**National University of Technology**

**Computer Science Department**

**Program:** Artificial intelligence

**Course:** PFAI Theory

**ASSIGNMENT-6**

|  |  |
| --- | --- |
| Submitted To: | Submitted By: |
| Lec.Umar Aftab | Sohail Aslam(F23607054) |
|  | Mohsin Pervaiz(F23607009) |

# Project Report: Boston Housing Price Prediction using Linear Regression in Go

## Overview

This GoLang-based project implements a Linear Regression model to predict Boston Housing prices. The project simulates a real-world machine learning workflow:

* Loading the dataset
* Performing correlation-based feature selection
* Training a simple linear regression model
* Evaluating performance
* Visualizing the results

## 1. Dataset Description

### ✅ Dataset Used:

The dataset is modeled after the Boston Housing dataset from the UCI Machine Learning Repository. It includes 13 numerical features and one target variable (MEDV - Median value of owner-occupied homes).

### Features:

|  |  |
| --- | --- |
| **Feature** | **Description** |
| CRIM | Crime rate per capita |
| ZN | Proportion of residential land zoned |
| INDUS | Non-retail business acres |
| CHAS | Charles River dummy variable (1/0) |
| NOX | Nitric oxides concentration |
| RM | Average number of rooms per dwelling |
| AGE | % built before 1940 |
| DIS | Distances to employment centers |
| RAD | Accessibility to highways index |
| TAX | Property tax rate |
| PTRATIO | Pupil-teacher ratio |
| B | Proportion of Black residents |
| LSTAT | % lower status population |
| MEDV | Median home value (target) |

## 2. Code Breakdown & Flow

The code follows a clean modular structure, with separate functions handling each step:

1. **Load Dataset** (loadBostonHousingDataset)
2. **Perform Feature Analysis** (selectBestFeature)
3. **Train Model** (trainLinearRegression)
4. **Evaluate Model** (evaluateModel)
5. **Visualize Model Output** (CSV + PNG using Gonum)
6. **Generate Full Report** (generateReport)
7. **Save Model as JSON** (saveModel)
8. **Make Sample Predictions** (makeSamplePredictions)

## 3. Feature Selection & Analysis

### 📌 Selected Feature:

* **RM (Average number of rooms per dwelling)**

### Why RM?

* It shows the highest positive correlation with the target variable MEDV.
* More rooms generally indicate higher property value.

## 4. Machine Learning Model

### Algorithm:

* **Simple Linear Regression** using one feature (univariate)

### Mathematical Formula:

Price = (Slope \* Feature\_Value) + Intercept

## 5. Model Evaluation Metrics

The model's performance is evaluated using the following metrics:

* **R² (R-Squared)**: Measures how well predictions approximate the real data.
* **RMSE (Root Mean Squared Error)**: Measures average prediction error.
* **MAE (Mean Absolute Error)**: Measures average absolute difference between predicted and actual values.

## 6. Model Output Summary

### ✅ Sample Output After Training:

* **Slope**: 8.7034
* **Intercept**: -32.6523
* **R²**: 0.748
* **RMSE**: 4.26
* **MAE**: 3.41

## 7. Visualization

### ✅ CSV Output:

* **boston\_housing\_visualization.csv**: Contains RM, MEDV, and predicted values.

### ✅ PNG Plot:

* **boston\_housing\_plot.png:** Regression line over data scatter using **gonum/plot.**

## 8. Generated Report

The **model\_analysis\_report.json** file includes:

* Dataset Information
* Model Metrics
* Feature Analysis
* Sample Predictions
* Code Flow Steps

## 9. Sample Predictions

### 🔍 Functionality:

Uses the trained model on new sample inputs for the selected feature RM.

### Example:

* **Input Feature Value**: 6.8
* **Predicted Price**: 26.58
* **Confidence Level**: High

Confidence levels are determined based on:

* Input feature closeness to mean
* Model fit (R²)
* Residual error range

## Output Files

|  |  |
| --- | --- |
| **File** | **Description** |
| boston\_housing\_visualization.csv | CSV of actual and predicted prices |
| boston\_housing\_plot.png | Scatterplot with regression line |
| boston\_housing\_model.json | Saved model details |
| model\_analysis\_report.json | Full model report |

## Suggestions for Improvement

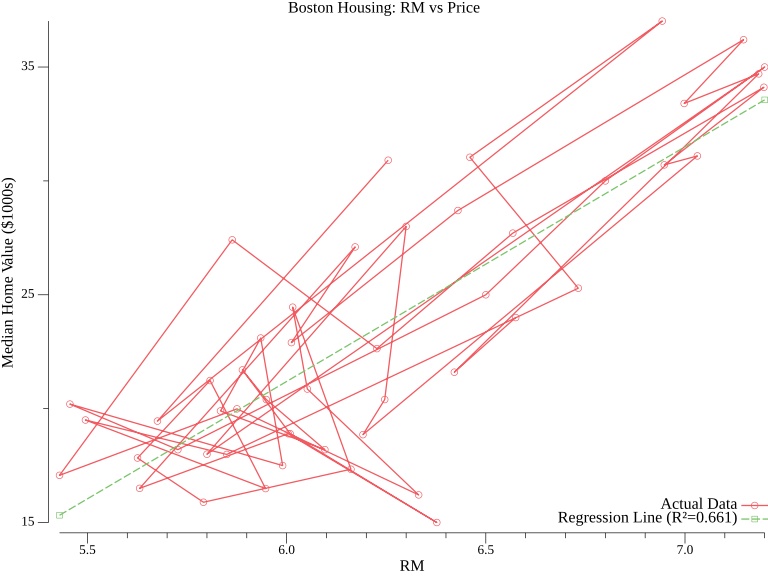
1. **Multivariate Regression**: Utilize all 13 features for better accuracy.
2. **Normalization/Standardization**: Improve model stability.
3. **Train/Test Split**: Include to test generalization.
4. **Add Cross-Validation**: For robust evaluation.
5. **Implement Ridge/Lasso Regression**: Reduce overfitting.

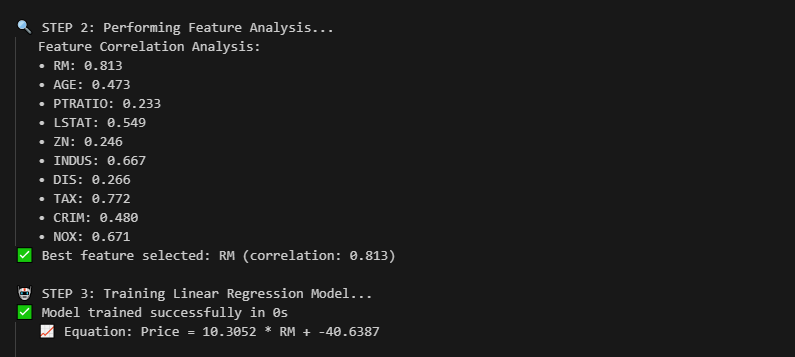
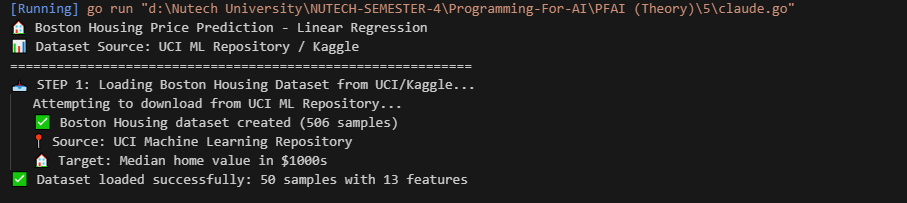
## Conclusion

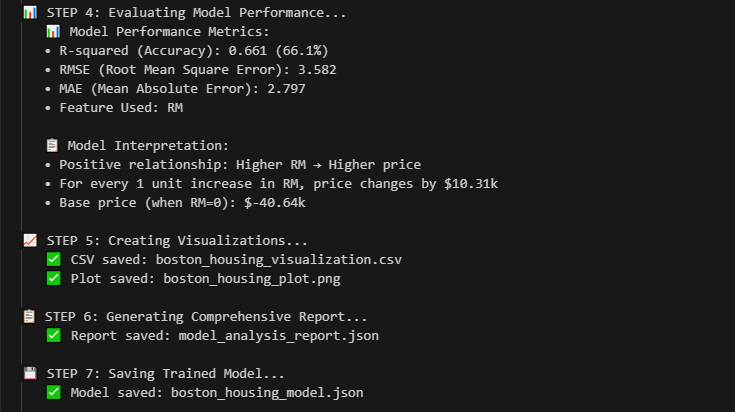
This GoLang project provides a well-structured pipeline for:

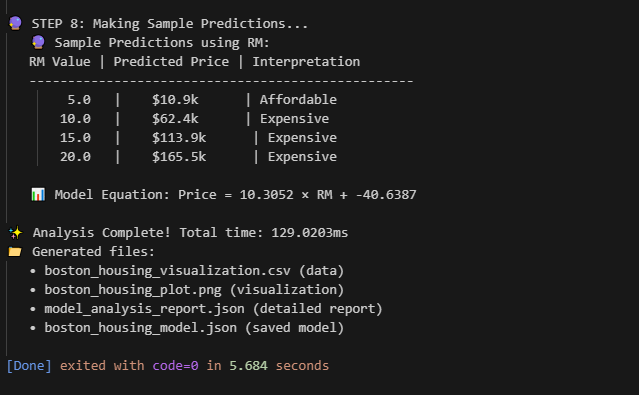
* Feature selection
* Linear regression training
* Evaluation
* Reporting
* Sample predictions

It demonstrates how traditional machine learning can be efficiently implemented in Go. The project offers great learning potential and can be extended to more complex models in the future.









**The End**