PREDICTING HOUSE PRICE USING MACHINE LEARNING

PHASE-3

Development Part 1

INTRODUCTION:

- Predicting house prices using machine learning is a common and practical application of data science in the real estate industry.
- It involves using historical data and various machine learning algorithms to create models that can predict the selling price of houses based on their features.
- Here, They were gave the dataset which contains 5001 datas. Which was in csv format. In this phase we have to building our project by loading and preprocessing the dataset and Perform different analysis as needed.

STEPS INVOLVED IN PHASE-3

STEP 1: IMPORT LIBRARIES

Import all the necessary libraries what we need. Here we have to install Pandas, Seaborn, Sklearn, Matplotlib.

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt
import xgboost as xgb
from sklearn.metrics import mean_squared_error
from sklearn.preprocessing import LabelEncoder
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.metrics import mean_squared_error
import seaborn as sns
```

STEP 2: LOAD THE DATASET

Load the dataset what they gave into pandas dataframe.

```
data = pd.read_csv('USA_Housing.csv')
```

STEP 3: DATA PREPROCESSING

- DATA CLEANING **:** Handle missing values: Fill missing values using mean, median, or advanced imputation techniques.

```
[38] data.dropna(subset=['Avg. Area Income', 'Avg. Area House Age'], inplace=True)
     data['Avg. Area Income'].fillna(data['Avg. Area Income'].mean(), inplace=True)
     data['Avg. Area House Age'].fillna('Unknown', inplace=True)
```

```
mean_price = data['Price'].mean()
median_price = data['Price'].median()

print('Mean House Price:', mean_price)
print('Median House Price:', median_price)
```

```
Mean House Price: 1232072.654142357
Median House Price: 1232669.3779657914
```

- DATA TRANSFORMATION **:** Convert categorical variables into numerical format (If required)

```
obj = (data.dtypes == 'object')
object_cols = list(obj[obj].index)
print("Categorical variables:",len(object_cols))

int_ = (data.dtypes == 'int')
num_cols = list(int_[int_].index)
print("Integer variables:",len(num_cols))

fl = (data.dtypes == 'float')
fl_cols = list(fl[fl].index)
print("Float variables:",len(fl_cols))
```

```
Categorical variables: 1
Integer variables: 0
Float variables: 6
```

using LabelEncoder

```
from sklearn.preprocessing import LabelEncoder
labelencoder=LabelEncoder()
for columns in data.columns:
  data[columns]=labelencoder.fit_transform(data[columns])
print(data.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5000 entries, 0 to 4999
Data columns (total 7 columns):
 #   Column                        Non-Null Count  Dtype
---  ------                        --------------  -----
 0   Avg. Area Income              5000 non-null   int64
 1   Avg. Area House Age           5000 non-null   int64
 2   Avg. Area Number of Rooms     5000 non-null   int64
 3   Avg. Area Number of Bedrooms  5000 non-null   int64
 4   Area Population               5000 non-null   int64
 5   Price                         5000 non-null   int64
 6   Address                       5000 non-null   int64
dtypes: int64(7)
memory usage: 273.6 KB
None
```

- REMOVE DUPLICATES : Check for and remove duplicate records if present.

```
39] data.drop_duplicates(inplace=True)
```

```
[7] print(data.isnull().sum())
```

```
Avg. Area Income                0
Avg. Area House Age             0
Avg. Area Number of Rooms       0
Avg. Area Number of Bedrooms    0
Area Population                 0
Price                           0
Address                         0
dtype: int64
```

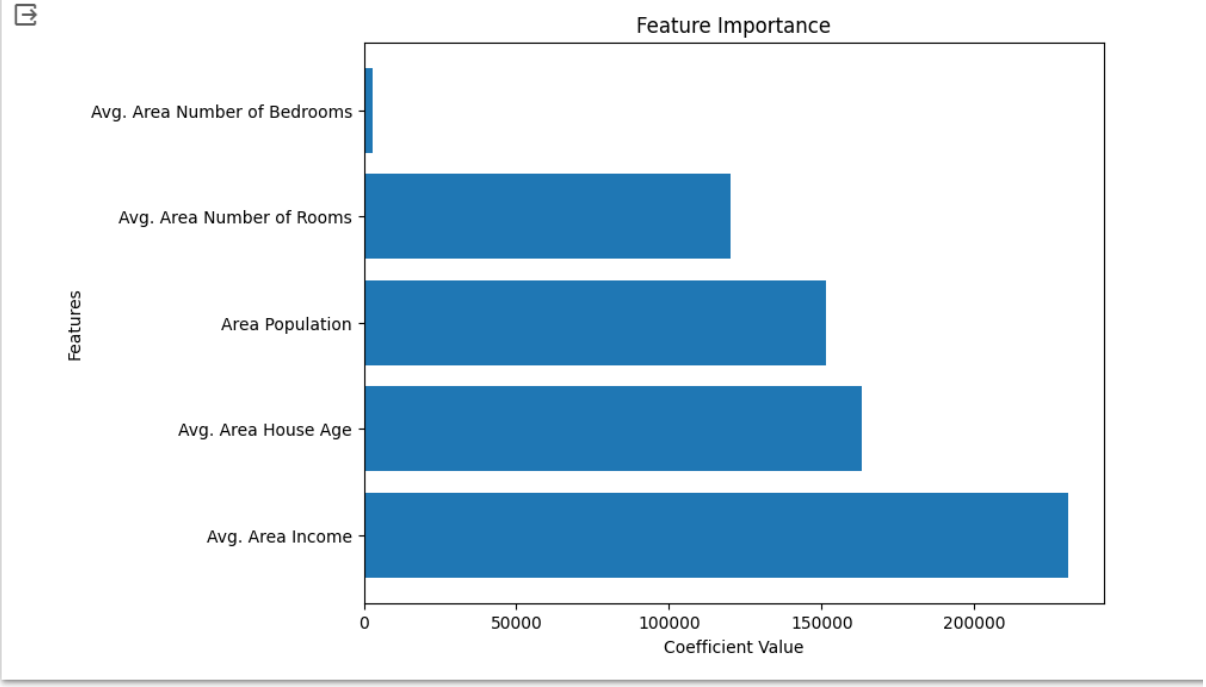
STEP 3 : DATA VISUALIZATION

Data visualization is the presentation of data in a graphical or pictorial format. It enables people to understand complex patterns, trends, and insights from data by representing it visually. Through data visualization, large and complex datasets can be represented in a clear, concise, and meaningful way. It can be

- ❖ Bar Charts
- ❖ Line Charts
- ❖ Pie Charts
- ❖ Scatter Plots
- ❖ Histograms
- ❖ Heat maps
- ❖ Box Plots

BAR CHART:

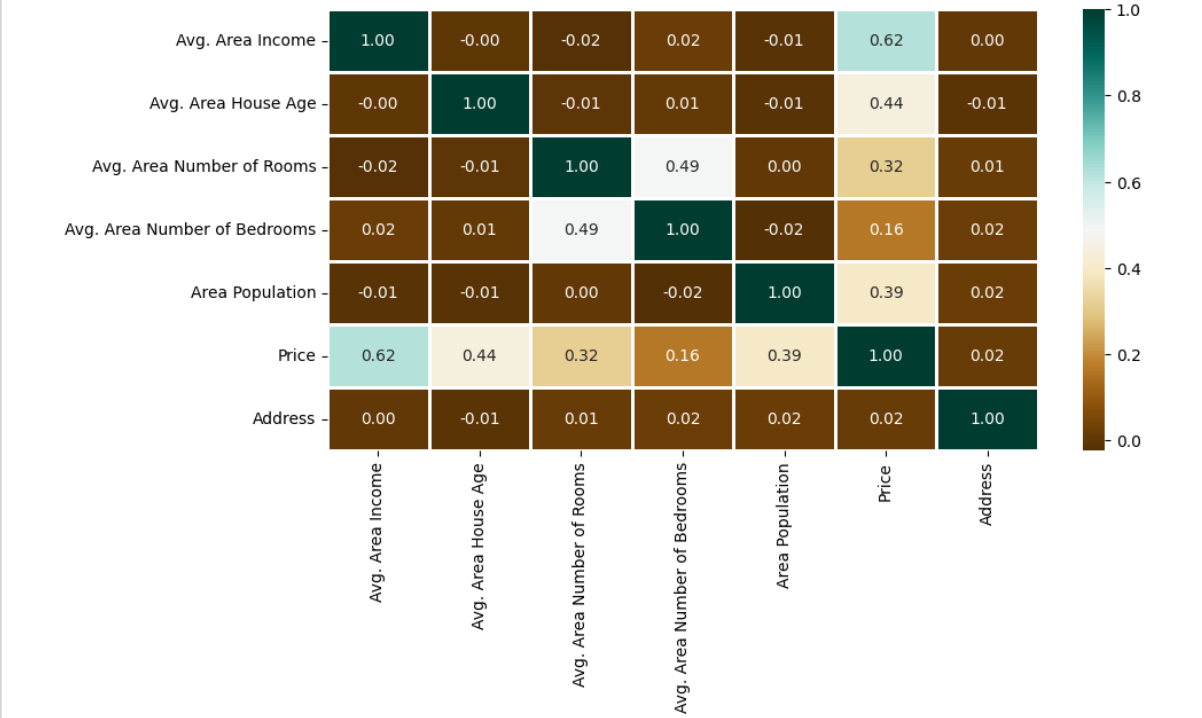
```
plt.figure(figsize=(8, 6))
plt.barh(feature_importance['Feature'], feature_importance['Coefficient'])
plt.xlabel('Coefficient Value')
plt.ylabel('Features')
plt.title('Feature Importance')
plt.show()
```



HEAT MAP:

```
plt.figure(figsize=(10, 5))
sns.heatmap(data.corr(),
        cmap = 'BrBG',
        fmt = '.2f',
        linewidths = 2,
        annot = True)
```

```
<Axes: >
```

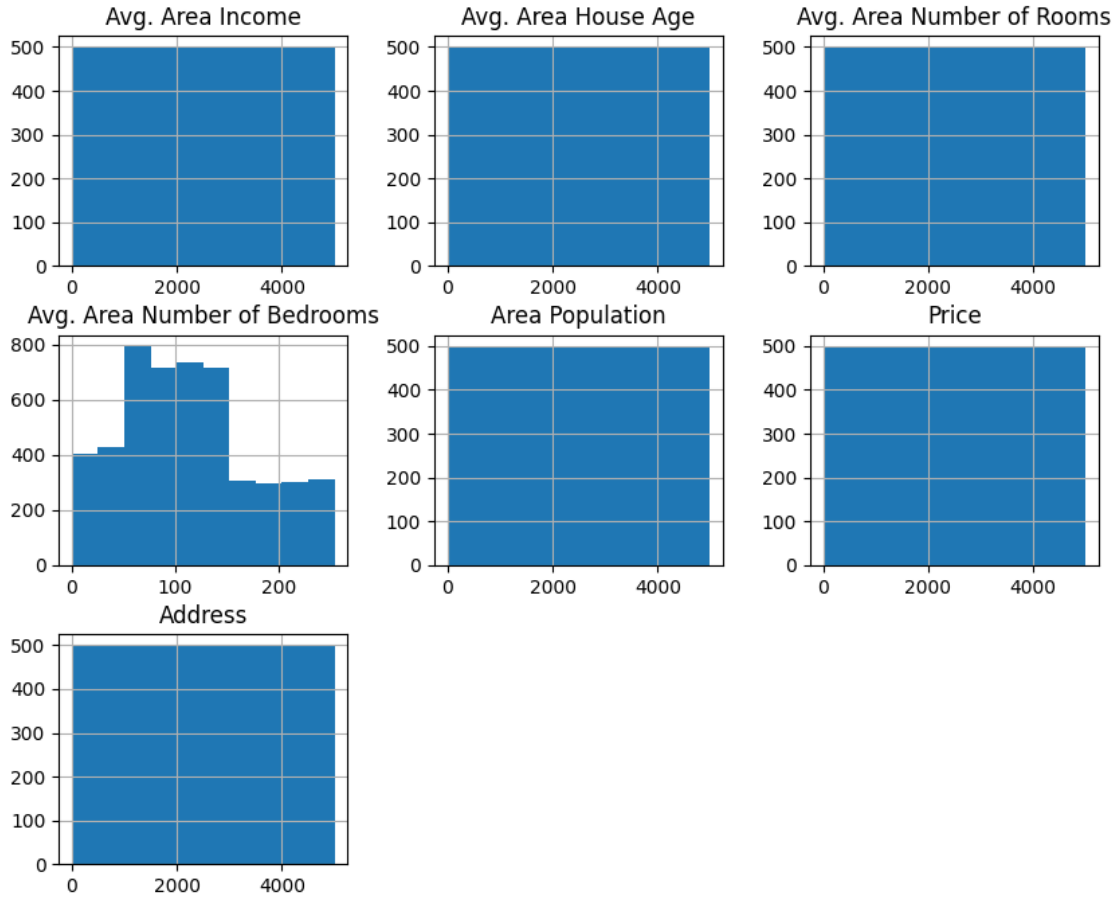


- ❖ This code uses Matplotlib and Seaborn to create a heatmap of the correlation matrix of your dataset.
- ❖ The code generates a heatmap that visually represents the correlation between pairs of variables in your dataset. Positive correlations are shown in one color, whilenegative correlations are shown in another color.

HISTOGRAM:

```
data.hist(figsize=(10,8))
```

```
array([[<Axes: title={'center': 'Avg. Area Income'}>,
        <Axes: title={'center': 'Avg. Area House Age'}>,
        <Axes: title={'center': 'Avg. Area Number of Rooms'}>],
       [<Axes: title={'center': 'Avg. Area Number of Bedrooms'}>,
        <Axes: title={'center': 'Area Population'}>,
        <Axes: title={'center': 'Price'}>],
       [<Axes: title={'center': 'Address'}>, <Axes: >, <Axes: >]],
      dtype=object)
```



- The code you've provided uses Seaborn to create a joint plot between the Avg.Area House Age and Price columns in your dataset. Specifically, it uses a hexbin plot for visualization.
- The code will generate a grid of histograms, with each histogram representing thedistribution of a numerical column in your dataset. This provides a quick visualoverview of the datas distribution, which can be useful for understanding the range,spread, and central tendency of each numerical variable.

SCATTER PLOT:

A scatter plot is a type of data visualization that displays individual data points along two axes to represent the relationship between two variables. Each data point in the scatter plot represents the values of the two variables, allowing you to observe patterns, correlations, or trends in the data.

```
plt.figure(figsize=(8,6))
sns.scatterplot(x='Area Population', y='Price', data=data, color='purple')
plt.title('House Price vs. Area Population')
plt.xlabel('Area Population')
plt.ylabel('Price')
plt.show()
```

CODE LINK:

https://colab.research.google.com/drive/1GUgmcE1Q5eecbzcBgrSor0jB2IxaxcJA?usp=sharing

CONCLUSION:

In this development part 1 i have completed data loading it is the initial step, which involves importing the dataset for analysis. and preprocessing which includes data cleaning, transformation and remove duplicate values and finally Data visualization is essential for understanding the data's distribution, relationships, and correlations, aiding in feature selection and model interpretation.