Analyzing the Dynamics of the U.S. Housing Prices

Rohan Krishnan, Yewon Kim, Dhwani Pareshkumar Kanani

2023-12-11

# Introduction

The current U.S. housing market is valued at 47 trillion USD (Rosen, 2023) , making it a pivotal factor in gauging the overall welfare of the nation’s economy. Homeownership is a widespread aspiration, contributing to the significance of this market. Investors and economists rely on the Housing Price Index (HPI) as one of the most crucial metrics for assessing investment attraction as well as determining the economic conditions. The HPI, is a comprehensive macroeconomic measure that carefully monitors the price fluctuations of single-family homes nationwide. In addition, it serves as an analytical tool for approximating the changes in the rates of mortgage defaults, prepayments, and housing affordability (Liberto, 2023). The Federal Housing Finance Agency compiles this data  by reviewing single-family housing mortgages purchased or securitized by Fannie Mac or Freddie Mac. For this project, our objective was to find an economic data set  that we could explore and analyze to better understand the U.S. economy. Our final dataset from kaggle talks about both the macroeconomics and microeconomic factors impacting house prices in the USA.

## Research Question(s)

The central focus of this research project revolves around the question of how the House Price Index (HPI)  is affected by several  macroeconomic factors. Conducting a project that revolves around the idea of the U.S. house price index can provide us with valuable insights into economic, social, and policy considerations. More precisely,  it can sculpt our knowledge on various aspects such as economic significance meaning that housing is a critical component of the economy. Understanding the factors influencing the HPI can shed light on economic trends, market dynamics, and overall marker of real estate. In addition, you can enrich your understanding about macroeconomic stability. That is to say, housing is often connected to broader economic stability and fluctuations in the HPI can have ripple effects on consumer spending, employment, and overall economic performance. It can help in analyzing consumer behavior where it can aid individuals and families make more informed decisions about homeownership, renting, or other house-related choices. In summary, a project aiming at factors affecting the House Price Index (HPI) is relevant and impactful because it addresses economic, social, and policy dimensions with implications for individuals, businesses, policymakers, and researchers. The final result can have practical applications and contribute to a better understanding of the complex dynamics within the real estate market. The main research questions we are trying analyze are:

1. What variables are most useful in predicting hpi?

* a. Which machine learning model best predicts hpi given new data?

## Statement of Purpose

The ultimate goal from this project was to broaden our knowledge about the economy, mortgage and finance, consumer behavior, and investment sectors associated within the U.S. housing market. Lastly, we wanted to apply our skills and understanding from the class to a real world data set.

# Methodology

## Data Collection

We found a “US Housing Market Factors” data set on Kaggle. Kaggle is a platform where one can find numerous types of datasets and competitions related to data science and machine learning. It uses various means of data collection methods and they are often compiled and shared by individuals, organizations, or communities for specific purposes. The data in our data set was collected from FRED (Federal Reserve Economic Data), an online database that offers a wide range of economic time series data.

#--- Load necessary libraries ---#  
library(dplyr)  
library(tidyverse)  
library(ggplot2)  
library(ggcorrplot)  
library(ggtext)  
library(gt)  
library(skimr)  
library(flextable)  
library(grDevices)

## Data Set(s)

The “U.S. Housing Market Factors” data set contained three different files labeled as “Annual\_Macroeconomic\_Factors”, “Housing\_Macroeconomic\_Factors”, and “Monthly\_Macroeconomic\_Factors”. After exploring all the files, we identified that the “Annual\_Macroeconomic\_Factors” data set was simply a summary of the “Monthly\_Macroeconomic\_Factors” data set.

In “Monthly\_Macroeconomic\_Factors” data set, there were eight macroeconomic factors including hpi, House Price Index. The seven other variables were Stock Price Index, Consumer Price Index, Population, Unemployment Rate, Real GDP, Mortgage Rate, and Real Disposable Income. It had 423 observations, from 1987 January to 2022 March. On the other hand, in “Housing\_Macroeconomic\_Factors” data set, there were twelve factors including hpi. Those variables were population, house supply, gdp, mortgage rate, employment rate, permit\_new, ppi\_res, m3, cci, delinquency rate, and hcai. Further description about each variable can be found below. This data set had 425 observations, from 1987 January to 2022 May.

Our working data set was created by left joining the “Monthly\_Macroeconomic\_Factors” and “Housing\_Macroeconomic\_Factors” data sets by their “Date” variables. We elected to not use the “Annual\_Macroeconomic\_Factors” in order to maximize the number of observations we could work with within our sample. We named the combined data set “data”.

#--- First steps ---#  
  
#Load in data  
housing <- read.csv("/Users/rohankrishnan/Downloads/archive (6)/Housing\_Macroeconomic\_Factors\_US (2).csv")  
macro <- read.csv("/Users/rohankrishnan/Downloads/archive (6)/Monthly\_Macroeconomic\_Factors.csv")  
  
#Join macro to housing by date  
data <- housing %>% left\_join(macro, by = "Date")  
  
#Preliminary examination of data  
glimpse(data)

After creating the initial combined data, we removed several duplicate, incorrect, and unnecessary columns (specified in the R code below). We then renamed the remaining columns for ease of syntax and converted all columns into the numeric class.

#--- Basic Data Cleaning ---#  
  
#Remove Housing\_Price\_Index (there are 2), Date (unnecessary for classification), Mortgage Rate (duplicate), and Population (wrong values)  
data0.5 <- data %>%  
 dplyr::select(-House\_Price\_Index, -Date, -Mortgage\_Rate, -Population)  
  
#Rename longer column names  
data0.5 <- data0.5 %>%   
 rename("hpi" = "house\_price\_index",  
 "hs" = "house\_supply",  
 "GDP"= "gdp",  
 "mort\_rt" = "mortgage\_rate" ,  
 "employ\_rt" = "employment\_rate",  
 "del\_rt" = "delinquency\_rate",  
 "spi" = "Stock\_Price\_Index",  
 "cpi" = "Consumer\_Price\_Index",  
 "unemploy\_rt" = "Unemployment\_Rate",  
 "rdi" = "Real\_Disposable\_Income")  
  
#Convert population to numeric variable  
data0.5$population <- data0.5$population %>% as.numeric()

*Table 1* below is the structure of our data after the above steps:

#View structure of data -- Internal view only no need to display in report  
data0.5 %>%  
 skimr:: skim() %>%  
 dplyr::select(skim\_type, skim\_variable, n\_missing, numeric.hist) %>%  
 flextable::flextable()%>%  
 set\_caption(  
 caption = as\_paragraph(  
 as\_chunk("Table 1: Structure of Intermediate Data",  
 props = fp\_text\_default(font.family = "Times New Roman",  
 font.size = 12,  
 italic = TRUE)))) %>%  
 set\_header\_labels(values = list(  
 skim\_type = "Type",  
 skim\_variable = "Name",  
 n\_missing = "NA values",  
 numeric.hist = "Histogram")) %>%  
 set\_table\_properties(layout = "autofit")%>%  
 bg(i = seq(2,16,2), bg = "light grey")%>%  
 font(part = "all", fontname = "Times New Roman") %>%  
 fontsize(size = 12) %>%  
 border(i = c(1:16), border.top = fp\_border\_default(color = "black"))

Table 1: Structure of Intermediate Data

| Type | Name | NA values | Histogram |
| --- | --- | --- | --- |
| numeric | hpi | 0 | ▇▅▅▁▁ |
| numeric | population | 0 | ▅▅▅▆▇ |
| numeric | hs | 0 | ▇▇▃▁▁ |
| numeric | GDP | 3 | ▁▁▁▅▇ |
| numeric | mort\_rt | 3 | ▇▆▇▅▃ |
| numeric | employ\_rt | 3 | ▁▁▅▇▇ |
| numeric | permit\_new | 3 | ▃▅▇▅▂ |
| numeric | ppi\_res | 3 | ▇▅▆▂▁ |
| numeric | m3 | 3 | ▇▃▂▁▁ |
| numeric | cci | 3 | ▂▃▃▇▂ |
| numeric | del\_rt | 51 | ▇▁▁▁▂ |
| numeric | hcai | 140 | ▇▁▁▃▂ |
| numeric | spi | 2 | ▇▇▅▃▁ |
| numeric | cpi | 2 | ▆▇▆▇▃ |
| numeric | unemploy\_rt | 2 | ▇▃▂▁▁ |
| numeric | Real\_GDP | 2 | ▁▁▂▇▁ |
| numeric | rdi | 2 | ▇▆▇▃▁ |

After converting all the columns to a numeric class and examining the structure of the data, we noticed that there were several variables with missing (NA) values due to a mismatch in dates. In particular, “Del\_rt” (Delinquency rate) and “hcai” (Housing credit availability index) had a significant number of NA values, 140 and 51 respectively. We elected to remove “hcai” from our data entirely, as it was nearly 40% NA values. We kept “del\_rt” because we suspected it may be significantly correlated with hpi and it had an allowable amount of NA values. Finally, we filtered out all of the remaining NA values in our data set and named our final cleaned data “data2”.

#Create new dataframe without hcai (too many NAs)  
data1 <- data0.5 %>% dplyr::select(-hcai)  
  
#Summarize data\_1 values (mean, quantiles, etc.)  
summary(data1)  
  
#Remove last three rows (NAs)  
data1 <- data1 %>%  
 filter(is.na(GDP) == FALSE)  
data1 %>% summary()  
  
#Create new dataframe with all NAs removed  
data2 <- data1 %>%   
 na.omit()

## Variables

Since the main research question is trying to uncover what are the most influential variables in predicting the Housing Price Index (HPI), the hpi variable will be our response (target) variable. Our explanatory variables are the remaining 15 variables: “population”, “hs”, “GDP”, “mort\_rt”, “employ\_rt”, “permit\_new”, “ppi\_res”, “m3”, “cci”, “del\_rt”, “spi”, “cpi”, “unemploy\_rt”, “Real\_GDP”, and “rdi”. *Table 2* below is a data dictionary with the meaning and class of each of our variables:

#Create data dictionary table for data2:  
  
#Create variable name list  
var\_name <- names(data2)  
  
#Create variable class list  
var\_type <- vector()  
for (i in 1:ncol(data2)){  
 var\_type[i] = class(data2[,i])  
}  
  
#Create variable meaning list  
var\_meaning <- c("Housing Price Index at the time of observation",  
 "Population of US at the time of observation",  
 "Housing supply at the time of observation",  
 "Nominal GDP of US at the time of observation",  
 "US mortgage rates at the time of observation",  
 "US employment rate at the time of observation",  
 "Total number of new privately owned housing units authorized in permit-issuing places at the time of observation",   
 "US producer price index at the time of observation",  
 "Total US money supply at the time of observation",  
 "US consumer confidence index at the time of observation",  
 "US deliquency rate at the time of observation",  
 "US stock price index at the time of observation",  
 "US consumer price index at the time of observation",  
 "US unemployment rate at the time of observation",  
 "Real GDP of US at the time of observation",  
 "US real disposable income at the time of observation"  
)  
  
#Create data frame of variable name and class  
data\_var <- data.frame(cbind(var\_name, var\_type, var\_meaning))  
data\_var <- data\_var %>%  
 rename("Name" = "var\_name",  
 "Class" = "var\_type",  
 "Meaning" = "var\_meaning")  
  
#Map data frame to flextable for display  
data\_var %>%  
 flextable()%>%  
 set\_caption(  
 caption = as\_paragraph(  
 as\_chunk("Table 2: Data Dictionary of Cleaned Data",  
 props = fp\_text\_default(font.family = "Times New Roman",  
 font.size = 12,  
 italic = TRUE)  
 )  
 )  
 ) %>%  
 set\_table\_properties(layout = "autofit") %>%  
 bg(i = seq(2,16,2), bg = "light grey") %>%  
 font(part = "all", fontname = "Times New Roman") %>%  
 fontsize(size = 12) %>%  
 border(i = c(1:16), border.top = fp\_border\_default(color = "black"))

Table 2: Data Dictionary of Cleaned Data

| Name | Class | Meaning |
| --- | --- | --- |
| hpi | numeric | Housing Price Index at the time of observation |
| population | numeric | Population of US at the time of observation |
| hs | numeric | Housing supply at the time of observation |
| GDP | numeric | Nominal GDP of US at the time of observation |
| mort\_rt | numeric | US mortgage rates at the time of observation |
| employ\_rt | numeric | US employment rate at the time of observation |
| permit\_new | numeric | Total number of new privately owned housing units authorized in permit-issuing places at the time of observation |
| ppi\_res | numeric | US producer price index at the time of observation |
| m3 | numeric | Total US money supply at the time of observation |
| cci | numeric | US consumer confidence index at the time of observation |
| del\_rt | numeric | US deliquency rate at the time of observation |
| spi | numeric | US stock price index at the time of observation |
| cpi | numeric | US consumer price index at the time of observation |
| unemploy\_rt | numeric | US unemployment rate at the time of observation |
| Real\_GDP | numeric | Real GDP of US at the time of observation |
| rdi | numeric | US real disposable income at the time of observation |

We also generated summary statistics for each variable in our cleaned data set.

## Modeling and Analysis Plan

In order to better understand our research question, we will first explore the features and distributions of the data. We will then employ several supervised and unsupervised learning methods to gain a better understanding of the relationships between each explanatory variable and hpi. This section will outline the specific machine learning methods we chose to use as well as the overall roadmap we intend to take when analyzing our data.

## *Description of Analysis*

Specifically, we will use a multiple linear regression (MLR), bootstrapped regression, regression tree (un-tuned and tuned) and a random forest (un-tuned and tuned) to understand which variables are most important in predicting hpi and to evaluate whether or not we can effectively predict hpi given out-of-sample values for our explanatory variables (in the case of MLR, regression tree, and random forest).

## *Analysis Plan*

We will follow the standard procedure of exploratory data analysis followed by iterative modelling to conduct our project. First, we will explore the descriptive statistics of each variable. We will then use data visualization to better understand the behavior of each variable. We will begin with individual density plots to assess each variables distribution and skew. We will then use box plots to better understand any imbalances and to visually identify any outliers. Finally, we will generate a correlation matrix to quantify the linear correlation of each of our explanatory variables with hpi. After finishing our exploratory analysis, we will begin to generate our models (see above). For each model, we will examine important explanatory variables and (when applicable) calculate the test mean absolute error (MAE), mean standard error (MSE), and root mean standard error (RMSE) to quantify each model’s predictive ability.

# Results

## Exploratory Data Analysis

This section will discuss our exploratory analysis of the data set.

## *Descriptive Statistics*

The first step in our exploratory analysis was to calculate descriptive statistics for all of our variables. *Table 3* below lists the main summary statistics for each of our variables:

#--- Final Data Descriptions ---#  
  
#Create summary statistics table for data2:  
  
#Load library  
library(psych)  
  
#Create a data frame of summary statistics  
summary\_stats <- describe(data2, fast = TRUE)  
summary\_stats$vars <- rownames(summary\_stats)  
  
#Map data frame to flextable for display  
summary\_stats %>%  
 flextable() %>%  
 set\_caption(  
 caption = as\_paragraph(  
 as\_chunk("Table 3: Summary Statistics of Cleaned Data",  
 props = fp\_text\_default(font.family = "Times New Roman",  
 font.size = 12,  
 italic = TRUE)))) %>%  
 set\_header\_labels(values = list(  
 vars = "Name",  
 n = "N",  
 mean = "Mean",  
 sd = "Std.Dev.",  
 min = "Min",  
 max = "Max",  
 range = "Range",  
 se = "Std. Error")) %>%  
 set\_table\_properties(layout = "autofit") %>%  
 bg(i = seq(2,16,2), bg = "light grey") %>%  
 font(part = "all", fontname = "Times New Roman") %>%  
 fontsize(size = 12) %>%  
 border(i = c(1:16), border.top = fp\_border\_default(color = "black"))

Table 3: Summary Statistics of Cleaned Data

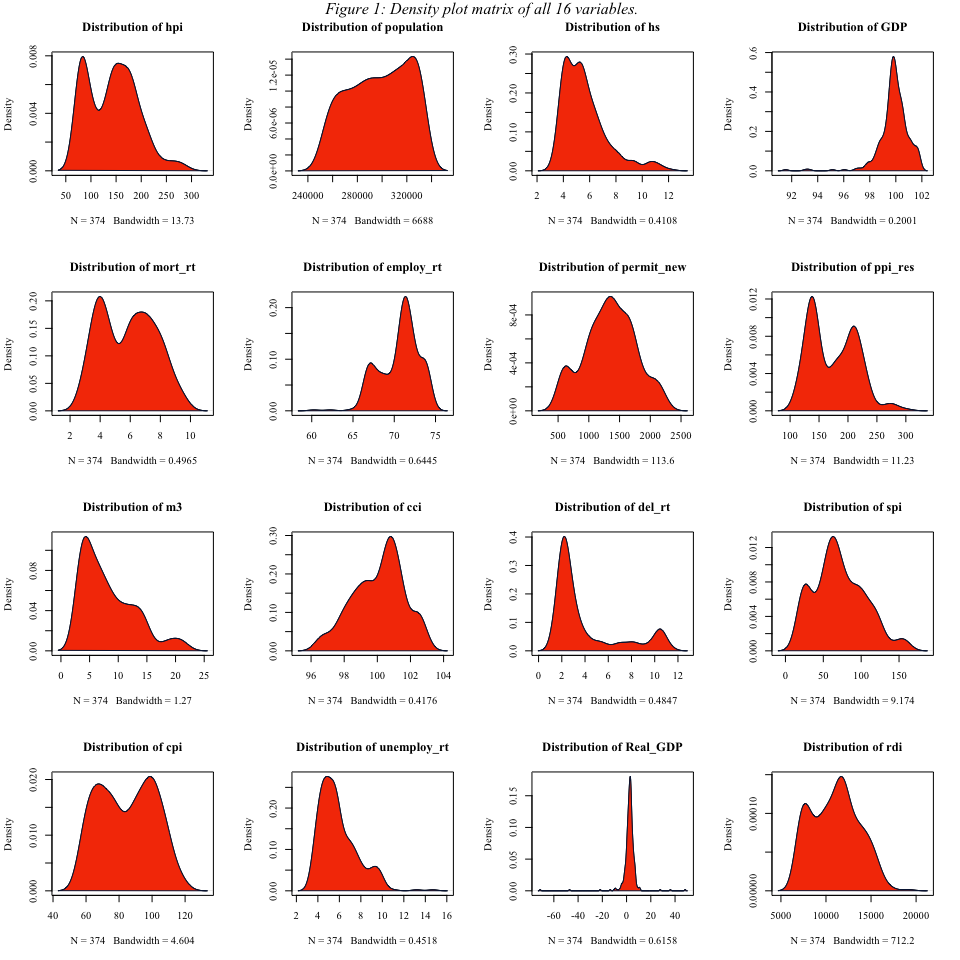
| Name | N | Mean | Std.Dev. | Min | Max | Range | Std. Error |
| --- | --- | --- | --- | --- | --- | --- | --- |
| hpi | 374 | 142.703599 | 49.899690 | 75.56700 | 290.37100 | 214.80400 | 2.58025152 |
| population | 374 | 297,515.339572 | 24,302.997309 | 252,012.00000 | 332,750.00000 | 80,738.00000 | 1,256.67807387 |
| hs | 374 | 5.634492 | 1.703100 | 3.30000 | 12.20000 | 8.90000 | 0.08806521 |
| GDP | 374 | 99.859629 | 1.141962 | 91.54306 | 101.82585 | 10.28279 | 0.05904946 |
| mort\_rt | 374 | 5.825602 | 1.804159 | 2.68400 | 9.63750 | 6.95350 | 0.09329086 |
| employ\_rt | 374 | 70.629780 | 2.341921 | 60.26101 | 74.50744 | 14.24642 | 0.12109784 |
| permit\_new | 374 | 1,344.409091 | 412.816821 | 513.00000 | 2,263.00000 | 1,750.00000 | 21.34624963 |
| ppi\_res | 374 | 174.428275 | 40.818932 | 112.70000 | 303.54600 | 190.84600 | 2.11069671 |
| m3 | 374 | 8.265692 | 4.615371 | 3.28770 | 21.74970 | 18.46200 | 0.23865517 |
| cci | 374 | 100.087868 | 1.517205 | 96.46033 | 102.97760 | 6.51727 | 0.07845279 |
| del\_rt | 374 | 3.984947 | 2.912758 | 1.40000 | 11.36000 | 9.96000 | 0.15061511 |
| spi | 374 | 71.398599 | 33.333211 | 17.62213 | 159.94283 | 142.32070 | 1.72361933 |
| cpi | 374 | 84.643172 | 16.730541 | 56.78918 | 119.70281 | 62.91363 | 0.86511569 |
| unemploy\_rt | 374 | 5.895187 | 1.743676 | 3.50000 | 14.70000 | 11.20000 | 0.09016334 |
| Real\_GDP | 374 | 2.411364 | 6.465178 | -71.28495 | 48.35587 | 119.64082 | 0.33430637 |
| rdi | 374 | 11,024.392781 | 2,587.684634 | 6,804.70000 | 19,119.50000 | 12,314.80000 | 133.80598698 |

Our variables all have very different scales, meaning we will need examine them each on their own plot to look at their distributions. As the table shows, our final data set contains 374 observations with 16 variables, each of which represents a specific indicator of the US economy.

## *Data Visualization*

After calculating descriptive statistics for our variables, we used density plots, box plots, and a correlation matrix to visualize our data. We started by plotting a density plot matrix to examine the distribution of each variable individually. *Figure 1*, a density plot matrix of all 16 variables is shown below:

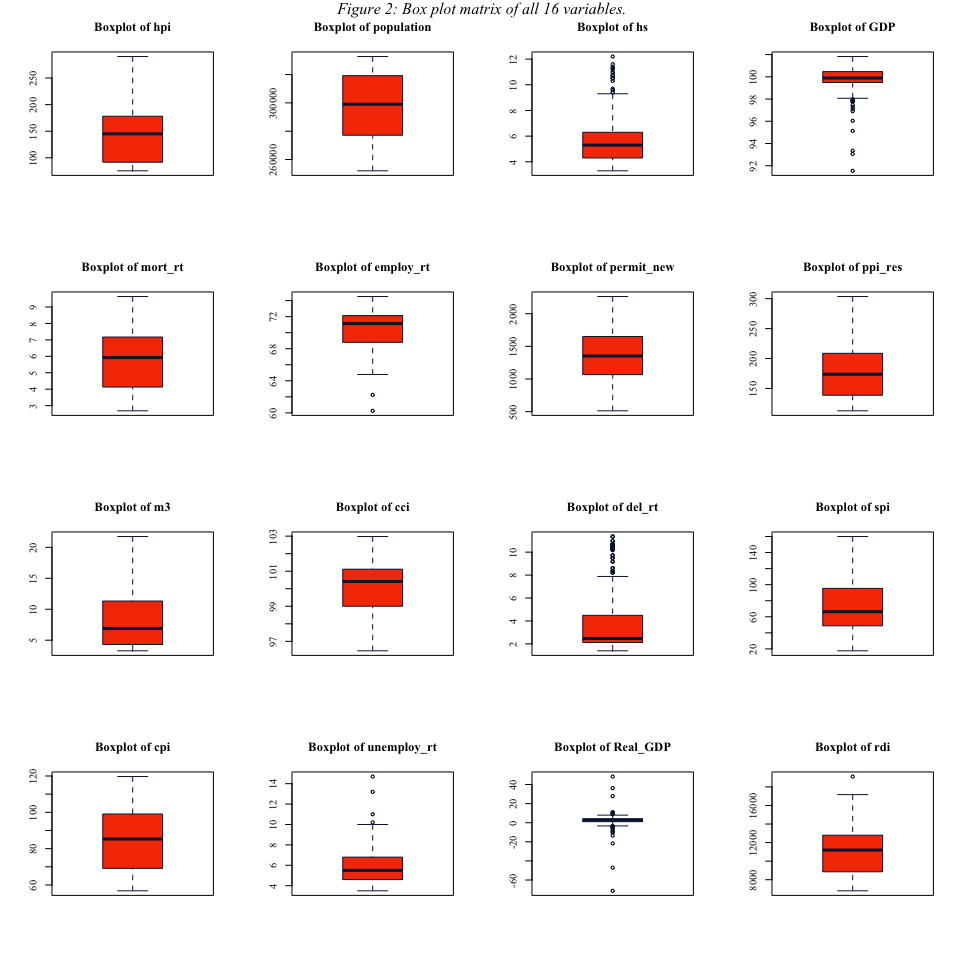
#--- Data visualization ---#  
  
#Create density plot matrix  
par(mfrow = c(4,4))  
for (i in 1:ncol(data2)){  
 colname = names(data2)  
 plot(density(data2[,i]), main = paste("Distribution of", colname[i]), family = "Times New Roman")  
 polygon(density(data2[,i]), col = "#F63E02", border = "#061A40")  
}  
mtext(~italic("Figure 1: Density plot matrix of all 16 variables."), side = 3, line = -1.5, outer = TRUE, family = "Times New Roman")



As is made clear by the matrix, none of our variables follow a perfect normal distribution. The housing price index (hpi) appears to be bimodal with a right skew. Population, unsurprisingly, appears to be relatively uniform with a slight left skew. Despite the fact that the US population continues to grow, the population values are relatively close together month-to-month. Housing supply (HS) is heavily right-skewed, indicating that we could potentially force some normality by removing outliers. On the other hand, Nominal GDP (GDP) is heavily left skewed. This is likely to be caused by the 2008 recession and could be removed when cleaning outliers. Mortgage rates (mort\_rt), like the housing price index, seem to be bimodal with a slight right skew. Employment rate (employ\_rt) was left skewed, meaning we can likely remove some outliers to get it to a normal distribution. The total number of new privately owned housing units authorized in permit-issuing places (permit\_new) is our first variable that appears to be approximately normal. The producer price index (ppi\_res) has a bimodal distribution with a slight right skew. This makes sense as there are likely two “common” values, one when the economy is doing better and one when it is doing worse. The total US money supply (m3) appears to be right skewed while the consumer confidence index (cci), which measures consumers’ overall optimism or pessimism regarding their financial situation, appears to be left skewed. The delinquency rate (del\_rt), which is the amount of debt that is past due in the economy, is heavily skewed to the right. This makes sense, as the delinquency rate should not take on extremely high values in a normal or healthy economy. The stock price index (spi) appears to have a slight right skew, though it is close to approximately normal. The consumer price index (cpi) appears to be bimodal. Again, this makes sense since consumer goods tend to occupy certain prices depending on the state of the economy. The unemployment rate (unemploy\_rt) is heavily right skewed, since there is usually not high unemployment in the normal economy. Real GDP (Real\_GDP) appears to have a very sharp spike around its mean and very narrow tails. This distribution appears to be an extreme version of a normal distribution. It makes sense that real GDP is less spread out than nominal GDP, as real GDP is adjusted for inflation making its values more similar across time. Finally, real disposable income (rdi), which is a household’s money left over after paying taxes, appears to be right skewed. This distribution makes sense since the majority of US citizens will not have a high amount of money left over after taxes, but a small minority (millionaires and billionaires) will have very large amounts of money left over.

After looking at the density plots, it is clear that many of our variables are skewed. In order to get a better understanding of if their skewness is an inherent characteristic of the data or if it is caused by data that may not fit within the broader pattern, we can look at boxplots and identify outliers. Below is *Figure 2*, a matrix of box plots for all 16 variables in data2:

#--- Data Visualization ---#  
  
#Create matrix of boxplots for all variables  
par(mfrow = c(4,4))  
for (i in 1:ncol(data2)){  
 par(family = "Times New Roman")  
 colname = names(data2)  
 boxplot(data2[,i], main = paste("Boxplot of", colname[i]),col = "#F63E02", border = "#061A40")  
}  
mtext(~italic("Figure 2: Box plot matrix of all 16 variables."), side = 3, line = -1.5, outer = TRUE, family = "Times New Roman")



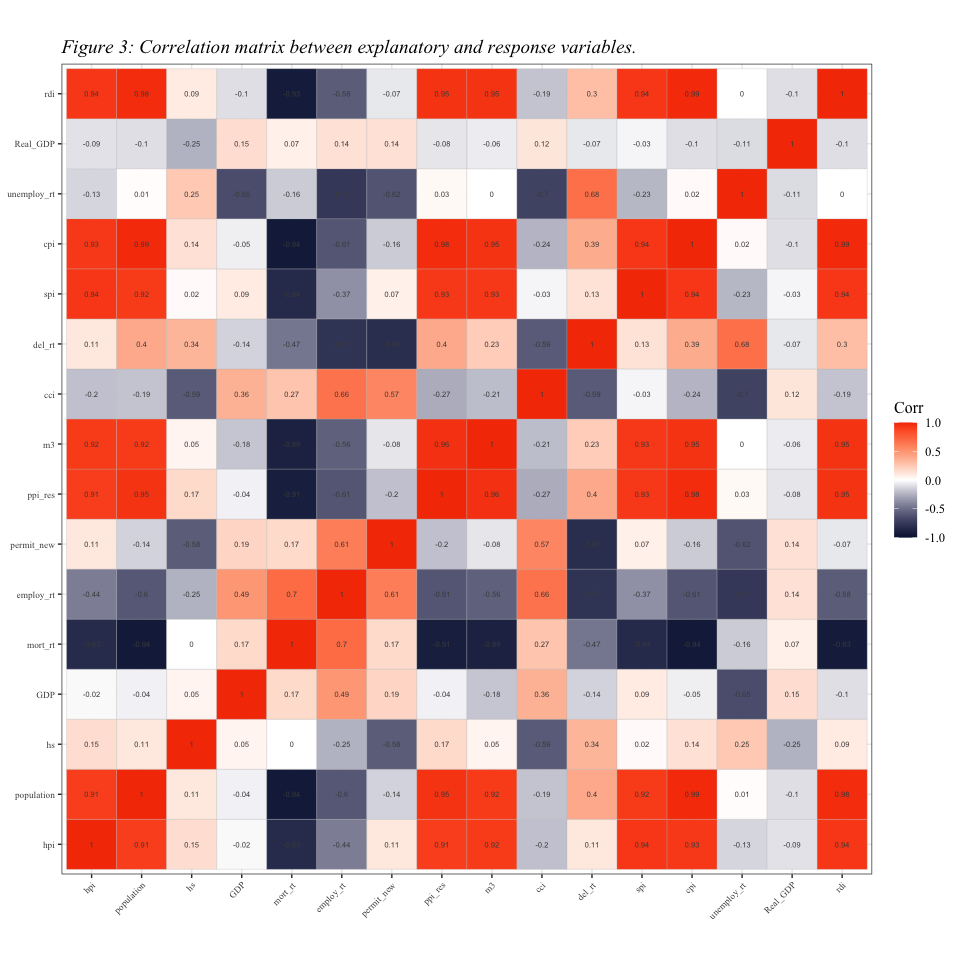
In order to get a better understanding of the range and outliers of each variable, we examined each of their box plots. Each box plot illustrates the range of data from Q1 to Q3. The lines at the top and bottom of each plot indicate the maximum and minimum and there are points outside of those bounds representing any outliers. Housing supply (hs) had a large amount of upper outliers while nominal GDP (GDP) had a large amount of lower outliers. This matches with their density distributions which were heavily right and left skewed, respectively. By removing these outliers before conducting analyses, we could create approximately normal data to satisfy any necessary normality requirements. Other variables with large amounts of outliers include the delinquency rate (del\_rt), which had a large amount of upper outliers, and the real GDP (Real\_GDP), which had an extremely small box and many upper and lower outliers. Both of these box plots match their respective density distributions of right skewed and thin unimodal. There were some variables with only a small amount of outliers, which corresponded to the variables with average or weak skews. Those variables include the employment rate (employ\_rt), which had two lower outliers; the unemployment rate (unemploy\_rt), which had four upper outliers; and real disposable income (rdi), which only had one upper outlier.

As discussed above, a potential solution to non-normal distributions would be to remove outliers. We could do this by either filtering out observations that fall outside of three standard deviations from the mean or that fall outside of 1.5 times the IQR away from the first and third quartiles respectively. Before modeling, we would try both methods, re-plot the box plots and density functions, and decide which method best removes outliers without sacrificing too many observations. After better understanding the distributions and outliers of our variables, we looked at each variable’s correlation with our response to understand what variables could be potentially significant.

## *Relationships*

The next part of our variable analyses was to create a correlation matrix to better understand the relationship between our 15 explanatory variables and hpi. Below is *Figure 3*, the full correlation matrix, created with ggcorrplot():

#--- Data Visualization ---#  
  
#Create correlation vector with correlations between all variables  
corr = data.frame(cor(data2))  
  
#Create correlation matrix of all variables in data2  
ggcorrplot(corr, ggtheme = theme\_bw(), show.diag = TRUE, tl.cex = 7, colors = c("#061A40", "white","#F63E02"), lab = TRUE, lab\_size = 2, lab\_col = "#4A4A4A") +  
 ggtitle("\*Figure 3: Correlation matrix between explanatory and response variables.\*") +  
 theme(plot.title = ggtext::element\_markdown(),  
 text = element\_text(family = "Times New Roman", size = 12))



Based on the correlation matrix plot above, hpi (Housing Price Index) has a strong positive correlation with population, ppi\_res (producer price Index), m3(total US money supply), spi (Stock Price Index), cpi(consumer price Index), and rdi(real disposable Income) variables. All of these variables have a positive correlation with values very close to 1. This signifies that as one of these factors increases or decreases, it will simultaneously increase or decrease the hpi variable. In contrast, hpi has a strong negative correlation with employ\_rt (employment rate), mort\_rt (mortgage rate), and cci (consumer confidence Index). The values are roughly close to -1, which implies that each of these variables would have an inverse relationship. Namely, as one increases, the other one decreases and vice versa. The darker the color, the stronger the correlation is between variables and the lighter the color, the weaker the relationship is between variables. For instance, hpi has the weakest negative correlation with factors known as Real\_GDP, GDP, and unemploy\_rt (unemployment rate). Likewise, it has the weakest positive correlation with permit\_new, hs(house\_supply), and del\_rt (delinquency rate). After viewing the correlation matrix, we filtered out the variables with the strongest correlation with hpi. We considered an absolute correlation of greater than 0.50 to be the threshold for strong correlation.

#--- Data Visualization ---#  
  
#Select variables with |correlation| greater than 0.50  
imp\_var <- corr[1] %>%  
 filter(abs(hpi) > 0.50)  
  
imp\_var$Name <- rownames(imp\_var)  
imp\_var <- imp\_var[,c(2,1)]

Below is *Table 4*, a table with our most correlated variables and their respective correlation with hpi:

#--- Data Visualization ---#  
  
#Map imp\_var data frame to flextable for display  
imp\_var %>%  
 flextable()%>%  
 set\_caption(  
 caption = as\_paragraph(  
 as\_chunk("Table 4: Variables with Highest Correlations with hpi",  
 props = fp\_text\_default(font.family = "Times New Roman",  
 font.size = 12,  
 italic = TRUE))))%>%  
 set\_header\_labels(values = list(  
 Name = "Name",  
 hpi = "Correlation with hpi")) %>%  
 set\_table\_properties(layout = "autofit") %>%  
 bg(i = seq(2,8,2), bg = "light grey") %>%  
 font(part = "all", fontname = "Times New Roman") %>%  
 fontsize(size = 12) %>%  
 border(i = c(1:8), border.top = fp\_border\_default(color = "black"))

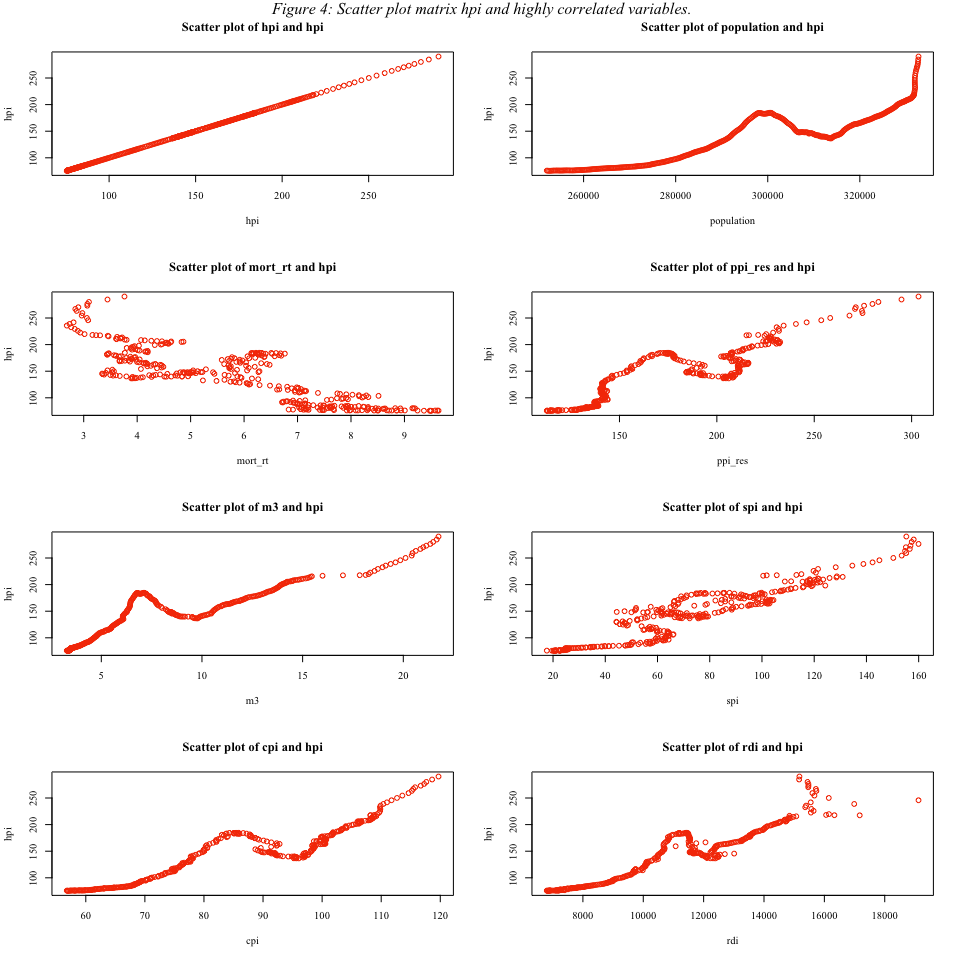
Table 4: Variables with Highest Correlations with hpi

| Name | Correlation with hpi |
| --- | --- |
| hpi | 1.0000000 |
| population | 0.9099283 |
| mort\_rt | -0.8299285 |
| ppi\_res | 0.9070114 |
| m3 | 0.9152493 |
| spi | 0.9369228 |
| cpi | 0.9272578 |
| rdi | 0.9404536 |

After obtaining these variables, we created scatter plots to more closely examine their relationships with hpi.

The final part of our analysis was creating scatter plots to examine the specific relationships between a subset of our variables and hpi. Below is *Figure 4*, a matrix of scatter plots of our most correlated variables and hpi:

#--- Data Visualization ---#  
#Create data set with only variables with strong correlation with hpi  
filtered\_data2 <- data2[,c(rownames(imp\_var))]  
  
#Create scatter plot matrix for 8 most important variables with hpi  
par(mfrow = c(4,2))  
for (i in 1:ncol(filtered\_data2)){  
 plot(x = filtered\_data2[,i], y = filtered\_data2[,1], xlab = colnames(filtered\_data2)[i], ylab = "hpi", main = paste("Scatter plot of", colnames(filtered\_data2)[i], "and hpi"), family = "Times New Roman", col = "#F63E02", pch = 21)  
}  
mtext(~italic("Figure 4: Scatter plot matrix hpi and highly correlated variables."), side = 3, line = -1.5, outer = TRUE, family = "Times New Roman")



Based on the above matrix, it is evident that, although these variables had linear correlations above 0.50 (or below -0.50), their relationships are not necessarily linear. For example, population and hpi appear to have a polynomial or exponential relationship with each other; though the relationship is clearly a positive one. Similarly, mortgage rates (mort\_rt) and hpi seem to have a strong negative relationship, though it may not be linear. Overall, the most linear relationship appears to be between total US money supply (m3) and hpi and the stock price index (spi) and hpi. This makes sense in that more money in the economy can result in inflation and stocks typically track alongside the housing market.

## Modeling

In order to model the behavior of hpi, we chose to explore a variety of models. We first started by splitting our data2 into a training and testing set.

#--- Pre-Model Splitting ---#  
  
#Load "Metrics" library"  
library(Metrics)  
  
#Change data naming and create train and test sets  
set.seed(100)  
train2 <- data2 %>% sample\_frac(size = 0.8)  
test2 <- data2 %>% setdiff(train2)

We used the training set to train our models to examine what are the most influential variables on hpi. We used the test set to validate our predictive models and assess which ones did the best at predicting out-of-sample hpi.

## *Multiple Linear Regression*

The first model we employed was a multiple linear regression. By running a multiple linear regression, we could get predictions of hpi on a test set and examine the coefficients and p-value of each of the explanatory variables to determine variable importance. We first ran a full formula linear regression. The output of our first model is shown below in *Table 5*:

#--- Multiple Linear Regression ---#  
  
#Generate multiple linear regression and obtain coefficients & p-values  
mod\_reg = lm(hpi~., data = train2)  
summary(mod\_reg)

##   
## Call:  
## lm(formula = hpi ~ ., data = train2)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -29.2734 -3.6199 -0.2711 3.3049 25.9542   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 9.598e+01 1.419e+02 0.676 0.499332   
## population 1.837e-04 4.306e-04 0.427 0.669981   
## hs 6.142e+00 4.963e-01 12.375 < 2e-16 \*\*\*  
## GDP 2.941e+00 8.224e-01 3.576 0.000409 \*\*\*  
## mort\_rt 1.006e+00 9.330e-01 1.078 0.281960   
## employ\_rt -3.435e+00 7.431e-01 -4.622 5.77e-06 \*\*\*  
## permit\_new 4.353e-02 2.828e-03 15.391 < 2e-16 \*\*\*  
## ppi\_res -1.979e-01 1.189e-01 -1.664 0.097295 .   
## m3 2.779e+00 7.815e-01 3.557 0.000440 \*\*\*  
## cci -3.084e+00 6.725e-01 -4.587 6.77e-06 \*\*\*  
## del\_rt -2.166e+00 6.166e-01 -3.513 0.000515 \*\*\*  
## spi 2.750e-01 6.702e-02 4.104 5.32e-05 \*\*\*  
## cpi 1.809e+00 8.131e-01 2.224 0.026914 \*   
## unemploy\_rt 7.195e-01 1.007e+00 0.715 0.475489   
## Real\_GDP 3.718e-02 6.559e-02 0.567 0.571271   
## rdi -5.621e-04 1.477e-03 -0.380 0.703892   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 6.351 on 283 degrees of freedom  
## Multiple R-squared: 0.985, Adjusted R-squared: 0.9842   
## F-statistic: 1238 on 15 and 283 DF, p-value: < 2.2e-16

#Map model to flextable for display  
mod\_reg %>%  
 as\_flextable()%>%  
 set\_caption(  
 caption = as\_paragraph(  
 as\_chunk("Table 5: Multiple Linear Regression Summary",  
 props = fp\_text\_default(font.family = "Times New Roman",  
 font.size = 12,  
 italic = TRUE)  
 )))%>%  
 set\_table\_properties(layout = "autofit") %>%  
 bg(i = seq(2,16,2), bg = "light grey") %>%  
 font(part = "all", fontname = "Times New Roman") %>%  
 fontsize(size = 12) %>%  
 border(i = c(1:16), border.top = fp\_border\_default(color = "black"))

Table 5: Multiple Linear Regression Summary

|  | Estimate | Standard Error | t value | Pr(>|t|) |  |
| --- | --- | --- | --- | --- | --- |
| (Intercept) | 95.977 | 141.892 | 0.676 | 0.4993 |  |
| population | 0.000 | 0.000 | 0.427 | 0.6700 |  |
| hs | 6.142 | 0.496 | 12.375 | 0.0000 | \*\*\* |
| GDP | 2.941 | 0.822 | 3.576 | 0.0004 | \*\*\* |
| mort\_rt | 1.006 | 0.933 | 1.078 | 0.2820 |  |
| employ\_rt | -3.435 | 0.743 | -4.622 | 0.0000 | \*\*\* |
| permit\_new | 0.044 | 0.003 | 15.391 | 0.0000 | \*\*\* |
| ppi\_res | -0.198 | 0.119 | -1.664 | 0.0973 | . |
| m3 | 2.779 | 0.781 | 3.557 | 0.0004 | \*\*\* |
| cci | -3.084 | 0.672 | -4.587 | 0.0000 | \*\*\* |
| del\_rt | -2.166 | 0.617 | -3.513 | 0.0005 | \*\*\* |
| spi | 0.275 | 0.067 | 4.104 | 0.0001 | \*\*\* |
| cpi | 1.809 | 0.813 | 2.224 | 0.0269 | \* |
| unemploy\_rt | 0.719 | 1.007 | 0.715 | 0.4755 |  |
| Real\_GDP | 0.037 | 0.066 | 0.567 | 0.5713 |  |
| rdi | -0.001 | 0.001 | -0.380 | 0.7039 |  |
| *Signif. codes: 0 <= '\*\*\*' < 0.001 < '\*\*' < 0.01 < '\*' < 0.05* | | | | | |
|  | | | | | |
| Residual standard error: 6.351 on 283 degrees of freedom | | | | | |
| Multiple R-squared: 0.985, Adjusted R-squared: 0.9842 | | | | | |
| F-statistic: 1238 on 283 and 15 DF, p-value: 0.0000 | | | | | |

From the above table, we found that hs, GDP, employ\_rt, permit\_new, m3, cci, del\_rt, spi, and cpi were all statistically significant predictors at least with an alpha of 0.05 among all 16 variables. After training the model, we evaluated it on our test set. For all of our predictive models, we will be using MAE, MSE, and RMSE as our three evaluation metrics. The values of these three metrics are shown in the code block below:

#--- Multiple Linear Regression ---#  
  
#Calculate model evaluation metrics on test set  
reg\_pred <- predict(mod\_reg, test2)  
Metrics::mae(test2$hpi, reg\_pred)

## [1] 4.667654

Metrics::mse(test2$hpi, reg\_pred)

## [1] 38.89087

Metrics::rmse(test2$hpi, reg\_pred)

## [1] 6.236255

After running a full formula multiple linear regression, we wanted to explore if a smaller model would more efficiently explain hpi. To do so, we calculated the AIC for all combinations of our multiple linear regression model and chose the model that minimized AIC. We then reran the regression with our optimal model, as shown below:

#Model selection AIC  
library(MASS)

##   
## Attaching package: 'MASS'

## The following object is masked \_by\_ '.GlobalEnv':  
##   
## housing

## The following object is masked from 'package:dplyr':  
##   
## select

final <- stepAIC(mod\_reg, direction = "both")

## Start: AIC=1121.07  
## hpi ~ population + hs + GDP + mort\_rt + employ\_rt + permit\_new +   
## ppi\_res + m3 + cci + del\_rt + spi + cpi + unemploy\_rt + Real\_GDP +   
## rdi  
##   
## Df Sum of Sq RSS AIC  
## - rdi 1 5.8 11422 1119.2  
## - population 1 7.3 11424 1119.3  
## - Real\_GDP 1 13.0 11430 1119.4  
## - unemploy\_rt 1 20.6 11437 1119.6  
## - mort\_rt 1 46.9 11463 1120.3  
## <none> 11417 1121.1  
## - ppi\_res 1 111.6 11528 1122.0  
## - cpi 1 199.6 11616 1124.2  
## - del\_rt 1 497.9 11914 1131.8  
## - m3 1 510.3 11927 1132.1  
## - GDP 1 516.0 11933 1132.3  
## - spi 1 679.4 12096 1136.4  
## - cci 1 848.7 12265 1140.5  
## - employ\_rt 1 862.0 12278 1140.8  
## - hs 1 6177.7 17594 1248.4  
## - permit\_new 1 9556.0 20973 1300.9  
##   
## Step: AIC=1119.22  
## hpi ~ population + hs + GDP + mort\_rt + employ\_rt + permit\_new +   
## ppi\_res + m3 + cci + del\_rt + spi + cpi + unemploy\_rt + Real\_GDP  
##   
## Df Sum of Sq RSS AIC  
## - population 1 3.8 11426 1117.3  
## - unemploy\_rt 1 15.2 11438 1117.6  
## - Real\_GDP 1 15.7 11438 1117.6  
## - mort\_rt 1 43.9 11466 1118.4  
## <none> 11422 1119.2  
## - ppi\_res 1 106.6 11529 1120.0  
## + rdi 1 5.8 11417 1121.1  
## - cpi 1 203.8 11626 1122.5  
## - del\_rt 1 499.2 11922 1130.0  
## - GDP 1 516.5 11939 1130.5  
## - m3 1 566.5 11989 1131.7  
## - spi 1 676.6 12099 1134.4  
## - cci 1 845.5 12268 1138.6  
## - employ\_rt 1 899.9 12322 1139.9  
## - hs 1 6185.0 17607 1246.6  
## - permit\_new 1 9557.1 20980 1299.0  
##   
## Step: AIC=1117.32  
## hpi ~ hs + GDP + mort\_rt + employ\_rt + permit\_new + ppi\_res +   
## m3 + cci + del\_rt + spi + cpi + unemploy\_rt + Real\_GDP  
##   
## Df Sum of Sq RSS AIC  
## - unemploy\_rt 1 13.3 11440 1115.7  
## - Real\_GDP 1 17.1 11443 1115.8  
## - mort\_rt 1 40.1 11466 1116.4  
## <none> 11426 1117.3  
## + population 1 3.8 11422 1119.2  
## + rdi 1 2.3 11424 1119.3  
## - ppi\_res 1 288.2 11714 1122.8  
## - GDP 1 518.3 11944 1128.6  
## - del\_rt 1 544.2 11970 1129.2  
## - m3 1 568.8 11995 1129.8  
## - spi 1 672.9 12099 1132.4  
## - employ\_rt 1 914.3 12340 1138.3  
## - cci 1 1482.9 12909 1151.8  
## - cpi 1 4307.7 15734 1211.0  
## - hs 1 6877.5 18304 1256.2  
## - permit\_new 1 10474.2 21900 1309.8  
##   
## Step: AIC=1115.67  
## hpi ~ hs + GDP + mort\_rt + employ\_rt + permit\_new + ppi\_res +   
## m3 + cci + del\_rt + spi + cpi + Real\_GDP  
##   
## Df Sum of Sq RSS AIC  
## - Real\_GDP 1 18.2 11458 1114.2  
## - mort\_rt 1 43.1 11483 1114.8  
## <none> 11440 1115.7  
## + unemploy\_rt 1 13.3 11426 1117.3  
## + population 1 1.9 11438 1117.6  
## + rdi 1 0.1 11439 1117.7  
## - ppi\_res 1 281.0 11720 1120.9  
## - del\_rt 1 549.0 11988 1127.7  
## - m3 1 556.4 11996 1127.9  
## - GDP 1 646.1 12086 1130.1  
## - spi 1 756.6 12196 1132.8  
## - cci 1 1613.7 13053 1153.1  
## - employ\_rt 1 2447.2 13887 1171.6  
## - cpi 1 4820.6 16260 1218.8  
## - hs 1 7275.0 18714 1260.8  
## - permit\_new 1 11268.2 22708 1318.7  
##   
## Step: AIC=1114.15  
## hpi ~ hs + GDP + mort\_rt + employ\_rt + permit\_new + ppi\_res +   
## m3 + cci + del\_rt + spi + cpi  
##   
## Df Sum of Sq RSS AIC  
## - mort\_rt 1 40.5 11498 1113.2  
## <none> 11458 1114.2  
## + Real\_GDP 1 18.2 11440 1115.7  
## + unemploy\_rt 1 14.4 11443 1115.8  
## + population 1 2.9 11455 1116.1  
## + rdi 1 0.3 11457 1116.1  
## - ppi\_res 1 307.3 11765 1120.1  
## - del\_rt 1 531.3 11989 1125.7  
## - m3 1 635.3 12093 1128.3  
## - GDP 1 747.5 12205 1131.0  
## - spi 1 763.4 12221 1131.4  
## - cci 1 1674.8 13132 1152.9  
## - employ\_rt 1 2438.3 13896 1169.8  
## - cpi 1 4826.7 16284 1217.3  
## - hs 1 7264.9 18722 1259.0  
## - permit\_new 1 11444.4 22902 1319.2  
##   
## Step: AIC=1113.2  
## hpi ~ hs + GDP + employ\_rt + permit\_new + ppi\_res + m3 + cci +   
## del\_rt + spi + cpi  
##   
## Df Sum of Sq RSS AIC  
## <none> 11498 1113.2  
## + mort\_rt 1 40.5 11458 1114.2  
## + unemploy\_rt 1 17.3 11481 1114.8  
## + Real\_GDP 1 15.6 11483 1114.8  
## + rdi 1 0.2 11498 1115.2  
## + population 1 0.1 11498 1115.2  
## - ppi\_res 1 285.1 11783 1118.5  
## - del\_rt 1 604.7 12103 1126.5  
## - m3 1 607.2 12105 1126.6  
## - GDP 1 739.1 12237 1129.8  
## - spi 1 907.0 12405 1133.9  
## - cci 1 1655.0 13153 1151.4  
## - employ\_rt 1 2398.1 13896 1167.8  
## - cpi 1 7100.9 18599 1255.0  
## - hs 1 8353.6 19852 1274.5  
## - permit\_new 1 11553.1 23051 1319.2

final$anova

## Stepwise Model Path   
## Analysis of Deviance Table  
##   
## Initial Model:  
## hpi ~ population + hs + GDP + mort\_rt + employ\_rt + permit\_new +   
## ppi\_res + m3 + cci + del\_rt + spi + cpi + unemploy\_rt + Real\_GDP +   
## rdi  
##   
## Final Model:  
## hpi ~ hs + GDP + employ\_rt + permit\_new + ppi\_res + m3 + cci +   
## del\_rt + spi + cpi  
##   
##   
## Step Df Deviance Resid. Df Resid. Dev AIC  
## 1 283 11416.56 1121.071  
## 2 - rdi 1 5.839255 284 11422.40 1119.224  
## 3 - population 1 3.789641 285 11426.19 1117.323  
## 4 - unemploy\_rt 1 13.295951 286 11439.49 1115.670  
## 5 - Real\_GDP 1 18.190683 287 11457.68 1114.146  
## 6 - mort\_rt 1 40.518270 288 11498.20 1113.201

#Fit AIC model  
mod\_AIC\_reg <- lm(hpi ~ hs + GDP + employ\_rt + permit\_new + ppi\_res + m3 + cci + del\_rt + spi + cpi, data = train2)

The output of our AIC-minimized model is shown below in *Table 6*:

mod\_AIC\_reg %>%  
 as\_flextable()%>%  
 set\_caption(  
 caption = as\_paragraph(  
 as\_chunk("Table 6: AIC-Minimized Multiple Linear Regression Summary",  
 props = fp\_text\_default(font.family = "Times New Roman",  
 font.size = 12,  
 italic = TRUE)  
 )))%>%  
 set\_table\_properties(layout = "autofit") %>%  
 bg(i = seq(2,11,2), bg = "light grey") %>%  
 font(part = "all", fontname = "Times New Roman") %>%  
 fontsize(size = 12) %>%  
 border(i = c(1:11), border.top = fp\_border\_default(color = "black"))

Table 6: AIC-Minimized Multiple Linear Regression Summary

|  | Estimate | Standard Error | t value | Pr(>|t|) |  |
| --- | --- | --- | --- | --- | --- |
| (Intercept) | 187.749 | 68.415 | 2.744 | 0.0064 | \*\* |
| hs | 6.339 | 0.438 | 14.465 | 0.0000 | \*\*\* |
| GDP | 2.655 | 0.617 | 4.303 | 0.0000 | \*\*\* |
| employ\_rt | -3.701 | 0.478 | -7.750 | 0.0000 | \*\*\* |
| permit\_new | 0.044 | 0.003 | 17.011 | 0.0000 | \*\*\* |
| ppi\_res | -0.212 | 0.079 | -2.672 | 0.0080 | \*\* |
| m3 | 2.597 | 0.666 | 3.900 | 0.0001 | \*\*\* |
| cci | -3.008 | 0.467 | -6.438 | 0.0000 | \*\*\* |
| del\_rt | -1.957 | 0.503 | -3.892 | 0.0001 | \*\*\* |
| spi | 0.295 | 0.062 | 4.766 | 0.0000 | \*\*\* |
| cpi | 1.891 | 0.142 | 13.336 | 0.0000 | \*\*\* |
| *Signif. codes: 0 <= '\*\*\*' < 0.001 < '\*\*' < 0.01 < '\*' < 0.05* | | | | | |
|  | | | | | |
| Residual standard error: 6.319 on 288 degrees of freedom | | | | | |
| Multiple R-squared: 0.9849, Adjusted R-squared: 0.9844 | | | | | |
| F-statistic: 1876 on 288 and 10 DF, p-value: 0.0000 | | | | | |

By comparing *Table 5* and *Table 6*, we can see the effects of AIC minimization. All of the coefficients in *Table 6* are significant at at least the alpha = 0.01 level. Also, the residual standard error is lower in *Table 6*, indicating that the AIC-minimized model explained more of the variation than the full formula model.

Finally, we calculated our evaluation metrics for the AIC-minimized model, as shown in the code block below:

#--- Multiple Linear Regression ---#  
  
#Calculate model evaluation metrics on test set -- slightly better MSE  
reg\_AIC\_pred <- predict(mod\_AIC\_reg, test2)  
Metrics::mae(test2$hpi, reg\_AIC\_pred)

## [1] 4.690139

Metrics::mse(test2$hpi, reg\_AIC\_pred)

## [1] 38.68806

Metrics::rmse(test2$hpi, reg\_AIC\_pred)

## [1] 6.219973

## *Bootstrap Regression*

Bootstrapping takes advantage of the concept of re sampling to generate a distribution-free interval for each variable’s coefficient. Because bootstrapping is rather computationally heavy, we decided to only run one bootstrap model on the full formula over 100 samples. After running the bootstrap regression, we calculated the 95% confidence interval for each explanatory variable’s coefficient to get a better idea of the specific effect each variable may have on hpi.

#--- Bootstrap Regression ---#  
  
#Load libraries  
library(broom)  
library(modelr)

##   
## Attaching package: 'modelr'

## The following object is masked from 'package:broom':  
##   
## bootstrap

## The following objects are masked from 'package:Metrics':  
##   
## mae, mape, mse, rmse

library(purrr)  
  
#Initialize bootstrap and generate models  
boot <- modelr:: bootstrap(train2, 100)  
mods <- map(boot$strap, ~lm(hpi~., data = .))  
mods\_boot <- map\_df(mods, broom::tidy, .id = "id")  
mods\_boot

## # A tibble: 1,600 × 6  
## id term estimate std.error statistic p.value  
## <chr> <chr> <dbl> <dbl> <dbl> <dbl>  
## 1 1 (Intercept) -153. 144. -1.06 2.89e- 1  
## 2 1 population 0.000792 0.000498 1.59 1.12e- 1  
## 3 1 hs 5.71 0.513 11.1 3.96e-24  
## 4 1 GDP 3.79 0.796 4.76 3.03e- 6  
## 5 1 mort\_rt -0.485 0.924 -0.524 6.00e- 1  
## 6 1 employ\_rt -3.14 0.725 -4.33 2.10e- 5  
## 7 1 permit\_new 0.0397 0.00282 14.1 1.46e-34  
## 8 1 ppi\_res -0.153 0.134 -1.14 2.53e- 1  
## 9 1 m3 2.83 0.842 3.36 8.72e- 4  
## 10 1 cci -2.51 0.720 -3.48 5.75e- 4  
## # ℹ 1,590 more rows

#Create list of column names (excluding "hpi") for data2  
columnNames = colnames(data2)  
columnNames = columnNames[-1]  
  
#Generate data frame to hold bootstrapped regression coefficients  
boot\_results <- data.frame(Name = character(),  
 LL = double(),  
 UL = double())  
  
#Calculate 95% confidence interval of coefficient for each variable  
for (i in 1:length(columnNames)){  
 x <- mods\_boot %>%  
 filter(term == columnNames[i]) %>%  
 summarize(Name = columnNames[i],  
 LL = quantile(estimate, 0.025),  
 UL = quantile(estimate, 0.975))  
 boot\_results <- rbind(boot\_results, x)  
}

*Table 7* below lists the lower and upper bounds of the 95% confidence estimate of each of our explanatory variable’s coefficients for predicting hpi:

#--- Bootstrap Regression ---#  
  
#Map 95% confidence interval of coefficients to flextable for display  
boot\_results %>%  
 flextable()%>%  
 set\_caption(  
 caption = as\_paragraph(  
 as\_chunk("Table 7: 95% Confidence Interval of Bootstrap Regression Coefficients",  
 props = fp\_text\_default(font.family = "Times New Roman",  
 font.size = 12,  
 italic = TRUE)  
 )))%>%  
 set\_table\_properties(layout = "autofit") %>%  
 bg(i = seq(2,14,2), bg = "light grey") %>%  
 font(part = "all", fontname = "Times New Roman") %>%  
 fontsize(size = 12) %>%  
 border(i = c(1:15), border.top = fp\_border\_default(color = "black"))

Table 7: 95% Confidence Interval of Bootstrap Regression Coefficients

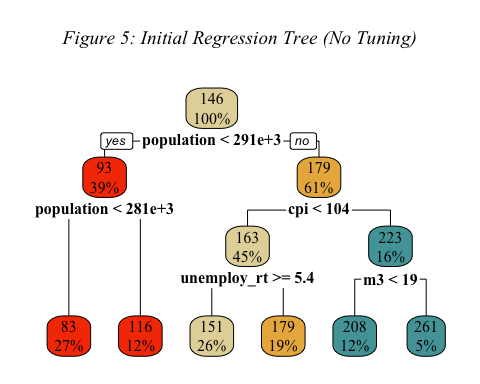
| Name | LL | UL |
| --- | --- | --- |
| population | -0.000888833 | 0.001361278 |
| hs | 4.977318690 | 7.169875596 |
| GDP | 1.817120801 | 5.013549996 |
| mort\_rt | -0.883564849 | 2.640298105 |
| employ\_rt | -4.720348670 | -1.870629598 |
| permit\_new | 0.034589210 | 0.048405272 |
| ppi\_res | -0.482813772 | 0.137006618 |
| m3 | 0.511579745 | 4.184742773 |
| cci | -4.367415434 | -1.252214392 |
| del\_rt | -4.276083127 | -0.879129987 |
| spi | 0.154161626 | 0.412403937 |
| cpi | -0.717557996 | 3.809914557 |
| unemploy\_rt | -1.352731526 | 4.812212958 |
| Real\_GDP | -0.399501220 | 0.208794493 |
| rdi | -0.003555624 | 0.003199015 |

The above table gives us insight into the specific range of effects each variable may have on hpi. Notably, several variables’ 95% confidence range of coefficients contain 0 (specifically rdi, Real\_GDP, unemploy\_rt, cpi, ppi\_res, mort\_rt, and population).

## *Regression Tree*

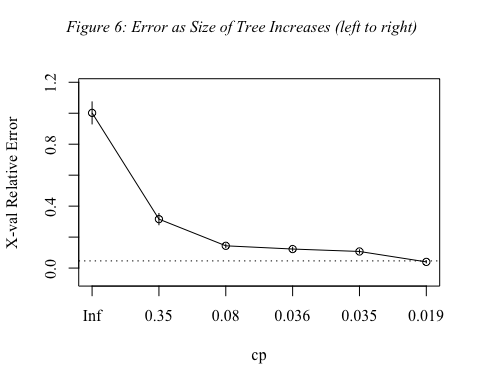
After examining our linear regression models, we decided to explore non-parametric methods to understand variable importance and develop a predictive model for hpi. The first non-parametric model we tried was a regression tree.

#--- Regression Tree ---#  
  
#Load libraries  
library(rpart)  
library(rpart.plot)  
  
#Set seed before creating tree  
set.seed(100)  
  
#Fit tree model  
mod\_tree <- rpart(hpi~.,data=train2, method = "anova")  
  
#Visualize regression tree  
rpart.plot(mod\_tree,   
 main = substitute(  
 paste(  
 italic("Figure 5: Initial Regression Tree (No Tuning)"))),  
 family = "Times New Roman",  
 box.palette = c("#F63E02","#E4D6A7","#E9B44C","#50A2A7"))



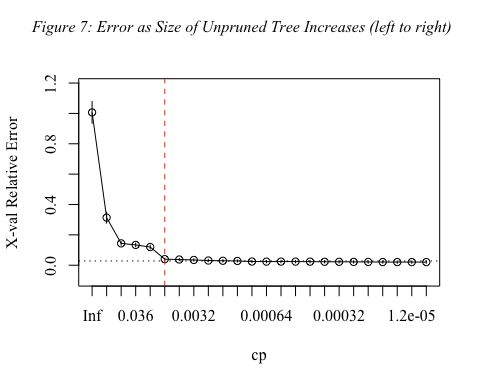
Based on our initial tree, population, cpi, unemploy\_rt, and m3 are the most important splits when regressing hpi. It is possible to prune a regression tree by finding the optimal complexity parameter, minimum observations in each split, and maximum number of terminal nodes. To optimize our model using these values, we first visualized the error as the size of the tree increased, as shown below in *Figure 6*:

#--- Regression Tree ---#  
  
#Visualize cp-size trade off  
plotcp(mod\_tree, upper = c("none"), family = "Times New Roman")  
mtext(~italic("Figure 6: Error as Size of Tree Increases (left to right)"), side = 3, line = -2, outer = TRUE, family = "Times New Roman")



From *Figure 6*, a maxdepth of 6 seemed to be optimal. We confirmed this by creating a regression tree with no cp and graphing its decrease in error as tree size increased. *Figure 7* below shows the un-pruned cp plot with a mark at maxdepth = 6. It is clear that the decrease in error drops off after the red line.

#--- Regression Tree ---#  
  
#Show that maxdepth = 6 seems best by visualizing a full tree with cp = 0  
full\_tree <- rpart(hpi~., data = train2, method = "anova",  
 control = list(cp = 0, xval = 10))  
plotcp(full\_tree, upper = c("none"), family = "Times New Roman")  
abline(v = 6, lty = "dashed", col = "#F63E02") #Decrease in error drops off after 6  
mtext(~italic("Figure 7: Error as Size of Unpruned Tree Increases (left to right)"), side = 3, line = -2, outer = TRUE, family = "Times New Roman")



After confirming the optimal maxdepth value, we calculated our evaluation metrics for the un-tuned regression tree on the test set, as shown in the code block below:

#--- Regression Tree ---#  
  
#Calculate evaluation metrics on testing set  
tree\_pred <- predict(mod\_tree, test2)  
Metrics::mae(test2$hpi, tree\_pred)

## [1] 7.766737

Metrics::mse(test2$hpi, tree\_pred)

## [1] 90.44806

Metrics::rmse(test2$hpi, tree\_pred)

## [1] 9.510418

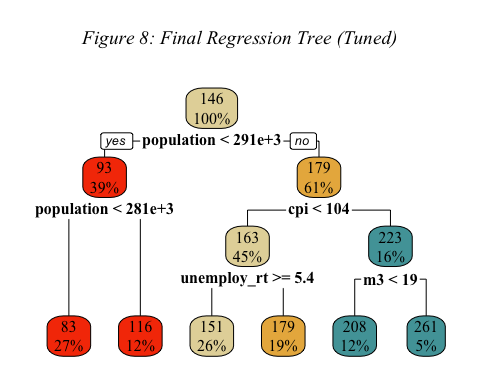
We then performed a grid search to find the minsplit and cp value that minimized the error in the regression tree.

#--- Regression Tree ---#  
  
#Grid search to tune minsplit and maxdepth  
tree\_grid <- expand.grid(  
 minsplit = seq(5,20,1),  
 maxdepth = seq(6,12,1)  
)  
  
#Iterate through grid and generate a tree for each combination of hyperparameter  
mods\_tree <- list()  
  
for (i in 1:nrow(tree\_grid)){  
 minsplit <- tree\_grid$minsplit[i]  
 maxdepth <- tree\_grid$maxdepth[i]  
   
 set.seed(100)  
 mods\_tree[[i]]<- rpart(hpi~., data = train2, method = "anova",  
 control = list(minsplit = minsplit, maxdepth = maxdepth))  
}  
  
#Extract minimum error associated with ccp value of each model:  
  
#Optimal cp function  
get\_cp <- function(y){  
 min <- which.min(y$cptable[,"xerror"])  
 cp <- y$cptable[min,"CP"]  
}  
  
#Minimum error function  
get\_min\_error <- function(y){  
 min <- which.min(y$cptable[,"xerror"])  
 xerror <- y$cptable[min,"xerror"]  
}  
  
#Find optimal values  
tree\_grid %>%  
 mutate(  
 cp = purrr::map\_dbl(mods\_tree, get\_cp),  
 error = purrr:: map\_dbl(mods\_tree, get\_min\_error)  
 ) %>%  
 arrange(error) %>%  
 top\_n(-5, wt = error) #minsplit = 5, cp = 0.01

## minsplit maxdepth cp error  
## 1 5 6 0.01 0.03982692  
## 2 6 6 0.01 0.03982692  
## 3 7 6 0.01 0.03982692  
## 4 8 6 0.01 0.03982692  
## 5 9 6 0.01 0.03982692  
## 6 10 6 0.01 0.03982692  
## 7 11 6 0.01 0.03982692  
## 8 12 6 0.01 0.03982692  
## 9 13 6 0.01 0.03982692  
## 10 14 6 0.01 0.03982692  
## 11 15 6 0.01 0.03982692  
## 12 16 6 0.01 0.03982692  
## 13 17 6 0.01 0.03982692  
## 14 18 6 0.01 0.03982692  
## 15 19 6 0.01 0.03982692  
## 16 20 6 0.01 0.03982692  
## 17 5 7 0.01 0.03982692  
## 18 6 7 0.01 0.03982692  
## 19 7 7 0.01 0.03982692  
## 20 8 7 0.01 0.03982692  
## 21 9 7 0.01 0.03982692  
## 22 10 7 0.01 0.03982692  
## 23 11 7 0.01 0.03982692  
## 24 12 7 0.01 0.03982692  
## 25 13 7 0.01 0.03982692  
## 26 14 7 0.01 0.03982692  
## 27 15 7 0.01 0.03982692  
## 28 16 7 0.01 0.03982692  
## 29 17 7 0.01 0.03982692  
## 30 18 7 0.01 0.03982692  
## 31 19 7 0.01 0.03982692  
## 32 20 7 0.01 0.03982692  
## 33 5 8 0.01 0.03982692  
## 34 6 8 0.01 0.03982692  
## 35 7 8 0.01 0.03982692  
## 36 8 8 0.01 0.03982692  
## 37 9 8 0.01 0.03982692  
## 38 10 8 0.01 0.03982692  
## 39 11 8 0.01 0.03982692  
## 40 12 8 0.01 0.03982692  
## 41 13 8 0.01 0.03982692  
## 42 14 8 0.01 0.03982692  
## 43 15 8 0.01 0.03982692  
## 44 16 8 0.01 0.03982692  
## 45 17 8 0.01 0.03982692  
## 46 18 8 0.01 0.03982692  
## 47 19 8 0.01 0.03982692  
## 48 20 8 0.01 0.03982692  
## 49 5 9 0.01 0.03982692  
## 50 6 9 0.01 0.03982692  
## 51 7 9 0.01 0.03982692  
## 52 8 9 0.01 0.03982692  
## 53 9 9 0.01 0.03982692  
## 54 10 9 0.01 0.03982692  
## 55 11 9 0.01 0.03982692  
## 56 12 9 0.01 0.03982692  
## 57 13 9 0.01 0.03982692  
## 58 14 9 0.01 0.03982692  
## 59 15 9 0.01 0.03982692  
## 60 16 9 0.01 0.03982692  
## 61 17 9 0.01 0.03982692  
## 62 18 9 0.01 0.03982692  
## 63 19 9 0.01 0.03982692  
## 64 20 9 0.01 0.03982692  
## 65 5 10 0.01 0.03982692  
## 66 6 10 0.01 0.03982692  
## 67 7 10 0.01 0.03982692  
## 68 8 10 0.01 0.03982692  
## 69 9 10 0.01 0.03982692  
## 70 10 10 0.01 0.03982692  
## 71 11 10 0.01 0.03982692  
## 72 12 10 0.01 0.03982692  
## 73 13 10 0.01 0.03982692  
## 74 14 10 0.01 0.03982692  
## 75 15 10 0.01 0.03982692  
## 76 16 10 0.01 0.03982692  
## 77 17 10 0.01 0.03982692  
## 78 18 10 0.01 0.03982692  
## 79 19 10 0.01 0.03982692  
## 80 20 10 0.01 0.03982692  
## 81 5 11 0.01 0.03982692  
## 82 6 11 0.01 0.03982692  
## 83 7 11 0.01 0.03982692  
## 84 8 11 0.01 0.03982692  
## 85 9 11 0.01 0.03982692  
## 86 10 11 0.01 0.03982692  
## 87 11 11 0.01 0.03982692  
## 88 12 11 0.01 0.03982692  
## 89 13 11 0.01 0.03982692  
## 90 14 11 0.01 0.03982692  
## 91 15 11 0.01 0.03982692  
## 92 16 11 0.01 0.03982692  
## 93 17 11 0.01 0.03982692  
## 94 18 11 0.01 0.03982692  
## 95 19 11 0.01 0.03982692  
## 96 20 11 0.01 0.03982692  
## 97 5 12 0.01 0.03982692  
## 98 6 12 0.01 0.03982692  
## 99 7 12 0.01 0.03982692  
## 100 8 12 0.01 0.03982692  
## 101 9 12 0.01 0.03982692  
## 102 10 12 0.01 0.03982692  
## 103 11 12 0.01 0.03982692  
## 104 12 12 0.01 0.03982692  
## 105 13 12 0.01 0.03982692  
## 106 14 12 0.01 0.03982692  
## 107 15 12 0.01 0.03982692  
## 108 16 12 0.01 0.03982692  
## 109 17 12 0.01 0.03982692  
## 110 18 12 0.01 0.03982692  
## 111 19 12 0.01 0.03982692  
## 112 20 12 0.01 0.03982692

After running the grid search, we found that the optimal parameters for our regression tree were a minsplit of 5, cp of 0.01, and maxdepth of 6. We then fit our optimized tree and visualized it, as shown by *Figure 8* below:

#--- Regression Tree ---#  
  
#Fit optimal tree  
set.seed(100)  
optimal\_mod\_tree <- rpart(hpi~., data = train2, method = "anova",  
 control = list(minsplit = 5, maxdepth = 6, cp = 0.01))  
  
#Visualize optimal regression tree  
rpart.plot(optimal\_mod\_tree,   
 main = substitute(  
 paste(  
 italic("Figure 8: Final Regression Tree (Tuned)"))),  
 family = "Times New Roman",  
 box.palette = c("#F63E02","#E4D6A7","#E9B44C","#50A2A7"))



Comparing *Figure 5* and *Figure 8*, it is clear that the two trees are identical. This means that the rpart algorithm was able to find the optimal values on its own, likely because our training data was quite small. After pruning our tree, we calculated its evaluation metrics on the test set, as shown in the code block below:

#Calculate evaluation metrics on testing set -- looks like origingal algorithm was optimal!  
optimal\_tree\_pred <- predict(optimal\_mod\_tree, test2)  
Metrics::mae(test2$hpi, optimal\_tree\_pred)

## [1] 7.766737

Metrics::mse(test2$hpi, optimal\_tree\_pred)

## [1] 90.44806

Metrics::rmse(test2$hpi, optimal\_tree\_pred)

## [1] 9.510418

## *Random Forest*

After our regression tree analysis, we wanted to add an ensemble-learning method to our analysis. So, we decided to run a random forest and then tune it. The below code block shows our initial random forest model:

#--- Random Forest ---#  
  
#Load library  
library(randomForest)

## randomForest 4.7-1.1

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

##   
## Attaching package: 'randomForest'

## The following object is masked from 'package:psych':  
##   
## outlier

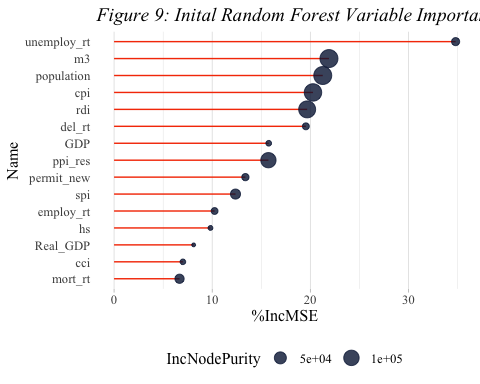
## The following object is masked from 'package:ggplot2':  
##   
## margin

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

#Set seed before running forest  
set.seed(100)  
  
#Fit random forest model  
mod\_rf<- randomForest(hpi~.,train2,ntree=1000, importance = TRUE)

After fitting our initial random forest, we extracted the variable importance and visualized it. *Figure 9* below illustrates the variable importance of the random forest sorted by percent increase in MSE:

#--- Random Forest ---#  
#Get variable importance from the model fit  
ImpData <- as.data.frame(importance(mod\_rf))  
ImpData$Var.Names <- row.names(ImpData)  
  
ImpData %>%  
 arrange(`%IncMSE`) %>%  
 mutate(Name = factor(Var.Names, levels = Var.Names)) %>%  
 ggplot(aes(x = Name, y = `%IncMSE`)) +   
 geom\_segment(aes(x = Name, xend = Name, y = 0, yend = `%IncMSE`),color="#F63E02") +  
 geom\_point(aes(size = IncNodePurity), color = "#061A40", alpha = 0.8) +   
 ggtitle("\*Figure 9: Inital Random Forest Variable Importance\*") +   
 theme\_light() +   
 coord\_flip() +   
 theme(  
 legend.position = "bottom",  
 panel.grid.major.y = element\_blank(),  
 panel.border = element\_blank(),  
 axis.ticks.y = element\_blank(),  
 text = element\_text(family = "Times New Roman", size = 12),  
 plot.title = ggtext::element\_markdown()  
 )



We ranked by percentage increase in MSE because it is the more informative metric. This metric tells us how much the prediction MSE increases when a particular variable is permuted. Based on *Figure 9*, unemploy\_rt, m3, and population are the three most important variables in predicting hpi.

After looking at the initial variable importance plot, we calculated the relevant model evaluation metrics for the un-tuned random forest, as shown in the code block below:

#--- Random Forest ---#  
  
#Calculate evaluation metrics on testing set  
rf\_pred <- predict(mod\_rf, test2)  
Metrics::mae(test2$hpi, rf\_pred)

## [1] 0.8147173

Metrics::mse(test2$hpi, rf\_pred)

## [1] 1.905132

Metrics::rmse(test2$hpi, rf\_pred)

## [1] 1.380265

After constructing our initial random forest, we conducted a grid search to find the optimal mtry value. The mtry parameter sets how many variables the model will randomly sample as candidates for each split. Optimizing mtry allows us to manage the trade off between performance and over fitting. The code block below shows the process of constructing and executing the grid search with a 5-fold CV:

#--- Random Forest ---#  
  
#Grid search to tune mtry:  
  
#Load caret library  
library(caret)

## Loading required package: lattice

##   
## Attaching package: 'caret'

## The following objects are masked from 'package:Metrics':  
##   
## precision, recall

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

#Set control procedure  
control <- trainControl(method="repeatedcv", number=5,   
 repeats=2, search="grid")  
  
#Set seed, define grid, and generate models for grid of mtry values  
set.seed(100)  
tunegrid <- expand.grid(.mtry=seq(1,15,1))  
rf\_gridsearch <- train(hpi~., data=train2, method="rf",   
 metric="RMSE", tuneGrid=tunegrid,   
 trControl=control)  
  
#Display grid search results -- mtry = 5 minimizes RMSE  
print(rf\_gridsearch)

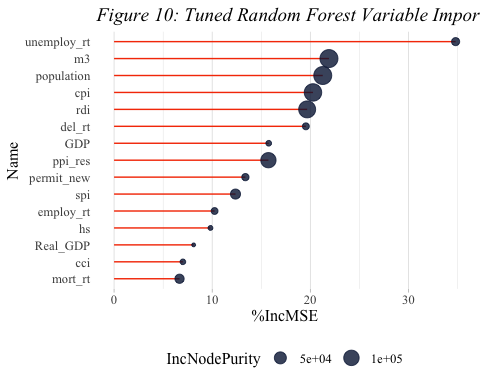
## Random Forest   
##   
## 299 samples  
## 15 predictor  
##   
## No pre-processing  
## Resampling: Cross-Validated (5 fold, repeated 2 times)   
## Summary of sample sizes: 239, 240, 239, 239, 239, 240, ...   
## Resampling results across tuning parameters:  
##   
## mtry RMSE Rsquared MAE   
## 1 4.104835 0.9952489 2.331992  
## 2 2.595205 0.9975025 1.348732  
## 3 2.156058 0.9982521 1.122895  
## 4 1.901123 0.9985992 1.044715  
## 5 1.821993 0.9987336 1.008271  
## 6 1.823465 0.9987256 1.018790  
## 7 1.823434 0.9987312 1.034446  
## 8 1.884837 0.9986404 1.060747  
## 9 1.895100 0.9986254 1.082900  
## 10 1.979051 0.9985289 1.122642  
## 11 2.026654 0.9984365 1.150372  
## 12 2.097375 0.9983291 1.178892  
## 13 2.162755 0.9982386 1.210058  
## 14 2.202240 0.9981663 1.237776  
## 15 2.260422 0.9980437 1.264242  
##   
## RMSE was used to select the optimal model using the smallest value.  
## The final value used for the model was mtry = 5.

As shown above, the optimal mtry to minimize RMSE is 5. After confirming the optimal mtry value, we fit our tuned random forest model, as shown in the code block below:

#--- Random Forest ---#  
  
#Fit tuned random forest model  
set.seed(100)  
mod\_tuned\_rf <- randomForest(hpi~., data=train2,   
 ntree=1000, mtry=5, importance = TRUE)

After fitting our tuned model, we extracted the variable importance values (percent decrease in MSE and increase in node purity) and visualized them, as shown in *Figure 10* below:

#--- Random Forest ---#  
  
#Get variable importance from the tuned model fit  
tuned\_ImpData <- as.data.frame(importance(mod\_tuned\_rf))  
tuned\_ImpData$Var.Names <- row.names(tuned\_ImpData)  
  
tuned\_ImpData %>%  
 arrange(`%IncMSE`) %>%  
 mutate(Name = factor(Var.Names, levels = Var.Names)) %>%  
 ggplot(aes(x = Name, y = `%IncMSE`)) +   
 geom\_segment(aes(x = Name, xend = Name, y = 0, yend = `%IncMSE`),color="#F63E02") +  
 geom\_point(aes(size = IncNodePurity), color = "#061A40", alpha = 0.8) +   
 ggtitle("\*Figure 10: Tuned Random Forest Variable Importance\*") +   
 theme\_light() +   
 coord\_flip() +   
 theme(  
 legend.position = "bottom",  
 panel.grid.major.y = element\_blank(),  
 panel.border = element\_blank(),  
 axis.ticks.y = element\_blank(),  
 text = element\_text(family = "Times New Roman", size = 12),  
 plot.title = ggtext::element\_markdown()  
 )



*Figure 10* above shows us that unemploy\_rt, m3, and population were the three most important variables in our tuned random forest (same as our un-tuned model). It is not too surprising that the top variables are still on top, as tuning primarily helps with our model’s predictive ability.

After examining the variable importance plot, we calculated the evaluation metrics for the tuned random forest. There was a noticeable improvement in MSE and RMSE after tuning, which was encouraging for our goal of predicting hpi. The evaluation metrics for the tuned random forest are listed in the code bock below:

#--- Random Forest ---#  
#Calculate evaluation metrics on the test set-- there was an improvement!  
tuned\_rf\_pred <- predict(mod\_tuned\_rf, test2)  
Metrics::mae(test2$hpi, tuned\_rf\_pred)

## [1] 0.8147173

Metrics::mse(test2$hpi, tuned\_rf\_pred)

## [1] 1.905132

Metrics::rmse(test2$hpi, tuned\_rf\_pred)

## [1] 1.380265

## Interpretation of Results

Across all of our models, only m3 showed up consistently as an influential variable. This could indicate some sustained effect m3 has hpi regardless of method of analysis. It is important to note that employ\_rt, which is directly correlated with unemploy\_rt was significant in our MLR. The fact that m3 showed up consistently across models indicates that it may be an important metric to track for economists interested in housing markets or in the growth of the housing sector.

In addition to looking at influential variables, we wanted to assess which of our models was best at predicting hpi using out-of-sample observations. *Table 8* below summarizes the model evaluation statistics for each of our predictive models:

#--- Results ---#  
  
#Create empty data frame to hold model evaluation metrics  
mod\_eval <- data.frame(Model = character(),  
 MAE = double(),  
 MSE = double(),  
 RMSE = double())  
  
#Add each model and its metric to the data frame  
mod\_eval <- rbind(mod\_eval, list("MLR", Metrics::mae(test2$hpi, reg\_pred), Metrics::mse(test2$hpi, reg\_pred), Metrics::rmse(test2$hpi, reg\_pred)))  
  
mod\_eval <- rbind(mod\_eval, list("AIC-Min MLR", Metrics::mae(test2$hpi, reg\_AIC\_pred), Metrics::mse(test2$hpi, reg\_AIC\_pred), Metrics::rmse(test2$hpi, reg\_AIC\_pred)))  
  
mod\_eval <- rbind(mod\_eval, list("Reg Tree", Metrics::mae(test2$hpi, tree\_pred), Metrics::mse(test2$hpi, tree\_pred), Metrics::rmse(test2$hpi, tree\_pred)))  
  
mod\_eval <- rbind(mod\_eval, list("Pruned Reg Tree", Metrics::mae(test2$hpi, optimal\_tree\_pred), Metrics::mse(test2$hpi, optimal\_tree\_pred), Metrics::rmse(test2$hpi, optimal\_tree\_pred)))  
  
mod\_eval <- rbind(mod\_eval, list("RF", Metrics::mae(test2$hpi, rf\_pred), Metrics::mse(test2$hpi, rf\_pred), Metrics::rmse(test2$hpi, rf\_pred)))  
  
mod\_eval <- rbind(mod\_eval, list("Tuned RF", Metrics::mae(test2$hpi, tuned\_rf\_pred), Metrics::mse(test2$hpi, tuned\_rf\_pred), Metrics::rmse(test2$hpi, tuned\_rf\_pred)))  
  
#Reset column names  
colnames(mod\_eval) <- c("Model", "MAE", "MSE", "RMSE")  
  
#Map data frame to flextable for display  
mod\_eval %>%  
 flextable()%>%  
 set\_caption(  
 caption = as\_paragraph(  
 as\_chunk("Table 8: Model Evaluation Metrics (On Test Set)",  
 props = fp\_text\_default(font.family = "Times New Roman",  
 font.size = 12,  
 italic = TRUE)  
 )))%>%  
 set\_table\_properties(layout = "autofit")%>%  
 bg(i = seq(2,6,2), bg = "light grey") %>%  
 font(part = "all", fontname = "Times New Roman") %>%  
 fontsize(size = 12) %>%  
 border(i = c(1:6), border.top = fp\_border\_default(color = "black"))

Table 8: Model Evaluation Metrics (On Test Set)

| Model | MAE | MSE | RMSE |
| --- | --- | --- | --- |
| MLR | 4.6676541 | 38.890874 | 6.236255 |
| AIC-Min MLR | 4.6901391 | 38.688064 | 6.219973 |
| Reg Tree | 7.7667367 | 90.448056 | 9.510418 |
| Pruned Reg Tree | 7.7667367 | 90.448056 | 9.510418 |
| RF | 0.8147173 | 1.905132 | 1.380265 |
| Tuned RF | 0.8147173 | 1.905132 | 1.380265 |

As seen above, the tuned random forest model was by far the best performing model. It outperforms the other models on all three metrics. This makes sense as the economy is a highly irregular system with many moving parts, not lending itself to just linear relationships between variables. Since our data was quite small, we did not run into any computational limitations when running our model.

These results tell us that it may be worth examining more non-linear or ensemble learning models in future examinations. When generalizing to the population, we can see that there are complex non-normal variables that require more complex models to analyze effectively.

## Post-Hoc Analysis

While analyzing economic factors and hpi, we wanted to find out hpi trends in similar economic situations. In order to achieve this, we decided to conduct unsupervised learning based on other variables except the response variable hpi. In this study, k-means and PAM (partitioning around medoids) were used as unsupervised learning methods.

After creating new data excluding hpi from the data used above, we went through a scaling process to obtain accurate weights.

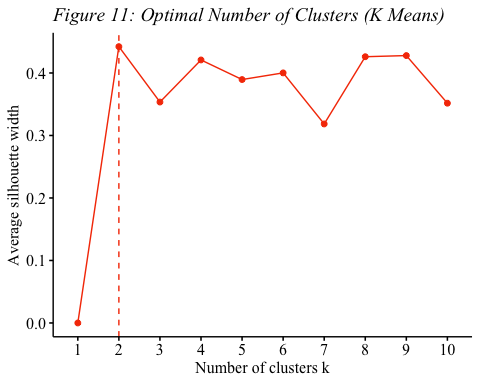
#--- Post Hoc Analysis: Clustering Techniques ---#  
  
#Load necessary libraries  
library(factoextra)

## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa

library(cluster)  
  
#Create data frame without reponse variable (hpi)  
data3<-data2 %>% dplyr::select(-hpi)  
  
#Scale data3  
data3\_scaled=scale(data3)

Then, we tried to find the appropriate number of clusters k, and as a result, k=2 was obtained using the silhouette score. It is important to note that there is no standardized way to select the number of clusters and our plots did not all agree. In this case, we chose to use the silhouette score plot for both of our models since it highlights the optimal number of clusters directly. *Figure 11* illustrates our silhouette score plot below:

#--- Post Hoc Analysis: Clustering Techniques ---#  
  
#Visualize silhouette score for k means -- optimal centers = 2  
fviz\_nbclust(data3\_scaled, kmeans, method = "silhouette", linecolor = "#F63E02") +  
 ggtitle("\*Figure 11: Optimal Number of Clusters (K Means)\*") +   
 theme(plot.title = ggtext::element\_markdown(),  
 text = element\_text(family = "Times New Roman", size = 12)) #2



#--- Post Hoc Analysis: Clustering Techniques ---#  
  
fviz\_nbclust(data3\_scaled, kmeans, method = "gap\_stat", linecolor = "#F63E02") #5  
fviz\_nbclust(data3\_scaled, kmeans, method = "wss", linecolor = "#F63E02") #No clear elbow

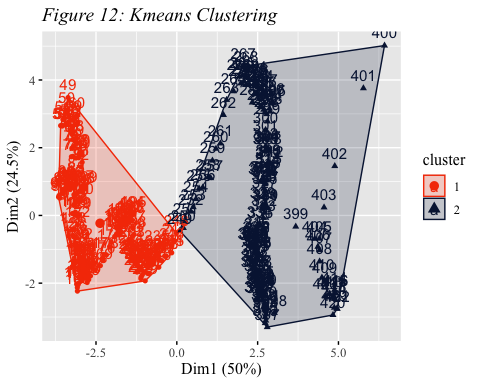
After deciding on the number of clusters, we ran our k means algorithm.

#--- Post Hoc Analysis: Clustering Techniques ---#  
  
#Run k means algorithm with 2 centers  
mod\_km <- kmeans(data3\_scaled,centers = 2)

#--- Post Hoc Analysis: Clustering Techniques ---#  
  
#Print k means results  
print(mod\_km)

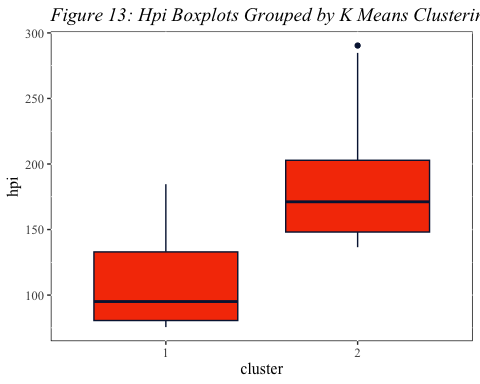
Finally, we visualized the k means clustering, as seen in *Figure 12* below:

#--- Post Hoc Analysis: Clustering Techniques ---#  
  
#Visualize k means clustering  
fviz\_cluster(mod\_km, data3\_scaled, stand = F, main = "\*Figure 12: Kmeans Clustering\*") +  
 scale\_color\_manual(values = c("#F63E02","#061A40")) +  
 scale\_fill\_manual(values = c("#F63E02","#061A40")) +   
 theme(plot.title = ggtext::element\_markdown(),  
 text = element\_text(family = "Times New Roman", size = 12))



We then plotted the box plots of the hpi distribution of the two different clusters, as shown in *Figure 13* below:

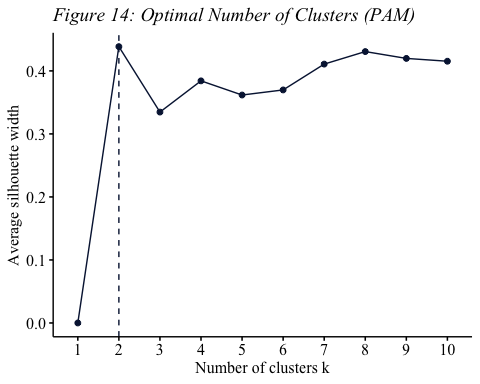
#--- Post Hoc Analysis: Clustering Techniques ---#  
  
#Add clusters to data2 and factorize to create new data frame  
data\_km<-cbind(data2, cluster=mod\_km$cluster)  
data\_km$cluster <-as.factor(data\_km$cluster)  
  
#Plot hpi boxplots separated by cluster  
ggplot(data\_km, aes(x=cluster,y=hpi))+  
 geom\_boxplot(fill = "#F63E02", col = "#061A40") +   
 labs(title="\*Figure 13: Hpi Boxplots Grouped by K Means Clustering\*") +   
 theme(panel.background = element\_rect(fill = "white", color = "black"),  
 plot.title = ggtext::element\_markdown(),  
 text = element\_text(family = "Times New Roman", size = 12))



Although there is some overlap, there is a significant jump in median hpi between the two clusters. This clustering may be most useful when looking at the upper ends of hpi values in each group as that is where there is the least overlap.

After conducting our k means clustering, we visualized the silhouette score for the PAM model, as shown by *Figure 14* below:

#--- Post Hoc Analysis: Clustering Techniques ---#  
  
#Visualize silhouette score to choose number of clusters  
fviz\_nbclust(data3\_scaled, pam, method = "silhouette", linecolor = "#061A40") +  
 ggtitle("\*Figure 14: Optimal Number of Clusters (PAM)\*") +  
 theme(plot.title = ggtext::element\_markdown(),  
 text = element\_text(family = "Times New Roman", size = 12))



Based on the silhouette scores, we decided to use 2 clusters for our PAM clustering model.

#--- Post Hoc Analysis: Clustering Analysis ---#  
fviz\_nbclust(data3\_scaled, pam, method = "gap\_stat", linecolor = "#061A40") #10  
fviz\_nbclust(data3\_scaled, pam, method = "wss", linecolor = "#061A40") #No clear elbow

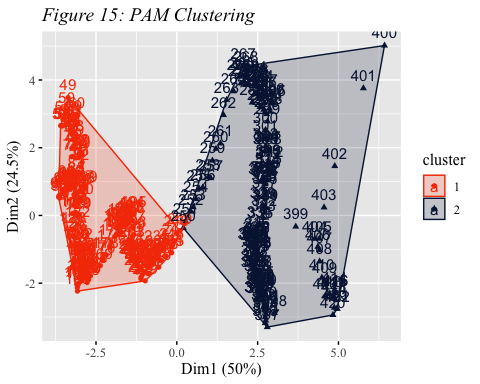
After deciding on the number of clusters, we fit our PAM clustering model.

#--- Post-Hoc Analysis: Clustering Techniques ---#  
  
#Run PAM algorithm -- how did you decide on 5??  
mod\_pam <- pam(data3\_scaled, stand = T, metric="manhattan",k=2)

#--- Post Hoc Analysis: Clustering Techniques ---#  
  
#Print PAM results  
print(mod\_pam)

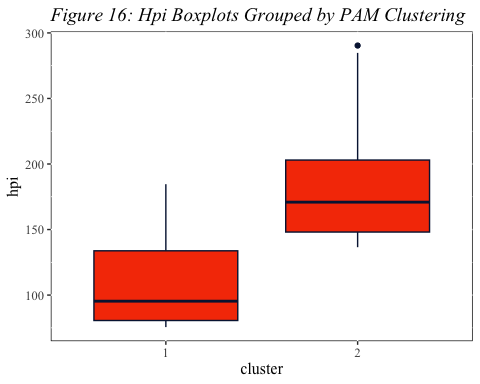
We then visualized the PAM clustering, as shown by *Figure 15* below:

#--- Post Hoc Analysis: Clustering Techniques ---#  
  
#Visualize PAM clustering  
fviz\_cluster(mod\_pam, main = "\*Figure 15: PAM Clustering\*") +   
 scale\_color\_manual(values = c("#F63E02","#061A40")) +  
 scale\_fill\_manual(values = c("#F63E02","#061A40")) +   
 theme(plot.title = ggtext::element\_markdown(),  
 text = element\_text(family = "Times New Roman", size = 12))



Finally, we plotted the box plot of hpi distribution for each of the two PAM clusters side-by-side, as shown by *Figure 16* below:

#--- Post Hoc Analysis: Clustering Techniques ---#  
  
#Add clusters to data2 and factorize to create new data frame  
data\_pam<-cbind(data2, cluster=mod\_pam$clustering)  
data\_pam$cluster <-as.factor(data\_pam$cluster)  
  
#Plot hpi boxplots separated by cluster  
ggplot(data\_pam, aes(x=cluster,y=hpi)) +   
 geom\_boxplot(fill = "#F63E02", col = "#061A40") +   
 labs(title="\*Figure 16: Hpi Boxplots Grouped by PAM Clustering\*") +   
 theme(panel.background = element\_rect(fill = "white", color = "black"),  
 plot.title = ggtext::element\_markdown(),  
 text = element\_text(family = "Times New Roman", size = 12))



Similar to k means, there is some overlap with a significant jump in median hpi between the two clusters. This clustering may be most useful when looking at the upper ends of hpi values in each group as that is where there is the least overlap.

#--- Post Hoc Analysis: Clustering Techniques ---#  
#Cacluate how many observations were classified differently by the two algorithms  
sum(mod\_pam$clustering != mod\_km$cluster)

## [1] 1

Meanwhile, when comparing the clustering results with k-means and pam, only 1 observation were found to be different. Analysis was performed with k=2 in k-means and pam, and as can be seen from the graph, the shape and order of the two clusters were quite similar between the two methods. In the future, we could look to rejoin the clustered data with their associated dates to allow further economic analysis.

# Conclusion

With the goal of better understanding the U.S. economy, our research focused on modeling the relationship between various economic indicators and the housing price index. Using housing and monthly macroeconomic factors data, we tried to determine which variables among different economic factors such as GDP, cpi (consumer price index), and employment rate would impact hpi (housing prices index) the most. Moreover, by attempting various modeling techniques, we tried to find a model that could well explain hpi with various economic factors.

To get a better understanding about our data, we conducted various exploratory analyses. We summarized the data and looked at the distributions of each variable by visualizing density functions and box plots. In addition to this, by creating a correlation matrix and scatter plots, we could learn about the overall relationship and correlations between hpi and other variables. Specifically, we could figure out that population, mortgage rate, producer price index, total US money supply, stock price index, consumer price index, and real disposable index are the variables that have strong correlation with hpi.

After exploring the data, we created several models to understand variable importance and to predict hpi. We created a multiple linear regression, bootstrapped regression, regression tree, and random forest. For each predictive model, we examined variable importance and calculated its mean absolute error, mean squared error, and root mean squared error. As a result of each modeling, various combinations of variables affecting hpi were found. Among them, m3 was commonly included in the results of all modeling conducted, and in addition, a combination of various economic factors, including cpi and population, influenced hpi. Furthermore, in terms of rmse, we concluded that random forest had the smallest rmse among the modeling methods implemented, making it the most appropriate modeling technique for our study. Finally, as a post-hoc analysis, we generated a k means and PAM clustering analysis. Through these unsupervised learning results, it was possible to cluster economic situations, and it was confirmed that hpi was distributed significantly differently depending on these clusters. This supports our hypothesis that economic factors have a significant impact on hpi.

Contrary to the existing expectation that important variables in predicting hpi could be clearly identified through modeling, results were obtained in which important variables appeared differently depending on the model. The reason why this phenomenon occurs can be thought of based on the characteristics of the model. While multiple regression analysis detects linear relationships between variables, regression trees or random forests can capture non-linear relationships. Additionally, in the case of bootstrap regression, the importance of predictor variables may vary depending on the bootstrapped sample. In addition, this phenomenon may be attributed to the characteristics of the data, such as the fact that the data we used in our analysis included several economic indicators.

Our research on analyzing the dynamics of the US housing prices can serve as a stepping stone to better understand the US real estate market and, by extension, the economy. By analyzing and predicting house prices through various economic indicators, various effects can be expected. Specifically, it will help people develop their investment strategy and make financial decisions, and anticipated fluctuations in housing prices can impact decisions related to home purchases, sales, and rentals. Housing price predictions can also contribute to evaluating the health of the real estate market, enabling governments and central banks to formulate policies to maintain stability in the housing and real estate market.

## Limitations

The real estate-related data we used in our analysis was time series data representing monthly indicators. Since the dependent variable, hpi, is likely affected by the passage of time, it would have been useful to apply time series modelling to the data. This was, however, outside of our scope.

Furthermore, in the original data sets, there was a variable ‘hcai’ that we ignored because of the amount of NAs. Hcai stands for ‘housing credit availability index’, and is a metric used in the United States to assess the ease of obtaining mortgage credit. As it represents credit availability in the housing market, hcai likely has a direct impact on demand in the housing and market dynamics. In other words, because hcai was a variable that had the potential to play an important role in the movement of hpi, different results may have been obtained if the study had been conducted including this variable.

In general, we a very low volume of data to work with. Having more observations, whether that was via daily observations or a larger time window of consideration would have likely made our models much more robust.

## Suggestions for Future Research

For this research project, we considered all of our variables as numeric variables. With numeric variables, we could apply few analysis methods such as regression trees, random forest, and multiple linear regression. In future research, changing the class of some variables  to categorical class can open up new possibilities for analysis and modeling. This can provide us with completely new results compared to the original method. In addition, we can evaluate the effectiveness of various government policies, such as monetary policies or fiscal policy, in influencing hpi. This research question can be looked at from a global and regional variances where one can investigate whether the influence of certain factors varies across different geographical locations and economic contexts. Lastly, one can integrate the role of technology and its impact on the real estate market and how that affects the hpi variable. For instance, analyzing the role of technology, like real estate technology platforms or smart home features, in shaping the relationship between macroeconomic factors and hpi.

# Works Cited

Liberto, D. (2023, August 29). Understanding the House price index (HPI) and how it is used. Understanding the House Price Index (HPI) and How It Is Used. <https://www.investopedia.com/terms/h/house-price-index-hpi.asp>

Rosen, P. (2023, August 11). The US housing market hits a record value of $47 trillion as the inventory shortage fuels a price boom. Business Insider. <https://markets.businessinsider.com/news/commodities/housing-market-inventory-shortage-home-prices-value-real-estate-property-2023-8.>