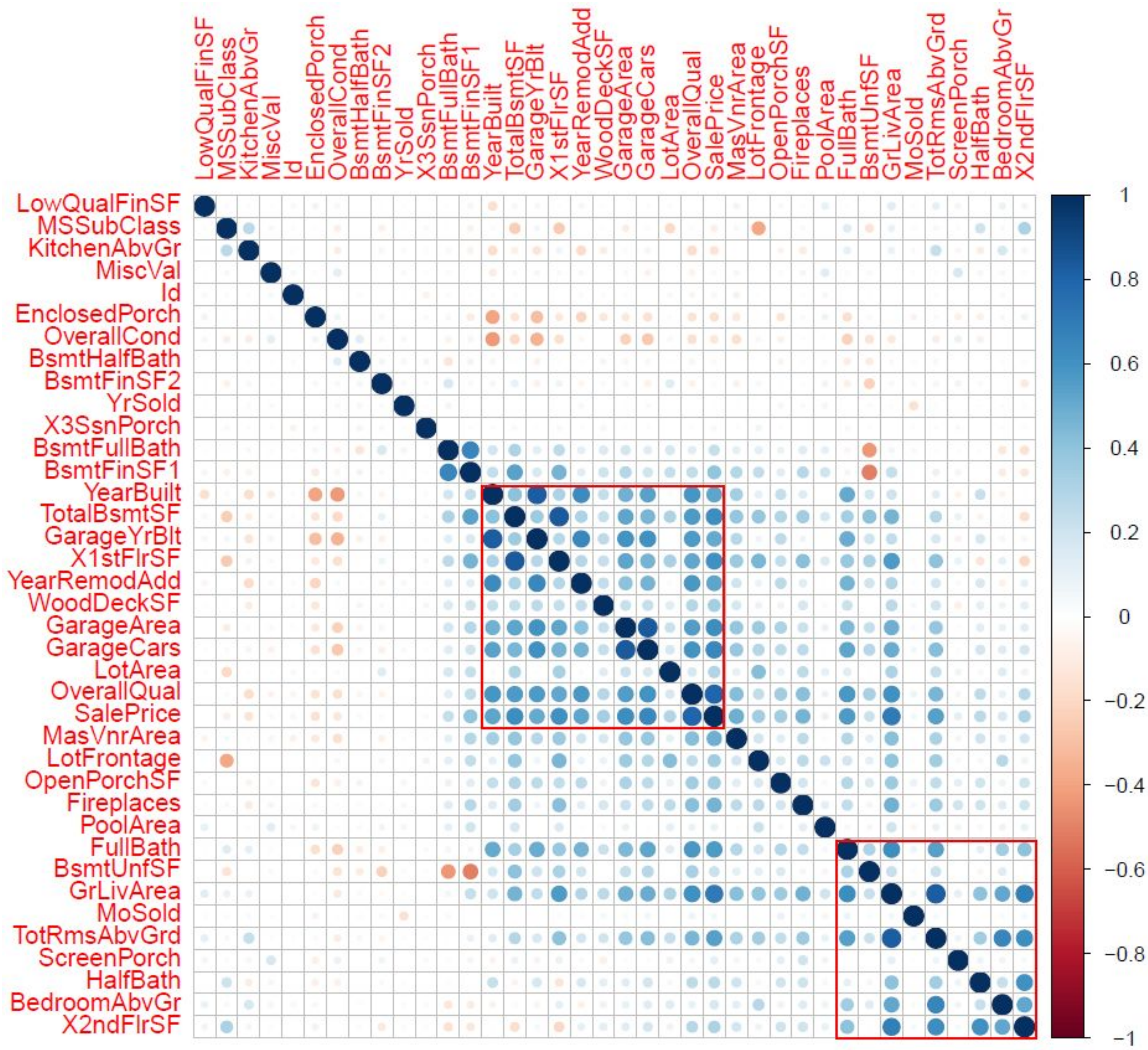


Above are the distributions of our categorical variables. There seems to be some imbalances amongst the variables that we will have to later address.

Correlation Analysis

A correlation matrix is a table showing correlation coefficients between sets of variables.

```
> # Compute the correlation matrix and visualize it
> cor.housing = cor(houseNum, use="complete.obs")
> corrplot(cor.housing, method="ellipse", order="AOE")
```



Following are the numeric variables which seems Strong Positively Correlated with our parameter of interest “SalePrice”: OverallQual, GrLivArea, TotalBsmtSf, X1stFlrSF, GarageArea, GarageCars.

Then there are some other variables which looks like moderately correlated with parameter of interest “SalePrice” in positive direction: Full Bath, TotRmsAbvGrd, YearBuilt, GarageYrBlt, YearRemodAdd, MasVnrArea, Fireplaces.

If we look at the grouping of variables there seems to be two distinct grouping (squared boxes) of variables which looks important for identifying multicollinearity and PCA/Factor analysis. The first big group is also a set of variables which we identified as strongly correlated with our parameter of interest “SalePrice”.

Techniques Covered

For the purpose of deep and thorough analysis of this large housing data set, we are going to explore and report following multivariate techniques in this project:

- Principal Components Analysis (PCA)
- Correspondence Analysis (CA)
- Multiple Correspondence Analysis (MCA)
- Linear Discriminant Analysis (LDA)
- Regularized Regression

Principal Component Analysis (PCA)

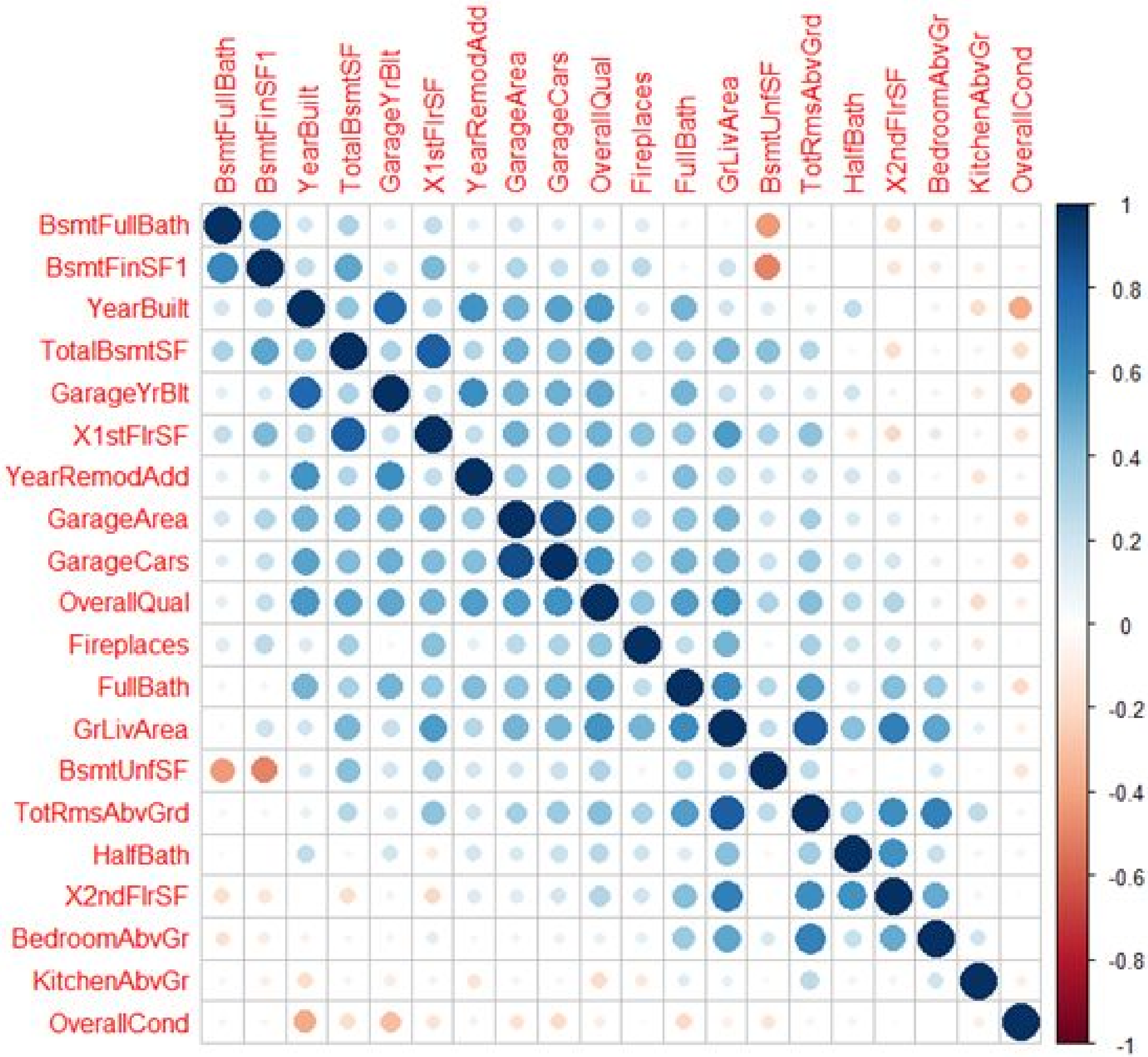
Principal Component Analysis (PCA) is a statistical procedure that allows us to summarize the information contained in a large set by means of a smaller set of “summary indices” that can be more easily visualized and analyzed. It is a very common technique for “dimensionality reduction” and finding the “latent/hidden” factors from the data.

- The goal of PCA is to simplify model features into fewer, uncorrelated features to help visualize patterns in the data and help it run faster.
- It reduces the number of variables while maintaining the majority of the important information.
- It can help solve the very common problem of “Curse of Dimensionality”.
- We should only apply PCA to continuous data and the data should be scaled before applying PCA technique.
- PCA can be very helpful in predicting the parameter of interest e.g. Sale Price of House.
- PCA can help us build the parsimonious model which is easier to explain.
- PCA can also be used in relation with other data analysis techniques for better results.

Testing Correlation Matrix

We began our PCA analysis by examining the data for correlations and there was a corrplot produced and also the correlation matrix test was run to analyze the significance of the entries in the correlation matrix for fields that are highly correlated or completely uncorrelated with the other fields. There were a total 20 variables in the corr test and none of them showed any problem so we kept all the variables in analysis and did not remove any. Most of the variables were positively correlated with each other. There were a couple of groupings among the variables which gave us the hint that what we are going to see in our factor analysis.

```
> # Compute the correlation matrix to see if there is significant
> # correlation to exploit
> corTests3 = cor(houseNum3)
>
> # Visualize correlation matrix
> cor.housing3 = cor(houseNum3, use="complete.obs")
> corrplot(cor.housing3, method="circle", order="AOE")
> houseCorrTest3 = corr.test(houseNum3, adjust="none")
> Mhouse3 = houseCorrTest3$p
> MTesthouse3 = ifelse(Mhouse3 < .01, T, F) # if Mhouse3 value <
> colSums(MTesthouse3) - 1
  OverallQual  OverallCond    YearBuilt YearRemodAdd   BsmtFinSF1
           19           12           18           18           15
 BsmtFullBath     FullBath     HalfBath BedroomAbvGr KitchenAbvGr
           13           17           14           14           14

.01 then TRUE else FALSE

    BsmtUnfSF  TotalBsmtSF    X1stFlrSF    X2ndFlrSF    GrLivArea
           15           17           19           15           18
 TotRmsAbvGrd   Fireplaces  GarageYrBlt   GarageCars   GarageArea
           16           16           17           18           17
```

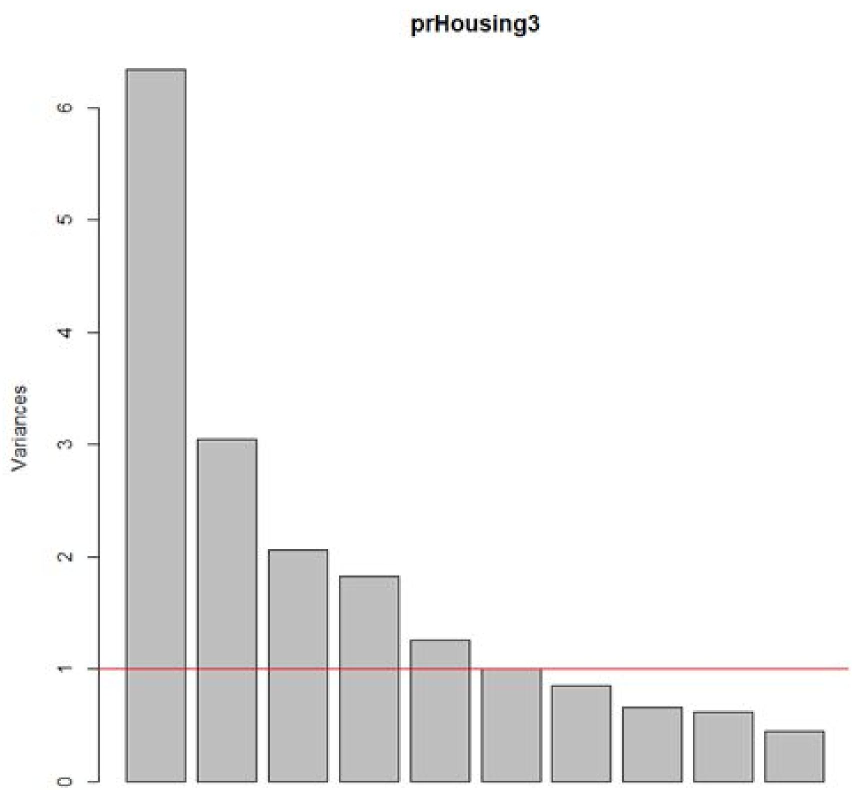


BsmtFullBath
BsmtFinSF1
YearBuilt
TotalBsmtSF
GarageYrBlt
X1stFlrSF
YearRemodAdd
GarageArea
GarageCars
OverallQual
Fireplaces
FullBath
GrLivArea
BsmtUnfSF
TotRmsAbvGrd
HalfBath
X2ndFlrSF
BedroomAbvGr
KitchenAbvGr
OverallCond
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1

Prcomp

Prcomp function in R was used to determine the appropriate number of components required to explain sufficient variance in data. It helps to select the number of components.

```
> ##########################################################################
> # Compute "prcomp" with scaling/correlation matrix
> # and determine number of components
> ##########################################################################
> prHousing3 = prcomp(houseNum3, scale=T)
> plot(prHousing3)           # The scree plot
> abline(1, 0, col="red")    # Put in a line at var=1
> summary(prHousing3)         # Get a summary including variances
Importance of components:
                         PC1    PC2    PC3     PC4     PC5     PC6     PC7
Standard deviation     2.518 1.7446 1.4351 1.35146 1.12379 0.99941 0.92469
Proportion of Variance 0.317 0.1522 0.1030 0.09132 0.06315 0.04994 0.04275
Cumulative Proportion  0.317 0.4691 0.5721 0.66344 0.72658 0.77652 0.81928
```



As per the cumulative proportion and knee in the scree plot, there are five principal components required to explain more than 72% of the variance for this data. The evening out pattern in the scree plot shows that after five, the components start containing unexplainable noise. PC1 has the most Proportion of Variance (32%). Most of the variables looked in the same units but the distribution of the variables varies so scaling or using correlation matrix was appropriate for this data set.

Loadings from Principal

Prcomp function in R is used to select the components and the "principal" function is used if we are required to rotate the components for better interpretability.

```
> ########################################################################
> # use "principal" to compute the Principal Component Analysis
> # with prHousing number of components, and with varimax factor rotation
> ########################################################################
> # prcomp is what we use to select our components
> # principal is used if we need to rotate the components
> principalHousing3 = principal(houseNum3, rotate="varimax", nfactors=5)
> # factors determined from prHousing scree-plot
> print(principalHousing3$loadings, cutoff=.5, sort=T)

Loadings:
             RC1    RC2    RC4    RC3    RC5
OverallQual   0.655
YearBuilt     0.893
YearRemodAdd  0.725
FullBath      0.506
GarageYrBlt   0.879
GarageCars    0.657
GarageArea    0.604
X2ndFlrSF            0.890
GrLivArea            0.811
HalfBath             0.600
BedroomAbvGr         0.726
TotRmsAbvGrd         0.834
TotalBsmtSF                 0.850
X1stFlrSF                   0.909
BsmtFinSF1                         0.834
BsmtUnfSF                         -0.801
BsmtFullBath                       0.804
OverallCond                              -0.638
KitchenAbvGr                              0.701
Fireplaces

                 RC1   RC2   RC4   RC3   RC5
SS loadings    4.181 3.674 3.202 2.143 1.332
Proportion Var 0.209 0.184 0.160 0.107 0.067
Cumulative Var 0.209 0.393 0.553 0.660 0.727
```

To compute the Principal Factor Analysis, we used the "principal" function with varimax rotation and "nfactors=5". The Cumulative Var at RC5 is 72% which is not bad for analyzing the factors. The loadings show a very nice set of components with much better separations of variables. The goal is to group original variables into distinct factors so each variable contributes to only 1 factor so they are easy to interpret.

Following are the five components which were built using 20 numerical variables:

- RC1 is a mix of **GARAGE** + **Age Of Property**;
- RC2 is mostly **Above Ground** + **2nd Floor**;
- RC4 is **Basement** + **1st Floor area**;
- RC3 is nothing but **BASEMENT**;
- RC5 is a negative association between **OverallCond** and **KitchenAbvGr**;

The five factors produced by PCA can be used in a regression analysis to predict the sale price of a house. The 20 variables which contribute to 5 factors can be used to predict around 72% of the variable contribution to sale price of the houses.

Correspondence Analysis (CA)

Correspondence Analysis (CA) is a multivariate graphical technique designed to explore relationships among categorical variables. It is conceptually similar to principal component analysis, but applies to categorical rather than continuous data. Using Correspondence Analysis for this data set, we can see which overall (house) condition corresponds most with each price class.

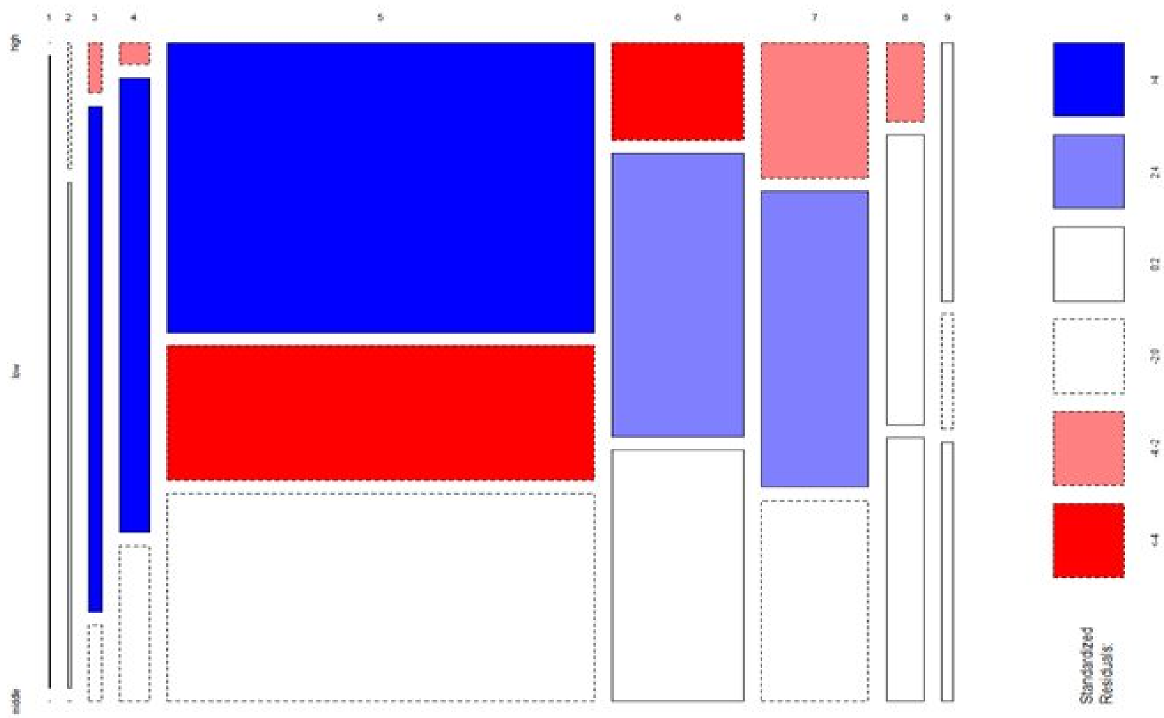
```
# A tibble: 3 x 4
  PriceClassName count    low   high
  <fct>          <int>  <int>  <int>
1 low              487  34900 139600
2 middle           490 139900 190000
3 high             483 191000 755000
```

Price Class broken into low, middle, and high: Near equal representation

Overall Condition: 10 Categories where 1 = Very Poor and 10 = Very Excellent

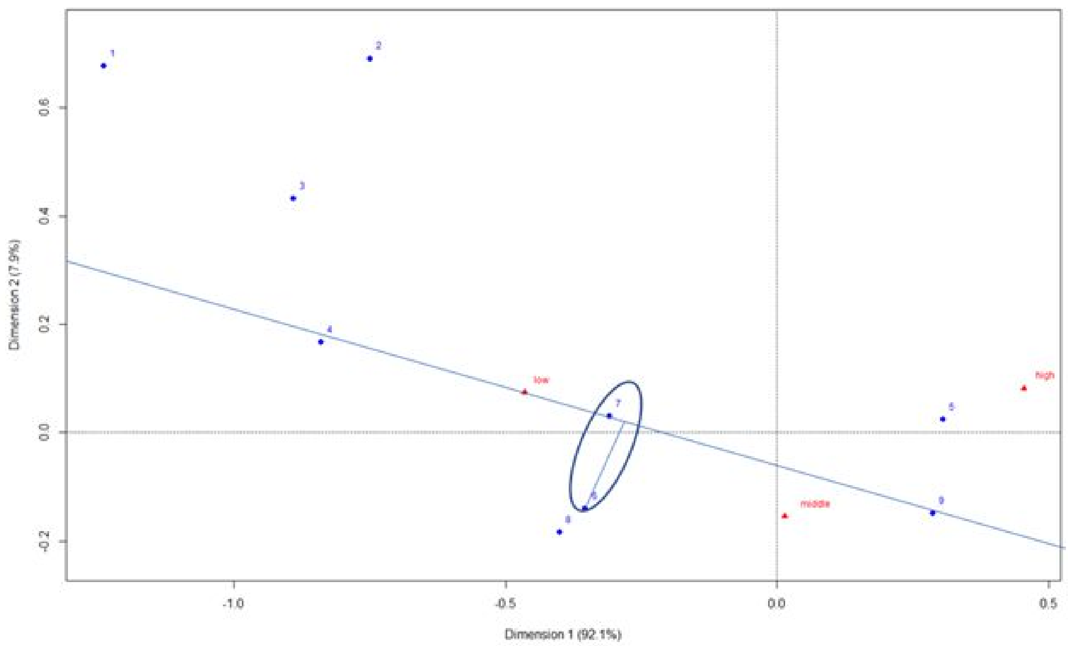
We are going to see if we can minimize this.

CA: Sale Price Class and Overall Condition

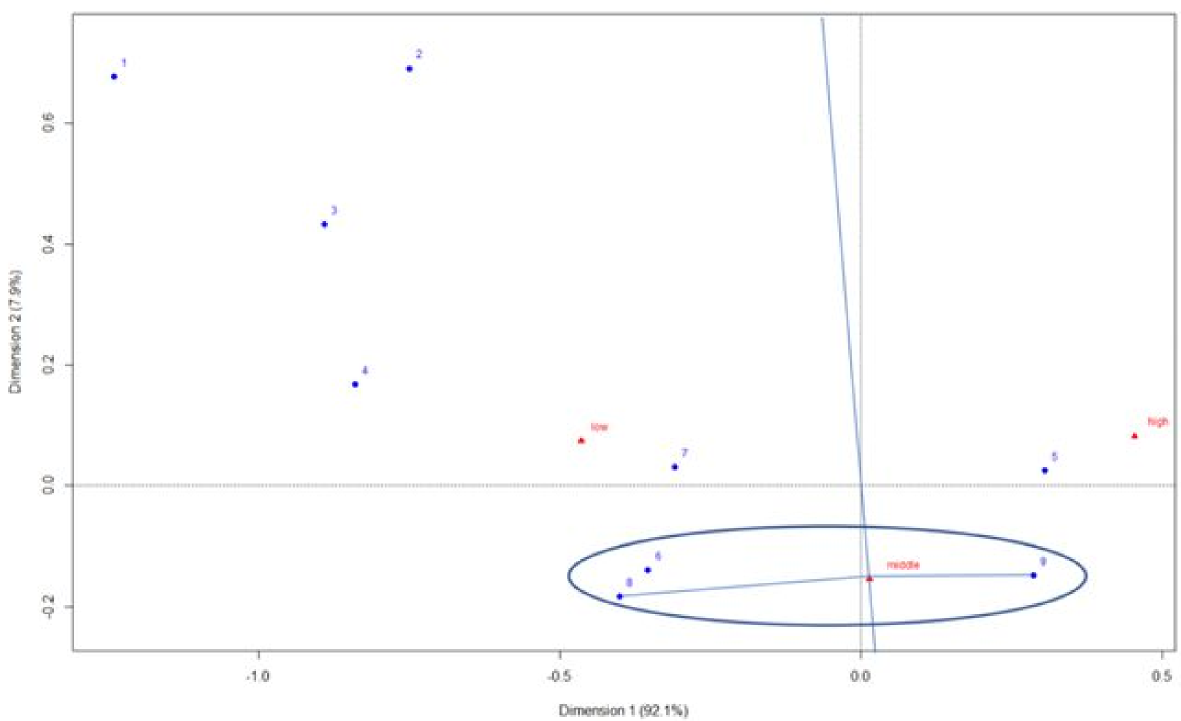



```
> summary(c3)

Principal inertias (eigenvalues):

 dim    value      %    cum%   scree plot
 1      0.140483  92.1  92.1   ***********************
 2      0.012064   7.9 100.0   **
        -------- -----
 Total: 0.152547 100.0
```

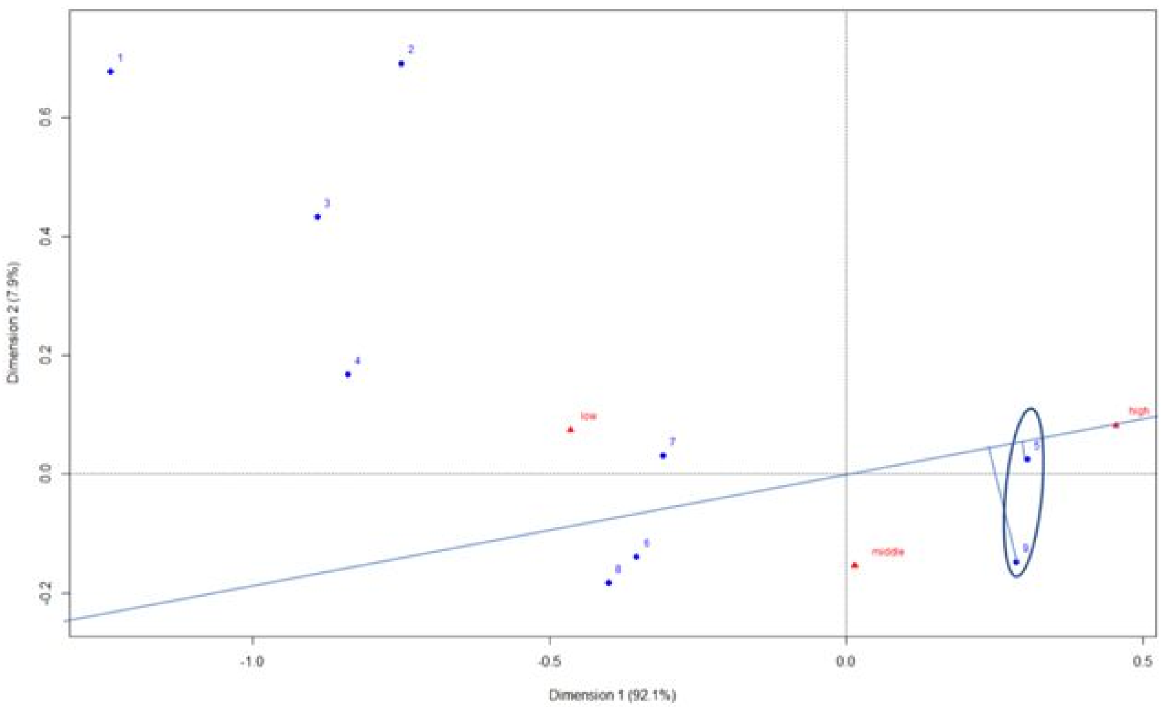
CA: Low Price Class



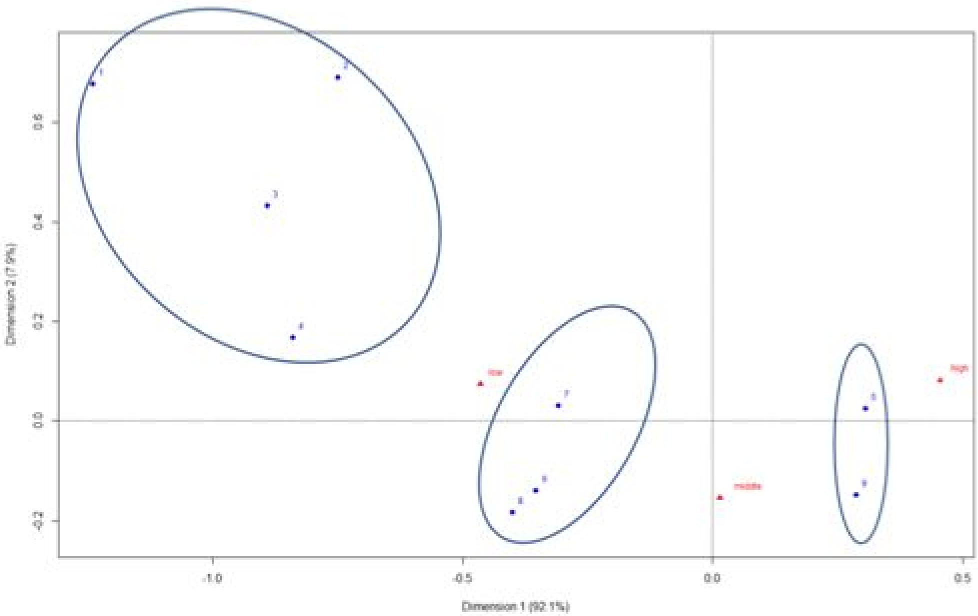
CA: Middle Price Class



CA: High Price Class



CA: An Interesting Pattern




```
> table(housing_train$OverallCond)

  1   2   3   4   5   6   7   8   9
  1   5  25  57 821 252 205  72  22
```

Notice the groupings:

- [1,2,3,4]
- [6,7,8]
- [5,9]

Begs the question “can we narrow down to 3 categories? We need to look further into why 5 is grouped with 9. That may be a case of class imbalance.

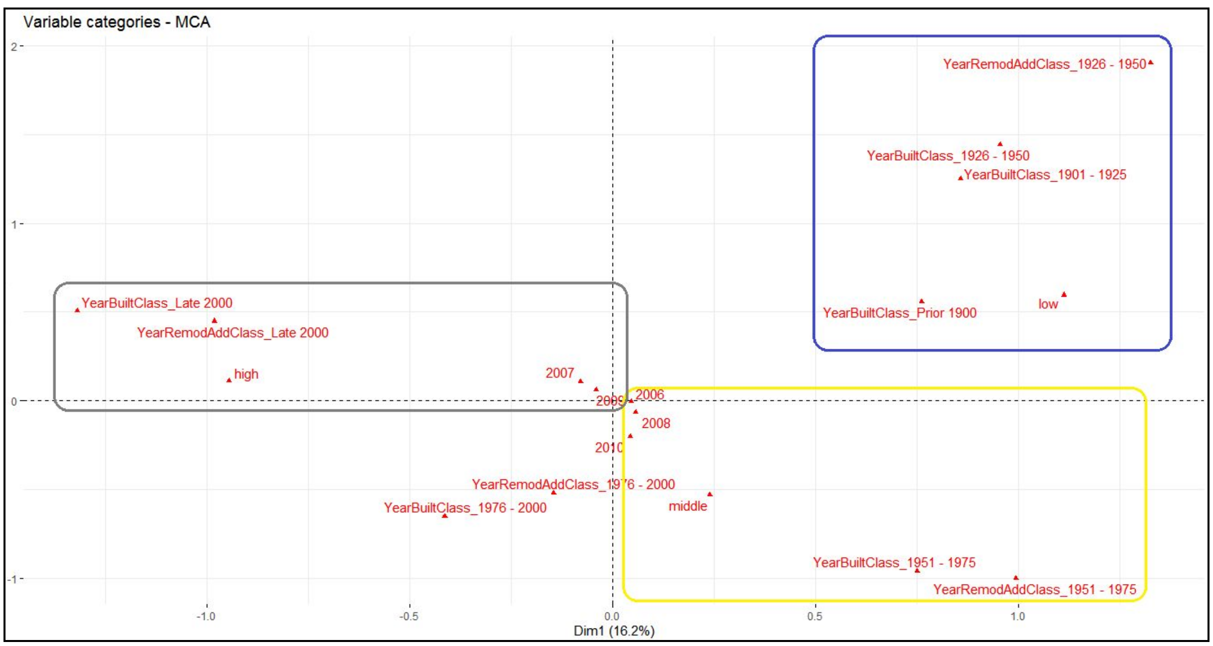
Multiple Correspondence Analysis (MCA)

The Multiple correspondence analysis (MCA) is an extension of the simple correspondence analysis for summarizing and visualizing a data table containing more than two categorical variables.

MCA for the following variables:

- SalesClass
- YearRemodAddClass
- YearBuiltClass
- YrSold

```
> fviz_mca_var(res.mca3, repel = TRUE, ggtheme = theme_minimal())
```



Following conclusion can be drawn after applying the MCA technique:

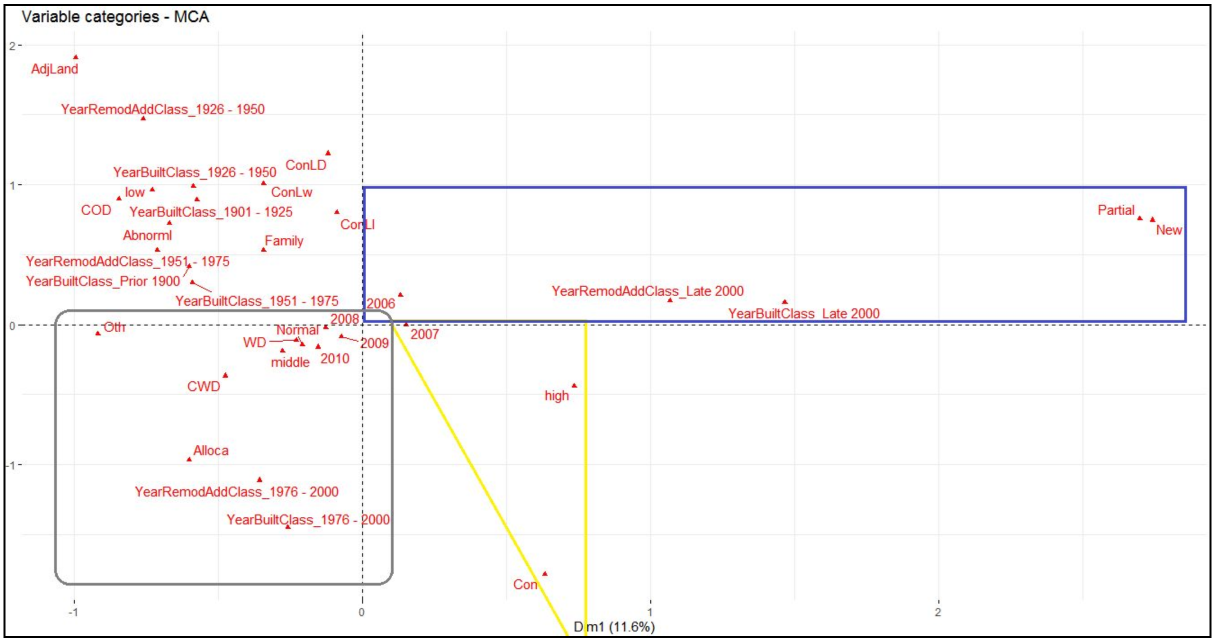
- Sales prices are high for the houses sold in the year 2007 and 2009. These houses were usually built and remodelled in Late 2000's (Highlighted in Grey).
- Houses sold in the year 2006, 2008 and 2010 have average sales price. The houses in this category were built and remodelled between the years 1951 - 1975 (Highlighted in Yellow).
- The year 2008 had a great economic recession. The slump in house prices can be attributed to this recession.
- The houses built and remodelled prior to 1951 have low prices.
- The houses built and remodelled between the year 1976 and 200 are plotted between high prices and average prices.

Extending this analysis to include SaleCondition and Sale Type.

MCA for the following variables:

- SalesClass
- YearRemodAddClass
- YearBuiltClass
- SaleCondition
- SaleType
- YrSold

```
> fviz_mca_var(res.mca1, repel = TRUE, ggtheme = theme_minimal())
```



Following conclusion can be drawn after extending MCA analysis to include SaleCondition and Sale Type:

- Sales prices are high for the houses sold in the year 2007 having the type of sale as Con 'Contract 15% Down payment regular terms'. (Highlighted in Yellow).
- Houses sold in the year 2008, 2009, 2010 have average sales price. The type of sale is CWD 'Warranty Deed – Cash' and WD 'Warranty Deed – Conventional'. The houses with average Sales Price are usually remodelled between 1976 – 2000 and newly built between 1976 – 2000. The sale condition was Normal for this category. (Highlighted in Grey).
- The houses built and remodelled before 1976 have low prices. The sale type of these houses was ConLw 'Contract Low Down and low interest', ConLI 'Contract Low Interest', ConLD 'Contract Low Down' and COD 'Court Officer Deed/Estate'. Most of the houses were sold between family members.

Linear Discriminant Analysis & Multidimensional Scaling

Linear Discriminant Analysis or LDA is a dimensionality reduction technique. It is used as a pre-processing step in Machine Learning and applications of pattern classification. For this housing data set, LDA will be implemented to locate a new feature space in order to project the data in a format that maximizes separability within the classes. LDA will produce a confusion matrix which will be implemented for MDS - to see if we have similar roof style profiles.

LDA: Parameter of Interest

In our analysis with LDA, we had decided to have RoofType as a parameter of interest. There are six types of roofs in our dataset: Flat, Gable, Gabrel, Hip, Mansard, and Shed. These were then transformed accordingly in ordinal fashion from 1 through 6. In our ldahist(), we noticed some separation between groups 5 & 6, and in our confusion matrix we noticed a high misclassification of groups 2 & 4.

Roofstyle adjusted to ordinal:

- Flat → 1
- Gable → 2
- Gambrel → 3
- Hip → 4
- Mansard → 5
- Shed → 6

LDA: Roofstyle Output

Coefficients of linear discriminants:

	LD1	LD2	LD3	LD4	LD5
MSSubClass	3.053046e-03	1.668768e-03	-1.571022e-03	-1.655543e-03	-4.398299e-04
LotFrontage	-2.915105e-03	1.275523e-03	-1.584230e-03	-9.635418e-03	-9.702532e-03
LotArea	1.592742e-07	-2.800454e-05	1.356591e-05	8.563029e-06	3.676230e-06
OverallQual	-1.912185e-01	1.983061e-01	1.792856e-02	-1.536920e-01	-9.768921e-02
OverallCond	5.221281e-03	-1.391219e-01	-1.381019e-01	8.546249e-03	1.155574e-01
YearBuilt	-1.377996e-04	-1.501641e-02	7.879580e-03	8.073257e-04	3.436394e-02
YearRemodAdd	7.224063e-03	-4.173004e-04	-2.202367e-02	-1.534933e-02	-9.018419e-03
MasVnrArea	-2.467827e-03	1.831616e-03	-4.533538e-04	-1.193892e-03	-1.064628e-03
BsmtFinSF1	4.780128e-04	3.575601e-03	-1.224698e-03	8.733541e-04	1.037841e-03
BsmtUnfSF	1.034583e-03	3.160859e-03	-1.396777e-03	9.224614e-04	1.128152e-03
TotalBsmtSF	-3.964853e-04	-2.109536e-03	6.758098e-04	-3.139419e-04	7.161889e-04
X1stFlrSF	-3.870494e-03	-2.643083e-03	-2.604441e-03	1.245989e-03	-4.023123e-03
X2ndFlrSF	-1.794859e-03	-2.399462e-03	-3.221391e-03	7.484138e-04	-1.581301e-03
GrLivArea	2.544914e-03	4.536562e-04	2.851137e-03	-3.048879e-04	1.949531e-03
BsmtFullBath	1.904736e-01	-1.569381e-01	7.503917e-02	9.501496e-02	4.171679e-01
BsmtHalfBath	2.910284e-01	-5.600432e-04	-2.057832e-01	1.689020e-02	-3.750106e-01
FullBath	6.479728e-01	3.425695e-01	1.648439e-01	-5.899530e-01	-1.607102e-01
HalfBath	1.925144e-01	2.722065e-01	-2.915071e-01	1.433682e+00	-1.073133e+00
BedroomAbvGr	8.969979e-02	5.405714e-01	2.312889e-01	-6.518310e-01	-6.649403e-01
TotRmsAbvGrd	-1.545988e-01	1.780219e-01	-2.863631e-01	3.223963e-01	4.716609e-01
Fireplaces	7.135947e-02	-1.634243e-01	2.693130e-01	-2.563163e-01	7.582861e-02
GarageYrBlt	3.640107e-03	2.226029e-02	3.120707e-02	1.930813e-02	-2.802424e-02
GarageCars	-1.846648e-01	-3.747829e-01	-1.201215e-01	-2.049278e-02	-8.261702e-01
GarageArea	8.928013e-04	-3.376275e-04	7.281896e-05	7.546617e-04	1.056108e-03
WoodDeckSF	-3.778125e-04	-7.033140e-04	1.397417e-04	7.689830e-04	-1.307550e-03
OpenPorchSF	-2.166068e-04	-1.458660e-03	6.411148e-03	-4.673998e-03	1.090069e-03
X3SsnPorch	-1.646876e-03	-1.719891e-03	2.597904e-03	-1.242355e-03	6.584669e-04
MoSold	-4.525366e-03	-1.743000e-02	-5.391476e-02	-5.558448e-03	2.312056e-02
YrSold	4.897112e-03	8.611996e-03	-5.101012e-02	6.190944e-02	5.128039e-02
SalePrice	-3.852126e-06	7.698980e-07	-1.641925e-06	-3.106803e-06	2.736284e-06

Proportion of trace:

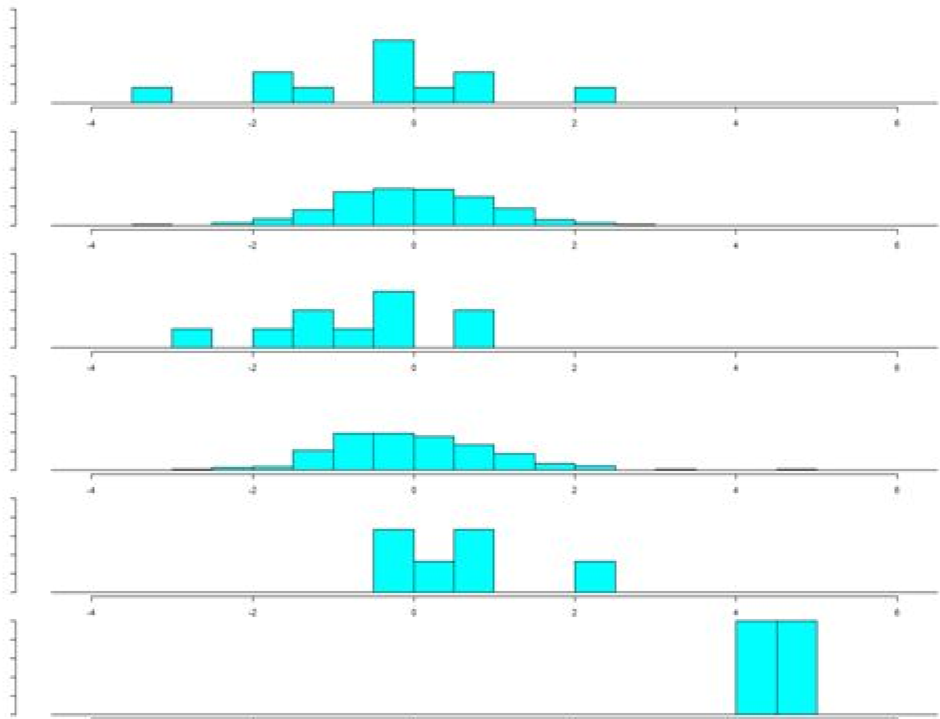
LD1	LD2	LD3	LD4	LD5
0.5096	0.1955	0.1441	0.0830	0.0678

LDA: Roofstyle Scalings

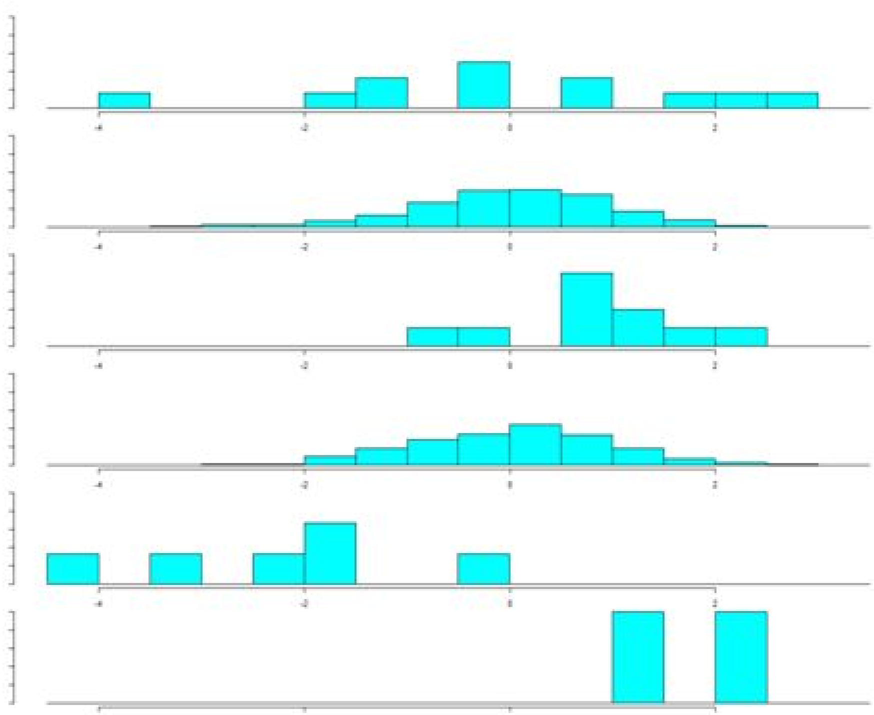
	LD1	LD2	LD3	LD4	LD5
GarageCars	-1.912185e-01	-3.747829e-01	-2.915071e-01	-6.518310e-01	-1.073133e+00
Fireplaces	-1.846648e-01	-1.634243e-01	-2.863631e-01	-5.899530e-01	-8.261702e-01
BsmtFullBath	-1.545988e-01	-1.569381e-01	-2.057832e-01	-2.563163e-01	-6.649403e-01
OverallCond	-4.525366e-03	-1.391219e-01	-1.381019e-01	-1.536920e-01	-3.750106e-01
MoSold	-3.870494e-03	-1.743000e-02	-1.201215e-01	-2.049278e-02	-1.607102e-01
YearBuilt	-2.915105e-03	-1.501641e-02	-5.391476e-02	-1.534933e-02	-9.768921e-02
X1stFlrSF	-2.467827e-03	-2.643083e-03	-5.101012e-02	-9.635418e-03	-2.802424e-02
X2ndFlrSF	-1.794859e-03	-2.399462e-03	-2.202367e-02	-5.558448e-03	-9.702532e-03
TotalBsmtSF	-1.646876e-03	-2.109536e-03	-3.221391e-03	-4.673998e-03	-9.018419e-03
X3SsnPorch	-3.964853e-04	-1.719891e-03	-2.604441e-03	-1.655543e-03	-4.023123e-03
OpenPorchSF	-3.778125e-04	-1.458660e-03	-1.584230e-03	-1.242355e-03	-1.581301e-03
WoodDeckSF	-2.166068e-04	-7.033140e-04	-1.571022e-03	-1.193892e-03	-1.307550e-03
BsmtHalfBath	-1.377996e-04	-5.600432e-04	-1.396777e-03	-3.139419e-04	-1.064628e-03
YearRemodAdd	-3.852126e-06	-4.173004e-04	-1.224698e-03	-3.048879e-04	-4.398299e-04
GarageArea	1.592742e-07	-3.376275e-04	-4.533538e-04	-3.106803e-06	2.736284e-06
LotArea	4.780128e-04	-2.800454e-05	-1.641925e-06	8.563029e-06	3.676230e-06
SalePrice	8.928013e-04	7.698980e-07	1.356591e-05	7.484138e-04	6.584669e-04
GrLivArea	1.034583e-03	4.536562e-04	7.281896e-05	7.546617e-04	7.161889e-04
LotFrontage	2.544914e-03	1.275523e-03	1.397417e-04	7.689830e-04	1.037841e-03
MSSubClass	3.053046e-03	1.668768e-03	6.758098e-04	8.073257e-04	1.056108e-03
MasVnrArea	3.640107e-03	1.831616e-03	2.597904e-03	8.733541e-04	1.090069e-03
BsmtUnfSF	4.897112e-03	3.160859e-03	2.851137e-03	9.224614e-04	1.128152e-03
BsmtFinSF1	5.221281e-03	3.575601e-03	6.411148e-03	1.245989e-03	1.949531e-03
YrSold	7.224063e-03	8.611996e-03	7.879580e-03	8.546249e-03	2.312056e-02
GarageYrBlt	7.135947e-02	2.226029e-02	1.792856e-02	1.689020e-02	3.436394e-02
TotRmsAbvGrd	8.969979e-02	1.780219e-01	3.120707e-02	1.930813e-02	5.128039e-02
OverallQual	1.904736e-01	1.983061e-01	7.503917e-02	6.190944e-02	7.582861e-02
HalfBath	1.925144e-01	2.722065e-01	1.648439e-01	9.501496e-02	1.155574e-01
FullBath	2.910284e-01	3.425695e-01	2.312889e-01	3.223963e-01	4.171679e-01
BedroomAbvGr	6.479728e-01	5.405714e-01	2.693130e-01	1.433682e+00	4.716609e-01

LDA: Separation Visualization

```
> ldahist(data = housingtrain.lda.values$x[,4],g=housing_train.edit$RoofStlyeNum)
```



```
> ldahist(data = housingtrain.lda.values$x[,5],g=housing_train.edit$RoofStlyeNum)
```



LDA: Confusion Matrix

We then proceed to execute MDS on the confusion matrix from this. Which led to the below results:

```
> Rooftbl<-as.data.frame.matrix(Rooftbl)
> Rooftbl
   1   2  3  4 5 6
1  6   5  0  0 1 0
2 14 990 14 35 7 3
3  0   5  5  0 0 0
4  9 192  5 71 0 1
5  0   3  0  0 3 0
6  0   0  0  0 0 2
```

```
$GOF
[1] 0.007516108 1.000000000

> roof.mds<-isoMDS(d)
initial  value 0.157686
iter   5 value 0.138204
iter  10 value 0.034102
iter  15 value 0.012837
iter  15 value 0.008584
final  value 0.008584
converged
> roof.mds$stress
[1] 0.008583677
```

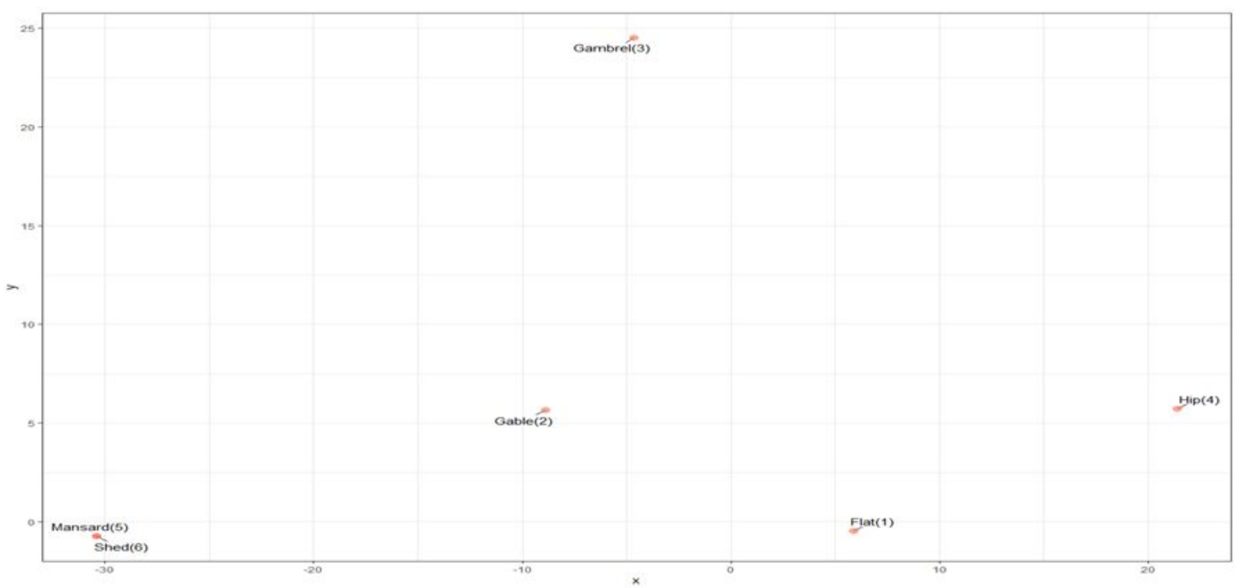
Above we can see the confusion matrix that is being used for MDS. Another part we see above is the amount of iterations that it took to identify the stress. Our stress looks good at .008, but I believe the goodness of fit may be a concern. Can run MDS by Roofstyle to look further into this.

MDS: Roofstyle

```
> fit <- cmdscale(Rooftbl,eig=TRUE, k=2) # k is the number of dim
> fit
$points
        [,1]       [,2]
1   5.891961 -0.4691393
2  -8.901380  5.6531402
3  -4.639301 24.5064059
4  21.393551  5.7322255
5 -30.401516 -0.7155886
6 -30.394414 -0.7366442

$eig
[1]  2.441229e+03  6.666551e+02 -4.541624e+00 -6.782711e+02 -5.036258e+03 -4.046696e+05

$x
NULL

$ac
[1] 0

$GOF
[1] 0.007516108 1.000000000
```

```
> roof.mds<-isoMDS(d)
initial  value 0.157686
iter   5 value 0.138204
iter  10 value 0.034102
iter  15 value 0.012837
iter  15 value 0.008584
final  value 0.008584
converged
> roof.mds$stress #very good
[1] 0.008583677
```

Goodness of fit is not good so we may need to look further into. The Stress value = .00858 which is very good.



Looking above at the plot for MDS, we can see that 5 & 6 overlap (Mansard & Shed). Our thought behind this, is that this is representative of the separation that we saw in between groups 5 & 6 in LDA as the separation was occurring because they were maximizing distance to reduce dimensionality. This is confirmed in MDS when they are not separated. Also we can see that group 2 & 4 (Gable & Hip) are at the same y level, we believe that this may confirm what we see in the confusion matrix in the fact that group 4 is misclassified as group 2.

Regularized Regression

Regularized regression is the type of regression where the coefficient estimates are constrained to zero. The magnitude (size) of coefficients, as well as the magnitude of the error term, are penalized. The complex models are discouraged, primarily to avoid overfitting. The common types of regularized regression methods are Ridge regression, Lasso regression, and Elastic Net.

In the final line of our analysis, we created the regularized regression models using LASSO and Ridge, and elastic net regression techniques to determine the best model to predict sale price. We took the sales price as a parameter of interest and took the log of SalePrice. After applying log transformation, the skew was quite low and the Q-Q plot looked much better. For independent variables, we considered all the variables and treated numerical and categorical variables accordingly. There were also few outliers in the dataset which were removed.

Parameter of Interest: SalePrice

Feature Engineering:

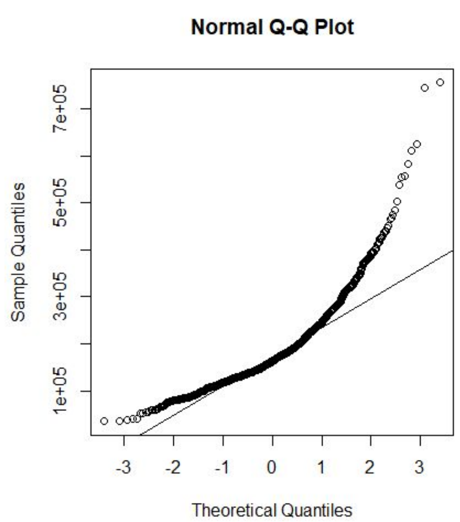
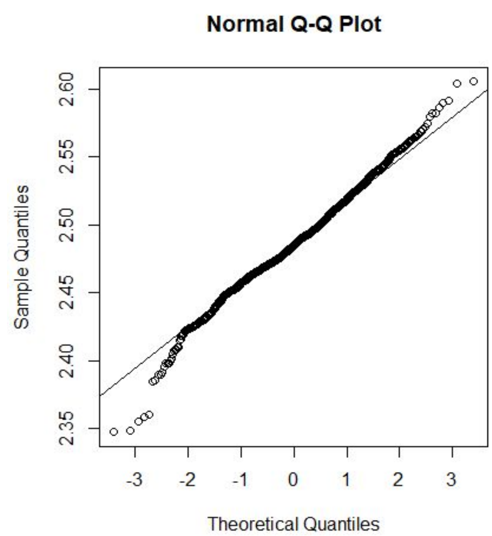
- Total number of bathrooms = FullBath + HalfBath*0.5 +BsmtFullBath + BsmtHalfBath*0.5
- Total square feet = GrLivArea+ TotalBsmtSF

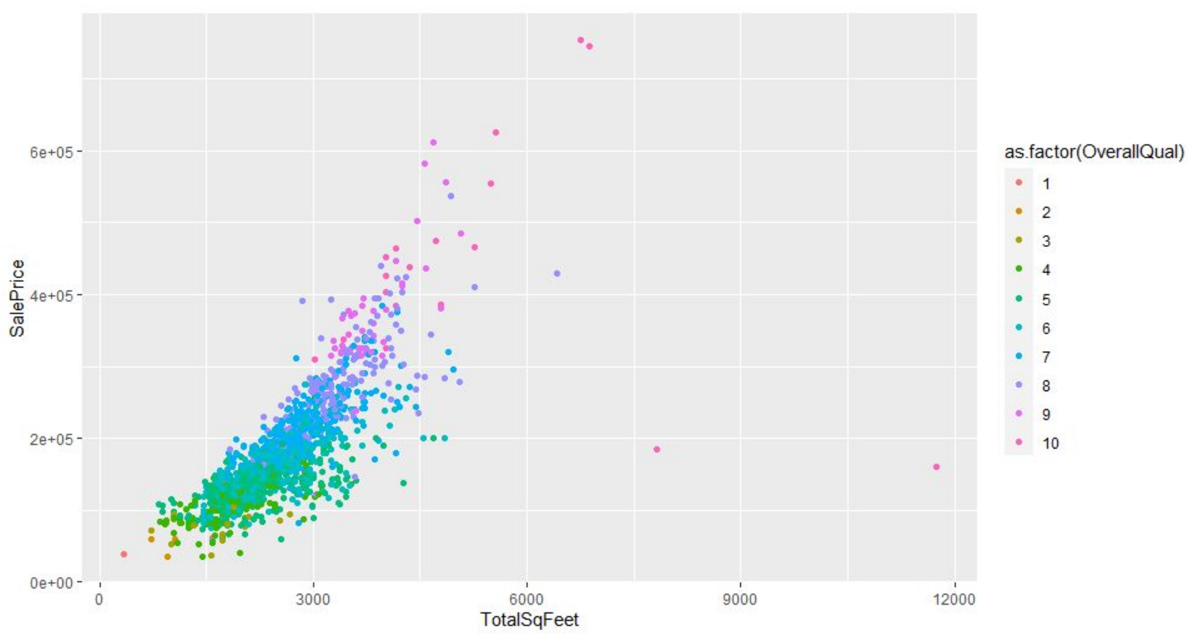
Data Preparation for modeling:

- Log transformation of response variable
- Removing outliers
- Label encoding
- Changed some categorical variables which have numerical order to ordinal variables:

  'Ex'=5,'Gd'=4,'TA'=3,'Fa'=2,'Po'=1,'None'=0

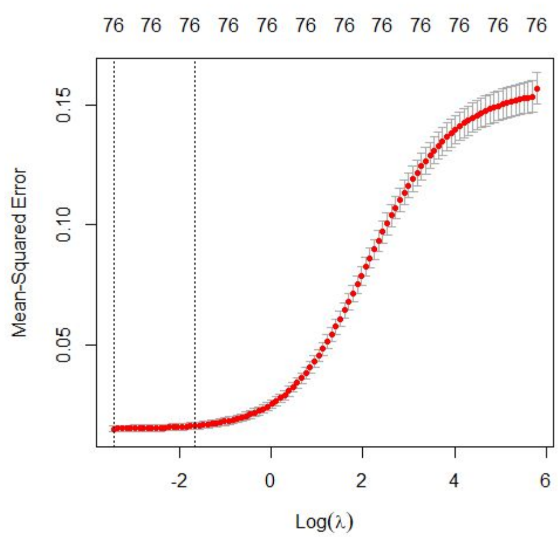
  'BsmtQual', 'BsmtCond', 'GarageQual', 'GarageCond', 'ExterQual', 'ExterCond', 'HeatingQC', 'PoolQC', 'KitchenQual’

<table>
<tr><th>Sale Price</th><th>Log (Sale Price)</th></tr>
<tr><td>> skewness(housingtrain$SalePrice)
[1] 1.880941
> kurtosis(housingtrain$SalePrice)
[1] 9.509812</td><td>> skewness(housingtrain$SalePrice)
[1] -0.02035577
> kurtosis(housingtrain$SalePrice)
[1] 3.890477</td></tr>
<tr><td>
</td><td>
</td></tr>
</table>

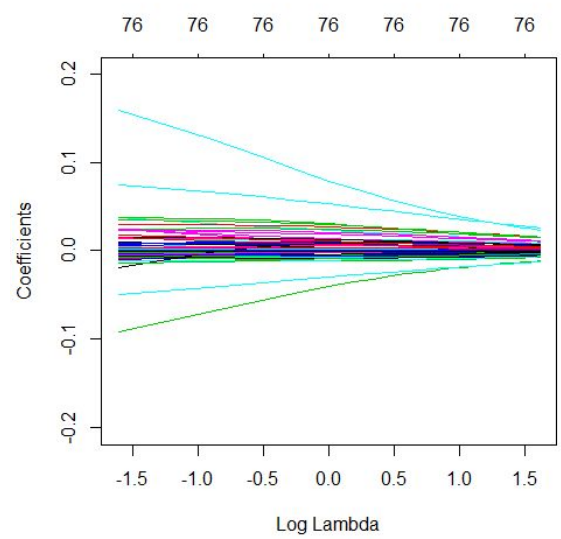


Ridge Regression

```
> RIDGEfit$lambda.min
[1] 0.0324373
> RIDGEfit$lambda.1se
[1] 0.1731081

Call:  glmnet(x = xTrain, y = yTrain,
alpha = 0, lambda = 0.1731081)

  Df   %Dev Lambda
1 76 0.9074 0.1731
```

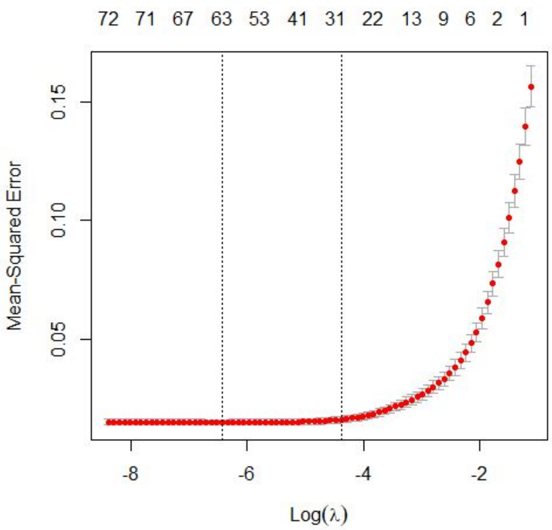



```
lRange = seq(0, 5, .2)
fitRidge = glmnet(xTrain, yTrain,
alpha=0, lambda=lRange)
plot(fitRidge xvar="lambda")

> rmseRidgetrain
[1] 0.1212441
```

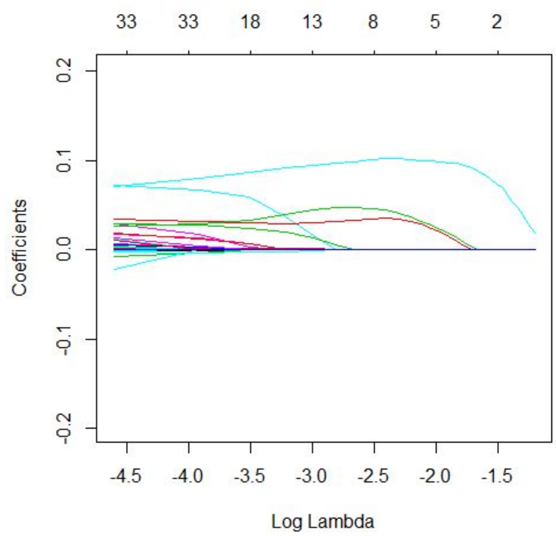


Lasso Regression

```
> LASSOfit$lambda.min
[1]0.001614412
> LASSOfit$lambda.1se
[1]0.01249981
Call:  glmnet(x = xTrain, y = yTrain,
alpha = 1, lambda = 0.01138936)
  Df   %Dev  Lambda
1 31 0.9043 0.01139
```

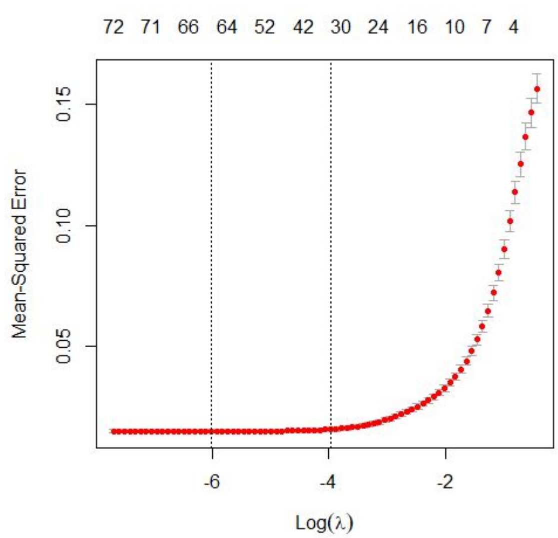



```
lRange = seq(0, 1, .01)
fitLASSO = glmnet(xTrain, yTrain,
alpha=1, lambda=lRange)
plot(fitLASSO, xvar="lambda")

> rmseLASSOtrain
[1] 0.1234957
```

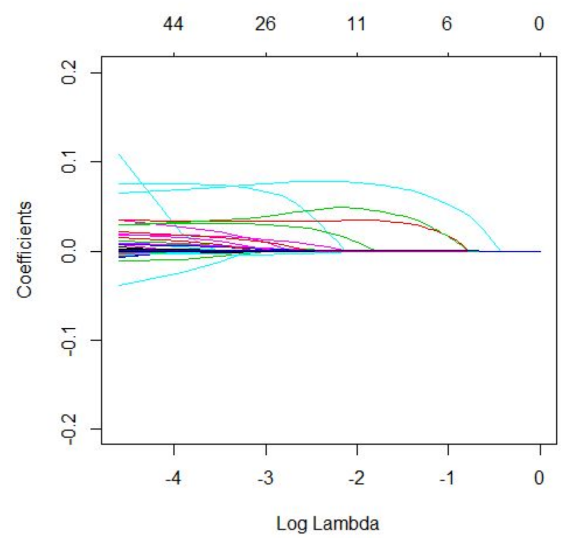


Elastic Net Regression

```
> Elasticfit = cv.glmnet(xTrain,
yTrain, alpha=0.5, nfolds=10)
> Elasticfit$lambda.min
[1] 0.002442487
>
> Elasticfit$lambda.1se
[1] 0.01891129

Call:  glmnet(x = xTrain, y = yTrain,
alpha = 1, lambda = 0.01891129)

  Df   %Dev  Lambda
1 24 0.8919 0.01891
```




```
lRange = seq(0, 1, .1)
fitElastic = glmnet(xTrain, yTrain,
alpha=0.5, lambda=lRange)
plot(fitElastic, xvar="lambda")

> rmseElastictrain
[1] 0.12092
```



Summary of Final Model

Final Model: Lasso regression

Standardized data to get standardized Beta:

```
xTrain <-scale(xTrain)
yTrain<- scale(yTrain)

LASSOfit$lambda.min
0.004471833

LASSOfit$lambda.1se
0.02619162
```

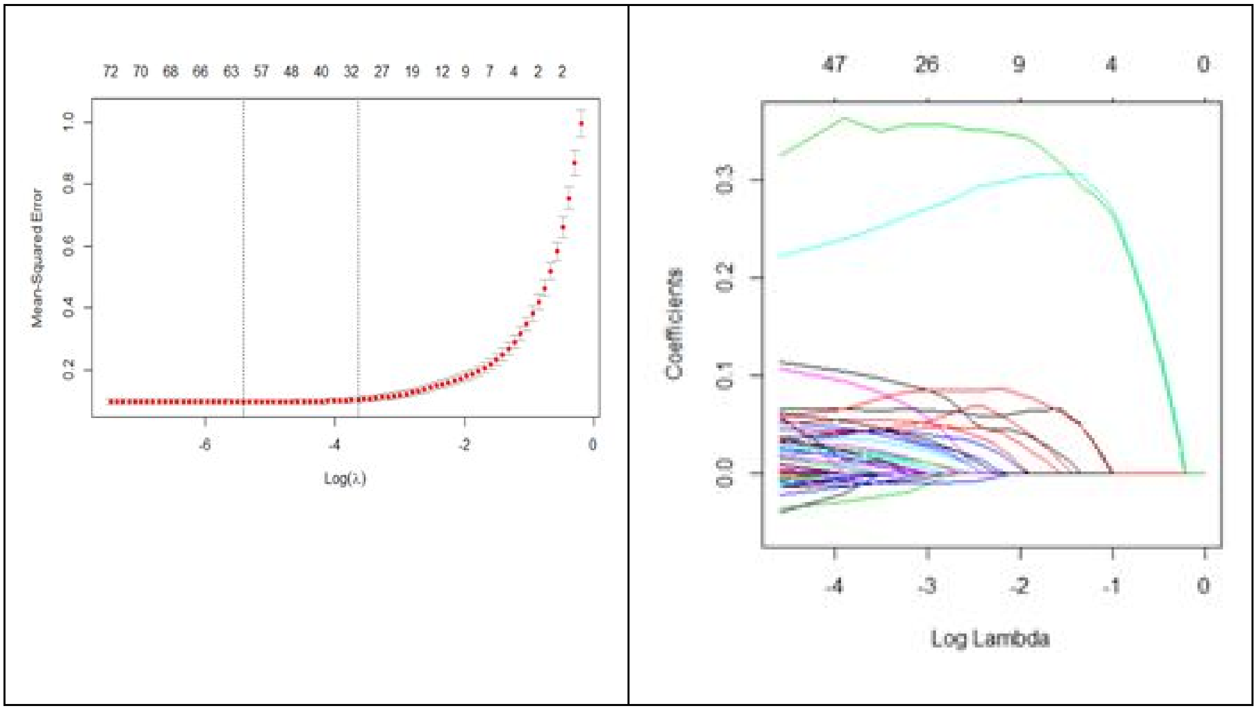
Increased lambda from lambda.1se to 0.1:

- Number of selected variables: 31 >> 12
- R square: 0.902 >> 0.848
- RMSE: 0.311 >> 0.389

Important variables: Coefficient of variables included in the model

	Df	%Dev	Lambda
86	7	0.81740	0.15
87	9	0.82370	0.14
88	9	0.83040	0.13
89	10	0.83640	0.12
90	11	0.84220	0.11
91	12	0.84810	0.10
92	13	0.85420	0.09
93	15	0.86230	0.08
94	16	0.87070	0.07
95	19	0.87850	0.06
96	21	0.88620	0.05
97	26	0.89340	0.04
98	28	0.90040	0.03
99	33	0.90690	0.02
100	47	0.91360	0.01
101	75	0.91840	0.00

Variable	Standardized Beta
TotalSqFeet	0.349245024
OverallQual	0.295903256
TotBathrooms	0.085753377
YearRemodAdd	0.064599293
GarageCars	0.058765700
YearBuilt	0.045731967
GarageArea	0.044595658
Fireplaces	0.030694746
CentralAir	0.023997537
BsmtFinSF1	0.007214968
GarageType	-0.004665385
SaleCondition	0.002253244



Comparing among the above three models, we can conclude that R-square and RMSE of the models are almost similar. As Lasso is a regression analysis method that performs both variable selection and regularization, we computed the coefficients of variables of the lasso model using lambda equal to lambda.lse. The results show that Lasso regression selected 31 predictors out of the initial 72 variables.