statistical_project

2024-05-27

Project Introduction

In this project we aim to study the influence of macroeconomics indicators on the house price index in Colombia. This indicator measure the evolution through time of the market prices of residential properties as a percentage change. The idea of the analysis is to present and study the possible effects of the macroeconomics indicators on the house prices through the construction of regression models, evaluating their results and interpreting the models in the context of the data.

Dataset Introduction and Preprocessing

The dataset was constructed gattering information from multiple sites: the Colombian department of statistics (DANE) , the Colombian central bank (Banco de la República), Google trends and the Federal Reserve Economic Data (FRED). In general, all the visited sites present the data as a .csv file with tables according to filters related with the time span of interest. With this we construct a consolidated database in .csv format with the following variables:

Variable	Description
House Price Index	Measure of the average change in the residential properties price as an index
Industrial Inputs Index	Measure of the average changes in the industry input costs as an index
Metals Price Index	Measure of the average price change in metal costs excluding gold as an index
Energy Price Index	Measure of the average price change of energy including Crude oil, Natural Gas, Coal Price and Propane Indices
Shipping Price Index	Measure the average price change of shipping costs
Forex Index	Indicator of the foreign exchange rate of the Colombian Peso (COP) with respect of the USD
Unemployment Rate	Indicator of the percentage of unemployment population in Colombia
Industrial Production Index	Measure of the level of production on industrial sectors as an index
Interest rate	Reference interest rate emited by the colombian central bank with respect of the others financial institutions
Construction Licences Area	Measure the total of squared meters given for construction licences in Colombia
Finished Constructions	Measure of the total constructions that have been finished in Colombia
Google Trends Housing	Google search trends on housing for Colombia

These variables present real values,and were selected as they have an initial coherent relation with the housing sector. Initially We present the import process of the data

Data Uploading

```
#Read the data using the read_excel function
#Change here for current data path
data <- read_excel("C:/Users/CAMILO/Documents/GitHub/unipd_sl_24/data/data.xlsx",sheet = "dataframe_col
```

Data pre-processing and cleaning

After have an initial version of the data we proceed with the pre-processing and cleaning steps. As firs step we change the original variable names for more appropriate ones. Then we identify the number of NA values in each one of the columns.

```
#change the variables names to have more consistency
colnames(data) <- c('MY','date','House_Price_Index','Industrial_Inputs_Index','Metals_Price_Index','Ene
# Counting NA values in each column
na_counts <- apply(data, 2, function(x) sum(is.na(x)))
# Print the counts of NA values per column
print(na_counts)
```

```
##                          MY                        date
##                           0                           0
##           House_Price_Index     Industrial_Inputs_Index
##                           0                           0
##          Metals_Price_Index          Energy_Price_Index
##                           0                           0
##        Shipping_Price_Index                 Forex_Index
##                           0                           0
##           Unemployment_Rate Industrial_Production_Index
##                           0                          12
##               Interest_Rate  Construction_Licences_Area
##                           0                          12
##      Finished_Constructions       Google_Trends_Housing
##                         158                           0
```

```
# Convert date column to Date type if it's not already
data$date <- as.Date(data$date)
# Copy the dataframe to avoid modifying the original one
data_filled <- data
```

We realize that the variable Finished_Constructions requires an additional pre-processing step because it is recorded by quarters while the others are registered by months. We propose the following reconstruction steps for those intermediate months. The idea is to take the quarterly value and divide it equally among the respective months of that quarter:

```
# Find the indices where the date is at the end of a trimester
trimester_end_indices <- which(format(data$date, "%m") %in% c("03", "06", "09", "12"))
# Loop through each trimester end distribute the value
for (i in trimester_end_indices) {
  if (i - 2 > 0) {
    # Distribute the value to the current month and the previous two months
```

```
    value_to_distribute <- data$Finished_Constructions[i] / 3
    data_filled$Finished_Constructions[i] <- value_to_distribute
    data_filled$Finished_Constructions[i - 1] <- value_to_distribute
    data_filled$Finished_Constructions[i - 2] <- value_to_distribute
  }
}
data_filled$Finished_Constructions <- na.locf(data_filled$Finished_Constructions, na.rm = FALSE)
```

Summary stats and final dataset

We proceed to remove the left rows with NA values eliminating from the dataset the registers before January 2005. Also, we drop the column MY also referred to the date in form of MONTH()YEAR(). Finally, we present an initial summary of the dataset after these transformations. In particular we see that the House Price Index range present values on [87.35,203.50] with an IQR=70.38.

```
# Removing rows with any NA values
clean_data <- na.omit(data_filled)
# Drop MY column
data_reduced <- subset(clean_data, select = -c(MY))
```

```
summary_stats <- summary(data_reduced[, !(names(data_reduced) %in% c("date"))])
```

Statistic	Min	1st Qu.	Median	Mean	3rd Qu.	Max
House_Price_Index	87.35	110.50	157.92	147.71	180.88	203.50
Industrial_Inputs_Index	78.67	116.02	136.26	137.78	157.65	221.14
Metals_Price_Index	74.89	121.21	143.74	145.88	172.53	234.53
Energy_Price_Index	55.89	128.19	162.61	173.36	223.35	376.41
Shipping_Price_Index	-213.3	242.2	262.1	280.7	294.1	478.3
Forex_Index	0.1712	1.1934	2.3762	2.6762	3.2021	4.9223
Unemployment_Rate	0.07563	0.09728	0.10970	0.11240	0.12099	0.21972
Industrial_Production_Index	59.82	77.94	94.44	92.87	104.85	132.46
Interest_Rate	0.01750	0.03860	0.04525	0.05551	0.06994	0.13250
Construction_Licences_Issued	233210	969559	1156424	1205587	1373585	2883878
Finished_Constructions	471310	775374	848663	871488	962168	1335811
Google_Trends_Housing	16.00	23.50	34.75	36.04	47.62	64.00

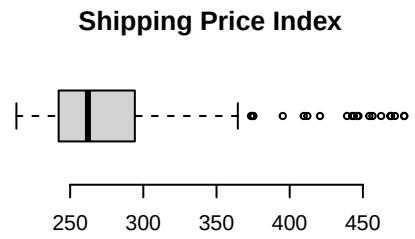
Exploratory Analysis

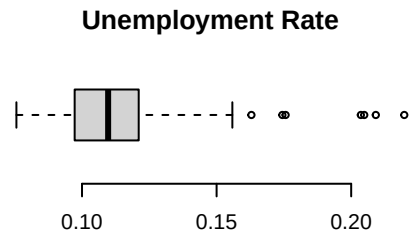
Outliers based on the IQR

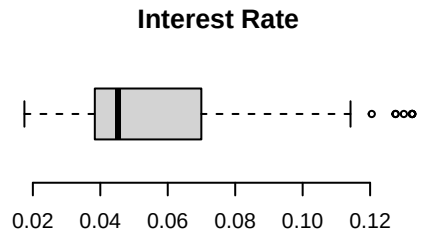
As an initial step to better understand the data and its behavior we find the outliers of each one of the variables using the IQR to select those points outside the bounds. As a result we found that the variables that present the most quantity of outliers are the Shipping Price Index, the Interest Rate and the Unemployment Rate. In addition, we present the boxplot of these three variables to obtain a clear summary of their distribution.

```r
# Function to detect outliers based on IQR
detect_outliers_single_var <- function(column) {
  column <- na.omit(column)
  Q1 <- quantile(column, 0.25, na.rm = TRUE)
  Q3 <- quantile(column, 0.75, na.rm = TRUE)
  IQR <- Q3 - Q1
  lower_bound <- Q1 - 1.5 * IQR
  upper_bound <- Q3 + 1.5 * IQR
  outliers <- column[column < lower_bound | column > upper_bound]
  return(outliers)
}
# List to store outliers for each variable
outliers_list <- list()
# Loop to determine outliers in each column
for (var in colnames(data_reduced[, !(names(data_reduced) %in% c("date"))])) {
  if (is.numeric(data[[var]])) {
    outliers_list[[var]] <- detect_outliers_single_var(data[[var]])
  }
}
print(outliers_list)
```

Variable	Values
House_Price_Index	0
Industrial_Inputs_Index	0
Metals_Price_Index	0
Energy_Price_Index	376.4121
Shipping_Price_Index	374.969, 375.187, 395.268, 446.983, 439.181, 454.407, 468.721, 478.263, 478.216, 471.877, 469.290, 442.591, 456.584, 412.082, 409.599, 443.799, 420.694, 446.214, 462.462
Forex_Index	0
Unemployment_Rate	0.1725640, 0.2048530, 0.2197200, 0.2035910, 0.2091470, 0.1744270, 0.1629280, 0.1756237
Industrial_Production_Index	0
Interest_Rate	0.1141935, 0.1204839, 0.1275000, 0.1275806, 0.1300000, 0.1324194, 0.1325000, 0.1325000
Construction_Licences_Area	2493085, 1988396, 2838778, 2337743, 2331210, 2859035, 2372414, 2052607
Finished_Constructions	4007432, 3762696, 1413931
Google_Trends_Housing	0

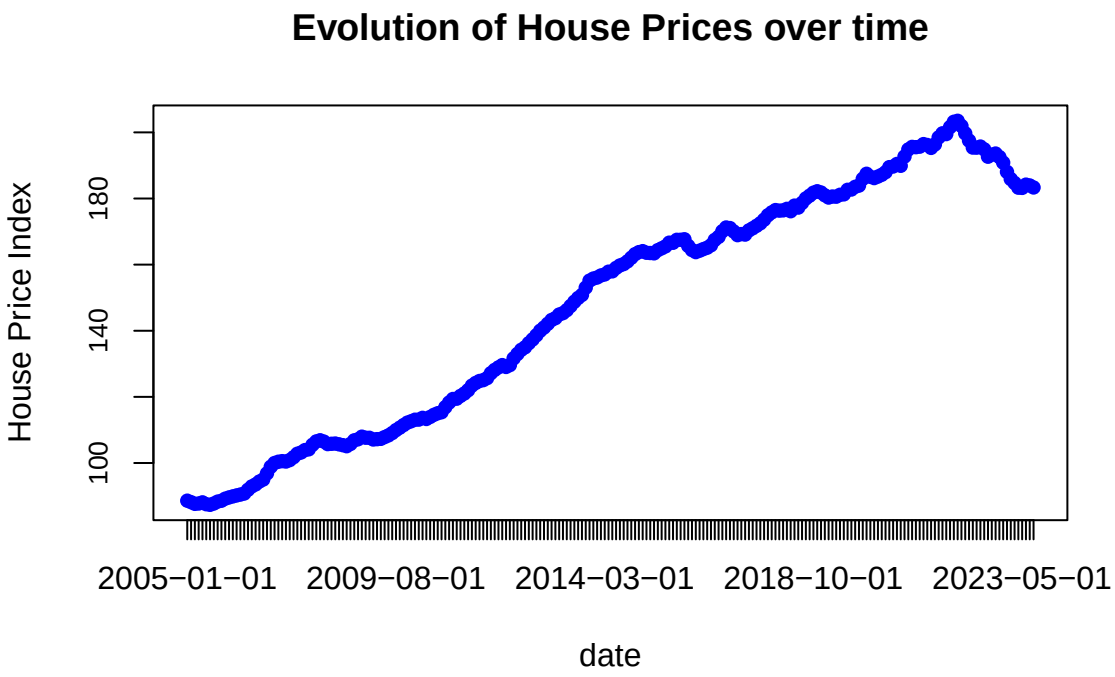






Ploting the House Price Index

Continuing with the exploratory analysis we plot the variable of House Price Index to see its behavior through time. It is possible to see a clear increase tendency starting from the year 2005 until the year 2021, then a little decay was recorded.



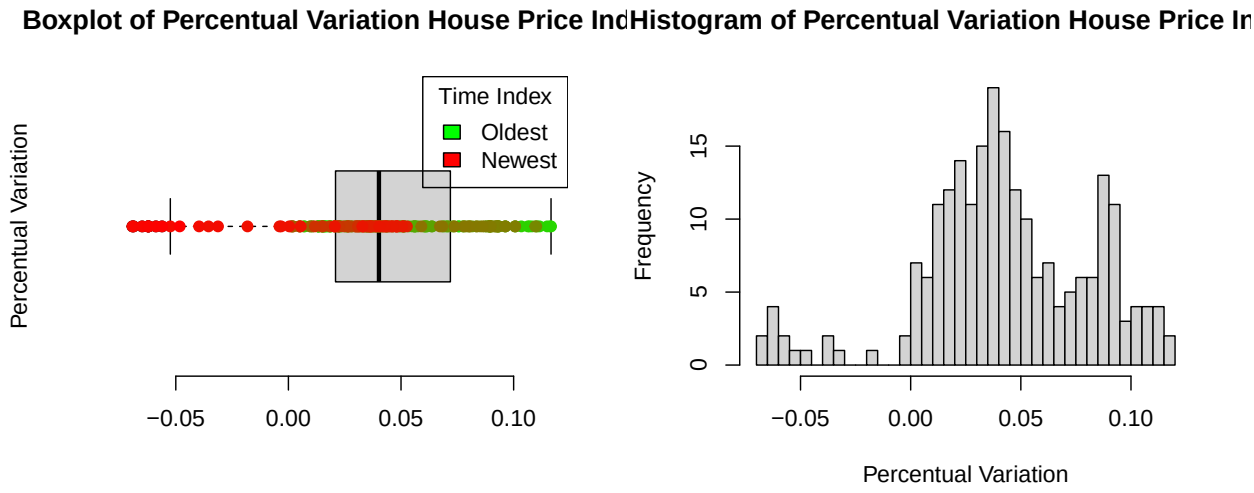
Ploting percentual variation of the House Price Index

Another useful information can be found in the box plots and histograms for the percentual variations. We take the percentual variation of the variables with the intention of identify patterns and tendencies in a clearly way, considering that the majority of it is composed by index, and the study of it changes is fundamental for a complete understanding of their behavior through time. We start with the House Price Index variable.

In these plots we see that the central value of the percentual variations is positive, indicating and overall increase in the House Price Index through time. Also the negatives outliers were expected because those negatives variations due to the decrease after the 2021. In the histogram it is possible to see a right skewed behavior, give by precisely those increments until the 2021.

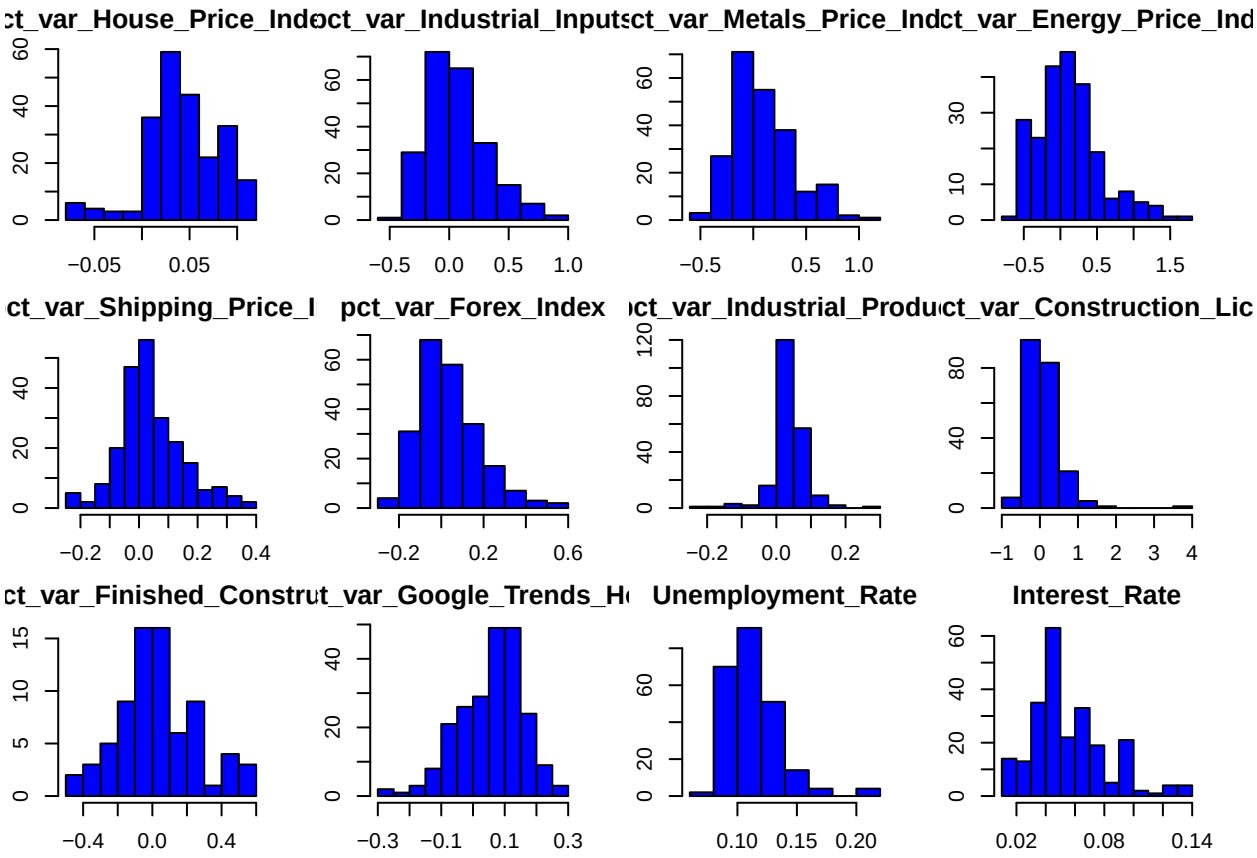
```
# Select subset of variables to difference
variables_to_calculate <- c("House_Price_Index", "Industrial_Inputs_Index", "Metals_Price_Index",
                            "Energy_Price_Index", "Shipping_Price_Index", "Forex_Index",
                            "Industrial_Production_Index", "Construction_Licences_Area",
                            "Finished_Constructions", "Google_Trends_Housing")

# Create a function to calculate the 12-month percentual variation
percentual_variation_12_months <- function(x) {
  return((x / lag(x, n = 12) - 1))
}
lag <- function(x, n) {
  c(rep(NA, n), head(x, -n))
}
data_percentual_variation <- data
# Apply the function to the select variables
```

```
for (var in variables_to_calculate) {
  new_var_name <- paste0("pct_var_", var)
  data_percentual_variation[[new_var_name]] <- percentual_variation_12_months(data[[var]])
}
```

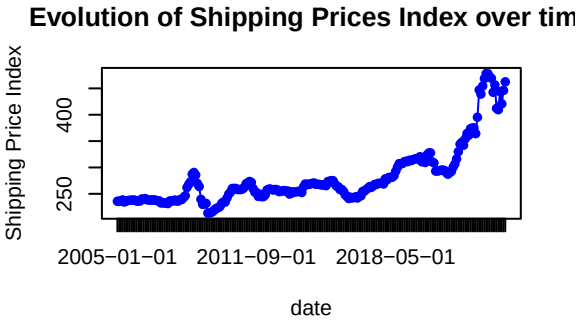


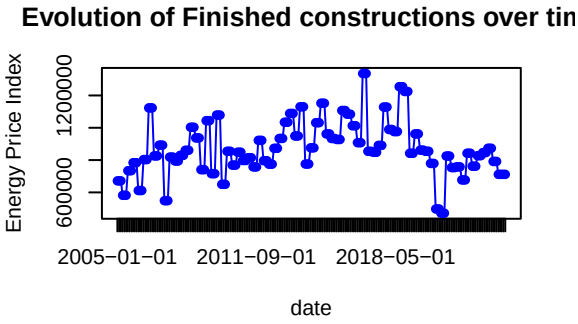
We did a similar procedure for the rest of the variables with respect to their percentual variation. For an easier interpretation we plot the histogram for the percentual variation of each variable. This plot shows that the index variables with a high variability in percentual variation are the shipping index price and the finished constructions. Also, the index variable that show less variability in the percentual variation is the one related with the industrial production.

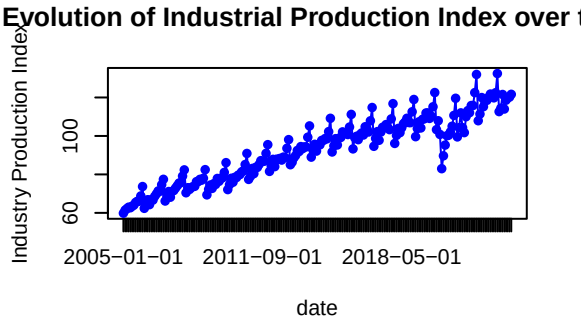
```
{par(mfrow = c(3,4 ), mar = c(2, 2, 2, 2) + 0.1)
data_percentual_variation <- data_percentual_variation[, columns_to_select]
for (i in 1:ncol(data_percentual_variation)) {
  hist(data_percentual_variation[[i]],
       main = substr(names(data_percentual_variation)[i], 1, 25), # Shorten title if necessary
       xlab = names(data_percentual_variation)[i],
       col = "blue",
       border = "black")
}
}
```



We plot these four variables to see its evolution through time. It is possible to see that the industrial production Index and the Shipping price Index have an increase tendency, with some peaks as for example the negative peak related with the COVID pandemic for the Industrial Production Index on the 2020. In another hand, the Finished Constructions have a less defined behavior over time.







Log-transformation

We apply a log transformation with the intention of normalize the data as some variables are index that change over time and others area real values in a wide range. Our idea is to reduce the variability of the data, compressing the scale and helping to linearize those relationships between the variables for the interpretability of the results in the subsequent models. We apply log transformation to all variables except those that are already percentages: the Unemployment Rate and the Interest Rate.

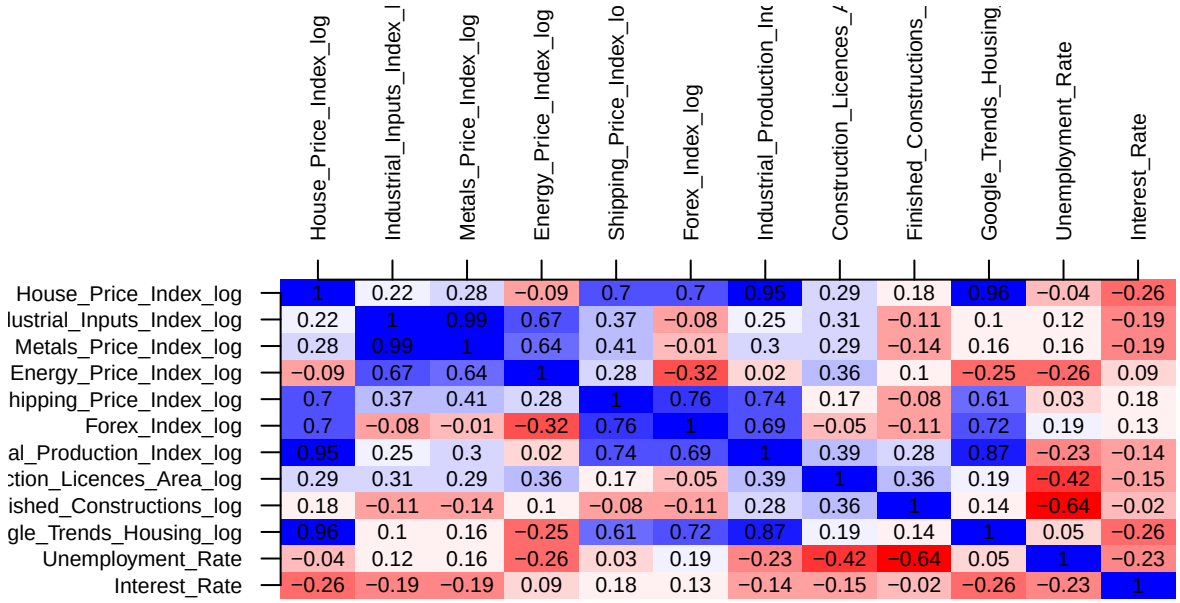
```
#Log transformation of the variables
variables_to_log <- c("House_Price_Index", "Industrial_Inputs_Index", "Metals_Price_Index",
                      "Energy_Price_Index", "Shipping_Price_Index", "Forex_Index",
                      "Industrial_Production_Index", "Construction_Licences_Area",
                      "Finished_Constructions", "Google_Trends_Housing")
```

```r
for (var in variables_to_log) {
  log_var_name <- paste0(var, "_log")
  data_reduced[[log_var_name]] <- log(data_reduced[[var]])
}

# Subset the dataframe
selected_columns <- c(paste0(variables_to_log, "_log"), "Unemployment_Rate", "Interest_Rate")
data_final <- data_reduced[, selected_columns]
```

Variable Correlation

Now we present the correlation between the variables of the dataset in order to understand the strength of their relation. We present the correlation matrix obtained. From the results we see a strong positive relationship between the House Price Index and the Google Trend and Industrial Production one,a more moderate correlation with the shipping Price and the Forex Index. While the weaker relationships are found with the Energy Price index and the Unemployment Rate.For the others variables, we found a high positive correlation between the metal price index and the industrial inputs.

```r
# Plot the correlation matrix

par(mar = c(7, 7, 7, 2) + 0.1)
cor_matrix<-correl_matrix
image(1:ncol(cor_matrix), 1:nrow(cor_matrix), t(cor_matrix)[, ncol(cor_matrix):1],
      axes = FALSE, xlab = "", ylab = "", col = colorRampPalette(c("red", "white", "blue"))(20))

# Add axis labels
axis(3, at = 1:ncol(cor_matrix), labels = colnames(cor_matrix), las = 2, cex.axis = 0.7)
axis(2, at = 1:nrow(cor_matrix), labels = rev(rownames(cor_matrix)), las = 2, cex.axis = 0.7)

#correlation coefficients
for (i in 1:ncol(cor_matrix)) {
  for (j in 1:nrow(cor_matrix)) {
    text(i, nrow(cor_matrix) - j + 1, round(cor_matrix[i, j], 2), cex = 0.7)
  }
}
```



Multicolinearity Check

Now present a multicollinearity test in top of the results related with the correlation of the variables in relation with the dependent variable of House Price Index. We first calculate the VIF for each predictor variable. We found that some variables like Shipping, Industrial Production Index, Metals Price Index and Industrial Inputs present a high VIF.

```
##     Industrial_Inputs_Index_log          Metals_Price_Index_log
##                      232.372573                      237.983290
##          Energy_Price_Index_log        Shipping_Price_Index_log
##                        6.176604                       12.188511
##                 Forex_Index_log Industrial_Production_Index_log
##                       10.143101                       17.856757
##  Construction_Licences_Area_log      Finished_Constructions_log
##                        1.807047                        2.148915
##       Google_Trends_Housing_log               Unemployment_Rate
##                        8.171922                        4.286584
##                   Interest_Rate
##                        1.982209
```

At this point we consider to remove the Metals Price Index as they show a strong correlation with Industrial Inputs, they both have a similar behavior and at the end the metal sector is consider on the Industrial Inputs Index. After this reduction we re run the VIF test. EN ESTE PUNTO HABLAR DE POR QUÉ DEJAR ALGUNAS AUNQUE TENGA VIF ALTO.

```
##     Industrial_Inputs_Index_log          Energy_Price_Index_log
##                        3.479427                        6.171898
##        Shipping_Price_Index_log                Forex_Index_log
##                       11.954143                       10.092636
## Industrial_Production_Index_log  Construction_Licences_Area_log
##                       14.752680                        1.698461
##      Finished_Constructions_log       Google_Trends_Housing_log
##                        2.102236                        8.037613
##               Unemployment_Rate                   Interest_Rate
##                        3.673045                        1.900964
```

Train Test Split

Before proceed with the linear models we apply a train test split in order to have a test set to compare the performance in a fair way between the data. For this we implement the following steps, considering that our train set will be 80% of the data:

```
set.seed(123)
# sample index for training data
trainIndex <- sample(seq_len(nrow(data_final)), size = 0.8 * nrow(data_final))
# Split the data into training and testing sets
data_train <- data_final[trainIndex, ]
data_test <- data_final[-trainIndex, ]

X_train <- model.matrix(House_Price_Index_log ~ ., data = data_train)[, -1]
y_train <- data_train$House_Price_Index_log
X_test <- model.matrix(House_Price_Index_log ~ ., data = data_test)[, -1]
y_test <- data_test$House_Price_Index_log
```

Modelling the data

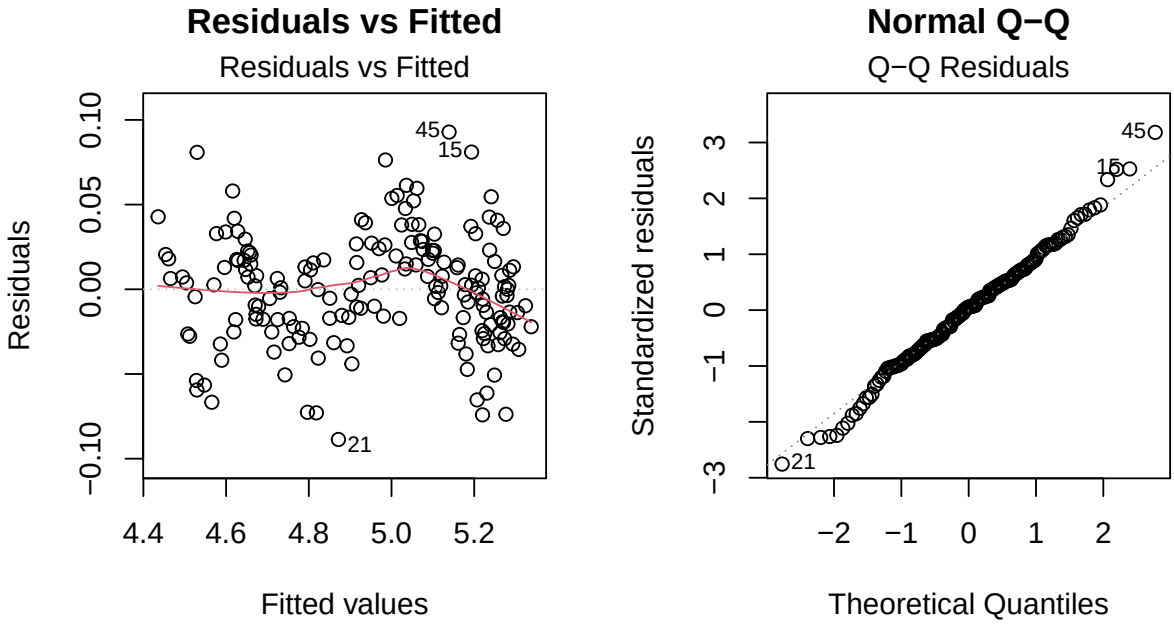
#Initial full linear model We start with a simple regression model for the data, remember that this process will be executed with the variables after the log normalization. In this first iteration we consider all the explanatory variables.

```
# OLS 1 Log-Log------------------
# Perform linear regression
model <- lm(House_Price_Index_log ~ ., data = data_train)
summary(model)
```

```
##
## Call:
## lm(formula = House_Price_Index_log ~ ., data = data_train)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.088670 -0.019984  0.001724  0.019942  0.092810
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
```

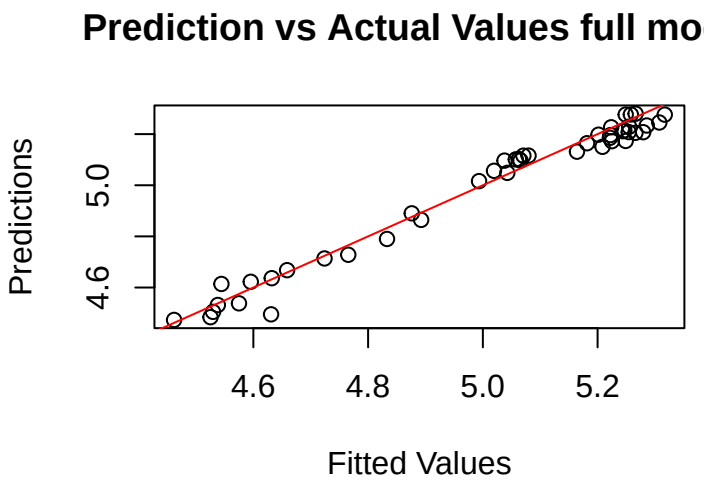
```
## (Intercept)                         0.79395    0.31588   2.513 0.012898 *
## Industrial_Inputs_Index_log        -0.03332    0.01994  -1.671 0.096516 .
## Energy_Price_Index_log              0.01794    0.01808   0.992 0.322477
## Shipping_Price_Index_log            0.07971    0.04815   1.656 0.099658 .
## Forex_Index_log                    -0.07641    0.02797  -2.732 0.006973 **
## Industrial_Production_Index_log     0.80968    0.05384  15.040  < 2e-16 ***
## Construction_Licences_Area_log     -0.02418    0.01060  -2.282 0.023760 *
## Finished_Constructions_log         -0.01036    0.01879  -0.551 0.582041
## Google_Trends_Housing_log           0.32995    0.01873  17.620  < 2e-16 ***
## Unemployment_Rate                   0.76654    0.23145   3.312 0.001134 **
## Interest_Rate                      -0.47954    0.12981  -3.694 0.000298 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.03321 on 168 degrees of freedom
## Multiple R-squared:  0.9846, Adjusted R-squared:  0.9837
## F-statistic:  1075 on 10 and 168 DF,  p-value: < 2.2e-16
```

In this first model we see a high R-squared, this mean that the model is powerful in terms of explain the variance in the House Price Index, but the model is no completely satisfactory as it includes several predictors without statistical significance, and other with different degree of significance as industrial production, google trends and interest rate. This can be explain because the R-squared statistic will always increase when more variables are added to the model. We proceed to plot the the results for the fitted results and the QQ-plot of the residuals:



After having the model we fit it to evaluate its performance. First we determine the MSE which has a value of 0.00269 suggesting the the predictions are close to the actual values, this could be observed in the plot of the predicted vs the actual values which presents an approximately 45° degrees line.

```
## [1] 0.00186722
```



Model dropping variables according to their significance

After the first iteration we consider the process of eliminating variables based on its significance to the model. The first variable that we consider to discard is the Energy Price Index, for this we proceed with an ANOVA F-test for the comparison of regression models. As the obtained value is 0.1328 is greater than 0.05 the Energy Price Index is not a significant predictor in the presence of the other variables. With this we remove it

```
##
## Call:
## lm(formula = House_Price_Index_log ~ ., data = data_train)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.088670 -0.019984  0.001724  0.019942  0.092810
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                      0.79395    0.31588   2.513 0.012898 *
## Industrial_Inputs_Index_log     -0.03332    0.01994  -1.671 0.096516 .
## Energy_Price_Index_log           0.01794    0.01808   0.992 0.322477
## Shipping_Price_Index_log         0.07971    0.04815   1.656 0.099658 .
## Forex_Index_log                 -0.07641    0.02797  -2.732 0.006973 **
## Industrial_Production_Index_log  0.80968    0.05384  15.040  < 2e-16 ***
## Construction_Licences_Area_log  -0.02418    0.01060  -2.282 0.023760 *
## Finished_Constructions_log      -0.01036    0.01879  -0.551 0.582041
## Google_Trends_Housing_log        0.32995    0.01873  17.620  < 2e-16 ***
## Unemployment_Rate                0.76654    0.23145   3.312 0.001134 **
## Interest_Rate                   -0.47954    0.12981  -3.694 0.000298 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.03321 on 168 degrees of freedom
```

```
## Multiple R-squared:  0.9846, Adjusted R-squared:  0.9837
## F-statistic:  1075 on 10 and 168 DF,  p-value: < 2.2e-16
```

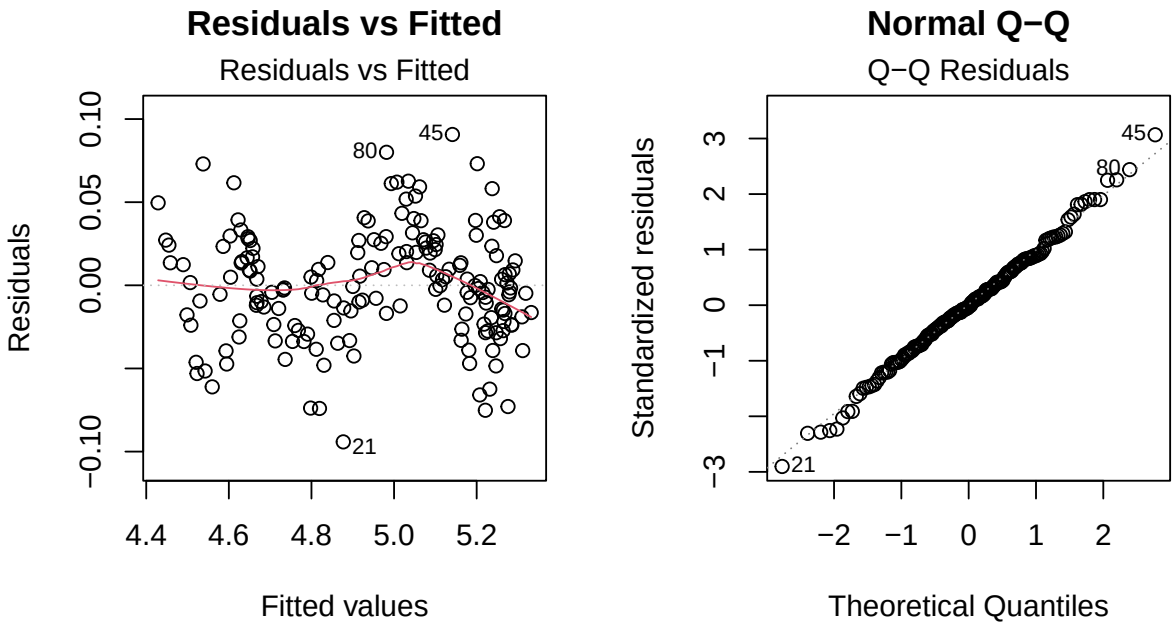
```
## Analysis of Variance Table
##
## Model 1: House_Price_Index_log ~ Industrial_Inputs_Index_log + Shipping_Price_Index_log +
##     Forex_Index_log + Industrial_Production_Index_log + Construction_Licences_Area_log +
##     Finished_Constructions_log + Google_Trends_Housing_log +
##     Unemployment_Rate + Interest_Rate
## Model 2: House_Price_Index_log ~ Industrial_Inputs_Index_log + Energy_Price_Index_log +
##     Shipping_Price_Index_log + Forex_Index_log + Industrial_Production_Index_log +
##     Construction_Licences_Area_log + Finished_Constructions_log +
##     Google_Trends_Housing_log + Unemployment_Rate + Interest_Rate
##   Res.Df     RSS Df Sum of Sq      F Pr(>F)
## 1    169 0.18643
## 2    168 0.18534  1 0.0010863 0.9847 0.3225
```

We observe again the significance of the variables, and the next no significant variables that we decide to drop were Industrial Inputs Index, Finished Constructions, Construction Licence Area and Unemployment. All of this after eliminating step by step each one following the previous procedure with the ANOVA F-test until all the predictors are significative. Justificar porque dejamos constructions licences aunque no es significativa

```
#Process of Variable dropping according to ther significance

# Energy does not improve model
summary(red.mod)
red.mod2 <- update(red.mod, . ~ . -Industrial_Inputs_Index_log)
anova(red.mod, red.mod2)
# Can safely remove industrial inputs log
summary(red.mod2)

# Remove finished_constructions_log
red.mod3 <- update(red.mod2, . ~ . -Finished_Constructions_log)
anova(red.mod2, red.mod3)
# Can safely remove finished constructions (but makes no sense)
summary(red.mod3)
```

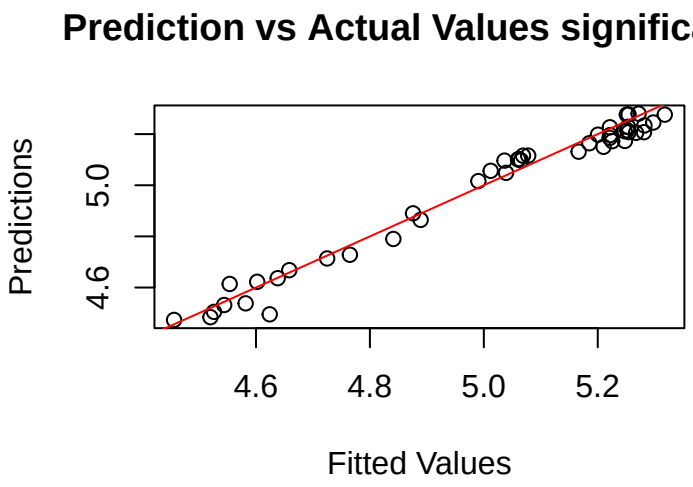
```
##
## Call:
## lm(formula = House_Price_Index_log ~ Shipping_Price_Index_log +
##     Forex_Index_log + Industrial_Production_Index_log + Construction_Licences_Area_log +
##     Google_Trends_Housing_log + Unemployment_Rate + Interest_Rate,
##     data = data_train)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.094159 -0.021217 -0.000685  0.021851  0.090717
##
## Coefficients:
##                                Estimate Std. Error t value Pr(>|t|)
## (Intercept)                     0.64293    0.16678   3.855 0.000164 ***
## Shipping_Price_Index_log        0.09061    0.02805   3.231 0.001481 **
## Forex_Index_log                -0.06931    0.01801  -3.849 0.000167 ***
```

```
## Industrial_Production_Index_log  0.77431    0.04870  15.901  < 2e-16 ***
## Construction_Licences_Area_log  -0.02490    0.01038  -2.399 0.017496 *
## Google_Trends_Housing_log        0.33191    0.01750  18.968  < 2e-16 ***
## Unemployment_Rate                0.61628    0.18912   3.259 0.001351 **
## Interest_Rate                   -0.46996    0.12859  -3.655 0.000342 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.03321 on 171 degrees of freedom
## Multiple R-squared:  0.9843, Adjusted R-squared:  0.9837
## F-statistic:  1535 on 7 and 171 DF,  p-value: < 2.2e-16
```



The final reduced model continue to show a high R-squared, while the QQ-Plot shows similar points outside the reference line. At this point even the fact that now all the variables are significative the model appears to has more or less the same explanatory power due to the residuals behaviour.

```
## [1] 0.001830836
```



Correction of the model with lagged value

As a following step we implement a procedure to consider a lag version of the House Price Index, this due to the possible influence of lagged affects, in terms of the intuition that the actual house price index will not be determined by the current explanatory variables, but by the explanatory variables of the previous period.

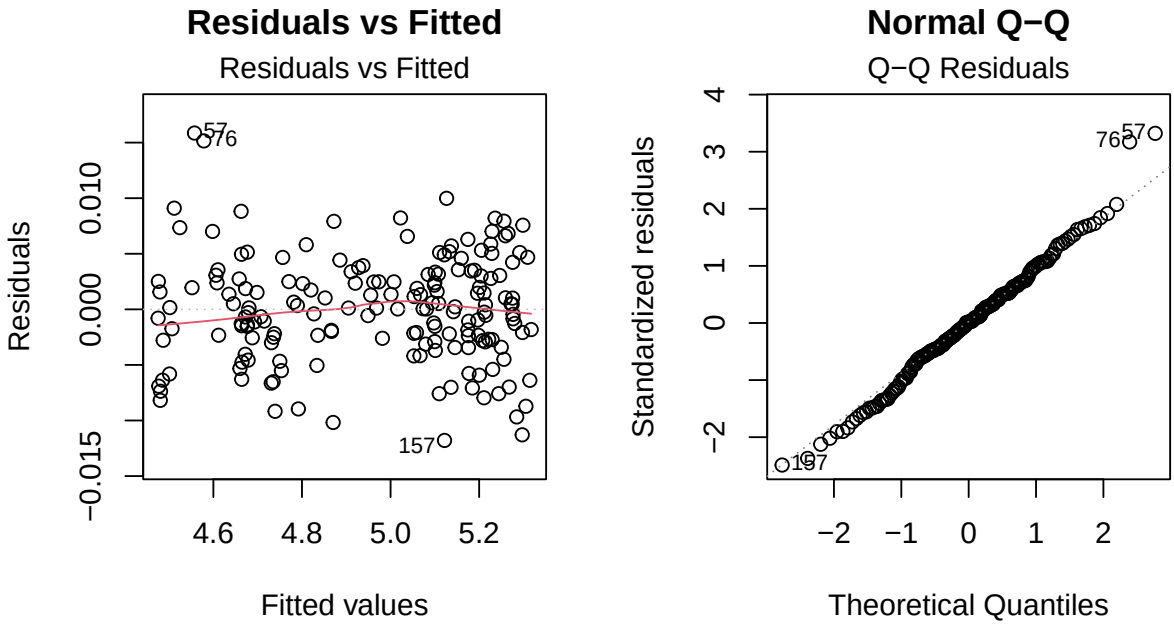
```
# Using a a lag version of the House Price Index

data_final_with_lag<- data_final

data_final_with_lag$House_Price_Index_log_lag1 <- c(NA, head(data_final$House_Price_Index_log, -1))

data_final_with_lag <- na.omit(data_final_with_lag)
```

```
# Set the seed for reproducibility
set.seed(123)

trainIndex_lag <- sample(seq_len(nrow(data_final_with_lag)), size = 0.8 * nrow(data_final_with_lag))

# Split the data into training and testing sets
data_train_lag <- data_final_with_lag[trainIndex_lag, ]
data_test_lag <- data_final_with_lag[-trainIndex_lag, ]


X_train_lag <- model.matrix(House_Price_Index_log ~ ., data = data_train_lag)[, -1]
y_train_lag <- data_train_lag$House_Price_Index_log


X_test_lag <- model.matrix(House_Price_Index_log ~ ., data = data_test_lag)[, -1]
y_test_lag <- data_test_lag$House_Price_Index_log
```

```r
model_lag <- lm(House_Price_Index_log ~ House_Price_Index_log_lag1 +Forex_Index_log+Industrial_Productio

# Summarize the model
summary(model_lag)
```

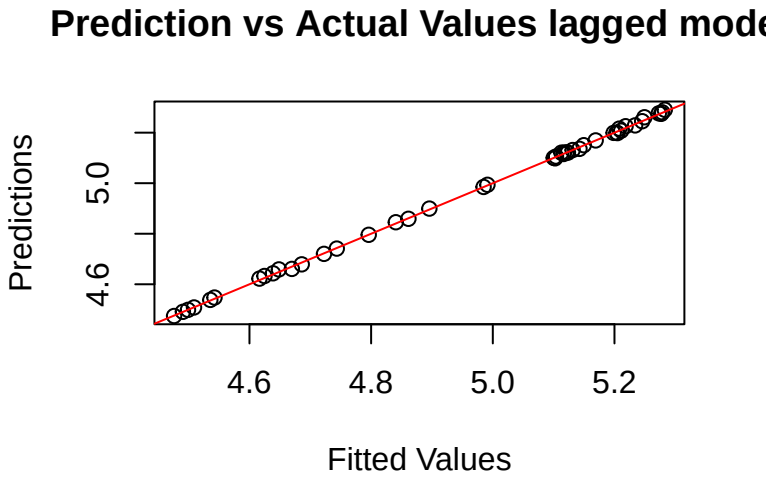
```
##
## Call:
## lm(formula = House_Price_Index_log ~ House_Price_Index_log_lag1 +
##     Forex_Index_log + Industrial_Production_Index_log + Construction_Licences_Area_log +
##     Google_Trends_Housing_log + Interest_Rate, data = data_train_lag)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.011794 -0.002766  0.000083  0.003037  0.015861
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                     0.110388   0.023991   4.601 8.15e-06 ***
## House_Price_Index_log_lag1      0.963768   0.009419 102.321  < 2e-16 ***
## Forex_Index_log                -0.009077   0.002165  -4.193 4.40e-05 ***
## Industrial_Production_Index_log  0.037624   0.008286   4.540 1.06e-05 ***
## Construction_Licences_Area_log -0.003591   0.001491  -2.408   0.0171 *
## Google_Trends_Housing_log       0.008054   0.004049   1.989   0.0483 *
## Interest_Rate                  -0.079478   0.017003  -4.674 5.96e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.00486 on 171 degrees of freedom
## Multiple R-squared:  0.9997, Adjusted R-squared:  0.9996
## F-statistic: 8.361e+04 on 6 and 171 DF,  p-value: < 2.2e-16
```

After include the lagged effect we consider the reduced model showing a high R-squared. Then we proceed to show the histogram of the residuals and their QQ plot. From the QQ-plot it is possible to see that now the residuals show a even strong normal behavior



Now, it is possible to see that the fit of the predictions and actual values is more correct, in terms that it follows almost a a perfect straight line.

```
## [1] 2.317564e-05
```



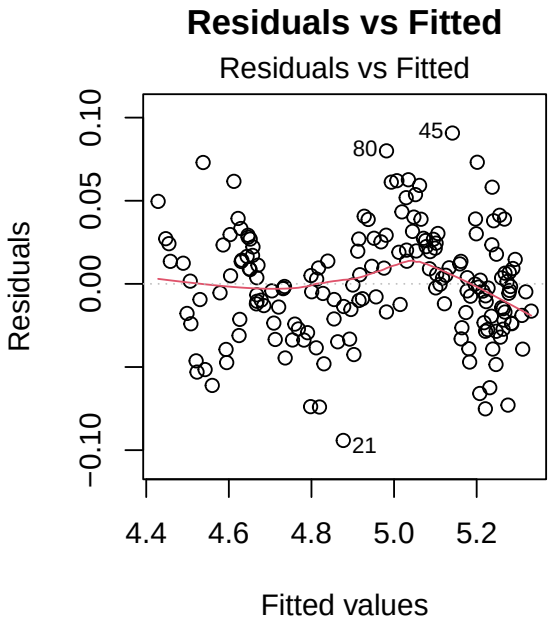
Forward AIC Selection model

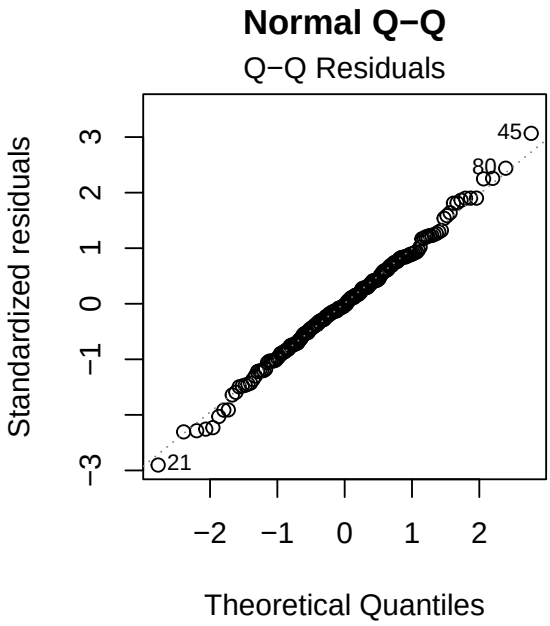
We Implement the forward selection based on the AIC obtained by the scores. The AIC approach allows a model comparison with the intention to find a good balance between the model fit and the complexity. The process of forward selection stars with no predictors and adding one by one. In thes case we have a no

significative predictor the Industrial Inputs Index, this happens because with the AIC criterion is possible to obtain non-significant predictors if their inclusion improves the general performance of the model.

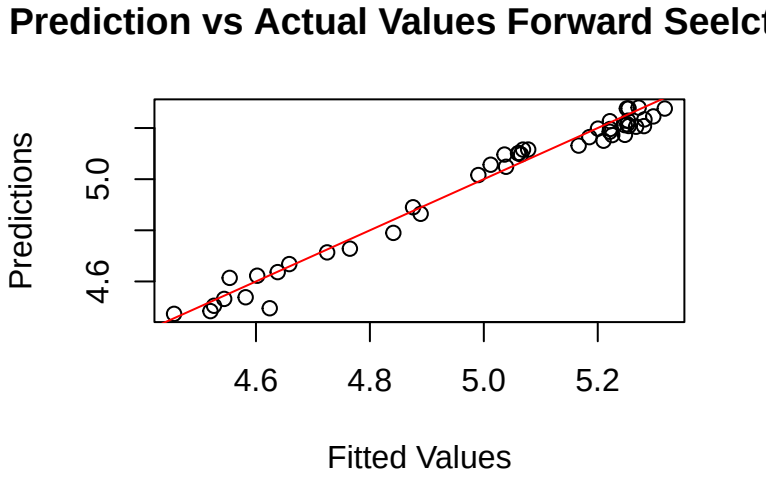
```r
# Forward AIC Stepwise selection--------------

# Fit the null model (intercept only)
null_model <- lm(House_Price_Index_log ~ 1, data = data_train)

# Specify the full model with all potential predictors
full_model <- lm(House_Price_Index_log ~ Industrial_Inputs_Index_log + Shipping_Price_Index_log + Forex_
                   Construction_Licences_Area_log + Google_Trends_Housing_log +
                   Unemployment_Rate + Interest_Rate, data = data_train)

# Perform forward selection based on AIC
forward_model <- step(null_model,
                      scope = list(lower = null_model, upper = full_model),
                      direction = "forward")
```

```
##
## Call:
## lm(formula = House_Price_Index_log ~ Google_Trends_Housing_log +
##     Industrial_Production_Index_log + Interest_Rate + Unemployment_Rate +
##     Forex_Index_log + Shipping_Price_Index_log + Construction_Licences_Area_log,
##     data = data_train)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.094159 -0.021217 -0.000685  0.021851  0.090717
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                      0.64293    0.16678   3.855 0.000164 ***
## Google_Trends_Housing_log        0.33191    0.01750  18.968  < 2e-16 ***
## Industrial_Production_Index_log  0.77431    0.04870  15.901  < 2e-16 ***
## Interest_Rate                   -0.46996    0.12859  -3.655 0.000342 ***
## Unemployment_Rate                0.61628    0.18912   3.259 0.001351 **
## Forex_Index_log                 -0.06931    0.01801  -3.849 0.000167 ***
## Shipping_Price_Index_log         0.09061    0.02805   3.231 0.001481 **
## Construction_Licences_Area_log  -0.02490    0.01038  -2.399 0.017496 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.03321 on 171 degrees of freedom
## Multiple R-squared:  0.9843, Adjusted R-squared:  0.9837
## F-statistic:  1535 on 7 and 171 DF,  p-value: < 2.2e-16
```






```
## [1] 0.001830836
```

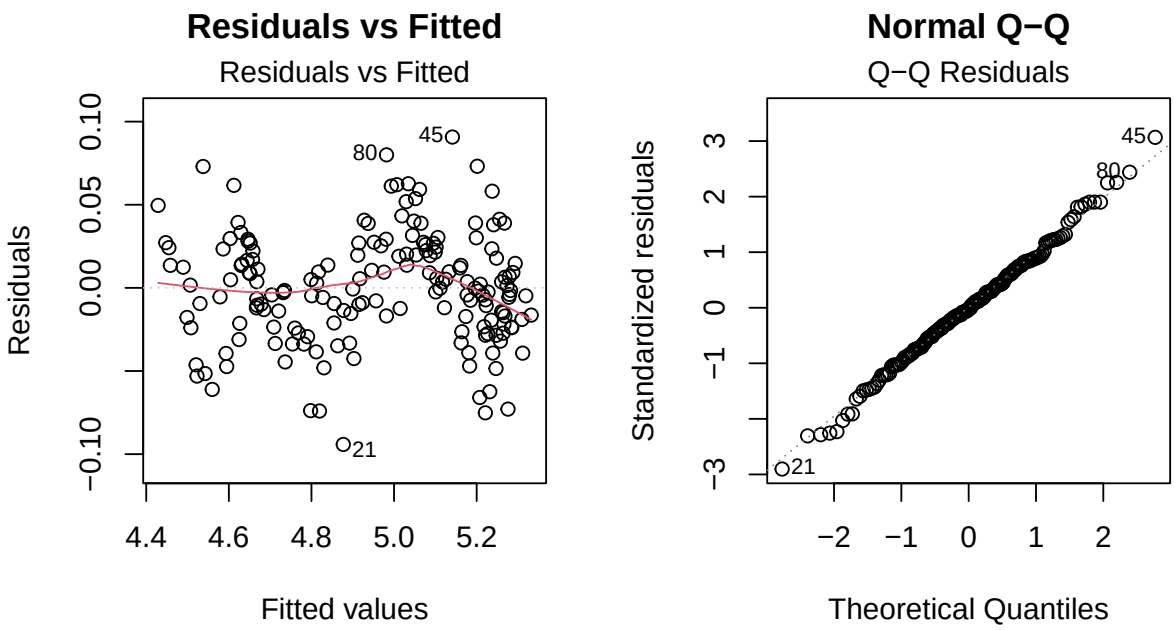


Backward AIC selection model

We implement the backward selection based also based in the AIC obtained by the different models. In this case, contrary to forward selection, all the variables left were in the reduced model obtained by leaving only the most significative variables. so for this case the results are the same obtained before.

```r
# Backward AIC Stepwise selection-------------

full_model <- lm(House_Price_Index_log ~ Industrial_Inputs_Index_log + Shipping_Price_Index_log + Forex_
                   Construction_Licences_Area_log + Google_Trends_Housing_log +
                   Unemployment_Rate + Interest_Rate, data = data_train)

# Perform backward selection based on AIC
backward_model <- step(full_model,
                       direction = "backward")
```

```
##
## Call:
## lm(formula = House_Price_Index_log ~ Shipping_Price_Index_log +
##     Forex_Index_log + Industrial_Production_Index_log + Construction_Licences_Area_log +
##     Google_Trends_Housing_log + Unemployment_Rate + Interest_Rate,
##     data = data_train)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.094159 -0.021217 -0.000685  0.021851  0.090717
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                      0.64293    0.16678   3.855 0.000164 ***
## Shipping_Price_Index_log         0.09061    0.02805   3.231 0.001481 **
## Forex_Index_log                 -0.06931    0.01801  -3.849 0.000167 ***
## Industrial_Production_Index_log  0.77431    0.04870  15.901  < 2e-16 ***
## Construction_Licences_Area_log  -0.02490    0.01038  -2.399 0.017496 *
## Google_Trends_Housing_log        0.33191    0.01750  18.968  < 2e-16 ***
## Unemployment_Rate                0.61628    0.18912   3.259 0.001351 **
## Interest_Rate                   -0.46996    0.12859  -3.655 0.000342 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.03321 on 171 degrees of freedom
## Multiple R-squared:  0.9843, Adjusted R-squared:  0.9837
## F-statistic:  1535 on 7 and 171 DF,  p-value: < 2.2e-16
```



Lasso regresion model

As another approach we decide to implement the lasso regression model. This with the intention of get a model that shrink the coefficients of the model helping to reduce the overfitting, and with the intention of see if some coefficients of the explanatory variables are setting to zero.For this, we use a cross validation in order to train the model

```
library(glmnet)
```

```
## Loading required package: Matrix
```

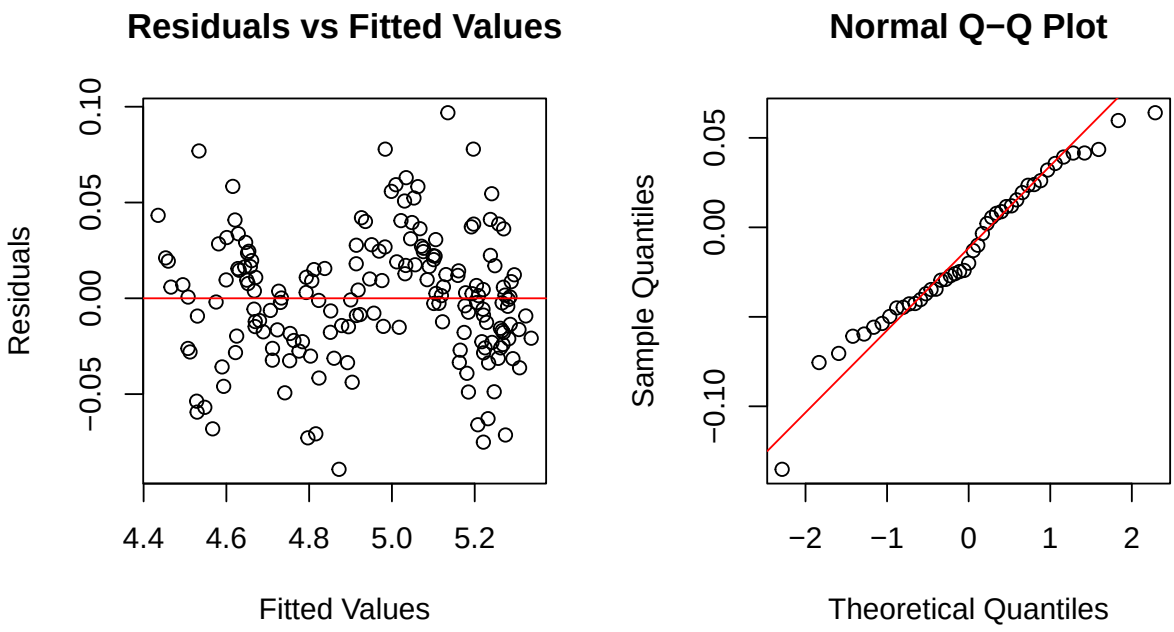
```
## Loaded glmnet 4.1-8
```

```
# Fit LASSO regression model
lasso_model <- cv.glmnet(X_train, y_train, alpha = 1)  # alpha = 1 for LASSO
lasso_best_lambda <- lasso_model$lambda.min
lasso_predictions <- predict(lasso_model, s = lasso_best_lambda, newx = X_test)
actuals <- y_test
```

We get the lasso coefficients, it is possible to see that no one was corrected to zero, but some of them get a way smaller coefficient, indicating a lower influence over the House Price Index, as for example the Finished constructions and the Energy Price Index.

```
# Coefficients at best lambda
lasso_coefficients <- coef(lasso_model, s = lasso_best_lambda)
print(lasso_coefficients)
```

```
## 11 x 1 sparse Matrix of class "dgCMatrix"
##                                         s1
## (Intercept)                      0.741327314
## Industrial_Inputs_Index_log     -0.022369516
## Energy_Price_Index_log           0.018196124
## Shipping_Price_Index_log         0.065335787
## Forex_Index_log                 -0.059385694
## Industrial_Production_Index_log  0.787242512
## Construction_Licences_Area_log  -0.022814422
## Finished_Constructions_log      -0.008274111
## Google_Trends_Housing_log        0.332610292
## Unemployment_Rate                0.674629707
## Interest_Rate                   -0.486292194
```



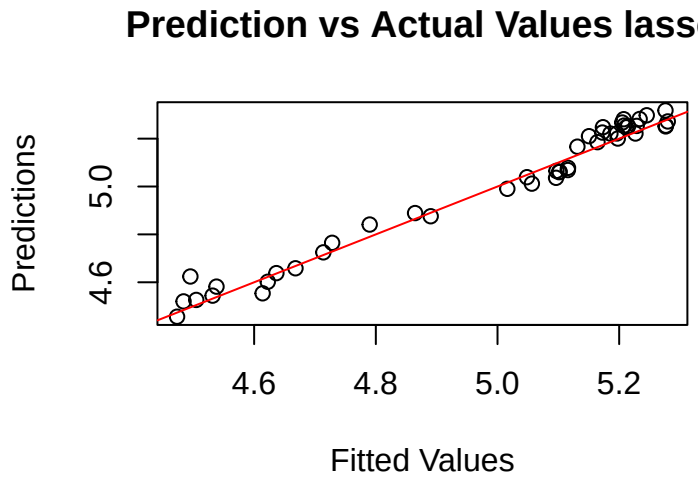

```r
# Calculate the Mean Squared Error (MSE)
mse_lasso <- mean((lasso_predictions - y_test)^2)
print(paste("Mean Squared Error:", mse_lasso))
```

```
## [1] "Mean Squared Error: 0.00183252365342529"
```

```r
# Calculate R-squared (R2)
y_mean <- mean(y_test)
ss_total <- sum((y_test - y_mean)^2)
ss_residual <- sum((y_test - lasso_predictions)^2)
r2 <- 1 - (ss_residual / ss_total)
print(paste("R-squared:", r2))
```

```
## [1] "R-squared: 0.974974383706247"
```

```
{
plot(actuals,predictions,main = "Prediction vs Actual Values lasso",
     xlab = "Fitted Values", ylab = "Predictions")
abline(0, 1, col = "red")}
```



Ridge regression model

We apply a similar procedure for the ridge regression to see if we get a different model with respecto to lasso, as it uses a different regularization term. We rpesent the results.

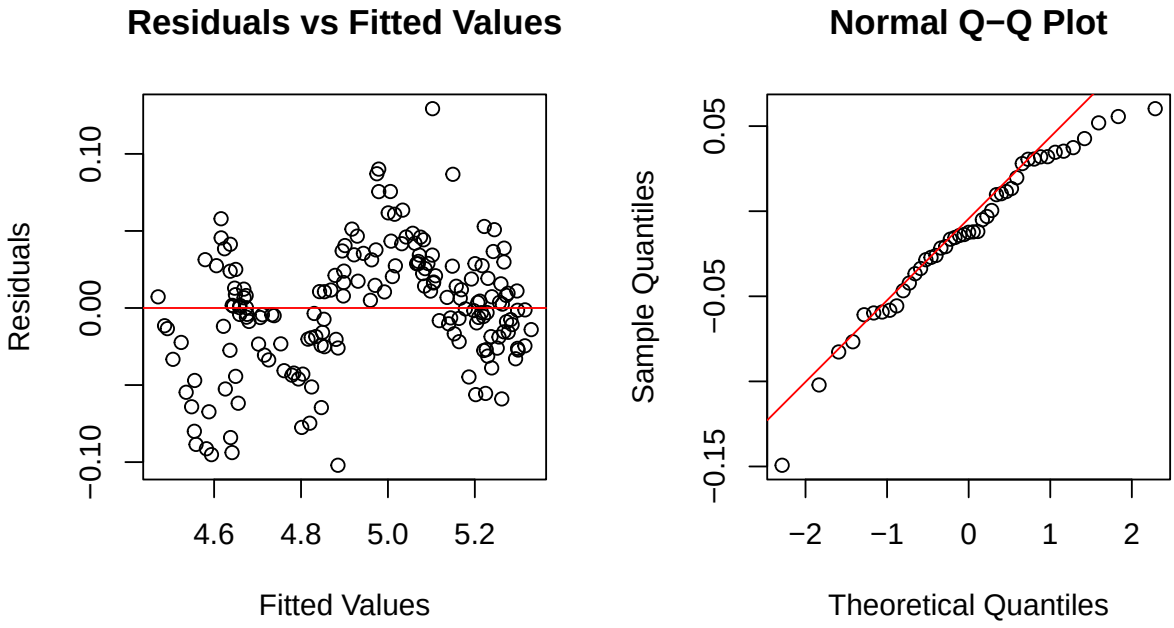
```
# Regularization------------
# Done over full model
# Load the packages
library(glmnet)



# Fit RIDGE regression model
ridge_model <- cv.glmnet(X_train, y_train, alpha = 0)  # alpha = 1 for LASSO
ridge_best_lambda <- ridge_model$lambda.min
ridge_predictions <- predict(ridge_model, s = ridge_best_lambda, newx = X_test)
actuals <- y_test
```

```
# Coefficients at best lambda
ridge_coefficients <- coef(ridge_model, s = ridge_best_lambda)
print(ridge_coefficients)
```

```
## 11 x 1 sparse Matrix of class "dgCMatrix"
##                                           s1
## (Intercept)                     -0.144744114
## Industrial_Inputs_Index_log      0.034484853
## Energy_Price_Index_log          -0.012393077
## Shipping_Price_Index_log         0.140560512
## Forex_Index_log                  0.039961309
## Industrial_Production_Index_log  0.553919098
## Construction_Licences_Area_log  -0.002977282
```

```
## Finished_Constructions_log       0.031146341
## Google_Trends_Housing_log        0.304693425
## Unemployment_Rate               -0.026640711
## Interest_Rate                   -0.923398135
```

In general we get a different results for the coefficients with ridge. It is possible to see that some of them change drastically as for example the one for the Unemployment Rate that with lasso have a postie high value, but with ridge we obtain a negative small one.



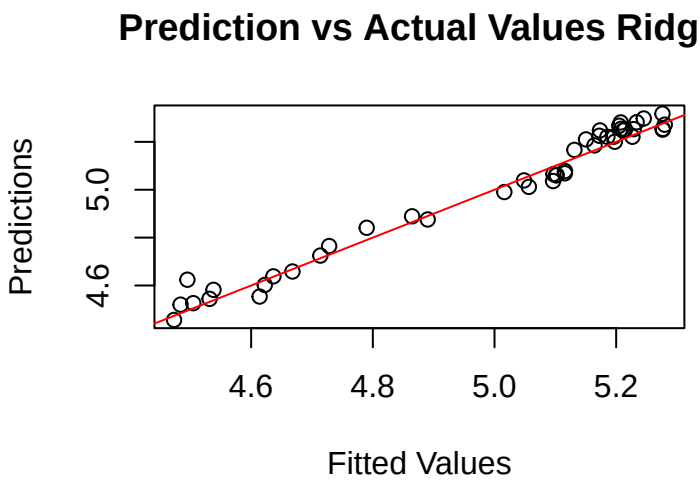

```r
# Calculate the Mean Squared Error (MSE)
mse_ridge <- mean((ridge_predictions - y_test)^2)
print(paste("Mean Squared Error:", mse_ridge))
```

```
## [1] "Mean Squared Error: 0.00209622139939288"
```

```r
# Calculate R-squared (R2)
y_mean <- mean(y_test)
ss_total <- sum((y_test - y_mean)^2)
ss_residual <- sum((y_test - ridge_predictions)^2)
r2 <- 1 - (ss_residual / ss_total)
print(paste("R-squared:", r2))
```

```
## [1] "R-squared: 0.971373230402836"
```

```r
{
plot(actuals,predictions,main = "Prediction vs Actual Values Ridge",
     xlab = "Fitted Values", ylab = "Predictions")
abline(0, 1, col = "red")}
```



#MSE Model comparison

Variable	Value
mse_full	1.867e-3
mse_red	1.830e-3
mse_lag	0.0231e-3
mse_forward	1.830e-3
mse_backward	1.830e-3
mse_lasso	1.832e-3
mse_ridge	2.096e-3

It is possible to see that the best model in terms of MSE is the one using the lag variable. With respect to the others, the majority of them give a similar value for the coefficients of the explanatory variables. In general the reduce model obtainind only with the significative variables, and the ones implemented selection based on the AIC perform in a similar way. And the lasso and ridge get a similar result with ridge getting a sligth gigher MSE.