Comparison of Ridge Regression With Random Forest Algorithm When Solving Boston Housing Problem

By: Fabio Eugenio dos Santos de Sampaio Doria

Introduction

Aims

- Implement Ridge Regression Machine learning method.
- Run and analyse performance on Boston Housing Problem Dataset.
- Repeat same process for Random Forest algorithm and compare the results.

Objectives

- Create program using Object-Oriented Design.
- Full implementation life cycle.
- Display data visualisation when using different parameters and Kernels.

Context – Why the issue matters

Significance of Boston Housing Dataset

- **Individual Impact** – Finding fair prices of houses for families
- **Large-scale Impact (Economy)** – House prices linked to consumer spending

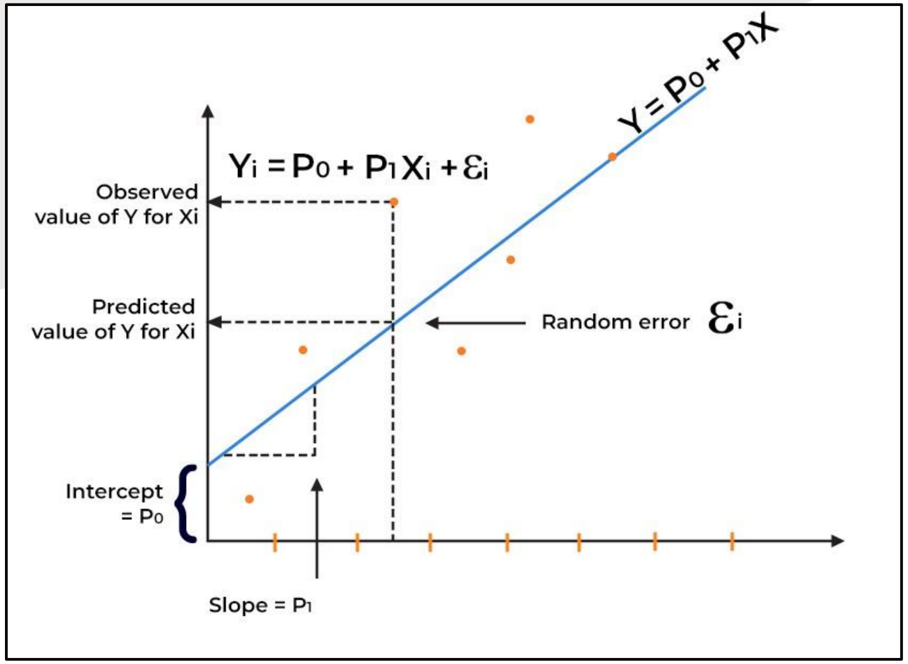
Labels

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	LSTAT	MEDV
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1	296	15.3	4.98	24.0
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2	242	17.8	9.14	21.6
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2	242	17.8	4.03	34.7
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3	222	18.7	2.94	33.4
5	0.02985	0.0	2.18	0.0	0.458	6.430	58.7	6.0622	3	222	18.7	5.21	28.7

Significance of Comparing Algorithms

- **Performance** – Can impact accuracy and speed of predictions
- **Generalisation** – Under and over fitting issues can occur

Main Concepts



<u>Regression in Machine Learning</u>

- **Supervised learning** – Provide labels to new unlabelled samples
    - Learn from a set of labelled samples, i.e., houses z
- Occurs when prediction set is infinite (house prices = Real Numbers)
- Sum of squares error equation is used to find line of best fit through data

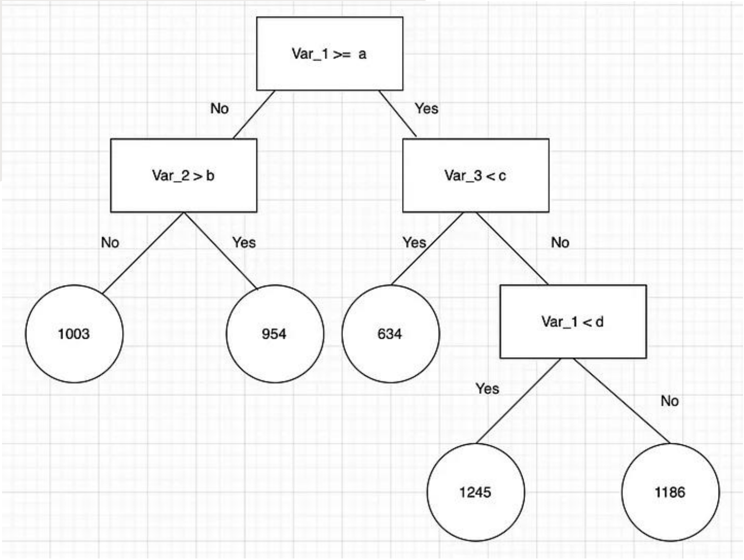
$$SSE = \sum_{i=1}^{n} (\hat{y}_i - \hat{y}_i)^2$$

<u>Ridge Regression</u>

- Improvement over regular regression algorithm, specifically overfitting
- Adds a 'penalty term' to the end of SSE equation
- Regularises the model by adding bias which decreases accuracy but improves overall performance

$$SSE_{L2} = \sum_{i=1}^{n} (\hat{y}_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{P} \beta_j^2$$

Main Concepts



Random Forest

- Breaks dataset down into smaller subsets.
  - At the same time deploys decision trees.
- Multiple models are trained, each using random subsets of the dataset.
- Average result of all models are average to find predicted value.

$$SSE = \sum_{i=1}^{n} (\hat{y}_i - \hat{y}_i)^2$$

Benefits

- Resilient to overfitting due to large number of random subsets.
- Multiple trees can be trained in parallel, computationally more efficient.
- Easy to identify important features as they are

$$SSE_{L2} = \sum_{i=1}^{n} (\hat{y}_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{P} \beta_j^2$$

My approach – Progress so far

- Preparation
  - In-depth research on Ridge Regression and Random Forest algorithms
  - Gather information from both articles and research papers
  - Found datasets with different number of features for testing

- Code
  - Completed Ridge Regression model implementation into program
  - Started implementation of Random Forest algorithm
  - Using Test Driven Development (TDD) to follow modern software engineering principles

Conclusion – Further Steps

- Code
  - Fully implement Random Forest algorithm
  - Implement a proper graphical user interface (GUI)
  - Add automatic tests that examine comparison of different kernels and parameters

- Write-up
  - Reflect and discuss findings, compare both results and come to a conclusion on both algorithms
  - Detail specific issues with implementation, such as data structures and numerical methods needed

Summary

Main Goal

- Define the difference in performance between Ridge Regression and Random Forest algorithm.

How to get there

- Create program implementing both algorithms and run them on Boston Housing problem Dataset
  - Do so in a proper manner, i.e., proper software principles followed, full OOP design, user graphical interface.
- Program shows complete breakdown of results.
  - Data visualisation.
  - Shows results when using different parameters etc, for optimisation.