



MASTER OF  
MANAGEMENT  
ANALYTICS

**MMA 867**

**Predictive Modeling**

**Anton Ovchinnikov**

**Individual Assignment 1**

Kaggle Contest Competition (House Prices: Advanced Regression Techniques)

Place: 1228 out of 4781 Teams (Top 33%)

May 3<sup>rd</sup> 2020

Neelan Muthurajah

Filename	Pages	Comments and/or Instructions
House Prices.R		
MMA 867 Individual Assignment 1	6	
Validation_1228thPlace.csv		
data_description.txt		

**Additional Comments: None**

The three competitions I identified for the Kaggle assignment included "House Prices: Advanced Regression Techniques", "Predict Future Sales" and "New York City Taxi Trip Duration". All three competitions were suitable for predictive modeling using regression analysis as each contest's goal was to predict a quantity. The chart below shows the number of participants per contest.

Contest	Participants/Teams	Goal
House Prices: Advanced Regression Techniques (Active)	4781	Predict house prices
Predict Future Sales (Active)	6568	Predict sales of products sold across shops
New York City Taxi Trip Duration (Completed)	1257	Predict trip duration of taxis around New York

Although Predict Future Sales and New York City Taxi Duration were tempting, they required more advanced techniques such as XGboost or time series modeling. Therefore, the competition I decided to try was House Prices as the dataset required regression analysis in order to predict the prices of houses based on a variety of independent variables. In addition, the dataset was very rich comprising of 80 variables in total. These included numerical variables such as land area, floor area, build quality etc. along with categorical variables such as neighborhood, type of foundation, exterior condition just to name a few. The dataset was clean with only minor issues such as missing values which made it perfect for feature engineering and modeling.

Link to the House Prices: Advanced Regression Techniques Competition is:  
<https://www.kaggle.com/c/house-prices-advanced-regression-techniques/overview>

After importing the training and validation sets into R, the first few commands I ran included the following:

```
head(train)
str(train)
tail(train)
summary(train)
```

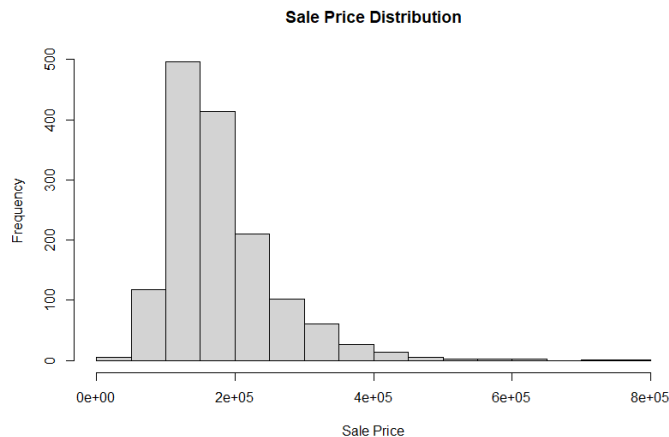
These commands allowed me to explore the data a bit more. For instance, using the summary command, I noticed that there were missing values in the dataset. For example, the column MasVnrArea included NA's.

```
      MasVnrArea
Min.   :  0.0
1st Qu.:  0.0
Median :  0.0
Mean   : 103.7
3rd Qu.: 166.0
Max.   :1600.0
NA's   : 8
```

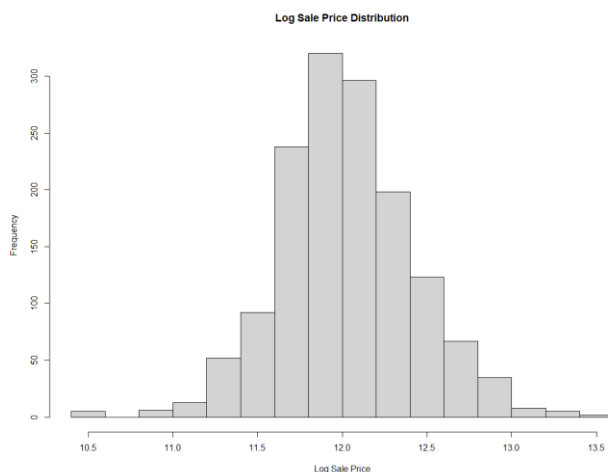
To help solve this issue, I divided the training data into two separate data frames. One data frame included all numeric/integer type variables whereas the second data frame included all character variables. For the **numeric data frame**, I simply replaced the NA's with the mean column value. For the **character data frame**, I first converted all variables to factors and then used the fastDummies package in R to convert all variables to dummy variables. In addition, NA values were replaced with 0's.

Furthermore, I noticed that surface area for different aspects of a home had a positive correlation with Sale price. However, the dataset did not provide a column that showed **Total Surface Area** of a home. Instead, the data had surface area split amongst different areas of a home such as basement SF, first/second floor SF and garage SF. Since feature engineering is an important aspect of modeling, I used these three variables to create a new column for **Total\_SF** of a home by summing these values together.

My next step involved using the **training numeric data frame** to develop a histogram for Sale Price (the variable of interest) in order to understand of any potential skewness.



I noticed some skewness in the sale price variable based on the distribution of sale price. It's better to log this variable.



Once sales price was logged, I noticed a much more normal distribution which is an important assumption of regression modeling.

### **Model 1: RMSLE of 0.1315043, MAPE of 9.797366**

**R-Code: Line 190 to 224**

Based on the case "Sarah gets a Diamond", my first instinct was to log the other numeric variables using only the **training numeric data frame**. Therefore, I logged all variables (regardless of actual skewness) to see if a linear model based on all logs would suffice. I decided to exclude all categorical variables in this model.

Next, I took the **training numeric data frame** and split this into training and test sets based on a 70%/30% split. Once this was complete, I developed a linear model using the log of Sale Price as the predictor variable and the log of **ALL** numeric values as the independent variables using the training dataset. Using this model, I ran the stepAIC command in R to help in the variable selection process. Based on the output of the stepAIC model, I calculated the MAPE as well as the RMSLE on the test data. This yielded results of 0.1315043 for RMSLE and a MAPE of 9.797366.

```
> mean(test_set_model2$percent_errors) #display Mean Absolute Percentage Error (MAPE)
[1] 9.797366
> rmsle(test_set_model2$SalePrice,test_set_model2$Prediction)
[1] 0.1315043
```

## **Model 2: RMSLE of 0.1184259 MAPE of 8.613581**

### **R-Code: Line 56 to 106**

For model 2, I wanted to incorporate categorical variables and expand upon model 1. Therefore, I combined the **numeric data frame** with the **character/categorical data frame** using the cbind command. This created a clean training data frame that had log transformations of all numerical values and dummy variables in place of all categorical variables.

Next, I took the training data frame and split this into training and test sets based on a 70%/30% split. Once this was complete, I developed a linear model in R using the logged Sale Price as the predictor variable and **ALL** other variables as the independent variables. Using this model, I ran the stepAIC command in R to help in the variable selection process. Based on the output of the stepAIC model, I calculated the MAPE as well as RMSLE on the test set. This yielded results of 0.1184259 for RMSLE and a MAPE of 8.613581.

```
> test_set_model1$percent_errors <- abs((test_set_model1$SalePrice-test_set_model1$Prediction)/test_set_model1$SalePrice)*100
> mean(test_set_model1$percent_errors) #display Mean Absolute Percentage Error (MAPE)
[1] 8.613581
> rmsle(test_set_model1$SalePrice,test_set_model1$Prediction)
[1] 0.1184259
```

## **Using Model 2 on the Validation Data from Kaggle**

Since model 2 performed better than model 1 I used this model in order to predict house prices of the validation data provided by Kaggle. Model 2 didn't exactly work as there were variables in model 2 that were not found in the validation data.

The following variables were found in the training data but not in the validation data:

```
MsZoning_c (all)
RoofMatl_ClyTile
GarageQual_Ex
Heating_Othw
PoolQC_Fa
```

As a result, these five variables needed to be removed from model 2. In addition, instead of splitting the training data 70/30 as I did for the previous two models, I trained the revised model 2 on the entire training set to further tune it. Once this was complete, I used the revised model 2, which accounted for the missing variables identified in red text above, to predict the house prices on the validation set. This resulted in an RMSLE of 0.12666 which was 1,228<sup>th</sup> place out of 4,781 teams (Top 33%). One area of improvement would be to use regularization methods such as lasso



or ridge on the revised model 2. However I couldn't get it to work but did include my attempt at using it in the R-code.

## Appendix

### Leaderboard Screenshot:

Overview	Data	Notebooks	Discussion	Leaderboard	Rules	Team	My Submissions	Submit Predictions
1223	ShawnTay						0.12661	5 12d
1224	Ivdao11						0.12662	1 19d
1225	minority2						0.12663	3 1mo
1226	kirankmadan						0.12664	14 1mo
1227	sheng						0.12665	4 1mo
1228	Neelan M						0.12666	8 now
<b>Your Best Entry ↑</b> You advanced 832 places on the leaderboard! Your submission scored 0.12666, which is an improvement of your previous score of 0.13875. Great job! <a href="#">Tweet this!</a>								
1229	João Moura		</> Data analysis and f...				0.12669	2 3d
1230	SeuMadruga		</> House Prices - EDA...				0.12670	4 2mo
1231	Zefeng Cai						0.12671	3 1mo
1232	WangYu #2						0.12671	3 2mo
1233	AK26995						0.12672	1 18d
1234	SARAVANA KUMAR MURTHY						0.12672	16 17d
1235	Leo Du						0.12673	4 2d

### Your Competitions

Active	Closed	Pinned	Hosted
	<b>Predict Future Sales</b> Final project for "How to win a data science competition" Coursera course Playground • 8 months to go • 6587 Teams		<b>Kudos</b>
	<b>House Prices: Advanced Regression Techniques</b> Predict sales prices and practice feature engineering, RFs, and gradient boosting Getting Started • Ongoing • 4781 Teams		<b>Knowledge</b>

## Model 1:

```
lm(formula = SalePrice ~ MSSubClass + LotArea + OverallQual +  
  OverallCond + YearBuilt + YearRemodAdd + BsmtFinsF1 + BsmtFinsF2 +  
  X2ndFlrSF + TotalSF + LowQualFinsF + GrLivArea + BsmtFullBath +  
  FullBath + HalfBath + BedroomAbvGr + KitchenAbvGr + TotRmsAbvGrd +  
  Fireplaces + GarageCars + GarageArea + WoodDeckSF + ScreenPorch +  
  PoolArea + MiscVal + MoSold, data = train_set)
```

## Model 2:

```
lm(formula = SalePrice ~ LotArea + OverallQual + OverallCond +  
  YearBuilt + YearRemodAdd + BsmtFinsF1 + BsmtFinsF2 + BsmtUnfSF +  
  TotalSF + LowQualFinsF + GrLivArea + BsmtFullBath + HalfBath +  
  Fireplaces + GarageCars + WoodDeckSF + `3SsnPorch` + ScreenPorch +  
  PoolArea + `MSZoning_C (all)` + MSZoning_FV + Alley_Pave +  
  LandContour_Bnk + LandContour_Low + Utilities_AllPub + LotConfig_Corner +  
  LotConfig_CulDSac + LotConfig_FR2 + Landslope_Gtl + Landslope_Mod +  
  Neighborhood_Blmngtn + Neighborhood_CollGr + Neighborhood_Crawfor +  
  Neighborhood_Edwards + Neighborhood_Gilbert + Neighborhood_IDOTRR +  
  Neighborhood_MeadowV + Neighborhood_Mitchel + Neighborhood_NAMES +  
  Neighborhood_NoRidge + Neighborhood_NPkVill + Neighborhood_NridgHt +  
  Neighborhood_NWAmes + Neighborhood_OldTown + Neighborhood_Sawyer +  
  Neighborhood_StoneBr + Condition1_Artery + Condition1_Feedr +  
  Condition1_PosA + Condition1_RRAe + Condition1_RRNe + Condition2_Feedr +  
  Condition2_PosN + BldgType_1Fam + Housestyle_2.5Unf + Housestyle_2Story +  
  Roofstyle_Flat + Roofstyle_Gable + RoofMatl_ClyTile + RoofMatl_CompShg +  
  Exterior1st_AsbShng + Exterior1st_BrkComm + Exterior1st_BrkFace +  
  Exterior1st_MetalSd + Exterior2nd_AsbShng + Exterior2nd_BrkFace +  
  Exterior2nd_CmentBd + Exterior2nd_Stone + Exterior2nd_VinylSd +  
  ExterQual_Ex + ExterCond_Ex + Foundation_BrkTil + Foundation_CBlock +  
  Foundation_PConc + Foundation_Slab + Foundation_Stone + BsmtQual_Fa +  
  BsmtQual_Gd + BsmtQual_TA + BsmtCond_Po + BsmtExposure_Av +  
  BsmtExposure_Gd + BsmtExposure_Mn + BsmtFinType1_ALQ + BsmtFinType1_BLQ +  
  BsmtFinType1_LwQ + BsmtFinType1_Rec + BsmtFinType2_ALQ +  
  BsmtFinType2_GLQ + BsmtFinType2_LwQ + BsmtFinType2_Rec +  
  Heating_GasA + Heating_Grav + Heating_Othw + HeatingQC_Ex +  
  HeatingQC_Fa + CentralAir_N + KitchenQual_Ex + Functional_Maj2 +  
  Functional_Min1 + Functional_Min2 + Functional_Mod + Functional_Sev +  
  FireplaceQu_Gd + FireplaceQu_TA + GarageType_CarPort + GarageQual_Ex +  
  GarageQual_Po + GarageCond_Ex + GarageCond_Fa + GarageCond_Po +  
  PoolQC_Fa + Fence_Gdwo + Fence_Mnww + SaleType_Con + SaleType_ConLD +  
  SaleType_CWD + SaleType_New + SaleCondition_AdjLand + SaleCondition_Normal +  
  GarageFinish_Unf + Exterior2nd_MetalSd + MasVnrType_NA +  
  BsmtCond_TA, data = train_set)
```

## Model 2 Revised for Validation Data:

```
model2_Improved_validation<-lm(formula = SalePrice ~ LotArea + OverallQual + OverallCond +  
  YearBuilt + YearRemodAdd + BsmtFinsF1 + BsmtFinsF2 + BsmtUnfsF +  
  TotalSF + LowQualFinsF + GrLivArea + BsmtFullBath + HalfBath +  
  Fireplaces + GarageCars + WoodDecksF + `3SsnPorch` + ScreenPorch +  
  PoolArea + MSZoning_FV + Alley_Pave +  
  LandContour_Bnk + LandContour_Low + Utilities_AllPub + LotConfig_Corner +  
  LotConfig_CulDSac + LotConfig_FR2 + Landslope_Gtl + Landslope_Mod +  
  Neighborhood_Blmngtn + Neighborhood_CollgCr + Neighborhood_Crawfor +  
  Neighborhood_Edwards + Neighborhood_Gilbert + Neighborhood_IDOTRR +  
  Neighborhood_Meadowv + Neighborhood_Mitchel + Neighborhood_NAMES +  
  Neighborhood_NoRidge + Neighborhood_NPkvill + Neighborhood_Nridght +  
  Neighborhood_NWAmes + Neighborhood_OldTown + Neighborhood_Sawyer +  
  Neighborhood_StoneBr + Condition1_Artery + Condition1_Feendr +  
  Condition1_PosA + Condition1_RRAe + Condition1_RRNe + Condition2_Feendr +  
  Condition2_PosN + BldgType_1Fam + HouseStyle_2.5Unf + HouseStyle_2Story +  
  RoofStyle_Flat + RoofStyle_Gable + RoofMatl_Compshg +  
  Exterior1st_Asbshng + Exterior1st_BrkComm + Exterior1st_BrkFace +  
  Exterior1st_MetalSd + Exterior2nd_Asbshng + Exterior2nd_BrkFace +  
  Exterior2nd_CmentBd + Exterior2nd_Stone + Exterior2nd_VinylSd +  
  ExterQual_Ex + ExterCond_Ex + Foundation_BrkTil + Foundation_CBlocl +  
  Foundation_PConc + Foundation_Slab + Foundation_Stone + BsmtQual_Fa +  
  BsmtQual_Gd + BsmtQual_TA + BsmtCond_Po + BsmtExposure_Av +  
  BsmtExposure_Gd + BsmtExposure_Mn + BsmtFinType1_ALQ + BsmtFinType1_BLQ +  
  BsmtFinType1_LwQ + BsmtFinType1_Rec + BsmtFinType2_ALQ +  
  BsmtFinType2_GLQ + BsmtFinType2_LwQ + BsmtFinType2_Rec +  
  Heating_GasA + Heating_Grav + HeatingQC_Ex +  
  HeatingQC_Fa + CentralAir_N + KitchenQual_Ex + Functional_Maj2 +  
  Functional_Min1 + Functional_Min2 + Functional_Mod + Functional_Sev +  
  FireplaceQu_Gd + FireplaceQu_TA + GarageType_CarPort +  
  GarageQual_Po + GarageCond_Ex + GarageCond_Fa + GarageCond_Po +  
  Fence_Gdwo + Fence_Mnww + SaleType_Con + SaleType_ConLD +  
  SaleType_CWD + SaleType_New + SaleCondition_AdjLand + SaleCondition_Normal +  
  GarageFinish_Unf + Exterior2nd_MetalSd + MasVnrType_NA +  
  BsmtCond_TA, data = train)
```

***\*Trained this model using entire train set from Kaggle and removed following variables:***

MSZoning\_c (all)  
RoofMatl\_ClyTile  
GarageQual\_Ex  
Heating\_Othw  
PoolQC\_Fa