"AMES HOUSE PRICES" PROJECT REPORT

I. Introduction of "House Prices - Advanced Regression Techniques"

The "House Prices - Advanced Regression Techniques" competition on Kaggle is one of the most popular and well-known data science competitions in the industry. The competition was launched in 2016 and aimed to challenge data scientists and machine learning practitioners to build accurate regression models that could predict the sale prices of residential homes in Ames, Iowa. With a large and complex dataset of 79 explanatory variables, the competition offered a unique opportunity for participants to develop and test their skills in data cleaning, feature engineering, and model selection.

The goal of the House Prices - Advanced Regression Techniques competition was to develop regression models that could accurately predict the sale prices of residential homes in Ames, Iowa based on a set of 79 explanatory variables. To evaluate the performance of the models, submissions were scored using the Root Mean Squared Error (RMSE) metric, which measures the average difference between the predicted sale prices and the true sale prices. A lower RMSE score indicates a better-performing model, as it means the model's predictions were closer to the true sale prices.

II. Brief Description of the Original Dataset

The dataset contains a total of 1460 observations or rows, each corresponding to a unique property sale, and includes 79 explanatory variables. The explanatory variables can be broadly categorized as numeric and categorical variables. The numeric variables include features such as the total square footage of the house, the size of the garage, and the lot frontage. The categorical variables include features such as the type of foundation, the type of roof material, and the type of heating system. Also, some ordinal variables include features such as the quality and condition ratings for different aspects of the house such as the overall material and finish, the exterior, the kitchen, and the basement. In addition to the explanatory variables, the dataset includes the target variable, which is the sale price of each property. The sale price is a variable that ranges from $34,900 to $755,000 in the dataset.

III. Cleaning the Dataset

I first combined the training dataset and test dataset into **cleaned_combined_set** to start cleaning by using **rbind()** function. Since **SalePrice** does not exist in the test dataset, I created a new column and assigned NA value to it. Additionally, to differentiate which observations belong to training or test dataset, I created a column called **isTrain**.

```
test$SalePrice <- NA
train$isTrain <- 1
```

```
test$isTrain <- 0
combined_set <- rbind(train,test)
cleaned_combined_set <- combined_set
```

1) Handling missing values

- Categorical variables: For missing values in categorical variables, I replace the missing values by the most frequent level of the variable. Additionally, sometimes, missing values are not missing values, but they actually represent one level of the categorical based on the description of the dataset. For example, for **BsmtQual**, there were 81 NAs but these NAs do not mean missing values, and actually represent "No Basement" according to the data description. Therefore, I simply replace NA values with "No Basement" to avoid confusion. Examples are provided below.

```
##COLUMN 3: MSZoning
sum(is.na(cleaned_combined_set$MSZoning))
#There are 4 missing values
#Therefore, I replace the missing values by the most frequent level for MSZoning
summary(factor(cleaned_combined_set$MSZoning))
#RL is the most frequent value for MSZoning, so I assign it to missing values
cleaned_combined_set$MSZoning[is.na(cleaned_combined_set$MSZoning)] <- "RL"
sum(is.na(cleaned_combined_set$MSZoning))
```

```
##COLUMN 31: BsmtQual
sum(is.na(cleaned_combined_set$BsmtQual))
summary(factor(cleaned_combined_set$BsmtQual))
#81 NAs but these NAs do not mean missing values.
#They actually represent "No Basement" according to data description.
#Replace NA with "No Basement" to avoid confusion.
cleaned_combined_set$BsmtQual[is.na(cleaned_combined_set$BsmtQual)] <- "No Basement"
```

- Numeric variables: To deal with missing values of numeric variables, I first identify the relationship of the missing values and the corresponding categorical variables. For example, for **LotFrontage**, I replace missing values with the mean that corresponds to the category. Then, I calculate the mean value for each factor.

```
##COLUMN 4: LotFrontage
cleaned_combined_set$LotFrontage <- as.numeric(cleaned_combined_set$LotFrontage)
sum(is.na(cleaned_combined_set$LotFrontage))
#There are 486 missing values
#I replace missing values with the mean that corresponds to the category of MSZoning.
#Then, I calculate the mean value of LotFrontage for each factor of MSZoning when
#LotFrontage is not an NA value
cleaned_combined_set %>%
  group_by(MSZoning) %>%
```

```
  select(MSZoning, LotFrontage) %>%
  summarize(mean = mean(LotFrontage, na.rm = TRUE))
#Mean LotFrontage for each type of MSZoning
#  C        65.6
#  FV       59.5
#  RH       55.4
#  RL       74.1
#  RM       52.2
#Impute the mean LotFrontage by MSZoning into all NAs of LotFrontage
```

Similarly, for other numeric variables, I also try to locate the relationship between the numeric variables and the categorical variables based on the description to decide what to do with the missing values of the numeric variables. For example, for **MasVnrArea**, it is actually related to **MasVnrType**, and by assessing their relationship, I found out the missing values of **MasVnArea** actually correspond to **MasVnrType** = None (i.e., no Masonry veneer), so I simply replace missing values with 0. Likewise, for **BsmtFinSF1**, I assess the relationship between the missing values of this variable and **BsmtFinType1**, and I found out the missing values of **BsmtFinSF1** correspond to **BsmtFinType1** = No Basement, so I simply replaced the missing values with 0.

```
##COLUMN 27: MasVnrArea
sum(is.na(cleaned_combined_set$MasVnrArea))
#23 missing values
#MasVnrArea and MasVnrType are related according to the description.
#Need to view the fields with missing values and decide what to do.
cleaned_combined_set %>%
  select(MasVnrType, MasVnrArea) %>%
  filter(is.na(MasVnrArea))
#Missing values of MasVnrArea correspond to MasVnrType = None (i.e., no Masonry veneer).
#So we need to replace missing values with 0.
cleaned_combined_set$MasVnrArea[is.na(cleaned_combined_set$MasVnrArea)] <- 0
```

2) Correcting typos or measurement errors

For each categorical variable, I always use **summary()** function and check the data description to see if there are any typos of the categories. If typos are identified, I would replace them with the correct values. For example, for **BldgType,** there is an error/typo in "Twnhs" because it doesn't match with TwnhsI in the data set description, so I replace values "Twnhs" with "TWnhsI"

```
##COLUMN 16: BldgType
sum(is.na(cleaned_combined_set$BldgType))
#No missing value
summary(factor(cleaned_combined_set$BldgType))
```

```
#There is an error/typo in "Twnhs" because it doesn't match with TwnhsI in the data set description.
#I will replace values "Twnhs" with "TWnhsI" because there are no values for "TWnhsI".
cleaned_combined_set$BldgType[cleaned_combined_set$BldgType == "Twnhs"] <- "TwnhsI"
```

3) Recoding and/or augmenting existing variables

In the cleaning stage, I also recode or augment some existing variables. As mentioned above, when dealing with missing values of some categorical variables, I would identify the most frequent value and replace missing values with them.

Additionally, because I will split the cleaned combined set into training and test dataset in the modeling stage, to avoid the issue where some observations belong to one dataset but not in the other dataset, I would merge those values of those variables into nearby values. For example, for **MSSubClass**, level "150" only belongs to the training dataset but does not exist in the test dataset, so I merge "150" into "160".

```
##COLUMN 2: MSSubClass
sum(is.na(cleaned_combined_set$MSSubClass))
summary(factor(cleaned_combined_set$MSSubClass))
#0 missing value
#To avoid the problem later in the modeling stage, in which level "150"
#of MSSubclass belongs to the training dataset but does not exist in
#the test dataset, I merge this level to the nearby level to avoid the issue.
cleaned_combined_set$MSSubClass[cleaned_combined_set$MSSubClass == "150"] <- 160
cleaned_combined_set$MSSubClass <- as.factor(cleaned_combined_set$MSSubClass)
```

For some variables related to the timeline of the observation, such as **MoSold** (month the house is sold), or **YrSold** (year the house is sold), I set the variable into a categorical variable as it would be more reasonable to explain the coefficient.

```
##COLUMN 77: MoSold
sum(is.na(cleaned_combined_set$MoSold))
#No missing value
#Similar to YearBuilt, it would be more reasonable to explain the coefficient of MoSold
#when MoSold is a categorical variable.
cleaned_combined_set$MoSold <- as.factor(cleaned_combined_set$MoSold)
```

4) Dealing with outliers

To deal with outliers of numeric variables, I used both **boxplot()** and **ggplot()** functions to construct a visualization between **SalePrice** and the variable. The pattern from both visualizations helps me better understand and identify the outliers. Once outliers are detected, I would replace them with the mean value of the variable. For example:

```
##COLUMN 4: LotFrontage
[...]
#Identify outliers.
```

```
boxplot(cleaned_combined_set$LotFrontage)
ggplot(cleaned_combined_set, aes(SalePrice, LotFrontage)) + geom_point()
#Outliers of LotFrontage seem to occur around 300.
#Replace outliers with mean.
cleaned_combined_set$LotFrontage <- ifelse(cleaned_combined_set$LotFrontage > 300,
                                mean(cleaned_combined_set$LotFrontage),
                                cleaned_combined_set$LotFrontage)
```

For variables such as **BedroomAbvGr** (number of bedrooms above grade), or **TotRmsAbvGrd** (total number of rooms above grade), with values ranging from 0~8, I use as.numeric() function, and then us either **ggplot()** or **boxplot()** to see the frequency of values. If there is only one observation belonging to one value, I will merge it with a nearby value.

```
##COLUMN 52: BedroomAbvGr
cleaned_combined_set$BedroomAbvGr <- as.numeric(cleaned_combined_set$BedroomAbvGr)
[...]
#Outliers
boxplot(cleaned_combined_set$BedroomAbvGr)
ggplot(data = cleaned_combined_set, aes(SalePrice, BedroomAbvGr)) + geom_point()
#Only 1 observation belongs to "8", therefore, I will merge it with the nearby
#level, which is "6"
cleaned_combined_set$BedroomAbvGr[cleaned_combined_set$BedroomAbvGr == "8"] <- 6
```

```
##COLUMN 55: TotRmsAbvGrd
cleaned_combined_set$TotRmsAbvGrd <- as.numeric(cleaned_combined_set$TotRmsAbvGrd)
[...]
ggplot(data = cleaned_combined_set, aes(SalePrice, TotRmsAbvGrd)) + geom_point()
#I will merge levels that have only one observations with the nearby levels
cleaned_combined_set$TotRmsAbvGrd[cleaned_combined_set$TotRmsAbvGrd == "2"] <- 3
cleaned_combined_set$TotRmsAbvGrd[cleaned_combined_set$TotRmsAbvGrd == "13"] <- 12
cleaned_combined_set$TotRmsAbvGrd[cleaned_combined_set$TotRmsAbvGrd == "14"] <- 12
cleaned_combined_set$TotRmsAbvGrd[cleaned_combined_set$TotRmsAbvGrd == "15"] <- 12
summary(factor(cleaned_combined_set$TotRmsAbvGrd))
```

5) Transformations

When cleaning the dataset, I also checked the skewness of the variables. For the skewness value between -0.5 to 0.5, which means the data is symmetric, and between -1 and -0.5 or between 0.5 and 1, which means the data is moderately skewed, I apply no transformation. However, if the skewness value is less than -1 or greater than 1, the data is highly skewed and I apply log transformation to improve the skewness. Example is provided below.

```
##COLUMN 5: LotArea
[...]
#Skewness
ggplot(cleaned_combined_set, aes(LotArea)) + geom_histogram()
```

```
skewness(cleaned_combined_set$LotArea)
#The value is 3.75, highly skewed.
#The histogram also displays right skewness.
#Try log transformation.
ggplot(cleaned_combined_set, aes(log10(LotArea))) + geom_histogram()
skewness(log10(cleaned_combined_set$LotArea))
#The value is -0.788, only moderately skewed, which is good
cleaned_combined_set$LotArea <- log10(cleaned_combined_set$LotArea)
```

6) Creating new variables

For **YearBuilt** and **YearRemodAdd** variables, I created two new variables called **HomeAge** and **AgeSinceRemod** respectively by calculating *(2022 – YearBuilt/YearRemodAdd) + 1*. In addition, to avoid redundant variables when there are variables related to each other, I combine them and create a new variable. New variables are **HomeTotalSF** (combing all the variables regarding the square feet of particular space), **TotalFullBath** (combing all the variables regarding the number of full bathrooms), **TotalHalfBath** (combing all the variables regarding the number of full bathrooms). Examples are provided below.

```
##COLUMN 20: YearBuilt
sum(is.na(cleaned_combined_set$YearBuilt))
#No missing value
#Create a new variable called "HomeAge"
cleaned_combined_set$HomeAge <- (2022 - cleaned_combined_set$YearBuilt) +1
```

```
#Create a new variable/column called "HomeTotalSF" by combing
#all the variables regarding the square feet of particular space.
cleaned_combined_set$HomeTotalSF <- cleaned_combined_set$TotalBsmtSF + cleaned_combined_set$X1stFlrSF +
cleaned_combined_set$X2ndFlrSF
[…]
#I will remove TotalBsmtSF, X1stFlrSF, and X2ndFlrSF because they have been combined into a single variable.
cleaned_combined_set <- subset(cleaned_combined_set, select = -c(TotalBsmtSF, X1stFlrSF, X2ndFlrSF))
```

7) Removing variables

After creating new variables, I remove the existing variables because they have been combined into a single variable. Example can be seen in above in the creation of **HomeTotalSF** variable. The second case I remove variables is when there are too many observations belong to one value. Based on my own analysis, when there are ~80% of the total number of observations belonging to one value, it will not provide much insightful data. Therefore, I would remove them. Example is provided below.

```
##COLUMN 7: Alley
sum(is.na(cleaned_combined_set$Alley))
summary(factor(cleaned_combined_set$Alley))
```

```
#There are 2721 missing values, which means "No alley access" according to
#the description
#However, given that 2721 out of 2919 obs belong to "No alley access",
#I am removing this varable
cleaned_combined_set <- subset(cleaned_combined_set, select = -c(Alley))
```

IV. Diagnostic Tests and Prescriptions Employed

After running the plain vanila model, I execute diagnostic tests.

- Multicollinearity issue: To check for multicollinearity, I use vif() function. Since this is a large dataset with various dimensions, I assess multicollinearity by looking at GVIF^(1/(2*Df)), and 5 is the cutoff value. MasVnrArea has the value of around 5.135, and seems to be the only culprit for multicollinearity. Prescription: I remove this variable from both training and test dataset. Then, I rerun the new model.
- Non-linearity issue: After using par() function and plot() function to product plots, I look at the residual-fitted plot to assess non-linearity issue. The residual-fitted plot shows a horizontal pattern, which is good and indicates no non-linearity issue exists. Therefore, no transformation is needed.
- Heterscedasticity issue: I look at the scale-location plot to assess heterscedasticity. The overall pattern shows pretty equally spread points. Therefore, I conclude that no heterscedasticity issue exists and no transformation is needed.
- Influential points: I apply Cook's Distance to evaluate if there are any influential points. The result is 0. Additionally, although we got a warning messages with the following points being high leverage points: 326, 376, 534, 949, 1012, 1188, 1371, it turns out they are not influential points. Therefore, no transformation is needed.

V. Results

Method	Score
Plain vanilla model	0.12989
Prescribed model	0.12981
Forward AIC	0.12874
Backward AIC	0.12893
Hybrid AIC	0.12853
Lasso	0.12891
Ridge	0.12891

Bagging	0.14962
Random forest	0.17029
Boosting	0.13658

After diagnostic tests and prescriptions, I use various analytical techniques to predict SalePrice. Of all the techniques, hybrid AIC method yields the lowest score. Considering the interpretability and the model accuracy, hybrid AIC seems to give the best regression model.

VI. Insights

After determining hybrid AIC gives out the best regression model, I look at summary() of the method to identifying important predictors in the dataset. When the p-values are below 0.05, the predictors can be considered as significant predictors to predict SalePrice. Based on this, the top factors that have the lowest p-values and are the most important to make sale prices predictions include:

- **HomeTotalSF**: This variable indicates the total square feet of the house, which is an important predictor of the sale price. Larger houses generally sell for more money, all else being equal.
- **OverallQual**: This variable is an ordinal variable that ranges from 1 to 10 and represents the overall quality of the house. The model has identified that houses with an OverallQual of 1 are particularly important in predicting the sale price.
- **HomeAge**: This variable indicates the age of the house, which is an important factor in determining its value. Older houses may have depreciated in value due to wear and tear, whereas newer houses may be more desirable and sell for a higher price.
- **MSSubClass**: This variable indicates the type of dwelling involved in the sale. The model has identified that houses that are Duplex – all styles and ages or 2-story pud – 1946 & newer are particularly important in predicting the sale price.
- **MSZoning**: This variable indicates the general zoning classification of the sale. Different types of zoning may affect the value of a property, so this variable is an important predictor.
- **GrLivArea**: This variable indicates the above-grade living area square feet, which is an important factor in determining the size of the house and its value.
- **LotArea**: This variable indicates the lot size in square feet, which is an important factor in determining the size of the property and its value.
- **Fireplaces**: This variable indicates the number of fireplaces. Fireplaces may be seen as a desirable feature and could increase the value of the property.

- **AgeSinceRemod**: This variable indicates the age of the house since its last remodeling, which is an important factor in determining the condition and value of the property.
- **BsmtFinSF1**: This variable indicates the type 1 finished square feet of the basement. A finished basement may be seen as a desirable feature and could increase the value of the property.
- **TotalHalfBath**: This variable indicates the total number of half bathrooms in the house. Bathrooms are an important feature of a house and could affect its value.
- **TotalFullBath**: This variable indicates the total number of full bathrooms in the house. Again, bathrooms are an important feature of a house and could affect its value.

Overall, the results highlight that larger houses with higher quality finishes and desirable features such as fireplaces and finished basements tend to sell for more money. This information can help real estate agents to better understand what types of properties are in demand and what features they should focus on when developing or renovating a property. In addition, our model has identified that certain types of houses, such as duplexes and 2-story puds, are particularly important in predicting the sale price. By sharing this information, we can help them to better understand what types of properties are currently in demand and what features they should focus on to attract potential buyers. The insights could also provide a better understanding of the market and help people to make informed decisions about pricing and property development. By understanding what features are important to potential buyers, they can develop or renovate properties that are more likely to attract interest and sell for a higher price.

The findings of the top important predictors are not out of my expectation. Below is the interpretation of the coefficients of the top five most important predictors which real estate agents should take into consideration:

- **HomeTotalSF**: Holding all other variables constant, for every 1 sqft increase of the house, the sale price will on average increase by 1.000093%
- **HomeAge**: Holding all other variables constant, for every 1 year increase in the age of the house, the sale price will on average increase by 0.99793%
- **OverallQual**: Holding everything else constant, the median price of homes that have the overall quality of 2 is on average around 80% higher than that of homes that have the overall quality of 1
- **MSZoning**:
    - MSZoningFV: Holding everything else constant, the median price of homes whose zoning is classified as Floating Village Residential is on average 56.89171% higher than that of homes that are classified as Commercial

  - MSZoningRH: Holding everything else constant, the median price of homes whose zoning is classified as Residential High Density is on average 50.41807% higher than that of homes that are classified as Commercial
  - MSZoningRL: Holding everything else constant, the median price of homes whose zoning is classified as Residential Low Density is on average 50.83426% higher than that of homes that are classified as Commercial
  - MSZoningRM: Holding everything else constant, the median price of homes whose zoning is classified as Residential Medium Density is on average 45.94862% higher than that of homes that are classified as Commercial
- **GrLivArea**: Holding all other variables constant, for every 1 sqft increase in the above-grade living area, the sale price will on average increase by 1.000148%

Apart from these predictors, there are other slightly less important predictors, whose p-values are below 0.05 but higher than the values of the predictors mentioned above. For example, some of them include LotFrontage (Linear feet of street connected to property), or GarageArea (Size of garage in square feet), and others. However, since they are not as important as predictors mentioned above, I will not mention or interpret all of them.

VII. Reflections

In conclusion, this project provided valuable insights into the process of developing a machine learning model to predict housing prices. Through data exploration, cleaning, and various analytical techniques, I was able to identify the most important predictors for predicting SalePrice, and build and evaluate several regression models to find the best performing one. From this project, I also learned about the importance of the techniques in developing an accurate and robust model. I also learned how to interpret and analyze the coefficients of the model to gain insights into the relationships between predictors and the response.

One obstacle that I encountered during this project was dealing with missing data. I solved this problem by using appropriate imputation techniques such as mean imputation.

Overall, this project was a great learning experience and provided a solid foundation in machine learning and regression modeling. This is especially helpful for me to work on the other datasets in the future to develop an accurate and more reliable model.