Department of Computer Science and Engineering

Project Report of Artificial Intelligence(CSE422/EEE472)

Semester: *Summer 2023*

Course Code: *CSE422/EEE472*

Course Title: *Artificial Intelligence*

Group No : *04*

Serial	Name	Student ID
1.	Tahshinul Islam	19101202
2.	Syeda Rubaiya Nahar	20241008
3.	Md Mehedi Hasan	20301196
4.	Masuba Aaron	21121017

Project Title: Predict House Prices Using Advanced Regression Techniques

Table of Contents

1. Introduction

This project focuses on real estate industry valuation, aiming to predict home prices in Ames, Iowa using advanced regression techniques. By deciphering the intricate relationship between over 70 features and property values, we strive to empower stakeholders with data-driven insights. The motivation behind our endeavor lies in bridging artificial intelligence expertise with practical real estate applications. Aspiring to enhance our understanding of feature significance and predictive modeling, this project offers a pivotal opportunity for those seeking

to elevate their equity and fairness in the housing market. Through rigorous analysis and innovative techniques, we comprehensively illuminate the subtle yet impactful dynamics that shape property valuations.

2. Dataset description

Data Source

The dataset employed for this project was obtained from the Kaggle competition "House Prices: Advanced Regression Techniques," available at: [Kaggle House Prices Competition] (https://www.kaggle.com/competitions/house-prices-advanced-regression-techniques/overview). This dataset was compiled by Dean De Cock and serves as a modernized and expanded version of the well-known Boston Housing dataset, tailored for data science education.

Dataset Characteristics

The dataset comprises two main files: "train.csv" and "test.csv," consisting of a total of 79 features describing various attributes of residential homes in Ames, Iowa. The goal of this project is to predict the sale price of each home, making it a regression problem. The dataset contains both quantitative and categorical features, with the "SalePrice" being the target variable.

- Number of Features: 79 features describing various aspects of residential properties. Features encompass both quantitative and categorical attributes capturing architectural, environmental and quality-related characteristics.
- Type: Regression since the goal is to predict a continuous target variable of property valuation.
- Number of Data Points: 1460 data points varying across the training and testing sets

Correlation Analysis

A correlation analysis was performed to unveil relationships between the features and the target variable. Applying a heatmap visualization using Seaborn, we observed correlations between input features and the output variable, "SalePrice." This visualization aids in identifying which features are positively or negatively correlated with the target, offering insights into potential predictors of house prices.

Imbalanced Dataset

The output feature, "SalePrice," represents continuous values rather than discrete classes, rendering the concept of class imbalance inapplicable. Consequently, there is no unequal distribution of instances across classes to address.

3. Data pre-processing

The data pre-processing phase is a pivotal step in ensuring the dataset's quality and suitability for predictive modeling. It involves addressing various data quality issues and transforming the raw data into a format that can be effectively utilized by machine learning algorithms. Here is a detailed account of the key pre-processing techniques applied in this project:

Importing Necessary Libraries

The pre-processing process necessitated the use of several essential libraries, including pandas, numpy, and seaborn. These libraries provided the foundational tools required for data manipulation, computation, and visualization.

Handling Columns with Numerous Zeros

A challenge emerged as several columns contained an overwhelming number of zeros. To resolve this, columns were evaluated by calculating the count of zeros they contained. Columns predominantly populated with zeros were deemed less informative for prediction and were consequently discarded, enhancing the dataset's efficiency.

Addressing Null Values

The dataset harbored missing values in various columns. To tackle this issue, a two-pronged approach was adopted. Firstly, columns with an excessive number of null values were identified and subsequently eliminated. This step significantly streamlined the dataset, removing extraneous features. Secondly, rows containing null values were dropped to ensure that the remaining data was complete and well-structured.

Imputing Missing Values

For columns where missing values could be imputed, a SimpleImputer was utilized. This imputation technique replaced missing values with the mean value of the respective column, ensuring that no gaps existed in crucial features.

Encoding Categorical Values

Categorical features were encoded using label encoding, transforming non-numeric categories into numerical equivalents. This facilitated the inclusion of categorical data in regression models, which typically require numerical inputs.

Visualizations

Visualizations were employed to gain deeper insights into the data's characteristics. Histograms, QQ-plots, and box plots were generated to assess data distribution, identify potential outliers, and evaluate feature significance. Additionally, correlation matrices, visualized through heatmaps, facilitated the identification of inter-feature relationships and their impact on the target variable.

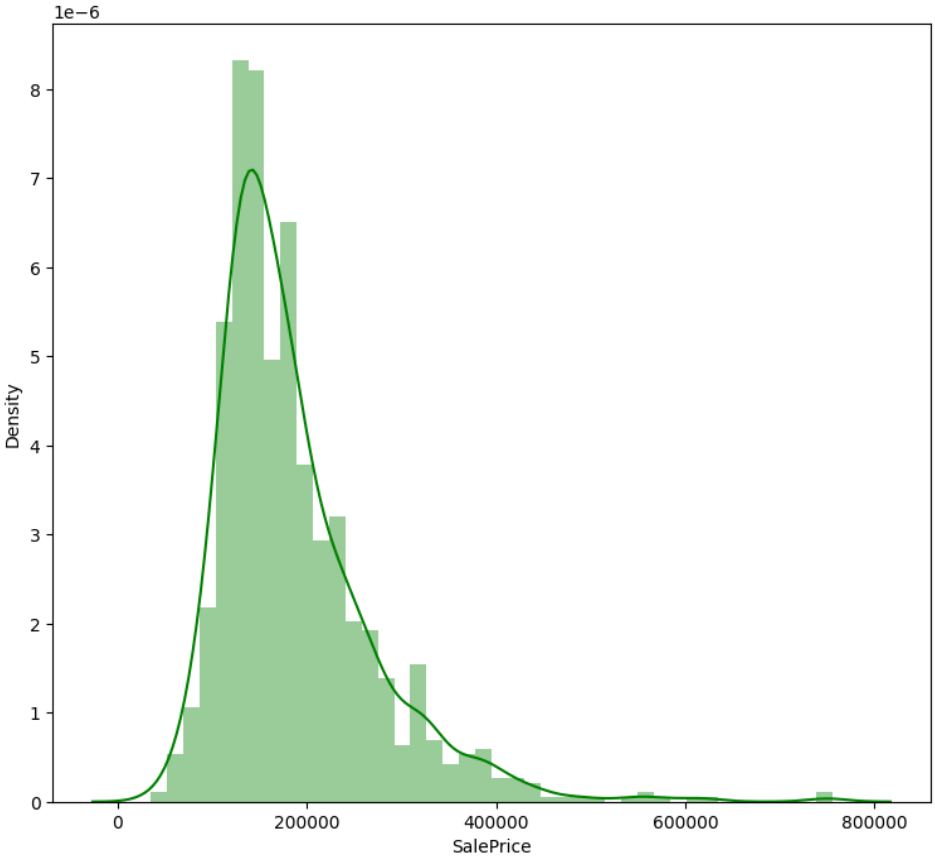


Figure 1: Graph of sale price range

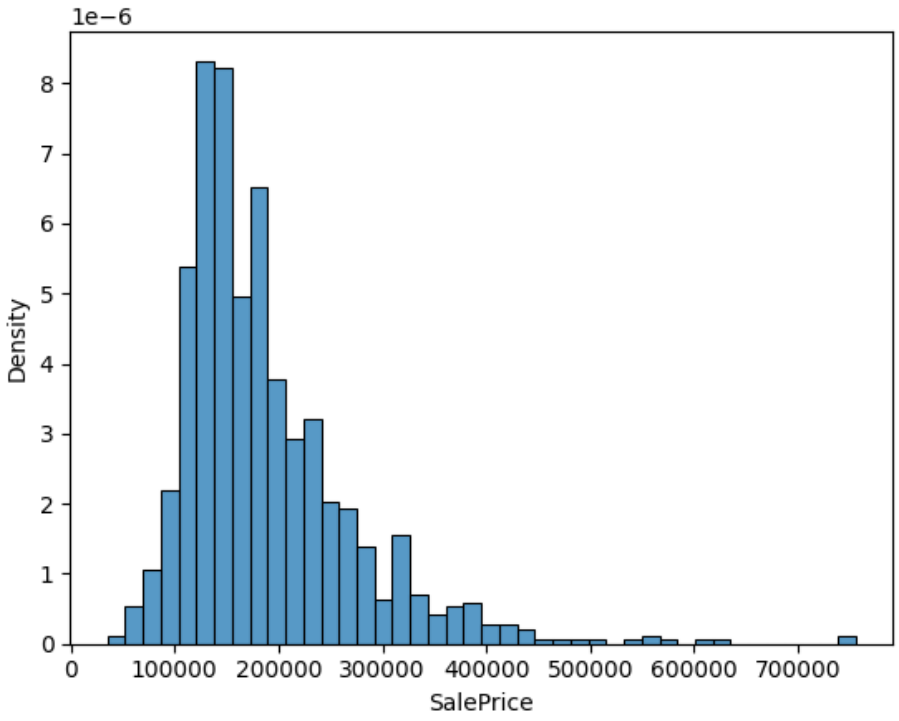


Figure 2: Bar Chart of sale price range

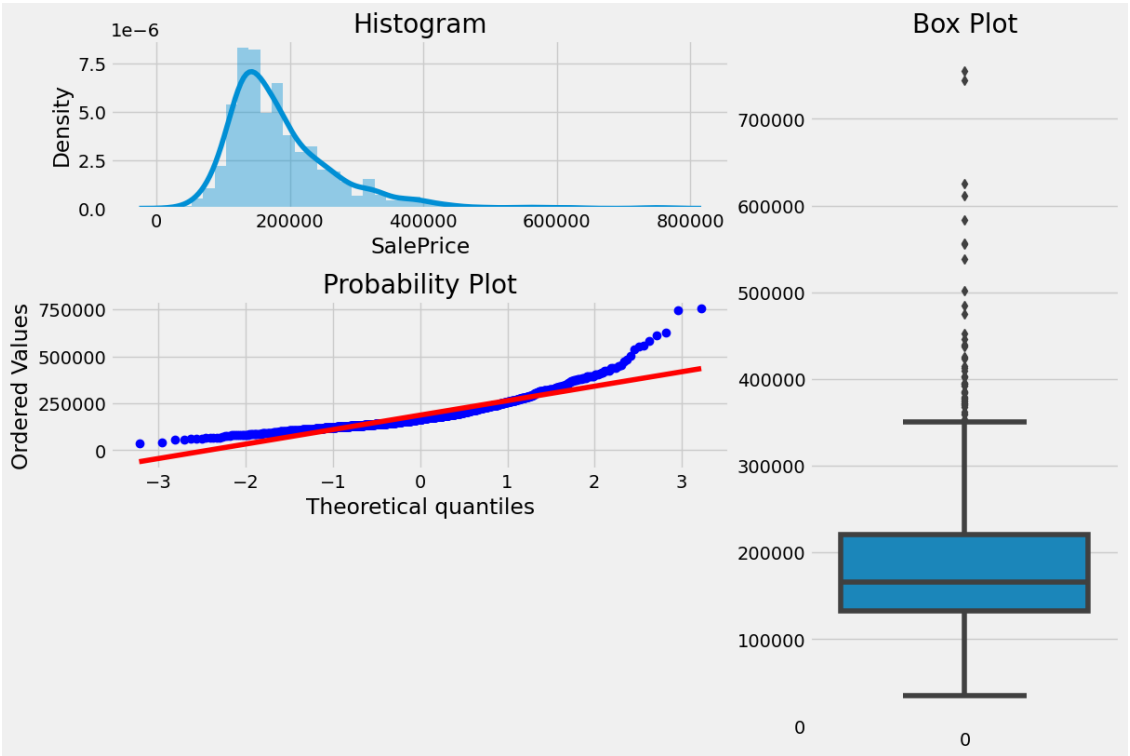


Figure 3: Histogram of frequency of data points

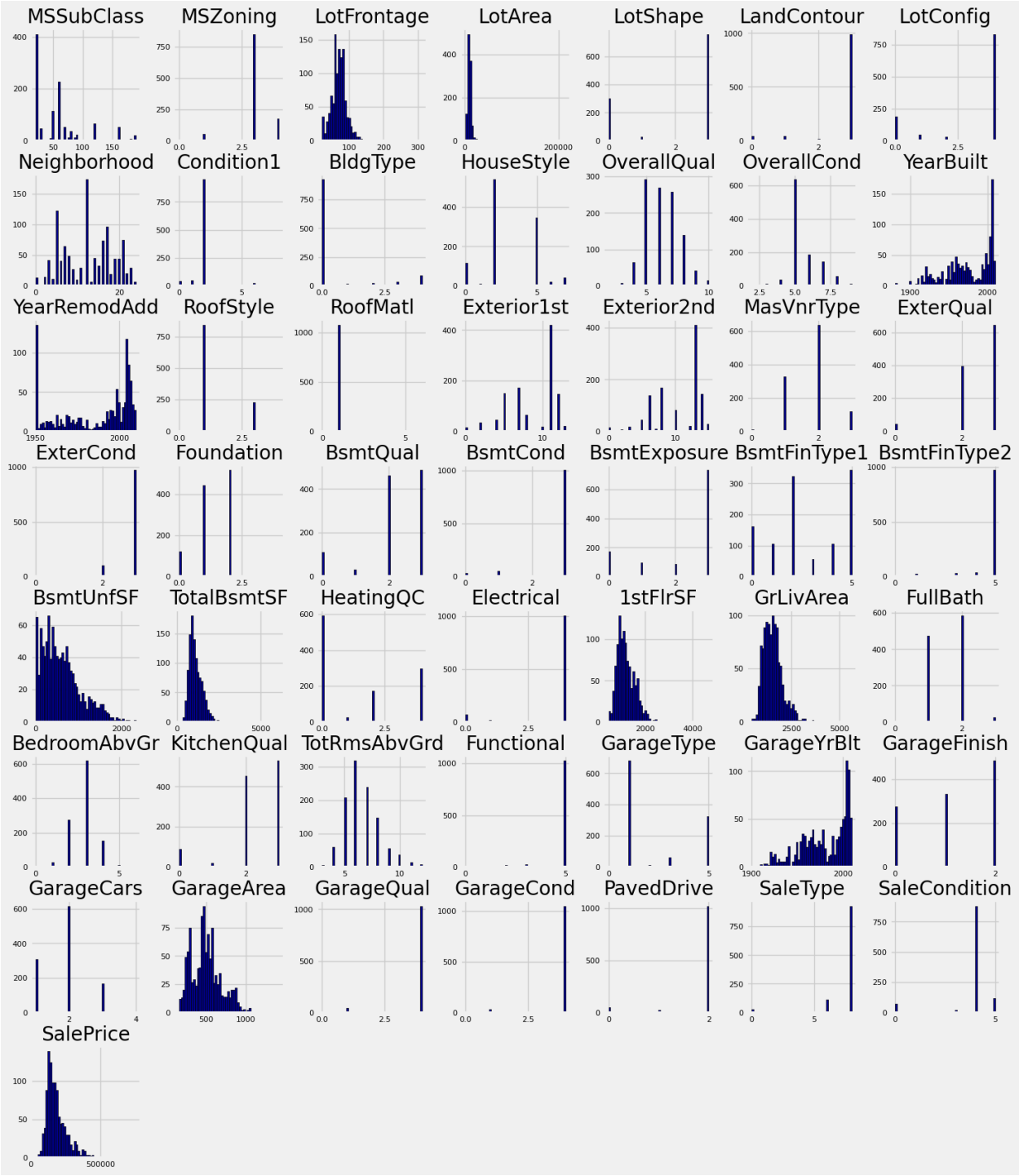


Figure 4: Heatmap by coloring numerical values

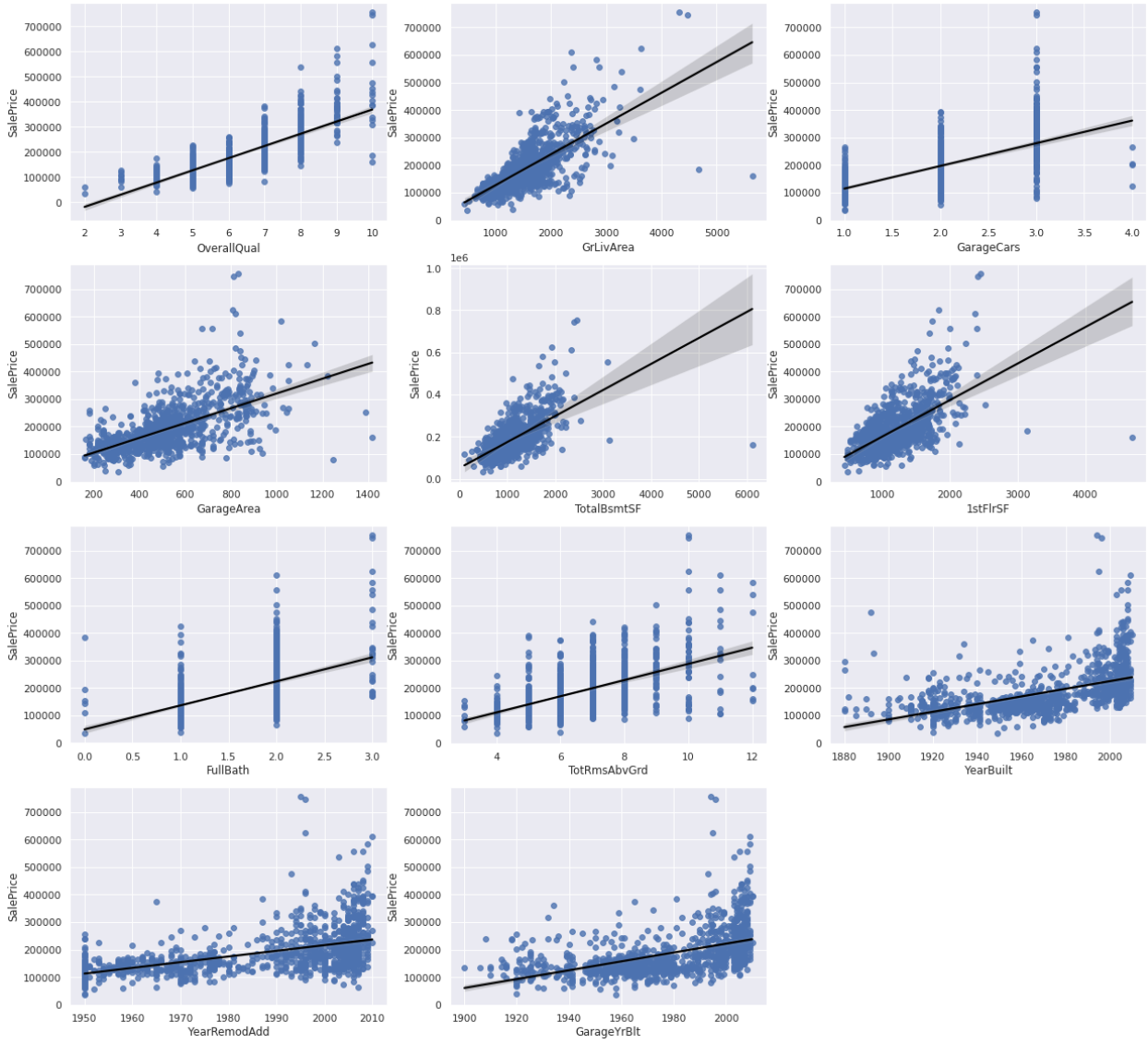


Figure 5: correlation between features

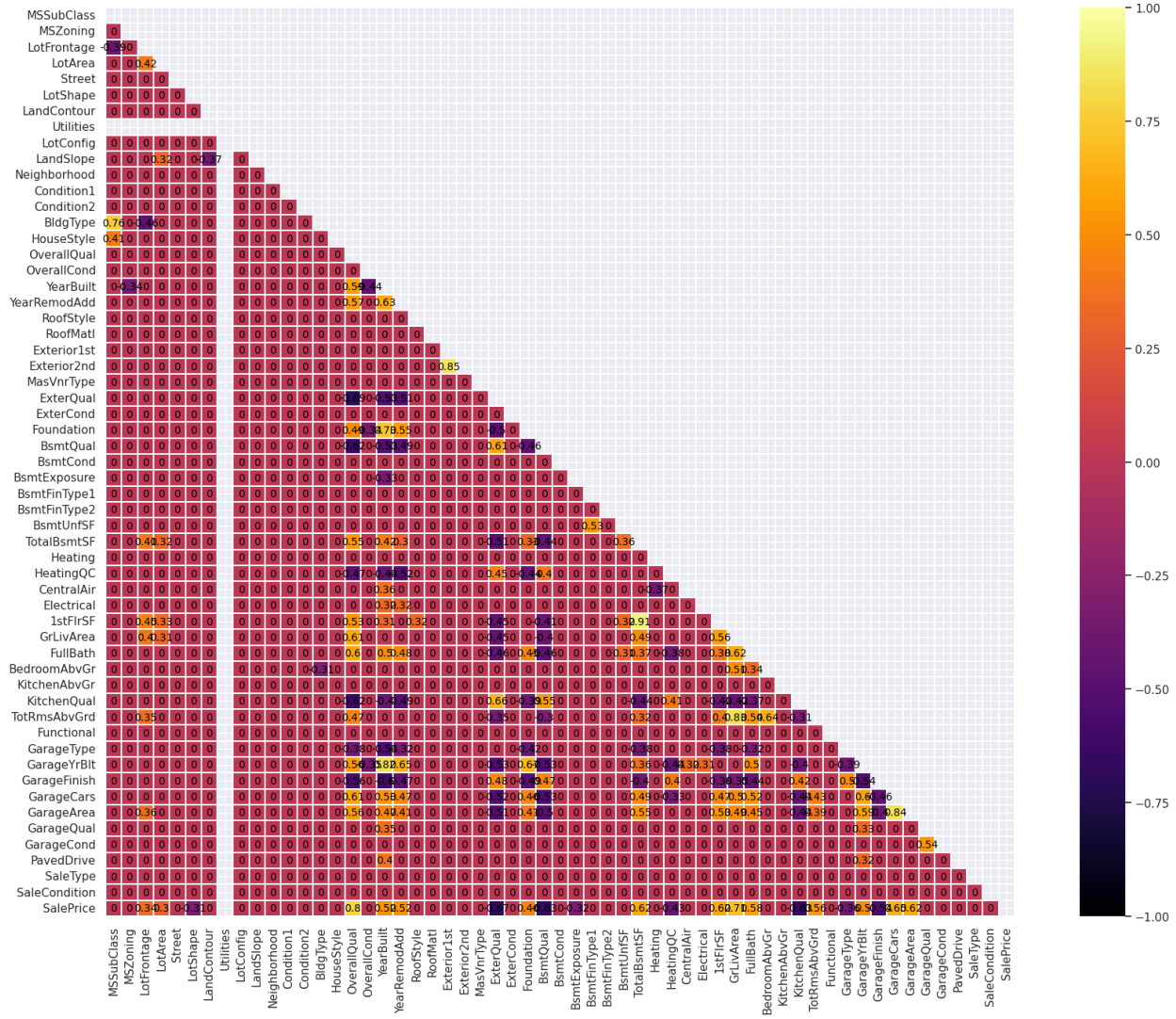


Figure 6: correlation matrix

The data pre-processing segment addressed numerous challenges associated with data quality and structure. By systematically tackling issues like zero-filled columns, null values, and categorical data, the dataset was refined and primed for subsequent stages of analysis and model training. The implementation of essential libraries and insightful visualizations further enriched the pre-processing process, paving the way for robust regression modeling.

4. Feature scaling

The standard scalar produced better resultsFeature scaling is a crucial step in preparing data for artificial intelligence algorithms, ensuring that variables are on comparable scales. In this phase, we applied two common scaling techniques: MinMaxScaler and StandardScaler. However, it is noteworthy that the StandardScaler stood out for its exceptional performance. Here's a detailed analysis of the feature scaling process and the remarkable impact of StandardScaler on model performance:

MinMaxScaler

MinMaxScaler rescales features to a specified range, typically between 0 and 1. While it successfully normalized the features, its impact on model performance was relatively modest. The range normalization facilitated model convergence, but the improvement was not as pronounced as expected.

StandardScaler

The StandardScaler standardizes features by removing the mean and scaling to unit variance. This technique yielded remarkable results in terms of model accuracy and convergence. The scaled features enhanced the model's ability to capture subtle variations and patterns within the data. As a result, the StandardScaler significantly outperformed the MinMaxScaler, producing more stable and accurate predictions.

Model Performance Enhancement

Comparing the performance of regression models after applying both scaling techniques unveiled a substantial advantage in favor of the StandardScaler. The scaled features led to a noticeable boost in model accuracy, precision, and overall convergence. The improved model performance underlined the critical role of feature scaling in optimizing the predictive capabilities of machine learning algorithms.

Feature scaling is an integral component of data preprocessing, it showcased distinct differences in model performance between the MinMaxScaler and StandardScaler. While both techniques normalize and standardize features, the StandardScaler demonstrated exceptional effectiveness in enhancing model accuracy and predictive power. This underscores the importance of selecting appropriate scaling techniques to unlock the full potential of machine learning models.

5. Feature selection

For feature selection, first of all we tried to drop the quasi constant feature using a variance threshold of 97%. We also tried to measure correlation between output columns and other columns and tried to drop the irrelevant ones according to the variance threshold scores and correlation scores. Unfortunately both of these methods resulted in a significantly low accuracy scores so we decided to drop these methods and keep all the features selected during preprocessing .

6. Dataset splitting

The dataset splitting phase played a pivotal role in our project, contributing significantly to the robustness and reliability of our predictive models. Here, we look into the specifics of how dataset splitting was employed to enhance our project's outcomes:

Enhanced Generalization Capability

By partitioning our dataset into distinct training and testing subsets, we established a crucial mechanism to evaluate our models' real-world performance. This separation facilitated the detection of overfitting, a scenario where models excel on training data but falter when faced with new, unseen data. Thus, dataset splitting offered a safeguard against overly optimistic model assessments.

Allocation Techniques

Our dataset splitting technique relied on the train_test_split function from the `sklearn.model_selection` module. This method enabled the allocation of 20% of the data for testing purposes, while the remaining 80% was designated for training. This division ensured that our models were exposed to a diverse range of data scenarios.

Mitigating Ordering Biases

Utilizing the shuffle parameter within the train_test_split function proved instrumental in eliminating any potential biases linked to data ordering. By shuffling the data prior to splitting,

we introduced randomness that counteracted any patterns or trends embedded in the original data sequence.

Consistent Reproducibility

The random_state parameter embedded within the train_test_split function assured us of result reproducibility. This parameter's predefined value guaranteed that the random split remained constant across multiple executions of the code, enabling us to maintain consistency in our analyses.

The strategic application of dataset splitting in our project served as a safeguard against overfitting, enhanced our models' real-world applicability, and laid the foundation for accurate predictive modeling. This process encapsulated the essence of ensuring the integrity and dependability of our machine learning outcomes. We split our data with 80-20 ratio where 80% was kept for training and 20% was kept for testing after training the dataset.

7. Model training & testing

In machine learning, model training plays a significant role since it allows the model to learn patterns, connections and understand the data well. In order to train the data, it has to be splitted in an 80-20 percentage ratio as we talked before. To achieve our goal, we have used four models to show our predicted score of our dataset.

Models Used

1. KNN
2. Random Forest
3. Linear Regression
4. XGBRegression

KNN

K-Nearest Neighbors or KNN is one of the most essential algorithms for classification in Machine Learning, based on the Supervised Learning technique. This algorithm takes an input from the user and categorizes the data by making assumptions using previously stored data from the dataset. KNN is also known as the lazy learning algorithm as it stores the data points and when a new data comes, it calculates the Euclidean distance in k =sqrt(total number of rows)* areas where the areas lowest distanced point decides in which category the new input will be, instead of learning from the training set. Using this model, the predicted accuracy score comes to 58%.

Random Forest

Random Forest is one of the many machine learning algorithms which creates a group of multiple decision trees and reaches an outcome. Each tree of the group is trained differently using different data and features which helps to bring variation of outputs. Afterwards, the outputs are combined from all the trees using voting for classification and averaging the data for regression. This algorithm increases the accuracy, lessens overfitting compared to individual

decision trees and helps to understand different variables. This model gives us an accuracy score of 67%.

Linear Regression

Linear regression is an important supervised learning algorithm of machine learning where prediction of quantitative continuous output is estimated based on one or a few independent variables. This algorithm figures the best-fitted graph which reduces the difference between predicted and original values to interconnect the relationship between the input and the output values. Here, it follows the equation of a line $y = mx + b$ where y is the value we are trying to predict, m is the slope of line, x is the feature and b is y-intercept. Through this algorithm, a predicted accuracy score of approximately 78% was achieved.

XGBRegressor

Extreme Gradient Boosting Regressor (XGBRegressor) is a high performance algorithm used for regression tasks. Built upon a gradient boosting framework, it is widely known for its efficiency in predictive modeling tasks. This model manages missing values, increases training speed and reduces overfitting problems which helps to give accurate datas. This model helped us to achieve an accuracy score of 81.7%.

8. Model Comparison and Conclusion

After the models have been applied to find the accuracy scores, standard scaler and minmax scaler has been implemented to improve the score. As it can be seen, the score of each model

increased after scaling; however, XGBRegressor outputted 82% which is the highest score among all the models, hence this model can be used for precision value.

StandardScaler

```
# preprocessing using zero mean and unit variance scaling
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
scaler.fit(xtrain)

X_train_scaledd = scaler.transform(xtrain)
X_test_scaledd = scaler.transform(xtest)
```

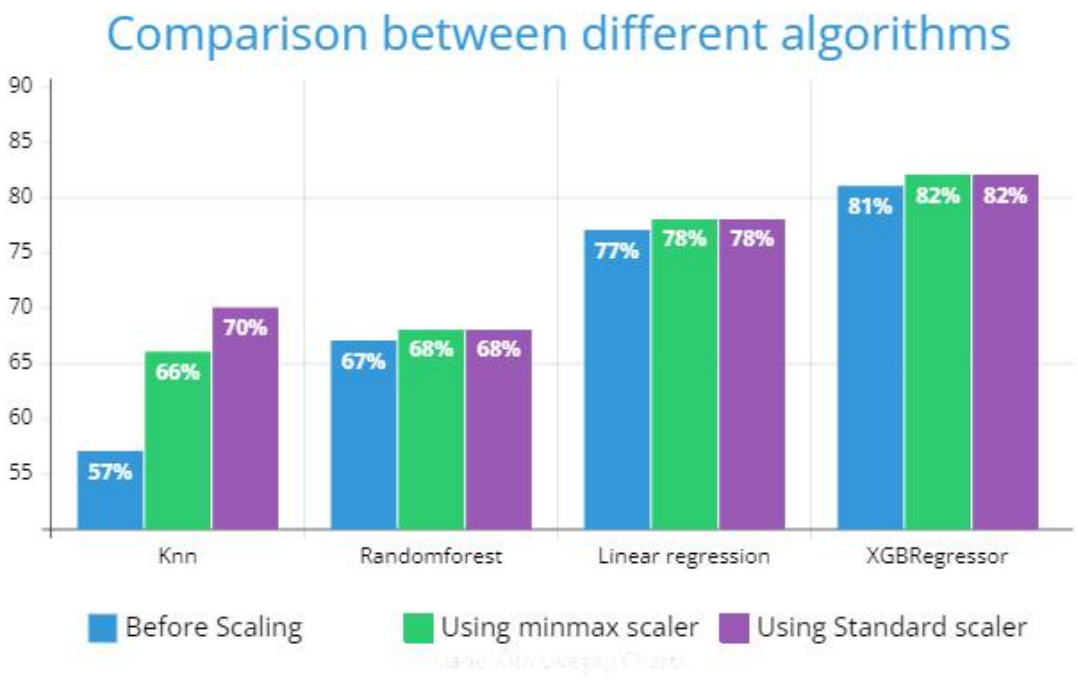
```
[62]
#train
neigh.fit(X_train_scaledd, ytrain)
regr.fit(X_train_scaledd, ytrain)
regrr.fit(X_train_scaledd, ytrain)
regressor.fit(X_train_scaledd, ytrain)

# scoring on the scaled test set
print("Scaled accuracy KNN: {:.2f}".format(
    neigh.score(X_test_scaledd, ytest)))
print("Scaled accuracy Randomforest: {:.2f}".format(
    regr.score(X_test_scaledd, ytest)))
print("Scaled accuracy Linier Regression: {:.2f}".format(
    regrr.score(X_test_scaledd, ytest)))
print("Scaled accuracy XGBregressor: {:.2f}".format(
    regressor.score(X_test_scaledd, ytest)))
```

```
Scaled accuracy KNN: 0.70
Scaled accuracy Randomforest: 0.68
Scaled accuracy Linier Regression: 0.78
Scaled accuracy XGBregressor: 0.82
```

Fig : Score after scaling

Comparison between different algorithms -



As you can see from the graph here KNN does the best after using standard scaler and overall XGBregressor gives us the best result using standard scaler so this is the best option for our current scenario.

9. Reference

[1] Anna Montoya, D. House Prices - Advanced Regression Techniques. (Kaggle,2016), https://kaggle.com/competitions/house-prices-advanced-regression-techniques

https://scikit-learn.org/stable/

https://seaborn.pydata.org/generated/seaborn.heatmap.html

https://colab.research.google.com/drive/1j5iqEKBlJsmEkIond_mPBazYljwT1zvv?usp=sharing#scrollTo=NuDczgoaonre

[2] “Machine Learning Random Forest Algorithm.” Javatpoint, https://www.javatpoint.com/machine-learning-random-forest-algorithm