HOUSE PRICE PREDICTION

Machine Learning Project-Phase1

Submitted by:
Vishvesh Rao
(AM.EN.U4CSE19161)
S5 CSE B

Dataset

This is the California Housing Prices, California is a suburb in USA.The data pertains to the houses found in a given California district and some summary stats about them based on the 1990 census data. It consistis of 20640 instances

The features of this dataset are as follows

1. **longitude**: A measure of how far west a house is; a higher value is farther west
2. **latitude**: A measure of how far north a house is; a higher value is farther north
3. **housingMedianAge**: Median age of a house within a block; a lower number is a newer building
4. **totalRooms**: Total number of rooms within a block
5. **totalBedrooms**: Total number of bedrooms within a block
6. **population**: Total number of people residing within a block
7. **households**: Total number of households, a group of people residing within a home unit, for a block
8. **medianIncome**: Median income for households within a block of houses (measured in tens of thousands of US Dollars)
9. **medianHouseValue**: Median house value for households within a block (measured in US Dollars)
10. **oceanProximity**: Location of the house w.r.t ocean/sea

Data Visualisation

The dataset consists of 9 features including the target feature which is "*House Price*"

It consists of 20,640 entries/samples

Dataset description

```
print(data.DESCR)
```

```
.. _california_housing_dataset:

California Housing dataset
--------------------------

**Data Set Characteristics:**

    :Number of Instances: 20640

    :Number of Attributes: 8 numeric, predictive attributes and the target

    :Attribute Information:
        - MedInc        median income in block
        - HouseAge      median house age in block
        - AveRooms      average number of rooms
        - AveBedrms     average number of bedrooms
        - Population    block population
        - AveOccup      average house occupancy
        - Latitude      house block latitude
        - Longitude     house block longitude

    :Missing Attribute Values: None
```

Data set values

	MedInc	HouseAge	AveRooms	AveBedrms	Population	AveOccup	Latitude	Longitude	House Price
0	8.3252	41.0	6.984127	1.023810	322.0	2.555556	37.88	-122.23	4.526
1	8.3014	21.0	6.238137	0.971880	2401.0	2.109842	37.86	-122.22	3.585
2	7.2574	52.0	8.288136	1.073446	496.0	2.802260	37.85	-122.24	3.521
3	5.6431	52.0	5.817352	1.073059	558.0	2.547945	37.85	-122.25	3.413
4	3.8462	52.0	6.281853	1.081081	565.0	2.181467	37.85	-122.25	3.422

<u>Exploratory Data Analysis</u>

Here we will look at each of the 8 individual features in detail analysing various statistical parameters and bar graphs.

This will help us in deciding on data cleaning i.e which features to remove and also analyse the relationship between all the different features.

1 MedInc

VALUES:	20,640 (100%)	MAX	15.0	RANGE	14.5
MISSING:	---	95%	7.3	IQR	2.18
DISTINCT:	12,928 (63%)	Q3	4.7	STD	1.90
ZEROES:	---	AVG	3.9	VAR	3.61
		MEDIAN	3.5	KURT.	4.95
		Q1	2.6	SKEW	1.65
		5%	1.6	SUM	79,891
		MIN	0.5		

2 HouseAge

VALUES:	20,640 (100%)	MAX	52.0	RANGE	51.0
MISSING:	---	95%	52.0	IQR	19.0
DISTINCT:	52 (<1%)	Q3	37.0	STD	12.6
ZEROES:	---	MEDIAN	29.0	VAR	158
		AVG	28.6	KURT.	-0.801
		Q1	18.0	SKEW	0.060
		5%	8.0	SUM	591k
		MIN	1.0		

3 AveRooms

VALUES:	20,640 (100%)	MAX	142	RANGE	141
MISSING:	---	95%	8	IQR	1.61
DISTINCT:	19,392 (94%)	Q3	6	STD	2.47
ZEROES:	---	AVG	5	VAR	6.12
		MEDIAN	5	KURT.	879
		Q1	4	SKEW	20.7
		5%	3	SUM	112k
		MIN	1		

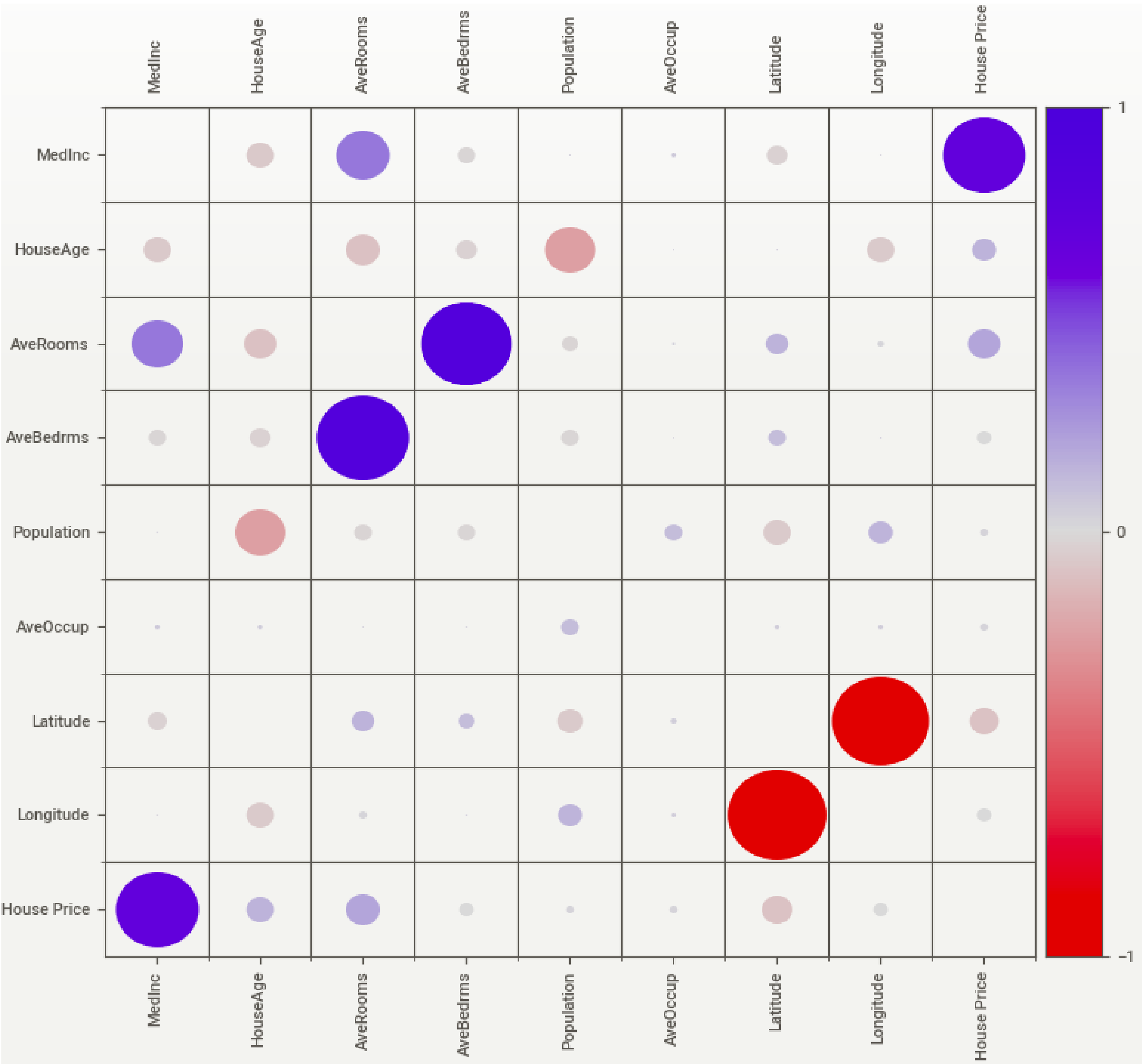
4 AveBedrms

VALUES:	20,640 (100%)	MAX	34.1	RANGE	33.7
MISSING:	---	95%	1.3	IQR	0.093
DISTINCT:	14,233 (69%)	Q3	1.1	STD	0.474
ZEROES:	---	AVG	1.1	VAR	0.225
		MEDIAN	1.0	KURT.	1,637
		Q1	1.0	SKEW	31.3
		5%	0.9	SUM	22,635
		MIN	0.3		

Looking at the above analysis we can infer that some features do not have much distinct values and therefore can be omitted.

Feature 2, Feature 8, Feature 9 can be neglected as we can see that their values are not distinct with 1%, 4%, 4% percent distictintion respectively which is very low.

<u>*Correlation Between Features*</u>



In this correlation plot, we can see that any positive value ie. Blue colored cirlces shows that an attribute is more dependent on the other attribute and red circles indicate a negetive value indicating lower dependency between the features.

Data Cleaning

Since not all features are relevant for our study we can ommit certain features based on parameters such as the level of distinction (i.e the varience in data values of a particular feature) and also count of null values in a particular feature.

Null Values in Features

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20640 entries, 0 to 20639
Data columns (total 9 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   MedInc       20640 non-null  float64
 1   HouseAge     20640 non-null  float64
 2   AveRooms     20640 non-null  float64
 3   AveBedrms    20640 non-null  float64
 4   Population   20640 non-null  float64
 5   AveOccup     20640 non-null  float64
 6   Latitude     20640 non-null  float64
 7   Longitude    20640 non-null  float64
 8   House Price  20640 non-null  float64
dtypes: float64(9)
memory usage: 1.4 MB
```

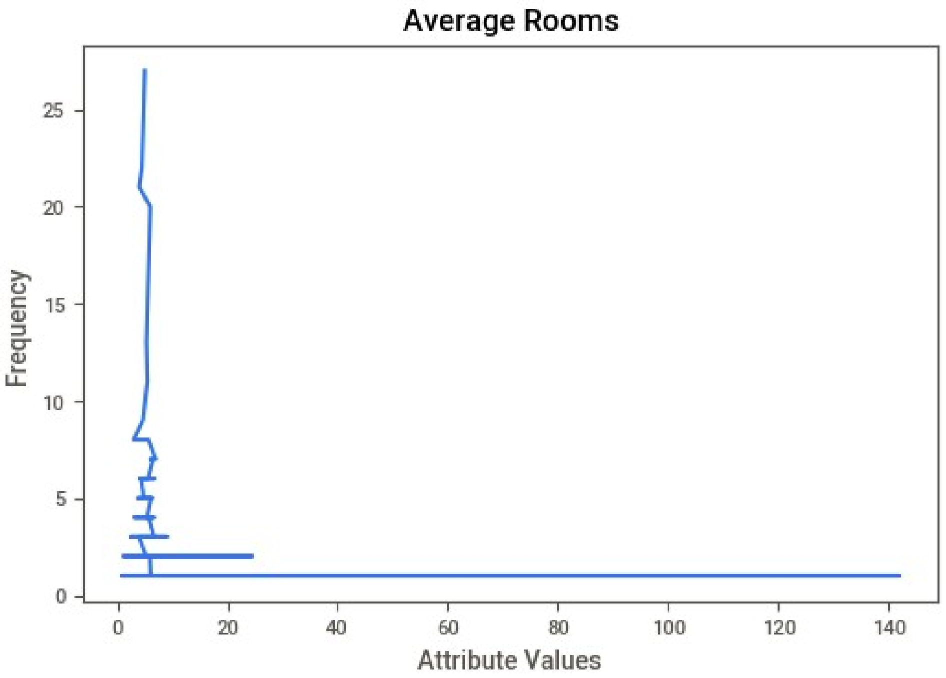
Since no null values are present here we can move on to the next criteria .

<u>*Varience in features*</u>

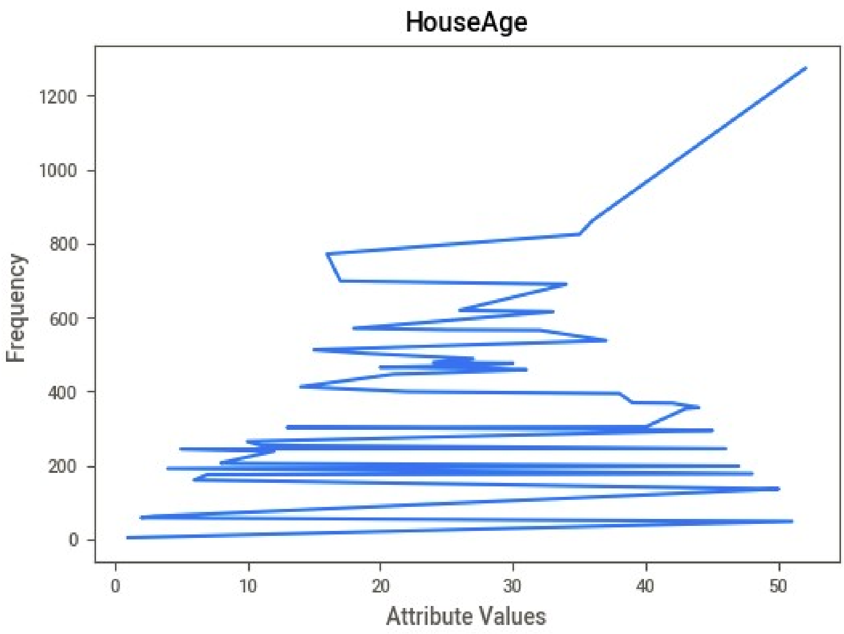
As we saw in last section HouseAge was the feature with least distinct percentage value at 1% along with Latitude and Longitude at 4% so we can drop these features

Below are two graphs illustrating the frequency of values in the feature having the least and most number of distinct values

<u>*Most no. Of distinct values*</u>



Least no. Of distinct values



Feature Reduction/Modification

The dataset has around 8 columns , which can certainly be reduced to a lower dimension as we saw in last section. For this we perform Feature Reduction to reduce the dimension of the dataset.

At the same time all though the latitude and Longitude have less distinct values we can get the exact street and road address of each house from these two features and the road and county might have a bit more variation.

Using geoloaction module we are able to get these two new attributes “*Road*” and “*County*”

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20640 entries, 0 to 20639
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   road    19507 non-null  object
 1   county  15368 non-null  object
dtypes: object(2)
memory usage: 322.6+ KB
```

dataset consisting only of road and county values which will be added to maain datset after further processing

<u>Inputting Missing Values</u>

As we can see some missing values are there in this we use logistic regression to fill up these missing values

Specifically we will be using SGD Classifier for for predicting the missing values

SGD is Stochastic Gradient Descent Classifier

Predicting missing Road values

```
[17] ## applying classification algorithm [ logistic regression ] to find missing road values

     missing_idx = []

     for i in range(df.shape[0]):

       if df["road"][i] is None:
         missing_idx.append(i)

     ## Independent Parameters
     missing_road_X_train = np.array([ [df["MedInc"][i], df["AveRooms"][i], df["AveBedrms"][i] ] for i in range(df.shape[0]) if i not in missing_idx ])

     ## Dependend Parameters
     missing_road_Y_train = np.array([  df["road"][i] for i in range(df.shape[0]) if i not in missing_idx ])

     missing_road_X_test = np.array([ [df["MedInc"][i], df["AveRooms"][i], df["AveBedrms"][i] ] for i in range(df.shape[0]) if i not in missing_idx ])
```

```
[18] from sklearn.linear_model import SGDClassifier

     # ## Model Initialisation

     model_1 = SGDClassifier()

     # ## Model Training
     model_1.fit(missing_road_X_train, missing_road_Y_train)

     missing_road_Y_pred = model_1.predict(missing_road_X_test)
```

We use the similar approach for filling in missing *County* values.

In the end no null values are presnt in dataset

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 20640 entries, 7273 to 2406
Data columns (total 8 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   MedInc       20640 non-null  float64
 1   AveRooms     20640 non-null  float64
 2   AveBedrms    20640 non-null  float64
 3   Population   20640 non-null  float64
 4   AveOccup     20640 non-null  float64
 5   House Price  20640 non-null  float64
 6   road         20640 non-null  object
 7   county       20640 non-null  int64
dtypes: float64(6), int64(1), object(1)
memory usage: 2.0+ MB
```