Boston Housing Price Prediction using Machine Learning

Muhammad Salman

August 26, 2025

1 Introduction

This project applies machine learning models to predict house prices from the **Boston Housing Dataset**. We compare *Linear Regression*, *Ridge Regression*, and *Random Forest Regressor*. We also evaluate performance using metrics such as **MAE, RMSE, $\mathbf{R^2}$**, and a **custom accuracy metric**. Additionally, the trained Random Forest model is saved and reloaded for future predictions.

2 Dataset

The dataset is loaded from the following URL:

`https://raw.githubusercontent.com/selva86/datasets/master/BostonHousing.csv`

2.1 Dataset Overview

- Rows: 506
- Columns: 14 (13 features + target)
- Target variable: `medv` (Median value of owner-occupied homes in $1000s).

2.2 Target Distribution

Figure 1 shows the distribution of the target variable `medv`.

3 Exploratory Data Analysis (EDA)

Correlation analysis was performed. The top 10 most correlated features with the target include:

- LSTAT (percentage of lower status population)

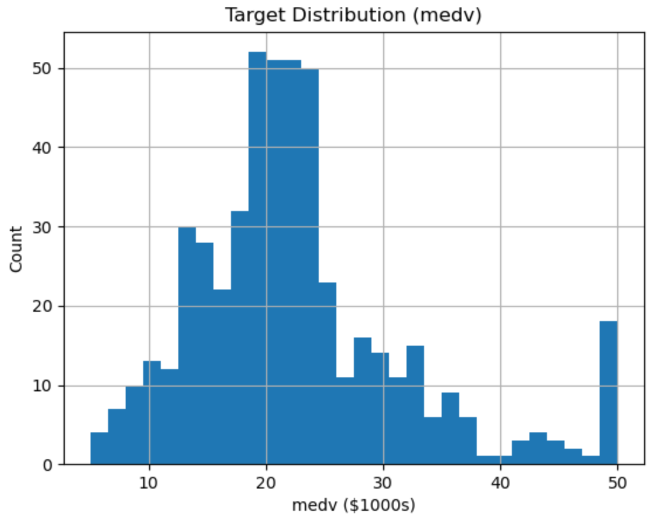

Figure 1: Distribution of Target Variable (`medv`)

- RM (average number of rooms per dwelling)
- PTRATIO (pupil-teacher ratio by town)

Figure 2 shows the correlation matrix heatmap.

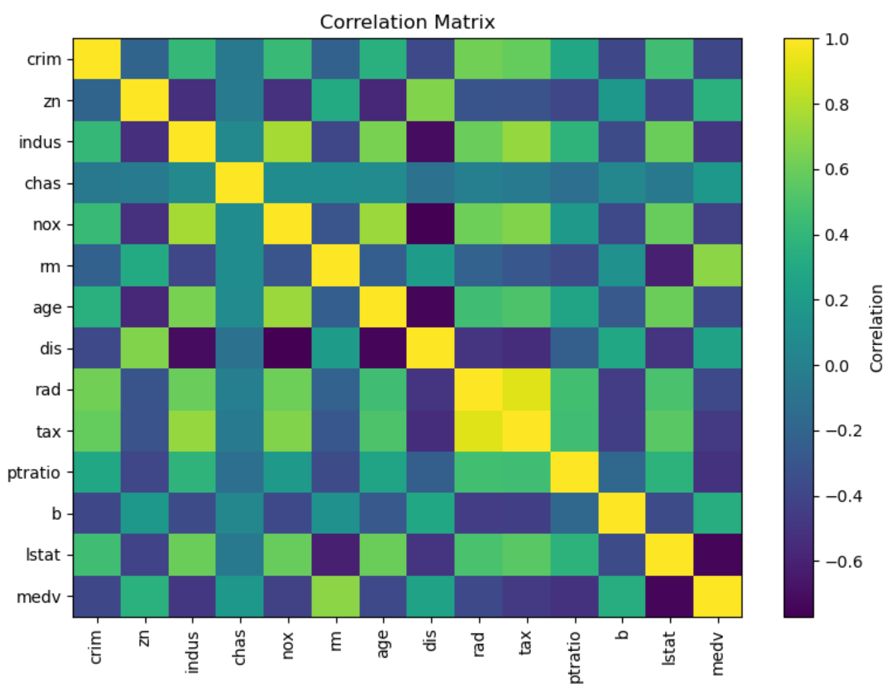

Figure 2: Correlation Matrix of Features

4 Models and Training

We split the dataset into 80% training and 20% testing sets.

4.1 Linear Regression

Achieved performance:

```
MAE  = 3.16
RMSE = 4.98
R²   = 0.711
```

4.2 Ridge Regression

Achieved performance:

```
MAE  = 3.13
RMSE = 4.95
R²   = 0.715
```

4.3 Random Forest Regressor

Achieved performance:

```
MAE  = 2.06
RMSE = 3.20
R²   = 0.876
```

Random Forest outperformed the linear models. Figure 3 shows the top 10 important features.

5 Model Evaluation

5.1 Predicted vs Actual

Figure 4 shows predicted vs actual prices for the Random Forest model.

5.2 Custom Accuracy

Since regression problems do not have accuracy by default, we define a custom accuracy metric. A prediction is considered correct if it lies within $\pm 10\%$ of the true value:

$$\text{Accuracy} = \frac{\text{Number of accurate predictions}}{\text{Total predictions}}$$

Custom Accuracy obtained:

```
Custom Accuracy (±10% tolerance): 82.45%
```

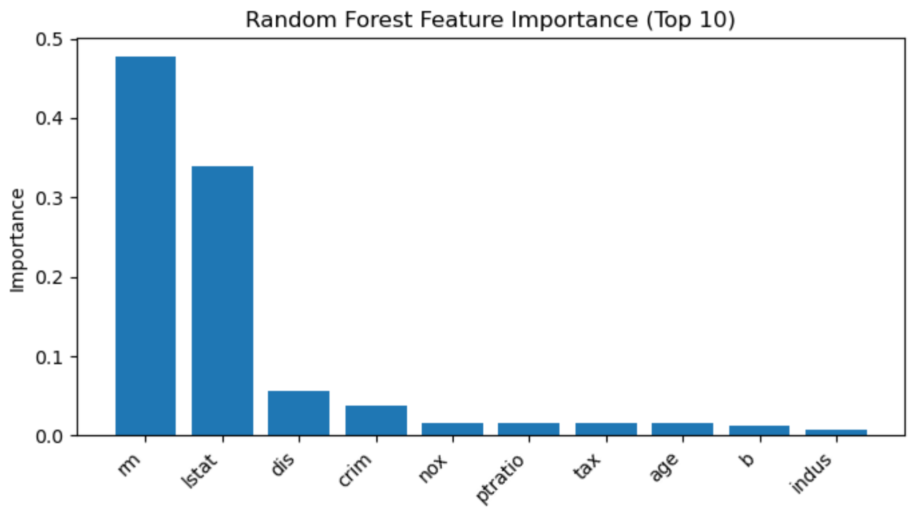

Figure 3: Random Forest Feature Importances (Top 10)

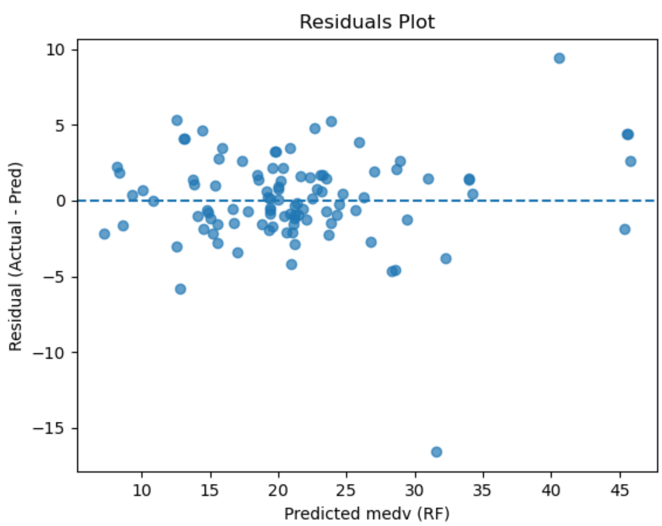

Figure 4: Residuals Plot (Random Forest)

6 Model Persistence

The trained Random Forest model is saved using `joblib`:

```
joblib.dump(rf, "house_price_rf_model.joblib")
```

It is later reloaded and used for predictions:

```
loaded_model = joblib.load("house_price_rf_model.joblib")
preds = loaded_model.predict(X_test)
```

7 Conclusion

- Random Forest performed best with lowest error and highest R^2.
- Feature importance revealed that `LSTAT` and `RM` are the most influential features.
- Custom accuracy provides an interpretable measure for regression tasks.
- The model was successfully saved and reloaded for deployment.