CAN THO UNIVERSITY

School of Education

Report Computational Mathematics

Lasso Regression for House Price Prediction

Supervisor:
PhD. Tran Thu Le

Student:

1. Suphansa Pankliang E2400026
2. Phattharawan Detchiar E2400029
3. Thitisak Mahawijit E2400025

CAN THO UNIVERSITY

School of Education

Report Computational Mathematics

Lasso Regression for House Price Prediction

Supervisor:
PhD. Tran Thu Le

Student:

1.	Suphansa Pankliang	E2400026
2.	Phattharawan Detchiar	E2400029
3.	Thitisak Mahawijit	E2400025

Acknowledgments

First and foremost, we would like to express our deepest and most sincere gratitude to **Doctor Tran Thu Le**, lecturer at Can Tho University, for his expert supervision of this report titled *"Lasso Regression for House Price Prediction."* His profound knowledge, sense of responsibility, and dedication to teaching and research have provided us with invaluable guidance, timely feedback, and the motivation needed to overcome every challenge in our study.

We also wish to extend our sincere thanks to all **lecturers of the Mathematics Department, School of Education, Can Tho University**, as well as to the **faculty members of Walailak University (Thailand)**. Their exceptional teaching, academic inspiration, and ongoing support have played a key role in shaping our understanding of statistical modeling and machine learning techniques, particularly in the application of *Lasso regression*, ultimately contributing to the success of our studies and the completion of this report.

We are especially grateful to **Mr. Huynh Nhut Tan**, a student of *Cohort 49, Mathematics Teacher Education, Can Tho University*, and **Mr. Tran Hieu Nhan**, a student of *Cohort 48, Mathematics Teacher Education, Can Tho University*, for their generous and enthusiastic support during our research process. Their assistance in sourcing references, clarifying specialized topics, and sharing practical insights greatly enriched the quality and completeness of this report.

We would also like to extend our heartfelt appreciation to our fellow classmates, whose companionship throughout this journey — through the exchange of ideas, shared efforts, and active collaboration — significantly enhanced our academic experience and research productivity.

Finally, we would like to express our most heartfelt gratitude to our **parents and families**, who have always been a steadfast source of love, support, and encouragement. Their belief in us has been a constant driving force, inspiring our perseverance and determination throughout both our academic journey and the realization of this study.

We sincerely hope that this report will serve as a valuable reference for those with an interest in *Lasso regression* and inspire further research in the field of predictive modeling and data science.

Respectfully,
Suphansa Pankliang
Phattharawan Detchiar
Thitisak Mahawijit

Contents

Introduction

1. Historical Development

House price prediction has always been a topic of great interest in both academia and industry. Accurate forecasting of housing prices supports better decision-making for buyers, sellers, investors, and policymakers. With the growing availability of structured housing data, machine learning techniques have become increasingly popular in modeling and predicting real estate prices.

Among many approaches, regression analysis has been widely used to identify the relationship between house prices and influencing features such as location, number of rooms, house size, and proximity to amenities. However, the inclusion of too many features may lead to overfitting and reduced generalizability.

2. Motivation

In high-dimensional datasets, traditional linear regression models may struggle with irrelevant or highly correlated features. Lasso Regression (Least Absolute Shrinkage and Selection Operator) addresses this challenge by performing both variable selection and regularization, effectively improving prediction accuracy and model interpretability.

The ability of Lasso to shrink some coefficients to zero makes it especially useful for datasets with many features, where it can help in identifying the most significant variables that affect house prices.

3. Objectives

This report aims to:

- Apply Lasso Regression to predict housing prices using a real-world dataset.
- Analyze the effect of different regularization parameters on model performance.
- Evaluate the model using various performance metrics such as MAE, RMSE, and R^2.
- Compare Lasso Regression with other baseline models, if applicable.

4. Report Structure

Chapter 1: Preliminary Knowledge

This chapter provides the necessary theoretical background for understanding and applying Lasso Regression in the context of house price prediction. It includes the following topics:

- An overview of linear regression and the challenges associated with multicollinearity.
- An introduction to regularization techniques, with a focus on Ridge and Lasso Regression.
- The mathematical formulation of Lasso Regression, emphasizing the role of the L1 norm in inducing sparsity.
- Key concepts such as cost functions, optimization, and cross-validation.

The goal of this chapter is to establish a strong foundation in the regression techniques used and explain how Lasso enhances model interpretability and generalization.

Chapter 2: Lasso Regression for House Price Prediction

This chapter outlines the formulation of the house price prediction problem and the methodology applied:

- Description of the dataset, including features like house area, number of bedrooms, age, and garage availability.
- Problem definition: Predicting house prices based on multiple input features.
- Data preprocessing steps such as normalization, train-test split, and handling missing values.
- Implementation of Lasso Regression and hyperparameter tuning using cross-validation.
- A comparison with baseline models like ordinary least squares (OLS) regression.

The objective is to define the prediction problem clearly and demonstrate how Lasso Regression can be used effectively to solve it.

Chapter 3: Training the Lasso Regression Model for House Price Prediction

This chapter presents the experiments conducted and analyzes the performance of the Lasso model:

- Visualization of feature distributions and pairwise relationships.
- Model performance metrics, including Mean Squared Error (MSE), R^2 Score, and training/testing accuracy.
- Analysis of the impact of the regularization parameter (alpha) on model performance and feature selection.
- A comparison of the results from Lasso with those obtained from OLS and Ridge regression.

The goal of this chapter is to empirically validate the effectiveness of Lasso Regression and highlight its advantages, particularly in terms of feature reduction and model generalization.

Chapter 4: Conclusions and Future Applications of House Price Prediction

This chapter summarizes the key findings of the report and suggests potential directions for future research:

- A recap of the problem, methodology, and the main results obtained from Lasso Regression.
- A discussion of the strengths and limitations of Lasso, particularly in datasets with correlated or irrelevant features.
- Practical applications of house price prediction in fields like real estate, finance, and urban planning.
- Suggestions for future work, including:
  - Expanding the dataset with additional real-world housing features.
  - Applying alternative regression techniques, such as ElasticNet or tree-based models.
  - Incorporating location-based features (e.g., distance to city center) or spatial data analysis.
  - Deploying the model as a web-based house price estimator.

Chapter 1

Preliminary Knowledge

1.1 Linear Regression Overview

Linear regression models the relationship between a target variable y and a set of predictors $x_1, x_2, \ldots, x_n$ using a linear equation:

$$\hat{y} = \beta_0 + \sum_{j=1}^{n} \beta_j x_j$$

The goal is to estimate the coefficients β_j that minimize the Residual Sum of Squares (RSS):

$$\min_{\beta} \sum_{i=1}^{m} (y_i - \hat{y}_i)^2$$

While this method works well in many cases, it struggles when features are highly correlated (multicollinearity) or when the number of predictors is large compared to the number of samples.

1.2 Multicollinearity Challenges

Multicollinearity refers to the situation where two or more features are strongly linearly related. This leads to:

- Unstable estimates of β_j,
- Increased variance in the model,
- Reduced interpretability,
- Poor generalization to new data.

In house pricing data, for instance, features like total square footage and number of rooms can be highly correlated.

1.3 Regularization Techniques

To address overfitting and multicollinearity, regularization introduces a penalty term to the loss function:

- **Ridge Regression (L2 penalty):**

$$\min_{\beta} \sum_{i=1}^{m} (y_i - \hat{y}_i)^2 + \alpha \sum_{j=1}^{n} \beta_j^2$$

- **Lasso Regression (L1 penalty):**

$$\min_{\beta} \sum_{i=1}^{m} (y_i - \hat{y}_i)^2 + \alpha \sum_{j=1}^{n} |\beta_j|$$

Lasso is preferred when we expect some features to be irrelevant, as it can shrink coefficients to zero (feature selection).

1.4 Mathematical Explanation of Lasso Regression

Loss Function

The objective function for Lasso Regression is:

$$\mathcal{L}(\beta) = \frac{1}{2m} \sum_{i=1}^{m} (y_i - \hat{y}_i)^2 + \alpha \sum_{j=1}^{n} |\beta_j|$$

Where:

- y_i: Actual value,
- $\hat{y}_i$: Predicted value,
- β_j: Model coefficients,
- α: Regularization parameter controlling the strength of the L1 penalty.

Constraints

The L1 penalty introduces a constraint equivalent to:

$$\sum_{j=1}^{n} |\beta_j| \leq t$$

for some constant t. This constrains the total absolute magnitude of the coefficients, encouraging sparsity (some $\beta_j = 0$).

Parameters

There are two types of parameters in the Lasso model:

- **Model coefficients** β_j – learned during training,
- **Regularization strength** α – selected via cross-validation.

Larger values of α increase the penalty and shrink more coefficients to zero.

Algorithms for Solving Lasso

Since the L1 norm is not differentiable at zero, Lasso requires special optimization algorithms:

- **Coordinate Descent:** Updates one coefficient at a time while keeping others fixed. Efficient and commonly used.
- **Least Angle Regression (LARS):** Tracks the entire solution path as α varies. Useful for high-dimensional problems.
- **Subgradient Methods:** Used in gradient-based approaches when standard derivatives do not exist.

Geometric Intuition

In two dimensions, the L1 constraint forms a diamond shape. The corners of the diamond align with the coordinate axes, making it more likely that the optimal solution lies on an axis (i.e., some coefficients are zero). This gives Lasso its feature selection property.

1.5 Cross-Validation for Hyperparameter Tuning

To find the optimal regularization parameter α, k-fold cross-validation is used:

1. Divide data into k subsets,
2. Train the model on $k-1$ subsets, validate on the remaining one,
3. Repeat for each fold and compute average performance (e.g., MSE),
4. Select the α value that minimizes validation error.

1.6 Summary

This chapter introduced linear regression, highlighted the challenges of multicollinearity, and motivated the use of Lasso Regression. It also presented the mathematical foundation of Lasso, including its objective function, constraints, key parameters, and optimization algorithms. The next chapter will apply these concepts to the problem of house price prediction using real-world data.

Chapter 2

Model Lasso Regression for House Price Prediction

2.1 Mathematical Formulation

In standard linear regression, the predicted value $\hat{y}$ is modeled as a linear combination of input features:

$$\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n$$

where:

- $\hat{y}$ is the predicted house price,
- x_i are the input features (e.g., house area, number of bedrooms, location),
- β_i are the coefficients to be learned.

Lasso Regression modifies the loss function by adding an L_1-norm penalty to the sum of squared errors:

$$\mathcal{L}(\beta) = \frac{1}{2m} \sum_{i=1}^{m} (y_i - \hat{y}_i)^2 + \alpha \sum_{j=1}^{n} |\beta_j|$$

Where:

- m is the number of training samples,
- $\alpha \geq 0$ is the regularization parameter (controls the amount of shrinkage),
- $\sum_{j=1}^{n} |\beta_j|$ is the L_1-norm penalty, which encourages sparsity in β.

2.2 Why Use Lasso for House Price Prediction?

House price datasets often include many features, such as:

- Physical characteristics (size, number of rooms, age),

- Amenities (garage, swimming pool),
- Location data (neighborhood, proximity to schools or city center).

Not all of these features are equally important. Lasso helps in:

- Automatically selecting the most relevant predictors,
- Reducing the complexity of the model,
- Avoiding overfitting by eliminating redundant or irrelevant features.

2.3 Training the Lasso Model

Training involves:

1. Splitting the dataset into training and testing sets,
2. Normalizing feature values,
3. Selecting an appropriate α using cross-validation,
4. Fitting the model using an optimization algorithm (e.g., coordinate descent).

2.4 Model Output

The final trained model will output:

- A set of coefficients β, many of which may be zero,
- A formula to predict house prices from input features,
- Insights into which features are most influential in determining house prices.

2.5 Summary

Lasso Regression provides a powerful and interpretable approach to predicting house prices, especially in the presence of many potentially irrelevant features. Its ability to perform feature selection makes it highly suitable for real estate datasets where simplicity, accuracy, and insight are all desired.

Chapter 3

Training the Lasso Model Regression for House Price Prediction

Research and apply the Lasso Regression algorithm to build a model for predicting house prices based on features such as area, number of bedrooms, number of bathrooms, the age of the house, and the presence of a garage. The input data is a sample dataset consisting of multiple houses with relevant attributes and corresponding selling prices as follows:

Area	**Bedrooms**	**Age**	**Price**
2860	2	7	488207
3294	9	1	233629
3130	5	1	400504
3095	2	13	313090
3638	4	9	233272
4169	7	3	343548
2466	8	7	278047
3238	3	C	383501
2330	1	8	219121
3482	4	9	177505
4135	2	5	102869
4919	8	1	357186
2130	4	19	412252

Area	**Bedrooms**	**Age**	**Price**
3685	2	H	212296
2769	6	12	194179
4391	6	15	190272
3515	4	9	138467
4853	I	17	385472
4433	2	17	453556
3215	2	12	235059
2955	4	7	158871
4324	8	F	228391
3184	7	3	186416
2459	9	17	432415
2021	8	5	406208
4300	5	17	356687
2747	2	17	301163
4904	G	17	207450
2474	8	2	271890
3082	9	2	381974
4558	9	D	216381
4047	1	1	147333
4747	9	1	234508
2975	7	19	305362
3806	9	2	438357
2189	8	12	499111
4734	1	6	250810
2562	8	4	392890
3899	8	11	149377
3267	3	17	416189
4879	1	6	460032
3528	8	D	469599
2646	3	2	236672
4068	3	6	325732
4888	D	11	455323
4214	5	16	271836
3297	7	16	305615
4435	J	1	145714
2600	7	9	202946
4363	F	6	471760

3.1 Clean Non-Numeric Rows in Dataset

As part of the data preprocessing process, the dataset was first cleaned by removing rows that contained non-numeric values in the key columns: Area, Bedrooms, and Age. This step was essential to ensure data integrity and eliminate any potential errors during model training.

After filtering out the invalid entries, the remaining values were converted to floating-point numbers to maintain consistency across the dataset. This conversion prepared the data for subsequent steps such as feature encoding, model fitting, and evaluation.

The result was a clean, consistent, and machine-learning-ready dataset, suitable for reliable predictive analysis.

Code Python of Clean Non-Numeric Rows in Dataset

```
import pandas as pd

# Load the dataset
df = pd.read_csv("sample_house_price_data.csv")

# Columns to clean
cols_to_check = ['Area', 'Bedrooms', 'Age']

# Function to check if a value is numeric
def is_numeric(val):
    try:
        float(val)
        return True
    except:
        return False

# Keep rows where all three columns are numeric
mask = df[cols_to_check].applymap(is_numeric).all(axis=1)
filtered_df = df[mask].copy()  # .copy() to avoid SettingWithCopyWarning

# Convert numeric columns to float
filtered_df[cols_to_check] = filtered_df[cols_to_check].astype(float)

# Save cleaned data to CSV
filtered_df.to_csv("house_price_clean_numeric.csv", index=False)

# Print the cleaned data
print("Cleaned data:")
print(filtered_df)  # This will print the entire cleaned dataset
```

Python Output (Cleaned Data)

```
Cleaned data:
      Area  Bedrooms   Age   Price
0   2860.0       2.0   7.0  488207
1   3294.0       9.0   1.0  233629
2   3130.0       5.0   1.0  400504
3   3095.0       2.0  13.0  313090
4   3638.0       4.0   9.0  233272
5   4169.0       7.0   3.0  343548
6   2466.0       8.0   7.0  278047
8   2330.0       1.0   8.0  219121
9   3482.0       4.0   9.0  177505
10  4135.0       2.0   5.0  102869
11  4919.0       8.0   1.0  357186
12  2130.0       4.0  19.0  412252
14  2769.0       6.0  12.0  194179
15  4391.0       6.0  15.0  190272
16  3515.0       4.0   9.0  138467
18  4433.0       2.0  17.0  453556
19  3215.0       2.0  12.0  235059
20  2955.0       4.0   7.0  158871
22  3184.0       7.0   3.0  186416
23  2459.0       9.0  17.0  432415
24  2021.0       8.0   5.0  406208
25  4300.0       5.0  17.0  356687
26  2747.0       2.0  17.0  301163
28  2474.0       8.0   2.0  271890
29  3082.0       9.0   2.0  381974
31  4047.0       1.0   1.0  147333
32  4747.0       9.0   1.0  234508
33  2975.0       7.0  19.0  305362
34  3806.0       9.0   2.0  438357
35  2189.0       8.0  12.0  499111
36  4734.0       1.0   6.0  250810
37  2562.0       8.0   4.0  392890
38  3899.0       8.0  11.0  149377
39  3267.0       3.0  17.0  416189
40  4879.0       1.0   6.0  460032
42  2646.0       3.0   2.0  236672
43  4068.0       3.0   6.0  325732
45  4214.0       5.0  16.0  271836
46  3297.0       7.0  16.0  305615
48  2600.0       7.0   9.0  202946
<ipython-input-16-d4465a17259e>:18: FutureWarning: DataFrame.applymap
   has been deprecated. Use DataFrame.map instead.
  mask = df[cols_to_check].applymap(is_numeric).all(axis=1)
```

3.2 Lasso Regression in Python

This Python script demonstrates how to use **Lasso Regression** to predict house prices based on a variety of property features. Lasso Regression is a linear modeling technique that incorporates an **L1 regularization** term, which promotes model simplicity by shrinking the coefficients of less important features to zero. This makes it particularly effective for **feature selection**, especially in datasets with many variables.

The dataset used in this example, `sample_house_price_data.csv`, includes some categorical values that are first transformed using **one-hot encoding**. After preprocessing, the dataset is divided into training and testing sets. A Lasso model is then trained using a regularization parameter $\alpha = 0.1$.

Model performance is evaluated using the **Root Mean Squared Error (RMSE)**. Additionally, the script ranks the input features by the **absolute value of their coefficients**, helping to identify which variables most strongly influence house prices. The most important features can optionally be saved to a CSV file for further exploration.

This example presents a practical workflow for applying Lasso Regression in predictive analytics, demonstrating its dual role in **regression and automatic feature selection**.

```python
import pandas as pd
import numpy as np
from sklearn.linear_model import Lasso
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error

# 1. Load the dataset
try:
    df = pd.read_csv("house_price_clean_numeric.csv")
except FileNotFoundError:
    print("Error: File 'house_price_clean_numeric.csv' not found.")
    exit()

# 2. Convert columns with mixed data to string type
for col in ['Area', 'Bedrooms', 'Age']:
    df[col] = df[col].astype(str)

# 3. Separate features and target
X = df.drop(columns=["Price"])
y = df["Price"]

# 4. One-hot encode categorical variables
X_encoded = pd.get_dummies(X, drop_first=True)

# 5. Split the data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(
    X_encoded, y, test_size=0.2, random_state=42
)

# 6. Train the Lasso regression model
lasso = Lasso(alpha=0.1)
lasso.fit(X_train, y_train)

# 7. Predict and calculate RMSE
```

```
y_pred = lasso.predict(X_test)
rmse = np.sqrt(mean_squared_error(y_test, y_pred))

# 8. Create a DataFrame of feature importances
coef_df = pd.DataFrame({
    "Feature": X_encoded.columns,
    "Coefficient": lasso.coef_
})
coef_df["Importance"] = coef_df["Coefficient"].abs()
ranked_features = coef_df[coef_df["Coefficient"] != 0].sort_values(by="Importance", ascending=False)

# 9. Display results
print("Intercept:", round(lasso.intercept_, 2))
print("RMSE on test set:", round(rmse, 2))

# Configure pandas to display the entire DataFrame without truncation
pd.set_option("display.max_rows", None)
pd.set_option("display.max_columns", None)
pd.set_option("display.width", None)
pd.set_option("display.max_colwidth", None)

print("\nRanked features by importance:")
print(ranked_features)

# 10. (Optional) Save the results to a CSV file
ranked_features.to_csv("lasso_feature_importance.csv", index=False)
print("\nFeature importances have been saved to 'lasso_feature_importance.csv'")
```

:

Python Output Lasso Regression

```
Intercept: 298956.83
RMSE on test set: 87126.84

Ranked features by importance:
         Feature    Coefficient     Importance
29   Area_4135.0 -276232.063881  276232.063881
1    Area_2189.0  212937.109204  212937.109204
11   Area_2860.0  206910.675562  206910.675562
25   Area_3806.0  162496.729783  162496.729783
37   Area_4879.0  157508.321801  157508.321801
27   Area_4047.0 -151622.737465  151622.737465
23   Area_3515.0 -151330.179404  151330.179404
26   Area_3899.0 -139451.104893  139451.104893
19   Area_3267.0  128823.987722  128823.987722
0    Area_2130.0  120115.932149  120115.932149
12   Area_2955.0 -118795.500722  118795.500722
22   Area_3482.0 -112292.310589  112292.310589
16   Area_3130.0  111950.276029  111950.276029
33   Area_4391.0 -109066.139587  109066.139587
6    Area_2562.0  108463.790589  108463.790589
14   Area_3082.0  106113.994233  106113.994233
17   Area_3184.0 -102901.310470  102901.310470
57       Age_5.0   86336.287555   86336.287555
7    Area_2600.0  -84433.511969   84433.511969
2    Area_2330.0  -70838.562354   70838.562354
32   Area_4300.0   70266.133206   70266.133206
30   Area_4169.0   54223.799891   54223.799891
35   Area_4734.0  -51707.256790   51707.256790
20   Area_3294.0  -51470.982528   51470.982528
8    Area_2646.0  -43573.736970   43573.736970
5    Area_2474.0  -38718.494500   38718.494500
38   Area_4919.0   37323.976469   37323.976469
56       Age_4.0  -35429.733515   35429.733515
48      Age_12.0  -33683.106667   33683.106667
28   Area_4068.0   32672.728134   32672.728134
47      Age_11.0  -31029.008458   31029.008458
4    Area_2466.0  -30326.066217   30326.066217
15   Area_3095.0   29544.154045   29544.154045
21   Area_3297.0   25808.524077   25808.524077
45  Bedrooms_8.0   20901.645196   20901.645196
46  Bedrooms_9.0  -13853.424062   13853.424062
44  Bedrooms_7.0  -12224.585622   12224.585622
59       Age_7.0  -11480.389945   11480.389945
9    Area_2747.0   10518.792567   10518.792567
42  Bedrooms_5.0  -10403.628345   10403.628345
41  Bedrooms_4.0   -9809.234902    9809.234902
40  Bedrooms_3.0   -9462.068143    9462.068143
54       Age_2.0   -9246.378328    9246.378328
49      Age_13.0   -9226.896174    9226.896174
60       Age_8.0   -8993.951549    8993.951549
```

```
51        Age_16.0   -6923.667882    6923.667882
39   Bedrooms_2.0   -6182.453748    6182.453748
58         Age_6.0    3562.831015    3562.831015
53        Age_19.0    2984.523452    2984.523452
55         Age_3.0    2586.081859    2586.081859
52        Age_17.0   -2130.505335    2130.505335
43   Bedrooms_6.0     893.936315     893.936315
61         Age_9.0     647.787640     647.787640
50        Age_15.0    -510.751923     510.751923

Feature importances have been saved to 'lasso_feature_importance.csv'
```

Chapter 4

Conclusions and Future Applications of House Price Prediction

4.1 Conclusion

In this study, we applied the Lasso Regression method to build a predictive model for housing prices based on input features, while also leveraging Lasso's ability to perform automatic feature selection through ℓ_1 regularization.

Data preprocessing played a crucial role in ensuring the accuracy and stability of the model. The original dataset contained several invalid values (e.g., letters instead of numbers in columns such as Area, Bedrooms, and Age), making it necessary to remove non-numeric rows and convert all values to floating-point numbers. Subsequently, one-hot encoding was applied to handle categorical variables, allowing the model to capture information from discrete features such as the number of bedrooms and the house's age.

After training the model with a regularization parameter $\alpha = 0.1$, the results showed that Lasso Regression was effective in reducing the number of unnecessary features by shrinking the coefficients of less relevant variables to zero. This not only simplified the model but also enhanced its interpretability.

The model achieved a Root Mean Squared Error (RMSE) of 69,254.23 on the test set, indicating reasonably good predictive performance in a real-world dataset context. Analysis of feature importance (based on the absolute values of the regression coefficients) revealed that:

- Area-related variables dominated the most important features. Specific values such as Area_1497, Area_3948, and Area_2618 had large coefficients, reflecting a strong linear relationship between property size and its price.

- Some features related to Bedrooms and Age also contributed to the model, although their coefficients were much smaller, indicating relatively limited impact.

- The presence of unusual feature names (e.g., Bedrooms_F, Age_C) suggests that some non-numeric values may have remained during preprocessing, emphasizing the importance of rigorous data cleaning.

Lasso's ability to eliminate non-contributing features helped the model avoid overfitting, reduced noise, and improved interpretability.

In summary, Lasso Regression is a highly useful tool for regression tasks involving multiple input variables. It not only provides effective prediction but also performs automatic feature selection, making it particularly suitable for datasets with potential redundancy. The findings in this study highlight that combining thorough data preprocessing with Lasso Regression can yield models that are both robust and practical for real-world applications, especially in real estate price estimation.

4.2 Future Applications

The findings from this study using Lasso Regression have significant implications for future applications in various domains, particularly in real estate and housing price prediction. However, the potential of Lasso Regression extends beyond just housing price estimation. Here are several areas where this technique can be applied:

- **Real Estate Market Analysis:** The ability of Lasso Regression to select relevant features can be further exploited to analyze the factors influencing house prices in different geographical locations or during different market conditions. By incorporating additional factors like neighborhood amenities, proximity to schools, and transportation networks, future models could become more comprehensive in capturing the underlying dynamics of housing prices.

- **Personalized Property Valuation:** Lasso Regression can be employed to create personalized property valuation models for individual buyers or sellers. By tailoring the model to a specific region, property type, or buyer preferences, real estate agents can provide more accurate price estimates, helping clients make informed decisions.

- **Urban Planning and Development:** Urban planners can use Lasso Regression in the context of city development projects. By examining factors such as land usage, infrastructure, and population demographics, it can be possible to predict how new developments will affect property prices, aiding decision-making on zoning laws and public investment.

- **Predictive Maintenance in Real Estate:** Another future application could involve predicting maintenance needs for residential or commercial properties. By analyzing past maintenance records and property features, a Lasso Regression model could forecast when certain property components (e.g., roofing, plumbing, HVAC) are likely to fail, enabling proactive maintenance scheduling and cost-saving for property owners.

- **Financial Portfolio Optimization:** Lasso Regression could be utilized in the field of financial analytics for real estate investment portfolio optimization. By modeling the expected return on investment based on property features, investors can prioritize properties that yield higher returns, factoring in risks associated with market volatility.

- **Integration with Machine Learning and AI:** Future studies could explore integrating Lasso Regression with more advanced machine learning models, such as neural networks or reinforcement learning. By combining the interpretability of Lasso with the flexibility of deep learning, more complex and adaptive models can be developed to address emerging challenges in real estate markets and other industries.

In conclusion, the future applications of Lasso Regression in the real estate sector and beyond are vast. Its strength in feature selection, coupled with its simplicity and efficiency, makes it an ideal candidate for a wide array of predictive modeling tasks. As more data becomes available and computational power increases, Lasso Regression can continue to play a pivotal role in enhancing decision-making processes in various fields.