How much is your next home in California worth?

A presentation on estimating housing price in California using demographic data

By Vikraant Pai
Mtech Data Science
Roll No. D005
SAP ID: 70271019005

Problem Definition

1. The problem that is addressed here is finding the median housing price of block given other demographic parameters of that particular block
2. This analysis would give an insight on investing in a particular region in california
3. Demographic data is used that can be easily obtained using public census dataset
4. Google Colaboratory(colab) is used for experimenting with the data set
5. Thus this is a supervised learning task with median housing price as the target variable and demographic features as predictor

Get and load the data

Glimpse of the Data that is taken from the census data

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	households	median_income	median_house_value	ocean_proximity
0	-122.23	37.88	41.0	880.0	129.0	322.0	126.0	8.3252	452600.0	NEAR BAY
1	-122.22	37.86	21.0	7099.0	1106.0	2401.0	1138.0	8.3014	358500.0	NEAR BAY
2	-122.24	37.85	52.0	1467.0	190.0	496.0	177.0	7.2574	352100.0	NEAR BAY
3	-122.25	37.85	52.0	1274.0	235.0	558.0	219.0	5.6431	341300.0	NEAR BAY
4	-122.25	37.85	52.0	1627.0	280.0	565.0	259.0	3.8462	342200.0	NEAR BAY

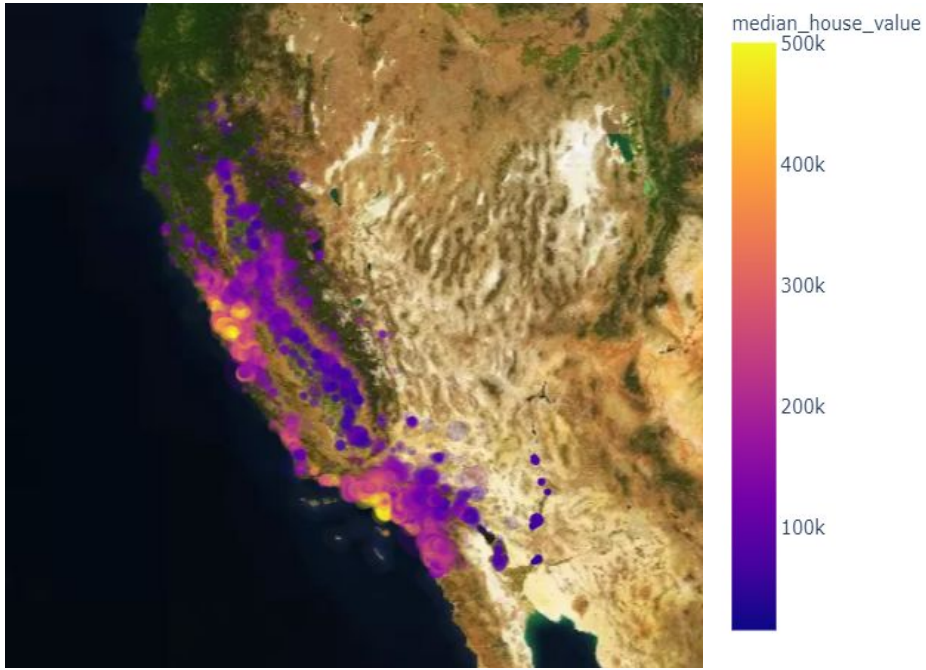
Get and load the data

1. data contains demographic and geospatial information about the region
2. It also contains information about the median Age of the house and Median number of rooms of the house which can be deciding factors for pricing a house
3. Following are summary statistics about the columns in the data set under consideration.
4. Apart from the missing value, there are no observable anomalies which means certain pre-processing is already done on the data for analysis purposes

	count	mean	std	min	25%	50%	75%	max
longitude	20640.0	-119.569704	2.003532	-124.3500	-121.8000	-118.4900	-118.01000	-114.3100
latitude	20640.0	35.631861	2.135952	32.5400	33.9300	34.2600	37.71000	41.9500
housing_median_age	20640.0	28.639486	12.585558	1.0000	18.0000	29.0000	37.00000	52.0000
total_rooms	20640.0	2635.763081	2181.615252	2.0000	1447.7500	2127.0000	3148.00000	39320.0000
total_bedrooms	20433.0	537.870553	421.385070	1.0000	296.0000	435.0000	647.00000	6445.0000
population	20640.0	1425.476744	1132.462122	3.0000	787.0000	1166.0000	1725.00000	35682.0000
households	20640.0	499.539680	382.329753	1.0000	280.0000	409.0000	605.00000	6082.0000
median_income	20640.0	3.870671	1.899822	0.4999	2.5634	3.5348	4.74325	15.0001
median_house_value	20640.0	206855.816909	115395.615874	14999.0000	119600.0000	179700.0000	264725.00000	500001.0000

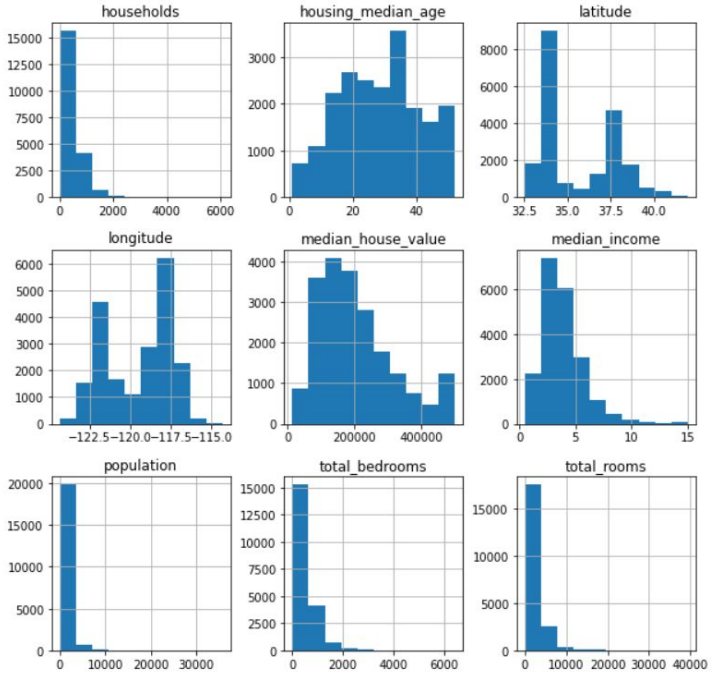
Exploring the data-Analysing the price

1. Purple regions in the adjacent map shows the regions of lower median prices of houses
2. The red and yellow regions show places with high median prices
3. Regions closer to the Bay are yellower which shows that regions closer to bay have higher median prices
4. Regions away from bay have relatively lesser cost
5. Except in the North California, where this pattern is not observed



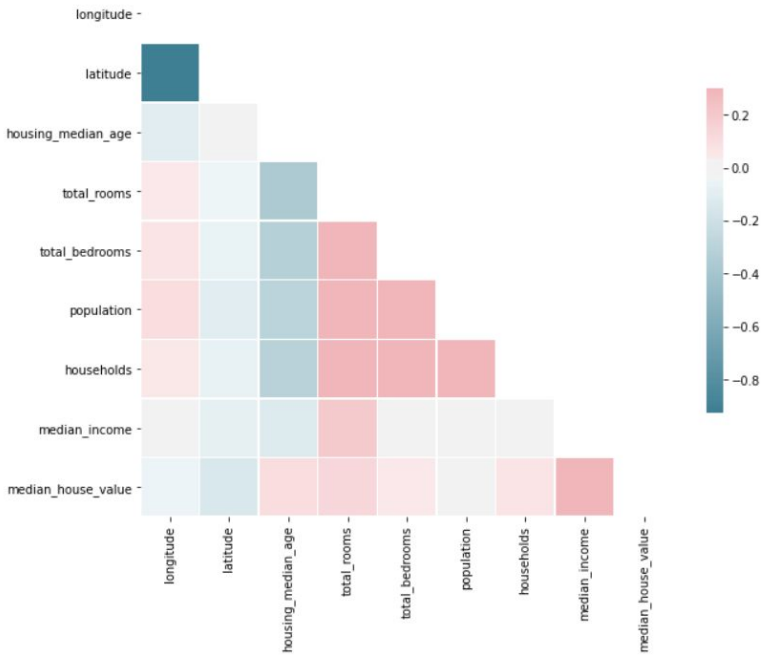
Exploring the data-Analysing the statistical distribution of data

1. Many histogram have larger tails(right skewed)
2. Some machine learning algorithms find harder to detect patterns in skewed distributions
3. We may need to apply standardization techniques to turn the distributions into bell shaped distributions



Checking for Correlations in the data

1. There are relatively low strong correlations in the dataset
2. Apart from latitude and longitude, rest of the predictors are positively correlated with the median housing price
3. Median Income has the highest positive correlation with the median housing price



Cleaning Data- Imputing Null Values

1. As per our earlier observation, we have null values in the total number of rooms.
2. We may impute it with median value of the rooms in the data set.
3. Replacing with central value will not impact the data much

```
df["total_bedrooms"].fillna(df["total_bedrooms"].median(), inplace=True)
```

```
df.isna().sum()
```

```
longitude             0
latitude              0
housing_median_age    0
total_rooms           0
total_bedrooms        0
population            0
households            0
median_income         0
median_house_value    0
ocean_proximity       0
dtype: int64
```

Cleaning Data- One Hot Encoding Categorical Values

1. Here in this step we one hot encode ocean proximity value
2. We replace the categorical data with dummy values so that the data can be fed to a model
3. We get following output after encoding the categorical data with dummy values

```
df.iloc[:, -5:-1].head()
```

	ocean_proximity_<1H OCEAN	ocean_proximity_INLAND	ocean_proximity_ISLAND	ocean_proximity_NEAR BAY
0	0	0	0	1
1	0	0	0	1
2	0	0	0	1
3	0	0	0	1
4	0	0	0	1

Cleaning Data- Standardizing the values

1. Since the values are skewed we need to transform it into bell shaped curves so that some of the ML algorithms can understand the pattern

```
[52] from sklearn.preprocessing import StandardScaler
     scaler = StandardScaler()
     df  = pd.DataFrame( scaler.fit_transform(df), columns=df.columns.values)
     df.head()
```

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	households	median_income	med
0	-1.327835	1.052548	0.982143	-0.804819	-0.972476	-0.974429	-0.977033	2.344766	
1	-1.322844	1.043185	-0.607019	2.045890	1.357143	0.861439	1.669961	2.332238	
2	-1.332827	1.038503	1.856182	-0.535746	-0.827024	-0.820777	-0.843637	1.782699	
3	-1.337818	1.038503	1.856182	-0.624215	-0.719723	-0.766028	-0.733781	0.932968	
4	-1.337818	1.038503	1.856182	-0.462404	-0.612423	-0.759847	-0.629157	-0.012881	

Train Test Split

1. We will split the data into train and test for training the model and testing it on out of the sample data

```
y = df["median_house_value"]
X = df.drop("median_house_value", axis=1)
X_train, X_test, y_train, y_test = train_test_split(X,y, train_size=0.8,random_state=123)
```

Exploring Machine Learning Models

For the given set of data, linear regression has the lowest test error

Linear Regression 

Lasso Regression

Ridge Regression

Linear Regression
RMSE ==> 68734.90050559722

	actual	predicted
368	142900.0	204039.578605
18722	94200.0	74881.743781
4854	156600.0	163138.551670
1038	107200.0	107759.078881
2618	90200.0	153316.333375
4609	316700.0	158566.829732
6106	140200.0	197537.034778
15684	500001.0	314689.449602
19628	115500.0	122812.843765
12882	118500.0	158690.496588

Lasso Regression
RMSE ==> 68735.1968763212

	actual	predicted
17974	365500.0	362186.513998
7945	219500.0	254944.502739
14353	125000.0	184770.424815
9951	267600.0	177635.691690
11510	457100.0	278207.639088
13862	119700.0	78605.177183
3639	183800.0	229535.929903
6251	163300.0	208569.169940
14658	183000.0	283545.021722
13122	160200.0	140952.343098

Ridge Regression
RMSE ==> 68735.29748177668

	actual	predicted
18447	233100.0	266240.231974
2296	74500.0	89674.966650
12275	112100.0	101668.159982
20109	137500.0	68192.548268
17206	275000.0	204344.514312
10329	270500.0	302236.588889
11767	134700.0	175672.070022
9685	152900.0	224203.646856
11513	485000.0	275225.153711
12686	82100.0	28103.394160