CS 451/551 - Assignment 2

April 23, 2024

1 Introduction

In this assignment, you are expected to solve a *Linear Regression* problem with a genetic algorithm. Five different datasets are provided, and you should implement a genetic algorithm approach to find a promising solution for each dataset. Also, you will analyze the effect of the algorithm's parameters on the performance in terms of the objective function and computational time. The deadline for this assignment is **May 12 2024**.

2 Problem Description

For this assignment, we provide five different datasets consisting of two independent features (x_1, x_2) and one dependent feature (y). You are expected to construct a linear function to predict the target value, as seen in Equation (1). Your ultimate goal is to minimize the objective value, which is the mean squared error between the actual and estimated target values, as seen in Equation (2).

$$f(x_1, x_2) = a \times x_1 + b \times x_2 + c = \hat{y} \tag{1}$$

$$z = \textbf{min}\ \frac{1}{N}\sum_{i=1}^{N}(y - \hat{y})^2 \tag{2}$$

To construct the linear function, you need to carefully determine the parameters (e.g., a, b, and c) to minimize the estimation error. For this purpose, you are expected to implement a genetic algorithm to optimize the parameters of the linear function for each given dataset.

Moreover, the genetic algorithm has several hyper-parameters impacting the algorithm's performance in terms of runtime and outcome. Therefore, you need to analyze the effect of each parameter on the algorithm's performance and then specify the most promising parameters. This process is called "Parameter Tunning".

3 Genetic Algorithm

Genetic Algorithm is a meta-heuristic approach inspired by the theory of evolution. Its population-based search strategy may provide a promising solution for an optimization or machine-learning problem. However, it does not guarantee the finding of a global optimum. A solution for the given problem should be represented as a set of genes (i.e., chromosomes) to apply the Genetic Algorithm. Besides, a fitness function should be defined to determine how good a chromosome is for the task. The Algorithm starts with a population of potential chromosomes. Iteratively, it applies a series of genetic operators to the population to create new generations of solutions. Each iteration tries to find fitter chromosomes by eliminating some chromosomes (i.e., replacement) and recombining the existing ones (i.e., reproduction). Figure 1 displays an example process of a Genetic Algorithm.

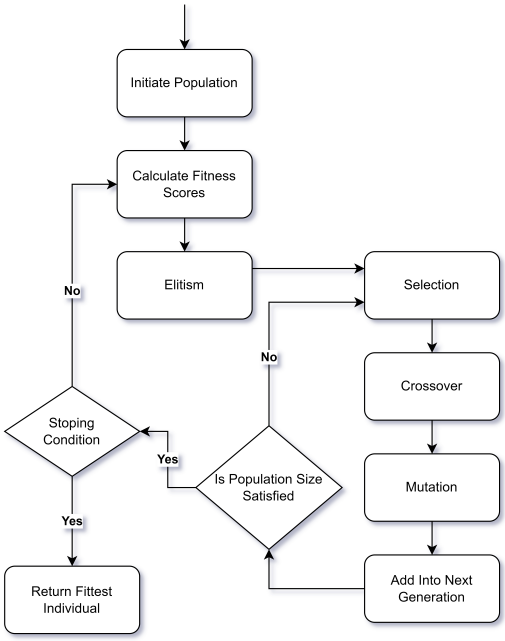

Figure 1: An Example Flowchart for Genetic Algorithm

As seen in the figure, Genetic Algorithm can obtain several components as listed below:

- **Initiate Population**: Generate an initial population of potential solutions randomly or based on prior knowledge. Each possible solution is represented as a chromosome, usually a binary string, a real-valued vector, a permutation, or any other data structure that can be manipulated. Note that the size of the population is a pre-defined parameter.

- **Elitism**: A strategy that preserves the best individuals in the population across generations. It involves selecting a pre-defined portion (i.e., Elitism Rate) of top-performing individuals and copying them directly to the next generation without undergoing any genetic operations.

- **Selection**: The process of selecting individuals from the current population for breeding and creating the next generation. The selection aims to increase the probability of selecting fitter individuals for breeding, as they are more likely to produce better offspring. Various selection techniques, such as Roulette Wheel Selection, Tournament Selection, Rank Selection, or Random Selection, can be used. Note that each selection approach has own advantages and disadvantages.

- **Crossover**: Combining genetic material from two parent chromosomes (chosen by selection method) to create a new individual for the next generation. In the literature, there are various strategies for combining two parent chromosomes with their own advantages and disadvantages.

- **Mutation**: The algorithm can get stuck in local optima if the gene pool does not contain the necessary genes to lead to the global optima. Mutation randomly alters the genetic material of a chromosome based on a mutation rate parameter to create a new solution. Thus, the gene pool in the current population is diverse. Various mutation methods can be implemented depending on the task and the chromosome structure.

- **Stopping Condition**: The algorithm iterates over generation until the stopping condition is met. The literature has several stopping conditions like maximum iteration, time limit, or stopping after convergence. You can try different strategies to analyze which one gives a better outcome.

- **Intensification**: The strategic focus on promising regions of the solution space by using selection, elitism, and exploitation of the best solutions. This ensures convergence towards optimal or near-optimal solutions by emphasizing exploration in the most promising areas while maintaining diversity for effective search. You can use different strategies to explore the most promising regions deeply.

- **Diversification**: Maintaining diversity within the population to prevent premature convergence to sub-optimal solutions. This is achieved via mutation, crossover, and introducing new individuals to explore different regions of the solution space. Diversification ensures that the algorithm continues to explore a wide range of potential solutions, improving the likelihood of finding the global optimum. You can use different strategies to cover more regions to get better solutions.

The Genetic Algorithm's components can vary depending on the implementation design. The methods of components (e.g., selection, crossover, chromosomes, initial population, mutation, stopping condition, etc.) are up to the

students. However, the performance of the chosen implementation is significant for the grade. Thus, the students can implement several methods and then compare their performance in their report. Moreover, the students are expected to evaluate the effect of the parameters (e.g., mutation rate, elitism rate, population size, maximum iteration, etc.) on the algorithm's performance in terms of computational time and objective function. An example graph for the parameter analysis on the algorithm's performance is demonstrated in Figure 2 where the task is a maximization problem. Note that it is just a representation; your graphs can differ from the example.

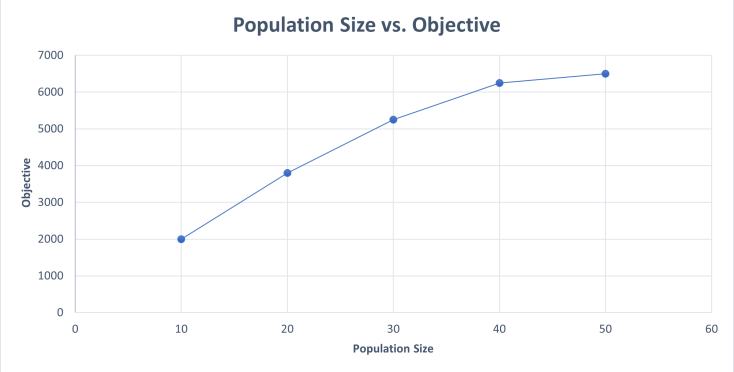

Figure 2: An Example Graph for Parameter Analyze

4 Implementation

In this text, the instructions for a programming assignment are given. The assignment involves implementing a Genetic Algorithm for the given problem in Python. As in the first assignment, the students are expected to implement their approach by utilizing the given code base. Students must utilize Python programming language with version 3.10, and Jupyter is not allowed. Students who need to learn how to code with Python can learn from available sources on the internet. Some helpful links to learn Python programming language from scratch are shared at the end of the document (Section 6). Inside the code base:

- **data_generator.py**: This Python script has two free functions to randomly generate new dataset and load a dataset from the file respectively.
- **GeneticSolver.py**: *GeneticSolver* class employs the Genetic Algorithm. For Assignment 2, you have to implement your Genetic Algorithm algorithm in **solve** method. You are free to make changes (i.e., defining variables and methods) in this class. However, you cannot change the name of the class, the constructor of the class and the definition of **solve** method.

- **Main.py**: This *Main* script is provided for your implementation. You are free to make any changes in this Python file. Note that, we will test your algorithm in our "Main" file. To use this Python script, all you need is defining a proper data file path in "*FILE_PATH*" variable. Moreover, Genetic Algorithm is a non-deterministic approach that the random seed can change the performance of the algorithm. Therefore, you need to also evaluate your algorithm with different random seeds.

For the evaluation, five distinct example datasets are provided. In each dataset (e.g., "dataset1.pkl", "dataset2.pkl", "dataset3.pkl", "dataset4.pkl", and "dataset5.pkl"), both dependent (y) and independent (x_1 and x_2) feature matrices are provided as Pickle format. The matrices are 2D NumPy array objects. Additionally, "dataset_generator.py" Python script is provided to generate more datasets for the evaluation.

5 Requirements

The requirements of this homework are listed below:

1. You have to implement a Genetic Algorithm to minimize the objective function (i.e., prediction error) for each dataset.
2. Your algorithms must contain some genetic operators (e.g., selection, crossover, mutation, etc.). Also, your report has to contain detail explanation of why/how you decide these strategies when using these genetic operators.
3. You need to demonstrate an example for each operator of your implementation in your report. You can draw them via draw.io, or you can also draw them manually.
4. You must use Python programming language with the 3.10 version. Jupyter is not allowed. You are free to choose any IDE for implementation, but PyCharm is recommended. You can create a student account with your "@ozu.edu.tr" e-mail address for an educational (free) license.
5. No library except *NumPy* and visualization libraries (e.g., Matplotlib & Seaborn & plotly, etc.) is allowed. Using other libraries will be penalized.
6. You must also write a detailed and well-organized report. Also, your report must be clear.
7. The report must cover the implementation details and design.
8. You must analyze the effect of parameters on the performance of the algorithm in terms of the computational time and the provided objective function. You are free to extend these analyzes. It would be best to put some graphs and tables in your report for better evaluation.

9. We provide five different datasets for you. Please, do not forget to test/evaluate your code with these datasets. With "dataset_generator.py", you can create more various datasets for your evaluation. We have more datasets we did not share with you to test your algorithm.

10. The performance of the Genetic Algorithm is depending on a random seed. Therefore, do not forget analyzing with different seeds.

11. You can compare various methods (e.g., selection methods, mutation methods, crossover methods, etc.) in your report.

12. Do not put any screenshots and snippets of the code or console. Because, we will already examine your code and run it to test. Instead, you can provide flowcharts, pseudo-codes, detail exhalations, graph, and so on.

13. You will submit your homework as *zip* format. Other formats, such as *rar*, will not be accepted. The *zip* file must contain your report as a PDF file and your implementation as **.py* file format. Do not forget to write your name in your Python files.

14. File name format is **NAME_SURNAME_STUDENTID_hw2.zip**.

15. Any compiler or run-time error will be penalized **(-20pt)**.

16. Any plagiarism will also be penalized (e.g., Sharing code, copy code from the Internet, asking someone to do your homework, using any language model such as **ChatGPT**, etc.).

17. **Grading**:

    - Implementation & Design & Parameter Tuning **(40pt)**
    - Report **(60pt)**
      - Implementation details and design choice with explanations and demonstrations of genetic operators **(20pt)**.
      - Analysis of the impact of the algorithm's parameters on the performance **(20pt)**.
      - Performance evaluation of the algorithm for each dataset **(10pt)**.
      - Overall (e.g., organization, writing skills, graphs, tables, etc.) **(10pt)**: Points will be awarded based on the overall quality of the report, including organization, clarity of writing, appropriate use of graphs and tables, and adherence to the guidelines.
    - Plagiarism Penalty: Any instance of plagiarism will result in a direct deduction of **-100pt** from this assignment.

18. The assignment must be done individually. While you may discuss the homework with your peers, you are strictly prohibited from sharing any ideas or code. Failure to comply with this rule will result in a penalty for plagiarism.

19. The deadline for this homework is **May 12 2024**. Late submissions will not be allowed according to the course policy.

If you have any questions regarding the homework, we encourage you to seek help during our office hours. Our office hours are held online via Zoom to ensure accessibility for all students, and the corresponding meeting links can be found on LMS. Additionally, if you prefer to communicate via e-mail, you can contact Anil Dogru. Please keep in mind that we cannot provide any extra code or ideas before the deadline, as this would be considered a violation of academic integrity. However, we will do our best to assist you with any conceptual or technical questions you may have.

6 Helpful Links for Python

You can learn the required Python programming with the helpful links below:

- Python Docs
- Tutorials Point
- Geeks for Geeks
- W3 Schools
- Programiz
- Learn Python