CS 584 Machine Learning

Project Report

On

Boston Housing Prices

By

Name of Student	**CWID**
Mohammed Abrar Ahmed	A20540340
Mohammad Anas Hussain	A20548229

Under The

Supervision of

Prof. Stephen Avsec

Illinois Institute of Technology

Chicago, Illinois

April 2024

CONTENT:

I. Overview:

In this project, we analyzed the housing prices in Boston by working with the Boston Housing Dataset from Kaggle [1]. Our objective was to accurately predict the median house value based on the provided input features. We successfully addressed missing data, mitigated skewness in feature distributions, and conducted correlation analyses to guide feature selection judiciously. Dimensionality reduction strategies were employed to gain a deeper understanding of the dataset.

The model training phase involved a training linear regression model, whose performance was assessed based on the MSE, RMSE and R-squared scores. Then we analyzed the performance of our linear regression model by comparing it with other regressor models like the Decision Tree Regressor and the Random Forest Regressor.

The Decision Tree Regressor outperformed the linear regression model, but it wasn't the best regressor model since it faced overfitting challenges. We optimized its performance through meticulous grid search, feature selection, and hyperparameter tuning. Despite achieving a simpler yet effective decision tree model, we were able to recognize the dataset's limitations, including potential biases and a relatively small size leading to an overfitting challenge.

This overfitting was mitigated by the random forest regressor, where the default model exhibited highly accurate predictions. Through an iterative process of evaluation and improvement, including cross-validation and feature analysis, the random forest regressor proved itself as a robust predictive model. However, we remained cognizant of the significant dataset limitations, underscoring the importance of critical evaluation for ethical and representational concerns in model development.

II. Data Description:

The Boston Housing Dataset includes 13 features and 1 target variable, i.e, 14 attributes.

1. CRIM - per capita crime rate by town
2. ZN - proportion of residential land zoned for lots over 25,000 sq.ft.
3. INDUS - proportion of non-retail business acres per town.
4. CHAS - Charles River dummy variable (1 if tract bounds river; 0 otherwise)
5. NOX - nitric oxides concentration (parts per 10 million)
6. RM - average number of rooms per dwelling
7. AGE - proportion of owner-occupied units built prior to 1940
8. DIS - weighted distances to five Boston employment centers
9. RAD - index of accessibility to radial highways
10. TAX - full-value property-tax rate per $10,000

11. PTRATIO - pupil-teacher ratio by town
12. B - 1000(Bk - 0.63)^2 where Bk is the proportion of blacks by town
13. LSTAT - % lower status of the population
14. MEDV - Median value of owner-occupied homes in $1000's

III. Data Preprocessing:

Handling Missing Values

The initial step in data preprocessing involved a meticulous examination of missing values within the dataset. The 'rm' column had 5 missing values, we addressed this by replacing the missing values with the median of the 'rm' column.

Exploring Feature Distribution

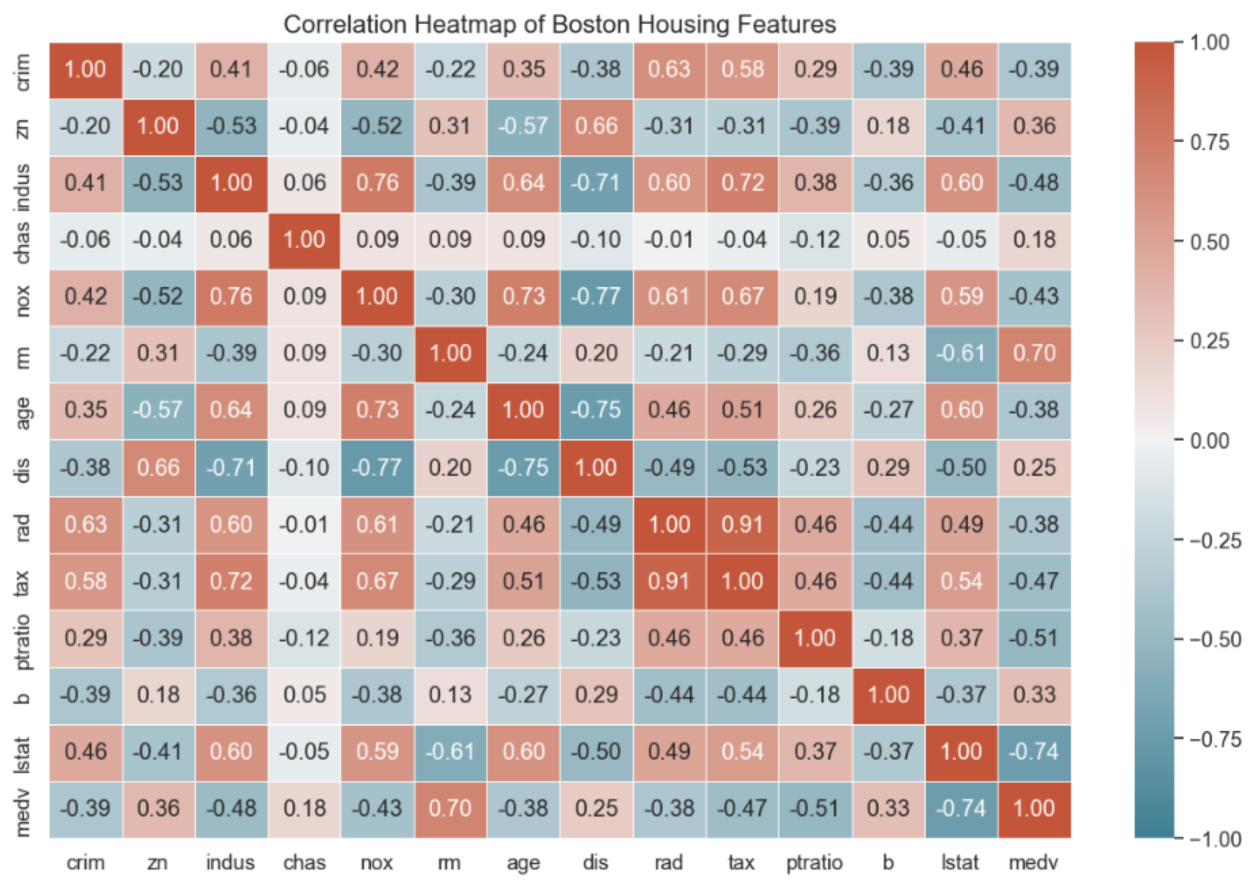
We performed EDA to understand the distribution of each feature and the target variable. Some features, such as 'crim,' 'zn,' and 'b,' exhibited high skewness. The 'chas' column was identified as categorical.

Visualization techniques, including histograms, were employed to analyze feature distributions. This exploration provided valuable information on the spread, central tendency, and potential outliers in each feature.

Correlation Analysis and Feature Selection

In exploring feature relationships, the 'Chas' feature showed limited correlation with others, indicating weak linear association. Hence, we decided to drop 'Chas' during preprocessing. Conversely, a significant correlation between 'Tax' and 'Rad' raised concerns about multicollinearity, prompting us to drop the 'Rad' feature.
The considered removal of 'Chas' and 'Rad' played a crucial role in refining the dataset for predictive modeling, emphasizing the importance of thoughtful feature correlation analysis during preprocessing.

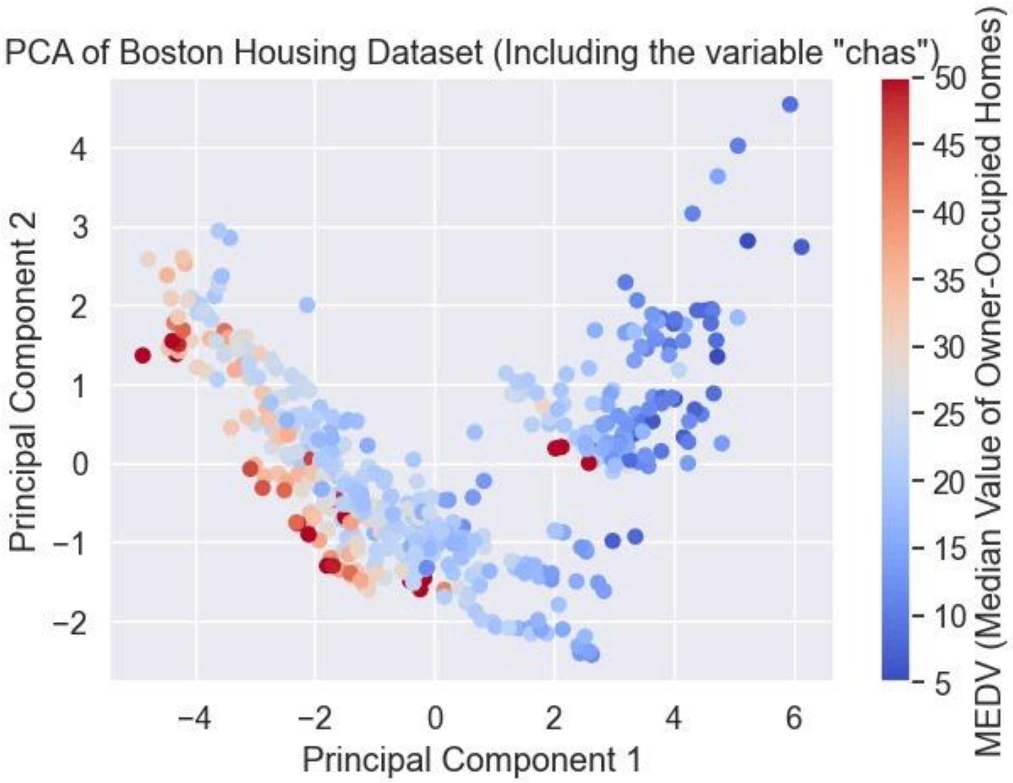
The correlation analysis heatmap below highlights the importance of thoughtful feature consideration in developing accurate and robust predictive models.

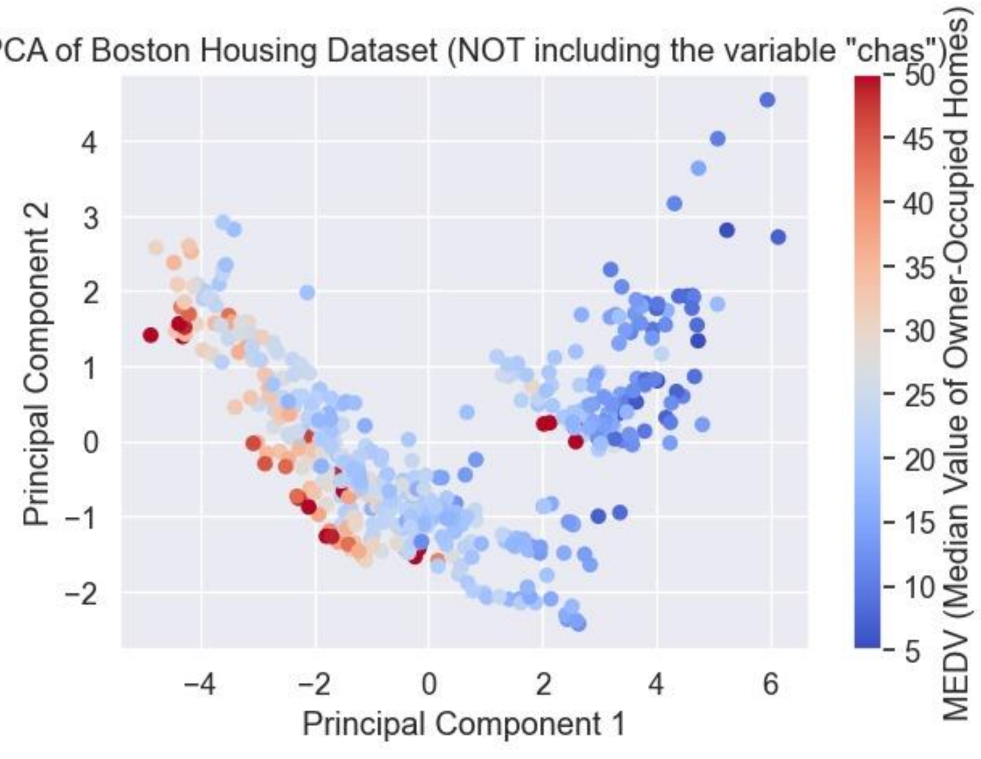


Dimensionality Reduction

Given the high dimensionality, Principal Component Analysis [2] was applied for dimensionality reduction. The impact of reduction, evaluated with and without the 'chas' feature, aimed to enhance model efficiency and interpretability. This visualization technique provided a comprehensive understanding of the dataset's characteristics, guiding preprocessing and improving overall data quality for model training.

It became clear that the first component (PCA) effectively grouped more expensive houses on the left and cheaper houses on the right of the x-axis. While interpretability decreases after dimensionality reduction, there is a promising indication that certain features play a crucial role in determining house prices.





The plots show almost no difference before and after dropping the "chas" variable.

Cross-Validation Strategy - Permutation of k-Fold:

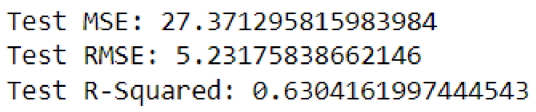
For model evaluation and training, a robust cross-validation strategy is crucial. The permutation of k-fold cross-validation was chosen for this project. The decision to use k-fold cross-validation was driven by the regression nature of the task, the absence of stratification needs, and the close-to-normal distribution of the target feature, 'medv.' This approach aimed to provide a comprehensive evaluation of the model across different subsets of the dataset.

IV. Model Training:

The model training phase involved implementing a linear regression model from scratch by using Numpy and then compared with other regression models (Decision Tree and Random Forest). To ensure an ample number of observations for model training, the dataset was split into a training set (80% of the data) and a testing set (20%). This larger split was deemed necessary, given the complexity introduced by the multitude of features in the dataset as well as a relatively small number of observations.

Linear Regression Model Scores

The linear regression models scores were calculated for the test set, this served as the initial baseline to compare with other regression models aiming to predict housing prices in Boston. The coefficient of determination score on the test set suggested that the current feature set lacked the predictive power needed for accurate housing price predictions.



```
Test MSE: 27.371295815983984
Test RMSE: 5.23175838662146
Test R-Squared: 0.6304161997444543
```

The figure illustrates the test-set results of the linear regression model

Feature Size Concerns

The main concern at this juncture was the size and relevance of the features. The suboptimal performance of the linear model prompted a realization that a more sophisticated model was necessary to enhance predictive accuracy.

Comparing Models: Linear Regression, Decision Tree Regressor

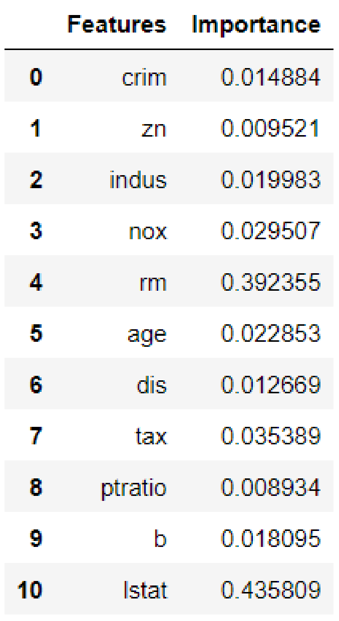
The results of the linear regression model when compared with a decision tree regressor. The linear regression model exhibited poor performance. The decision tree regressor surpassed the linear model, achieving higher R-squared scores on both training and test sets [3].

The decision tree model was then also subject to a k=10 k-fold cross-validation. These results were shocking as the cross-validation MSE scores were worse than the linear model when it was subject to cross-validation.

Grid Search CV, Feature Selection, Hyper-parameter Tuning

Numerous steps were taken after the initial simple linear and decision tree model training to improve model performance. The first was to implement a grid search cross validation to find the best hyper-parameters for the decision tree model in order to decrease the complexity. As found in the analysis, the default decision tree parameter was very deep, with a depth of 19, and wide, with 383 leaf nodes. This was why on cross-validation, the model performed poorly since there was an obvious over-fitting to training data. Hyper-parameter selection through grid search led to a much smaller tree with depth of 7 and 29 leaf nodes. This smaller model, however, did not perform on the test set as well as the default parameter tree, yet we see that a much smaller tree still only resulted in a small drop in performance.

Next, it was determined to analyze the feature importance of the decision tree, and upon the findings two features were heavily favored, “rm” and “lstat”. These were important to the model, so they were kept and instead the worst feature was dropped, “ptratio”. The default



	Features	Importance
0	crim	0.014884
1	zn	0.009521
2	indus	0.019983
3	nox	0.029507
4	rm	0.392355
5	age	0.022853
6	dis	0.012669
7	tax	0.035389
8	ptratio	0.008934
9	b	0.018095
10	lstat	0.435809

decision tree with the dropped feature, performed better than the default tree with previous features on the test set. Now to decrease complexity another grid search was performed, and while a less complex tree was found, its performance drop was extreme and was deemed to be a model not worth the trade-off between performance and complexity.

Finally, the random forest regressor was implemented and it proved to solve the problem of over-fitting because ensemble methods increase accuracy by combining prediction power of many uncorrelated models. The test R-squared was .969 and the test MSE was 2.247. There was an attempt to prune the model which resulted in a model that performed better than the decision tree with feature selection, but still not as good as the default random forest model.

V. Results:

The Linear Regression model's assumption that features are scaled and distributed evenly was its greatest downfall. This was not the case in our dataset, and it makes sense as to why the linear model performed as poorly as it did. The pitfall for decision tree models can be that they severely overfit training data. This is why it was apparent that during cross-validation, most of the folds had a worse MSE compared to the cross-validation MSE of the linear model. For this reason, an ensemble method was chosen to strike a balance between performance and complexity However, the decision tree model, after feature selection (excluding 'ptratio'), showed improved performance with a higher R-squared score on the test set and reduced complexity. The grid search, however, provided a simpler model with slightly worse metrics, emphasizing the trade-off between model complexity and performance. The random forest regressor, with default parameters, was by far the best model. The cross-validation scores looked low and more balanced than for previous models.

References:

[1]. https://www.kaggle.com/datasets/arunjangir245/boston-housing-dataset

[2]. https://jamesmccaffrey.wordpress.com/2021/07/16/computing-pca-using-numpy-without-scikit/

[3]. https://medium.com/@vk.viswa/unveiling-decision-tree-regression-exploring-its-pr inciples-implementation-beb882d756c6