Machine Learning Techniques to Predict Flat Prices

Name:	**Divyesh Krishna Lagisetty**
Registration No./Roll No.:	19178
Institute/University Name:	IISER Bhopal
Program/Stream:	PHY
Problem Release date:	August 17, 2023
Date of Submission:	November 19, 2023

1 Introduction

This project focuses on predicting flat prices in Indian cities using machine learning. The dataset encompasses 26,505 entries with 9 key factors, including construction status, regulatory approvals (RERA), room count (BHK_NO.), square footage (SQUARE_FT), and location details (ADDRESS, LONGITUDE, LATITUDE). Understanding these elements impact on prices is vital in the complex real estate landscape. The aim is to construct a robust predictive model leveraging this dataset. By employing advanced machine learning algorithms, the project aims to uncover correlations, patterns, and insights. Ultimately, this endeavor seeks to aid stakeholders in the real estate sector by providing a reliable tool for estimating flat prices. This report's findings aim to contribute valuable insights into housing market dynamics, empowering informed decision-making for both buyers and sellers by employing different feature selection processes and regression techniques.

2 Methods

Click this text for Github page of this project.

Preprocessing

The data has no missing values and followed the following steps to scale some features. Initially, isolated city names from the 'ADDRESS' column, potentially for regional analysis. Next, employing K-Means clustering on latitude and longitude coordinates, it categorizes flats into 20 clusters, likely to capture geographical proximity patterns. Performed a logarithmic transformation on the 'SQUARE_FT' feature, aiming to normalize its distribution for improved model performance. Subsequently, standardized several numerical features ('LOG_SQFT', 'SQUARE_FT', 'LATITUDE', 'LONGITUDE') using StandardScaler. This step ensures that all features share a mean of 0 and a standard deviation of 1, facilitating fair comparisons between different features and preventing disproportionate influence due to varying scales.

Overall, these preprocessing steps aim to enhance the dataset's readiness for machine learning models. By organizing geographical data, normalizing distributions, and standardizing feature scales, this sets the stage for models to better discern patterns and make accurate predictions based on the dataset.

The subsequent step involves **one-hot encoding** for 'BHK_NO' and 'ADDRESS' columns along with converting boolean values to their respective boolean representations (True/False). One-hot encoding transforms categorical variables into binary vectors, assigning a binary column for each category present in the original data. This process helps machine learning models interpret categorical data, preventing ordinal assumptions and potential bias.

Employing Regression Models

Linear Regression

Initialized a GridSearchCV to optimize a Linear Regression [1] model by exploring fit_intercept hyperparameters (True/False) using 10-fold cross-validation. It searches for the best-fit model based on the R-squared (r2) score and negative mean squared error (neg_mean_squared_error), indicating how well the model fits the data. Finally, it saves the model's predictions on test data as 'linear_regression_results.npy'. This process systematically determines the optimal model configuration, ensuring better predictions by evaluating different intercept settings for Linear Regression through 10 fold cross-validation.

Ridge Regression:

It explored various hyperparameters—alpha (regularization strength), max_iter (maximum number of iterations), and solver (optimization method) using GridSearchCV. With 10-fold cross-validation, it seeks the best model based on the negative mean squared error (scoring=neg_mean_squared_error) and R-squared (r2) score. This process rigorously evaluates different combinations of hyperparameters to determine the optimal settings for the Ridge Regression model. The 'grid_result' captures the best model configuration, enhancing prediction accuracy by fine-tuning regularization, iteration limits, and solving methods within Ridge Regression.

Decision Tree Regressor:

Similar to the above employed models explored various hyperparameters—max_depth (maximum depth of the tree), min_samples_split (minimum samples required to split a node), and min_samples_leaf (minimum samples required at a leaf node). Employing 10-fold cross-validation and assessing based on the R-squared score and negative mean squared error, it identifies the best model configuration. This allowed the selection of optimal hyperparameters for the Decision Tree Regressor[2]. This thorough search across different parameter combinations aims to improve the model's predictive capability by fine-tuning the tree structure.

Adaptive Boosting

Employed AdaBoost Regressor, an ensemble method combining multiple base estimators to enhance predictive performance. It utilized Linear Regression, Decision Tree Regressor, and Ridge Regression as potential base estimators, each with specific hyperparameter configurations obtained from the previous grid searches. The grid search for AdaBoostRegressor is conducted using 'scoring' parameter of 'r2' and negative mean squared error. This ensemble approach aims to leverage the strengths of diverse base estimators, utilizing their best hyperparameters to collectively improve the overall predictive capacity of the AdaBoostRegressor.

Random Forest Regressor

This a powerful ensemble learning algorithm. It explored a range of hyperparameters, including the number of trees, the maximum depth of trees, and the minimum number of samples required to split an internal node and to be a leaf node. Utilizing 10-fold cross-validation and evaluating performance based on the R-squared score and negative mean squared error, the grid search identified the optimal combination of hyperparameters for the RandomForestRegressor[3].

XGBoost Regressor

XGBoost combines gradient boosting with regularization techniques. Grid search involved n_estimators, max depth, learning rate, and booster.

Employing Regression Models with PCA on the Training Data

In this, Principal Component Analysis (PCA) was applied to reduce dimensionality on the one-hot-encoded columns, specifically 'BHK_NO' and 'ADDRESS' columns. PCA helps capture essential information while reducing the feature space's complexity. The reduced dimensions from PCA were then utilized as input for various above-mentioned regression models, namely Linear Regression, Decision Tree Regressor, Ridge Regression, AdaBoost Regressor, RandomForest Regressor, and XGBoost Regressor. GridSearchCV systematically explored similar hyperparameter combinations for each model, optimizing for both negative mean squared error and R-squared metrics. This dual evaluation ensures a comprehensive assessment of predictive accuracy and generalization performance. By employing PCA and grid search with multiple regression models, the approach aims to strike a balance between feature reduction and model optimization, ultimately enhancing the models' ability to make accurate predictions on the given dataset. Click this text for Github page of this project.

3 Experimental Analysis

R^2 and negative MSE are used to find the best hyperparameters for above-mentioned regression techniques. The coefficient of determination (R-squared, R^2) measures the proportion of variance in the dependent variable (target) explained by the independent variables in a regression model. Mathematically, it's calculated as $R^2 = 1 - \frac{SS_{\text{res}}}{SS_{\text{tot}}}$, where SS_{res} is the sum of squared residuals (errors) and SS_{tot} is the total sum of squares. R^2 ranges from 0 to 1, where 1 indicates a perfect fit, and 0 indicates that the model does not explain any variance.

Negative Mean Squared Error (Negative MSE) measures the average squared difference between predicted values and actual values, indicating the model's predictive accuracy. Mathematically, MSE is $\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$, where y_i represents actual values, $\hat{y}_i$ represents predicted values, and n is the number of samples. Negative MSE is used in GridSearchCV to maximize the performance metric, as minimizing MSE and maximizing negative MSE are equivalent. It's essential to note that a higher R^2 value and a lower negative MSE indicate better model performance.

Table 1: Performance of Different Regression Techniques Using All Features After One-hot Encoding

Regressor	R^2	MSE
Linear Regression	0.08	8.97
Ridge Regression	0.58	1.80
Decision Tree	0.85	0.16
Random Forest	0.90	0.15
Adaptive Boosting	0.88	0.13
XG Boost	0.89	0.11

Table 2: Performance of Different Regression Techniques After Applyoing **PCA** on the Data Set

Regressor	R^2	MSE
Linear Regression	0.17	17
Ridge Regression	0.48	3.4
Decision Tree	0.85	0.16
Random Forest	0.89	0.14
Adaptive Boosting	0.89	0.13
XG Boost	0.87	0.11

The outcomes in Table 1 offer a comprehensive overview of the training data analysis using various evaluation metrics like 'r^2' and 'MSE'. The evident observation from Table 1 is the subpar performance of linear regression and ridge regression models on our dataset. This inadequacy stems from our data, which exhibits non-linear dependencies among its factors concerning flat prices. Conversely, Decision

Tree and Random Forest regressors showcase remarkable accuracy in regression tasks based on our dataset. Particularly noteworthy is the Adaptive Boosting technique employing the best decision tree regressor as its base estimator, yielding superior results. Notably, XGBoost regressor, a widely acclaimed gradient boosting algorithm, achieves remarkable accuracy on our dataset, boasting an impressive MSE of 0.11. We also performed these above-mentioned regressions on the data which we have applied principle component analysis to reduce the dimensions of the data which also gave us the similar insights and results as earlier which is evident from table 2.

4 Conclusion & Discussion

Regression model evaluations showed linear regression and ridge regressor methods struggled, while Decision Tree, Random Forest, Adaboost and XGBoost regressors performed better, even post-PCA. We have explored and compared different regression models for predicting flat prices and gained valuable insights in addressing the complexities of real estate valuation and achieved precise predictions with diverse preprocessing and rigorous metric-based validation for Indian flat price forecasting. XG-

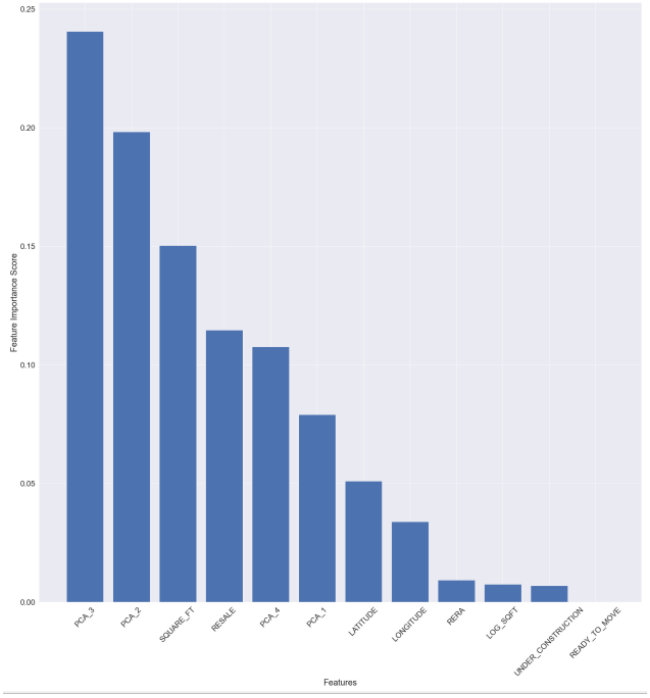

Figure 1: Feature importance scores per feature using the XGBoost regressor

Boost emerged as the top-performing model, and examining the feature importance scores reveals that the most critical features are those on which PCA was applied—specifically, BHK NO, ADDRESS, and GRID NUM following one-hot encoding. This analysis strongly indicates that factors related to the property's location and the number of rooms significantly influence the predicted price. Results could improve by exploring area impact on city prices, unexplored due to computational complexity constraints.

References

[1] Robert Tibshirani Jerome H. Friedman and Trevor Hastie. *The Elements of Statistical Learning*. Springer, second edition, 2008.

[2] R. A. Olshen L. Breiman, J. H. Friedman and C. J. Stone. Classification and regression trees. metrika. *Information*, 33:128–128, 1986.

[3] Leo Breiman. Random forests. machine learning. *Information*, 45(1):5–32, 2001.