

**VIET NAM NATIONAL UNIVERSITY HO CHI MINH CITY UNIVERSITY OF INFORMATION TECHNOLOGY**

**INFORMATION SYSTEM FACULTY**

——— 🕮 ———

SUBJECT: DATA MINING

**FINAL PROJECT REPORT**

**TOPIC:**

**FORECASTING THE HOUSE PRICE**

**Lecturer**: Mrs Cao Thi Nhan

Mr Vu Minh Sang

**Class:** IS252.O22.HTCL

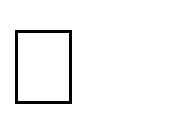
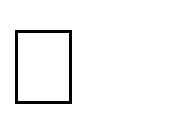
**Group:** Team 1

**Student performance**:

| Nguyen Khanh Van | 21522781 |
| --- | --- |
| Dang Quang Nhat | 21522413 |
| Ho Dac Khai | 21522183 |

**Ho Chi Minh City, June 2024**

**VIET NAM NATIONAL UNIVERSITY HO CHI MINH CITY UNIVERSITY OF INFORMATION TECHNOLOGY INFORMATION SYSTEM FACULTY**

——— 🕮 ———



SUBJECT: DATA MINING

**FINAL PROJECT REPORT**

**TOPIC:**

**PREDICT THE MUSIC GENRE OF A SONG**

**Lecturer**: Mrs Cao Thi Nhan

Mr Vu Minh Sang

**Class:** IS252.O22.HTCL

**Group:** Team 1

**Student performance**:

| Nguyen Khanh Van | 21522781 |
| --- | --- |
| Dang Quang Nhat | 21522413 |
| Ho Dac Khai | 21522183 |
|  |  |

**Ho Chi Minh City, June 2024**

# TEACHER’S COMMENTS

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

……………………………………………………………………………………………………

# ACKNOWLEDGEMENT

In fact, there is no success that is not tied to the support and help, whether more or less, directly or indirectly from others. With the deepest gratitude, first of all, our group would like to express our sincere thanks to the teachers of the University of Information Technology - Vietnam National University, Ho Chi Minh City and the teachers of the Faculty of Information Systems helped the group to have the basic knowledge as a basis to carry out this topic.

In particular, our team would like to express our sincere thanks to Mrs. Cao Thi Nhan - theoretical lecturer and Mr. Vu Minh Sang -practical lecturer of Data Mining who wholeheartedly helped, directly instructed and guided the group throughout the process a project.

Thanks to that, we have gained a lot of useful knowledge in applying as well as project-making skills. Without the guidance and teachings of the teacher, our group thinks this project of the group would be very difficult to complete. Once again, I sincerely thank teacher. In addition, for the project to be completed, it is impossible to thank the people who did it, thank you to the team members who worked hard and completed the task on schedule.

**Finally, thank you to all the team members who worked at their best to complete their thesis well. Sincerely thank!**

# TABLE OF CONTENTS

Contents

[**TEACHER’S COMMENTS 3**](#_heading=h.gjdgxs)

[**ACKNOWLEDGEMENT 4**](#_heading=h.30j0zll)

[**TABLE OF CONTENTS 5**](#_heading=h.1fob9te)

[**CHAPTER I: INTRODUCTION 7**](#_heading=h.3znysh7)

[1.1](#_heading=h.2et92p0) Problems 7

[1.2](#_heading=h.tyjcwt) Project goal 7

[1.3](#_heading=h.3dy6vkm) Developer tools & Technology 7

[**CHAPTER II: DATA PREPROCESSING 9**](#_heading=h.1t3h5sf)

[2.1](#_heading=h.4d34og8) Description of original data 9

[2.2](#_heading=h.2s8eyo1) Data preprocessing 16

[**CHAPTER III: ALGORITHMS 48**](#_heading=h.17dp8vu)

[3.1 Linear Regression 48](#_heading=h.3rdcrjn)

[3.2 Lasso Regression 49](#_heading=h.26in1rg)

[3.3 Extreme Gradient Boost Model 50](#_heading=h.lnxbz9)

[3.4 Random Forest 52](#_heading=h.35nkun2)

[Random Forest explain 52](#_heading=h.1ksv4uv)

[Random Forests Algorithm 53](#_heading=h.44sinio)

[3.5 Gradient Boosting Model 57](#_heading=h.2jxsxqh)

[3.6 K-Nearest Neighbors 58](#_heading=h.z337ya)

[K-Nearest Neighbors explain 58](#_heading=h.3j2qqm3)

[K-nearest neighbors Algorithm 58](#_heading=h.1y810tw)

[**CHAPTER IV: PREDICTIVE SOFTWARE AND EXPERIMENT 61**](#_heading=h.4i7ojhp)

[4.1 Predictive sofeware 61](#_heading=h.2xcytpi)

[4.1.1 Building Pipelines and Params Searching 61](#_heading=h.1ci93xb)

[4.1.2 Eval models and find best processed data 63](#_heading=h.3whwml4)

[4.1.3 Models training for Demo application. 73](#_heading=h.2bn6wsx)

[4.2 Experimement 75](#_heading=h.qsh70q)

[4.2.1 Dataset 75](#_heading=h.3as4poj)

[4.2.2 Evaluation forecasting models 77](#_heading=h.1pxezwc)

[**CHAPTER V: CONCLUSION 79**](#_heading=h.49x2ik5)

[**REFERENCES 80**](#_heading=h.2p2csry)

# CHAPTER I: INTRODUCTION

## Problems

In the realm of real estate, accurately predicting house prices is a complex task influenced by numerous factors. Traditional methodologies for estimating property values often fall short of providing accurate and dynamic pricing models. As a result, there is a growing demand for more advanced techniques that can bridge the gap between traditional approaches and cutting-edge technology. This study aims to address these challenges by leveraging the power of machine learning algorithms to unravel the intricacies of house price dynamics.

## Project goal

* + - The primary objective of this project is to develop a solution that can assist users in referring to and predicting reasonable house prices in the market. By utilizing regression algorithms and artificial intelligence techniques such as eXtreme Gradient Boosting (XGBoost), k-Nearest Neighbors (KNN), Gradient Boosting, Linear Regression, Lasso Regression and Random Forest, we seek to provide accurate predictions for house prices. Through the application of data mining methods, our study aims to identify the most suitable algorithm that ensures the accuracy of the results. This will enable users to make informed decisions and mitigate risks when investing in houses.

## Developer tools & Technology

**In the process of implementation, the group used several software for researching and developing the topic:**

* Information collection and analysis using the python library and programming language.
* Data sources: [House Price | Kaggle](https://www.kaggle.com/datasets/juhibhojani/house-price?fbclid=IwZXh0bgNhZW0CMTAAAR341R7PK4OcMLBStyvWsDQlbaAwloWArzUqx0RndtgqzkkC3eXiJ8_T9Jc_aem_ARH4chYIieE0nRUbGQwUJuXb7xSmGrIarW6khfj6VwZJ3CfGG_RpNcwxCaPDuKuwqVPrYjoggwGawz2oAuNUJW9Y)

All the above software is installed and used by the team on Microsoft Windows 11 operating system. The compatibility of the above software with other operating systems is not within the scope of this study.

# CHAPTER II: DATA PREPROCESSING

## Description of original data

#### Data Sources

**Author**: JUHI BHOJANI

#### Data file

**Total data rows**: 187532

#### Attribute number and value

**Total columns**: 21

**Dataset characteristics**: Multivariable

**Attribute number characteristics**: Characters, real numbers, integers

**Lost value**: None

#### Attribute number and value

**Symbol: # -**number**,  -**character **Sources**: [House Price | Kaggle](https://www.kaggle.com/datasets/juhibhojani/house-price?fbclid=IwZXh0bgNhZW0CMTAAAR341R7PK4OcMLBStyvWsDQlbaAwloWArzUqx0RndtgqzkkC3eXiJ8_T9Jc_aem_ARH4chYIieE0nRUbGQwUJuXb7xSmGrIarW6khfj6VwZJ3CfGG_RpNcwxCaPDuKuwqVPrYjoggwGawz2oAuNUJW9Y)

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A close-up of a text

Description automatically generated

#### Attribute statical information of categorical features

**A screenshot of a computer

Description automatically generated**

#### Statistical information of numerical features

**A screenshot of a computer

Description automatically generated**

## Data preprocessing

**Purpose**:

* DataTransformation
* Data Collection
* Data visualization and comments

#### Import Library

**A white background with a few green objects

Description automatically generated with medium confidence**

#### Import Dataset

**A screenshot of a computer

Description automatically generated**

#### Check data type, information



#### Take a look at missing values

A screenshot of a computer

Description automatically generated

A black and white chart with white text

Description automatically generated

#### Remove attributes with high missing rate.

A close-up of a computer screen

Description automatically generated

Boxplot after remove attributes with high missing rate

A black and white image of a chart

Description automatically generated with medium confidence

A screenshot of a computer

Description automatically generated

#### Exploratory data analysis.

A screenshot of a computer

Description automatically generated

A screen shot of a graph

Description automatically generated

##### Attribute “Title”

The title has special structure: Number + BHK + Real Title, which BHK means

“Bedroom, Hall and Kitchen”. So, we will divide this column into 2

columns which are “BHK” and Title.

A screenshot of a computer

Description automatically generated

##### Attribute “Amount”

With this attribute, it had 3 different unit “Lac”, “Cr”, “Call for price”, so we will be converted all different unit to Rupees. The “Call for price” will be defined as missing values.

A screenshot of a computer

Description automatically generated

A computer screen shot of numbers

Description automatically generated

A screenshot of a computer

Description automatically generated

##### Attribute “Floor”

Firstly, it is defined that “ground”/”Ground” is “0”, “nan” is “nan”, “Lower

Basement”/ “lower”/”upper” is “-1”. The Floor column is splitted into 2 columns which is “Floor Location” and “Total Floor”.

A screenshot of a computer

Description automatically generated

A computer screen shot of a program code

Description automatically generated

A screenshot of a computer

Description automatically generated

##### Attribute “Transaction”

A screenshot of a computer

Description automatically generated

##### Attribute “Bathroom”

In this column, there is one strange unit “> 10”, it is converted to “11”. And the same with “Balcony” column.

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

##### Attribute “Furnishing”

A screenshot of a computer code

Description automatically generated

##### Attribute “Balcony”

This attribute has the same situation as the attribute “Bathroom”, so we will handle with the same way when we handle the attribute “Bathroom”.

A screenshot of a computer

Description automatically generated

##### Attribute “Carpet Area”

There are total 9 different measures in Caper Area Unit column, all will be

converted to “sqft” to ensure consistency.

A screen shot of a computer

Description automatically generated

A screenshot of a computer code

Description automatically generated

A screenshot of a computer program

Description automatically generated

##### Attribute “Facing”

In Facing column, there is one wrong syntax “South –West”, which is fixed into “South – West”.

A screenshot of a computer

Description automatically generated

##### Attribute “Status”

A screenshot of a computer code

Description automatically generated

##### Attribute “Overlooking”

It can be seen that there are 4 different views in this attribute, including “Not Available”, “Main Road”, “Garden/Park”, “Pool”. And Overlooking column will also be divided into 4 corresponding columns for better identification.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

##### Attribute “Ownership”

A screenshot of a computer program

Description automatically generated

##### Handle outlier



A graph with numbers and lines

Description automatically generated

A graph of a box plot

Description automatically generated

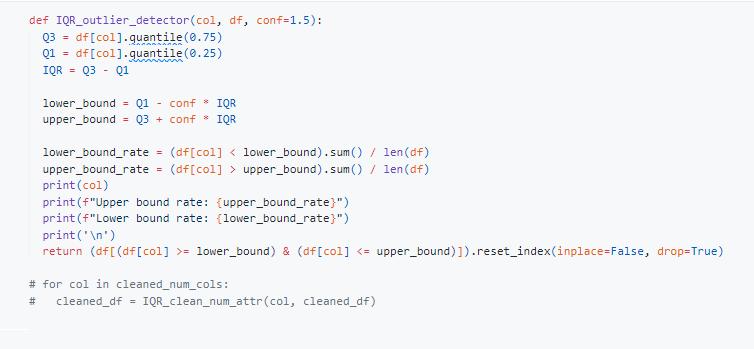
A graph of a box plot

Description automatically generated

A graph of a box plot

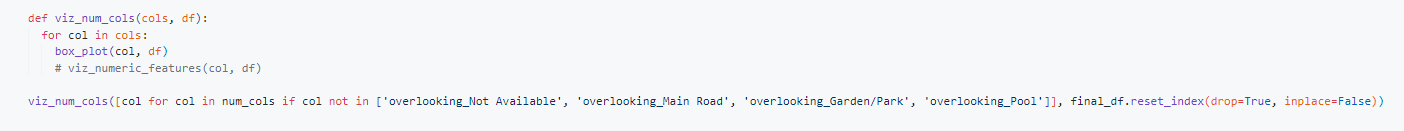
Description automatically generated

It can be seen that the level of outliers is warning, and the solution is use Interquartile Range to calculate and remove relevant outliers.



A screenshot of a computer

Description automatically generated



A graph of a box plot

Description automatically generated

A diagram of a box plot

Description automatically generated

A graph of a box plot

Description automatically generated

A blue rectangular object with a black line

Description automatically generated

A graph with a line

Description automatically generated

#### Handle missing value

A screenshot of a computer

Description automatically generated

##### Approach 1 : Remove any rows with missing values

The first solution and and least effective solution is removing any rows with missing values. This solution left behind a lot of harmful effects to our training later, including the loss of information, reduction in sample size, reduction in model instability. It is not recommended to use this solution.

A computer code with text

Description automatically generated with medium confidence

##### Approach 2 : Impute missing values with Descriptive Statistics

##### (Mean/Median/Mode)

The second solution is impute missing values with Descriptive Statistics. Using mean/median/mode values to fill in misinformation can guarantee the sample size. But it is still a simple solution, and using a single value (mean, median, or mode) to impute missing values can lead to a loss of variability in the dataset. It doesn't capture the individual nuances of the missing values. And this solution can artificially inflate correlations between variables, especially in situations where missing values are not

missing completely at random (MCAR). This can lead to biased results.

A screenshot of a computer code

Description automatically generated

##### Approach 3 : Impute missing values by finding correlation groups

The fourth solution is impute missing values by finding correlation groups. It is a strategy that involves identifying groups of variables that are correlated and then using the information from correlated variables to impute missing values.

This solution helps preserve relationships between variables, especially when missing data is related to other variables in a systematic way. Unlike machine learning-based imputation methods, correlation-based imputation does not require training a separate model, which can simplify the imputation process and reduce computational overhead. This solution is very sensitive to outliers, which has been handle earlier.

A close-up of a computer screen

Description automatically generated

A screenshot of a computer

Description automatically generated

A computer code with many colorful text

Description automatically generated with medium confidence

A close-up of a text

Description automatically generated

A computer screen shot of a program code

Description automatically generated

A screenshot of a computer

Description automatically generated

A close-up of a computer screen

Description automatically generated

#### Feature selection

A screen shot of a computer code

Description automatically generated

A screenshot of a computer

Description automatically generated

A screen shot of a computer

Description automatically generated

A white background with black text

Description automatically generated

A computer screen shot of a code

Description automatically generated

A graph with blue rectangles and black text

Description automatically generated

After using Labelencoder from sklearn and MDI, it is decided to keep all of the feature. Because these features all directly or indirectly affect each other.

# CHAPTER III: ALGORITHMS

## 3.1 Linear Regression

Linear regression is a type of supervised machine learning algorithm that computes the linear relationship between the dependent variable and one or more independent features by fitting a linear equation to observed data.

When there is only one independent feature, it is known as Simple Linear Regression, and when there are more than one feature, it is known as Multiple Linear Regression.

Similarly, when there is only one dependent variable, it is considered Univariate Linear Regression, while when there are more than one dependent variables, it is known as Multivariate Regression.

**Why Linear Regression is Important?**

The interpretability of linear regression is a notable strength. The model’s equation provides clear coefficients that elucidate the impact of each independent variable on the dependent variable, facilitating a deeper understanding of the underlying dynamics. Its simplicity is a virtue, as linear regression is transparent, easy to implement, and serves as a foundational concept for more complex algorithms.

Linear regression is not merely a predictive tool; it forms the basis for various advanced models. Techniques like regularization and support vector machines draw inspiration from linear regression, expanding its utility. Additionally, linear regression is a cornerstone in assumption testing, enabling researchers to validate key assumptions about the data.

**Types of Linear Regression**

There are two main types of linear regression:

**Simple Linear Regression**

This is the simplest form of linear regression, and it involves only one independent variable and one dependent variable. The equation for simple linear regression is:

Where:

* Y is the dependent variable
* X is the independent variable
* β0 is the intercept
* β1 is the slope

**Multiple Linear Regression**

This involves more than one independent variable and one dependent variable. The equation for multiple linear regression is:

Where:

* Y is the dependent variable
* X1, X2, …, Xp are the independent variables
* β0 is the intercept
* β1, β2, …, βn are the slopes
* The goal of the algorithm is to find the best Fit Line equation that can predict the values based on the independent variables.
* In regression set of records are present with X and Y values and these values are used to learn a function so if you want to predict Y from an unknown X this learned function can be used. In regression we have to find the value of Y, So, a function is required that predicts continuous Y in the case of regression given X as independent features.

## 3.2 Lasso Regression

Lasso Regression is also another linear model derived from Linear Regression which shares the same hypothetical function for prediction. The cost function of Linear Regression is represented by J.

* Here, m is the total number of training examples in the dataset.
* represents the hypothetical function for prediction.
* represents the value of target variable for ith training example.

The Linear Regression model considers all the features equally relevant for prediction. When there are many features in the dataset and even some of them are not relevant for the predictive model. This makes the model more complex with a too inaccurate prediction on the test set (or overfitting). Such a model with high variance does not generalize on the new data. So, Lasso Regression comes for the rescue. It introduced an L1 penalty (or equal to the absolute value of the magnitude of weights) in the cost function of Linear Regression. The modified cost function for Lasso Regression is given below.

* Here, represents the weight for jth feature.
* n is the number of features in the dataset.
* lambda is the regularization strength.

Lasso Regression performs both variable selection and regularization too.

**Mathematical Intuition**

During gradient descent optimization, added l1 penalty shrunk weights close to zero or zero. Those weights which are shrunk to zero eliminate the features present in the hypothetical function. Due to this, irrelevant features don’t participate in the predictive model. This penalization of weights makes the hypothesis simpler which encourages sparsity (model with few parameters).

If the intercept is added, it remains unchanged.

We can control the strength of regularization by hyperparameter lambda. All weights are reduced by the same factor lambda.

Different cases for tuning values of lambda.

* If lambda is set to be 0, Lasso Regression equals Linear Regression.
* If lambda is set to be infinite, all weights are shrunk to zero.

If we increase lambda, bias increases if we decrease the lambda variance increase. As

lambda increases, more and more weights are shrunk to zero and eliminates features from

the model.

**How does the Lasso Regression work?**

To avoid overfitting and promote feature selection, Lasso Regression combines conventional linear regression with a regularization term. The linear regression loss function is expanded by the regularization term, denoted by λ. During training, this term drives some of the coefficients to exactly zero by penalizing their absolute values.

The objective of the model is to simultaneously minimize the sum of the absolute values of the coefficients and the sum of squared discrepancies between the actual and anticipated values. Only the most pertinent features are kept in a sparse model, which is the result of this dual optimization goal.

The following stages can be used to summarize how Lasso Regression operates:

* Set the intercept and coefficients to zero. Set the intercept and coefficients to zero.
* Gradient descent is used to update coefficients iteratively through training instances.
* Sparsity is enforced by the L1 penalty term, and certain coefficients may become exactly zero.
* By focusing on the most significant elements, the model adjusts to become a more straightforward and comprehensible model.

## 3.3 Extreme Gradient Boost Model

XGBoost Regression (eXtreme Gradient Boosting) initially started as a research project by Tianqi Chen as part of the Distributed (Deep) Machine Learning Community (DMLC) group in 2016.

XGBoost operates through an iterative and additive approach, wherein weak learners (typically decision trees) are sequentially added to the model to correct errors made by the existing ones. Each tree is trained to predict the residual errors of the previous ensemble, and their predictions are aggregated to produce the final output.

Mathematically, we can write our model in the form:

where, K is the number of trees, f is the functional space of F, F is the set of possible

CARTs.

The strength of XGBoost lies in its ability to optimize and combine a large number of

trees efficiently, taking into account their individual contributions to the overall

predictive power. The results in a highly accurate and robust regression model suitable

for a wide range of applications.

## 3.4 Random Forest

### Random Forest explain

**Random Forest** is one of the most popular and commonly used algorithms by Data Scientists. **Random forest is a Supervised Machine Learning Algorithm that is used widely in Classification and Regression problems**. It builds decision trees on different samples and takes their majority vote for classification and average in case of regression.

One of the most important features of the Random Forest Algorithm is that it can handle the data set containing continuous variables, as in the case of regression, and categorical variables, as in the case of classification. It performs better for classification and regression tasks.

* + - * + **Steps Involved in Random Forest Algorithm**

**Step 1:** In the Random forest model, a subset of data points and a subset of features is selected for constructing each decision tree. Simply put, n random records and m features are taken from the data set having k number of records.

**Step 2**: Individual decision trees are constructed for each sample.

**Step 3**: Each decision tree will generate an output.

**Step 4**: Final output is considered based on Majority Voting or Averaging for Classification and regression, respectively.

### Random Forests Algorithm



**The following steps explain the working Random Forest Algorithm: Step 1:** Select random samples from a given data or training set.

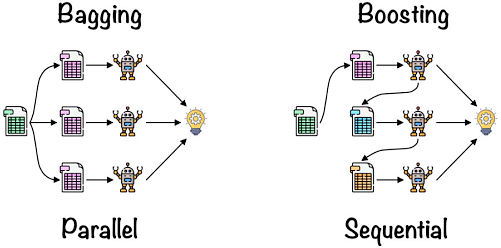
**Step 2**: This algorithm will construct a decision tree for every training data.

**Step 3**: Voting will take place by averaging the decision tree.

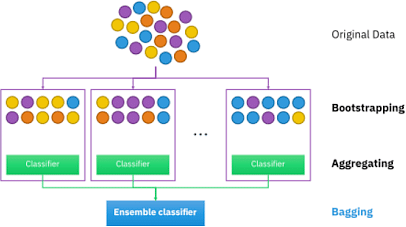
**Step 4**: Finally, select the most voted prediction result as the final prediction result.

**This combination of multiple models is called Ensemble. Ensemble uses two methods:**

* **Bagging:** Creating a different training subset from sample training data with replacement is called Bagging. The final output is based on majority voting.
* **Boosting:** Combing weak learners into strong learners by creating sequential models such that the final model has the highest accuracy is called Boosting. Example: ADA BOOST, XG BOOST.



BaggingBagging is also known as Bootstrap Aggregation used by random forest. The process begins with any original random data. After arranging, it is organised into samples known as Bootstrap Sample. This process is known as Bootstrapping.Further, the models are trained individually, yielding different results known as Aggregation. In the last step, all the results are combined, and the generated output is based on majority voting. This step is known as Bagging and is done using an Ensemble Classifier.



* + **Essential Features of Random Forest**
* **Miscellany:** Each tree has a unique attribute, variety and features concerning other trees.

Not all trees are the same.

* **Immune to the curse of dimensionality**: Since a tree is a conceptual idea, it requires no features to be considered. Hence, the feature space is reduced.
* **Parallelization:** We can fully use the CPU to build random forests since each tree is created autonomously from different data and features.
* **Train-Test split:** In a Random Forest, we don’t have to differentiate the data for train and test because the decision tree never sees 30% of the data.
* **Stability:** The final result is based on Bagging, meaning the result is based on majority voting or average.
  + **How Random Forest is applied?**

**Random Forest has a wide range of applications across various domains due to its versatility and robustness.**

* **Classification Problems**: Random Forest is often used for classification tasks, such as spam detection, sentiment analysis, customer churn prediction, disease diagnosis, and image

classification. Its ability to handle both numerical and categorical features makes it suitable for diverse datasets.

* **Regression Problems:** Random Forest can be applied to regression tasks, including predicting housing prices, stock market trends, energy consumption, and demand forecasting. It can capture complex non-linear relationships between input variables and the target variable.
* **Feature Importance**: Random Forest provides a measure of feature importance, which can be utilized for feature selection in data preprocessing. This information helps identify the most relevant features for prediction and can guide feature engineering efforts.
* **Anomaly Detection**: Random Forest can be used for anomaly detection by training on normal data and identifying instances that deviate significantly from the learned patterns. This is useful in fraud detection, network intrusion detection, and detecting unusual behaviors in various domains.
* **Ensemble Learning**: Random Forest is a type of ensemble learning method, which combines multiple models to improve overall performance. It can be used as a base learner in ensemble techniques such as bagging, boosting, and stacking to further enhance predictive accuracy.
* **Imputation of Missing Values**: Random Forest can handle missing data effectively. It can be used to impute missing values in a dataset by using the available features to predict missing values, making it valuable for data preprocessing tasks.
* **Recommender Systems**: Random Forest can be employed in building recommendation systems to suggest relevant products, movies, or content to users based on their preferences and behavior.
* **Bioinformatics and Genomics**: Random Forest finds applications in analyzing DNA sequences, gene expression data, and protein-protein interactions. It can be used for tasks like gene expression classification, protein structure prediction, and identifying disease biomarkers.

## 3.5 Gradient Boosting Model

The idea of gradient boosting originated in the observation by Leo Breiman that boosting can be interpreted as an optimization algorithm on a suitable cost function. Explicit regression gradient boosting algorithms were subsequently developed, by Jerome H. Friedman, simultaneously with the more general functional gradient boosting perspective of Llew Mason, Jonathan Baxter, Peter Bartlett and Marcus Frean.

Gradient Boosting is a popular boosting algorithm in machine learning used for classification and regression tasks.

Like other boosting methods, gradient boosting combines weak "learners" into a single strong learner in an iterative fashion. The goal is to "teach" a model F to predict values of the form by minimizing the mean squared error:

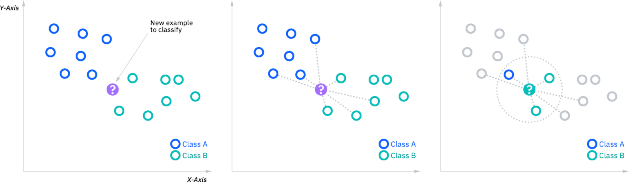
where i indexes over some training set of size n of actual values of the output variable y. ŷ equals the predicted value F(), equals the observed value, n equals the number of samples in y.

## 3.6 K-Nearest Neighbors

### K-Nearest Neighbors explain

The k-nearest neighbors algorithm, also known as KNN or k-NN, is a non-parametric, supervised learning classifier, which uses proximity to make classifications or predictions about the grouping of an individual data point. While it can be used for either regression or classification problems, it is typically used as a classification algorithm, working off the assumption that similar points can be found near one another.

For classification problems, a class label is assigned on the basis of a majority vote—i.e. the label that is most frequently represented around a given data point is used. While this is technically considered “plurality voting”, the term, “majority vote” is more commonly used in literature. The distinction between these terminologies is that “majority voting” technically requires a majority greater than 50%, which primarily works when there are only two categories. When you have multiple classes—e.g. four categories, you don’t necessarily need 50% of the vote to make a conclusion about a class; you could assign a class label with a vote of greater than 25%.



### K-nearest neighbors Algorithm

**Compute KNN: distance metrics**

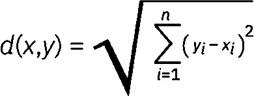
To recap, the goal of the k-nearest neighbor algorithm is to identify the nearest neighbors of a given query point, so that we can assign a class label to that point. In order to do this, KNN has a few requirements.

* **Determine your distance metrics**

In order to determine which data points are closest to a given query point, the distance between the query point and the other data points will need to be calculated. These distance metrics help to form decision boundaries, which partitions query points into different regions. You commonly will see decision boundaries visualized with Voronoi diagrams.

**While there are several distance measures that you can choose from, this article will only cover the following**:

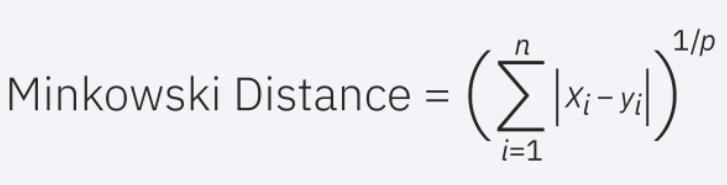
* **Euclidean distance (p=2):** This is the most commonly used distance measure, and it is limited to real-valued vectors. Using the below formula, it measures a straight line between the query point and the other point being measured.



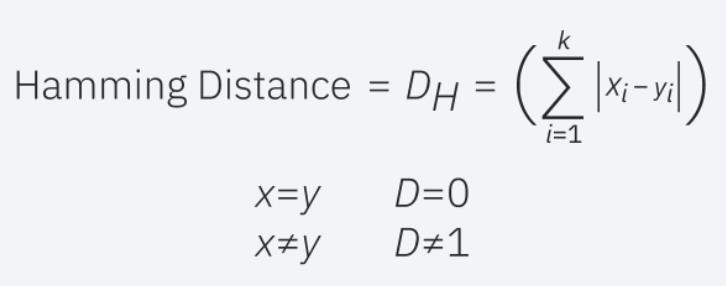
* + **Manhattan distance (p=1)**: This is also another popular distance metric, which measures the absolute value between two points. It is also referred to as taxicab distance or city block distance as it is commonly visualized with a grid, illustrating how one might navigate from one address to another via city streets.



* + **Minkowski distance**: This distance measure is the generalized form of Euclidean and Manhattan distance metrics. The parameter, p, in the formula below, allows for the creation of other distance metrics. Euclidean distance is represented by this formula when p is equal to two, and Manhattan distance is denoted with p equal to one.



* + **Hamming distance:** This technique is used typically used with Boolean or string vectors, identifying the points where the vectors do not match. As a result, it has also been referred to as the overlap metric. This can be represented with the following formula:



# CHAPTER IV: PREDICTIVE SOFTWARE AND EXPERIMENT

## 4.1 Predictive sofeware

### 4.1.1 Building Pipelines and Params Searching

Building pipelines and performing parameter searching are essential practices in machine learning to automate workflows, improve model performance, ensure reproducibility, and efficiently handle the complexities of model development and deployment.

We have 6 models, including Random Forest Regressor, XGBoost Regressor, K-

Nearest Neighbor Regressor, Gradient Boosting Regressor, Linear Regression and Lasso Regression. It is necessary to find suitable parameters for each of them.

For Lasso Regression, it can tune the following parameters: regressor\_\_alpha: The regularization strength. It controls the degree of regularization, where higher values result in more regularization. It is typically a positive value.

For Linear Regression, there are no specific parameters to tune as it is a simple linear model.

For Random Forest Regressor, it’s needed to define regressor\_\_n\_estimators,

regressor\_\_max\_depth.

For XGBoost Regressor, it’s needed to define regressor\_\_learning\_rate, regressor\_\_max\_depth, regressor\_\_min\_child\_weight, regressor\_\_gamma,

regressor\_\_colsample\_bytree.

For K-Nearest Neighbor Regressor, it’s needed to define regressor\_\_n\_neighbors.

For Gradient Boosting Regressor, it’s needed to define regressor\_\_max\_depth,

regressor\_\_max\_features, regressor\_\_min\_samples\_leaf, regressor\_\_min\_samples\_split, regressor\_\_n\_estimators.

A screen shot of a computer code

Description automatically generated

A screenshot of a computer program

Description automatically generated

### 4.1.2 Eval models and find best processed data

In terms of finding the most suitable models for this study, R-squared, Mean Absolute

Error (MAE), Root Mean Squared Error (RMSE) parameters will be used.

A computer code with text

Description automatically generated with medium confidence

A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screen shot of a graph

Description automatically generated

A screen shot of a graph

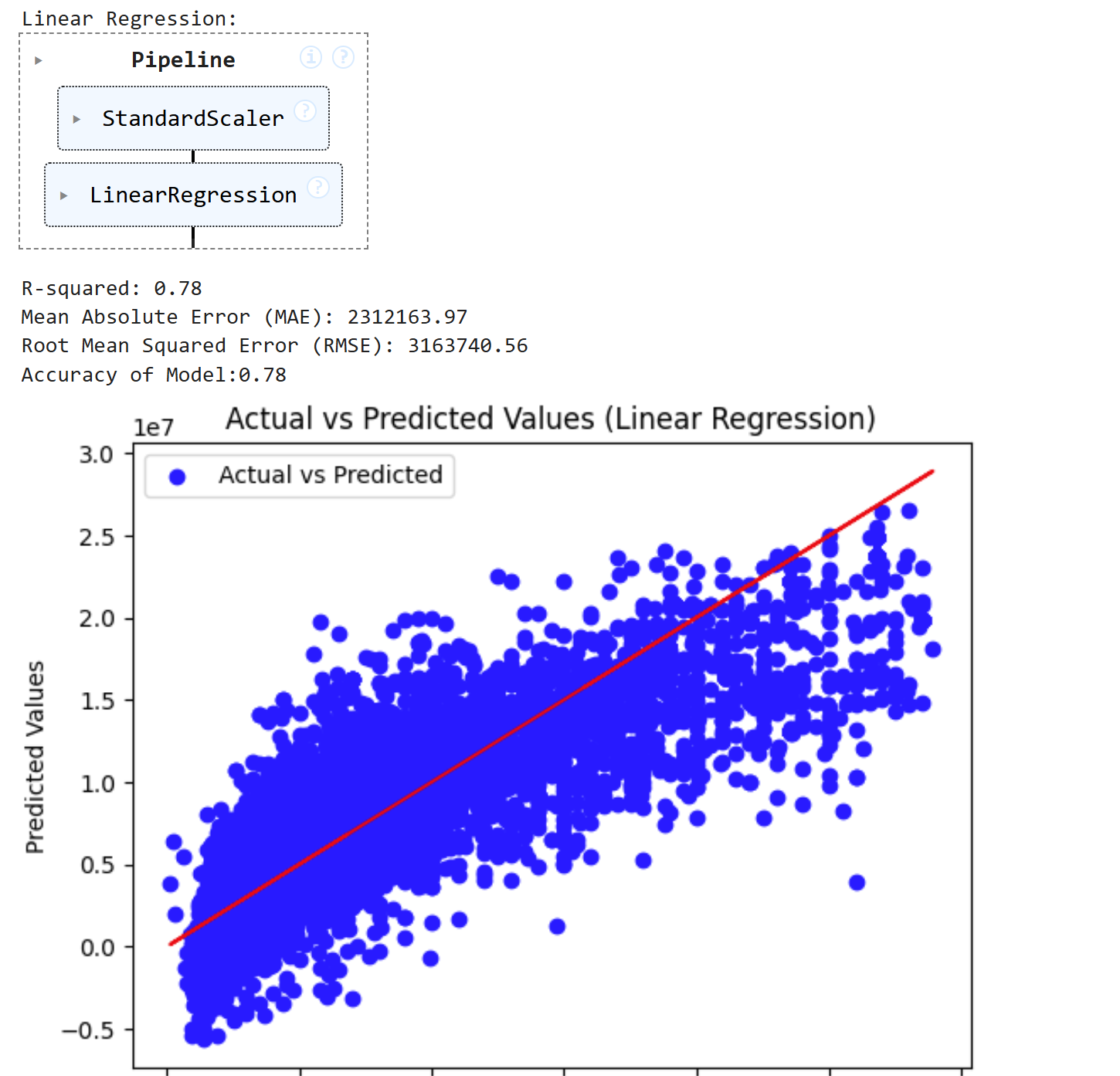
Description automatically generated

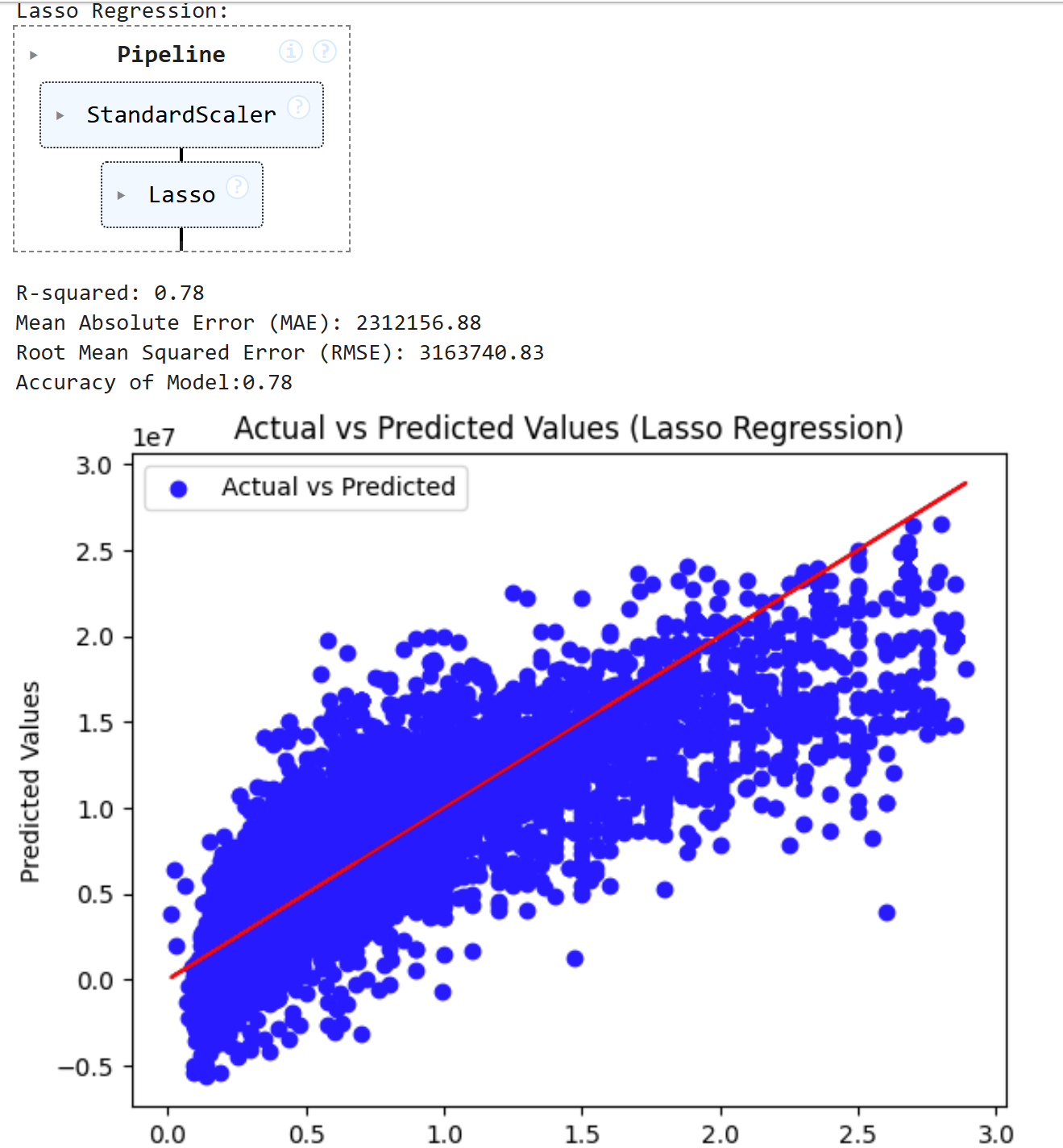
A screen shot of a graph

Description automatically generated

A screen shot of a computer screen

Description automatically generated





So, the best approach3 because it has the highest accuracy of Model (0.95%) so we will choose this approach.

A computer screen shot of a computer code

Description automatically generated

### 4.1.3 Models training for Demo application.

A computer code with many text

Description automatically generated with medium confidence

A white screen with black text

Description automatically generated

A screenshot of a computer program

Description automatically generated

## 4.2 Experimement

### 4.2.1 Dataset

**Original Dataset**

| **Attribute** | **Describe** | **Data Type** |
| --- | --- | --- |
| **Index** | Index or identifier for each entry in the dataset | Int64 |
| **Title** | Title of the property | Object |
| **Description** | Description providing additional details about the property | Object |
| **Amount (in rupees)** | The total amount associated with the property transaction (in rupees) | Object |
| **Price (in rupees)** | The price of the property (in rupees) | Float64 |
| **location** | Description of the property's location | Object |
| **Carpet Area** | Floor area of the property | Object |
| **Status** | - Status of the property  - Ex: Ready to move | Object |
| **Floor** | Number of floors in the property | Object |
| **Transaction** | - Type of transaction  - Ex: Resale, New Property | Object |
| **Furnishing** | Furnishing status  Ex: Furnished, Semi-Furnished, Unfurnished | Object |
| **facing** | Direction the property faces | Object |
| **overlooking** | View from the property | Object |
| **Society** | Name of the residential society | Object |
| **Bathroom** | Number of bathrooms | Object |
| **Balcony** | Number of balconies | Object |
| **Car Parking** | Availability of parking space | Object |
| **Super Area** | Total built-up area of the property | Object |
| **Dimensions** | Dimensions of the property | Float64 |
| **Plot Area** | Land area of the property | Float64 |

**Dataset after pre-processing**

| **Attribute** | **Describe** | **Data Type** |
| --- | --- | --- |
| **Amount (in rupees)** | The total amount associated with the property transaction (in rupees) | Float64 |
| **location** | Description of the property's location | Object |
| **Carpet Area (in sqft)** | Floor area of the property (in sqft) | Float64 |
| **Transaction** | - Type of transaction  - Ex: Resale, New Property | Object |
| **Furnishing** | Furnishing status  Ex: Furnished, Semi-  Furnished, Unfurnished | Float64 |
| **facing** | Direction the property faces | Object |
| **Bathroom** | Number of bathrooms | Int64 |
| **Balcony** | Number of balconies | Int64 |
| **Ownership** | Type of ownership  Ex: Freehold, Co-operative Society | Object |
| **Bedroom, Hall and Kitchen** | Number of bedroom, hall and kitchen | Object |
| **Floor location** | Location of floor | Float64 |
| **Total floor** | The total number of floors | Float64 |
| **overlooking\_Not Available** | View: Not Available | Float64 |
| **overlooking\_Main Road** | View: Main Road | Float64 |
| **overlooking\_Garden/Park** | View: Garden/Park | Float64 |
| **overlooking\_Pool** | View: Pool | Float64 |

### 4.2.2 Evaluation forecasting models

To evaluate the accuracy of the single and combined regression models, we use three parameters, mean absolute error (MAE), Root mean squared error (RMSE) and Accuracy of Model. The algorithm with the lowest value of those three parameters has the best performance.

#### 4.2.2.1 Mean Absolute Error (MAE)

MAE is calculated as the average of the absolute error between the actual value and the predicted value

With n is the number of predicted data samples

#### 4.2.2.2 Mean Squared Error (MSE)

MSE is defined as the average of squared error between actual value and predicted value

With n is the number of predicted data samples

#### 4.2.2.3 Root Mean Squared Error (RMSE)

With n is the number of predicted data samples

#### 4.2.2.4 Accuracy of Model

Model accuracy is defined as the number of classifications a model correctly predicts divided by the total number of predictions made.

#### 4.2.2.5 Predicting the price of house.

|  | **Linear Regression** | **Lasso Regression** | **Random Forest Regressor** | **XgBoost** | **KNeighbors** | **GradientBoosting**  **Regressor** |
| --- | --- | --- | --- | --- | --- | --- |
| **R-squared** | 0.78 | 0.78 | 0.94 | 0.94 | 0.92 | 0.95 |
| **Mean Absolute Error (MAE)** | 2312163.97 | 2312156.88 | 620608.62 | 752070.36 | 785336.78 | 626927.19 |
| **Root Mean**  **Squared**  **Error (RMSE)** | 3163740.56 | 3163740.83 | 1581404.82 | 1643808.42 | 1955964.14 | 1555986.96 |
| **Accuracy of**  **Model** | 0.78 | 0.78 | 0.94 | 0.94 | 0.92 | 0.95 |

# CHAPTER V: CONCLUSION

Summary, the regression algorithm applied for predicting house prices has shown positive results in our analysis. From the analysis results, we can observe that the Gradient Boosting algorithms is the algorithm with the highest accuracy, as MAE and RMSE have the lowest values. Next is the XgBoost algorithm and

random forest algorithm. Finally, the Linear Regression and Lasso Regression algorithm are two algorithms with the lowest accuracy, as MAE and RMSE have the highest values.

However, it's important to note that predicting house prices is a complex issue with many influencing factors. While our model has achieved positive results, additional scrutiny may be needed for stability and generalization of the model across various real-world scenarios.

In conclusion, the applied regression algorithm for predicting house prices has shown promise, providing a useful tool to meet the demand for predicting house prices for users. Fine-tuning and maintaining sensitivity to market dynamics are crucial to maintaining the realism and effectiveness the model over time

# REFERENCES

##### <https://www.stat.berkeley.edu/~breiman/randomforest2001.pdf>

1. [**https://www.analyticsvidhya.com/blog/2021/06/understanding-random-forest/**](https://www.analyticsvidhya.com/blog/2021/06/understanding-random-forest/)
2. [**http://myweb.sabanciuniv.edu/rdehkharghani/files/2016/02/The-Morgan-Kaufmann-**](http://myweb.sabanciuniv.edu/rdehkharghani/files/2016/02/The-Morgan-Kaufmann-Series-in-Data-Management-Systems-Jiawei-Han-Micheline-Kamber-Jian-Pei-Data-Mining.-Concepts-and-Techniques-3rd-Edition-Morgan-Kaufmann-2011.pdf)[**Series-in-Data-Management-Systems-Jiawei-Han-Micheline-Kamber-Jian-Pei-Data-**](http://myweb.sabanciuniv.edu/rdehkharghani/files/2016/02/The-Morgan-Kaufmann-Series-in-Data-Management-Systems-Jiawei-Han-Micheline-Kamber-Jian-Pei-Data-Mining.-Concepts-and-Techniques-3rd-Edition-Morgan-Kaufmann-2011.pdf)[**Mining.-Concepts-and-Techniques-3rd-Edition-Morgan-Kaufmann-2011.pdf**](http://myweb.sabanciuniv.edu/rdehkharghani/files/2016/02/The-Morgan-Kaufmann-Series-in-Data-Management-Systems-Jiawei-Han-Micheline-Kamber-Jian-Pei-Data-Mining.-Concepts-and-Techniques-3rd-Edition-Morgan-Kaufmann-2011.pdf)
3. [**https://www.ibm.com/docs/en/spss-modeler/saas?topic=nodes-support-vector-machine-**](https://www.ibm.com/docs/en/spss-modeler/saas?topic=nodes-support-vector-machine-models)[**models**](https://www.ibm.com/docs/en/spss-modeler/saas?topic=nodes-support-vector-machine-models)
4. [**https://scikit-learn.org/stable/modules/svm.html**](https://scikit-learn.org/stable/modules/svm.html)