California House Price Analysis

Betti Gianmarco (2097050), Marinelli Andrea (2091700), Rinaldi Giorgia (2092226)

2023-06-27

```
knitr::opts_chunk$set(warning = "all")
library(ggplot2)
library(RColorBrewer)
library(ggcorrplot)
library(knitr)
library(visdat)
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
library(cowplot)
library(gridExtra)
library(leaps)
library(caret)
```

```
## Caricamento del pacchetto richiesto: lattice
```

```
library(dplyr)
```

```
##
## Caricamento pacchetto: 'dplyr'
```

```
## Il seguente oggetto è mascherato da 'package:gridExtra':
##
##     combine
```

```
## I seguenti oggetti sono mascherati da 'package:stats':
##
##     filter, lag
```

```
## I seguenti oggetti sono mascherati da 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(glmnet)
```

```
## Caricamento del pacchetto richiesto: Matrix
```

```
## Loaded glmnet 4.1-7
```

```
library(cartogram)
library(car)
```

```
## Caricamento del pacchetto richiesto: carData
```

```
##
## Caricamento pacchetto: 'car'
```

```
## Il seguente oggetto è mascherato da 'package:dplyr':
##
##     recode
```

```
library(ggmap)
```

```
## The legacy packages maptools, rgdal, and rgeos, underpinning the sp package,
## which was just loaded, will retire in October 2023.
## Please refer to R-spatial evolution reports for details, especially
## https://r-spatial.org/r/2023/05/15/evolution4.html.
## It may be desirable to make the sf package available;
## package maintainers should consider adding sf to Suggests:.
## The sp package is now running under evolution status 2
##      (status 2 uses the sf package in place of rgdal)
```

```
## ℹ Google's Terms of Service: <https://mapsplatform.google.com>
## ℹ Please cite ggmap if you use it! Use `citation("ggmap")` for details.
##
## Caricamento pacchetto: 'ggmap'
##
##
## Il seguente oggetto è mascherato da 'package:cowplot':
##
##     theme_nothing
```

```
library(MASS)
```

```
##
## Caricamento pacchetto: 'MASS'
##
## Il seguente oggetto è mascherato da 'package:dplyr':
##
##     select
```

Introduction

The US Census Bureau has released the California Census Data, which provides comprehensive information for each block group in California. This dataset encompasses a diverse range of metrics, including population figures, median income levels, median housing prices, and various other indicators. The following analysis has

been conducted examining California House Price dataset in order to construct a regression model that can effectively predict the median house value by considering the primary influencing factors. The goal is to develop a model that achieves a high level of accuracy and efficiency in its predictions.

Data Collection

California House Price it is a publicly available dataset (Kaggle: https://www.kaggle.com/datasets/shibumohapatra/house-price (https://www.kaggle.com/datasets/shibumohapatra/house-price)). In this dataset, districts, or block groups, represent the smallest geographical units for which sample data is published by the US Census Bureau. Typically, a block group comprises a population ranging from 600 to 3,000 individuals. It has 20,640 observations corresponding to districts and 10 variables corresponding to the following metrics:

- longitude (signed numeric - float) : Longitude value for the block in California, USA
- latitude (numeric - float) : Latitude value for the block in California, USA
- housing_median_age (numeric - int) : Median age of the house in the block
- total_rooms (numeric - int) : Count of the total number of rooms (excluding bedrooms) in all houses in the block
- total_bedrooms (numeric - float) : Count of the total number of bedrooms in all houses in the block
- population (numeric - int) : Count of the total number of population in the block
- households (numeric - int) : Count of the total number of households in the block
- median_income (numeric - float) : Median of the total household income of all the houses in the block
- ocean_proximity (numeric - categorical) : Type of the landscape of the block
- median_house_value (numeric - int) : Median of the household prices of all the houses in the block (US$)

```
data <- read.csv("california_housing.csv")
```

Data cleaning and filtering

Once the dataset has been imported, we explore it to check the eventual presence of duplicated rows and missing values.

```
anyDuplicated(data)
```

```
## [1] 0
```

The output “0” means that there aren’t duplicated rows.

```
summary(data)
```

```
##    longitude         latitude     housing_median_age  total_rooms
##  Min.   :-124.3   Min.   :32.54   Min.   : 1.00      Min.   :    2
##  1st Qu.:-121.8   1st Qu.:33.93   1st Qu.:18.00      1st Qu.: 1448
##  Median :-118.5   Median :34.26   Median :29.00      Median : 2127
##  Mean   :-119.6   Mean   :35.63   Mean   :28.64      Mean   : 2636
##  3rd Qu.:-118.0   3rd Qu.:37.71   3rd Qu.:37.00      3rd Qu.: 3148
##  Max.   :-114.3   Max.   :41.95   Max.   :52.00      Max.   :39320
##
##  total_bedrooms     population      households     median_income
##  Min.   :   1.0   Min.   :    3   Min.   :   1.0   Min.   : 0.4999
##  1st Qu.: 296.0   1st Qu.:  787   1st Qu.: 280.0   1st Qu.: 2.5634
##  Median : 435.0   Median : 1166   Median : 409.0   Median : 3.5348
##  Mean   : 537.9   Mean   : 1425   Mean   : 499.5   Mean   : 3.8707
##  3rd Qu.: 647.0   3rd Qu.: 1725   3rd Qu.: 605.0   3rd Qu.: 4.7432
##  Max.   :6445.0   Max.   :35682   Max.   :6082.0   Max.   :15.0001
##  NA's   :207
##  ocean_proximity    median_house_value
##  Length:20640       Min.   : 14999
##  Class :character   1st Qu.:119600
##  Mode  :character   Median :179700
##                     Mean   :206856
##                     3rd Qu.:264725
##                     Max.   :500001
##
```

```
anyNA(data)
```

```
## [1] TRUE
```

Using “summary” and “anyNA” it is possible to detect the presence of missing values. Since the variables in the dataset are both numerical and categorical, we proceeds analyzing every kind of variables separately. First of all, we analyze numerical variables.

```
num_col <- sapply(data, is.numeric)
df_num <- data[, num_col]
anyNA(df_num)
```

```
## [1] TRUE
```

```
col_with_na <- colSums(is.na(df_num)) > 0
vis_miss(df_num)
```

As it can be seen from the plot above, missing values represents 0.1% and they are concentrated in the fifth column (total_bedrooms), so we decide to drop them.

```
# dropping rows with NA
rows_with_na <- is.na(data[, 5])
data <- data[!rows_with_na, ]
anyNA(data)
```

```
## [1] FALSE
```

Now we detect whether there is a label in the categorical variable that can testify the presence of missing values.

```
cat_colums <- !num_col
cat_colums[cat_colums == TRUE]
```

```
## ocean_proximity
##            TRUE
```

```
unique(data$ocean_proximity)
```

```
## [1] "NEAR BAY"   "<1H OCEAN"  "INLAND"     "NEAR OCEAN" "ISLAND"
```

```
# creating a copy of dataframe
cleaned_data <- data
```

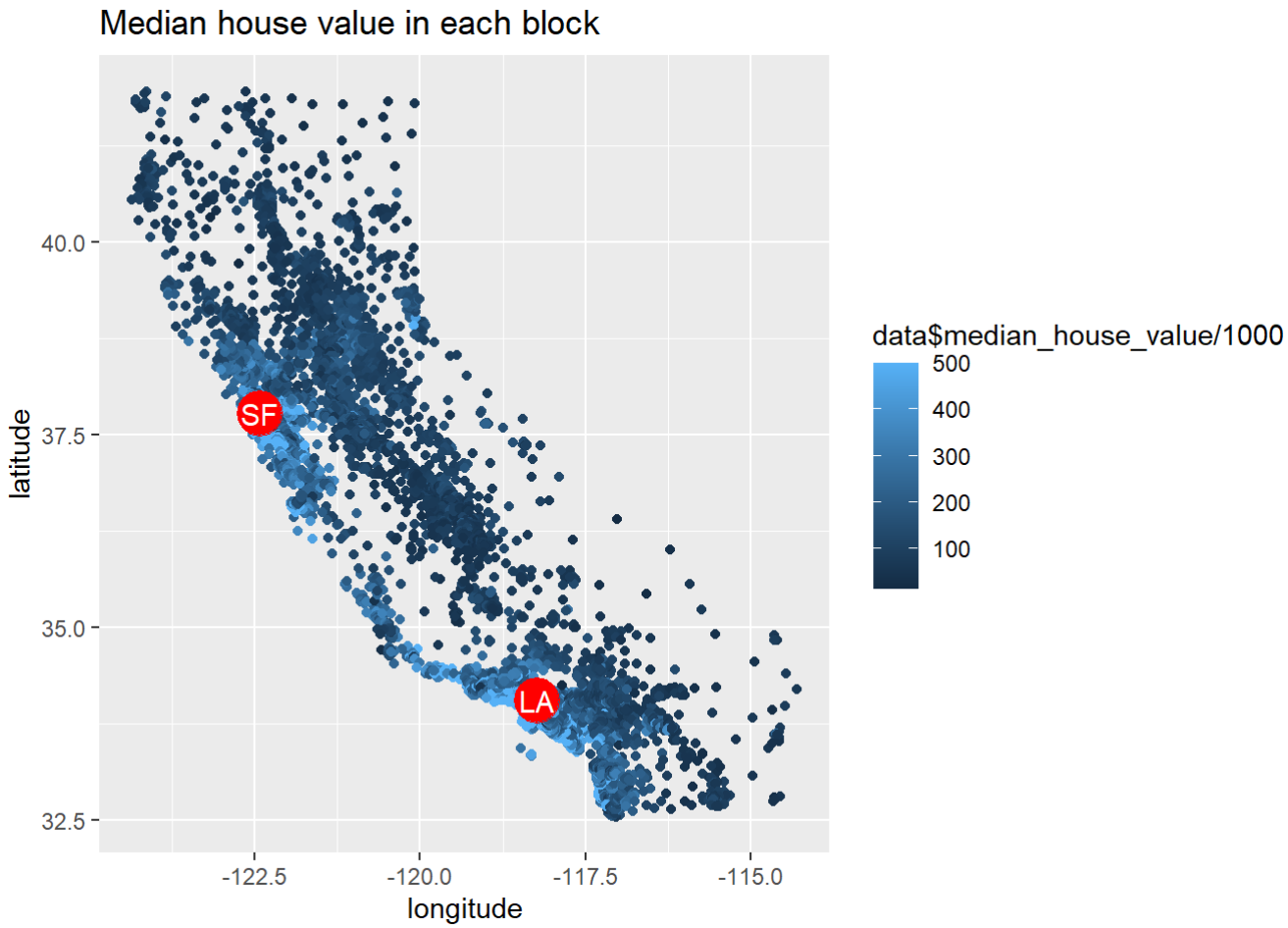
There is no label different from the expected ones, so there are no missing values in the categorical variable “ocean proximity”. The dataset is now cleaned from missing values.

Exploratory Data Analysis (EDA)

New features

At this stage of the analysis, we examine the meaning of the variables. Variables “latitude” and “longitude” individually are not particularly informative. Therefore, we decide to combine them to create a new numerical variable called “distance”. This decision stems from the observation, as evident from the graph below, that areas in close proximity to Los Angeles and San Francisco tend to exhibit higher values of the target variable “median house value.”

```
# Los Angeles and San Francisco coordinates
LA_long <- -118.24
LA_lat <- 34.05
SF_long <- -122.43
SF_lat <- 37.77

# creating a dataframe with LA and SF coordinates
latitude_mc <- c(LA_lat, SF_lat)
longitude_mc <- c(LA_long, SF_long)
labels <- c("LA", "SF")
point <- as.data.frame(cbind(latitude_mc, longitude_mc))
rownames(point) = labels

ggplot() +
  geom_point(data = data, mapping = aes(x = data$longitude, y = data$latitude,
                                        color = data$median_house_value/1000))+
  geom_point(data = point , aes(x = longitude_mc, y = latitude_mc),
             color = "red", size = 8)+
  geom_text(data = point , aes(x = longitude_mc, y = latitude_mc, label = labels),
            color = "white", size = 4)+
  labs(x = "longitude", y = "latitude", title = "Median house value in each block")
```



We have therefore decided to include the variable “distance” in the dataset, which captures the distances of the blocks from the nearest major city (either Los Angeles or San Francisco). To create this variable, we assigned to the “distance” variable the distance of each block from the nearest city (either LA or SF). We calculated the distance using the Euclidean distance method. Let $dist_{SF}$(x) and $dist_{LA}$(x) be the distance of block *x* respectively from San Francisco and Los Angeles. The variable *Distance* is defined as:

$$Distance = min\{dist_{SF}; dist_{LA}\}$$

Then we include “distance” in the dataframe.

```
# Euclidean distance
E_dist <- function(x, y, x_1, y_1) {
  sqrt(sum((x_1 - x)**2 , (y_1 - y)**2 ))
}

# computing distances of each block from LA and SF
dist_LA <- mapply(E_dist, data$longitude, data$latitude, LA_long, LA_lat)
dist_SF <- mapply(E_dist, data$longitude, data$latitude, SF_long, SF_lat)


# initializing an empty vector
n <- dim(data)[1]
distance <- numeric(n)

# assigning values to "distance" variable
for (i in 1:n) {
  if (dist_LA[i] >= dist_SF[i]){
    distance[i] <- dist_SF[i]
  }
  else{
    distance[i] <- dist_LA[i]
  }
}

# adding "distance" column to the dataframe
data <- cbind(distance, data)

# deleting "latitude" and "longitude" columns from the dataset
data <- data[, -c(2, 3)]
```

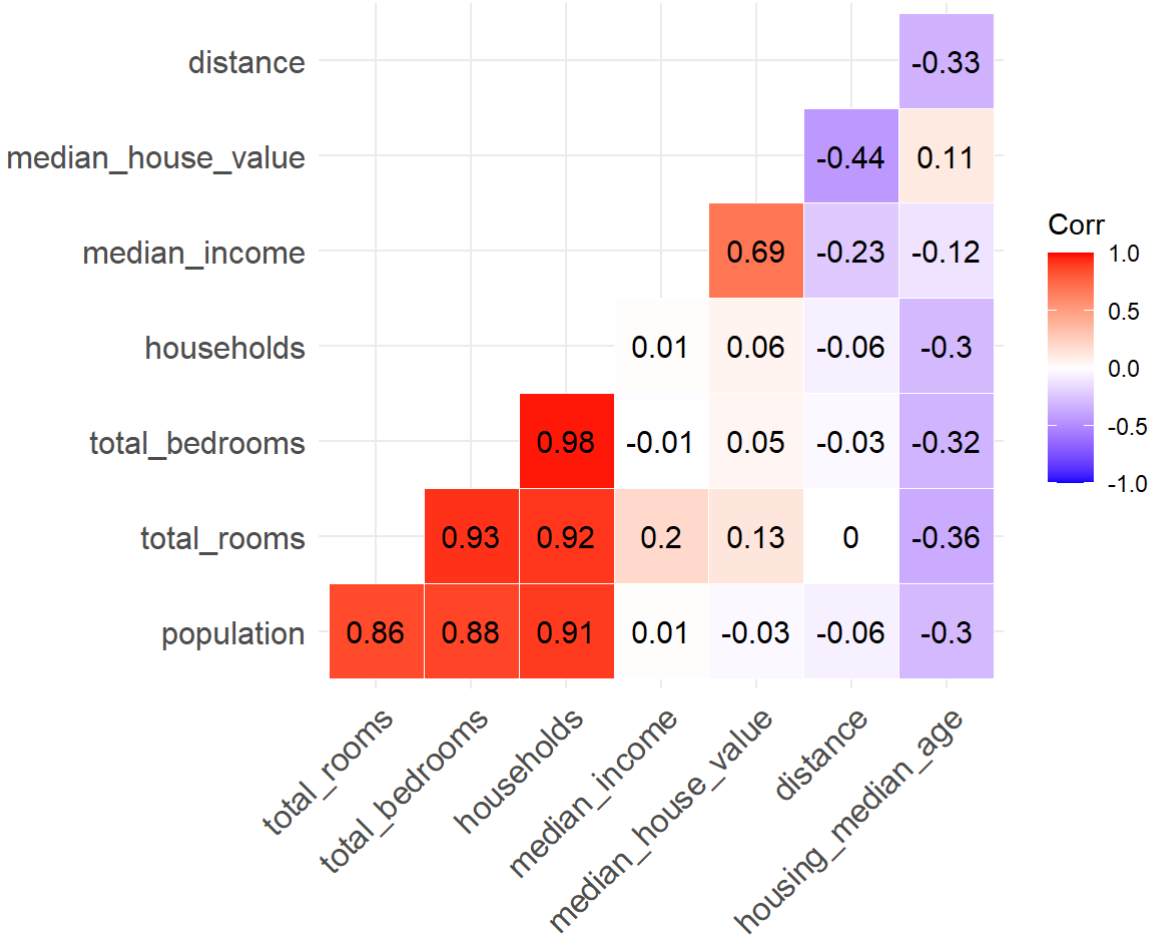
In order to check if “distance” variable makes sense, we plot our response variable “median house values” with respect to two groups representing blocks close to big cities and blocks far from them. Blocks are split using as threshold the median distance.

```
# threshold computed as the median distance
med_dist <- median(data$distance)

# mask to filter blocks
near_cities <- data$distance <= med_dist

# colors
color_1 <- "#FFCC99"
color_2 <- "black"

# plot densities
 dens_dist <- ggplot(data, aes(x = median_house_value / 1000, color = near_cities, fill = nea
r_cities)) +
  geom_density(alpha = 0.5)+
  labs(x ="Median House Value (in 1000 $)", y="Density")+
  ggtitle("Median house values vs nearess from main cities")

# distance histogram
dist_histogram <- ggplot(data, aes(x = distance)) +
  geom_histogram(aes(y = after_stat(density)), fill=color_1, color="black", alpha=1, bins = 1
5) +
  geom_density(color= color_2, linewidth = 0.8) +
  geom_vline(aes(xintercept = mean(distance)), color="blue",linetype="dashed", linewidth=1) +
  geom_vline(aes(xintercept = median(distance)), color="red",linetype="dashed", linewidth=1)
+
  labs(x ="distance", y="Count")+
  ggtitle("Histogram of the distance from main cities (LA-SF)")

options(repr.plot.width = 2, repr.plot.height =3)
plot_grid(dist_histogram, dens_dist, nrow = 1)
```

The graphs above show that blocks close to big cities have higher median house value than blocks far from LA and SF, as expected.

Then we proceed detecting correlation among numerical variables

```
data2 <- data[sapply(data, is.numeric)]
ggcorrplot(cor(data2), hc.order=TRUE, type = "lower", outline.col= "white", lab=TRUE)
```



As it can be seen from the plot above, the variable “households” has high correlation with “total rooms” and “total bedrooms”. Therefore we decide to make them interact creating two new variables dividing “total rooms” and “total bedrooms” by “households”. Those two new variables are called “bedrooms_ph” and “rooms_ph”.

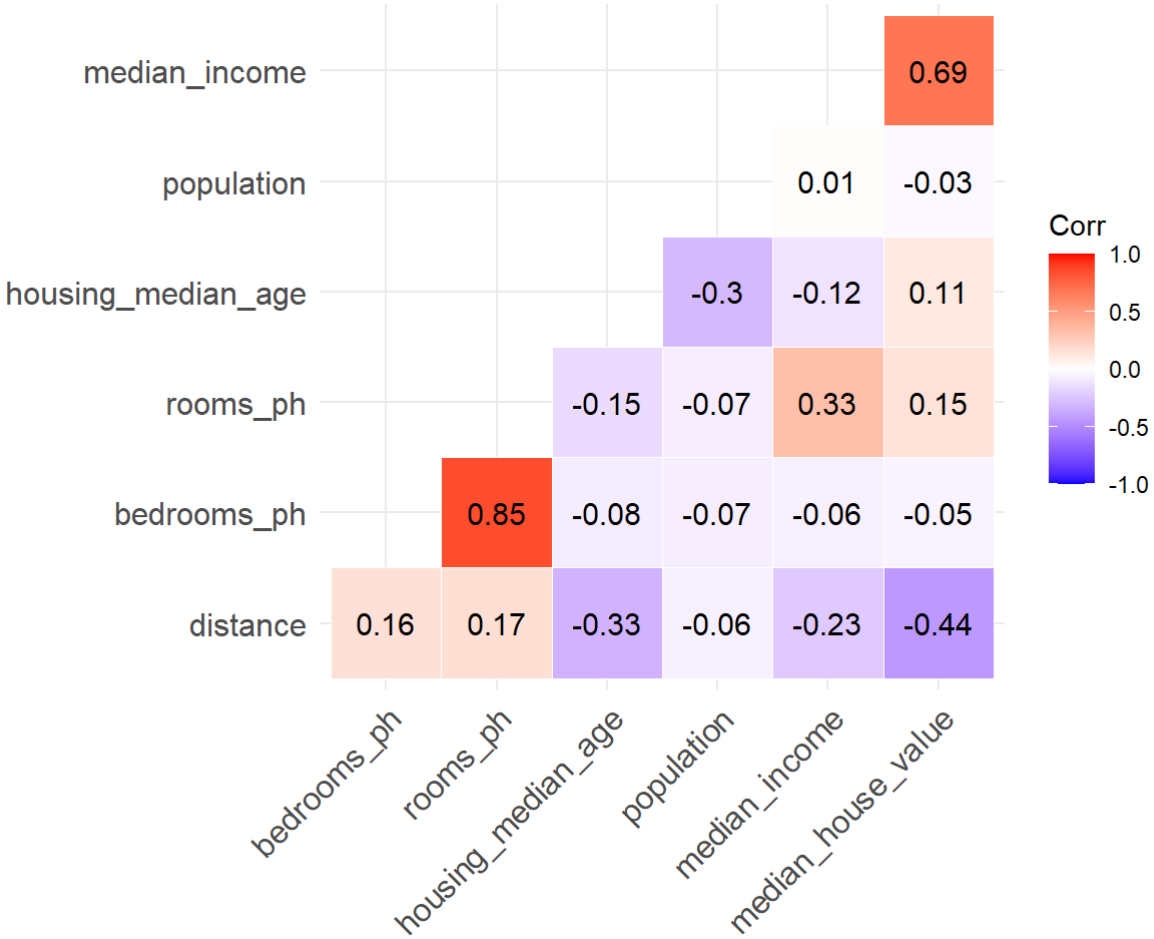
```
bedrooms_ph <- data$total_bedrooms/data$households
rooms_ph <- data$total_rooms/data$households
```

Then we update the dataframe adding columns “bedrooms_ph” and “rooms_ph” and delating “households”.

```
data <- cbind(distance, bedrooms_ph, rooms_ph, data[, -c(1,3,4,6)])
rm(distance) # deleting distance variable to avoid conflicts
```

Now we can check correlation matrix to see how correlation between variables is changed

```
data3 <- data[sapply(data, is.numeric)]
ggcorrplot(cor(data3), hc.order=TRUE, type = "lower", outline.col= "white", lab=TRUE)
```



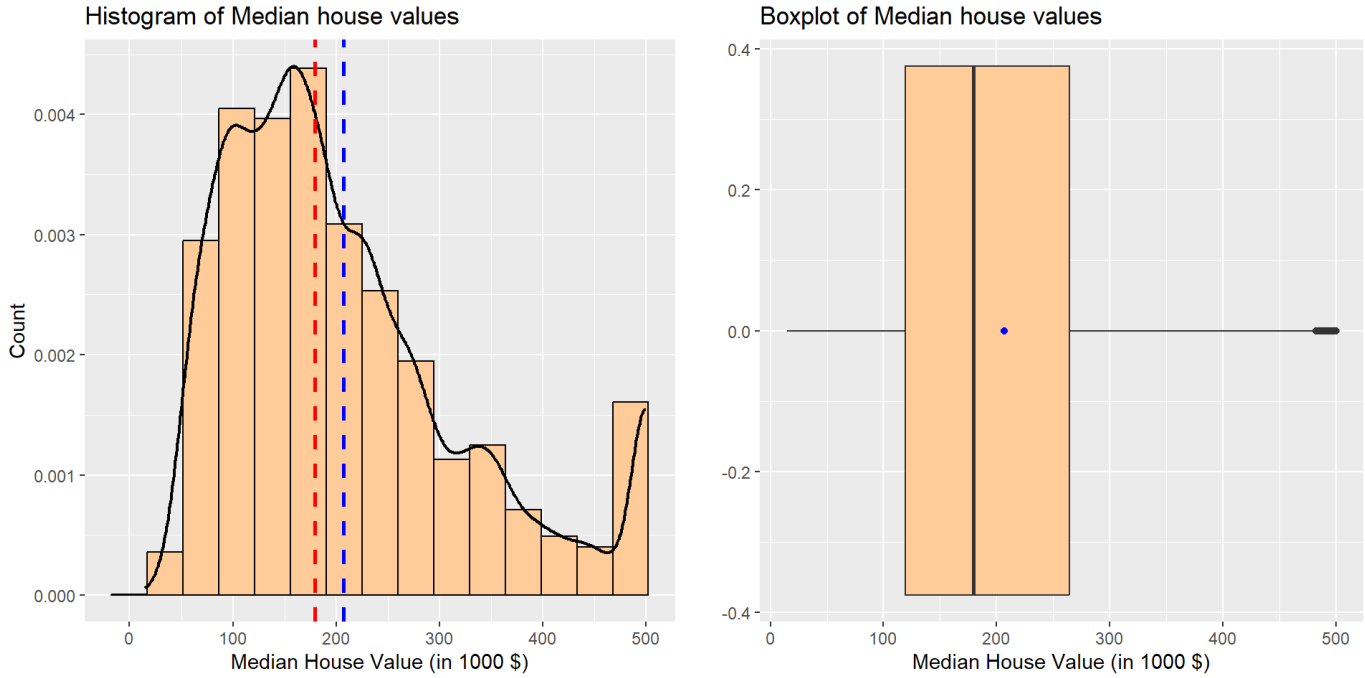
The correlation matrix above shows an improvement. We are going to detect it again at the end of the Exploratory Data Analysis, right after the analysis of eventual outliers.

First investigation on distribution

In the following paragraph, graphs will be plotted in order to see variables distribution and eventual anomalous data such as outliers.

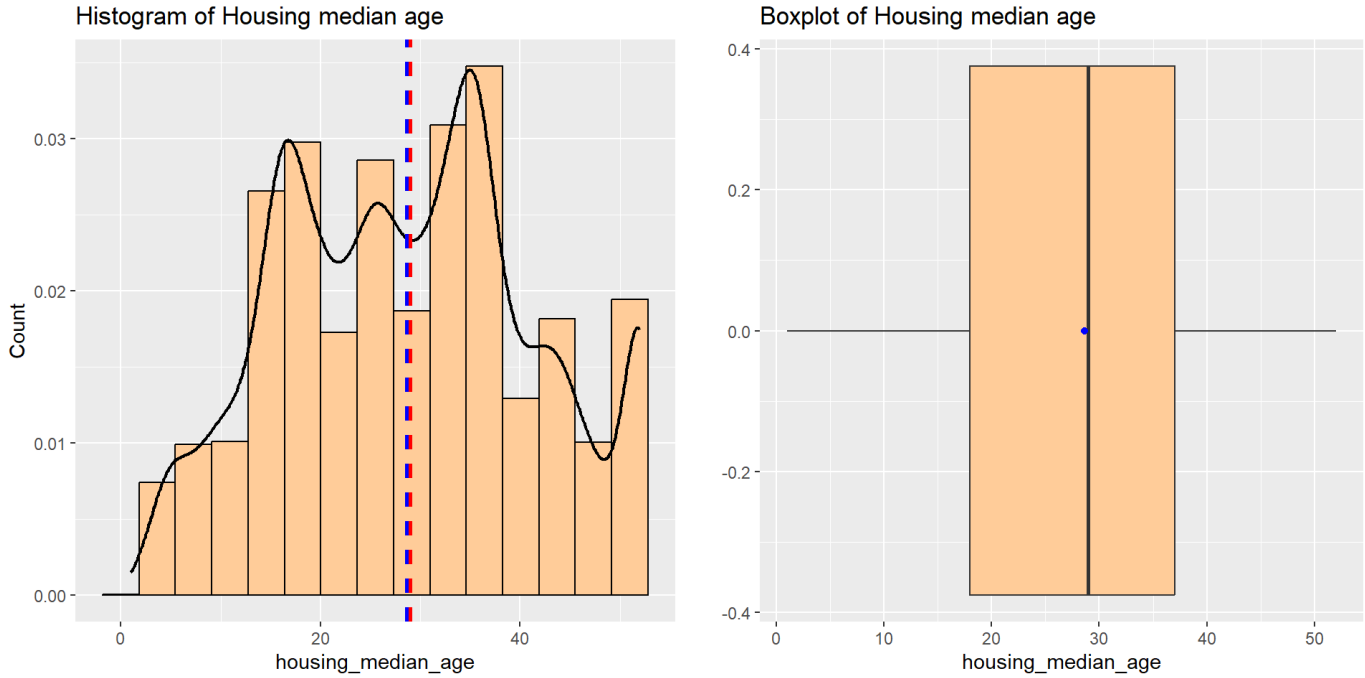
We are going to analyze one variable at a time beginning with the response variable “median house value”.

```
# median_house_values histogram and density curve
mhv_histogram <- ggplot(data, aes(x = median_house_value/1000)) +
  geom_histogram(aes(y =after_stat(density)), fill= color_1, color="black", alpha=1, bins = 1
5) +
  geom_density(color= color_2, linewidth =0.8)+
  geom_vline(aes(xintercept = mean(median_house_value/1000)), color="blue",
             linetype="dashed", linewidth=1) +
  geom_vline(aes(xintercept = median(median_house_value/1000)), color="red",
             linetype="dashed", linewidth=1) +
  labs(x ="Median House Value (in 1000 $)", y="Count")+
  ggtitle("Histogram of Median house values")

# median_house_values boxplot
mhv_boxplot <- ggplot(data, aes(x = median_house_value/1000)) +
  geom_boxplot(fill = color_1) +
  geom_point(aes(x= mean(median_house_value)/1000, y=0), color="blue")+
  labs(x ="Median House Value (in 1000 $)", y = " ")+
  ggtitle("Boxplot of Median house values")

plot_grid(mhv_histogram, mhv_boxplot, nrow = 1)
```



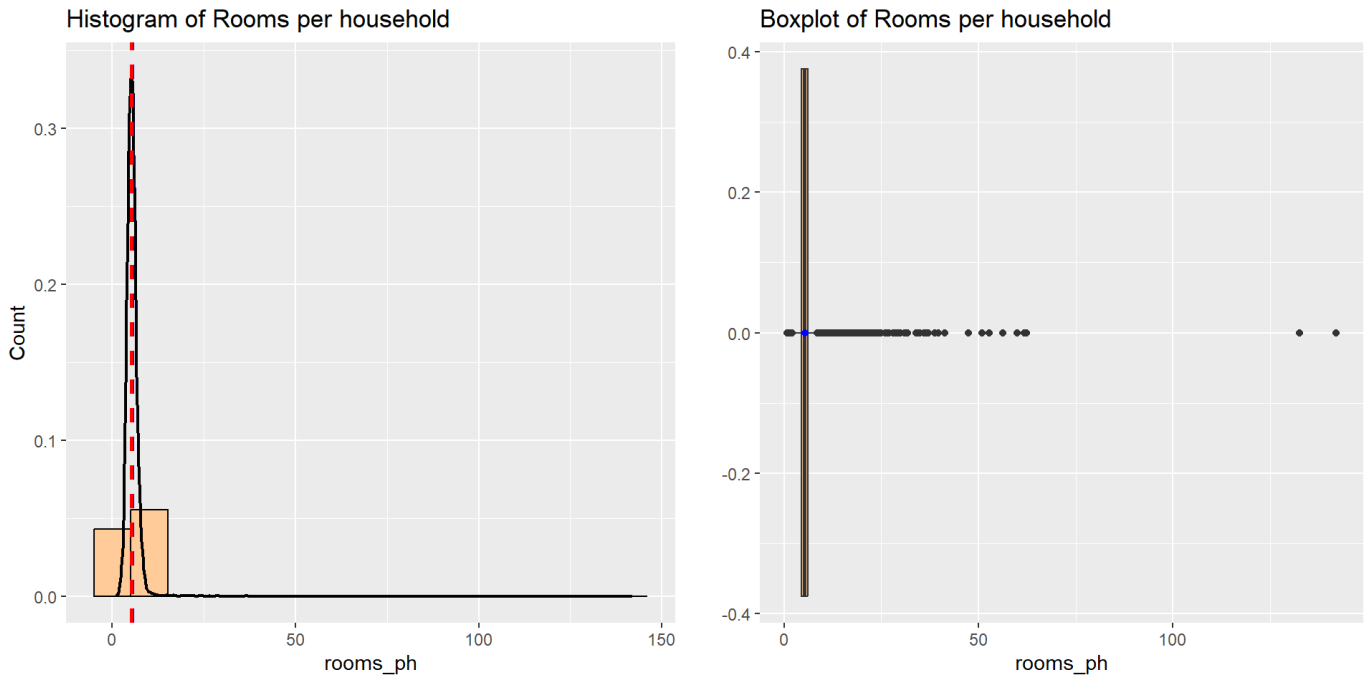
As it can be seen from the plots above, the distribution is asymmetrical and multimodal (?). Since mean (blue line) is greater than median (red line), the graph shows positive skewness. Furthermore, there are data with high values: they could be anomalous data.

```
# housing_median_age histogram and density curve
hma_histogram <- ggplot(data, aes(x = housing_median_age)) +
  geom_histogram(aes(y =after_stat(density)), fill=color_1, color="black", alpha=1, bins = 1
5) +
  geom_density(color= color_2, linewidth =0.8)+
  geom_vline(aes(xintercept = mean(housing_median_age)), color="blue",
             linetype="dashed", linewidth=1) +
  geom_vline(aes(xintercept = median(housing_median_age)), color="red",
             linetype="dashed", linewidth=1) +
  labs(x ="housing_median_age", y="Count")+
  ggtitle("Histogram of Housing median age")

# housing_median_age boxplot
hma_boxplot <- ggplot(data, aes(x = housing_median_age)) +
  geom_boxplot(fill = color_1) +
  labs(x ="housing_median_age", y = " ")+
  geom_point(aes(x= mean(housing_median_age), y=0), color="blue")+
  ggtitle("Boxplot of Housing median age")

plot_grid(hma_histogram, hma_boxplot, nrow = 1)
```



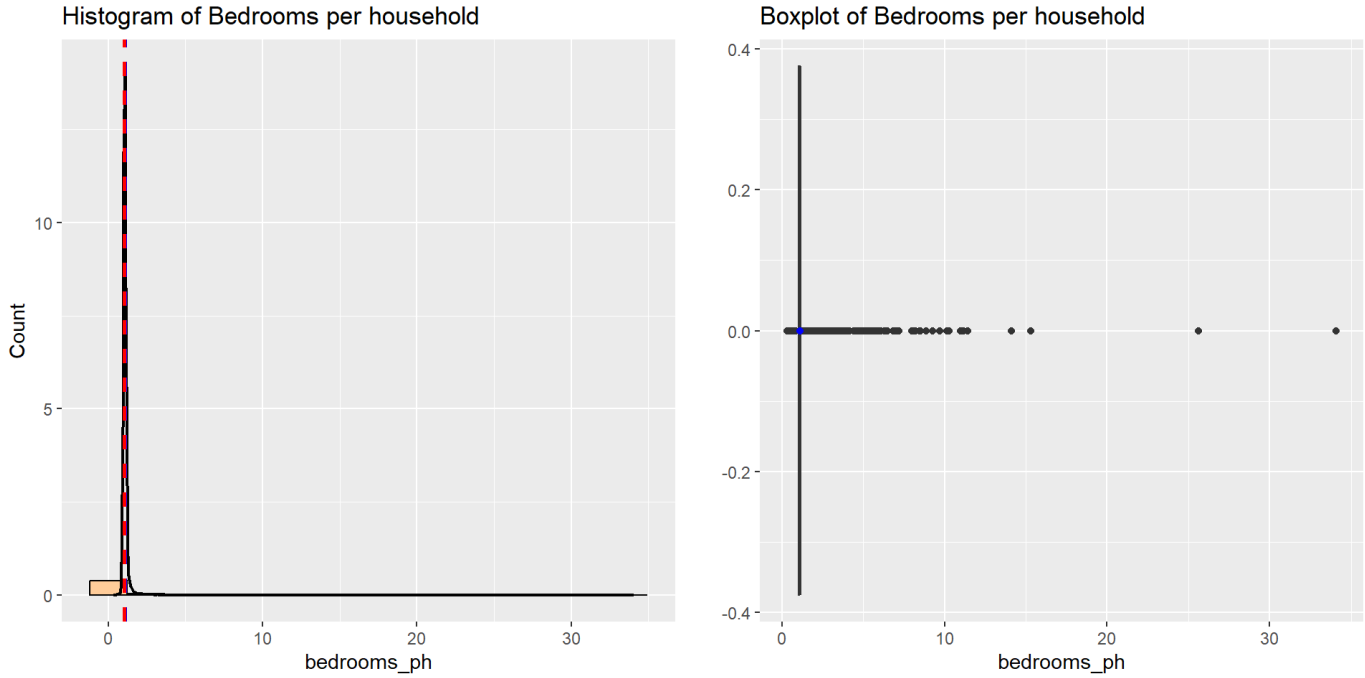
Housing median age variable chart shows multimodal distribution. We are going to check it again after the outliers removal.

```
# rooms_ph histogram and density curve
rph_histogram <- ggplot(data, aes(x = rooms_ph)) +
  geom_histogram(aes(y =after_stat(density)), fill=color_1, color="black", alpha=1, bins = 1
5) +
  geom_density(color= color_2, linewidth =0.8)+
  geom_vline(aes(xintercept = mean(rooms_ph)), color="blue",linetype="dashed", linewidth=1) +
  geom_vline(aes(xintercept = median(rooms_ph)), color="red",linetype="dashed", linewidth=1)
+
  labs(x ="rooms_ph", y="Count")+
  ggtitle("Histogram of Rooms per household")

# rooms_ph boxplot
rph_boxplot <- ggplot(data, aes(x = rooms_ph)) +
  geom_boxplot(fill = color_1) +
  geom_point(aes(x= mean(rooms_ph), y=0), color="blue")+
  labs(x ="rooms_ph", y = " ")+
  ggtitle("Boxplot of Rooms per household")

plot_grid(rph_histogram, rph_boxplot, nrow = 1 )
```



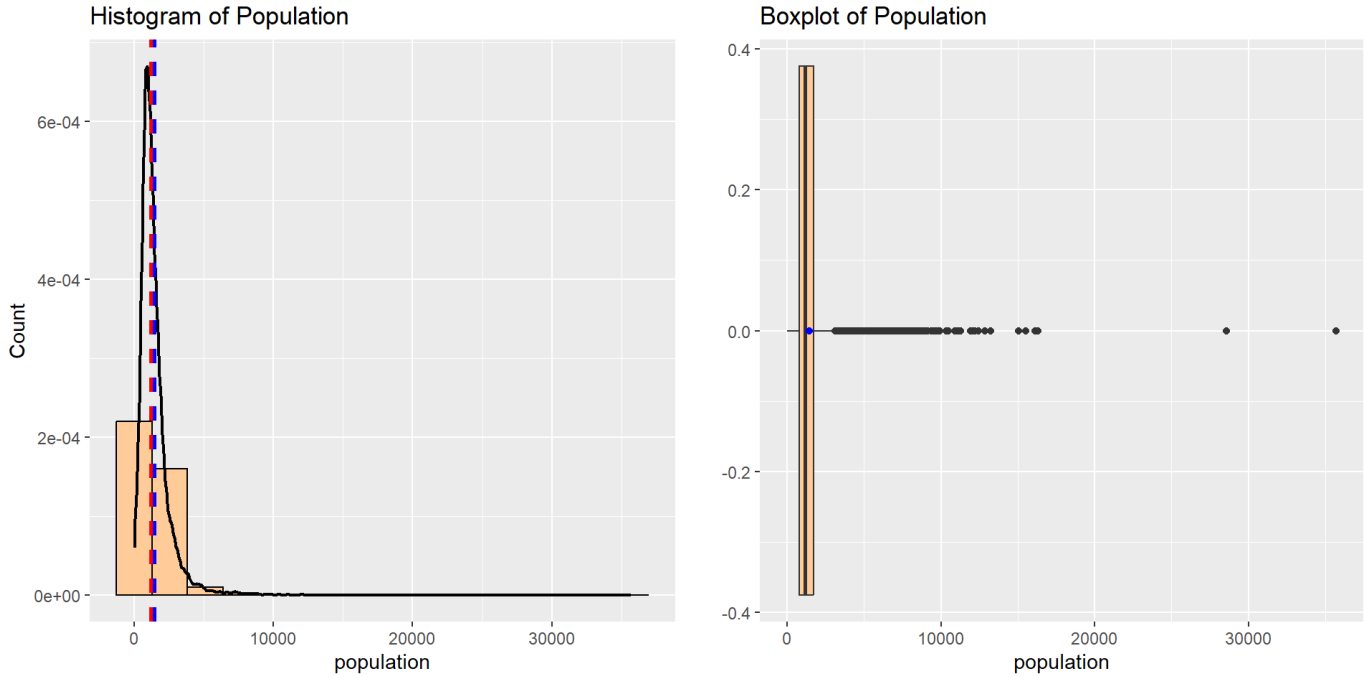
The distribution of rooms_ph variable shows a strong skewness since it has a long right tail due to the presence of outliers. Analyzing boxplot, in fact, it can be seen that there are lots of value far from the right tail of the graph.

```
# bedrooms_ph histogram and density curve
bph_histogram <- ggplot(data, aes(x = bedrooms_ph)) +
  geom_histogram(aes(y =after_stat(density)), fill=color_1, color="black", alpha=1, bins = 1
5) +
  geom_density(color= color_2, linewidth =0.8)+
  geom_vline(aes(xintercept = mean(bedrooms_ph)), color="blue",
             linetype="dashed", linewidth=1) +
  geom_vline(aes(xintercept = median(bedrooms_ph)), color="red",
             linetype="dashed", linewidth=1) +
  labs(x ="bedrooms_ph", y="Count")+
  ggtitle("Histogram of Bedrooms per household")

# bedrooms_ph boxplot
bph_boxplot <- ggplot(data, aes(x = bedrooms_ph)) +
  geom_boxplot(fill = color_1) +
  geom_point(aes(x= mean(bedrooms_ph), y=0), color="blue")+
  labs(x ="bedrooms_ph", y = " ")+
  ggtitle("Boxplot of Bedrooms per household")

plot_grid(bph_histogram, bph_boxplot, nrow = 1)
```



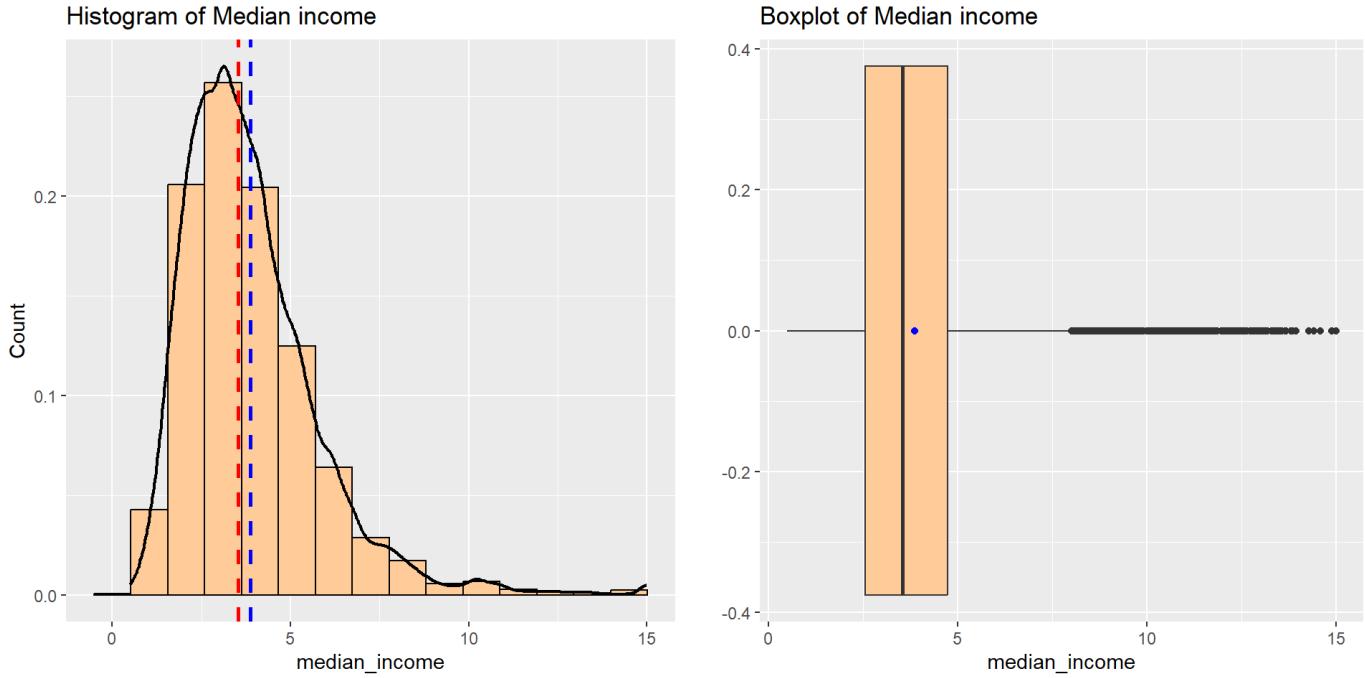
As rooms_ph, bedrooms_ph has a strong positive skewness too. Even in this case, it can be deduced by the boxplot and by the right tail of density curve that there are outliers.

```
# population histogram and density curve
pop_histogram <- ggplot(data, aes(x = population)) +
  geom_histogram(aes(y =after_stat(density)), fill=color_1, color="black", alpha=1, bins = 1
5) +
  geom_density(color= color_2, linewidth =0.8)+
  geom_vline(aes(xintercept = mean(population)), color="blue",
             linetype="dashed", linewidth=1) +
  geom_vline(aes(xintercept = median(population)), color="red",
             linetype="dashed", linewidth=1) +
  labs(x ="population", y="Count")+
  ggtitle("Histogram of Population")

# population boxplot
pop_boxplot <- ggplot(data, aes(x = population)) +
  geom_boxplot(fill = color_1) +
  geom_point(aes(x= mean(population), y=0), color="blue")+
  labs(x ="population", y = " ")+
  ggtitle("Boxplot of Population")

plot_grid(pop_histogram, pop_boxplot, nrow = 1)
```



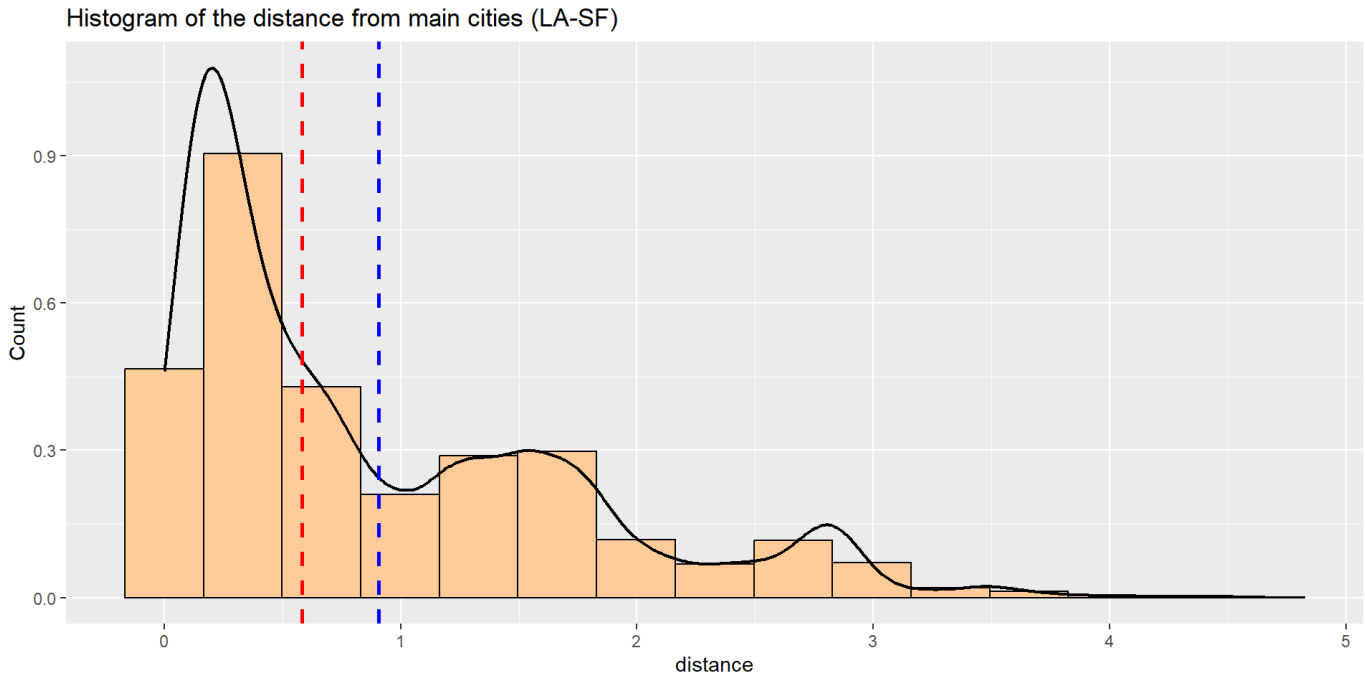
Population distribution shows positive skewness too. It means that population variable also suffers from outliers.

```
# median_income and density curve
mi_histogram <- ggplot(data, aes(x = median_income)) +
  geom_histogram(aes(y =after_stat(density)), fill=color_1, color="black", alpha=1, bins = 1
5) +
  geom_density(color= color_2, linewidth =0.8)+
  geom_vline(aes(xintercept = mean(median_income)), color="blue",
             linetype="dashed", linewidth=1) +
  geom_vline(aes(xintercept = median(median_income)), color="red",
             linetype="dashed", linewidth=1) +
  labs(x ="median_income", y="Count")+
  ggtitle("Histogram of Median income")

# boxplot median_income
mi_boxplot <- ggplot(data, aes(x = median_income)) +
  geom_boxplot(fill = color_1) +
  geom_point(aes(x= mean(median_income), y=0), color="blue")+
  labs(x ="median_income", y = " ")+
  ggtitle("Boxplot of Median income")

plot_grid(mi_histogram, mi_boxplot, nrow = 1)
```

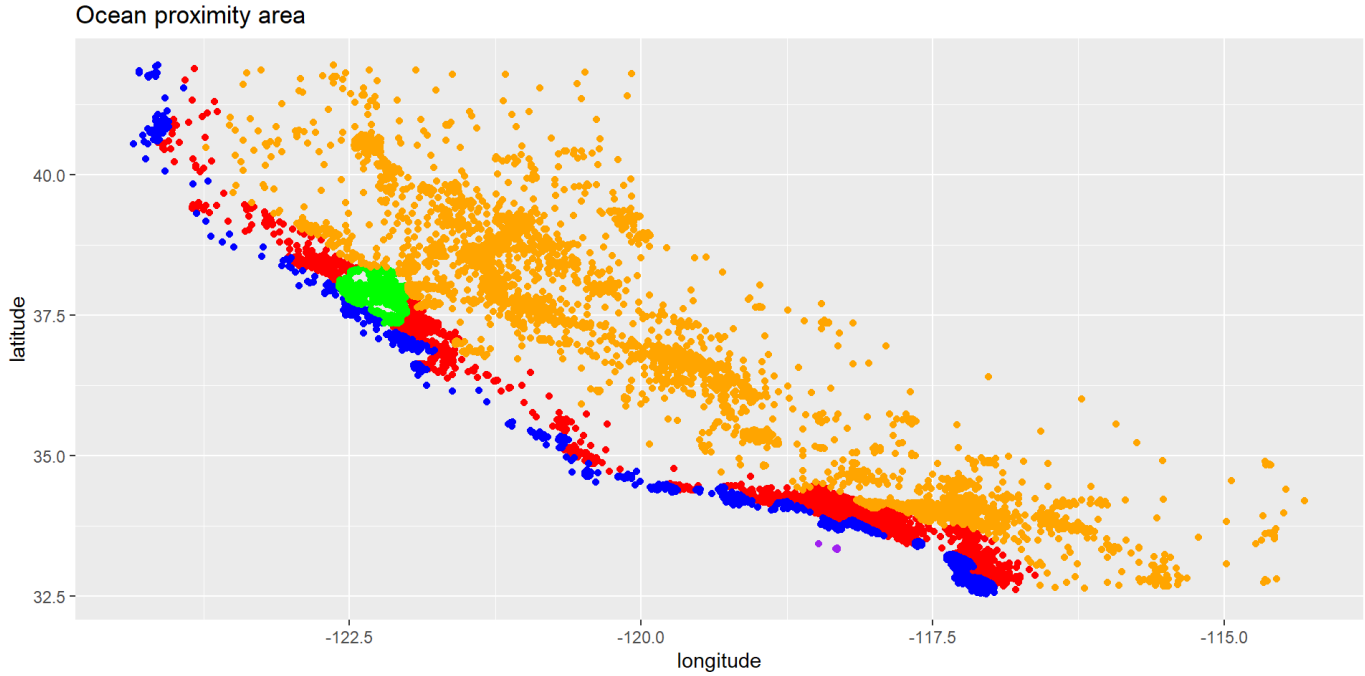


Median income variable has low positive skewness, but it suffers from outliers presence too. We are going to check if after outliers elimination it becomes less asymmetrical.

```
# distance histogram and density curve
ggplot(data, aes(x = distance)) +
  geom_histogram(aes(y =after_stat(density)), fill=color_1, color="black", alpha=1, bins = 1
5) +
  geom_density(color= color_2, linewidth =0.8)+
  geom_vline(aes(xintercept = mean(distance)), color="blue",linetype="dashed", linewidth=1) +
  geom_vline(aes(xintercept = median(distance)), color="red",linetype="dashed", linewidth=1)
+
  labs(x ="distance", y="Count")+
  ggtitle("Histogram of the distance from main cities (LA-SF)")
```

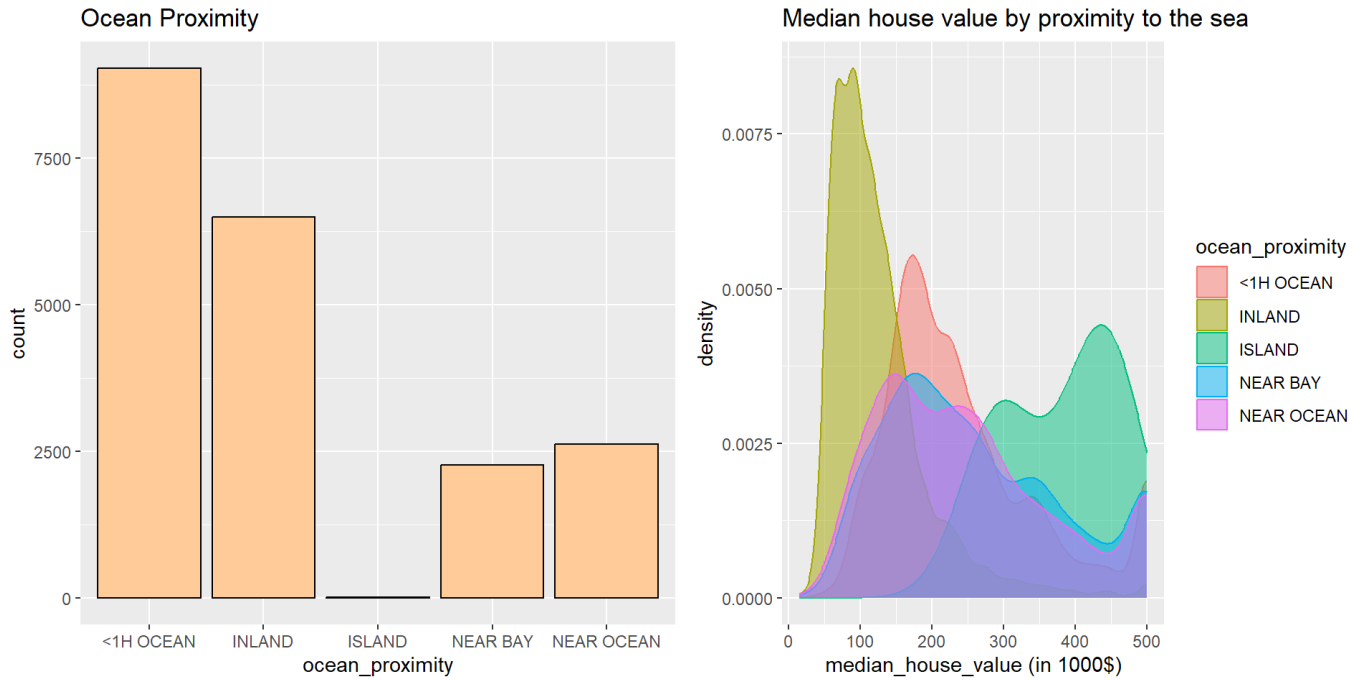



```
# plot coordinate
ggplot() +
  geom_point(data=cleaned_data[cleaned_data$ocean_proximity == "<1H OCEAN", ] ,
             mapping = aes(x = longitude, y = latitude), color="red") +
  geom_point(data=cleaned_data[cleaned_data$ocean_proximity == "NEAR OCEAN", ] ,
             mapping = aes(x = longitude, y = latitude), color="blue") +
  geom_point(data=cleaned_data[cleaned_data$ocean_proximity == "NEAR BAY", ] ,
             mapping = aes(x = longitude, y = latitude), color="green") +
  geom_point(data=cleaned_data[cleaned_data$ocean_proximity == "INLAND", ] ,
             mapping = aes(x = longitude, y = latitude), color="orange") +
  geom_point(data=cleaned_data[cleaned_data$ocean_proximity == "ISLAND", ] ,
             mapping = aes(x = longitude, y = latitude), color="purple")+
  ggtitle("Ocean proximity area")
```



The chart above shows the spread of blocks according to ocean proximity labels: - green: “NEAR BAY”; - orange: “INLAND”; - red: “<1H OCEAN”; - purple: “ISLAND”; - blue: “NEAR OCEAN”.

```
# barplot ocean_proximity
bar_op <- ggplot(data, aes(x = ocean_proximity)) +
  geom_bar(color="black", fill=color_1)+
  ggtitle("Ocean Proximity")

# plot densities
dens_op <- ggplot(data, aes(x = median_house_value / 1000, color = ocean_proximity,
                            fill = ocean_proximity)) +
  geom_density(alpha = 0.5)+
  labs(x ="median_house_value (in 1000$)", y = "density")+
  ggtitle("Median house value by proximity to the sea")

plot_grid(bar_op, dens_op, nrow = 1)
```



The barplot shows the spread of blocks according to ocean proximity labels. “ISLAND” has very few blocks and lots of blocks correspond to “<1H OCEAN” label.

Then we plot density for each label of “ocean proximity” with respect to the response variable median house value. It can be seen that “INLAND” blocks have lowest median house values and “ISLAND” blocks have highest median house values. Since we know that blocks under the label “ISLAND” are really few, we can deduct that those blocks have houses with very high values, that’s why median house value in ISLAND blocks is greater that blocks in different areas.

Outliers

Previous charts have highlighted the presence of outliers. Since the dataset has more than 20 thousands observations, it could be useful to use a criterion to determine which data have to be consider as outliers in order to remove them. We opted for a rule connected with the notion of Interquartile Range:

$$IQR = Q_3 - Q_1$$

Outliers are data points which fall below:

$$Q_1 - 1,5\ IQR$$

Or above:

$$Q_3 + 1,5\ IQR$$

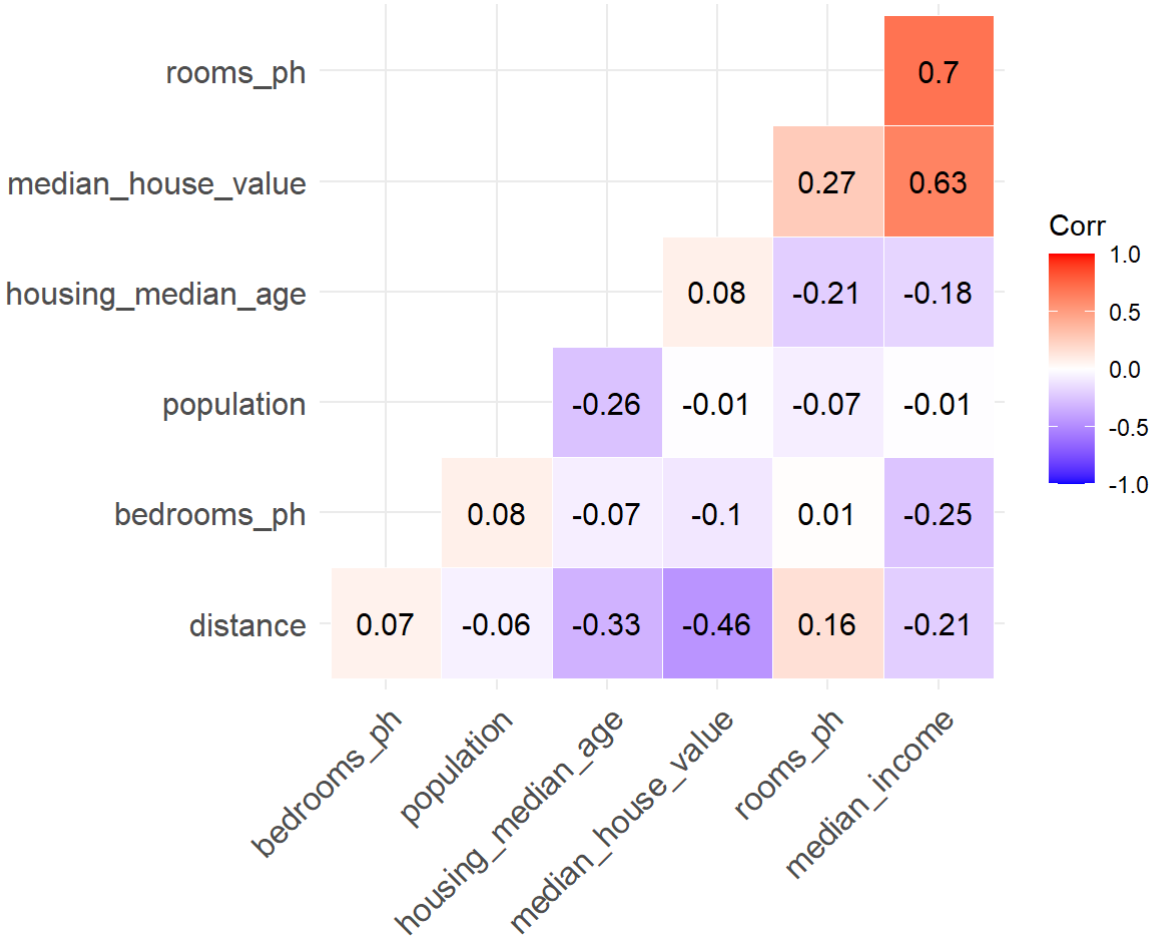
So we proceed with outliers detection and removal.

```
# Outliers in bedrooms_ph
outliers_b <- data$bedrooms_ph > IQR(data$bedrooms_ph)*1.5+quantile(bedrooms_ph, 0.75) |
  data$bedrooms_ph < quantile(bedrooms_ph, 0.25) - IQR(data$bedrooms_ph) * 1.5

# Outliers in rooms_ph
outliers_r <- data$rooms_ph > IQR(data$rooms_ph)*1.5+quantile(data$rooms_ph, 0.75) |
  data$rooms_ph < quantile(data$rooms_ph, 0.25) - IQR(data$rooms_ph) * 1.5

# Outliers in housing_median_age
outliers_hma <- data$housing_median_age > IQR(data$housing_median_age)*1.5+
  quantile(data$housing_median_age, 0.75) |
  data$housing_median_age < quantile(data$housing_median_age, 0.25) -
  IQR(data$housing_median_age) * 1.5

# Outliers in population
outliers_p <- data$population > IQR(data$population)*1.5+quantile(data$population, 0.75) |
  data$population < quantile(data$population, 0.25) - IQR(data$population) * 1.5

# Outliers in median_income
outliers_mi <- data$median_income > IQR(data$median_income)*1.5+
  quantile(data$median_income, 0.75) |
  data$median_income < quantile(data$median_income, 0.25) - IQR(data$median_income) * 1.5

# Outliers in median_house_value
outliers_mhv <- data$median_house_value > IQR(data$median_house_value)*1.5+
  quantile(data$median_house_value, 0.75) |
  data$median_house_value < quantile(data$median_house_value, 0.25)-
  IQR(data$median_house_value)*1.5

# Deleting outliers in order to update the dataframe
data <- data[!(outliers_b|outliers_r|outliers_hma|outliers_p|outliers_mi|outliers_mhv), ]
```

Now the cleaned dataframe has 16665 observations and 8 variables.

Now we check again correlation matrix in order to see if correlation among variables improves deleting outliers.

```
data2 <- data[sapply(data, is.numeric)]
ggcorrplot(cor(data2), hc.order=TRUE, type = "lower", outline.col= "white", lab=TRUE)
```

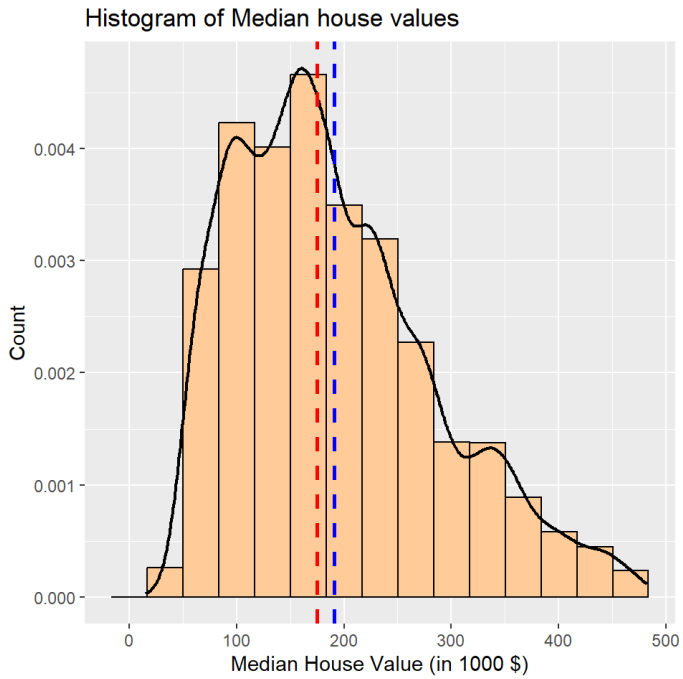


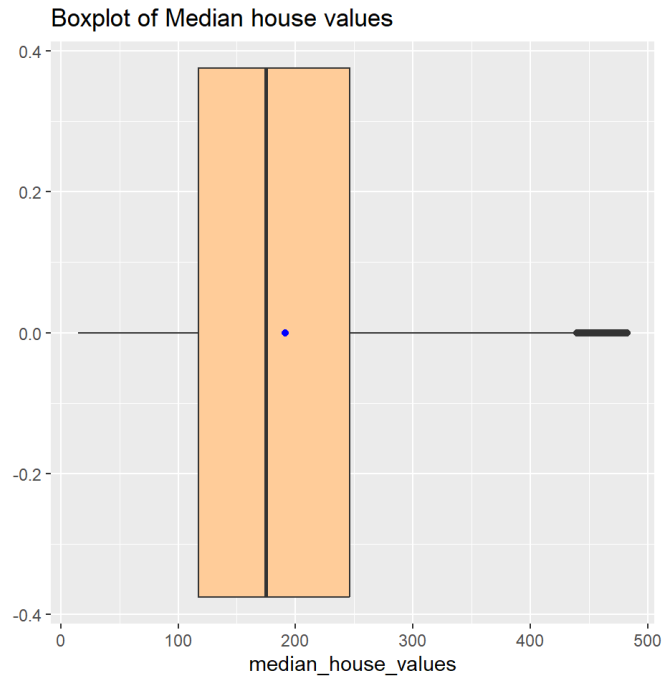
Before the outliers removal the correlation between “bedrooms_ph” and “rooms_ph” was 0.85. After outliers have been deleted, the correlation become 0.01. Correlation among “median income” and “rooms_ph” passes from 0.33 to 0.7, but it is not a critical value for correlation.

Second investigation on distribution

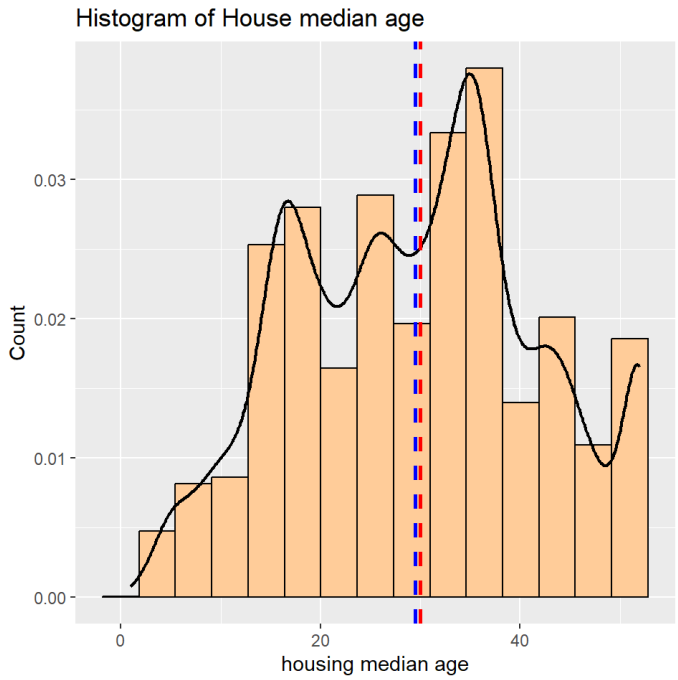
Now we check plots of each variable after outliers deletion in order to see changes on variables distributions.

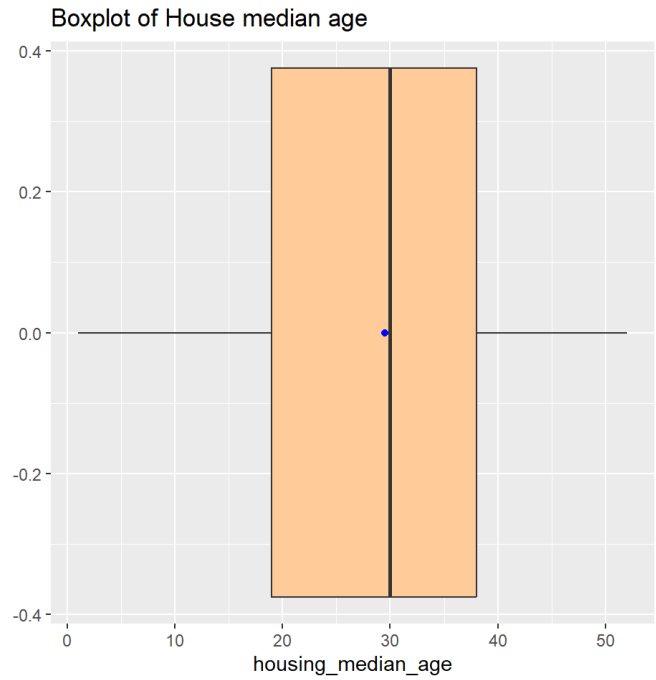
```
# median_house_values histogram and density curve
mhv_histogram <- ggplot(data, aes(x = median_house_value/1000)) +
  geom_histogram(aes(y =after_stat(density)), fill=color_1, color="black", alpha=1, bins = 1
5) +
  geom_density(color= color_2, linewidth =0.8)+
  geom_vline(aes(xintercept = mean(median_house_value/1000)), color="blue",
             linetype="dashed", linewidth=1) +
  geom_vline(aes(xintercept = median(median_house_value/1000)), color="red",
             linetype="dashed", linewidth=1) +
  labs(x ="Median House Value (in 1000 $)", y="Count")+
  ggtitle("Histogram of Median house values")

# median_house_values boxplot
mhv_boxplot <- ggplot(data, aes(x = median_house_value/1000)) +
  geom_boxplot(fill = color_1) +
  geom_point(aes(x= mean(median_house_value)/1000, y=0), color="blue")+
  labs(x ="median_house_values", y = " ")+
  ggtitle("Boxplot of Median house values")

plot_grid(mhv_histogram, mhv_boxplot, nrow = 1)
```



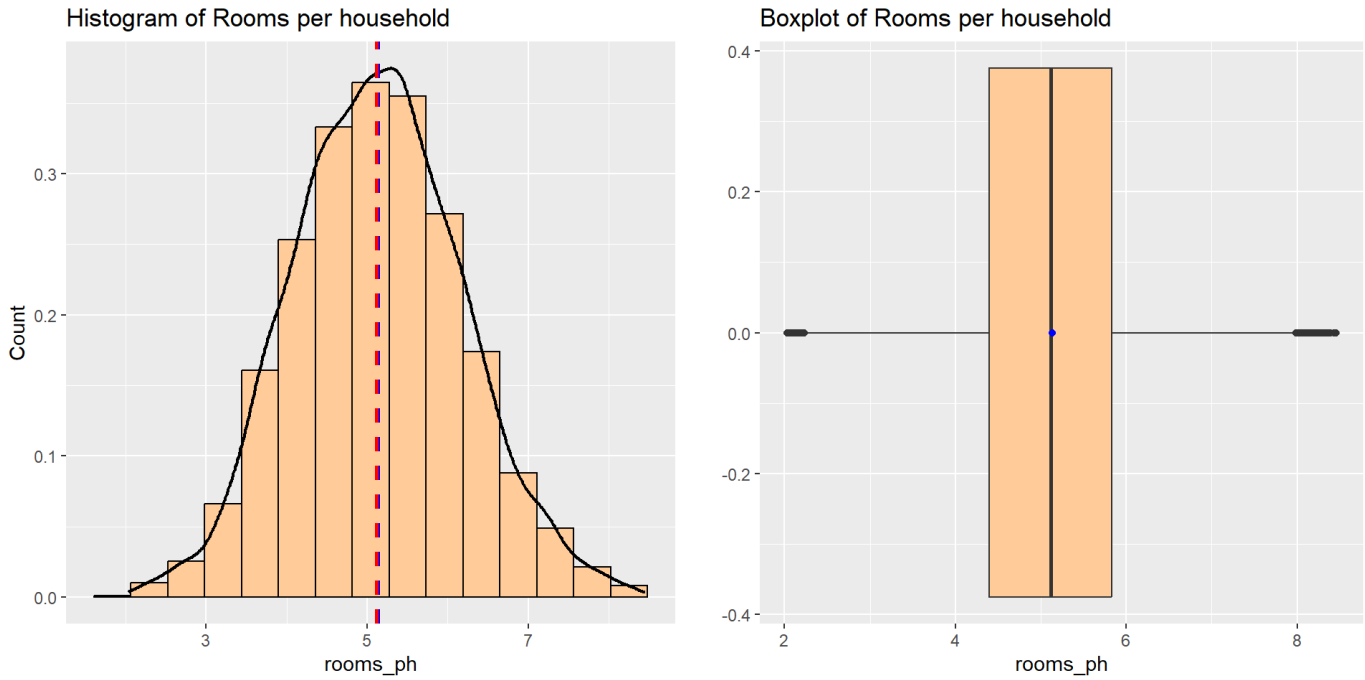



```
# housing_median_age histogram and density curve
hma_histogram <- ggplot(data, aes(x = housing_median_age)) +
  geom_histogram(aes(y =after_stat(density)), fill=color_1, color="black", alpha=1, bins = 1
5) +
  geom_density(color= color_2, linewidth =0.8)+
  geom_vline(aes(xintercept = mean(housing_median_age)), color="blue",
             linetype="dashed", linewidth=1) +
  geom_vline(aes(xintercept = median(housing_median_age)), color="red",
             linetype="dashed", linewidth=1) +
  labs(x ="housing median age", y="Count")+
  ggtitle("Histogram of House median age")

# housing_median_age boxplot
hma_boxplot <- ggplot(data, aes(x = housing_median_age)) +
  geom_boxplot(fill = color_1) +
  geom_point(aes(x= mean(housing_median_age), y=0), color="blue")+
  labs(x ="housing_median_age", y = " ")+
  ggtitle("Boxplot of House median age")

plot_grid(hma_histogram, hma_boxplot, nrow = 1)
```



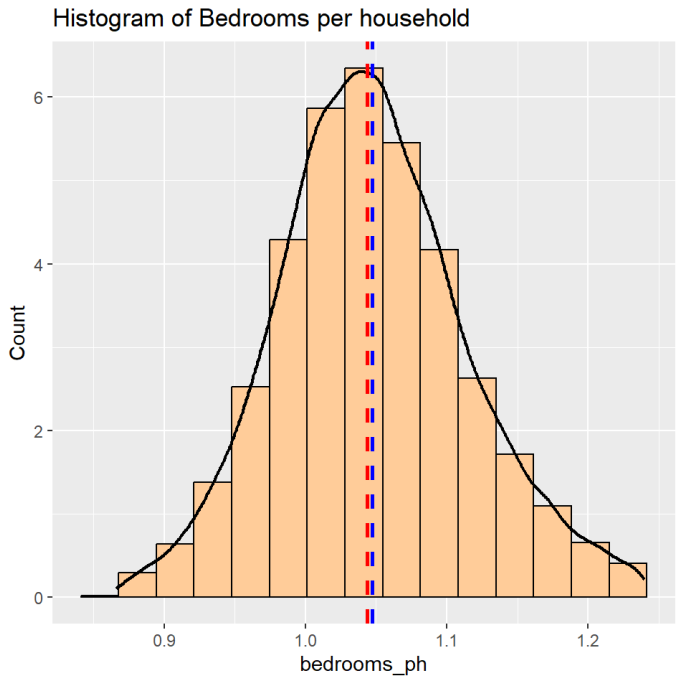


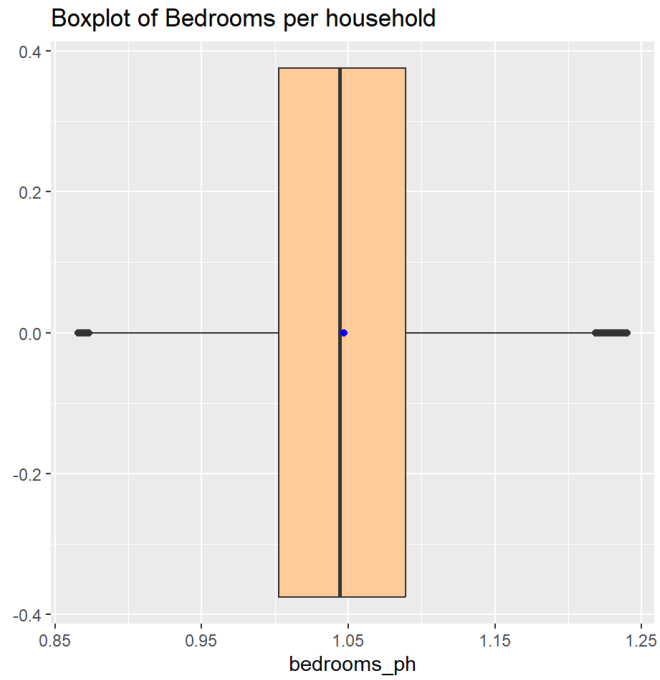

```
# rooms_ph histogram and density curve
rph_histogram <- ggplot(data, aes(x = rooms_ph)) +
  geom_histogram(aes(y =after_stat(density)), fill=color_1, color="black", alpha=1, bins = 1
5) +
  geom_density(color= color_2, linewidth =0.8)+
  geom_vline(aes(xintercept = mean(rooms_ph)), color="blue",linetype="dashed", linewidth=1) +
  geom_vline(aes(xintercept = median(rooms_ph)), color="red",linetype="dashed", linewidth=1)
+
  labs(x ="rooms_ph", y="Count")+
  ggtitle("Histogram of Rooms per household")

# rooms_ph boxplot
rph_boxplot <- ggplot(data, aes(x = rooms_ph)) +
  geom_boxplot(fill = color_1) +
  geom_point(aes(x= mean(rooms_ph), y=0), color="blue")+
  labs(x ="rooms_ph", y = " ")+
  ggtitle("Boxplot of Rooms per household")

plot_grid(rph_histogram, rph_boxplot, nrow = 1)
```



It is possible to be seen that after the outliers elimination, rooms_ph variable shows a symmetrical distribution since median is quite equal to mean. Boxplots shows the same results: outliers removal had made the distribution much more symmetrical.

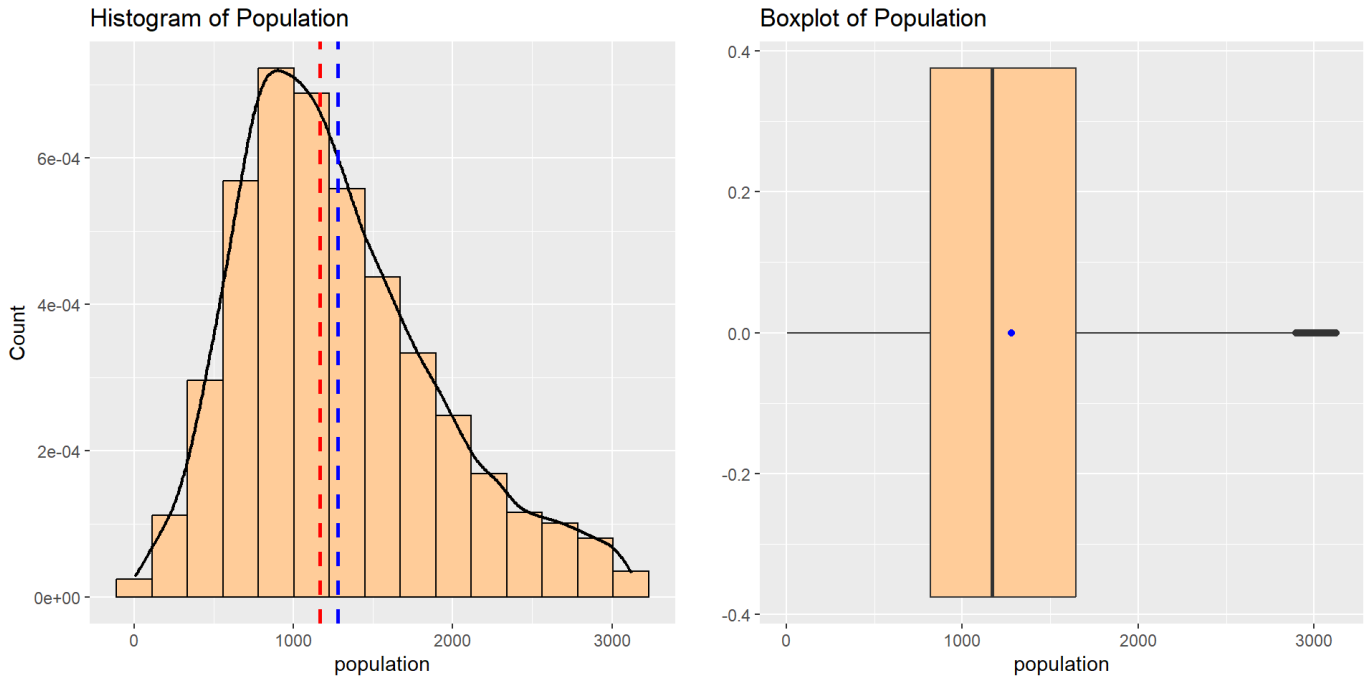
```
# bedrooms_ph histogram and density curve
bph_histogram <- ggplot(data, aes(x = bedrooms_ph)) +
  geom_histogram(aes(y =after_stat(density)), fill=color_1, color="black", alpha=1, bins = 1
5) +
  geom_density(color= color_2, linewidth =0.8)+
  geom_vline(aes(xintercept=mean(bedrooms_ph)), color="blue",linetype="dashed", linewidth=1)
+
  geom_vline(aes(xintercept=median(bedrooms_ph)), color="red",linetype="dashed", linewidth=1)
+
  labs(x ="bedrooms_ph", y="Count")+
  ggtitle("Histogram of Bedrooms per household")

# bedrooms_ph boxplot
bph_boxplot <- ggplot(data, aes(x = bedrooms_ph)) +
  geom_boxplot(fill = color_1) +
  geom_point(aes(x= mean(bedrooms_ph), y=0), color="blue")+
  labs(x ="bedrooms_ph", y = " ")+
  ggtitle("Boxplot of Bedrooms per household")

plot_grid(bph_histogram, bph_boxplot, nrow = 1)
```





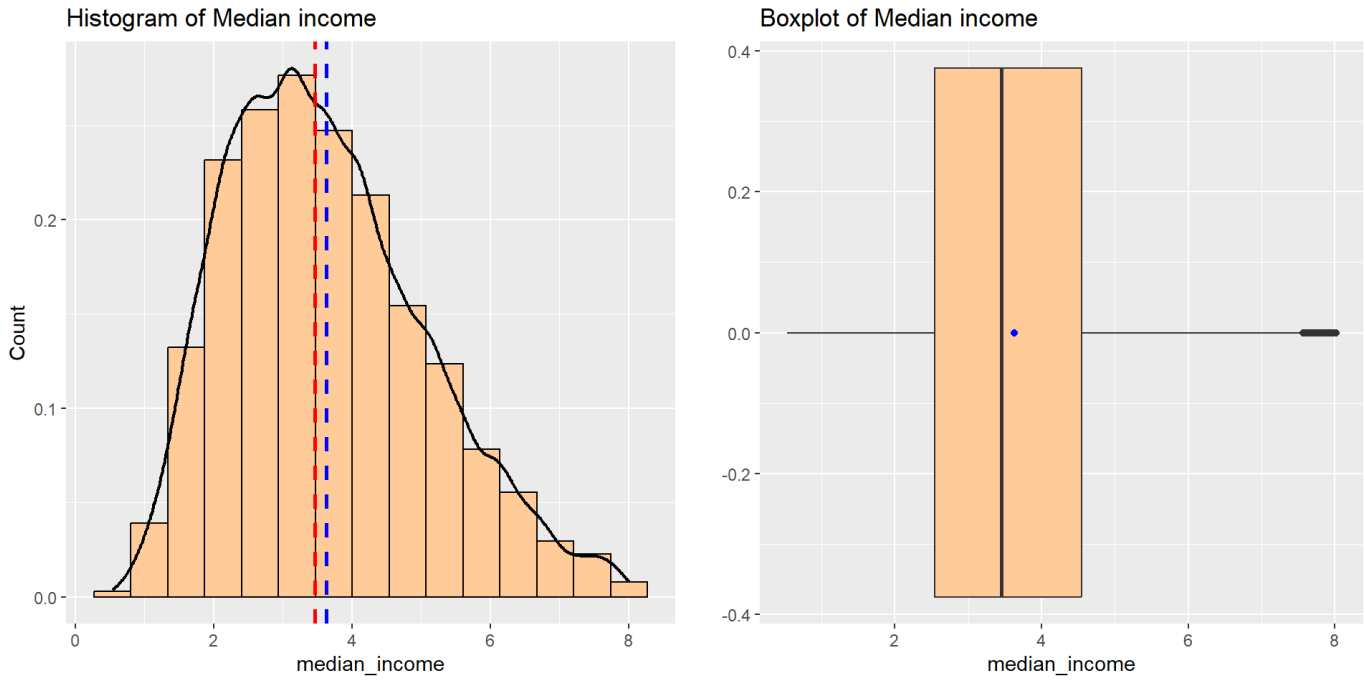
The same results reached for rooms_ph variable is shown in bedrooms_ph plots: outliers removal has led to a symmetrical distribution of bedrooms_ph variable.

```
# population histogram and density curve
pop_histogram <- ggplot(data, aes(x = population)) +
  geom_histogram(aes(y =after_stat(density)), fill=color_1, color="black", alpha=1, bins = 1
5) +
  geom_density(color= color_2, linewidth =0.8)+
  geom_vline(aes(xintercept=mean(population)), color="blue",linetype="dashed", linewidth=1) +
  geom_vline(aes(xintercept=median(population)), color="red",linetype="dashed", linewidth=1)
+
  labs(x ="population", y="Count")+
  ggtitle("Histogram of Population")

# population boxplot
pop_boxplot <- ggplot(data, aes(x = population)) +
  geom_boxplot(fill = color_1) +
  geom_point(aes(x= mean(population), y=0), color="blue")+
  labs(x ="population", y = " ")+
  ggtitle("Boxplot of Population")

plot_grid(pop_histogram, pop_boxplot, nrow = 1)
```



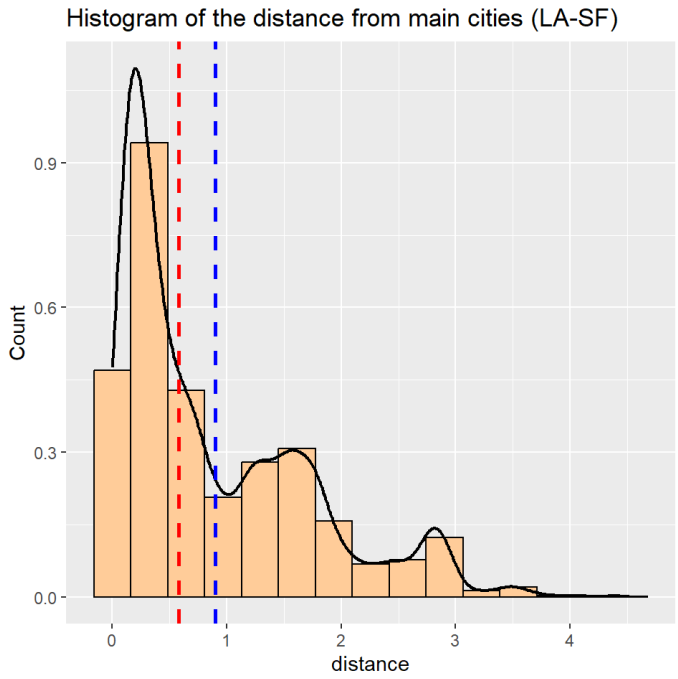
The variable population was strongly asymmetric with long tail on the right: once outliers have been deleted, the skewness has been significantly reduced.

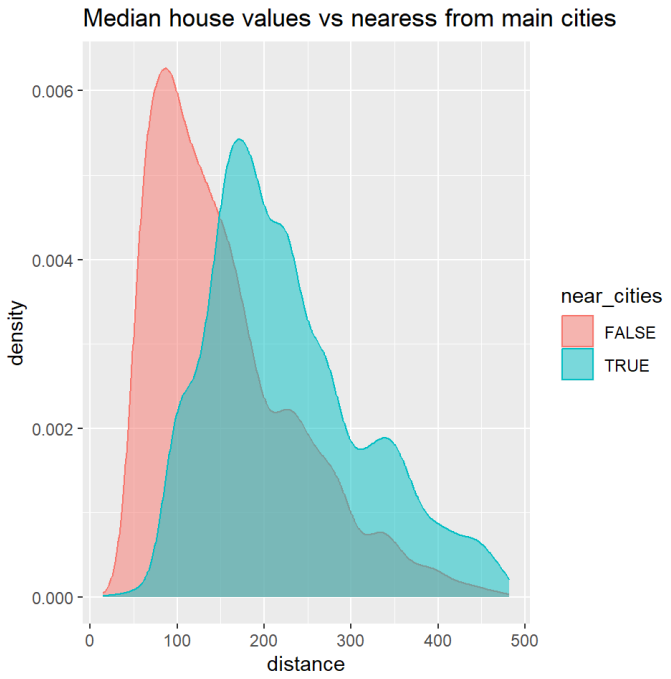
```
# median_income and density curve
mi_histogram <- ggplot(data, aes(x = median_income)) +
  geom_histogram(aes(y =after_stat(density)), fill=color_1, color="black", alpha=1, bins = 1
5) +
  geom_density(color= color_2, linewidth =0.8)+
  geom_vline(aes(xintercept = mean(median_income)), color="blue",
             linetype="dashed", linewidth=1) +
  geom_vline(aes(xintercept = median(median_income)), color="red",
             linetype="dashed", linewidth=1) +
  labs(x ="median_income", y="Count")+
  ggtitle("Histogram of Median income")

# boxplot median_income
mi_boxplot <- ggplot(data, aes(x = median_income)) +
  geom_boxplot(fill = color_1) +
  geom_point(aes(x= mean(median_income), y=0), color="blue")+
  labs(x ="median_income", y = " ")+
  ggtitle("Boxplot of Median income")

plot_grid(mi_histogram, mi_boxplot, nrow = 1)
```



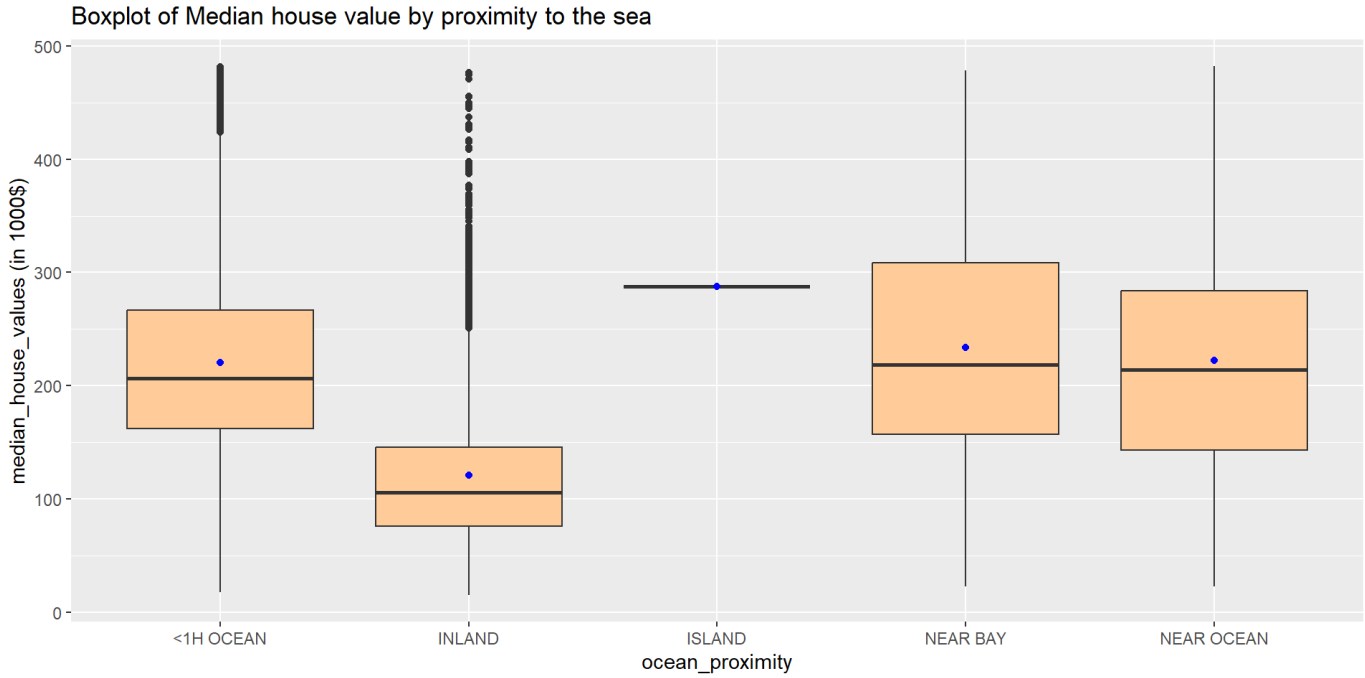
Median income variable had a long right tail and the boxplot showed a huge amount of anomalous data. Now the skewness has been reduced due the outliers removal. In fact, even if the distribution is still positively asymmetric, the right tail is shorter and the boxplots shows few data greater than the right tail.

```
# We are going to split the data according to "distance" variable
# using median distance as threshold
med_dist <- median(data$distance)
near_cities <- data$distance <= med_dist

# densities
dens_dist <- ggplot(data, aes(x = median_house_value / 1000, color = near_cities,
                              fill = near_cities)) +
  geom_density(alpha = 0.5) +
  labs(x ="distance", y = "density")+
  ggtitle("Median house values vs nearess from main cities")

# distance histogram and density curve
dist_histogram <- ggplot(data, aes(x = distance)) +
  geom_histogram(aes(y =after_stat(density)), fill=color_1, color="black", alpha=1, bins = 1
5) +
  geom_density(color= color_2, linewidth =0.8)+
  geom_vline(aes(xintercept = mean(distance)), color="blue",linetype="dashed", linewidth=1) +
  geom_vline(aes(xintercept = median(distance)), color="red",linetype="dashed", linewidth=1)
+
  labs(x ="distance", y="Count")+
  ggtitle("Histogram of the distance from main cities (LA-SF)")

plot_grid(dist_histogram, dens_dist, nrow = 1)
```






```
# boxplot ocean_proximity (without outliers)
box_op <- ggplot(data, aes(x = ocean_proximity, y = median_house_value / 1000)) +
  geom_boxplot(fill=color_1) +
  stat_summary(fun=mean, geom="point", color="blue")+
  labs(x ="ocean_proximity", y = "median_house_values (in 1000$)")+
  ggtitle("Boxplot of Median house value by proximity to the sea")

plot_grid(box_op)
```



It is possible to be seen that the boxplot corresponding to “ISLAND” label shows the presence of a unique value. In order to simplify our analysis, we decide to check which is the unique “ISLAND” value and to give it the label “NEAR OCEAN”. Then we plot the updated charts.

```
# checking what is the data with "ISLAND" label
which(data$ocean_proximity == "ISLAND")
```

```
## [1] 6895
```

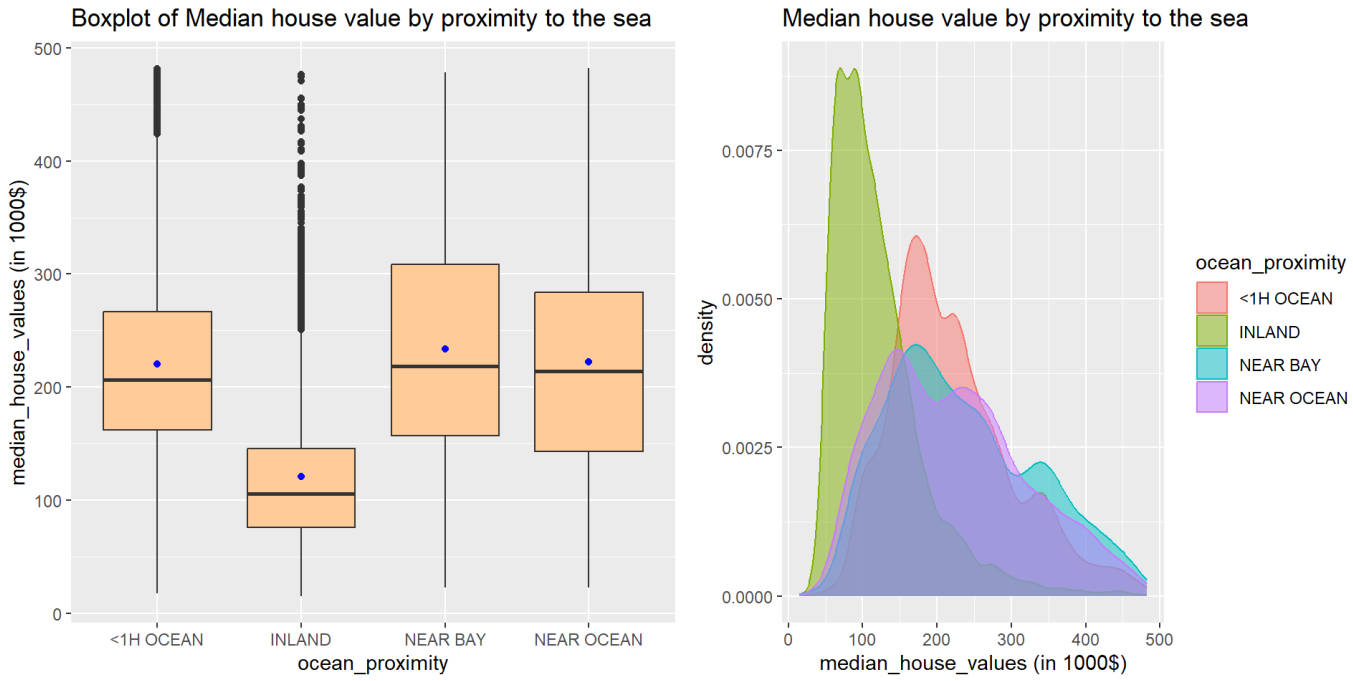
```
data$ocean_proximity[6895]
```

```
## [1] "ISLAND"
```

```
# Assuming your dataset is named 'data' and the label column is named 'label'
data$ocean_proximity[data$ocean_proximity == "ISLAND"] <- "NEAR OCEAN"

# checking if the old ISLAND value has now NEAR OCEAN label
data$ocean_proximity[6895]
```

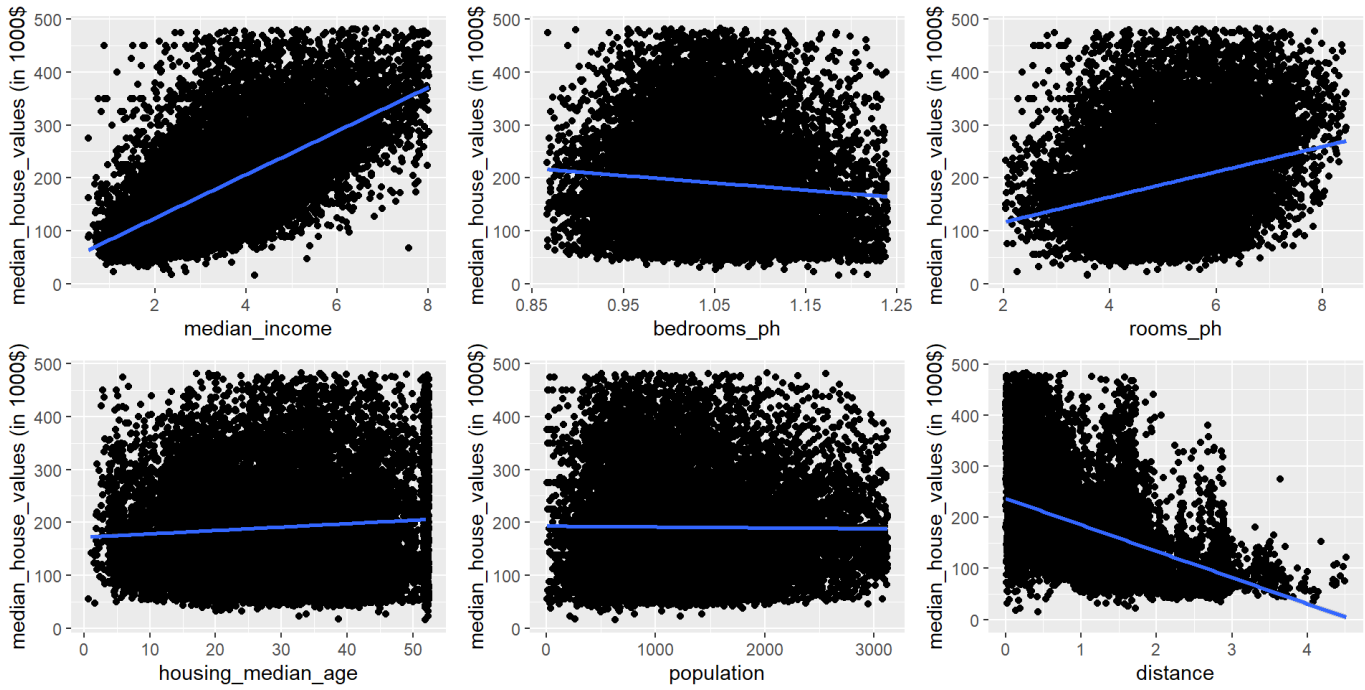
```
## [1] "NEAR OCEAN"
```

```
# boxplot ocean_proximity (without outliers)
box_op <- ggplot(data, aes(x = ocean_proximity, y = median_house_value / 1000)) +
  geom_boxplot(fill=color_1) +
  stat_summary(fun=mean, geom="point", color="blue")+
  labs(x ="ocean_proximity", y = "median_house_values (in 1000$)")+
  ggtitle("Boxplot of Median house value by proximity to the sea")

# plot densities
dens_op <- ggplot(data, aes(x = median_house_value / 1000, color = ocean_proximity,
                            fill = ocean_proximity)) +
  geom_density(alpha = 0.5)+
  labs(x ="median_house_values (in 1000$)", y = "density")+
  ggtitle("Median house value by proximity to the sea")

plot_grid(box_op, dens_op, nrow = 1)
```



Next, we investigate the relations between the response variable “median house value” and the other numerical features.

```
g1 <- ggplot(data, aes(x = median_income, y = median_house_value / 1000)) +
  geom_jitter() +
  labs(y ="median_house_values (in 1000$)")+
  geom_smooth(method = lm, formula = y~x)

g2 <- ggplot(data, aes(x = bedrooms_ph, y = median_house_value / 1000)) +
  geom_jitter() +
  labs(y ="median_house_values (in 1000$)")+
  geom_smooth(method = lm, formula = y~x)

g3 <- ggplot(data, aes(x = rooms_ph, y = median_house_value / 1000)) +
  geom_jitter() +
  labs(y ="median_house_values (in 1000$)")+
  geom_smooth(method = lm, formula = y~x)

g4 <- ggplot(data, aes(x = housing_median_age, y = median_house_value / 1000)) +
  geom_jitter() +
  labs(y ="median_house_values (in 1000$)")+
  geom_smooth(method = lm, formula = y~x)

g5 <- ggplot(data, aes(x = population, y = median_house_value / 1000)) +
  geom_jitter() +
  labs(y ="median_house_values (in 1000$)")+
  geom_smooth(method = lm, formula = y~x)

g6 <- ggplot(data, aes(x = distance, y = median_house_value / 1000)) +
  geom_jitter() +
  labs(y ="median_house_values (in 1000$)")+
  geom_smooth(method = lm, formula = y~x)

plot_grid(g1, g2, g3, g4, g5, g6, nrow = 2)
```



The plot above show the quite linear relation between each feature and the response variable.

Model Data & Analysis: Multiple Linear Regression Model

In this section we will use a multiple linear regression model to consider the effect of features on the response variable “median house value”.

Multiple linear regression is a statistical modeling technique used to examine the relationship between a dependent variable and multiple independent variables. It extends the concept of simple linear regression by considering multiple predictors simultaneously. The goal is to find the best-fitting linear equation that predicts the dependent variable based on the values of the independent variables. In multiple linear regression, each independent variable is assigned a regression coefficient that represents its influence on the dependent variable, while controlling for the effects of other predictors. This allows for a more comprehensive analysis, capturing the combined effects of multiple factors on the outcome of interest.

Model preparation

Before the model computation, since our dataframe is composed by 7 numerical variables and a categorical one, data have to be modified considering the kind of variables. The categorical variable “ocean proximity” has four labels. It means that it is necessary to create a dummy variable for each label in order to add them in the model. Then, one on those dummies shall be considered as a reference: we decide to use “NEAR BAY” for that purpose.

```
# dividing the response variable by 1000 and updating the dataframe
data$median_house_value <- data$median_house_value/1000

# creating dummy variables for each label of the categorical variable "ocean proximity"
unique(data$ocean_proximity)
```

```
## [1] "NEAR BAY"   "<1H OCEAN"  "INLAND"     "NEAR OCEAN"
```

```
NEAR_BAY <- ifelse(data$ocean_proximity == 'NEAR BAY', 1, 0)
Min_1H_OCEAN <- ifelse(data$ocean_proximity == '<1H OCEAN', 1, 0)
INLAND <- ifelse(data$ocean_proximity == 'INLAND', 1, 0)
NEAR_OCEAN <- ifelse(data$ocean_proximity == 'NEAR OCEAN', 1, 0)

# updating dataframe with dummies
dataset_with_dummy <- cbind(data[,-c(7)], INLAND, Min_1H_OCEAN, NEAR_OCEAN)
```

The next step is building the model. We try to build a linear model with median house value as Y and all the other variables as features (X) and then we print coefficients of the model. The coefficients represent the estimates of the parameters that describe the relationship between the independent variables (or predictors) and the dependent variable. Each independent variable in the linear model has an associated coefficient that indicates how much the variable influences the value of the dependent variable, while accounting for the other predictors in the model.

The coefficients can be interpreted as follows:

Intercept coefficient: this is the coefficient associated with the constant or intercept independent variable. It represents the predicted value of the dependent variable when all other independent variables are zero.

Coefficients of the independent variables: these coefficients represent the estimated effect of the corresponding independent variable on the value of the dependent variable. A positive coefficient indicates that an increase in the independent variable is associated with an increase in the dependent variable, while a

negative coefficient indicates an inverse relationship.

```
# building the model
model <- lm(median_house_value ~ ., data = dataset_with_dummy)

# summary() to check p-value
summary(model)
```

```
##
## Call:
## lm(formula = median_house_value ~ ., data = dataset_with_dummy)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -353.60  -39.40   -8.79   28.26  398.42
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)        -4.459e+01  8.779e+00  -5.079 3.83e-07 ***
## distance           -2.139e+01  7.985e-01 -26.795  < 2e-16 ***
## bedrooms_ph         1.479e+02  7.656e+00  19.323  < 2e-16 ***
## rooms_ph           -1.114e+01  7.696e-01 -14.474  < 2e-16 ***
## housing_median_age  5.500e-01  4.592e-02  11.977  < 2e-16 ***
## population         -2.979e-03  7.852e-04  -3.794 0.000149 ***
## median_income       4.395e+01  6.033e-01  72.851  < 2e-16 ***
## INLAND             -4.627e+01  1.922e+00 -24.072  < 2e-16 ***
## Min_1H_OCEAN       -7.700e+00  1.596e+00  -4.823 1.42e-06 ***
## NEAR_OCEAN          2.214e+01  2.060e+00  10.746  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 60.05 on 16655 degrees of freedom
## Multiple R-squared:  0.5908, Adjusted R-squared:  0.5906
## F-statistic:  2672 on 9 and 16655 DF,  p-value: < 2.2e-16
```

```
# printing model's coefficients
beta_hat <- coefficients(model)
beta_hat
```

```
##        (Intercept)           distance        bedrooms_ph           rooms_ph
##      -44.590373086      -21.394802952      147.936711512      -11.139726968
## housing_median_age         population      median_income             INLAND
##        0.549982467       -0.002979102       43.950456289      -46.272458370
##       Min_1H_OCEAN         NEAR_OCEAN
##       -7.699758081       22.136334528
```

In order to assess the goodness-of-fit of the regression model, we compute R^2 and $Adjusted\ R^2$. R^2 represents the coefficient of determination. It is a measure that indicates the proportion of the variance in the dependent variable that can be explained by the independent variables in a regression model. Its range goes from 0 (indicating that the independent variables have no explanatory power in predicting the dependent variable) up to 1 (perfect fit).

$Adjusted\ R^2$ takes into account the number of predictors in a regression model. It adjusts R^2 by penalizing the inclusion of additional predictors that do not significantly improve the model's explanatory power. $Adjusted\ R^2$ is used to evaluate the goodness-of-fit of a regression model while considering model complexity.

```
R2 <- summary(model)$r.squared
R2
```

```
## [1] 0.5908068
```

```
adjusted_R2 <- summary(model)$adj.r.squared
adjusted_R2
```

```
## [1] 0.5905856
```

The value of R^2 is 0.59081: it means that 59.081% of the variability in the dependent variable can be explained by the independent variables used in the regression model. In other words, 59.081% of the fluctuations or differences observed in the dependent variable can be attributed to the independent variables included in the model. This value indicates a moderate relationship between the independent variables and the dependent variable.

The $Adjusted\ R^2$ is 0.5905856: it means that 59.0585% of the variability in the dependent variable can be explained by the independent variables in the regression model, while taking into account the number of predictors and model complexity.

Then we compute RSE: it provides an indication of the average amount of error in the predictions made by the model. It quantifies the variability of the residuals around the regression line. A lower RSE value suggests that the model has a better fit to the data, as it indicates less dispersion of the residuals.

```
RSE <- summary(model)$sigma
RSE
```

```
## [1] 60.05008
```

The Variance Inflation Factor (VIF), is a measure used to assess multicollinearity in regression models. Multicollinearity occurs when independent variables in a regression model are highly correlated with each other, which can lead to issues in interpreting the coefficients and destabilize the model's predictions. Typically, a VIF value above 5 or 10 is considered a threshold for concern, indicating moderate to high multicollinearity. In such cases, it may be necessary to address multicollinearity by removing one or more correlated variables. No one of variables shows a concerning VIF value.

```
vif(model)
```

```
##           distance        bedrooms_ph           rooms_ph housing_median_age
##           2.049730           1.207103           3.008486           1.436901
##         population      median_income              INLAND       Min_1H_OCEAN
##           1.128676           3.492837           3.661619           2.913252
##         NEAR_OCEAN
##           2.185327
```

Now we compute Confidence Intervals for the estimated coefficients. Confidence intervals provide an estimate of the range in which the true value of the estimated coefficient is expected to fall, with a certain level of confidence. When training a regression model, the estimated coefficients provide estimates of the parameters that describe the relationship between the independent variables and the dependent variable.

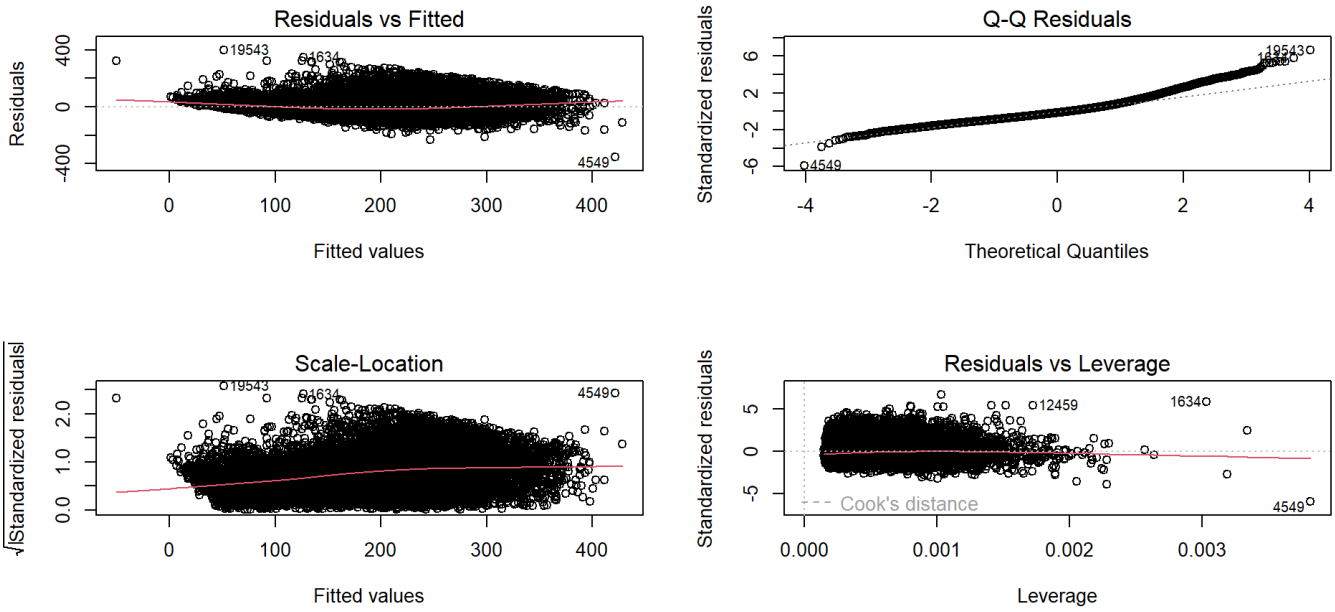
Lastly, residuals and QQ plots are shown to check model features concerning linear model assumptions, which are:

- Linearity: it is assumed that the relationship between the independent and dependent variables is linear. This implies that the change in the independent variable is proportional to the change in the dependent variable, holding other predictors constant.
- Independence of residuals: it is assumed that the residuals are independent of each other.
- Homoscedasticity: it is assumed that the variance of the residuals is constant across all levels of the independent variables. This means that the dispersion of the residuals does not depend on the values of the independent variables.
- Normal distribution of error terms: it is assumed that the error terms are normally distributed and that they have null mean.

```
# confidence interval
confint(model)
```

```
##                          2.5 %        97.5 %
## (Intercept)        -61.79801296 -27.382733211
## distance           -22.95988615 -19.829719756
## bedrooms_ph        132.93034063 162.943082397
## rooms_ph           -12.64829032  -9.631163615
## housing_median_age   0.45997531   0.639989629
## population          -0.00451819  -0.001440013
## median_income       42.76793188  45.132980694
## INLAND             -50.04032399 -42.504592748
## Min_1H_OCEAN       -10.82867881  -4.570837355
## NEAR_OCEAN          18.09854380  26.174125255
```

```
par(mfrow=c(2, 2))
plot(model) #osservo che ho problemi nel QQ-plot
```




```
par(mfrow=c(1, 1))
```

The purpose of residuals vs fitted values plot is to check for any patterns or trends in the residuals. Ideally, the residuals should be randomly scattered around zero without any discernible pattern. As it can be seen from the graph above, residuals are not perfectly randomly distributed around zero. Furthermore, analyzing the left part of the graph, it could detect a light funnel shape: it means that the assumption of homoscedasticity is not totally verified. QQ-Residuals plot shows confirms that residuals are not normally distributed: in fact, QQ-plot shows a slight deviation from a straight line in the QQ norm plot suggesting a departure from normality.

Since the model including all independent variables seems not to perform so well, we try a stepwise approach to select variables using backward selection. Backward selection is a stepwise variable selection method involves iteratively removing variables from a model to identify the most significant predictors. The backward selection process starts with a model that includes all potential predictors. Then, at each step, the least significant variable is removed based on a predefined criterion: AIC for a first attempt, then BIC.

The Akaike Information Criterion (AIC) is based on information theory and provides a trade-off between model fit and complexity. It balances the goodness of fit by minimizing the residual sum of squares or maximizing the likelihood function, while penalizing models with a larger number of parameters. The AIC formula is: $AIC = -2\,l(\hat{\theta}) + 2\,k$ where $l(\hat{\theta})$ is log-likelihood is the logarithm of the likelihood function and k is the number of parameters. A lower AIC indicates a better fit with a balance between model complexity and fit to the data.

The Bayesian Information Criterion (BIC) is derived from Bayesian principles. It penalizes model complexity, but the penalty term is more severe than the AIC penalty term. The BIC formula is: $BIC = -2\,l(\hat{\theta}) + log(n)\,k$ where n is the sample size. The BIC puts a stronger emphasis on model parsimony compared to the AIC, leading to the selection of simpler models.

```
# using AIC to detect best variable to use
library(MASS)
AIC <- stepAIC(model, direction ='backward')
```

```
## Start:  AIC=136502.3
## median_house_value ~ distance + bedrooms_ph + rooms_ph + housing_median_age +
##     population + median_income + INLAND + Min_1H_OCEAN + NEAR_OCEAN
##
##                      Df Sum of Sq      RSS    AIC
## <none>                             60058135 136502
## - population          1     51907 60110042 136515
## - Min_1H_OCEAN        1     83898 60142033 136524
## - NEAR_OCEAN          1    416400 60474534 136615
## - housing_median_age  1    517285 60575420 136643
## - rooms_ph            1    755455 60813590 136709
## - bedrooms_ph         1   1346440 61404575 136870
## - INLAND              1   2089493 62147627 137070
## - distance            1   2588977 62647112 137204
## - median_income       1  19137846 79195981 141110
```

```
summary(AIC)
```

```
##
## Call:
## lm(formula = median_house_value ~ distance + bedrooms_ph + rooms_ph +
##     housing_median_age + population + median_income + INLAND +
##     Min_1H_OCEAN + NEAR_OCEAN, data = dataset_with_dummy)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -353.60  -39.40   -8.79   28.26  398.42
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)        -4.459e+01  8.779e+00  -5.079 3.83e-07 ***
## distance           -2.139e+01  7.985e-01 -26.795  < 2e-16 ***
## bedrooms_ph         1.479e+02  7.656e+00  19.323  < 2e-16 ***
## rooms_ph           -1.114e+01  7.696e-01 -14.474  < 2e-16 ***
## housing_median_age  5.500e-01  4.592e-02  11.977  < 2e-16 ***
## population         -2.979e-03  7.852e-04  -3.794 0.000149 ***
## median_income       4.395e+01  6.033e-01  72.851  < 2e-16 ***
## INLAND             -4.627e+01  1.922e+00 -24.072  < 2e-16 ***
## Min_1H_OCEAN       -7.700e+00  1.596e+00  -4.823 1.42e-06 ***
## NEAR_OCEAN          2.214e+01  2.060e+00  10.746  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 60.05 on 16655 degrees of freedom
## Multiple R-squared:  0.5908, Adjusted R-squared:  0.5906
## F-statistic:  2672 on 9 and 16655 DF,  p-value: < 2.2e-16
```

```
BIC <- step(model, direction = "backward", k = log(dim(dataset_with_dummy)[1]))
```

```
## Start:  AIC=136579.5
## median_house_value ~ distance + bedrooms_ph + rooms_ph + housing_median_age +
##     population + median_income + INLAND + Min_1H_OCEAN + NEAR_OCEAN
##
##                      Df Sum of Sq      RSS    AIC
## <none>                             60058135 136580
## - population          1     51907 60110042 136584
## - Min_1H_OCEAN        1     83898 60142033 136593
## - NEAR_OCEAN          1    416400 60474534 136685
## - housing_median_age  1    517285 60575420 136713
## - rooms_ph            1    755455 60813590 136778
## - bedrooms_ph         1   1346440 61404575 136939
## - INLAND              1   2089493 62147627 137140
## - distance            1   2588977 62647112 137273
## - median_income       1  19137846 79195981 141180
```

```
summary(BIC)
```

```
##
## Call:
## lm(formula = median_house_value ~ distance + bedrooms_ph + rooms_ph +
##     housing_median_age + population + median_income + INLAND +
##     Min_1H_OCEAN + NEAR_OCEAN, data = dataset_with_dummy)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -353.60  -39.40   -8.79   28.26  398.42
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)        -4.459e+01  8.779e+00  -5.079 3.83e-07 ***
## distance           -2.139e+01  7.985e-01 -26.795  < 2e-16 ***
## bedrooms_ph         1.479e+02  7.656e+00  19.323  < 2e-16 ***
## rooms_ph           -1.114e+01  7.696e-01 -14.474  < 2e-16 ***
## housing_median_age  5.500e-01  4.592e-02  11.977  < 2e-16 ***
## population         -2.979e-03  7.852e-04  -3.794 0.000149 ***
## median_income       4.395e+01  6.033e-01  72.851  < 2e-16 ***
## INLAND             -4.627e+01  1.922e+00 -24.072  < 2e-16 ***
## Min_1H_OCEAN       -7.700e+00  1.596e+00  -4.823 1.42e-06 ***
## NEAR_OCEAN          2.214e+01  2.060e+00  10.746  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 60.05 on 16655 degrees of freedom
## Multiple R-squared:  0.5908, Adjusted R-squared:  0.5906
## F-statistic:  2672 on 9 and 16655 DF,  p-value: < 2.2e-16
```

```
AIC(model)
```

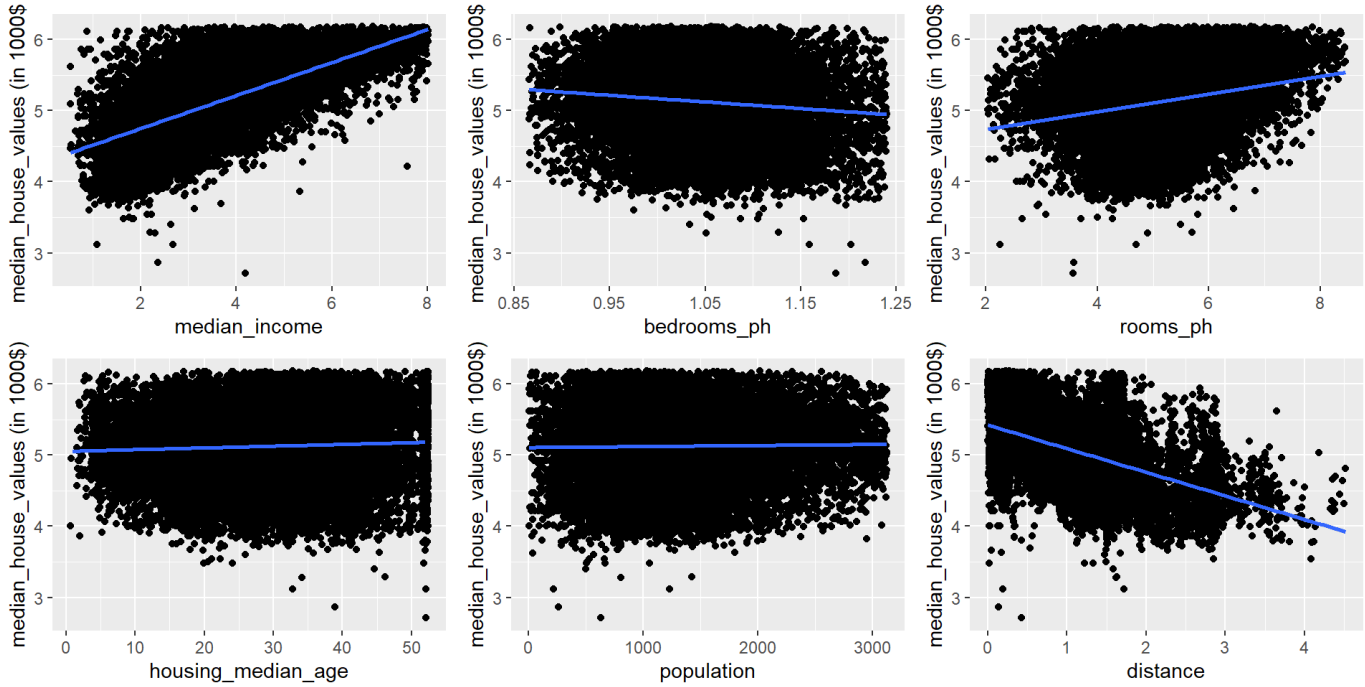
```
## [1] 183797.5
```

```
BIC(model)
```

```
## [1] 183882.5
```

The outputs suggest to maintain all features in the model, but AIC and BIC are quite high. Since we are not satisfied with results obtained, we try to transform the response variable using logarithmic transformation.

```
# relations between features and log(y)
g1_log <- ggplot(data, aes(x = median_income, y = log(median_house_value))) +
  geom_jitter() +
  labs(y ="median_house_values (in 1000$)")+
  geom_smooth(method = lm, formula = y~x)

g2_log <- ggplot(data, aes(x = bedrooms_ph, y =log(median_house_value))) +
  geom_jitter() +
  labs(y ="median_house_values (in 1000$)")+
  geom_smooth(method = lm, formula = y~x)

g3_log <- ggplot(data, aes(x = rooms_ph, y = log(median_house_value))) +
  geom_jitter() +
  labs(y ="median_house_values (in 1000$)")+
  geom_smooth(method = lm, formula = y~x)

g4_log <- ggplot(data, aes(x = housing_median_age, y =log(median_house_value))) +
  geom_jitter() +
  labs(y ="median_house_values (in 1000$)")+
  geom_smooth(method = lm, formula = y~x)

g5_log <- ggplot(data, aes(x = population, y =log(median_house_value))) +
  geom_jitter() +
  labs(y ="median_house_values (in 1000$)")+
  geom_smooth(method = lm, formula = y~x)

g6_log <- ggplot(data, aes(x = distance, y = log(median_house_value))) +
  geom_jitter() +
  labs(y ="median_house_values (in 1000$)")+
  geom_smooth(method = lm, formula = y~x)

plot_grid(g1_log, g2_log, g3_log, g4_log, g5_log, g6_log, nrow = 2)
```



Using log-transformation of Y, relations between variables seem to get better.

```
# model with log(Y)
model_log <- lm(log(median_house_value) ~ . , data = dataset_with_dummy)

summary(model_log)
```

```
##
## Call:
## lm(formula = log(median_house_value) ~ ., data = dataset_with_dummy)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.54484 -0.20042 -0.02433  0.18142  1.90254
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)         4.213e+00  4.418e-02  95.353  < 2e-16 ***
## distance           -1.748e-01  4.018e-03 -43.491  < 2e-16 ***
## bedrooms_ph         5.442e-01  3.853e-02  14.125  < 2e-16 ***
## rooms_ph           -3.799e-02  3.873e-03  -9.808  < 2e-16 ***
## housing_median_age  4.825e-05  2.311e-04   0.209  0.83460
## population         -1.134e-05  3.952e-06  -2.868  0.00413 **
## median_income       2.154e-01  3.036e-03  70.931  < 2e-16 ***
## INLAND             -2.944e-01  9.674e-03 -30.433  < 2e-16 ***
## Min_1H_OCEAN       -8.252e-03  8.034e-03  -1.027  0.30435
## NEAR_OCEAN          1.600e-01  1.037e-02  15.432  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3022 on 16655 degrees of freedom
## Multiple R-squared:  0.6638, Adjusted R-squared:  0.6636
## F-statistic:  3654 on 9 and 16655 DF,  p-value: < 2.2e-16
```

```
beta_hat_log <- coefficients(model_log)
beta_hat_log
```

```
##        (Intercept)           distance        bedrooms_ph           rooms_ph
##       4.212814e+00      -1.747642e-01       5.442275e-01      -3.798806e-02
## housing_median_age         population      median_income             INLAND
##       4.825418e-05      -1.133468e-05       2.153582e-01      -2.944177e-01
##       Min_1H_OCEAN         NEAR_OCEAN
##      -8.251981e-03       1.599898e-01
```

```
R2_log <- summary(model_log)$r.squared
R2_log
```

```
## [1] 0.6638152
```

```
adjusted_R2_log <- summary(model_log)$adj.r.squared
adjusted_R2_log
```

```
## [1] 0.6636335
```

```
RSE_log <- summary(model_log)$sigma
RSE_log
```

```
## [1] 0.3022116
```

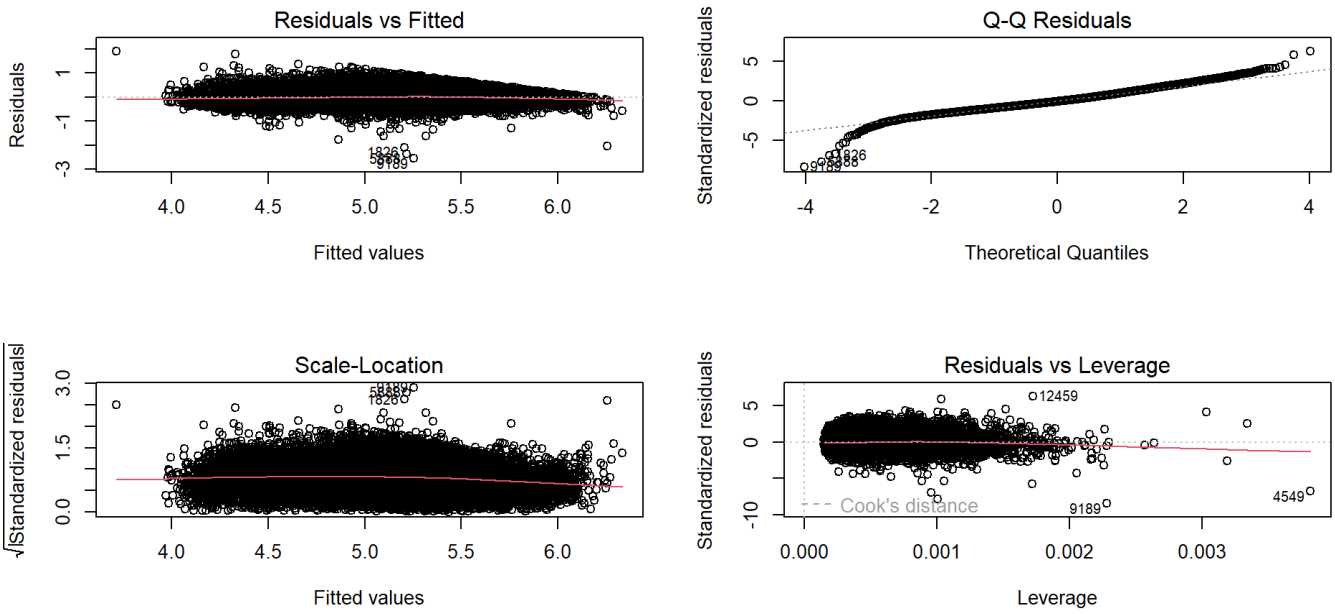
```
vif(model_log)
```

```
##           distance        bedrooms_ph           rooms_ph housing_median_age
##           2.049730           1.207103           3.008486           1.436901
##         population      median_income              INLAND       Min_1H_OCEAN
##           1.128676           3.492837           3.661619           2.913252
##         NEAR_OCEAN
##           2.185327
```

```
confint(model_log)
```

```
##                            2.5 %        97.5 %
## (Intercept)         4.126214e+00  4.299415e+00
## distance           -1.826407e-01 -1.668877e-01
## bedrooms_ph         4.687055e-01  6.197494e-01
## rooms_ph           -4.558014e-02 -3.039597e-02
## housing_median_age -4.047212e-04  5.012295e-04
## population         -1.908039e-05 -3.588971e-06
## median_income       2.094070e-01  2.213095e-01
## INLAND             -3.133800e-01 -2.754553e-01
## Min_1H_OCEAN       -2.399877e-02  7.494811e-03
## NEAR_OCEAN          1.396690e-01  1.803106e-01
```

```
par(mfrow=c(2, 2))
plot(model_log)
```




```
par(mfrow=c(1, 1))
```

The outputs obtained show a clear improvement: - R^2 passes from 0.591 to 0.6638; - $Adjusted\ R^2$ passes from 0.59 to 0.6636; - RSE passes from 60.05 to 0.302.

Plots confirm this result: residuals are more sparse and their distribution seems closer to normal distribution.

Now, using the same variable selection method as before, we check AIC and BIC.

```
AIC_log <- stepAIC(model_log, direction ='backward')
```

```
## Start:  AIC=-39873.61
## log(median_house_value) ~ distance + bedrooms_ph + rooms_ph +
##     housing_median_age + population + median_income + INLAND +
##     Min_1H_OCEAN + NEAR_OCEAN
##
##                      Df Sum of Sq    RSS    AIC
## - housing_median_age  1      0.00 1521.1 -39876
## - Min_1H_OCEAN        1      0.10 1521.2 -39875
## <none>                            1521.1 -39874
## - population          1      0.75 1521.9 -39867
## - rooms_ph            1      8.79 1529.9 -39780
## - bedrooms_ph         1     18.22 1539.3 -39677
## - NEAR_OCEAN          1     21.75 1542.9 -39639
## - INLAND              1     84.59 1605.7 -38974
## - distance            1    172.75 1693.9 -38083
## - median_income       1    459.50 1980.6 -35477
##
## Step:  AIC=-39875.57
## log(median_house_value) ~ distance + bedrooms_ph + rooms_ph +
##     population + median_income + INLAND + Min_1H_OCEAN + NEAR_OCEAN
##
##                 Df Sum of Sq    RSS    AIC
## - Min_1H_OCEAN   1      0.10 1521.2 -39876
## <none>                       1521.1 -39876
## - population     1      0.86 1522.0 -39868
## - rooms_ph       1      8.83 1530.0 -39781
## - bedrooms_ph    1     18.49 1539.6 -39676
## - NEAR_OCEAN     1     21.77 1542.9 -39641
## - INLAND         1     87.32 1608.5 -38947
## - distance       1    194.02 1715.2 -37877
## - median_income  1    494.79 2015.9 -35184
##
## Step:  AIC=-39876.42
## log(median_house_value) ~ distance + bedrooms_ph + rooms_ph +
##     population + median_income + INLAND + NEAR_OCEAN
##
##                 Df Sum of Sq    RSS    AIC
## <none>                       1521.2 -39876
## - population     1      0.92 1522.2 -39868
## - rooms_ph       1      8.73 1530.0 -39783
## - bedrooms_ph    1     18.49 1539.7 -39677
## - NEAR_OCEAN     1     41.12 1562.4 -39434
## - INLAND         1    159.97 1681.2 -38212
## - distance       1    199.05 1720.3 -37829
## - median_income  1    497.32 2018.6 -35165
```

```
summary(AIC_log)
```

```
##
## Call:
## lm(formula = log(median_house_value) ~ distance + bedrooms_ph +
##     rooms_ph + population + median_income + INLAND + NEAR_OCEAN,
##     data = dataset_with_dummy)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.54347 -0.19999 -0.02458  0.18085  1.90206
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    4.210e+00  4.023e-02 104.641   <2e-16 ***
## distance      -1.756e-01  3.761e-03 -46.685   <2e-16 ***
## bedrooms_ph    5.431e-01  3.817e-02  14.228   <2e-16 ***
## rooms_ph      -3.754e-02  3.839e-03  -9.779   <2e-16 ***
## population    -1.194e-05  3.759e-06  -3.175   0.0015 **
## median_income  2.149e-01  2.912e-03  73.794   <2e-16 ***
## INLAND        -2.877e-01  6.874e-03 -41.852   <2e-16 ***
## NEAR_OCEAN     1.671e-01  7.874e-03  21.220   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3022 on 16657 degrees of freedom
## Multiple R-squared:  0.6638, Adjusted R-squared:  0.6636
## F-statistic:  4698 on 7 and 16657 DF,  p-value: < 2.2e-16
```

AIC stepwise backward selection suggest to remove variables “housing median age” and “<1H OCEAN” variables.

```
BIC_log <- step(model_log, direction = "backward", k = log(dim(dataset_with_dummy)[1]))
```

```
## Start:  AIC=-39796.4
## log(median_house_value) ~ distance + bedrooms_ph + rooms_ph +
##     housing_median_age + population + median_income + INLAND +
##     Min_1H_OCEAN + NEAR_OCEAN
##
##                      Df Sum of Sq    RSS    AIC
## - housing_median_age  1      0.00 1521.1 -39806
## - Min_1H_OCEAN        1      0.10 1521.2 -39805
## - population          1      0.75 1521.9 -39798
## <none>                            1521.1 -39796
## - rooms_ph            1      8.79 1529.9 -39710
## - bedrooms_ph         1     18.22 1539.3 -39608
## - NEAR_OCEAN          1     21.75 1542.9 -39570
## - INLAND              1     84.59 1605.7 -38904
## - distance            1    172.75 1693.9 -38014
## - median_income       1    459.50 1980.6 -35407
##
## Step:  AIC=-39806.08
## log(median_house_value) ~ distance + bedrooms_ph + rooms_ph +
##     population + median_income + INLAND + Min_1H_OCEAN + NEAR_OCEAN
##
##                 Df Sum of Sq    RSS    AIC
## - Min_1H_OCEAN   1      0.10 1521.2 -39815
## - population     1      0.86 1522.0 -39806
## <none>                       1521.1 -39806
## - rooms_ph       1      8.83 1530.0 -39719
## - bedrooms_ph    1     18.49 1539.6 -39614
## - NEAR_OCEAN     1     21.77 1542.9 -39579
## - INLAND         1     87.32 1608.5 -38886
## - distance       1    194.02 1715.2 -37815
## - median_income  1    494.79 2015.9 -35123
##
## Step:  AIC=-39814.65
## log(median_house_value) ~ distance + bedrooms_ph + rooms_ph +
##     population + median_income + INLAND + NEAR_OCEAN
##
##                 Df Sum of Sq    RSS    AIC
## <none>                       1521.2 -39815
## - population     1      0.92 1522.2 -39814
## - rooms_ph       1      8.73 1530.0 -39729
## - bedrooms_ph    1     18.49 1539.7 -39623
## - NEAR_OCEAN     1     41.12 1562.4 -39380
## - INLAND         1    159.97 1681.2 -38158
## - distance       1    199.05 1720.3 -37775
## - median_income  1    497.32 2018.6 -35110
```

```
summary(BIC_log)
```

```
##
## Call:
## lm(formula = log(median_house_value) ~ distance + bedrooms_ph +
##     rooms_ph + population + median_income + INLAND + NEAR_OCEAN,
##     data = dataset_with_dummy)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.54347 -0.19999 -0.02458  0.18085  1.90206
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    4.210e+00  4.023e-02 104.641   <2e-16 ***
## distance      -1.756e-01  3.761e-03 -46.685   <2e-16 ***
## bedrooms_ph    5.431e-01  3.817e-02  14.228   <2e-16 ***
## rooms_ph      -3.754e-02  3.839e-03  -9.779   <2e-16 ***
## population    -1.194e-05  3.759e-06  -3.175   0.0015 **
## median_income  2.149e-01  2.912e-03  73.794   <2e-16 ***
## INLAND        -2.877e-01  6.874e-03 -41.852   <2e-16 ***
## NEAR_OCEAN     1.671e-01  7.874e-03  21.220   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3022 on 16657 degrees of freedom
## Multiple R-squared:  0.6638, Adjusted R-squared:  0.6636
## F-statistic:  4698 on 7 and 16657 DF,  p-value: < 2.2e-16
```

BIC stepwise backward selection reach the same conclusion as AIC stepwise backward: we should delete “housing median age” and “<1H OCEAN” variables.

```
AIC(model_log)
```

```
## [1] 7421.612
```

```
BIC(model_log)
```

```
## [1] 7506.544
```

AIC and BIC improves too: - AIC passes from 183797.5 to 7421.612; - BIC passes from 183882.5 to 7506.544.

Given that we update our model considering all variables but “housing median age” and “<1H OCEAN” variables.

```
# model with log(Y), without housing median age and <1H OCEAN
model_log_2 <- lm(log(median_house_value) ~ . -Min_1H_OCEAN -housing_median_age, data = datas
et_with_dummy)

summary(model_log_2)
```

```
##
## Call:
## lm(formula = log(median_house_value) ~ . - Min_1H_OCEAN - housing_median_age,
##     data = dataset_with_dummy)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.54347 -0.19999 -0.02458  0.18085  1.90206
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    4.210e+00  4.023e-02 104.641   <2e-16 ***
## distance      -1.756e-01  3.761e-03 -46.685   <2e-16 ***
## bedrooms_ph    5.431e-01  3.817e-02  14.228   <2e-16 ***
## rooms_ph      -3.754e-02  3.839e-03  -9.779   <2e-16 ***
## population    -1.194e-05  3.759e-06  -3.175   0.0015 **
## median_income  2.149e-01  2.912e-03  73.794   <2e-16 ***
## INLAND        -2.877e-01  6.874e-03 -41.852   <2e-16 ***
## NEAR_OCEAN     1.671e-01  7.874e-03  21.220   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3022 on 16657 degrees of freedom
## Multiple R-squared:  0.6638, Adjusted R-squared:  0.6636
## F-statistic:  4698 on 7 and 16657 DF,  p-value: < 2.2e-16
```

```
beta_hat_log_2 <- coefficients(model_log_2)
beta_hat_log_2
```

```
##   (Intercept)      distance   bedrooms_ph      rooms_ph    population
##  4.209521e+00 -1.755995e-01  5.431404e-01 -3.754319e-02 -1.193537e-05
## median_income        INLAND    NEAR_OCEAN
##  2.149138e-01 -2.876875e-01  1.670865e-01
```

```
R2_log_2 <- summary(model_log_2)$r.squared
R2_log_2
```

```
## [1] 0.6637912
```

```
adjusted_R2_log_2 <- summary(model_log_2)$adj.r.squared
adjusted_R2_log_2
```

```
## [1] 0.6636499
```

```
RSE_log_2 <- summary(model_log_2)$sigma
RSE_log_2
```

```
## [1] 0.3022043
```

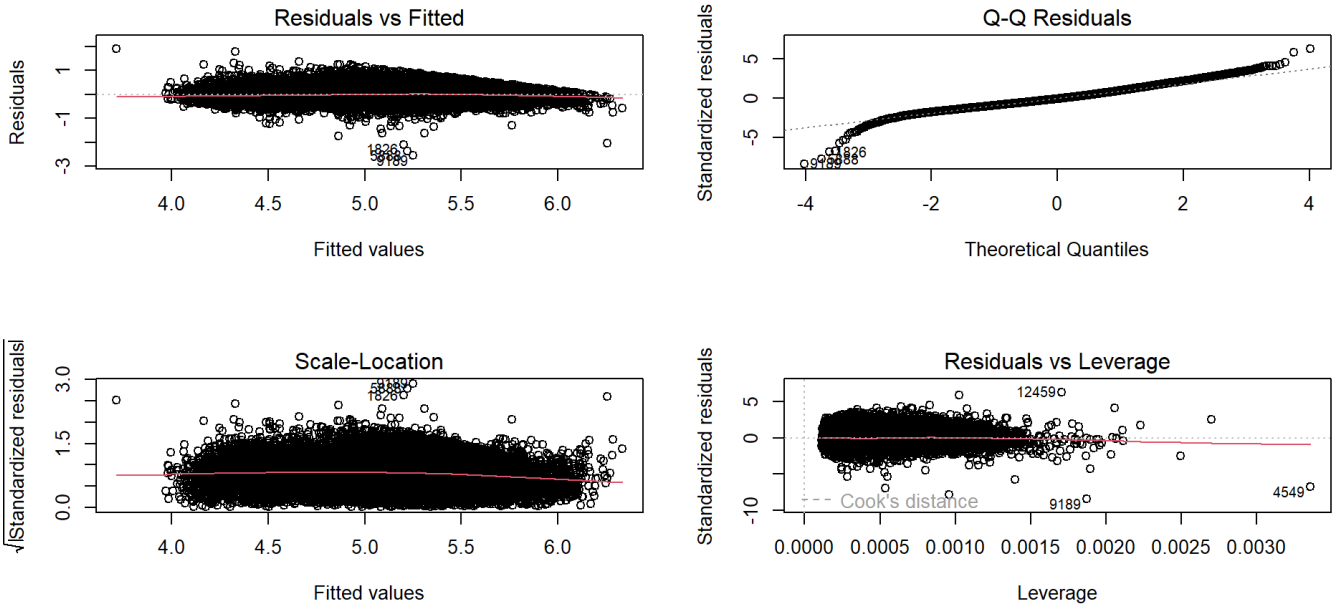
```
vif(model_log_2)
```

```
##      distance   bedrooms_ph      rooms_ph    population median_income
##      1.795960      1.185015      2.956093      1.021415      3.213906
##        INLAND    NEAR_OCEAN
##      1.848730      1.260737
```

```
confint(model_log_2)
```

```
##                     2.5 %        97.5 %
## (Intercept)    4.130669e+00  4.288372e+00
## distance      -1.829721e-01 -1.682268e-01
## bedrooms_ph    4.683144e-01  6.179664e-01
## rooms_ph      -4.506870e-02 -3.001769e-02
## population    -1.930367e-05 -4.567077e-06
## median_income  2.092053e-01  2.206224e-01
## INLAND        -3.011611e-01 -2.742140e-01
## NEAR_OCEAN     1.516523e-01  1.825207e-01
```

```
par(mfrow=c(2, 2))
plot(model_log_2)
```




```
par(mfrow=c(1, 1))
```

```
# backward stepwise: AIC
AIC_log_2 <- stepAIC(model_log_2, direction ='backward')
```

```
## Start:  AIC=-39876.42
## log(median_house_value) ~ (distance + bedrooms_ph + rooms_ph +
##     housing_median_age + population + median_income + INLAND +
##     Min_1H_OCEAN + NEAR_OCEAN) - Min_1H_OCEAN - housing_median_age
##
##                 Df Sum of Sq    RSS    AIC
## <none>                       1521.2 -39876
## - population     1      0.92 1522.2 -39868
## - rooms_ph       1      8.73 1530.0 -39783
## - bedrooms_ph    1     18.49 1539.7 -39677
## - NEAR_OCEAN     1     41.12 1562.4 -39434
## - INLAND         1    159.97 1681.2 -38212
## - distance       1    199.05 1720.3 -37829
## - median_income  1    497.32 2018.6 -35165
```

```
summary(AIC_log_2)
```

```
##
## Call:
## lm(formula = log(median_house_value) ~ (distance + bedrooms_ph +
##     rooms_ph + housing_median_age + population + median_income +
##     INLAND + Min_1H_OCEAN + NEAR_OCEAN) - Min_1H_OCEAN - housing_median_age,
##     data = dataset_with_dummy)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.54347 -0.19999 -0.02458  0.18085  1.90206
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    4.210e+00  4.023e-02 104.641   <2e-16 ***
## distance      -1.756e-01  3.761e-03 -46.685   <2e-16 ***
## bedrooms_ph    5.431e-01  3.817e-02  14.228   <2e-16 ***
## rooms_ph      -3.754e-02  3.839e-03  -9.779   <2e-16 ***
## population    -1.194e-05  3.759e-06  -3.175   0.0015 **
## median_income  2.149e-01  2.912e-03  73.794   <2e-16 ***
## INLAND        -2.877e-01  6.874e-03 -41.852   <2e-16 ***
## NEAR_OCEAN     1.671e-01  7.874e-03  21.220   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3022 on 16657 degrees of freedom
## Multiple R-squared:  0.6638, Adjusted R-squared:  0.6636
## F-statistic:  4698 on 7 and 16657 DF,  p-value: < 2.2e-16
```

```
# backward stepwise: BIC
BIC_log_2 <- step(model_log_2, direction = "backward", k = log(dim(dataset_with_dummy)[1]))
```

```
## Start:  AIC=-39814.65
## log(median_house_value) ~ (distance + bedrooms_ph + rooms_ph +
##     housing_median_age + population + median_income + INLAND +
##     Min_1H_OCEAN + NEAR_OCEAN) - Min_1H_OCEAN - housing_median_age
##
##                 Df Sum of Sq    RSS    AIC
## <none>                       1521.2 -39815
## - population     1      0.92 1522.2 -39814
## - rooms_ph       1      8.73 1530.0 -39729
## - bedrooms_ph    1     18.49 1539.7 -39623
## - NEAR_OCEAN     1     41.12 1562.4 -39380
## - INLAND         1    159.97 1681.2 -38158
## - distance       1    199.05 1720.3 -37775
## - median_income  1    497.32 2018.6 -35110
```

```
summary(BIC_log_2)
```

```
##
## Call:
## lm(formula = log(median_house_value) ~ (distance + bedrooms_ph +
##     rooms_ph + housing_median_age + population + median_income +
##     INLAND + Min_1H_OCEAN + NEAR_OCEAN) - Min_1H_OCEAN - housing_median_age,
##     data = dataset_with_dummy)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.54347 -0.19999 -0.02458  0.18085  1.90206
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    4.210e+00  4.023e-02 104.641   <2e-16 ***
## distance      -1.756e-01  3.761e-03 -46.685   <2e-16 ***
## bedrooms_ph    5.431e-01  3.817e-02  14.228   <2e-16 ***
## rooms_ph      -3.754e-02  3.839e-03  -9.779   <2e-16 ***
## population    -1.194e-05  3.759e-06  -3.175   0.0015 **
## median_income  2.149e-01  2.912e-03  73.794   <2e-16 ***
## INLAND        -2.877e-01  6.874e-03 -41.852   <2e-16 ***
## NEAR_OCEAN     1.671e-01  7.874e-03  21.220   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3022 on 16657 degrees of freedom
## Multiple R-squared:  0.6638, Adjusted R-squared:  0.6636
## F-statistic:  4698 on 7 and 16657 DF,  p-value: < 2.2e-16
```

```
AIC(model_log_2)
```

```
## [1] 7418.803
```

```
BIC(model_log_2)
```

```
## [1] 7488.293
```

The two models are similar but the model without “housing median age” and “<1H OCEAN” variables shows slightly better results, expecially in AIC and BIC values. In fact, AIC and BIC decrease after the variables removal.

AIC: 7418.803 BIC: 7488.293

We have now reached the best model for our response variable “median house value”.

ANOVA: model_log; model_log_2

ANOVA is used to determine if there are significant differences between models in terms of how well they fit the data. We use ANOVA to compare model_log and model_log_2.

```
anova(model_log, model_log_2)
```

```
## Analysis of Variance Table
##
## Model 1: log(median_house_value) ~ distance + bedrooms_ph + rooms_ph +
##     housing_median_age + population + median_income + INLAND +
##     Min_1H_OCEAN + NEAR_OCEAN
## Model 2: log(median_house_value) ~ (distance + bedrooms_ph + rooms_ph +
##     housing_median_age + population + median_income + INLAND +
##     Min_1H_OCEAN + NEAR_OCEAN) - Min_1H_OCEAN - housing_median_age
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1  16655 1521.1
## 2  16657 1521.2 -2  -0.10877 0.5954 0.5513
```

Those outputs show that there aren’t significant differences among the two models, since p-value (0.5513) is greater than alpha (0.05). That suggests that those two models don’t differ significantly on their ability to explain response variable variation.

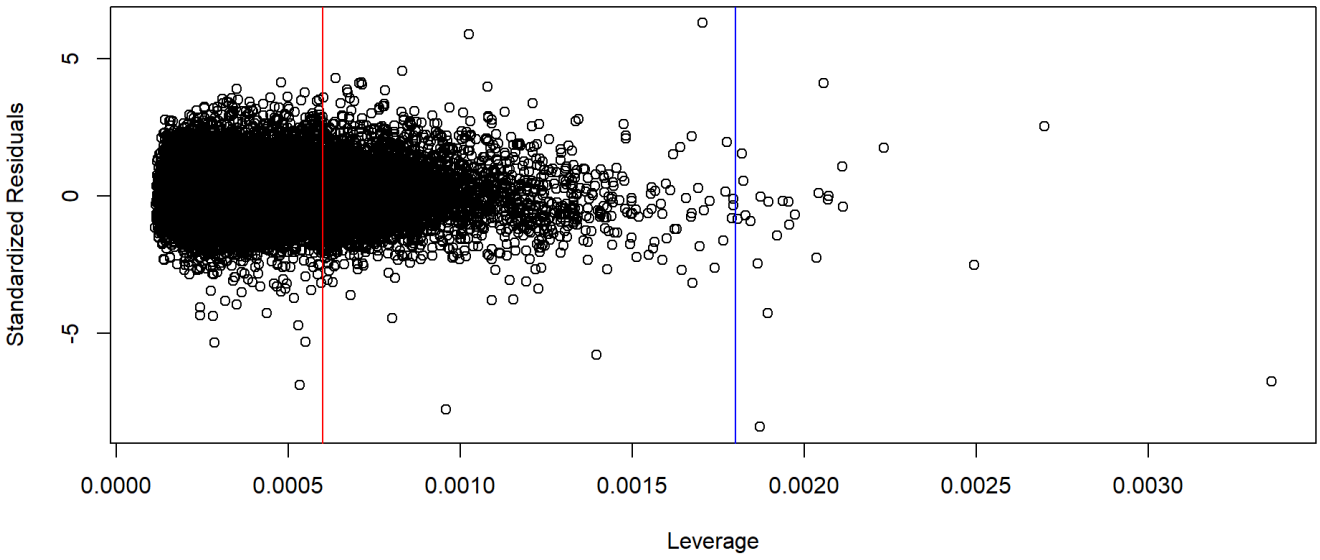
High Leverage Points

However, looking at residuals versus leverage plot of every model, it can be detected the presence of observations with higher residuals compared to other observations or observations with very high leverage. Leverage represents the influence that an observation has on the estimation of the model coefficients. An observation with high leverage has a greater weight in estimating the coefficients compared to observations with low leverage. Therefore, observations with high leverage can significantly affect the shape and slope of the regression. Those data are called High Leverage Points. So, we check for high leverage points using as criterion the rule of thumb that there is high leverage when the leverage statistic is more than 3 times its average value $(p+1)/n$:

if $h_i = \frac{1}{n} + \frac{(x_i - \bar{x})^2}{\sum_{j=1}^{n}(x_j - \bar{x})^2} > 3\,\frac{(p+1)}{n}$

The “hat” values represent the estimate of the distance of each observation from the center of the dataset. In other words, they measure how much each observation influences the results of the model.

```
# hat values
hat_values <- hatvalues(model_log_2)

# plotting high leverage points (on the right of blue line)
p <- dim(dataset_with_dummy)[2]-1
n <- dim(dataset_with_dummy)[1]
plot(hat_values, rstandard(model_log_2), xlab="Leverage", ylab = "Standardized Residuals") +
abline(v=(p+1)/n, col = "red") +
abline(v=3*(p+1)/n, col = "blue")
```




```
## integer(0)
```

In the graph above high leverage points are shown. Red line represents average high leverage value and line blue represents the threshold: points on the right of blue line are considered high leverage points.

```
# computing high leverage points and corresponding rows removal
high_leverage_points <- names(hat_values[hat_values>3*(p+1)/n])
high_leverage_points
```

```
##  [1] "1634"  "1851"  "1852"  "1854"  "1855"  "1856"  "1857"  "1858"  "1859"
## [10] "1860"  "1861"  "1862"  "1864"  "2005"  "2560"  "2604"  "2622"  "4549"
## [19] "8223"  "9189"  "12105" "14675" "15630" "16532" "16862" "17866"
```

```
dataset_with_dummy[high_leverage_points, c(1,2,3,4,5,6,7,8,9,10)]
```

```
##         distance bedrooms_ph rooms_ph housing_median_age population
## 1634  0.23706539   0.8666667 6.333333                 20         31
## 1851  4.38958996   1.1009174 6.282110                 16       1259
## 1852  4.44272439   1.1548117 5.589958                 19       1298
## 1854  4.37817313   1.1159196 5.349304                 17       1947
## 1855  4.37923509   1.1156250 4.906250                 15       1645
## 1856  4.34576806   1.0704441 4.598775                 28       1530
## 1857  4.35990826   1.0915395 5.284327                 20       1993
## 1858  4.35289559   1.0466926 5.200389                 20       1282
## 1859  4.33056578   1.0735849 5.122642                 15       1532
## 1860  4.51624844   1.1700405 5.457490                 21       1208
## 1861  4.49615391   1.1448763 5.893993                 19        841
## 1862  4.47904008   1.1644737 5.870614                 17       1244
## 1864  4.18072960   1.1464968 5.987261                 22        743
## 2005  2.83381368   1.0357143 2.059524                 52        401
## 2560  3.49284984   0.8915663 4.096386                 50        235
## 2604  3.95102518   1.2193878 5.250000                 29        509
## 2622  3.55995786   1.1988950 8.022099                 14        516
## 4549  0.04242641   1.0000000 3.142857                 52         55
## 8223  0.29017236   0.8750000 3.062500                 21         29
## 9189  0.42485292   1.1866667 3.568889                 52        628
## 12105 0.91350972   0.9500000 7.600000                  8       1275
## 14675 1.67764716   1.0102881 7.053498                 15       2303
## 15630 0.05000000   1.0449102 4.038922                 12        536
## 16532 1.23065023   1.0277778 5.486111                 26        193
## 16862 0.14035669   0.9166667 5.500000                 46         30
## 17866 0.64350602   1.1428571 5.714286                 26         52
##       median_income median_house_value INLAND Min_1H_OCEAN NEAR_OCEAN
## 1634         2.4444            475.000      0            0          0
## 1851         3.7557            109.400      0            0          1
## 1852         1.9797             85.800      0            0          1
## 1854         2.5795             68.400      0            0          1
## 1855         1.6654             74.600      0            0          1
## 1856         1.7038             78.300      0            0          1
## 1857         2.0074             66.900      0            0          1
## 1858         2.4605            105.900      0            0          1
## 1859         2.1829             69.500      0            0          1
## 1860         2.2750            122.400      0            0          1
## 1861         2.1336             75.000      0            0          1
## 1862         3.0313            103.600      0            0          1
## 1864         2.9688            152.700      0            1          0
## 2005         2.1094             75.000      1            0          0
## 2560         1.7500             67.500      0            0          1
## 2604         2.0156             62.800      0            0          1
## 2622         5.0329            165.600      0            0          1
## 4549         7.5752             67.500      0            1          0
## 8223         5.0000             87.500      0            0          1
## 9189         4.1932             14.999      1            0          0
## 12105        1.6250            162.500      1            0          0
## 14675        2.5953             67.500      0            0          1
## 15630        7.7852            250.000      0            0          0
## 16532        7.3718            212.500      1            0          0
## 16862        2.3750            275.000      0            0          1
## 17866        7.7197            225.000      0            1          0
```

```
number_high_leverage_points <-length(high_leverage_points)
number_high_leverage_points
```

```
## [1] 26
```

```
# dataframe without high leverage points creation
data_w_hlp <- dataset_with_dummy[!(rownames(dataset_with_dummy) %in% high_leverage_points),]
```

Since by definition high leverage points are data which have a strong impact on the model, we decide to delete them in order to check if the model without high leverage could be better. Now it is possible to build a model without the high leverage points influence.

```
# model without high leverage points
model_w_hlp <- lm(log(median_house_value)~. -housing_median_age -Min_1H_OCEAN, data = data_w_
hlp)
summary(model_w_hlp)
```

```
##
## Call:
## lm(formula = log(median_house_value) ~ . - housing_median_age -
##     Min_1H_OCEAN, data = data_w_hlp)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.36284 -0.20031 -0.02406  0.18059  1.91115
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    4.195e+00  4.013e-02 104.545  < 2e-16 ***
## distance      -1.746e-01  3.779e-03 -46.201  < 2e-16 ***
## bedrooms_ph    5.669e-01  3.811e-02  14.875  < 2e-16 ***
## rooms_ph      -4.201e-02  3.849e-03 -10.916  < 2e-16 ***
## population    -1.316e-05  3.746e-06  -3.514 0.000443 ***
## median_income  2.184e-01  2.921e-03  74.775  < 2e-16 ***
## INLAND        -2.848e-01  6.874e-03 -41.432  < 2e-16 ***
## NEAR_OCEAN     1.678e-01  7.845e-03  21.385  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3007 on 16631 degrees of freedom
## Multiple R-squared:  0.6663, Adjusted R-squared:  0.6662
## F-statistic:  4745 on 7 and 16631 DF,  p-value: < 2.2e-16
```

```
beta_hat_w_hlp <- coefficients(model_w_hlp)
beta_hat_w_hlp
```

```
##   (Intercept)      distance   bedrooms_ph      rooms_ph    population
##  4.195060e+00 -1.745850e-01  5.668878e-01 -4.201467e-02 -1.316367e-05
## median_income        INLAND    NEAR_OCEAN
##  2.183830e-01 -2.847822e-01  1.677633e-01
```

```
R2_w_hlp <- summary(model_w_hlp)$r.squared
R2_w_hlp
```

```
## [1] 0.6663296
```

```
adjusted_R2_w_hlp <- summary(model_w_hlp)$adj.r.squared
adjusted_R2_w_hlp
```

```
## [1] 0.6661891
```

```
RSE_w_hlp <- summary(model_w_hlp)$sigma
RSE_w_hlp
```

```
## [1] 0.3007113
```

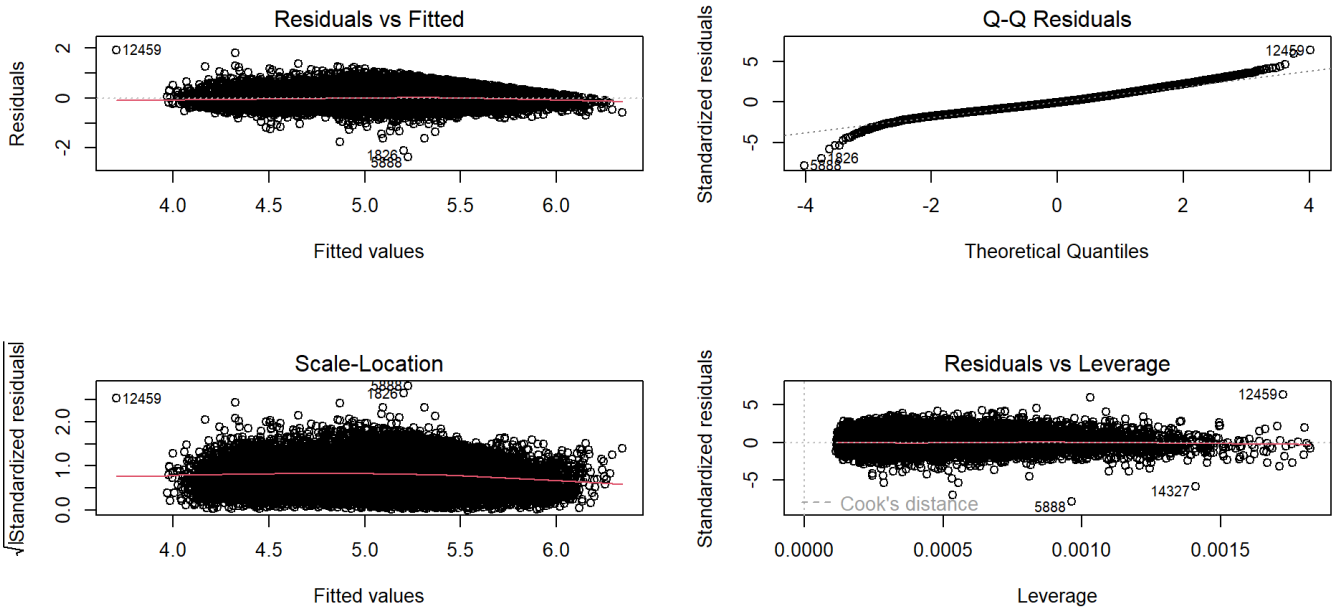
```
vif(model_w_hlp)
```

```
##      distance   bedrooms_ph      rooms_ph    population median_income
##      1.802759      1.188387      2.993048      1.022095      3.254074
##        INLAND    NEAR_OCEAN
##      1.864847      1.254928
```

```
confint(model_w_hlp)
```

```
##                       2.5 %        97.5 %
## (Intercept)    4.116406e+00  4.273713e+00
## distance      -1.819919e-01 -1.671781e-01
## bedrooms_ph    4.921880e-01  6.415877e-01
## rooms_ph      -4.955924e-02 -3.447010e-02
## population    -2.050669e-05 -5.820645e-06
## median_income  2.126585e-01  2.241076e-01
## INLAND        -2.982551e-01 -2.713092e-01
## NEAR_OCEAN     1.523865e-01  1.831402e-01
```

```
par(mfrow=c(2, 2))
plot(model_w_hlp)
```




```
par(mfrow=c(1, 1))
```

The outputs obtained show an improvement: - R^2 passes from 0.6637 to 0.6663; - $Adjusted\ R^2$ passes from 0.6636 to 0.6662; - RSE passes from 0,302 to 0.3007113.

This conclusion can be desumed from plots too: residuals become more normally distributed.

```
# AIC and BIC for the final model
AIC_b_w_hlp <- stepAIC(model_w_hlp, direction ='backward')
```

```
## Start:  AIC=-39979
## log(median_house_value) ~ (distance + bedrooms_ph + rooms_ph +
##     housing_median_age + population + median_income + INLAND +
##     Min_1H_OCEAN + NEAR_OCEAN) - housing_median_age - Min_1H_OCEAN
##
##                Df Sum of Sq    RSS    AIC
## <none>                      1503.9 -39979
## - population    1      1.12 1505.0 -39969
## - rooms_ph      1     10.77 1514.7 -39862
## - bedrooms_ph   1     20.01 1523.9 -39761
## - NEAR_OCEAN    1     41.35 1545.2 -39530
## - INLAND        1    155.22 1659.1 -38347
## - distance      1    193.02 1696.9 -37972
## - median_income 1    505.61 2009.5 -35159
```

```
summary(AIC_b_w_hlp)
```

```
##
## Call:
## lm(formula = log(median_house_value) ~ (distance + bedrooms_ph +
##     rooms_ph + housing_median_age + population + median_income +
##     INLAND + Min_1H_OCEAN + NEAR_OCEAN) - housing_median_age -
##     Min_1H_OCEAN, data = data_w_hlp)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.36284 -0.20031 -0.02406  0.18059  1.91115
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    4.195e+00  4.013e-02 104.545  < 2e-16 ***
## distance      -1.746e-01  3.779e-03 -46.201  < 2e-16 ***
## bedrooms_ph    5.669e-01  3.811e-02  14.875  < 2e-16 ***
## rooms_ph      -4.201e-02  3.849e-03 -10.916  < 2e-16 ***
## population    -1.316e-05  3.746e-06  -3.514 0.000443 ***
## median_income  2.184e-01  2.921e-03  74.775  < 2e-16 ***
## INLAND        -2.848e-01  6.874e-03 -41.432  < 2e-16 ***
## NEAR_OCEAN     1.678e-01  7.845e-03  21.385  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3007 on 16631 degrees of freedom
## Multiple R-squared:  0.6663, Adjusted R-squared:  0.6662
## F-statistic:  4745 on 7 and 16631 DF,  p-value: < 2.2e-16
```

```
BIC_b_w_hlp <- step(model_w_hlp, direction = "backward", k = log(dim(data_w_hlp)[1]))
```

```
## Start:  AIC=-39917.25
## log(median_house_value) ~ (distance + bedrooms_ph + rooms_ph +
##     housing_median_age + population + median_income + INLAND +
##     Min_1H_OCEAN + NEAR_OCEAN) - housing_median_age - Min_1H_OCEAN
##
##                 Df Sum of Sq    RSS    AIC
## <none>                       1503.9 -39917
## - population     1      1.12 1505.0 -39915
## - rooms_ph       1     10.77 1514.7 -39808
## - bedrooms_ph    1     20.01 1523.9 -39707
## - NEAR_OCEAN     1     41.35 1545.2 -39476
## - INLAND         1    155.22 1659.1 -38293
## - distance       1    193.02 1696.9 -37918
## - median_income  1    505.61 2009.5 -35105
```

```
summary(BIC_b_w_hlp)
```

```
##
## Call:
## lm(formula = log(median_house_value) ~ (distance + bedrooms_ph +
##     rooms_ph + housing_median_age + population + median_income +
##     INLAND + Min_1H_OCEAN + NEAR_OCEAN) - housing_median_age -
##     Min_1H_OCEAN, data = data_w_hlp)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.36284 -0.20031 -0.02406  0.18059  1.91115
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    4.195e+00  4.013e-02 104.545  < 2e-16 ***
## distance      -1.746e-01  3.779e-03 -46.201  < 2e-16 ***
## bedrooms_ph    5.669e-01  3.811e-02  14.875  < 2e-16 ***
## rooms_ph      -4.201e-02  3.849e-03 -10.916  < 2e-16 ***
## population    -1.316e-05  3.746e-06  -3.514 0.000443 ***
## median_income  2.184e-01  2.921e-03  74.775  < 2e-16 ***
## INLAND        -2.848e-01  6.874e-03 -41.432  < 2e-16 ***
## NEAR_OCEAN     1.678e-01  7.845e-03  21.385  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3007 on 16631 degrees of freedom
## Multiple R-squared:  0.6663, Adjusted R-squared:  0.6662
## F-statistic:  4745 on 7 and 16631 DF,  p-value: < 2.2e-16
```

```
AIC(AIC_b_w_hlp)
```

```
## [1] 7242.433
```

```
BIC(AIC_b_w_hlp)
```

```
## [1] 7311.908
```

AIC: 7242.433 BIC: 7311.908

Summary Table

```
c_R2 = c(R2, R2_log, R2_log_2, R2_w_hlp)
c_adjustedR2 =  c(adjusted_R2, adjusted_R2_log, adjusted_R2_log_2, adjusted_R2_w_hlp)
c_RSE = c(RSE, RSE_log, RSE_log_2, RSE_w_hlp)
c_AIC = c(AIC(model), AIC(model_log), AIC(model_log_2), AIC(model_w_hlp))
C_BIC = c(BIC(model), BIC(model_log), BIC(model_log_2), BIC(model_w_hlp))

table <- data.frame(R2 = c_R2, adjusted_R2=c_adjustedR2, RSE=c_RSE, AIC=c_AIC, BIC=C_BIC, ro
w.names = c("model", "model_log", "model_log_2","model_w_hlp"))
table
```

```
##                    R2 adjusted_R2        RSE        AIC        BIC
## model       0.5908068   0.5905856 60.0500816 183797.531 183882.463
## model_log   0.6638152   0.6636335  0.3022116   7421.612   7506.544
## model_log_2 0.6637912   0.6636499  0.3022043   7418.803   7488.293
## model_w_hlp 0.6663296   0.6661891  0.3007113   7242.433   7311.908
```

Model evaluation

In order to evaluate the model performance, we compute MSE (Mean Squared Errors) of test set splotting the data in test and training. Mean Squared Error is a metric used to evaluate the performance of a predictive model. It provides a measure of how well the model's predictions align with the actual values or observations, measuring the average squared difference between predicted and actual values.

First of all, we split the data in training and test sets: we decide to use the 80% of data randomly selected as training set and the other 20% as test set. Then we build the linear model with training data and make predictions.

```
# Split the data into train and test and compute MSE

# setting seed to generate a reproducible random sampling
set.seed(569)

# Define training subset
n_train = floor(dim(data_w_hlp)[1]*0.8) # 13243 samples for test(80 % of the data)
i_train <- sample( 1:n, size = n_train, replace = FALSE) # indexes of training samples

# linear model with training data
model_train <- lm(log(median_house_value) ~ . -Min_1H_OCEAN -housing_median_age, data = data_
w_hlp , subset = i_train)

# Prediction based on fitted model
y_pred <- predict(model_train, newdata = data_w_hlp[-i_train, ])

# mean squared error on test data
MSE <- mean((log(data_w_hlp$median_house_value[-i_train])-y_pred)^2)
MSE
```

```
## [1] 0.09299337
```

```
# MSE su log(y) che implica MSE=exp(MSE) su median_house_value/1000
MSE_c = exp(MSE) * 1000
MSE_c
```

```
## [1] 1097.454
```

The output is a MSE equal to 0.09299337 for log(Y) as response variable. In order to have a measure of the error relative to out initial target variable (median house value), we have applied inverse transformations: it lead to a MSE equal to 1097.454. It means that, on average, the predictions made by the model deviate from the actual prices by 1097.454 squared units (dollars)

K-Fold Cross-Validation

K-fold cross-validation is a technique used to evaluate the performance of a predictive model by splitting the data into multiple subsets or folds. It provides a more robust estimation of the model's effectiveness compared to a simple test-train-split approach. K-fold cross-validation is a technique that splits the data into k subsets. The model is trained and tested k times, each time using a different subset as the test set and the remaining subsets as the training set. Performance metrics are calculated for each iteration, and the average metric value provides an overall assessment of the model's performance. This method reduces bias and variance, leading to a more robust evaluation of the model's generalization ability. We decide to use firstly K = 10 folds and then K = 20 to check differences.

```
# Define the value of k
k=10

# Split the subsets of data_w_hlp
folds_10 <- sample(1:k, nrow(data_w_hlp), replace=TRUE)
table(folds_10)
```

```
## folds_10
##    1    2    3    4    5    6    7    8    9   10
## 1686 1638 1666 1647 1647 1654 1689 1685 1702 1625
```

```
# Make the error's vector
cv.errors_10 <- matrix(NA,k,1)
colnames(cv.errors_10) <- "Errors"

# Compute 10-fold Cross Validation and compute the error
for(j in 1:k){
  best.fit <- lm(log(median_house_value)~.-housing_median_age -Min_1H_OCEAN, data=data_w_hlp
[folds_10!=j,])
  test.mat <- model.matrix(log(median_house_value)~.-housing_median_age -Min_1H_OCEAN, data =
data_w_hlp[folds_10==j,])
  y_pred_k <- predict(model_train, newdata = data_w_hlp[folds_10==j,])
  cv.errors_10[j,1] <- mean((log(data_w_hlp$median_house_value[folds_10==j]) - y_pred_k)^2)
}
mean.cv.errors_10 <- apply(cv.errors_10, 2, mean)
names(mean.cv.errors_10) <- c("Error")
mean.cv.errors_10
```

```
##      Error
## 0.09040445
```

```
error_10 = exp(mean.cv.errors_10) * 1000
error_10
```

```
##    Error
## 1094.617
```

The error of 10-fold Cross Validation is 1094.617.

Now we use K = 20.

```
# Define the value of k
k=20

# Split the subsets of data_w_hlp
folds_20 <- sample(1:k, nrow(data_w_hlp), replace=TRUE)
table(folds_20)
```

```
## folds_20
##   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20
## 841 880 787 778 885 788 878 797 882 853 820 820 846 825 848 814 883 799 768 847
```

```
# Make the error's vector
cv.errors_20 <- matrix(NA,k,1)
colnames(cv.errors_20) <- "Errors"

# Compute 20-fold Cross Validation and compute the error
for(j in 1:k){
  best.fit <- lm(log(median_house_value)~.-housing_median_age -Min_1H_OCEAN, data=data_w_hlp
[folds_20!=j,])
  test.mat <- model.matrix(log(median_house_value)~.-housing_median_age -Min_1H_OCEAN, data =
data_w_hlp[folds_20==j,])
  y_pred_k <- predict(model_train, newdata = data_w_hlp[folds_20==j,])
  cv.errors_20[j,1] <- mean((log(data_w_hlp$median_house_value[folds_20==j]) - y_pred_k)^2)
}
mean.cv.errors_20 <- apply(cv.errors_20, 2, mean)
names(mean.cv.errors_20) <- c("Error")
mean.cv.errors_20
```

```
##      Error
## 0.09037392
```

```
# mean.cv.errors_20 e mean.cv.errors_10 errori su log(y)
error_20 = exp(mean.cv.errors_20) * 1000
error_20
```

```
##    Error
## 1094.583
```

The error of 10-fold Cross Validation is 1094.583. 20-folds shows a bit lower error value. Generally, a higher number of folds can provide a more accurate estimation of the model's performance, as each test fold becomes more representative of the data. However, when both cross-validations produce very similar errors, it indicates that the increased subdivision into more folds has not significantly affected the variability of the model's performance.

This implies that the available data is sufficiently representative, and the model has good generalization ability.

RIDGE regression with k-Fold Cross-Validation

RIDGE regression is a shrinkage method. Those methods provide a viable alternative to the subset selection methods discussed earlier. These methods offer a way to reduce model complexity while also serving a regularization purpose. While backward and forward elimination utilize least squares to fit a linear model with a subset of predictors, shrinkage methods fit a model with all p predictors by employing a technique that shrinks the coefficient estimates towards zero. Unlike subset selection methods, which choose a subset of predictors, shrinkage methods retain all predictors in the model. However, they apply a shrinkage or regularization technique that reduces the magnitude of the coefficient estimates. This shrinkage prevents overfitting and helps to address issues such as multicollinearity.

First of all, we create a matrix (X) from the model, remove the intercept column in X and create responses vector. In order to determine the best value of the regularization parameter we perform a 10-fold cross-validation on values. We randomly select a 80% of values and assign them to “train” vector. Then, we create the test vector using all data but those on train vector: these elements are used for testing the trained model. On the training data we perform 10-fold cross validation using RIDGE regression: we use cv.glmnet() that automatically select the best value of regularization parameter (lambda) based on cross-validation. Then we compute RIDGE regression model coefficients at the optimal lambda value.

```
# design matrix
X <- model.matrix(log(data_w_hlp$median_house_value) ~. -housing_median_age -Min_1H_OCEAN, da
ta=data_w_hlp)

# remove the first column relative to the intercept
X <- X[,-1]

# vector of responses
y <- log(data_w_hlp$median_house_value)

# In order to determine the best value of the regularization parameter we perform
# a 10-fold cross-validation on  values

train <- sample(1:nrow(X), nrow(X)*0.8)
y_test <- y[-train]

ridge_model <- cv.glmnet(X[train, ], y[train], alpha = 0, nfold=10) # alpha = 0 regressione R
IDGE
lambda_min <- ridge_model$lambda.min
lambda_min
```

```
## [1] 0.03346832
```

```
coefficient_ridge <- coef(ridge_model, s = "lambda.min")
coefficient_ridge
```

```
## 8 x 1 sparse Matrix of class "dgCMatrix"
##                          s1
## (Intercept)    4.334212e+00
## distance      -1.750514e-01
## bedrooms_ph    3.795451e-01
## rooms_ph      -8.211852e-03
## population    -6.410539e-06
## median_income  1.851018e-01
## INLAND        -2.993836e-01
## NEAR_OCEAN     1.636651e-01
```

We have predicted test set values comparing them with the real ones. Then we computed statistic metrics.

```
# using optimal value of lambda selected by cross-validation
predictions <- predict(ridge_model, newx = X[-train, ], s = "lambda.min")
predictions_exp <- exp(predictions)

y_exp <- exp(y_test)

n_R <- dim(X[-train, ])[1]
n_R
```

```
## [1] 3328
```

```
p_R <- dim(X)[2]
p_R
```

```
## [1] 7
```

```
# Residual Sum of Squares
RSS <- sum((y_test - predictions)^2)
RSS
```

```
## [1] 302.3859
```

```
# Explained Sum of Squares
ESS <- sum((predictions - mean(y_test))^2)
ESS
```

```
## [1] 534.1679
```

```
# Total Sum of Squares
TSS <- ESS + RSS
TSS
```

```
## [1] 836.5538
```

```
# Residual Standard Error
RSE <- sqrt(RSS/(n_R - p_R - 1))
RSE
```

```
## [1] 0.3017948
```

```
# R Squared statistic
R2_R <- 1 - RSS/TSS
R2_R
```

```
## [1] 0.6385339
```

```
# adjusted R square
adjR2_R <- 1 - (1-R2_R)*((n_R-1)/(n_R-p_R-1))
adjR2_R
```

```
## [1] 0.6377717
```

```
#MSE (both using RSS and its definition)
MSE_2 <- RSS/n_R
MSE_2
```

```
## [1] 0.09086114
```

```
MSE_R <- mean((predictions - y_test)^2)
MSE_R
```

```
## [1] 0.09086114
```

```
MSE_R_c = exp(MSE_R) * 1000
MSE_R_c
```

```
## [1] 1095.117
```

ANOVA

In this part our aim it to apply ANOVA to evaluate is there are significant differences between means of different groups of data. In order to compute ANOVA, a foundamental assumption is homoscedasticity between groups. To verify this assumption we use Bartlett test.

```
dataset_last <- data[!(rownames(dataset_with_dummy) %in% high_leverage_points),]

ocean_proximity <- as.factor(dataset_last$ocean_proximity)
contrasts(ocean_proximity)
```

```
##            INLAND NEAR BAY NEAR OCEAN
## <1H OCEAN       0        0          0
## INLAND          1        0          0
## NEAR BAY        0        1          0
## NEAR OCEAN      0        0          1
```

```
# checking homoscedasticity between categorical variables
bartlett.test(log(median_house_value) ~ ocean_proximity , data = dataset_last)
```

```
##
##  Bartlett test of homogeneity of variances
##
## data:  log(median_house_value) by ocean_proximity
## Bartlett's K-squared = 327.2, df = 3, p-value < 2.2e-16
```

The result shows that p-value is less than alpha = 0.05. Therefore the null hypotesis is not verified. It means that ANOVA cannot be applied.

Conclusions

To summarize, we built a multiple linear regression model in order to forecast the trend of the response variable Median House Value as a function of specific metrics. To do that, we worked on the dataset cleaning and filtering data in order to make analysis on them. Then, we detect the meaning of each variable at a time in order to investigate on their information capacity. Therefore, we decided to create a new variable called “distance” using “longitude” and “latitude”. Checking correlations between variables, we noticed high values of correlation between some of them. Therefore we divided two of them by the third creating more informative variables: “rooms_ph” and “bedrooms_ph”. At this point of analysis we examined the distribution of each feature, detecting outliers presence and dropping them according to Interquartile Range rule. The second part of the analysis consists on building the best linear regression model, using backward stepwise selection AIC and BIC to find the most suitable variables and checking statistical metrics to evaluate each model created. Plots about residuals show the presence of high leverage points: so, we dropped them and compute a new model with the appropriate transformation of response variable (log-transformation), without high leverege and without those variables excluded by stepwise. Then we evaluated the goodness-in-prediction of our model splitting the dataset in test and train subtest and computing MSE. Moreover, we made two K-fold cross validation with different values of K and computed respective errors. Finally, we applied a RIDGE regression method to improve our model.