# California_Housing_Report

January 18, 2020

```
[7]: <IPython.core.display.HTML object>
```

Welcome! This notebook is a Data Science/Machine Learning Project

## 1 Table of Contents

#### 1.0.1 Abstract (tl;dr)

The passing of Tenant Protection Act of 2019 in California ushers in an era of tighter rent controls that California landlords must understand and adjust to. This projects main goal is to provide current landlords with insights towards determining the rental price of their property based on attributes of their property and city real estate market. The objective was to build a model of median rental rate using a subset of variables collected for each city in California from Towncharts.com (a website that provides free housing market data mostly taking from the most recent 2018 American Community Survey). Currently estimates that take into account landlords property attributes and attributes of their cities real estate market are outsourced and often come at a cost. A minimum performance measure of a Root Mean Squared Error of 192 dollars two days of work for an individual working minimum wage was desired. This project applies an Ensemble Learning model consisting of two estimators an Elastic Net and Random Forest Regressor which achieves a RMSE on the test set of ~198 dollars just shy of our business goal. We find that the monthly cost of housing for property owners including mortgage payment, taxes, insurance,and utilities is the number one predictor of median rental rates. While median housing cost for homeowners with a mortgage(including the cost of the mortgage or other debt) is the number-two predictor of median rental rates and both share a linear relationship with median rental rate. We note that some interesting outlier cities Bradbury, Vernon, Industry, Indian Wells. For future research the model may be improved by obtaining further predictors of median rental rate from other sources. Specifically, Towncharts.com has Education, Economy, and Demographic Data for California cities that may be further gathered and could prove to be valuable in reducing the RMSE performance measure of the Ensemble model. Lastly, a good idea would be to seek domain expertise in regards to what

valuable categorical variables should be accounted for when predicting median rental price for a property in a certain city.

**Keywords** California Housing Market, Median Rental Rate, Ensemble Learning, Towncharts.com, python, scikit-learn

Section **??**

## 2 Framing the Problem and Looking at the Big Picture

Motivation: Whether you are currently renting or a landlord being informed about what affects median rental rate will give you the power to be a smart decision maker.

Make sure to check out the Section **??** about Bob at the end to see the applicable results from the end to end project.

The objective of this project is to build a model of median rental rates for cities in California using publicly available information about these cities.

The main goal is to provide current landlords with insights towards determining their optimal rental rate amount based on attributes of their properties. Additional questions that will hopefully be answered are: Which attributes of properties are the most informative on predicting the median rental amount to charge? Out of those attributes that are most informative whats their relationship with median rental rate is it exponential, linear, constant as they increase/decrease? All in all the goal is to share major insights and surprises about the California housing market using publicly available data.

So let's begin

For landlords charging the optimal rent is critical, you want to ensure that the price is low enough to attract applicants, while high enough to cover costs. Pricing units lower may result in problematic tenants, but higher prices can lead to longer vacancies.

The current solutions/workarounds are that currently the estimates of the median rental price of property for landlords might be typically non-holistically based or is outsourced.Meaning consulting directly with a real estate agent or property management company to take a look at comparable rentals in your area which often comes at a cost. Additionally estimates are never precise and perhaps don't provide a margin of error. Our goal here is to have our estimate based on publicly available data be less than 192 dollars that is less than two days of work for an individual working minimum wage in California that will be our minimum performance needed to reach a satisfactory estimate.

In technical terms: This project is a typical supervised learning task given the fact that the training examples are labeled (each instance (a city) comes with the expected output i.e, the cities median rental rate). Moreover, we are predicting the value of $Y$ (median rental rate) using $X_1...X_m$ where $\mathbf{m}$ = *number of features* = 39. This task is a a typical regression task in particualr a multiple regression problem. Additionally, seeing as there is no continous flow of data coming in the ML system, there is no need to adjust to changing data rapidly, and the data is small enough to fit in memory,so plain batch learning should do just fine.

Performance measure: Root Mean Square Error(RMSE) this performance measure will be aligned with the objective as it will give an idea of how much error the system typically makes in its predictions. Our minimum performance needed to reach the business objective would be a RMSE of less than 192.

$$RMSE(X,h) = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(h(x^i) - y^i)^2}$$

For robustness the Mean Absolute Error will be used to check if the final model performs well across other than just the main performance measure > $MAE(X,h) = \frac{1}{m}\sum_{i=1}^{m} \mid h(x^i) - y^i \mid$

*Note in a Business setting*: After framing the problem and looking at the big picture I would list and verify the assumptions that were made so far (by me and others) for example, the variables to be looked at are satisfactory to the department for investigation, a numerical response of median rental rate is the expectation my department has and not a classification of whether its above or below a certain price. This will help avoid a difference in expectations and help verify that the characteristics of my system output correspond to the expected signal my project manager/supervisor has for my project.

## 3 Data Collection

Most of the data to be used in this project was gathered via webscraping from Towncharts.com on August ,2019. This website provides free housing market data mostly taking from the most recent 2018 American Community Surveys as well as data collated from a large variety of government agencies and public data sources.

Here is a snapshot of a web page whose information was scraped for a given city

Additional variables were merged from open sourced datasets

For more information about how regular expressions, pandas, beautiful soup and other tools were used to produce the california_housing.csv the following two links outline the process * Scraping the data * Merging datasets

Checking the legal obligations from the user aggreement on Towncharts.com use of this data is allowed and encouraged.

*Note in a business setting*: it would be good practice to import the csv file into a table in the companies internal database. This provides the opportunity for collaboration as all a coworker would need to access the data would be to get their credentials and access authorizations.

**Download the Data**

Best practice calls for me to use a python file which i have created for fetching the data and writing a function to load the data.

#### 3.0.1 Data dictionary: 39 variables collected

- housing_units:Total number of housing units in the area
- housing_density : The number of housing units per square mile in the area
- change_hunits: Change in housing units from 2010 to 2017
- %rent_home_percent: The percent of all occupied housing units that are rental housing units

- % owned_homes: The percent of all occupied housing units that are owned housing units
- $ med_homeval: Median home value i.e how much property is worth( house and lot, mobile homes and lot, or condominium unit) if it was for sale
- $ med_rental_rate: the median monthly rental amount for a rental unit in this area
- $ med_owner_cost: The monthly cost of housing for property owners including mortgage payment, taxes, insurance,and utilities.
- % med_own_cost_aspercentof_income: The monthly owner cost as a percent of the household income. This measure is an excellent way to understand how affordable housing is for owners in an area
- % med_hval_aspercentof_medearn: How much the property is worth(house and lot, mobile home and lot, or condominium unit) if it was for sale as a percent of the median earnings for a worker in the area
- $ med_hcost_ownmortg: Median housing cost for homeowners with a mortgage(including the cost of the mortgage or other debt)
- $ med_hcost_own_wo_mortg: Median housing cost for homeowners who do not have a mortgage. This isolates the cost of ownership seperate from the financing cost of debt
- % hcost_aspercentof_hincome_ownmortg: Homeowners with a mortgage showing total cost (including mortgage debt) as a percent of household income
- hcost_as_perc_of_hincome_womortg: Homeowners without a mortgage showing total cost as a percent of household income.
- $ med_real_estate_taxes: The median real estate taxes paid by owners of homes in the area
- family_members_per_hunit: The average size of related families members who live together in a housing unit.
- median_num_ofrooms: The average number of rooms of total rooms for housing units in the area
- median_year_house_built: The average year the housing units were built in the area. This indicates the average age of housing units in the area.
- household_size_of_howners: For people who own their homes how many people on average are living in them whether they are part of family or related or not.
- household_size_for_renters: The average size of a household for people who are renting.
- med_year_moved_in_for_owners: The median year that a home owner moved into their home
- med_year_renter_moved_in: The median year that a renter moved into their home
- The following variables are proportion of monthly rental rates by size of Rental in Bedrooms for the city: studio_1000_1499,studio_1500_more, studio_750_999, onebed_1000_1499, onebed_1500_more, onebed_750_999, twobed_1000_1499, twobed_1500_more, twobed_750_999, threebed_1000_1499, threebed_1500_more, threebed_750_999
- city: Name of city

- Latitude: Latitude of the city
- Longitude: Longitude of the city
- Population: Population of the city

## 4 To skip the EDA click Section ??

## 5 Exploring the Data

Checking out the Dataframe

```
   housing_units  housing_density  change_hunits  rent_home_percent  \
0         8751.0            156.2           -3.7               50.7   
1         7674.0            984.7            1.2               25.6   
2        32414.0           3104.1            0.2               53.0   
3         7724.0           4319.4           -2.1               52.4   
4        30990.0           4061.1            0.2               59.7   

   owned_homes  med_homeval  med_rental_rate  med_owner_cost  \
0         49.3     151600.0           1059.0          1093.0   
1         74.4     745000.0           2261.0          2488.0   
2         47.0     729100.0           1607.0          2259.0   
3         47.6     766000.0           1739.0          2501.0   
4         40.3     553800.0           1286.0          1629.0   

   med_own_cost_aspercentof_income  med_hval_aspercentof_medearn  ...  \
0                             24.0                         427.0  ...   
1                             23.0                         928.0  ...   
2                             21.0                        1006.0  ...   
3                             21.0                         991.0  ...   
4                             22.0                        1296.0  ...   

   twobed_1500_more  twobed_750_999  threebed_1000_1499  threebed_1500_more  \
0             0.032           0.508               0.583               0.137   
1             0.899           0.038               0.000               0.962   
2             0.633           0.019               0.119               0.667   
3             0.821           0.012               0.084               0.849   
4             0.312           0.054               0.180               0.629   

   threebed_750_999         city   Latitude   Longitude  population  \
0             0.181     Adelanto  34.582769 -117.409214     32867.0   
1             0.000  AgouraHills  34.153339 -118.761675     20736.0   
2             0.036      Alameda  37.765206 -122.241636     78246.0   
3             0.000       Albany  37.886869 -122.297747     19682.0   
4             0.039     Alhambra  34.095286 -118.127014     85168.0   

   area_total_sq_mi  
0            56.027  
```

```
1           7.822
2          22.960
3           5.465
4           7.632

[5 rows x 39 columns]


<class 'pandas.core.frame.DataFrame'>
Int64Index: 452 entries, 0 to 451
Data columns (total 39 columns):
housing_units                        452 non-null float64
housing_density                      452 non-null float64
change_hunits                        450 non-null float64
rent_home_percent                    452 non-null float64
owned_homes                          452 non-null float64
med_homeval                          452 non-null float64
med_rental_rate                      451 non-null float64
med_owner_cost                       449 non-null float64
med_own_cost_aspercentof_income      452 non-null float64
med_hval_aspercentof_medearn         452 non-null float64
med_hcost_ownmortg                   451 non-null float64
med_hcost_own_wo_mortg               450 non-null float64
hcost_aspercentof_hincome_ownmortg   451 non-null float64
hcost_as_perc_of_hincome_womortg     452 non-null float64
med_real_estate_taxes                452 non-null float64
family_members_per_hunit             452 non-null float64
median_num_ofrooms                   452 non-null float64
median_year_house_built              452 non-null float64
household_size_of_howners            452 non-null float64
household_size_for_renters           452 non-null float64
med_year_moved_in_for_owners         452 non-null float64
med_year_renter_moved_in             434 non-null float64
studio_1000_1499                     446 non-null float64
studio_1500_more                     442 non-null float64
studio_750_999                       441 non-null float64
onebed_1000_1499                     446 non-null float64
onebed_1500_more                     442 non-null float64
onebed_750_999                       441 non-null float64
twobed_1000_1499                     446 non-null float64
twobed_1500_more                     442 non-null float64
twobed_750_999                       441 non-null float64
threebed_1000_1499                   446 non-null float64
threebed_1500_more                   442 non-null float64
threebed_750_999                     441 non-null float64
city                                 452 non-null object
Latitude                             452 non-null float64
Longitude                            452 non-null float64
```

```
population                             452 non-null float64
area_total_sq_mi                       448 non-null float64
dtypes: float64(38), object(1)
memory usage: 141.2+ KB
```

```
None
```

```
change_hunits has a count of missing values = 2
med_rental_rate has a count of missing values = 1
med_owner_cost has a count of missing values = 3
med_hcost_ownmortg has a count of missing values = 1
med_hcost_own_wo_mortg has a count of missing values = 2
hcost_aspercentof_hincome_ownmortg has a count of missing values = 1
med_year_renter_moved_in has a count of missing values = 18
studio_1000_1499 has a count of missing values = 6
studio_1500_more has a count of missing values = 10
studio_750_999 has a count of missing values = 11
onebed_1000_1499 has a count of missing values = 6
onebed_1500_more has a count of missing values = 10
onebed_750_999 has a count of missing values = 11
twobed_1000_1499 has a count of missing values = 6
twobed_1500_more has a count of missing values = 10
twobed_750_999 has a count of missing values = 11
threebed_1000_1499 has a count of missing values = 6
threebed_1500_more has a count of missing values = 10
threebed_750_999 has a count of missing values = 11
area_total_sq_mi has a count of missing values = 4
```

Each row represents a city in california we have a total collection of 452 cities/instances in the dataset. **Some attributes contain null values**. In our case this means the variable was not in the text section of the housing information provided by Towncharts.com or had missing values from the datasets that were merged. Decisions about which method to use to treat these missing values will be performed later in the notebook. Although the data set is fairly small by Machine Learning standards this data set is great as a first step in understanding median rental rates in California.

It is important to note that all attributes are numerical. No dummy variables need to be created for an ordinal character variable. If there was a categorical attribute with a large number of possible categories it is recommended to avoid the computational complexity by replacing each category with a learnable low dimensional vector called an embedding.

Summary for some of the numerical attributes

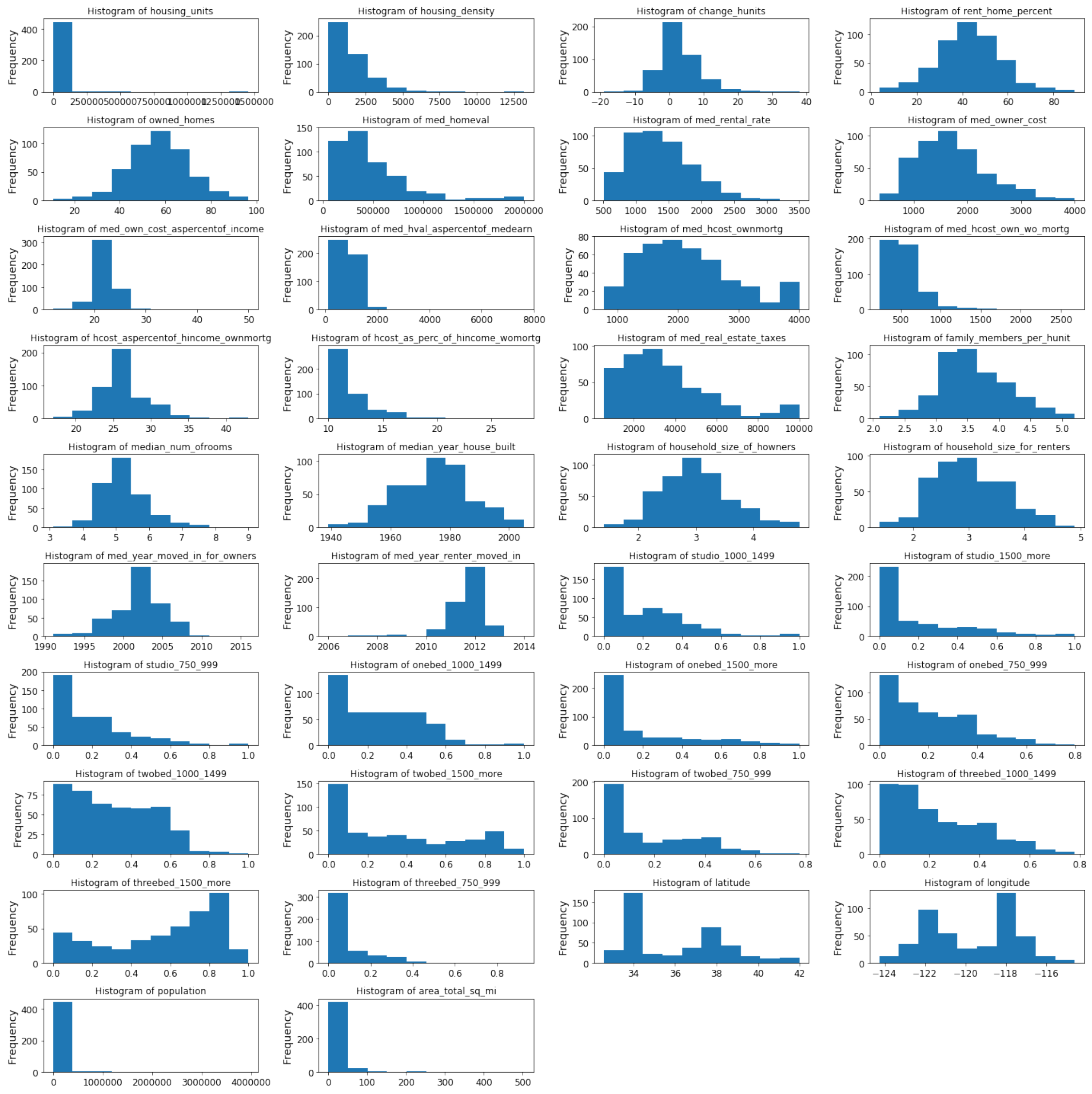
[8]:
```
       housing_units  housing_density  change_hunits  rent_home_percent  \
count      452.00000        452.00000      450.00000          452.00000
mean     24963.13274       1580.13518        2.94800           42.98119
std      79012.58597       1238.87668        5.90521           13.60165
min         37.00000          6.50000      -19.00000            3.60000
25%       4385.75000        863.85000       -0.87500           34.30000
50%      11841.00000       1259.15000        2.10000           43.30000
```

```
75%      26010.25000       1983.20000        5.60000            52.22500
max    1457762.00000      13166.80000       37.90000            89.30000

       owned_homes   med_homeval  med_rental_rate  med_owner_cost  \
count    452.00000     452.00000        451.00000       449.00000
mean      57.01881  491502.22566       1383.59645      1708.65702
std       13.60165  378745.53673        485.59675       659.87260
min       10.70000   56100.00000        508.00000       358.00000
25%       47.77500  237625.00000       1001.50000      1189.00000
50%       56.70000  389750.00000       1314.00000      1627.00000
75%       65.70000  622075.00000       1675.00000      2085.00000
max       96.40000 2000001.00000       3501.00000      4001.00000

       med_own_cost_aspercentof_income  med_hval_aspercentof_medearn  …  \
count                        452.00000                     452.00000  …
mean                          22.19027                     861.29867  …
std                            2.68098                     473.54670  …
min                           12.00000                     122.00000  …
25%                           21.00000                     561.50000  …
50%                           22.00000                     837.00000  …
75%                           23.00000                    1078.25000  …
max                           50.00000                    7619.00000  …

       onebed_1500_more  onebed_750_999  twobed_1000_1499  twobed_1500_more  \
count         442.00000       441.00000         446.00000         442.00000
mean            0.19514         0.19875           0.30597           0.34552
std             0.24949         0.16315           0.20365           0.31083
min             0.00000         0.00000           0.00000           0.00000
25%             0.00425         0.06200           0.12925           0.04700
50%             0.07800         0.16800           0.28050           0.26650
75%             0.31925         0.31700           0.47475           0.63075
max             1.00000         0.79700           1.00000           1.00000

       twobed_750_999  threebed_1000_1499  threebed_1500_more  \
count       441.00000           446.00000           442.00000
mean          0.18066             0.23041             0.56978
std           0.17337             0.17530             0.28140
min           0.00000             0.00000             0.00000
25%           0.03300             0.08500             0.36825
50%           0.10600             0.18650             0.66700
75%           0.32000             0.35675             0.80675
max           0.77400             0.77900             1.00000

       threebed_750_999    population  area_total_sq_mi
count         441.00000     452.00000         448.00000
mean            0.08724   70550.57743          18.85446
std             0.11300  213760.71418          37.40000
```

```
min            0.00000      76.00000          0.31400
25%            0.01800   12164.00000          3.72650
50%            0.04700   33029.00000          9.19400
75%            0.10900   75114.50000         20.09025
max            0.92800 3949780.00000        503.00000

[8 rows x 36 columns]
```

Lets get a further look into our data with a histogram for each numerical attribute in our data

```
C:\Users\Crist\Anaconda3\envs\projectone\lib\site-
packages\numpy\lib\histograms.py:824: RuntimeWarning: invalid value encountered
in greater_equal
  keep = (tmp_a >= first_edge)
C:\Users\Crist\Anaconda3\envs\projectone\lib\site-
packages\numpy\lib\histograms.py:825: RuntimeWarning: invalid value encountered
in less_equal
  keep &= (tmp_a <= last_edge)
```



### A few takeaways: * **Attributes have different scales** some are in percentage, some are in units, and some are in dollars. * **Many histograms are right tail heavy**: Meaning their mean is greater than their median i.e the mean is being 'pulled' by outliers. This tells us that we are going to want a transformer that can handle these outliers when we get to choosing how we are going to scale the data. * Interestingly housing units (mean: 24963, std:79012, median:11841) and change in housing units from 2010 to 2017 (mean:3, std 5.9, median:2.1) are two attributes whose standard deviation are considerably greater than their mean. Looking at the IQR for the range of the middle half of values in change of housing units is Q3(5.6) to Q1(-.875%) = 6.475 and the range of the middle half of housing units is Q3(26010.25) to Q1(4385.75) = 21624.5 * Two attributes, median housing cost for homeowners with a mortgage (including the cost of the mortgage or other debt) and median real estate taxes paid by owners of homes in the area have a noticeable number of high values (spike) at the tail end of their distribution of values. * The mean percent of all occupied housing units that are owned housing units across the cities is greater than the mean of

all occupied housing units that are rental housing units mean of 57% vs mean of 43% both have a fairly normal distribution

- How much the property is worth(house and lot, mobile home and lot, or condominium unit) if it was for sale as a percent of the median earnings for a worker in the area this attribute has a mean of 861% and standard deviation of 473%, and a median of 837%. It is a slightly right skewed distribution.

Recall our goal is to see how these affect median rental rate. But knowing the distributions of these attributes will provide us with an underlying understanding about your cities housing market in relation to the rest of the markets.

#### 5.0.1 Creating a test set

The goal when creating a test set is to create a unique test set and avoid generating a different test every time you work. There are some important considerations when creating a test set. And they depend on whether I will be fetching an updated dataset on occasion or if I would be working on just one dataset. The latter has a simple solution of setting the random number generator seed in sklearns ***train_test_split()*** function, so that it always generates the same shuffled indices for extracting the instances to include in my test set. While the former involves using each instance's identifier to decide whether or not it should go in the test set (assuming instances have a unique and immutable identifer). This eliminates the possibility of leakage of instances previously in the training set into the new test set. Additionally, other sampling methods for creating a test set might be preferred over random sampling. For example in the occasion that you want to ensure that the test set is representative of some attribute in your data set or contain a representative ratio of each class during a classification task than stratified sampling would be the method to use for creating the test set.

In this project implementation of stratified random sampling based on med_homeval: Median home value i.e how much property is worth( house and lot, mobile homes and lot, or condominium unit) if it was for sale a continuous numerical attribute was used to create the test set. My belief is that this is a vital attribute in predicting median rental prices. For example, if my test set contained a grand majority of instances in which med_homeval was between the 3rd Quantile and the max of med_homeval, than no matter what machine learning pipeline I implemented my test error would always be large because my training data did not reflect that stratification of cities with that characteristic.

**Note that one instance was missing a label value this value will be deleted**

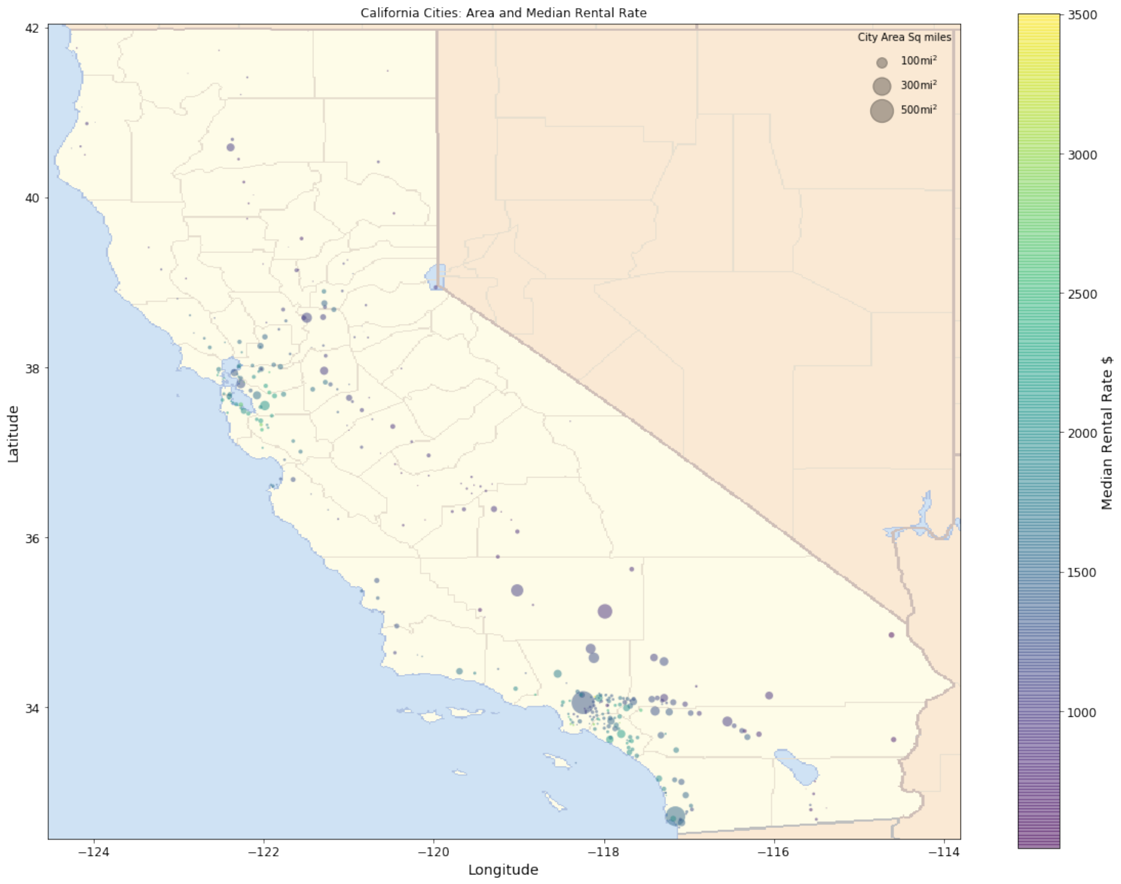
Henceforth why we create the column `property_cost_cat`i.e. property value category in order to perform stratified sampling for our test set with sklearns `StratifiedShuffleSplit` and drop the column after words.

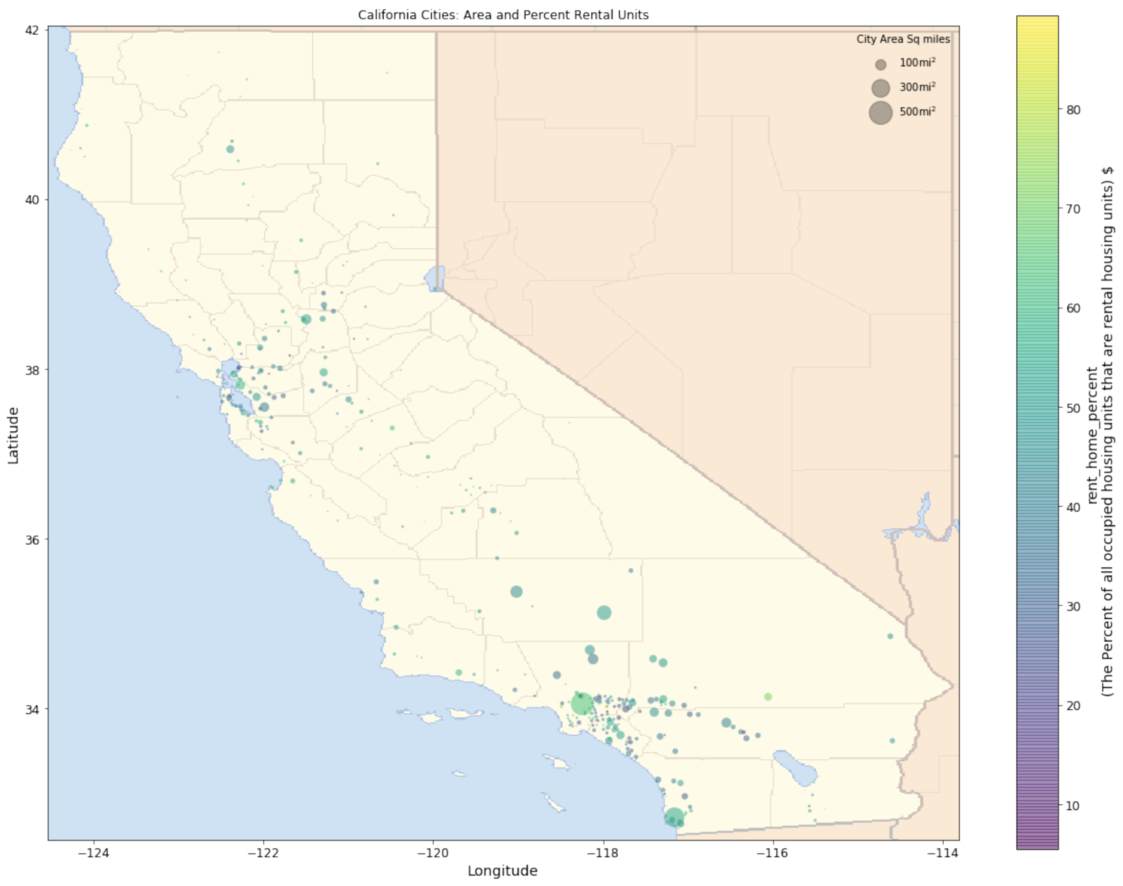
```
Property cost category proportions in the overall dataset:
4   0.25055
2   0.25055
1   0.25055
3   0.24834
Name: property_cost_cat, dtype: float64
Property cost category proportions in the test set:
4   0.25275
```

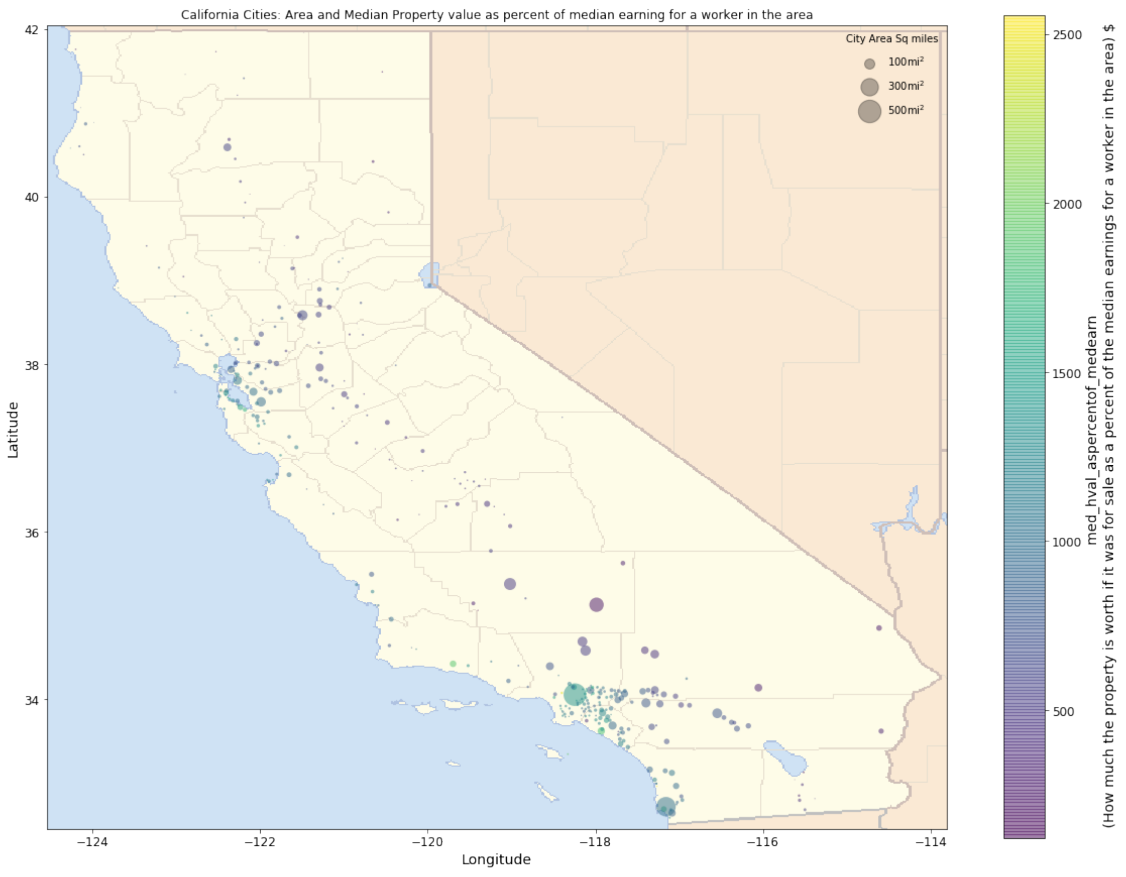
```
2    0.25275
1    0.25275
3    0.24176
Name: property_cost_cat, dtype: float64
```

### 5.0.2 Visualizing the data

There are three interesting plots that come to mind to visualize they are as follows:




California Cities: Area and Percent Rental Units
City Area Sq miles
100mi²
300mi²
500mi²
Latitude
Longitude
42
40
38
36
34
−124
−122
−120
−118
−116
−114
80
70
60
50
40
30
20
10
rent_home_percent
(The Percent of all occupied housing units that are rental housing units) $



California Cities: Area and Median Property value as percent of median earning for a worker in the area
City Area Sq miles
100mi²
300mi²
500mi²
Latitude
Longitude
med_hval_aspercentof_medearn
(How much the property is worth if it was for sale as a percent of the median earnings for a worker in the area) $

These graphs hint at the indication that median rental rate is cheaper in the central valley as opposed to metropolitan areas. Not surprisingly its also where there is a higher percent of rental units available. In addition, the values of the properties in the central valley are not disproportionately valued as a percent of the median earnings for a worker in the area.

**Lets start to investigate which features are strongly correlated with the target variable?**

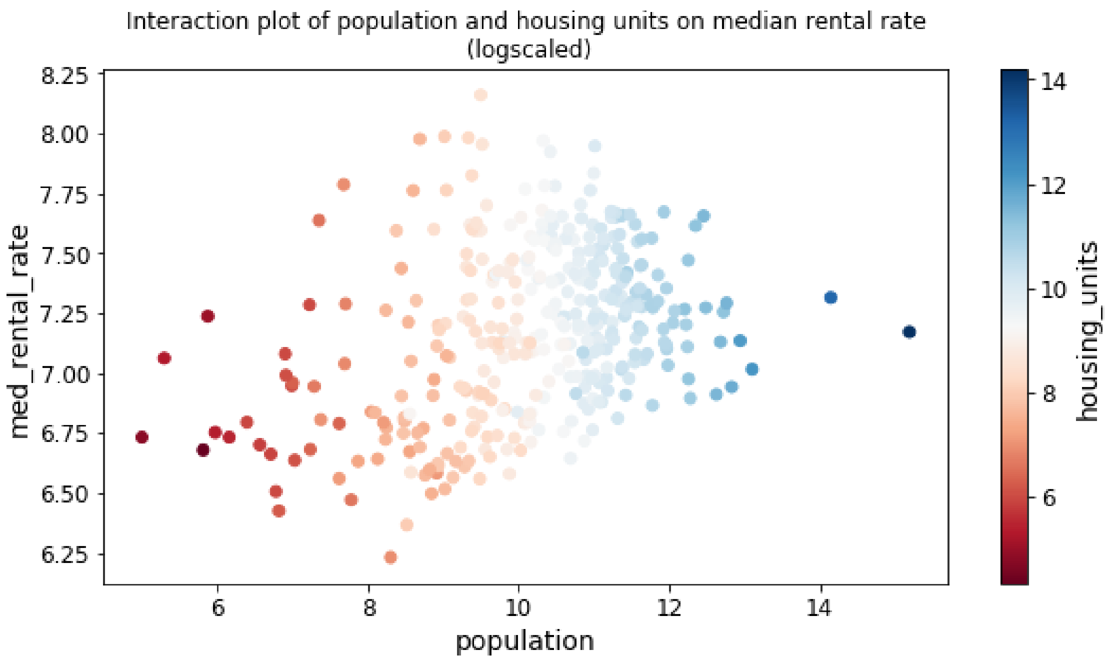
[19]:
```
med_rental_rate                        1.00000
med_hcost_ownmortg                     0.87065
med_owner_cost                         0.86482
twobed_1500_more                       0.85438
onebed_1500_more                       0.84262
med_real_estate_taxes                  0.83948
med_homeval                            0.80378
threebed_1500_more                     0.79335
med_hcost_own_wo_mortg                 0.61931
med_hval_aspercentof_medearn           0.57261
studio_1500_more                       0.55133
median_num_ofrooms                     0.46696
owned_homes                            0.45250
onebed_1000_1499                       0.35833
```

```
studio_1000_1499                    0.34437
housing_density                     0.27741
med_year_renter_moved_in            0.16840
housing_units                       0.02817
population                          0.02304
med_own_cost_aspercentof_income     0.01865
area_total_sq_mi                    0.00550
Longitude                          -0.00718
change_hunits                      -0.03019
hcost_aspercentof_hincome_ownmortg -0.09839
median_year_house_built            -0.11817
studio_750_999                     -0.12784
med_year_moved_in_for_owners       -0.13266
hcost_as_perc_of_hincome_womortg   -0.13416
household_size_of_howners          -0.17191
household_size_for_renters         -0.19199
twobed_1000_1499                   -0.19831
Latitude                           -0.23818
family_members_per_hunit           -0.31677
onebed_750_999                     -0.32697
rent_home_percent                  -0.45250
threebed_750_999                   -0.59450
twobed_750_999                     -0.70974
threebed_1000_1499                 -0.72493
Name: med_rental_rate, dtype: float64
```

This is great information! There are several attributes that are highly correlated with the median rental rates. For example median rental rate tends to increase when the variables: > med_hcost_ownmortg,med_owner_cost, twobed_1500_more,onebed_1500_more,med_real_estate_taxes,med_homeval etc. increase.

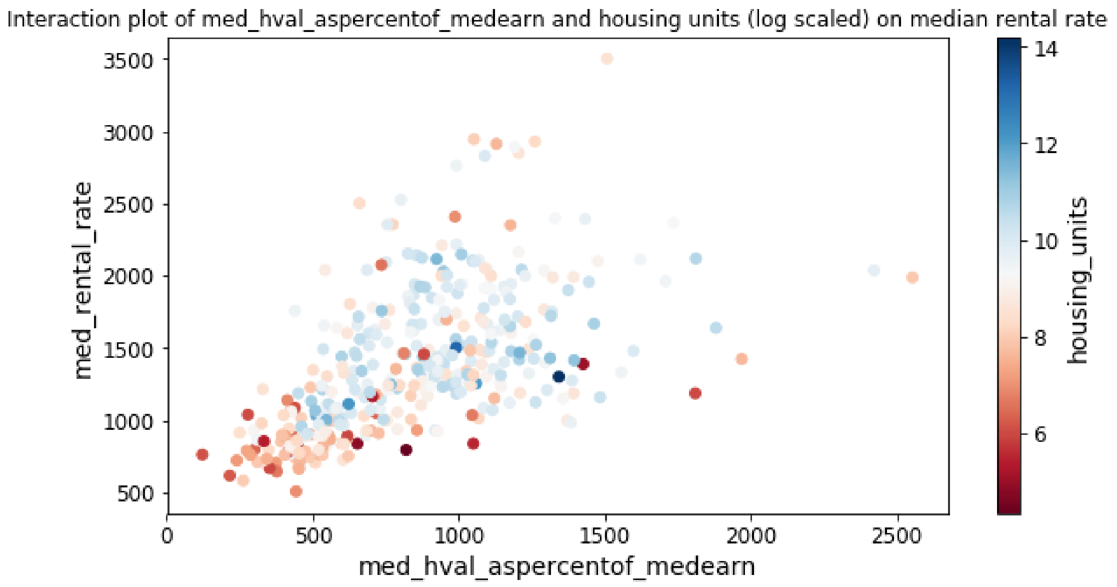
On the other hand the median rental rate tends to decrease when the variables: > threebed_1000_1499,threebed_750_999,twobed_750_999 etc. increase.

Finally lets note that the variables: > area_total_sq_mi, med_own_cost_aspercentof_income, Longitude, change_hunits,hcost_aspercentof_hincome_ownmortg and a other variables are very weakly associated with median rental rate. To my surprise housing density and housing units are positive however this does not tell the whole story and will investigate further with an interaction plot



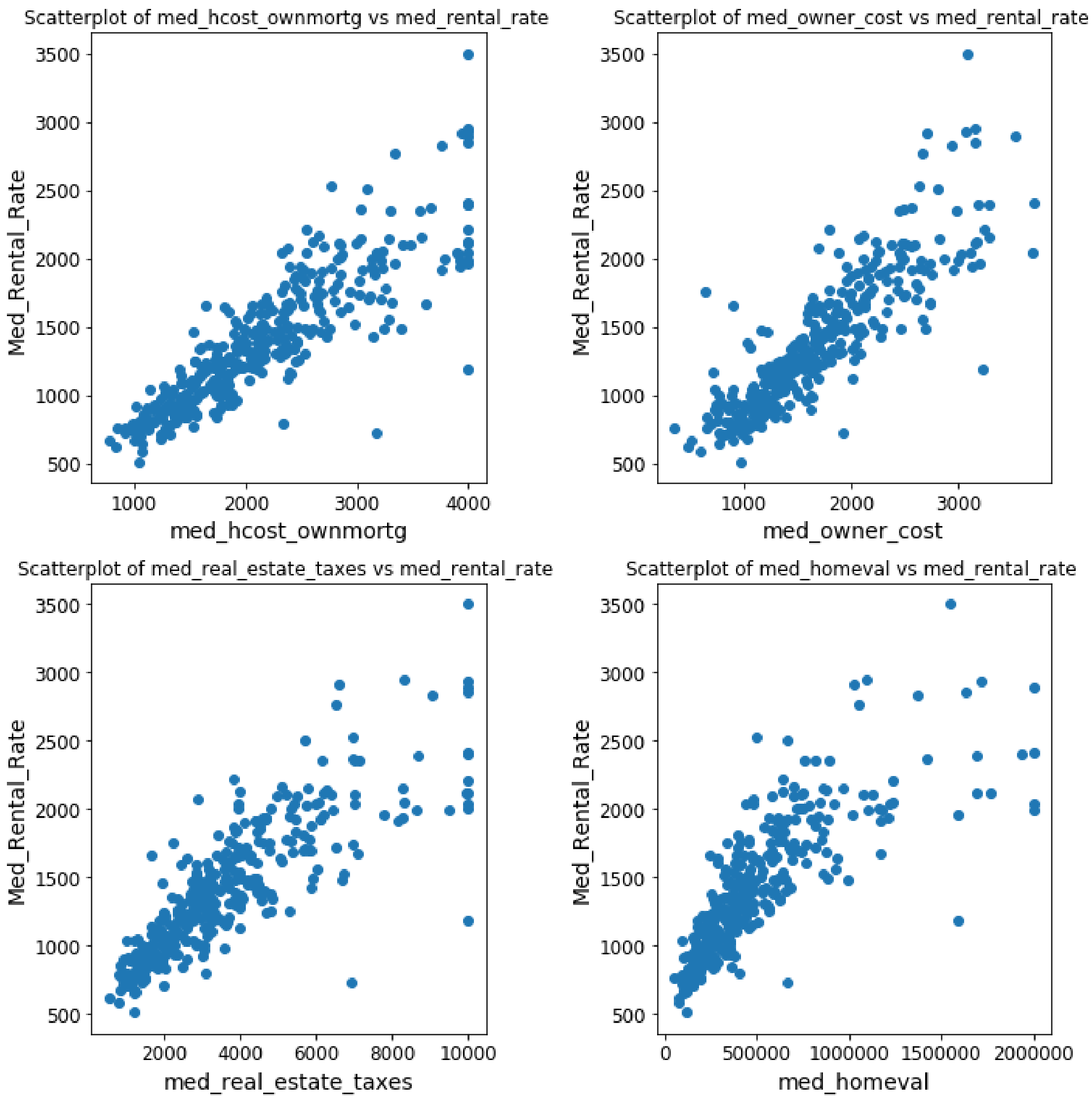
Median rental rates for cities with large populations and numerous housing units appear to be associated with median rental rates for cities with small populations and few housing units. Meaning the effect of housing units on median rental rate on California cities does not demonstrate to be codependent with the population of the cities.

I also wanted to take a closer look at * med_hval_aspercentof_medearn: How much the property is worth (house and lot, mobile home and lot, or condominium unit) if it was for sale as a percent of the median earnings for a worker in the area (%). To my surprise this only has a ~.57 correlation with median rental rate so i want to look into it further



Graph demonstrates that locations with fewer housing units than the average have a large variability in their associated med_hval_aspercentof_medearn.

Below we take a a deeper look at a few strongly correlated attributes.



There are some interesting points from the displayed scatter plots. Let us note where the med_real_estate_taxes >6000 & med_rental_rate < 1500

[23]:
```
            city
174  IndianWells
39      Bradbury
13       Arcadia
```

And let us note where the med_owner cost >3000 and median rental rate is < 1500

```
[24]:        city
      39  Bradbury
```

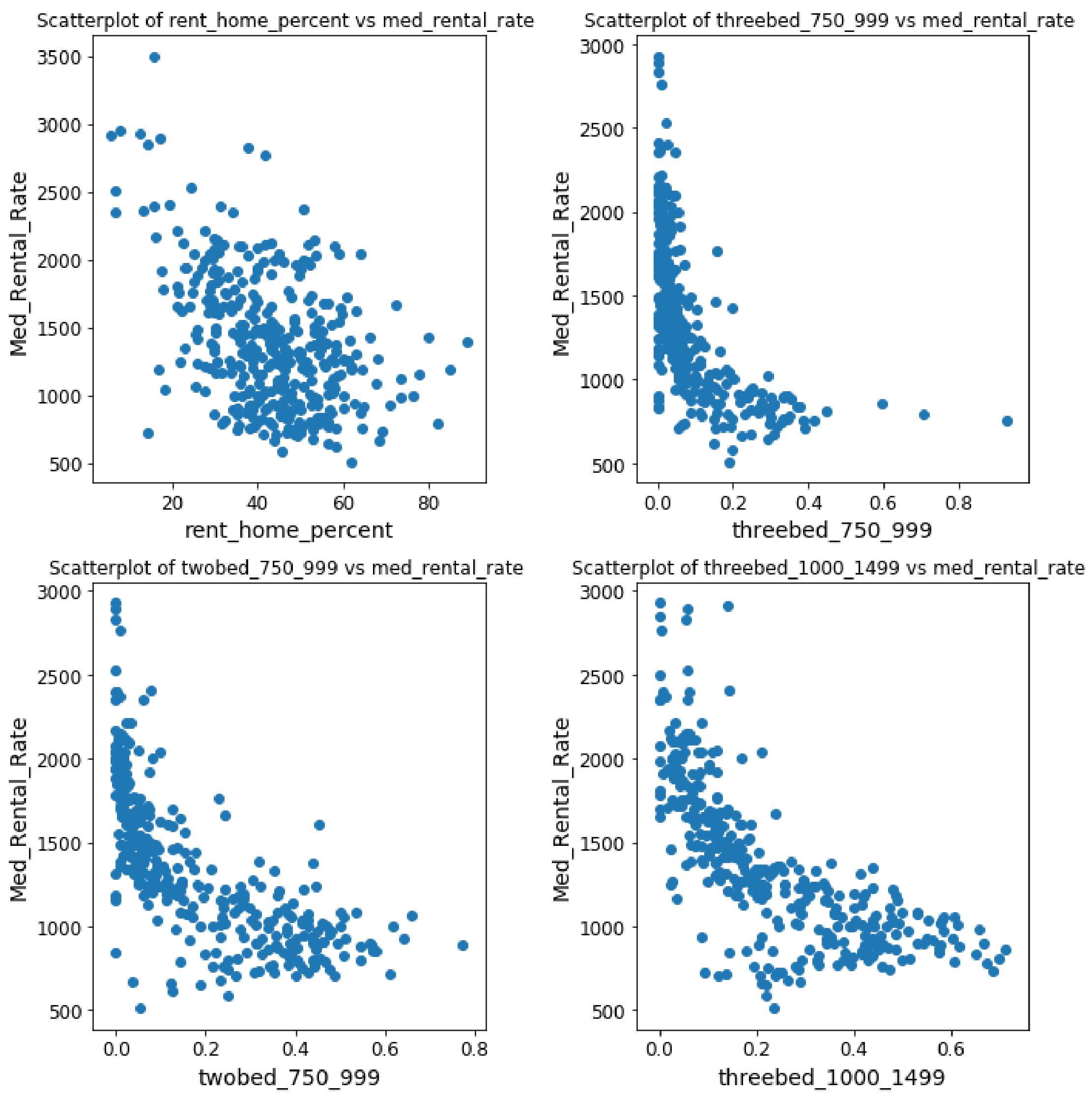
The city in our training dataset with med_homeval greater than 1000000 and median rental rate < 1500 is

```
[25]:        city
      39  Bradbury
```

The city in our training dataset with med_homeval greater than 500000 and median rental rate < 1500 and where med_hcost_ownmortg >3000 & med_rental_rate <1000

```
[26]:           city
      174  IndianWells
```

Lets take a look at some negative correlated variables.



Besides the variable rent_home_percent the Median Rental rate is associated with an ***exponentially*** decrease as the attributes above increase

Lets take a look at the unusual cities we have come to find and a subset of their attributes

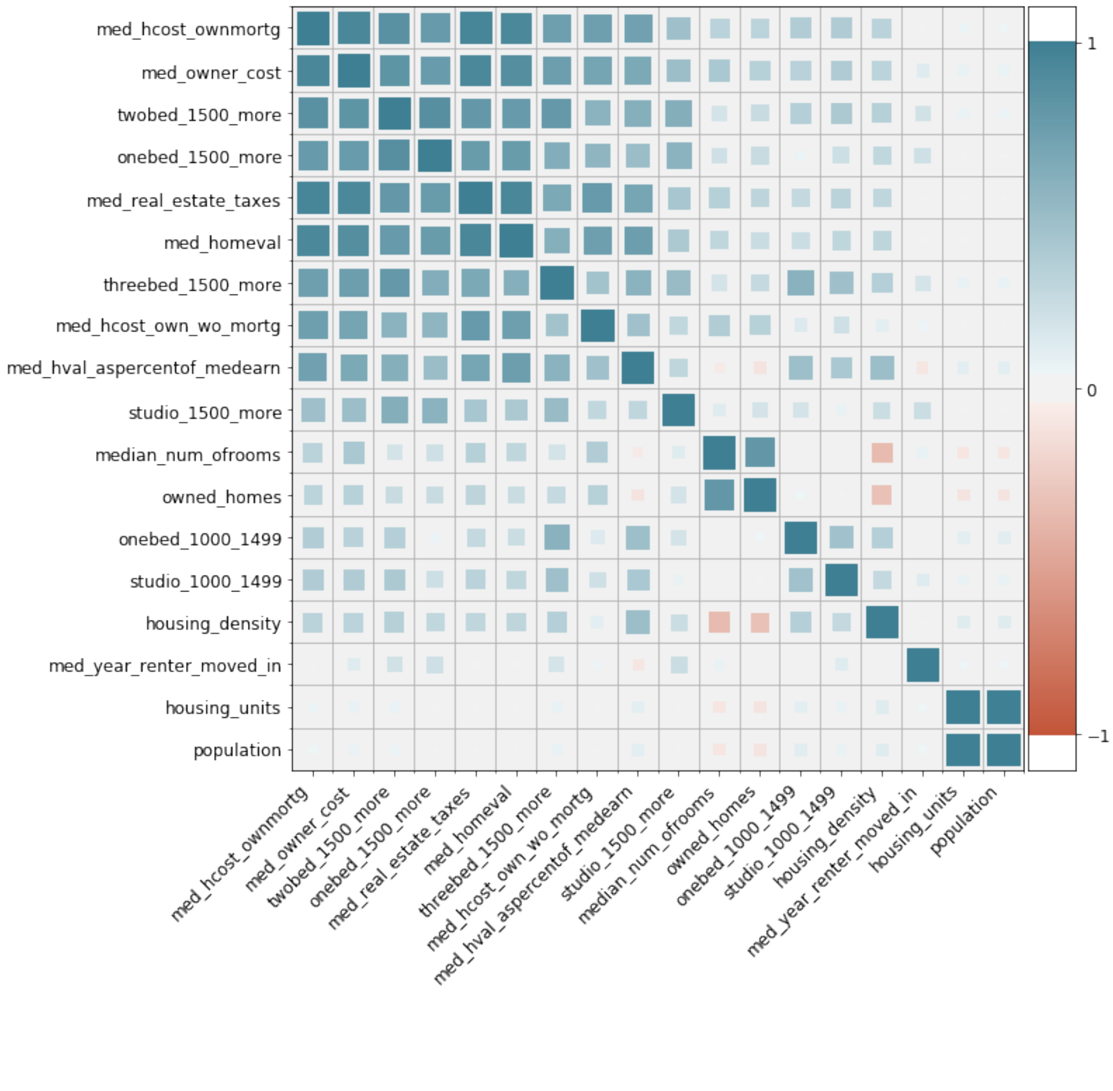
[28]:
```
      housing_units  rent_home_percent  med_rental_rate  med_owner_cost  \
174      5440.00000           14.10000        725.00000      1932.00000
39        418.00000           16.80000       1188.00000      3227.00000
13      21253.00000           39.80000       1481.00000      2287.00000

       med_homeval  med_real_estate_taxes  med_hval_aspercentof_medearn  \
174  667000.00000             6922.00000                     603.00000
39  1595200.00000            10001.00000                    1810.00000
13   994400.00000             6663.00000                    1598.00000
```

```
             city
174   IndianWells
39       Bradbury
13        Arcadia
```

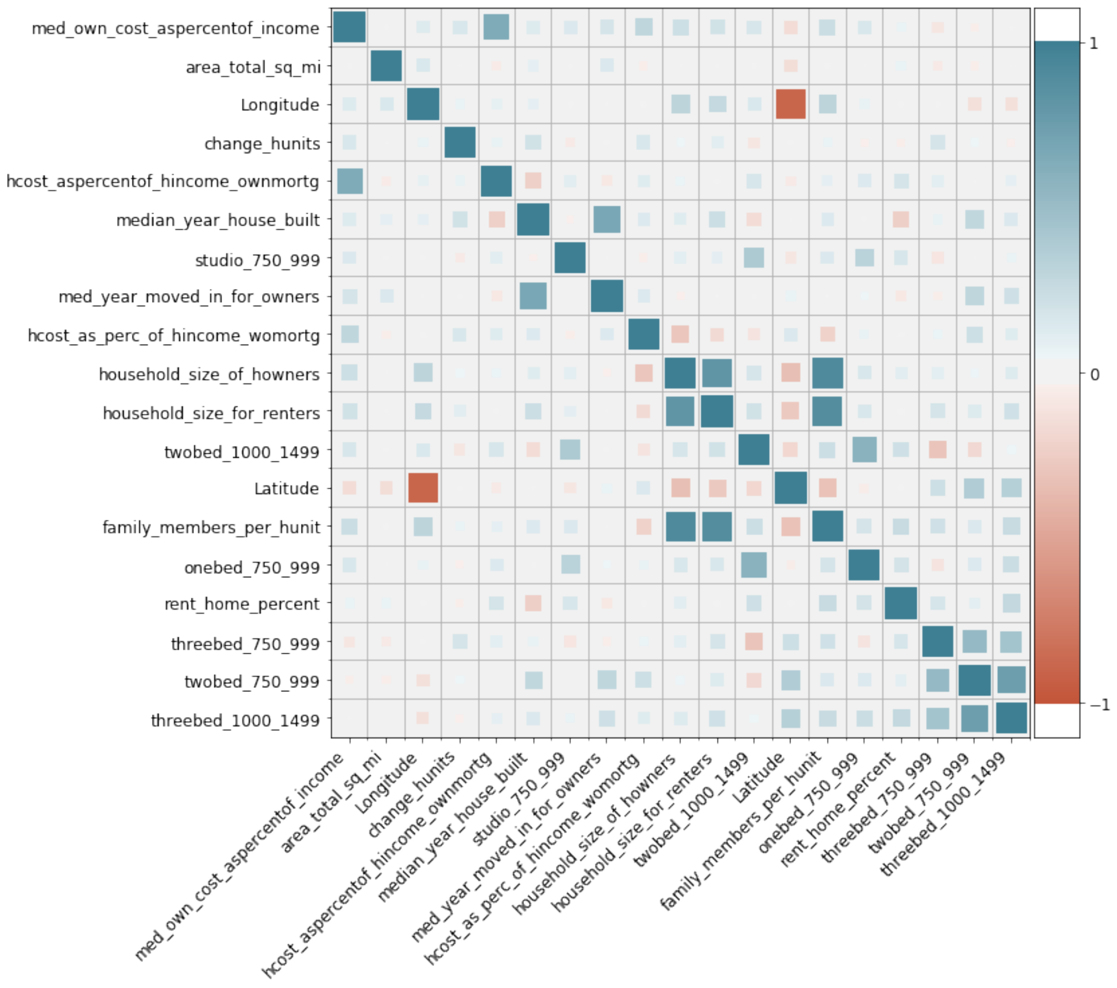
**Are there interesting or unexpected strong correlations between other features?**

Heat maps are a great way to accomplish this investigation.This will help when deciding to use regularization. Because between elastic net/ridge/lasso we know lasso does not handle multicollinearity very well. Additionally, this information may add some insight into feature engineering.

In the following heat maps note that the size of each square corresponds to the magnitude of the correlation it represents. Credit for The function `corrplot` that creates these heatmaps is to Drazen Zaric(https://github.com/drazenz) he wrote a good post on how to create them here.



The heatmap demonstrates that ‘med__hcost__ownmortg’,‘med__hval__aspercentof__medearn’,‘med__owner__cost’, ‘twobed__1500__more’, ‘med__real__estate__taxes’, ‘onebed__1500__more’, ‘med__homeval’, ‘population', 'hunits' have some multicolinearity between them



The heat map above demonstrates some intuitive correlation between variables. For example, its quite expected to have family members per housing unit to be correlated with household size of housing owners and household size for renters this extends to the other variables that are exhibiting correlation between each other.

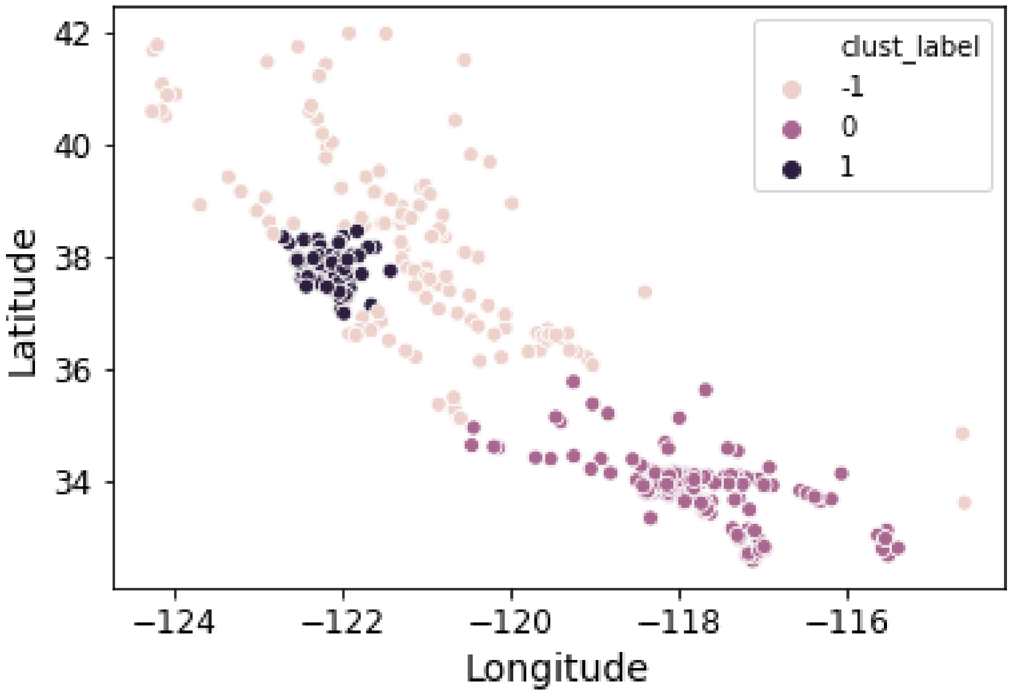
## 6 Preparing and Cleaning the Data for Machine Learning Algorithms

After our data exploration let us start prepping the data and cleaning the data i.e transform our data so that it is in a suitable representation for the machine learning algorithms i.e estimators.

**Predictions coming from informative features with simple algorithms beat predictions from uninformative features with a complex learning algorithms.** For anyone who is familiar with California knows that location may be a large influencer on median rental price. **Thus, in order to make better predictions lets create a location feature. To do this we**

**will utilize a clustering method.** Popular clustering approaches include K-Means,DBSCAN and OPTICS. K-means is not the most appropriate algorithm here. The reason is that k-means is designed to minimize variance. From a statistical and signal processing point of view latitude and longitude data is not "linear". Rather we will use an algorithm that can handle arbitrary distance functions, in particular geodetic distance functions. DBSCAN is an algorithm that can handle this however it has some difficulties in distinguishing separated clusters if they are located too close to each other, even though they have different densities. To overcome this difficulty, OPTICS algorithm was developed. OPTICS ensures good quality clustering by maintaining the order in which the data objects are processed, i.e., high-density clusters are given priority over lower density clusters. Therefore, OPTICS clustering is chosen to provide us with our location feature,specifically sk-learns OPTICS clustering

```
[32]: OPTICS(algorithm='auto', cluster_method='xi', eps=None, leaf_size=30,
             max_eps=inf, metric='minkowski', metric_params=None,
             min_cluster_size=0.05, min_samples=50, n_jobs=None, p=2,
             predecessor_correction=True, xi=0.05)
```



As we can see the optics on our training data provided two clusters with -1 being considered noise.

## 6.1 Cleaning the Data

**Better data will beat fancier algorithms. If i have a properly cleaned dataset, even simple algorithms can learn some valuable insights from the data.**

The first recommended step in data cleaning is to **remove unwanted observations these include duplicate or irrelevant observations**.In our case during data collection preliminary steps were taken to avoid unwanted observations

Next up would be checking the data set for **structural errors**. These may include inconsistencies in attribute units,typos, mislabeled classes etc.This very ‘sexy’ necessary part of data cleaning was already performed. For more details check out data collection

We continue with **handling missing data** lets display once more which attributes have some missing values

```
change_hunits has a count of missing values = 1
med_owner_cost has a count of missing values = 3
med_hcost_own_wo_mortg has a count of missing values = 2
med_year_renter_moved_in has a count of missing values = 13
studio_1000_1499 has a count of missing values = 5
studio_1500_more has a count of missing values = 10
studio_750_999 has a count of missing values = 8
onebed_1000_1499 has a count of missing values = 5
onebed_1500_more has a count of missing values = 10
onebed_750_999 has a count of missing values = 8
twobed_1000_1499 has a count of missing values = 5
twobed_1500_more has a count of missing values = 10
twobed_750_999 has a count of missing values = 8
threebed_1000_1499 has a count of missing values = 5
threebed_1500_more has a count of missing values = 10
threebed_750_999 has a count of missing values = 8
area_total_sq_mi has a count of missing values = 3
```

Dealing with missing values is nuanced as it depends on why the data is missing and it also comes down to the nature of the data. Sunil Ray outlines some popular reasons for why instances have missing values they are: 1. Data Extraction 2. Data Collection a) Missing completely at random b) Missing at random c) Missing that depends on unobserved predictors d) Missing that depends on the missing value itself

And methods to treat missing values include: 1. Deletion (list wise, pairwise) 2. Mean/Mode /Median Imputation 3. Prediction Model **the chosen way in this project using scikit learn IterativeImputer** which imputes missing values by modeling each feature with missing values as a function of other features in a round-robin fashion 4. KNN Imputation

However, it can be noted that perhaps imputing missing values is sub-optimal as it lead to a loss in information as the value was originally missing. Telling your algorithm that a value was missing is informative in itself.One common way is you flag the observation using an indicator variable and fill the missing values with 0. There is also the application of collaborative filtering.

Afterwards we apply **Feature Scaling** scikit-learn has some great documentation on Comparing the effect of different scalers on data with outliers Shay Geller also wrote an excellent article. The **chosen way in this project was scikit learn RobustScaler** a scaler that is robust against outliers.

## 7 Exploring many different models and short-listing the best ones

We are going to proceed to use our training data and our training labels without the variables `city`,`Latitude`,`Longitude`,`change_hunits` `change_hunits`,`studio_1000_1499`, `studio_1500_more`, `studio_750_999`, `onebed_1000_1499`, `onebed_1500_more`, `onebed_750_999`, `twobed_1000_1499`, `twobed_1500_more`, `twobed_750_999`, `threebed_1000_1499`, `threebed_1500_more`, `threebed_750_999`

Next its recommended to try out some models from various categories of Machine Learning algorithms **Note best practice recommends that we cannot simply standardize all of the data once at the beginning and run cross validation on the standardized data. To do so would be allowing the model to peek at the validation set during training.** And so when measuring the performance its is best to standardize our data within each fold in the N-fold cross-validation being done.

Here we train a list of different models in order to shortlist the top promising models.

```
Ordinary Least Squares Linear Regression 5 fold Cross-Validation Results:
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
Scores [[228.70962322 176.52633545 224.31023843 193.71690671 239.92708296]]
Mean 212.6380373537479
Standard Deviation 23.66977971984869
--------------------------------------------------------------------------------
----------------------------------
Ridge(alpha=1, copy_X=True, fit_intercept=True, max_iter=None, normalize=False,
      random_state=None, solver='cholesky', tol=0.001)
Ridge Regression 5 fold Cross-Validation Results:
Scores [[228.41757545 177.22545992 220.96739757 192.37498221 235.90847068]]
Mean 210.97877716511599
Standard Deviation 22.408754778383287
--------------------------------------------------------------------------------
----------------------------------
Lasso Regression 5 fold Cross-Validation Results:
Lasso(alpha=0.1, copy_X=True, fit_intercept=True, max_iter=1000,
      normalize=False, positive=False, precompute=False, random_state=22,
      selection='cyclic', tol=0.09, warm_start=False)
Scores [[239.38792648 195.17038822 220.81849914 193.15404639 259.1010623 ]]
Mean 221.5263845060566
Standard Deviation 25.420544076574924
--------------------------------------------------------------------------------
----------------------------------
Elastic Net 5 fold Cross-Validation Results:
ElasticNet(alpha=0.1, copy_X=True, fit_intercept=True, l1_ratio=0.5,
           max_iter=1000, normalize=False, positive=False, precompute=False,
           random_state=22, selection='cyclic', tol=0.01, warm_start=False)
Scores [[228.91154503 182.52314844 213.02067397 193.27925272 220.2842728 ]]
Mean 207.60377858959833
Standard Deviation 17.195282199301207
--------------------------------------------------------------------------------
```

```
----------------------------------
RF-Regressor 5 fold Cross-Validation Results:
RandomForestRegressor(bootstrap=True, criterion='mse', max_depth=None,
                      max_features='auto', max_leaf_nodes=None,
                      min_impurity_decrease=0.0, min_impurity_split=None,
                      min_samples_leaf=1, min_samples_split=2,
                      min_weight_fraction_leaf=0.0, n_estimators=100,
                      n_jobs=None, oob_score=False, random_state=22, verbose=0,
                      warm_start=False)
Scores [[223.63810401 208.34103968 217.19849917 248.28014952 250.14372876]]
Mean 229.5203042292232
Standard Deviation 16.80622409083839
```

Elastic Net is our winner at this point in time with a mean RMSE of ~207 and std ~17 at this point the model has not achieved our goal of providing an estimate of the median rental rate that is less than 192 dollars. Lets us see if we can improve upon this.

A helpful thing to do here is to check the noise of the data and how much variance in median rental price our best model can explain currently

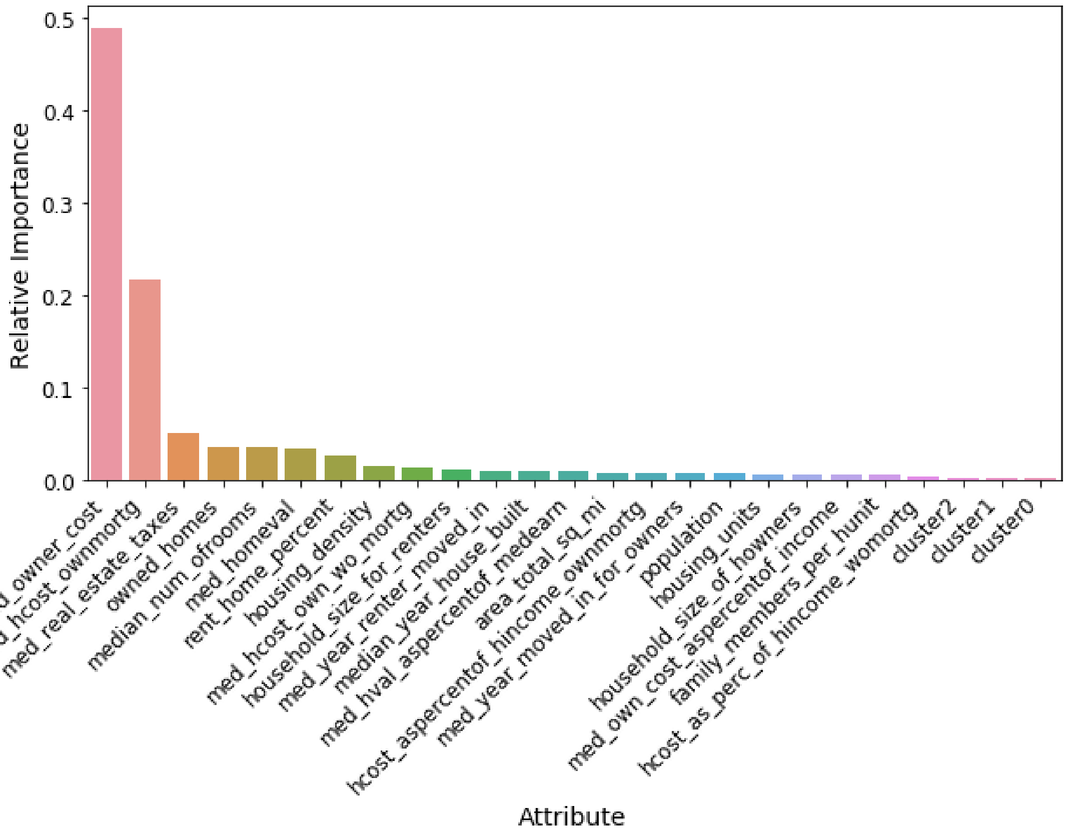
```
Explained variance with an Elastic Net regresion model
ElasticNet(alpha=0.1, copy_X=True, fit_intercept=True, l1_ratio=0.5,
           max_iter=1000, normalize=False, positive=False, precompute=False,
           random_state=22, selection='cyclic', tol=0.01, warm_start=False)
Scores [[0.83867501 0.88496919 0.8248445  0.78324435 0.74007055]]
Mean 0.8143607210858583
Standard Deviation 0.04934350514441448
```

ElasticNet accounts for ~ 81% of the variation in our given training data set with a sd of 4%

Lets investigate the relative importance of each attribute for making accurate prediction of median rental rate for feature selection purposes to see if reducing the noise i.e. dropping uninformative attributes improves our predictions.

```
(0.4887781138536719, 'med_owner_cost')
(0.21675649535699323, 'med_hcost_ownmortg')
(0.050201590357428647, 'med_real_estate_taxes')
(0.03587588836602332, 'owned_homes')
(0.0347595014957407, 'median_num_ofrooms')
(0.0324637039464469, 'med_homeval')
(0.025615844525265767, 'rent_home_percent')
(0.013820970048462571, 'housing_density')
(0.013256862434699219, 'med_hcost_own_wo_mortg')
(0.010148870136003196, 'household_size_for_renters')
(0.009420439127731117, 'med_year_renter_moved_in')
(0.008661245901148296, 'median_year_house_built')
(0.00801330904523532, 'med_hval_aspercentof_medearn')
(0.007343128332141725, 'area_total_sq_mi')
(0.007072207368963794, 'hcost_aspercentof_hincome_ownmortg')
(0.007069719982840823, 'med_year_moved_in_for_owners')
(0.005911235719694841, 'population')
```

```
(0.004938470868514768, 'housing_units')
(0.004910073403746763, 'household_size_of_howners')
(0.004566983165365608, 'med_own_cost_aspercentof_income')
(0.00409571060983884, 'family_members_per_hunit')
(0.0030364234055547264, 'hcost_as_perc_of_hincome_womortg')
(0.0018569607973755054, 'cluster2')
(0.000860147955821826, 'cluster1')
(0.0005661037952903825, 'cluster0')
Total attributes:  25
```



Median Owner cost and median housing cost for those who own a mortgage are our 2 most informative attributes in predicting the median rental rate.

What if you wanted to look at how attributes for a city contribute to our machine learning estimators model predictions for a given city. In this case the SHAP library along with our random forest model can be used to explain the estimators predictions for San Diego and Los Angeles

```
[45]: RandomForestRegressor(bootstrap=True, criterion='mse', max_depth=None,
                            max_features='auto', max_leaf_nodes=None,
                            min_impurity_decrease=0.0, min_impurity_split=None,
                            min_samples_leaf=1, min_samples_split=2,
                            min_weight_fraction_leaf=0.0, n_estimators=100,
                            n_jobs=None, oob_score=False, random_state=22, verbose=0,
                            warm_start=False)
```

For San Diego the above explanation shows features each contributing to push the model output from the base value (the average model output over the training dataset we passed) to the model output. Features pushing the prediction higher are shown in red, those pushing the prediction lower are in blue

The actual value for San Diego is

```
[47]: 1503.0
```

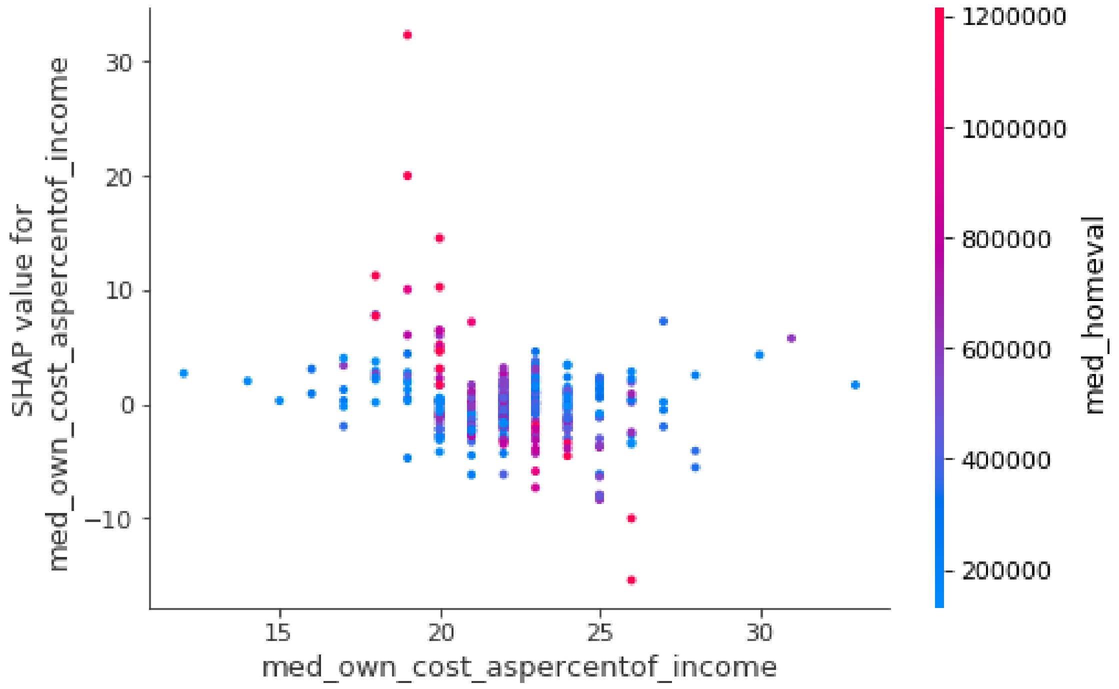
Our model is underpredicting

Now lets do the same for Los Angeles

The actual value for Los Angeles is

```
[49]: 1302.0
```

Coloring by Median Home Evaluation highlights that med_own_cost_aspercentof_income (median owner cost as percent of household income) had less impact on median rental rate for areas with a low median home evaluation even as it increased.



Using the information given to us by the random forest feature importance let us now reuse our `preparedf_or_featureselection` function to explore how many features GridSearchCV recommends.

```
[52]: 24
```

Which are:

```
[53]: ['med_owner_cost',
       'med_hcost_ownmortg',
       'med_real_estate_taxes',
       'owned_homes',
       'median_num_ofrooms',
       'med_homeval',
       'rent_home_percent',
       'housing_density',
       'med_hcost_own_wo_mortg',
       'household_size_for_renters',
       'med_year_renter_moved_in',
       'median_year_house_built',
       'med_hval_aspercentof_medearn',
       'area_total_sq_mi',
       'hcost_aspercentof_hincome_ownmortg',
       'med_year_moved_in_for_owners',
       'population',
       'housing_units',
       'household_size_of_howners',
       'med_own_cost_aspercentof_income',
       'family_members_per_hunit',
       'hcost_as_perc_of_hincome_womortg',
       'cluster2',
       'cluster1']
```

Apparently almost all features are useful except one (cluster 0). Lets recheck the models we tried out above now with our 24 features and see by having less noise i.e. reducing the number of irrelevant features the models is trying to fit we can reduce our MSE and see if there is a new winner

```
Ordinary Least Squares Linear Regression 5 fold Cross-Validation Results:
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
Scores [[228.70939878 176.52778395 224.31561971 193.71462661 239.92271662]]
Mean 212.63802913340592
Standard Deviation 23.66919595400194
--------------------------------------------------------------------------------
----------------------------------
Ridge(alpha=1, copy_X=True, fit_intercept=True, max_iter=None, normalize=False,
      random_state=None, solver='cholesky', tol=0.001)
Ridge Regression 5 fold Cross-Validation Results:
Scores [[228.33146514 177.2107947  220.96301358 192.35024036 235.92261824]]
Mean 210.95562640192261
Standard Deviation 22.406661097537945
--------------------------------------------------------------------------------
----------------------------------
```

```
Lasso Regression 5 fold Cross-Validation Results:
Lasso(alpha=0.1, copy_X=True, fit_intercept=True, max_iter=1000,
      normalize=False, positive=False, precompute=False, random_state=22,
      selection='cyclic', tol=0.09, warm_start=False)
Scores [[251.79292729 203.11322586 295.89401629 202.0268536  211.01464028]]
Mean 232.76833266566382
Standard Deviation 36.45400982258304
--------------------------------------------------------------------------------
----------------------------------
Elastic Net 5 fold Cross-Validation Results:
ElasticNet(alpha=0.1, copy_X=True, fit_intercept=True, l1_ratio=0.5,
           max_iter=1000, normalize=False, positive=False, precompute=False,
           random_state=22, selection='cyclic', tol=0.01, warm_start=False)
Scores [[228.72091745 182.30475736 212.98792569 193.02908072 220.24501657]]
Mean 207.45753955906352
Standard Deviation 17.245746459721897
--------------------------------------------------------------------------------
----------------------------------
RF-Regressor 5 fold Cross-Validation Results:
RandomForestRegressor(bootstrap=True, criterion='mse', max_depth=None,
                      max_features='auto', max_leaf_nodes=None,
                      min_impurity_decrease=0.0, min_impurity_split=None,
                      min_samples_leaf=1, min_samples_split=2,
                      min_weight_fraction_leaf=0.0, n_estimators=100,
                      n_jobs=None, oob_score=False, random_state=22, verbose=0,
                      warm_start=False)
Scores [[223.1179371  210.06569356 220.38540657 252.98851248 257.09764473]]
Mean 232.73103888747133
Standard Deviation 18.77573411858096
```

As expected reducing the number of features down by just one does not improve our RMSE

## 8 Fine-tuning models and combine them into a great solution

#### 8.0.1 Tuning the hyper-parameters of the best estimators

Elastic Net Grid Search

To understand the effects of the parameters of ElasticNet this link was very helpful!

```
[56]: GridSearchCV(cv=5, error_score='raise-deprecating',
                   estimator=ElasticNet(alpha=1.0, copy_X=True, fit_intercept=True,
                                        l1_ratio=0.5, max_iter=1000, normalize=False,
                                        positive=False, precompute=False,
                                        random_state=22, selection='cyclic',
                                        tol=0.0001, warm_start=False),
                   iid='warn', n_jobs=-1,
                   param_grid={'alpha': [0.0001, 0.001, 0.01, 0.1, 1, 10, 100],
                               'l1_ratio': array([0.  , 0.1, 0.2, 0.3, 0.4, 0.5, 0.6,
      0.7, 0.8, 0.9]),
                               'tol': array([0.01, 0.02, 0.03, 0.04, 0.05, 0.06, 0.07,
      0.08, 0.09])},
                   pre_dispatch='2*n_jobs', refit=True, return_train_score=True,
                   scoring='neg_mean_squared_error', verbose=0)
```

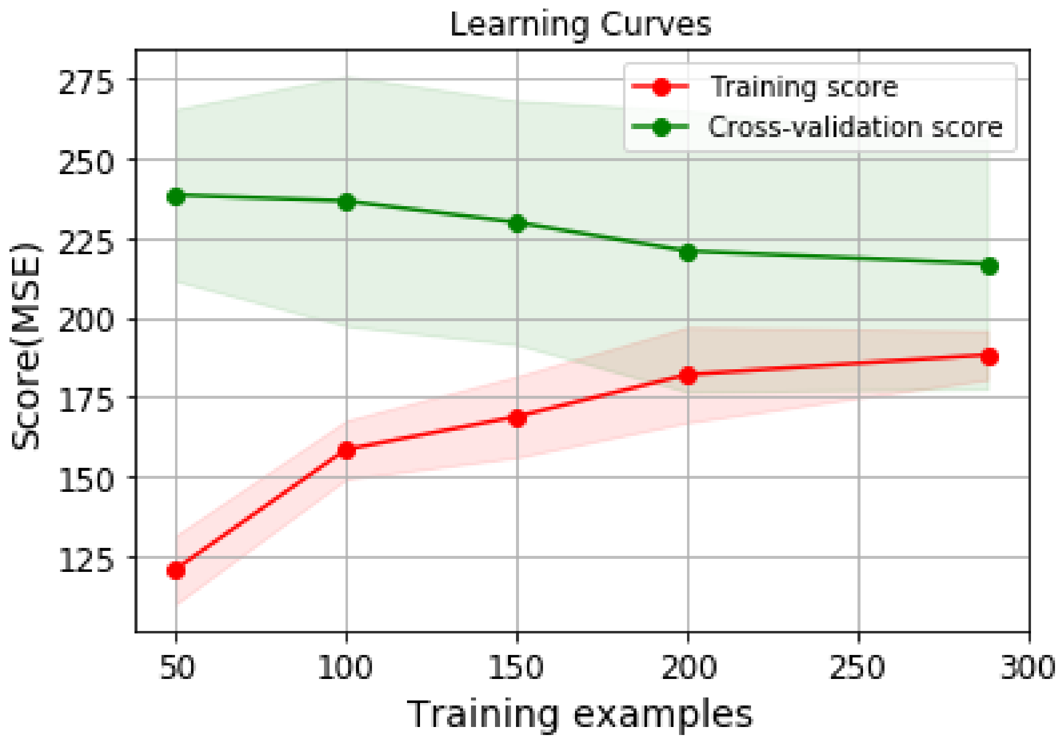
```
[57]: {'alpha': 0.1, 'l1_ratio': 0.7000000000000001, 'tol': 0.09}
```

```
[58]: ElasticNet(alpha=0.1, copy_X=True, fit_intercept=True,
                 l1_ratio=0.7000000000000001, max_iter=1000, normalize=False,
                 positive=False, precompute=False, random_state=22,
                 selection='cyclic', tol=0.09, warm_start=False)
```

```
[59]: 216.70849202484342
```

Our cross validation mean for the MSE score is actually higher lets see why that might be with a learning curve.Also note that the complete training data is scaled before cross-validation rather than scaled for each fold during cross validation which might affect our above grid search results

```
[61]: <matplotlib.legend.Legend at 0x1877421beb8>
```



Our MSE might converge to a certain limit even as we gather more training data this could mean that in order to reduce our MSE we are likely going to need to gather further variables of importance in the future.

**Lets check if ensemble methods or boosting might give some improvement**

```
A voting Regressor Ensemble(ElasticNet with Random Forest) 5 fold Cross-
Validation Results:
Scores [[216.19043793 181.9209085  203.31474648 214.55382382 227.5918772 ]]
Mean 208.71435878501106
Standard Deviation 15.449249845405017
```

An ensemble of our Elastic Net and our Random Forest model increased our RMSE slightly from ~207 to ~208 but reduces our std from ~17 to ~15

How about a Gradient Boosting Regressor

```
A Gradient Boosting Regressor 5 fold Cross-Validation Results:
Scores [[223.48480983 214.1537     210.76298904 241.86636742 228.37260652]]
Mean 223.7280945593885
Standard Deviation 11.04889689925249
```

The std for RMSE for a GradientBoostingRegressor is our lowest yet ~11 but it does have a higher mean RMSE from our lowest of 223

## 9 The System that we will evaluate on the Test Set will be the ensemble model

Our final Root Mean Squared Error is:

```
[69]: 197.90470449210102
```

Our explained variance in median rental rate by our final model is:

```
[70]: 0.8062712315772751
```

How precise is this estimate lets compute a 95% confidence interval for the generalization error

```
[71]: array([ 70.81698686, 270.77204156])
```

For future research perhaps some categorical variables should be considered after seeking domain expertise about valuable categorical variables to consider in the prediction of median rental price.

This model may not have produced the desired solution to our business problem however in this project we learned quite a lot. In the story to follow we will summarize the key findings of this project.

## 10 A story about applicable results from the project.

Bob Ross is a 40 year old college professor. He has 1 kid in college and is widowed. Unfortunately for Bob he has had a series of unexpected financial expenses that have rendered him to temporarily fall behind on his mortgage payments on his 3 bedroom house in San Luis Obispo, CA. While talking to his friend Tim about his troubles Tim tells him that he can flip the script by renting his home for a period of time. He explains that this may be a good option when two factors are present: Your home would rent for at or more than your mortgage payment and you were able to find an affordable place to stay. San Luis Obispo is a College Town and Bob believes he can rent his home during the school year while he rents a small apartment to help cover the costs before moving back in. Tim says that to determine the rental price of your property, consult directly with a real estate agent or property management company to take a look at comparable rentals in your area. But, before doing so Tim invites Bob over to do some research of their own.Tim is savvy with python and has done some research that he presents to Bob.

He explains that across California the top 3 predictors of median rental rates are monthly cost of housing for property owners including mortgage payment, taxes, insurance,and utilities, median housing cost for homeowners with a mortgage(including the cost of the mortgage or other debt), and the median real estate taxes paid by owners of homes in the area.

Additionally, Tim also shows Bob the figure below and suggest that after taking into account the variables displayed above his real estate taxes paid, home evaluation i.e. how much the property is worth if it was for sale, and housing cost(including the cost of the mortgage or other debt) contribute to increasing the rental price of his property. On the other hand it suggest his owner cost(monthly cost of housing including mortgage payment, taxes, insurance,and utilities), the number of rooms, and the fact the percent of all occupied housing units that are owned housing units is 38(%) in San Luis Obispo contribute to decreasing rental price of his property.

Tim tells Bob that he can come up with a prediction of the rental price of his property within a margin of error of ~198 dollars that takes into account several pieces of information about his property and the local real estate market in San Luis Obispo. Bob thanks Tim for his help he tells him that this will give him a better understanding of whether the quote he receives from the consultant is an accurate estimate. Bob then goes to Linda a real estate agent and rather than be in a state of unknowing he knows that the estimate Linda provides to him aligns with the prediction Tim had provided him. As a result Bob has piece of mind that he has received a fair estimate of the rental price of his property.

## 11 Acknowledgements

The community is great and I appreciate all the knowledge that is shared by incredible individuals. * Hands-On Machine Learning with Scikit-Learn & TensorFlow * https://www.analyticsvidhya.com/blog/2016/01/guide-data-exploration/#three * https://towardsdatascience.com/normalization-vs-standardization-quantitative-analysis-a91e8a79cebf * https://elitedatascience.com/primer * https://towardsdatascience.com/better-heatmaps-and-correlation-matrix-plots-in-python-41445d0f2bec * "Towncharts.com - United States Demographics Data." United States Demographics data. N.p., 15 Dec. 2016. Web. 04 Sep. 2019. http://www.towncharts.com/.