AI_PHASE3

Domain:**A**rtificial **I**ntelligence

PROJECT 9: PREDICTING HOUSE PRICES USING MACHINE LEARNING

Introduction:

- In this project, the goal is to build a predictive model for house prices. We will follow a series of steps to load, explore, preprocess, and create a baseline model for house price prediction using a sample dataset. The model will use features from the dataset to predict the 'Avg. Area Income.

Steps to start building the house price prediction model by loading and preprocessing the dataset:

1.Import Necessary Libraries

2. Load the Dataset

3. Data Exploration and Preprocessing

    a. Explore the Data

    b.Handling Missing Values

    c.Encoding Categorical Variables

4. Splitting the Dataset

5. Building and Training the Model

6. Making Predictions

7. Evaluating the Model

1.Import Necessary Libraries:

- In the first step, we imported essential Python libraries, including Pandas, NumPy, and Scikit-Learn. These libraries provide tools for data manipulation, preprocessing, modeling, and evaluation.

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
```

2. Load the Dataset:

We loaded the dataset from the provided file path. The dataset is assumed to contain information relevant to house price prediction, and we used the Pandas library to read it.

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

3.Data Exploration and Preprocessing:

- In this phase, we performed data exploration and preprocessing to prepare the data for modeling.

a.Explore the Data:

- We examined the dataset by printing the first few rows and checking the data information to understand its structure and characteristics.

print(data.head())

print(data.info())

OUTPUT:

```
Avg. Area Income  Avg. Area House Age  Avg. Area Number of Rooms  \
0      79545.458574             5.682861                   7.009188
1      79248.642455             6.002900                   6.730821
2      61287.067179             5.865890                   8.512727
3      63345.240046             7.188236                   5.586729
4      59982.197226             5.040555                   7.839388

   Avg. Area Number of Bedrooms  Area Population         Price  \
0                          4.09     23086.800503  1.059034e+06
1                          3.09     40173.072174  1.505891e+06
2                          5.13     36882.159400  1.058988e+06
3                          3.26     34310.242831  1.260617e+06
4                          4.23     26354.109472  6.309435e+05

                                          Address
0  208 Michael Ferry Apt. 674\nLaurabury, NE 3701...
1  188 Johnson Views Suite 079\nLake Kathleen, CA...
2  9127 Elizabeth Stravenue\nDanieltown, WI 06482...
3                     USS Barnett\nFPO AP 44820
4                    USNS Raymond\nFPO AE 09386
```

```
<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   float64
 1   Avg. Area House Age           5000 non-null   float64
 2   Avg. Area Number of Rooms     5000 non-null   float64
 3   Avg. Area Number of Bedrooms  5000 non-null   float64
 4   Area Population               5000 non-null   float64
 5   Price                         5000 non-null   float64
 6   Address                       5000 non-null   object
dtypes: float64(6), object(1)
memory usage: 273.6+ KB
      None
```

b. Handling Missing Values:

- Missing values, if any, were filled with zeros in this code. It's important to note that a more sophisticated strategy for handling missing data may be necessary in real-world datasets.

data.fillna(0, inplace=True)

c. Encoding Categorical Variables:

- We used one-hot encoding to convert the 'Address' column into a numerical format suitable for machine learning.

data = pd.get_dummies(data, columns=['Address'])

4. Splitting the Dataset:

- We divided the data into two parts: the feature matrix 'X' (excluding the target variable, 'Avg. Area Income') and the target variable 'y' (which is the 'Avg. Area Income' column).

Feature scaling was applied to standardize the features using StandardScaler from Scikit-Learn.

```
X = data.drop('Avg. Area Income', axis=1)
y = data['Avg. Area Income']
scaler = StandardScaler()
X = scaler.fit_transform(X)
```

- The dataset was split into a training set (80%) and a testing set (20%). The random_state parameter was set to 42 for reproducibility.

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

5. Building and Training the Model:

- In this step, we created and trained a Linear Regression model using the training data. Linear Regression is a suitable choice for house price prediction tasks.

```
model = LinearRegression()
model.fit(X_train, y_train)
```

6: Making Predictions:

- We used the trained Linear Regression model to make predictions on the test data.

```
y_pred = model.predict(X_test)
```

7: Evaluating the Model:

- To assess the model's performance on the test data, we calculated the Mean Squared Error (MSE). The MSE measures the average squared difference between the predicted and actual 'Avg. Area Income' values.

mse = mean_squared_error(y_test, y_pred)

print(f"Mean Squared Error: {mse}")

FINAL OUTPUT :

```
Avg. Area Income  Avg. Area House Age  Avg. Area Number of Rooms  \
0      79545.458574             5.682861                  7.009188
1      79248.642455             6.002900                  6.730821
2      61287.067179             5.865890                  8.512727
3      63345.240046             7.188236                  5.586729
4      59982.197226             5.040555                  7.839388

   Avg. Area Number of Bedrooms  Area Population         Price  \
0                          4.09     23086.800503  1.059034e+06
1                          3.09     40173.072174  1.505891e+06
2                          5.13     36882.159400  1.058988e+06
3                          3.26     34310.242831  1.260617e+06
4                          4.23     26354.109472  6.309435e+05

                                             Address
0  208 Michael Ferry Apt. 674\nLaurabury, NE 3701...
1  188 Johnson Views Suite 079\nLake Kathleen, CA...
2  9127 Elizabeth Stravenue\nDanieltown, WI 06482...
3                     USS Barnett\nFPO AP 44820
4                    USNS Raymond\nFPO AE 09386
<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   float64
 1   Avg. Area House Age          5000 non-null   float64
 2   Avg. Area Number of Rooms    5000 non-null   float64
 3   Avg. Area Number of Bedrooms  5000 non-null   float64
 4   Area Population              5000 non-null   float64
```

```
 5   Price                         5000 non-null   float64
 6   Address                       5000 non-null   object
dtypes: float64(6), object(1)
memory usage: 273.6+ KB
None
Mean Squared Error: 72310314.15909015
```

Conclusion:

- In this phase, we successfully loaded, explored, and preprocessed the dataset to prepare it for modeling. We built a Linear Regression model and evaluated its performance using the Mean Squared Error.

- The model can serve as a baseline for house price prediction, and further enhancements, including feature engineering and model tuning, can be explored to improve predictive accuracy.

- It's important to emphasize that the choice of features, data preprocessing, and model selection should be adapted to the specific characteristics of your dataset and problem.

- Additionally, handling missing values should be approached with domain-specific knowledge and consideration for real-world data quality.


Thank
You