|  |
| --- |
| **MA321- APPLIED STATISTICS**    **(Department of Mathematical Sciences)**    **20th MARCH 2023**        **Group Project**  **Data Analysis of Housing Market**  ***Word Count: 2548***    **Submitted by:**  ` 2202227  2202486  2200964 |

***Contents***

**1.0 Abstract**…………………………………………………………………………….**3**

# 2.0 Introduction………………………………………………………………………..3 3.0 Task 1……………………………………………………………………………….3

*3.1 Numerical Summary………………………………………………………………………………****3***

*3.2 Graphical Summary* ***……………………………………………………………………...4***

# 4.0 Task2………………………………………………………………………………..5 *4.1 Preliminary Analysis* ..............................................................................................5 *4.2 Logistic Regression* .................................................................................................6 *4.3 Random Forest* ........................................................................................................7 5.0 Task3…………………………………………………………………………………8 *5.1 Multiple Linear Regression* ……………………………………………..................8

*5.2 Random Forest Regression***…..……………………………………………………..9**

*5.3 Resampling methods****……………………………………………………………….10***

**6.0 Task4 ……………………………………………………………………………….10** *6.1 Clusters***…………………………………………………………………………….10 7.0 Conclusion .................................................................................................................11 8.0 References ..................................................................................................................11 Appendix 1 ...................................................................................................................... 12**

**Figures ............................................................................................................................13 Appendix 2 .......................................................................................................................17**

# Contribution Table

|  |  |
| --- | --- |
| Question 1 | 2202227,2202486,2200964 |
| Question 2 | 2202227, 2200964 |
| Question 3 | 2202486, 2202227 |
| Question 4 | 2202486, 2200964 |
| Report | 2202227,2202486,2200964 |

## 1.0 Abstract

The housing data consists of multiple features that impact the response variables of the overall condition and sale price. After conducting an analysis of the data, we were able to draw several conclusions. We used classification techniques to predict the overall condition and to forecast the sale price. Additionally, we employed clustering analysis to gain further insights into the data.

# 2.0 Introduction

The primary goal of this task is to analyse the "house-data.csv". Firstly, the dataset is cleaned to handle null values and ensure that there are no irrelevant data when fitting it into machine learning model. The 'missForest()' function imputes the missing values & assigns values to each missing value using either the mean or mode, then models are fitted to predict the overall condition of houses. Then, factors and labels are used so that there are three different classes corresponding to the overall condition (OverallCond), which are Poor (1-3), Average (4-6) and Good. (7-10). Logistic regression and Random Forest are used to classify the house condition, and both are analysed to see which model better suits the problem. Furthermore, multiple linear regression and random forest methods predict housing prices. Comparing these two models uses mean squared error to see which model gives us a better prediction. Similarly, the bootstrapping resampling method estimates the test error for the multiple linear regression and random forest methods. Finally, by employing a clustering analysis, the project considers if it would be a good idea to know the number of clusters between the variables OverallCond and OverallQual.

## 3.0 Task1

*3.1 Numerical Summary* -It is observed from the dataset that there are 51 attributes and 1460 rows. The following table shows the numerical summaries of all the numerical columns:

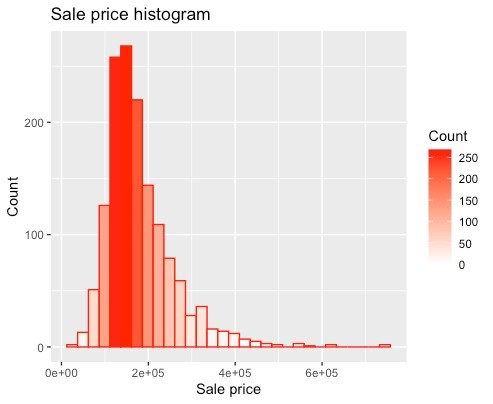
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Column  Name | Min  Value | 1st  Quartile | Median | Mean | 3rd  Quartile | Max  Value | Null  Values |
| LotFrontage | **21** | **59** | **69** | **70.05** | **80** | **313** | **259** |
| LotArea | **1300** | **7554** | **9458** | **10517** | **11602** | **215254** | **0** |
| OverallQual | **1** | **5** | **5** | **6.099** | **7** | **10** | **0** |
| OverallCond | **1** | **5** | **6** | **5.575** | **6** | **9** | **0** |
| MasVnrArea | **0** | **0** | **0** | **103.7** | **166** | **1600** | **8** |
| TotalBsmtSF | **0** | **795.8** | **991.5** | **1057.4** | **1298.2** | **6110** | **0** |
| X1stFlrSF | **334** | **882** | **1087** | **1163** | **1391** | **4692** | **0** |
| X2ndFlrSF | **0** | **0** | **0** | **347** | **728** | **2065** | **0** |
| SalePrice | **34900** | **129975** | **163000** | **180921** | **214000** | **755000** | **0** |

***Table 1: Numerical Summaries***

*3.2 Graphical Summary -* Created different types of plots to visualize the data. The plots include histograms and boxplots for different variables, and the data is segmented by different categorical variables to study their impact on the sale price of the houses and figure out whether the dataset has any outliners.

There are four categorical columns which have more than 75% of NA values (Figure 1.1 Appendix 1). Alley, PoolQC, Fence & MiscFeature*.* The missing values indicate that many of the houses do not have alley access, no pool, no fence and no elevator, 2nd garage, shed or tennis court that is covered by the MiscFeature.

* The utility column has only one value, 'AllPub' and no null values, henceforth, this variable is eliminated from the dataset (Figure 1.2 Appendix 1).
* Similarly, the LotConfig column has no null values perhaps it consists of 5 different categories, with category 'Inside' having the highest number of values. (Figure 1.3 Appendix 1)
* In contrast, the 'Neighborhood' column has the greatest number of categories among all other columns, and the Names category has the highest number of values, whereas the Blueste category has lowest number of values (Figure 1.4 Appendix 1).
* On one hand, another column with a highest number of categories is Condition1, where Norm is around 1500, where other columns are less than 100. Both the above columns Neighborhood, Conditional 1 have 0 null values (Figure 1.5 Appendix 1).
* The prices of most houses range in between 0 and 200000. Having the maximum sale price of 755000 while the minimum sale price of 34900. Hence, it can be inferenced from figure1 that the number/count of houses decrease with the increase in price after the price value of 200000.

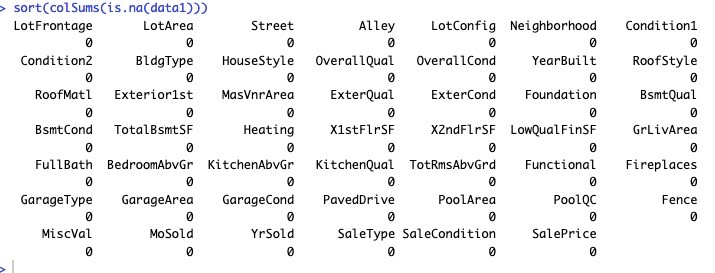
. 

## Figure 1: Distribution of Sale Price

**Condition 2 vs SalePrice (fig 1.6) -** There are five varieties of buildings. Most preferred building are of single family detached(1Farm) and least used are of two-family conversion (2fmCon). *There are outliers for all types of except Twnhs.* **Neighborhood vs Sale price (fig 1.7) -** 13 types of neighbourhoods are present which determine in pricing. Houses with Stone Brook (StoneBr) are priced high and North Park villa (NPKVill) are the least priced ones. *There are outliers for almost all columns.* **Distribution of LotFrontage vs Sale Price (fig 1.8)-** This is a numerical column and most of the houses are in the range of 0 to 250000. *Many outliers are observed for this feature.* **House Style vs Sale Price (fig 1.9) -** Eight types of house styles are present where houses with 2 storey (2 story) have highest range of prices while split foyer (SFoyer) style being the least priced. *There are outliers for all features except 1.5Unf.*

# 4.0 Task 2 –

*4.1 Preliminary Analysis* **-** To get accurate predictions for the data, it is significant that we ensure that the dataset has no missing values or outliers. Therefore, the first step taken in the project is cleaning the data and making sure it is suitable to fit a model. Some unnecessary columns, such as ID and Utilities are removed from the dataset as they do not affect the data. The Utilities column values correspond to the same value as *'AllPub'*. Therefore, the whole column has been removed. Next up, it is vital to deal with the missing values. As the 'house.csv' dataset has a mixed datatype. Therefore, *'missForest'* is implemented so that we can impute the missing values. Several iterations of training and predicting the values occur and NRMSE and PFC's imputation error is calculated. The value of NRMSE is 1.044832e-07 which suggests the imputation model has very high accuracy in predicting missing values in the dataset. Whereas the PFC value is 6.602006e-02, which is approximately 6.6% thereby the imputation model was correctly predicted & there is only a small fraction of the missing values present in the dataset. Finally, we look for missing values in any of the dataset's columns, which are none, as it can be seen in fig.2 :

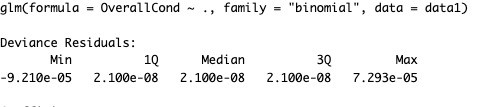


## Figure 2: Result of Imputation Process

Now that the data is clean, we have divided the categories for the 'OverallCond' column into three variables: Poor, Average, and Good. We will first analyse trying to fit a logistic regression model which predicts the overall condition.

## data1$OverallCond <- factor (data1$OverallCond, levels = 1:10, labels = c('Poor', 'Poor', 'Poor', 'Average', 'Average', 'Average', 'Good', 'Good', 'Good', 'Good'))

*4.2 Logistic Regression* **–** Logistic regression involves modelling the probability of a discrete outcome when given an input variable. Almost every problem solved by logistic regression models represents a binary outcome, which means that it can take two values, true and false.

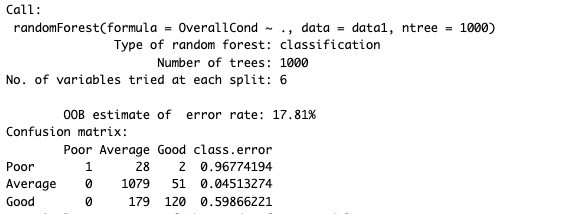


|  |  |
| --- | --- |
| Null deviance | 3.0017e+02 on 1459 degrees of freedom |
| Residual deviance | 1.5662e-07 on 1283 degrees of freedom |
| AIC | 354 |
| Number of Fisher Scoring iterations | 25 |

### Table 2: Summary of Logistic Regression

The output suggests that the logistic regression model is a good fit for the data, with a very low residual deviance and an acceptable AIC value. The dispersion parameter for the binomial family is assumed to be 1, which is the default value. The first and third quartiles are also very small, indicating that most of the residuals are close to zero which suggests that the model is a good fit for the data.

*4.3 Random Forest***:** In random forest, cross-validation or any other tests are not required as it is estimated internally, known as bag (OOB) error estimate. The OOB cases are noted in all the trees grown in the forest, and the number of votes cast for each correct class is counted. Furthermore, all these cases are listed on the tree. Then, the values of variable m in the OOB cases are permuted in a random state. Finally, the number of votes for each correct class in the variable-m-permuted OOB data is subtracted from the number of votes cast for the proper class in the election new OOB data. The raw significance score for variable m is the average of this value over all trees in the forest. When the random Forest classifier is fit into the house data, here are the results:

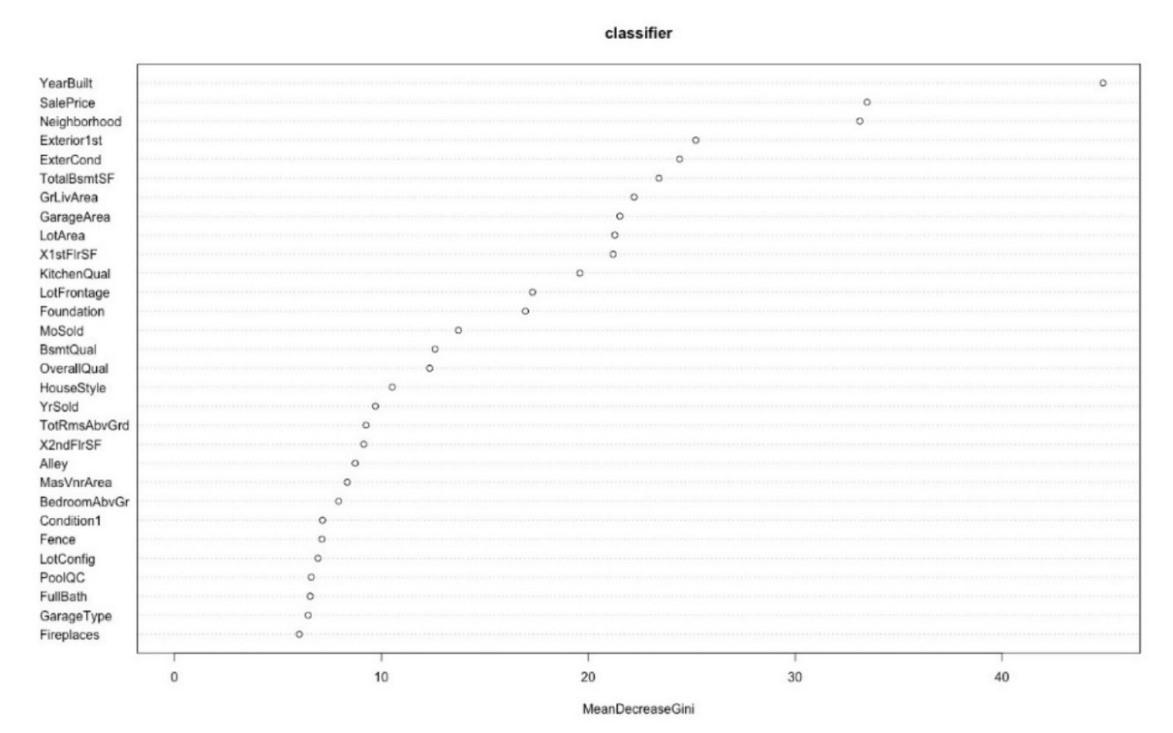


### Figure 3: Result of Random Forest Classifier

As seen in figure 3, the OOB estimate of error rate was 18.22%, and the class error can be seen on the confusion matrix above. Similarly, when plotting the importance, as seen in the figure 4, here are the observations:

1. Year Built affected the condition of the house most significantly with just below value of 50 of Mean Decrease Gini whereas the Functional column had the most negligible impact on the condition of the house with the Mean Decrease Gini value of around 5 to

7.



### Figure 4: Random Forest

1. The other variables that affected the house's condition most after the Year Built were Neighborhood and Sale Price, with the value for Mean Decrease Gini above 30 but less than 40.
2. X1stFirSF, Garage Area, Lot Area, GrLivArea, TotalBsmtSF, ExterCond, Exterior1st all affected the house mildly, with the value for Mean Decrease Gini between 20 and

30.

1. TotRmsAbvGrd, X2ndFirSF, YrSold, HouseStyle, QverallQual, BsmtQual, MoSold, Foundation, LotFrontage, KitchenQual affected the condition of the house only slightly, with the value of Mean Decrease Gini between 10 and 20.
2. Finally, Functional, Fireplaces, ExterQual, LotConfig, Condition1, Garage Type, FullBath, BedroomAbvGr, Fence, MasVnrArea all variables had significantly less impact on the condition of the house in comparison with others as the value of Mean Decrease Gini was between 5 and 10.

# 5.0 Task 3

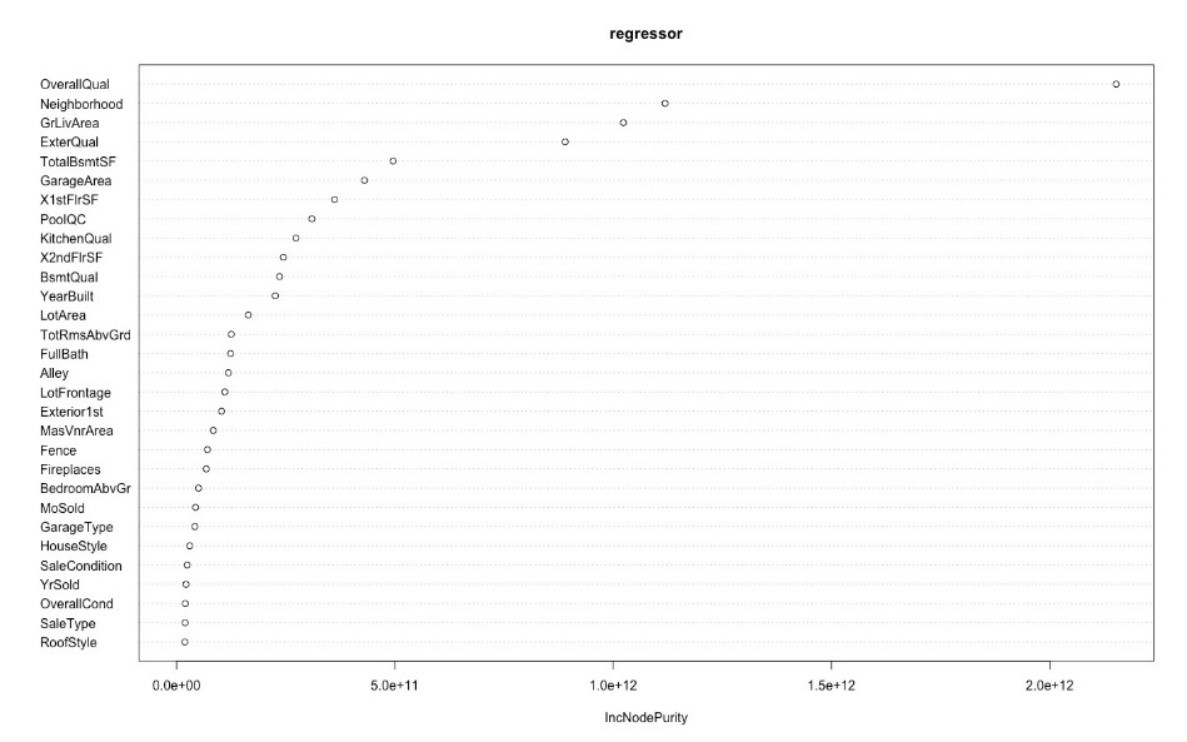
*5.1 Multiple Linear Regression-*

The residual standard deviation (or residual standard error) is a metric for evaluating the fit of a linear regression model to data. With the data that we extracted from the overall performance of the Multiple linear regression. we note that the Residual standard error is a high number. Therefore, this model is does not fit well to this dataset. However, the R-Squared value of 0.9 indicates that the variation of the independent variable explains 90% of the variance of the dependent variable under investigation. The dispersion of data points around the fitted regression line is calculated using R-squared. For multiple regression, it's also known as the coefficient of determination or the coefficient of multiple determination. Higher R-squared values indicate less discrepancies between the observed data and the fitted values for the same data set.

|  |  |
| --- | --- |
| Residual standard error | 24670 on 1282 degrees of freedom |
| Multiple R-squared | 0.9153 |
| F-statistic | 78.26 on 177 and 1282 DF |
| p - value | < 2.2e-16 |

***Table 3:*** Linear Regression Performance

*5.2 Random Forest Regression* **–** Random Forest regression gives us two outputs: MSE and node purity. Usually, MSE is a more reliable measure of variable importance. Prediction



***Figure 5***: Node Purity

error, described as MSE, is based on permuting out-of-bag sections of the data per individual tree and predictor, and the errors are then averaged. Similarly, node purity is the total decrease in the residual sum of squares when splitting on a variable averaged over all trees. In other words, it means how well a predictor decreases variance.

Figure 5 shows the node purity in terms of the variables. When making comparisons between random forest regression and multiple linear regression in terms of the MSE, it can be observed that the mean squared error for random forest regression is much lower than the mean squared error for the multiple linear regression, as shown in Table 4.



|  |  |
| --- | --- |
| MSE\_rf | MSE\_lm |
| 13872892 | 534594895 |

## *Table 4: Model Comparison Results*

Therefore, random forest regression is a better model than multiple linear regression.

*5.3 Resampling methods –* Train-test resampling:

Resampling for multiple linear regression:

* Residual standard error: 24650 on 990 degrees of freedom
* Multiple R-squared: 0.9243, Adjusted R-squared: 0.9107
* F-statistic: 68.26 on 177 and 990 DF, p-value: < 2.2e-16
* MSE: 734418981

Resampling for random forest regression:

* Number of trees: 500
* No. of variables tried at each split: 15
* Mean of squared residuals: 8451111502
* % Var explained: 87.63
* MSE: 594124234

Bootstrapping:

Resampling for multiple linear regression:

* MSE: 492441114

Resampling for random forest regression:

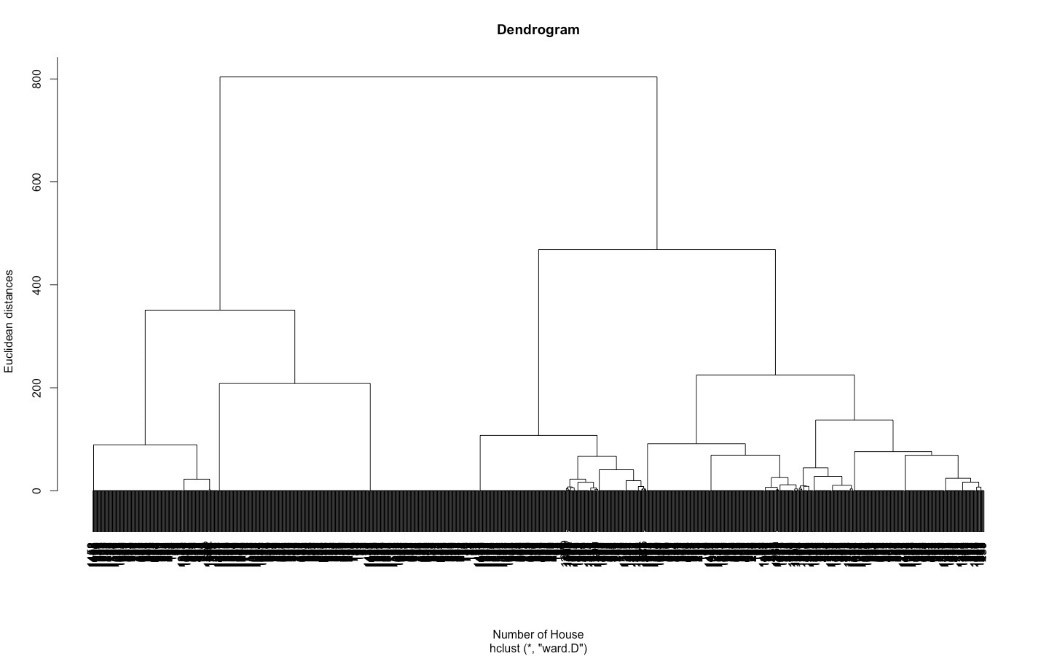
* MSE: 141817007

# 6.0 Task 4

*6.1 Clusters -*

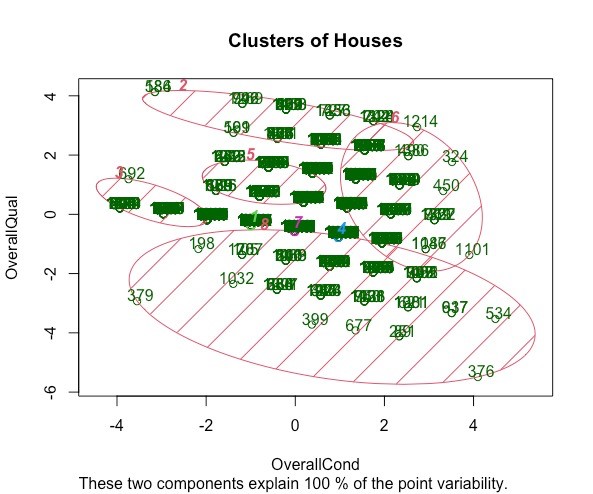
There are several exciting investigations we can infer using the 'house-data’.csv. However, the most interesting one we found out were the number of clusters between the variables

‘OverallCond’ and ‘OverallQual’. We use a dendrogram to find the optimal number of clusters as a clustering analysis methodology. The data points are represented on the horizontal axis. The distance between clusters is represented by the vertical axis' height. As a result, the graph's vertical lines depict clusters. The distance between these clusters is represented by the height of these lines.



## *Figure 6: Dendrogram*

The number of clusters that best depict the data is determined by the vertical lines with the longest distances between them, i.e. the highest height on the same level. The number of clusters in this case is four, whereas the number of clusters in the highest level is seven. Figure 7 shows the visual representation of the clusters. It can be observed that both the components explain 100% of the variability.



***Figure 7: Cluster Analysis***

# 7.0 Conclusion

Examining the results of the study as mentioned earlier and comparing the mean squared error of all approaches. We can conclude that the random forest regression is the best model to fit the data and predict the house prices most accurately. In addition, random forest is also a better classification method than logistic regression to classify into three categories: Poor, Average or Good. This is mainly because logistic regression requires binary outcomes, either True/False, Yes or No, whereas our dataset has three outcomes: Poor, Average or Good.

Looking at the methods used for resampling, it was clear that bootstrapping resampling method came ahead of train-test resampling. The bootstrap resampling method gave MSE for the random forest model 141386127 compared to the train-testing resampling method, which was 590988851. Similarly, for the multiple linear regression model, the MSE for bootstrapping was 457094207, while the train-testing resampling method produced an MSE of 724357411, which was far more than the bootstrapping method. Hence, the bootstrapping method produced far less MSE error than the train-testing method.

# 8.0 References -

[1]. Logistic Regression - an overview - https://www.sciencedirect.com/topics/computerscience/logistic-regression

[2]. Random Forest Approach for Regression in R Programming - **https://www.geeksforgeeks.org/random-forest-approach-for-regression-in-rprogramming/**

[3]. Random Forests - https://uc-r.github.io/random\_forests

[4]. Bonaccorso, Giuseppe - Mastering Machine Learning Algorithms Expert techniques for implementing popular machine learning algorithms, fine-tuning your models, and understanding how they work (2020).

***Appendix 1:***

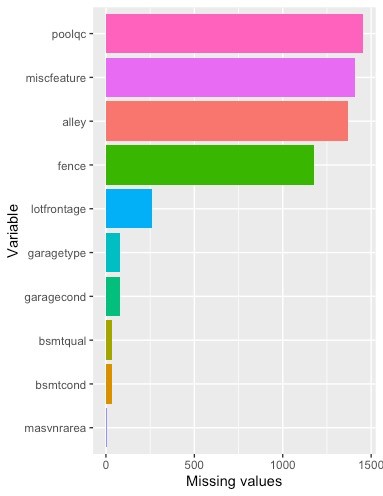


Figure 1.1 Missing values

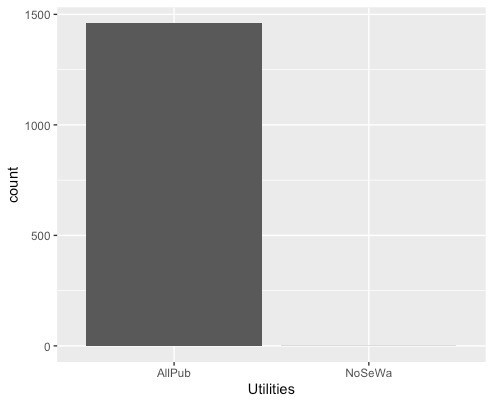


Figure 1.2 Utilities Distribution

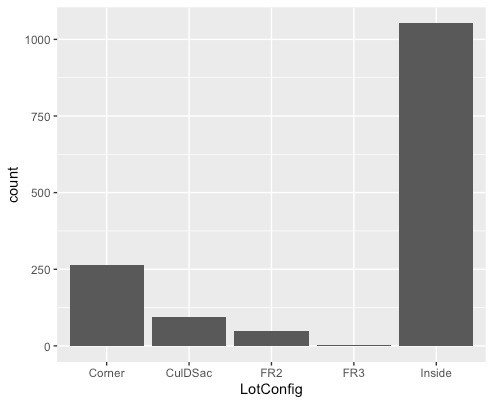


Figure 1.3 : LotConfiguration Distribution

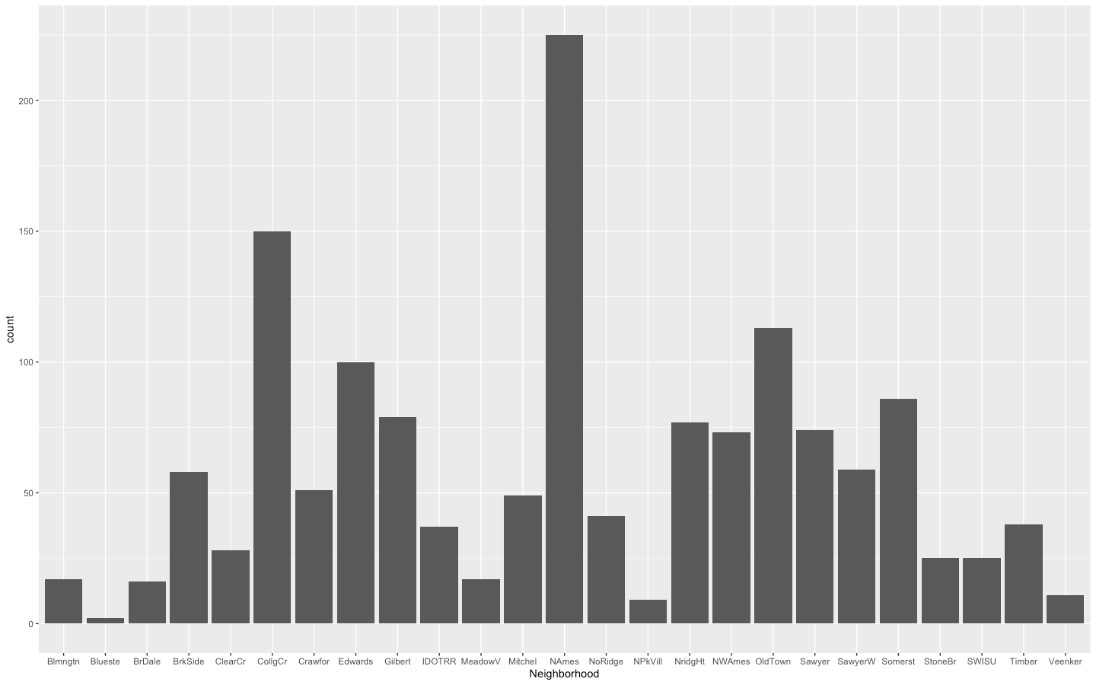


Figure 1.4 Neighbourhood Distribution

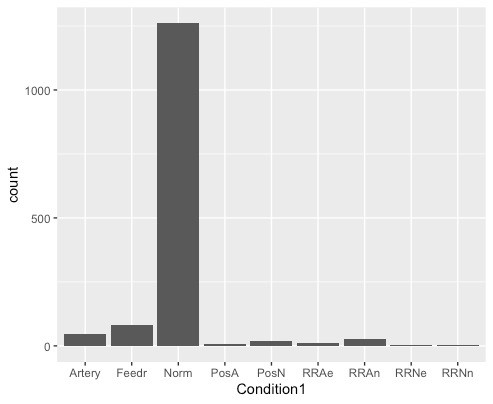


Figure 1.5 Condition1

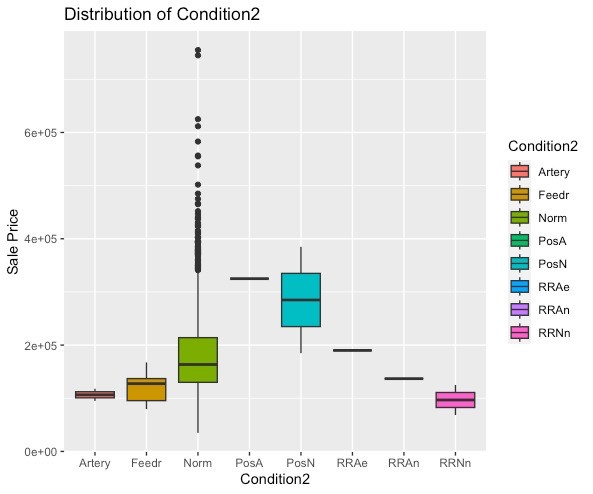


Figure 1.6 Condition2 Vs Sale Price

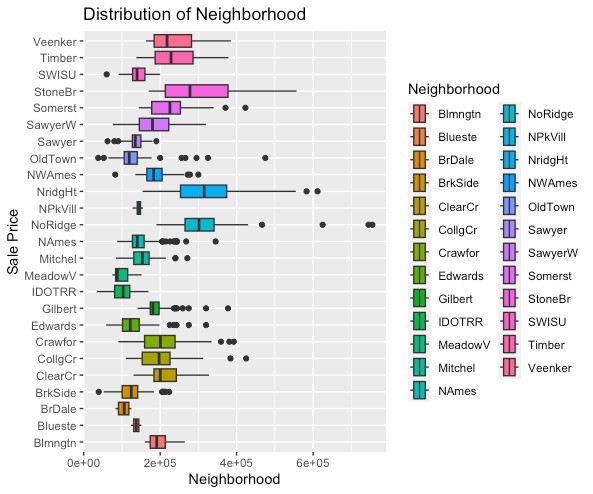


Figure 1.7 Neighborhood Vs Sale Price

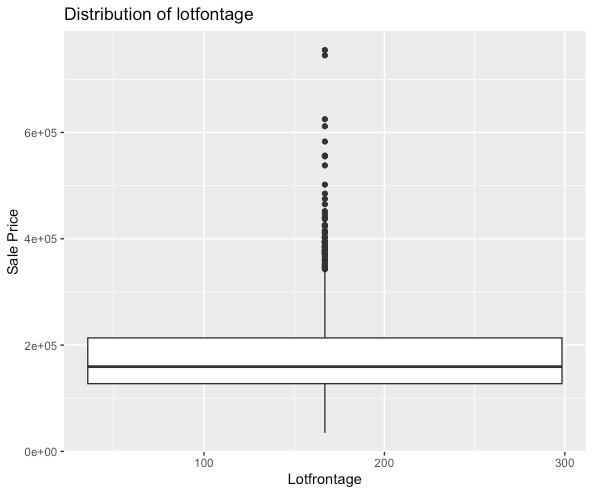


Figure 1.8 Lotfrontage Vs Sale Price

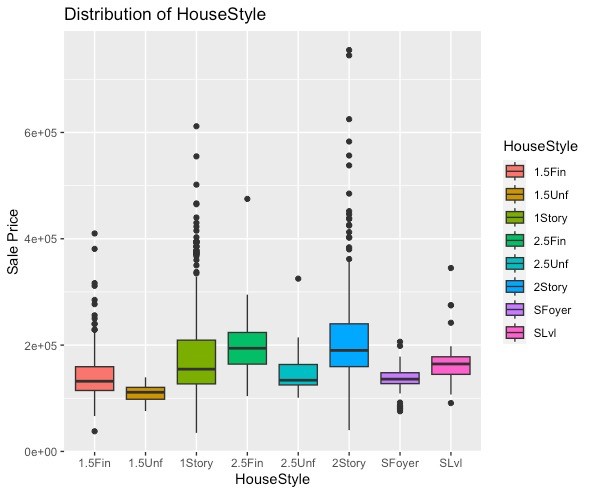


Figure 1.9 HouseStyle Vs Sale Price

***Appendix 2:***

***#Loading required packages*** library('ggplot2') library('tidyverse') library('scales') library('dplyr') library('caret') library('MASS') library('randomForest') ***#Load the dataset***

*setwd('/Users/ankitrajsingh/Desktop/MA321 group') data<-read.csv('house-data.csv')*

***#Task 1: Summary statistics and visualizations***

***# Numerical Summary*** *summary(data)*

*num\_data<-names(which(sapply(data,is.numeric)))# check the data types of variables char\_data<-names(which(sapply(data,is.character)))# check the number of levels of categorical variables*

***#Graphical summaries*** *hist(data$LotArea) hist(data$YearBuilt) hist(data$OverallQual)*

***# Histogram of numerical variable 'SalePrice' with a gradient fill*** *ggplot(data, aes(SalePrice)) + geom\_histogram(col="red", aes(fill=..count..)) + scale\_fill\_gradient("Count", low="white", high="red") + labs(title = "Sale price histogram", x = "Sale price", y = "Count") summary(data$SalePrice)* ***# Bar plots*** *ggplot(data, aes(x = LotConfig)) + geom\_bar() ggplot(data, aes(x = Neighborhood)) + geom\_bar() ggplot(data, aes(x = Utilities)) + geom\_bar() ggplot(data, aes(x = Condition1)) + geom\_bar() ggplot(data, aes(x = Condition2)) + geom\_bar()*

***# Box plots of 'SalePrice' grouped by various categorical variables*** *ggplot(data, aes(y= SalePrice, x=HouseStyle, fill=HouseStyle) ) + geom\_boxplot() + ggtitle("Distribution of HouseStyle") + ylab("Sale Price") + xlab("HouseStyle")*

*ggplot(data, aes(y= SalePrice, x=RoofMatl, fill=RoofMatl ) ) + geom\_boxplot() + ggtitle("Distribution of RoofMatl") + ylab("Sale Price") + xlab("RoofMatl")*

*ggplot(data, aes(x= SalePrice, y=Heating, fill=Heating ) ) + geom\_boxplot() + ggtitle("Distribution of Heating") + ylab("Sale Price") + xlab("Heating")*

*ggplot(data, aes(y= SalePrice, x=Condition1, fill=Condition1) ) + geom\_boxplot() +*

*ggtitle("Distribution of Condition1") + ylab("Sale Price") + xlab("Condition1")*

*ggplot(data, aes(y= SalePrice, x=Condition2, fill=Condition2) ) + geom\_boxplot() +*

*ggtitle("Distribution of Condition2") + ylab("Sale Price") + xlab("Condition2")*

*ggplot(data, aes(y= SalePrice, x=RoofStyle, fill=RoofStyle) ) + geom\_boxplot() +*

*ggtitle("Distribution of RoofStyle") + ylab("Sale Price") + xlab("RoofStyle")*

*ggplot(data, aes(y= SalePrice, x=ExterQual, fill=ExterQual ) ) + geom\_boxplot() +*

*ggtitle("Distribution of ExterQual") + ylab("Sale Price") + xlab("ExterQual")*

*ggplot(data, aes(y= SalePrice, x=BldgType, fill=BldgType) ) + geom\_boxplot() + ggtitle("Distribution of BldgType") + ylab("Sale Price") + xlab("BldgType")*

*ggplot(data, aes(y= SalePrice, x=ExterCond, fill=ExterCond) ) + geom\_boxplot() + ggtitle("Distribution of ExterCond") + ylab("Sale Price") + xlab("ExterCond")*

*ggplot(data, aes(y= SalePrice, x=Foundation, fill=Foundation) ) + geom\_boxplot() + ggtitle("Distribution of Foundation") + ylab("Sale Price") + xlab("Foundation")*

*ggplot(data, aes(y= Neighborhood, x=SalePrice, fill=Neighborhood) ) + geom\_boxplot() + ggtitle("Distribution of Neighborhood") + ylab("Sale Price") + xlab("Neighborhood")*

*ggplot(data, aes(y= SalePrice, x=LotConfig, fill=LotConfig) ) + geom\_boxplot() +*

*ggtitle("Distribution of LotConfig") + ylab("Sale Price") + xlab("LotConfig")*

*ggplot(data, aes(y= SalePrice, x=Street, fill=Street) ) + geom\_boxplot() + ggtitle("Distribution of Street") + ylab("Sale Price") + xlab("Street")*

*ggplot(data, aes(y= SalePrice, x=PavedDrive, fill=PavedDrive) ) + geom\_boxplot() + ggtitle("Distribution of PavedDrive") + ylab("Sale Price") + xlab("PavedDrive")*

*ggplot(data, aes(y= SalePrice, x=SaleCondition, fill=SaleCondition) ) + geom\_boxplot() + ggtitle("Distribution of SaleCondition") + ylab("Sale Price") + xlab("SaleCondition")*

***################################################################***

***# storing data into data1*** *data1<-data*

***#Task 2: Divide houses based on their overall condition as follows:***

***#Poor if the overall condition is between 1 to 3.***

***#Average if the overall condition is between 4 and 6 #Good if the overall condition is between 7 and 10*** *data1$OverallCond <- factor(data1$OverallCond, levels = 1:10,*

*labels = c('Poor', 'Poor', 'Poor', 'Average', 'Average', 'Average',*

*'Good', 'Good', 'Good', 'Good'))*

*str(data1)# looking at the structure of the data data1 = data1[-c(1,6,45)] # removing the unnecessary variables* ***# separating the categorical and numerical variables*** *num\_vec\_indx = c(1,2, 11, 13, 17, 23, 25:28, 33, 37, 40, 43:45)*

***# transformation of the numerical variables***

*# min-max and log transformation at a time addition with 5*

*log\_t = function(y) log((y - min(y, na.rm = T)) / (max(y, na.rm = T) - min(y, na.rm = T)) + 5)*

*data1[num\_vec\_indx] =lapply(data1[num\_vec\_indx], log\_t) num\_vec\_indx = c(num\_vec\_indx, 48)*

***# making the categorical variables as factor***

*data1[-num\_vec\_indx] = as.data.frame(lapply(data1[-num\_vec\_indx], factor))*

***# to treat the missing value*** *library(missForest) data1 <- missForest(data1) # applying the missForest function to impute missing values data1$OOBerror # to see the imputation error*

*data1 = data1$ximp # to extract the data only*

*# checking the number of missing values sort(colSums(is.na(data1)))*

***# Task -2 (a) Fit the logistic regression model***

*model <- glm(OverallCond ~ ., data = data1, family = "binomial")*

*# View the summary of the model summary(model)*

***# (b) Carry out a similar study using a different classification method to classify*** *install.packages('randomForest', dependencies = TRUE) library(randomForest)*

*set.seed(13) # for setting the randomness fixed for all the runs classifier <- randomForest(OverallCond ~ ., data = data1, ntree = 1000) # Display the random forest model and its goodness-of-fit measures classifier*

*# Display a summary of the random forest model summary(classifier) y\_pred = predict(classifier) # to predict by using the model importance(classifier) # to know the importance of the independent variables varImpPlot(classifier)*

***############################################################## # Task 3: Predicting house prices: # fitting multiple linear regression*** *regressor =lm(SalePrice ~ ., data = data1) regressor # to show the regressor*

*# To test of hypothesis and see the summary of the model and to know the goodness of fit of the model summary(regressor)*

*# predicting the test set results y\_pred = predict(regressor, newdata = data1)*

*MSE\_lm = mean(resid(regressor)^2) # Mean sum of squares of errors*

***# random forest*** *set.seed(120) regressor = randomForest(SalePrice ~ ., data =data1, ntree = 500) regressor summary(regressor) importance(regressor) # to know the importances of the independent variables varImpPlot(regressor) # to plot the importances*

***# predicting the test set results***

*y\_pred = predict (regressor, newdata = data1) # sum squares of errors*

*MSE\_rf = mean((data1$SalePrice - y\_pred)^2)*

*# to compare mean sum of squares of errors from different models*

*MSE = data.frame(MSE\_lm, MSE\_rf)*

*MSE\_sorted <- sort(MSE) # sort the MSE data frame by increasing order*

*MSE\_sorted # print the sorted MSE data frame*

***# Task 3(b)***

***# train-test resampling for multiple regression model*** *set.seed(137) indx = sample(1:nrow(data1), round(0.8 \* nrow(data1))) train\_set = data1[indx,] test\_set = data1[-indx,]*

***# fitting multiple linear regression*** *regressor = lm(SalePrice ~ ., data = train\_set) regressor # to show the regressor*

***# to test of hypothesis and see the summary of the model*** *summary(regressor)*

***# predicting the test set results*** *y\_pred = predict(regressor, newdata = test\_set) MSE\_lm\_tt = mean((test\_set$SalePrice - y\_pred)^2) # sum squares of errors*

***# bootstrap resampling for multiple regression model***

*MSE\_lm\_boot = 2^1000*

*# setting a very large value*

*# loop for bootstrap*

*# Run a loop 5 times to perform bootstrapping and linear regression modeling for (i in 1:5) {*

*# Randomly select 90% of the rows in data2 indx <- sample(1:nrow(data1), round(0.9 \* nrow(data1))) # Create a new data frame with the selected rows data2 <- data1[indx,]*

*# Fit a linear regression model to predict SalePrice using all other variables in data3 regressor <- lm(SalePrice ~ ., data = data2)*

*# Use the fitted model to predict SalePrice for the rows in data3 y\_pred <- predict(regressor, newdata = data2)*

*# Calculate the mean squared error between the predicted and actual SalePrice in data3*

*MSE\_lm <- mean((data2$SalePrice - y\_pred)^2)*

*# Update the "best" linear regression model and its corresponding MSE if the current model has a lower MSE if (MSE\_lm < MSE\_lm\_boot) { regressor\_best <- regressor*

*MSE\_lm\_boot <- MSE\_lm*

*}*

*}*

*# train-test resampling for random forest model set.seed(137)*

*indx = sample(1:nrow(data1), round(0.8 \*nrow(data1))) train\_set = data1[indx,] test\_set = data1[-indx,]*

*# fitting multiple linear regression*

*regressor = randomForest(SalePrice ~ ., data = train\_set, ntree = 500) regressor # to show the regressor*

*# to test of hypothesis and see the summary of the model summary(regressor)*

*# predicting the test set results*

*y\_pred = predict(regressor, newdata = test\_set)*

*MSE\_rf\_tt = mean((test\_set$SalePrice - y\_pred)^2) # sum squares of errors*

*# bootstrap resampling for random forest model*

*MSE\_rf\_boot = 2^1000 # setting a very large value*

*# loop for bootstrap*

*# Run a loop 5 times to perform bootstrapping and random forest modeling*

*for (i in 1:5) {*

*# Randomly select 90% of the rows in data2 indx <- sample(1:nrow(data1), round(0.9 \* nrow(data1))) # Create a new data frame with the selected rows data2 <- data1[indx,]*

*# Fit a random forest model to predict SalePrice using all other variables in data3 regressor <- randomForest(SalePrice ~ ., data = data2, ntree = 500) # Use the fitted model to predict SalePrice for the rows in data3 y\_pred <- predict(regressor, newdata = data2)*

*# Calculate the mean squared error between the predicted and actual SalePrice in data3*

*MSE\_lm <- mean((data2$SalePrice - y\_pred)^2)*

*# Update the "best" random forest model and its corresponding MSE if the current model has a lower MSE if (MSE\_lm < MSE\_rf\_boot) { regressor\_best <- regressor*

*MSE\_rf\_boot <- MSE\_lm*

*}*

*}*

*# to compare mean sum of squares of errors from different models*

*MSE = data.frame(MSE\_lm, MSE\_rf, MSE\_lm\_tt, MSE\_rf\_tt, MSE\_lm\_boot, MSE\_rf\_boot) sort(MSE) # to the model having lowest MSE*

***# Task 4***

***#we may consider to know the number of clusters between the variables OverallCond and OverallQual.***

***# taking the required data only*** *data2 = data.frame(data$OverallCond, data$OverallQual)* ***# using the dendrogram to find the optimal number of clusters*** *dendrogram = hclust(d = dist(data2, method = 'euclidean'), method = 'ward.D') plot(dendrogram,main ='Dendrogram', xlab= 'Number of House',ylab = 'Euclidean distances')*

*# fitting hierarchical clustering to the data3*

*hc = hclust(d = dist(data2, method = 'euclidean'), method = 'ward.D') y\_hc = cutree(hc, 8) # visualizing the clusters install.packages('cluster',dep = T) library(cluster) clusplot(data2, y\_hc, lines = 0, shade = TRUE, color = TRUE, labels =*

*2, plotchar = FALSE, span = TRUE, main = 'Clusters of Houses', xlab = 'OverallCond', ylab = 'OverallQual')*