Research on Increased Housing Prices

Michael Pitts

8/17/2023

Research question and Research Goals:

The goal of this report is to identify and possibly rank contributing factors to large national increases in median home prices. Additionally policy suggestions will be made in accordance to this finding in the conclusion section.

Step 1: Import data

Response Variable:

- Quarter over Quarter Change in Median home price.

Predictor Variables:

- Inflation.
- Average 30 year mortgage rate.
- Median Household Income.

Each data frame is to be organize such that each row is correlated with a quarter and a year. To do this I will be using an unorthodox date format in my date column. The date format I will be using will take the form: Year Quarter, for example: 1963 Q2.

```
# National inflation
# Start: Jan 1914, End: Dec 2022, Monthly.
# https://www.usinflationcalculator.com/inflation/historical-inflation
```

```
  n_infl<- read_csv('US_inflation.csv', show_col_types = FALSE)

  # Median house Price
  # Start: Jan 1963, End: Apr 2023, Quarterly.
  # https://fred.stlouisfed.org/series/MSPUS
  # TODO: Convert into quarter over quarter changes.
  mhp <- read_csv('median_home_price.csv', show_col_types = FALSE)

  # Average 30 year mortgage rate
  # Start: Apr 1971, End: Aug 2023. Quarterly.
  # https://fred.stlouisfed.org/series/MORTGAGE30US
  mort_30 <-  read_csv('mortgage_30.csv', show_col_types = FALSE)

  # Median household income
  # Start 1984, End: 2021, Yearly
  # https://fred.stlouisfed.org/series/MEHOINUSA672N
  # TODO: Want to convert to year over year increase,
  mhi <- read_csv("median_houshold_income.csv", show_col_types = FALSE)
```

Given the data above we can see that the larges time frame we can use is from the year 1984 to 2021. Because these years will come up again in the data wrangling portion of this research I will create a vector containing all these years. This time frame will be denoted as the observation window, and will be stored in the yrs vector.

```
yrs <- seq(1985, 2021, 1)
```

Step 2: Data Wrangling

Many of the data sets selected for analysis are structured in way such that they incompatible with each other in their original state. In order to make these data set compatible with each other they will have to be transformed to a standard format to which they will all comply. Additionally the target variable's data set does not measure quarter over quarter changes, but instead tracts the quarterly median house prices as it stands for that quarter. This data set will have to be transform to comply to the formatting standard of this analysis and additional statistics will have to be calculated to achieve the goal of this study.

National Inflation Data:

National inflation data is tricky because of the way it is structured. Each row represents a year and each column represents a month, with the exception of some of the tailing columns which are used to store some summary statistics for a given year. Because our target data format is so different from this original data format the transformation will have to be done in a 2-part process and be stored in a new tibble, this tibble will be called: national_inflation.

```r
head(n_infl)
```

```
## # A tibble: 6 x 14
##    Year   Jan   Feb   Mar   Apr   May   Jun   Jul   Aug   Sep   Oct   I
##   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dl
## 1  1914   2     1     1     0     2.1   1     1     3     2     1
## 2  1915   1     1     0     2     2     2     1    -1    -1     1
## 3  1916   3     4     6.1   6     5.9   6.9   6.9   7.9   9.9  10.8  1
## 4  1917  12.5  15.4  14.3  18.9  19.6  20.4  18.5  19.3  19.8  19.5  17
## 5  1918  19.7  17.5  16.7  12.7  13.3  13.1  18    18.5  18    18.5  20
## 6  1919  17.9  14.9  17.1  17.6  16.6  15    15.2  14.9  13.4  13.1  13
## # i 1 more variable: Average <dbl>
```

Firstly the quarterly statistics that we want will be appended to the original data. The quarterly statistics are calculated by taking the mean of the target months, for example if Jan = 2, Feb = 6, and Mar = 5, then Q1 = (2 + 6 + 5)/3 = 4.333. Once the statistics are calculated the original monthly measurements can be discarded.

```r
# Create columns for quarters using average over 3 months.
n_infl["Q1"] = rowMeans(n_infl[2:4])
n_infl["Q2"] = rowMeans(n_infl[5:7])
n_infl["Q3"] = rowMeans(n_infl[8:10])
n_infl["Q4"] = rowMeans(n_infl[11:13])
```

```r
# Select just quarterly columns and year and start at 1984.
n_infl <- n_infl %>%
  filter(Year >= 1984) %>%
  dplyr::select(Year, Q1, Q2, Q3, Q4)

head(n_infl)
```

```
## # A tibble: 6 x 5
##     Year    Q1    Q2    Q3    Q4
##    <dbl> <dbl> <dbl> <dbl> <dbl>
## 1  1984  4.53  4.33  4.27  4.1
## 2  1985  3.57  3.77  3.33  3.5
## 3  1986  3.1   1.63  1.67  1.3
## 4  1987  2.2   3.8   4.2   4.47
## 5  1988  3.93  3.93  4.1   4.27
## 6  1989  4.83  5.23  4.67  4.6
```

At this point we have the data we want, now we just need to arrange it in the format we want. To do this anew tibble is created and the existing data is reorganized into the new tibble.

```r
# New inflation data tibble.
national_inflation <- tibble(date = "0 Q0", inflation = 123456)

# For every target year create row for the quarter and year.
for (yr in yrs) {
  row <- n_infl %>% filter(Year == yr)
  national_inflation <- national_inflation %>%
    add_row(date = paste(as.character(yr), "Q1"),
            inflation = unlist(row[2]) ) %>%
    add_row(date = paste(as.character(yr), "Q2"),
            inflation = unlist(row[3]) ) %>%
    add_row(date = paste(as.character(yr), "Q3"),
            inflation = unlist(row[4]) ) %>%
    add_row(date = paste(as.character(yr), "Q4"),
            inflation = unlist(row[5]) )
}

# Filter out dummy row.
national_inflation <- national_inflation %>%
  filter(inflation != 123456)

head(national_inflation)
```

```
## # A tibble: 6 x 2
##   date    inflation
##   <chr>       <dbl>
## 1 1985 Q1      3.57
## 2 1985 Q2      3.77
## 3 1985 Q3      3.33
## 4 1985 Q4      3.5
## 5 1986 Q1      3.1
## 6 1986 Q2      1.63
```

Median Home Price

```
head(mhp)
```

```
## # A tibble: 6 x 2
##   DATE       MSPUS
##   <date>     <dbl>
## 1 1963-01-01 17800
## 2 1963-04-01 18000
## 3 1963-07-01 17900
## 4 1963-10-01 18500
## 5 1964-01-01 18500
## 6 1964-04-01 18900
```

Because the structure of the median home price data is so similar to the format we want we won't have to create a new tibble for it. Instead I will add a new date column with the correct format, rename the date format to something more readable, and remove the old date format. Lastly I will convert the data from telling us the median home price every quarter to the change in median home price from the previous quarter by percent. For example if the previous median home price was 320,000 and this quarter it is 330,000 then our measured quarterly increase would be (330,000-320,000)/320,000 * 100 = 3.125. I will also keep the raw data for the median home price incase it proves a better source of analysis.

Firstly I will rename the rate home price to something more intuitive, I like to run this in a separate code block because it permanently changes the structure of the data and can make debugging problematic code blocks harder.

```
mhp <- mhp %>% rename(median_home_price = MSPUS)
```

```
Q1 <- c("01","02","03")
Q2 <- c("04","05","06")
Q3 <- c("07","08","09")
Q4 <- c("10","11","12")

mhp <- mhp %>%
  mutate(
    date = case_when(
      substr(DATE,6, 7) %in% Q1 ~ paste(substr(DATE,1, 4), "Q1"),
      substr(DATE,6, 7) %in% Q2 ~ paste(substr(DATE,1, 4), "Q2"),
      substr(DATE,6, 7) %in% Q3 ~ paste(substr(DATE,1, 4), "Q3"),
      substr(DATE,6, 7) %in% Q4 ~ paste(substr(DATE,1, 4), "Q4"),
      TRUE ~ "ERROR"
    )
  ) %>%
  mutate(
    median_home_price_change =
      (median_home_price - lag(median_home_price))
      / lag(median_home_price)*100
  ) %>%
  filter(as.numeric(substr(date,1, 4)) %in% yrs) %>%
    dplyr::select(date, median_home_price_change, median_home_price)
```

```
head(mhp)
```

```
## # A tibble: 6 x 3
##   date    median_home_price_change median_home_price
##   <chr>                      <dbl>             <dbl>
## 1 1985 Q1                     3.63             82800
## 2 1985 Q2                     1.81             84300
## 3 1985 Q3                    -1.30             83200
## 4 1985 Q4                     4.33             86800
## 5 1986 Q1                     1.38             88000
## 6 1986 Q2                     4.66             92100
```

Average 30 Year Mortgage rate

Like the median home price data, the average 30 year mortgage rate date is already in a quarterly format. Unlike the median home price data, the 30 year mortgage rate data does not need to be transformed into quarter of quarter changes. Therefor all that needs to be done to the data is to change the formatting of the date column, it is currently stored as yyyy-mm-dd, after the translation is will be yyyy Q#.

Firstly I will rename the rate column to something more intuitive, I like to run this in a separate code block because it permanently changes the structure of the data and can make debugging problematic code blocks harder.

```
mort_30 <- mort_30 %>% rename(mort_rate = MORTGAGE30US)
```

```
mort_30 <- mort_30 %>%
  filter(as.numeric(paste(substr(DATE,1, 4))) %in% yrs) %>%
  mutate(
    date = case_when(
      substr(DATE,6, 7) %in% Q1 ~ paste(substr(DATE,1, 4), "Q1"),
      substr(DATE,6, 7) %in% Q2 ~ paste(substr(DATE,1, 4), "Q2"),
      substr(DATE,6, 7) %in% Q3 ~ paste(substr(DATE,1, 4), "Q3"),
      substr(DATE,6, 7) %in% Q4 ~ paste(substr(DATE,1, 4), "Q4"),
   TRUE ~ "ERROR"
  ),
  mort_rate = as.double(mort_rate)
 ) %>%
 dplyr::select(date, mort_rate)


head(mort_30)
```

```
## # A tibble: 6 x 2
##   date    mort_rate
##   <chr>       <dbl>
## 1 1985 Q1      13.1
## 2 1985 Q2      12.8
## 3 1985 Q3      12.1
## 4 1985 Q4      11.7
## 5 1986 Q1      10.6
## 6 1986 Q2      10.2
```

Median Household Income

Median household income data is structured identically to the median home price data. Because that it should be processed similarly to the way the median home price data was processed. Additionally median household income should be measured through percent increase like new_house_sale_change and median house price. Lastly the median household income data was collected year, thus for each quarter of the year the quarterly change will be constant. The year over year change will be divided by 4 and used as the quarter over quarter change. Because of this lack of quarterly or monthly data the assumption being made is, median household income increased changes linearly over the course of a year.

```
median_household_change <- tibble(date = "0 Q0",
                                  median_income_change = 12345,
                                  median_income = 12345)
```

```r
mhi <- mhi %>%
  mutate(
    median_household =
      (MEHOINUSA672N - lag(MEHOINUSA672N))/lag(MEHOINUSA672N) * 100
  )

# For every target year create row for the quarter and year.
for (yr in yrs) {
  row <- mhi %>% filter(substr(DATE, 1,4) == yr)
  median_household_change <- median_household_change %>%
    add_row(date = paste(as.character(yr), "Q1"),
            median_income_change  = unlist(row[3])/4,
            median_income = unlist(row[2])) %>%
    add_row(date = paste(as.character(yr), "Q2"),
            median_income_change  = unlist(row[3])/4,
            median_income = unlist(row[2]) ) %>%
    add_row(date = paste(as.character(yr), "Q3"),
            median_income_change  = unlist(row[3])/4,
            median_income = unlist(row[2]) ) %>%
    add_row(date = paste(as.character(yr), "Q4"),
            median_income_change  = unlist(row[3])/4,
            median_income = unlist(row[2]) )
}

median_household_change <- median_household_change %>%
  filter(median_income_change != 12345)


head(median_household_change)
```

```
## # A tibble: 6 x 3
##   date    median_income_change median_income
##   <chr>                  <dbl>         <dbl>
## 1 1985 Q1                0.467         56871
## 2 1985 Q2                0.467         56871
## 3 1985 Q3                0.467         56871
## 4 1985 Q4                0.467         56871
## 5 1986 Q1                0.901         58920
## 6 1986 Q2                0.901         58920
```

Joining Data Sets

In order to perform proper analysis on the data gather the data sets we cleaned must be joined. Because the data was properly cleaned and formatted uniformly this step should be straight forward. The data sets will be right joined such that the data column will remain on the far right-side of the output tibble. The results of this joining will be stored in the tibble: df, it will become the primary source of analysis for the remained of the report.

Before joining the working data sets will be listed:

- *mhp*
- *national_inflation*
- *mort*_30
- *median_household_change*

```
df <- inner_join(mhp, national_inflation)

df <- inner_join(df, mort_30)

df <- inner_join(df, median_household_change)
```

The data sets have been joined and analysis will be solely conducted from the data set *df* which contains all the predictors, and the response variable along with our key column date.

```
head(df)
```

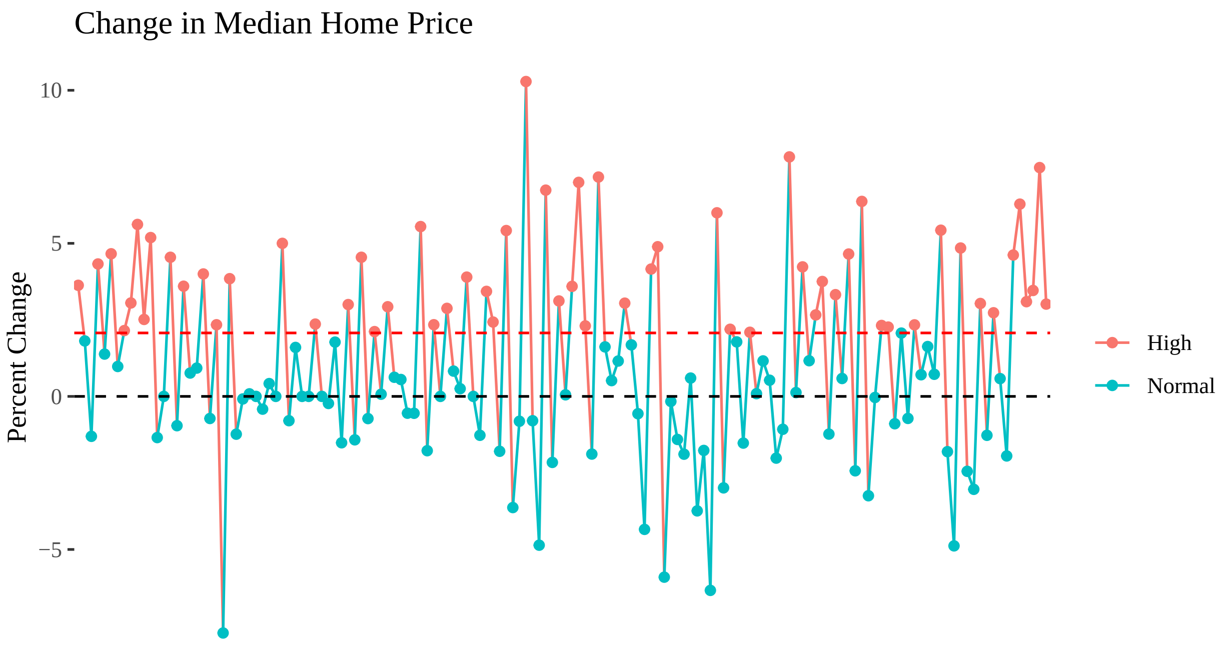
```
## # A tibble: 6 x 7
##   date    median_home_price_change median_home_price inflation mort_ra
##   <chr>                      <dbl>             <dbl>     <dbl>     <db
## 1 1985 Q1                     3.63             82800      3.57       13
## 2 1985 Q2                     1.81             84300      3.77      12
## 3 1985 Q3                    -1.30             83200      3.33      12
## 4 1985 Q4                     4.33             86800      3.5       11
## 5 1986 Q1                     1.38             88000      3.1       10
## 6 1986 Q2                     4.66             92100      1.63      10
## # i 2 more variables: median_income_change <dbl>, median_income <dbl>
```

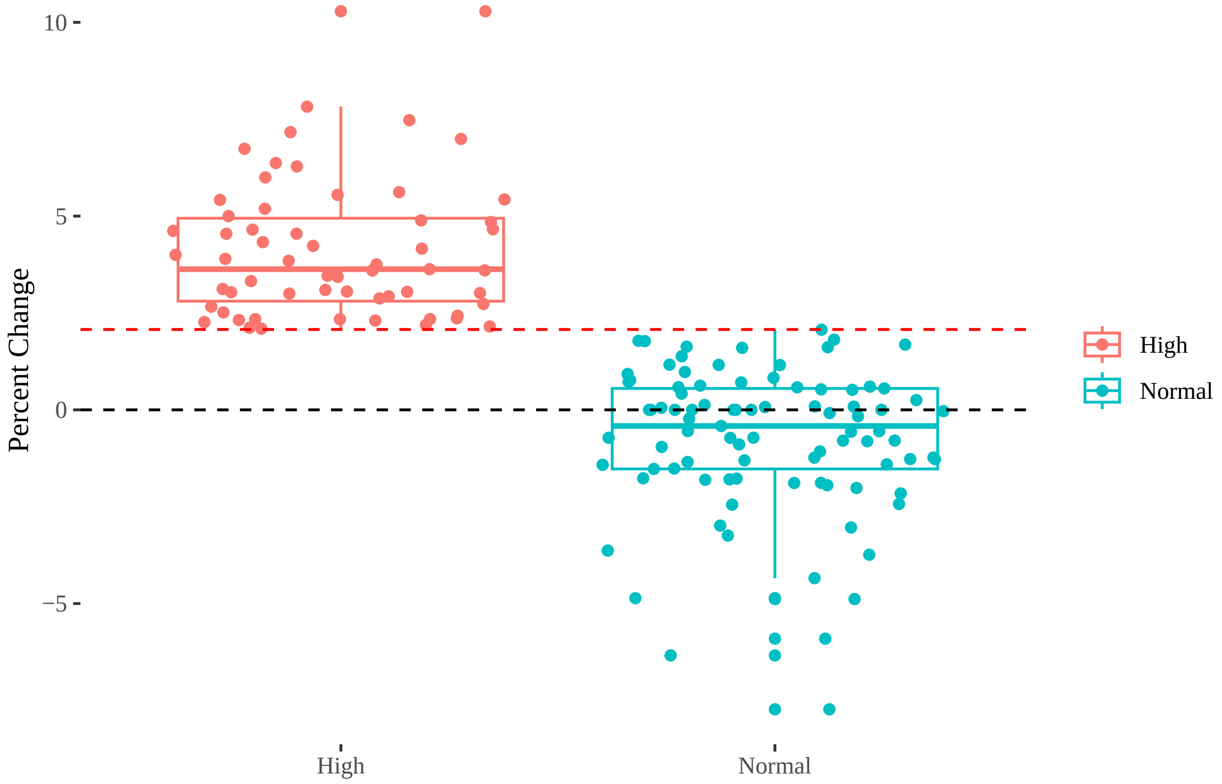
Step 3: Exploritory Data Analysis

Firstly lets look at graph of change in median home price over time with a black line to indicate at 0 percent to indicate no growth and a red line at 2.07 which is the 6th percentile mark for home price change for our given time window. All growth above that indicates that home price changes are notably high. Points with growth greater than or equal to that will be marked with red. Under that graph there will be a series box plots that sort the points into 2 groups, those who fall in the average quarterly home price growth, and who do not. The group names respectively are: High, Normal.

```r
# High Change threshold
# Threshold will be set to 60th percentile.

thresh <- quantile(df$median_home_price_change, probs = c(0.6))
```

```r
#Create factor column that classifies the data based on
# median_home_price_change.
df <- df %>%
  mutate(
    home_price_class = case_when(
      median_home_price_change >= thresh~ "High",
      TRUE ~ "Normal"

    )
  )
```





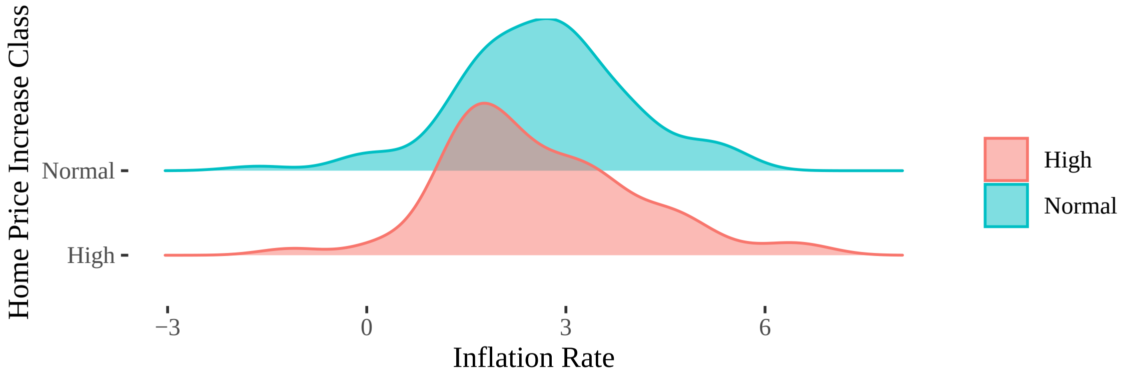
From these graph it can be seen on the box plot that the median value for normal quarter over quarter home value growth is close to zero. This is interesting because it could suggest that a majority of the home price growth made over the past 36 years is due to these high quarterly increases rather than an accumulation of “Normal” increases.

Ploting Relationships

One Dimensional Analysis

Because the goal of this report is to determine which of the predictor variables are most effective in classifying whether or not a quarter will produce a “High” home price change. It would be advantages to look at how these relationships look on paper. This will allows speed up the process of modeling building and selection because not every relationship will have to have a corresponding model built and validated.

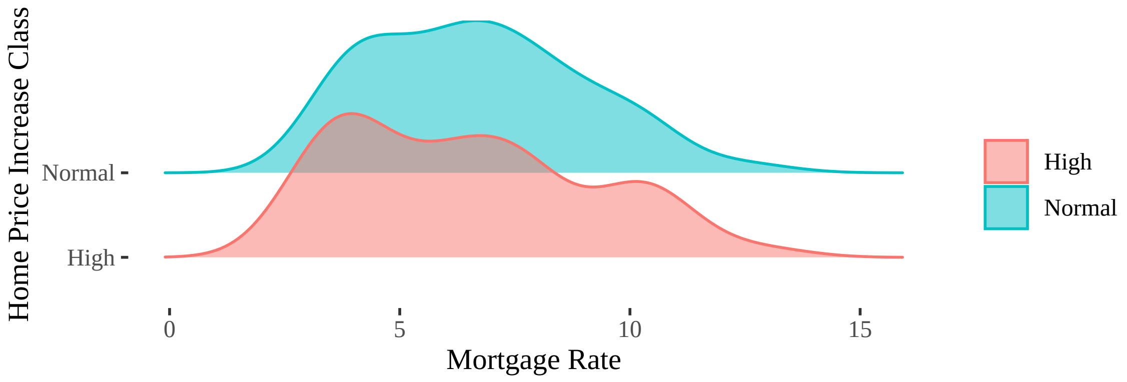
Inflation



Comments:

The High and Normal Ridges are closely related, the only noteworthy difference is that the High group is sharper and slightly more skewed towards lower inflation.

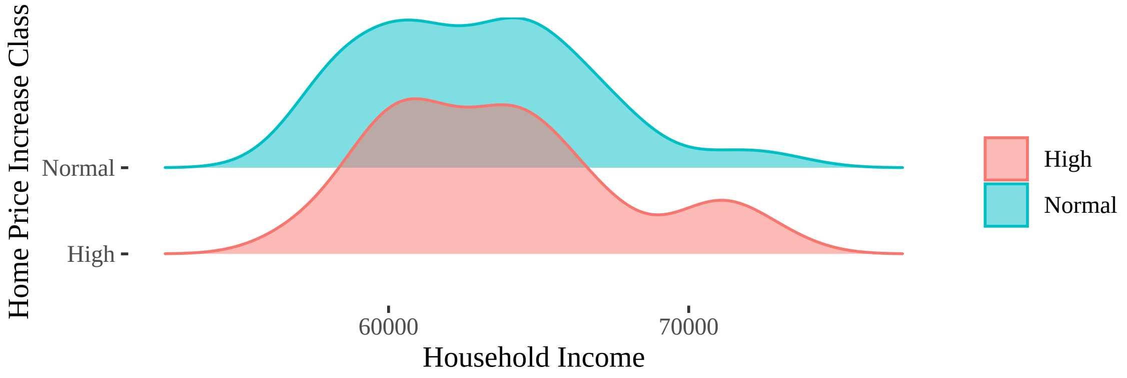
30 Year Mortgage Rate



Comments:

Again the High and Normal Ridges are closely related, But notably there is a large subgroup of High quarters centered around the 10 percent mortgage rate. This is interesting because I would normally expect High quarters to tend to lower mortgage rates because increased consumer power.

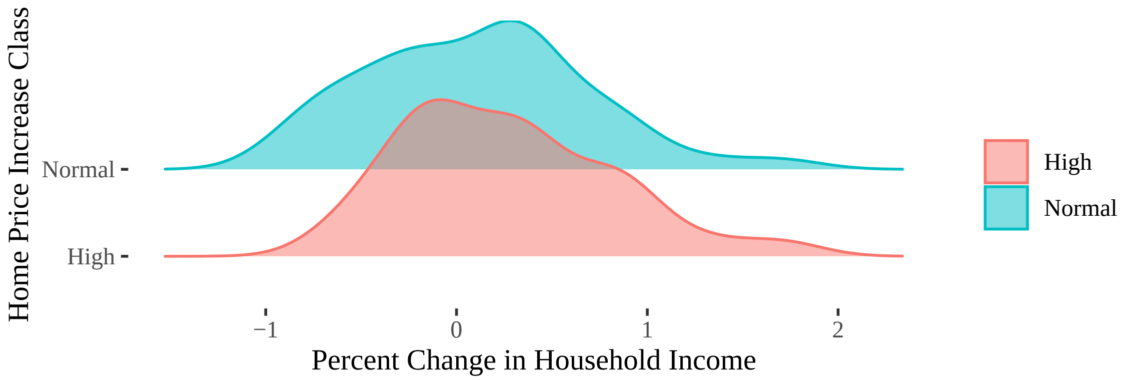
Median Household Income



Comments:

Once again a bimodal distribution appears in the High group ridge around the mean. Additionally there is a noticeable spike when household income reaches 70000.

Change in Median Household Income



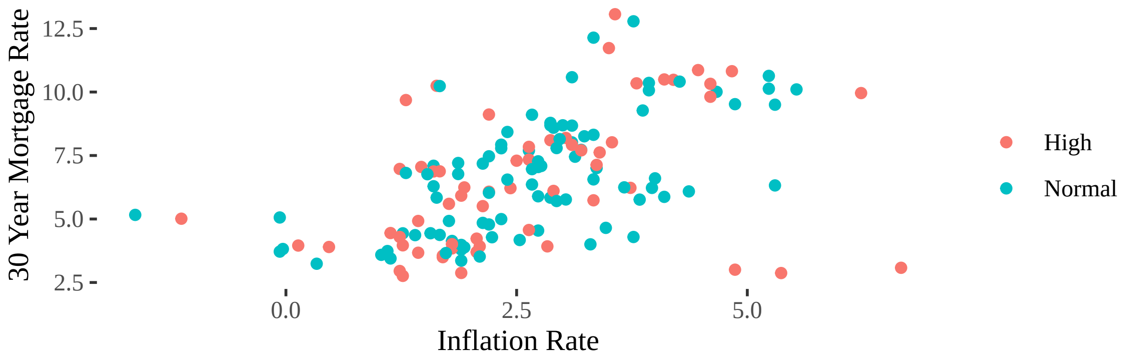
Comments:

The most notable aspect of this graph is how the High group clearly favors a slight decrease in household income where as the Normal group favors a slight increase in household income. Also the variance the High group is much smaller which could signify that large quarterly increase are partially motivated by economic stability.

Two Dimensional Analysis

No predictor variable has a distinct pattern with respect to the high quarterly increase in home prices. This could infer that none of these factors, within the time frame observed, have any notable influence on large quarter over quarter increases home prices. Because no notable relationship can be seen in one dimensional data, the analysis must be bumped up to two dimensions. There are 5 unique pairs of predictor variables that will be plotted.

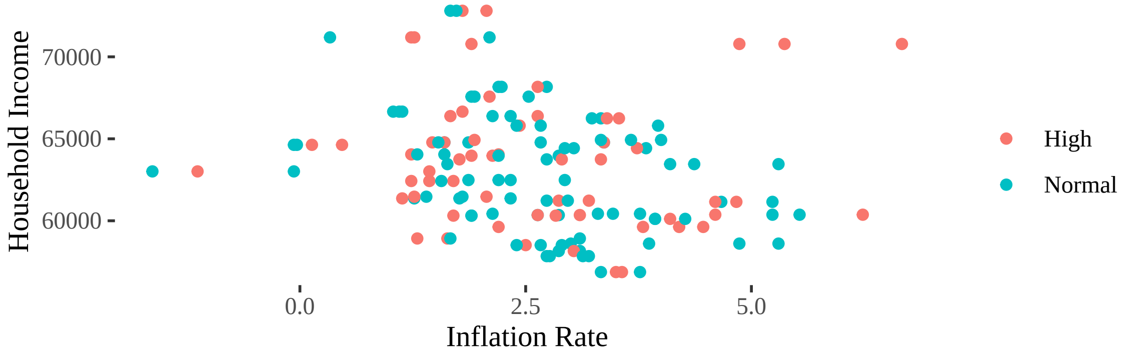
Inflation vs 30 Year Mortgage



Comments:

More sparse towards 5 percent inflation and more dense around 2.5 percent inflation. Minimal clustering for High class, enough for further analysis.

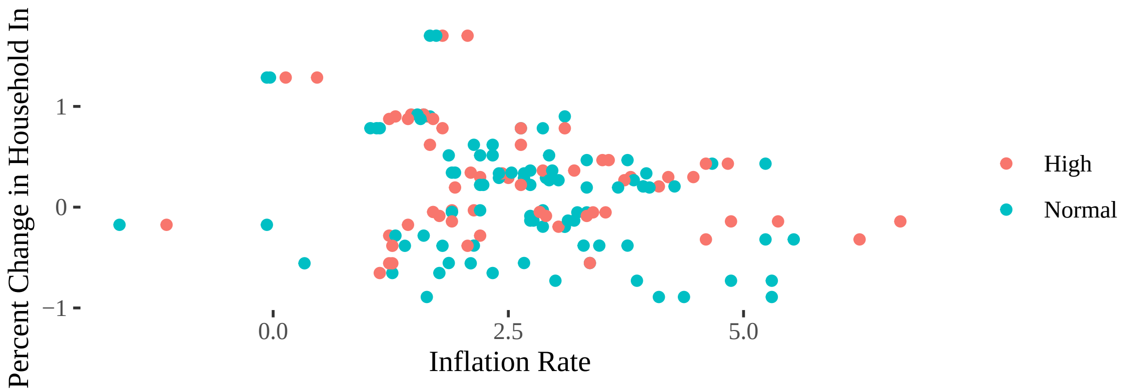
Inflation vs Household Income



Comments:

No notable clustering, will not proceed with analysis.

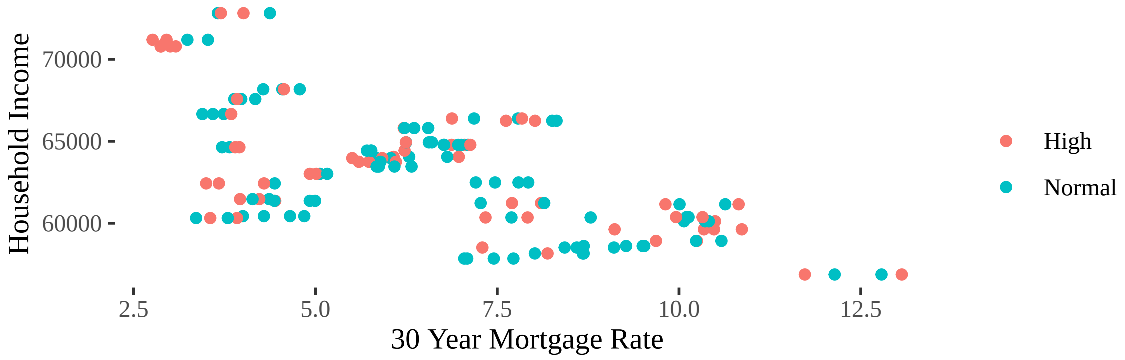
Inflation vs Change Household Income



Comments:

Light clustering around [-.5,1] percent change area, will proceed with analysis.

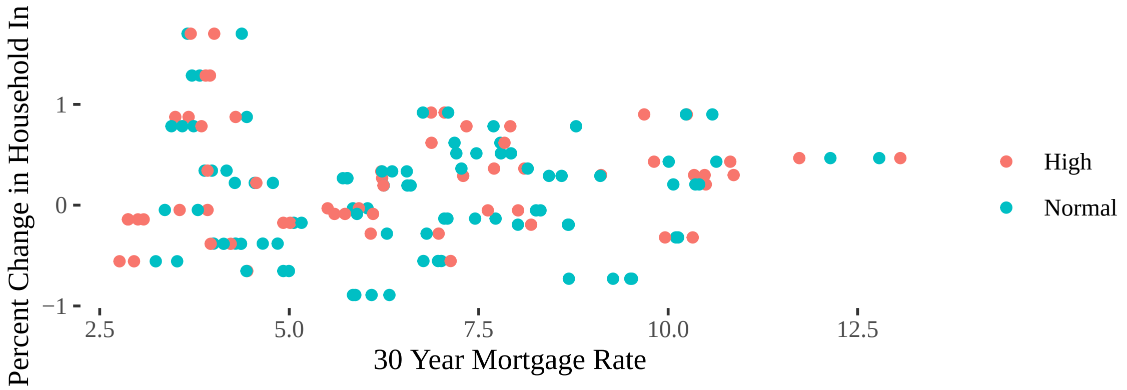
30 Year Mortgage Rate vs Household Income



Comments:

No notable clustering, will not proceed with analysis.

30 Year Mortgage Rate vs Change in Household Income



Comments:

Light clustering around throughout, will proceed with analysis.

Part 4: Model Building

Selected relationships that will be used in model building:

- 30 Year Mortgage Rate vs Change in Household Income
- Inflation vs 30 Year Mortgage
- Inflation vs Change in Household Income

Additionally models will be built utilizing all three predictors to determine how all the predictor work together

Classification Through kNN, Random Forest, and Classification Trees.

Firstly k-nearest neighbor models will be built with every selected relationship. Each model will tuned and it's accuracy evaded. The accuracy will be evaluated through k-fold validation. kNN is being used because it has a flexible decision boundary which will allow it to be fit to patterns that are harder to see.

Create Training set

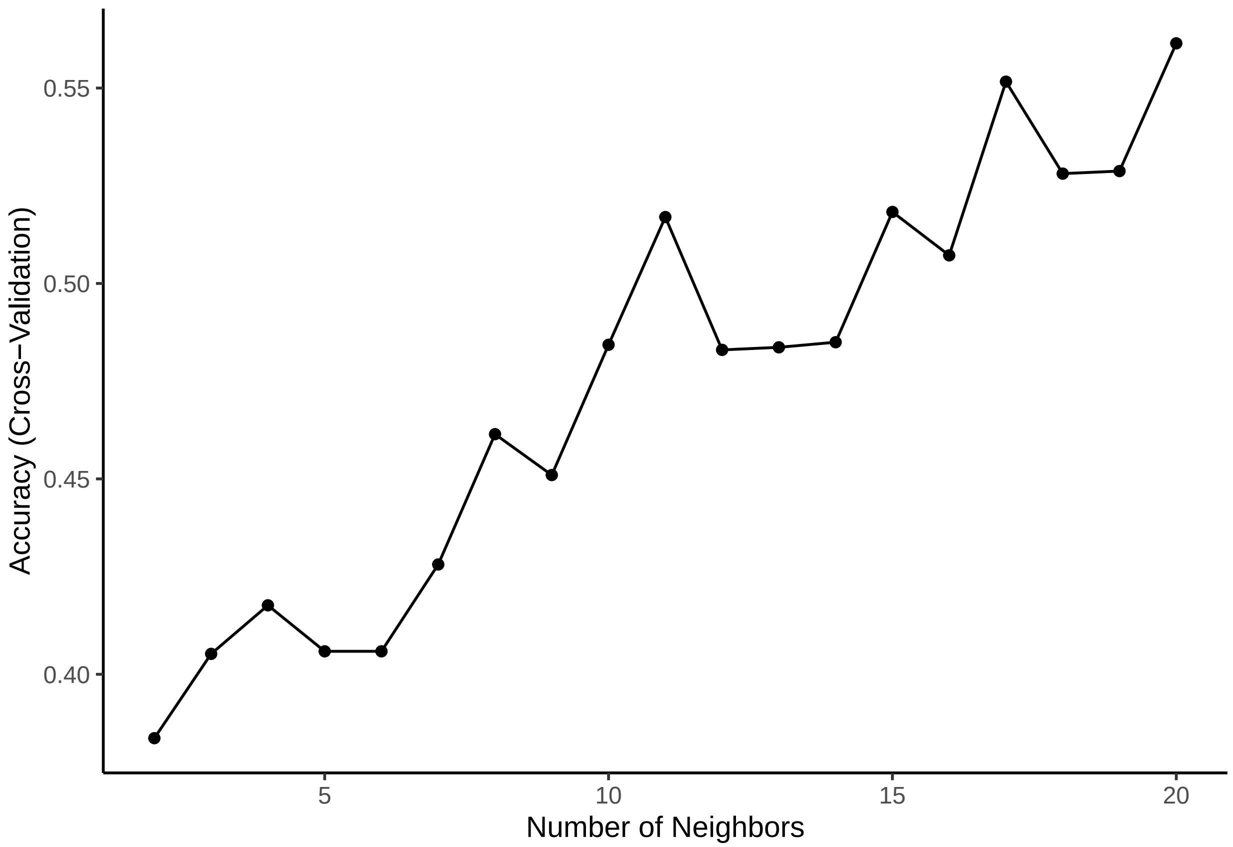
```r
# Number of entries in the df.
num_entries <- length(df$date)

# Number of testing entries rounded to nearest integer.
num_test = round(num_entries*.4, 0)

# Sample 20 percent of entries from data set.
test_indexs = sample(1:148, num_test, replace = FALSE)

# Use 20 samples to create test set.
test_set = df[test_indexs,]

# Use remaining entries to create training set.
train_set = df[-test_indexs,]
```

kNN Models:

30 Year Mortgage Rate vs Change in Household Income

```r
# Create training set the works with kNN.
tr_data <- train_set %>%
  dplyr::select(mort_rate, median_income, home_price_class) %>%
  rename(y = mort_rate,
         x = median_income)
# Train kNN
train_knn_1 <- train(home_price_class~.,
                   method="knn",
                   data = tr_data,
                   tuneGrid = data.frame(k=2:20),
                   trControl = trainControl(method="cv", number = 5))
```

```r
# Plot Accuracy vs number of neighbors
ggplot(train_knn_1)+
  theme_classic()+
  xlab("Number of Neighbors")
```



```r
train_knn_1$bestTune$k
```

```
## [1] 20
```

```r
# Create testing set the works with kNN.
tt_data <- test_set %>%
  dplyr::select(mort_rate, median_income_change, home_price_class) %>%
  rename(y = mort_rate,
         x = median_income_change)

# Produce confusion matrix for analysis
confusionMatrix(
  predict(train_knn_1, tt_data, type="raw"),
  as.factor(test_set$home_price_class))
```

```
## Confusion Matrix and Statistics
##
```

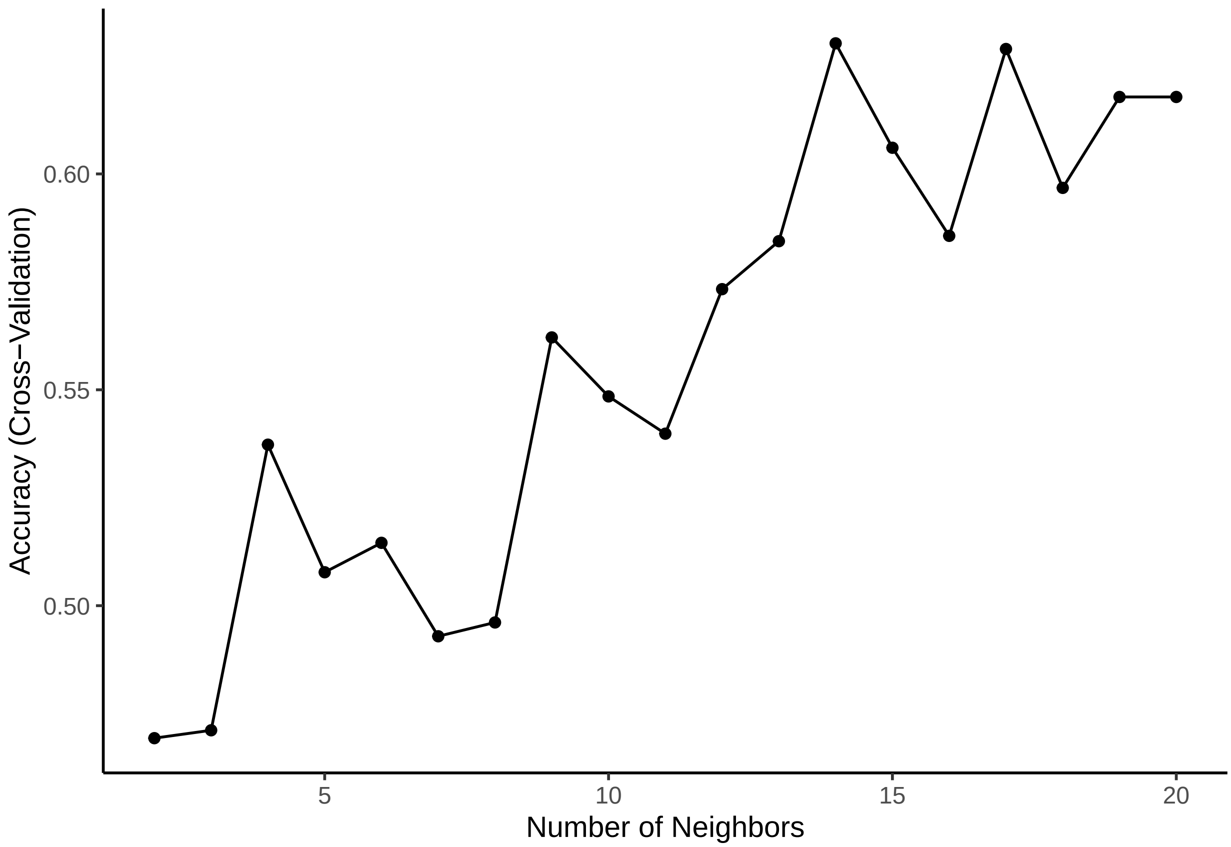
```
##           Reference
## Prediction High Normal
##     High      0      0
##     Normal   22     37
##
##                Accuracy : 0.6271
##                  95% CI : (0.4915, 0.7496)
##     No Information Rate : 0.6271
##     P-Value [Acc > NIR] : 0.5579
##
##                   Kappa : 0
##
##  Mcnemar’s Test P-Value : 7.562e-06
##
##             Sensitivity : 0.0000
##             Specificity : 1.0000
##          Pos Pred Value :    NaN
##          Neg Pred Value : 0.6271
##              Prevalence : 0.3729
##          Detection Rate : 0.0000
##    Detection Prevalence : 0.0000
##       Balanced Accuracy : 0.5000
##
##        ’Positive’ Class : High
##
```

The result above indicate that the model attempted to fit to the data by simply predicting normal for all the data points. This is the case because in the confusion matrix it can be seen that 0 predictions were made from the High class. Because of this the above model will provide us with 0 benefit. Thus the relationship between change in household income and 30 year mortgage rate do not correlate strongly with our ability to predict high or normal change.

Inflation Rate vs 30 Year Mortgage Rate

```r
# Create training set the works with kNN.
tr_data <- train_set %>%
  dplyr::select(mort_rate, inflation, home_price_class) %>%
  rename(y = mort_rate,
         x = inflation)
# Train kNN
train_knn_2 <- train(home_price_class~.,
                     method="knn",
                     data = tr_data,
                     tuneGrid = data.frame(k=2:20),
                     trControl = trainControl(method="cv", number = 5))
```

```r
# Plot Accuracy vs number of neighbors
ggplot(train_knn_2)+
  theme_classic()+
  xlab("Number of Neighbors")
```



```
train_knn_2$bestTune$k
```

```
## [1] 14
```

```
# Create test set the works with kNN.
tt_data <- test_set %>%
  dplyr::select(mort_rate, inflation, home_price_class) %>%
  rename(y = mort_rate,
         x = inflation)
# Produce confusion matrix for analysis
confusionMatrix(
  predict(train_knn_2, tt_data, type="raw"),
  as.factor(test_set$home_price_class))
```

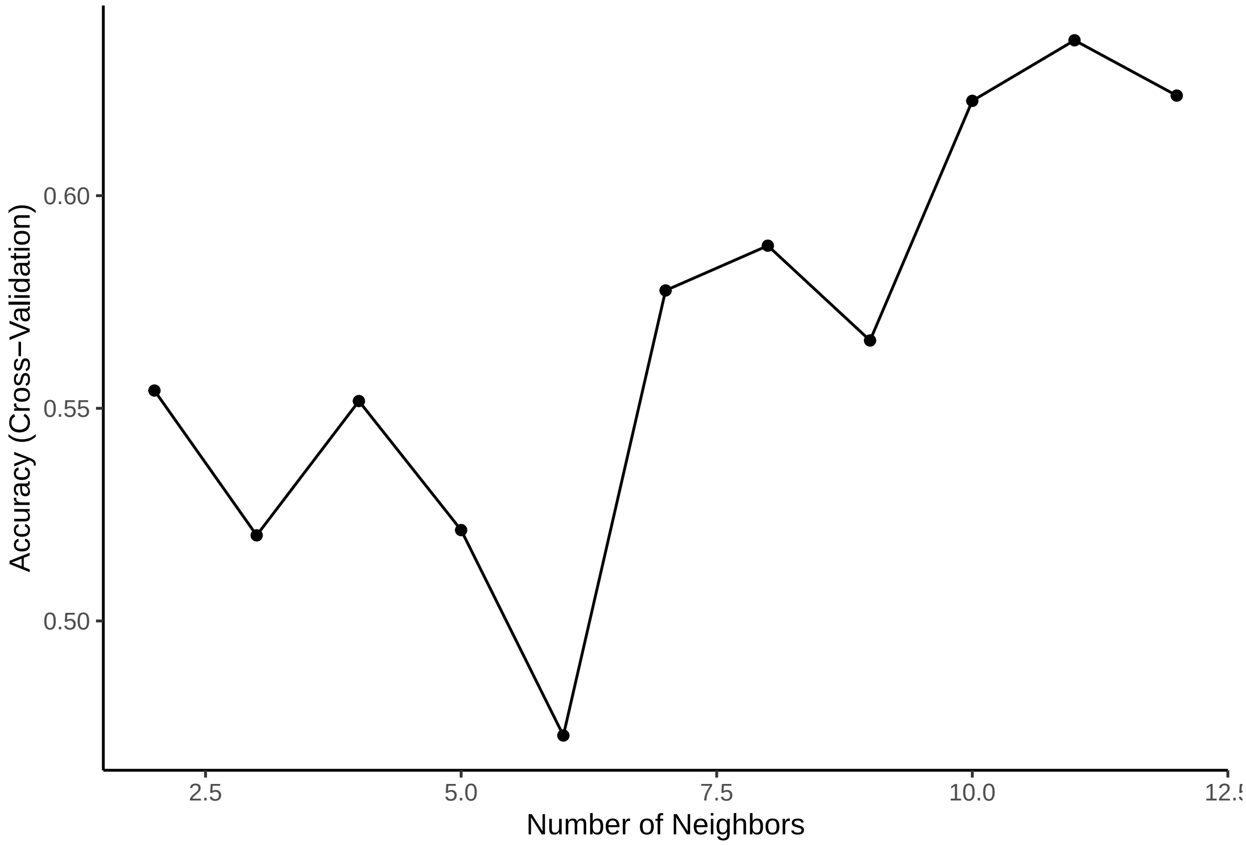
```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction High Normal
##     High       1       7
##     Normal   21      30
##
##                Accuracy : 0.5254
##                  95% CI : (0.3912, 0.657)
##     No Information Rate : 0.6271
##     P-Value [Acc > NIR] : 0.95840
##
##                   Kappa : -0.165
##
##  Mcnemar’s Test P-Value : 0.01402
##
##             Sensitivity : 0.04545
##             Specificity : 0.81081
##          Pos Pred Value : 0.12500
##          Neg Pred Value : 0.58824
##              Prevalence : 0.37288
##          Detection Rate : 0.01695
##    Detection Prevalence : 0.13559
##       Balanced Accuracy : 0.42813
##
##        ’Positive’ Class : High
##
```

The result above show that this model did not attempt to predict all Normal class to increase accuracy. This is an improvement but the overall accuracy as well as the sensitivity show that the model is still useless. Thus the relationship between the inflation rate and 30 year mortgage rate do not correlate strongly with our ability to predict high or normal change.

Inflation Rate vs Change in Household Income

```
# Create training the works with kNN.
tr_data <- train_set %>%
  dplyr::select(median_income_change, inflation, home_price_class) %>%
  rename(y = median_income_change,
         x = inflation)
# Train knn
train_knn_3 <- train(home_price_class~.,
                     method="knn",
                     data = tr_data,
                     tuneGrid = data.frame(k=2:12),
                     trControl = trainControl(method="cv", number = 5))
```

```
# Plot Accuracy vs number of neighbors
ggplot(train_knn_3)+
  theme_classic()+
  xlab("Number of Neighbors")
```



```
train_knn_3$bestTune$k
```

```
## [1] 11
```

```
# Create testing set the works with kNN.
tt_data <- test_set %>%
  dplyr::select(median_income_change, inflation, home_price_class) %>%
  rename(y = median_income_change,
         x = inflation)
# Produce confusion matrix for analysis
confusionMatrix(
  predict(train_knn_3, tt_data, type="raw"),
  as.factor(test_set$home_price_class))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction High Normal
##     High      3     10
##     Normal   19     27
##
##                Accuracy : 0.5085
##                  95% CI : (0.375, 0.6411)
##     No Information Rate : 0.6271
##     P-Value [Acc > NIR] : 0.9769
##
##                   Kappa : -0.146
##
##  Mcnemar's Test P-Value : 0.1374
##
##             Sensitivity : 0.13636
##             Specificity : 0.72973
##          Pos Pred Value : 0.23077
##          Neg Pred Value : 0.58696
##              Prevalence : 0.37288
##          Detection Rate : 0.05085
##    Detection Prevalence : 0.22034
##       Balanced Accuracy : 0.43305
##
##        'Positive' Class : High
```

```
##
```

This model clearly performed the best out of our 3 kNN models. The sensitivity is slightly over 50 percent. This is a promising result and points to a possibility that this information could be further refined into something useful. But as is this model still only performs as good as a random guess for our purposes, thus it a strong correlation between our predictors and target variable cannot be made.

kNN Review:

Overall the kNN models did not perform as desired. The sensitivity remained low for all the models and no conclusions could be made. This bodes poorly for the future models as I fear they may suffer a similar fate.

Classification Tree Testing

The next series of models that will be built and tested are classification trees. These models were selected because like kNN models they have a flexible decision boundary and can adapt to data with a more subtle pattern.

30 Year Mortgage Rate vs Change in Household Income

```
tr_data <- train_set %>%
  dplyr::select(mort_rate, median_income, home_price_class) %>%
  rename(y = mort_rate,
         x = median_income)

train_ctree_1 <- train(home_price_class~.,
                   method="ctree",
                   data = tr_data,
                   trControl = trainControl(method="cv", number = 5))
```

```
tt_data <- test_set %>%
  dplyr::select(mort_rate, median_income, home_price_class) %>%
  rename(y = mort_rate,
         x = median_income)

confusionMatrix(
  predict(train_ctree_1, tt_data, type="raw"),
  as.factor(test_set$home_price_class))
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction High Normal
##     High       0      0
##     Normal    22     37
##
##                Accuracy : 0.6271
##                  95% CI : (0.4915, 0.7496)
##     No Information Rate : 0.6271
##     P-Value [Acc > NIR] : 0.5579
##
##                   Kappa : 0
##
##  Mcnemar’s Test P-Value : 7.562e-06
##
##             Sensitivity : 0.0000
##             Specificity : 1.0000
##          Pos Pred Value :    NaN
##          Neg Pred Value : 0.6271
##              Prevalence : 0.3729
##          Detection Rate : 0.0000
##    Detection Prevalence : 0.0000
##       Balanced Accuracy : 0.5000
##
##        ’Positive’ Class : High
##
```

30 Year Mortgage Rate vs Inflation

```
tr_data <- train_set %>%
  dplyr::select(mort_rate, inflation, home_price_class) %>%
  rename(y = mort_rate,
         x = inflation)

train_ctree_2 <- train(home_price_class~.,
                       method="ctree",
                       data = tr_data,
                       trControl = trainControl(method="cv", number = 5))
```

```
tt_data <- test_set %>%
  dplyr::select(mort_rate, inflation, home_price_class) %>%
  rename(y = mort_rate,
         x = inflation)

confusionMatrix(
  predict(train_ctree_2, tt_data, type="raw"),
  as.factor(test_set$home_price_class))
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction High Normal
##     High      0      0
##     Normal   22     37
##
##                Accuracy : 0.6271
##                  95% CI : (0.4915, 0.7496)
##     No Information Rate : 0.6271
##     P-Value [Acc > NIR] : 0.5579
##
##                   Kappa : 0
##
##  Mcnemar's Test P-Value : 7.562e-06
```

```
##
##             Sensitivity : 0.0000
##             Specificity : 1.0000
##          Pos Pred Value :    NaN
##          Neg Pred Value : 0.6271
##              Prevalence : 0.3729
##          Detection Rate : 0.0000
##    Detection Prevalence : 0.0000
##       Balanced Accuracy : 0.5000
##
##        'Positive' Class : High
##
```

Change in Median Income vs Inflation

```
tr_data <- train_set %>%
  dplyr::select(median_income_change, inflation, home_price_class) %>%
  rename(y = median_income_change,
         x = inflation)

train_ctree_3 <- train(home_price_class~.,
                   method="ctree",
                   data = tr_data,
                   trControl = trainControl(method="cv", number = 5))
```

```
tt_data <- test_set %>%
  dplyr::select(median_income_change, inflation, home_price_class) %>%
  rename(y = median_income_change,
         x = inflation)

confusionMatrix(
  predict(train_ctree_3, tt_data, type="raw"),
  as.factor(test_set$home_price_class))
```

```
## Confusion Matrix and Statistics
```

```
##
##           Reference
## Prediction High Normal
##     High      0      0
##     Normal   22     37
##
##                Accuracy : 0.6271
##                  95% CI : (0.4915, 0.7496)
##     No Information Rate : 0.6271
##     P-Value [Acc > NIR] : 0.5579
##
##                   Kappa : 0
##
##  Mcnemar’s Test P-Value : 7.562e-06
##
##             Sensitivity : 0.0000
##             Specificity : 1.0000
##          Pos Pred Value :    NaN
##          Neg Pred Value : 0.6271
##              Prevalence : 0.3729
##          Detection Rate : 0.0000
##    Detection Prevalence : 0.0000
##       Balanced Accuracy : 0.5000
##
##        ’Positive’ Class : High
##
```

Classification Tree review:

Individual reports were not generated for each relationship modeled for the classification trees. This is because each relationship produced the same disappointing result. Each tree optimized for accuracy by predicting normal change for every input. This makes every model produced here useless. Additionally this does not increase our understanding on the relationship between these predictors and our target variable.

Random Forest Tree Testing:

Lastly a random forest model will be built leveraging all three predictors to try to enhance the sensitivity of the model. The decision to use random forest was similar to the two previous models, it too has a flexible decision boundary, additionally like the classification tree the model's logic can be easily interpreted.

```
tr_data <- train_set %>%
  dplyr::select(mort_rate, median_income_change, home_price_class, inflat
  mutate(home_price_class = as.factor(home_price_class))



train_rf <- randomForest(home_price_class ~ ., data = tr_data, ntrees = 8
                         keep.forest = TRUE, importance = TRUE)

train_rf
```

```
##
## Call:
##  randomForest(formula = home_price_class ~ ., data = tr_data,      ntre
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 1
##
##         OOB estimate of  error rate: 55.06%
## Confusion matrix:
##        High Normal class.error
## High     13     24   0.6486486
## Normal   25     27   0.4807692
```

For our last model we can see that the predictor error for our target class is very high at nearly .70 . This is an unacceptable level and means this model is still useless for out purposes.

Conclusion

The predictors and relationships I used to try to predict whether or not a quarter would have a large home price growth were ineffective. This could be because of a few reason:

- User Error, I could have conducted the analysis wrong or introduced some error into the analysis at some point. A note was made to me that I dichotomized the data which removed some of the information present, thus a better approach could have been regression analysis and classifying the results.
- These predictors/relationship cannot fully explain home price growth. It is possible that more predictors are needed to fully explain the changes in home price and I did not include them. This would suggest that the predictors I selected did have an impact on home price changes but not by themselves.
- These predictors/relationship do not relate to home price growth.

Final Note:

If I were to continue this project I would look for more predictors and explore more interactions between these variable to hopefully produce better results. One key predictor that I feel would've helped were I able to obtain the data was Housing inventory. Lastly I would take the note to use regression analysis rather than classification, feel like it was a waste to ignore the finer relationship between the numeric values of median house price change and the predictors in favor of house price class values.