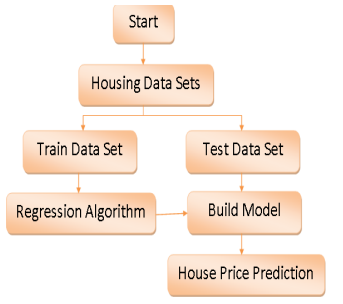
**A Machine Learning Approach to House Price Prediction**

**ABSTRACT:-**

The "House Price Prediction using Machine Learning" project presents a comprehensive approach to predicting real estate prices by harnessing the power of advanced data analysis techniques. Developed primarily using Python programming language, the project employs the Random Forest Regressor algorithm as its core predictive model. The objective is to accurately estimate the prices of residential properties, contributing to informed decision-making in the real estate market. In this project, a dataset containing 42,703 individual data points from the United States of America is utilized for training and evaluation. The dataset encompasses various essential features that influence property prices, including location, square footage, number of bedrooms and bathrooms, amenities, and more. By leveraging this diverse set of attributes, the Random Forest Regressor algorithm learns intricate patterns and relationships within the data, enabling it to make reliable predictions. The project's success is measured by the achieved performance metrics. During the training phase, the model attains a Mean Absolute Error (MAE) of 1.4606, indicating the average absolute difference between predicted and actual prices on the training set. Furthermore, on the test set, the model demonstrates its generalization capability by achieving a MAE of 3.8313. These metrics underscore the model's ability to make accurate predictions on unseen data, enhancing its practical utility in real-world scenarios. The Proposed House Price Prediction using Machine Learning showcases the efficacy of the Random Forest Regressor algorithm in forecasting residential property prices. The Python-based implementation leverages a dataset comprising thousands of data points from the United States, contributing to a robust and reliable predictive model. The achieved low Mean Absolute Error values on both training and test sets emphasize the model's accuracy and generalization potential. This project holds significant implications for individuals, investors, and real estate professionals seeking data-driven insights to navigate the dynamic real estate market.

|  |  |
| --- | --- |
| **EXSISTING SYSTEM** | **PROPOSED SYSTEM** |
| * Gradient Boosting Regressor is a machine learning algorithm that falls under the ensemble learning category. It builds a predictive model in the form of an ensemble of weak learners, typically decision trees, to make accurate predictions. * The idea behind gradient boosting is to sequentially add new models to correct the errors of the existing ensemble. The algorithm minimizes a loss function, such as mean squared error for regression problems, by iteratively fitting new models to the residuals. | * Random Forest is a robust ensemble learning algorithm that leverages the strength of multiple decision trees. By constructing a diverse set of trees through techniques like bagging and random feature selection, it achieves higher predictive accuracy and resilience compared to individual trees. * Known for its versatility, Random Forest is applicable to both classification and regression tasks. Its ability to handle high-dimensional datasets, mitigate overfitting, and provide insights into feature importance makes it a popular and effective choice across various domains. |
| **EXISTING ALGORITHM**   * Gradient Boosting Regressor | **PROPOSED ALGORITHM**   * Random Forest Regressor |
| * Gradient Boosting Regressor is a boosting algorithm that builds an additive model in a forward stage-wise fashion. It combines the predictions from multiple weak learners to create a strong predictive model. * The algorithm minimizes the loss function by adjusting the parameters of each weak learner during each iteration, emphasizing areas where the current model makes errors. | **ALGORITHM DEFINITION**   * Random Forest is a machine learning algorithm that builds a multitude of decision trees during training, creating an ensemble model for classification or regression tasks. It operates by employing two key sources of randomness: bagging (bootstrap aggregating) and random feature selection. * Bagging involves training each tree on a different subset of the training data, sampled with replacement, while random feature selection considers only a subset of features at each split in the decision trees. |
| **DRAWBACKS: -**   * Gradient Boosting Regressors, especially when using a large number of trees or deep trees, can be computationally expensive and time-consuming. * Gradient Boosting models, if not properly tuned, are susceptible to overfitting. * Gradient Boosting Regressors have several hyper parameters that need to be tuned for optimal performance. | **ADVANTAGES: -**   * Random Forests often provide high accuracy in classification tasks. * Random Forests can handle large datasets with a large number of features. * They are computationally efficient and can parallelize the training process, making them suitable for big data applications. |

**SYSTEM ARCHITECTURE:**

****

**MINIMUMSYSTEM REQUIREMENTS**

**HARDWARE REQUIREMENTS**

• PROCESSOR : Pentium i3 Processor

• RAM : 4GB DD RAM

• HARD DISK : 450 GB

**SOFTWARE REQUIREMENTS**

• BACK END : PYTHON

• OPERATING SYSTEM : WINDOWS 10

• IDE : Spyder3