AI\_PHASE - 3: HOUSE PRICE PREDICTION

*DEVELOPMENT PART 1*

## 1.loading and preprocessing the dataset

Loading the Data:

Load the dataset into your machine learning environment. The dataset is typically provided in a structured format like a CSV (Comma-Separated Values) file. You can use libraries like Pandas in Python to read the dataset.

import pandas as pd

# Load the dataset

df = pd.read\_csv('USA\_Housing.csv')

# Check the dataset

print(df.head())

pre processing the data:

Data preprocessing is essential to clean and prepare the dataset for machine learning. Common preprocessing tasks include:

Handling Missing Values: Identify and handle missing values, which can be done by removing rows with missing values or imputing them.

Encoding Categorical Variables: If the dataset contains categorical variables (e.g., "Location"), you may need to encode them into numerical format, often using one-hot encoding.

Feature Scaling: Normalize or standardize numerical features to bring them to a similar scale. This can be important for certain machine learning algorithms.

Feature Selection: Decide which features are relevant for your prediction task and remove irrelevant ones.

Splitting Data: Divide the data into input features (X) and the target variable (y).

# Preprocess the data

# Drop any missing values

df = df.dropna()

# Convert categorical features to numerical features

# For example, we can use one-hot encoding to convert the ' State' feature to numerical features

df = pd.get\_dummies(df, columns=[' State'])

# Scale the numerical features

from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

X = scaler.fit\_transform(df.drop('Price', axis=1))

# Split the data into training and test sets

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, df['Price'], test\_size=0.25)

This code will load the dataset into a Pandas DataFrame, drop any missing values, convert categorical features to numerical features, scale the numerical features, and split the data into training and test sets

Once the data is preprocessed, we can train a machine learning model to predict house prices. Some popular machine learning algorithms for house price prediction include:

* Linear regression
* Random forest
* Gradient boosting machines

3.Start building the house price prediction model by loading and preprocessing the dataset.

We can use the following Python code to train a linear regression model:

from sklearn.linear\_model import LinearRegression

# Create a linear regression model

model = LinearRegression()

# Train the model

model.fit(X\_train, y\_train)

# Evaluate the model on the test set

y\_pred = model.predict(X\_test)

print('Mean squared error:', mean\_squared\_error(y\_test, y\_pred))

This code will train a linear regression model on the training data and evaluate it on the test data. The mean squared error (MSE) is a metric that measures how well the model fits the data. A lower MSE indicates a better fit.

Once the model is trained, we can use it to predict house prices for new data. For example, to predict the price of a house with the following features:

n**ew\_features = {**

**'Avg. Area Income': 75000,**

**'Avg. Area House Age': 30,**

**'Avg. Area Number of Rooms': 4,**

**'Avg. Area Number of Bedrooms': 3,**

**'Area Population': 50000,**

**'State': 'CA'**

**}**

4.Load the housing dataset and preprocess the data

use the following Python code to predict the price of the house:

# Scale the new features

new\_features = scaler.transform([new\_features])

# Make a prediction

prediction = model.predict(new\_features)

# Print the prediction

print('Predicted price:', prediction[0])

This code will predict the price of the house to be $500,000.