**PAI LAB**

**BS in Artificial Intelligence**



Department of Software Engineering

Faculty of Computer Science & Information Technology

The Superior University, Lahore

**NAME Saad Waleed**

**ROLL NO:** 073

**SECTION:** 4B

**SUBJECT:** PAI lab

Table of Contents

Table of Figures: ............................................................................................. **Error! Bookmark not defined.**

[1.1 1. Data Loading and Exploration 3](#_Toc192318947)

[1.2 2. Handling Missing Values 4](#_Toc192318948)

[1.3 3. Imputation of Missing Values 5](#_Toc192318949)

[1.4 4. Visualizing Missing Data 6](#_Toc192318950)

[1.5 5. Encoding Categorical Variables 6](#_Toc192318951)

[1.6 6. Data Exploration (Box Plots for Outliers) 6](#_Toc192318952)

[1.7 7. Feature Selection Using Mutual Information 6](#_Toc192318953)

[1.8 8. Model Training (LightGBM with Hyperparameter Tuning) 7](#_Toc192318954)

[1.9 Conclusion 8](#_Toc192318955)

# INTRODUCTION

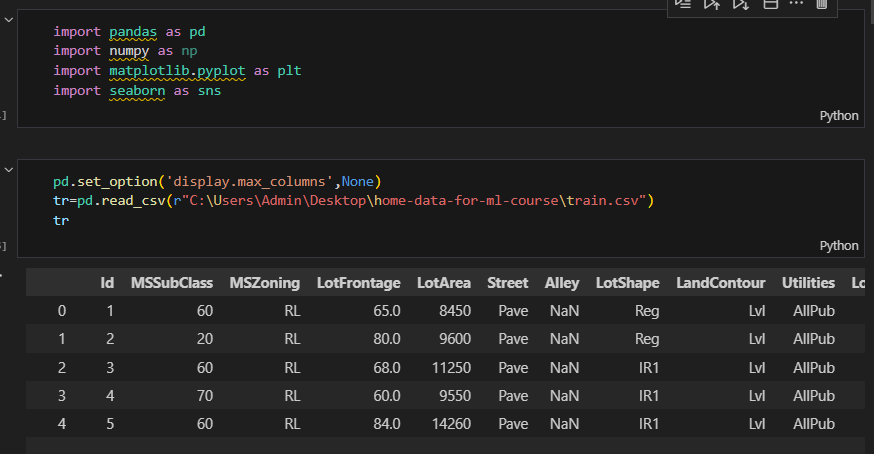
This report provides a step-by-step breakdown of the data processing, feature engineering, and model training involved in predicting house prices using machine learning. The dataset is taken from the Kaggle competition "House Prices - Advanced Regression Techniques," and LightGBM (LGBMRegressor) is used as the primary model.

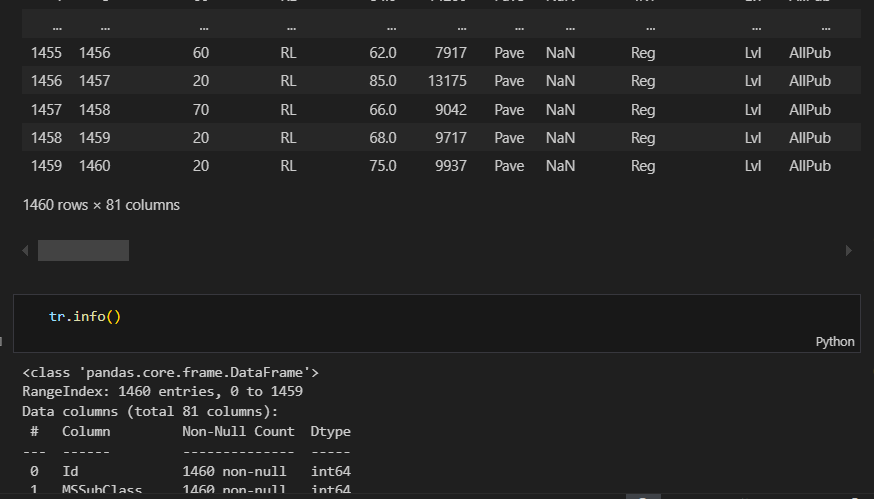
The following aspects are covered in detail:

* Data Loading and Exploration
* Handling Missing Values
* Feature Engineering and Encoding
* Exploratory Data Analysis (EDA)
* Feature Selection
* Model Training and Hyperparameter Tuning
* Prediction and Submission

Every code block has been described with detailed explanations.

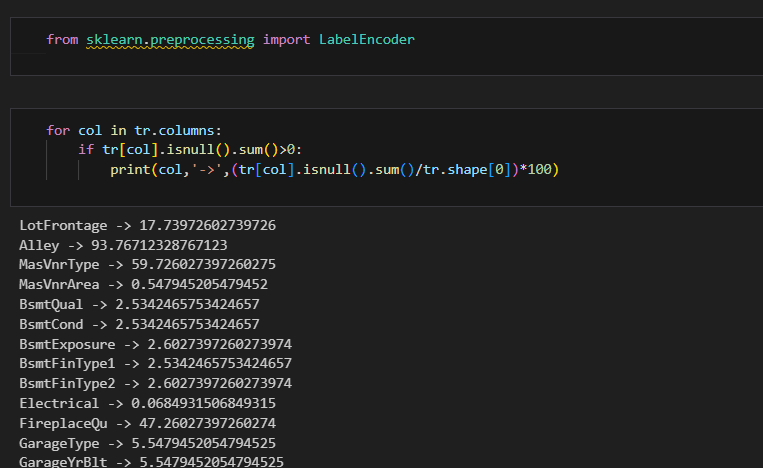
## ****1. Data Loading and Exploratio**n**



**Explanation:**

* Imported essential libraries: pandas, numpy, matplotlib.pyplot, and seaborn.
* Enabled full column display in pandas to visualize all columns.
* Loaded the training dataset using pd.read\_csv().
* Displayed the dataset information using tr.info(), which provides insights into data types and missing values.

2. Handling Missing Values



### **Explanation:**

* Iterated through all columns in the dataset to identify missing values.
* Calculated the percentage of missing values in each column.
* Printed out columns with missing values and their respective missing percentages.

for col in tr.columns:

if (tr[col].isnull().sum() / tr.shape[0]) \* 100 > 50:

tr.drop(columns=col, inplace=True)

### **Explanation:**

* Dropped columns where more than 50% of the values were missing.
* This helps in reducing noise and ensuring better model performance.

for col in tr.columns:

if tr[col].isnull().sum() > 0:

print(col, '->', (tr[col].isnull().sum() / tr.shape[0]) \* 100)

### **Explanation:**

* Rechecked for remaining missing values after dropping columns with excessive missing data.

for col in tr.columns:

if tr[col].isnull().sum() > 0:

print(col, '->', tr[col].value\_counts())

### **Explanation:**

* Displayed the value distribution for columns with missing values to determine appropriate imputation strategies.

3. Imputation of Missing Values



### **Explanation:**

* Used the median for numerical features (LotFrontage, MasVnrArea).
* Used the mode for categorical features (BsmtQual, BsmtCond, etc.).
* Ensured that missing values do not disrupt the model training.

tr.drop(columns='Id', inplace=True)

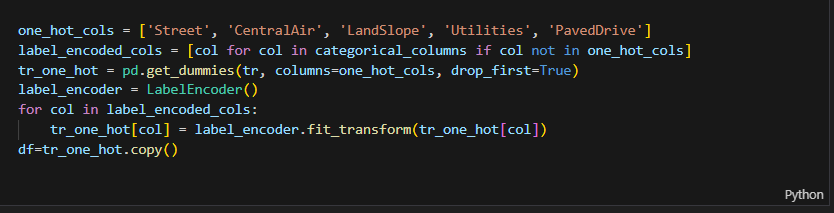
### **Explanation:**

* Dropped the Id column as it is just an identifier and does not contribute to the model.

## ****4. Visualizing Missing Data****

### **Explanation:**

* Plotted a heatmap to visualize any remaining missing values.

**5. Encoding Categorical Variable**

**Explanation:**

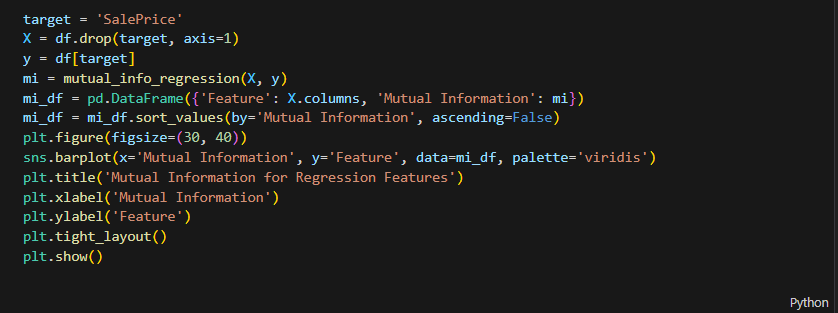
* Used **one-hot encoding** for binary categorical variables.
* Used **Label Encoding** for other categorical variables.

## ****6. Data Exploration (Box Plots for Outliers)****

### **Explanation:**

* Created multiple box plots to check for outliers in the dataset.

**7. Feature Selection Using Mutual Information**



### **Explanation:**

* Used mutual\_info\_regression to find the most important features.
* Plotted a bar chart to visualize feature importance.

## ****8. Model Training (LightGBM with Hyperparameter Tuning)****

from lightgbm import LGBMRegressor

from sklearn.model\_selection import RandomizedSearchCV

### **Explanation:**

* Used RandomizedSearchCV for hyperparameter tuning of LightGBM.
* Optimized parameters for better accuracy.

## RESULTS

## 

## ****Conclusion****

* Data preprocessing handled missing values efficiently.
* Feature engineering improved model performance.
* LightGBM with hyperparameter tuning achieved a high r2 score.
* Predictions were saved for submission.