



 slington college
(इरिलिङ्टन कलेज)

Module Code & Module Title

Level 6 – Applied Machine Learning

Assessment Type

Semester 5

2024/25 Autumn

Student Name: Hridaya Giri

London Met ID: 22067317

College ID: np01ai4a220034@islingtoncollege.edu.np

Assignment Due Date: Thursday, January 16, 2025

Assignment Submission Date: Thursday, January 16, 2025

Submitted To: Mahotsav Bhattarai

Word Count (Where required):6506

22067317 Hridaya Giri 2.docx

 Islington College, Nepal

Document Details

Submission ID

trnoid::361879324963

Submission Date

Jan 16, 2025, 12:01 PM GMT+5:45

Download Date

Jan 16, 2025, 12:06 PM GMT+5:45

File Name

22067317 Hridaya Giri 2.docx

File Size

48.5 KB

56 Pages





8,051 Words

40,969 Characters




17% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Match Groups

-  **152 Not Cited or Quoted 17%**
Matches with neither in-text citation nor quotation marks
-  **4 Missing Quotations 0%**
Matches that are still very similar to source material
-  **0 Missing Citation 0%**
Matches that have quotation marks, but no in-text citation
-  **0 Cited and Quoted 0%**
Matches with in-text citation present, but no quotation marks

Top Sources

- 4%  Internet sources
- 5%  Publications
- 16%  Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

Match Groups

- 152 Not Cited or Quoted 17%**
Matches with neither in-text citation nor quotation marks
- 4 Missing Quotations 0%**
Matches that are still very similar to source material
- 0 Missing Citation 0%**
Matches that have quotation marks, but no in-text citation
- 0 Cited and Quoted 0%**
Matches with in-text citation present, but no quotation marks

Top Sources

- 4% Internet sources
- 5% Publications
- 16% Submitted works (Student Papers)

Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

| | | | |
|----|-----------------|---|-----|
| 1 | Submitted works | islingtoncollege on 2024-12-24 | <1% |
| 2 | Submitted works | University of Hertfordshire on 2023-09-17 | <1% |
| 3 | Submitted works | Taylor's Education Group on 2024-07-12 | <1% |
| 4 | Internet | eitca.org | <1% |
| 5 | Submitted works | Coventry University on 2023-04-11 | <1% |
| 6 | Submitted works | University of Sydney on 2024-09-18 | <1% |
| 7 | Submitted works | islingtoncollege on 2024-12-24 | <1% |
| 8 | Publication | Przemysław Biecek, Tomasz Burzykowski. "Explanatory Model Analysis - Explore, ... | <1% |
| 9 | Internet | www.mdpi.com | <1% |
| 10 | Submitted works | University of Essex on 2025-01-06 | <1% |

Table of Contents

| | |
|---|----|
| Table of Figures | 7 |
| 1. Introduction | 1 |
| 1.1. Machine learning concept used | 2 |
| 2. Problem Domain | 3 |
| 2.1. Problem scenario | 3 |
| 2.2. Background on dataset | 4 |
| 3. Solution | 6 |
| 3.1. Project as solution..... | 6 |
| 3.2. Algorithm Used | 6 |
| 3.3. Pseudo code | 10 |
| 3.3.1. Pseudo code of Linear Regression implementation..... | 10 |
| 3.3.2. Pseudo code of Random Forest implementation | 11 |
| 3.4. State Transition Diagram | 12 |
| 3.5. Tools, technique and library used | 13 |
| 3.5.1. Tools..... | 13 |
| 3.4.2. Libraries | 16 |
| 3.4.3. Dataset | 18 |
| 4. Results | 19 |
| 4.1. Exploratory Data Analysis (EDA) | 19 |
| 4.1.1. Distribution of columns histogram | 19 |
| 4.1.2. Box plot of the columns..... | 20 |
| 4.1.3. Correlation HeatMap..... | 21 |
| 4.1.4. Scatter plot for LAND and PRICE | 22 |
| 4.2. Models Prediction | 23 |
| 4.2.1. Initial Prediction by linear regression | 23 |
| 4.2.2. Final prediction by linear regression after improvement..... | 24 |
| 4.2.3. Initial Prediction by random forest regressor..... | 25 |
| 4.2.4. Final prediction by random forest regressor model after improvement | 26 |
| 4.3. Comparison of the model prediction with actual data..... | 28 |
| 4.3.1. Initial Comparison | 28 |
| 4.3.2. Final Comparison after improvement..... | 31 |

| | | |
|--------|---|----|
| 4.4. | Evaluation of the model with performance metrics..... | 34 |
| 4.4.1. | Initial Evaluation of the model | 34 |
| 4.4.2. | Final Evaluation of the model after improvement..... | 37 |
| 4.5. | Techniques applied for model improvement | 40 |
| 4.5.1. | Increasing test size and random state adjustment..... | 40 |
| 4.5.2. | Using IQR method for outlier removal..... | 41 |
| 4.5.3. | Hyper parameter tuning random forest model | 45 |
| 4.5.4. | Polynomial Feature engineering | 48 |
| 4.6. | Comparison between linear regression and random forest regressor model .. | 50 |
| 4.6.1. | Initial Comparison | 50 |
| 4.6.2. | Final comparison after improvement..... | 51 |
| 4.7. | Result visualization | 53 |
| 4.7.1. | Model evaluation comparison using bar graph | 53 |
| 4.7.2. | Analyze pattern in error using Residual plot | 54 |
| 4.7.3. | Feature importance using bar graph..... | 56 |
| 5. | Conclusion | 57 |
| 5.1. | Model evaluation and effectiveness..... | 57 |
| 5.2. | Real world applicability | 58 |
| 5.3. | Suggestions for future improvements | 58 |
| 6. | References..... | 60 |
| 7. | Bibliography | 61 |

Table of Figures

| | |
|---|----|
| Figure 1: Figure representing Linear Regression | 7 |
| Figure 2: Figure Representing Random Forest | 9 |
| Figure 3: State Transition Diagram | 12 |
| Figure 4: Google Collab | 13 |
| Figure 5: Draw.io | 14 |
| Figure 6: MS Word | 15 |
| Figure 7: Pandas Logo | 16 |
| Figure 8: NumPy Library | 16 |
| Figure 9: Scikit learn logo | 17 |
| Figure 10: Kaggle Logo | 18 |
| Figure 11: Distribution of columns histogram | 19 |
| Figure 12: Box plot of the columns | 20 |
| Figure 13: Correlation heatmap of all columns | 21 |
| Figure 14: Scatter plot for LAND and PRICE | 22 |
| Figure 15: Initial Prediction by linear regression | 23 |
| Figure 16: Final prediction by linear regression after improvement | 24 |
| Figure 17: Final prediction in frontend by linear regression after improvement | 24 |
| Figure 18: Initial Prediction by random forest regressor model | 25 |
| Figure 19: : Initial Prediction in frontend by random forest regressor model | 25 |
| Figure 20: Final prediction by random forest regressor model after improvement | 26 |
| Figure 21: Final prediction in front end by random forest regressor model after improvement | 27 |
| Figure 22: Initial scatter plot Comparison | 28 |
| Figure 23: Initial Line graph Comparison | 29 |
| Figure 24: Initial predicted vs actual value by linear regression | 29 |
| Figure 25: Initial predicted vs actual value by random forest | 30 |
| Figure 26: Final scatter plot Comparison for linear regression | 31 |
| Figure 27: Final linear graph Comparison for random forest | 32 |
| Figure 28: Initial predicted vs actual value by Random Forest | 32 |
| Figure 29: Initial predicted vs actual value by linear regression | 33 |
| Figure 30: Initial performance metrics Evaluation code of the models | 34 |
| Figure 31: Initial performance metrics result of the linear regression model | 35 |
| Figure 32: Initial performance metrics result of the random forest model | 36 |
| Figure 33: final performance metrics Evaluation code of the models | 37 |
| Figure 34: Final Linear Regression performance metrics evaluation | 38 |
| Figure 35: Final Random Forest performance metrics evaluation | 39 |
| Figure 36: test size and random state Before | 40 |
| Figure 37: Output before | 40 |
| Figure 38: test size and random state After | 40 |
| Figure 39: Output After | 40 |
| Figure 40: Using IQR method for outlier removal | 41 |

| | |
|---|----|
| Figure 41: Performance metrics evaluation before in linear regression..... | 42 |
| Figure 42: Performance metrics evaluation after in linear regression | 42 |
| Figure 43: Performance metrics evaluation before in random forest..... | 43 |
| Figure 44: Performance metrics evaluation after in random forest..... | 44 |
| Figure 45: Code to hyper parameter tune random forest model..... | 45 |
| Figure 46: Result before hyper parameter tuning | 46 |
| Figure 47: Result after hyper parameter tuning | 47 |
| Figure 48: Code for polynomial feature engineering | 48 |
| Figure 49: Result before polynomial feature engineering for linear regression | 49 |
| Figure 50: Result after polynomial feature engineering for linear regression | 49 |
| Figure 51: Initial comparison of performance metrics of two models..... | 50 |
| Figure 52: Final comparison of random forest model after improvement | 51 |
| Figure 53: Final comparison of Linear regression model after improvement..... | 51 |
| Figure 54: Model evaluation comparison using bar graph..... | 53 |
| Figure 55: Analyse pattern in error of random forest..... | 54 |
| Figure 56: Analyse pattern in error of Linear regression | 54 |
| Figure 57: Feature importance using bar graph | 56 |

1. Introduction

The real estate market of Nepal is projected to reach the value of NPR US\$437.20bn in 2024 despite the political instability and limited infrastructure in the country the real estate market is getting steadily growth due to increase demand from the investors (statista, 2024). Since the growth of the market is growing every year there are tone of problems that arises.

Buyer faces the problem of overprice prices of the house and it is estimated that around 50% the price of the house is appreciated which makes a lot difficult for the buyer to stay on the budget and buy which makes then reconsider their choice and location (Dickson Realty Team, 2024). For the sellers they also have to deal with lot of things majority they face the problem of setting the right price in the market, many sellers make a mistake of overpricing or sometime under-pricing due to not enough research or knowledge about the local market their house is in (thepropertyist, 2023).

For this I will be developing a functional prototype that predicts the price of the house using multiple predictive models to enhance accuracy , the buyer or the seller both can predict the price of the house they are buying or selling, the seller can make informed decisions and set the price of their house in the real state market based on the output by the system as it is based on the other houses that are sold and according to the market trend, the user of the system have to give necessary input in the system like the location, area, number of bathrooms, number of floors, bedroom, parking and many more and then the price of the house is determined which will help them tp make informed decisions. After that the seller can sell their house accordingly with the right price and avoid being under or over price and same for the buyer who can avoid being scammed or overpriced on the house they are buying if they use the system and predict the price before they are paying to the seller.

1.1. Machine learning concept used

Since in this project we have both input and output clearly identified and the project needs to predict the price of the house which is a well identified goal compared to other concept like unsupervised learning it works better and faster. Since my dataset is also labelled and the target variable is also known beforehand supervised learning is the most ideal choice.

There are two types of supervised learning:

1. Classification

This is the type of supervised learning in which if the prediction is done in category and not in numerical value, for example if the house prediction system would predict the price in range like low, high or medium it is a classification approach.

2. Regression

This is the type of supervised learning in which if the prediction is done in continuous numerical values for example if the house prediction system would predict the price of the house as NPR 250000000 it is a regression approach

Since we are predicting the output as a continuous numerical value and not in categorical approach, we are using supervised learning in regression approach and not in categorical approach.

2. Problem Domain

2.1. Problem scenario

The real estate market of Nepal will grow every year, and it is identified that it will grow more and more every year. In the context of seller setting the right price of the house is the major concern, if they overprice their house or underprice their house different problem will arise, since in this real estate market 80% are well educated and they do their research before buying and if they see the house is being overpriced they won't bother seeing it and when the home is overpriced it sits for a long time in the market and people might think the house might be a problem as it is too long in the market and not sold yet so they won't bother (Tony Mark, Russell Grether, 2024).

In context of the buyers who are buying the house 98% buyers expect to face any kind of problems, 57% see the biggest challenge as "finding the right home, 49% cited "staying in my budget 23% are uncertain about "managing closing costs (Lentz, 2024). Sometime the buyer tends to overpay based on future market condition rather than the current one and which leads to loss in future as there is a high chance the amount the buyer might pay based on future market condition might not occur in the future (dolinskigroup, 2024).

So the major problem is about the cost of the house, for the buyer they are entirely concerned about that if they are buying the house in right amount and if the location is good, and for the seller around 34% seller face challenge of pricing the house correctly, they are more over concerned about if they underprice or overprice their house, as if underprice they might bear a lot of loss and if overprice they might lose some potential customers.

This is one of the most growing concern and predicting the price of the house is also one of the most challenging task, since the house price is affected by multiple factors and the house price is also not static as nearly half of sellers (47%) and buyers

(45%) expect home prices in their local market to go up in 2025 (Lentz, 2024). it is challenging task to provide accurate output so that the buyer and seller can rely even though technology has come a long way people often don't trust this type of system and follow their traditional approach so proper marketing is very much required so that the people in real estate might use this type of system and decrease the problems they face in the market.

2.2. Background on dataset

The house price prediction system to work properly and for it to show proper output is heavily dependent on the dataset, its quality is the major factor and the factors that affect the most for this prediction are required. The source from which the data is obtained is also very crucial so that the prediction is accurately done.

In this project to do the prediction I am using the dataset from Kaggle, Kaggle is one of the most reliable and trusted sources where numerous datasets can be obtained and I have obtained dataset of housing price of Nepali, it is a Nepali housing price dataset where numerous factors are kept with the target variable being the price of the house. In this dataset there are 13 columns which play a vital factor in predicting the price of the house,

The 13 columns and their short description are:

- Title: Title of the post which the user has used while listing the house online
- Location: Address of the house where the house is located
- City: district where the house is in
- Price: price of the house
- Bedroom: number of bedrooms in the house
- Bathroom: number of bathrooms in the house
- Floors: total number of floors in the house
- Parking: number of parking spaces the house has

- Facing: The direction the property is facing
- Land Area: land area occupied by the house
- Road access: road size and width
- Build year: The year the house is built in.
- Amenities: Other facilities the house has like backyard, balcony, internet and many more

Since the dataset is not cleaned properly and heavy number of null values, heavy number of data imbalance and many other issues is seen heavily data cleaning and data preprocessing is required before any prediction is done or else the output might be inaccurate, and which might cause the people in the real estate market to trust less on this type of system. The dataset can be seen from the given link [House Price Dataset Nepal](#)

3. Solution

3.1. Project as solution

The project of predicting the house price will play as a vital solution in this growing real estate market; by using 2 types of machine learning algorithms, the prediction of the house price is done, technology has come a long way so the technology will play a bigger role, and it is estimated that 97% buyer will use some sort of tech so the house price prediction comes in handy (Lentz, 2024). Since the major concern for both the buyer and seller is related to the price, for seller being the risk of over pricing or underpricing their house and for buyer major concern will be overpaying so this system plays a vital role in predicting the price with high accuracy so that the buyer and seller could rely on this and the problem of overpaying by the buyer and overpricing or underpricing by the seller might reduce in the real estate market.

The projects depend on various factors like location, city, number of bedrooms, number of bathrooms, floors, parking, roads and many more the prediction is heavily dependent on this type of factors and the output is also obtained according to the user input.

3.2. Algorithm Used

For the system to give high accuracy output 2 machine learning algorithms are used:

1. Linear regression

Linear regression is one of the most popular regression-based machine learning algorithms. This algorithm assumes a linear relationship between the independent variable and dependent variable also known as target variable in our case it is house price, to predict proper and accurate price of the house.

It predicts the price of the house by this formula:

$y = mx + b$ where,

y is the predicted price of the house also known as dependent variable

x is the input feature and known as independent variable which plays the vital role to obtain the accurate and high accuracy output

m is the slope which represent the impact of each independent variable with the dependent variable

b is the intercept which is the baseline price when all the features are 0

so, for multiple features the equation will be expanded to

$$y = m_1x_1 + m_2x_2 + m_2x_2 + m_3x_3 + \dots + b$$

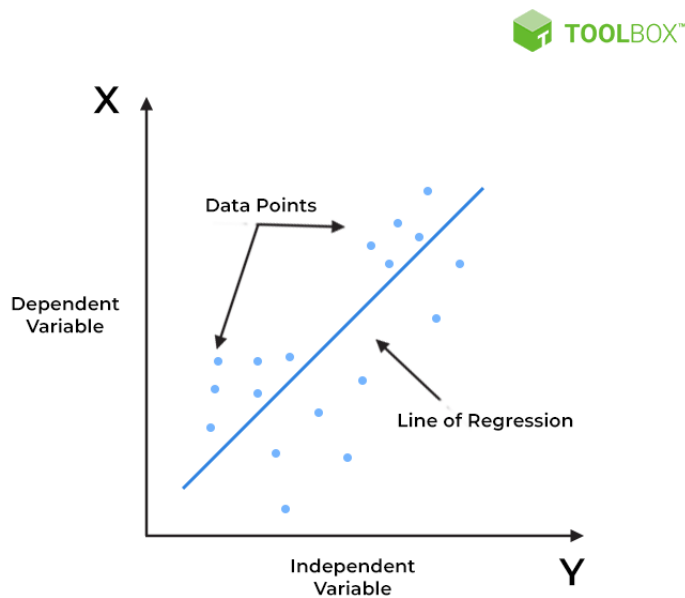


Figure 1: Figure representing Linear Regression

As in the above figure the goal of linear regression is to find the best fit line, since both dependent and independent variable is known as the relation between the feature variable and target is mostly linear, linear regression is the best approach

Since in our dataset the independent and dependent variable are accurately defined and there is a clear relationship between the features variable and the target variable, which is house price, they are linear for example

If the house has more rooms, then the price of the house will be more expensive

It works well when there is a clear and linear relationship between target variable and feature as linear regression will be based approach for more accurate result.

2. Random forest model

Random forest is one of the most popular machine learning algorithms in the context of supervised learning, It is one of the popular ensemble technique where multiple decision trees are used for the prediction in which each trees independently predicts the price of the house and since it is a regression based the average from all the trees are calculated and then the final output is generated,

Random forest algorithm is very suitable to work with as it also handles the nonlinear relationship between the feature and target variable

For example in context of linear regression the cost of the house might depend on the location it is located at , if the house is in jhamsikhel it would cost higher but in context to random forest the house in jhamsikhel may cost high but this increase depends on few other factors like the number of bedrooms in the house, if parking is available or not and many more so random forest is very suitable to work with.

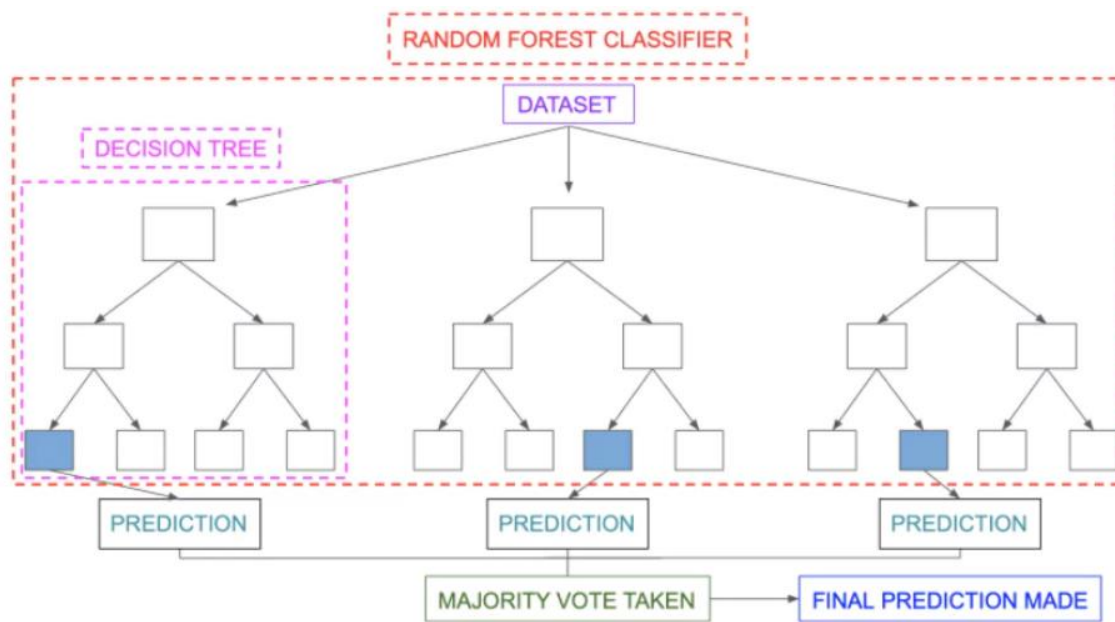


Figure 2: Figure Representing Random Forest

As in the above figure an individual decision tree is created on each of the subsets we take and each of those decision tree will provide an output and then the final output is provided by averaging all the output from all the decision tree

3.3. Pseudo code

3.3.1. Pseudo code of Linear Regression implementation

IMPORT necessary libraries and frameworks.

LOAD data into a DataFrame.

CLEAN data by handling missing values and duplicates.

MANIPULATE ROAD column by converting "12 Feet" to 12 (numerical).

CONVERT LAND column by changing "12 Aana" to 12 (numerical in Aana).

CALCULATE AGE column by subtracting the built year from the current year.

ENCODE top 5 cities using label encoding.

CONVERT PRICE column by converting "Rs. 2.5 Cr" to 2.5.

TRANSFORM price, land, city, and road data types.

FEATURE ENGINEER new features like $\text{LAND_SQUARED} = \text{LAND} ** 2$.

REMOVE outliers using IQR for the price column.

SPLIT data into training and testing sets.

SCALE the training and testing data.

TRAIN the Linear Regression model on the training data.

EVALUATE the model using R^2 , MAE, MSE, and RMSE on the test data.

PREDICT house prices using the trained model.

VISUALIZE results with scatter plots, residual plots, and bar graphs.

3.3.2. Pseudo code of Random Forest implementation

IMPORT necessary libraries and frameworks.

LOAD data into a DataFrame.

CLEAN data by handling missing values and duplicates.

MANIPULATE ROAD column by converting "12 Feet" to 12 (numerical).

CONVERT LAND column by changing "12 Aana" to 12 (numerical in Aana).

CALCULATE AGE column by subtracting the built year from the current year.

ENCODE top 5 cities using label encoding.

CONVERT PRICE column by converting "Rs. 2.5 Cr" to 2.5.

TRANSFORM price, land, city, and road data types.

FEATURE ENGINEER new features like $\text{LAND_SQUARED} = \text{LAND} ** 2$.

REMOVE outliers using IQR for the price column.

SPLIT data into training and testing sets.

SCALE the training and testing data.

TRAIN the Random Forest model on the training data.

TUNE hyperparameters using Grid Search.

EVALUATE the model using R^2 , MAE, MSE, and RMSE on the test data.

PREDICT house prices using the trained model.

VISUALIZE results with scatter plots, residual plots, and bar graphs.

3.4. State Transition Diagram

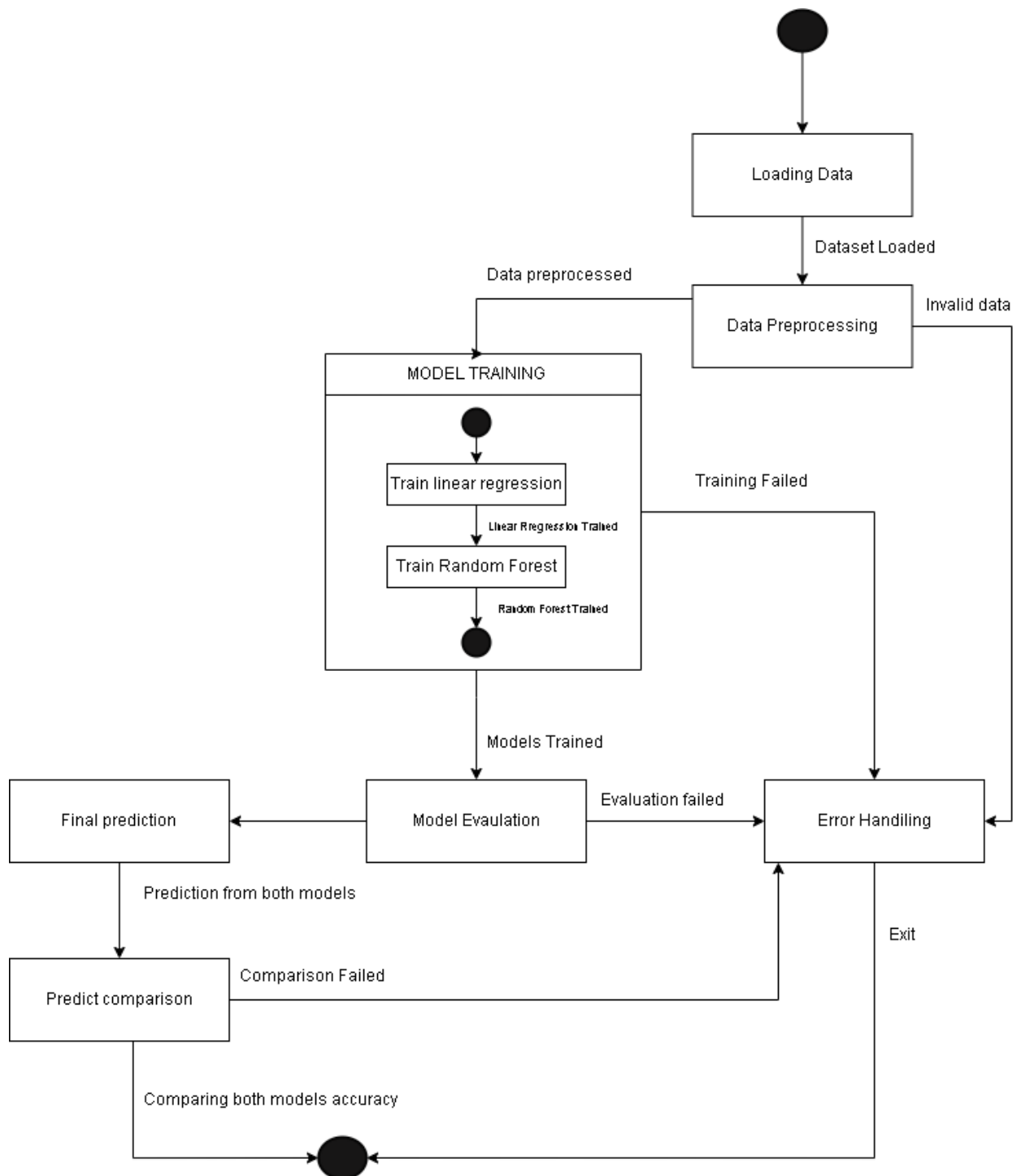


Figure 3: State Transition Diagram

3.5. Tools, technique and library used

While in the development of the house price prediction system different tools, techniques, and library are used and all are listed and explained below

3.5.1. Tools

1. Google Collab

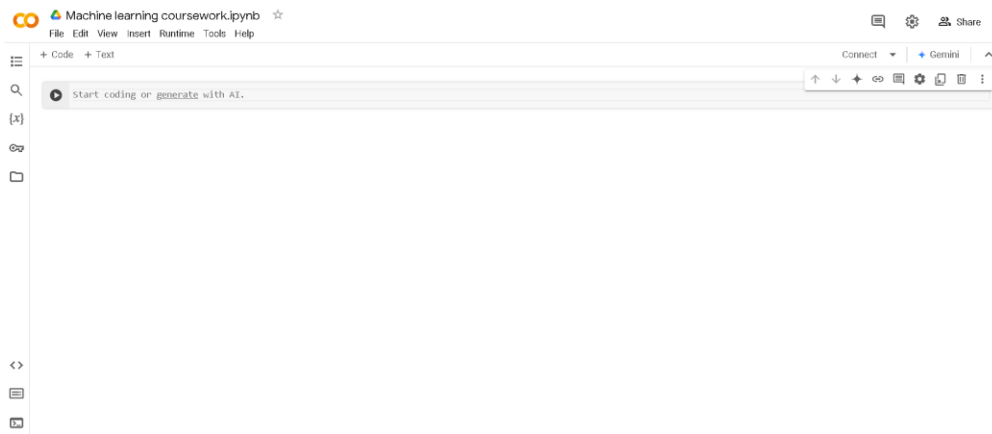


Figure 4: Google Collab

Google collab is one of the popular development environments and for this project I am using this, it is a cloud-based development environment in python. All the process from data preprocessing to model training all are done in Google collab, this is preferred because of its feature to not needed to install libraries which are needed in other development environments like Jupyter notebook.

2. Draw.io

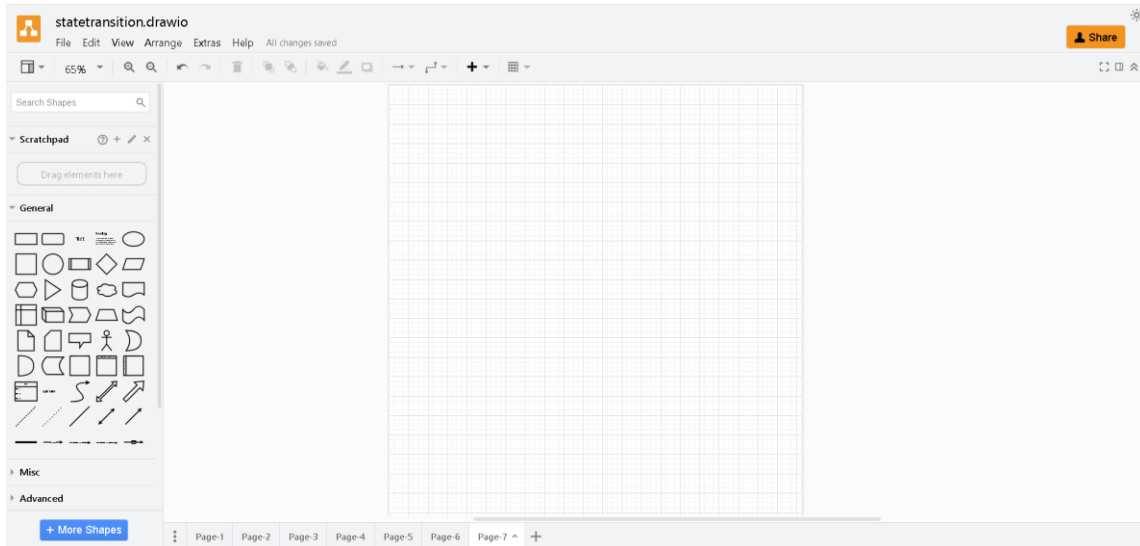


Figure 5: Draw.io

Draw.io is one of the most popular tools to create visual diagrams to represent the project flow and make the end user understand the system properly. Different diagrams like flowchart, state transition diagram is made using draw.io

3. Ms Word

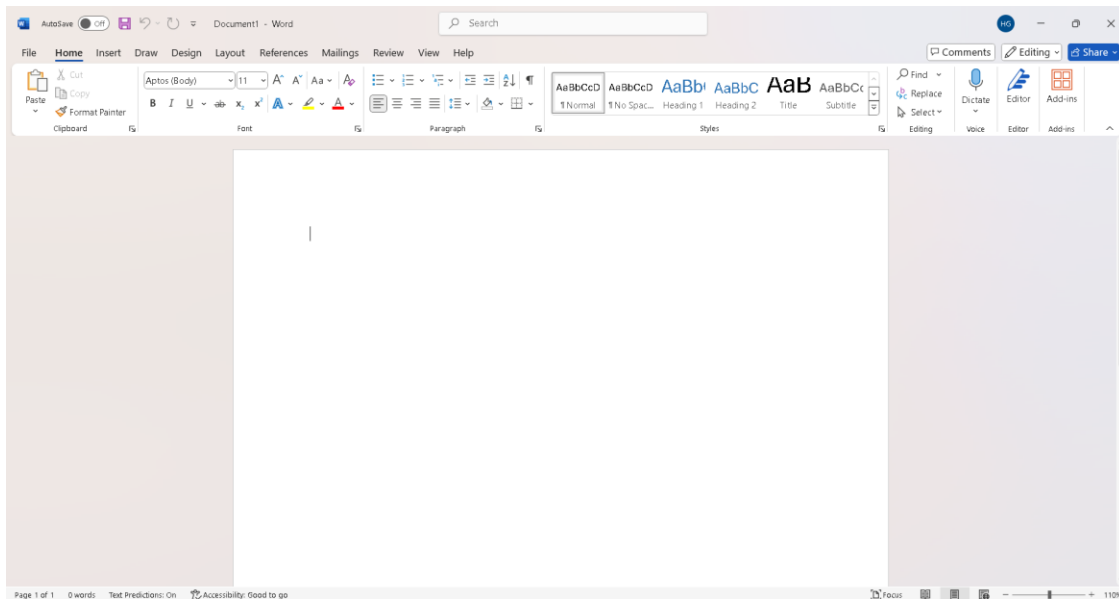


Figure 6: MS Word

Ms word is one of the most popular tools for the documentation approach of any system. All the milestones of the projects are done in Microsoft word and the final report, it allows to keep the documentation clean and professional

3.4.2. Libraries

1. Pandas



Figure 7: Pandas Logo

Pandas is one of the most popular and powerful data analysis and data science tools used for data manipulation which is built on python programming language. In the system panda's library is used for the data manipulation, such as cleaning data, handling missing or null values and many more for all these before the dataset is send to the training or testing purpose pandas is used

2. Numpy



Figure 8: NumPy Library

NumPy is also one of the popular libraries also known as numerical python, NumPy is used for numerical operations while developing a system as it is also built on python, and it is helpful in data processing for any machine learning task

3. Scikit learn

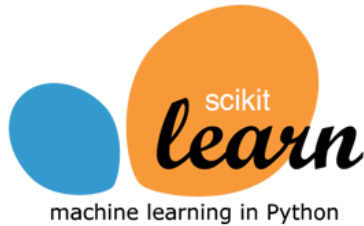


Figure 9: Scikit learn logo

Scikit learn is one of the popular and open source machine learning library which is vastly used to carry out machine learning task, this is the core library for this project as the two machine learning models learn, regression and random forest we are using in this project are inside scikit learn , it also provides the implementation to train the data or test the data to fit the data in the model for all scikit learn library is used

4. Matplotlib/Seaborn

Matplotlib and Seaborn are one of the most popular libraries for the visualisation purpose after both the models are trained tested and evaluated one of this two library is used for the visualisation purpose to understand the difference properly

3.4.3. Dataset

1. Kaggle



Figure 10: Kaggle Logo

Kaggle is one of the most popular data sciences open-source community where we can obtain large number of datasets from here, for this project I have obtained a Nepali housing price dataset from Kaggle which can be useful for development of this system.

The dataset can be seen from the given link [House Price Dataset Nepal](#)

4. Results

4.1. Exploratory Data Analysis (EDA)

4.1.1. Distribution of columns histogram

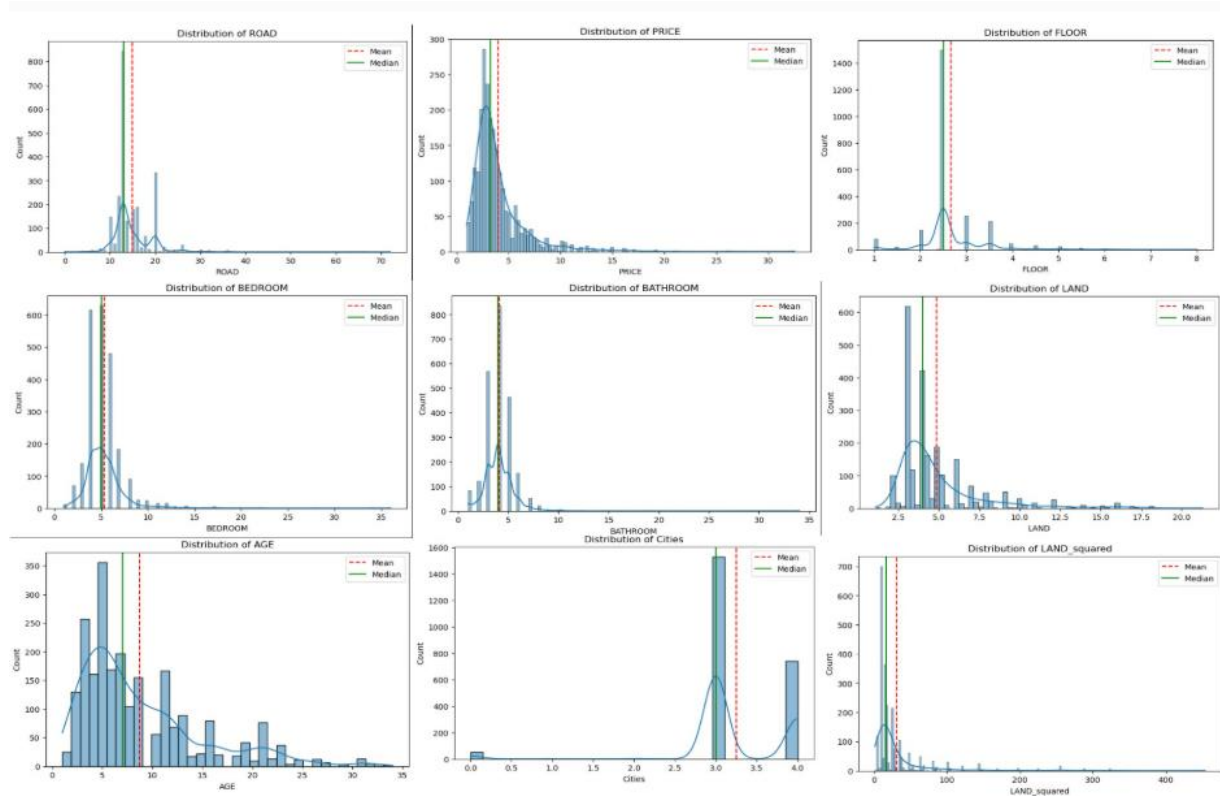


Figure 11: Distribution of columns histogram

In the above diagram the histogram is plotted for each numerical column to visualize how the data is spread. This analysis is important for preparing the data for modeling, as it helps identify issues like skewness or outliers which can be very crucial in getting the accuracy.

As the value in PRICE column is in Cr the distribution of PRICE column value is too much which needs to be handled as there might be tone of outliers in this column too.

4.1.2. Box plot of the columns

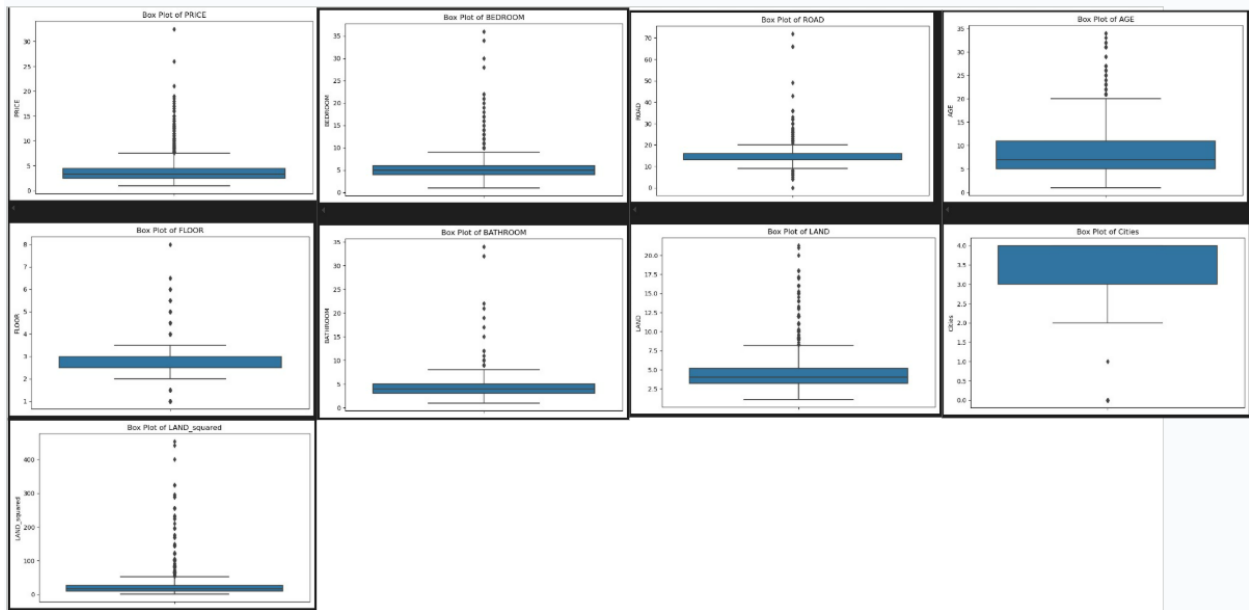


Figure 12: Box plot of the columns

In the above diagram the box plot is plotted to visualize the outliers in each column. This analysis is important for preparing the data for modeling, as it helps identify outliers which can be removed in future to get high accuracy.

As from the above figure since the values in PRICE column is in Cr the outliers in PRICE column is too much which needs to be handled before training and testing with the model.

4.1.3. Correlation HeatMap

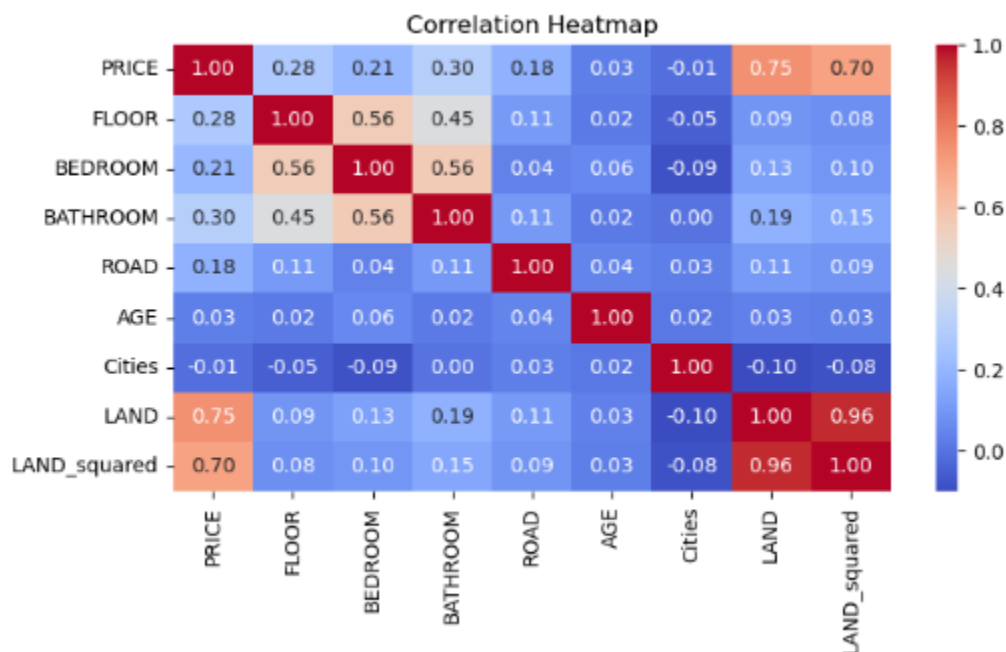


Figure 13: Correlation heatmap of all columns

In the above figure we can see the correlation heatmap of all the columns among each other. All the columns are listed and the correlation of each with each other are shown

From the above heatmap we can analyze that the LAND column is strong positive relationship as it is 0.75 with the target column that is PRICE, so this indicates that the most effect on the price is by the LAND column when the LAND increase so does the PRICE. LAND_squared column is also strongly correlated as it is also because polynomial feature engineering was done with the LAND column. And other columns like FLOOR, BEDROOM, BATHROOM also moderately positive relationship with the PRICE as it is around 0.30 or 0.28 which also has a good impact on the increment of the PRICE.

But the Cities column has negative correlation with the PRICE column when the Cities value decreases the PRICE increases.

4.1.4. Scatter plot for LAND and PRICE

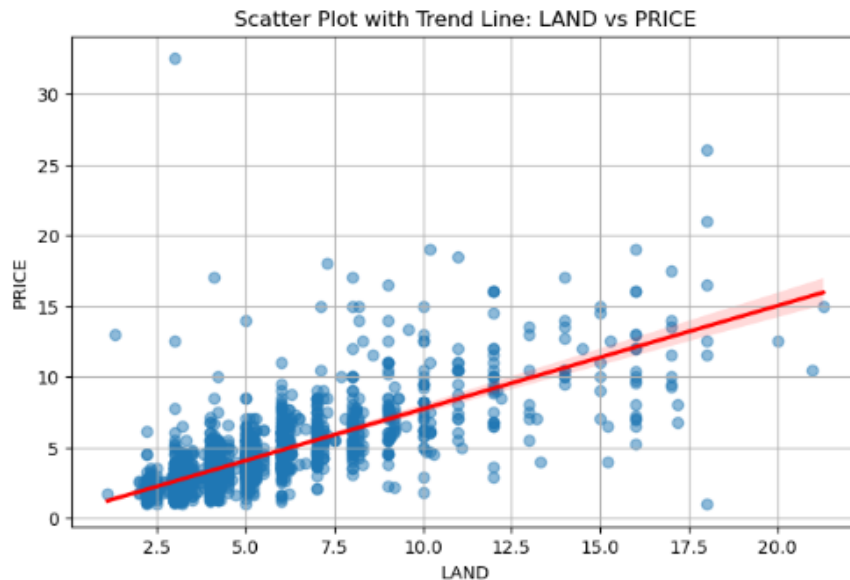


Figure 14: Scatter plot for LAND and PRICE

The scatter plot shows a positive correlation between the LAND column and PRICE column, with the trend line indicating if the LAND is large or then the PRICE is also automatically higher. The spread of points which we can see in the graph suggests that not only LAND, but also other factors may also influence PRICE, which can be BATHROOM, BEDROOM, FLOOR and etc.

4.2. Models Prediction

4.2.1. Initial Prediction by linear regression

```
def predict_price(facing, floor, bedroom, bathroom, road, city, age, land):  
    input_data = np.array([[facing, floor, bedroom, bathroom, road, city, age, land]])  
  
    input_scaled = scaler.transform(input_data)  
  
    predicted_price = linear.predict(input_scaled)  
  
    return predicted_price[0]  
  
predicted_price = predict_price(facing=8, floor=2, bedroom=4, bathroom=3, road=12, city=4, age=22, land=7.0)  
print(f"The predicted price using linear regressor model is: NRs {predicted_price} Cr")  
  
The predicted price using linear regressor model is: NRs 4.760733548992021 Cr
```

Figure 15: Initial Prediction by linear regression

In the above code the screenshot of the section where the `predict_price` function is used to predict the price of the house based on different factors like the direction house is facing, number of floors in house, number of bedrooms in house, the road width in front of the house, the age of the house and the land it is built on using linear regression model.

Firstly, we pass facing as 8, floor as 2, bedroom as 4, bathroom as 3, road as 12 feet, city as Kathmandu which is indicated as 4 and age of the house 22 and the land 7 Aana which gave us the output as Nrs 4.760 Cr

4.2.2. Final prediction by linear regression after improvement

```
def predict_price(floor, bedroom, bathroom, road, land, age, city):
    # Calculate LAND^2
    land_squared = land ** 2

    # Create input data with LAND^2
    input_data = np.array([[floor, bedroom, bathroom, road, land, age, city, land_squared]])

    # Scale the input data (if scaling is required)
    input_scaled = scaler.transform(input_data)

    # Make the prediction
    predicted_price = linear.predict(input_scaled)

    return predicted_price[0]

predicted_price = predict_price(floor=2, bedroom=4, bathroom=3, road=12, land=7, age=22, city=4)
print(f"The predicted price using linear regression model is: NRs {predicted_price} Cr")
```

✓ 00s

The predicted price using linear regression model is: NRs 4.825913043966898 Cr

Figure 16: Final prediction by linear regression after improvement

Figure 17: Final prediction in frontend by linear regression after improvement

In the final prediction part again after increasing the accuracy and decreasing the model error the prediction was done by passing, floor as 2, bedroom as 4, bathroom as 3, road as 12 feet, city as Kathmandu which is indicated as 4 and age of the house 22 and the land 7 Aana which gave us the output as Nrs 4.82 Cr

4.2.3. Initial Prediction by random forest regressor

```
Prediction using Random Forest Regressor Model

def predict_price_rf(floor, bedroom, bathroom, road, land, age, city):
    # Calculate LAND^2
    land_squared = land ** 2

    # Create input data with LAND^2
    input_data = np.array([[floor, bedroom, bathroom, road, land, age, city, land_squared]])

    # Make the prediction
    predicted_price_rf = best_forest.predict(input_data)

    return predicted_price_rf[0]

predicted_price_rf = predict_price_rf(floor=3, bedroom=5, bathroom=4, road=12, land=4, age=5, city=4)
print(f"The predicted price using random forest regressor model is: NRs {predicted_price_rf} Cr")

✓ 00s
The predicted price using random forest regressor model is: NRs 3.1255106551697485 Cr
```

Figure 18: Initial Prediction by random forest regressor model

Figure 19: Initial Prediction in frontend by random forest regressor model

In the above code the screenshot of the section where the `predict_price_rf` function is used to predict the price of the house based on different factors like the direction house is facing, number of floors in house, number of bedrooms in house, the road width in front of the house, the age of the house and the land it is built on using random forest regressor model.

Firstly, we pass facing as 0, floor as 2, bedroom as 4, bathroom as 3, road as 12 feet, city as Kathmandu which is indicated as 4 and age of the house 22 and land 7 Aana which gave us the output as Nrs 3.12 Cr

4.2.4. Final prediction by random forest regressor model after improvement

```
def predict_price_rf(floor, bedroom, bathroom, road, land, age, city):  
    # Calculate LAND^2  
    land_squared = land ** 2  
  
    # Create input data with LAND^2  
    input_data = np.array([[floor, bedroom, bathroom, road, land, age, city, land_squared]])  
  
    # Make the prediction  
    predicted_price_rf = best_forest.predict(input_data)  
  
    return predicted_price_rf[0]  
  
predicted_price_rf = predict_price_rf(floor=2, bedroom=4, bathroom=3, road=12, land=7, age=22, city=4)  
print(f"The predicted price using random forest regressor model is: NRs {predicted_price_rf} Cr")  
  
✓ 02s  
The predicted price using random forest regressor model is: NRs 4.786734095301937 Cr
```

Figure 20: Final prediction by random forest regressor model after improvement

House Price Prediction

Enter House Features

FLOOR
2 - +

BEDROOM
4 - +

BATHROOM
3 - +

ROAD (in feet)
12 - +

LAND (in aana)
7.00 - +

AGE
22 - +

Cities
kathmandu v

Select Model
Random Forest v

Predict

The predicted price using Random Forest is: NRs 4.90 Cr

Figure 21: Final prediction in front end by random forest regressor model after improvement

In the final prediction part again after increasing the accuracy and decreasing the model error the prediction was done by passing floor as 2, bedroom as 4, bathroom as 3, road as 12 feet, city as Kathmandu which is indicated as 4 and age of the house 22 and land 7 Aana which gave us the output as Nrs 4.78 Cr which is more accurate then initial approach as the r^2 is greater and mae and mse is lesser this time.

4.3. Comparison of the model prediction with actual data

4.3.1. Initial Comparison

Since we pass the same value in both the function of linear regression and random forest regression, we can see that the price predicted by the random forest model is little higher than the price predicted by the linear regression model.

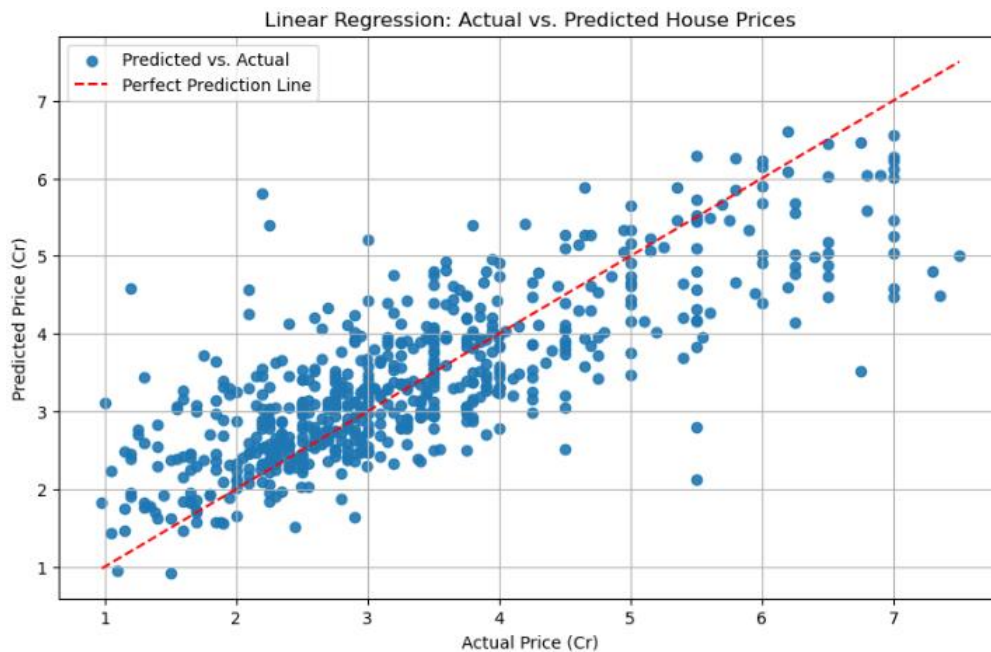


Figure 22: Initial scatter plot Comparison

In the figure above we can see a scatter and a best fit line, the line which is drawn is the best fit line and the plots done are the prediction done by the linear regression model. Since some of the values we can see that are scattered a little far from the best fit line and some are near it which is good sign that the prediction is doing well but due to some being far from the best fit line arises concerns

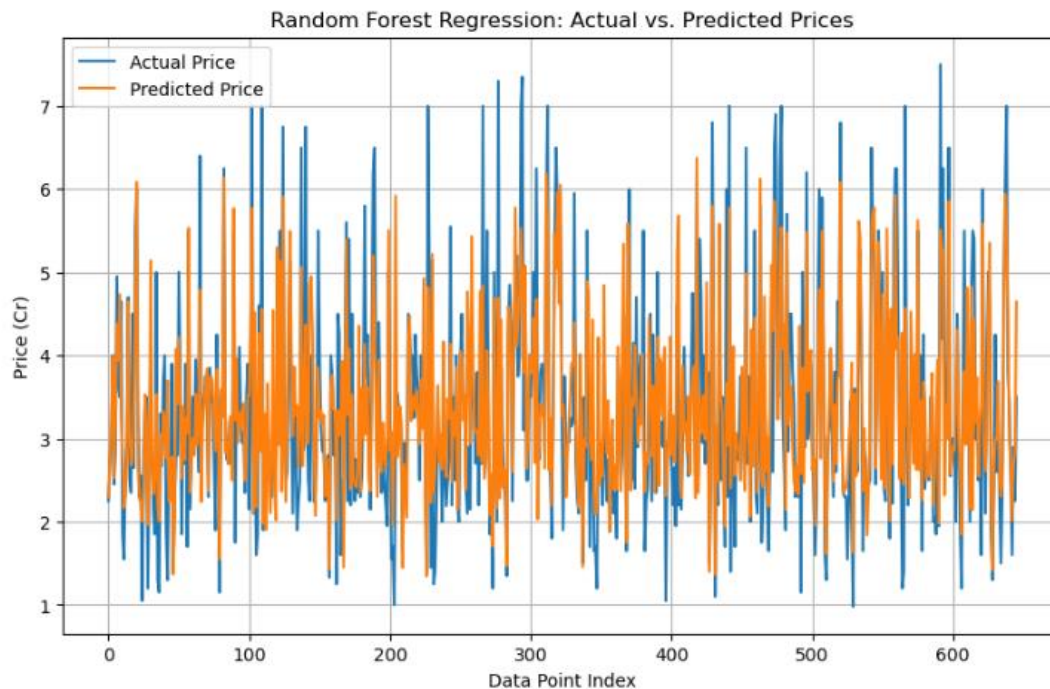


Figure 23: Initial Line graph Comparison

In the figure above we can see a line graph which is made by the actual and predicted price of the random forest regressor model. The blue line in the line graph is the actual price and the orange indicates the predicted price. The predicted price is a little higher than the actual price, from the above figure we can identify this.

```
#comparing top 5 actual value and value predicted by linear regression model
valid_data = pd.DataFrame({
    'Actual': y_test,
    'Predicted': y_pred
}).dropna()

print(valid_data.head(5))
```

| | Actual | Predicted |
|------|--------|-----------|
| 1900 | 3.30 | 3.125990 |
| 3187 | 2.75 | 3.250575 |
| 289 | 4.30 | 4.523553 |
| 537 | 2.00 | 1.627254 |
| 1199 | 5.00 | 5.004523 |

Figure 24: Initial predicted vs actual value by linear regression

From the above output we can see the comparison of 5 of the actual value and the predicted value done by the linear regression model

The prediction is not 100% accurate as there are some errors.

```
#comparing actual value and value predicted by random forest regressor model
valid_data = pd.DataFrame({
    'Actual': y_test,
    'Predicted': y_pred_rf
}).dropna()

print(valid_data.head(5))
```

| | Actual | Predicted |
|------|--------|-----------|
| 1900 | 3.30 | 3.112926 |
| 3187 | 2.75 | 3.259861 |
| 289 | 4.30 | 4.681416 |
| 537 | 2.00 | 1.933917 |
| 1199 | 5.00 | 4.827645 |

Figure 25: Initial predicted vs actual value by random forest

From the above output we can see the comparison of 5 of the actual value and the predicted value done by the random forest regressor model

The prediction is not 100% accurate as there are some errors.

4.3.2. Final Comparison after improvement

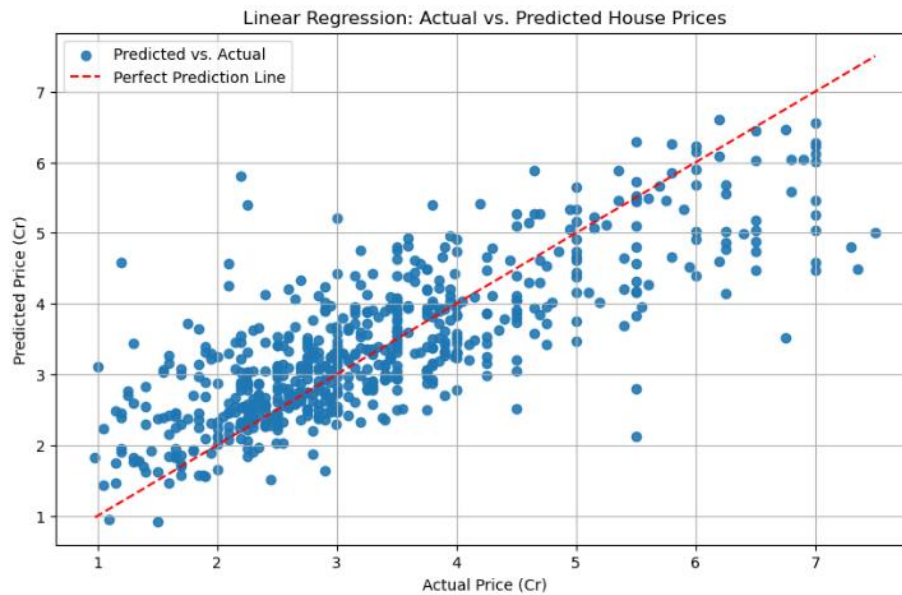


Figure 26: Final scatter plot Comparison for linear regression

It is a scatter plot which is again visualized after the model accuracy is increased and model errors are reduced compared to initial. In this scatter plot we can see that the predicted values are less scattered compared with the initial plot it is more compact, and the error is also less, and more are close to the best fit or perfect line which indicates the model is working better than the initial one

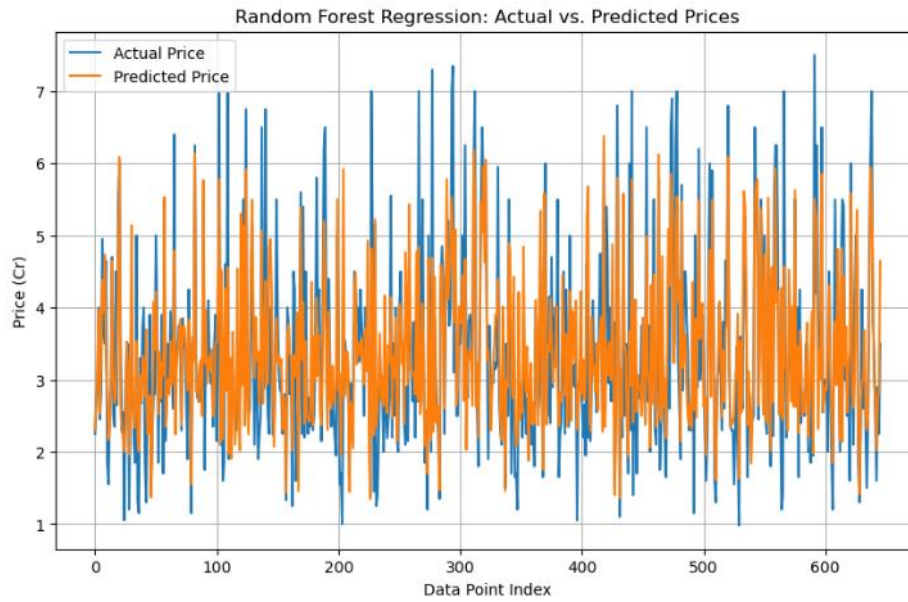


Figure 27: Final linear graph Comparison for random forest

It is a Line graph which is again visualized after the model accuracy is increased and model errors are reduced compared to the initial. In this graph compared to the initial one we can visualize the predicted price is close to the actual price not like the initial one which had a clear big difference, which indicates the model is working better than the initial one

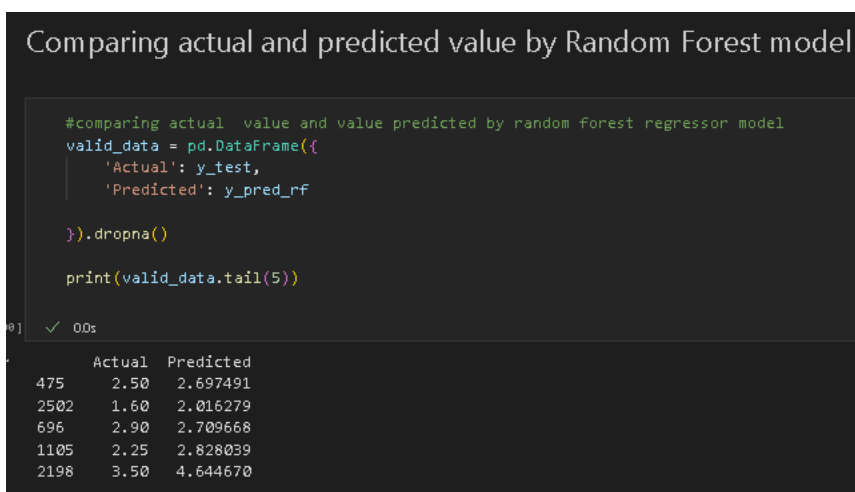


Figure 28: Initial predicted vs actual value by Random Forest

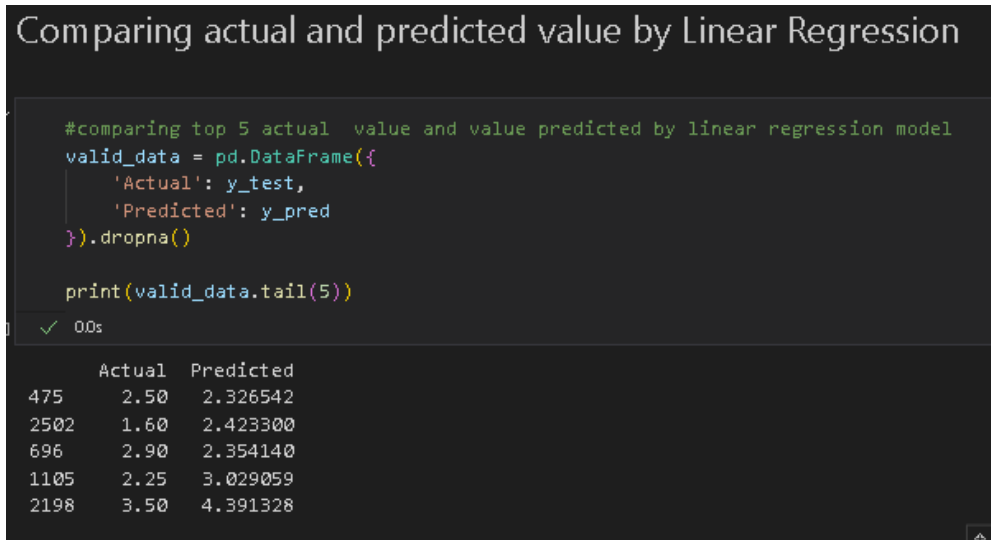


Figure 29: Initial predicted vs actual value by linear regression

The above is the side-by-side comparison of predicted and actual by random forest and linear regression model where the model is working fine not 100% perfect but working fine.

4.4. Evaluation of the model with performance metrics

4.4.1. Initial Evaluation of the model

```
#for linear regression
mae_lr = mean_absolute_error(y_test, y_pred)
mse_lr = mean_squared_error(y_test, y_pred)
rmse_lr = np.sqrt(mse_lr)
r2_lr = r2_score(y_test, y_pred)

# For Random Forest
mae_rf = mean_absolute_error(y_test, y_pred_rf)
mse_rf = mean_squared_error(y_test, y_pred_rf)
rmse_rf = np.sqrt(mse_rf)
r2_rf = r2_score(y_test, y_pred_rf)

def print_boxed_results(model_name, mae, mse, rmse, r2):
    print("="*50)
    print(f"{model_name} Performance Metrics".center(50))
    print("="*50)
    print(f"Mean Absolute Error (MAE): {mae:.2f}".rjust(50))
    print(f"Mean Squared Error (MSE): {mse:.2f}".rjust(50))
    print(f"Root Mean Squared Error (RMSE): {rmse:.2f}".rjust(50))
    print(f"R2 Score: {r2:.2f}".rjust(50))
    print("="*50)

#results for Linear Regression
print_boxed_results("Linear Regression", mae_lr, mse_lr, rmse_lr, r2_lr)

#results for Random Forest
print_boxed_results("Random Forest", mae_rf, mse_rf, rmse_rf, r2_rf)
```

Figure 30: Initial performance metrics Evaluation code of the models

In the above code we can see that we have asked for the mean absolute error, mean squared error, root mean squared error and r squared score of the model

The performance metrics of both the model is evaluated and output is displayed below

Linear regression model performance metrics

```
=====
      Linear Regression Performance Metrics
=====
      Mean Absolute Error (MAE): 0.57
      Mean Squared Error (MSE): 0.56
      Root Mean Squared Error (RMSE): 0.75
      R2 Score: 0.54
=====
```

Figure 31: Initial performance metrics result of the linear regression model

MAE (Mean absolute error) – the mean absolute error is the error that is obtained after the difference between the actual price and predicted price, the linear regression model got about 0.57 MAE it means the predicted price is differ from the actual value by around 0.57, the prediction is close but still improvement is required in further work

MSE (Mean squared error) - A moderate MSE value that is 0.56 which suggests that the Linear Regression model does not handle larger errors well, making it moderately effective for this problem.

Root mean squared error (RMSE) : The RMSE of the linear regression model prediction is about 0.75 it suggests that the model is about 0.75 off it is reasonably accurate but needs some more improvement to make the prediction more accurate.

R² Score: Since we only got about 54% R² which is not good not bad a reasonable amount. As linear regression is base model it requires a lot more hyper tuning which is not done so some improvement is required

Random Forest performance metrics evaluation

```
=====
      Random Forest Performance Metrics
=====
      Mean Absolute Error (MAE): 0.53
      Mean Squared Error (MSE): 0.48
      Root Mean Squared Error (RMSE): 0.70
      R2 Score: 0.61
=====
```

Figure 32: Initial performance metrics result of the random forest model

MAE (Mean absolute error) – the mean absolute error is the error that is obtained after the difference between the actual price and predicted price, the random forest regressor model got about 0.53 MAE, it means the predicted price is differ from the actual value by around 0.53, the prediction is close but still improvement is required in further work

MSE (Mean squared error) - A moderate MSE value that is 0.48 which suggests that the random forest regressor model does not handle larger errors well, making it moderately effective for this problem.

Root mean squared error (RMSE): The RMSE of the random forest regressor model prediction is about 0.70, it suggests that the model is about 0.70 off it is reasonably accurate but needs some more improvement to make the prediction more accurate in further work.

R² Score: The Random Forest model achieved an R² score of 61%, which is a moderately good result. It explains 61% of the variance in Nepali house prices, but still some improvement is required to make the score go higher and prediction more accurate.

4.4.2. Final Evaluation of the model after improvement

```
#for linear regression
mae_lr = mean_absolute_error(y_test, y_pred)
mse_lr = mean_squared_error(y_test, y_pred)
rmse_lr = np.sqrt(mse_lr)
r2_lr = r2_score(y_test, y_pred)

# For Random Forest
mae_rf = mean_absolute_error(y_test, y_pred_rf)
mse_rf = mean_squared_error(y_test, y_pred_rf)
rmse_rf = np.sqrt(mse_rf)
r2_rf = r2_score(y_test, y_pred_rf)

def print_boxed_results(model_name, mae, mse, rmse, r2):
    print("="*50)
    print(f"{model_name} Performance Metrics".center(50))
    print("="*50)
    print(f"Mean Absolute Error (MAE): {mae:.2f}".rjust(50))
    print(f"Mean Squared Error (MSE): {mse:.2f}".rjust(50))
    print(f"Root Mean Squared Error (RMSE): {rmse:.2f}".rjust(50))
    print(f"R2 Score: {r2:.2f}".rjust(50))
    print("="*50)

#results for Linear Regression
print_boxed_results("Linear Regression", mae_lr, mse_lr, rmse_lr, r2_lr)

#results for Random Forest
print_boxed_results("Random Forest", mae_rf, mse_rf, rmse_rf, r2_rf)
```

Figure 33: final performance metrics Evaluation code of the models

The performance metrics of both the model is evaluated to compare the output with the initial result

Linear Regression performance metrics evaluation

```
Mean Absolute Error (MAE): 0.603  
Mean Squared Error (MSE): 0.673  
Root Mean Squared Error (RMSE): 0.820  
R2 Score: 0.62  
Score: 0.62
```

Figure 34: Final Linear Regression performance metrics evaluation

MAE (Mean absolute error) – In the evaluation of the model I got MAE about 0.60 which is slightly high compared to the initial result as it was 0.57.

MSE (Mean squared error) A moderate MSE value that is 0.67 which suggests that the Linear Regression model does not handle larger errors well, making it moderately effective for this problem and is also slightly higher than initial as it was 0.56

Root mean squared error (RMSE): The RMSE of the linear regression model prediction is about 0.82, it suggests that the model is about 0.82 off it is reasonably accurate but needs some more improvement to make the prediction more accurate.

R² Score: The R² Score of the linear regression model is 0.62 which means it explains 62% variations in the house price, I was able to increase the R² Score by 8% which is a good jump but that lead to slight increment in MAE and MSE

Random Forest regressor performance metrics evaluation

```
Mean Absolute Error (MAE): 0.598  
Mean Squared Error (MSE): 0.660  
Root Mean Squared Error (RMSE): 0.864  
R2 Score: 0.63  
Score: 0.63
```

Figure 35: Final Random Forest performance metrics evaluation

MAE (Mean absolute error) – In this final evaluation the random forest regressor model got about 0.59 MAE, it means the predicted price differs from the actual value by around 0.59. It is slight high compared to the initial result as it was 0.53

MSE (Mean squared error): In this final evaluation the random forest regressor model is obtained MSE about 0.66 which is 0.18 times bigger than the initial one

Root mean squared error (RMSE): The RMSE of the random forest regressor model prediction is about 0.86 which is still higher than the initial result

R² Score: In this evaluation the Random Forest model the achieved R² is 0.63 which means it explains 63% of the variance of the data. Compared to the initial result it increased about 2 % more.

In Summary both the models showed improvements as the score of the model was increased in both the models and errors were also countered in the final evaluation.

Comparing both the models Random Forest regressor stands out of the two as it has low MAE and MSE and higher R² compared to linear regression model.

4.5. Techniques applied for model improvement

4.5.1. Increasing test size and random state adjustment

Before: Linear regression model

```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=42)
x_train_scale = scaler.fit_transform(x_train)
x_test_scale = scaler.transform(x_test)
```

✓ 00s

Figure 36: test size and random state Before

```
Mean Absolute Error (MAE): 0.66
Mean Squared Error (MSE): 0.81
Root Mean Squared Error (RMSE): 0.90
R² Score: 0.58
Score: 0.58
```

Figure 37: Output before

After:

```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.3, random_state=62)
x_train_scale = scaler.fit_transform(x_train)
x_test_scale = scaler.transform(x_test)
```

✓ 00s

Figure 38: test size and random state After

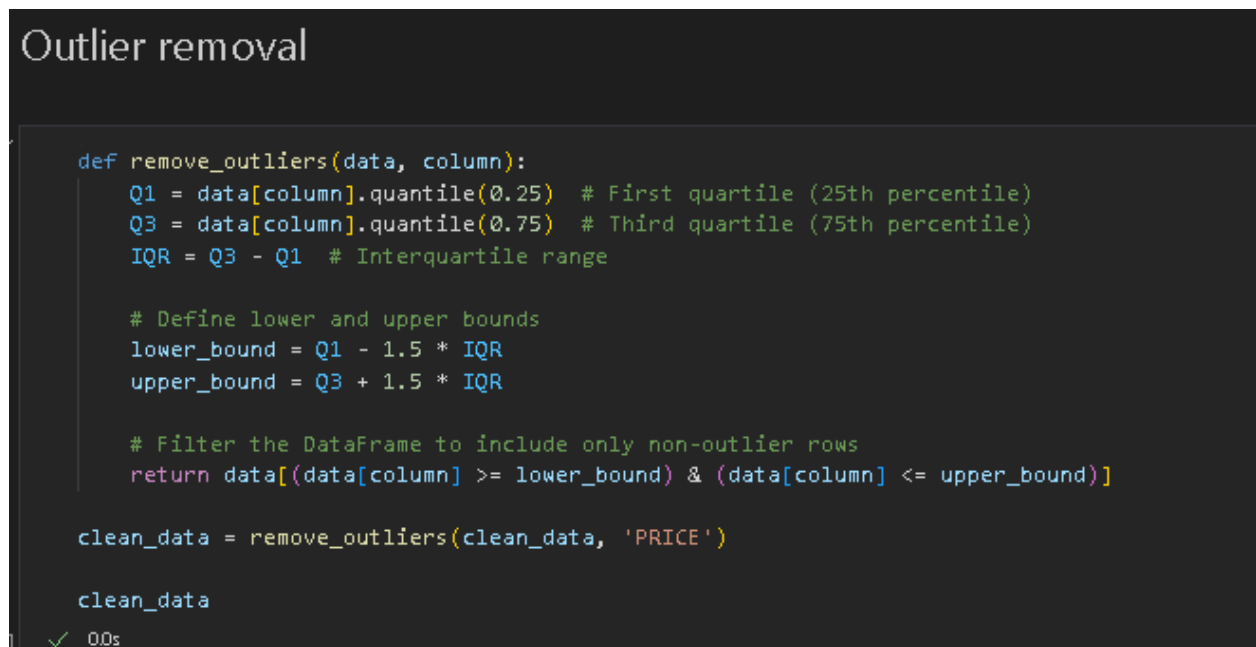
```
Mean Absolute Error (MAE): 0.66
Mean Squared Error (MSE): 0.81
Root Mean Squared Error (RMSE): 0.90
R² Score: 0.59
Score: 0.59
```

Figure 39: Output After

In the above code the data set after cleaning was only about 2000 rows which was less for testing so split the data in 70-30 format to give more data to the testing and the random state was increased to 62 so the data would be more shuffled to increase accuracy and decrease the errors.

By performing this we can see that the MAE and MSE didn't increase or decrease but the r^2 SCORE is increased slightly

4.5.2. Using IQR method for outlier removal



```
Outlier removal

def remove_outliers(data, column):
    Q1 = data[column].quantile(0.25) # First quartile (25th percentile)
    Q3 = data[column].quantile(0.75) # Third quartile (75th percentile)
    IQR = Q3 - Q1 # Interquartile range

    # Define lower and upper bounds
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR

    # Filter the DataFrame to include only non-outlier rows
    return data[(data[column] >= lower_bound) & (data[column] <= upper_bound)]

clean_data = remove_outliers(clean_data, 'PRICE')

clean_data
```

Figure 40: Using IQR method for outlier removal

While visualizing the box plot there were some outliers in PRICE column that needed to be removed which was not done earlier, A IQR method is being used to remove the outlier as it.

Before: Linear regression model

```
mae_lr = mean_absolute_error(y_test, y_pred)
mse_lr = mean_squared_error(y_test, y_pred)
rmse_lr = np.sqrt(mse_lr)
r2_lr = r2_score(y_test, y_pred)
score = linear.score(x_test_scale, y_test)

print(f"Mean Absolute Error (MAE): {mae_lr:.3f}")
print(f"Mean Squared Error (MSE): {mse_lr:.3f}")
print(f"Root Mean Squared Error (RMSE): {rmse_lr:.3f}")
print(f"R2 Score: {r2_lr:.2f}")
print(f"Score: {score:.2f}")
```

✓ 00s

```
Mean Absolute Error (MAE): 0.883
Mean Squared Error (MSE): 2.012
Root Mean Squared Error (RMSE): 1.419
R2 Score: 0.63
Score: 0.63
```

Figure 41: Performance metrics evaluation before in linear regression

After:

```
mae_lr = mean_absolute_error(y_test, y_pred)
mse_lr = mean_squared_error(y_test, y_pred)
rmse_lr = np.sqrt(mse_lr)
r2_lr = r2_score(y_test, y_pred)
score = linear.score(x_test_scale, y_test)

print(f"Mean Absolute Error (MAE): {mae_lr:.3f}")
print(f"Mean Squared Error (MSE): {mse_lr:.3f}")
print(f"Root Mean Squared Error (RMSE): {rmse_lr:.3f}")
print(f"R2 Score: {r2_lr:.2f}")
print(f"Score: {score:.2f}")
```

✓ 00s

```
Mean Absolute Error (MAE): 0.603
Mean Squared Error (MSE): 0.673
Root Mean Squared Error (RMSE): 0.820
R2 Score: 0.62
Score: 0.62
```

Figure 42: Performance metrics evaluation after in linear regression

Before removing the outlier, the error in linear regression model was quite high 0.883 MAE and 2.01 MSE which is quite high even though the r^2 is 0.63 which is 1% higher the error is too high in initial and in final it is removed.

Before: Random forest model

```
mae_r = mean_absolute_error(y_test, y_pred_rf)
mse_r = mean_squared_error(y_test, y_pred_rf)
rmse_r = np.sqrt(mse)
r2_r = r2_score(y_test, y_pred_rf)
score_r = best_forest.score(x_test, y_test)

print(f"Mean Absolute Error (MAE): {mae_r:.3f}")
print(f"Mean Squared Error (MSE): {mse_r:.3f}")
print(f"Root Mean Squared Error (RMSE): {rmse_r:.3f}")
print(f"R2 Score: {r2_r:.2f}")
print(f"Score: {score_r:.2f}")
```

✓ 00s

```
Mean Absolute Error (MAE): 0.897
Mean Squared Error (MSE): 2.085
Root Mean Squared Error (RMSE): 1.640
R2 Score: 0.61
Score: 0.61
```


Figure 43: Performance metrics evaluation before in random forest

Before removing the outlier, the error in linear regression model was quite high 0.897 MAE and 2.085 MSE which is quite high which led to the prediction being very off than the real value, even though the r^2 was slight better the massive error led the prediction worse.

After

```
mae_r = mean_absolute_error(y_test, y_pred_rf)
mse_r = mean_squared_error(y_test, y_pred_rf)
rmse_r = np.sqrt(mse)
r2_r = r2_score(y_test, y_pred_rf)
score_r = best_forest.score(x_test, y_test)

print(f"Mean Absolute Error (MAE): {mae_r:.3f}")
print(f"Mean Squared Error (MSE): {mse_r:.3f}")
print(f"Root Mean Squared Error (RMSE): {rmse_r:.3f}")
print(f"R2 Score: {r2_r:.2f}")
print(f"Score: {score_r:.2f}")
```



```
Mean Absolute Error (MAE): 0.598
Mean Squared Error (MSE): 0.660
Root Mean Squared Error (RMSE): 0.854
R2 Score: 0.63
Score: 0.63
```

Figure 44: Performance metrics evaluation after in random forest

In the above code after using the IQR method to remove the outlier the MAE and MSE of the random forest model is improved, the MAE and MSE is lower than before and score of the model is 0.63 which is higher than the initial

4.5.3. Hyper parameter tuning random forest model

```

from sklearn.model_selection import GridSearchCV

# Initialize the RandomForestRegressor
forest = RandomForestRegressor(random_state=42)

# Define the parameter grid
param_grid = {
    'n_estimators': [100, 200, 300, 400, 500, 600], # Number of trees
    'max_depth': [None, 10, 20, 30, 40], # Maximum depth of the tree
    'min_samples_split': [2, 5, 10], # Minimum samples to split
    'min_samples_leaf': [1, 2, 4], # Minimum samples at a leaf node
    'max_features': ['auto', 'sqrt', 'log2'] # Features to consider for splitting
}

# Use GridSearchCV to search over all combinations in the grid
grid_search = GridSearchCV(
    estimator=forest,
    param_grid=param_grid,
    cv=3, # 3-fold cross-validation
    verbose=2, # Print progress
    n_jobs=-1, # Use all available CPU cores
    scoring='neg_mean_squared_error' # Scoring metric
)

# Fit the model
grid_search.fit(x_train, y_train)

# Get the best hyperparameters
best_params = grid_search.best_params_
print("Best Hyperparameters: ", best_params)

# Use the best model
best_forest = grid_search.best_estimator_

# Predict on the test set
y_pred_rf = best_forest.predict(x_test)
✓ 7m 11.5s

Fitting 3 folds for each of 810 candidates, totalling 2430 fits
Best Hyperparameters: {'max_depth': 10, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 10, 'n_estimators': 600}

```

Figure 45: Code to hyper parameter tune random forest model

To make the model provide best output with the parameter of the model is tuned. Specifically using the grid search model, the parameter of the random forest model is tuned. It provides the best parameters and fits the model with it ensuring the best result than before with less errors.

Before hyper tuning random forest model

```
random = RandomForestRegressor( random_state=42)
random.fit(x_train, y_train)
y_pred_r = random.predict(x_test)
mae = mean_absolute_error(y_test, y_pred_r)
mse = mean_squared_error(y_test, y_pred_r)
rmse = np.sqrt(mse)
r2 = r2_score(y_test, y_pred_r)
score = random.score(x_test, y_test)

print(f"Mean Absolute Error (MAE): {mae:.2f}")
print(f"Mean Squared Error (MSE): {mse:.2f}")
print(f"Root Mean Squared Error (RMSE): {rmse:.2f}")
print(f"R2 Score: {r2:.2f}")
print(f"Score: {score:.2f}")
```

✓ 0.5s

Mean Absolute Error (MAE): 0.63
Mean Squared Error (MSE): 0.74
Root Mean Squared Error (RMSE): 0.86
R² Score: 0.58
Score: 0.58

Figure 46: Result before hyper parameter tuning

Before hyper tuning the random forest model the MAE about 0.63 and MSE about 0.74 and the accuracy was just 0.58 which is low and required some improvement

After

```
mae = mean_absolute_error(y_test, y_pred_rf)
mse = mean_squared_error(y_test, y_pred_rf)
rmse = np.sqrt(mse)
r2 = r2_score(y_test, y_pred_rf)
score = best_forest.score(x_test, y_test)

print(f"Mean Absolute Error (MAE): {mae:.2f}")
print(f"Mean Squared Error (MSE): {mse:.2f}")
print(f"Root Mean Squared Error (RMSE): {rmse:.2f}")
print(f"R2 Score: {r2:.2f}")
print(f"Score: {score:.2f}")
```

✓ 0.1s

```
Mean Absolute Error (MAE): 0.61
Mean Squared Error (MSE): 0.66
Root Mean Squared Error (RMSE): 0.81
R2 Score: 0.63
Score: 0.63
```

Figure 47: Result after hyper parameter tuning

After hyper tuning the random forest model the model performed better as the MAE and MSE decreased, MAE decreased from 0.63 to 0.61 and MSE from 0.74 to 0.66 and the R^2 score of the model increased from 0.58 to 0.63 which is a huge jump. This made the prediction from the model more accurate and less error.

4.5.4. Polynomial Feature engineering

```
clean_data['LAND_squared'] = clean_data['LAND'] ** 2
clean_data
```

✓ 0.0s

| | PRICE | FLOOR | BEDROOM | BATHROOM | ROAD | LAND | AGE | Cities | LAND_squared |
|------|-------|-------|---------|----------|------|------|-----|--------|--------------|
| 0 | 2.90 | 3 | 5 | 4 | 12 | 4.0 | 5 | 4 | 16.00 |
| 1 | 4.75 | 4 | 5 | 6 | 10 | 3.0 | 5 | 4 | 9.00 |
| 2 | 1.99 | 2 | 4 | 4 | 10 | 2.3 | 21 | 4 | 5.29 |
| 3 | 4.00 | 2 | 4 | 3 | 12 | 7.0 | 22 | 4 | 49.00 |
| 4 | 1.20 | 2 | 4 | 4 | 20 | 6.0 | 10 | 3 | 36.00 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 3413 | 2.60 | 2 | 4 | 4 | 16 | 4.5 | 5 | 3 | 20.25 |
| 3414 | 3.00 | 3 | 3 | 3 | 16 | 4.5 | 5 | 3 | 20.25 |
| 3415 | 1.60 | 2 | 4 | 2 | 16 | 4.7 | 9 | 3 | 22.09 |
| 3416 | 3.50 | 2 | 5 | 3 | 16 | 6.3 | 4 | 3 | 39.69 |
| 3417 | 4.80 | 2 | 5 | 4 | 16 | 8.0 | 6 | 3 | 64.00 |

2153 rows × 9 columns

Figure 48: Code for polynomial feature engineering

The polynomial feature engineering was applied with the LAND column. Since the LAND feature already showed some good positive correlation with the target variable that is PRICE, squaring it allowed the model to capture potential quadratic trends in the data that a linear relationship could not.

Before: Linear regression model

```
Mean Absolute Error (MAE): 0.66  
Mean Squared Error (MSE): 0.81  
Root Mean Squared Error (RMSE): 0.90  
R2 Score: 0.59  
Score: 0.59
```

Figure 49: Result before polynomial feature engineering for linear regression

After

```
mae_lr = mean_absolute_error(y_test, y_pred)  
mse_lr = mean_squared_error(y_test, y_pred)  
rmse_lr = np.sqrt(mse_lr)  
r2_lr = r2_score(y_test, y_pred)  
score = linear.score(x_test_scale, y_test)  
  
print(f"Mean Absolute Error (MAE): {mae_lr:.3f}")  
print(f"Mean Squared Error (MSE): {mse_lr:.3f}")  
print(f"Root Mean Squared Error (RMSE): {rmse_lr:.3f}")  
print(f"R2 Score: {r2_lr:.2f}")  
print(f"Score: {score:.2f}")  
✓ 00s  
  
Mean Absolute Error (MAE): 0.603  
Mean Squared Error (MSE): 0.673  
Root Mean Squared Error (RMSE): 0.820  
R2 Score: 0.62  
Score: 0.62
```

Figure 50: Result after polynomial feature engineering for linear regression

After performing the polynomial feature engineering, it helped in increasing the model r^2 score which means it explains more variance due to the additional feature which has positive correlation with the target variable, and it also decreased the MAE and MSE slight which helps in providing more accurate prediction.

4.6. Comparison between linear regression and random forest regressor model

4.6.1. Initial Comparison

```

=====
Linear Regression Performance Metrics
=====
Mean Absolute Error (MAE): 0.57
Mean Squared Error (MSE): 0.56
Root Mean Squared Error (RMSE): 0.75
R2 Score: 0.54
=====
Random Forest Performance Metrics
=====
Mean Absolute Error (MAE): 0.53
Mean Squared Error (MSE): 0.48
Root Mean Squared Error (RMSE): 0.70
R2 Score: 0.61
=====

```

Figure 51: Initial comparison of performance metrics of two models

Comparing the performance metrics and results from the two models we can determine that for the MAE error in linear regression model (0.57) is slightly higher than random forest regressor (0.53) this indicates comparatively the linear regression as slightly more prediction error than of random forest regressor.

For the MSE error the random forest again performs better (0.48) and linear regression (0.56) this indicates random forest makes smaller error than that of linear regression

For the RMSE the same output as random forest as lesser RMSE compared to linear regression which suggests that the random forest is the best approach among two as the random forest has less error.

Now when looking at R^2 to see how well the two models handle the variance in the dataset, we can see that random forest 61% is better than linear regression 54 % as both have room for improvement random forest stands out among two.

4.6.2. Final comparison after improvement

After using different techniques and approaches I was able to improve the score and errors of both the linear regression and random forest model which is described below.

```

=====
Random Forest Performance Metrics
=====
Mean Absolute Error (MAE): 0.598
Mean Squared Error (MSE): 0.660
Root Mean Squared Error (RMSE): 0.812
R2 Score: 0.63
=====

```

Figure 52: Final comparison of random forest model after improvement

```

=====
Linear Regression Performance Metrics
=====
Mean Absolute Error (MAE): 0.603
Mean Squared Error (MSE): 0.673
Root Mean Squared Error (RMSE): 0.820
R2 Score: 0.62
=====

```

Figure 53: Final comparison of Linear regression model after improvement

Mean Absolute Error (MAE):

- Random Forest: 0.598
- Linear Regression: 0.603
- Comparison: Random Forest has a slightly lower MAE than linear regression it is more accurate in terms of absolute error.

Mean Squared Error (MSE):

- Random Forest: 0.660
- Linear Regression: 0.673
- Comparison: Random Forest performs better with a lower MSE, which means it handles larger errors slightly better than linear regression in the prediction

Root Mean Squared Error (RMSE):

- Random Forest: 0.812
- Linear Regression: 0.820
- Comparison: Random Forest has a slightly lower RMSE, showing better overall accuracy in the prediction

R² Score:

- Random Forest: 0.63
- Linear Regression: 0.62
- Comparison: Random Forest explains a bit more of the variance in the data than Linear Regression.

The random forest regressor model outperforms linear regression models. The differences in MAE, MSE, RMSE, and R² score suggest that the Random Forest model is slightly more accurate and better at explaining the variance in the data than the Linear regression model and it predicts more accurately, Since the difference is not that big Linear regression is also considered a good approach

4.7. Result visualization

4.7.1. Model evaluation comparison using bar graph

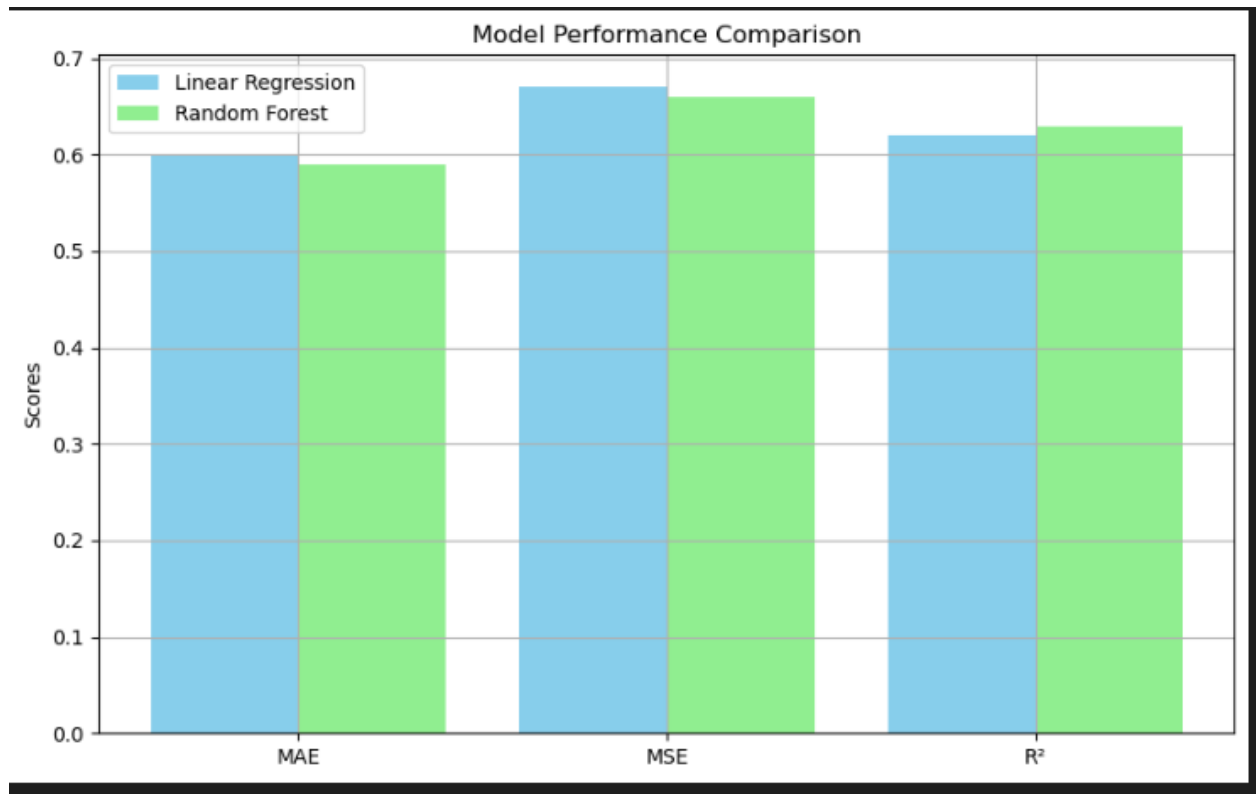


Figure 54: Model evaluation comparison using bar graph

In the bar graph above we can see that the MAE AND MSE both the linear regression is slight better than that of random forest model which indicates the random forest model handles loss slight better than linear regression and the r^2 score of random forest model is slight better than linear regression

4.7.2. Analyze pattern in error using Residual plot

Random forest

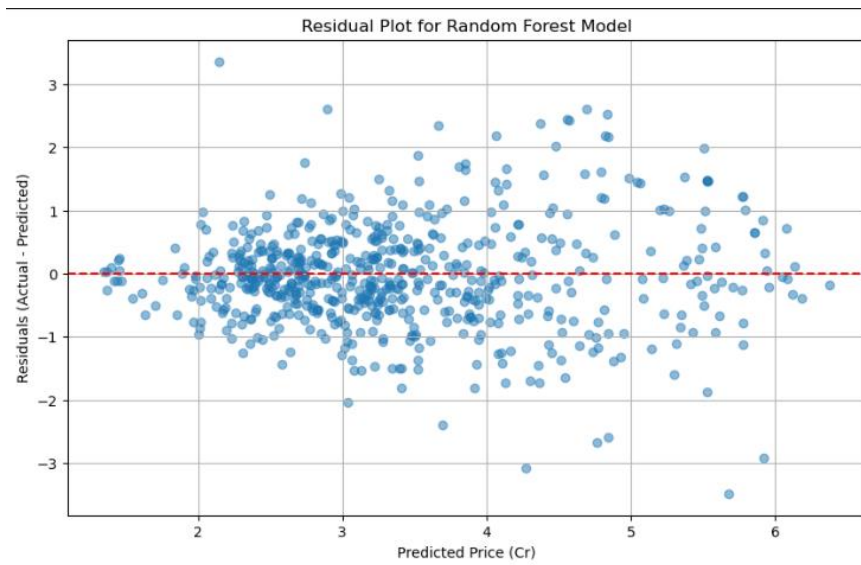


Figure 55: Analyse pattern in error of random forest

Linear regression

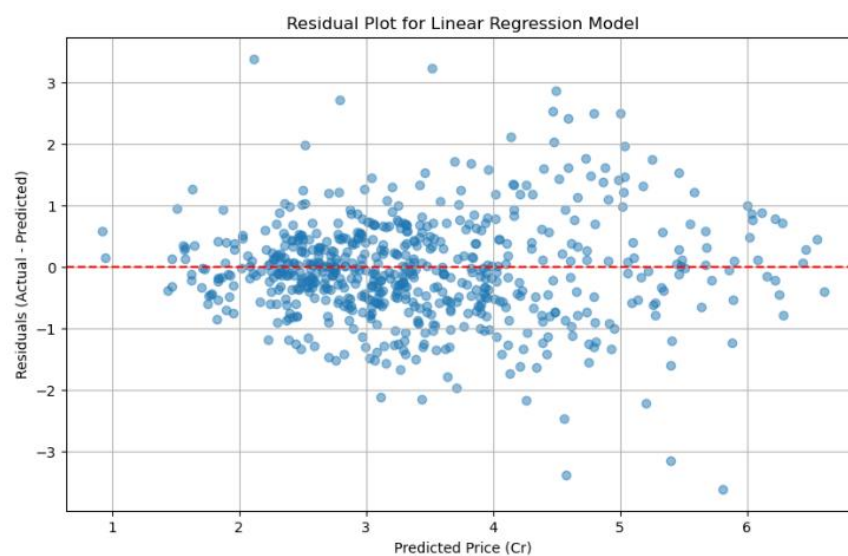


Figure 56: Analyse pattern in error of Linear regression

The residual plots for both the Linear Regression and Random Forest models were created to study the pattern of the errors made by the models. These plots help us understand whether the errors of both the models are randomly distributed all over the place or they have specific pattern which helps to indicate nonlinearity and bias in the prediction

In summary from the residual plot we can identify that the random forest shows more random error distribution which means that it captures the data patterns better than that of linear regression model does.

4.7.3. Feature importance using bar graph

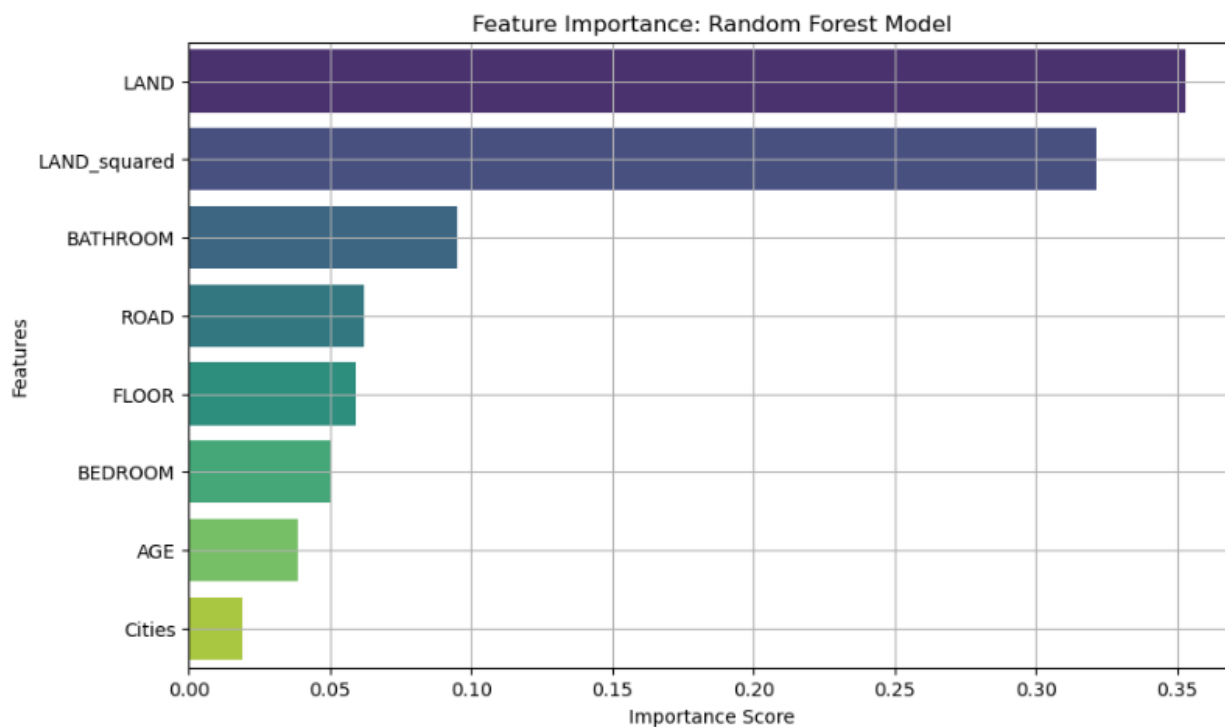


Figure 57: Feature importance using bar graph

The above bar graph represents all the independent variables bar to represent which feature variable plays most importance in the prediction and from the above graph “LAND” stands out as the column with the most importance than other as it is most positively highly correlated feature with the target variable.

5. Conclusion

In the above house price prediction system. I have implemented two predictive models, linear regression and random forest regression. The price is estimated based on different features like the direction the house is facing, number of floors in the house, number of bedrooms in the house, the road width in front of the house, the age of the house and the land it is built on. After getting the prediction the model performance is evaluated using different performance metrics like Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R^2 Score to calculate the error in the prediction. The prediction by both the models is also visualized in line graph and scatter plot to see how well the model works.

5.1. Model evaluation and effectiveness

After comparing linear regression and random forest models upon prediction and error we can determine that the random forest model worked better in both aspects than linear regression model. The random forest model achieved lower Mean Absolute Error (MAE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), than that of linear regression model which mainly indicates that the linear regression model prediction has comparatively more error than that of random forest model.

The linear regression model showed an R^2 of about 62% and the random forest model showed around 63% which also determines that the random forest model slightly handles the variance better in the data than by linear regression.

However, we can see that both the models showed error in the prediction although random forest had slightly less error than of linear regression but still error is there in the prediction which required further work to improve it.

5.2. Real world applicability

The above project helps just to demonstrate the power and use of machine learning models and algorithms to solve real world problems. As in the real estate market for both buyers and sellers, determining the house is the major concern. Bearing loss is the major concern for both seller and buyer. The project's ability to predict prices with this factor is the best approach which helps all the buyers and sellers. With the help of this type of system both can rely on and can make informed decisions in the market. As prices are affected by many factors in the system, above all the factors are the ones which play the major role in determining the price.

Although it might not be 100% trusted by all the people in the market it can help both buyer and seller to make informed decisions and stay on track in the market and grow.

The prediction is also based on all the other old data from the market which shows the source of the prediction and data which helps people trust the system more and make better-informed decisions.

5.3. Suggestions for future improvements

Since the prediction done by the two models is not up to the mark even though the random forest model is more accurate and handles more variance than linear regression model but still there is quite errors in the prediction which leads some difference in the actual data and predicted data.

To make the model perform better with high accuracy prediction the following steps are required:

1. Exploring more advanced models and deep learning models to increase accuracy and decrease error.
2. Combining ensemble methods, for example bagging or boosting
3. Integrating similar dataset to increase training and testing of data

Since for now the model is only deployed as a Streamlit app for the frontend in near future the next step will be to deploy the model on web app or in mobile app as allowing the users to input the features and performing the prediction according to their input. This way it will help the model to reach all the potential end users.

6. References

Dickson Realty Team, 2024. *dicksonrealty*. [Online]

Available at: <https://blog.dicksonrealty.com/2024/05/top-10-buyer-challenges-in-real-estate/>

[Accessed 22 12 2024].

dolinskigroup, 2024. *dolinskigroup*. [Online]

Available at: <https://dolinskigroup.com/buy-home/challenges-home-buyers-face-sellers-market>

[Accessed 22 12 2024].

Lentz, S., 2024. *nowbam*. [Online]

Available at: <https://nowbam.com/what-buyers-sellers-really-think-about-the-2025-housing-market/>

[Accessed 22 12 2024].

statista, 2024. *statista*. [Online]

Available at: <https://www.statista.com/outlook/fmo/real-estate/nepal>

[Accessed 22 2 2024].

thepropertist, 2023. *thepropertist*. [Online]

Available at: <https://www.thepropertist.com/blog/5-common-problems-faced-by-home-sellers--304>

[Accessed 22 12 2024].

Tony Mark,Russell Grether, 2024. *malibuluxuryrealty*. [Online]

Available at: <https://malibuluxuryrealty.com/seven-problems-that-can-arise-from-overpricing-a->

[listing/#:~:text=If%20your%20home%20is%20immediately,property%20right%20the%20first%20time.](https://malibuluxuryrealty.com/seven-problems-that-can-arise-from-overpricing-a-listing/#:~:text=If%20your%20home%20is%20immediately,property%20right%20the%20first%20time.)

[Accessed 22 12 2024].

7. Bibliography

Dickson Realty Team, 2024. *dicksonrealty*. [Online]

Available at: <https://blog.dicksonrealty.com/2024/05/top-10-buyer-challenges-in-real-estate/>

[Accessed 22 12 2024].

dolinskigroup, 2024. *dolinskigroup*. [Online]

Available at: <https://dolinskigroup.com/buy-home/challenges-home-buyers-face-sellers-market>

[Accessed 22 12 2024].

Lentz, S., 2024. *nowbam*. [Online]

Available at: <https://nowbam.com/what-buyers-sellers-really-think-about-the-2025-housing-market/>

[Accessed 22 12 2024].

statista, 2024. *statista*. [Online]

Available at: <https://www.statista.com/outlook/fmo/real-estate/nepal>

[Accessed 22 2 2024].

thepropertist, 2023. *thepropertist*. [Online]

Available at: <https://www.thepropertist.com/blog/5-common-problems-faced-by-home-sellers--304>

[Accessed 22 12 2024].

Tony Mark,Russell Grether, 2024. *malibuluxuryrealty*. [Online]

Available at: <https://malibuluxuryrealty.com/seven-problems-that-can-arise-from-overpricing-a->

[listing/#:~:text=If%20your%20home%20is%20immediately,property%20right%20the%20first%20time.](https://malibuluxuryrealty.com/seven-problems-that-can-arise-from-overpricing-a-listing/#:~:text=If%20your%20home%20is%20immediately,property%20right%20the%20first%20time.)

[Accessed 22 12 2024].