

STAT 420 Final Report: Modeling California Housing Prices

Shashank Thakur, Avinash Tiwari, Kathryn DeWitt

Modeling California Housing Prices

Introduction

Executive Summary

In this report we will discuss the creation of a linear model to model median house value of a block group in California (CA) in the 1990's based on administrative (Census) data. Given the recent turbulence in the real estate market and associated bidding wars, there is a renewed interest in what the value of a house is. Utilizing methods from STAT 420, we underwent model selection to identify the optimal model that minimized test RMSE, met the assumptions of linear modelling, and had the fewest parameters to enhance explainability. The best linear model include the parameters <!TODO: add params from best linear model> and a test RMSE of <!TODO: add test RMSE>. Our model provides utility in explaining the relationship between how features of a California Block Group relate to the median house value of the block group.

The Data

For our project we leveraged a data set from kaggle [California Housing Prices](#). The data contains information from the 1990 California census and pertains to houses found in a given California district. It contains one row per census block group. A block group is the smallest geographical unit for which the U.S. Census Bureau publishes sample data (a block group typically has a population of 600 to 3,000 people).

The *California Housing Prices* data set pertains to the houses found in a given California district and some summary stats about them based on the 1990 census data. There are 10 columns and 20,640 rows in the data. The columns are as mentioned below:

Column Name	Description
longitude	Longitudinal location of block group of houses.
latitude	Latitudinal location of block group of houses
housing_median_age	Median age of the houses in the block group
total_rooms	Total rooms in the group of houses
total_bedrooms	Total bedrooms in the block group
population	Total number of people in the block group
households	Households comprised in the block group
median_income	Median income calculated from the individual income of house.
median_house_value	Median house value of a block group.
ocean_proximity	Indicating whether each block group is near the ocean, near the Bay Area, inland or on an island.

Limitations of the Data

In order to ensure the reader understands the caveats of working with our data, we will first underscore the differences between census housing data and typical sources of real estate data. The census data set, unlike scraped real estate data, is gathered by an administrative body, the Census. Since the stakeholder involved in collecting the data is not interested in selling a house, we see different features represented that relate more to geography (Latitude, longitude), population density (population, number of households), and age of the building (median_housing_age). The data set is aggregated at the block level (rather than at the house level). This means our analysis is not directly modelling housing values, but the median housing value of a neighborhood. We believe this will help us better to understand the value of the neighborhood as opposed to the value of the amenities of the individual houses.

Another important limitation of the source of this data is the date range. Since the data were gathered as part of the 1990 Census, we cannot extrapolate to current housing prices or utilize the model to explain what features are important to median house value at any other time period. For this particular reason, we focused our project on modelling rather than prediction. While there are few use cases for a prediction algorithm for median house value of a census block group in the 1990's at the present date, understanding historical features that related to a census block groups' price point provides an opportunity for understanding how median house price varied based on parameters at the time of data collection.

Methods

We will now discuss the steps we used to generate our linear model. We began by conducting exploratory data analysis (EDA). Based on the EDA, we identified steps necessary to clean our data. In order to ensure our model was not overfit, we split the data set into a 70-30 train-test split. Next we underwent model selection process to minimize the test RMSE of our model. In order to conduct model selection and graph the results, we leveraged helper functions that we will describe before diving into the model selection.

Helper Functions

Function : `model_performance_metrics` will generate model performance metrics such as RMSE and R-squared, etc.

Inputs:

`models` : A list of models to generate the performance metrics

`model_names` : Names of the models passed to the function

Returns : Returns a data frame with following performance metrics as columns and a row for each model provided in `model_names` as input.

R2: R-Squared
ADJR2: Adjusted R-Squared
RMSE: Root Mean Square Error
LOOCV_RMSE: Leave One Out Cross Validated RMSE
Coefs: Total number of coefficients in the model

```
model_performance_metrics = function(models, model_names){  
  # Calculate model names and total numbers  
  n_mod = length(models)  
  # Initialize the data frame to be built by the function  
  perf_metrics_df = data.frame(
```

```

    row.names = model_names,
    "R2" = rep(0L, n_mod),
    "ADJR2" = rep(0L, n_mod),
    "RMSE" = rep(0L, n_mod),
    "LOOCV_RMSE" = rep(0L, n_mod),
    "Coefs" = rep(0L, n_mod)
  )

  for (idx in 1:n_mod){
    perf_metrics_df[model_names[idx], "R2"] = summary(models[[idx]])$r.squared
    perf_metrics_df[model_names[idx], "ADJR2"] = summary(models[[idx]])$adj.r.squared
    perf_metrics_df[model_names[idx], "RMSE"] = sqrt(mean(resid(models[[idx]])) ^ 2))
    perf_metrics_df[model_names[idx], "LOOCV_RMSE"] = sqrt(mean((resid(models[[idx]])) /
      (1 - hatvalues(models[[idx]]))) ^ 2))
    perf_metrics_df[model_names[idx], "Coefs"] = length(coef(models[[idx]])))
  }
  return(perf_metrics_df)
}

```

Data Cleaning

Our first step to cleaning the data was reading in the CSV file from kaggle.com. Our team members reviewed the first five rows to ensure that there were no encoding issues reading in the raw data into R.

```

#read in the data
read_in <- function(){
  ca_housing_data = read.csv('../000_Data/california-housing-prices/housing.csv')
  ca_housing_data
}

ca_housing_data <- read_in()
head(ca_housing_data)

##   longitude latitude housing_median_age total_rooms total_bedrooms population
## 1     -122.2     37.88                 41        880          129       322
## 2     -122.2     37.86                 21       7099         1106      2401
## 3     -122.2     37.85                 52       1467          190       496
## 4     -122.2     37.85                 52       1274          235       558
## 5     -122.2     37.85                 52       1627          280       565
## 6     -122.2     37.85                 52        919          213       413
##   households median_income median_house_value ocean_proximity
## 1           126        8.325        452600    NEAR BAY
## 2          1138        8.301        358500    NEAR BAY
## 3           177        7.257        352100    NEAR BAY
## 4           219        5.643        341300    NEAR BAY
## 5           259        3.846        342200    NEAR BAY
## 6           193        4.037        269700    NEAR BAY

```

The first thing we looked at were basic summary statistics and missingness. We found that there were 207 observations missing `total_bedrooms`. We cannot generate a linear model with missing data. However, looking at the collinearity, we will find a solution that allows us to keep these observations.

```

summary(ca_housing_data)

##      longitude      latitude      housing_median_age      total_rooms
##  Min.   :-124   Min.   :32.5   Min.   : 1.0   Min.   :    2
##  1st Qu.:-122   1st Qu.:33.9   1st Qu.:18.0   1st Qu.: 1448
##  Median :-118   Median :34.3   Median :29.0   Median : 2127
##  Mean   :-120   Mean   :35.6   Mean   :28.6   Mean   : 2636
##  3rd Qu.:-118   3rd Qu.:37.7   3rd Qu.:37.0   3rd Qu.: 3148
##  Max.   :-114   Max.   :42.0   Max.   :52.0   Max.   :39320
##
##      total_bedrooms      population      households      median_income
##  Min.   :     1   Min.   :     3   Min.   :     1   Min.   : 0.50
##  1st Qu.: 296   1st Qu.:  787   1st Qu.: 280   1st Qu.: 2.56
##  Median :435   Median :1166   Median : 409   Median : 3.54
##  Mean   :538   Mean   :1425   Mean   : 500   Mean   : 3.87
##  3rd Qu.:647   3rd Qu.:1725   3rd Qu.: 605   3rd Qu.: 4.74
##  Max.   :6445  Max.   :35682  Max.   :6082  Max.   :15.00
##  NA's   :207
##
##      median_house_value      ocean_proximity
##  Min.   :14999      Length:20640
##  1st Qu.:119600     Class :character
##  Median :179700     Mode  :character
##  Mean   :206856
##  3rd Qu.:264725
##  Max.   :500001
##

```

```

#How many missing values do we have?
colSums(is.na(ca_housing_data))

```

```

##      longitude      latitude      housing_median_age      total_rooms
##          0             0             0             0
##      total_bedrooms      population      households      median_income
##         207             0             0             0
##      median_house_value      ocean_proximity
##          0             0

```

As part of data cleaning, we plotted the `median_house_value` against latitude and longitude to visualize the locations of the different houses. One of the first things we noticed were two distinct clusters of highly priced houses that appear to correspond to Los Angeles and the San Francisco Bay Area in bright red. The lighter green dots in the plot illustrate the lowest priced houses. Note that this scale differentiates by color using the median house value as a midpoint, which helps to illustrate concentrated areas of high price values.

```

## Map the data
#Source: https://stackoverflow.com/questions/65233613/plot-a-map-using-lat-and-long-with-r
my_sf <- st_as_sf(ca_housing_data, coords = c('longitude', 'latitude'))
my_sf <- st_set_crs(my_sf, value = 4326)

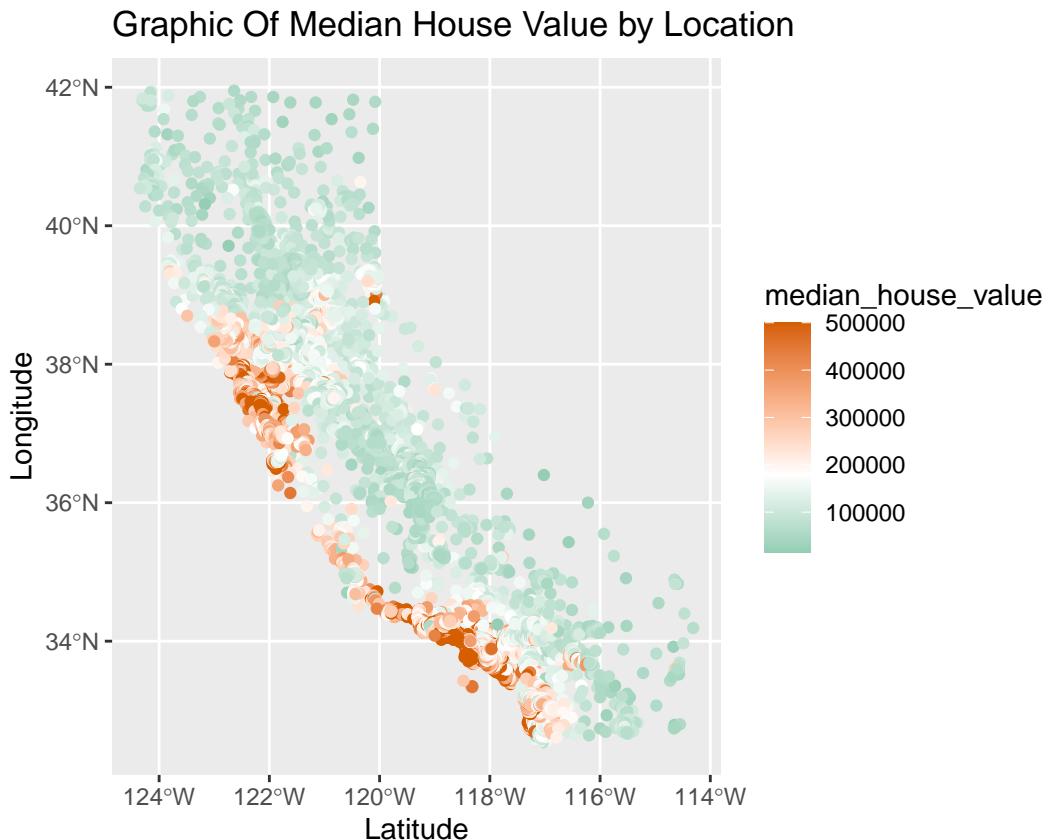
ggplot(my_sf) +
  geom_sf(aes(color = median_house_value)) +
  labs( y = "Longitude",
       x = "Latitude",

```

```

        title = "Graphic Of Median House Value by Location") +
scale_colour_gradient2(low = "#009E73",
midpoint = median(ca_housing_data$median_house_value),
high = "#D55E00")

```



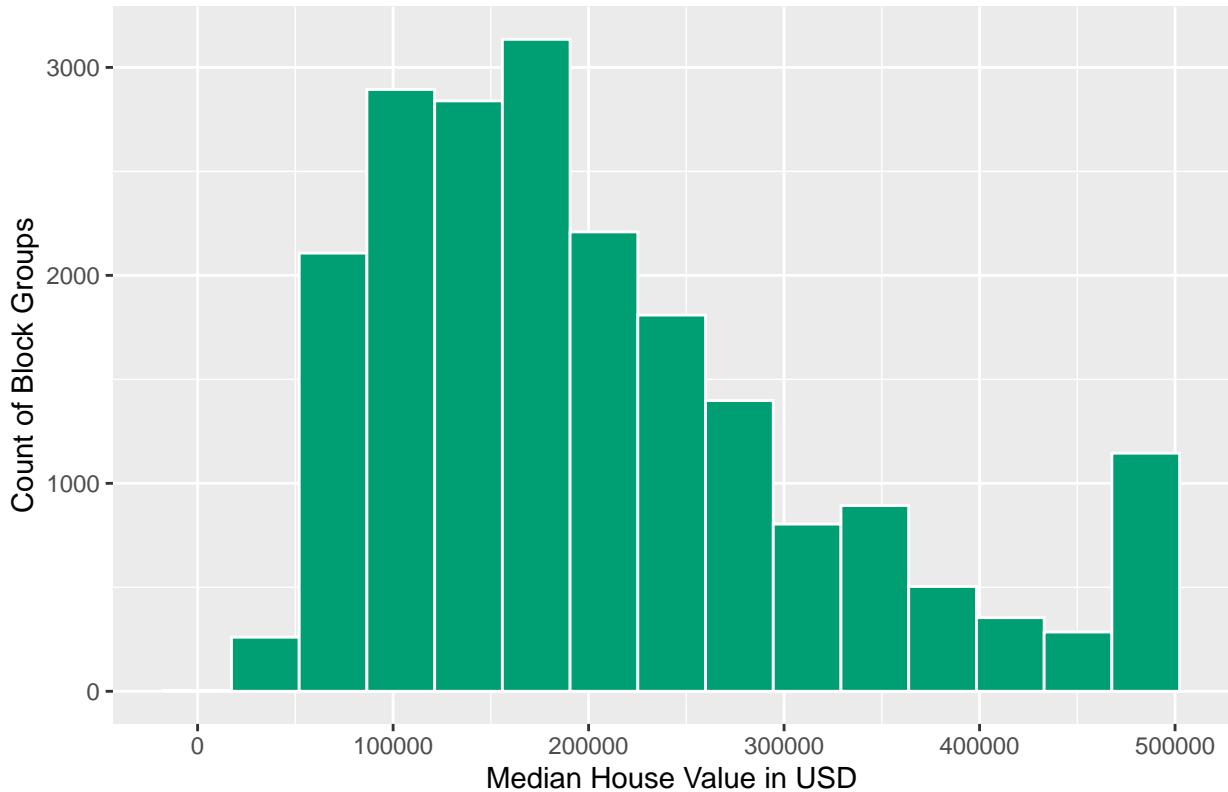
We also generated histograms for all relevant predictors. The most important finding discovered while performing EDA was the long tail of our outcome variable `median_house_value`.

```

ggplot(ca_housing_data, aes(median_house_value)) +
  geom_histogram(fill = "#009E73", color= "#FFFFFF", bins = 15) +
  labs( y = "Count of Block Groups",
        x = "Median House Value in USD",
        title = "Histogram of Median House Values")

```

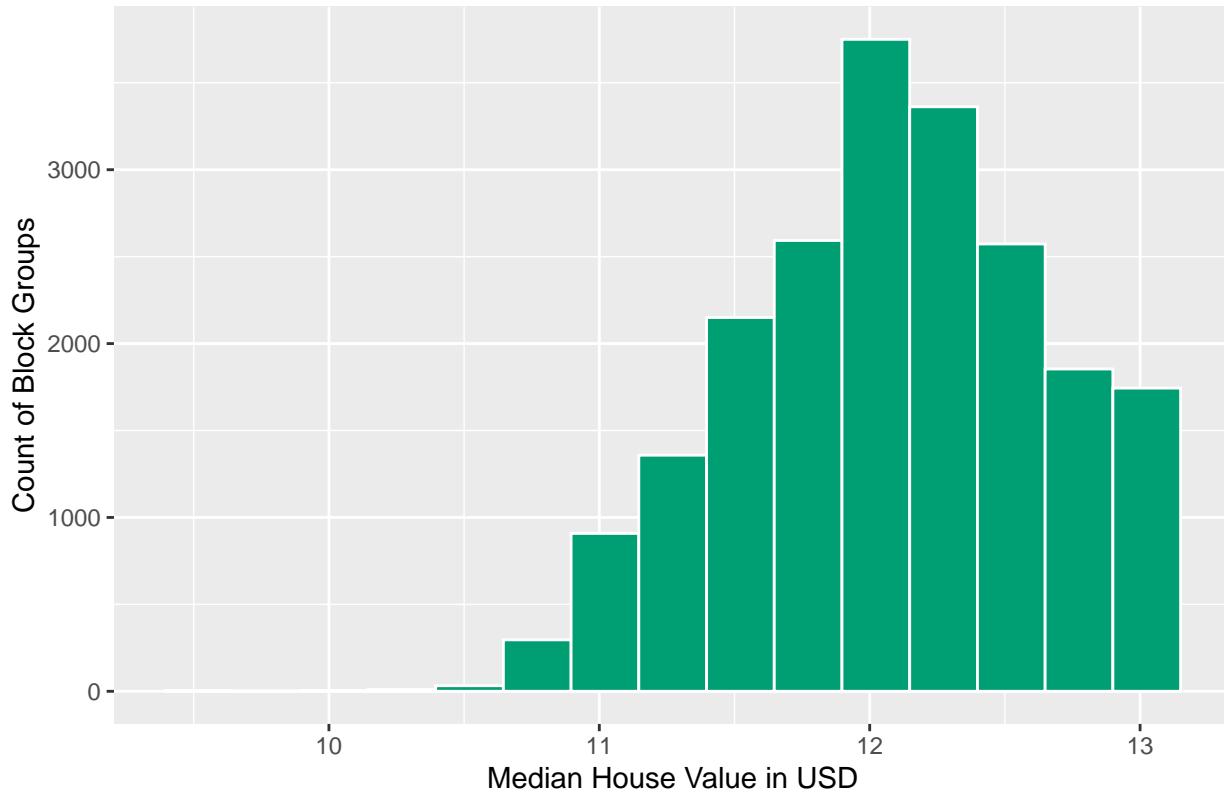
Histogram of Median House Values



Based on the histogram, we see that our response variable could benefit from a log transformation. Generally most dollar variables require a log transformation, and `median_house_value` is no different. This log transformation will also assist with linear model assumptions during model selection.

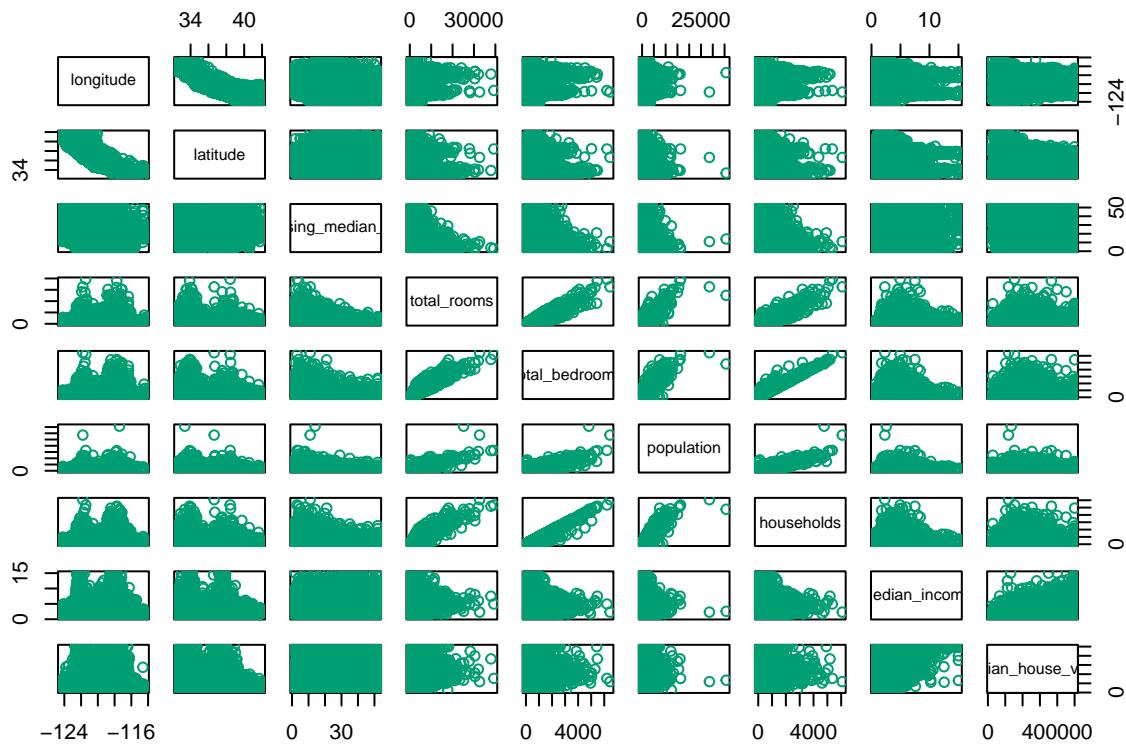
```
ggplot(ca_housing_data, aes(log(median_house_value))) +  
  geom_histogram(fill = "#009E73", color= "#FFFFFF", bins = 15) +  
  labs( y = "Count of Block Groups",  
        x = "Median House Value in USD",  
        title = "Histogram of Median House Values")
```

Histogram of Median House Values



In addition to looking at graphs and reviewing summary statistics, we also began to identify variables with collinearity issues via a correlation matrix and pairwise plot. In the pairwise plot we see that `total_bedrooms` is highly collinear with `total_rooms`, but `total_rooms` is not missing any values. With this in mind, we can utilize the variable `total_rooms` to preserve the 207 observations.

```
## pairs & Cor matrix
numeric_vals <- unlist(lapply(ca_housing_data, is.numeric), use.names = FALSE)
pairs(ca_housing_data[, numeric_vals], col="#009E73")
```



```
round(cor(ca_housing_data[,numeric_vals]), 2)
```

```
##          longitude latitude housing_median_age total_rooms
## longitude      1.00     -0.92        -0.11       0.04
## latitude      -0.92      1.00         0.01      -0.04
## housing_median_age -0.11      0.01        1.00      -0.36
## total_rooms      0.04     -0.04        -0.36       1.00
## total_bedrooms     NA       NA          NA        NA
## population       0.10     -0.11        -0.30       0.86
## households       0.06     -0.07        -0.30       0.92
## median_income     -0.02    -0.08        -0.12       0.20
## median_house_value -0.05   -0.14         0.11       0.13
##                      total_bedrooms population households median_income
## longitude            NA        0.10       0.06      -0.02
## latitude             NA       -0.11      -0.07      -0.08
## housing_median_age   NA       -0.30      -0.30      -0.12
## total_rooms           NA        0.86       0.92       0.20
## total_bedrooms         1        NA        NA        NA
## population            NA       1.00       0.91       0.00
## households            NA       0.91       1.00       0.01
## median_income          NA       0.00       0.01      1.00
## median_house_value     NA      -0.02       0.07      0.69
##                      median_house_value
## longitude            -0.05
## latitude             -0.14
```

```

## housing_median_age           0.11
## total_rooms                  0.13
## total_bedrooms                 NA
## population                   -0.02
## households                     0.07
## median_income                  0.69
## median_house_value                1.00

#Population and households are collinear
#Population and Total rooms are also collinear
#latitude and longitude are also highly correlated

#Remove total_bedrooms
ca_housing_data <- subset(ca_housing_data, select=-c(total_bedrooms))

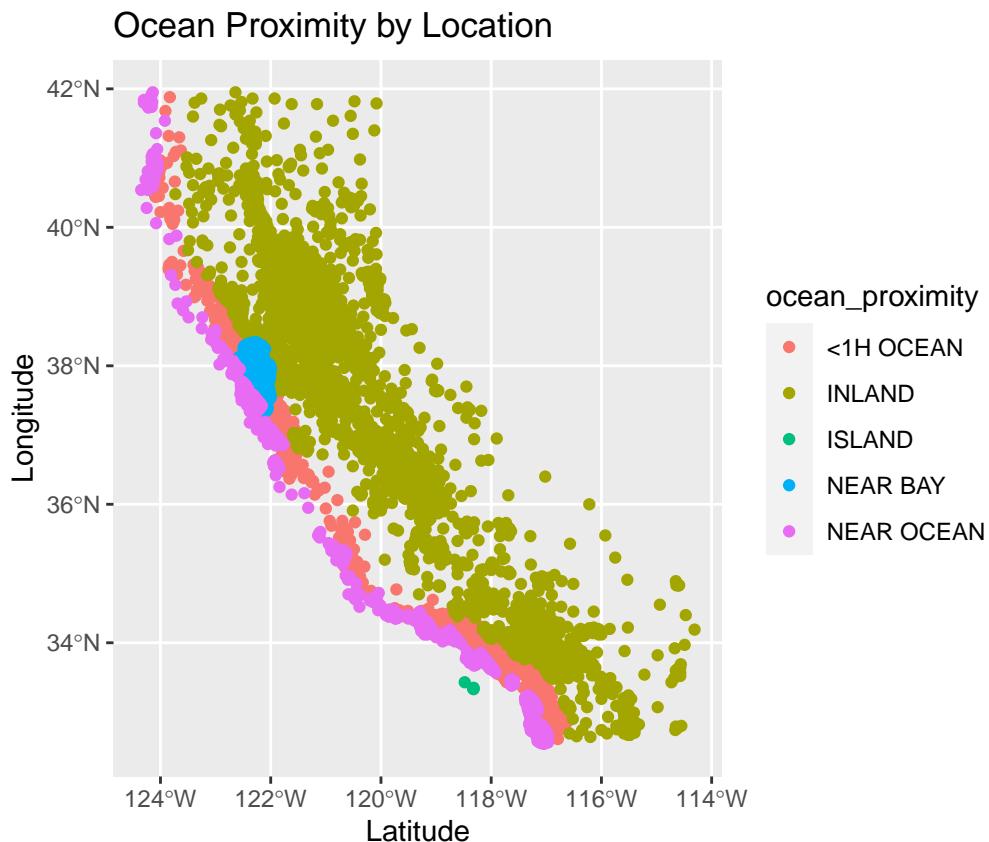
```

We have another column `ocean_proximity` which is a categorical variable indicating how close the block is from ocean. Using latitude and longitude, we can see how this factor variable helps to classify different census blocks. Keeping in mind that we saw distinct clusters of highly priced median house value in the Bay Area (see the blue “NEAR BAY” region) and LA (around 34N, 118W “NEAR OCEAN” region), this factor along with latitude and longitude will help us to capture the geographic nature of housing value.

```

ggplot(my_sf) +
  geom_sf(aes(color = ocean_proximity)) +
  labs( y = "Longitude",
        x = "Latitude",
        title = "Ocean Proximity by Location")

```

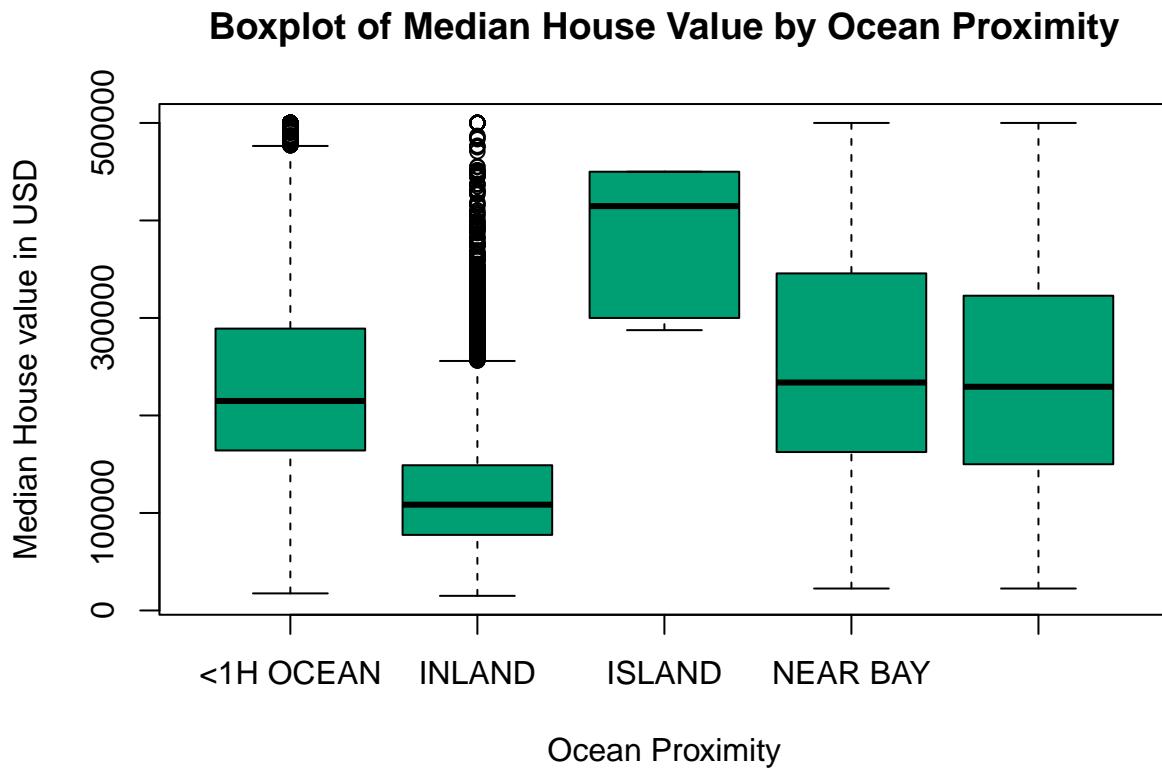


In the table below, the reader can see this column has 5 different categories. It is important to note that the ISLAND category has only 5 observations. During initial analysis, our team determined that ISLAND has a substantially higher average value of response variable `median_house_value` than the other categories as illustrated by the below boxplot. This fact is problematic for our purpose. We will be dropping the rows from data set with ISLAND category and coercing the predictor `ocean_proximity` to be categorical.

```
#Ocean proximity - only 5 in island, and substantially different. I propose removing from consideration
table(ca_housing_data$ocean_proximity)
```

```
##
##   <1H OCEAN      INLAND      ISLAND    NEAR BAY NEAR OCEAN
##       9136        6551         5        2290       2658
```

```
boxplot(median_house_value ~ ocean_proximity,
ca_housing_data,
main = "Boxplot of Median House Value by Ocean Proximity",
ylab = "Median House value in USD",
xlab = "Ocean Proximity",
col = "#009E73")
```



```
#Remove the islands rows
ca_housing_data = ca_housing_data[ca_housing_data$ocean_proximity != "ISLAND",]

#Now we turn it into a factor after dropping that level
ca_housing_data$ocean_proximity <- as.factor(ca_housing_data$ocean_proximity)
```

```

nrows <- nrow(ca_housing_data)
ncols <- ncol(ca_housing_data)

```

At this point, we have concluded the prerequisite data cleaning and can move onto model selection. Our final data set used for modelling has 20635 and 9. The summary statistics can be found below.

```

#Final dataset and quality assurance
summary(ca_housing_data)

```

```

##      longitude      latitude      housing_median_age      total_rooms
##  Min.   :-124   Min.   :32.5   Min.   : 1.0   Min.   :    2
##  1st Qu.:-122   1st Qu.:33.9   1st Qu.:18.0   1st Qu.: 1448
##  Median : -118   Median :34.3   Median :29.0   Median : 2127
##  Mean   : -120   Mean   :35.6   Mean   :28.6   Mean   : 2636
##  3rd Qu.:-118   3rd Qu.:37.7   3rd Qu.:37.0   3rd Qu.: 3148
##  Max.   : -114   Max.   :42.0   Max.   :52.0   Max.   :39320
##      population      households      median_income      median_house_value
##  Min.   :     3   Min.   :    1   Min.   : 0.50   Min.   :14999
##  1st Qu.:  787   1st Qu.: 280   1st Qu.: 2.56   1st Qu.:119600
##  Median :1166   Median : 409   Median : 3.54   Median :179700
##  Mean   :1426   Mean   : 500   Mean   : 3.87   Mean   :206814
##  3rd Qu.:1725   3rd Qu.: 605   3rd Qu.: 4.74   3rd Qu.:264700
##  Max.   :35682  Max.   :6082   Max.   :15.00   Max.   :500001
##      ocean_proximity
##  <1H OCEAN :9136
##  INLAND    :6551
##  NEAR BAY   :2290
##  NEAR OCEAN:2658
##
##
```

```

str(ca_housing_data)

```

```

## 'data.frame': 20635 obs. of 9 variables:
## $ longitude      : num -122 -122 -122 -122 -122 ...
## $ latitude       : num 37.9 37.9 37.9 37.9 37.9 ...
## $ housing_median_age: num 41 21 52 52 52 52 52 52 42 52 ...
## $ total_rooms     : num 880 7099 1467 1274 1627 ...
## $ population      : num 322 2401 496 558 565 ...
## $ households      : num 126 1138 177 219 259 ...
## $ median_income    : num 8.33 8.3 7.26 5.64 3.85 ...
## $ median_house_value: num 452600 358500 352100 341300 342200 ...
## $ ocean_proximity : Factor w/ 4 levels "<1H OCEAN","INLAND",...: 3 3 3 3 3 3 3 3 3 ...
```

```

#QA: assert no missing values

```

```

if(any(colSums(is.na(ca_housing_data)) > 0)){
  stop("Fatal error. Missing data is one or more columns")
}

```

```

#QA: Assert that there are only 4 levels in ocean_proximity

```

```

if(length(levels(ca_housing_data$ocean_proximity)) != 4){
  stop("Fatal error. There are not 4 levels in ocean_proximity")
}

```

Model Selection Process

Now that we have data cleaned, we will proceed with building a model. The very first step we will take is to divide data into train and test. Based on total number of observations we have, team has decided to use 70%-30% split for train-test data.

We first initialized global variables and created helper functions.

```
# build different models to test the performance
n_models = 7
# create a vector list of models to pass to various functions
cahoupta_models = vector("list", length = n_models)
cahoupta_model_names = vector("character", length = n_models)
# Train-Test Split Ration
tr_pct = 0.70
```

In order to replicate the results, we set a seed to ensure that the test train split of our data was consistent between team members. After creating both the test and train dataset, we began building models.

```
set.seed(420072022)
# Randomly select train indexes
cahoupta_trn_idx = sample(nrow(ca_housing_data),
                           size = trunc(tr_pct * nrow(ca_housing_data)))
# Split into train and test data sets based on the index
ca_housing_data.tr = ca_housing_data[cahoupta_trn_idx, ]
ca_housing_data.te = ca_housing_data[-cahoupta_trn_idx, ]

n_tr_rows = nrow(ca_housing_data.tr)
n_te_rows = nrow(ca_housing_data.te)
```

Originally, we started with a simple additive linear model.... <!TODO: add more>

We then tried to reduce the model size using backwards AIC/BIC selection.... <!TODO: add more>

We also evaluated the impact of using interaction terms... <!TODO: add more>

<!TODO: add more about model selection!>

Results

Chosen Model

Discussion

Utility of Model

section should contain discussion of your results and should frame your results in the context of the data. How is your final model useful?

Future Directions and Lessons Learned

<!TODO: Avi to add other directions we did not have time to implement>

Appendix

<!TODO: add details in the appendix on model selection and possibly EDA>

Authors

Team Member's Names:

- Kathryn DeWitt
- Shashank Thakur
- Avinash Tiwari

NetIDs:

- kdewitt3
- sthakur5
- tiwari6