**AI PHASE-2 PROJECT: HOUSE PRICE PREDICTION**

**Improving Prediction Accuracy**

**Data Source:**

Choose a dataset containing information about houses, including features like location, square footage, bedrooms, bathrooms, and price.

import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

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

from sklearn.preprocessing import StandardScaler

from sklearn.metrics import r2\_score, mean\_absolute\_error,mean\_squared\_error

from sklearn.linear\_model import LinearRegression

from sklearn.linear\_model import Lasso

from sklearn.ensemble import RandomForestRegressor

from sklearn.svm import SVR

import xgboost as xg

%matplotlib inline

import warnings

warnings.filterwarnings("ignore")

**Data Preprocessing:**

Start by cleaning and preprocessing your dataset. This includes handling missing values, encoding categorical variables, and scaling features if necessary.

dataset = pd.read\_csv(‘/content/USA\_Housing.csv’)

**Feature Selection:**

Experiment with feature engineering techniques, such as creating interaction terms, polynomial features, or using feature selection methods to improve the model's performance.

dataset

dataset.info()

dataset.describe()

dataset.columns

Visualisation and Pre-Processing of Data

sns.histplot(dataset, x='Price', bins=50, color='y')

sns.boxplot(dataset, x='Price', palette='Blues')

sns.jointplot(dataset, x='Avg. Area House Age', y='Price', kind='hex')

sns.jointplot(dataset, x='Avg. Area Income', y='Price')

plt.figure(figsize=(12,8))

sns.pairplot(dataset)

dataset.hist(figsize=(10,8))

## **Visualising Correlation**

Visualizing correlation in house price prediction data can be essential for understanding the relationships between different variables and their impact on the final predictions. To visualize these correlations effectively, you can use scatter plots and correlation matrices.

dataset.corr(numeric\_only=True)

plt.figure(figsize=(10,5))

sns.heatmap(dataset.corr(numeric\_only = True), annot=True)

# Dividing Dataset in to features and target variable

# In house price prediction or any regression task, you need to divide your dataset into features (independent variables) and the target variable (dependent variable). Features are the input variables that you use to make predictions, while the target variable is the variable you want to predict.

X = dataset[['Avg. Area Income', 'Avg. Area House Age', 'Avg. Area Number of Rooms',

'Avg. Area Number of Bedrooms', 'Area Population']]

Y = dataset['Price']

# Using Train Test Split

# Divide your dataset into two parts: one containing the features (X) and the other containing the target variable (y)

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X, Y, test\_size=0.2, random\_state=101)

linkcode

Y\_train.head()

Y\_train.shape

Y\_test.head()

Y\_test.shape

# Standardizing the data

# Standardizing the data in house price prediction is an important preprocessing step, especially when you're working with features that have different scales. Standardization transforms your data so that it has a mean of 0 and a standard deviation of 1, making it easier for many machine learning algorithms to converge and perform well.

sc = StandardScaler()

X\_train\_scal = sc.fit\_transform(X\_train)

X\_test\_scal = sc.fit\_transform(X\_test)

# Model Selection: Model Building and Evaluation using XG Boost Regressor

# Building a house price prediction model using XGBoost (Extreme Gradient Boosting) Regressor is a common approach in machine learning. XGBoost is a powerful algorithm known for its performance in regression tasks.

model\_xg = xg.XGBRegressor()

model\_xg.fit(X\_train\_scal, Y\_train)

## **Predicting Prices**

Prediction = model\_xg.predict(X\_test\_scal)

**Evaluation: Evaluate the model's performance using metrics like Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and R-squared.**

plt.figure(figsize=(12,6))

plt.plot(np.arange(len(Y\_test)), Y\_test, label='Actual Trend')

plt.plot(np.arange(len(Y\_test)), Prediction5, label='Predicted Trend')

plt.xlabel('Data')

plt.ylabel('Trend')

plt.legend()

plt.title('Actual vs Predicted')

sns.histplot((Y\_test-Prediction4), bins=50)

print(r2\_score(Y\_test, Prediction2))

print(mean\_absolute\_error(Y\_test, Prediction2))

print(mean\_squared\_error(Y\_test, Prediction2))