CS 488/508: Introduction to Data Mining

Final Report

Project Title: House Price Prediction California
Instructor: Tuan Le
Number of students: 4

Student Name	ID Number
Pravallika Bonthu	800807440
Aditya Bandaru	800811963
Sai Vivek Kodam	800810818
Humesh Reddy Venkatapuram	800810540

• **Data**
• **What kind of data?**
Kaggle: https://www.kaggle.com/datasets/camnugent/california-housing-prices
The numerical data is used as input features to predict the target variable, which is **'median_house_value'** The categorical data ('ocean_proximity') is typically one-hot encoded for use in the model.
• **Dataset information such as number of instances, number of attributes, type of attributes**
About the dataset:

1. 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
2. **Number of Instances:** The number of instances in the dataset is 20641. Each row represents a distinct housing unit in California.

	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	households	median_income	ocean_proximity	median_house_value
12851	-121.37	38.68	29.0	3757.0	646.0	2022.0	611.0	3.5429	INLAND	88200.0
19598	-120.91	37.57	26.0	3396.0	705.0	2446.0	694.0	2.0521	INLAND	65400.0
8945	-118.47	34.01	27.0	1782.0	471.0	837.0	422.0	3.7727	<1H OCEAN	413000.0
12186	-117.34	33.71	10.0	2591.0	486.0	1255.0	425.0	3.1513	<1H OCEAN	154300.0
19324	-122.97	38.53	48.0	3939.0	860.0	1257.0	571.0	2.1165	<1H OCEAN	98700.0
...	...	...	...	...	...	...	...	...	...	...
19577	-120.75	37.69	24.0	2282.0	423.0	1167.0	398.0	3.8214	INLAND	116100.0
19798	-123.12	40.54	23.0	1091.0	217.0	539.0	201.0	1.8696	INLAND	61500.0
13772	-117.01	34.01	15.0	5592.0	891.0	2419.0	840.0	4.7193	INLAND	135200.0
15210	-117.09	32.99	16.0	2175.0	327.0	1037.0	326.0	5.1909	<1H OCEAN	201400.0
10981	-117.84	33.76	26.0	2110.0	409.0	1146.0	407.0	4.3698	<1H OCEAN	229600.0

16346 rows × 10 columns

3. **Number of Attributes:** The dataset has 10 attributes or columns, each providing information about different aspects of the housing units. 'longitude,' 'latitude,' 'housing_median_age,' 'total_rooms,' 'total_bedrooms,' 'population,' 'households,' 'median_income,' ,'median_house_value', ‘ocean_proximity
4. **Type of attributes:** The attributes can be categorized into numerical and categorical types.
    - **Numerical Attributes**: 'longitude,' 'latitude,' 'housing_median_age,' 'total_rooms,' 'total_bedrooms,' 'population,' 'households,' 'median_income,' and 'median_house_value' are continuous numerical attributes.
    - **Categorical Attribute:** 'ocean_proximity' is a categorical attribute representing the proximity of houses to the ocean.

• How do you process your data?

Data processing steps: It involves several steps such as log-transforming certain numerical attributes, one-hot encoding the categorical attribute, and creating additional features like bedroom_ratio and household_rooms

1. **Handling Missing Data:** Identify missing values in the dataset. Here we remove any rows in the data that have missing values and updates data in place by removing those rows.

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 20433 entries, 0 to 20639
Data columns (total 10 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   longitude           20433 non-null  float64
 1   latitude            20433 non-null  float64
 2   housing_median_age  20433 non-null  float64
 3   total_rooms         20433 non-null  float64
 4   total_bedrooms      20433 non-null  float64
 5   population          20433 non-null  float64
 6   households          20433 non-null  float64
 7   median_income       20433 non-null  float64
 8   median_house_value  20433 non-null  float64
 9   ocean_proximity     20433 non-null  object
dtypes: float64(9), object(1)
memory usage: 1.7+ MB
```

2. Encoding Categorical Variables:

Convert categorical variables, like ocean_proximity, into numerical format since most machine learning algorithms require numerical input.

Here we have used one-hot encoding Techniques i.e., (creating binary columns for each category).

```
train_data
```

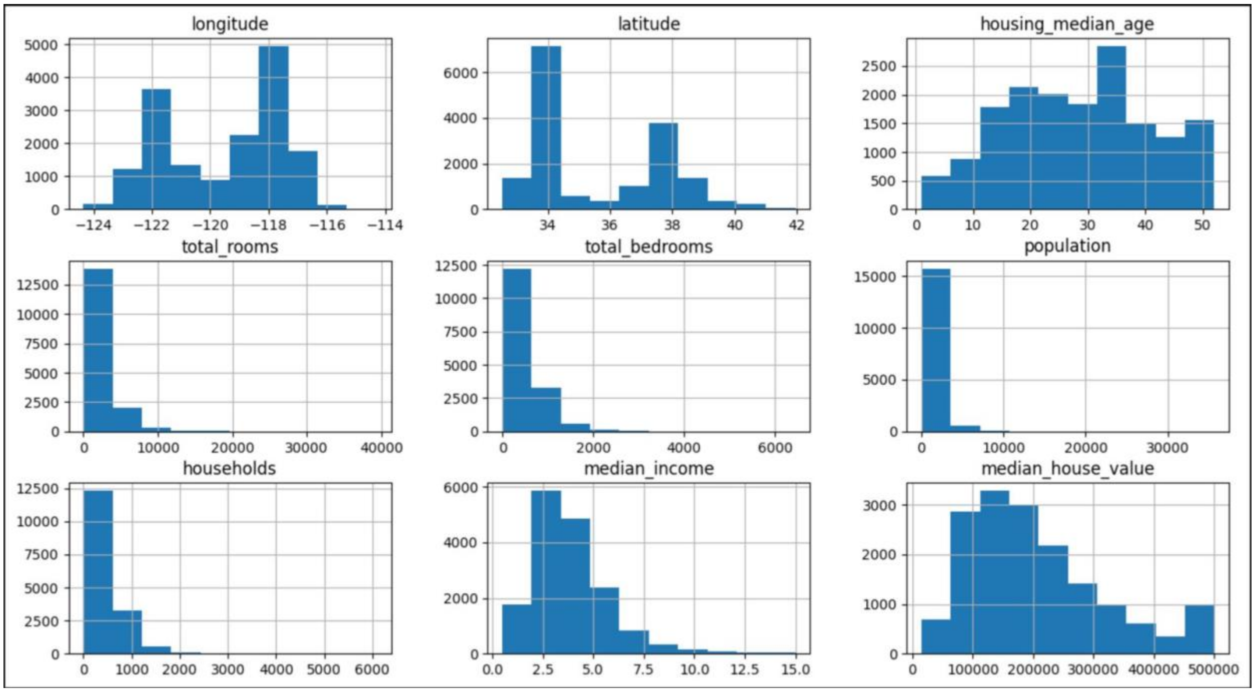
de	latitude	housing_median_age	total_rooms	total_bedrooms	population	households	median_income	median_house_value	<1H OCEAN	INLAND	ISLAND	NEAR BAY	NEAR OCEAN
37	38.68	29.0	8.231642	6.472346	7.612337	6.416732	3.5429	88200.0	0	1	0	0	0
91	37.57	26.0	8.130648	6.559615	7.802618	6.543912	2.0521	65400.0	0	1	0	0	0
47	34.01	27.0	7.486053	6.156979	6.731018	6.047372	3.7727	413000.0	1	0	0	0	0
34	33.71	10.0	7.860185	6.188264	7.135687	6.054439	3.1513	154300.0	1	0	0	0	0
97	38.53	48.0	8.278936	6.758095	7.137278	6.349139	2.1165	98700.0	1	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
75	37.69	24.0	7.733246	6.049733	7.063048	5.988961	3.8214	116100.0	0	1	0	0	0
12	40.54	23.0	6.995766	5.384495	6.291569	5.308268	1.8696	61500.0	0	1	0	0	0
01	34.01	15.0	8.629271	6.793466	7.791523	6.734592	4.7193	135200.0	0	1	0	0	0
09	32.99	16.0	7.685244	5.793014	6.945051	5.789960	5.1909	201400.0	1	0	0	0	0
84	33.76	26.0	7.654917	6.016157	7.044905	6.011267	4.3698	229600.0	1	0	0	0	0

lumns

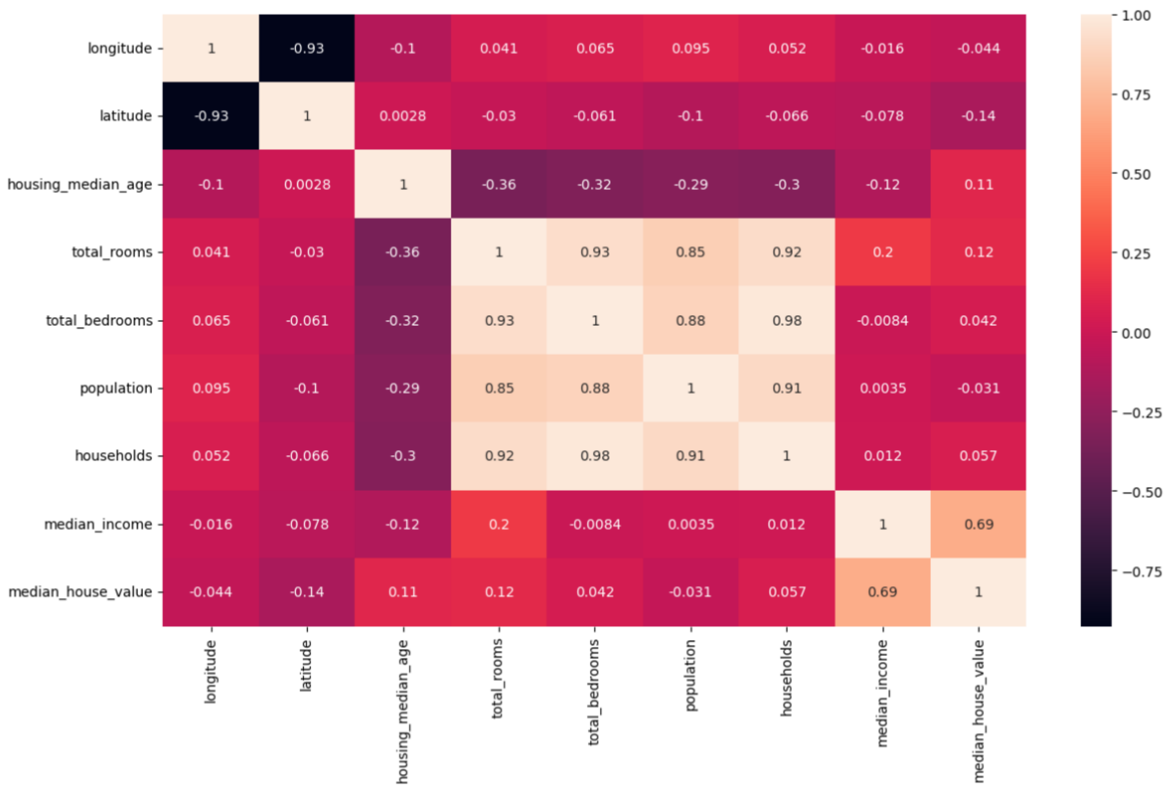
3. Data Splitting:

Split the dataset into training and testing sets to assess the model's performance accurately. Here we have split the data into 80% for training and rest for the Testing.

4. **Data Visualization:** For Data visualization we used histogram. This plot will display histograms for all the numerical attributes in ‘train_data’.



For the Visualizing the relationships between attributes we used heatmap. This will tell you to quickly identify which attributes are strongly related and which are not, which can be helpful in features selection and understanding the data’s pattern.



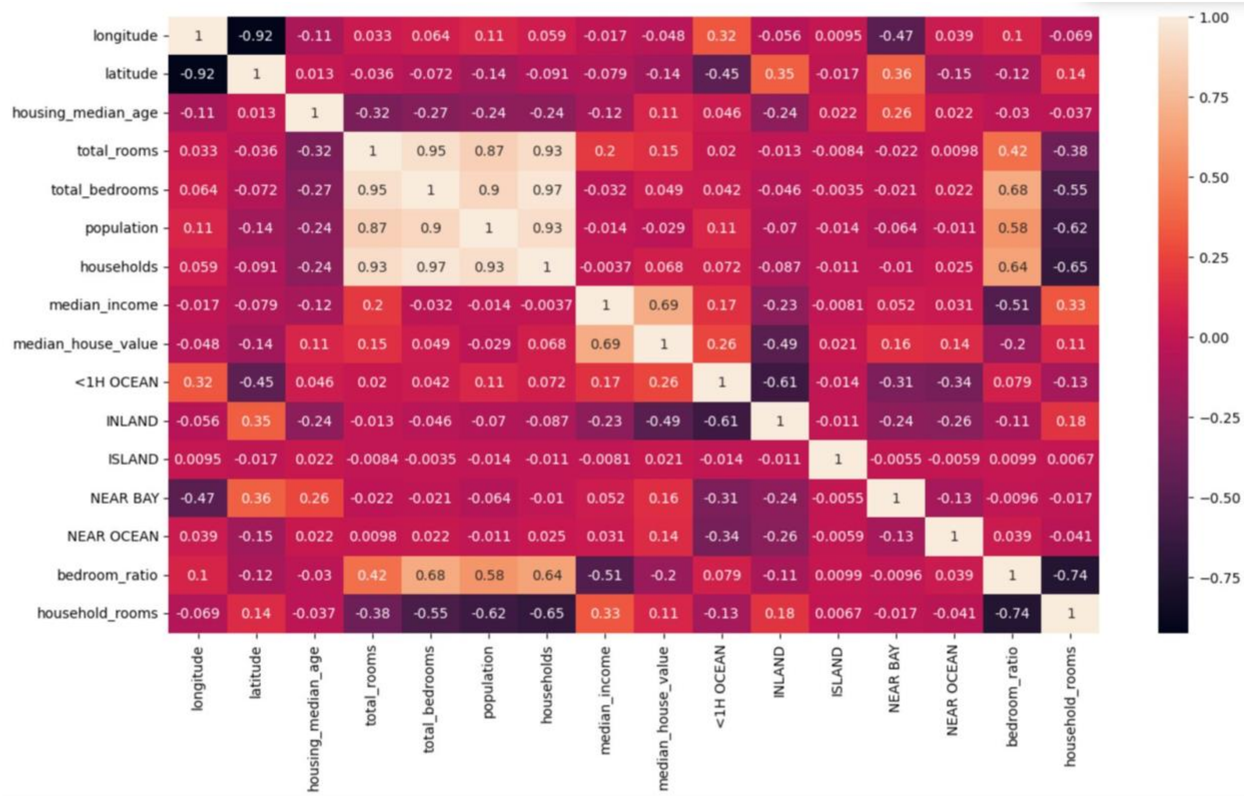
• **Which attributes you use and which one you don't use? Why?**

Attributes: we are using the following attributes - 'longitude','latitude','housing_median_age','total_rooms','total_bedrooms','population',' house','median_income',’ocean_proximity’

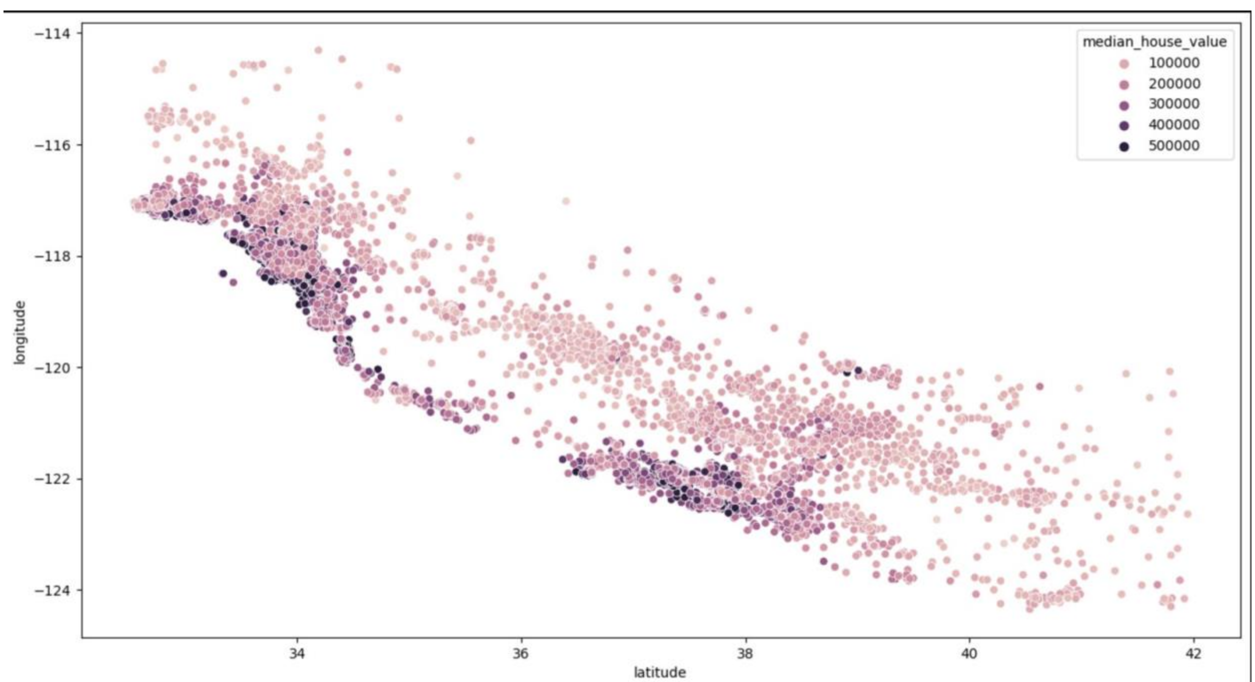
Don't use:

We are not using ‘**median_house_value’** as a feature attribute because it is typically the target variable in a regression task, not a feature for the prediction.

After performing the pre-processing. We did log transformation to alter the distribution of the data to make more suitable for certain types of analysis or modeling.



This scatterplot helps to visualize the geographical distribution of house prices based on latitude and longitude. It allows you to observe patterns or clusters in house values across different regions. The use of hue to represent **'median_house_value'** adds an extra dimension to the plot by indicating how house values vary spatially.



- **Data mining task**

What task? Classification, Clustering, Anomaly detection

In this data mining project, our primary objective is to leverage the California housing prices dataset to perform a regression analysis. The focus of our task is to predict the future prices of housing units based on historical data from previous years. To achieve this, we will be implementing various data mining techniques and machine learning algorithms.

Regression is the chosen data mining task as it aligns with our objective of estimating median house prices, which are continuous numerical values. Our goal is to build a predictive model that can accurately forecast housing prices for various units within California. This task involves extracting valuable insights from the dataset and using those insights to make informed predictions.

Which algorithms you have tried?

In this Project as now, we have used.

Linear Regression: Linear regression is simple to use where the goal is to predict a continuous numerical value.

Training Data: The 'train_data' DataFrame is used for training the linear regression model. It contains features such as longitude, latitude, housing attributes, and one-hot encoded 'ocean_proximity' values, as well as the target variable **'median_house_value**.'

Model Fitting: The code uses the **LinearRegression** class from Scikit-Learn to create a linear regression model the fit method is then used to fit the model to the training data. During this fitting process, the algorithm determines the linear relationship between the feature variables and the target variable **'median_house_value**'. The model finds the coefficients for each feature that minimize the difference between the predicted values and the actual values in the training data.

Testing and Predictions: Once the model is trained, it can be used to make predictions. 'x_test' is used to represent the test data and the trained **'reg'** model is used to predict the values of **'median_house_value'** based on the test data.

Evaluation: After making predictions, we evaluated the model's performance by comparing the predicted values to the actual values in 'y_test.'

Common evaluation metrics for regression tasks Mean Absolute Error (MAE), Mean Squared Error (MSE), and R-squared (R2).

KNN:

The K-Nearest Neighbors (KNN) algorithm is a simple, non-parametric algorithm used for classification and regression. In regression tasks, like the one described in our code, KNN predicts the output based on the average of the 'k' nearest neighbors' values.

we have used the Number of neighbors (n_neighbors=6)

Decision Tree:

A Decision Tree is a flowchart-like tree structure where an internal node represents a feature (or attribute), the branch represents a decision rule, and each leaf node represents the outcome. In the context of regression, a Decision Tree splits the data into distinct subsets based on different values of the input features.

we have used the following parameters:
Max_depth: 10
Min_samples_split:5
Min_samples_leaf:2

GradientBoost:
The Gradient Boosting Regressor is an ensemble learning technique that builds a predictive model in a stage-wise fashion. It constructs new models that predict the residuals or errors of prior models and then combines them to make the final prediction.
We have used the following parameters:
N_estimators:100
Learning_rate:0.1
Max_depth:5

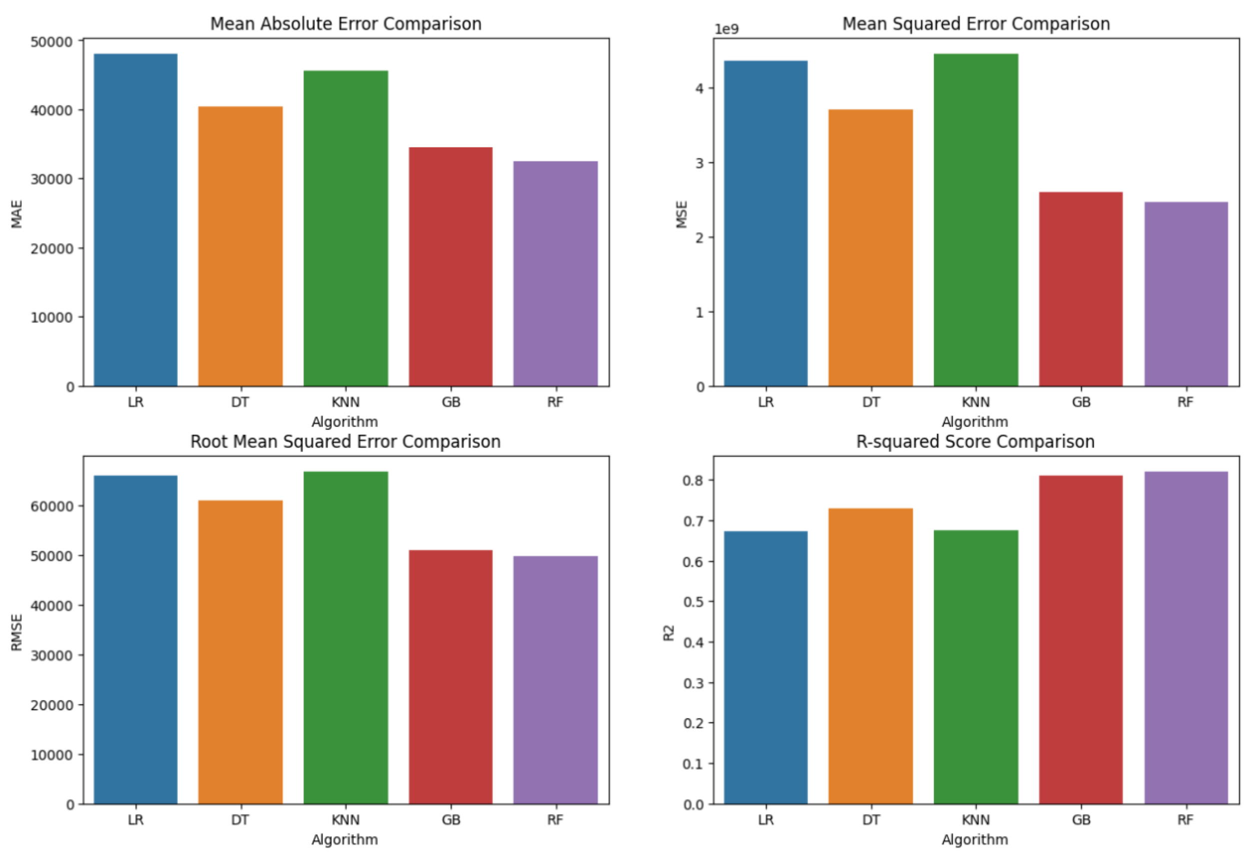
Random Forest:
The Random Forest Regressor is an ensemble learning method that operates by constructing multiple decision trees during training and outputting the mean prediction of the individual trees to form a more accurate and robust prediction. It handles both continuous and categorical data well.
We have used the following parameters:
N_estimators:150
Max_depth:15

For all these algorithms we have used several values of these parameters and came up with these values after doing several trial-and-error operations to get the best fit for the given dataset

- **How are the Experiment results ?**

We have got almost similar results for Gradient boost Regressor and Random Forest. With Random Forest having a slightly better R2 score than Gradient Boost

	Mean Absolute Error	Mean Squared Error	Root Mean Squared Error	R-squared (R2) Score	RunTime
Linear Regressor	47248.91870178438	4280535442.226238	65425.8010438255	0.67	0.074846 seconds
Decision Tree	40409.91	3707728515.50	60891.12	0.73	0.177644 seconds
KNN	45665.59	4444095639.64	66664.05	0.68	0.078018 seconds
Gradient Boost Regressor	34501.13	2595738884.27	50948.39	0.81	7.043264 seconds
Random Forest Regressor	32547.35	2467128404.95	49670.20	0.82	17.224092 seconds



References:

https://www.kaggle.com/datasets/camnugent/california-housing-prices

https://cse.msu.edu/~ptan/dmbook/software/

https://scikit-learn.org/stable/supervised_learning.html#supervised-learning