House Price Prediction Using Machine Learning

Building a house price prediction model involves several key phases, including feature selection, model training, and evaluation. Here's a detailed explanation of each phase:

1. **Feature Selection**

Feature selection is a critical step in building a house price prediction model. It involves choosing the most relevant and informative attributes (features) that will be used to make predictions. Here's how you can approach this phase:

- ***Data Collection***: Gather a dataset that contains information about houses, such as square footage, number of bedrooms, location, age, and other relevant features. You might also include historical sales data.

- ***Exploratory Data Analytics (EDA)***: Perform EDA to understand the data's characteristics. This includes data visualization and statistical analysis to identify patterns and correlations among features.

- ***Feature Engineering*** : Create new features or modify existing ones to extract more valuable information. For example, you can calculate the price per square foot or create categorical variables for the neighborhood.

- ***Feature Selection Methods*** : Use techniques like correlation analysis, mutual information, or feature importance from machine learning models to identify the most important features. Remove irrelevant or redundant features that do not contribute significantly to the model's predictive power.

2.**Model Training**

Once you've selected the relevant features, it's time to build and train a machine learning model:

- ***Data Splitting***: Split your dataset into training and testing subsets. Typically, you'd use a significant portion of the data for training (e.g., 70-80%) and the rest for testing to evaluate the model's performance.

- ***Choose a Model***: Select an appropriate regression model for predicting house prices. Common choices include Linear Regression, Decision Trees, Random Forest, or Gradient Boosting.

- ***Data Preprocessing*** : Standardize or normalize the data to ensure that all features are on the same scale. Handle missing values and outliers as well.

-***Model Training***: Use the training data to train the chosen model. The model learns the relationships between the features and the target variable (house prices).

3.**Evaluation**

Once the model is trained, you need to assess its performance to ensure it's making accurate predictions:

-***Predictions***: Use the testing dataset to make predictions for house prices. You can also use cross-validation for a more robust assessment.

- ***Evaluation Metrics***: Calculate evaluation metrics like Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R-squared (R^2) to measure the model's accuracy and how well it fits the data.

-***Model Tuning***: If the model's performance is not satisfactory, you can fine-tune hyperparameters or consider different algorithms to improve the results.

- ***Visualization***: Visualize the model's predictions compared to actual prices to gain insights into its strengths and weaknesses.

This iterative process of feature selection, model training, and evaluation may need to be repeated several times to build the most accurate house price prediction model. The ultimate goal is to have a model that can reliably estimate house prices based on the selected features.

Program for House Price Prediction

-->**Importing Libraries and Dataset**

* Pandas – To load the Dataframe
* Matplotlib – To visualize the data features i.e. barplot
* Seaborn – To see the correlation between features using heatmap

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
dataset = pd.read_excel("HousePricePrediction.xlsx")

# Printing first 5 records of the dataset
print(dataset.head(5))
```

-->**Data Preprocessing**

Now, we categorize the features depending on their datatype (int, float, object) and then calculate the number of them.

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

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

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

-->**Exploratory Data Analysis**

EDA refers to the deep analysis of data so as to discover different patterns and spot anomalies. Before making inferences from data it is essential to examine all your variables.

So here let's make a heatmap using seaborn library.

```
plt.figure(figsize=(12, 6))
sns.heatmap(dataset.corr(),
                        cmap = 'BrBG',
                        fmt = '.2f',
                        linewidths = 2,
                        annot = True)
```

-->**Data Cleaning**

Data Cleaning is the way to improvise the data or remove incorrect, corrupted or irrelevant data.

As in our dataset, there are some columns that are not important and irrelevant for the model training. So, we can drop that column before training. There are 2 approaches to dealing with empty/null values

We can easily delete the column/row (if the feature or record is not much important).
Filling the empty slots with mean/mode/0/NA/etc. (depending on the dataset requirement).
As Id Column will not be participating in any prediction. So we can Drop it.

```
dataset.drop(['Id'],
                axis=1,
                inplace=True)

dataset['SalePrice'] = dataset['SalePrice'].fillna(   dataset['SalePrice'].mean())

new_dataset = dataset.dropna()
new_dataset.isnull().sum()
```

-->Linear Regression

Linear Regression predicts the final output-dependent value based on the given independent features. Like, here we have to predict SalePrice depending on features like MSSubClass, YearBuilt, BldgType, Exterior1st etc

```
from sklearn.linear_model import LinearRegression

  model_LR = LinearRegression()
  model_LR.fit(X_train, Y_train)
  Y_pred = model_LR.predict(X_valid)

print(mean_absolute_percentage_error(Y_valid, Y_pred))
```

-->**Conclusion**

Clearly, SVM model is giving better accuracy as the mean absolute error is the least among all the other regressor models i.e. 0.18 approx. To get much better results ensemble learning techniques like Bagging and Boosting can also be used.house price prediction using machine learning is a valuable application that leverages data and various tools and technologies to estimate housing prices accurately. It's a versatile field with the potential for widespread use in the real estate industry and beyond.