House price Prediction using Machine Learning

Summary:

This report presents a comprehensive machine learning approach to predict house prices. The process begins with preparing the dataset for analysis, including cleaning the data, handling missing values, and partitioning the data into training and test sets. Next, we develop a regression model, evaluate its performance, and analyze the model coefficients to interpret the influence of different features on house prices.

Dataset for Implementation:

This database contains 13 attributes listed below:

1. Id
2. MSSubClass
3. MSZoning
4. LotArea
5. LotConfig
6. BldgType
7. OverallCond
8. YearBuilt
9. YearRemodAdd
10. Exterior1st
11. BsmtFinSF2
12. TotalBsmtSF
13. SalePrice

The dataset contains information about residential properties and their corresponding characteristics. Each observation is identified by a unique identifier in the "Id" column. The "MSSubClass" column describes the building class of the property, while "MSZoning" categorizes the general zoning classification.

Property size is represented by the "LotArea" column, and the configuration of the lot, such as "LotConfig," varies across observations. The "BldgType" column specifies the type of dwelling, and the overall condition of each property is rated in the "OverallCond" column. "YearBuilt" indicates the year the property was constructed, and "YearRemodAdd" denotes the year of the last remodel or addition. Exterior covering material is recorded in the "Exterior1st" column. Additionally, basement-related features include "BsmtFinSF2" for finished square feet in the basement type 2 and "TotalBsmtSF" for the total square footage of the basement area. Finally, the target variable "SalePrice" indicates the sale price of each property in dollars. This dataset provides a comprehensive set of features for predicting house prices.

Importing Libraries like:

1. Pandas *(for data manipulation)*
2. Matplotlib *(for data visualization)*
3. Seaborn *(for data visualization)*
4. SkLearn *(for data modeling)*

Data Preparation:

Initially, the dataset reads the file named "HousePricePrediction.xlsx" using pd.read_excel() and then It converts the DataFrame into a CSV file named "HousePricePredictioncsvfile.csv" using the to_csv() function. After that, Exploration of the dataset is conducted using the **info()** and **describe()** functions to understand its structure and summary statistics.

Data Preprocessing:

Converting the categorial data into binary data:

In order to prepare the house price feature for analysis, it needs to be converted into binary format. An instance of OrdinalEncoder is created with the parameter categories set to 'auto', which automatically determines the categories for encoding based on the unique values present in the data. The code applies the

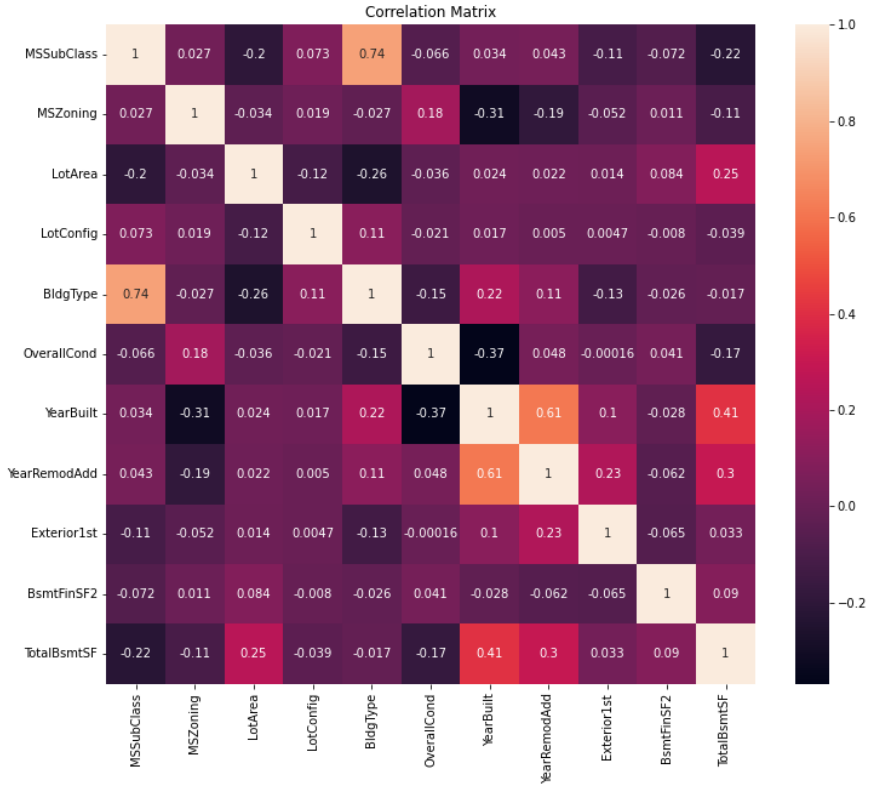
encoder to each categorical column individually: 'MSZoning', 'LotConfig', 'BldgType', and 'Exterior1st'. The fit_transform method is used to encode each column and replace its values with the corresponding encoded numerical values.

Identifying and handling missing values:

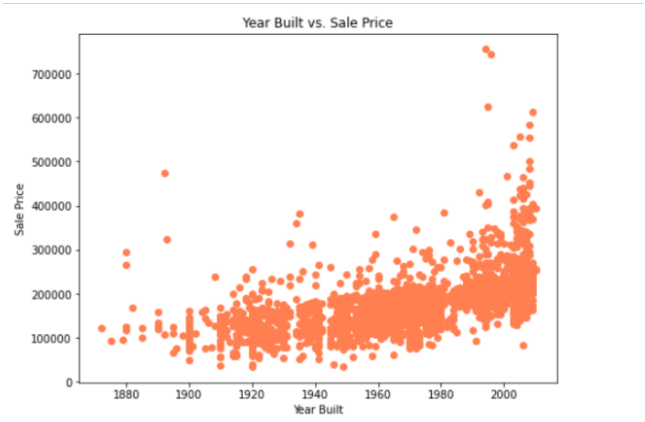
To find whether they are missing values in dataset or not we use the isnull() method to identify missing values in each column of the DataFrame. The sum() method is then applied to count the number of missing values in each column. They are 5 columns which consists of missing values. For instance, the 'MSZoning' column has 4 missing values, while 'Exterior1st', 'BsmtFinSF2', 'TotalBsmtSF', and 'SalePrice' columns have 1, 1, 1, and 1459 missing values, respectively. We can handle the 4 columns like 'MSZoning' ,'Exterior1st', 'BsmtFinSF2', 'TotalBsmtSF' as followed. Initially,it calculates the mode, which represents the most frequent value in the column, using .mode() .Subsequently, it employs the fillna() function to replace the missing values in columns with the calculated mode value, ensuring that every entry in the column has a non-null value. To confirm the success of the operation, the code then checks for any remaining null values in the column by counting occurrences of True (indicating null values) and False (indicating non-null values) using isnull().value_count(). After this , we predict the missing values column named 'SalePrice' and storing the dataset in a new data Frame 'new_df '.

Data visualization and plots:

We began the analysis by constructing a correlation matrix to visualize the pairwise correlations between numerical variables in the dataset. The correlation matrix provides a comprehensive overview of the linear relationships between variables, ranging from -1 (perfect negative correlation) to 1 (perfect positive correlation), with 0 indicating no linear correlation. We calculated the correlation matrix between features to identify highly correlated variables. This helped in understanding which features have the most significant impact on house prices.



The histogram provides valuable insights into the distribution of sale prices in the dataset, allowing for a better understanding of the range, central tendency, and presence of outliers. The scatter plot provides valuable insights into the relationship between the year a house was built and its corresponding sale price.



Splitting the Data:

We partitioned the dataset into a 80:20 ratio, allocating 80% for training and 20% for testing. The data is split into training and test sets using the train_test_split function with a random state of 42. This split ensures sufficient data for learning while retaining a substantial subset for an unbiased evaluation of the model's performance.

Model Development:

Linear Regression Model:

We chose a linear regression model for its interpretability and efficiency in binary classification tasks. The model was implemented using the scikit-learn library in Python, with regularization to prevent overfitting. Linear Regression model to predict house prices based on features like 'Exterior1st', 'TotalBsmtSF', 'BldgType', 'YearBuilt', and 'YearRemodAdd'. The model is then fitted to the training data, and predictions are made on the test set.

Model Evaluation:

Performance Metrics:

Performance metrics including Mean Squared Error (MSE) and R-squared (R2) are calculated both for the training and test sets.The model achieved an MSE of approximately 1,711,688,111 on the training set and 1,834,141,921 on the test set. Additionally, the R-squared value, which indicates the proportion of the variance in the dependent variable that is predictable from the independent variables, is approximately 0.59 for the training set and 0.61 for the test set.

Conclusion:

The linear regression model showed promising results in predicting house prices based on available features. However, further experimentation with different models and feature engineering techniques could potentially improve the predictive performance. Overall, this project provides a foundation for building more advanced machine learning models for house price prediction.