- **Christella Nissanthan**

- **Final Project_Ma 3904**
- **11:59 May 24, 2020**

Modeling and Forecasting the Housing price in Queens, NY

Abstract

In this paper, we will model and predict the sales price using three different methods, which are supervised techniques used in the case of the quantitative dependent variable. In fact, many exogenous factors impact the housing sale price, and every single method brings a different insight into the housing database. For this purpose, we will start by the most important step, namely, the wrangling and cleansing of the dataset. By next, and after feature selection, we will train our models and make a comparison in terms of prediction errors.

Keywords: Modeling, forecasting, housing price, R, RMarkDown, Regression, Random Forest, Decision Tree.

1. Introduction

Predictive modeling is widely used by researchers and scientists to solve a plethora of economic, social, and health problems. A large variety of techniques have been developed and implemented. The choice of the right predictive modeling method is a meticulous work that implies the comparison and benchmarking of many models.

Theoretically, a predictive model is written as a linear or non-linear mathematical formula that aims to explain and predict a variable of interest, which is independent/endogenous. The inputs are called explanatory or exogeneous variables and help to better understand the variable of interest and ensure a great quality of fit.

Selecting the right variable to include in a predictive model can be very difficult and especially decisive of the quality of the model. For structural predictive models, such as linear regression, it's possible to quantify the relationship between the dependent and independent variables, which allows the model's high interpretability and results.

In this paper, we will use three predictive modeling techniques to predict the sale price of housings in Queens, NY. There is an important number of explicative variables that can be included in the analysis, characteristics of the house, financial indicators, and social aspects.

We will try to include all the relevant factors to minimize the prediction error and keep certain interpretability in the results. Among the used techniques, random forest that was proposed by Leo Breiman and Adèle Cutler, 2001. In its most classical formula, it performs parallel learning on multiple decision trees constructed randomly and trained on subsets of data differently. The ideal number of trees, which can go up to several hundred or more, is an important parameter: it is very variable and depends on the problem.

The linear regression is a structural model that links linearly the dependent variable and the explicative ones and aims to minimize the root squared mean error between the fitted and actual values. The regression allows interpreting the link between the variable in term of the elasticity.

A regression tree is a classic method in machine learning. It aims to predict the value of a target variable from the value of several input variables. One of the input variables is selected at each internal node (or internal, node which is not terminal) of the tree according to a method that depends on the algorithm and which will be discussed later. Each edge to a child node corresponds to a set of values of an input variable so that the set of edges to the child nodes covers all the possible values of the input variable.

2. Data

In this section, we will introduce our database used to predict the housing sale price in Queens, NY.

The dataset is found on GitHub, housing_data_2016_2017.csv, where the outcome (dependent variable) to be predicted is the column named sale_price. The data contains a plethora of exogenous quantitative and qualitative explicative variables that impact the sale price.

The raw database contains 2230 observations and 55 variables; not all the variables would be selected for the analysis because some of the variables are just informative, and others contain a lot of missing values, and that be irrelevant to include in the modeling part.

The database needs great wrangling and cleansing to avoid any computational problems and also any bias in model estimates. By next, we will tidy the data and fix the problems among variables and observations.

We will move to the next section, where we will provide more details and descriptive statistics about the data.

3. Data exploration and cleansing

The first step is to provide the descriptive and exploratory summary (Like median, mean, standard deviation, number of missing values.) of the continuous variables:

```
setwd("/Users/christellanissanthan/Desktop/Final_Project")
house_data<-read.csv("housing_data_2016_2017.csv", sep=",", dec=".", header=T, stringsAsFactors = F)
library(pastecs)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:pastecs':
##
##     first, last

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
summary(house_data %>% select_if(is.numeric))
```

```
##  MaxAssignments AssignmentDurationInSeconds AutoApprovalDelayInSeconds
##  Min.   :1      Min.   :900                 Min.   :60
##  1st Qu.:1      1st Qu.:900                 1st Qu.:60
##  Median :1      Median :900                 Median :60
##  Mean   :1      Mean   :900                 Mean   :60
##  3rd Qu.:1      3rd Qu.:900                 3rd Qu.:60
##  Max.   :1      Max.   :900                 Max.   :60
##  NA's   :758    NA's   :758                 NA's   :758
```

```
##  WorkTimeInSeconds approx_year_built community_district_num  num_bedrooms
##  Min.   : 22.0     Min.   :1893      Min.   : 3.00          Min.   :0.000
##  1st Qu.: 89.0     1st Qu.:1950      1st Qu.:25.00          1st Qu.:1.000
##  Median :127.0     Median :1958      Median :26.00          Median :2.000
##  Mean   :162.4     Mean   :1963      Mean   :26.33          Mean   :1.653
##  3rd Qu.:197.0     3rd Qu.:1970      3rd Qu.:28.00          3rd Qu.:2.000
##  Max.   :815.0     Max.   :2017      Max.   :32.00          Max.   :6.000
##  NA's   :758       NA's   :40        NA's   :19             NA's   :115
##  num_floors_in_building num_full_bathrooms num_half_bathrooms num_total_rooms
##  Min.   : 1.000         Min.   :1.000      Min.   :0.0000     Min.   : 0.000
##  1st Qu.: 3.000         1st Qu.:1.000      1st Qu.:1.0000     1st Qu.: 3.000
##  Median : 6.000         Median :1.000      Median :1.0000     Median : 4.000
##  Mean   : 7.785         Mean   :1.231      Mean   :0.9535     Mean   : 4.139
##  3rd Qu.: 7.000         3rd Qu.:1.000      3rd Qu.:1.0000     3rd Qu.: 5.000
##  Max.   :34.000         Max.   :3.000      Max.   :2.0000     Max.   :14.000
##  NA's   :650                               NA's   :2058       NA's   :2
##  pct_tax_deductibl   sq_footage       walk_score
##  Min.   :20.0      Min.   : 100.0   Min.   : 7.00
##  1st Qu.:40.0      1st Qu.: 743.0   1st Qu.:77.00
##  Median :50.0      Median : 881.0   Median :89.00
##  Mean   :45.4      Mean   : 955.4   Mean   :83.92
##  3rd Qu.:50.0      3rd Qu.:1100.0   3rd Qu.:95.00
##  Max.   :75.0      Max.   :6215.0   Max.   :99.00
##  NA's   :1754      NA's   :1210
```

As we can remark, many columns contain missing values with a high percent like MaxAssignmentS, AssignmentDurationInSeconds, AutoApprovalDelayInSeconds, WorkTimeInSeconds... These variables should not be included in the model identification or selection because it may bias the estimation.

Now, we will present the character variable in our dataset, using the str() function:

```
str(house_data %>% select_if(is.character))
```

```
## 'data.frame':    2230 obs. of  36 variables:
##  $ HITId                       : chr  "3OID399FXG7F26JWONXF0Y86J90FD4" "3MQY1YVHS3K2MF90MWR2LPQH7KJ
##  $ HITTypeId                   : chr  "36BILMLQB75QQNBTYKGYCZWDN8TVAU" "36BILMLQB75QQNBTYKGYCZWDN8T
##  $ Title                       : chr  "Find Information about Housing To Help a Student Project -- 
##  $ Description                 : chr  "Go to a link and copy information into the HIT" "Go to a lin
##  $ Reward                      : chr  "$0.05 " "$0.05 " "$0.05 " "$0.05 " ...
##  $ CreationTime                : chr  "Wed Feb 15 22:13:37 PST 2017" "Wed Feb 15 22:13:37 PST 2017"
##  $ RequesterAnnotation         : chr  "BatchId:2689947;OriginalHitTemplateId:920937336;" "BatchId:2
##  $ Expiration                  : chr  "Wed Feb 22 22:13:37 PST 2017" "Wed Feb 22 22:13:37 PST 2017"
##  $ AssignmentId                : chr  "32KTQ2V7RDFCSAWQOW1SXC5AZIC9MB" "35LDD5557A4W96FHSTSHNLJQAB7
##  $ WorkerId                    : chr  "A231MNJJDDF3LS" "A394B5QVCVKU7A" "A231MNJJDDF3LS" "AHXBZXWIZ
##  $ AssignmentStatus            : chr  "Approved" "Approved" "Approved" "Approved" ...
##  $ AcceptTime                  : chr  "Thu Feb 16 05:32:36 PST 2017" "Wed Feb 15 22:19:51 PST 2017"
##  $ SubmitTime                  : chr  "Thu Feb 16 05:35:37 PST 2017" "Wed Feb 15 22:21:52 PST 2017"
##  $ AutoApprovalTime            : chr  "Thu Feb 16 05:36:37 PST 2017" "Wed Feb 15 22:22:52 PST 2017"
##  $ ApprovalTime                : chr  "2017-02-16 13:37:11 UTC" "2017-02-16 06:23:11 UTC" "2017-02-
##  $ LifetimeApprovalRate        : chr  "100% (187/187)" "100% (8/8)" "100% (187/187)" "100% (115/115
##  $ Last30DaysApprovalRate      : chr  "100% (187/187)" "100% (8/8)" "100% (187/187)" "100% (115/115
##  $ Last7DaysApprovalRate       : chr  "100% (187/187)" "100% (8/8)" "100% (187/187)" "100% (103/103
##  $ URL                         : chr  "http://www.mlsli.com/homes-for-sale/address-not-available-fr
##  $ cats_allowed                : chr  "no" "no" "no" "no" ...
```

```
##  $ common_charges               : chr  "$767 " NA "$167 " "$275 " ...
##  $ coop_condo                   : chr  "co-op" "co-op" "condo" "condo" ...
##  $ date_of_sale                 : chr  "2/16/2016" "2/16/2016" "2/17/2016" "2/17/2016" ...
##  $ dining_room_type             : chr  "combo" "formal" "combo" "combo" ...
##  $ dogs_allowed                 : chr  "no" "no" "no" "no" ...
##  $ fuel_type                    : chr  "gas" "oil" NA "gas" ...
##  $ full_address_or_zip_code     : chr  "Flushing NY, 11355" "30-11 Parsons Blvd,  Flushing NY, 11354
##  $ garage_exists                : chr  NA NA NA NA ...
##  $ kitchen_type                 : chr  "eat in" "eat in" "efficiency" "eat in" ...
##  $ maintenance_cost             : chr  NA "$604 " NA NA ...
##  $ model_type                   : chr  "Mitchell Garden 3" "Jr-4 Model" "Apt In Bldg" "144-48 Roosev
##  $ parking_charges              : chr  NA NA NA NA ...
##  $ sale_price                   : chr  "$228,000 " "$235,500 " "$137,550 " "$545,000 " ...
##  $ total_taxes                  : chr  NA NA "$5,500 " "$2,260 " ...
##  $ listing_price_to_nearest_1000: chr  NA NA NA NA ...
##  $ url                          : chr  NA NA NA NA ...
```

```
str(house_data)
```

```
## 'data.frame':    2230 obs. of  55 variables:
##  $ HITId                        : chr  "3OID399FXG7F26JWONXF0Y86J90FD4" "3MQY1YVHS3K2MF90MWR2LPQH7KJ
##  $ HITTypeId                    : chr  "36BILMLQB75QQNBTYKGYCZWDN8TVAU" "36BILMLQB75QQNBTYKGYCZWDN8T
##  $ Title                        : chr  "Find Information about Housing To Help a Student Project -- 
##  $ Description                  : chr  "Go to a link and copy information into the HIT" "Go to a lin
##  $ Keywords                     : logi  NA NA NA NA NA NA ...
##  $ Reward                       : chr  "$0.05 " "$0.05 " "$0.05 " "$0.05 " ...
##  $ CreationTime                 : chr  "Wed Feb 15 22:13:37 PST 2017" "Wed Feb 15 22:13:37 PST 2017"
##  $ MaxAssignments               : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ RequesterAnnotation          : chr  "BatchId:2689947;OriginalHitTemplateId:920937336;" "BatchId:2
##  $ AssignmentDurationInSeconds  : int  900 900 900 900 900 900 900 900 900 900 ...
##  $ AutoApprovalDelayInSeconds   : int  60 60 60 60 60 60 60 60 60 60 ...
##  $ Expiration                   : chr  "Wed Feb 22 22:13:37 PST 2017" "Wed Feb 22 22:13:37 PST 2017"
##  $ NumberOfSimilarHITs          : logi  NA NA NA NA NA NA ...
##  $ LifetimeInSeconds            : logi  NA NA NA NA NA NA ...
##  $ AssignmentId                 : chr  "32KTQ2V7RDFCSAWQOW1SXC5AZIC9MB" "35LDD5557A4W96FHSTSHNLJQAB7
##  $ WorkerId                     : chr  "A231MNJJDDF3LS" "A394B5QVCVKU7A" "A231MNJJDDF3LS" "AHXBZXWIZ
##  $ AssignmentStatus             : chr  "Approved" "Approved" "Approved" "Approved" ...
##  $ AcceptTime                   : chr  "Thu Feb 16 05:32:36 PST 2017" "Wed Feb 15 22:19:51 PST 2017"
##  $ SubmitTime                   : chr  "Thu Feb 16 05:35:37 PST 2017" "Wed Feb 15 22:21:52 PST 2017"
##  $ AutoApprovalTime             : chr  "Thu Feb 16 05:36:37 PST 2017" "Wed Feb 15 22:22:52 PST 2017"
##  $ ApprovalTime                 : chr  "2017-02-16 13:37:11 UTC" "2017-02-16 06:23:11 UTC" "2017-02-
##  $ RejectionTime                : logi  NA NA NA NA NA NA ...
##  $ RequesterFeedback            : logi  NA NA NA NA NA NA ...
##  $ WorkTimeInSeconds            : int  181 121 120 160 136 249 85 132 198 130 ...
##  $ LifetimeApprovalRate         : chr  "100% (187/187)" "100% (8/8)" "100% (187/187)" "100% (115/115
##  $ Last30DaysApprovalRate       : chr  "100% (187/187)" "100% (8/8)" "100% (187/187)" "100% (115/115
##  $ Last7DaysApprovalRate        : chr  "100% (187/187)" "100% (8/8)" "100% (187/187)" "100% (103/103
##  $ URL                          : chr  "http://www.mlsli.com/homes-for-sale/address-not-available-fr
##  $ approx_year_built            : int  1955 1955 2004 2002 1949 1938 1950 1960 1960 2005 ...
##  $ cats_allowed                 : chr  "no" "no" "no" "no" ...
##  $ common_charges               : chr  "$767 " NA "$167 " "$275 " ...
##  $ community_district_num       : int  25 25 24 25 26 28 29 28 25 30 ...
##  $ coop_condo                   : chr  "co-op" "co-op" "condo" "condo" ...
##  $ date_of_sale                 : chr  "2/16/2016" "2/16/2016" "2/17/2016" "2/17/2016" ...
```

```
##  $ dining_room_type             : chr  "combo" "formal" "combo" "combo" ...
##  $ dogs_allowed                 : chr  "no" "no" "no" "no" ...
##  $ fuel_type                    : chr  "gas" "oil" NA "gas" ...
##  $ full_address_or_zip_code     : chr  "Flushing NY, 11355" "30-11 Parsons Blvd,  Flushing NY, 11354
##  $ garage_exists                : chr  NA NA NA NA ...
##  $ kitchen_type                 : chr  "eat in" "eat in" "efficiency" "eat in" ...
##  $ maintenance_cost             : chr  NA "$604 " NA NA ...
##  $ model_type                   : chr  "Mitchell Garden 3" "Jr-4 Model" "Apt In Bldg" "144-48 Roosev
##  $ num_bedrooms                 : int  2 1 1 3 2 2 1 0 1 1 ...
##  $ num_floors_in_building       : int  6 7 1 NA 2 6 NA 2 NA 4 ...
##  $ num_full_bathrooms           : int  1 1 1 2 1 1 1 1 1 1 ...
##  $ num_half_bathrooms           : int  NA NA NA NA NA NA NA NA NA NA ...
##  $ num_total_rooms              : int  5 4 3 5 4 4 3 2 4 3 ...
##  $ parking_charges              : chr  NA NA NA NA ...
##  $ pct_tax_deductibl            : int  NA NA NA NA 39 NA NA NA NA NA ...
##  $ sale_price                   : chr  "$228,000 " "$235,500 " "$137,550 " "$545,000 " ...
##  $ sq_footage                   : int  NA 890 550 NA 675 1000 NA 375 NA 681 ...
##  $ total_taxes                  : chr  NA NA "$5,500 " "$2,260 " ...
##  $ walk_score                   : int  82 89 90 94 71 90 72 93 70 98 ...
##  $ listing_price_to_nearest_1000: chr  NA NA NA NA ...
##  $ url                          : chr  NA NA NA NA ...
```

The output shows 36 string variables. Many of these variables are actually numerical and need the be formatted to delete string characters on it. For example, the dependent variable "sale_price" is categorized as a character because it contains the $ symbol that must be deleted and formatted.

The main library we will use for this purpose is dplyr, which provides a plethora of functions that helps filtering, formatting, and selecting features.

The main features used to format the database: - Remove irrelevant strings form observations like $ symbol - Recategorize dummy and multinomial variables by correcting the factors - Add new variables as a combination of others - Change the type of variables

For this purpose, we create a new database "house_data_cleaned" that contains all this featurizations.

```
library(stringr)
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.0     v readr   1.3.1
## v tibble  3.0.1     v purrr   0.3.3
## v tidyr   1.0.2     v forcats 0.5.0

## -- Conflicts ------------------------------------------------------ tidyverse_conflicts() --
## x tidyr::extract() masks pastecs::extract()
## x dplyr::filter()  masks stats::filter()
## x dplyr::first()   masks pastecs::first()
## x dplyr::lag()     masks stats::lag()
## x dplyr::last()    masks pastecs::last()
```

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:dplyr':
##
##     intersect, setdiff, union

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
house_data_cleaned<-house_data%>%
#First, we must convert the sale_price and other variables to a numerical by removing the $ symbol
mutate(sale_price = as.numeric(gsub("[^0-9A-Za-z///' ]","" , sale_price))) %>%
mutate(total_taxes = as.numeric(gsub("[^0-9A-Za-z///' ]","" , total_taxes))) %>%
mutate(maintenance_cost = as.numeric(gsub("[^0-9A-Za-z///' ]","" , maintenance_cost))) %>%
mutate(Reward = as.numeric(gsub("[^0-9A-Za-z///' ]","" , Reward))) %>%
mutate(zip_code = str_extract(full_address_or_zip_code, "[0-9]{5}")) %>%
mutate(common_charges = as.numeric(gsub("[^0-9A-Za-z///' ]","" , common_charges))) %>%
mutate(listing_price_to_nearest_1000 = as.numeric(gsub("[^0-9A-Za-z///' ]","" , listing_price_to_neares
#The items of some variables should be recategorised-->
mutate(dogs_allowed = ifelse(substr(house_data$dogs_allowed, 1, 3) == "yes", 1, 0)) %>%
mutate(cats_allowed = ifelse(substr(house_data$cats_allowed, 1, 3) == "yes", 1, 0)) %>%
mutate(pets_allowed = ifelse( cats_allowed + dogs_allowed > 0, 1, 0)) %>%
mutate(coop_condo = factor(tolower(coop_condo))) %>%
mutate(fuel_type=ifelse(fuel_type==c("Other","other"), "other",fuel_type)) %>%
mutate(fuel_type=ifelse(fuel_type==c("Other","other"), "other",fuel_type)) %>%
mutate(kitchen_type=ifelse(kitchen_type==c("efficiemcy","efficiency","efficiency kitchen","efficiency k
mutate(kitchen_type=as.factor(kitchen_type)) %>%
mutate(kitchen_type=ifelse(kitchen_type=="0", "other",kitchen_type)) %>%
mutate(dining_room_type=ifelse(dining_room_type=="none","other", dining_room_type)) %>%
mutate(dining_room_type=ifelse(dining_room_type=="dining area","other",dining_room_type)) %>%
#Change the type of some variables
mutate(dining_room_type = as.factor(dining_room_type)) %>%
mutate(zip_code = as.numeric(zip_code)) %>%
mutate(garage_exists = as.character(garage_exists)) %>%
mutate(garage_exists = as.numeric(garage_exists)) %>%
mutate(parking_charges = as.numeric(parking_charges)) %>%
mutate(date_of_sale=as.Date(date_of_sale, format="%m/%d/%Y")) %>%
mutate(maintenance_cost_sq = maintenance_cost^2)%>%
mutate(month_sale=month(date_of_sale), year_sale=year(date_of_sale))   %>%
select(-c(HITId, HITTypeId, Title, Description, Keywords, Reward, CreationTime, MaxAssignments, Request
```

```
## Warning in kitchen_type == c("efficiemcy", "efficiency", "efficiency kitchen", :
## longer object length is not a multiple of shorter object length

## Warning in kitchen_type == c("eat in", "Eat in", "Eat In", "eatin"): longer
## object length is not a multiple of shorter object length

## Warning: NAs introduced by coercion

## Warning: NAs introduced by coercion
```

```
house_data_cleaned$listing_price_to_nearest_1000<-ifelse(house_data_cleaned$listing_price_to_nearest_100
#Now, we will replace all the remaining NAs with 0
house_data_cleaned[is.na(house_data_cleaned)]=0
```

```
## Warning in `[<-.factor`(`*tmp*`, thisvar, value = 0): invalid factor level, NA
## generated
```

4. Modeling

4.1. Data splitting

It's important to split the dataset into at least two sets, the training dataset and the test data test. The performance of the model is calculated in both sets. First, the model is trained using the training dataset and then it would be used in a test dataset to properly test the adequation of the model.

```
train_index = sample(1 : nrow(house_data_cleaned), nrow(house_data_cleaned)*80/100)
training_set = house_data_cleaned[train_index, ]
testing_set = house_data_cleaned[-train_index, ]
Both_sets = rbind(training_set, testing_set)
```

```
summary(training_set)
```

```
##  AssignmentStatus   WorkTimeInSeconds approx_year_built  cats_allowed
##  Length:1784        Min.   :  0.0     Min.   :   0      Min.   :0.00
##  Class :character   1st Qu.:  0.0     1st Qu.:1950      1st Qu.:0.00
##  Mode  :character   Median : 92.0     Median :1958      Median :0.00
##                     Mean   :109.6     Mean   :1933      Mean   :0.37
##                     3rd Qu.:156.0     3rd Qu.:1970      3rd Qu.:1.00
##                     Max.   :815.0     Max.   :2017      Max.   :1.00
##  common_charges   community_district_num coop_condo   dining_room_type
##  Min.   :   0.0   Min.   : 0.00          co-op:1327   combo :770
##  1st Qu.:   0.0   1st Qu.:25.00          condo: 457   formal:493
##  Median :   0.0   Median :26.00                       other :164
##  Mean   : 107.6   Mean   :26.09                       NA's  :357
##  3rd Qu.:   0.0   3rd Qu.:28.00
##  Max.   :2200.0   Max.   :32.00
##   dogs_allowed     kitchen_type   maintenance_cost  num_bedrooms
##  Min.   :0.0000   Min.   :0.000   Min.   :   0.0   Min.   :0.000
##  1st Qu.:0.0000   1st Qu.:3.000   1st Qu.:   0.0   1st Qu.:1.000
##  Median :0.0000   Median :3.000   Median : 655.0   Median :2.000
##  Mean   :0.2455   Mean   :2.677   Mean   : 615.6   Mean   :1.561
##  3rd Qu.:0.0000   3rd Qu.:3.000   3rd Qu.: 877.0   3rd Qu.:2.000
##  Max.   :1.0000   Max.   :3.000   Max.   :4659.0   Max.   :6.000
##  num_floors_in_building num_full_bathrooms num_half_bathrooms num_total_rooms
##  Min.   : 0.000         Min.   :1.000      Min.   :0.00000    Min.   : 0.000
##  1st Qu.: 0.000         1st Qu.:1.000      1st Qu.:0.00000    1st Qu.: 3.000
##  Median : 3.000         Median :1.000      Median :0.00000    Median : 4.000
##  Mean   : 5.511         Mean   :1.233      Mean   :0.07455    Mean   : 4.143
##  3rd Qu.: 6.000         3rd Qu.:1.000      3rd Qu.:0.00000    3rd Qu.: 5.000
##  Max.   :34.000         Max.   :3.000      Max.   :2.00000    Max.   :14.000
##  pct_tax_deductibl   sale_price       sq_footage      total_taxes
##  Min.   : 0.000    Min.   :     0   Min.   :   0.0   Min.   :   0.0
##  1st Qu.: 0.000    1st Qu.:     0   1st Qu.:   0.0   1st Qu.:   0.0
##  Median : 0.000    Median :     0   Median :   0.0   Median :   0.0
##  Mean   : 9.582    Mean   : 77946   Mean   : 440.5   Mean   : 596.2
##  3rd Qu.: 0.000    3rd Qu.:     0   3rd Qu.: 850.0   3rd Qu.:  30.0
```

```
##  Max.   :75.000    Max.   :999999   Max.   :6215.0   Max.   :9300.0
##    walk_score    listing_price_to_nearest_1000    zip_code      pets_allowed
##  Min.   : 7.00   Min.   :   0.0                Min.   :    0   Min.   :0.0000
##  1st Qu.:77.00   1st Qu.:  98.5                1st Qu.:11360   1st Qu.:0.0000
##  Median :89.00   Median : 264.5                Median :11372   Median :0.0000
##  Mean   :83.98   Mean   : 289.9                Mean   :11587   Mean   :0.3744
##  3rd Qu.:95.00   3rd Qu.: 439.0                3rd Qu.:11375   3rd Qu.:1.0000
##  Max.   :99.00   Max.   :1000.0                Max.   :27110   Max.   :1.0000
##  maintenance_cost_sq
##  Min.   :       0
##  1st Qu.:       0
##  Median :  429025
##  Mean   :  640129
##  3rd Qu.:  769129
##  Max.   :21706281
```

```
summary(testing_set)
```

```
##  AssignmentStatus   WorkTimeInSeconds approx_year_built  cats_allowed
##  Length:446         Min.   :  0.00    Min.   :   0      Min.   :0.0000
##  Class :character   1st Qu.:  0.00    1st Qu.:1950      1st Qu.:0.0000
##  Mode  :character   Median : 75.00    Median :1958      Median :0.0000
##                     Mean   : 97.42    Mean   :1906      Mean   :0.3722
##                     3rd Qu.:138.50    3rd Qu.:1969      3rd Qu.:1.0000
##                     Max.   :772.00    Max.   :2016      Max.   :1.0000
##  common_charges   community_district_num coop_condo  dining_room_type
##  Min.   :   0.0   Min.   : 0.00          co-op:334   combo :187
##  1st Qu.:   0.0   1st Qu.:25.00          condo:112   formal:127
##  Median :   0.0   Median :26.00                      other : 41
##  Mean   : 110.5   Mean   :26.15                      NA's  : 91
##  3rd Qu.:   0.0   3rd Qu.:28.00
##  Max.   :2499.0   Max.   :31.00
##   dogs_allowed     kitchen_type   maintenance_cost  num_bedrooms
##  Min.   :0.0000   Min.   :0.000   Min.   :   0.0   Min.   :0.000
##  1st Qu.:0.0000   1st Qu.:3.000   1st Qu.:   0.0   1st Qu.:1.000
##  Median :0.0000   Median :3.000   Median : 668.5   Median :2.000
##  Mean   :0.2422   Mean   :2.617   Mean   : 632.3   Mean   :1.596
##  3rd Qu.:0.0000   3rd Qu.:3.000   3rd Qu.: 891.0   3rd Qu.:2.000
##  Max.   :1.0000   Max.   :3.000   Max.   :2321.0   Max.   :4.000
##  num_floors_in_building num_full_bathrooms num_half_bathrooms num_total_rooms
##  Min.   : 0.000         Min.   :1.000      Min.   :0.00000    Min.   :0.000
##  1st Qu.: 0.000         1st Qu.:1.000      1st Qu.:0.00000    1st Qu.:3.000
##  Median : 3.000         Median :1.000      Median :0.00000    Median :4.000
##  Mean   : 5.538         Mean   :1.226      Mean   :0.06951    Mean   :4.103
##  3rd Qu.: 6.000         3rd Qu.:1.000      3rd Qu.:0.00000    3rd Qu.:5.000
##  Max.   :34.000         Max.   :3.000      Max.   :2.00000    Max.   :8.000
##  pct_tax_deductibl   sale_price       sq_footage      total_taxes
##  Min.   : 0.00     Min.   :     0   Min.   :   0.0   Min.   :   0
##  1st Qu.: 0.00     1st Qu.:     0   1st Qu.:   0.0   1st Qu.:   0
##  Median : 0.00     Median :     0   Median :   0.0   Median :   0
##  Mean   :10.12     Mean   : 61077   Mean   : 422.7   Mean   : 530
##  3rd Qu.: 0.00     3rd Qu.:     0   3rd Qu.: 850.0   3rd Qu.:  13
##  Max.   :57.00     Max.   :850000   Max.   :2200.0   Max.   :6533
##    walk_score    listing_price_to_nearest_1000    zip_code      pets_allowed
```

```
##  Min.   :23.00   Min.   :  0.0                Min.   :    0   Min.   :0.0000
##  1st Qu.:77.00   1st Qu.:146.8                1st Qu.:11360   1st Qu.:0.0000
##  Median :89.00   Median :286.0                Median :11372   Median :0.0000
##  Mean   :83.67   Mean   :306.9                Mean   :11743   Mean   :0.3812
##  3rd Qu.:95.00   3rd Qu.:466.8                3rd Qu.:11375   3rd Qu.:1.0000
##  Max.   :99.00   Max.   :999.0                Max.   :27110   Max.   :1.0000
##  maintenance_cost_sq
##  Min.   :      0
##  1st Qu.:      0
##  Median : 446894
##  Mean   : 633063
##  3rd Qu.: 793881
##  Max.   :5387041
```

4.2. Linear regression

We will start our modeling par with the linear regression, the most used modeling technique of continious variables.

```
linear_regression<-lm(sale_price~., training_set)
summary(linear_regression)
```

```
##
## Call:
## lm(formula = sale_price ~ ., data = training_set)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -385137  -50787   -7133   42520  478587
##
## Coefficients:
##                                Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  -1.726e+05  3.720e+04  -4.641 3.79e-06 ***
## AssignmentStatusApproved      3.813e+04  6.801e+03   5.607 2.48e-08 ***
## WorkTimeInSeconds            -3.926e+01  2.643e+01  -1.486  0.13757
## approx_year_built            -1.305e+01  1.094e+01  -1.192  0.23329
## cats_allowed                 -1.198e+04  3.645e+04  -0.329  0.74249
## common_charges                7.860e+01  1.597e+01   4.922 9.57e-07 ***
## community_district_num        1.056e+03  6.664e+02   1.584  0.11341
## coop_condocondo               1.530e+05  1.074e+04  14.253  < 2e-16 ***
## dining_room_typeformal        7.493e+03  5.496e+03   1.363  0.17296
## dining_room_typeother         6.707e+03  7.704e+03   0.871  0.38413
## dogs_allowed                  9.031e+03  8.323e+03   1.085  0.27806
## kitchen_type                  2.990e+03  3.612e+03   0.828  0.40793
## maintenance_cost              1.969e+01  1.444e+01   1.364  0.17289
## num_bedrooms                  3.306e+04  5.310e+03   6.226 6.30e-10 ***
## num_floors_in_building        1.811e+03  4.159e+02   4.354 1.44e-05 ***
## num_full_bathrooms            6.784e+04  7.388e+03   9.183  < 2e-16 ***
## num_half_bathrooms            2.112e+04  9.664e+03   2.185  0.02903 *
## num_total_rooms              -6.498e+02  3.279e+03  -0.198  0.84293
## pct_tax_deductibl            -9.955e+01  1.400e+02  -0.711  0.47726
## sq_footage                    1.535e+01  4.782e+00   3.209  0.00136 **
## total_taxes                   3.004e+00  2.401e+00   1.251  0.21107
```

```
## walk_score                      1.427e+03  1.779e+02   8.022 2.18e-15 ***
## listing_price_to_nearest_1000 -6.169e+02  1.210e+01 -50.967  < 2e-16 ***
## zip_code                        3.594e+00  1.276e+00   2.817  0.00491 **
## pets_allowed                    1.402e+04  3.728e+04   0.376  0.70688
## maintenance_cost_sq             2.878e-02  5.542e-03   5.194 2.36e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 87610 on 1401 degrees of freedom
##   (357 observations deleted due to missingness)
## Multiple R-squared:  0.7109, Adjusted R-squared:  0.7058
## F-statistic: 137.8 on 25 and 1401 DF,  p-value: < 2.2e-16
```

```
dependent_linear<- attributes(alias(linear_regression)$Complete)$dimnames[[1]]
```

```
library(car)
```

```
## Loading required package: carData
```

```
##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:purrr':
##
##     some
```

```
## The following object is masked from 'package:dplyr':
##
##     recode
```

```
Multicollinearity<-vif(linear_regression)
Multicollinearity
```

```
##                                 GVIF Df GVIF^(1/(2*Df))
## AssignmentStatus            1.897157  1        1.377373
## WorkTimeInSeconds           1.779531  1        1.333990
## approx_year_built           1.019443  1        1.009675
## cats_allowed               57.930061  1        7.611180
## common_charges              2.269063  1        1.506341
## community_district_num      1.054310  1        1.026796
## coop_condo                  4.166802  1        2.041275
## dining_room_type            1.244841  2        1.056278
## dogs_allowed                2.433731  1        1.560042
## kitchen_type                1.010723  1        1.005347
## maintenance_cost           10.005387  1        3.163129
## num_bedrooms                3.220522  1        1.794581
## num_floors_in_building      1.497888  1        1.223882
## num_full_bathrooms          2.113026  1        1.453625
## num_half_bathrooms          1.256449  1        1.120914
## num_total_rooms             3.483801  1        1.866494
## pct_tax_deductibl           1.279803  1        1.131284
```

```
## sq_footage                     1.286839  1        1.134389
## total_taxes                    2.084513  1        1.443784
## walk_score                     1.284415  1        1.133320
## listing_price_to_nearest_1000  1.619947  1        1.272772
## zip_code                       1.407971  1        1.186580
## pets_allowed                  60.863302  1        7.801494
## maintenance_cost_sq            6.029645  1        2.455534
```

We should delete from the training and test datasets:

```
training_set<-training_set %>% select(-cats_allowed)
testing_set<-testing_set %>% select(-cats_allowed)
training_set$cats_allowed
```

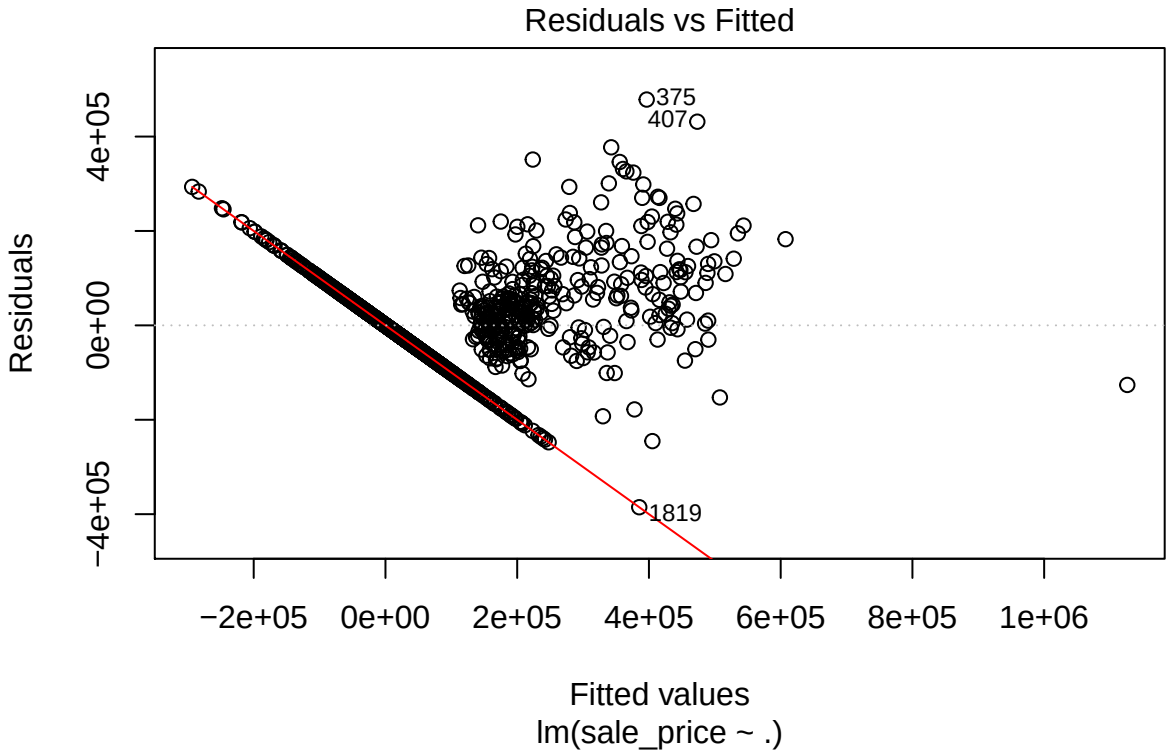
```
## NULL
```

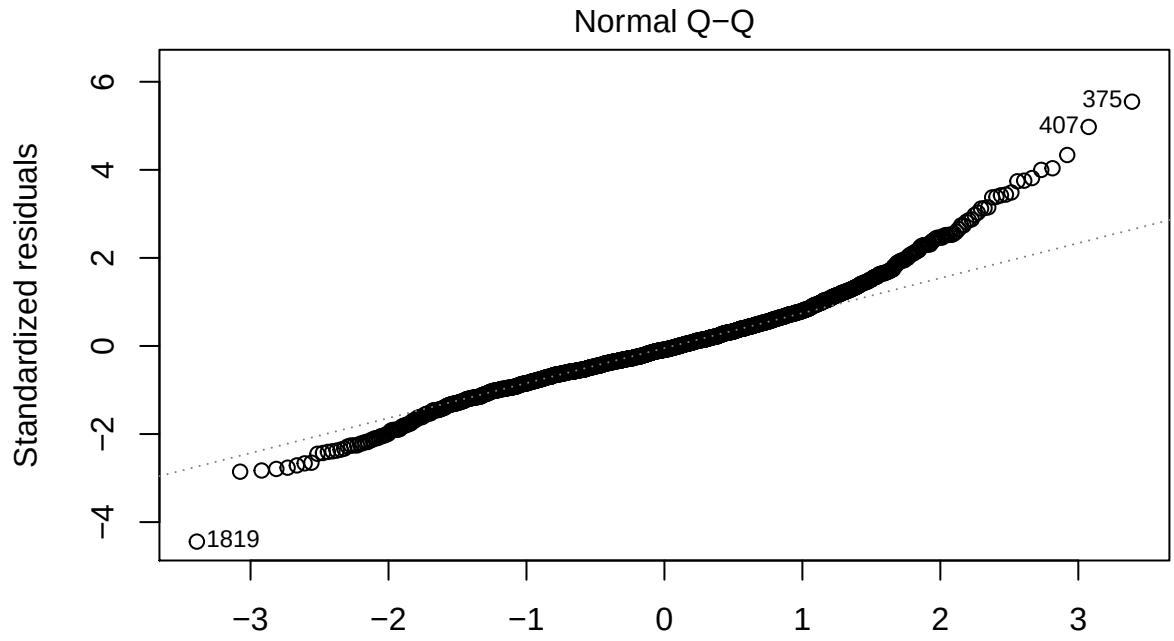
```
linear_regression_2<-lm(sale_price~., training_set)
summary(linear_regression_2)
```

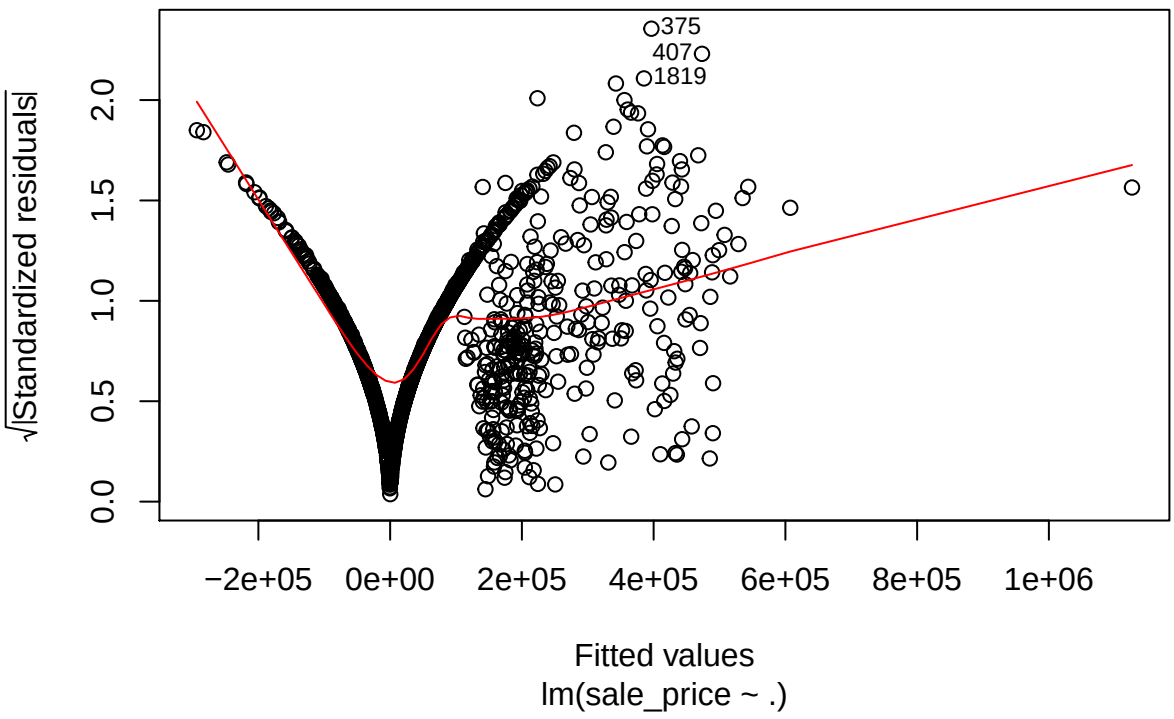
```
##
## Call:
## lm(formula = sale_price ~ ., data = training_set)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -385299  -50611   -7146   42540  478424
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   -1.722e+05  3.716e+04  -4.633 3.94e-06 ***
## AssignmentStatusApproved       3.818e+04  6.797e+03   5.616 2.35e-08 ***
## WorkTimeInSeconds             -3.948e+01  2.641e+01  -1.495  0.13518
## approx_year_built             -1.302e+01  1.094e+01  -1.190  0.23414
## common_charges                 7.853e+01  1.596e+01   4.920 9.68e-07 ***
## community_district_num         1.058e+03  6.661e+02   1.589  0.11232
## coop_condocondo                1.530e+05  1.073e+04  14.257  < 2e-16 ***
## dining_room_typeformal         7.457e+03  5.493e+03   1.358  0.17483
## dining_room_typeother          6.612e+03  7.696e+03   0.859  0.39039
## dogs_allowed                   9.237e+03  8.297e+03   1.113  0.26574
## kitchen_type                   2.951e+03  3.609e+03   0.818  0.41356
## maintenance_cost               1.973e+01  1.444e+01   1.367  0.17190
## num_bedrooms                   3.305e+04  5.308e+03   6.226 6.32e-10 ***
## num_floors_in_building         1.811e+03  4.158e+02   4.355 1.43e-05 ***
## num_full_bathrooms             6.789e+04  7.383e+03   9.195  < 2e-16 ***
## num_half_bathrooms             2.130e+04  9.645e+03   2.208  0.02741 *
## num_total_rooms               -6.646e+02  3.277e+03  -0.203  0.83933
## pct_tax_deductibl             -1.004e+02  1.400e+02  -0.717  0.47324
## sq_footage                     1.534e+01  4.781e+00   3.210  0.00136 **
## total_taxes                    3.001e+00  2.400e+00   1.250  0.21136
## walk_score                     1.428e+03  1.779e+02   8.026 2.11e-15 ***
## listing_price_to_nearest_1000 -6.168e+02  1.210e+01 -50.984  < 2e-16 ***
## zip_code                       3.549e+00  1.268e+00   2.799  0.00519 **
```

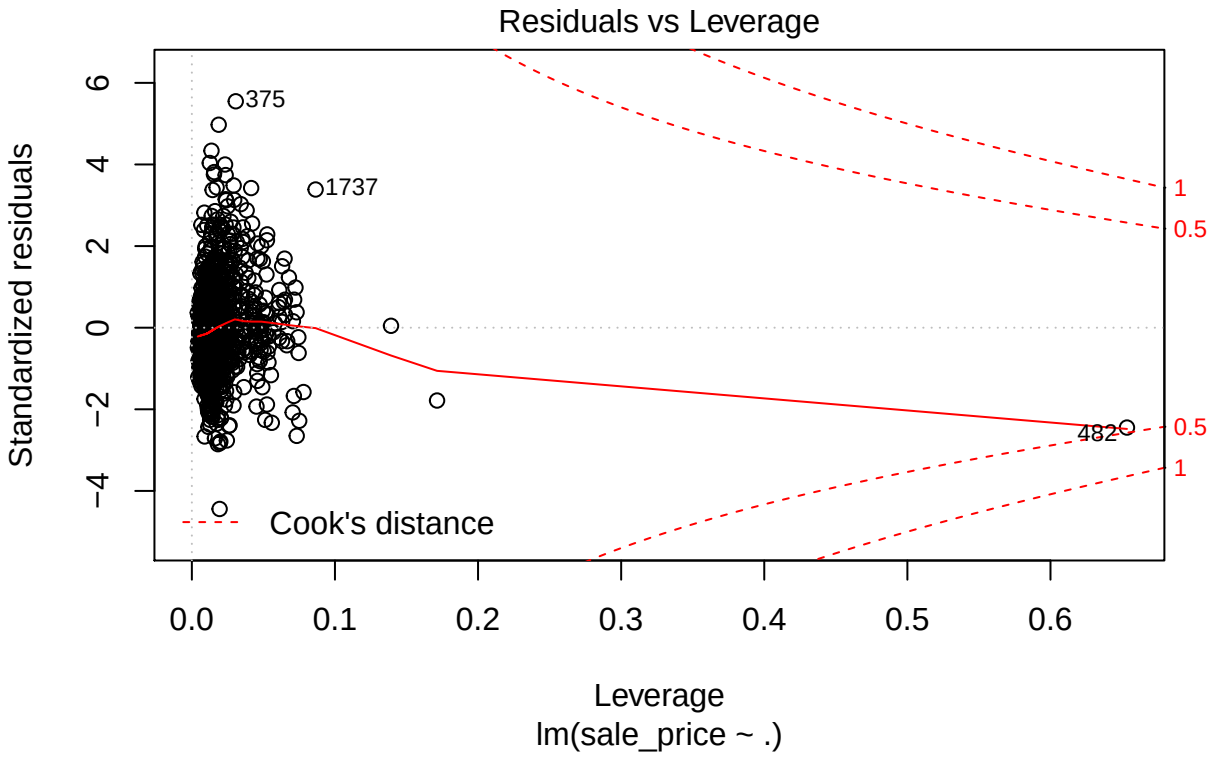
```
## pets_allowed                   2.022e+03  7.504e+03   0.269  0.78762
## maintenance_cost_sq            2.878e-02  5.540e-03   5.196 2.34e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 87590 on 1402 degrees of freedom
##   (357 observations deleted due to missingness)
## Multiple R-squared:  0.7109, Adjusted R-squared:  0.706
## F-statistic: 143.7 on 24 and 1402 DF,  p-value: < 2.2e-16
```

```
plot(linear_regression_2)
```




Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(sale_price ~ .)
375
407
1819
−4
−2
0
2
4
6
−3
−2
−1
0
1
2
3


Scale−Location
√|Standardized residuals|
Fitted values
lm(sale_price ~ .)
375
407
1819
0.0
0.5
1.0
1.5
2.0
−2e+05
0e+00
2e+05
4e+05
6e+05
8e+05
1e+06




```
test_predict = predict(linear_regression_2, testing_set %>% select(-sale_price))
error = test_predict - testing_set$sale_price
```

```
mae <- function(error)
{
    mean(abs(error))
}
```

```
rmse <- function(error)
{
    sqrt(mean(error^2))
}
```

```
mae(error[!is.na(error)])
```

```
## [1] 65062.97
```

```
rmse(error[!is.na(error)])
```

```
## [1] 88595.67
```

4.3. Regression tree

```
library(rsample)#data spliting
library(rpart) #performing reg tree
```

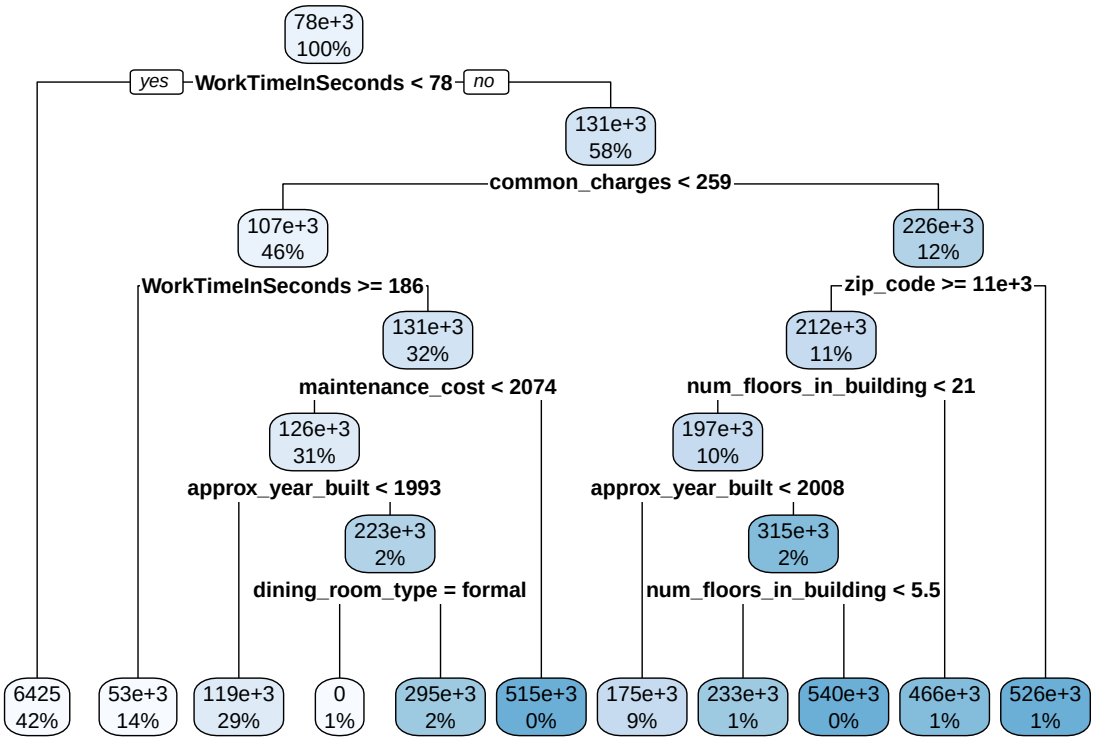
```
library(rpart.plot) #ploting reg tree
library(ipred) #bagging
library(caret) #bagging
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
regression_tree = rpart(
formula = training_set$sale_price ~ .,
data = training_set %>% select(-c(sale_price, listing_price_to_nearest_1000)),
method = "anova",
control = list(minsplit = 10, maxdepth = 6, xval = 5)
)
rpart.plot(regression_tree, roundint = FALSE)
```




```
summary(regression_tree)
```

```
## Call:
## rpart(formula = training_set$sale_price ~ ., data = training_set %>%
##     select(-c(sale_price, listing_price_to_nearest_1000)), method = "anova",
##     control = list(minsplit = 10, maxdepth = 6, xval = 5))
```

```
##   n= 1784
##
##           CP nsplit rel error    xerror       xstd
## 1 0.14098759      0 1.0000000 1.0008053 0.06382014
## 2 0.04919025      1 0.8590124 0.8674295 0.05353607
## 3 0.02245054      2 0.8098222 0.8299930 0.04979637
## 4 0.02188757      3 0.7873716 0.8567103 0.05127746
## 5 0.01776794      4 0.7654840 0.8600008 0.05179934
## 6 0.01567282      5 0.7477161 0.8773395 0.05201251
## 7 0.01098311      6 0.7320433 0.9101428 0.05534898
## 8 0.01008575      8 0.7100771 0.9235481 0.05614990
## 9 0.01000000     10 0.6899056 0.9241985 0.05617929
##
## Variable importance
##      WorkTimeInSeconds       AssignmentStatus         common_charges
##                     28                     19                      9
##       maintenance_cost    maintenance_cost_sq      approx_year_built
##                      8                      8                      5
## num_floors_in_building               zip_code             coop_condo
##                      5                      4                      4
##            total_taxes       dining_room_type             sq_footage
##                      4                      2                      1
##
## Node number 1: 1784 observations,    complexity param=0.1409876
##   mean=77946.52, MSE=2.674331e+10
##   left son=2 (757 obs) right son=3 (1027 obs)
##   Primary splits:
##       WorkTimeInSeconds < 77.5    to the left,  improve=0.14098760, (0 missing)
##       AssignmentStatus  splits as  LR,          improve=0.11056300, (0 missing)
##       common_charges    < 186     to the left,  improve=0.03018641, (0 missing)
##       approx_year_built < 1981.5  to the left,  improve=0.02843430, (0 missing)
##       coop_condo        splits as  LR,          improve=0.02753006, (0 missing)
##   Surrogate splits:
##       AssignmentStatus       splits as  LR,          agree=0.903, adj=0.771, (0 split)
##       zip_code               < 5502    to the left,  agree=0.579, adj=0.008, (0 split)
##       community_district_num < 30.5    to the right, agree=0.577, adj=0.004, (0 split)
##       walk_score             < 51.5    to the left,  agree=0.576, adj=0.001, (0 split)
##
## Node number 2: 757 observations
##   mean=6425.231, MSE=1.623346e+09
##
## Node number 3: 1027 observations,    complexity param=0.04919025
##   mean=130664.7, MSE=3.870951e+10
##   left son=6 (821 obs) right son=7 (206 obs)
##   Primary splits:
##       common_charges     < 258.5   to the left,  improve=0.05903383, (0 missing)
##       approx_year_built  < 1981.5  to the left,  improve=0.05063810, (0 missing)
##       coop_condo         splits as  LR,          improve=0.05047676, (0 missing)
##       total_taxes        < 5.5     to the left,  improve=0.04273838, (0 missing)
##       num_full_bathrooms < 1.5     to the left,  improve=0.03864647, (0 missing)
##   Surrogate splits:
##       maintenance_cost    < 158.5   to the right, agree=0.904, adj=0.519, (0 split)
##       maintenance_cost_sq < 25134.5 to the right, agree=0.904, adj=0.519, (0 split)
##       coop_condo          splits as  LR,          agree=0.888, adj=0.442, (0 split)
```

```
##       total_taxes         < 59.5    to the left,  agree=0.886, adj=0.432, (0 split)
##       approx_year_built   < 1979.5  to the left,  agree=0.854, adj=0.272, (0 split)
##
## Node number 6: 821 observations,    complexity param=0.02245054
##   mean=106719.4, MSE=2.767805e+10
##   left son=12 (258 obs) right son=13 (563 obs)
##   Primary splits:
##       WorkTimeInSeconds   < 185.5   to the right, improve=0.04713661, (0 missing)
##       sq_footage          < 1636    to the left,  improve=0.03766815, (0 missing)
##       maintenance_cost    < 2159    to the left,  improve=0.03575009, (0 missing)
##       maintenance_cost_sq < 4661450 to the left,  improve=0.03575009, (0 missing)
##       num_full_bathrooms  < 1.5     to the left,  improve=0.02470266, (0 missing)
##   Surrogate splits:
##       zip_code               < 22912.5 to the right, agree=0.691, adj=0.016, (0 split)
##       maintenance_cost       < 2211    to the right, agree=0.689, adj=0.012, (0 split)
##       maintenance_cost_sq    < 4890042 to the right, agree=0.689, adj=0.012, (0 split)
##       community_district_num < 7       to the left,  agree=0.688, adj=0.008, (0 split)
##       pct_tax_deductibl      < 52.5    to the right, agree=0.688, adj=0.008, (0 split)
##
## Node number 7: 206 observations,    complexity param=0.01776794
##   mean=226097.5, MSE=7.128214e+10
##   left son=14 (197 obs) right son=15 (9 obs)
##   Primary splits:
##       zip_code               < 11103   to the right, improve=0.05772969, (0 missing)
##       num_floors_in_building < 20.5    to the left,  improve=0.05641973, (0 missing)
##       approx_year_built      < 2007.5  to the left,  improve=0.04451022, (0 missing)
##       community_district_num < 29.5    to the left,  improve=0.03368921, (0 missing)
##       total_taxes            < 5329.5  to the left,  improve=0.02889533, (0 missing)
##
## Node number 12: 258 observations
##   mean=53362.4, MSE=1.670248e+10
##
## Node number 13: 563 observations,    complexity param=0.02188757
##   mean=131170.7, MSE=3.080519e+10
##   left son=26 (556 obs) right son=27 (7 obs)
##   Primary splits:
##       maintenance_cost    < 2074    to the left,  improve=0.06021093, (0 missing)
##       maintenance_cost_sq < 4302152 to the left,  improve=0.06021093, (0 missing)
##       sq_footage          < 1625    to the left,  improve=0.04372964, (0 missing)
##       num_full_bathrooms  < 1.5     to the left,  improve=0.03085749, (0 missing)
##       num_bedrooms        < 2.5     to the left,  improve=0.02516562, (0 missing)
##   Surrogate splits:
##       maintenance_cost_sq < 4302152 to the left,  agree=1, adj=1, (0 split)
##
## Node number 14: 197 observations,    complexity param=0.01567282
##   mean=212386.2, MSE=6.625119e+10
##   left son=28 (186 obs) right son=29 (11 obs)
##   Primary splits:
##       num_floors_in_building < 20.5    to the left,  improve=0.05729245, (0 missing)
##       community_district_num < 26.5    to the right, improve=0.03767048, (0 missing)
##       total_taxes            < 5329.5  to the left,  improve=0.03743928, (0 missing)
##       zip_code               < 11366   to the right, improve=0.03452671, (0 missing)
##       approx_year_built      < 1967    to the left,  improve=0.02905635, (0 missing)
##
```

```
## Node number 15: 9 observations
##   mean=526222.2, MSE=8.721417e+10
##
## Node number 26: 556 observations,    complexity param=0.01008575
##   mean=126338.3, MSE=2.739481e+10
##   left son=52 (519 obs) right son=53 (37 obs)
##   Primary splits:
##       approx_year_built  < 1992.5  to the left,  improve=0.02433866, (0 missing)
##       zip_code           < 11376   to the right, improve=0.02314308, (0 missing)
##       num_full_bathrooms < 1.5     to the left,  improve=0.01656416, (0 missing)
##       num_bedrooms       < 1.5     to the left,  improve=0.01502873, (0 missing)
##       coop_condo         splits as  LR,          improve=0.01376899, (0 missing)
##   Surrogate splits:
##       common_charges < 134     to the left,  agree=0.957, adj=0.351, (0 split)
##       coop_condo     splits as  LR,          agree=0.955, adj=0.324, (0 split)
##       total_taxes    < 5.5     to the left,  agree=0.939, adj=0.081, (0 split)
##
## Node number 27: 7 observations
##   mean=514999.9, MSE=1.52507e+11
##
## Node number 28: 186 observations,    complexity param=0.01098311
##   mean=197403.6, MSE=6.191024e+10
##   left son=56 (156 obs) right son=57 (30 obs)
##   Primary splits:
##       approx_year_built      < 2007.5  to the left,  improve=0.04292660, (0 missing)
##       common_charges         < 319.5   to the right, improve=0.03081828, (0 missing)
##       num_floors_in_building < 2.5     to the left,  improve=0.03011195, (0 missing)
##       WorkTimeInSeconds      < 180.5   to the right, improve=0.02747310, (0 missing)
##       community_district_num < 26.5    to the right, improve=0.02589884, (0 missing)
##
## Node number 29: 11 observations
##   mean=465727.3, MSE=7.167529e+10
##
## Node number 52: 519 observations
##   mean=119443.9, MSE=2.334929e+10
##
## Node number 53: 37 observations,    complexity param=0.01008575
##   mean=223047, MSE=7.412214e+10
##   left son=106 (9 obs) right son=107 (28 obs)
##   Primary splits:
##       dining_room_type       splits as  RLR,         improve=0.19732890, (2 missing)
##       zip_code               < 11375.5 to the right, improve=0.11469140, (0 missing)
##       common_charges         < 242     to the right, improve=0.08106746, (0 missing)
##       sq_footage             < 760     to the left,  improve=0.05408841, (0 missing)
##       community_district_num < 24.5    to the left,  improve=0.04988056, (0 missing)
##   Surrogate splits:
##       zip_code        < 11382   to the right, agree=0.857, adj=0.444, (2 split)
##       sq_footage      < 1450    to the right, agree=0.829, adj=0.333, (0 split)
##       common_charges  < 247     to the right, agree=0.800, adj=0.222, (0 split)
##       num_bedrooms    < 2.5     to the right, agree=0.800, adj=0.222, (0 split)
##       num_total_rooms < 5.5     to the right, agree=0.800, adj=0.222, (0 split)
##
## Node number 56: 156 observations
##   mean=174796.6, MSE=5.547625e+10
```

```
##
## Node number 57: 30 observations,    complexity param=0.01098311
##   mean=314960, MSE=7.888987e+10
##   left son=114 (22 obs) right son=115 (8 obs)
##   Primary splits:
##       num_floors_in_building < 5.5     to the left,  improve=0.2339537, (0 missing)
##       common_charges         < 313     to the right, improve=0.2067666, (0 missing)
##       num_full_bathrooms     < 1.5     to the left,  improve=0.1789606, (0 missing)
##       sq_footage             < 690     to the left,  improve=0.1179421, (0 missing)
##       WorkTimeInSeconds      < 177.5   to the right, improve=0.1079093, (0 missing)
##   Surrogate splits:
##       walk_score < 88      to the right, agree=0.767, adj=0.125, (0 split)
##
## Node number 106: 9 observations
##   mean=0, MSE=0
##
## Node number 107: 28 observations
##   mean=294740.6, MSE=7.681608e+10
##
## Node number 114: 22 observations
##   mean=233036.4, MSE=8.167225e+10
##
## Node number 115: 8 observations
##   mean=540250, MSE=2.026188e+09
```

```
test_predict = predict(regression_tree, testing_set %>% select(-sale_price))
error = test_predict - testing_set$sale_price
```

```
mae(error[!is.na(error)])
```

```
## [1] 77782.62
```

```
rmse(error[!is.na(error)])
```

```
## [1] 135319.3
```

4.4. Random Forest

```
library(randomForest)
```

```
## randomForest 4.6-14

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
training_set1<- na.exclude(training_set %>% filter(sale_price>0, ! is.na(sale_price)))
str(training_set1)
```

```
## 'data.frame':    340 obs. of  24 variables:
##  $ AssignmentStatus             : chr  "Approved" "Approved" "Approved" "Approved" ...
##  $ WorkTimeInSeconds            : num  130 100 124 134 117 428 224 120 356 117 ...
##  $ approx_year_built            : num  2005 1972 1965 1950 1982 ...
##  $ common_charges               : num  0 626 0 0 1017 ...
##  $ community_district_num       : num  30 25 27 26 25 28 25 26 28 24 ...
##  $ coop_condo                   : Factor w/ 2 levels "co-op","condo": 2 2 1 1 2 1 2 1 1 1 ...
##  $ dining_room_type             : Factor w/ 3 levels "combo","formal",..: 3 1 3 1 1 1 1 1 1 2 ...
##  $ dogs_allowed                 : num  0 0 0 0 0 1 1 0 0 1 ...
##  $ kitchen_type                 : num  3 3 2 3 3 3 3 3 3 3 ...
##  $ maintenance_cost             : num  0 0 745 830 0 592 0 642 655 591 ...
##  $ num_bedrooms                 : num  1 1 1 2 2 1 2 1 1 1 ...
##  $ num_floors_in_building       : num  4 22 6 3 22 4 13 6 12 6 ...
##  $ num_full_bathrooms           : int  1 1 1 1 3 1 1 1 1 1 ...
##  $ num_half_bathrooms           : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ num_total_rooms              : num  3 4 3 4 7 1 4 4 3 3 ...
##  $ pct_tax_deductibl            : num  0 0 0 0 0 50 0 0 0 40 ...
##  $ sale_price                   : num  535000 475000 100000 226000 790000 125000 480000 168000 140000
##  $ sq_footage                   : num  681 874 0 0 1419 ...
##  $ total_taxes                  : num  1320 4000 0 0 5807 ...
##  $ walk_score                   : int  98 82 77 88 82 99 90 76 90 91 ...
##  $ listing_price_to_nearest_1000: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ zip_code                     : num  11102 11360 11414 11364 11360 ...
##  $ pets_allowed                 : num  0 0 1 1 1 1 1 1 0 1 ...
##  $ maintenance_cost_sq          : num  0 0 555025 688900 0 ...
##  - attr(*, "na.action")= 'exclude' Named int  6 16 17 18 21 26 28 29 43 45 ...
##   ..- attr(*, "names")= chr  "6" "16" "17" "18" ...
```

```
training_set1$AssignmentStatus=as.factor(training_set1$AssignmentStatus)
random_forest = randomForest(training_set1$sale_price ~ .,
training_set1 %>% select(-c(sale_price, listing_price_to_nearest_1000)))
summary(random_forest)
```

```
##                 Length Class  Mode
## call              3    -none- call
## type              1    -none- character
## predicted       340    -none- numeric
## mse             500    -none- numeric
## rsq             500    -none- numeric
## oob.times       340    -none- numeric
## importance       22    -none- numeric
## importanceSD      0    -none- NULL
## localImportance   0    -none- NULL
## proximity         0    -none- NULL
## ntree             1    -none- numeric
## mtry              1    -none- numeric
```

```
## forest           11    -none- list
## coefs             0    -none- NULL
## y               340    -none- numeric
## test              0    -none- NULL
## inbag             0    -none- NULL
## terms             3    terms  call
```

```
random_forest
```

```
##
## Call:
##  randomForest(formula = training_set1$sale_price ~ ., data = training_set1 %>%      select(-c(sale_pr
##                Type of random forest: regression
##                      Number of trees: 500
## No. of variables tried at each split: 7
##
##           Mean of squared residuals: 6803995116
##                     % Var explained: 78.1
```

```
sqrt(which.min(random_forest$mse))
```

```
## [1] 22
```

5. Performance results

In this section, we will compare the performance of the found model using the RMSE and MAE metrics. The models that show the lowest value of these metrics are the best ones to fit the housing sale price.

The Random Forest a robust machine learing algorithm provides the best performance looking at the values of the RMSE and MAE that are highly inferior to the ones found in other models.

It's trivial, because while the linear regression and regression tree estimates once the model, the random forest can compute lot of combinations to find the optimal model that better suit the dependent variable and reduce more the error of prediction.

6. Discussion

The used raw database has needed lot of work to clean it and format it. That would have been more relevant for database maker to better collect the data and include more control techniques to avoid all the incoherence found in the data cleansing part.

Acknowledgments

I would like to thank my brother & friend that those who helped me develop my expertise in R programming language, Machine learing and predictive modeling.