AMES House market report

Xieyuan Huang

```
install.packages("rmarkdown")

install.packages("Tidyverse")

install.packages("Metrics")

install.packages("GGally")

library(knitr)
library(tidyverse)

library(Metrics)

library(GGally)
```

Abstract

In this report, I am aiming to to provide an insights of Ames city housing market and then provide a solid model for predicting the sale price by analysising the dataset with research and data support. The report starts from conducting deep background research and future policy of the housing market from the government of Ames city. Then, I identified 5 variables on neighborhood, living areas, overall quality, sale condition and house style that are interested and valuable to investigate. After that, Cleaning the dataset was performed by finding the best treatment for each variables instead of just deleting them. In the EDA section, I used barplot, histogram, heat map, scatterplot plot matrix, scatter plot, box plot, violin plot and jitter points with introducing different color and smooths to explain the defined problems. To construct a reliable model, the first model was created by the top five numeric variables with highest correlation with the sale price. The second model was created by the five selected variables. The third model was created by the combination of the first model's variables and the second model's. In conclusion, the best result reaches the R squared value 0.845 which provides a solid model for investor to invest and investment suggestion were stated at the end of the report. The future improvement and work also stated at the end of the report.

Problem Identification

In order to conduct a high quality exploratory data analysis, understand the housing market and the nature of Ames are necessary.

The housing market refers to the supply and demand for houses in a typical region. The one of the most important elements of the housing market is the average house prices and trend.

There are 66,023 population in Ames, Iowa. It is regarded as one of the lowest unemployment rate among the US. According to the Ames Plan 2040 conducted by Ames city council, it estimates that 6,400 housing needs to be built with an average size 2.3 people so that it can match the estimated population growth about 15,000. According to the survey provided by the coucil, there are 67% of residents believe that more housing options can enhance there life quality in the next 20 years. The summarized principles have been defined as following: Expand housing choice and attainability for people of all income ranges. Maintain the quality of existing neighborhoods while also encouraging reinvestment and enhancement of existing housing stock. Advance identification and redevelopment of redirection areas. Those three policies have been decomposed with detail policies. More information and discusstion will be taken in the later sections.

Overall, the housing market in Ames city has become important and in demand. In this report, I am aiming to extract the insights and patterns for all investers who are interested in the housing market in Ames city. There are five questions that the report mainly focuses on building the model.

First, Identify which suburb/location had the biggest SalePrice by plotting and examining the sale prices cross different suburbs Second, identify the relation between GrLivArea against sales price Third, Identify whether there is a corralation between OverallQual and SalePrice? Fourth, Identify whether there is a corralation between SaleCondition and SalePrice? Fifth, Identify whether there is a corralation between HouseStyle and SalePrice?

The annswers of those questions will be covered in the later part.

Date processing

First of all, import the “train” data for eda analysis. As we can see from the graph, there are 81 columns in the dataframe and 1460 rows. The first 6 rows of data are displayed.

```
df= read.csv(file.choose(), header=TRUE) #Import data to df
dim(df) #Display the dimension of the data
```

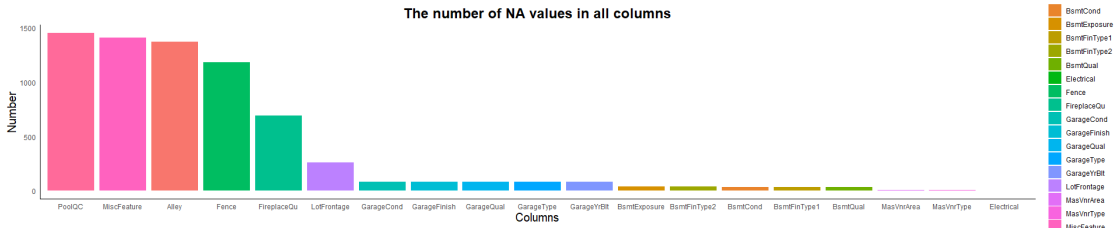
```
## [1] 1460   81
```

Second, checking the Na value in the dataset According to the graph, the variables with Na value are LotFrontage(259), Alley(1369), MasVnrType(8), MasVnrArea(8), BsmtQual(37), BsmtCond(37), BsmtExposure(38), BsmtFinType1(37), BsmtFinType2(38), Electrical(1), FireplaceQu(690), GarageType(81), GarageYrBlt(81), GarageFinish(81), PoolQC(1453), Fence(1179) and MiscFeature(1406).

It will not be a good idea to remove all Na value in this dataset since some of the Na values do not represent missing value in the scenarios. Thus, the missing values should be either replaced by the mean for the numeric variables, the suitable variables for categorical

variables or the whole rows should be removed. Here are the analysis and treatment for the Na values of this dataset. Here are the visualization of the number of NA values in different columns

```
# Create NA_sum dataframe to store the summary of NA values
NA_sum= as.data.frame(colSums(is.na(df)))
# Process the data to feed the ggplot code

colnames(NA_sum) = "Number"
NA_sum$Variables= row.names(NA_sum)
NA_sum = NA_sum %>% filter(NA_sum$Number !=0)

#Display the bar plot of the distribution of NA
ggplot(NA_sum, aes(x= reorder(Variables, -Number), y=Number))+
  geom_bar(aes(fill= Variables), stat = "identity")+
  ylab("Number") +
  xlab("Columns") +
  ggtitle("The number of NA values in all columns") +
  theme_classic()+
  theme(plot.title = element_text(size= 18, face= "bold", hjust= 0.5),
        axis.title.x = element_text(size= 15, hjust= 0.5),
        axis.title.y = element_text(size= 15, hjust= 0.5),
        axis.ticks = element_blank())
```



LotFrontage(259): According to the description of the dataset from Kaggle, it represents the linear feet of street connected to property, which indicates that this is a numeric variables. The Na value can be considerred as missing values or 0. In this case, I assume those Na values represent 0 feet connected to the street since it is possible that the property just next to the street which means that the linear feet of street connected to property are 0.

```
df$LotFrontage[is.na(df$LotFrontage)]=0 #Replace NA with 0
```

Alley(1369): According to the description of the dataset from Kaggle, it represents the type of alley access to property, which indicates that it is a categorical variable. The NA value in here represents “No alley access”. They are not a missing value. So, I am going to replace “NO” to all Na values in Alley.

```
df$Alley[is.na(df$Alley)]="NO" #Replace NA with "NO"
```

MasVnrType(8): According to the description of the dataset from Kaggle, it represents the masonry veneer type, which indicates that it is a categorical variable. However, the Na values in this column are considered as missing values, not representing no type of alley

access to property, because there is a value named “None” to represent no mansonry veneer type. In this case, the best treatment for this column is to remove those observations since we cannot get the average of the categorical data. If we use the most frequent values, it will create an uncertainty and bias for the analysis later on. 8 observations only take 0.5% out of the whole observations, so removing them is reasonable with an extremely small impact for the dataset. Also, MasVnrArea(8) and MasVnrType(8) are in the same observations. If these 8 observations are removed, MasVnrType(8) and MasVnrArea(8) can be addressed at the same time.

```
#Instead of removing the rows that contains Na values in MasVnrType, I only pick the data that there is no Na values in MasVnrType. It gets the same result as I expected.
df=df[!is.na(df$MasVnrType), ]
```

As we cans see from the summary of Na values in df, MasVnrType(8) and MasVnrArea(8) got 0 Na values now. So, MasVnrType(8) and MasVnrArea(8) have been addressed at the same time.

BsmtQual(37): According to the description of the dataset from Kaggle, it represents Evaluates the height of the basement, which indicates that it is a categorical variable. Since Na values represent “No Basement” not missing values, the best option to solve this problem is to replace Na values with “No Basement”.

```
df$BsmtQual[is.na(df$BsmtQual)]="No Basement" #Replace NA with "No Basement"
```

BsmtCond(37): According to the description of the dataset from Kaggle, it represents the evaluates the general condition of the basement, which indicates that it is a categorical variable. Since Na values represent “No Basement” not missing values, the best option to solve this problem is to replace Na values with “No Basement”.

```
df$BsmtCond[is.na(df$BsmtCond)]="No Basement" #Replace NA with "No Basement"
```

BsmtExposure(38): According to the description of the dataset from Kaggle, it represents the refers to walkout or garden level walls, which indicates that it is a categorical variable. Since Na values represent “No Basement” not missing values, the best option to solve this problem is to replace Na values with “No Basement”.

```
df$BsmtExposure[is.na(df$BsmtExposure)]="No Basement" #Replace NA with "No Basement"
```

BsmtFinType1(37): According to the description of the dataset from Kaggle, it represents the rating of basement finished area, which indicates that it is a categorical variable. Since Na values represent “No Basement” not missing values, the best option to solve this problem is to replace Na values with “No Basement”.

```
df$BsmtFinType1[is.na(df$BsmtFinType1)]="No Basement" #Replace NA with "No Basement"
```

BsmtFinType2(38): According to the description of the dataset from Kaggle, it represents the rating of basement finished area (if multiple types), which indicates that it is a

categorical variable. Since Na values represent “No Basement” not missing values, the best option to solve this problem is to replace Na values with “No Basement”.

```
df$BsmtFinType2[is.na(df$BsmtFinType2)]="No Basement" #Replace NA with "No Basement"
```

Electrical(1): According to the description of the dataset from Kaggle, it represents the electrical system, which indicates that it is a categorical variable. However, there is no description for any Na values, so this one Na value either represents no electric system or an unknown electric system. In order to maintain the integrity of the data, I decide to remove this one observation.

```
#Instead of removing the rows that contains Na values in Electrical[1]
df=df[!is.na(df$Electrical), ]
```

FireplaceQu(690): According to the description of the dataset from Kaggle, it represents the fireplace quality, which indicates that it is a categorical variable. Since Na values represent “No fireplace”, not missing values, the best option to solve this problem is to replace Na values with “No fireplace”.

```
df$FireplaceQu[is.na(df$FireplaceQu)]="No Fireplace" #Replace NA with "No Fireplace"
```

GarageType(81): According to the description of the dataset from Kaggle, it represents the garage location, which indicates that it is a categorical variable. Since Na values represent “No Garage”, not missing values, the best option to solve this problem is to replace Na values with “No Garage”.

```
df$GarageType[is.na(df$GarageType)]="No Garage" #Replace NA with "No Garage"
```

GarageYrBlt(81): According to the description of the dataset from Kaggle, it represents the year garage was built, which indicates that it is a numeric variable. Since Na values represent “No Garage”, not missing values, the best option to solve this problem is to replace Na values with zero to represent the fact that there is no garage.

```
df$GarageYrBlt[is.na(df$GarageYrBlt)]=0 #Replace NA with zero since this column is integer type
```

GarageFinish(81): According to the description of the dataset from Kaggle, it represents the interior finish of the garage, which indicates that it is a categorical variable. Since Na values represent “No Garage”, not missing values, the best option to solve this problem is to replace Na values with “No Garage”.

```
df$GarageFinish[is.na(df$GarageFinish)]="No Garage" #Replace NA with "No Garage"
```

GarageQual(81): According to the description of the dataset from Kaggle, it represents the garage quality, which indicates that it is a categorical variable. Since Na values represent “No Garage”, not missing values, the best option to solve this problem is to replace Na values with “No Garage”.

```
df$GarageQual[is.na(df$GarageQual)]="No Garage" #Replace NA with "No Garage"
```

GarageCond(81): According to the description of the dataset from Kaggle, it represents the garage condition, which indicates that it is a categorical variable. Since Na values represent “No Garage”, not missing values, the best option to solve this problem is to replace Na values with “No Garage”.

```
df$GarageCond[is.na(df$GarageCond)]="No Garage" #Replace NA with "No Garage"
```

PoolQC(1453): According to the description of the dataset from Kaggle, it represents the pool quality, which indicates that it is a categorical variable. Since Na values represent “No Pool”, not missing values, the best option is to replace NA with “No Pool”

```
df$PoolQC[is.na(df$PoolQC)]="No Pool" #Replace NA with "No Pool"
```

Fence(1179): According to the description of the dataset from Kaggle, it represents the fence quality, which indicates that it is a categorical variable. Since Na values represent “No Fence”, not missing values, the best option is to replace NA with “No Fence”

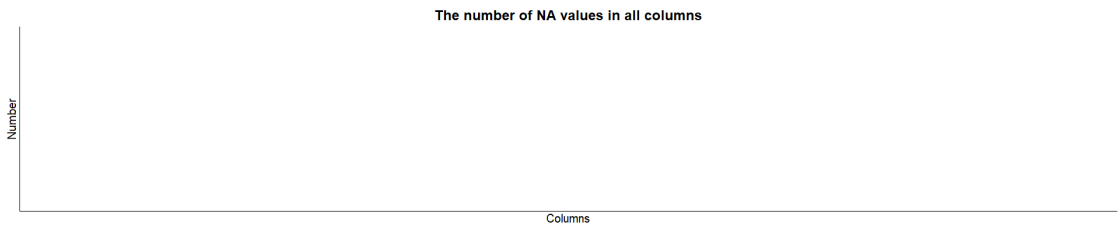
```
df$Fence[is.na(df$Fence)]="No Fence" #Replace NA with "No Fence"
```

MiscFeature(1406): According to the description of the dataset from Kaggle, it represents the miscellaneous feature not covered in other categories, which indicates that it is a categorical variable. Since Na values represent “No MixFeature”, not missing values, the best option is to replace NA with “None”

```
df$MiscFeature[is.na(df$MiscFeature)]="None" #Replace NA with "No
MiscFeature"
```

Display the result that there is no NA value in the trainnig set.

```
# Create NA_sum dataframe to store the summary of NA values
NA_sum= as.data.frame(colSums(is.na(df)))
# Process the data to feed the ggplot code

colnames(NA_sum) = "Number"
NA_sum$Variables= row.names(NA_sum)
NA_sum = NA_sum %>% filter(NA_sum$Number !=0)

#Display the bar plot of the distribution of NA
ggplot(NA_sum, aes(x= reorder(Variables, -Number), y=Number))+
  geom_bar(aes(fill= Variables), stat = "identity")+
  ylab("Number") +
  xlab("Columns") +
  ggtitle("The number of NA values in all columns") +
  theme_classic()+
  theme(plot.title = element_text(size= 18, face= "bold", hjust= 0.5),
        axis.title.x = element_text(size= 15, hjust= 0.5),
        axis.title.y = element_text(size= 15, hjust= 0.5),
        axis.ticks = element_blank())
```



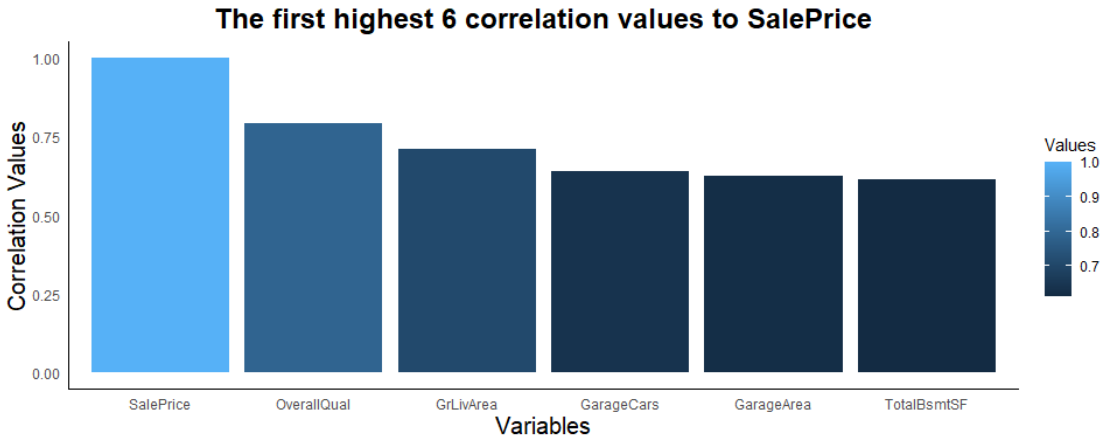
EDA

In this section, I am going to identify 5 variables that are correlated with the sales price based on the 5 problems I proposed in the first section by plotting graph and emerging with the specialists' advices.

Beofore starting to answer the question, to extract insight of the dataset is preferred. Since building a model for predicting the sale price of the property in Ames and the NA values have been removed, finding the correlation between SalePrice and each variables is taken into considerations. However, there are 38 numeric variables in this dataset, which makes impossible to use correlation matrix to give a clear view. So, I am going to use bar plot to show the 5 variables with the highest correlation value using cor().

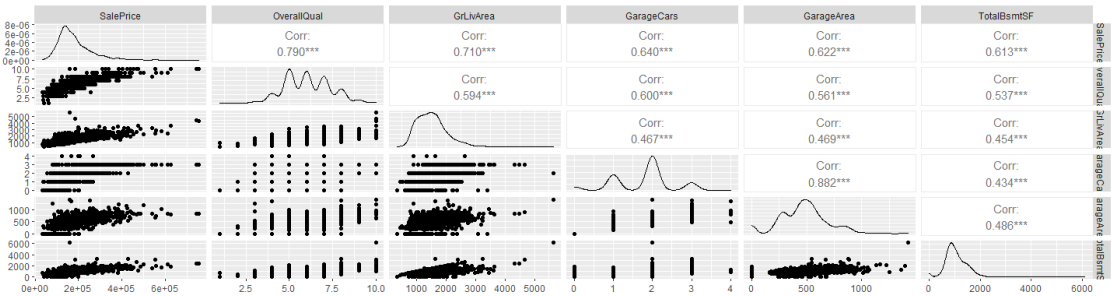
```
#Extract numeric columns
df_num= select_if(df, is.numeric)
#Create a dataset to store the correlation values and the corresponding
variables

cor_sum= data.frame( Variables= names(df_num), Values=0)#Assume default value
is 0

#Assign correlation value to each row
for (i in seq(1:length(names(df_num)))) {

  cor_sum$Values[i]= cor(df_num$SalePrice, df_num[i])
}

#Extract the 5 variables with top 5 correlation values except SalePrice
itself
cor_sum= cor_sum[order(-cor_sum$Values),]
#Only take the first 6 rows including SalePrice itself
cor_sum= head(cor_sum,6)
#Display the bar plot of the correlation values among all numeric variables
ggplot(cor_sum, aes(x= reorder(Variables, -Values), y=Values))+
  geom_bar(aes(fill= Values), stat = "identity")+
  ylab("Correlation Values") +
  xlab("Variables") +
  ggtitle("The first highest 6 correlation values to SalePrice") +
  theme_classic()+
  theme(plot.title = element_text(size= 18, face= "bold", hjust= 0.5),
        axis.title.x = element_text(size= 15, hjust= 0.5),
```

```
        axis.title.y = element_text(size= 15, hjust= 0.5),
        axis.ticks = element_blank())
```



As we can see from the scatterplot plot matrix, the first row has clearly shows the correlation between those 5 variables and SalePrice. Moreover, the histogram of SalePrice indicates that the distribution is skewed to the right, not symmetric. Log transformation can be applied in the later section.

```
#Create a dataset for creating a scatterplot plot matrix
ScatterPlot_p= select(df, c("SalePrice","OverallQual", "GrLivArea",
"GarageCars", "GarageArea",  "TotalBsmtSF"))
ggpairs(ScatterPlot_p)
```



So, as we can see from the graph, the highest numeric correlated variables with Saleprice are OverallQual, GrLivArea, GarageCars, GarageArea and TotalBsmtSF. So this group will be used as one of my model.

The first question

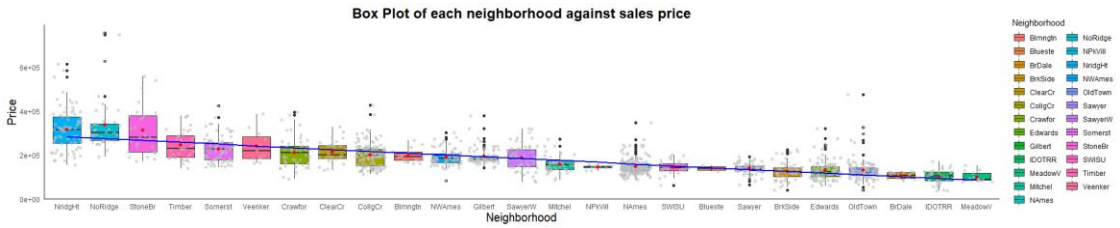
First, Identify which suburb/location had the biggest SalePrice by plotting and examining the sale prices cross different suburbs. The reason why I will look at neighborhoods first is that the location of the housing property is one of the most important factor that determines the price of the house. Also, despite of the location, the behaviour of the neighbors around the property can also impact on the price of the property. According to the Appraisal Institute, a bad neighbor could potentially reduce your home's value up to 10%. It includes neighbors' criminal background, financial level and level of

maintainance.(How Your Neighbors Affect Your Property Value, 2021) For investigating the relation between neighborhoods and sale price, I plot a boxplot with displaying the mean sales price as the red point and the sort those neighborhoods in descending order by the median. Here are the boxplot. As we can see from the graph, NridgHt has the highest median sale price while MeadowV has the lowest meidan sale price. However, the most popular area are CollgCr, NAmes, Edwards and OldTown.

In conclusion, it has been clearly showed that neighborhoods has a significant correlation with the sale price supported by some researchs and the relation between sale price and neighborhoods is shown below by box plot. As the graph shown, the neighborhoods are sorted by the median of each neighborhoods since the median represents the majority of the price range. The highest one is NridgHT and the lowest one is MeadowV. Also the number of residuals of each group is displayed by the point in grey. The more details will be discuessed on Further Processing section.

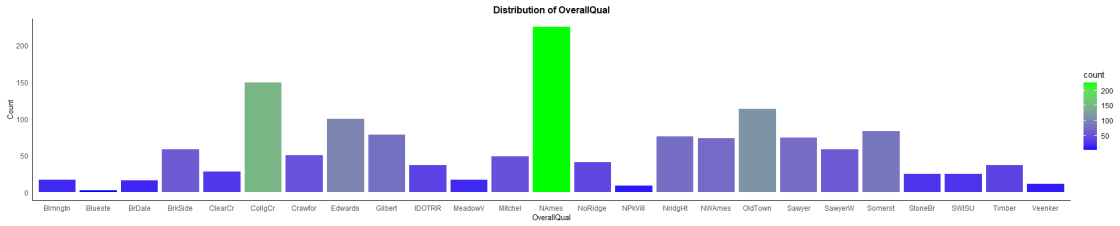
```
#Display boxplot of each neighborhood against sales price with a linear trend
ggplot(df, aes(x= reorder(Neighborhood, -SalePrice, FUN = median),
y=SalePrice))+
  geom_boxplot(aes(fill= Neighborhood))+
  geom_jitter(aes(alpha = I(.5)), color="grey")+
  stat_summary(fun = mean,
               geom = "point",
               shape =20,
               size =3,
               color = "red",
               fill = "black") +
   geom_smooth(method = "lm", se=FALSE, color="blue",lwd=1, aes(group=1))+
  ylab("Price") +
  xlab("Neighborhood") +
  ggtitle("Box Plot of each neighborhood against sales price") +
  theme_classic()+
  theme(plot.title = element_text(size= 18, face= "bold", hjust= 0.5),
        axis.title.x = element_text(size= 15, hjust= 0.5),
        axis.title.y = element_text(size= 15, hjust= 0.5),
        axis.ticks = element_blank())
```

```
## `geom_smooth()` using formula 'y ~ x'
```




```
#Display the distribution of Neighborhood
ggplot(df, aes(x=Neighborhood, fill=..count..))+
  geom_bar()+
  scale_fill_gradient(low="blue", high="green")+
```

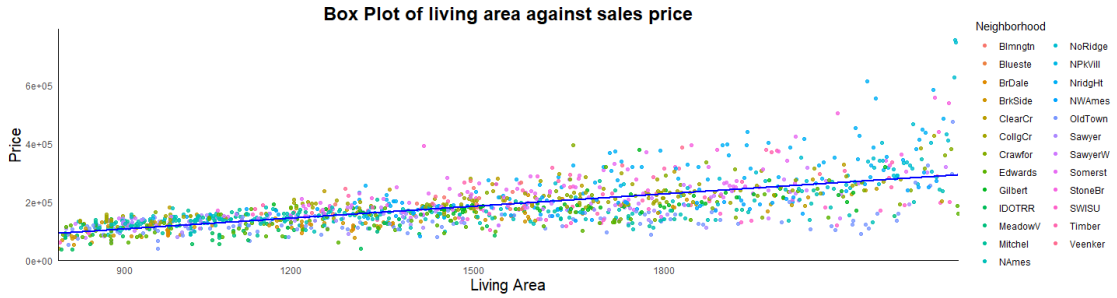
```
  ylab("Count") +
  xlab("OverallQual") +
  ggtitle("Distribution of OverallQual") +
  theme_classic()+
  theme(plot.title = element_text(size= 12, face= "bold", hjust= 0.5),
        axis.title.x = element_text(size= 10, hjust= 0.5),
        axis.title.y = element_text(size= 10, hjust= 0.5),
        axis.ticks = element_blank())
```



The second question

Second, identify the distribution of GrLivArea against SalePrice. The reason why the square feet of the living areas is related to sale price is obvious. In general, the more space for the living area, the more materials are needed which results the higher price. The trend is positive. Also, the neighborhood variables are introduced in this graph. We can see from the graph that nearly all the price ranges are evenly distributed to all neighborhood. However, for NridgHt, this neiborhood is generally higher than the tendency line which match the previous analysis. The distribution of living area against sale price is that the smaller living areas the property has, the smaller price range it tends to get. On the other hand, the larger living area the property has, the larger price range it tends to have. For example, the small living areas less than 1300 are more likely to have a certain price.

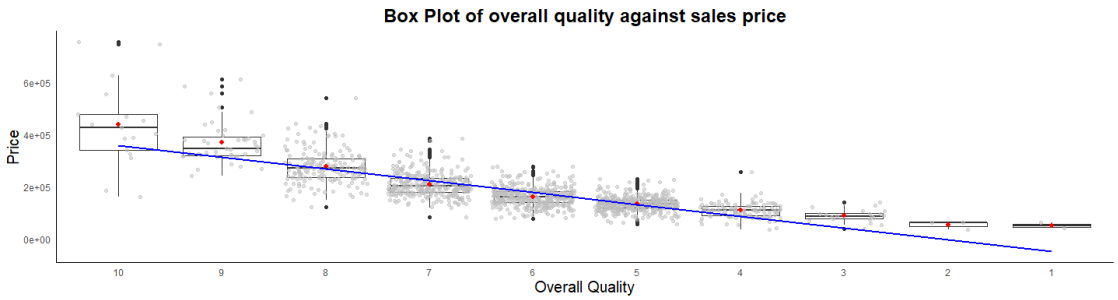
```
#Display boxplot of LotArea against sales price with a linear trend
ggplot(df,aes(x= reorder( GrLivArea, GrLivArea), y=SalePrice ))+
  geom_point(aes(color = Neighborhood, alpha = I(.8)))+
    geom_smooth(method = "lm", se=FALSE, color="blue",lwd=1, aes(group=1))+
 # scale_color_gradient( low = "navy", high="red")+
  ylab("Price") +
  xlab("Living Area") +
  scale_x_discrete(breaks=seq(0,6000,300))+
  ggtitle("Box Plot of living area against sales price") +
  theme_classic()+
  theme(plot.title = element_text(size= 18, face= "bold", hjust= 0.5),
        axis.title.x = element_text(size= 15, hjust= 0.5),
        axis.title.y = element_text(size= 15, hjust= 0.5),
        axis.ticks = element_blank())

## `geom_smooth()` using formula 'y ~ x'
```



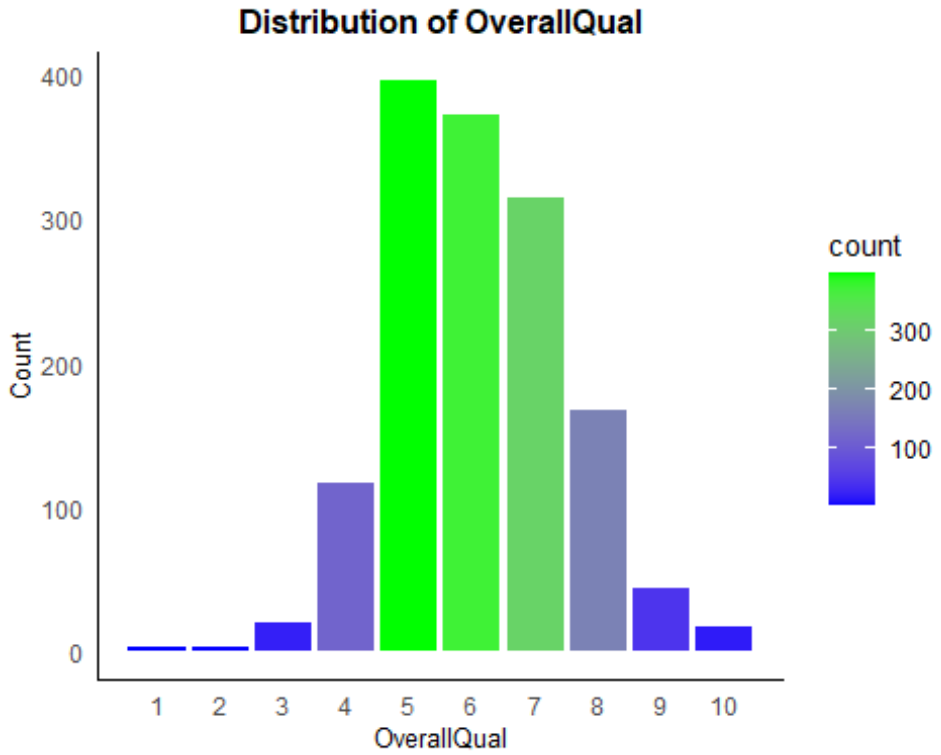
The third question

Third, Identify whether there is a corralation between OverallQual and SalePrice? It is obvious that the overall quality of material and finish of the house is significantly correlated with sale price. The hypothesis was made: the better materials are used, the higher price will be. According to the graph, the order of the variables is sorted by the median of SalePrice of each OverallQual group. As we can see, the order is in an ascending order which indicates that the higher overall quality the house has, the higher price it tends to be. Also, the number of residuals is shown by the grey points in each group. The graph matches my hypothesis. Additional insights can be extracted by the graph as well. The most of the residuals are under OverallQual 5, 6, 7. The distribution of OverallQual is roughly a symmetric distribution shown y the second graph.

```
#Display boxplot of overall quality against sales price with a linear trend
ggplot(df, aes(x= reorder(OverallQual, -SalePrice,Fun= median),
y=SalePrice))+
  geom_boxplot()+
  geom_jitter(aes(alpha = I(.5)), color="grey")+
  stat_summary(fun = mean,
               geom = "point",
               shape =20,
               size =3,
               color = "red",
               fill = "black") +
  geom_smooth(method = "lm", se=FALSE, color="blue",lwd=1, aes(group=1))+
  ylab("Price") +
  xlab("Overall Quality") +
  ggtitle("Box Plot of overall quality against sales price") +
  theme_classic()+
  theme(plot.title = element_text(size= 18, face= "bold", hjust= 0.5),
        axis.title.x = element_text(size= 15, hjust= 0.5),
        axis.title.y = element_text(size= 15, hjust= 0.5),
        axis.ticks = element_blank())

## `geom_smooth()` using formula 'y ~ x'
```




```r
#Display the distribution of OverallQual
ggplot(df, aes(x=OverallQual, fill=..count..))+
  geom_bar()+
  scale_fill_gradient(low="blue", high="green")+
  scale_x_continuous(breaks= seq(1,10,1))+
  ylab("Count") +
  xlab("OverallQual") +
  ggtitle("Distribution of OverallQual") +
  theme_classic()+
  theme(plot.title = element_text(size= 12, face= "bold", hjust= 0.5),
        axis.title.x = element_text(size= 10, hjust= 0.5),
        axis.title.y = element_text(size= 10, hjust= 0.5),
        axis.ticks = element_blank())
```

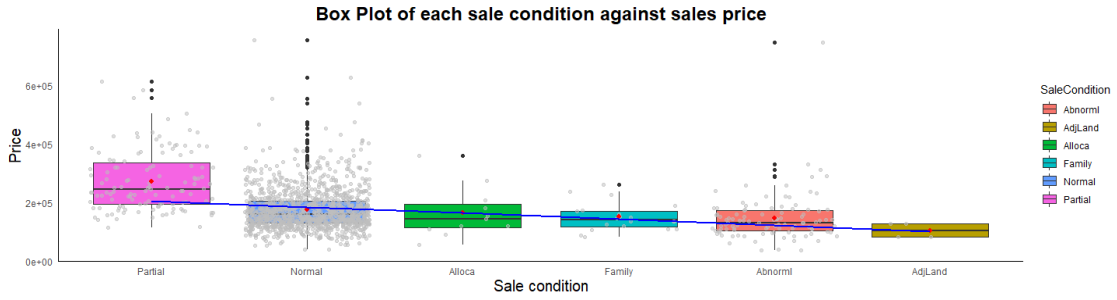


The fourth question

Fourth, Identify whether there is a corralation between SaleCondition and SalePrice? There are many conditions of sale in housing market. Abnormal sale is usually about family transactions and ignorance of true values and risks (Evans, 2021). Partial is about home is not completed when last assessed. Allocation is about two linked properties with separate deeds, typically condo with a garage unit. Also there is a type of condition of sale is adjoining Land Purchase. Different conditions of sale imply different expectation of the purchas, which indicates the different amount of money. So, investigating the relation between SaleCondition and SalePrice is valuable. As the graph shown, the order of SaleCondition is based on the ranking of the median of SalePrice in an ascending order. It shows what type of condition of sale is more likely to have a higher price. For example, the sale condition is partial is more likely to be higher than the sale condition of normal. However, according to the number of residuals in each group, the most popular condition of sale is normal. the least popular condition of sale is AdjLand.

```
#Display boxplot of each sale condition against sales price with a linear
trend
ggplot(df, aes(x= reorder(SaleCondition, -SalePrice,FUN = median),
y=SalePrice))+
  geom_boxplot(aes(fill= SaleCondition))+
  geom_jitter(aes(alpha = I(.5)), color="grey")+
  stat_summary(fun = mean,
               geom = "point",
               shape =20,
               size =3,
               color = "red",
               fill = "black") +
  geom_smooth(method = "lm", se=FALSE, color="blue",lwd=1, aes(group=1))+
  ylab("Price") +
  xlab("Sale condition") +
  ggtitle("Box Plot of each sale condition against sales price") +
  theme_classic()+
  theme(plot.title = element_text(size= 18, face= "bold", hjust= 0.5),
        axis.title.x = element_text(size= 15, hjust= 0.5),
        axis.title.y = element_text(size= 15, hjust= 0.5),
        axis.ticks = element_blank())
```

```
## `geom_smooth()` using formula 'y ~ x'
```

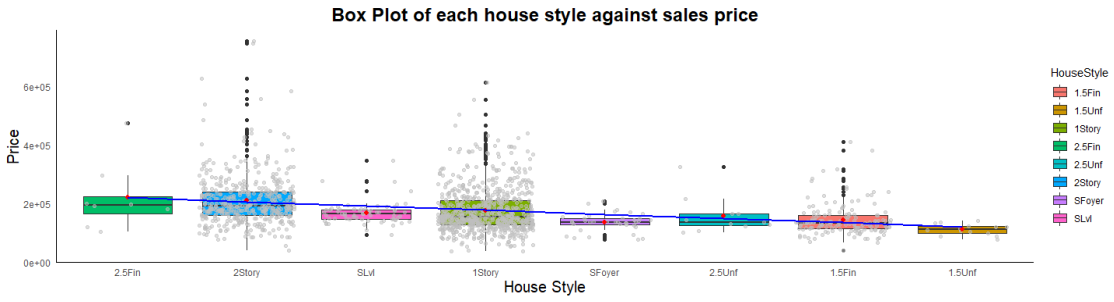


The fifth question

Fifth, Identify whether there is a corralation between HouseStyle and SalePrice? The HouseStyle is the style of dwelling which contains one story, one and one-half story: 2nd level finished, one and one-half story: 2nd level unfinished, two story, two and one-half story: 2nd level finished, two and one-half story: 2nd level unfinished, split Foyer, and split level. Those differnt dwelling style indicates the information of living areas and possible future expense required. So, it is valuable to investigate the relation between HouseStyle and SalePrice. As the graph shown, the order of HouseStyle is based on the ranking of the median of SalePrice in an ascending order. It shows what type of the style of dwelling is more likely to have a higher price. In Ames city, the most popular dwelling styles are 2 story and 1 story.

```
#Display boxplot of each HouseStyle against sales price with a linear trend
ggplot(df, aes(x= reorder(HouseStyle, -SalePrice,FUN = median),
y=SalePrice))+
  geom_boxplot(aes(fill= HouseStyle))+
  geom_jitter(aes(alpha = I(.5)), color="grey")+
  stat_summary(fun = mean,
               geom = "point",
               shape =20,
               size =3,
               color = "red",
               fill = "black") +
  geom_smooth(method = "lm", se=FALSE, color="blue",lwd=1, aes(group=1))+
  ylab("Price") +
  xlab("House Style") +
  ggtitle("Box Plot of each house style against sales price") +
  theme_classic()+
  theme(plot.title = element_text(size= 18, face= "bold", hjust= 0.5),
        axis.title.x = element_text(size= 15, hjust= 0.5),
        axis.title.y = element_text(size= 15, hjust= 0.5),
        axis.ticks = element_blank())
```

```
## `geom_smooth()` using formula 'y ~ x'
```



Further processing

This section mainly focuses on further data processing to train the model. To summarize the previous section, 5 variables (Neighborhood, LotArea, OverallQual, SaleCondition and HouseStyle) have been demostrated that they are highly correlated with sales price. However, Neighborhood, SaleCondition and HouseStyle are categorial variables that cannot be processed into linear regression model. So those categorical variable should be quantitated to numeric variable. The method for converting categorical variables to quantitative variables is to use zero to represents the category which has the lowest SalePrice. This method will be applied to all categorical variables (Neighborhood, SaleCondition and HouseStyle).

Convert Neighborhood, SaleCondition and HouseStyle to dummy variables

First, I sort the names of the neiborhoods (25 in total) based on the Ascending order. For example, NridgHt got the highest median sales price, so the numeric value for NridgHt will be 24. The lowest sale price is from MeadowV, so the value for it will be 0.

Second, do the same thing for SaleCondition and HouseStyle.

Third, display the same graph from the previous section to see whether the x-axis is in right order.

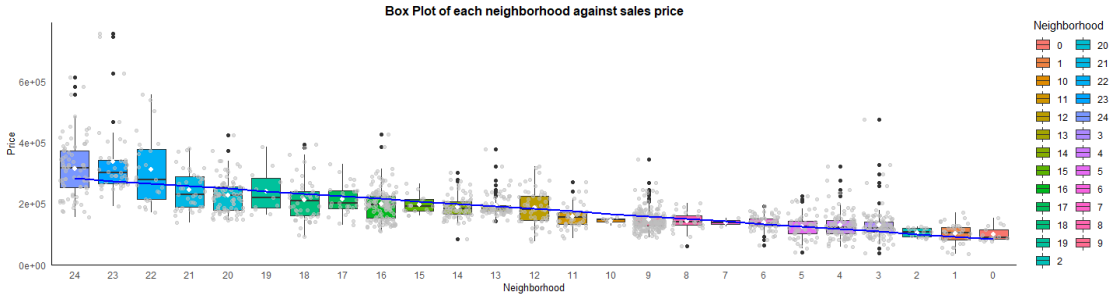
```
#Store the order of the categorical values based on the ranking of the median
of sale price into differnt group
Nei_name=
c("NridgHt","NoRidge","StoneBr","Timber","Somerst","Veenker","Crawfor","Clear
Cr","CollgCr","Blmngtn","NWAmes","Gilbert","SawyerW","Mitchel","NPkVill","NAm
es","SWISU","Blueste","Sawyer","BrkSide","Edwards","OldTown",
"BrDale","IDOTRR","MeadowV")

Sale_Cond_name= c("Partial","Normal", "Alloca", "Family", "Abnorml",
"AdjLand")

HouseS_name= c("2.5Fin", "2Story", "SLvl", "1Story", "SFoyer", "2.5Unf",
"1.5Fin", "1.5Unf")

  #Create a copy of df called FP (Further processing)
  FP= df

  #Using the for loop to convert categorical variables to numeric variables
  for (j in seq(1:length(Nei_name))) {
  #since the name is in ascending order, we need to assign the largest number
to the first object.
  FP$Neighborhood[FP$Neighborhood==Nei_name[j]]= length(Nei_name)-j
  }
  for (j in seq(1:length(Sale_Cond_name))) {
```

```
  #since the name is in ascending order, we need to assign the largest number
to the first object.
  FP$SaleCondition[FP$SaleCondition==Sale_Cond_name[j]]=
length(Sale_Cond_name)-j
  }

  for (j in seq(1:length(HouseS_name))) {
  #since the name is in ascending order, we need to assign the largest number
to the first object.
  FP$HouseStyle[FP$HouseStyle==HouseS_name[j]]= length(HouseS_name)-j


  }
```

After converting, display the boxplot for neiborhood against saleprice using the same code as the previous section. The x-axis should be from 24 to 0. As the graph shown below, the convertion is completed for neighborhood.

```
#Display boxplot of each neighborhood against sales price with a linear trend
ggplot(FP, aes(x= reorder(Neighborhood, -SalePrice, FUN = median),
y=SalePrice))+
  geom_boxplot(aes(fill=Neighborhood))+
  geom_jitter(aes(alpha = I(.5)), color="grey")+
  stat_summary(fun = mean,
               geom = "point",
               shape =20,
               size =3,
               color = "white",
               fill = "black") +
   geom_smooth(method = "lm", se=FALSE, color="blue",lwd=1, aes(group=1))+
  ylab("Price") +
  xlab("Neighborhood") +
  ggtitle("Box Plot of each neighborhood against sales price") +
  theme_classic()+
  theme(plot.title = element_text(size= 12, face= "bold", hjust= 0.5),
        axis.title.x = element_text(size= 10, hjust= 0.5),
        axis.title.y = element_text(size= 10, hjust= 0.5),
        axis.ticks = element_blank())
```

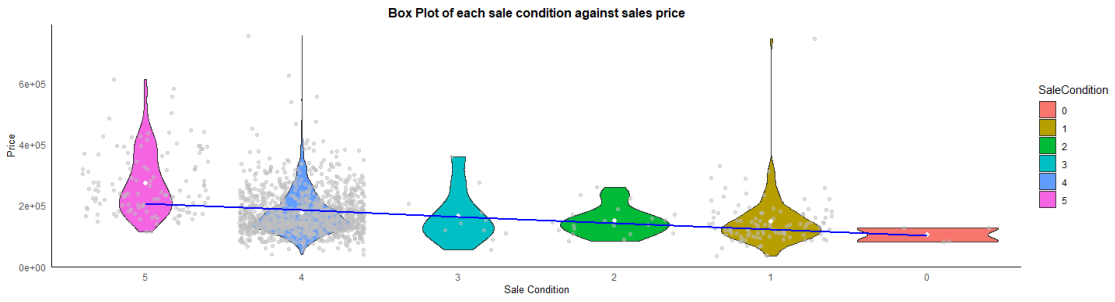
```
## `geom_smooth()` using formula 'y ~ x'
```



After converting, display the boxplot for SaleCondition against SalePrice using the same code as the previous section. The x-axis should be from 5 to 0. As the graph shown below, the convertion is completed for sale condition.

```
#Display boxplot of each sale condition against sales price with a linear
trend
ggplot(FP, aes(x= reorder(SaleCondition, -SalePrice,FUN = median),
y=SalePrice))+
  geom_violin(aes(fill= SaleCondition))+
  geom_jitter(aes(alpha = I(.5)), color="grey")+
  stat_summary(fun = mean,
               geom = "point",
               shape =20,
               size =3,
               color = "white",
               fill = "black") +
  geom_smooth(method = "lm", se=FALSE, color="blue",lwd=1, aes(group=1))+
  ylab("Price") +
  xlab("Sale Condition") +
  ggtitle("Box Plot of each sale condition against sales price") +
  theme_classic()+
  theme(plot.title = element_text(size= 12, face= "bold", hjust= 0.5),
        axis.title.x = element_text(size= 10, hjust= 0.5),
        axis.title.y = element_text(size= 10, hjust= 0.5),
        axis.ticks = element_blank())
```

```
## `geom_smooth()` using formula 'y ~ x'
```



After converting, display the boxplot for HouseStyle against SalePrice using the same code as the previous section. The x-axis should be from 7 to 0. As the graph shown below, the convertion is completed for house style.

```
#Display boxplot of each HouseStyle against sales price with a linear trend
ggplot(FP, aes(x= reorder(HouseStyle, -SalePrice,FUN = median),
y=SalePrice))+
  geom_boxplot(aes(fill= HouseStyle))+
  geom_jitter(aes(alpha = I(.5)), color="grey")+
  stat_summary(fun = mean,
               geom = "point",
               shape =20,
               size =3,
```

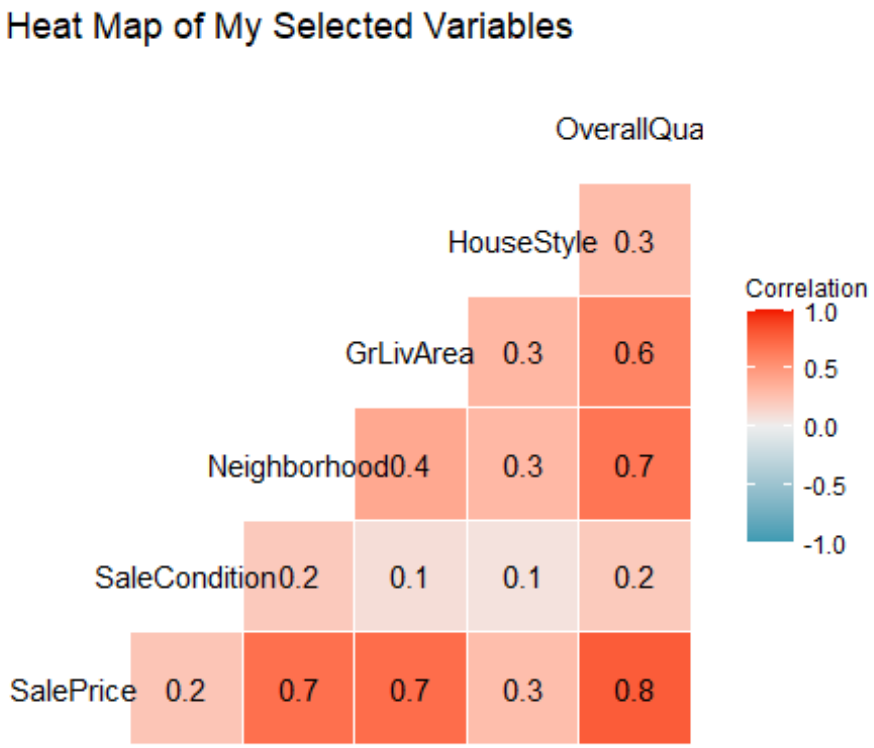
```
                color = "white",
                fill = "black") +
  geom_smooth(method = "lm", se=FALSE, color="blue",lwd=1, aes(group=1))+
  ylab("Price") +
  xlab("House Style") +
  ggtitle("Box Plot of each House style against sales price") +
  theme_classic()+
  theme(plot.title = element_text(size= 12, face= "bold", hjust= 0.5),
        axis.title.x = element_text(size= 10, hjust= 0.5),
        axis.title.y = element_text(size= 10, hjust= 0.5),
        axis.ticks = element_blank())
```

```
## `geom_smooth()` using formula 'y ~ x'
```

In conclusion, the heat map of my selected variables is shown.

As the graph shown, in my 5 selected variable, 3 of them have been proved that they have high correlation with SalePrice, but 2 of them which are SaleCondition and HouseStyle are not significantly correlated to SalePrice.

```
Heat_map= select(FP, c("SalePrice","SaleCondition",
"Neighborhood","GrLivArea","HouseStyle","OverallQual"))
#Convert the categorical data to numeric format so that I can create a heat
maps for my selected variables
Heat_map$Neighborhood= as.numeric(Heat_map$Neighborhood)
Heat_map$SaleCondition= as.numeric(Heat_map$SaleCondition)
Heat_map$HouseStyle= as.numeric(Heat_map$HouseStyle)
#Create Heat map
ggcorr(Heat_map, name="Correlation", label = T)+
  ggplot2::labs(title = "Heat Map of My Selected Variables")
```



After converting those categorical variables, we can start modelling

Modeling

In this section, I am going to apply multi-linear model to There are 3 models will be developed to be compared. Here are the variable selections: 1. The best 5 numeric variables correlated to SalePrice which are OverallQual, GrLivArea, GarageCars, GarageArea and TotalBsmtSF. 2. The 5 variables I defined in the previous sections which are SaleCondition, Neighborhood, GrLivArea, HouseStyle and OverallQual. 3. Combine all the variables from model 1 and model 2.

```
#First model: train linear regression model by the best numeric model
First_M =lm(SalePrice ~
OverallQual+GrLivArea+GarageCars+GarageArea+TotalBsmtSF  , data= FP)
#Second model: train linear regression model by Neighborhood, HouseStyle,
LotArea, OverallQual, SaleCondition
Second_M = lm(SalePrice ~ SaleCondition+
Neighborhood+GrLivArea+HouseStyle+OverallQual, data= FP)
#Third model: train linear regression model with all variables of model 1 and
2
Third_M= lm(SalePrice ~SaleCondition+
Neighborhood+GrLivArea+HouseStyle+OverallQual+GarageCars+GarageArea+TotalBsmt
SF, data= FP)
```

```r
#Display each summary of the model
summary(First_M) #Adjusted R-squared 0.76
```

```
##
## Call:
## lm(formula = SalePrice ~ OverallQual + GrLivArea + GarageCars +
##     GarageArea + TotalBsmtSF, data = FP)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -479921  -19935   -1596   16553  286989
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -98468.964   4647.389 -21.188  < 2e-16 ***
## OverallQual  23461.117   1076.159  21.801  < 2e-16 ***
## GrLivArea       45.598      2.495  18.276  < 2e-16 ***
## GarageCars   14564.711   3023.924   4.816 1.61e-06 ***
## GarageArea      16.693     10.472   1.594    0.111
## TotalBsmtSF     31.705      2.912  10.886  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 38850 on 1445 degrees of freedom
## Multiple R-squared:  0.7609, Adjusted R-squared:  0.7601
## F-statistic: 919.7 on 5 and 1445 DF,  p-value: < 2.2e-16
```

```r
summary(Second_M)#Adjusted R-squared 0.8051
```

```
##
## Call:
## lm(formula = SalePrice ~ SaleCondition + Neighborhood + GrLivArea +
##     HouseStyle + OverallQual, data = FP)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -372629  -16363     277   14834  254218
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -73310.674  22768.309  -3.220 0.001312 **
## SaleCondition1  -5817.571  18278.091  -0.318 0.750319
## SaleCondition2  -9571.536  19643.792  -0.487 0.626153
## SaleCondition3  -3333.862  20878.031  -0.160 0.873154
## SaleCondition4   6229.963  17961.402   0.347 0.728754
## SaleCondition5  27619.104  18339.210   1.506 0.132287
## Neighborhood1   -1445.978  10662.655  -0.136 0.892148
## Neighborhood10   7131.821  14700.320   0.485 0.627647
## Neighborhood11  13487.092  10134.619   1.331 0.183472
```

```
## Neighborhood12  16900.388  10078.760   1.677 0.093797 .
## Neighborhood13  20996.892   9844.009   2.133 0.033100 *
## Neighborhood14  10085.930   9888.417   1.020 0.307915
## Neighborhood15   1694.087  12662.764   0.134 0.893592
## Neighborhood16  24778.087   9487.851   2.612 0.009109 **
## Neighborhood17  38492.293  11164.833   3.448 0.000582 ***
## Neighborhood18  34675.817  10290.046   3.370 0.000772 ***
## Neighborhood19  55949.233  13991.014   3.999 6.69e-05 ***
## Neighborhood2   -3319.279  12580.929  -0.264 0.791946
## Neighborhood20  32103.771  10070.005   3.188 0.001464 **
## Neighborhood21  38281.401  10907.098   3.510 0.000463 ***
## Neighborhood22  73901.649  11900.667   6.210 6.96e-10 ***
## Neighborhood23  81175.623  10993.210   7.384 2.61e-13 ***
## Neighborhood24  73092.695  10461.614   6.987 4.32e-12 ***
## Neighborhood3   -7102.856   9498.724  -0.748 0.454724
## Neighborhood4   -2923.502   9517.100  -0.307 0.758749
## Neighborhood5   11023.138  10110.767   1.090 0.275795
## Neighborhood6   10291.661   9672.270   1.064 0.287493
## Neighborhood7    2698.700  26334.822   0.102 0.918393
## Neighborhood8   -9754.077  11585.114  -0.842 0.399959
## Neighborhood9    8161.124   9169.721   0.890 0.373613
## GrLivArea          71.310      2.902  24.573  < 2e-16 ***
## HouseStyle1     -8818.030  10074.552  -0.875 0.381572
## HouseStyle2    -35359.072  14566.973  -2.427 0.015334 *
## HouseStyle3     24898.517  11463.125   2.172 0.030018 *
## HouseStyle4     15195.319   9799.963   1.551 0.121234
## HouseStyle5      9743.307  10741.988   0.907 0.364546
## HouseStyle6    -11911.998  10085.326  -1.181 0.237753
## HouseStyle7    -31443.052  16907.281  -1.860 0.063131 .
## OverallQual     19082.449   1128.679  16.907  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 35020 on 1412 degrees of freedom
## Multiple R-squared:  0.8102, Adjusted R-squared:  0.8051
## F-statistic: 158.6 on 38 and 1412 DF,  p-value: < 2.2e-16
```

```
summary(Third_M) #Adjusted R-squared 0.8126
```

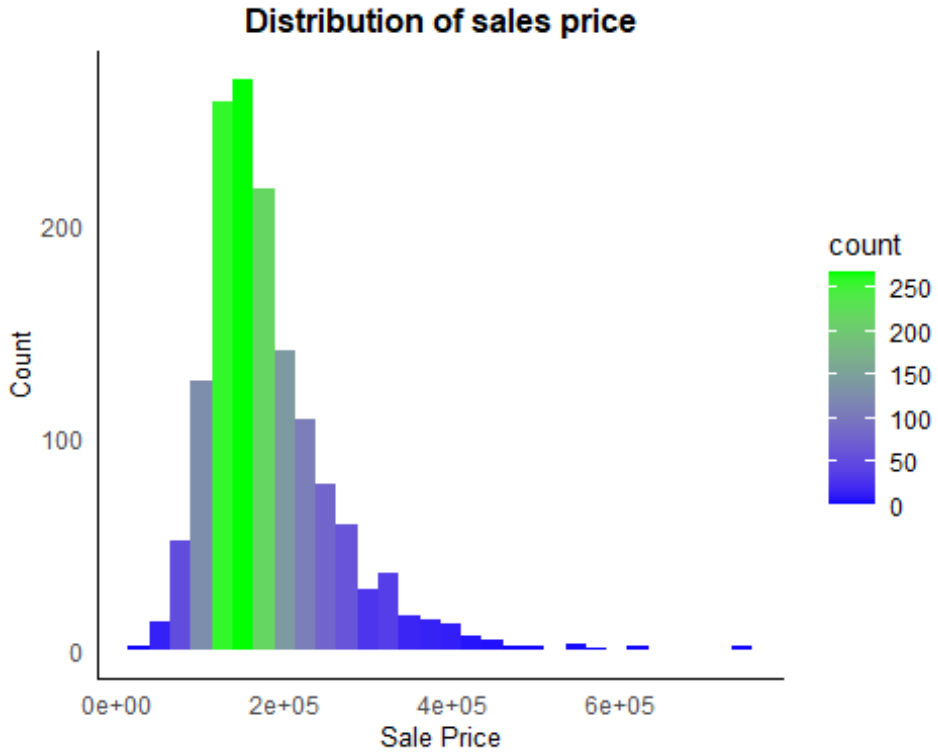
```
##
## Call:
## lm(formula = SalePrice ~ SaleCondition + Neighborhood + GrLivArea +
##     HouseStyle + OverallQual + GarageCars + GarageArea + TotalBsmtSF,
##     data = FP)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -392223  -15481     264   14313  262547
##
## Coefficients:
```

```
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -58984.471  22411.442  -2.632 0.008584 **
## SaleCondition1 -16739.731  17977.011  -0.931 0.351923
## SaleCondition2 -22603.903  19333.949  -1.169 0.242549
## SaleCondition3 -13005.479  20584.034  -0.632 0.527605
## SaleCondition4  -5670.141  17679.980  -0.321 0.748478
## SaleCondition5  13274.227  18082.084   0.734 0.463004
## Neighborhood1   -3466.202  10467.008  -0.331 0.740577
## Neighborhood10  -4285.449  14525.278  -0.295 0.768011
## Neighborhood11   5729.621  10003.671   0.573 0.566904
## Neighborhood12  10737.442   9949.725   1.079 0.280697
## Neighborhood13  13545.479   9797.820   1.382 0.167038
## Neighborhood14   3023.670   9771.248   0.309 0.757028
## Neighborhood15  -7576.762  12599.367  -0.601 0.547697
## Neighborhood16  16617.542   9382.379   1.771 0.076753 .
## Neighborhood17  32669.336  10982.667   2.975 0.002983 **
## Neighborhood18  32785.187  10127.597   3.237 0.001235 **
## Neighborhood19  47379.761  13780.093   3.438 0.000602 ***
## Neighborhood2   -6838.579  12343.723  -0.554 0.579658
## Neighborhood20  22616.209   9984.527   2.265 0.023656 *
## Neighborhood21  28178.442  10807.427   2.607 0.009222 **
## Neighborhood22  66085.622  11742.346   5.628 2.20e-08 ***
## Neighborhood23  71479.334  10883.660   6.568 7.17e-11 ***
## Neighborhood24  61469.583  10388.404   5.917 4.11e-09 ***
## Neighborhood3  -10430.735   9347.882  -1.116 0.264681
## Neighborhood4   -4027.309   9342.027  -0.431 0.666465
## Neighborhood5    8849.155   9931.948   0.891 0.373093
## Neighborhood6    5910.650   9516.413   0.621 0.534634
## Neighborhood7   -7666.005  25874.423  -0.296 0.767062
## Neighborhood8   -7875.924  11365.416  -0.693 0.488441
## Neighborhood9    3746.415   9024.409   0.415 0.678101
## GrLivArea          59.400      3.520  16.876  < 2e-16 ***
## HouseStyle1     -3426.780   9954.933  -0.344 0.730725
## HouseStyle2    -25175.826  14422.078  -1.746 0.081091 .
## HouseStyle3     20958.137  11270.244   1.860 0.063151 .
## HouseStyle4     12322.492   9624.516   1.280 0.200642
## HouseStyle5     11002.546  10592.656   1.039 0.299124
## HouseStyle6     -4779.147  10082.612  -0.474 0.635574
## HouseStyle7    -14928.269  16946.424  -0.881 0.378517
## OverallQual     16805.788   1155.113  14.549  < 2e-16 ***
## GarageCars      10858.565   2876.570   3.775 0.000167 ***
## GarageArea          2.532      9.840   0.257 0.796983
## TotalBsmtSF        12.805      3.542   3.615 0.000311 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 34330 on 1409 degrees of freedom
## Multiple R-squared:  0.8179, Adjusted R-squared:  0.8126
## F-statistic: 154.4 on 41 and 1409 DF,  p-value: < 2.2e-16
```

As we can see from the result, model 3 is the best model with the R squared value 0.8126. **After getting the best model, I am going to optimize the model by using log transformation.** First, display the distribution of SalePrice by histogram. Second, applying log scale transformation to SalePrice. Third, display the distribution of log(SalePrice) by histogram. As we can see from the graph, the distribution of log(SalePrice) is much closer to normal distribution. So, using log transformation can actually improve the performance of the model.

```
#Display histagram of sales price
ggplot(FP, aes(x= SalePrice, fill=..count..))+
  geom_histogram()+
  scale_fill_gradient(low="blue", high="green")+
  ylab("Count") +
  xlab("Sale Price") +
  ggtitle("Distribution of sales price") +
  theme_classic()+
  theme(plot.title = element_text(size= 12, face= "bold", hjust= 0.5),
        axis.title.x = element_text(size= 10, hjust= 0.5),
        axis.title.y = element_text(size= 10, hjust= 0.5),
        axis.ticks = element_blank())
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
#Display histagram of sales price after log transformation
ggplot(FP, aes(x= log(SalePrice), fill=..count..))+
  geom_histogram()+
  scale_fill_gradient(low="blue", high="green")+
```

```
  ylab("Count") +
  xlab("Sale Price after log transformation") +
  ggtitle("Distribution of sales price after log transformation") +
  theme_classic()+
  theme(plot.title = element_text(size= 12, face= "bold", hjust= 0.5),
        axis.title.x = element_text(size= 10, hjust= 0.5),
        axis.title.y = element_text(size= 10, hjust= 0.5),
        axis.ticks = element_blank())
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

Appley log transformation to all models As we can see from the result, the R squared value is improved.

```
#Scale SalePrice with log()
FP$LogSalePrice= log(FP$SalePrice)
#Perform log tranformation to all model
#First model: train linear regression model by the best numeric model
First_M =lm(LogSalePrice ~
OverallQual+GrLivArea+GarageCars+GarageArea+TotalBsmtSF   , data= FP)
#Second model: train linear regression model by Neighborhood, HouseStyle,
LotArea, OverallQual, SaleCondition
Second_M = lm(LogSalePrice ~ SaleCondition+
Neighborhood+GrLivArea+HouseStyle+OverallQual, data= FP)
#Third model: train linear regression model with all variables of model 1 and
2
Third_M= lm(LogSalePrice ~SaleCondition+
Neighborhood+GrLivArea+HouseStyle+OverallQual+GarageCars+GarageArea+TotalBsmt
```

```
SF, data= FP)

summary(First_M) #Adjusted R-squared 0.7931 where the original one is 0.76

##
## Call:
## lm(formula = LogSalePrice ~ OverallQual + GrLivArea + GarageCars +
##     GarageArea + TotalBsmtSF, data = FP)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.14740 -0.07361  0.01423  0.10550  0.59648
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1.058e+01  2.174e-02 486.646  < 2e-16 ***
## OverallQual 1.277e-01  5.034e-03  25.370  < 2e-16 ***
## GrLivArea   1.988e-04  1.167e-05  17.035  < 2e-16 ***
## GarageCars  1.084e-01  1.415e-02   7.666 3.24e-14 ***
## GarageArea  4.540e-05  4.899e-05   0.927    0.354
## TotalBsmtSF 1.425e-04  1.362e-05  10.457  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1817 on 1445 degrees of freedom
## Multiple R-squared:  0.7938, Adjusted R-squared:  0.7931
## F-statistic:  1112 on 5 and 1445 DF,  p-value: < 2.2e-16

summary(Second_M)#Adjusted R-squared 0.8309 where the original one is 0.8051

##
## Call:
## lm(formula = LogSalePrice ~ SaleCondition + Neighborhood + GrLivArea +
##     HouseStyle + OverallQual, data = FP)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.64092 -0.07745  0.01034  0.09203  0.56031
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     1.045e+01  1.068e-01  97.809  < 2e-16 ***
## SaleCondition1  6.143e-03  8.576e-02   0.072 0.942904
## SaleCondition2  3.455e-02  9.217e-02   0.375 0.707864
## SaleCondition3  3.853e-02  9.796e-02   0.393 0.694183
## SaleCondition4  1.203e-01  8.428e-02   1.427 0.153758
## SaleCondition5  1.765e-01  8.605e-02   2.052 0.040384 *
## Neighborhood1  -3.824e-02  5.003e-02  -0.764 0.444813
## Neighborhood10  1.959e-01  6.898e-02   2.840 0.004578 **
## Neighborhood11  2.448e-01  4.755e-02   5.149 2.99e-07 ***
```

```
## Neighborhood12  2.616e-01  4.729e-02   5.532 3.77e-08 ***
## Neighborhood13  3.011e-01  4.619e-02   6.520 9.77e-11 ***
## Neighborhood14  2.570e-01  4.640e-02   5.539 3.62e-08 ***
## Neighborhood15  2.284e-01  5.942e-02   3.845 0.000126 ***
## Neighborhood16  3.115e-01  4.452e-02   6.997 4.03e-12 ***
## Neighborhood17  3.976e-01  5.239e-02   7.590 5.77e-14 ***
## Neighborhood18  3.433e-01  4.828e-02   7.111 1.82e-12 ***
## Neighborhood19  4.422e-01  6.565e-02   6.736 2.37e-11 ***
## Neighborhood2   4.609e-03  5.903e-02   0.078 0.937782
## Neighborhood20  3.428e-01  4.725e-02   7.254 6.64e-13 ***
## Neighborhood21  3.645e-01  5.118e-02   7.123 1.68e-12 ***
## Neighborhood22  4.223e-01  5.584e-02   7.562 7.11e-14 ***
## Neighborhood23  4.186e-01  5.158e-02   8.115 1.05e-15 ***
## Neighborhood24  4.275e-01  4.909e-02   8.709  < 2e-16 ***
## Neighborhood3   4.348e-02  4.457e-02   0.976 0.329446
## Neighborhood4   8.935e-02  4.466e-02   2.001 0.045607 *
## Neighborhood5   1.262e-01  4.744e-02   2.660 0.007908 **
## Neighborhood6   2.105e-01  4.538e-02   4.637 3.86e-06 ***
## Neighborhood7   1.426e-01  1.236e-01   1.154 0.248756
## Neighborhood8   9.249e-02  5.436e-02   1.701 0.089072 .
## Neighborhood9   2.099e-01  4.303e-02   4.878 1.20e-06 ***
## GrLivArea       3.173e-04  1.362e-05  23.301  < 2e-16 ***
## HouseStyle1     2.840e-02  4.727e-02   0.601 0.548084
## HouseStyle2    -6.649e-02  6.835e-02  -0.973 0.330823
## HouseStyle3     1.624e-01  5.379e-02   3.019 0.002578 **
## HouseStyle4     1.019e-01  4.598e-02   2.215 0.026907 *
## HouseStyle5     1.080e-01  5.040e-02   2.143 0.032266 *
## HouseStyle6     3.807e-03  4.732e-02   0.080 0.935892
## HouseStyle7    -6.624e-02  7.933e-02  -0.835 0.403852
## OverallQual     1.115e-01  5.296e-03  21.055  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1643 on 1412 degrees of freedom
## Multiple R-squared:  0.8353, Adjusted R-squared:  0.8309
## F-statistic: 188.4 on 38 and 1412 DF,  p-value: < 2.2e-16
```

```
summary(Third_M) #Adjusted R-squared 0.8453 where the original one is 0.8126
```

```
##
## Call:
## lm(formula = LogSalePrice ~ SaleCondition + Neighborhood + GrLivArea +
##     HouseStyle + OverallQual + GarageCars + GarageArea + TotalBsmtSF,
##     data = FP)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.80872 -0.07168  0.00901  0.09172  0.53763
##
## Coefficients:
```

```
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     1.055e+01  1.026e-01 102.838  < 2e-16 ***
## SaleCondition1 -6.914e-02  8.228e-02  -0.840  0.40085
## SaleCondition2 -5.523e-02  8.849e-02  -0.624  0.53259
## SaleCondition3 -2.320e-02  9.421e-02  -0.246  0.80553
## SaleCondition4  3.839e-02  8.092e-02   0.474  0.63523
## SaleCondition5  7.719e-02  8.276e-02   0.933  0.35111
## Neighborhood1  -5.267e-02  4.790e-02  -1.099  0.27179
## Neighborhood10  1.209e-01  6.648e-02   1.818  0.06921 .
## Neighborhood11  1.932e-01  4.578e-02   4.219 2.61e-05 ***
## Neighborhood12  2.216e-01  4.554e-02   4.866 1.27e-06 ***
## Neighborhood13  2.544e-01  4.484e-02   5.674 1.69e-08 ***
## Neighborhood14  2.108e-01  4.472e-02   4.713 2.69e-06 ***
## Neighborhood15  1.705e-01  5.766e-02   2.957  0.00315 **
## Neighborhood16  2.567e-01  4.294e-02   5.979 2.85e-09 ***
## Neighborhood17  3.592e-01  5.026e-02   7.147 1.42e-12 ***
## Neighborhood18  3.329e-01  4.635e-02   7.183 1.10e-12 ***
## Neighborhood19  3.855e-01  6.307e-02   6.113 1.26e-09 ***
## Neighborhood2  -1.944e-02  5.649e-02  -0.344  0.73088
## Neighborhood20  2.789e-01  4.570e-02   6.103 1.34e-09 ***
## Neighborhood21  2.976e-01  4.946e-02   6.017 2.26e-09 ***
## Neighborhood22  3.711e-01  5.374e-02   6.906 7.52e-12 ***
## Neighborhood23  3.544e-01  4.981e-02   7.115 1.78e-12 ***
## Neighborhood24  3.490e-01  4.755e-02   7.341 3.58e-13 ***
## Neighborhood3   2.227e-02  4.278e-02   0.521  0.60274
## Neighborhood4   8.295e-02  4.276e-02   1.940  0.05256 .
## Neighborhood5   1.127e-01  4.546e-02   2.480  0.01326 *
## Neighborhood6   1.819e-01  4.355e-02   4.176 3.15e-05 ***
## Neighborhood7   7.457e-02  1.184e-01   0.630  0.52897
## Neighborhood8   1.064e-01  5.202e-02   2.045  0.04109 *
## Neighborhood9   1.807e-01  4.130e-02   4.375 1.30e-05 ***
## GrLivArea       2.321e-04  1.611e-05  14.406  < 2e-16 ***
## HouseStyle1     6.665e-02  4.556e-02   1.463  0.14370
## HouseStyle2     6.648e-03  6.601e-02   0.101  0.91978
## HouseStyle3     1.343e-01  5.158e-02   2.603  0.00934 **
## HouseStyle4     8.092e-02  4.405e-02   1.837  0.06640 .
## HouseStyle5     1.167e-01  4.848e-02   2.406  0.01624 *
## HouseStyle6     5.552e-02  4.615e-02   1.203  0.22916
## HouseStyle7     5.100e-02  7.756e-02   0.658  0.51094
## OverallQual     9.573e-02  5.287e-03  18.107  < 2e-16 ***
## GarageCars      6.687e-02  1.317e-02   5.079 4.30e-07 ***
## GarageArea      4.213e-05  4.504e-05   0.936  0.34967
## TotalBsmtSF     9.291e-05  1.621e-05   5.731 1.22e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1571 on 1409 degrees of freedom
## Multiple R-squared:  0.8497, Adjusted R-squared:  0.8453
## F-statistic: 194.3 on 41 and 1409 DF,  p-value: < 2.2e-16
```

Evaluation

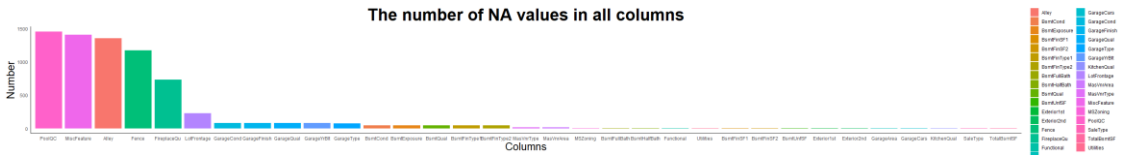
This section will maily focus on cleaning the test dataset and then evaluate the performance of my models. However, before using the models to predict, cleaning the test dataset and convert the targeting categorical variables to numeric variables are preferred, otherwise the model cannot predict the test dataset.

Here is the code for cleaning the NA values of test dataset. **First, import the test data.**

```
ts= read.csv(file.choose(), header=TRUE) #Import data to testing_set
```

Second, display the summary of NA values in test dataset As the graph shown, there are 33 variables contained NA values. So I am going to handle them one by one based on the description of the data.

```
# Create NA_sum dataframe to store the summary of NA values
NA_sum_ts= as.data.frame(colSums(is.na(ts)))
# Process the data to feed the ggplot code

colnames(NA_sum_ts) = "Number"
NA_sum_ts$Variables= row.names(NA_sum_ts)
NA_sum_ts = NA_sum_ts %>% filter(NA_sum_ts$Number !=0)

#Display the bar plot of the distribution of NA
ggplot(NA_sum_ts, aes(x= reorder(Variables, -Number), y=Number))+
  geom_bar(aes(fill= Variables), stat = "identity")+
  ylab("Number") +
  xlab("Columns") +
  ggtitle("The number of NA values in all columns") +
  theme_classic()+
  theme(plot.title = element_text(size= 32, face= "bold", hjust= 0.5),
        axis.title.x = element_text(size= 20, hjust= 0.5),
        axis.title.y = element_text(size= 20, hjust= 0.5),
        axis.ticks = element_blank())
```



Third, start cleaning the NA values Finally, display the summary of NA values by matrix

```
ts$LotFrontage[is.na(ts$LotFrontage)]=0 #Replace NA with 0
ts$Alley[is.na(ts$Alley)]="NO" #Replace NA with "NO"

ts=ts[!is.na(ts$MasVnrType), ]

ts$BsmtCond[is.na(ts$BsmtCond)]="No Basement" #Replace NA with "No Basement"
ts$BsmtQual[is.na(ts$BsmtQual)]="No Basement" #Replace NA with "No Basement"
```

```r
ts$BsmtExposure[is.na(ts$BsmtExposure)]="No Basement" #Replace NA with "No Basement"

ts$BsmtFinType1[is.na(ts$BsmtFinType1)]="No Basement" #Replace NA with "No Basement"

ts$BsmtFinType2[is.na(ts$BsmtFinType2)]="No Basement" #Replace NA with "No Basement"

ts$FireplaceQu[is.na(ts$FireplaceQu)]="No Fireplace" #Replace NA with "No Fireplace"

ts$GarageType[is.na(ts$GarageType)]="No Garage" #Replace NA with "No Garage"

ts$GarageYrBlt[is.na(ts$GarageYrBlt)]=0 #Replace NA with zero since this column is integer type

ts$GarageFinish[is.na(ts$GarageFinish)]="No Garage" #Replace NA with "No Garage"

ts$GarageQual[is.na(ts$GarageQual)]="No Garage" #Replace NA with "No Garage"

ts$GarageCond[is.na(ts$GarageCond)]="No Garage" #Replace NA with "No Garage"

ts$PoolQC[is.na(ts$PoolQC)]="No Pool" #Replace NA with "No Pool"

ts$Fence[is.na(ts$Fence)]="No Fence" #Replace NA with "No Fence"

ts$MiscFeature[is.na(ts$MiscFeature)]="None" #Replace NA with "No MiscFeature"

# The codes below are to remove all the rows with NA values since those values are missing
ts=ts[!is.na(ts$MasVnrType), ]

ts=ts[!is.na(ts$BsmtUnfSF), ]

ts=ts[!is.na(ts$BsmtHalfBath), ]

ts=ts[!is.na(ts$BsmtFullBath), ]

ts=ts[!is.na(ts$MSZoning), ]

ts=ts[!is.na(ts$KitchenQual), ]

ts=ts[!is.na(ts$Functional), ]
```

```
ts=ts[!is.na(ts$GarageCars), ]

ts=ts[!is.na(ts$SaleType), ]

ts=ts[!is.na(ts$Exterior1st), ]

ts=ts[!is.na(ts$Utilities), ]

#show the NA values
colSums(is.na(ts))
```

```
##            Id    MSSubClass      MSZoning   LotFrontage       LotArea
##             0             0             0             0             0
##        Street         Alley      LotShape   LandContour     Utilities
##             0             0             0             0             0
##     LotConfig     LandSlope  Neighborhood    Condition1    Condition2
##             0             0             0             0             0
##      BldgType    HouseStyle   OverallQual   OverallCond     YearBuilt
##             0             0             0             0             0
##  YearRemodAdd     RoofStyle      RoofMatl   Exterior1st   Exterior2nd
##             0             0             0             0             0
##    MasVnrType    MasVnrArea     ExterQual     ExterCond    Foundation
##             0             0             0             0             0
##      BsmtQual      BsmtCond  BsmtExposure  BsmtFinType1    BsmtFinSF1
##             0             0             0             0             0
##  BsmtFinType2    BsmtFinSF2     BsmtUnfSF   TotalBsmtSF       Heating
##             0             0             0             0             0
##     HeatingQC    CentralAir    Electrical     X1stFlrSF     X2ndFlrSF
##             0             0             0             0             0
##  LowQualFinSF     GrLivArea  BsmtFullBath  BsmtHalfBath      FullBath
##             0             0             0             0             0
##      HalfBath  BedroomAbvGr  KitchenAbvGr   KitchenQual  TotRmsAbvGrd
##             0             0             0             0             0
##    Functional    Fireplaces   FireplaceQu    GarageType   GarageYrBlt
##             0             0             0             0             0
##  GarageFinish    GarageCars    GarageArea    GarageQual    GarageCond
##             0             0             0             0             0
##    PavedDrive    WoodDeckSF   OpenPorchSF EnclosedPorch    X3SsnPorch
##             0             0             0             0             0
##   ScreenPorch      PoolArea        PoolQC         Fence   MiscFeature
##             0             0             0             0             0
##       MiscVal        MoSold        YrSold      SaleType SaleCondition
##             0             0             0             0             0
##     SalePrice
##             0
```

So, there are no NA values in test dataset.

Converting categorical variables to numeric variables

In this subsection, it mainly focuses on converting the targeted categorical variables to numeric variables same as the way where training data was processed.

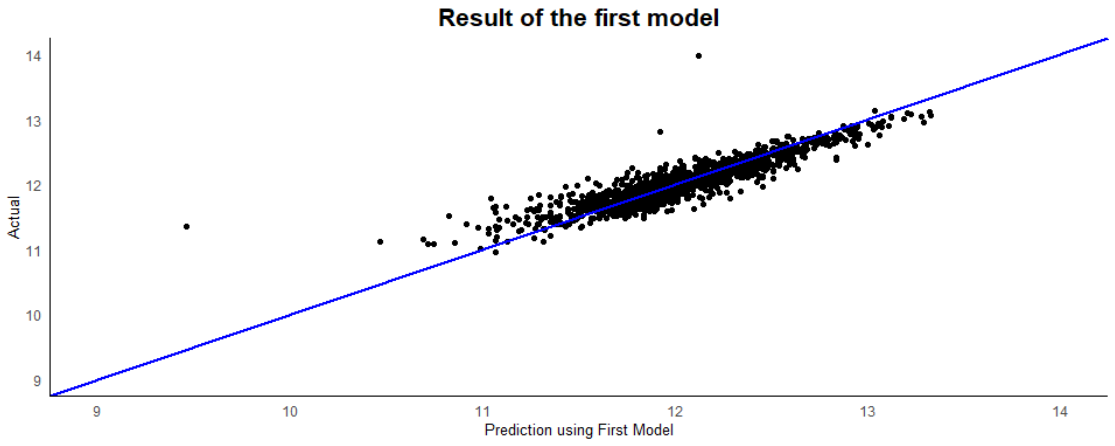
```
#Using the for loop to convert categorical variables to numeric variables
  for (j in seq(1:length(Nei_name))) {
  #since the name is in ascending order, we need to assign the largest number
to the first object.
  ts$Neighborhood[ts$Neighborhood==Nei_name[j]]= length(Nei_name)-j
  }
  for (j in seq(1:length(Sale_Cond_name))) {
  #since the name is in ascending order, we need to assign the largest number
to the first object.
  ts$SaleCondition[ts$SaleCondition==Sale_Cond_name[j]]=
length(Sale_Cond_name)-j
  }

  for (j in seq(1:length(HouseS_name))) {
  #since the name is in ascending order, we need to assign the largest number
to the first object.
  ts$HouseStyle[ts$HouseStyle==HouseS_name[j]]= length(HouseS_name)-j


  }
```

Prediction

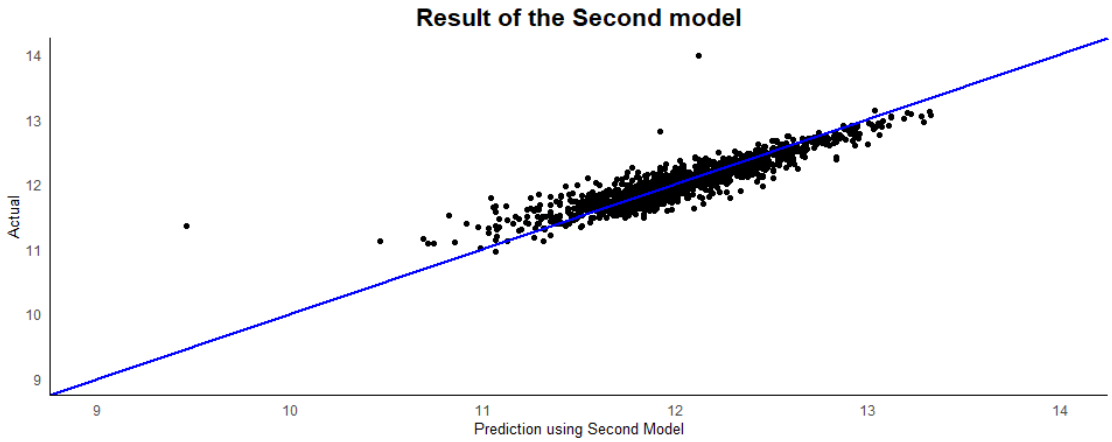
In this section, I will apply my best model to predict the test dataset and evaluate the prediction result by RMSE. In conclusion, the best RMSE is 0.1534255 and the graph visualize the accuracy of the model. The detials of finding will be presented in the next section.

```
#Convert ts$Saleprice to log scale
ts$LogSalePrice= log(ts$SalePrice)

#Make prediction with the forth model
ts$Prediction1= predict(First_M,ts)
ts$Prediction2= predict(Second_M,ts)
ts$Prediction3= predict(Third_M,ts)

#Calculate RMSE
rmse(ts$LogSalePrice, ts$Prediction1) #RMSE is 0.1726761

## [1] 0.1726761

rmse(ts$LogSalePrice, ts$Prediction2) #RMSE is 0.1600108

## [1] 0.1600108
```

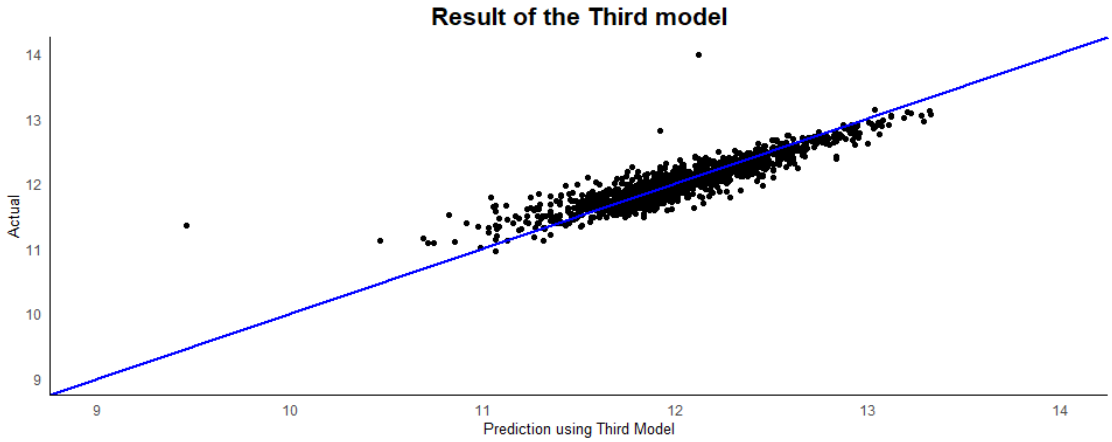
```
rmse(ts$LogSalePrice, ts$Prediction3) #RMSE is 0.1534255

## [1] 0.1534255

#Result of the first model
ggplot(ts,aes(x=LogSalePrice, y=Prediction1))+
  geom_point()+
  geom_abline( slope = 1, intercept = 0, color="blue" ,lwd=1)+
  scale_x_continuous(limits=c(9,14))+
  scale_y_continuous(limits=c(9,14))+
  ylab("Actual") +
  xlab("Prediction using First Model") +
  ggtitle("Result of the first model") +
  theme_classic()+
  theme(plot.title = element_text(size= 16, face= "bold", hjust= 0.5),
        axis.title.x = element_text(size= 10, hjust= 0.5),
        axis.title.y = element_text(size= 10, hjust= 0.5),
        axis.ticks = element_blank())
```




```
#Result of the second model
ggplot(ts,aes(x=LogSalePrice, y=Prediction1))+
  geom_point()+
  geom_abline( slope = 1, intercept = 0, color="blue" ,lwd=1)+
  scale_x_continuous(limits=c(9,14))+
  scale_y_continuous(limits=c(9,14))+
  ylab("Actual") +
  xlab("Prediction using Second Model") +
  ggtitle("Result of the Second model") +
  theme_classic()+
  theme(plot.title = element_text(size= 16, face= "bold", hjust= 0.5),
        axis.title.x = element_text(size= 10, hjust= 0.5),
        axis.title.y = element_text(size= 10, hjust= 0.5),
        axis.ticks = element_blank())
```




```
#Result of the third model
ggplot(ts,aes(x=LogSalePrice, y=Prediction1))+
  geom_point()+
  geom_abline( slope = 1, intercept = 0, color="blue" ,lwd=1)+
  scale_x_continuous(limits=c(9,14))+
  scale_y_continuous(limits=c(9,14))+
  ylab("Actual") +
  xlab("Prediction using Third Model") +
  ggtitle("Result of the Third model") +
  theme_classic()+
  theme(plot.title = element_text(size= 16, face= "bold", hjust= 0.5),
        axis.title.x = element_text(size= 10, hjust= 0.5),
        axis.title.y = element_text(size= 10, hjust= 0.5),
        axis.ticks = element_blank())
```



Recommendation and findings

In conclusion, firstly, I extracted the 5 variables with the highest correlation to SalePrice and then add them as my first model. Second, I selected 5 variables that could be the highest correlation to SalePrice by getting advices from the professionals and research and then, add them as my second model. For my third model, I combine the first model's

variables and the second model's variables together to create my third models. The R squared value of each model were calculated where the third model is considered as the best model. Since in the EDA section, I have spotted that the distribution of SalePrice was not a normal distribution. So, I apply log transformation to make a log normal distribution for SalePrice to optimize the performance of the models. Then, I trained all 3 models with the log-scale SalePrice. The significant improvement can be seen and I have stated in the previous section. Finally, I applied all the models to test dataset after it have been preprocessed. The best model is still the third model which contains all highest 5 numeric variables and my selected variables. The best model has roughly 0.85 R square value.

The models definitely could provide an estimation for all investor who is interested in Ames housing market. Except the prediction model, some insteresting findings are also spotted in the previous section. For instance, the most populor house styles are one story and 2 story, so the investors are encouraged to invest the house with these 2 styles. Also, for the neiborhood, the location is one of the most important factor of the property. NridgHt has the highest median sale price while MeadowV has the lowest meidan sale price. However, the most popular area are CollgCr, NAmes, Edwards and OldTown. The investors are suggested to invest CollgCr since the price range of CollgCr and the popularity of CollgCr are both high. For the sale condition, the most popular sale condition is normal, but it only at the second place in terms of the sale price. The sale condition partial took the first place and the median of the sale price is much geater than the sale condition normal, but it is much less popular to the sale condition normal. The investor should consider their own situations to decide which sale condition they should apply. For the ovreall quality, the additional insights were extracted. The most common OverallQual are 5, 6, 7. The distribution of OverallQual is roughly a symmetric distribution. For living area, the distribution of living area against sale price is that the smaller living areas the property has, the smaller price range it tends to get. On the other hand, the larger living area the property has, the larger price range it tends to have. For example, the small living areas less than 1300 are more likely to have a certain price.

However, for my selected variables, 3 of them have been proved that they have high correlation with SalePrice, but 2 of them which are SaleCondition and HouseStyle are not significantly correlated to SalePrice. I was overestimate the correlation of HouseStyle and SalesCondition where they only has around 0.3 correlation value to the price. So, the further work can be finding another 2 better variables to optimzie the models such as the year built and other categorical variables.

Reference

Pettinger, T., 2021. Definition of the housing market - Economics Help. [online] Economics Help. Available at: https://www.economicshelp.org/blog/glossary/definition-of-the-housing-market/ [Accessed 1 May 2021].

Evans, D., 2021. abnormal sale. [online] TheFreeDictionary.com. Available at: https://financial-dictionary.thefreedictionary.com/abnormal+sale [Accessed 7 May 2021].

Sarafian, R., 2021. House Price Prediction. [online] Rstudio-pubs-static.s3.amazonaws.com. Available at: https://rstudio-pubs-static.s3.amazonaws.com/247652_bb5c001d6f7642d88f9ff66ecf1e28a3.html?fbclid=IwAR0oNc8USLIStQgoaMAq7vYX4cUHgJCZ46o9FQZP12I5SzDeT2E9cbsHO5Q [Accessed 12 May 2021].

StatCanada, 2021. Dwelling type of dwelling. [online] Www23.statcan.gc.ca. Available at: https://www23.statcan.gc.ca/imdb/p3Var.pl?Function=DEC&Id=323163 [Accessed 5 May 2021].

Ames Gov, 2021. Reports | City of Ames, IA. [online] Cityofames.org. Available at: https://www.cityofames.org/government/departments-divisions-a-h/city-assessor/reports [Accessed 5 May 2021].

Ames Mayor John Haila, 2021. [online] Cityofames.org. Available at: https://www.cityofames.org/home/showpublisheddocument/61546/637552045376700000 [Accessed 12 May 2021].