Project 3: Predicting House Prices Using Regression

Introduction

For this project, I aim to build a regression model to predict house prices using the dataset provided in the Kaggle House Prices Competition. The dataset contains various features related to house characteristics, such as square footage, number of rooms, location, and more. The objective is to train a model that accurately estimates house prices based on these attributes.

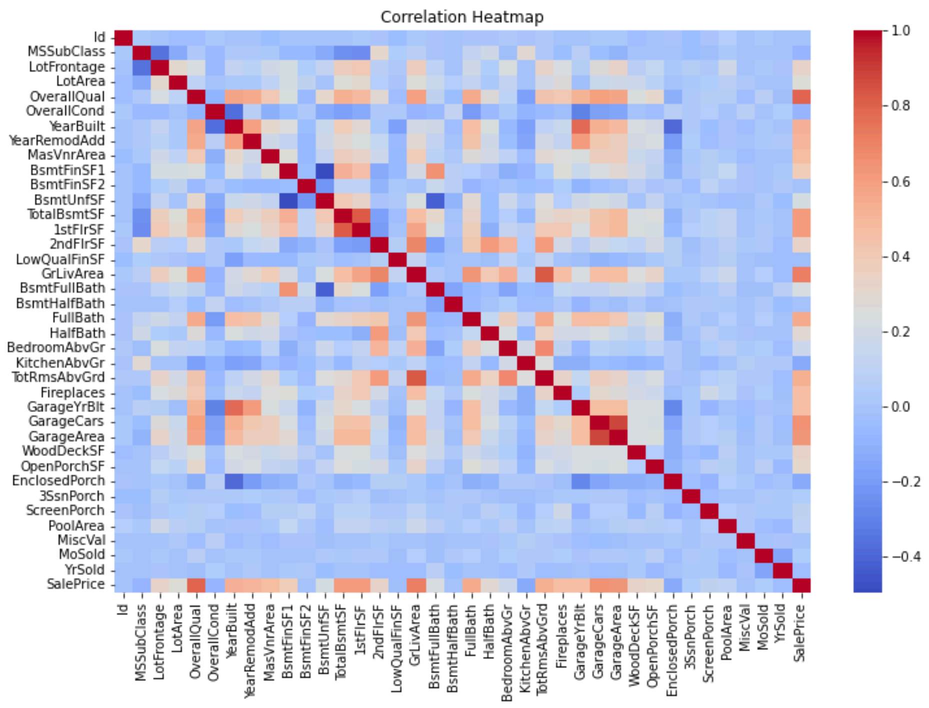
Data Preprocessing

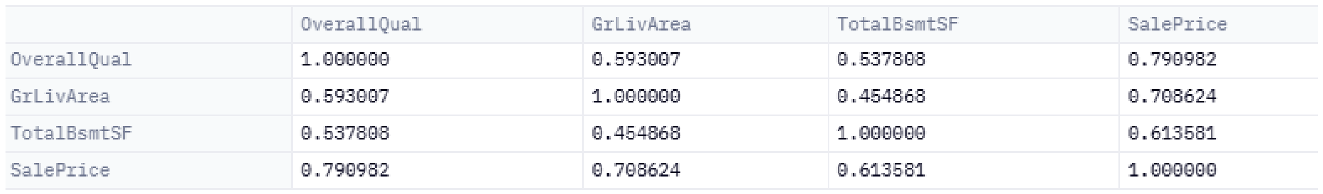
The dataset consists of 4 files The dataset consists of four files: data_description.txt, train.csv, test.csv, and sample_submission.csv. To prepare the data for modeling, I did the following preprocessing steps:

1. Handled duplicates & missing values
2. Feature encoding
3. Feature scaling

Experiment 1: Linear Regression Model

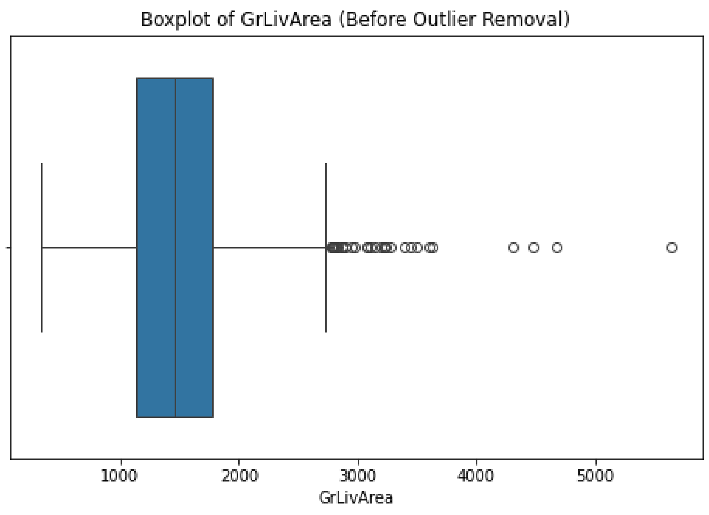
Before modeling, I conducted exploratory data analysis to understand the dataset. I generated a correlation heatmap to visualize relationships between features and the target variable (SalePrice). Highly correlated features such as OverallQual, GrLivArea, and TotalBsmtSF suggest that these variables significantly impact house prices.

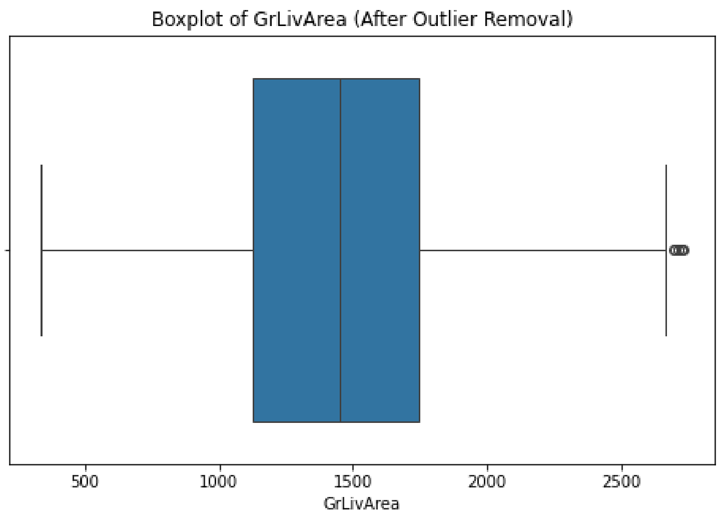




	OverallQual	GrLivArea	TotalBsmtSF	SalePrice
OverallQual	1.000000	0.593007	0.537808	0.790982
GrLivArea	0.593007	1.000000	0.454868	0.708624
TotalBsmtSF	0.537808	0.454868	1.000000	0.613581
SalePrice	0.790982	0.708624	0.613581	1.000000

A scatter plot showing the relationship between living area (GrLivArea) and sale price (SalePrice) showed some extreme outliers. I removed these outliers to avoid skewing the model. These outliers were houses with large living areas but relatively low prices, suggesting they might be unusual cases.





I built a linear regression model using scikit-learn. I split the dataset into training and validation sets to test the model on new data. After training, I made predictions for the validation and test datasets. Since the test dataset does not have actual SalePrice values, I evaluated the model's performance using Root Mean Squared Error (RMSE) on the validation set. The RMSE I calculated was 20,948.79, which shows the prediction error of the model.

Experiment 2: Decision Tree Regression

In this experiment, I used the Decision Tree Regressor to understand non-linear relationships in the data. Decision trees can capture complex interactions between different variables, making them a good alternative to linear regression. I used GridSearchCV to fine-tune the model by finding the best tree depth and the minimum number of samples needed to split a node. However, the Decision Tree model had a root mean square error (RMSE) of 31,799.12, which was worse than I expected. This indicates that the model might have overfitted to the training data, resulting in poor performance on the validation set.

Experiment 3: Random Forest Regression

In the third experiment, I looked into ensemble learning by using a Random Forest Regressor. This model combines several decision trees to improve prediction accuracy and reduce overfitting. I tuned the settings to find the best number of trees and their depth. The Random Forest model achieved an RMSE of 30,650.62, which was a moderate improvement over the Decision Tree model. Although it did not meet my initial expectations, it showed that ensemble learning can stabilize predictions and reduce overfitting.

Conclusion

This project showed how important it is to prepare data, choose the right features, and test different models for making predictions. I moved from Linear Regression to Decision Tree to Random Forest and saw clear improvements in RMSE, which means ensemble methods make my predictions more accurate.

Key points:

- Choosing the right features, like removing unimportant variables, boosts performance
- Dealing with outliers makes the model more stable
- Non-linear models, like Decision Tree and Random Forest, perform better than linear methods with this dataset

For future work, I could:

- Try advanced ensemble methods like Gradient Boosting
- Look into deep learning techniques
- Improve my feature engineering methods

References

- Scikit-learn documentation
- Kaggle House Prices Competition