30th March 2022

GROUP PROJECT FOR MA 321: APPLIED STATISTICS

TOPIC: Descriptive Analysis of Housing Dataset

Faculty in-charge: Dr. Fanlin Meng

No.	Full Name	Registration Number	Email	Question
1	Zia Ullah Khan	2101142	zk21867@essex.ac.uk	1,2,3,4+ Report
2	Wahyu Maiwa	2107460	wm21615@essex.ac.uk	1,2,3,4+ Report
3	Muhammad Ehsan	2100504	me21594@essex.ac.uk	1,2,3,4+ Report
4	Muhammad Hamza Shehzad	2100509	ms21730@essex.ac.uk	1,2,3,4+ Report
5	Muhammad Taimoor Khan Malik	2101295	mm21571@essex.ac.uk	1,2,3,4+ Report
6	Areej Sharafat	2101262	as21185@essex.ac.uk	1,2,3,4+ Report
7	Aeman Zehra	2100681	az21206@essex.ac.uk	1,2,3,4+ Report
8	Prateek Tiwary	2100920	pt21775@essex.ac.uk	1,2,3,4+ Report

Abstract

Property valuation is a difficult discipline to grasp. Each appraiser and valuer bring their own set of abilities, measures, and knowledge to the table. According to studies undertaken by most of the organisation across the group, valuations between two professionals can differ by a large amount.

A well-trained machine might be able to perform this task more consistently and precisely than a human. Let's put this theory to the test and train some machine learning models to predict a house's value based on information about its features, costs, and neighbourhood profile. The machine learning problem is regression since the goal variable, property price, is numerical. Classification may have been explored if the dataset had been categorical.

Content

A. Introduction

Accurately estimating the value of real estate is an important problem for many stakeholders including house owners, house buyers, agents, creditors, and investors. It is also a difficult one. Though it is common knowledge that factors such as the size, number of rooms and location affect the price, there are many other things at play. Additionally, prices are sensitive to changes in market demand and the peculiarities of each situation, such as when a property needs to be urgently sold. The sales price of a property can be predicted in various ways but is often based on regression techniques. All regression techniques essentially involve one or more predictor variables as input and a single target variable as output. In this paper, we compare different machine learning methods performance in effect on the selling price of houses based on a few features such as the area, the year of build, year of sold, Garage area. Also, on basis of various model for predicting sales price.

B. Methodology:

To begin with, providing numerical and graphical summaries of the data set and make any initial comments that are appropriate. The process which has several steps such as Processing of the data; Exploratory data analysis: Resulting in graphs and summaries; Applying Feature Engineering; Based on it run suitable models Regression, Random Forest, and logistic regression.

1. Pre-processing of data

The process of pre-processing of data starts by installing of all the necessary packages and loading of the of the data. Post to this one has to run few commands such as creating a data frame, identifying of the data structure. This is followed with locating and dealing with the missing values. Filling of missing value is very important, and it can be done via mean imputation to the numerical column. As a result, all the null values will be imputed. The data we have is normally distributed.

There are several correlation testing's done of various variables on sales price, their outcomes are mentioned below:

a) **Year build**: The result is only 52 %, Which is not the same as earlier figure of 1980s.
b) **Year Sold**: There is no direct relationship between the two.
c) **Lot Area**: Lot area has not much effect on sale price, there are however, outliers which if removed, this can become a positive corelation.
d) **Masonry veneer Area**: There positive relationship between sale price and Masonry veneer Area
e) **Total square feet of basement area**: There positive relationship between sale price and total square feet of basement area, having basement increased sale price.
f) **living area square feet**: There is very strong positive correlation, the more living area square feet, more is sale price
g) **Garage Area**: Strong positive correlation, Garage location has great impact on sale price

2. Primary analysis (Question1)

Through monthly density it can be easily observed that sales price has significantly increase from Mid-April to August.

We are reporting the dataset based on Neighbourhood location that is grater then to median sales prices with respect to the Neighbourhood, and on the other hand Statistics help to identify the neighbourhood's most costly, least expensive, and most diverse housing prices. We also plot Price Variation per Neighbourhood and calculate the most and least expensive neighbourhood and we assume that the most expensive neighbourhood that is NridgHt (Northridge Heights) and it's price is 315000 and the least expensive neighbourhood is MeadowV (Meadow Village) and it's price is 88000, and after that we calculate the variation in prices between least and most expensive Neighbourhood and we note that the most variation in price is NoRidge (Northridge) 121413 and least variation in price is NPkVill (Northpark Villa) 9377.

Now, let's plot boxplots to examine outliers, the fig1 boxplot we analyse that there are some outliers here and we need to remove outliers.

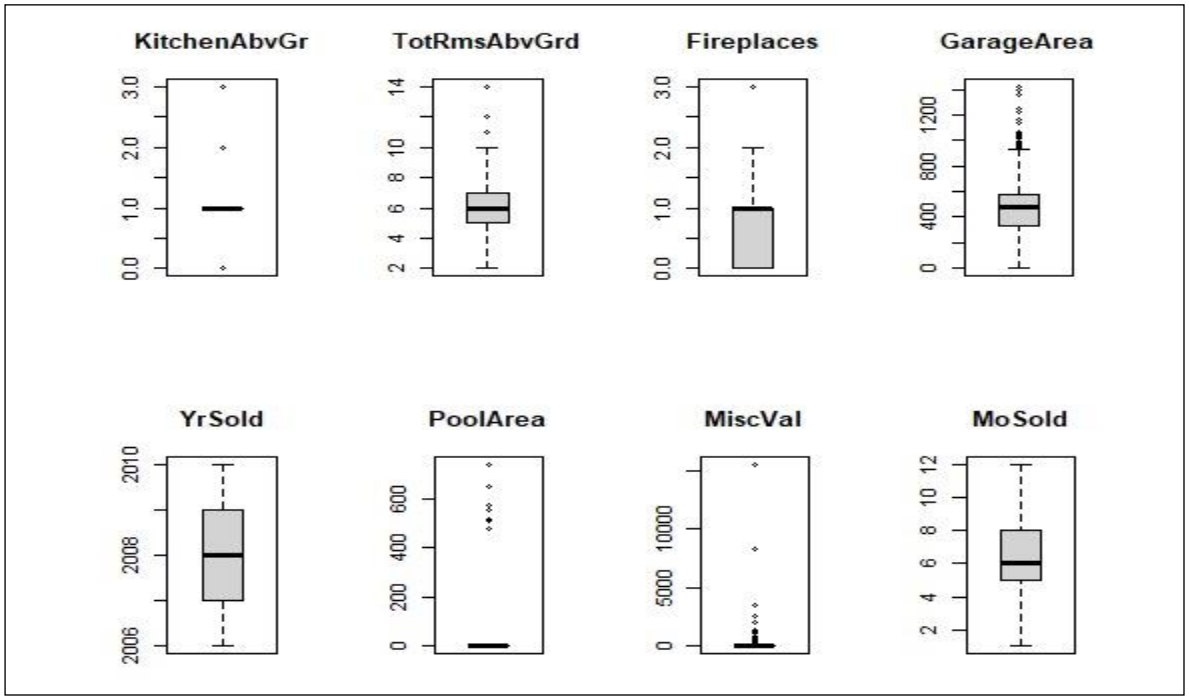

Figure 1: boxplot

The distribution of features is shown by plotting the histograms that are shown below:

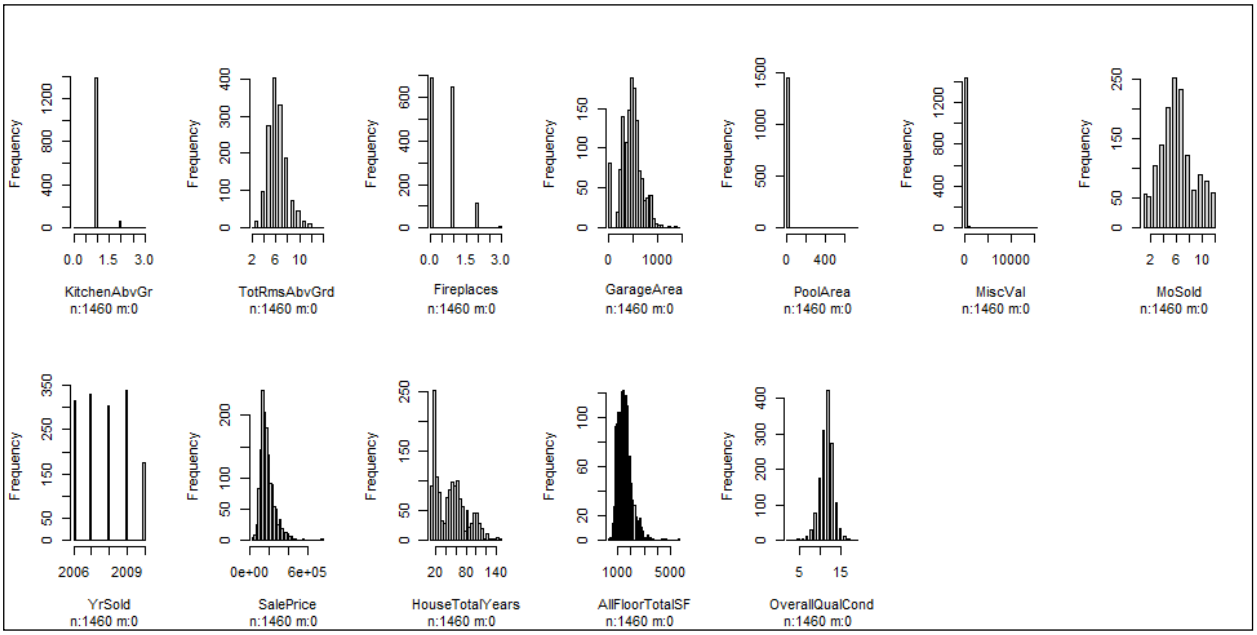

Figure 2: Histogram

By creating correlation plot we are seeming variations in correlation and plot correlation matrix. It is vivid from the graph that Garage Area, Overall Quall are highly correlated with sales price, hence their relationship is quite linear. Now, eliminating columns on the bases of correlation matrix because these columns are corelated, and they will be removed from the matrix.

C. Analysis and discussion (Question 2)

The nominal target variable is predicted using multinomial regression. We must apply ordinal logistic regression if the target variable is of the ordinal type. We will learn how to perform multinomial logistic regression in this lesson. Ensure that data is devoid of multicollinearity, outliers, and high influential leverage points as part of data preparation.

In this tutorial, we'll use numerical data from housing to classify it. Originally, the data consisted of several factors that were divided into three categories: poor, average, and good. Combining target variable levels and removing the 'overall condition' because it is a distinct variable. Using a function from the **dplyr** package to split the data.

In multinomial logistic regression, we must determine the reference level, unlike binary logistic regression. Please keep in mind that this is specific to the function from the nnet package in R that I'm using. Some functions from other R packages don't require the reference level to be specified before the model can be built. The **multinorm** function from the **nnet** package will be used to train the model. The summary () function will be used to check the model coefficients once the model has been trained.

We must convert the coefficients to odds using the exponential of the coefficients, same as we did with binary logistic regression. In the model object, the predicted values are kept as fitted values. Let's have a look at the top six observations.

The multinomial regression predicts the likelihood that a given observation will be included in the given level. In the table below Figure, we can see that this is the case. The classification levels are represented by columns, while the observations are represented by rows. The accuracy of the model will be examined to validate it. The classification table can be used to calculate this accuracy.

In the test dataset, we were able to reach 100% accuracy, which is very near to the train, therefore we can infer that the model is good and stable. We learned how to create a multinomial logistic regression model, validate it, and make a prediction on an unknown dataset in this course.

Figure bellow show data frame created from the housing dataset.

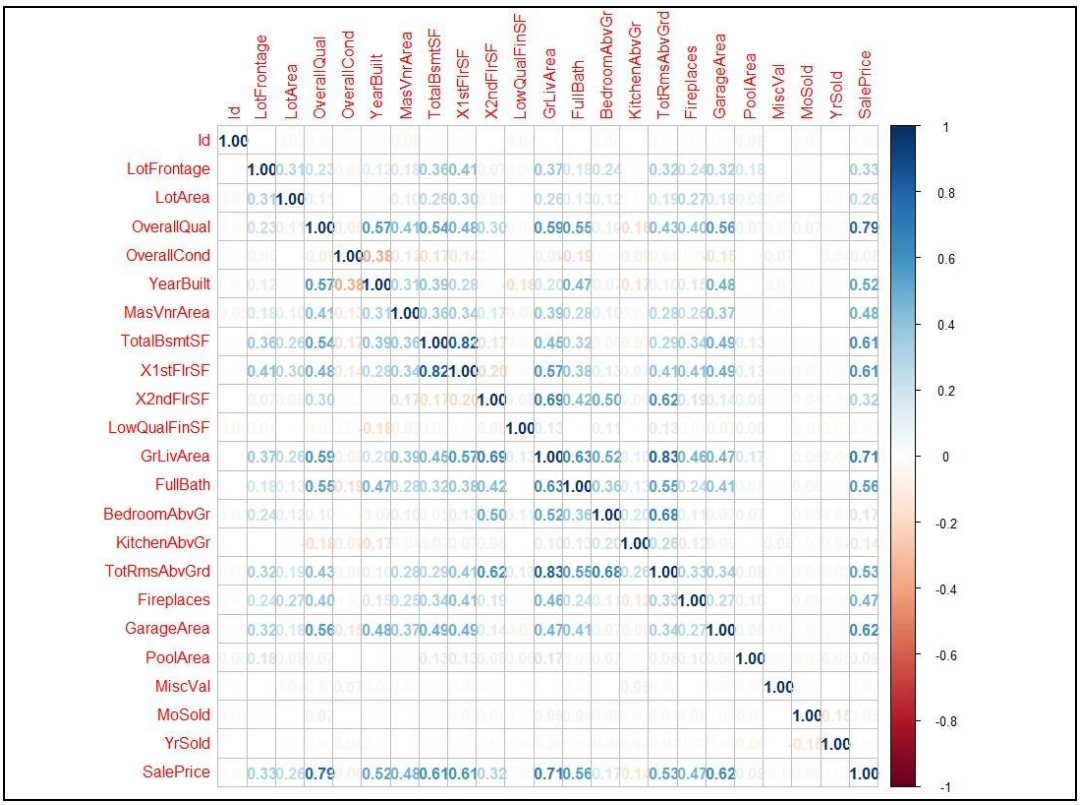

Figure 3: Showing Data frame

It can be observed from the graph that Garage Area, OverallQual, general living area are highly correlated with price variable. It is also visible that they show linear relationship removing columns based on the correlation matrix following columns will be eliminated from the matrix as these are not correlated.

SVM is used to train a support vector machine. It can be used to carry out general regression and classification (of nu and epsilon-type), as well as density-estimation. A formula interface is provided. Supervised learning is when you train a machine learning model using labelled data. It means that you have data that already have the right classification associated with them. One common use of supervised learning is to help you predict values for new data.

Support vector regression (SVR), which is an extension of support vector classification, is one example of a sort of SVM that can be used for specific machine learning issues (SVC). What distinguishes the linear SVM algorithm from others, such as k-nearest neighbours, is that it selects the best line for classifying your data points. It selects the line that divides the data and is the farthest distant from the closest data points.

a) SVMs come in a variety of shapes and sizes.

b) SVMs are divided into two categories, each of which is employed for a different purpose:

- Simple SVM: Typically used for tasks involving linear regression and classification.
- Kernel SVM: Has more flexibility for non-linear data because it can fit a hyperplane instead of a two-dimensional space with more features.

c) SVMs are employed in machine learning for a variety of reasons.

Handwriting recognition, intrusion detection, face detection, email categorization, gene classification, and web page classification all use SVMs. One of the reasons we employ SVMs in machine learning is for this reason. On both linear and non-linear data, it can do classification and regression. Another reason we utilise SVMs is that they can uncover intricate associations between your data without requiring you to perform a lot of manual modifications.

Both Figures show us the prediction and summary

```
          prediction
           Average Good Poor
  Average      271    0    0
  Good           6   78    0
  Poor           4    0    6
>                (271    +78    +6)/nrow(test_svm)  # Compute prediction accuracy
[1] 0.9726027
>                svm1$labels
[1] 1 3 2
```

Figure 4: Sales Price Prediction

```
>                summary(svm1)

Call:
svm(formula = Class ~ ., data = train_svm, method = "C-classification", kernal = "radial", gamma = 0.1, cost = 10)


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  10

Number of Support Vectors:  402

 ( 273 21 108 )


Number of Classes:  3

Levels:
 Average Good Poor
```

Figure 5: Summary

SVM in this context is supposed to guess the correct house ratings as per the features feeded. There are 3 levels of ratings and it requires 402 support vectors to fit the line while applying radial kernel and gamma of .25. The goal of SVM is to minimize the space between different data points. Kernel trick is used so that to estimate the overall distance without computing different parameters and its calculations across all of the dimensions.

D. Analysis and discussion (Question 3)

The prediction of sales prices or housing prices as in our case can be performed through various machine learning algorithms. As per the researcher's thought, the importance of ensemble learning has improved the models more and made them suitable for our use considering how the data itself is of importance when added to the models. The data sets have to be dealt with pre-processing by making sure how the numerical and categorical data will be handled. In our case, both numerical and non-numerical data sets exist but for our findings, we require only numerical data as the non-numerical seems to show no such impact on the response variables. Amongst numerous Machine Learning models such as Tree-based, K-NN, Support vector machines, neural networks, and Naive Bayes. But, it is highly advised to use classification or Regression approaches using the simplest yet the most powerful analysis which is by deploying random forest and linear regression. Our studies and findings have also suggested the same. Initially, the dataset is partitioned into two parts; training at 75% and validation at 25%. As per our understanding, we will be taking into account the most significant numerical variables such as LotArea, OverallQual, OverallCond, TotalBsmtSF, GrLivArea, FullBath, BedroomAbvGr, KitchenAbvGr, GarageArea, and PoolArea". All of these have a huge impact on the sales prices as they determine the cost of the housing.

The first method we going to use for predicting the prices is linear regression method is linear regression.

1. Linear regression

Linear regression analysis is a predictive modeling method that estimates the relationship between two variables or more than two variables. In linear regression, we focus on the relationship between the dependent (targeted one) variable and independent variables or predictors. More specifically we can say that the regression analysis helps us to understand how the value of a dependent variable changes when one of any independent variable's values changes. We use the linear regression in our table because of the following reasons:

- The simple implementation
- Performance on linearly separatable datasets
- Regularization can reduce the overfitting

In this question, we have to conduct several tests among them the most prefer one is linear regression analysis under this a model needed to be fit it for predicting the value the variable that be considered the variable that very shortlisted were Garage Area, Lot Area, Neighbourhood, overall call and many similar properties with respect to the sale prices. The tabulation that was conducted were mainly on some of square error and some of square total. All together there were 8 possibilities tested they had separate summary score**.**

Linear regression best model with stepwise model path (backward) selection method, here we are getting comparatively less AIC value after removing categorical feature ExterQual as mentioned in the graph.

2. Random Forest

Random forest is an example of supervised ensemble learning as mentioned previously. It is an example of a tree-based approach where the results are not dependent on one tree analysis instead many trees. Random forest is said to take the wisdom of the crowd where various parameters help in finding the strong and accurate result by keeping the cp (complexity parameter) optimal and letting the tree grow till it is required. MTRY and Ntrees are other few which enable the variables to be randomly sampled at each split. We use the random forest regression because it is used to solve the variety of business problems like Predicting the upcoming prices, upcoming revenue and compare performance.

The second method that was tested was using random forest under this as well the objective was the same the predictive sale prices. The purpose of running the model was to reach and comparing with earlier model. So, under the random forest will be helpful for the resampling and use of multiple decision trees bootstrapping approach was followed in which mean all the models were consider and the best fit value was taken. Random forest residuals and RMSE value: 34107 which has been reduced from a larger value.

```
#test2.SalePrice pred.house  residuals
#7           307000   258590.1  48409.925
#13          144000   127942.2  16057.767
#15          157000   147572.4   9427.612
#18           90000   110798.1 -20798.059
#23          230000   250193.6 -20193.633
#28          306000   291588.2  14411.824

#Calculating the root square mean error (RSME)
#RMSE(pred.house, test2$SalePrice)      RMSE:  34107.52
```

Table 1: Random Forest

On comparing the two methods that is linear regression and random forest it can be clearly the noted that random forest has a better accuracy (97%) with the match lower root mean square error.

E. Analysis and discussion (Question 4)

The house-data set can be explored in various ways and numerous research questions can be raised from within. The data set has quite a few important variables and it can be explored numerically as well as categorically. The important variables in the data set are but not limited to sale price with respect to building size, neighbourhood, availability of garages, pools, quality of construction, current condition, proximity to various conditions, type of utilities available and lot size. and such questions are discussed in detail below:

- What is the relationship between Neighbourhood and the sale price?
- Difference between the prices of houses for different lot size or building size.
- How does the availability of garages and pools effect the prices of the building?
- What is the effect of type of utilities availability?

- Does the proximity to various conditions effect the prices of the house?
- What is the relationship between the current condition of the house and the prices of the property?

Clustering analysis is a statistical approach of organising the data into clusters based on their similarity and closeness to each other reflecting association. It is the analysis for classification to make groups that have similar attributes.

1. K-Means & Hierarchical Method

Specific methods for the clustering analysis here are K-mean and hierarchical method. K-mean find the clusters based on centroids and their Euclidean distance between each point. And in hierarchical method, every data point is considered a separate point and similar clusters merge on each iteration. Principle component analysis is used to reduce the number of dimensions through the algorithm. Nan values in the numeric data set were imputed using single mean imputation method to make the model more prone to outliers.

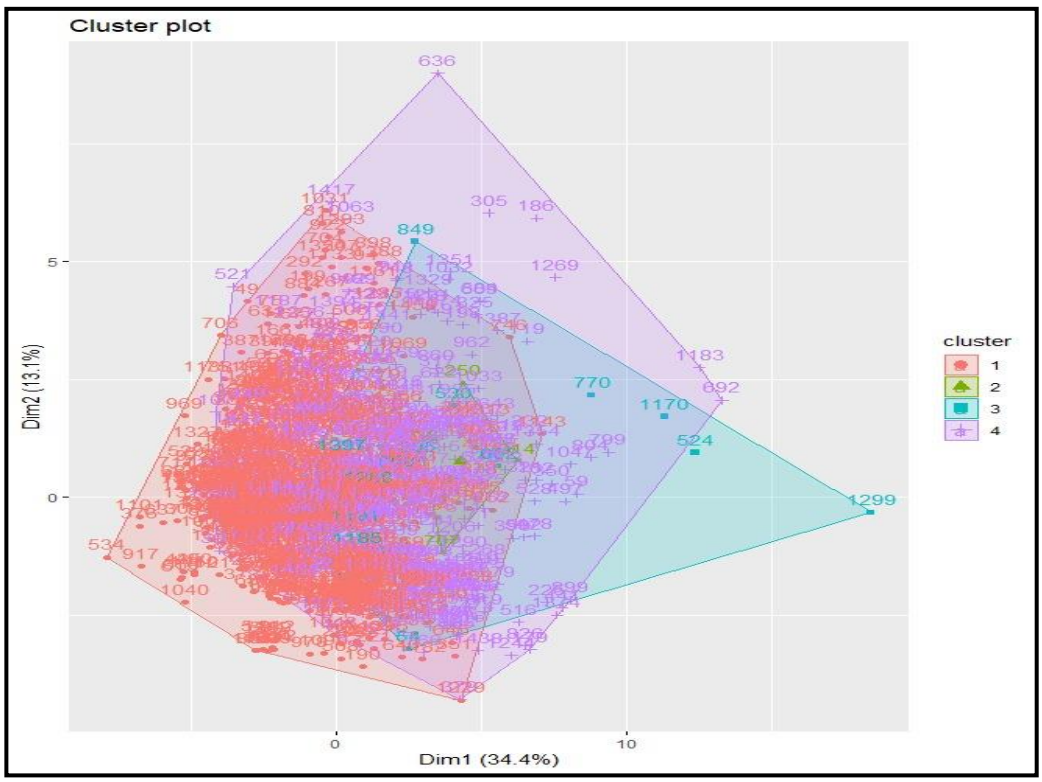

Figure 6: Clustering Analysis

In the figure shown adjacently, we have obtained the correlation percentage of each component and included ones with the highest percentage for the iteration of K in k-means. Therefore, the number of K we used in the k-mean are 12. It is also one of the drawbacks of the k-means clustering analysis that prior knowledge must be assumed, and appropriate number of K are required to be chosen. Also, while we have also used the principal component analysis, it is said with confidence that the PCA increased the variance as the number of components increased.

Components	Correlation %
1	2.75%
2	1.70%
3	1.35%
4	1.31%
5	1.03%
6	1.01%
7	0.95%
8	0.88%
9	0.87%
10	0.84%
11	0.77%
12	0.74%

Table 2: Component Correlation Percentage

However, if we talk about the hierarchical method of clustering analysis, it can be observed from the dendrogram figure which clearly states the distance matrix and similarities between the components.

Even if we observe while looking at the dendrogram obtained by applying the hierarchical clustering method, it re-iterates the finding of the K-Mean that most of the components belong to different clusters and 14 of the clusters are majorly similar out of 51 individual clusters.

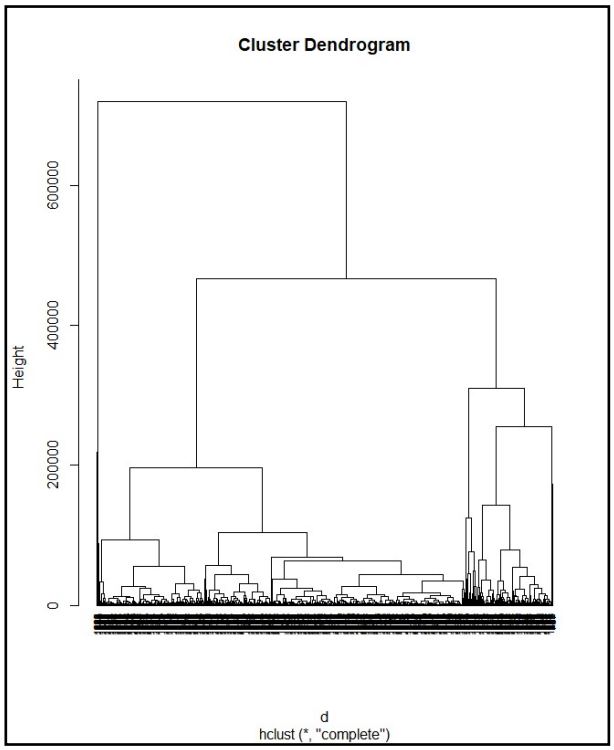

Figure 7 - Dendrogram

F. Conclusion

A multiple regression model was developed to predict the selling price of a given homes as per housing data. The model is restricted to house that were sold under normal conditions, with a prementioned LOT area, only living and non-commercial properties were open for sale.

The model is suitable only for non-atypical houses, that have significantly different characteristics that majority of houses in the data used for the model creation.

In this model the price dependent on living area, LOT area and Garage Area; exponentially on the year built, year of sold. The model has been tested and validated and has shown to perform similarly on all sets.

The model underpredicts the prices of the expensive house. From investment point of view, this will not be acceptable as actual profits would be higher than accepted.

Reference:

1. University of Essex
2. Brad Boehmke, Brandon Greenwell. HandsOn Machine Learning with R. 1st. Boca Raton: Chapman and Hall/CRC, 2019.
3. Breiman, Leo. “Bagging Predictors”. In: Mach. Learn. 24.2 (Aug. 1996), pp. 123–140. ISSN: 08856125. DOI: 10 . 1023 / A : 1018054314350. URL: https://doi.org/10.1023/A:1018054314350.
4. Pyle, Dorian. Data Preparation for Data Mining. 1st. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1999. ISBN: 1558605290.
5. Varma, A. et al. “House Price Prediction Using Machine Learning and Neural Networks”. In: 2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT). 2018, pp. 1936–1939. DOI: 10.1109/ICICCT.2018.8473231.

Appendix:

```
#Packages required
install.packages('gapminder')
install.packages('finalfit')
install.packages('Hmisc')
install.packages('ggpubr')
install.packages('psych')
install.packages("mice")
install.packages('faraway')
install.packages('corrplot')
install.packages('mlbench')
install.packages('caret')
```

```
#loading libraries
library(ggplot2)
library(finalfit) #package for finishing tabulation % reference: https://finalfit.org/
library(gapminder) #package for finding missing values % reference: https://cran.r-project.org/web/packages/gapminder/README.html
library(Hmisc)
library("ggpubr") # package must be installed first
library(psych)
library(mice) #reference https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4701517/
library(VIM) #reference https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4701517/
library(faraway)
```

```
library(corrplot)
library(mlbench)
library(caret)
library(dplyr)
```

##1. Provide numerical and graphical summaries of the data set and make any initial comments that you deem appropriate.

```
#1. Worked on the process by following steps:
#2. Process the data
#3. Exploratory Analysis: Graphs, Summaries
#4. Feature Engineering
#5. Models - Regression, Random Forest, Logistic regression
#6. Model Validation

#load in data
house_df <- read.csv("house-data.csv")

## List characteristics of the dataframe
head(house_df)

colnames(house_df)
summary(house_df)

# Structure of dataset
 str(house_df)    # 1460 obs. of  51 variables

# Finding Missing values in all columns
```

```
missing_glimpse(house_df) # missing data for each variable

#label                  var_type   n missing_n missing_percent

# LotFrontage     LotFrontage    <int> 1201      259          17.7
# Alley           Alley    <chr>   91     1369          93.8
# MasVnrArea      MasVnrArea    <int> 1452       8           0.5
# BsmtQual        BsmtQual    <chr> 1423       37           2.5
# BsmtCond        BsmtCond    <chr> 1423       37           2.5
# GarageType      GarageType    <chr> 1379       81           5.5
# GarageCond      GarageCond    <chr> 1379       81           5.5
# PoolQC          PoolQC    <chr>    7     1453          99.5
# Fence           Fence    <chr>  281     1179          80.8
# MiscFeature     MiscFeature    <chr>   54     1406          96.3

sapply(house_df[, c(1:51)], function(x) {sum(is.na(x))})

table(house_df$Fence)
table(house_df$Alley)
table(house_df$PoolQC)
table(house_df$MiscFeature)
table(house_df$GarageCond)
table(house_df$GarageType)
table(house_df$BsmtCond)
table(house_df$BsmtQual)
table(house_df$Alley)

#fill missing values / imputing missing values
```

```
house_df$Fence[is.na(house_df$Fence)] <- 'MnPrv'

house_df$MiscFeature[is.na(house_df$MiscFeature)] <- 'Shed'

house_df$PoolQC[is.na(house_df$PoolQC)] <- 'Gd'

house_df$GarageCond[is.na(house_df$GarageCond)] <- 'TA'

house_df$GarageType[is.na(house_df$GarageType)] <- 'Attchd'

house_df$BsmtCond[is.na(house_df$BsmtCond)] <- 'TA'

house_df$BsmtQual[is.na(house_df$BsmtQual)] <- 'TA'

house_df$Alley[is.na(house_df$Alley)] <- 'Grvl'

house_df$MasVnrArea[is.na(house_df$MasVnrArea)] <- mean(house_df$MasVnrArea, na.rm = T)

house_df$LotFrontage[is.na(house_df$LotFrontage)] <- mean(house_df$LotFrontage, na.rm = T)

#It can be seen that it is a better option to apply mean imputation to the numeric columns data

summary(house_df$MasVnrArea)

summary(house_df$LotFrontage)

# Null values are all imputed as can be seen by running below function

sapply(house_df[, c(1:51)], function(x) {sum(is.na(x))})
```

```
#Graphing & Summaries


# Splitting Categoric, and Numeric  features (splitting house dataframe in two)
numeric_cols <- names(which(sapply(house_df, is.numeric)))
categorical_cols <- names(which(sapply(house_df, is.character)))

numeric_features <- house_df[, names(house_df) %in% numeric_cols]
categorical_features <- house_df[, names(house_df) %in% categorical_cols]


#Plots ~ boxplot, scatterplots


pairs( ~ YearBuilt  + OverallQual + TotalBsmtSF + GrLivArea, data = numeric_features,
     main = "Scatterplot")

pairs( ~ YearBuilt + OverallQual + TotalBsmtSF + GrLivArea, data = numeric_features, main = "Scatterplot")


house_df[which.max(house_df$SalePrice),]  # max sale price : 755000
house_df[which.min(house_df$SalePrice),]  # min sale price : 34900


summary(numeric_features)

# Most of the numerical columns are normally distributed
hist.data.frame(numeric_features)  # distribution of numeric features in histogram
```

 # 1) year built ~ sale price relationship

#It can be observed that correlation between the two variables is 52%, which is does not show positive relation

#However, it can be seen that after 1980, the relationship is quite strong,

#Overall the relationship shows if the house is newer has no positive correlation on price, means price does not increased as such

```
plot(numeric_features$SalePrice ~ numeric_features$YearBuilt, ylab = 'Sale Price', xlab= 'Year Built')
```

 # 2) year sold ~ sale price relationship

#There is no special relationship between two

```
plot(numeric_features$SalePrice ~ numeric_features$YrSold, ylab = 'Sale Price', xlab= 'Year Sold')
```

 # 3) Lot Area ~ sale price relationship

#Lot area has not much effect on sale price, there are however, outliers which if removed, this can become a positive corelation

```
plot(numeric_features$SalePrice ~ numeric_features$LotArea, ylab = 'Sale Price', xlab= 'Lot Area')
```

 # 4) Masonry veneer Area ~ sale price relationship

#There positive relationship between sale price and Masonry veneer Area

```
plot(numeric_features$SalePrice ~ numeric_features$MasVnrArea, ylab = 'Sale Price', xlab= 'Masonry veneer Area')
```

 # 5) total square feet of basement area ~ sale price relationship

#There positive relationship between sale price and total square feet of basement area, having basement increased sale price.

```
plot(numeric_features$SalePrice ~ numeric_features$TotalBsmtSF, ylab = 'Sale Price', xlab= 'Basement area sq.feet')
```

 # 6) First Floor square feet ~ sale price relationship

 # There is strong positive correlation, the more fist floor square feet, the more is the sale price

```
plot(numeric_features$SalePrice ~ numeric_features$X1stFlrSF, ylab = 'Sale Price', xlab= 'First Floor square feet')
```

```
colnames(numeric_features)
```

 # 7) Second Floor square feet ~ sale price relationship

 # There is again positive correlation, the more second floor square feet, the more is the sale price

```
plot(numeric_features$SalePrice ~ numeric_features$X2ndFlrSF, ylab = 'Sale Price', xlab= 'Second Floor square feet')
```

 # 8) living area square feet ~ sale price relationship

 # There is very strong positive correlation, the more living area square feet, more is sale price

```
plot(numeric_features$SalePrice ~ numeric_features$GrLivArea, ylab = 'Sale Price', xlab= 'living area square feet')
```

 # 9) GarageArea ~ sale price relationship

 # Strong positive correlation, Garage location has great impact on sale price

```
plot(numeric_features$SalePrice ~ numeric_features$GarageArea, ylab = 'Sale Price', xlab= 'Garage Area')
```

```
#Other variables too had less affect on sale price


# Monthly Density

# to find out which months have higher sale prices compare to others

# It is clear that from the mid april till August sale prices are significantly increased

ggplot(data = numeric_features[numeric_features$SalePrice > 34000,], aes(x = MoSold)) +

  geom_histogram() +

  stat_bin(bins = 12, binwidth = 1) +

  geom_density(aes(y = ..density..)) + # Density plot

  theme_minimal() +

  labs(x = "Month Sold", y = "Sales Price", title = "sale prices in different months") +

  scale_x_continuous(breaks = c(1,2,3,4,5,6,7,8,9,10,11,12))



#Reporting on the basis of Neighborhood location > median sales prices with respect to Neighborhood

# Statistics to determine the most expensive, least expensive and highly varied house prices in the neighborhood


ggplot(data = house_df , aes(SalePrice)) + geom_histogram() + theme_classic() +

  xlab("sale price of homes") +

  ggtitle("Distribution of home prices")


Location_based_reporting <- house_df %>%

  group_by(Neighborhood) %>%

  summarise(Price_Neighborhood = median(SalePrice)) %>%

  arrange(Price_Neighborhood)
```

```
head(Location_based_reporting)


ggplot(data = Location_based_reporting, aes(Neighborhood, Price_Neighborhood)) +
  geom_jitter(aes(color = Neighborhood, size = Price_Neighborhood)) +
  theme(axis.text.x = element_text(angle = 90)) +
  theme(legend.position = "none") +
  ggtitle("Median Prices in relation to Neighborhoods") +
  xlab("Neighborhoods") +
  ylab("Median Price") +
  theme(plot.title = element_text(size = 10)) +
  theme(axis.title = element_text(size = 10))



# Price Variation per Neighborhood
Location_based_reporting_Var <- house_df %>%
  group_by(Neighborhood) %>%
  summarise(Price_Var_Nhood = sd(SalePrice)) %>%
  arrange(Price_Var_Nhood)


ggplot(data = Location_based_reporting_Var, aes(Neighborhood, Price_Var_Nhood)) +
  geom_jitter(aes(color = Neighborhood, size = Price_Var_Nhood)) +
  theme(axis.text.x = element_text(angle = 90)) +
  theme(legend.position = "none") +
  ggtitle("Price Variation in relation to Neighborhoods") +
  xlab("Neighborhoods") +
  ylab("Price Variation") +
  theme(plot.title = element_text(size = 10)) +
  theme(axis.title = element_text(size = 10))
```

#Most and least expensive neighborhood

```
Location_based_reporting[which.max(Location_based_reporting$Price_Neighborhood)
,]
```

#Neighborhood Price_Neighborhood

#1 NridgHt 315000

#least expensive neighborhood

```
Location_based_reporting[which.min(Location_based_reporting$Price_Neighborhood)
,]
```

#Neighborhood Price_Neighborhood

 # 1 MeadowV 88000

 # Neighbor hoods with most, least Variation in price

#Most variation in price

```
Location_based_reporting_Var[which.max(Location_based_reporting_Var$Price_Var_
Nhood),]
```

 # Neighborhood Price_Var_Nhood

 # NoRidge 121413.

```
Location_based_reporting_Var[which.min(Location_based_reporting_Var$Price_Var_
Nhood),]
```

#Neighborhood Price_Var_Nhood

 # NPkVill 9377.

```
 # boxplots to examine outliers


par(mfrow=c(2,7))
for(i in 1:22) {
  boxplot(numeric_features[,i], main=names(numeric_features)[i])
}



#Showing the distirubtion of age of the houses is right skewed
summary(numeric_features$HouseTotalYears)

#Histogram to show distribution of features
hist.data.frame(numeric_features)



#Boxplot to Analyze

par(mfrow=c(2,7))
for(i in 1:22) {
  boxplot(numeric_features[,i], main=names(numeric_features)[i])
}

 # Feature Engineering


numeric_features$HouseTotalYears <- (2022 - numeric_features$YearBuilt)

numeric_features$NewHouse <- (numeric_features$YearBuilt ==
numeric_features$YrSold) * 1
```

```
numeric_features$AllFloorTotalSF <- numeric_features$X1stFlrSF + numeric_features$X2ndFlrSF

numeric_features$OverallQualCond <- numeric_features$OverallCond + numeric_features$OverallQual


# Create correlation plot variations
par(mfrow = c(1,1))
correlation_matrix <- cor(numeric_features,  use = 'everything')

correlation_matrix

#Correlation plot
corrplot(correlation_matrix, method = "number", type = "full")
# It can be observed from the graph that Garage Area, OverallQual, general living area are highly correlated with price variable
# It is also visible that they show linear relationship

# Removing Columns based on the correlation matrix
# Following columns will be eliminated rom the matrix as these are not correlated

colnames(categorical_features)

numeric_features <- numeric_features[,!colnames(numeric_features) == "LowQualFinSF"]
numeric_features <- numeric_features[,!colnames(numeric_features) == "MiscVal"]
numeric_features <- numeric_features[,!colnames(numeric_features) == "BsmtFinSF2"]
numeric_features <- numeric_features[,!colnames(numeric_features) == "MoSold"]
```

```
numeric_features <- numeric_features[,!colnames(numeric_features) == "PoolArea"]

categorical_features <- categorical_features[,!colnames(categorical_features) == "PoolQC"]

numeric_features <- numeric_features[,!colnames(numeric_features) == "Id"]


colnames(categorical_features)

colnames(numeric_features)



## Transformation:

# At this section we will transform some of the feature to log transformation or sqrt transformation so if some feature's

# distribution is left or right skewed, it will normally distribute it.


grouped_df2 <- house_df


# for normal distribution

grouped_df2$LotArea <- log(grouped_df2$LotArea)

grouped_df2$GarageArea <- log(grouped_df2$GarageArea)

grouped_df2$SalePrice <- log(grouped_df2$SalePrice)

grouped_df2$LotFrontage <- log(grouped_df2$LotFrontage)

grouped_df2$MasVnrArea <- log(grouped_df2$MasVnrArea)

grouped_df2$TotalBsmtSF <- log(grouped_df2$TotalBsmtSF)

grouped_df2$X1stFlrSF <- log(grouped_df2$X1stFlrSF)

grouped_df2$X2ndFlrSF <- log(grouped_df2$X2ndFlrSF)

grouped_df2$GrLivArea <- log(grouped_df2$GrLivArea)



#Grouping categorical + continuous features

grouped_df <- cbind(categorical_features, numeric_features)
```

```
#Label encoding/factorizing the character variables
grouped_df <- grouped_df %>%
  mutate_if(is.character, as.factor)
grouped_df

str(grouped_df)




#----------Question 2 ------------------------#

install.packages('nnet')  # This package is used for multi-classification features
library(nnet)
library(caret)
library(dplyr)

# Find and remove highly correlated variables to see effect on linear models

set.seed(7)
cutoff <- 0.70
correlations <- cor(numeric_features)
highlyCorrelated <- findCorrelation(correlations, cutoff=cutoff)
for (value in highlyCorrelated) {
  print(names(numeric_features)[value])
}

#Highly correlated features -> to be removed later for training/testing models
#[1] "SalePrice"
```

```
#[1] "AllFloorTotalSF"
#[1] "GrLivArea"
#[1] "OverallQual"
#[1] "X1stFlrSF"
#[1] "YearBuilt"


numeric_cols_logit <- names(which(sapply(grouped_df, is.numeric)))
numeric_df_logit <- grouped_df[, names(grouped_df) %in% numeric_cols_logit]


dummy_model <- grouped_df %>% mutate(HouseQuality =
                    case_when(OverallCond >= 1 & OverallCond <= 3 ~ "Poor",
                              OverallCond >= 4 & OverallCond <= 6 ~ "Average",
                              OverallCond >= 7 & OverallCond <= 10 ~ "Good"))

numeric_df_logit$Class <- dummy_model$HouseQuality

numeric_df_logit$Class <- as.factor(nc1$Class)
levels(numeric_df_logit$Class)


index <- createDataPartition(numeric_df_logit$Class, p = .70, list = FALSE)
train_logit <- numeric_df_logit[index,]
test_logit <- numeric_df_logit[-index,]

train_logit$Class <- relevel(train_logit$Class, ref = "Average")
str(train_logit)
```

```
# Training the multinomial model
multinom_model <- multinom(Class ~ ., data = train_logit)
# Checking the model
summary(multinom_model)

library("MASS")
install.packages("MASS")

exp(coef(multinom_model))

#This will show first 6 predicted values of house condition
head(round(fitted(multinom_model), 2))

#Average Good Poor
#1      1    0    0
#2      0    1    0
#3      1    0    0
#5      1    0    0
#7      1    0    0
#8      1    0    0


# Predicting the values for train_logit dataset
train_logit$ClassPredicted <- predict(multinom_model, newdata = train_logit, "class")

# Building classification table
tab <- table(train_logit$Class, train_logit$ClassPredicted)
```

```
# Calculating accuracy - sum of diagonal elements divided by total obs
round((sum(diag(tab))/sum(tab))*100,2)



# Predicting the class for test_logit dataset
test_logit$ClassPredicted <- predict(multinom_model, newdata = test_logit, "class")
# Building classification table
tab <- table(test_logit$Class, test_logit$ClassPredicted)
tab

#confusion Matrix of Prediction -> logistic regression

#Average Good Poor
#Average    339    0    0
#Good         1   88    0
#Poor         0    0    9

# confusion matrix
table(test_logit$Class, test_logit$Class)



##question 2 part B

install.packages('e1071')
library(e1071)

dataSVM <- numeric_df_logit

dataSVM$Class <- as.factor(dataSVM$Class)
```

```
levels(dataSVM$Class)

n <- nrow(dataSVM)  # Number of observations
ntrain <- round(n*0.75)  # 75% for training set
set.seed(314)    # Set seed for reproducible results
tindex <- sample(n, ntrain)   # Create a random index
train_svm <- dataSVM[tindex,]   # Create training set
test_svm <- dataSVM[-tindex,]   # Create test set
svm1 <- svm(Class~., data=train_svm,
          method="C-classification", kernal="radial",
          gamma=0.1, cost=10)

summary(svm1)
 # Number of Support Vectors:  402
 # ( 273 21 108 )
 #Number of Classes:  3

svm1$SV


prediction <- predict(svm1, test_svm)
xtab <- table(test_svm$Class, prediction)
xtab

#Confusion Matrix
#Average Good Poor
#Average     271    0    0
#Good         6   78    0
#Poor         4    0    6
```

```
(271 + 78    +6)/nrow(test_svm)  # Compute prediction accuracy
 # 0.9726027  accuracy obtained

 svm1$fitted # Results



#-------------------------question 3-----------------------------#

#Question 3

#Creating indices

numeric_cols_q3 <- names(which(sapply(grouped_df, is.numeric)))
numeric_features_df <- grouped_df[, names(grouped_df) %in% numeric_cols_q3]

training_sample <- createDataPartition(numeric_features_df$SalePrice ,p=0.70, list = FALSE)

#Partitioning data into 70% and 30%
df_train <- numeric_features_df[training_sample,] #training data 70
df_test <- numeric_features_df[-training_sample, ] #test data 30



##Method 1   Complete linear regression Analysis



#Fitting a model to predict values

   set.seed(2018)
```

```
split <- sample(seq_len(nrow(grouped_df)), size = floor(0.70 * nrow(grouped_df)))

train2 <- grouped_df[split, ]

test2 <- grouped_df[-split, ]

dim(train2)



train2 <- subset(train2, select=c(SalePrice,GarageArea, LotArea,FullBath, Neighborhood, OverallQual, TotRmsAbvGrd, KitchenAbvGr, GrLivArea, BedroomAbvGr, YearBuilt, OverallCond))

head(train2)


### Fit the linear model


fit7 <- lm(SalePrice ~ log(LotArea) + log(GarageArea+1) + OverallQual + FullBath + Neighborhood + TotRmsAbvGrd + KitchenAbvGr + GrLivArea + BedroomAbvGr + YearBuilt + OverallCond, data=train2)

summary(fit7)


fit7 <- lm(log(SalePrice) ~ log(LotArea) + log(GarageArea+1) + OverallQual + FullBath + Neighborhood + TotRmsAbvGrd + KitchenAbvGr + GrLivArea + BedroomAbvGr + YearBuilt + OverallCond, data=train2)

summary(fit7)



test2 <- subset(test2, select=c(SalePrice,GarageArea, LotArea,FullBath, Neighborhood, OverallQual, TotRmsAbvGrd, KitchenAbvGr, GrLivArea, BedroomAbvGr, YearBuilt, OverallCond))

prediction <- predict(fit7, newdata = test2)


head(prediction)


head(test2$SalePrice)
```

```
sumOfSquaredError <- sum((test2$SalePrice - prediction) ^ 2)
SumOfSquaredTotal <- sum((test2$SalePrice - mean(test2$SalePrice)) ^ 2)
1 - sumOfSquaredError/SumOfSquaredTotal

sumOfSquaredError
SumOfSquaredTotal

#5       7       8       9      10      11
#270318.8 229809.2 249908.9 148457.9  76962.5 111200.8

#Residual standard error: 0.1516 on 986 degrees of freedom
#Multiple R-squared:  0.862,      Adjusted R-squared:  0.8573
#F-statistic: 181.2 on 34 and 986 DF,  p-value: < 0.00000000000000022
```

```
# Working on different models to get the best model

model <- lm(formula = SalePrice ~ LotArea + OverallQual + OverallCond + TotalBsmtSF + GrLivArea +
              FullBath + BedroomAbvGr + KitchenAbvGr + GarageArea, data = numeric_features_df, subset = training_sample)
summary(model)

#model 1  -> very high standard error and it is reduced in 6th model fit6 where Residual standard error: 0.162 is quite less
fit1 <- lm(log(SalePrice) ~ log(LotFrontage) + log(LotArea) + log(GarageArea + 1) + log(GrLivArea) + YearBuilt + TotRmsAbvGrd + MasVnrArea  +
             OverallQual + log(TotalBsmtSF), data = numeric_features_df)
summary(fit1)
```

```
#Residual standard error: 39620 on 1014 degrees of freedom
#Multiple R-squared:  0.757,        Adjusted R-squared:  0.7548
#F-statistic:   351 on 9 and 1014 DF,  p-value: < 0.00000000000000022

#model 2
fit1 <- lm(SalePrice ~ LotFrontage + LotArea + GarageArea + GrLivArea + YearBuilt + TotRmsAbvGrd + MasVnrArea  +
           OverallQual + TotalBsmtSF, data = numeric_features_df)
summary(fit1)

#model3
fit2 <- lm(SalePrice ~  LotArea + GarageArea + GrLivArea + YearBuilt  + MasVnrArea  +
           OverallQual + TotalBsmtSF, data = numeric_features_df)
summary(fit2)

#model4
fit3 <- lm(log(SalePrice) ~  LotArea + GarageArea + GrLivArea + YearBuilt  + MasVnrArea  +
           OverallQual + TotalBsmtSF, data = numeric_features_df)
summary(fit3)

#model5
fit4 <- lm(log(SalePrice) ~  LotArea + GarageArea + GrLivArea + YearBuilt  +
           OverallQual + TotalBsmtSF, data = numeric_features_df)
summary(fit4)


#model6
fit5 <- lm(log(SalePrice) ~  LotArea + GarageArea + GrLivArea + YearBuilt  +
```

```
            OverallQual + TotalBsmtSF, data = numeric_features_df)
summary(fit5)
#Residual standard error: 0.1724 on 1453 degrees of freedom
#Multiple R-squared:  0.8145,    Adjusted R-squared:  0.8137
#F-statistic:  1063 on 6 and 1453 DF,  p-value: < 0.00000000000000022


#model7
init_fit <- lm(log(SalePrice) ~  log(LotArea) + log(GarageArea+1) +
            YearBuilt + TotRmsAbvGrd + FullBath + factor(Neighborhood) +
            OverallQual + factor(ExterQual) + factor(BsmtQual), data = grouped_df)
summary(init_fit)
#Residual standard error: 0.162 on 1423 degrees of freedom  (Here the standard error is less compare to other models)
#Multiple R-squared:  0.8395,    Adjusted R-squared:  0.8355
#F-statistic: 206.8 on 36 and 1423 DF,  p-value: < 0.00000000000000022


#model8
fit6 <- lm(log(SalePrice) ~  log(LotArea) + log(GarageArea+1) +
            YearBuilt + TotRmsAbvGrd + FullBath + factor(Neighborhood) +
            OverallQual + factor(ExterQual) + factor(BsmtQual), data = grouped_df)
summary(fit6)


#Here we are getting high accuracy and small residul standard error
#Residual standard error: 0.162 on 1423 degrees of freedom
#Multiple R-squared:  0.8395,    Adjusted R-squared:  0.8355
#F-statistic: 206.8 on 36 and 1423 DF,  p-value: < 0.00000000000000022


install.packages('MASS')
```

```
library(MASS)


stepAIC_lm <- stepAIC(init_fit, direction="backward", k=2, trace=FALSE)

stepAIC_lm$anova


stepBIC_lm <- stepAIC(init_fit, direction="backward", k=log(dim(grouped_df)[1]), trace=FALSE)

stepBIC_lm$anova


step__p_lm <- stepAIC(init_fit, direction="backward", k=qchisq(0.05, 1, lower.tail = F), trace=FALSE)

step__p_lm$anova


#Note  1) from the above, it is clear that BIC is little better than AIC, it can be noticed that by removing ExterQual have some effect on model AIC gets better

   # 2) From the above Linear models, we achieve F-Statistic: 222.8, Residual standard error: 0.1628 and P-value is quite significant.

   # Adjusted R-squared:  0.8338


 #best Model

 #model8

  fit6 <- lm(log(SalePrice) ~  log(LotArea) + log(GarageArea+1) +

        YearBuilt + TotRmsAbvGrd + FullBath + factor(Neighborhood) +

        OverallQual + factor(ExterQual) + factor(BsmtQual), data = grouped_df)

 summary(fit6)



plot(fit6$residuals, pch = 16, col = "blue")



# Method 2
```

```
#Using Random forest
install.packages('randomForest')
library(randomForest)
forest_df <- randomForest(SalePrice ~ LotArea + OverallQual + OverallCond + GrLivArea +
                          FullBath + BedroomAbvGr + KitchenAbvGr + GarageArea , data=train2)


#Using the predict features, we will test our model
pred.house <- predict(forest_df, test2)


#Testing our prediction and prediction results
test_pred <- data.frame(test2$SalePrice, pred.house, residuals = test2$SalePrice - pred.house)
head(test_pred, 30)


#test2.SalePrice pred.house  residuals
#7          307000   258590.1  48409.925
#13         144000   127942.2  16057.767
#15         157000   147572.4   9427.612
#18          90000   110798.1 -20798.059
#23         230000   250193.6 -20193.633
#28         306000   291588.2  14411.824


#Calculating the root square mean error (RSME)
#RMSE(pred.house, test2$SalePrice)      RMSE:  34107.52


#re-sampling methods
library(ipred)
```

```
# cross validation based on random forest


mypredict.randomForest <- function(object, newdata)
  predict(object, newdata = newdata, type = "response")


errorest(formula = SalePrice ~ LotArea + OverallQual + OverallCond + GrLivArea +
           FullBath + BedroomAbvGr + KitchenAbvGr + GarageArea ,model=randomForest,data = train2,
         estimator = "cv", predict= mypredict.randomForest)


#10-fold cross-validation estimator of root mean squared error
#Root mean squared error:  32236.2


# bootstraping
errorest(formula = SalePrice ~ LotArea + OverallQual + OverallCond + GrLivArea +
           FullBath + BedroomAbvGr + KitchenAbvGr + GarageArea,model=randomForest,data = train2,
         estimator = "boot", est.para=control.errorest(nboot = 25), predict= mypredict.randomForest)


#Bootstrap estimator of root mean squared error
#with 25 bootstrap replications


#Root mean squared error:  33792.15
```

```
#--------------------------------Question 4 ---------------------------------

## install relevant packages before access to the function and data

install.packages("HSAUR2") # install package
install.packages("ISLR")
library(HSAUR2)   #load package
library(ISLR)   #load package

install.packages("xtable")  #install package before use it
library(xtable)           # load package


## conduct PCA
## two methods: prcomp() - The calculation is done by a singular value decomposition of the (centered and possibly scaled) data matrix, not by using eigen on the covariance matrix.
## princomp()- The calculation is done using eigen on the correlation or covariance matrix, as determined by cor
# cutoff- loadings smaller than this (in absolute value) are suppressed, default value 0.1.

library(caret)

missing_glimpse(numeric_features) # missing data for each variable


numeric_features.pca <- princomp(numeric_features, cor = TRUE)
# cor=TRUE indicates correlation matrix is used.

summary_df <- summary(numeric_features.pca, loadings = TRUE,cutoff=0)
```

```
summary_df$sdev

K <- 4;
x.price_prediction <- numeric_features[,c(1:12)];
km.price_prediction <- kmeans(x.price_prediction,centers=K,nstart=20, iter=20)

km.price_prediction$cluster

plot(x.price_prediction[km.price_prediction$cluster==2,1:2],pch="x")

plot(km.price_prediction$centers[1,],type="l",col="red",ylab="")
lines(km.price_prediction$centers[2,],type="l", col="green",ylab="")

cor(x.price_prediction$LotArea, x.price_prediction$OverallCond)
str(km.price_prediction)

install.packages('factoextra')
library(factoextra)

fviz_cluster(km.price_prediction, data = numeric_features)

#-----------------------

## Hierichical clustering example


# Finding distance matrix
distance_mat <- dist(numeric_features[1:12], method = 'euclidean')
```

```
distance_mat

# Fitting Hierarchical clustering Model
# to training dataset
set.seed(240)  # Setting seed
Hierar_cl <- hclust(distance_mat, method = "average")
Hierar_cl

# Plotting dendrogram
plot(Hierar_cl)

# Choosing no. of clusters
# Cutting tree by height
abline(h = 110, col = "green")

# Cutting tree by no. of clusters
fit <- cutree(Hierar_cl, k = 3 )
fit

table(fit)
rect.hclust(Hierar_cl, k = 3, border = "green")

#hclust(d = distance_mat, method = "average")

#Cluster method   : average
#Distance         : euclidean
#Number of objects: 1460
```

```
install.packages("tidyverse")
install.packages("cluster")
install.packages("dendextend")
library(tidyverse)
library(cluster)
library(dendextend)

clusters <- hclust(dist(numeric_features[3:4]))
plot(clusters)

# Dissimilarity matrix
d <- dist(numeric_features, method = "euclidean")

# Hierarchical clustering using Complete Linkage
hc1 <- hclust(d, method = "complete" )

# Plot the obtained dendrogram
plot(hc1, cex = 0.6, hang = -1)

# Compute with agnes
hc2 <- agnes(numeric_features, method = "complete")

# Agglomerative coefficient
hc2$ac


# methods to assess
m <- c( "average", "single", "complete", "ward")
names(m) <- c( "average", "single", "complete", "ward")
```

```
# function to compute coefficient
ac <- function(x) {
  agnes(numeric_features, method = x)$ac
}

map_dbl(m, ac)
#average    single  complete      ward
#0.9967656 0.9864195 0.9973119 0.9994193

#It can be seen that all methods show strong hierarchical structures

hc3 <- agnes(numeric_features, method = "ward")
pltree(hc3, cex = 0.6, hang = -1, main = "Dendrogram of agnes")


##compute divisive hierarchical clustering
hc4 <- diana(numeric_features)

# Divise coefficient; amount of clustering structure found
hc4$dc
## [1]  0.9969774

# plot dendrogram
pltree(hc4, cex = 0.6, hang = -1, main = "Dendrogram of diana")


# Compute distance matrix
res.dist <- dist(numeric_features, method = "euclidean")

# Compute 2 hierarchical clusterings
```

```
hc1 <- hclust(res.dist, method = "complete")
hc2 <- hclust(res.dist, method = "ward.D2")

# Create two dendrograms
dend1 <- as.dendrogram (hc1)
dend2 <- as.dendrogram (hc2)

tanglegram(dend1, dend2)

#--------------------------------Question 4 ----------------------------------
```

#References:

#Classwork/labs code snippets

#http://www.sthda.com/english/articles/36-classification-methods-essentials/147-multinomial-logistic-regression-essentials-in-r/

#https://bookdown.org/chua/ber642_advanced_regression/multinomial-logistic-regression.html

#https://www.r-bloggers.com/2020/05/multinomial-logistic-regression-with-r/

#https://www.learnbymarketing.com/tutorials/k-means-clustering-in-r-example/

#https://odsc.medium.com/build-a-multi-class-support-vector-machine-in-r-abcdd4b7dab6

#https://www.rpubs.com/prakharprasad/511734

#https://www.r-bloggers.com/2016/01/hierarchical-clustering-in-r-2/

#Google