**M.Sc. Dissertation**

**On**

**DEVELOPMENT OF MARINE PREDATOR ALGORITHM WITH IMPROVED POPULATION INITIALIZATION FOR FEATURE SELECTION**

**By**

**AHMED, Abdulrazaq Bello**

**P18PSCS8040**

**Department of Computer Science,**

**Faculty of Physical Sciences,**

**Ahmadu Bello University, Zaria**

**Dissertation Supervisors**

**Prof. M. Abdullahi**

**Dr. S. Aliyu**

**JANUARY, 2025**

# **DECLARATION**

I declare that the work in this project dissertation entitled “**Development of Marine Predator Algorithm with Improved Population Initialization For Feature Selection”** has been performed by me in the Department of Computer Science, Faculty of Physical Science, Ahmadu Bello University, Zaria. The information derived from the literature has been duly acknowledged in the text and a list of references is provided. No part of this project dissertation was previously presented for another degree or diploma at this or any other Institution.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Name of Student Signature Date**

# **CERTIFICATION**

This project dissertation entitled “**DEVELOPMENT OF MARINE PREDATOR ALGORITHM WITH IMPROVED POPULATION INITIALIZATION FOR FEATURE SELECTION”** by **AHMAD, Abdulrazaq Bello** meets the regulations governing the award of the degree of M.Sc. in Computer Science of the Ahmadu Bello University and is approved for its’ contribution to knowledge and literary presentation.

**Prof. M. Abdullahi \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Chairman, Supervisory Committee Signature Date**

**Dr. S. Aliyu \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Member, Supervisory Committee Signature Date**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Head of Department Signature Date**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Dean, School of Postgraduate Studies Signature Date**

# ACKNOWLEDGEMENT

I begin by expressing my deepest gratitude to the Almighty Allah for bestowing upon me the strength, guidance, and blessings throughout my MSc. journey. Indeed, His divine grace has illuminated my path and granted me the perseverance to overcome challenges.

I would like to express my heartfelt gratitude to my parents for their unwavering love, encouragement, and sacrifices that have shaped me into the person I am today. Their endless support has been my pillar of strength. I extend my appreciation to my siblings for their constant belief in my abilities and for always cheering me on, even during the most challenging times. To my loving wife and child, your patience, understanding, and unwavering support have been my driving force. Your presence has made every hurdle seem conquerable, and for that, I am eternally grateful.

A special word of thanks to my supervisors, Prof. M. Abdullahi and Dr. S. Aliyu for their guidance, expertise, and invaluable feedback throughout this dissertation process. Your mentorship has been instrumental in shaping my research and academic growth. I am indebted to my lecturers for their profound knowledge, insightful teachings, and dedication to nurturing our intellectual curiosity. Your passion for education has inspired me beyond measure. To the Head of Department, thank you for fostering an environment of academic excellence and providing the resources necessary for our success.

To my esteemed course mates, thank you for the camaraderie, collaboration, and shared learning experiences that have enriched my academic journey. Your support and friendship have made this endeavor truly fulfilling. I am also grateful to my extended family members for their encouragement and understanding during my academic pursuits. Lastly, I extend my appreciation to my colleagues at Federal Polytechnic Bauchi, for their understanding and flexibility, allowing me to balance work commitments with my studies.

This dissertation would not have been possible without the collective support and encouragement of each individual mentioned above. Thank you for being an integral part of my academic success.

# ABSTRACT

*Feature Selection (FS) is considered an important but yet difficult data pre-processing technique because of the large search space. Several metaheuristics algorithm (MA) such as Particle Swarm Optimization (PSO), Salp Swarm Algorithm (SSA), Dragonfly Algorithm (DA), Grey Wolf Optimization (GWO) and Marine Predators Algorithm (MPA) have being considered in optimizing the FS process. MPA is a recent MA which is a population based algorithm modeled after the foraging actions of the marine predators. MPA as a recent and efficient population based MA has limitations in FS tasks with regards to its method of initial population generation and exploration/exploitation phase imbalance. This led to the development of more approach for initializing population and also different approach to balancing the exploitation phase so as to maximize the classification accuracy and minimize the number of features. This work proposed a new population initialization mechanism in MPA. Furthermore, the resulting algorithm is combined with a local search to help improve the exploitation phase. Ten benchmark datasets from the University of California in Irvine (UCI) repository were used throughout the experiment. Throughout these experiment, the k-Nearest Neighbor (KNN) was used as the classifier. To help with the overfitting problem, each of the datasets is divided into training and testing parts using the K-fold cross-validation. The experiments performed were divided into three parts. In the first part, the MPA was initialized using logistic chaotic map with two different control parameter value (3.7 and 3.75). The generation of the initial population with a control parameter of 3.75 recorded the best performance in terms of average classification accuracy and average standard deviation with a score of 91.91% and 0.002642 respectively. However, regarding the average number of selected features, it produced 9.4. In the second part of the experiment, random bit flipping was introduced to the last phase of the exploitation phase so as to intensify and refine solution. The experiment uses two different values for mutation rate which are 0.1 and 0.6 respectively, so as to see which give better performance. The experimental results shows that MPA performed better with a mutation rate of 0.1 with an accuracy of 92.4%, sensitivity of 92.6%, specificity of 91.98% and average selected features of 9.3. The standard deviation obtained for accuracy, sensitivity and specificity are 0.005522, 0.004396 and 0.003814 respectively. The third experiment uses the best performing control parameter rate of 3.75 from experiment I and mutation rate of 0.1 from experiment 2 to implement the proposed algorithm. The proposed algorithm was compared against the standard MPA. The proposed algorithm performed better with accuracy of 95.55% and standard deviation of 0.003083. It also performed better in sensitivity metric with a score of 89.6% and standard deviation of 0.003634 and average number of selected features of 7.7. It also perform higher in the specificity metric in few dataset. Of the 7 dataset used for comparison, the proposed algorithm has higher accuracy in 6 out of 7 and a lower number of selected features in four dataset, obtained same number of features on two dataset but obtained higher number of selected features in one dataset.*

Table of Contents

[DECLARATION i](#__RefHeading___Toc673119_23724211841)

[CERTIFICATION i](#__RefHeading___Toc673119_23724211842)

[ACKNOWLEDGEMENT ii](#__RefHeading___Toc32512_302867105)

[ABSTRACT i](#__RefHeading___Toc673119_2372421184)

[LIST OF FIGURES v](#__RefHeading___Toc32623_302867105)

[LIST OF TABLES vi](#__RefHeading___Toc32625_302867105)

[CHAPTER ONE 1](#__RefHeading___Toc673121_2372421184)

[INTRODUCTION 1](#__RefHeading___Toc673123_2372421184)

[1.1 Background of study 1](#__RefHeading___Toc673125_2372421184)

[1.2 Motivation of the Study 2](#__RefHeading___Toc673127_2372421184)

[1.3 Statement of problem 3](#__RefHeading___Toc673129_2372421184)

[1.4 Significance of the Study 3](#__RefHeading___Toc673131_2372421184)

[1.5 Aim and Objectives 4](#__RefHeading___Toc673133_2372421184)

[1.6 Research Methodology 4](#__RefHeading___Toc673135_2372421184)

[CHAPTER TWO 6](#__RefHeading___Toc673137_2372421184)

[LITERATURE REVIEW 6](#__RefHeading___Toc673139_2372421184)

[2.1 Introduction 6](#__RefHeading___Toc673141_2372421184)

[2.2 Dimensionality Reduction Technique (DR) 6](#__RefHeading___Toc673143_2372421184)

[2.3 Feature Selection 7](#__RefHeading___Toc673145_2372421184)

[2.3.1 Filter Method 7](#__RefHeading___Toc673147_2372421184)

[2.3.2 Wrapper Method 8](#__RefHeading___Toc673149_2372421184)

[2.3.3 Embedded/Hybrid Method 8](#__RefHeading___Toc673151_2372421184)

[2.4 Metaheuristic Algorithm 9](#__RefHeading___Toc673153_2372421184)

[2.4.1 Trajectory-Based Optimization 9](#__RefHeading___Toc673155_2372421184)

[2.4.2 Population-Based Optimization 10](#__RefHeading___Toc673157_2372421184)

[2.5 Evolutionary-Based Algorithm (EA) 10](#__RefHeading___Toc673159_2372421184)

[2.6 Nature-Inspired Algorithms (NIA) 10](#__RefHeading___Toc673161_2372421184)

[2.6.1 Swarm-Based Algorithms 11](#__RefHeading___Toc673163_2372421184)

[2.6.2 Physics/Chemistry-Based Algorithm 11](#__RefHeading___Toc673165_2372421184)

[2.6.3 Human-Based Algorithms 12](#__RefHeading___Toc673167_2372421184)

[2.6.4 Plant-Based Algorithm 12](#__RefHeading___Toc673169_2372421184)

[2.6.5 Maths-Based Algorithm 12](#__RefHeading___Toc673171_2372421184)

[2.7 Marine Predators Algorithm 12](#__RefHeading___Toc673173_2372421184)

[2.7.1 MPA Mathematical Model 13](#__RefHeading___Toc673175_2372421184)

[2.7.2 MPA Optimization Process 15](#__RefHeading___Toc673179_2372421184)

[2.8 Population Initialization 19](#__RefHeading___Toc673181_2372421184)

[2.8.1 Chaotic Initialization 19](#__RefHeading___Toc673183_2372421184)

[2.8.2 Logistic Chaotic Map 20](#__RefHeading___Toc673185_2372421184)

[2.9 Mutation Operation 20](#__RefHeading___Toc673187_2372421184)

[2.10 Machine Learning Algorithm 21](#__RefHeading___Toc673189_2372421184)

[2.10.1 Classification Algorithm 21](#__RefHeading___Toc673191_2372421184)

[2.10.2 K-nearest neighbors (KNN) 21](#__RefHeading___Toc673193_2372421184)

[2.11 Related Works 22](#__RefHeading___Toc673195_2372421184)

[2.12 Summary of Related Work 26](#__RefHeading___Toc1068846_2372421184)

[2.13 Research Gap 27](#__RefHeading___Toc673197_2372421184)

[CHAPTER THREE 28](#__RefHeading___Toc673137_23724211841)

[RESEARCH METHODOLOGY 28](#__RefHeading___Toc673139_23724211841)

[3.1 Introduction 28](#__RefHeading___Toc147499_2056266451)

[3.2 Implementation Setup and System Specification 28](#__RefHeading___Toc147499_205626645)

[3.3 Data Collection 29](#__RefHeading___Toc147499_20562664511)

[3.4 Dataset Description 30](#__RefHeading___Toc147499_20562664512)

[3.5 Fitness Function 31](#__RefHeading___Toc147499_205626645121)

[3.6 Classifier 31](#__RefHeading___Toc147499_205626645122)

[3.7 Transfer Function 32](#__RefHeading___Toc147499_205626645123)

[3.8 Integrating marine predator algorithm with chaotic mapping and mutation operation 32](#__RefHeading___Toc147503_205626645)

[3.8.1 Initialization phase 32](#__RefHeading___Toc673183_23724211841)

[3.8.2 Update phase 33](#__RefHeading___Toc673183_23724211842)

[3.8.3 Mutation Operation Introduction 36](#__RefHeading___Toc673183_23724211843)

[3.9 Framework of the Proposed Approach 36](#__RefHeading___Toc147505_205626645)

[3.10 Compared Algorithm and Parameter Settings 39](#__RefHeading___Toc147499_205626645124)

[3.11 Performance Metric 40](#__RefHeading___Toc147507_205626645)

[CHAPTER FOUR 43](#__RefHeading___Toc147509_205626645)

[RESULT AND DISCUSSION 43](#__RefHeading___Toc147511_205626645)

[4.1 Introduction 43](#__RefHeading___Toc147513_2056266451)

[4.2 Experiment I: Comparison of proposed initialization using different value of ‘r’ 43](#__RefHeading___Toc147513_2056266452)

[4.3 Experiment II: Comparison of proposed introduction of random bit flipping operation using different mutation rate 47](#__RefHeading___Toc147513_205626645)

[4.4 Experiment III: Proposed algorithm versus Standard MPA 51](#__RefHeading___Toc147515_205626645)

[CHAPTER FIVE 57](#__RefHeading___Toc147517_205626645)

[SUMMARY, CONCLUSION AND RECOMMENDATION 57](#__RefHeading___Toc147519_205626645)

[5.1 SUMMARY 57](#__RefHeading___Toc147521_205626645)

[5.2 CONCLUSION 57](#__RefHeading___Toc147523_205626645)

[5.2 RECOMMENDATIONS 58](#__RefHeading___Toc147525_205626645)

[REFERENCE 59](#__RefHeading___Toc673199_2372421184)

# LIST OF FIGURES

Figure 2.1: Process of Feature Selection …...………………………………………………………..7

Figure 2.2: Classification of Feature Selection Methods…….………………………………………7

Figure 2.3: NIAs FS modification categories………………………………………………………11

Figure 2.4: Marine Predators Algorithm Optimization Phases……………………………………..13

Figure 2.5: Various types of Machine learning techniques…………………………………………21

Figure 3.1: Flowchart diagram of the proposed approach……...…………………………………...38

Figure 4.1: Average Accuracy using different control parameter rate………………………………44

Figure 4.2: Average Standard Deviation for the Accuracy………………………………………….44

Figure 4.3: Average Number of features selected using different control parameter……………….44

Figure 4.6: Average Specificity using different control parameter rate…………………………….46

Figure 4.7: Average Standard Deviation for Specificity…………………………………………….46

Figure 4.8: Average Accuracy using different mutation rate………………………………………..48

Figure 4.9: Average Standard Deviation for Accuracy……………………………………………...48

Figure 4.10: Average number of features…………………………………………………………...48

Figure 4.11: Average Sensitivity for different mutatation rate……………………………………...50

Figure 4.12: Average Standard Deviation for Sensitivity…………………………………………...50

Figure 4.13: Average Specificity for different mutation rate………………………………………..50

Figure 4.14: Average Standard Deviation for Specificity…………………………………………...50

Figure 4.15: Accuracy of the Proposed Algorithm Vs MPA for each dataset………………………52

Figure 4.16: Number of selected features for the Proposed Algorithm Vs MPA…………………...52

Figure 4.17: Average sensitivity for each dataset using the Proposed Algorithm Vs MPA………...55

Figure 4.18: Average Specificity for each dataset using the Proposed Algorithm Vs MPA………..55

# LIST OF TABLES

Table 2.1: Summary of Related Work………………………………………………………………26

Table 3.1: Summary of hardware and software specifications……………………………………...29

Table 3.2: Description of the 7 UCI datasets used in the existing method………………………….30

Table 3.2: Description of the 3 additional datasets used in the proposed method…………………..31

Table 3.3: Parameter settings of both algorithm for all experiments………………………………..40

Table 4.1: Comparison based on average accuracy, average standard deviation and number of

selected features of the two values of ‘r’ used in the study……………………………...44

Table 4.2: Comparison based on average sensitivity, average specificity and average standard

deviation of the two values of ‘r’ used in the study……………………………………...45

Table 4.3: Comparison based on average accuracy, average standard deviation and number of

selected features of the two mutation rates used in the study…………………………...48

Table 4.4: Comparison based on average sensitivity, average specificity and average standard

deviation of the two values of mutation used in the study………………………………49

Table 4.5: Comparison of the proposed algorithm performance against the Standard MPA based on

average accuracy, average standard deviation and number of selected features………..52

Table 4.6: Proposed algorithm performance based on average accuracy, average standard deviation

and number of selected features for the extra datasets…………………………………..53

Table 4.7: Comparison of the proposed algorithm performance against the Standard MPA based on

sensitivity, specificity and average standard deviation and number of selected features..54

Table 4.8: Proposed algorithm performance based on average sensitivity, average specificity and

average standard deviation for the extra datasets………………………………………..55

# CHAPTER ONE

# INTRODUCTION

## 1.1 Background of study

In the last few years, machine learning (ML) which is a subset of artificial intelligence have seen a new wave of publicity fueled by the huge and ever-increasing amount of data and computational power as well as the discovery of improved learning algorithms. However, the idea of a computer learning some abstract concept from data and applying them to yet unseen situations is not new and has been around at least since the 1950s (Badillo et al., 2020). Machine learning is an evolving branch of computational algorithms that are designed to emulate human intelligence by learning from the surrounding environment. They are considered the working horse in the new era of the so-called big data. Techniques based on machine learning have been applied successfully in diverse fields ranging from pattern recognition, computer vision, spacecraft engineering, finance, entertainment, and computational biology to biomedical and medical applications (El Naqa & Murphy, 2015). Presently, data are recorded from various sources in different form thereby storing both relevant and irrelevant aspect of data not minding the cost of data. ML is not a technique that exist in isolation; it comprises of different steps which include data collection, data preprocessing, feature selection (FS), model training etc. Feature selection, as a data preprocessing strategy, has been proven to be effective (especially high-dimensional data) for various data-mining and machine-learning problems. The objectives of feature selection include building simpler and more comprehensible models, improving data-mining performance, and preparing clean data. The recent proliferation of big data has presented some substantial challenges and opportunities to feature selection (Li et al., 2017) .

Bellman (1957) coined the phrase “the curse of dimensionality” to describe the extraordinarily rapid growth in the difficulty of problems as the number of variables (or the dimension) increases. A common experience is that the cost of an algorithm grows exponentially with dimension, making the cost prohibitive for moderate or large values of the dimension (Kuo & Sloan, 2005).

Metaheuristic approaches have been widely used in recent years and are more efficient in dealing with the optimization process, including machine learning and FS, than other existing techniques (Holland, 1992). There are many metaheuristics algorithm that have been used to solve feature selection problem. Examples of Metaheuristic algorithm include Genetic Algorithm (GA) (Holland, 1992), Ant Colony Optimization (ACO) (Dorigo & Di Caro, 1999), Particle Swarm Optimization (PSO) (Kennedy & Eberhart, 1995), Salp Swarm Algorithm (SSA) (Mirjalili et al., 2017) and recently the Marine Predators Algorithm (MPA) (Faramarzi et al., 2020).

In the context of feature selection, different studies have focused on the application of the MPA to handle the feature selection problems. Despite the fact that MPA is effective at solving the FS problem, stagnation in local optima, exploitation and exploration phase imbalance, and the diversity problem, are still challenges affecting the algorithm (Mugemanyi et al., 2023).

This study seek to propose an improve version of Marine Predator Algorithm for FS, to address the problem of population initialization diversity by initializing the initial population using logistics chaotic mapping, thereby, also improving the exploration. In addition, the improved MPA seeks to address the imbalance between the exploration and the exploitation phase by introducing the bit flipping operation.

## 1.2 Motivation of the Study

The motivation for this study lies in seeking to enhance the efficiency of the Marine Predator Algorithm (MPA) for feature selection since it has shown promising result in solving optimization problem. In seeking to enhance the efficiency of the MPA, the study hope to integrate some concept into the existing MPA and combine it with a learning algorithm to solve the problem of feature selection. There exist lots of study on initial population generation using various techniques. Metaheuristics Algorithm like PSO, Whale Optimization Algorithm, Grey-Wolf Optimization, GA, Differential Evolution, ACO have being adapted to use different initial population generation scheme beside the general random generation that is common in most metaheuristic algorithm; this also serve as a motivation to this study.

## 1.3 Statement of problem

Feature Selection problem has been an active research areas in recent years as its goal is to reduce dimensionality of data while trying to maintain a high classification accuracy. Researchers have made numerous contribution to this field, however, there is still room for improvement. Marine Predator Algorithm achieves a good performance in feature selection problem but suffers from having an initial diverse population and also having imbalance between the exploration and exploitation phase which results to having all search space not optimally explored, thereby stucking in local optima convergence (Elminaam et al., 2021). This challenge hinder existing feature selection model from achieving the best global optimal solution in selecting optimal features subset while maintaining high accuracy with regards to high-dimensional dataset. Therefore, generating a diverse initial population and also improving the exploration and exploitaion phase of MPA such that all search space are fully explored for FS problem is still an area of active research. This study aim to solve this limitation by developing an improved MPA for FS by generating a diverse initial population using logistics chaotic mapping. A diverse initial population would lead to enhanced exploration as it would allow more search space to be fully explored. The study also aims to include the random bit flipping into the MPA to serve as a local search thereby enhancing the exploitation phase of the MPA.

## 1.4 Significance of the Study

The result from this study will play an important role in improving the efficiency of data mining and analysis in real-world application. The result will also be of importance for Machine learning task as it will also help to reduce the risk of over-fitting during training. To avoid the risk of over-fitting, the dimensionality of the data must be reduced by selecting only important features for the classification task.

## 1.5 Aim and Objectives

The aim of this study is to develop an improved Marine Predator Algorithm for efficient feature selection to enhance classification performance.

The objectives are:

1. Investigate the effect of introducing logistics chaotic map as a new initial population generation scheme for MPA to enhance the initial population.
2. To introduce mutation operation in MPA which will help the algorithm enhance its exploitation phase thereby improving the feature selection process.
3. To evaluate the performance of the proposed improvement with benchmark datasets against the work of Elminaam et al. (2021) in terms of average number of features, averages accuracy, sensitivity, and specificity rate.

## 1.6 Research Methodology

In order to achieve the proposed objectives, we intend to follow the steps below:

1. Marine Predator Algorithm (MPA)
2. Design and implement the MPA using the random initialization.
3. Developing the Proposed improvement to the MPA
4. Change the initialization of the MPA from random initialization to use logistics chaotic initialization
5. Introduce the mutation operation into the algorithm at the last phase of the optimization phases.
6. Dataset Preparation:
   1. Obtain the dataset for the classification task from the UCI repository.
   2. Conduct thorough data pre-processing, including data cleaning, data normalization and handling missing values to prepare the dataset for model training and evaluation.
7. Feature Selection using the proposed improved MPA:
8. Use the developed improved MPA for feature selection, emphasizing the identification of relevant features that contribute significantly to the classification performance.
9. Use an appropriate fitness function that reflects the model’s classification accuracy taking into account the selected features.
10. Model Training and Evaluation:
11. Split the dataset into training and testing sets for model development and evaluation
12. Train the MPA with KNN on the training set for 30 independent runs of 1000 iterations.
13. Evaluate the model’s performance on the testing set using metrics such as average number of selected features, accuracy, sensitivity and specificity.
14. Comparison with Elminaam et al., (2021):
15. Replicate the experimental setup and dataset used in the work of Elminaam et al., (2021) to ensure fair and consistent comparison.
16. Apply the proposed model to the same dataset used in the reference work.
17. Quantitatively compare the performance of the model against Elminaam et al., (2021) using the average number of selected features, accuracy, sensitivity and specificity.

# CHAPTER TWO

# LITERATURE REVIEW

## 2.1 Introduction

Datasets with many features/attributes are called high-dimensional data. This term has received increased attention from people. It is due to the difficulty of high-dimensional datasets that the dimensionality reduction problem became a fundamental challenge for machine learning tasks (e.g., classification, regression and clustering). Dimensionality Reduction technique have been applied in various important field like pattern recognition (Rodriguez-Lujan et al., 2013), medicine (Berisha et al., 2021), text mining (Tubishat et al., 2018) and image processing (Medjahed et al., 2016). The importance of this techniques are determined by their ability to convert massive amounts of data into a usable format. This will make large datasets more intelligible, analyzable, predictable and easy for visualization; hence enhancing knowledge discovery.

## 2.2 Dimensionality Reduction Technique (DR)

High-dimensional dataset contains a large number of noisy features, which makes data processing, knowledge mining, and pattern classification difficult. By reducing the high-dimensional space to the low-dimensional space, DR can filter out some noise and redundant information, which is a key method for solving this problem (Golay & Kanevski, 2017). DR techniques have been proposed and implemented by using feature selection (FS) and feature extraction (FE) method (Velliangiri et al., 2019). FS is also called variable selection or feature subset selection, and it is a process of selecting feature subsets that are applied to model construction. FE generates new features from the original features, that is, the new feature is a mapping of the original features. In FE, the compression of new features is more efficient but if the original feature set has an obvious physical meaning, the new features may lose meaning.

## **2.3 Feature Selection**

The most important prerequisite for using FS techniques is that the data contains many redundant or related features that can be deleted without losing a lot of information. FS is generally used in areas where there are many features and relatively few samples (or data points) (Hira & Gillies, 2015; Jie, 2008). FS techniques can be divided into three methods based on the level of the learning algorithm use in evaluation; Filter, Wrapper and Embedded.

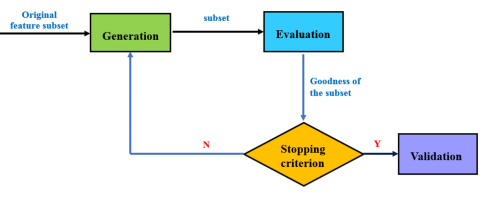


Fig 2.1: Process of feature selection (W. Jia et al., 2022)

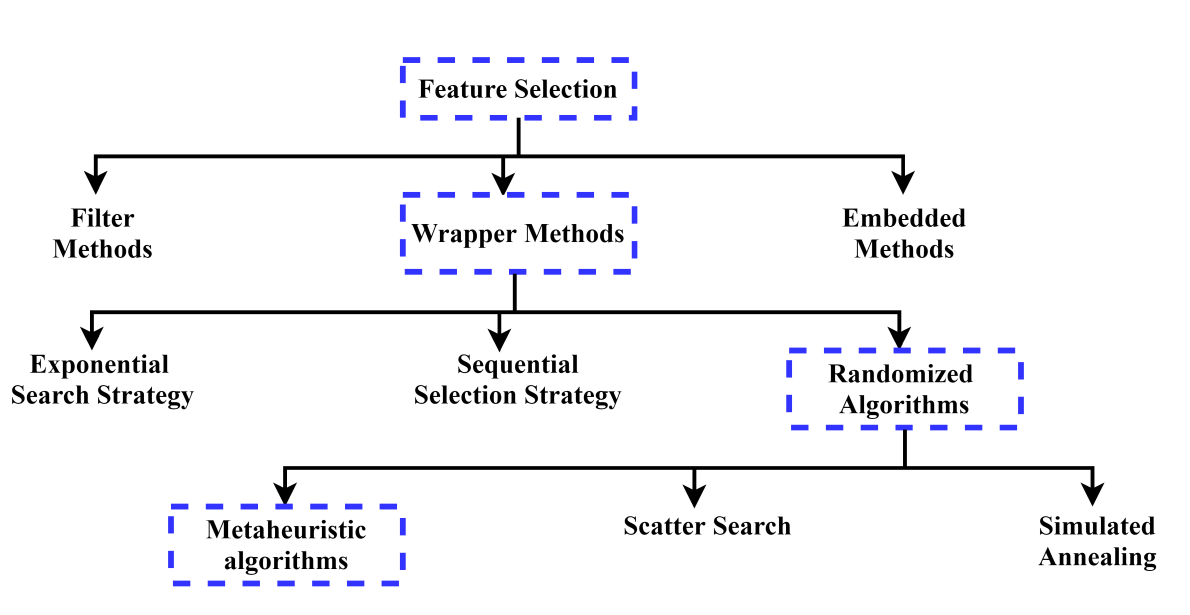


Fig 2.2: Classification of feature selection methods (Agrawal et al., 2021)

### **2.3.1 Filter Method**

Filter methods select features based on a performance measure regardless of the employed data learning algorithm. Only after the best features are found, the learning algorithms can use them. Filter methods can rank individual features or evaluate entire feature subsets. We can classify the developed measures for feature filtering into: information, distance, consistency, similarity, and statistical measures (Jovic et al., 2015). In the filter method, features are ranked according to the internal relationships among the data. Some of the popular filter models include Chi-Square (Huan Liu & Setiono, 1995), information Gain (IG) (Quinlan, 1986), Gain Ratio (Salzberg, 1994), and ReliefF (Robnik-Šikonja & Kononenko, 2003).

### 2.3.2 Wrapper Method

Wrappers consider feature subsets by the quality of the performance on a learning algorithm, which is taken as a black box evaluator. For classification tasks, a wrapper will evaluate subsets based on the classifier performance (e.g. Naïve Bayes or SVM) (Bradley & Mangasarian, 1998; Maldonado et al., 2014), while for clustering, a wrapper will evaluate subsets based on the performance of a clustering algorithm (e.g. K-means) (Kim et al., 2002). The evaluation is repeated for each subset, and the subset generation is dependent on the search strategy. Wrappers are much slower than filters in finding sufficiently good subsets because they depend on the resource demands of the learning algorithm. Practically, any combination of search strategy and learning algorithm can be used as a wrapper (Jovic et al., 2015). Searching for the optimal feature subset is a challenging problem for FS methods. Three search strategies can be used;

A complete search that tries to generate all possible feature subsets to select the best subset. In Random searches, feature subsets are selected randomly with the hope of finding the best set. In the worst cases this can become a complete search. Heuristic Search, which are guided by the random search process based on heuristic information.

### **2.3.3 Embedded/Hybrid Method**

Embedded methods perform FS during the modelling algorithm's execution. These methods are thus embedded in the algorithm either as its normal or extended functionality (Jovic et al., 2015). Common embedded methods include various types of decision tree algorithms: CART, C4.5, random forest (Sandri & Zuccolotto, 2006). Hybrid methods were proposed to combine the best properties of filters and wrappers. First, a filter method is used in order to reduce the feature space dimension, possibly obtaining several candidate subsets. Then, a wrapper is employed to find the best candidate subset (Das, 2001). While practically any combination of filter and wrapper can be used for constructing the hybrid methodology, several interesting methodologies were recently proposed, such as: fuzzy random forest based feature selection (Cadenas et al., 2013), hybrid genetic algorithms (Oh et al., 2004) and hybrid ant colony optimization (Ali & Shahzad, 2012).

## **2.4 Metaheuristic Algorithm**

Metaheuristic algorithms are characterized by flexibility, simplicity, low cost in computations, and they are derivation-free methods. They are problem-independent and aim to guide the search process in an intelligent way. The best-known and most common approach is to divide Metaheuristics based on the number of candidate solutions encountered during the optimization process into the trajectory (single-solution) and population-based (Abu Khurma et al., 2022). One of the main difference between the two approaches is that population-based metaheuristics generate initial solutions simultaneously, while single-solution metaheuristics generate the solutions sequentially (Sarhani et al., 2023). Metaheuristic techniques were employed to address the shortcomings of other search method such as time complexity, high search space, entrapment in local optima. Their approaches to global optimization mimic the physical, biological, and animal social behaviors in nature (Mirjalili et al., 2017).

### 2.4.1 Trajectory-Based Optimization

Trajectory algorithm begins with one random solution and tries to optimize the solution until a stop condition is satisfied. Computation cost is reduced significantly because only one solution is being improved upon and evaluated during the optimization process. They are local search techniques. They depend on making a few changes in the components of the current solution to find a better one. Local search implies searching within a limited region (exploitation). Examples of trajectory algorithms are Simulated Annealing (SA) (Mafarja & Mirjalili, 2017) and Tabu search (TS) (Glover, 1989).

### 2.4.2 Population-Based Optimization

Population algorithm begins with a set of randomly generated solutions and tries to enhance them during the optimization process. Each candidate solution fluctuates outward or converges toward the best solution following a certain mathematical framework. The predominance of these algorithms is because of their simplicity and flexibility. They can be adopted to deal with real-world problems without structural modifications. All that is required is an accurate representation of the problem and the structure of the optimizer is left untouched (Abu Khurma et al., 2022).

Population algorithms are more efficient in mitigating local minima compared with trajectory algorithms because more individuals and more information are shared between them. However, multiplicity in solutions increases the computation burden because more evaluations are required.

Population-based algorithms compromise of Evolutionary-Based Algorithm and Nature-Inspired Algorithms (NIA) (Dhal et al., 2020; Rai et al., 2022).

## **2.5 Evolutionary-Based Algorithm (EA)**

This is considered as the foremost class of population-based meta-heuristic optimization algorithms instigated from evolutionary phenomena of nature and biological evolution. It make use of three main operators (i.e., selection, recombination and mutation) (Rai et al., 2023). A few popular EAs are: GA (Holland, 1992), Differential Evolution (Storn & Price, 1997), Evolutionary Programming (Xin Yao et al., 1999) and Memetic Algorithm (Moscato et al., 2004).

## 2.6 Nature-Inspired Algorithms (NIA)

NIAs are the result of the union of nature with different scientific fields including physics, biology, mathematics, and engineering. Computer science utilized these relations between science and nature and turned it into a well-defined discipline for optimizing different challenging problems. NIAs are categorized into Swarm-based algorithms, Physics/Chemistry-Based algorithms, Human-Based algorithms, Plant-Based algorithm and Maths-Based algorithm (Diao & Shen, 2015; Rai et al., 2023).

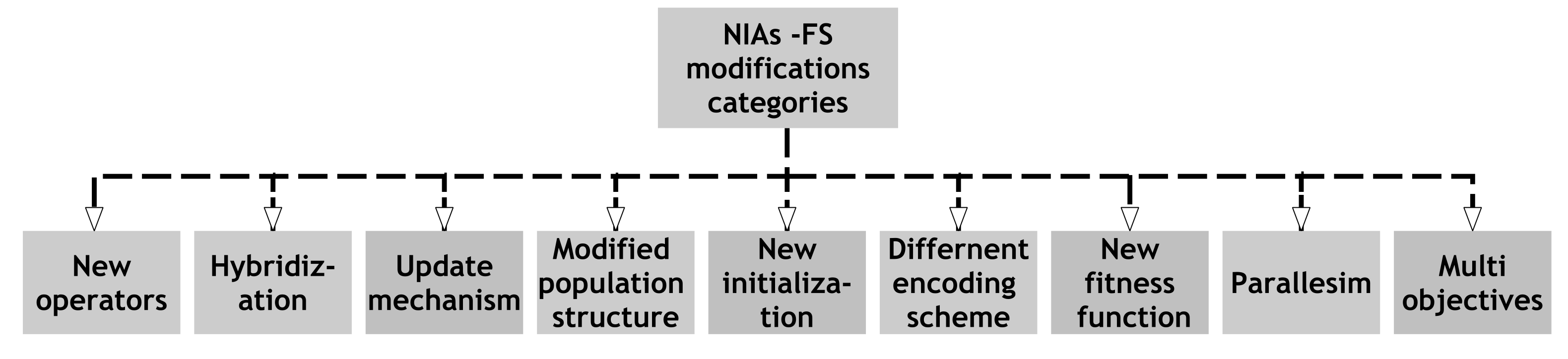


Fig 2.3: NIAs FS modifications categories (Abu Khurma et al., 2022)

### **2.6.1 Swarm-Based Algorithms**

In this algorithm, the swarms tend to mimic the behavior of the social animals or agents available in our nature such as ants foraging, birds flocking, fish schooling, bacteria moulding and animals herding. The main characteristics of swarm systems are adaptability, self-organization, distributed control, scalability, and flexibility (Brezočnik et al., 2018). Some of the Swarm-Based algorithm include: ACO (Dorigo & Di Caro, 1999), PSO (Kennedy & Eberhart, 1995), Dragonfly Algorithm (Mirjalili, 2016a), SSA (Mirjalili et al., 2017) and MPA (Faramarzi et al., 2020).

### **2.6.2 Physics/Chemistry-Based Algorithm**

The main source of inspirations is the physical processes or nature of chemical reactions which are further formulated into solutions to resolve the problems. Examples include: Henry Gas Solubility Optimization (Hashim et al., 2019), Chemical Reaction Optimization (Lam & Li, 2012), Harmony Search (Geem et al., 2001), Multi-Verse Optimization (Mirjalili et al., 2016) and Gravitational Search Algorithm (Rashedi et al., 2009).

### 2.6.3 Human-Based Algorithms

Human-Based algorithms imitates human behaviour, supremacy and intelligence. Examples include: Cultural Algorithm (Reynolds, 1994), Teaching Learning-Based Optimization (R. V. Rao et al., 2011), Jaya Algorithm (R. Rao, 2016), Anti Corona virus Optimization Algorithm (Emami, 2021) and Backtracking Search Optimization Algorithm (Civicioglu, 2013).

### 2.6.4 Plant-Based Algorithm

They mimics the intelligent behavior exhibited by plants. Examples are: Flower Pollination Algorithm (Yang, 2012), Paddy Field Algorithm (Kong et al., 2012), Photosynthetic Algorithm (Murase, 2000) and Rooted Tree Optimization (Labbi et al., 2016).

### 2.6.5 Maths-Based Algorithm

They imitate the procedure of numerical techniques, mathematical programming and its orientation to resolve numerous constraints and optimization issues of the real environment (Rai et al., 2023). Examples are Golden Ratio Optimization Method (Nematollahi et al., 2020), Sine Cosine Algorithm (Gabis et al., 2021; Mirjalili, 2016b) and Arithmetic Optimization Algorithm (Abualigah et al., 2021).

## 2.7 Marine Predators Algorithm

The foraging pattern of many animals in nature is effectively a random walk strategy; a stochastic process in which the next state/position is dependent on the current state and a transition probability to the next location which can be mathematically modeled (Bartumeus et al., 2005). These optimal strategies evolved by the ecosystem and naturally picked by predators to survive. Among marine creatures, many species including sharks, tunas, marlines, sunﬁsh and swordﬁsh exhibits Lévy-like behavior in searching for prey (Humphries et al., 2010). MPA is a new algorithm that follows the foraging strategy called Lévy and Brownian movements in ocean predators. In MPA, both Predator and Prey are hunting each other, and meanwhile, both are also looking for food.

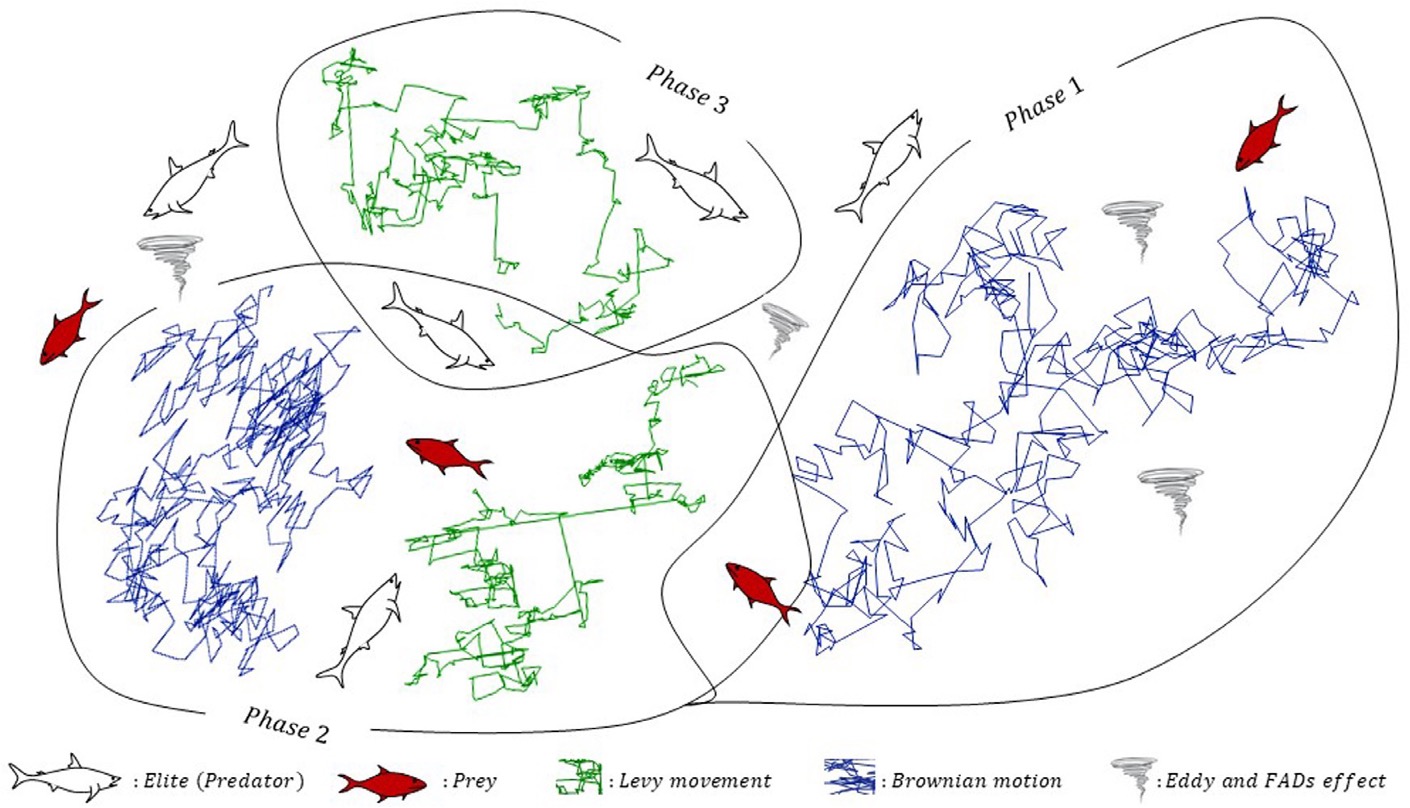


Fig 2.4: Marine Predators Algorithm Optimization phases (Faramarzi et al., 2020)

### 2.7.1 MPA Mathematical Model

2.7.1.1 Lévy Motion

Lévy process is known as the continuous-time analog of a random walk. Lévy flight is one of random walk type in which the step sizes are demonstrated by a probability function defined by Lévy distribution (power-law tail) as shown in Equation 2.1:

(2.1)

where the flight length denoted to *xj* while is the power-law exponent .

The integral form of the probability density of the Lévy stable process is determined by as shown in Equation 2.2:

(2.2)

where α shows the distribution index and controls the properties of the process’s scale while γ selects the unit of scale. When α = 2, it shows a Gaussian distribution, and when α = 1, it represents a Cauchy distribution. In equation 2.3, the solution generally requires using the method of series expansion only when x has enormous value as follows:

(2.3)

Gamma function in which for integer α numbers (1+α) is equal to α!

Using the Magneta method for extracting random numbers based on Lévy distribution as showed in equation 2.4 where the arbitrary value of index distribution (α) ranged in 0.3 and 1.99:

(2.4)

where 𝑥 and 𝑦 represent the variables from Normal distribution whose standard deviations are 𝜎𝑥 and 𝜎𝑦 respectively as shown in Equation 2.5, 2.6 and 2.7:

(2.5)

(2.6)

(2.7)

2.7.1.2 Brownian Motion

Brownian motion is the process in which the probability function of Gaussian distribution determines the step length when the mean is equal to zero (µ = 1) and variance (σ2 = 1 ). At point x for this motion, the Probably Density Function (PDF) is defined as Equation 2.8.

(2.8)

### 2.7.2 MPA Optimization Process

The MPA algorithm revolves around the relationship between predators and preys. When preys seek their food, it means that the preys then become predators. Therefore, the algorithm shows the relationship between the top predator (also known as Elite) and the prey.

Similar to various other population-based meta-heuristic algorithms, MPA also initializes search process by randomly locating N search agents around search space.

(2.9)

During initialization, along the main population matrix, another P × U matrix comprising of search agents with best fitness values is created, where P and U denote population size and problem dimensions respectively. MPA calls it Elite. Firstly, a prey matrix is constructed to contain the initial preys.

(2.10)

A top predator vector represents the fittest solution obtained from the prey matrix that is identified as:

(2.11)

Then Elite matrix is built by replicating p times the top predator vector. The Elite matrix is used to update the better values of the top predator.

(2.12)

As with other optimization algorithms, MPA also focuses on exploration in the initial loops and exploitation in the last loops. To do this, MPA relies on velocity ratio between predators and preys. Therefore, the ability of MPA in exploration and exploitation is exhibited in three main phases. One-third of the total number of iterations is used in each phase. This process is divided into three main phases of optimization based on various velocity ratios and at the same time imitating the natural life of predators and prey. These phases are defined as follows:

**Phase 1:** when there is a high-velocity ratio (v ≥ 10) and the prey is moving faster than the predator. In this initial iteration of optimization, exploration is important. Therefore, the optimal strategy for a predator is standing still. Meanwhile, the prey can move in Lévy or Brownian, and it moves forward to the predator. This phase is interpreted in a mathematical model as equation 2.13:

(2.13)

Based on the Normal distribution of Brownian motion, the vector of RB has random numbers. Multiplying RB by Prey imitates the movement of prey. P is a constant number equal to 0.5, and R is a uniform random numbers vector in [0,1]. This phase happens in 1/3 of iterations, when the velocity of movement is high to allow high levels of exploration.

**Second Phase:** when both Predator and Prey are moving simultaneously. This scenario happens when the exploration tries to be fleetingly converted to exploitation. Exploration and exploitation matters are included in this Phase, when Prey is responsible for exploitation and Predator for exploration. Based on unit velocity ratio (v ≈ 1), Prey’s best strategy is to move in Lévy, the best strategy for Predator is Brownian. The mathematical model of this rule is applied as Equation 2.14:

(2.14)

For the first half of the population, RL denotes a vector of random numbers based on Lévy distribution. The movement of Prey in the Lévy manner is simulated by the multiplication of RL Adding step size to the prey position further simulates the movement of prey. Most step sizes in the Lévy distribution are small. For the second half of the population, the equation model is 2.15:

(2.15)

While CF is an adaptive parameter to control the stepsize for predator movement. The movement of Predator in the Brownian manner is simulated by the multiplication of RB Furthermore, Elite simulates the movement of Prey as it updates its position depending on the movement of predators in Brownian motion.

**Third Phase:** In this phase, when Predator is moving faster than Prey with a low-velocity ratio (v = 0.1), it has high exploitation capability. The movement of Predator in Lévy strategy is simulated by the multiplication of RL Furthermore, while the Predator’s movement helps Prey update its position, Elite is simulated by adding the step size to the Elite position. Equation 2.16 represent the model:

(2.16)

Environmental issues can also cause changes in the behavior of marine predators. One example of this is the effects of Fish Aggregating Devices (FADs), also known as eddy formation. Sharks spend more than 80% of their time near FADs, and during the other 20%, of their time, they take long jumps in different dimensions, probably to find environments with different prey. The FADs are considered local optima and their effect is trapping these points in search space. The mathematical model of FAD’s effect is defined in Equation 2.17:

(2.17)

The uniform random number in [0, 1] is r, while the vectors that form the lower and upper boundary of the dimensions are X\_max, and X\_min. FADs will affect the optimization process when FADs = 0.2. The binary vector U is defined by generating a random vector in [0, 1].

|  |
| --- |
| Algorithm 1: Pseudo-Code of MPA Algorithm (Faramarzi et al., 2020) |
| Initialize a population of search agents (Prey) i=1,2,. . . ,n |
| **while** stop condition not met **do** |
| Compute the initial fittness function and construct the matrix of Elite |
| **if** Iter < Max\_Iter/3 **then** |
| Update the prey with the aid of Eq. (2.13) |
| **if** Max\_Iter/3 < Iter < 2 \* Max\_Iter/3 **then** |
| For the first half of population (i =1,. . . ,n/2) |
| Update the prey via Eq. (2.14) |
| For the second half of population (i =n/2,. . . ,n) |
| Update the prey via Eq. (2.15) |
| **if** Iter > 2 \* Max\_Iter/3 **then** |
| Update the prey via Eq. (2.16) |
| **end** **if** |
| **end** **if** |
| **end** **if** |
| Complete the memory saving and Elite update |
| Implementing FADs effect and update using Eq. (2.17) |
| **end** **while** |

## **2.8 Population Initialization**

As an essential step of metaheuristic optimizers, initialization seriously affects the convergence speed and solution accuracy. The state-of-the-art initialization method is to generate a small initial population to cover the search space as much as possible uniformly. However, these approaches have suffered from the curse of dimensionality, high computational cost, and sensitivity to parameters, which ultimately reduce the algorithm’s convergence speed (Li et al., 2021). Some popular initialization methods are: Chaotic initialization, Opposition-based learning, Diagonal uniform distribution, Latin hypercube sampling, and Sequence-based deterministic initialization. It is widely accepted in the research community that initial population of solutions plays an important role in the optimization process, because all the solutions generated thereafter depend, to a certain extent, on their preceding solutions (Sarhani et al., 2023).

### 2.8.1 Chaotic Initialization

The nature of chaos is apparently random and unpredictable, and chaotic sequences have been proven easy and fast to generate and store (Heidari-Bateni & McGillem, 1994). There are many kinds of chaotic maps, such as Chebyshev map, circle map, iterative map, logistic map, tent map, sine map, gauss map etc. Chaotic sequence has the advantages of introducing chaos or unpredictability into the optimisation, increasing the range of chaotic motion, and using these chaotic induced variables to search the space effectively.

The starting positions in metaheuristics algorithm are generated randomly, diversity of initial population is very important for helping the population spread in search space. Therefore, when initial populations are generated by chaotic maps, which can produce a well distribution by the properties of random and ergodicity of chaos. The chaotic sequence can accelerate the convergence and enhance the global search capability.

### **2.8.2 Logistic Chaotic Map**

One of the simplest and most studied nonlinear system is the logistic map. It was originally introduced as a demographic model by Pierre Franois Verhulst in 1838 (Kanso & Smaoui, 2009).

The logistic map is given by Equation 2.18 below:

(2.18)

where r is the control parameter and n is the number of iterations.

Using the logistic chaotic function, Suresh et al., (2017) proposed novel improvements on the CS, and one of these improvements is the use of the logistic chaotic function to initialise the population. Logistic chaotic function was also used in the work of Afrabandpey et al., (2014) where the bat population was initialised using chaotic sequences, instead of the random number generator.

## **2.9 Mutation Operation**

In an organism, mutation is an error that occurs during DNA replication (meiosis). The error specifically results from a permanent deletion, insertion, or alternation on the DNA segment (nucleotide sequence of the genome). Even though this is a small genome error, it causes abnormal changes in the characteristics of an organism. Evolutionary and GA inspired the same idea to make changes and increase the diversity in the population. The advantages of mutation come from preventing solutions becoming similar and thus ensuring that evolution does not stop. Mutation operators alter one or more gene values (a bit in chromosome vector) which causes the solution to be changed from its previous state. Besides diversity, the mutation could contribute to mitigating the local minima problem. Equation 2.19 identifies the mutation process where is the ith element at the dth dimension in the solution.

(2.19)

## 2.10 Machine Learning Algorithm

Machine learning is a paradigm that refers to improving future performance by learning from existing experience (in this case, historical data). This field's sole concentration is on automatic learning methods. Learning is the automatic adjustment or enhancement of an algorithm based on previous "experiences" without the need for human intervention. To solve data challenges, Machine Learning employs a variety of algorithms. Researchers like to point out that there is no such thing as a "one-size-fits-all" algorithm for solving a problem. The type of method used is determined by the type of problem you're trying to solve, the number of variables involved, the best model to use, and so on (Mahesh, 2020). Figure 2.5 shows various types of machine learning technique.

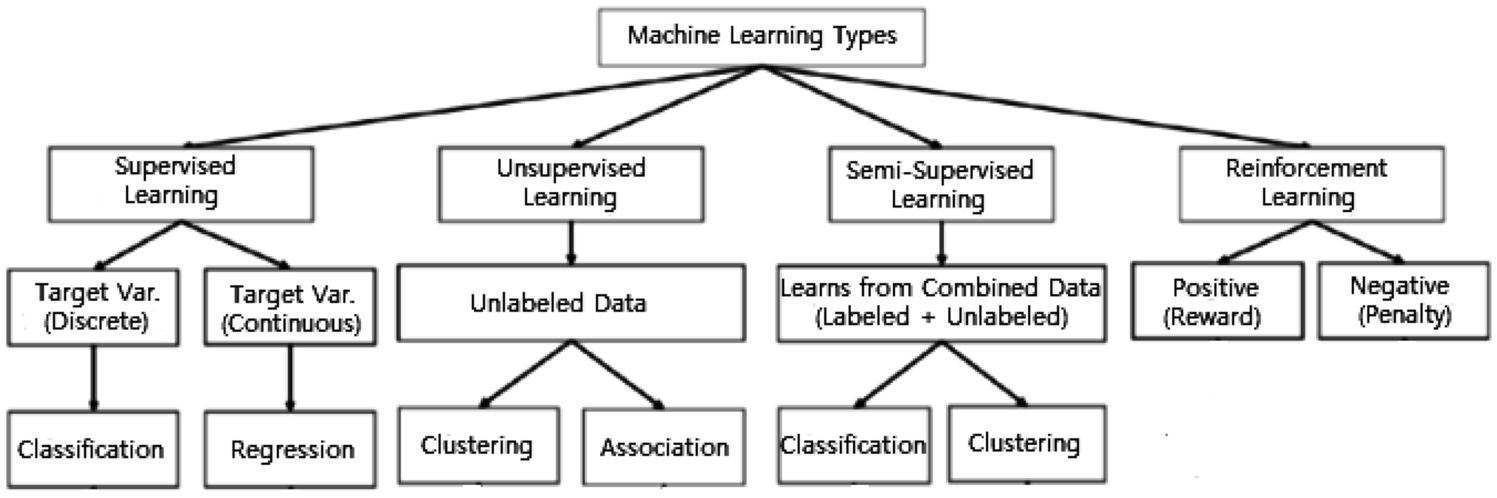


Fig 2.5: Various types of machine learning techniques (Sarker, 2021)

### 2.10.1 Classification Algorithm

Supervised learning is the machine learning task of learning a function that maps an input to an output based on example input-output pairs. It infers a function from labelled training data consisting of a set of training examples (Mahesh, 2020). The supervised machine learning algorithms which deals more with classification includes the following: Logistic Regression, Naïve Bayes, Perceptron, Support Vector Machine, Boosting, Decision Tree, Random Forest (RF); Neural networks, Bayesian Networks etc.

### 2.10.2 K-nearest neighbors (KNN)

K-Nearest Neighbors (KNN) (Aha et al., 1991) is an “instance-based learning” or non-generalizing learning, also known as a “lazy learning” algorithm. It does not focus on constructing a general internal model; instead, it stores all instances corresponding to training data in n-dimensional space. KNN uses data and classifies new data points based on similarity measures (e.g., Euclidean distance function) (Pedregosa et al., 2011). Classification is computed from a simple majority vote of the k nearest neighbors of each point. It is quite robust to noisy training data, and accuracy depends on the data quality. The biggest issue with KNN is to choose the optimal number of neighbors to be considered. KNN can be used both for classification as well as regression. The k-nearest neighbors (KNN) algorithm is a simple, supervised machine learning algorithm that can be used to solve both classification and regression problems. It's easy to implement and understand, but has a major drawback of becoming significantly slows as the size of that data in use grows (Mahesh, 2020).

## 2.11 Related Works

The MPA by Faramarzi et al., (2020) is a recent MA from the swarm intelligence (SI) family that was proposed in 2020. MPA shares most of the characteristic available with other MAs, namely, simplicity in implementation, flexibility in hybridization with other MA, few adjustable parameters, high calculation accuracy and excellent memory retention. MPA has shown high effectiveness in solving real-world optimization problems. MPA has performed excellently in the many areas of application, yet it still has some flaws, such as all search space not optimally explored, imbalance between the exploration and exploitation phases for higher dimensional problems, inability to generate diverse initial population with high effectiveness and its prone to premature convergence due to the division of the MPA optimization process into three (3) phases. Sometime, Predators are gathered at the global optimum point and this may lead to local optima stagnation (Mugemanyi et al., 2023). Since MPA was proposed, many researchers have proposed different enhancement/improvement to the original MPA to increase its performance on a variety of optimization problems which include Feature Selection.

To show the importance of initial population, Q. Li et al., (2020) studied the sensitivity of five algorithms, including PSO, GA, and DE, to the initialization phase. The authors pointed out that some algorithms are more sensitive to initialization than others. The authors compared several different initialization methods, based on different probability distributions and found that the PSO performs differently for different initialization methods while DE is more robust when it comes to the initialization method.

A study by Beheshti (2022) proposed a Binary Marine Predator Algorithm (BMPA-TVSinV) for feature selection. MPA algorithm was been introduced for the continuous search space, while the feature selection problem is a binary optimization problem. Therefore, MPA was converted to binary form by transfer functions. Two novel time-varying Sine and V-shaped transfer functions were applied in BMPA. The proposed algorithm was evaluated against high-dimensional and COVID-19 datasets. BMPA-TVSinV archives a higher accuracy and feature reduction rate on datasets. Also, the K-Nearest Neighborhood (KNN) was applied for classification.

In a study by Abdel-Basset et al., (2021), MPA was used for the first time for PV models. The MPA was improved to accurately estimate the parameters of different PV models. The proposed algorithm was compared with recent state-of-the-art algorithms. The improved marine predators algorithm employs a population improvement strategy to enhance the quality of the solutions by utilizing two different ways to handle the solutions inside the population based on the population mean fitness. The location of a high-quality solution is improved using an adaptive mutation operation, while the location of a low-quality solution is updated according to the location of the best-obtained solution and the location of a good solution selected from the population. The results showed that the proposed algorithm is highly correlated with the measured current–voltage data so that it can offer a useful alternative for parameter estimation of photovoltaic models.

In a study by Pan et al., (2022), the authors proposes a Multigroup Marine Predator Algorithm (MGMPA) so as to improve the initial population and solve Economic Load Dispatch problem (ELD). The multigroup mechanism divide the initial population into several independent groups. These groups generate the top predator and the Elite matrix based on different strategies and share information after a fixed iteration. The experimental results showed that the proposed MGMPA has significant advantages under the CEC2013 suite and obtains the minimum cost of power system operation and the maximum economic benefits in the application when compared with classic algorithm like PSO, SSA and MPA.

The author Oszust, (2021) introduce a local escaping operator (LEO) as an Hybridization of the MPA algorithm to solve the premature convergent due to an imbalance between the exploration and exploitation capabilities of the MPA. In the proposed approach, the worst candidates were replaced with solutions created by the LEO. The results show the superiority of the LEO-MPA over the MPA and recent algorithms.

In a study by Alrasheedi et al., (2022), the author proposes a chaos embed marine predator algorithm (CMPA) for feature selection. The author alter the decision making of the position update phase of stage-2 with a chaotic sequence. The proposed binary algorithm was tested with over 17 data sets. Graphical analysis along with statistical comparison of the proposed algorithm with others revealed that a modification in the stage-2 of MPA algorithm has some positive implications on the optimization performance of MPA. It was compared with enhanced chaotic grasshopper optimization algorithm (ECGOA) (with sine map), sinusoidal bridging mechanism-based grasshopper algorithm (with sine map) and enhanced chaotic artificial bee colony algorithm (ECABC) (with sine map).

In a study to improve the search space of MPA by H. Jia et al., (2022), an improved algorithm called co-evolutionary cultural mechanism-based marine predators algorithm (CECMPA) was proposed. The CECMPA has a higher probability of avoiding local optimum and can search the global optimum quickly. The author use CECMPA to perform feature subset selection and optimize hyperparameters in support vector machine (SVM) simultaneously. The proposed method was tested on twelve datasets from the University of California Irvine (UCI) repository. The authors divides the population into different sub-populations, the sub-populations evolve in parallel in their respective spaces based on MPA. The CECMPA increases the diversity of the population and the probability of avoiding trapping into the local optimum.

A study by Al-qaness et al., (2022) develop a new time-series forecasting approach for wind power production. The authors propose an efficient variant of the MPA using the mutation operators called MPAmu, to prevent its premature convergence on local optima. The developed MPAmu is used to optimize the ANFIS parameters and to boost its configuration process. The MPAmu depends on boosting the searchability of the traditional MPA using the mutation operators, which showed significant impacts on the quality of solutions of the MPA. More so, the developed MPAmu is utilized to optimize and boost the parameter configuration process of the conventional ANFIS model. The result was compared with the traditional ANFIS, neural networks (NN), support vector machine (SVM), and long short term memory (LSTM).

In a study to improve the search ability of MPA, Balakrishnan et al., (2022) improve the MPA by using ROBL for FS problem using a high-dimensional dataset. The incorporation of ROBL in the MPA enhances its ability to reconnoiter bigger search space. The proposed algorithm generates a new population based on the initial and random opposite population. The performance of ROBL-MPA is inspected on six high-dimensional microarray datasets. The proposed ROBL-MPA outperforms traditional MPA based on several benchmark performance analysis tests that was used.

## 2.12 Summary of Related Work

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S/N | Author | Title | Findings/Method Used | Weakness |
| 1 | (Elminaam et al., 2021) | An Efficient Marine Predators Algorithm for  Feature Selection | Use the MPA to solve FS problem using KNN classifier | Initial population not diversified. Exploration and Exploitation phase not enhanced. |
| 2 | (Alrasheedi et al., 2022) | Chaos Embed Marine Predator (CMPA) Algorithm for  Feature Selection | Use the 𝛽-chaotic sequence to balance Exploration and exploitation phases in the second phase | β-chaotic sequence used might get stuck in local optima. |
| 3 | (Al-qaness et al., 2022) | Boosted ANFIS model using augmented MPA with Mutation | Local optima avoided using Mutation operator | Balance between exploration and exploitation not considered likewise the initial population. |
| 4 | (Abdel-Basset et al., 2021) | Parameter Estimation of Photovoltaic models using an improved MPA | Exploration phase was ameliorated using adaptive mutation operation | The exploitation phase was not handled to maintain balance. |
| 5 | (Oszust, 2021) | Enhanced Marine Predators Algorithm with Local Escaping Operator for global optimization | Population diversity boosted using Local Escaping Operator | The exploitation phase was not handled to maintain balance. |
| 6 | (Jiang et al., 2022) | Ridge Regression based on t – distribution Marine Predatos Algorithm | Initial population was improved using t-distribution | The t-distribution assumes the underlying population data is normally distributed and is beneficial for small sample size. |
| 7 | (H. Jia et al., 2022) | Improved marine predators algorithm for  feature selection and SVM optimization | Population diversity improved and local optima avoided using Co-evolutionary cultural strategy | Increase in computational cost |
| 8 | (Balakrishnan et al., 2022) | Excogitating Marine Predators Algorithm based on random opposition based learning for Feature Selection | Local optima avoided, exploration and exploitation phases ameliorated using ROBL strategy | Longer time to find good feature subsets. Lack of control over the opposition generation limit the effectiveness of ROBL. |
| 9 | (Khaire et al., 2022) | Hybrid marine predator  algorithm with simulated annealing for feature selection | Exploration and exploitation phases improved, local minima avoided using SA | Increase in computational cost |
| 10 | (Abd Elaziz et al., 2022) | Modified marine predators algorithm for feature selection: case study  metabolomics | Exploration phase enhanced using SCA | The high exploration ability of SCA cause disruption in the MPA convergence process |

## 2.13 Research Gap

From the related works in section 2.11, it was observed that the MPA has been applied in diverse optimization task for example, feature selection task, image segmentation task, power economic dispatch problem. Some of the results obtained looks promising, however, the problem of having a diverse initial population, search space not fully explored and search not intensified in promising region still exist despite efforts from a number of researchers. Hence, this study seeks to improve the population initialization of MPA for an enhanced feature selection task. In addition, the study seeks to help the algorithm balance its exploration and exploitation phase through introduction of mutation operation thereby improving the feature selection process.

# CHAPTER THREE

# **RESEARCH METHODOLOGY**

## **Introduction**

This chapter describes the research method utilized in the study. It describes the detailed structure and flow of the proposed technique, design, and algorithms that will be used to for the feature selection. The methodology consists of the following stages:

1. Implementation Setup and System Specification
2. Data collection and description
3. Data Preprocessing
4. Parameters and settings used in the proposed and existing algorithms
5. Objective functions
6. Integrating marine predator algorithm with logistics chaotic mapping
7. Integrating marine predator algorithm with mutation operation
8. Evaluation metrics

## **Implementation Setup and System Specification**

Python programming language is mostly used for the data preprocessing, training, and implementation of the classification model. Python's advantages such as ease of use, huge framework library and flexibility make it the best choice for creating machine learning models (Hao & Ho, 2019). Python offers machine learning settings a remarkable level of power and customization. The language's straightforward syntax makes it easier to validate data and speeds the scraping, processing, refining, cleaning, arranging, and analyzing operations, which reduces the difficulty of working with other programmers (Srinath, 2017).

The following are the hardware and software specifications used for the implementation of the model:

**Table 3.1**: Summary of hardware and software specifications

|  |  |
| --- | --- |
| **Software specification** | |
| Language | Python 3.11.0 |
| Platform | Google Colab, RAM 13GB, Microsoft excel |
| Operating system | Ubuntu 22.04.4 LTS |
| **Hardware specification** | |
| Processor | Intel(R) Core(TM) i5-8350U CPU @ 1.70GHz x 8 |
| Computer model | Dell Inc Latitude 5490 Intel(R) at 1.70GHz. |
| Memory | 16.00 GB of RAM |
| Storage | 256.1 GB of Solid State Drive |

## **Data Collection**

In order to evaluate and compare the performance of the existing and proposed models, ten (10) datasets from the University College Irvine (UCI) Machine Learning repository were used, which are available for download from https://archive.ics.uci.edu/. The UCI Machine Learning Repository is a collection of databases, domain theories, and data generators that are used by the machine learning community for the empirical analysis of machine learning algorithms. The archive was created as file transfer protocol (ftp) archive in 1987 by UCI PhD student, David Aha. Since that time, it has been widely used by students, educators, and researchers all over the world as a primary source of machine learning datasets. These datasets poses a challenge in machine learning due to their high-dimensionality, which is characterized sometimes by having a large number of features compared to the number of instances. It is also characterized with having class imbalance in some datasets. High-dimensionality in this datasets can lead to issues such as overfitting, increased computational time, and the curse of dimensionality, thereby making the performance of machine learning algorithms decreases as the number of features increases.

## **Dataset Description**

In this analysis, ten (10) popular datasets from the UCI repository will be used for the evaluation of the proposed and existing methods. Several researchers have used these datasets as a benchmark for comparison in the field of FS. The datasets available in the repository are usually grouped into 3 categories based on sizes: small (features < 15), medium (15 ≤ features ≤ 50), and large (features > 50) (Ghosh et al., 2019). During the training of machine learning model, choosing relevant features for classification is a very important task which this study try to solve. Seven (7) datasets were carefully selected from the study of Elminaam et al., (2021) so as to be use for comparison with the result of the proposed improvement to the standard MPA. The characteristics of the seven (7) datasets are summarized in Table 3.2 below. Three (3) other datasets in the same categories of the other seven (7) were also selected from the UCI machine learning repository. The characteristics of the other three (3) datasets are summarized in Table 3.3 below.

**Table 3.2:** Description of the 7 UCI datasets used in the existing method

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S/N** | **Datasets** | **No. of samples** | **No. of classes** | **No. of features** |
| **1** | BreastEW | 569 | 2 | 30 |
| **2** | CongressEW | 435 | 2 | 16 |
| **3** | Ionosphere | 351 | 2 | 34 |
| **4** | Sonar | 208 | 2 | 60 |
| **5** | SpectEW | 267 | 2 | 22 |
| **6** | Vote | 300 | 2 | 16 |
| **7** | WaveformEW | 5000 | 3 | 40 |

**Table 3.3:** Description of the 3 additional datasets used in the proposed method

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S/N** | **Datasets** | **No. of samples** | **No. of classes** | **No. of features** |
| **1** | HeartEW | 270 | 2 | 13 |
| **2** | Vowel | 528 | 11 | 10 |
| **3** | Glass | 214 | 7 | 10 |

## **Fitness Function**

Finding the feature subset that accurately classify items with lower error is the goal of F.S methods.

A critical aspect of F.S. methods is the assessment of the quality of the selected samples. With Wrapper technique, a learning algorithm is involved in the evaluation process. In this study, the well-known and commonly use classifier, i.e., k Nearest Neighbors (k-NN) classifier, is used as a classifier. The fitness function controls the Accuracy of the selected features. When the features selected in a particular subset are relevant, the obtained Accuracy will be better. The solutions obtained by MPA must be evaluated during the iterative process to verify each iteration’s performance. The fitness function used by the MPA is defined as:

(3.1)

The classification error rate is defined as R, where C refers to the total number of features in the data set. The two parameters, α, and β, refer to the importance of classification quality and subset length. α is defined in the range [0, 1].

## **Classifier**

Among the classification algorithm in machine learning, K-Nearest Neighbors is one of the simplest, powerful and easy to implement classifier (Kataria & Singh, 2013). KNN is non-parametric; it does not make any underlying assumptions about the distribution of data; new sample point are classified base on the class of the nearest neighbors. The KNN classifier will be used in this work to introduce a FS based on the wrapper method.

## **Transfer Function**

MPA is an optimization algorithm that was created primarily to solve problem in the continuous space ranging from the field of engineering to science and other domain. However, FS problem is a discrete problem; so, the algorithm need to be transformed to search in a discrete or binary search spaces. Transfer function (TF) is a basic method for converting continuous algorithms into binary. A mathematical operator called TF determines the likelihood that an input value will be either 1 or 0. TF is therefore responsible for mapping of continuous search space to binary search space i.e., 0 and 1, where ‘0’ represents unselected features and ‘1’ represents selected features (Wei et al., 2020). Each solution in the search space is depicted in the form of a binary vector of size D, where D is the obtainable features in the dataset. This work uses a variant of the sigmoid function from the study of Chaudhuri & Sahu, (2021) which is represented in equation 3.2 due to its ability to produce output in the range [0, 1].

(3.2)

(3.3)

where and represent a possible solution

## **Integrating marine predator algorithm with chaotic mapping and mutation operation**

### **3.8.1 Initialization phase**

Depending on the size of the input dataset, the algorithm generates the initial population of solutions randomly across the search space.

This study changes the initialization from random to use logistics chaotic mapping. This change in initialization will create a diverse solution which help promote the exploration of the search space.

### **3.8.2 Update phase**

During this stage, the algorithm modifies each search agent positions at each iteration. The algorithm revolves around the relationship between predators and preys. However, preys also seek their food. It means that the preys then become predators. Therefore, the algorithm shows the relationship between the top predator (also known as Elite) and the prey. To initiate the MPA optimization process two same dimension matrices for Elite and Prey are constructed based on the population of search agents p, and the number of updating parameters u. Firstly, a prey matrix is constructed to contain the initial preys.

(3.4)

A top predator vector represents the fittest solution obtained from the prey matrix that is identified as:

(3.5)

Then Elite matrix is built by replicating p times the top predator vector. The Elite matrix is used to update the better values of the top predator.

(3.6)

The ability of the in exploration and exploitation is exhibited in three main phases. One-third of the total number of iterations is used in each phase. These phases are defined as follows which also shows how the predator/prey position update happen:

**Phase 1:** the prey is moving faster than the predator. Therefore, the optimal strategy for a predator is standing still. Meanwhile, the prey can move in Lévy or Brownian, and it moves forward to the predator. This phase is interpreted in a mathematical model as equation 3.7:

(3.7)

Based on the Normal distribution of Brownian motion, the vector of RB has random numbers. Multiplying RB by Prey imitates the movement of prey. P is a constant number equal to 0.5, and R is a uniform random numbers vector in [0,1]. This phase happens in 1/3 of iterations, when the velocity of movement is high to allow high levels of exploration.

**Second Phase:** when both Predator and Prey are moving simultaneously. This scenario happens when the exploration tries to be fleetingly converted to exploitation. Exploration and exploitation matters are included in this Phase, when Prey is responsible for exploitation and Predator for exploration. Prey’s best strategy is to move in Lévy and the best strategy for Predator is Brownian. The mathematical model of this rule is applied as Equation 3.8:

(3.8)

For the first half of the population, RL denotes a vector of random numbers based on Lévy distribution. In the first half population, the motion in Lévy of the prey is considered as exploration capability. The movement of Prey in the Lévy manner is simulated by the multiplication of RL Adding step size to the prey position further simulates the movement of prey. Most step sizes in the Lévy distribution are small.

In the second half of the population, the predator performs the Brownian motion. A new position of the prey is updated based on the predator’s movement. In other terms, exploitation capability mainly depends on the movement of the predator. For the second half of the population, the equation model is 3.9:

(3.9)

While CF is an adaptive parameter to control the stepsize for predator movement. The movement of Predator in the Brownian manner is simulated by the multiplication of RB Furthermore, Elite simulates the movement of Prey as it updates its position depending on the movement of predators in Brownian motion.

**Third Phase:** This stage mainly focuses on exploitation. The movement of the predator is simulated in Lévy strategy. The position of prey is updated based on the predator’s motion. It has high exploitation capability. The movement of Predator in Lévy strategy is simulated by the multiplication of RL Furthermore, while the Predator’s movement helps Prey update its position, Elite is simulated by adding the step size to the Elite position. Equation 3.10 represent the model:

(3.10)

In this phase, no modification is done to how position update happens to the predator/prey in the algorithm.

### 3.8.3 Mutation Operation Introduction

After the last phase of the optimization process of the algorithm where all population are engaged in exploitation, before the memory update is applied, mutation operation is applied. There are various types of mutation operation but this study uses the Random bit flipping mutation. The advantages of mutation come from preventing solutions becoming similar and thus ensuring the evolution does not stop. Mutation operators alter one or more gene values (a bit in chromosome vector) which causes the solution to be changed from its previous state. Besides diversity, the mutation could contribute to mitigating the local minima problem.

Equation 3.11 identifies the mutation process where is the ith element at the dth dimension in the solution.

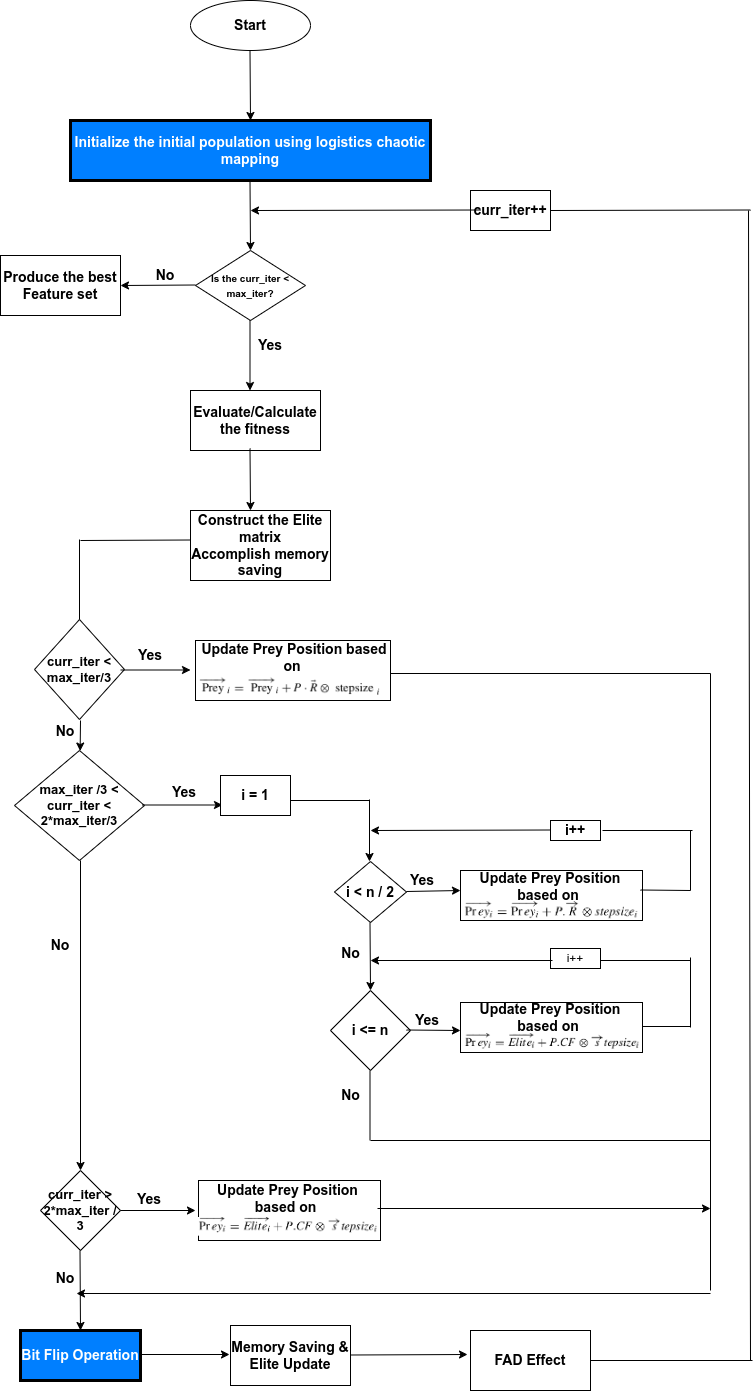
(3.11)

## **Framework of the Proposed Approach**

The goal of this section is to present a wrapper feature selection technique that effectively identifies features subset that reduces the rate of classification error. To prevent stucking at local optimal solutions, a well-designed metaheuristic algorithm should strike a balance between performing local and global search operations.

This proposed method, is based on a modified binary version of MPA that is used to carry out the search procedure. The MPA starts with population initialization using logistic chaotic mapping. This method injects randomness into the initial positions of predators (potential solutions) in the search space. Predators primarily utilize Lévy flights for long-distance exploration. This mimics how real predators search vast areas for prey. When a predator encounters a promising food source (good solution), it switches to Brownian motion. This simulates the fine-tuned search around a potential solution for further improvement. During the exploitation phase, to prevent premature convergence and encourage diversification, a bit flipping mutation operation is introduced. This randomly flips bits in a predator's position (solution), potentially leading to the discovery of new, potentially better solutions. The MPA evaluates each predator's fitness based on the objective functions which is minimizing classification error and number of features selected. By combining these elements, the MPA aims to achieve a good balance between finding new areas in the search space (exploration) and improving existing solutions (exploitation), leading to a diverse set of solutions which means converging on high-quality solutions.

|  |
| --- |
| Algorithm 1: Pseudocode of the proposed Improvement with kNN Classifier |
| Inputs: The population size N, the maximum number of iterations T, Dataset D, feature X and fitness function (fobj) |
| Outputs: The accuracy value for each iteration (best location) likewise sensitivity, specificity. |
| Initialize the population using logistics chaotic mapping |
| **While**  stopping condition is not met **do** |
| Fitness function calculated based on call feature selection method, Call kNN classifier |
| Update prey based on Equation **9** |
| **for** the first half of the populations (i = 1, …, n/2) **do** |
| Update prey based on Equation 10 |
| **for** the other half of the populations (i = n/2, …, n) **do** |
| Update prey based on Equation 11 |
| **end for** |
| **end for** |
| Update prey based on Equation 12 |
| Bit flipping operating based equation (3.11) |
| Accomplish memory saving and Elite update. Applying FADs effect and update based on Equation 13 |
| **end while** |



**Figure 3.1:** Flowchart diagram of the proposed approach

## **Compared Algorithm and Parameter Settings**

All of the experiments were carried out using Google Colab on a PC running Ubuntu 22.04.4 LTS with an Intel(R) Core(TM) i5-8350U CPU @ 1.70GHz, 16 GB of RAM, a 256 GB SSD storage, and other specifications.

The conventional MPA and the proposed improved MPA are used as wrapper feature selection techniques; the result from the both are compared for evaluation. Since the both algorithm involved are wrapper techniques, they incorporate the K-Nearest Neighbor learning algorithm with k=5, which is chosen for its ease of use and quick execution. Both algorithm in use has been tested in identical settings. The parameter settings for the both algorithms in use are shown in Table 3.4.

Training and testing sets were created from the datasets in the ratio of 70:30 respectively. K-fold cross-validation is a perfect choice to prevent the problem of overfitting. The five folds cross-validation was used to divide the training set into five subsets. An internal assessment of the generated feature subsets was carried out using the five subsets. However, in the last phase, the testing set is employed to evaluate the output non-dominated feature subsets that are stored in the repository in terms of performance. As a result, the repository's candidate solutions are assessed using the testing set, and the average is reported. To make sure the results are statistically plausible, the experiments were repeated thirty times. The maximum number of agents in the population and the maximum number of iterations for the improved algorithms are 30 and 1000 respectively.

**Table 3.3:** Parameter settings of both algorithms for all experiments.

|  |  |
| --- | --- |
| **Parameter** | **Value** |
| Population Size | 30 |
| Number of iteration | 1000 |
| Number of runs for algorithms | 30 |
|  | 0.99 |
|  | 0.01 |
| The dimension of initial search space (D) | Number of columns in the input data |
| Search domain (lower and upper bound) | [0,1] |
| Fish Aggregating Device (FAD) | 0.2 |
| P | 0.5 |

## **Performance Metric**

Common evaluation metrics, such as accuracy, the number of features, sensitivity and specificity were used. The definition for some of the evaluation metrics are given below:

The average (mean) of the fitness function values obtained by running algorithm for M times. The Mean fitness function can be calculated by:

(3.12)

Standard Deviation (StdDev): It is used to vary the fitness function value computed from the running algorithm M times. It shows an indicator of the stability of an algorithm. It indicates the measure of the spread of values around the mean. Larger values indicate wandering outcomes while smaller value implies the algorithm almost all the time converges towards the same value which is most preferred. It is also used to check the stability of the obtained results. StdDev can be calculated by:

(3.13)

Accuracy: It is a measure that is defined as a total, correctly identified examples out of all the examples. Accuracy is determined as:

(3.14)

where TP refers to True Positive, which means correctly identified, (FP) refers to incorrectly identified or False Positive, TN is defined as True Negative or correctly rejected. FN refers to False-Negative, which means incorrectly rejected.

Average Accuracy ( AVGACC ): It is a measure that is defined as a average, correctly identified examples out of all the examples. Average Accuracy is calculated by:

(3.15)

where M is the number of run times of the optimization algorithm to select a subset of feature.

Sensitivity: this is the True Positive Rate (TPR). It is the degree to which the learning algorithm is able to identify the data records that have been positively classified. It is calculated by:

(3.16)

Specificity: this is the True Negative Rate (TNR). It defines the proportion of actual negatives that are correctly identified. The learning algorithm's capacity to identify records of data with the negative class is demonstrated by the true negative rate, which is used to measure specificity. It's calculated by:

(3.17)

The number of features: It is a result of the feature selection process of all algorithms. It usually has the minimal most significant number of features which results in the minimum classification error rate.

# **CHAPTER FOUR**

# **RESULT AND DISCUSSION**

## **Introduction**

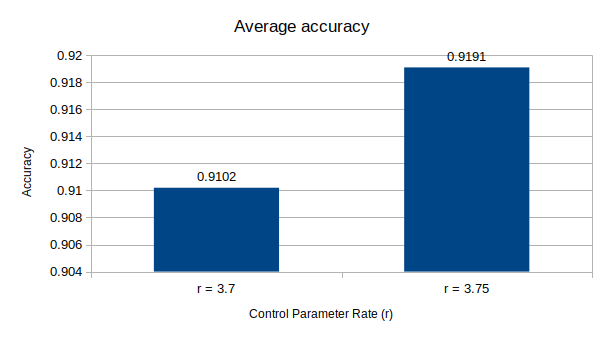
This section discusses the results obtained from the proposed improvement of the marine predator algorithm versus the original marine predator algorithm used for the feature selection task. The improvement of the algorithm is to investigate the effect of logistics chaotic map and also the introduction of the bit flipping operation for balancing exploration and exploitation phase.

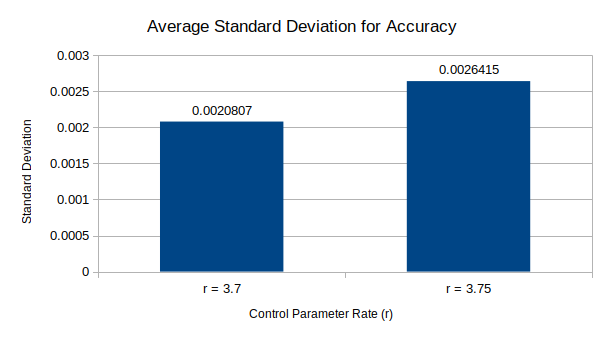
## **Experiment I: Comparison of proposed initialization using different value of ‘r’**

The purpose of this experiment is to investigate the effectiveness of logistics chaotic mapping for population initialization against the random initialization that is mostly used for population initialization. The experiment uses two (2) different values of r which is the control parameter of the equation 2.18. The chaotic behavior of logistic map with r = 3.7 and 3.75 respectively, have been leveraged in various applications, such as pseudo-random bit generation and image encryption (Moysis et al., 2020). The experiment uses four (4) evaluation metrics which are average classification accuracy, average sensitivity, average specificity and average number of features selected. Table 4.1 presents the results of the experiment in terms of average classification accuracy with the standard deviation associated with each accuracy. The table shows the average accuracy of 0.9102 and 0.9191 with standard deviation of 0.002081 and 0.002642 which is also represented in figure 4.1 and figure 4.2 respectively. Using r = 3.75 yield a higher accuracy of 0.9191 with a lower standard deviation of 0.002642 which is the best result obtained for this experiment. Standard deviation as defined in section 3.10, says that, lower standard deviation suggest stability of the algorithm. Considering the number of selected features in this experiment, the table 4.1 presents the results of the experiment in terms of average number of features selected. When the value of r = 3.75, the experiment produces a lower average number of selected features which is 9.4 compared to when the value of r = 3.7 which yield a higher average number of selected features (10.9). Figure 7 shows how using r = 3.75 yield a better result compared to when r = 3.7.

**Table 4.1:** Comparison based on average accuracy, average standard deviation and number of selected features of the two values of ‘r’ used in the study.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Dataset** | Accuracy | | Standard Deviation | | Number of features | Number of features |
| r = 3.7 | r = 3.75 | r = 3.7 | r = 3.75 | r = 3.7 | r = 3.75 |
| BreastEW | 0.983 | **0.986** | 0.001455 | **0.000339** | 19 | **11** |
| CongressEW | 0.975 | **0.98** | 0.000707 | **0** | **3** | 5 |
| IonosphereEW | 0.894 | **0.923** | **0.000465** | 0.000595 | 8 | **5** |
| Sonar | **0.8** | 0.796 | 0.002121 | **0.001414** | 26 | 26 |
| SpectEW | 0.845 | **0.848** | 0.000353 | **0.000212** | 9 | **4** |
| Vote | 0.962 | **0.966** | 0.004243 | **0.003536** | 4 | **3** |
| WaveformEW | 0.897 | **0.9** | **0.000707** | 0.009798 | 19 | 19 |
| Glass | 0.934 | **0.955** | 0.002243 | **0.001973** | 7 | 7 |
| HeartEW | 0.812 | **0.837** | **0.008513** | 0.008548 | 6 | 6 |
| Vowel | 1 | 1 | 0 | 0 | 8 | 8 |
| **AVERAGE** | **0.9102** | **0.9191** | **0.002081** | **0.002642** | **10.9** | **9.4** |

**Figure 4.1:** Average Accuracy using different control parameter rate

**Figure 4.2:** Average Standard Deviation for the Accuracy

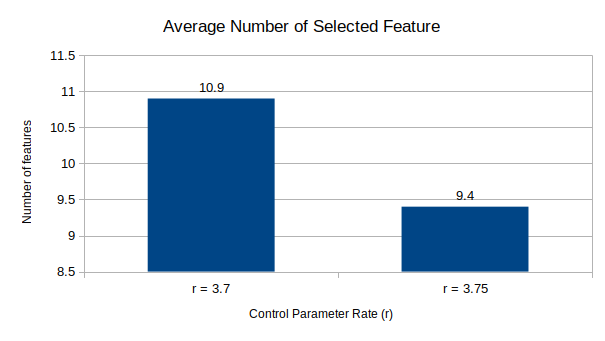
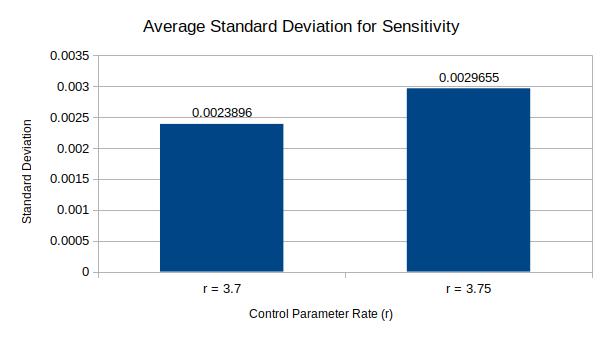
**Figure 4.3:** Average Number of features selected using different control parameter

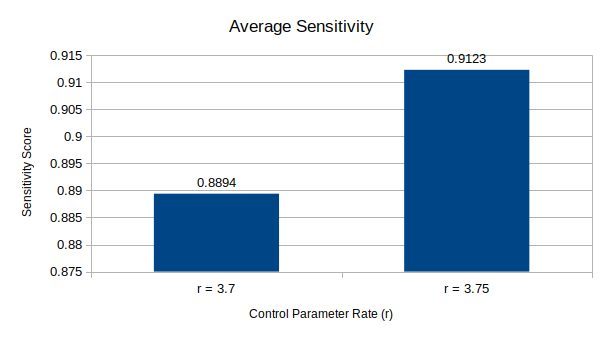
Table 4.2 presents the results of the experiment in terms of average sensitivity and specificity with the standard deviation associated with each dataset. The table shows the average sensitivity of 0.8894 and 0.9123 with standard deviation of 0.002390 and 0.002966 which is also represented in figure 4.4 and figure 4.5 respectively. Using r = 3.75 yield a higher sensitivity of 0.9191 with a higher standard deviation of 0.002642 which is still a good result since the standard deviation is relatively small though it is higher when compared to that of when r = 3.7. Average Sensitivity of 0.9123 means the model was able to actually identify the data that been positively classified with a rate of 91.23% which is also known as True Positive Rate (TPR). With r =3.7, the experiment produce a lower and more statistically significant standard deviation of 0.002390 but the Recall rate is lower (88.94%). The table shows the average specificity of 0.9023 and 0.9267 with standard deviation of 0.003159 and 0.003281 which is also represented in figure 4.6 and figure 4.7 respectively. Using r = 3.75 yield a higher specificity of 0.9267 with a higher standard deviation of 0.003281 which is still a good result since the standard deviation is relatively small though it is higher when compared to that of r when = 3.7.

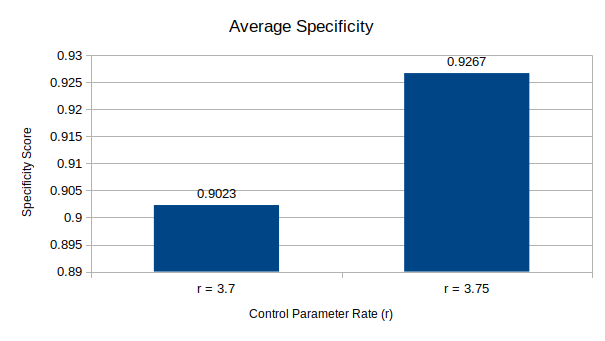
**Table 4.2:** Comparison based on average sensitivity, average specificity and average standard deviation of the two values of ‘r’ used in the study.

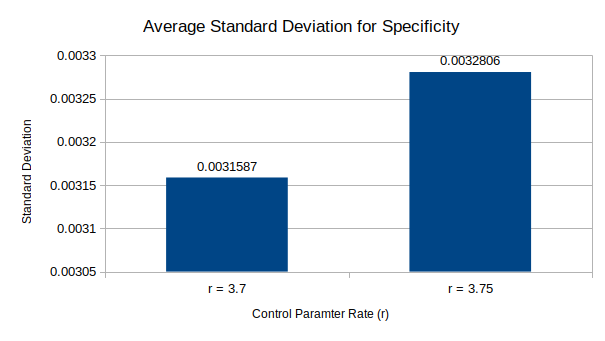
|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Dataset** | Sensitivity | | Standard Deviation | | Specificity | | Standard Deviation | |
| r = 3.7 | r = 3.75 | r = 3.7 | r = 3.75 | r = 3.7 | r = 3.75 | r = 3.7 | r = 3.75 |
| BreastEW | 0.97 | **0.981** | 0.000775 | **0.000498** | 0.974 | **0.991** | **0.000707** | 0.007368 |
| CongressEW | **0.962** | 0.96 | **0.001312** | 0.001537 | 0.964 | **0.98** | **0.006289** | 0.006998 |
| IonosphereEW | 0.853 | **0.92** | **0.003356** | 0.003377 | **0.904** | 0.901 | **0.005168** | 0.005884 |
| Sonar | 0.786 | **0.8** | 0.001922 | **0.001317** | **0.797** | 0.793 | **0.001486** | 0.001567 |
| SpectEW | 0.746 | **0.9** | **0.002002** | 0.002443 | 0.76 | **0.86** | 0.007402 | **0.000877** |
| Vote | 0.961 | **1** | 0.002587 | **0** | 0.95 | **0.973** | **0.002404** | 0.002475 |
| WaveformEW | 0.876 | **0.901** | **0.000707** | 0.003356 | 0.874 | **0.93** | **0.001767** | 0.003394 |
| Glass | 0.947 | **0.963** | **0.002587** | 0.0072 | 1 | 1 | 0 | 0 |
| HeartEW | **0.822** | 0.799 | 0.008648 | **0.008513** | 0.8 | **0.839** | 0.006364 | **0.004243** |
| Vowel | **0.971** | 0.899 | **0** | 0.001414 | 1 | 1 | 0 | 0 |
| **AVERAGE** | **0.8894** | **0.9123** | **0.0023896** | **0.0029655** | **0.9023** | **0.9267** | **0.0031587** | **0.0032806** |

Average Specificity of 0.9267 means the model was able to actually identify the data that been negatively classified with a rate of 92.67% which is also known as True Negative Rate. With r =3.7, the experiment produce a lower and more statistically significant standard deviation of 0.003159 but the Recall rate is lower at 90.23%.

**Figure 4.5:** Average Standard Deviation for Sensitivity

**Figure 4.4:** Average Sensitivity using different control parameter rate

**Figure 4.6:** Average Specificity using different control parameter rate

**Figure 4.7:** Average Standard Deviation for Specificity

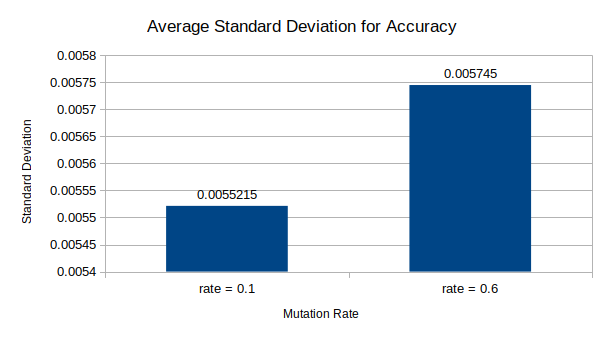
The result from Experiment I shows how introducing logistic chaotic map to the initial population generation enhances the exploration of the search space. Higher rates were observed from the early stage of the iteration which is where the exploration actually happens. In conclusion, the proposed method of using logistics chaotic mapping to initialize the population with a control parameter of 3.75 is selected as the best algorithm because of its higher statistical results.

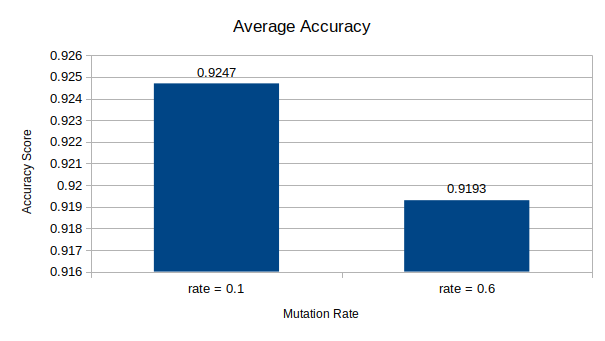
## **Experiment II: Comparison of proposed introduction of random bit flipping operation using different mutation rate**

The purpose of this experiment is to introduce mutation operation in the optimization phases of the standard MPA. It is to be use as a local search of the algorithm which is where intensification happen, that is the exploitation phase of the algorithm. The experiment uses two (2) different mutation rate. The value of the mutation rate is a very important factor considered during the optimization phase. A low mutation rate might not introduce enough diversity while a high rate can also destroy good solutions. A study by Yousri et al., (2022) uses a mutation rate of 0.1 and it achieve a good result in managing the exchange of energy between microgrid elements based on multi-objective enhanced marine predators algorithm. The study did enhanced the standard MPA using the mutation rate (0.1) to minimize the cost and emission while also providing very high profit. Also, in a study by Chen et al., (2022), mutation rate of 0.1, 0.6 and 0.9 respectively were used to improve the predator position updating mechanism and improve the MPA’s global search ability. The study stated 0.1 and 0.6 as having a better performance which form the basis for this experiment. This experiment introduce the random bit flipping operation as a form of enhancing the exploitation phase. The experiment also uses four (4) evaluation metrics which are average classification accuracy, average sensitivity, average specificity and average number of features selected. Table 4.3 presents the results of the experiment in terms of average classification accuracy with the standard deviation associated with each dataset. The table also present the number of selected features for each dataset and the average number of selected features. The table shows the average accuracy of 0.9247 and 0.9193 with standard deviation of 0.005522 and 0.005745 which is also represented in figure 4.8 and figure 4.9 respectively. Using a mutation rate of 0.1 yield a higher accuracy of 0.9247 with a lower standard deviation of 0.005522 which is the best result obtained for this experiment. Considering the metrics of average number of selected features in this experiment, with a mutataion rate of 0.1, the average number of selected features is 9.4 which is lower when compared to the value of 10.4 obtained when the mutation rate is 0.6. Figure 4.10 shows the average number of selected features using the two different mutation rate.

**Table 4.3:** Comparison based on average accuracy, average standard deviation and number of selected features of the two mutation rates used in the study.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Dataset** | **Accuracy** | | **Standard Deviation** | | **Number of Features** | **Number of Features** |
| **rate**  **= 0.1** | **rate**  **= 0.6** | **rate**  **= 0.1** | **rate**  **= 0.6** | **rate**  **= 0.1** | **rate**  **= 0.6** |
| BreastEW | **0.996** | 0.992 | **0.003196** | 0.003673 | 19 | **11** |
| CongressEW | **0.982** | 0.977 | **0.004243** | 0.00459 | 7 | **5** |
| IonosphereEW | **0.96** | 0.891 | 0.008425 | **0.0084** | **6** | 16 |
| Sonar | 0.759 | **0.771** | **0.006169** | 0.007157 | 26 | **13** |
| SpectEW | 0.864 | **0.883** | **0.009213** | 0.009698 | **5** | 7 |
| Vote | **0.981** | 0.966 | **0.006264** | 0.006353 | 6 | 6 |
| WaveformEW | **0.908** | 0.901 | 0.007829 | **0.007157** | 19 | 19 |
| Glass | 0.959 | **0.99** | **0.003628** | 0.004069 | 1 | 1 |
| HeartEW | **0.838** | 0.822 | **0.006248** | 0.006353 | 7 | 7 |
| Vowel | 1 | 1 | 0 | 0 | 8 | 8 |
| **AVERAGE** | **0.9247** | **0.9193** | **0.0055215** | **0.005745** | **10.4** | **9.3** |

**Figure 4.9:** Average Standard Deviation for Accuracy

**Figure 4.8:** Average Accuracy using different mutation rate

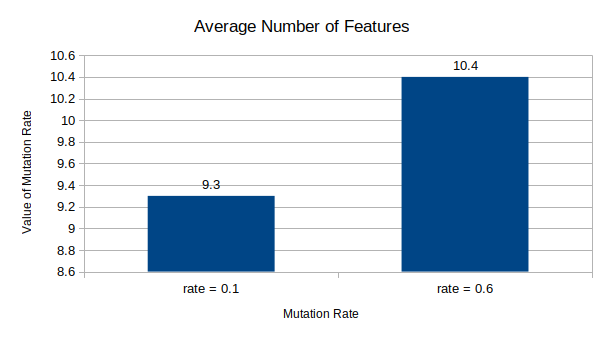
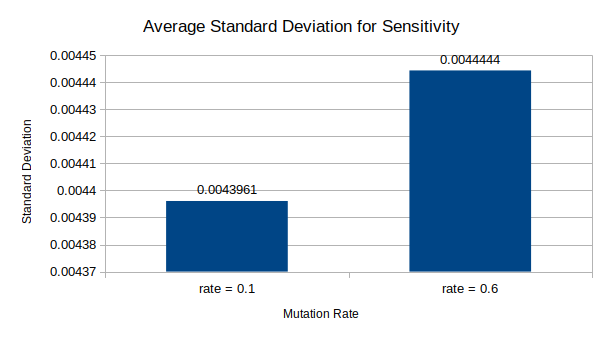
**Figure 4.10:** Average number of features

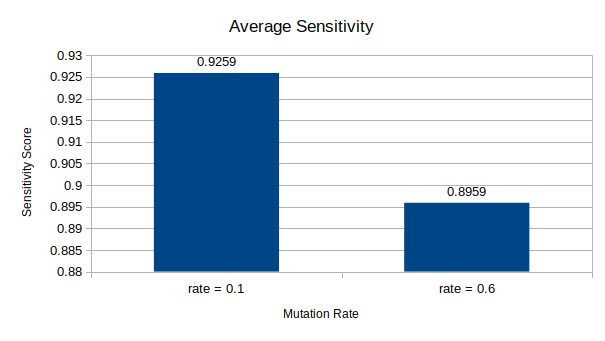
Table 4.4 presents the results of the experiment in terms of average sensitivity with the standard deviation associated with each dataset. The table shows the average sensitivity of 0.9259 and 0.8959 with standard deviation of 0.004396 and 0.004444 which is also represented in figure 4.11 and figure 4.12 respectively. Using mutation rate of 0.1 yielded a higher sensitivity of 0.9259 with a lower standard deviation of 0.004396 which is still a good result since the standard deviation is relatively small. Average Sensitivity of 0.9259 means the model was able to actually identify the data that been positively classified with a rate of 92.59% which is also known as the True Positive Rate (TPR). With mutation rate of 0.6, the experiment produce a recall rate of 89.59%. The table also shows the average specificity of 0.9198 and 0.912 with standard deviation of 0.003814 and 0.006563 which is also represented in figure 4.13 and figure 4.14 respectively. Using mutation rate of 0.1 also yielded a slightly higher specificity of 0.9198 against 0.912 that resulted from a mutation rate of 0.6. Mutation rate of 0.1 also has the better significant since it has a lower standard deviation when compared with that of 0.6. The standard deviation obtained for mutation rate of 0.1 is 0.003814, which is lower than 0.006563, thereby having a higher significant than the other.

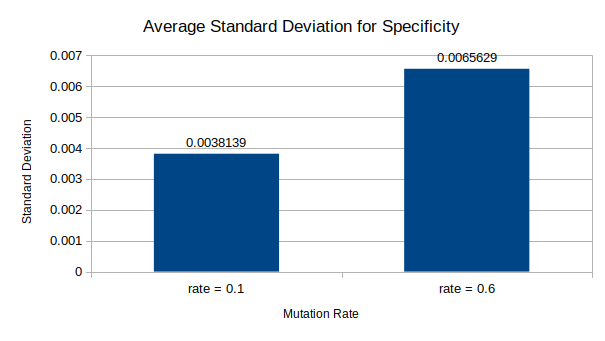
**Table 4.4:** Comparison based on average sensitivity, average specificity and average standard deviation of the two values of mutation used in the study.

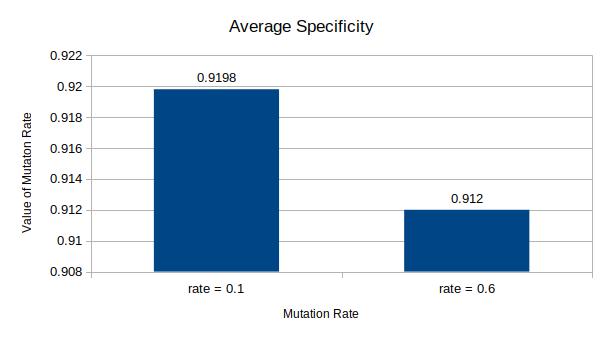
|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Dataset** | **Sensitivity** | | **Standard Deviation** | | **Specificity** | | **Standard Deviation** | |
| **rate**  **= 0.1** | **rate**  **= 0.6** | **rate**  **= 0.1** | **rate**  **= 0.6** | **rate**  **= 0.1** | **rate**  **= 0.6** | **rate**  **= 0.1** | **rate**  **= 0.6** |
| BreastEW | **0.987** | 0.977 | **0.001878** | 0.002588 | **0.996** | 0.985 | **0.003072** | 0.004536 |
| CongressEW | 0.96 | **0.966** | 0.004275 | **0.001131** | **0.99** | 0.966 | **0** | 0.007287 |
| IonosphereEW | 0.786 | **0.846** | **0.000921** | 0.004536 | **0.95** | 0.909 | 0.002721 | **0.002697** |
| Sonar | **0.881** | 0.761 | 0.023246 | **0.019647** | **0.8** | 0.796 | **0.003674** | 0.027828 |
| SpectEW | **0.93** | 0.765 | 0.002551 | **0.001234** | 0.782 | **0.814** | 0.001682 | **0.001133** |
| Vote | **1** | 0.963 | **0** | 0.004515 | **0.983** | 0.954 | **0.00779** | 0.008748 |
| WaveformEW | **0.901** | 0.879 | 0.002324 | **0.001964** | 0.878 | **0.91** | **0.0022** | 0.002782 |
| Glass | **0.973** | 0.97 | **0.00452** | 0.004714 | **0.97** | 0.957 | 0.004817 | **0.001414** |
| HeartEW | **0.841** | 0.832 | 0.004246 | **0.004115** | **0.859** | 0.835 | 0.008748 | **0.00779** |
| Vowel | 1 | 1 | 0 | 0 | 0.99 | **0.994** | 0.003435 | **0.001414** |
| **AVERAGE** | **0.9259** | **0.8959** | **0.0043961** | **0.0044444** | **0.9198** | **0.912** | **0.0038139** | **0.0065629** |

Average Specificity of 0.9198 means the model was able to actually identify the data that been negatively classified with a rate of 91.98% which is the precision rate for the algorithm. With a mutation rate of 0.6, the experiment produce a little lower precision rate of 91.2% and a higher significant standard deviation of 0.006563.

**Figure 4.12:** Average Standard Deviation for Sensitivity

**Figure 4.11:** Average Sensitivity for different mutatation rate

**Figure 4.14:** Average Standard Deviation for Specificity

**Figure 4.13:** Average Specificity for different mutation rate

In conclusion, the proposed approach of introducing the random bit flipping operation to enhance the exploitation phase with a mutation rate of 0.1 is selected as the best algorithm because of its higher statistical results.

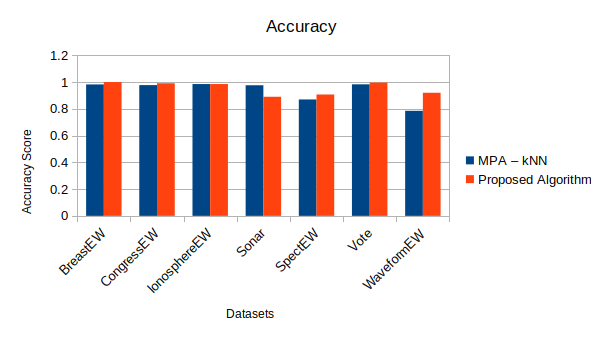
## **Experiment III: Proposed algorithm versus Standard MPA**

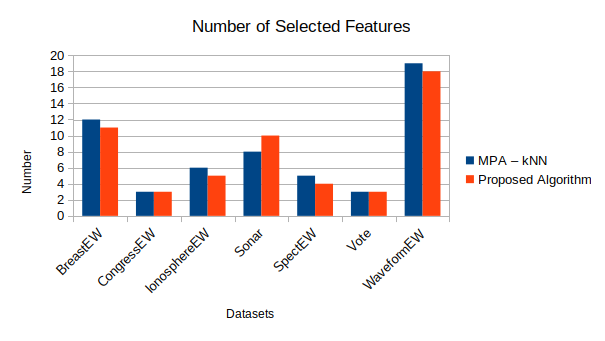
From Experiment I, it is concluded the best performance was obtained using control parameter rate of 3.75 in the logistics chaotic map while in Experiment II, the best performance was obtained using mutation rate of 0.1. The purpose of this experiment is to use the best value obtained from the both experiment and use the value as parameter into the proposed improvement to the MPA. Experiment I would enhance the diversity of the initial population while the Experiment II would be enhanciing the exploitation phase of the MPA, thereby, this experiment would help to balance the exploration and exploitation phase of the MPA. The results from the experiment would be evaluated against the work of Elminaam et al., (2021) using four (4) evaluation metrics which are average classification accuracy, average sensitivity, average specificity and average number of features selected.

Table 4.5 presents the results of the experiment in terms of average classification accuracy with the standard deviation associated with each dataset. Likewise, the table also present the number of features obtained for each dataset and the average number of features. Also, the result of Elminaam et al., (2021) is presented in the table for comparison. The table shows the average accuracy of 0.9555 for the proposed algorithm with standard deviation of 0.003083 which is better when compared with the stanadard MPA which have an average accuracy of 0.9368 with a standard deviation of 0.002785. From the table, it is observed that the proposed algorithm performed better in the WaveformEW dataset with a higher accuracy of 92% against the 78.5% obtained using the standard MPA based on percentage increase. The proposed algorithm also obtained a reduced number of selected features for the same dataset which 18 against the 19 obtained by the standard MPA. The proposed algorithm performed worst with the Sonar dataset as it obtained a lower accuracy of 89% against the 97.6% obtained by the standard MPA. It also recorded an increase in the number of selected feature, which is 10 against the 8 obtained by the standard MPA. The proposed algorithm also obtained 100% accuracy against 98.2% in the BreastEW dataset with a lesser feature of 11 against 12 of the MPA. For other datasets, the proposed algorithm obtained a higher accuracy in CongressEW (99.1% against 97.7%), SpectEW (90.7% against 87%) and Vote (99.6% against 98.3%). Figure 4.15 shows the performance of each dataset using the proposed algorithm against the standard MPA based on the accuracy metric. The standard deviation which is use to show the significant of the result make the result obtained in Ionosphere dataset (98.5%) to be better than that of the MPA because of the lower standard deviation obtained, which is 0.001414 against 0.008326. On the number of features, the proposed algorithm performed better in the datasets of BreastEW, IonosphereEW, SpectEW and WaveformEW. The algorithm also selected equal number of features for the CongressEW and Vote dataset (3 each).

**Table 4.5:** Comparison of the proposed algorithm performance against the Standard MPA based on average accuracy, average standard deviation and number of selected features

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Dataset** | Accuracy | | Standard Deviation | | Number of features | |
| MPA – kNN | Proposed Algorithm | MPA – kNN | Proposed Algorithm | MPA – kNN | Proposed Algorithm |
| BreastEW | 0.982 | **1** | 0.002263 | **0** | 12 | **11** |
| CongressEW | 0.977 | **0.991** | **0.000707** | 0.001023 | **3** | 3 |
| IonosphereEW | 0.985 | **0.985** | 0.008326 | **0.001414** | 6 | **5** |
| Sonar | **0.976** | 0.89 | **0.002828** | 0.003092 | **8** | 10 |
| SpectEW | 0.87 | **0.907** | **0.000707** | 0.007402 | 5 | **4** |
| Vote | 0.983 | **0.996** | 0.003323 | **0.001654** | 3 | 3 |
| WaveformEW | 0.785 | **0.92** | **0.001344** | 0.006998 | 19 | **18** |
| **AVERAGE** | **0.936** | **0.9555** | **0.002785** | **0.003083** | **8** | **7.7** |

**Figure 4.15:** Accuracy of the Proposed Algorithm Vs MPA for each dataset

**Figure 4.16:** Number of selected features for the Proposed Algorithm Vs MPA

The algorithm still perform poorly in the Sonar dataset when compared with the standard MPA as it selected a higher number of features and it has a lower accuracy. Figure 4.16 shows the results of the experiment based on the number of selected features. It shows the number selected by the proposed algorithm against the standard MPA.

Table 4.6 presents the results of the experiment in terms of average classification accuracy with the standard deviation associated with the extra dataset. The table also present the number of features obtained for each dataset and the average number of features. From the table, it is seen that the algorithm obtains an accuracy of 100% in Glass dataset with number of features selected as 3. It also obtains an accuracy of 100% in Vowel dataset with number of features selected as 5. For the HeartEW dataset, it showed an accuracy of 85.6% having standard deviation of 0.005195 while the number of selected feature is 5.

**Table 4.6:** Proposed algorithm performance based on average accuracy, average standard deviation and number of selected features for the extra datasets

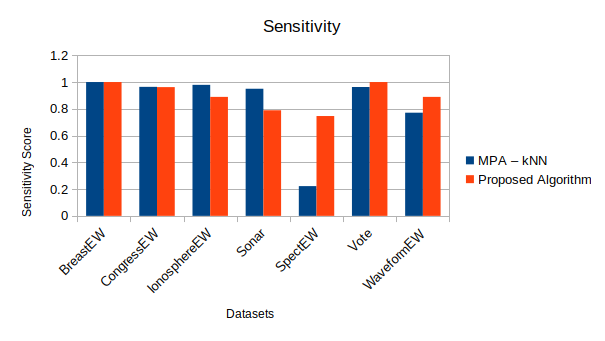
|  |  |  |  |
| --- | --- | --- | --- |
| **Dataset** | Accuracy | Standard Deviation | Number of features |
| Proposed Algorithm | Proposed Algorithm | Proposed Algorithm |
| Glass | **1** | **0** | **3** |
| HeartEW | **0.856** | **0.005195** | **6** |
| Vowel | **1** | **0** | **5** |
| **AVERAGE** | **0.952** | **0.001731** | **4.6** |

Table 4.7 presents the results of the experiment in terms of average sensitivity and specificity with the standard deviation associated with each dataset. The table shows the average sensitivity of 0.835 with average standard deviation of 0.001767 for the MPA while the proposed algorithm obtained average sensitivity of 0.896 with average standard deviation of 0.003634. The proposed algorithm score a higher average sensitivity and lower average standard deviation which make it a good algorithm to compare against the standard MPA. Average Sensitivity of 0.896 for the proposed algorithm means the model was able to actually identify the data that been positively classified with a rate of 89.6% agaisnt 83.5% of the MPA. This is also known as the True Positive Rate (TPR). Figure 4.17 shows the performance of each dataset using the proposed algorithm against the standard MPA based on the accuracy metric. The table shows the average specificity of 0.974 with average standard deviation of 0.000808 for the MPA while the proposed algorithm obtained average specificity of 0.0.902 with average standard deviation of 0.004973. The proposed algorithm score a lower average specificity and higher average standard deviation which means the standard MPA performs better than the proposed algorithm based on the specificity metric. The standard MPA scored a higher average specificity of 0.974 and a lower specificity of 0.000808. Average Specificity of 0.974 for the standard MPA means the model was able to actually identify the data that have been negatively classified with a rate of 97.% against the 90.2% obtained by the proposed algorithm. This is known as the True Negative Rate (TNR). For most of the dataset like CongressEW, IonosphereEW, Sonar, SpectEW and Vote dataset, the standard MPA performed better than the proposed algorithm by having a 100% precision rate.

**Table 4.7:** Comparison of the proposed algorithm performance against the Standard MPA based on sensitivity, specificity and average standard deviation and number of selected features.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Dataset** | Sensitivity | | Standard Deviation | | Specificity | | Standard Deviation | |
| MPA – kNN | Proposed Algorithm | MPA – kNN | Proposed Algorithm | MPA – kNN | Proposed Algorithm | MPA – kNN | Proposed Algorithm |
| BreastEW | 1 | 1 | 0 | 0 | 0.951 | **0.979** | **0.003536** | 0.004243 |
| CongressEW | **0.964** | 0.962 | **0.001414** | 0.006065 | **1** | 0.956 | **0** | 0.001328 |
| IonosphereEW | **0.979** | 0.889 | 0.004243 | **0.001118** | **1** | 0.932 | **0** | 0.008833 |
| Sonar | **0.95** | 0.789 | **0.002475** | 0.003335 | **1** | 0.817 | **0** | 0.004007 |
| SpectEW | 0.222 | **0.746** | 0.002121 | 0.007946 | **1** | 0.795 | **0** | 0.008771 |
| Vote | 0.963 | **1** | 0.001414 | **0** | **1** | 0.953 | **0** | 0.000707 |
| WaveformEW | 0.771 | **0.889** | **0.000707** | 0.00698 | 0.873 | **0.884** | **0.002121** | 0.006926 |
| AVERAGE | 0.835 | 0.896 | 0.001767 | 0.003634 | 0.974 | 0.902 | 0.000808 | 0.004973 |

The proposed algorithm did have a better performance in two (2) of the dataset which are BreastEW and WaveformEW by having precision rate of 97.9% and 88.4% against 95.1% and 87.3%. Figure 4.18 shows the performance of each dataset using the proposed algorithm against the standard MPA based on the accuracy metric.

**Figure 4.17:** Average sensitivity for each dataset using the Proposed Algorithm Vs MPA

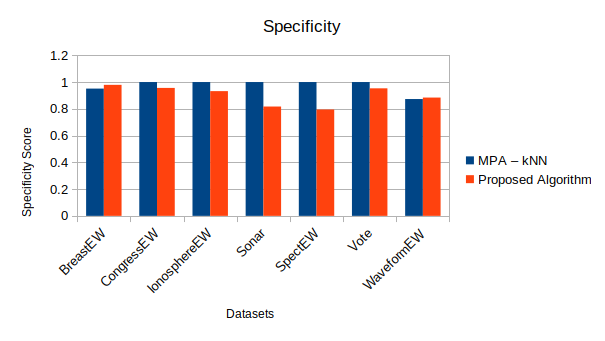
**Figure 4.18:** Average Specificity for each dataset using the Proposed Algorithm Vs MPA

Table 4.8 presents the results of the experiment in terms of average sensitivity and average specifity with the standard deviation for the extra dataset. From the table, it is seen that the algorithm obtained 100% sensitivity and 100% specificity in the Glass dataset. It also obtained 100% sensitivity and 100% specificity in Vowel dataset. This means the precision rate and recall rate for both dataset is 100%, i.e the algorithm has TNR and TPR of 100%. For the HeartEW dataset, it showed sensitivity of 0.841 and having standard deviation of 0.005168 while the specificity is 0.859 with standard deviation of 0.005884.

**Table 4.8:** Proposed algorithm performance based on average sensitivity, average specificity and average standard deviation for the extra datasets

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Dataset** | Sensitivity | Standard Deviation | Specificity | Standard Deviation |
| Proposed Algorithm | Proposed Algorithm | Proposed Algorithm | Proposed Algorithm |
| Glass | **1** | **0** | **1** | **0** |
| HeartEW | **0.841** | **0.005168** | **0.859** | **0.005884** |
| Vowel | **1** | **0** | **1** | **0** |
| AVERAGE | 0.947 | 0.001722 | 0.953 | 0.001961 |

The results from the various tables presented above shows that there is a statistically significant difference between the standard MPA and the proposed algorithms in some of the datasets with a few showing otherwise. It also reemphasize the fact that, there is no single method that does well for all datasets. The reported results confirm the importance of the FS phase as the proposed algorithm attain better average classification accuracy in some of the datasets which can be attributed to the decrease in the dimensionality of these datasets as where higher classification accuracy was obtained, there was reduction in the number of selected features even though it was minimal. The proposed algorithm ranked as the first in average Accuracy, Sensitivity and obtained the lower number of selected features. Finally, based on the No-Free Lunch theorem which states that no one optimization algorithm performs best on all datasets, the proposed algorithm can be seen as a worthy competitor for feature selection problems.

# **CHAPTER FIVE**

# **SUMMARY, CONCLUSION AND RECOMMENDATION**

## **SUMMARY**

This dissertation introduces a variation of the Marine Predator Algorithm (MPA) for feature selection using machine learning datasets. This approach emphasizes improved exploration and exploitation capabilities during the optimization process. The key modifications are in the initialization phase and introduction of the bit flipping operation. These modifications aim to achieve a balance between exploration and exploitation. This modification strives to outperform the standard MPA. The algorithm minimizes the number of selected features as well as improving the rate of accuracy. The classifier used is the K-Nearest Neighbor (KNN) algorithm. To help with the data imbalance and overfitting problem, each of the datasets was divided into training and testing parts using five folds cross-validation. The proposed algorithm was evaluated on a number of publicly benchmark datasets and compared with the MPA. The results shows that the proposed algorithm obtains a competitive and high-quality solutions that outperformed the MPA in most of the test cases. Specifically, the proposed algorithm achieved an equivalent or higher accuracy (i.e lower classification error rate) with fewer selected features compared to the MPA in most of the datasets. This demonstrates the algorithm's ability to effectively search the high-dimensional feature space and converge towards the optimal set of minimal feature subsets.

## **5.2 CONCLUSION**

In conclusion, this variation of the Marine Predator Algorithm (MPA), incorporating logistic chaotic mapping and bit flipping mutation, proves to be valuable for feature selection in Machine Learning Classification task. The use of logistic chaotic mapping for initialization injects controlled randomness, leading to a diverse initial population. Additionally, the introduction of bit flipping mutation during the final phase enhances exploration and prevents premature convergence, allowing the MPA to discover a wider range of potentially useful features. The utilization of K-Nearest Neighbors (KNN) as the classifier ensures the practical applicability of the proposed approach. KNN's simplicity and effectiveness make it well-suited for feature selection tasks. The combination of improved exploration, exploitation, and a well-suited classifier leads to effective feature selection with reduced classification errors.

## **RECOMMENDATIONS**

Based on the promising performance of the proposed improvement to the MPA observed in this dissertation, the following are recommendation for future directions:

1. Investigate the use of other classification algorithms like Support Vector Machines (SVM), Decision Trees, Naive Bayes and Random Forests into the algorithm for feature selection.
2. Investigate the effect of introducing other chaotic mapping such as circle map, tent map, duffing map, gauss map and exponential map to enhance the initial population of the MPA.
3. Investigate the effect of introducing other method of initial population generation like Latin hypercube sampling (LHS), Probability Distributions, Quasirandom methods and hybridization with other metaheuristic algorithm into the MPA.
4. Apply the proposed approach for gene selection, intrusion detection and analysis on other high-dimensional datasets.
5. Explore the addition of more objectives like model interpretability along with model accuracy and number of features.

# REFERENCE

Abd Elaziz, M., Ewees, A. A., Yousri, D., Abualigah, L., & Al-qaness, M. A. A. (2022). Modified marine predators algorithm for feature selection: Case study metabolomics. *Knowledge and Information Systems*, *64*(1), 261–287. https://doi.org/10.1007/s10115-021-01641-w

Abdel-Basset, M., El-Shahat, D., Chakrabortty, R. K., & Ryan, M. (2021). Parameter estimation of photovoltaic models using an improved marine predators algorithm. *Energy Conversion and Management*, *227*, 113491.

Abu Khurma, R., Aljarah, I., Sharieh, A., Abd Elaziz, M., Damaševičius, R., & Krilavičius, T. (2022). A Review of the Modification Strategies of the Nature Inspired Algorithms for Feature Selection Problem. *Mathematics*, *10*(3), 464. https://doi.org/10.3390/math10030464

Abualigah, L., Diabat, A., Mirjalili, S., Abd Elaziz, M., & Gandomi, A. H. (2021). The arithmetic optimization algorithm. *Computer Methods in Applied Mechanics and Engineering*, *376*, 113609.

Afrabandpey, H., Ghaffari, M., Mirzaei, A., & Safayani, M. (2014). A novel bat algorithm based on chaos for optimization tasks. *2014 Iranian Conference on Intelligent Systems (ICIS)*, 1–6.

Agrawal, P., Abutarboush, H. F., Ganesh, T., & Mohamed, A. W. (2021). Metaheuristic Algorithms on Feature Selection: A Survey of One Decade of Research (2009-2019). *IEEE Access*, *9*, 26766–26791. https://doi.org/10.1109/ACCESS.2021.3056407

Aha, D. W., Kibler, D., & Albert, M. K. (1991). Instance-based learning algorithms. *Machine Learning*, *6*(1), 37–66. https://doi.org/10.1007/BF00153759

Ali, S. I., & Shahzad, W. (2012). A feature subset selection method based on conditional mutual information and ant colony optimization. *International Journal of Computer Applications*, *60*(11), 5–10.

Al-qaness, M. A., Ewees, A. A., Fan, H., Abualigah, L., & Abd Elaziz, M. (2022). Boosted ANFIS model using augmented marine predator algorithm with mutation operators for wind power forecasting. *Applied Energy*, *314*, 118851.

Alrasheedi, A. F., Alnowibet, K. A., Saxena, A., Sallam, K. M., & Mohamed, A. W. (2022). Chaos Embed Marine Predator (CMPA) Algorithm for Feature Selection. *Mathematics*, *10*(9), 1411. https://doi.org/10.3390/math10091411

Badillo, S., Banfai, B., Birzele, F., Davydov, I. I., Hutchinson, L., Kam-Thong, T., Siebourg-Polster, J., Steiert, B., & Zhang, J. D. (2020). An Introduction to Machine Learning. *Clinical Pharmacology & Therapeutics*, *107*(4), 871–885. https://doi.org/10.1002/cpt.1796

Balakrishnan, K., Dhanalakshmi, R., & Mahadeo Khaire, U. (2022). Excogitating marine predators algorithm based on random opposition-based learning for feature selection. *Concurrency and Computation: Practice and Experience*, *34*(4), e6630.

Bartumeus, F., Da Luz, M. G. E., Viswanathan, G. M., & Catalan, J. (2005). ANIMAL SEARCH STRATEGIES: A QUANTITATIVE RANDOM-WALK ANALYSIS. *Ecology*, *86*(11), 3078–3087. https://doi.org/10.1890/04-1806

Beheshti, Z. (2022). BMPA-TVSinV: A Binary Marine Predators Algorithm using time-varying sine and V-shaped transfer functions for wrapper-based feature selection. *Knowledge-Based Systems*, *252*, 109446. https://doi.org/10.1016/j.knosys.2022.109446

Bellman, R. (1957). In *Dynamic Programming*. Princeton University Press.

Berisha, V., Krantsevich, C., Hahn, P. R., Hahn, S., Dasarathy, G., Turaga, P., & Liss, J. (2021). Digital medicine and the curse of dimensionality. *Npj Digital Medicine*, *4*(1), 153. https://doi.org/10.1038/s41746-021-00521-5

Bradley, P. S., & Mangasarian, O. L. (1998). Feature selection via concave minimization and support vector machines. *ICML*, *98*, 82–90.

Brezočnik, L., Fister, I., & Podgorelec, V. (2018). Swarm Intelligence Algorithms for Feature Selection: A Review. *Applied Sciences*, *8*(9), 1521. https://doi.org/10.3390/app8091521

Cadenas, J. M., Garrido, M. C., & MartíNez, R. (2013). Feature subset selection filter–wrapper based on low quality data. *Expert Systems with Applications*, *40*(16), 6241–6252.

Chaudhuri, A., & Sahu, T. P. (2021). Feature selection using Binary Crow Search Algorithm with time varying flight length. *Expert Systems with Applications*, *168*, 114288.

Chen, L., Hao, C., & Ma, Y. (2022). A Multi-Disturbance Marine Predator Algorithm Based on Oppositional Learning and Compound Mutation. *Electronics*, *11*(24), 4087. https://doi.org/10.3390/electronics11244087

Civicioglu, P. (2013). Backtracking search optimization algorithm for numerical optimization problems. *Applied Mathematics and Computation*, *219*(15), 8121–8144.

Das, S. (2001). Filters, wrappers and a boosting-based hybrid for feature selection. *Icml*, *1*, 74–81.

Dhal, K. G., Das, A., Ray, S., Gálvez, J., & Das, S. (2020). Nature-Inspired Optimization Algorithms and Their Application in Multi-Thresholding Image Segmentation. *Archives of Computational Methods in Engineering*, *27*(3), 855–888. https://doi.org/10.1007/s11831-019-09334-y

Diao, R., & Shen, Q. (2015). Nature inspired feature selection meta-heuristics. *Artificial Intelligence Review*, *44*(3), 311–340. https://doi.org/10.1007/s10462-015-9428-8

Dorigo, M., & Di Caro, G. (1999). Ant colony optimization: A new meta-heuristic. *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406)*, 1470–1477. https://doi.org/10.1109/CEC.1999.782657

El Naqa, I., & Murphy, M. J. (2015). What Is Machine Learning? In I. El Naqa, R. Li, & M. J. Murphy (Eds.), *Machine Learning in Radiation Oncology: Theory and Applications* (pp. 3–11). Springer International Publishing. https://doi.org/10.1007/978-3-319-18305-3\_1

Elminaam, D. S. A., Nabil, A., Ibraheem, S. A., & Houssein, E. H. (2021). An Efficient Marine Predators Algorithm for Feature Selection. *IEEE Access*, *9*, 60136–60153. https://doi.org/10.1109/ACCESS.2021.3073261

Emami, H. (2021). *Anti coronavirus optimization algorithm: A socio-inspired meta-heuristic for numerical and engineering optimization problems*.

Faramarzi, A., Heidarinejad, M., Mirjalili, S., & Gandomi, A. H. (2020). Marine Predators Algorithm: A nature-inspired metaheuristic. *Expert Systems with Applications*, *152*, 113377. https://doi.org/10.1016/j.eswa.2020.113377

Gabis, A. B., Meraihi, Y., Mirjalili, S., & Ramdane-Cherif, A. (2021). A comprehensive survey of sine cosine algorithm: Variants and applications. *Artificial Intelligence Review*, *54*(7), 5469–5540.

Geem, Z. W., Kim, J. H., & Loganathan, G. V. (2001). A new heuristic optimization algorithm: Harmony search. *Simulation*, *76*(2), 60–68.

Ghosh, M., Guha, R., Alam, I., Lohariwal, P., Jalan, D., & Sarkar, R. (2019). Binary genetic swarm optimization: A combination of GA and PSO for feature selection. *Journal of Intelligent Systems*, *29*(1), 1598–1610.

Glover, F. (1989). Tabu Search—Part I. *ORSA Journal on Computing*, *1*(3), 190–206. https://doi.org/10.1287/ijoc.1.3.190

Golay, J., & Kanevski, M. (2017). Unsupervised feature selection based on the Morisita estimator of intrinsic dimension. *Knowledge-Based Systems*, *135*, 125–134. https://doi.org/10.1016/j.knosys.2017.08.009

Hao, J., & Ho, T. K. (2019). Machine Learning Made Easy: A Review of *Scikit-learn* Package in Python Programming Language. *Journal of Educational and Behavioral Statistics*, *44*(3), 348–361. https://doi.org/10.3102/1076998619832248

Hashim, F. A., Houssein, E. H., Mabrouk, M. S., Al-Atabany, W., & Mirjalili, S. (2019). Henry gas solubility optimization: A novel physics-based algorithm. *Future Generation Computer Systems*, *101*, 646–667.

Heidari-Bateni, G., & McGillem, C. D. (1994). A chaotic direct-sequence spread-spectrum communication system. *IEEE Transactions on Communications*, *42*(2/3/4), 1524–1527. https://doi.org/10.1109/TCOMM.1994.582834

Hira, Z. M., & Gillies, D. F. (2015). A review of feature selection and feature extraction methods applied on microarray data. *Advances in Bioinformatics*, *2015*.

Holland, J. H. (1992). Genetic Algorithms. *Scientific American*, *267*(1), 66–72. https://doi.org/10.1038/scientificamerican0792-66

Huan Liu, & Setiono, R. (1995). Chi2: Feature selection and discretization of numeric attributes. *Proceedings of 7th IEEE International Conference on Tools with Artificial Intelligence*, 388–391. https://doi.org/10.1109/TAI.1995.479783

Humphries, N. E., Queiroz, N., Dyer, J. R. M., Pade, N. G., Musyl, M. K., Schaefer, K. M., Fuller, D. W., Brunnschweiler, J. M., Doyle, T. K., Houghton, J. D. R., Hays, G. C., Jones, C. S., Noble, L. R., Wearmouth, V. J., Southall, E. J., & Sims, D. W. (2010). Environmental context explains Lévy and Brownian movement patterns of marine predators. *Nature*, *465*(7301), 1066–1069. https://doi.org/10.1038/nature09116

Jia, H., Sun, K., Li, Y., & Cao, N. (2022). Improved marine predators algorithm for feature selection and SVM optimization. *KSII Transactions on Internet and Information Systems (TIIS)*, *16*(4), 1128–1145.

Jia, W., Sun, M., Lian, J., & Hou, S. (2022). Feature dimensionality reduction: A review. *Complex & Intelligent Systems*, *8*(3), 2663–2693. https://doi.org/10.1007/s40747-021-00637-x

Jiang, Y., Huang, S., Wu, H., Yang, Z., Nai, W., & Li, D. (2022). Ridge Regression Based on t-Distribution Marine Predators Algorithm. *2022 IEEE 6th Information Technology and Mechatronics Engineering Conference (ITOEC)*, 1900–1904. https://doi.org/10.1109/ITOEC53115.2022.9734555

Jie, H. (2008). Survey on feature dimension reduction for high-dimensional data. *Application Research of Computers*, *25*(9), 2601–2606.

Jovic, A., Brkic, K., & Bogunovic, N. (2015). A review of feature selection methods with applications. *2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, 1200–1205. https://doi.org/10.1109/MIPRO.2015.7160458

Kanso, A., & Smaoui, N. (2009). Logistic chaotic maps for binary numbers generations. *Chaos, Solitons & Fractals*, *40*(5), 2557–2568. https://doi.org/10.1016/j.chaos.2007.10.049

Kataria, A., & Singh, M. (2013). A review of data classification using k-nearest neighbour algorithm. *International Journal of Emerging Technology and Advanced Engineering*, *3*(6), 354–360.

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. *Proceedings of ICNN’95 - International Conference on Neural Networks*, *4*, 1942–1948. https://doi.org/10.1109/ICNN.1995.488968

Khaire, U. M., Dhanalakshmi, R., & Balakrishnan, K. (2022). Hybrid Marine Predator Algorithm with Simulated Annealing for Feature Selection. In O. P. Jena, B. Bhushan, & U. Kose, *Machine Learning and Deep Learning in Medical Data Analytics and Healthcare Applications* (1st ed., pp. 131–150). CRC Press. https://doi.org/10.1201/9781003226147-7

Kim, Y., Street, W. N., & Menczer, F. (2002). Evolutionary model selection in unsupervised learning. *Intelligent Data Analysis*, *6*(6), 531–556.

Kong, X., Chen, Y.-L., Xie, W., & Wu, X. (2012). A novel paddy field algorithm based on pattern search method. *2012 IEEE International Conference on Information and Automation*, 686–690.

Kuo, F. Y., & Sloan, I. H. (2005). Lifting the curse of dimensionality. *Notices of the AMS*, *52*(11), 1320–1328.

Labbi, Y., Attous, D. B., Gabbar, H. A., Mahdad, B., & Zidan, A. (2016). A new rooted tree optimization algorithm for economic dispatch with valve-point effect. *International Journal of Electrical Power & Energy Systems*, *79*, 298–311.

Lam, A. Y., & Li, V. O. (2012). Chemical reaction optimization: A tutorial. *Memetic Computing*, *4*, 3–17.

Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R. P., Tang, J., & Liu, H. (2017). Feature Selection: A Data Perspective. *ACM Comput. Surv.*, *50*(6). https://doi.org/10.1145/3136625

Li, Q., Bai, Y., & Gao, W. (2021). Improved Initialization Method for Metaheuristic Algorithms: A Novel Search Space View. *IEEE Access*, *9*, 121366–121384. https://doi.org/10.1109/ACCESS.2021.3073480

Li, Q., Liu, S.-Y., & Yang, X.-S. (2020). Influence of initialization on the performance of metaheuristic optimizers. *Applied Soft Computing*, *91*, 106193.

Mafarja, M. M., & Mirjalili, S. (2017). Hybrid Whale Optimization Algorithm with simulated annealing for feature selection. *Neurocomputing*, *260*, 302–312. https://doi.org/10.1016/j.neucom.2017.04.053

Mahesh, B. (2020). Machine learning algorithms-a review. *International Journal of Science and Research (IJSR).[Internet]*, *9*(1), 381–386.

Maldonado, S., Weber, R., & Famili, F. (2014). Feature selection for high-dimensional class-imbalanced data sets using support vector machines. *Information Sciences*, *286*, 228–246.

Medjahed, S. A., Ait Saadi, T., Benyettou, A., & Ouali, M. (2016). Gray Wolf Optimizer for hyperspectral band selection. *Applied Soft Computing*, *40*, 178–186. https://doi.org/10.1016/j.asoc.2015.09.045

Mirjalili, S. (2016a). Dragonfly algorithm: A new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems. *Neural Computing and Applications*, *27*(4), 1053–1073. https://doi.org/10.1007/s00521-015-1920-1

Mirjalili, S. (2016b). SCA: a sine cosine algorithm for solving optimization problems. *Knowledge-Based Systems*, *96*, 120–133.

Mirjalili, S., Gandomi, A. H., Mirjalili, S. Z., Saremi, S., Faris, H., & Mirjalili, S. M. (2017). Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems. *Advances in Engineering Software*, *114*, 163–191. https://doi.org/10.1016/j.advengsoft.2017.07.002

Mirjalili, S., Mirjalili, S. M., & Hatamlou, A. (2016). Multi-verse optimizer: A nature-inspired algorithm for global optimization. *Neural Computing and Applications*, *27*, 495–513.

Moscato, P., Cotta, C., Mendes, A., & others. (2004). Memetic algorithms. *New Optimization Techniques in Engineering*, *141*, 53–85.

Moysis, L., Volos, C., Jafari, S., Munoz-Pacheco, J. M., Kengne, J., Rajagopal, K., & Stouboulos, I. (2020). Modification of the logistic map using fuzzy numbers with application to pseudorandom number generation and image encryption. *Entropy*, *22*(4), 474.

Mugemanyi, S., Qu, Z., Rugema, F. X., Dong, Y., Wang, L., Bananeza, C., Nshimiyimana, A., & Mutabazi, E. (2023). Marine predators algorithm: A comprehensive review. *Machine Learning with Applications*, *12*, 100471. https://doi.org/10.1016/j.mlwa.2023.100471

Murase, H. (2000). Finite element inverse analysis using a photosynthetic algorithm. *Computers and Electronics in Agriculture*, *29*(1–2), 115–123.

Nematollahi, A. F., Rahiminejad, A., & Vahidi, B. (2020). A novel meta-heuristic optimization method based on golden ratio in nature. *Soft Computing*, *24*(2), 1117–1151.

Oh, I.-S., Lee, J.-S., & Moon, B.-R. (2004). Hybrid genetic algorithms for feature selection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *26*(11), 1424–1437. https://doi.org/10.1109/TPAMI.2004.105

Oszust, M. (2021). Enhanced Marine Predators Algorithm with Local Escaping Operator for Global Optimization. *Knowledge-Based Systems*, *232*, 107467. https://doi.org/10.1016/j.knosys.2021.107467

Pan, J.-S., Shan, J., Chu, S.-C., Jiang, S.-J., Zheng, S.-G., & Liao, L. (2022). A multigroup marine predator algorithm and its application for the power system economic load dispatch. *Energy Science & Engineering*, *10*(6), 1840–1854.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., & others. (2011). Scikit-learn: Machine learning in Python. *The Journal of Machine Learning Research*, *12*, 2825–2830.

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning*, *1*(1), 81–106. https://doi.org/10.1007/BF00116251

Rai, R., Das, A., & Dhal, K. G. (2022). Nature-inspired optimization algorithms and their significance in multi-thresholding image segmentation: An inclusive review. *Evolving Systems*, *13*(6), 889–945. https://doi.org/10.1007/s12530-022-09425-5

Rai, R., Dhal, K. G., Das, A., & Ray, S. (2023). An Inclusive Survey on Marine Predators Algorithm: Variants and Applications. *Archives of Computational Methods in Engineering*, *30*(5), 3133–3172. https://doi.org/10.1007/s11831-023-09897-x

Rao, R. (2016). Jaya: A simple and new optimization algorithm for solving constrained and unconstrained optimization problems. *International Journal of Industrial Engineering Computations*, *7*(1), 19–34.

Rao, R. V., Savsani, V. J., & Vakharia, D. P. (2011). Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems. *Computer-Aided Design*, *43*(3), 303–315.

Rashedi, E., Nezamabadi-Pour, H., & Saryazdi, S. (2009). GSA: a gravitational search algorithm. *Information Sciences*, *179*(13), 2232–2248.

Reynolds, R. G. (1994). An introduction to cultural algorithms. *Proceedings of the Third Annual Conference on Evolutionary Programming*, *24*, 131–139.

Robnik-Šikonja, M., & Kononenko, I. (2003). Theoretical and Empirical Analysis of ReliefF and RReliefF. *Machine Learning*, *53*(1), 23–69. https://doi.org/10.1023/A:1025667309714

Rodriguez-Lujan, I., Bailador, G., Sanchez-Avila, C., Herrero, A., & Vidal-de-Miguel, G. (2013). Analysis of pattern recognition and dimensionality reduction techniques for odor biometrics. *Knowledge-Based Systems*, *52*, 279–289. https://doi.org/10.1016/j.knosys.2013.08.002

Salzberg, S. L. (1994). C4.5: Programs for Machine Learning by J. Ross Quinlan. Morgan Kaufmann Publishers, Inc., 1993. *Machine Learning*, *16*(3), 235–240. https://doi.org/10.1007/BF00993309

Sandri, M., & Zuccolotto, P. (2006). Variable selection using random forests. *Data Analysis, Classification and the Forward Search: Proceedings of the Meeting of the Classification and Data Analysis Group (CLADAG) of the Italian Statistical Society, University of Parma, June 6–8, 2005*, 263–270.

Sarhani, M., Voß, S., & Jovanovic, R. (2023). Initialization of metaheuristics: Comprehensive review, critical analysis, and research directions. *International Transactions in Operational Research*, *30*(6), 3361–3397. https://doi.org/10.1111/itor.13237

Sarker, I. H. (2021). Machine Learning: Algorithms, Real-World Applications and Research Directions. *SN Computer Science*, *2*(3), 160. https://doi.org/10.1007/s42979-021-00592-x

Srinath, K. R. (2017). Python – The Fastest Growing Programming Language. *International Research Journal of Engineering and Technology*, *04*(12), 354–357.

Storn, R., & Price, K. (1997). Differential Evolution – A Simple and Efficient Heuristic for global Optimization over Continuous Spaces. *Journal of Global Optimization*, *11*(4), 341–359. https://doi.org/10.1023/A:1008202821328

Suresh, S., Lal, S., Reddy, C. S., & Kiran, M. S. (2017). A Novel Adaptive Cuckoo Search Algorithm for Contrast Enhancement of Satellite Images. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, *10*(8), 3665–3676. https://doi.org/10.1109/JSTARS.2017.2699200

Tubishat, M., Idris, N., & Abushariah, M. A. M. (2018). Implicit aspect extraction in sentiment analysis: Review, taxonomy, oppportunities, and open challenges. *Information Processing & Management*, *54*(4), 545–563. https://doi.org/10.1016/j.ipm.2018.03.008

Velliangiri, S., Alagumuthukrishnan, S., & joseph, S. I. T. (2019). A Review of Dimensionality Reduction Techniques for Efficient Computation. *Procedia Computer Science*, *165*, 104–111. https://doi.org/10.1016/j.procs.2020.01.079

Wei, W., Chen, S., Lin, Q., Ji, J., & Chen, J. (2020). A multi-objective immune algorithm for intrusion feature selection. *Applied Soft Computing*, *95*, 106522.

Xin Yao, Yong Liu, & Guangming Lin. (1999). Evolutionary programming made faster. *IEEE Transactions on Evolutionary Computation*, *3*(2), 82–102. https://doi.org/10.1109/4235.771163

Yang, X.-S. (2012). Flower pollination algorithm for global optimization. *International Conference on Unconventional Computing and Natural Computation*, 240–249.

Yousri, D., Ousama, A., Fathy, A., Babu, T. S., Allam, D., & others. (2022). Managing the exchange of energy between microgrid elements based on multi-objective enhanced marine predators algorithm. *Alexandria Engineering Journal*, *61*(11), 8487–8505.