PREDICTING HOUSE PRICE USING MACHINE LEARNING

Phase 3: Development Part 1

Project Title: House Price Prediction

TEAM MEMBERS:

SIVARANJINI J (311521106092)

PRITHISHA V (311521106073)

SWETHA S (311521106103)

YAMINI R (311521106115)

Topic: Building the house price prediction model by loading and pre-processing the dataset.

INTRODUCTION:

House price prediction is the process of forecasting the future selling price of a house based on historical data and current market trends. This can be done using a variety of machine learning techniques, such as regression analysis, random forests, and gradient boosting machines.

House price prediction is a challenging task, as there are many factors that can affect the price of a house, including:

- Location
- Size

- Age
- Condition
- Number of bedrooms and bathrooms
- Amenities
- Neighbourhood quality
- Market trends

Despite the challenges, house price prediction can be a valuable tool for both home buyers and sellers. For buyers, it can help them to estimate the value of a home they are considering purchasing and to make sure that they are not overpaying. For sellers, it can help them to set a competitive selling price and to sell their home quickly.

Here are some of the potential applications of house price prediction:

- Home buyers can use house price prediction to estimate the value of a home they are considering purchasing and to make sure that they are not overpaying.
- Home sellers can use house price prediction to set a competitive selling price and to sell their home quickly.
- Real estate investors can use house price prediction to identify undervalued properties that may be good investments.
- Mortgage lenders can use house price prediction to assess the risk of a loan and to set appropriate interest rates.
- Government agencies can use house price prediction to develop policies that promote affordable housing and to track the overall health of the housing market.

House price prediction is a complex and challenging task, but it has the potential to be a valuable tool for a variety of stakeholders. By using machine learning to analyze historical data and current market trends, house price prediction models can provide accurate estimates of the future selling price of a house.

GIVEN DATA SET:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTAT
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1.0	296.0	15.3	396.90	4.98
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2.0	242.0	17.8	396.90	9.14
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2.0	242.0	17.8	392.83	4.03
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3.0	222.0	18.7	394.63	2.94
4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3.0	222.0	18.7	396.90	5.33
5	0.02985	0.0	2.18	0.0	0.458	6.430	58.7	6.0622	3.0	222.0	18.7	394.12	5.21
6	0.08829	12.5	7.87	0.0	0.524	6.012	66.6	5.5605	5.0	311.0	15.2	395.60	12.43
7	0.14455	12.5	7.87	0.0	0.524	6.172	96.1	5.9505	5.0	311.0	15.2	396.90	19.15
8	0.21124	12.5	7.87	0.0	0.524	5.631	100.0	6.0821	5.0	311.0	15.2	386.63	29.93
9	0.17004	12.5	7.87	0.0	0.524	6.004	85.9	6.5921	5.0	311.0	15.2	386.71	17.10

Necessary step to follow:

1.Import Libraries:

Start by importing the necessary libraries:

Program:

```
import pandas as pd

import numpy as np

from sklearn.model_selection import train_test_split

from sklearn. preprocessing import StandardScaler
```

2.LOADING OF DATA SET:

There are a variety of ways to load a dataset in Python. The most common way is to use the pandas library. Pandas provides a variety of functions for reading and writing data in a variety of formats, including CSV, Excel, and JSON.

To load a CSV dataset in Python using pandas, you can use the read_csv () function. This function takes the path to the CSV file as input and returns a pandas Data Frame object. The Data Frame object is a two-dimensional data structure that is similar to a spreadsheet.

Program:

```
df = pd.read_csv(' E:\USA_Housing.csv ')

Pd.read()
```

3. Exploratory Data Analysis (EDA):

Perform EDA to understand your data better. This includes checking for missing values, exploring the data's statistics, and visualizing it to identify patterns.

Program:

```
# Check for missing values

print(df.isnull().sum())

# Explore statistics

print(df.describe())

# Visualize the data (e.g., histograms, scatter plots, etc.)
```

4. Feature Engineering:

Depending on your dataset, you may need to create new features or transform existing ones. This can involve one-hot encoding categorical variables, handling date/time data, or scaling numerical features.

Program:

```
# Example: One-hot encoding for categorical variables
```

df = pd.get_dummies(df, columns=[' Avg. Area Income ', ' Avg. Area

House Age '])

5. Split the Data:

Split your dataset into training and testing sets. This helps you evaluate

your model's performance later.

X = df.drop('price', axis=1) # Features

y = df['price'] # Target variable

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,

random_state=42)

6. Feature Scaling:

Apply feature scaling to normalize your data, ensuring that all

features have similar scales. Standardization (scaling to mean=0 and

std=1) is a common choice.

Program:

scaler = StandardScaler()

X_train = scaler.fit_transform(X_train)

X_test = scaler.transform(X_test)

Importance of loading and processing dataset:

Loading the dataset and preprocessing the data are two of the most important steps in any machine learning project. By loading the dataset and preprocessing the data correctly, you can ensure that your machine learning model has the best

possible data to work with. This will lead to more accurate and reliable predictions.

Here are some of the specific benefits of loading the dataset and preprocessing the data correctly:

- Improved accuracy: Machine learning models are only as good as the data they are trained on. If your data is noisy or incomplete, your model will not be able to learn the patterns in the data effectively. As a result, your model will make less accurate predictions.
- Reduced training time: Preprocessing the data can help to reduce the training time of your machine learning model. This is because preprocessing the data can remove noise and outliers from the data, which can make the training process more efficient.
- Improved interpretability: Preprocessing the data can also help to improve the interpretability of your machine learning model. This is because preprocessing the data can help you to understand the relationships between the different variables in your data. As a result, you will be better able to understand how your model is making predictions.

In addition to these specific benefits, loading the dataset and preprocessing the data correctly is also essential for ensuring that your machine learning project is reproducible. This is because reproducibility is important for ensuring that the results of your project can be trusted and verified by others.

Overall, loading the dataset and preprocessing the data are two of the most important steps in any machine learning project. By following the best practices for loading and preprocessing data, you can improve the accuracy, efficiency, and interpretability of your machine learning models.

Challenges involved in loading and preprocessing a house price dataset:

There are a number of challenges involved in loading and preprocessing a house price dataset. Some of the most common challenges include:

- Data quality: House price datasets can often be noisy and incomplete. This is because data can be collected from a variety of sources, such

as public records, real estate listings, and appraisal data. The quality of data from these sources can vary widely.

- Data format: House price datasets can be in a variety of formats, such as CSV, Excel, and JSON. This can make it difficult to load and preprocess the data in a consistent way.
- Missing values: House price datasets often contain missing values. This can happen for a variety of reasons, such as data collection errors or user opt-outs. Missing values can skew the results of machine learning models, so it is important to handle them carefully.
- Outliers: House price datasets can often contain outliers. Outliers are data points that are significantly different from the rest of the data. They can be caused by errors in data collection or by unusual events. Outliers can skew the results of machine learning models, so it is important to identify and remove them before training a model.
- Feature engineering: House price datasets may require some feature engineering. Feature engineering is the process of creating new features by combining existing features or by transforming existing features. For example, you could create a new feature for the age of the house by subtracting the year the house was built from the current year.

Here are some tips for overcoming the challenges involved in loading and preprocessing a house price dataset:

- Use a variety of data sources: The more data you have, the better your machine learning model will perform. Consider using multiple data sources, such as public records, real estate listings, and appraisal data.
- Clean the data: Once you have loaded your data, it is important to clean it. This involves removing outliers, handling missing values, and correcting errors.
- Encode categorical variables: Categorical variables are variables that can take on a limited number of values, such as "good", "fair", and "poor". Machine learning models can only work with numerical data, so categorical variables need to be encoded into numerical values before training a model.

- Scale numerical variables: Numerical variables should be scaled so that they have a similar range of values. This is important to prevent variables with larger ranges from dominating the training process.
- Split the data into training and testing sets: Once the data has been preprocessed, it should be split into a training set and a testing set. The training set will be used to train the machine learning model, and the testing set will be used to evaluate the performance of the model.

By following these tips, you can overcome the challenges involved in loading and preprocessing a house price dataset and prepare your data for machine learning.

How to overcome the challenges of loading and preprocessing a house price dataset:

Here are some tips on how to overcome the challenges of loading and preprocessing a house price dataset:

1. Use a data cleaning library. There are a number of data cleaning libraries available in Python, such as Pandas and NumPy. These libraries provide a variety of functions for cleaning and preprocessing data.

2. Handle missing values carefully. There are a number of ways to handle missing values, such as imputing the mean or median value, or using a more sophisticated technique such as multiple imputation. The best way to handle missing values will depend on the specific dataset and the machine learning model that you are using.

3. Encode categorical variables. Categorical variables are variables that can take on a limited number of values, such as "good", "fair", and "poor". Machine learning models can only work with numerical data, so categorical variables need to be encoded into numerical values before training a model. There are a number of ways to encode categorical variables, such as one-hot encoding and label encoding.

4. Scale numerical variables. Numerical variables should be scaled so that they have a similar range of values. This is important to prevent variables with larger ranges from dominating the training process. There are a number of ways to scale numerical variables, such as min-max scaling and standard scaling.

5. Split the data into training and testing sets. Once the data has been preprocessed, it should be split into a training set and a testing set. The training set will be used to train the machine learning model, and the testing set will be used to evaluate the performance of the model.

   Here are some additional tips:

- Use a variety of data sources. The more data you have, the better your machine learning model will perform. Consider using multiple data sources, such as public records, real estate listings, and appraisal data.

- Create new features. You can create new features by combining existing features or by transforming existing features. For example, you could create a new feature for the age of the house by subtracting the year the house was built from the current year.

- Visualize the data. Data visualization can help you to identify outliers, missing values, and other problems with your data.

- Document your process. It is important to document your data preprocessing process so that you can reproduce it later. This is especially important if you are working with a team of people.

  By following these tips, you can overcome the challenges of loading and preprocessing a house price dataset and prepare your data for machine learning.

Program:

```
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")
/opt/conda/lib/python3.10/site-packages/scipy/__init__.py:146:
UserWarning: A NumPy version >=1.16.5 and <1.23.0 is required for
this version of SciPy (detected version 1.23.5
warnings.warn(f"A NumPy version >={np_minversion} and
<{np_maxversion}"
```

Loading Dataset:

dataset = pd.read_csv('E:/USA_Housing.csv')

Some common data preprocessing tasks include:

- Data cleaning: This involves identifying and correcting errors or inconsistencies in the data, such as missing values, outliers, and duplicates.
- Data integration: This involves combining data from multiple sources to create a unified dataset.
- Data reduction: This involves reducing the size of the dataset while preserving the important information. This can be done by removing redundant features, sampling the data, or aggregating the data.
- Data transformation: This involves converting the data into a format that is suitable for the machine learning algorithm that will be used. This may involve scaling the data, encoding categorical variables, or creating new features.

Here are some specific examples of data preprocessing tasks for house price prediction:

- Handling missing values: Missing values can be handled by imputing the mean or median value, or using a more sophisticated technique such as multiple imputation.
- Encoding categorical variables: Categorical variables such as neighborhood quality and condition can be encoded using one-hot encoding or label encoding.
- Scaling numerical variables: Numerical variables such as square footage and number of bedrooms should be scaled so that they have a similar range of values.
- Creating new features: New features can be created by combining existing features. For example, you could create a new feature for the age of the house by subtracting the year the house was built from the current year.

By performing these data preprocessing tasks, you can improve the quality and accuracy of your machine learning model.

OUTPUT:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LS
count	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.000000	506.00
mean	3.593761	11.363636	11.136779	0.069170	0.554695	6.284634	68.574901	3.795043	9.549407	408.237154	18.455534	356.674032	12.65
std	8.596783	23.322453	6.860353	0.253994	0.115878	0.702617	28.148861	2.105710	8.707259	168.537116	2.164946	91.294864	7.14
min	0.006320	0.000000	0.460000	0.000000	0.385000	3.561000	2.900000	1.129600	1.000000	187.000000	12.600000	0.320000	1.73
25%	0.082045	0.000000	5.190000	0.000000	0.449000	5.885500	45.025000	2.100175	4.000000	279.000000	17.400000	375.377500	6.95
50%	0.256510	0.000000	9.690000	0.000000	0.538000	6.208500	77.500000	3.207450	5.000000	330.000000	19.050000	391.440000	11.36
75%	3.647423	12.500000	18.100000	0.000000	0.624000	6.623500	94.075000	5.188425	24.000000	666.000000	20.200000	396.225000	16.95
max	88.976200	100.000000	27.740000	1.000000	0.871000	8.780000	100.000000	12.126500	24.000000	711.000000	22.000000	396.900000	37.97

Conclusion:

 In the quest to build a house price prediction model, we have embarked on a critical journey that begins with loading and preprocessing the dataset.We have traversed through essential steps, starting with importing the necessary libraries to facilitate data manipulation and analysis.

 Understanding the data's structure, characteristics, and any potential issues through exploratory data analysis (EDA) is essential for informed decision-making.

 Data preprocessing emerged as a pivotal aspect of this process. It involves cleaning, transforming, and refining the dataset to ensure that it aligns with the requirements of machine learning algorithms.

 With these foundational steps completed, our dataset is now

primed for the subsequent stages of building and training a house price prediction model.