Prediction of Heart Disease Dataset Using Hybrid Optimization and Machine Learning Techniques

By

Prayushi Patel

A thesis submitted in partial fulfillment

of the requirements for the degree of

Master of Science (MSc) in Computational Sciences

The Faculty of Graduate Studies

Laurentian University

Sudbury, Ontario, Canada

THESIS DEFENCE COMMITTEE/COMITÉ DE SOUTENANCE DE THÈSE
Laurentian Université/Université Laurentienne
Faculty of Graduate Studies/Faculté des études supérieures

Title of Thesis
Titre de la thèse — Prediction of Heart Disease Dataset Using Hybrid Optimization and Machine Learning Techniques

Name of Candidate
Nom du candidat — Patel, Prayushi

Degree
Diplôme — Master of Science

Department/Program
Département/Programme — Computational Sciences

Date of Defence
Date de la soutenance
September 27, 2019

APPROVED/APPROUVÉ

Thesis Examiners/Examinateurs de thèse:

Dr. Kalpdrum Passi
(Supervisor/Directeur(trice) de thèse)

Dr. Ratvinder Grewal
(Committee member/Membre du comité)

Dr. Thomas Merrit
(Committee member/Membre du comité)

Dr. Sanjeena Dang
(External Examiner/Examinateur externe)

Approved for the Faculty of Graduate Studies
Approuvé pour la Faculté des études supérieures
Dr. David Lesbarrères
Monsieur David Lesbarrères
Dean, Faculty of Graduate Studies
Doyen, Faculté des études supérieures

ACCESSIBILITY CLAUSE AND PERMISSION TO USE

Abstract

The traditional methods of cancer diagnosis and cancer-type recognition have quite a large number of limitations in terms of speed and accuracy. However, recent studies on cancer diagnosis are focused on molecular level identification so as to improve the capability of diagnosis process. Several recent research studies have used data mining techniques, machine learning algorithms and statistical methods to study the issue of cancer classification to achieve an effective analysis on gene expression profiles. The process has been supported by data mining and machine learning techniques such that a smart combination of various algorithms and techniques can generate comparatively efficient and reliable output. In this thesis, four distinct classification methods are used – Support Vector Machine (SVM) (SVM with Gaussian Kernel and SVM with Radial Basis Kernel), K – Nearest Neighbor, Naïve Bayes and Random Forest to study the accuracy of prediction using heart disease dataset. The models were trained using different training-testing ratios to get the optimal results. Three optimization techniques were used to further improve the prediction accuracy of heart disease. Various performance parameters including accuracy and area under the ROC curve (AUC) were computed to test the optimization algorithms, Grey Wolf Optimization (GWO), Oppositional Grey Wolf Optimizer (OGWO) and Hybrid Particle Swarm Optimization with Grey Wolf Optimizer (HPSOGWO) with the four classifiers to obtain the best optimized classifier for the heart disease datasets. The datasets used in this thesis are of heart disease taken from three different locations: Cleveland, Hungarian and Switzerland. The results show that best accuracy was achieved by using the hybrid PSO with GWO (HPSOGWO) in Switzerland (with Random Forest) and Hungarian (with KNN) datasets whereas GWO gave the highest accuracy for the Cleveland (with Random Forest) dataset with the HPSOGWO very close in accuracy to GWO.

Keywords: Hungarian, Microarray Dataset, Gene Expression, Cleveland, Switzerland, Grey Wolf Optimization, Particle Swarm Optimization, Hybrid Particle Swarm Optimization with Grey Wolf Optimizer (HPSOGWO)

Acknowledgements

In the first place, I would like to express my sincere gratitude to my supervisor, Dr. Kalpdrum Passi, for the continuous support of my Master's study and research, for his patience, motivation, enthusiasm and enormous expertise. His guidance assisted me throughout the thesis' studies and writing.

I would also like to express special thanks to all my friends for their care, support and encouragement to me. In the previous two years, we have shared our memorable moments together. I am indebted to my parents forever for their kind understanding, constant patience, and continuous encouragement when most desired. I am always thankful to them for their support and encouragement. Finally, and most importantly, I would like to acknowledge my beloved family's constant support.

Table of Contents

Abstract……………………………………………………………………............ iii

Acknowledgments…………………………………………………………...... v

Table of Contents…………………………………………………………... vi

List of Figures……………………………………………………………… ix

List of Tables……………………………………………………………… x

1 Introduction……………………………………………………………….. 1

1.1 Introduction to Cancer…………………………………………….…..... 1

1.1.1 Cancer development and cell division……………………………. 1

1.1.2 Heart Cancer…………………………………………………….... 2

1.2 Bioinformatics…………………………………………………………. 2

1.3 Gene expression………………………………………………………... 4

1.4 Classification techniques………………………………………………. 6

1.5 Objectives of the thesis……………………...……………………………. 6

2 Literature Review...………………………………………………………. 8

3 Datasets and Classification Methods………………………………….…. 13

3.1 Dataset Description……………………………………………………. 13

3.2 Classification methods.…...…………………………………………… 14

3.2.1 Support Vector Machine (SVM)…………………………………... 15

3.2.1.1 SVM with Gaussian kernel….……………………………..…. 17

3.2.1.2 SVM with Radial basis kernel…………..……..…………... 17

3.2.2 Random forest…………………………………………………..….. 18

3.2.3 KNN (k-nearest neighbor) ………………………………………... 19

3.2.4 Naïve Bayes…………………………………………………….. 20

4 Optimization Techniques…………………..…………………………….. 22

4.1 Introduction to Optimization Techniques.……..……………………………. 22

4.2 Meta-heuristic optimization technique……………………….................. 23

4.2.1 Classification on the basis of properties…………………………… 23

4.2.2 Characteristic of Meta heuristic……...……………………………. 23

4.2.3 Working of Meta heuristic Optimization.…….……………………....... 24

4.3 Meta heuristic algorithms…………...………………………………………. 25

4.3.1 Particle Swarm Optimization (PSO)……..………………….………… 26

4.3.2 Grey Wolf Optimization (GWO)………………………...……… 27

4.3.3 Oppositional Grey Wolf Optimizer (OGWO)………………...…… 32

4.3.4 Hybrid PSO with GWO……………………………………………. 36

5 Software Tools and Experiment Setup……………….….....……………….. 39

5.1 Tool.…..……………...……………………………………………………. 39

5.2 10-fold cross validation………………………………………………….... 39

5.3 Ratio Comparison………………………………………………………… 40

5.4 Performance Parameters…………………..………………………………… 41

6 Results and Discussion………………………………………………………. 44

6.1 Cleveland Dataset Output…......……………………………………...….... 44

6.2 Hungarian Dataset Output…………...………………………………………. 52

6.3 Switzerland Dataset Output………...……………………………………….. 59

6.4 Discussion………………………………………………………………… 69

7 Conclusions and Future work…………………………………………........ 72

7.1 Conclusion.…………………………………………………………………….. 72

References……………………………………………………………………. 73

List of Figures

Figure 1	Hyperplane……………………………………………………...….	16
Figure 2	Updating the positions of grey wolves after every iteration..................	29
Figure 3	Solutions by alpha, beta, delta and omega wolves…………………….	30
Figure 4	Flowchart of Grey Wolf Optimizer……..…………...............................	32
Figure 5	Graphical Analysis of Cleveland Dataset……..…………...................................	69
Figure 6	Graphical Analysis of Hungarian Dataset...…………………………………..	70
Figure 7	Graphical Analysis of Switzerland Dataset...……………………….………...	70
Figure 8	Combined Results Analysis Chart…………………….........................	71

List of Tables

Table 1	Detailed results of Cleveland dataset for Ratio (60:40) and (70:30)……	45
Table 2	Detailed results of Cleveland dataset for Ratio (80:20) and (90:10) ...…	48
Table 3	Highest Accuracy and AUC value for each classifier with best optimization technique for Cleveland Dataset……………...................	51
Table 4	Detailed results of Hungarian dataset for Ratio (60:40) and (70:30)…...	52
Table 5	Detailed results of Hungarian dataset for Ratio (80:20) and (90:10) …..	55
Table 6	Highest Accuracy and AUC value for each classifier with best optimization technique for Hungarian Dataset……………...................	58
Table 7	Detailed results of Switzerland dataset for Ratio (60:40) and (70:30).…	60
Table 8	Detailed results of Switzerland dataset for Ratio (80:20) and (90:10) …	62
Table 9	Highest Accuracy and AUC value for each classifier with best optimization technique for Switzerland Dataset…………….................	65
Table 10	Result Analysis of Cleveland Dataset.……………………………………	67
Table 11	Result Analysis of Hungarian Dataset...…………………………………	67
Table 12	Result Analysis of Switzerland Dataset..………………………………	68
Table 13	Combined Results…………………………………………….……………	71

Chapter 1

Introduction

1.1 Introduction to cancer

Cancer is a comprehensive term it discusses the disease resulting in uncontrolled development and cell division caused by cell modifications. Some cancers trigger fast development of cells, while others trigger cells to grow and divide more slowly. Some types of disease cause in noticeable tumor growth, while others, such as leukemia, do not [1]. Most cells in the body have particular tasks and set lifespan. Cell death is a component of a normal and useful occurrence called apoptosis.

A cell is instructed to die so that it can be replaced by a newer cell that works better. Errors in the instructions can cause the cell to stop its normal function and allow a cell to become cancerous. As a consequence, they create up in the body, using water and nutrients that would generally feed other cells. Cancer cells may create tumors, impair the immune system, and trigger other modifications that stop the organ from frequently working. In one region, cancerous cells may occur, then disperse through the lymph nodes. These are clusters of body-wide immune cells.

1.1.1 Cancer development and cell division

Doctors identify cancer by:

- Its place in the body
- The tissues it grows in

For instance, sarcomas grow in joints or smooth tissues, while carcinomas form in cells that cover the body's inner or outer walls [1]. In the skin, basal cell carcinomas grow, while in the breast adenocarcinomas may grow. The medical word for this is metastasis when cancer cells spread to other parts of the body. A person may have more than one cancer type at a time as well.

1.1.2 Heart cancer

Cardiac cancer (main respiratory tumor) is heart cancer. Cancerous (malignant) tumors that start in the core are most commonly sarcomas, a sort of cancer that arises in the body's smooth tissues. Noncancerous (benign) are the vast majority of heart tumors.

Heart cancer is extremely rare. One research, for instance, examined more than 12,000 autopsies and discovered only seven primary cardiac tumor cases. On average, only one case of heart cancer is seen in a year at the Mayo Clinic. [2].

Although still rare, most heart cancers have come from other parts of the body. Cancer such as lung cancer, which begins near to the heart, can grow to involve the heart or lining around the heart or cancer can start anywhere else in the body and spread through the bloodstream to the heart. Among other cancers that can affect the heart are breast cancer, kidney cancer, lung cancer, leukemia, lymphoma, and melanoma.

1.2 Bioinformatics

Bioinformatics is an interdisciplinary field in which methods and software tools are developed to understand biological data. It was proposed by Paulien Hogeweg and Ben Hesper [3] to refer to the study of biotic systems information processes. This definition placed bioinformatics parallel to biochemistry (the study of chemical processes in biological systems) as a field. Bioinformatics, as an interdisciplinary field of science, combines biology, computer science, information engineering, mathematics and statistics to analyze and interpret biological data. Bioinformatics has become a major part of many biological fields. Bioinformatics techniques such as image and signal processing allow useful results to be extracted from large amounts of raw data in experimental molecular biology. Bioinformatics' primary goal is to enhance knowledge of biological processes. However, what distinguishes it from other methods is its focus on developing and applying computationally intensive methods to achieve this goal.

Bioinformatics now involves creating and advancing databases, algorithms, computational and statistical techniques, and theory to solve formal and practical issues resulting from biological data management and analysis. Application of machine learning in bioinformatics has resulted in a large amount of implementation from prediction of diseases, diagnosis and survival analysis [3]. In bioinformatics, data science as an emerging field has contributed from reducing the dimensionality of large datasets to visualizing data.

However, Bioinformatics, often represents a two-sided problem: (1) Researchers in software engineering and other related fields see bioinformatics simply as an explicit application of hypotheses and methods due to their inability to provide exact answers to complex subatomic science problems. (2) Biologists perform hypothesis testing in wet laboratories so that bioinformatics fills in as a tool to separate common data from their preliminaries. It isn't hard to see that the two sides have their own constraints. Computational researchers need a good understanding of science and biomedical sciences, while biology researchers need to understand the concept of data review from an algorithmic point of view. In this way, the lack of coordination between these different sides not only restricts the development of life science inquiries but also limits the improvement of computational approaches in bioinformatics.

To understand bioinformatics, a basic understanding of biology is necessary. This section provides a brief introduction to some fundamental molecular biology concepts that are relevant to the problems of bioinformatics discussed in subsequent sections. The cell is the basic unit of life. Despite the diversity of cells, they all have a life cycle: they are born, eat, replicate, and die. A cell makes different decisions during the life cycle through the expression of pathways. A cell contains three types of essential atoms: Deoxyribonucleic acid (DNA), Ribonucleic acid (RNA), and proteins. Instinctively, DNA, RNA, and proteins can be seen as strings. DNA is a very long molecule that is composed of four types of bases: adenine (A), thymine (T), guanine (G), and cytosine (C). Similar to DNA, there are four bases in RNA as well. The main difference is that in RNA the base uracil (U) replaces the T base. Each protein is a string of 20 types of amino acids. DNA carries a cell's genetic information and is made up of thousands of genes. Each cell contains the genetic information so that before a cell divides (replication) the DNA is replicated. When proteins are needed, the corresponding genes are converted into RNA (transcription). Hence, the primary responsibility of RNA is to synthesize the particular protein within the DNA (translation) according to the protein encoding information. Proteins are responsible for biochemical reactions, sending signals to other cells, and forming the significant parts of the body.

1.3 Gene expression

Genes encode proteins and proteins dictate cell function. The thousands of genes expressed in a specific cell determine what that cell can do. In addition, each stage in the stream of data from DNA to RNA to protein offers the cell with a prospective control point for self-regulating its tasks by changing the quantity and sort of protein it produces [4].

The quantity of a specific protein in a cell at any specified moment represents the equilibrium between the synthetic and degrading biochemical pathways of that protein. On the synthetic side of this equilibrium, protein manufacturing begins at transcription (DNA to RNA) and proceeds with conversion (RNA to protein). Thus, controlling these procedures form a critical part in determining what proteins are present in a cell and in what quantity. In fact, the manner a cell processes its RNA transcripts and newly created proteins also significantly affects protein concentrations.

The quantities and kind of mRNA molecules in a cell represent the cell's role. In reality, in every cell, thousands of transcripts are generated every second. Given this statistic, it is not surprising that the main control point for gene expression is generally at the very start of the method of protein manufacturing— the initiation of transcription. RNA transcription allows an effective control point because many proteins can be produced from a single mRNA molecule.

Transcript processing offers an extra level of regulation for eukaryotes, and this is made possible by the existence of a nucleus. Because of the proximity of ribosomes to the new mRNA molecules, translation of a transcript starts before the transcript is complete in prokaryotes. However, in eukaryotes, transcripts are altered in the nucleus before being transferred to the cytoplasm for transcription.

Eukaryotic transcripts are also more complicated than prokaryotic transcripts. For example, the primary transcripts synthesized by RNA polymerase contain sequences that will not be part of the mature RNA. These interrupting parts are called introns and are separated before the mature mRNA reaches the nucleus. The remaining areas of transcript, which include the protein-coding areas, are called exons and spliced together to create the mature mRNA. At their ends, eukaryotic transcripts are also changed, affecting their structure and translation.

1.4 Classification techniques

Classification is a supervised learning method in machine learning and statistics in which the computer program learns from the given data input and then uses this learning to classify new observation. Classification is a technique for categorizing our data into a required and different number of classes in which each class can be assigned a label. This set of data can be either simply bi-class (such as identifying whether the person is male or female or whether the mail is spam or non-spam) or multi-class as well. Some examples of classification problems are speech recognition, handwriting recognition, biometric identification, document classification etc.

In this research, we are using the classification techniques to compare the differences among them. The included classification techniques in the research are:

1. Support Vector Machine (SVM)
    a. SVM with Gaussian kernel
    b. SVM with Radial basis kernel
2. K - nearest neighbor (KNN)
3. Naive Bayes (NB)
4. Random Forest (RF)

1.5 Objectives of the thesis

The main goal of this research is to increase the accuracy of heart disease prediction using three different datasets: Cleveland dataset, Hungarian dataset and Switzerland dataset by the combination of three different optimization techniques and four distinct classifiers. The three different optimization techniques used in this research are Particle Swarm Optimization, Grey Wolf Optimizer and Hybrid Particle Swarm Optimization with Grey Wolf Optimizer. The four different classifiers used in this research are SVM, SVM with Radial Basis Kernel, SVM with Gaussian Kernel, KNN, Naive Bayes and Random Forest. 10-fold cross validation is used further in each classification to avoid overfitting. Different training to testing ratios were used to optimize the accuracy and AUC (area under the curve). The dataset used in this research is taken from UCI machine learning repository. The algorithms were implemented in MATLAB.

Various optimization techniques have been applied to obtain global maxima of the problem set. It is an attempt to test the performance of three optimization techniques mentioned above to find global optimum for heart disease prediction and thereby increase the accuracy of prediction. The hybridization of PSO and GWO algorithms have not been tested in literature to increase heart disease prediction accuracy and to the best of our knowledge is the first attempt at investigating the efficacy of the hybrid optimization technique in this thesis.

Chapter 2

Literature Review

Microarray data in dimensionality are often extremely asymmetric, highly redundant and noisy [32]. Most genes are considered to be non-informative in relation to the classes. This paper suggested a novel feature selection strategy to classify high-dimensional cancer microarray data using filtering techniques such as signal-to-noise (SNR) score and optimization techniques such as Particle Swarm Optimization (PSO). The suggested technique is divided into two phases. The data set is grouped in the first phase using k-means clustering, using SNR to rate each gene in each group. The highest rated genes are gathered from each group and a new subset of features is created. The largest classified genes are gathered from each group and a new subset of features is generated. The new feature subset is used as feedback to the PSO in the second phase and optimized feature subset will be generated. Support vector machine (SVM), K-nearest neighbor (K-NN) and Probabilistic Neural Network (PNN) are used as evaluators and one out cross validation strategy is used for validation. In the literature, they compared their strategy and other methods using PSO. Their strategy of using PSO has been shown to yield better results than others. Experimental results on Leukemia using SVM, DLBCL using SVM, Colon Cancer using KNN and Breast cancer using KNN and SVM showed that their suggested algorithm is more effective and gives the highest result as compared to other approaches. Their algorithm is therefore a helpful tool for selecting subset features for cancer microarray data. In this paper, they have used four different cancer datasets, as mentioned above, using PSO.

Moumita Pradhan *et al.* [39] proposed an efficient meta-heuristic approach, namely the Oppositional Grey Wolf Optimization (OGWO) algorithm to solve the optimal operating strategy of Economic Load Dispatch (ELD) problem with optimal operating strategy. The suggested algorithm contains two fundamental ideas. First, the hunting behavior of grey wolves and social hierarchy are used to strive for optimal options, and second, oppositional concept is integrated into the grey wolf evaluation algorithm (GWO) to increase the convergence rate of the conventional GWO algorithm. It is applied to small, medium and large-scale test systems to solve ELD problems of 13-unit, 40-unit and 160-unit systems to show the performance of the proposed algorithm.

Seyedali Mirjalili *et al.* [37] proposed a new meta-heuristic called Grey Wolf Optimizer (GWO) inspired by the nature of grey wolves (Canis lupus). The GWO algorithm imitates the hierarchy of leadership and the mechanism of hunting grey wolves in nature. Four kinds of grey wolves are used to simulate the leadership hierarchy, such as alpha, beta, delta, and omega. Furthermore, the three main stages of hunting, searching for prey, encircling prey, and attacking prey are introduced. The algorithm is then benchmarked on 29 well-known test functions and the results are checked through a comparison with Particle Swarm Optimization (PSO), GSA, Differential Evolution (DE), Evolutionary Programming (EP), and Evolution Strategy (ES). The results show that compared to these well-known meta-heuristics, the GWO algorithm is able to deliver very competitive results. The paper also considers solving three classical engineering design problems (tension / compression spring, welded beam designs, and pressure vessel designs) and provides a real implementation of the proposed optical engineering technique. The results of the several engineering design problems and real applications prove that the proposed algorithm applies to challenging problems with unknown search spaces. In this paper, they have implemented different numerical problems and experiments were tested on different test functions.

A novel feature selection method using a combination of differential evolution (DE) optimization method and a repair mechanism based on feature distribution measures has been suggested by Rami N. Khushaba *et al.* [6]. In the combinatorial optimization problem of feature selection, the new method, abbreviated as DEFS, uses the DE float number optimizer. In order to make the solutions controlled by the float-optimizer suitable for feature selection, a roulette wheel structure was constructed and supplied with the probabilities of features distribution. Those probabilities were constructed among the iterations by distinguishing the features that contribute to the most promising solutions. The proposed DEFS is used to search for optimal subsets of features in datasets with varying dimensionality. It is then utilized to aid in the selection of Wavelet Packet Transform (WPT) best basis for classification problems, thus acting as a part of a feature extraction process.

A method named as HPSO-LS (Hybrid Particle Swarm Optimization - Local Search strategy) applies a local search strategy embedded in the particle swarm optimization to choose the less correlated and salient feature subset proposed by Parham Moradi *et al.* [7]. The goal of the local search technique was to guide the search process of the particle swarm optimization to choose distinct features by conceiving their correlation information. Moreover, the suggested method applies a subset size determination scheme to choose a subset of features with minimized size. The performance of the suggested method has been estimated on 13 benchmark classification problems and compared with five state-of-the-art feature selection methods. Moreover, HPSO-LS has been compared with four well-known filter-based methods including the information gain, term variance, fisher score and mRMR and five well-known wrapper-based methods including genetic algorithm, particle swarm optimization, simulated annealing and ant colony optimization. The results established that the introduced method develops the classification accuracy compared with those of the filter based and wrapper-based feature selection methods.

A very recent PSO variant, known as competitive swarm optimizer (CSO) that was dedicated to large-scale optimization for resolving high-dimensional feature selection problems has been developed by Shenkai Gu *et al.* [8]. In addition, the CSO, which was originally developed for continuous optimization, was adapted to perform feature selection that can be conceived as a combinatorial optimization problem. An archive technique was also introduced to minimize computational cost. Experiments on six benchmark datasets established that as compared to the canonical PSO-based feature selection, CSO-based feature selection algorithm selects much smaller number of features and results in better classification performance as well.

A multi-kernel support vector machine has been suggested by Shankar *et al.* [9]. They introduced a model to separate thyroid data by applying optimal feature selection and kernel-based classifier. They created classifications models to show the classification of data using “multi-kernel support vector machine”. The novelty and objective of this proposed model as feature selection was used to enhance the performance of classification with the help of improved grey wolf optimization. It results in an optimal feature selection as the insignificant features are removed from the dataset and it increases the computationally performance of the model. The introduced thyroid classification results in 97.49% accuracy, 99.05% sensitivity, and 94.5% specificity. The performance measures were calculated from the confusion matrix and compared with individual models of existing classifiers and optimization techniques.

Narinder Singh *et al.* [40] proposed a nature-inspired algorithm called HPSOGWO, the combination of Particle Swarm Optimization (PSO) and Grey Wolf Optimizer (GWO). The main concept is to improve the ability of exploitation in Particle Swarm Optimization with the ability of exploration of Grey Wolf Optimizer to produce the strength of both variants. Some unimodal, multimodal, and fixed measurement multimodal test capacities are used to check the arrangement quality and execution of HPSOGWO variation. The numerical and statistical solutions show that the hybrid variant significantly outperforms the variants of PSO and GWO in terms of solution quality, solution stability, speed of convergence, and the ability to find the global optimum. In this paper, twenty-three benchmark problems are used to test the ability of HPSOGWO. These problems can be divided into three different groups: unimodal, multimodal, and fixed dimension multimodal functions.

Chapter 3

Datasets and Classification Methods

3.1 Dataset Description

The analysis and results in this research are based on the dataset of heart disease. The primary source is the dataset which has been provided in the UCI machine learning repository [10].

It is used for evaluating the efficiency of the suggested methods based on the parameters like accuracy, sensitivity and specificity and other performance metrics. The implementation is done in MATLAB. The directory has three databases in regards with the diagnosis of heart disease. All the utilized attributes are numeric-valued. The data has been selected from the following three locations:

1. Cleveland Clinic Foundation (Cleveland.data)
2. Hungarian Institute of Cardiology, Budapest (Hungarian.data)
3. University Hospital, Zurich, Switzerland (Switzerland.data)

The instance format is the same for all the databases. While the databases have 76 raw attributes, but all the published experiments refer to using a subset of 14 of them. Particularly, the Cleveland database is the only one that ML (Machine Learning) researchers have used to date. The field "goal" relates to the presence of heart disease in the patient. It is estimated to be an integer between 0 (no presence) and 4. Simulation with the Cleveland database has focused on differentiating the existence (values 1,2,3,4) from absence (value 0). There are some missing values in the given dataset.

Attribute Information:

The 14 attributes used are:

1. #3 (age) – age in years

2. #4 (sex) – sex (1 = male; 0 = female)
3. #9 (cp) – chest pain type
4. #10 (trestbps) – resting blood pressure (in mm Hg on admission to the hospital)
5. #12 (chol) – serum cholesterol in mg/dl
6. #16 (fbs) – (fasting blood sugar > 120 mg/dl) (1 = true; 0 = false)
7. #19 (restecg) – resting electrocardiographic results
8. #32 (thalach) – maximum heart rate achieved
9. #38 (exang) – exercise induced angina (1 = yes; 0 = no)
10. #40 (oldpeak) – ST depression included by exercise relative to rest
11. #41 (slope) – the slope of the peak exercise ST segment
12. #44 (ca) – number of major vessels (0-3) colored by flourosopy
13. #51 (thal) – 3 = normal; 6 = fixed defect; 7 = reversable defect
14. #58 (num) (the predicted attribute) – diagnosis of heart disease (angiographic disease status)

3.2 Classification methods

Classification is the action of assigning an object to a category based on the object's characteristics. Classification in data mining relates to the task of evaluating a number of pre-classified data objects to learn a model (or function) that can be used to classify an unknown data object as one of several pre-defined classes. A set of features or variables describe a data object, referred to as an instance [11]. One of the attributes describes the class belonging to an example and is therefore called the class attribute or class variable. Other attributes are often referred to as predictor attributes or independent attributes (or variables). The set of examples used to understand the model of classification is called the training data set. Classification tasks include regression, which builds a model to predict numerical values from training data, and clustering, which group examples to form categories. Classification belongs to the supervised learning category, which is differentiated from unsupervised learning. Training data in supervised learning consists of pairs of input data (usually vectors) and specified outputs, whereas there is no apriori output in unsupervised learning.

There are various applications of classification, such as learning from a patient database to diagnose a disease based on a patient's symptoms, analyzing credit card transactions to detect fraudulent transactions, automatically recognizing letters or numbers based on handwriting samples, and distinguishing extremely active compounds from inactive compounds based on drug discovery structures. The community of data mining inherits the classification techniques developed in statistics and machine learning and applies them to different problems in the real world.

In our research, we have used four classification methods: SVM (Support Vector Machine), SVM with Gaussian kernel, SVM with radial basis kernel, Random Forest, Naive Bayes, and KNN (k-nearest-neighbor). The methods are briefly described below.

3.2.1 Support Vector Machine (SVM)

The Support Vector Machine (SVM) was first proposed by Vapnik and has since attracted a high level of interest in the research community of machine learning [12]. Several recent studies have reported that in terms of classification accuracy, SVM (Support Vector Machines) can offer high performance compared to other classification algorithms.

For classification and regression, support vector machine is the learning algorithm that has been applied effectively to a broad spectrum of pattern recognition problems and is also a new and unique approach to supervised pattern classification [13]. A support vector machine (SVM) is a supervised learning algorithm which can be used for binary classification or regression. In applications such as natural language processing, speech and image recognition, and computer vision, support vector machines are popular. A support vector machine builds an optimal hyperplane as a decision surface to maximize the margin of separation in the data between the two classes. The dashed line in Figure 1 separates the data is called the ‘Decision Boundary’. The other two lines are called the hyperplanes that make the right decision boundary. An ideal hyperplane is produced by the algorithm that categorizes new instances. This hyperplane is a row separating a jet into two sections in two dimensional spaces where it lies on either hand in each category.

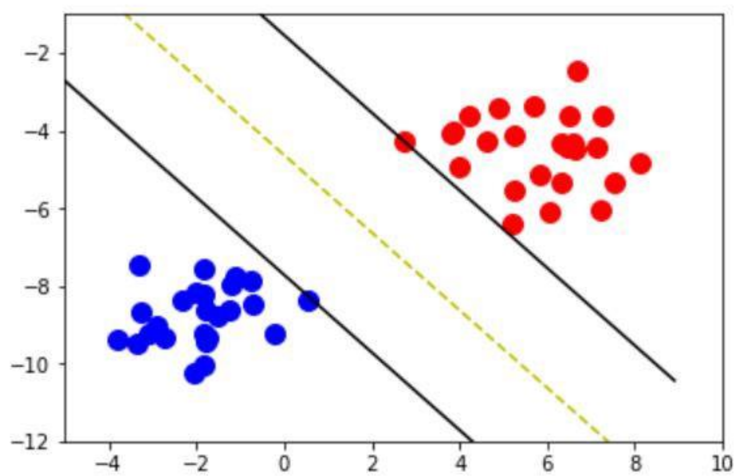

Figure 1: Hyperplane (Madhu Sanjeevi, 2017) [14]

Support vectors refer to a small subset of training observations that are used to support the decision surface's optimal location. Besides that, SVM is best adapted to operate with high dimensionality function spaces properly and efficiently as in a simple manner it is based on strong mathematical foundations and results in very powerful algorithm. Support Vector Machines (SVM) calculate the distance between two observations and discover a decision boundary that maximizes the distance between the closest members of different classes and uses a scheme called kernels. In machine learning algorithms, support vector machines are classified as kernel techniques and are also referred to as kernel machines. Few of the SVM kernel techniques have been discussed below.

3.2.1.1 SVM with Gaussian Kernel

In machine learning systems, based on the statistical learning process, the most prevalent classification techniques are kernel techniques such as support vector machines (SVMs). Support vector classification with Gaussian kernel is sensitive to the width of the kernel. Small kernel width may result in over-fitting while large width may result in under-fitting [15]. When linear algorithms are used to evaluate, they have a significant computational gap with computational time in $O(N^3)$ and space cost in $O(N^2)$, where N is the size of the training set. Hence it is difficult for kernel methods to scale up to large-scale problems.

Equation for SVM with Gaussian kernel is:

$$k(x, y) = \exp\left(-\frac{\|x-y\|^2}{2\sigma^2}\right) \tag{1}$$

Here, x is a training vector, vector $y \in \{+1, -1\}$ and σ is free parameter.

The Gaussian kernel calculated with a support vector is an exponentially decaying function in the input feature space, the maximum value of which is achieved in the support vector and which decays uniformly in all directions around the support vector, resulting in hyper-sphere contours of the kernel function. The Gaussian kernel with SVM classifier is merely a weighted linear combination of the kernel function, calculated between a data point and each of the support vectors. The role of a support vector in classifying a data point is tempered by α, the global prediction of the support vector's usefulness, and K(x, y), the local influence of a support vector in prediction of a specific data point.

3.2.1.2 SVM with Radial basis kernel

In machine learning, a popular kernel function used in various kernelized learning algorithms is the radial basis function kernel or RBF kernel. Especially, it is widely used in the classification of support vector machines. Unlike linear kernel, Radial basis kernel not only examines the situation when the connection between class labels and attributes appears to be nonlinear, but the samples are also mapped to a higher dimensional space in a nonlinear strategy [16]. When implemented separately, the Radial Basis Function (RBF) kernel and the Polynomial kernel will perform better, resulting in better kernel function instead of being applied together as the polynomial kernel of the radial basis has the feature of the two kernel functions [17]. There are situations in which the RBF kernel is not suitable. In particular, one can only use the linear kernel when the number of features is very large.

The RBF kernel on two samples x and x', depicted in some input space as feature vectors, is described as:

$$K(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2\sigma^2}\right) \tag{2}$$

$\| x - x' \|^2$ may be recognized as the squared Euclidean distance between the two feature vectors, σ is a free parameter.

3.2.2 Random forest

Random forests, first implemented by Breiman in 2001 [18], are a set of classification and regression trees. Random forests, also known as random decision forests, is a common ensemble method that can be used to construct predictive models for problems of both classification and regression. Ensemble methods combine more than one algorithm of the same or different kind for the classification of objects. It can be trained and then used to make predictions that uses various learning models to achieve better predictive results. In the case of a random forest, the model produces an entire forest of randomly uncorrelated decision trees to reach the best possible answer. In [19], the amount of decision tree classifiers on different sub-samples of the chosen datasets is fitted into a meta estimator known as random forest. A random forest also improves the precision of forecast and over-fitting control on average. The sub-sample size is always the same as the original input sample size but the samples are drawn with replacement if bootstrap=True (default). For classification, regression and other tasks, random forests or random decision forest are an ensemble learning method. It works by building a number of decision trees at training time and outputting the class which is the mode of the individual trees' classes (classification) or mean prediction (regression). Random decision forests are correct for the practice of decision trees to overfit their training set.

A random forest is a meta estimator that fits into different sub-samples of the dataset a set of decision tree classifiers. It then utilizes average to enhance predictive accuracy and over-fitting control. One can use the Random Forest classifier, a decision tree that doesn't have to be combined with a bagging classifier. To deal with regression tasks in Random Forest, one can use the Random Forest regressor.

3.2.3 KNN (k-nearest neighbor)

The K-nearest neighbors (KNN) algorithm is a lucid and easy-to-implement machine learning algorithm. KNN is one of the most common learning algorithms as well as it is one of the easiest classification algorithms. KNN normally solves both classification and regression issues. In both cases, the input consists in the feature space of the nearest k training examples. The output depends on the K neighbors of KNN whether it is used for classification or regression. The algorithm determines the close proximity of similar things. In other words, similar things are assumed to be close to each other. The distances between the query and all data space examples is calculated by selecting the specified number of examples (K) closest to the query, and then voting for the most common label (in the classification case) or label the average (in the regression case). When KNN is used for regression, the prediction is done on the basis of their mean or median values [20]. When using KNN for classification, it is possible to determine the output as the class with the highest frequency out of the most similar instances. Each instance is considered as the prediction in votes for their class and the class with the most votes. This algorithm is a non-parametric, lazy learning algorithm. Its purpose is to use a database to predict a new sample point classification by dividing the data points into multiple classes. The distance between points is defined in equation (3).

$$d(X,Y)=\sqrt{\sum_{i=1}^{N}(Xi-Yi)^{2}} \qquad (3)$$

The distance measures are valid only for continuous variables. The best way to select the optimal value for N is to inspect the data first [21]. Generally, a large N value is more accurate as it reduces the overall noise, but there is no guarantee. Cross-validation is another way to use an independent dataset to validate the N value to retrospectively determine a good N value.

3.2.4 Naive Bayes

Naïve Bayes is used for predictive modeling, an extremely powerful yet simple algorithm. Naïve Bayes can be used for both binary (two-class) and multiclass classification. When done using binary or categorical input values, the method seems to be easy. It is known as Naive Bayes because to make the calculations tractable, probability calculations for each hypothesis are simplified [23].

Bayes’ theorem has many uses in probability theory and statistics [22]. Using Bayes' theorem, from a new set of features, one can model a learner predicting the likelihood of a reaction variable belonging to some class. Bayesian Networks (BN) and Bayesian Classifiers (BC) are traditional probabilistic techniques successfully used by various machine learning methods to help solve a variety of problems in many different fields. A Naïve Bayes or probabilistic directed acyclic graph model is a probabilistic model representing a collection of variables and their conditional dependence by means of a directed acyclic graph (DAG). Naive Bayes is ideal for taking an occurring event and predicting the likelihood that the contributing factor is any of the several known causes. For example, a Naive Bayes could represent the probabilistic relationship between diseases and symptoms. The network can be used to calculate the probabilities of the existence of various diseases given the symptoms. They can be used for a broad spectrum of functions, including prediction, detection of anomalies, diagnosis, automated insight, reasoning, prediction of time series and uncertain decision-making. Bayesian classifier is based on the Bayes theorem, which states that given the training data B, posteriori probability of a hypothesis A, P(A|B) is given by equation (4).

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{4}$$

Using Bayes theorem, given that B is observed, we can find the probability of A happening. Here, the training data is B and the hypothesis is A [24]. The assumption made here is that there are independent predictors / features. That's one particular feature's presence doesn't influence the other. It is therefore called naïve.

Chapter 4

Optimization Techniques

4.1 Introduction to Optimization Techniques

We are always attempting to optimize something in almost all the applications in engineering and industry, it can be minimizing the cost and energy consumption, or it can be maximizing the profit, production, output, efficiency and performance. Optimization is everywhere, and with a broad spectrum of applications is a significant paradigm of its own [25]. Based on the different features and focus, we try to compare optimization algorithms which can be categorized in many different and important ways. These algorithms can be categorized as gradient-based also called derivative-based and gradient-free also known as derivative-free methods.

Optimization algorithms further categorized into deterministic or stochastic. It is said to be deterministic when an algorithm operates without any random nature in a mechanically deterministic way. For these algorithms, if we begin with the same original point, it will achieve the same final solution. Some good examples of deterministic algorithms are hill-climbing and downhill-simplex.

On the second side, the stochastic algorithms usually reach a different point whenever we run the algorithm, even when we begin with the same initial point, this happens because algorithms have some randomness. Some good examples of stochastic algorithms are hill-climbing with a random restart and genetic algorithms.

4.2 Meta-heuristic optimization techniques

Metaheuristic optimization uses metaheuristic algorithms to address optimization problems. Essentially, optimization is everywhere, from engineering design to economics, from holiday scheduling to internet routing. Most real-world optimizations are extremely nonlinear and multimodal, under multiple complicated limitations. Different goals often conflict. Even for a given goal, there may be no ideal alternatives at all. Overall, it is not a simple job to find an ideal alternative or even sub-optimal alternatives. In a modern world, metaheuristic calculations will generally be suitable for global development, however it is not constantly efficient. It may very well be a mixed type or cross breed, using a mixture of determinist components with irregularity, or joining one calculation to another in order to plan progressively effective calculations

4.2.1 Classification on the basis of properties

From the theory point of view, the optimization techniques classify according to their algorithmic structure and fundamental values [26]. However, a software engineer or user who wants to solve a problem with such a strategy is more interested in its "interfacing features" such as speed and precision. In terms of probabilistic algorithms, speed and precision are conflicting objectives. A general rule of thumb is that to achieve improvements in accuracy of optimization one needs to invest more time.

4.2.2 Characteristics of meta heuristic

In the past, algorithms with stochastic components have often been termed as heuristic, although recent literature tends to refer to them as metaheuristics. Here meta - means beyond or higher level, and metaheuristics usually do better than simple heuristics [27]. A metaheuristic can be regarded as a "master strategy that guides and modifies other heuristics in order to produce solutions beyond those that are normally generated in a quest for local optimality". Furthermore, all metaheuristic algorithms use some randomization tradeoff and local search. In a reasonable amount of time, quality solutions to difficult optimization problems can be discovered, but there is no assurance that ideal solutions can be achieved. These algorithms are assumed to work most of the time, but not all of the time. Nearly all metaheuristic algorithms tend to be appropriate for global optimization.

Any metaheuristic algorithm has two main parts: exploitation and exploration. Exploration means generating varied solutions to explore the search space on a worldwide scale, while exploitation means focus the search in a local region knowing that this region is currently finding a good solution. In selecting the best solutions to enhance the rate of convergence of algorithms, a good balance should be found between exploitation and exploration. The selection of the best ensures that solutions converge to the optimum, while exploration through randomization enables the search to escape from local optima and, at the same time, enhances the diversity of solutions. A good mixture of these two significant components will generally make it possible to achieve global optimality.

4.2.3 Working of meta heuristic optimization

An optimization can be regarded as a problem of minimization or maximization in the simple context. For example, the $f(x) = x^2$ function has a minimum $f_{min} = 0$ at $x = 0$ in the entire domain $-\infty < x < \infty$ representation [27]. Usually, if a function is simple enough, we can use the first derivative $f'(x)=0$ to determine potential locations and use the second derivative $f''(x)$ to determine whether the solution is a maximum or a minimum. Moreover, this is not a simple task for nonlinear, multimodal, multivariate features. Additionally, some features may have discontinuities, so it is not easy to acquire derivative information. Many traditional techniques, such as hill-climbing, may face various challenges.

4.3 Meta heuristic algorithms

There are many different meta heuristic algorithm such as:

1. Simulated Annealing
2. Genetic Algorithm
3. Ant Colony Optimization
4. Artificial Bee Colony Optimization
5. Differential Evolution
6. Gravitational Search Algorithm
7. Particle Swarm Optimization
8. Grey Wolf Optimizer
9. Hybrid Particle Swarm Optimization and Grey Wolf Optimizer and many more

Among them the following meta heuristic algorithms were considered in this research: Simulated Annealing, Genetic Algorithm, Ant Colony Optimization, Bee Optimization, Particle Swarm Optimization, Grey Wolf Optimizer and Hybrid Particle Swarm Optimization with Grey Wolf Optimizer.

In this research, Grey Wolf Optimizer (GWO), Oppositional Grey Wolf Optimizer (OGWO) and Hybrid Particle Swarm Optimization with Grey Wolf Optimizer (HPSOGWO) were implemented.

4.3.1 Particle Swarm Optimization (PSO)

Kennedy and Eberhart [30] developed Particle Swarm Optimization (PSO) in 1995, based on swarm behaviors found in nature such as fish and bird schooling. Since then, in the field of swarm intelligence, PSO has generated a lot of attention and is now creating an interesting, ever-expanding study topic. In optimization, computational intelligence, and design/scheduling applications, PSO has been implemented to almost every area [30]. There are at least two dozen variations of PSO as well as hybrid algorithms obtained by combining PSO with other existing algorithms that are also becoming increasingly popular.

By adjusting the trajectories of individual agents, called particles, PSO searches the space of an objective function. Each particle traces a route that can be represented as a positional vector that depends on time [30]. A swarming particle's motion comprises of two main parts: a stochastic component and a deterministic component. Each particle is drawn to the current global best g* position and its own best-known place x_i* while at the same moment showing a tendency to move randomly.

At the point when a particle finds an area that is better than any recently found areas, at that point it updates this area as the new current best for particle i. There is a current best for all particles at any time t at each iteration. The point is to locate the worldwide best among all the present best arrangements until the target no longer improves or after a specific number of cycles.

Let x_i and v_i be the vector and speed of particle i respectively. The following formula determines the new velocity vector:

$$v_i^{t+1} = v_i^{\,t} + \alpha\epsilon_1[g^* - x_i^t] + \beta\epsilon_2[x_i^* - x_i^t]$$

where ϵ_1 and ϵ_2 are two irregular vectors, and every passage takes an incentive somewhere in the range of 0 and 1. The parameters α and β are the learning parameters or acceleration constants, which are typically equal to, say, $\alpha \approx \beta \approx 2$.

The underlying areas of all particles ought to be conveyed moderately consistently with the goal that they can test over most locales, which is particularly significant for multimodal issues. The underlying speed of a molecule can be set to zero, that is,

$$v_i^{t=0} = 0 \tag{5}$$

The new position would then be updated by the equation

$$x_i^{t+1} = x_i^t + v_i^{t+1} \tag{6}$$

In spite of the fact that v_i can take any value, it is generally limited in some range $[0, v_{max}]$.

There are numerous variations which expand the standard PSO calculation, and the most observable improvement is likely to use a inertia function $\theta(t)$ so that v^t_i is replaced by $\theta(t)v^t_i$ where θ takes a value between the range of 0 and 1. In the simplest case, the inertia function can be taken as a constant, typically $\theta \in [0.5,0.9]$. This is proportionate to introducing a virtual mass which balances out the movement of the particles, and thus the algorithm is expected to converge more quickly.

As the iterations proceed, the particle system swarms and may converge towards a global optimum.

4.3.2 Grey Wolf Optimization (GWO)

Feature extraction selects a lesser number of optimized features from a set of all features without affecting the performance of the system, thus, maintaining higher accuracy. As number of features involved in machine learning tasks is large, various types of methods have been implemented which attempt to solve the problem of reducing redundant and irrelevant data features. Grey Wolf Optimization (GWO) is a meta-heuristic algorithm inspired by grey wolves and proposed by Mirjalili et al. [37] which imitates the leadership hierarchy and hunting behavior of grey wolves. Alpha, beta, gamma and delta wolf are the four types of wolves in which wolves are divided in the simulation of leadership hierarchy while the hunting behavior includes searching for prey, encircling prey and attacking prey [37]. The alpha wolf has been the leader wolf and pecking orders and decisions to the other wolves in the pack. Beta wolf is below the alpha wolf, these wolves obey the alpha wolf and they give instructions to other wolves. Omega wolves stand lowest in the hierarchy and must report to the wolves above. Delta wolf is the category which includes the wolves that don’t belong to alpha, beta or omega category. Various roles are played by them, they could be spies that lookout the boundaries and generates alert during the situation of danger or they could play the role of guard that ensures the protection and safety of the pack. To achieve optimization, GWO algorithm uses a mathematical model that represents the hunting behavior of grey wolves. The grey wolves surround their prey by making a circle as shown in Figure 2, the mathematical equations representing this encircling behavior are proposed in the paper Grey Wolf Optimizer [37]. The equations are:

$$\vec{D} = \left|\vec{C} \cdot \vec{X}_p(t) - \vec{X}(t)\right| \tag{7}$$

$$\vec{X}(t+1) = \vec{X}_p(t) - \vec{A} \cdot \vec{D} \tag{8}$$

$$\vec{A} = 2\vec{a} \cdot \vec{r}_1 - \vec{a} \tag{9}$$

$$\vec{C} = 2 \cdot \vec{r}_2 \tag{10}$$

where t is the current iteration, A and C are coefficient vectors, X_p is the position vector of the prey, and the position vector of a grey wolf is indicated by X. Components of $\vec{a}$ are linearly decreased from 2 to 0 over the course of iterations and r_1, r_2 are random vectors in [0, 1].

In the beginning, the wolves are positioned at random locations with reference to the location of the prey and further value of fitness function is computed for various random locations of wolves, consequently alpha, beta, gamma, delta solutions are taken note of. Throughout the simulation of GWO, the iteration apprises in the locations of all the grey wolves as depicted in Figure 2. Fitness function is reliant on distance assessment amongst the prey and the various wolves. The distance evaluated acts as a decisive factor in finding out the optimized solution determined by fitness function

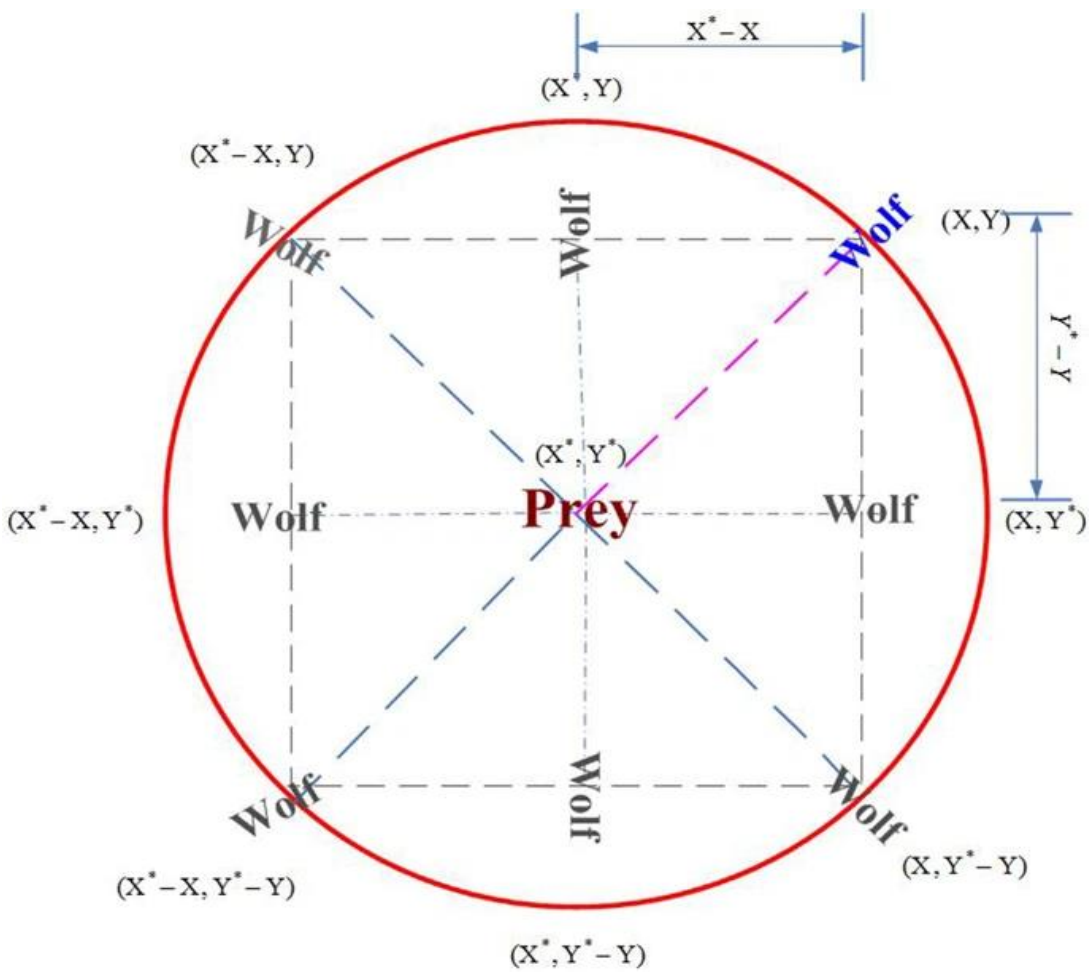

Figure 2: Updating the positions of grey wolves after every iteration. (Jie-Sheng Wang & Shu-Xia Li et al. 2019) [38]

i.e. the wolf that is found to be nearest in location to the prey has the higher probability of continuing to be closest to the prey in near future. Figure 2 illustrates the repeated iterations solutions as depicted by the alpha, beta, gamma and delta wolves. Algorithm 1 exemplifies Grey Wolf Optimization. The GWO has been represented by the following equations and Figure 3.

$$\vec{D}_\alpha = |\vec{C}_1 . \vec{X}_\alpha - \vec{X}|, \vec{D}_\beta = |\vec{C}_2 . \vec{X}_\beta - \vec{X}|, \vec{D}_\delta = |\vec{C}_3 . \vec{X}_\delta - \vec{X}| \quad (11)$$

$$\vec{X}_1 = \vec{X}_\alpha - \vec{A}_1 . (\vec{D}_\alpha), \vec{X}_2 = \vec{X}_\beta - \vec{A}_2 . (\vec{D}_\beta), \vec{X}_3 = \vec{X}_\delta - \vec{A}_3 . (\vec{D}_\delta) \quad (12)$$

$$\vec{X}(t+1) = \frac{\vec{X}_1+\vec{X}_2+\vec{X}_3}{3} \quad (13)$$

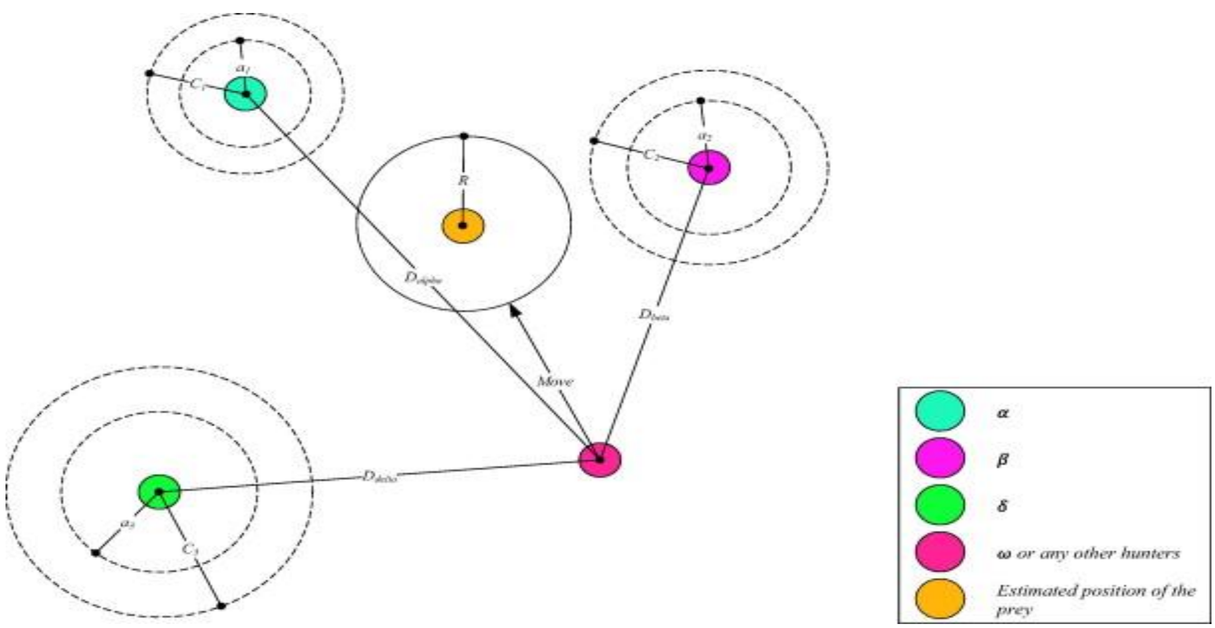

Figure 3: Solutions by alpha, beta, delta and omega wolves. (Seyedali Mirjalili et al. 2014) [37]

Algorithm 1: Traditional Grey Wolf Optimizer (GWO)

Input: objective functions, constraints and decision variables.

Output: Search agent having the best solution.

1. Initialize the grey wolf population, a, A and C
2. Calculate the fitness of each search agents
3. Find X_α, X_β, X_δ: best, second best and third best search agents
4. while t < Itermax:
    - Update the position using equation (13)
    - Update a, A and C
    - Calculate the fitness of all search agents
    - Update X_α, X_β, X_δ
    - Increment t by 1.
5. Return X_α
6. End.

Algorithm 1: Grey Wolf Optimizer (GWO) [37]

The traditional algorithm of GWO for solving complex problems is given in algorithm 1 and the flowchart is depicted in Figure 4.

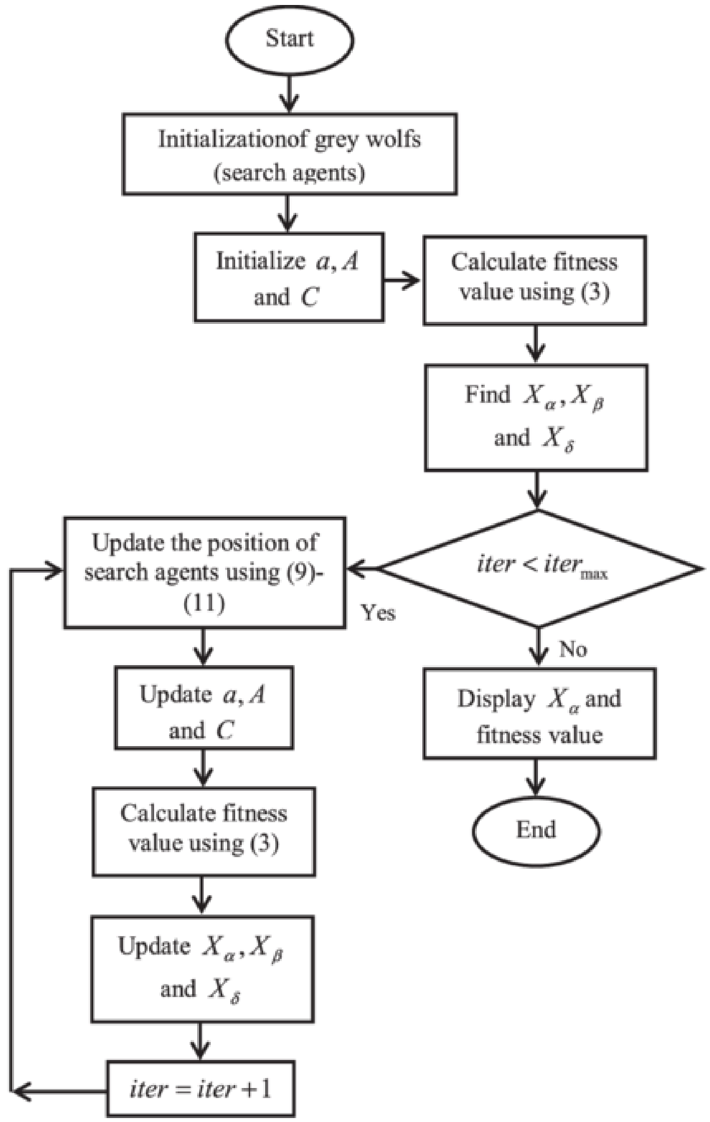

Figure 4: Flowchart of Grey Wolf Optimizer

4.3.3 Oppositional Grey Wolf Optimizer (OGWO)

Opposition-based learning (OBL) is one of Taizhou's strong optimization tools to improve the convergence velocity of different heuristic optimization techniques. Successful implementation of the OBL involves evaluating the opposite population and current population in the same generation in order to find the best candidate solution to a particular problem. The notion of OBL has been used effectively in multiple meta-heuristics [39] used to improve convergence velocity. The concept of opposite number needs to be defined to explain OBL.

Let N be a real number. The opposite number is characterized by

$$\boldsymbol{N^{0} = x + y - N} \tag{14}$$

The definition can be expanded as follows for d-dimensional search space:

$$\boldsymbol{N_i^{0} = x_i + y_i - N_i} \tag{15}$$

where there is a (N_1, N_2…. N_d) point in d-dimensional search space and $N_i \in [x_i, y_i]$ ***; i= {1,2, 3,…..d}***

The notion of OBL is used in any opposition-based optimization (OBO) using the jumping rate of generation in the initialization phase and in each iteration [39]. OBO's various steps are articulated in the steps below.

Step 1: Population people within working boundaries are initialized randomly.

Step 2: Generate opposite population:

for j = 1: population size

for I = 1: number of control variables

end

end

Step 3: Sort current population and the opposite population from best to worst

Step 4: Depending upon the population size, select number of fittest solutions from the current and the opposite population

Step 5: Update the control variables of the specified problem using the proposed optimization technique

Step 6: Using jumping rate, opposite population are generated from the current population.

Step 7: Classify the population and the reverse population from the best to the worst and pick the amount of best alternative from the population and the reverse population

Step 8: If the termination requirement is met, stop iteration for next generation to step 5.

Algorithms steps for Opposition-based Optimization (OBO) [39]

Step 1: The positions of grey wolves (search agents) are initialized randomly in the search space. Population size (number of wolves) and number of iterations are also needed to be fixed.

Step 2: Calculate fitness value of each search agent which represents the distance of wolf from the prey.

Step 3: Initialize opposite points and use them to generate opposite population and calculate the fitness of each individual populations.

Step 4: Perform sorting among current population and opposite population , based on their fitness values.

Step 5: Select number of fittest solutions from the combination of the current and the corresponding opposite population.

Step 6: Based on the fitness values, best (), second best () and third best () solutions are identified which represent the alpha, beta and delta category wolves, respectively.

Step 7: Modify the positions of the grey wolves.

Step 8: Update the fitness value using the modified position of the grey wolves.

Step 9: Using jumping rate, the opposite population are generated from the current population.

Step 10: Select number of fittest solution from the combination of the current and the opposite population.

Step 11: Repeat step 6 to step 10 until the maximum number of iteration is reached.

Step 12: Output the best solution.

Algorithm steps of Oppositional Grey Wolf Optimizer (OGWO) [39]

4.3.4 Hybrid Particle Swarm Optimization with Grey Wolf Optimization (HPSOGWO)

In HPSOGWO, PSO and GWO are hybridized using a low level co-evolutionary hybrid. The method is co-evolutionary because instead of applying one after another, we implement both the optimizers in parallel.

For heuristic variants, many scientists have submitted several variations of hybridization [40]. With relay or coevolutionary methods as heterogeneous or homogeneous, two versions can be hybridized at low or high levels.

In this document, we use low-level coevolutionary blended hybrid to hybridize Particle Swarm Optimization with Grey Wolf Optimizer algorithm [40]. The hybrid is low level because the functionalities of both variants are merged. It's coevolutionary because we don't use both the optimizers one after the other instead, they are implemented and run in parallel. It is mixed because there are two different variants involved in the final problem solving. Based on this modification, we improve the ability of Particle Swarm Optimization to exploit with the ability of exploration in Grey Wolf Optimizer to produce the strength of both variants.

In HPSOGWO, the proposed mathematical equations update the position of the first three agents in the search space. We regulate the discovery and exploitation of the grey wolf in the search space by inertia constant instead of using usual mathematical equations. The altered set of regulatory equations:

$$\vec{d}_{\alpha}=|\vec{c_1}\cdot\vec{x}_{\alpha}-w*\vec{x}| \tag{16}$$

$$\vec{d}_{\beta}=|\vec{c_2}\cdot\vec{x}_{\beta}-w*\vec{x}| \tag{17}$$

$$\vec{d}_{\delta} = |\vec{c_3} \cdot \vec{x}_{\delta} - w * \vec{x}| \tag{18}$$

Here $\vec{d}_{\alpha}$, $\vec{d}_{\beta}$ and $\vec{d}_{\delta}$ are the distances of α, β and δ respectively from the prey. Here c_1, c_2 and c_3 are acceleration coefficients, w is inertial parameter (weight). This parameter affects the movement propagation given by the last velocity value. Here $\vec{x}_{\alpha}$, $\vec{x}_{\beta}$ and $\vec{x}_{\delta}$ indicates the position vector of α, β and δ respectively and $\vec{x}$ indicates the position vector of a grey wolf.

The velocity and updated equation are suggested as follows in order to combine PSO and GWO variants:

$$v_i^{k+1} = w * (v_i^k + c_1 r_1 (x_1 - x_i^k) + c_2 r_2 (x_2 - x_i^k) + c_3 r_3 (x_3 - x_i^k)) \tag{19}$$

Here v is the particle velocity, v_i^k indicates the current iteration and v_i^{k+1} indicates the next iteration and w is inertial parameter (weight). Here c_1, c_2 and c_3 are acceleration coefficients and r_1, r_2 and r_3 are random numbers, where $0 \leq \text{rand} \leq 1$ and they control the influence of each value. Here i indicates the particle and x indicates the particle's best position and x_i^k indicates the updated position of particle. For each particle i, it will calculate the velocity according to the equation given above.

$$x_i^{k+1} = x_i^k + v_i^k \tag{20}$$

The particle's position will be updated by the equation given above.

Pseudocode of Modified Hybrid Algorithm (HPSOGWO) [40]

Initialization

Initialize *l*,*a*,*w* and *c*

//w = 0.5 + rand()/2

Evaluate the fitness of agents by using

$\vec{d}_{\alpha} = |\vec{c}_1 \cdot \vec{x}_{\alpha} - w * \vec{x}|$

$\vec{d}_{\beta} = |\vec{c}_2 \cdot \vec{x}_{\beta} - w * \vec{x}|$

$\vec{d}_{\delta} = |\vec{c}_3 \cdot \vec{x}_{\delta} - w * \vec{x}|$

while (t < max no. of iter)

For each search agent

Calculate objective function for each search agent

Update the velocity and position by using

$v_i^{k+1} = w * (v_i^k + c_1 r_1 (x_1 - x_i^k) + c_2 r_2 (x_2 - x_i^k) + c_3 r_3 (x_3 - x_i^k))$

$x_i^{k+1} = x_i^k + v_i^k$

end for

Update *l*,*a*,*w* and *c*

Evaluate the fitness of all search agents

$t = t + 1$

end while

return// first best search agent position

Chapter 5

Software Tools and Experiment Setup

5.1 Tool

We have used Microsoft excel for the storage of the dataset. For the analysis and implementation of algorithms, we have used MATLAB R2018a.

Microsoft Excel

Microsoft Excel is a Microsoft developed spreadsheet for Windows, MacOS, Android and iOS. It works with calculation, graphing tools, pivot tables, and a micro programming language of Visual Basic for Applications. Excel is component of the software suite for Microsoft Office.

MATLAB

MATLAB is a computing structure of high performance. It integrates computing, visualization, and programming into a user-friendly environment where identifiable mathematical notation represents problems and responses. MATLAB is an interactive structure whose basic element of data is a cluster that requires no dimensioning.

5.2 10-fold cross validation

Cross-validation is a technique for improving frequent holdout. Cross-validation is an extensive technique of performing ongoing holdout that enhances it efficiently by reducing the variance of assessment.

We're taking a practice set and creating a classifier. Then we're looking to evaluate the effectiveness of that classifier, and there's some degree of variance in that evaluation because it's all below the statistics. In the evaluation, we want to keep the distinction as low as possible. Cross-validation is a way of reducing the variance and is further decreased by a cross-validation variant called "stratified cross-validation."

In the cross-validation the dataset is divided into 10 parts. At that point, 9 parts are used to train and one part to test. At that point we carry another 9 sections with a similar detachment and use them for training and experimentation with the hold-out. We do the entire procedure 10 times, every time we use a distinct area to test. In different cases, we split the dataset into 10 sections, and after that we keep each piece for observing, preparing on the rest, and averaging the 10 results. That would be a "cross-validation of multiple times." Divide the dataset into 10 sections, thereby keeping each section of dataset and evaluating the results. Therefore, every datum point in the dataset is utilized for testing once and for preparing multiple times. That is a cross-validation of multiple times.

5.3 Ratio Comparison

Comparison of ratio is the ratio taken using cross validation.

In our research, we have used ratios as follows:

1. **90:10** where 90% is training set and 10% is testing
2. **80:20** where 80% is training set and 20% is testing
3. **70:30** where 70% is training set and 30% is testing
4. **60:40** where 60% is training set and 40% is testing

5.4 Performance Parameters

We have used 11 different parameters in this research. On the basis of these parameters we have obtained the most accurate classifier and optimization technique. List of parameters as follows:

1. Sensitivity (True positive rate): Sensitivity (SN) is calculated as the number of true positive predictions (TP) divided by the total number of positives (P) i.e., the summation of true positive and false negative (TP + FN). It is also referred as true positive rate (TPR). The highest sensitivity is 1.0, while the worst is 0.0.

$$\mathrm{SN} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} = \frac{\mathrm{TP}}{\mathrm{P}} \tag{21}$$

2. Specificity (True negative rate): Specificity (SP) is calculated as the number of true negative predictions (TN) divided by the total number of negatives (N) i.e., the summation of true negative and false positive (TN + FP). It is also referred to as the true negative rate (TNR). The highest specificity is 1.0, while the worst is 0.0.

$$\mathrm{SP} = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FP}} = \frac{\mathrm{TN}}{\mathrm{N}} \tag{22}$$

3. Accuracy: Accuracy (ACC) is calculated as the number of all true predictions (TP + TN) divided by the total number of data sets (P + N) i.e., the summation of all true predictions (TP +TN) and all true false predictions (FN + FP). The highest accuracy is 1.0, while the worst is 0.0. It can also be calculated by 1-ERR.

$$\mathrm{ACC} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FN} + \mathrm{FP}} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{P} + \mathrm{N}} \tag{23}$$

4. PPV (Positive Predictive Value or Precision): PPV is calculated as the number of true positive predictions (TP) divided by the total number of positive predictions (TP + FP). It is also referred to as Precision (PREC). The best precision or PPV value is 1.0, whereas the worst is 0.0.

$$\mathrm{PREC} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \quad (24)$$

5. NPV (Negative Predictive Value): NPV is calculated as the number of true negative predictions (TN) divided by the total number of negative predictions (TN + FN). The best NPV value is 1.0 and the worst is 0.0.

$$\mathrm{NPV} = \frac{\mathrm{TN}}{(\mathrm{FN}+\mathrm{TN})} \quad (25)$$

6. FPR (False positive rate): FPR is calculated as the number of false positive predictions (FP) divided by the total number of negatives (N) i.e., the summation of true negative and false positive (TN + FP). The best false positive rate is 0.0 while the worst is 1.0. It can also be calculated as 1 – Specificity.

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{TN} + \mathrm{FP}} = 1 - \mathrm{SP}$$

(26)

7. FNR (False negative rate): FNR is calculated as the number of false negative predictions (FN) divided by the total number of positives (P) i.e., the summation of true positive and false negative (TP + FN). It can also be calculated as 1 – Sensitivity.
8. FDR (False discovery rate):- FDR is calculated as the number of false predictions (FP) divided by the total number of positive predictions (TP + FP). It can also be calculated as 1 – PPV.
9. AUC (Area Under the ROC Curve): AUC is used in the analysis of classification to determine which of the models used best predicts the classes. ROC curves are an example of its application. AUC curve is a classification problem performance measurement at different threshold settings. ROC is a curve of probability and AUC is a degree or

separability measure. It states how much model is capable of distinguishing between classes. The higher the AUC, the better the prediction of the model.

10. Recall: Recall is calculated as the number of true positive predictions (TP) divided by the total number of positives (P) i.e., the summation of true positive (TP) and false negative (FN). The highest recall value is 1.0, while the worst is 0.0.

$$\text{Recall} = \frac{True\ Positive}{True\ Positive + False\ Negative} \tag{27}$$

Chapter 6

Results and Discussion

6.1 Cleveland Dataset Output

To optimize the heart disease data, three different types of data were used, and three different kinds of optimization techniques were used to distinguish the features. Furthermore, four distinct classifiers give the prediction results. For different training and testing ratios (90:10, 80:20, 70:30 and 60:40), 10-fold cross validation is performed. Performance parameters like sensitivity, specificity, accuracy, PPV, NPV, FPR, FNR, FDR, AUC, precision and recall are calculated for different ratios. Table 1 shows detailed results for Cleveland output data. Parameters like sensitivity, specificity, accuracy, PPV, NPV, FPR, FNR, FDR, AUC, precision and recall are calculated for different training-testing ratios with different classifiers. The highest accuracy and AUC values are highlighted with different colors. In the tables, light blue color indicates highest AUC among ratio whereas dark blue color indicates the highest AUC for the classifier. Similarly, yellow color represents the highest accuracy for the ratio and peach color represents the highest accuracy for the classifier. When we used SVM classifier with the grey wolf optimizer we got the highest accuracy 0.83457 for 60:40 ratio. Similarly, for AUC, we got the highest value 0.89635 when we used the combination of SVM classifier with the grey wolf optimizer for the ratio 60:40. For 70:30 ratio, we got highest accuracy 0.89781 when we used the combination of Gaussian kernel classifier with the oppositional learning based grey wolf optimizer. Similarly, for AUC, we got highest value 0.89478 when Gaussian kernel is used with the oppositional learning based grey wolf optimizer. As is shown in Table 1, Sensitivity is 1 when we use Radial Basis Kernel with 60:40 and 70:30 ratio. Here, False positive ratio is 0.2 in Naive Bayes and Random Forest, which is quite good, and we can say that these classifiers are more accurate. False negative ratio is seen best in the classifier Radial Basis Kernel, Gaussian Kernel and Random Forest with the value 0, so we can say the combination with these classifiers are more accurate.

Table 1: Detailed results of Cleveland dataset for Ratio (60:40) and (70:30)

Ratio	**(60:40)**				**(70:30)**			
Optimizers	**OGWO**	**GWO**	**No Optimization**	**HPSOGWO**	**OGWO**	**GWO**	**No Optimization**	**HPSOGWO**
Classifier: SVM								
Sensitivity	0.78049	0.87805	0.86585	0.82927	0.82927	0.85366	0.89024	0.85366
Specificity	0.68116	0.73913	0.72464	0.68116	0.68116	0.71014	0.69565	0.71014
Accuracy	0.7351	0.83457	0.80132	0.76159	0.76159	0.78808	0.80132	0.78808
PPV	0.74419	0.8	0.78889	0.75556	0.75556	0.77778	0.7766	0.77778
NPV	0.72308	0.83607	0.81967	0.77049	0.77049	0.80328	0.84211	0.80328
FPR	0.31884	0.26087	0.27536	0.31884	0.31884	0.28986	0.30435	0.28986
FNR	0.21951	0.12195	0.13415	0.17073	0.17073	0.14634	0.10976	0.14634
FDR	0.25581	0.2	0.21111	0.24444	0.24444	0.22222	0.2234	0.22222
AUC	0.78544	0.89635	0.88671	0.83669	0.83015	0.86709	0.8936	0.83369
Precision	0.74419	0.8	0.78889	0.75556	0.75556	0.77778	0.7766	0.77778
Recall	0.78049	0.87805	0.86585	0.82927	0.82927	0.85366	0.89024	0.85366
Classifier: SVM with Radial Basis Kernel								
Sensitivity	1	1	0.86585	1	1	1	0.89024	1
Specificity	0.18841	0.18841	0.72464	0.18841	0.3913	0.3913	0.69565	0.3913
Accuracy	0.62914	0.62914	0.80132	0.62914	0.72185	0.72185	0.80132	0.72185
PPV	0.5942	0.5942	0.78889	0.5942	0.66129	0.66129	0.7766	0.66129
NPV	1	1	0.81967	1	1	1	0.84211	1
FPR	0.81159	0.81159	0.27536	0.81159	0.6087	0.6087	0.30435	0.6087
FNR	0	0	0.13415	0	0	0	0.10976	0
FDR	0.4058	0.4058	0.21111	0.4058	0.33871	0.33871	0.2234	0.33871
AUC	0.39113	0.39113	0.88671	0.39113	0.66914	0.66914	0.8936	0.66914
Precision	0.5942	0.5942	0.78889	0.5942	0.66129	0.66129	0.7766	0.66129

Recall	1	1	0.86585	1	1	1	0.89024	1
Classifier: SVM with Gaussian Kernel								
Sensitivity	0.9878	0.9878	0.86585	0.9878	0.90244	0.9878	0.89024	0.9878
Specificity	0	0	0.72464	0	0.18841	0	0.69565	0
Accuracy	0.53642	0.53642	0.80132	0.53642	0.89781	0.53642	0.80132	0.53642
PPV	0.54	0.54	0.78889	0.54	0.56923	0.54	0.7766	0.54
NPV	0	0	0.81967	0	0.61905	0	0.84211	0
FPR	1	1	0.27536	1	0.81159	1	0.30435	1
FNR	0.012195	0.012195	0.13415	0.012195	0.097561	0.012195	0.10976	0.012195
FDR	0.46	0.46	0.21111	0.46	0.43077	0.46	0.2234	0.46
AUC	0.57989	0.56716	0.88671	0.58501	0.89478	0.58519	0.8936	0.58519
Precision	0.54	0.54	0.78889	0.54	0.56923	0.54	0.7766	0.54
Recall	0.9878	0.9878	0.86585	0.9878	0.90244	0.9878	0.89024	0.9878
Classifier: K-Nearest Neighbor								
Sensitivity	0.79268	0.79268	0.86585	0.79268	0.85366	0.85366	0.89024	0.85366
Specificity	0.5942	0.5942	0.72464	0.5942	0.71014	0.71014	0.69565	0.71014
Accuracy	0.70199	0.70199	0.80132	0.70199	0.78808	0.78808	0.80132	0.78808
PPV	0.69892	0.69892	0.78889	0.69892	0.77778	0.77778	0.7766	0.77778
NPV	0.7069	0.7069	0.81967	0.7069	0.80328	0.80328	0.84211	0.80328
FPR	0.4058	0.4058	0.27536	0.4058	0.28986	0.28986	0.30435	0.28986
FNR	0.20732	0.20732	0.13415	0.20732	0.14634	0.14634	0.10976	0.14634
FDR	0.30108	0.30108	0.21111	0.30108	0.22222	0.22222	0.2234	0.22222
AUC	0.69344	0.69344	0.88671	0.69344	0.7819	0.7819	0.8936	0.7819
Precision	0.69892	0.69892	0.78889	0.69892	0.77778	0.77778	0.7766	0.77778
Recall	0.79268	0.79268	0.86585	0.79268	0.85366	0.85366	0.89024	0.85366
Classifier: Naive Bayes								
Sensitivity	0.79268	0.81707	0.86585	0.80488	0.85366	0.87805	0.89024	0.84146
Specificity	0.76812	0.78261	0.72464	0.78261	0.65217	0.62319	0.69565	0.76812
Accuracy	0.78146	0.88457	0.80132	0.7947	0.76159	0.76159	0.80132	0.80795
PPV	0.80247	0.81707	0.78889	0.81481	0.74468	0.73469	0.7766	0.81176
NPV	0.75714	0.78261	0.81967	0.77143	0.78947	0.81132	0.84211	0.80303
FPR	0.23188	0.21739	0.27536	0.21739	0.34783	0.37681	0.30435	0.23188
FNR	0.20732	0.18293	0.13415	0.19512	0.14634	0.12195	0.10976	0.15854

FDR	0.19753	0.18293	0.21111	0.18519	0.25532	0.26531	0.2234	0.18824
AUC	0.87222	0.886	0.88671	0.87964	0.85666	0.86214	0.8936	0.8685
Precision	0.80247	0.81707	0.78889	0.81481	0.74468	0.73469	0.7766	0.81176
Recall	0.79268	0.81707	0.86585	0.80488	0.85366	0.87805	0.89024	0.84146
Classifier: Random Forest								
Sensitivity	0.89024	0.91463	0.91463	0.90244	0.91463	0.91463	0.92683	0.93902
Specificity	0.75362	0.73913	0.73913	0.73913	0.7971	0.78261	0.7971	0.78261
Accuracy	0.82781	0.83444	0.83444	0.82781	0.86093	0.8543	0.86755	0.86755
PPV	0.81111	0.80645	0.80645	0.80435	0.8427	0.83333	0.84444	0.83696
NPV	0.85246	0.87931	0.87931	0.86441	0.8871	0.88525	0.90164	0.91525
FPR	0.24638	0.26087	0.26087	0.26087	0.2029	0.21739	0.2029	0.21739
FNR	0.10976	0.085366	0.085366	0.097561	0.085366	0.085366	0.073171	0.060976
FDR	0.18889	0.19355	0.19355	0.19565	0.1573	0.16667	0.15556	0.16304
AUC	0.92259	0.92365	0.91861	0.92595	0.93726	0.93982	0.93522	0.94406
Precision	0.81111	0.80645	0.80645	0.80435	0.8427	0.83333	0.84444	0.83696
Recall	0.89024	0.91463	0.91463	0.90244	0.91463	0.91463	0.92683	0.93902

Table 2 below shows the detailed results for Cleveland output data for the ratio 80:20 and 90:10. Parameters like sensitivity, specificity, accuracy, PPV, NPV, FPR, FNR, FDR, AUC, precision and recall are calculated with different classifiers.

In Table 2, when we used SVM with Radial basis kernel, we get the highest accuracy among the classifier in the ratio 90-10 for HPSOGWO with 0.90728 value. For AUC, we get the highest value 0.98277 among the classifiers in the Improved SVM with radial basis kernel for 90-10 ratio. For KNN, we got the highest accuracy 0.91391 for the ratio 90-10 using HPSOGWO optimization technique. Similarly, for AUC, we got the highest value 0.91154 for KNN classifier for the ratio 90-10 with HPSOGWO. For Random Forest, we got the highest accuracy among all the classifiers with 0.96026 for 90:10 ratio with GWO optimization technique. For AUC, we got the highest value 0.99434 in random forest classifier for the ratio 90:10 with OGWO optimization technique. In the dataset of Cleveland for ratio 80:20 and 90:10, False positive ratio is 0.0 in SVM with Gaussian Kernel which is best. Here, Sensitivity is also near to 1 for all the classifiers.

Table 2: Detailed results of Cleveland dataset for Ratio (80:20) and (90:10)

Ratio	**(80:20)**				**(90:10)**			
Optimizers	**OGWO**	**GWO**	**No Optimization**	**HPSOGWO**	**OGWO**	**GWO**	**No Optimization**	**HPSOGWO**
Classifier: SVM								
Sensitivity	0.81707	0.86585	0.87805	0.76829	0.86585	0.81707	0.85366	0.85366
Specificity	0.72464	0.73913	0.76812	0.65217	0.73913	0.75362	0.75362	0.75362
Accuracy	0.77483	0.80795	0.80781	0.71523	0.80795	0.78808	0.80795	0.80795
PPV	0.77907	0.79775	0.81818	0.72414	0.79775	0.79762	0.8046	0.8046
NPV	0.76923	0.82258	0.84127	0.70313	0.82258	0.77612	0.8125	0.8125
FPR	0.27536	0.26087	0.23188	0.34783	0.26087	0.24638	0.24638	0.24638
FNR	0.18293	0.13415	0.12195	0.23171	0.13415	0.18293	0.14634	0.14634
FDR	0.22093	0.20225	0.18182	0.27586	0.20225	0.20238	0.1954	0.1954
AUC	0.84641	0.85755	0.89201	0.79604	0.85967	0.87363	0.84466	0.85949
Precision	0.77907	0.79775	0.81818	0.72414	0.79775	0.79762	0.8046	0.8046
Recall	0.81707	0.86585	0.87805	0.76829	0.86585	0.81707	0.85366	0.85366
Classifier: SVM with Radial Basis Kernel								
Sensitivity	1	1	0.87805	1	1	1	0.85366	1
Specificity	0.5942	0.5942	0.76812	0.5942	0.7971	0.7971	0.75362	0.7971
Accuracy	0.81457	0.81457	0.82781	0.81457	0.90728	0.90728	0.80795	0.90728
PPV	0.74545	0.74545	0.81818	0.74545	0.85417	0.85417	0.8046	0.85417
NPV	1	1	0.84127	1	1	1	0.8125	1
FPR	0.4058	0.4058	0.23188	0.4058	0.2029	0.2029	0.24638	0.2029
FNR	0	0	0.12195	0	0	0	0.14634	0
FDR	0.25455	0.25455	0.18182	0.25455	0.14583	0.14583	0.1954	0.14583
AUC	0.85861	0.85861	0.89201	0.85861	0.98277	0.98277	0.89466	0.98277
Precision	0.74545	0.74545	0.81818	0.74545	0.85417	0.85417	0.8046	0.85417
Recall	1	1	0.87805	1	1	1	0.85366	1
Classifier: SVM with Gaussian Kernel								

Sensitivity	0.12195	0.12195	0.87805	0.12195	0.04878	0.04878	0.85366	0.04878
Specificity	0.86957	0.86957	0.76812	0.86957	0.91304	0.91304	0.75362	0.91304
Accuracy	0.46358	0.46358	0.54642	0.46358	0.44371	0.44371	0.80795	0.44371
PPV	0.52632	0.52632	0.81818	0.52632	0.4	0.4	0.8046	0.4
NPV	0.45455	0.45455	0.84127	0.45455	0.44681	0.44681	0.8125	0.44681
FPR	0.13043	0.13043	0.23188	0.13043	0.086957	0.086957	0.24638	0.086957
FNR	0.87805	0.87805	0.12195	0.87805	0.95122	0.95122	0.14634	0.95122
FDR	0.47368	0.47368	0.18182	0.47368	0.6	0.6	0.1954	0.6
AUC	0.53093	0.53093	0.89201	0.53093	0.45069	0.45069	0.60248	0.45069
Precision	0.52632	0.52632	0.81818	0.52632	0.4	0.4	0.8046	0.4
Recall	0.12195	0.12195	0.87805	0.12195	0.04878	0.04878	0.85366	0.04878
Classifier: K-Nearest Neighbor								
Sensitivity	0.84146	0.84146	0.87805	0.84146	0.93902	0.93902	0.85366	0.93902
Specificity	0.81159	0.81159	0.76812	0.81159	0.88406	0.88406	0.75362	0.88406
Accuracy	0.82781	0.82781	0.82781	0.82781	0.91391	0.91391	0.80795	0.91391
PPV	0.84146	0.84146	0.81818	0.84146	0.90588	0.90588	0.8046	0.90588
NPV	0.81159	0.81159	0.84127	0.81159	0.92424	0.92424	0.8125	0.92424
FPR	0.18841	0.18841	0.23188	0.18841	0.11594	0.11594	0.24638	0.11594
FNR	0.15854	0.15854	0.12195	0.15854	0.060976	0.060976	0.14634	0.060976
FDR	0.15854	0.15854	0.18182	0.15854	0.094118	0.094118	0.1954	0.094118
AUC	0.82653	0.82653	0.89201	0.82653	0.91154	0.91154	0.89466	0.91154
Precision	0.84146	0.84146	0.81818	0.84146	0.90588	0.90588	0.8046	0.90588
Recall	0.84146	0.84146	0.87805	0.84146	0.93902	0.93902	0.85366	0.93902
Classifier: Naive Bayes								
Sensitivity	0.78049	0.81707	0.87805	0.76829	0.87805	0.89024	0.85366	0.87805
Specificity	0.65217	0.76812	0.76812	0.68116	0.73913	0.68116	0.75362	0.63768
Accuracy	0.72185	0.7947	0.80781	0.72848	0.81457	0.7947	0.80795	0.76821
PPV	0.72727	0.80723	0.81818	0.74118	0.8	0.76842	0.8046	0.74227
NPV	0.71429	0.77941	0.84127	0.71212	0.83607	0.83929	0.8125	0.81481
FPR	0.34783	0.23188	0.23188	0.31884	0.26087	0.31884	0.24638	0.36232
FNR	0.21951	0.18293	0.12195	0.23171	0.12195	0.10976	0.14634	0.12195
FDR	0.27273	0.19277	0.18182	0.25882	0.2	0.23158	0.1954	0.25773
AUC	0.83934	0.86886	0.89201	0.79198	0.8982	0.88317	0.89466	0.87416

Precision	0.72727	0.80723	0.81818	0.74118	0.8	0.76842	0.8046	0.74227
Recall	0.78049	0.81707	0.87805	0.76829	0.87805	0.89024	0.85366	0.87805
Classifier: Random Forest								
Sensitivity	0.93902	0.95122	0.93902	0.95122	0.97561	0.95122	0.97561	0.96341
Specificity	0.86957	0.86957	0.85507	0.85507	0.92754	0.97101	0.94203	0.94203
Accuracy	0.90728	0.91391	0.90066	0.90728	0.95364	0.96026	0.96026	0.95364
PPV	0.89535	0.89655	0.88506	0.88636	0.94118	0.975	0.95238	0.95181
NPV	0.92308	0.9375	0.92188	0.93651	0.9697	0.94366	0.97015	0.95588
FPR	0.13043	0.13045	0.14493	0.14493	0.072464	0.028986	0.057971	0.057971
FNR	0.060976	0.04878	0.060976	0.04878	0.02439	0.04878	0.02439	0.036585
FDR	0.10465	0.10345	0.11494	0.11364	0.058824	0.025	0.047619	0.048193
AUC	0.96165	0.95422	0.95688	0.96121	0.99434	0.99364	0.99205	0.99373
Precision	0.89535	0.89655	0.88506	0.88696	0.94118	0.975	0.95238	0.95181
Recall	0.93902	0.95122	0.93902	0.95122	0.97561	0.95122	0.97561	0.96341

Table 3 shows the output of Cleveland data evaluated on the dataset of heart disease. The analysis has been done on different ratios like 90:10, 80:20, 70:30 and 60:40 respectively. The result is then classified using classifiers like Support Vector Machine (SVM), Radial Basis Kernel, Gaussian Kernel, KNN and, Random Forest. Optimization techniques like GWO, OGWO and HPSOGWO were applied so as to optimize the random dataset. Along with that, Raw data is also taken into consideration. A comparative result is presented so as to comprehend the analysis. For Cleveland, the highest accuracy for SVM was 0.83457 and highest AUC was 0.89635. For Radial Basis Kernel, highest accuracy was 0.90728 and highest AUC was 0.98277. For Gaussian Kernel, highest accuracy was 0.89781 and highest AUC was 0.89478. For KNN, 0.91391 was the highest recorded accuracy and 0.91154 was the highest recorded AUC. For Naive Bayes, 0.88457 was the highest accuracy and 0.8982 was the highest AUC. 0.96026 was the highest recorded accuracy for Random Forest classification and 0.99434 was the highest AUC.

Table 3: Highest Accuracy and AUC values for each classifier with best optimization technique for Cleveland Dataset

Ratio	60:40		70:30		80:20		90:10	
Optimizers	**Accuracy**	**AUC**	**Accuracy**	**AUC**	**Accuracy**	**AUC**	**Accuracy**	**AUC**
Classifiers: SVM								
RAW	0.80132	0.88671	0.80132	0.8936	0.80781	0.89201	0.80795	0.84466
OGWO	0.7351	0.78544	0.76159	0.83015	0.77483	0.84641	0.80795	0.85967
GWO	0.83457	0.89635	0.78808	0.86709	0.80795	0.85755	0.78808	0.87363
HPSOGWO	0.76159	0.83669	0.78808	0.83369	0.71523	0.79604	0.80795	0.85949
Classifiers: SVM with Radial Basis Kernel								
RAW	0.80132	0.88671	0.80132	0.8936	0.82781	0.89201	0.80795	0.89466
OGWO	0.62914	0.39113	0.72185	0.66914	0.81457	0.85861	0.90728	0.98277
GWO	0.62914	0.39113	0.72185	0.66914	0.81457	0.85861	0.90728	0.98277
HPSOGWO	0.62914	0.39113	0.72185	0.66914	0.81457	0.85861	0.90728	0.98277
Classifiers: SVM with Gaussian Kernel								
RAW	0.80132	0.88671	0.80132	0.8936	0.54642	0.89201	0.80795	0.60248
OGWO	0.53642	0.57989	0.89781	0.89478	0.46358	0.53093	0.44371	0.45069
GWO	0.53642	0.56716	0.53642	0.58519	0.46358	0.53093	0.44371	0.45069
HPSOGWO	0.53642	0.58501	0.53642	0.58519	0.46358	0.53093	0.44371	0.45069
Classifiers: KNN								
RAW	0.80132	0.88671	0.80132	0.8936	0.82781	0.89201	0.80795	0.89466
OGWO	0.70199	0.69344	0.78808	0.7819	0.82781	0.82653	0.91391	0.91154
GWO	0.70199	0.69344	0.78808	0.7819	0.82781	0.82653	0.91391	0.91154
HPSOGWO	0.70199	0.69344	0.78808	0.7819	0.82781	0.82653	0.91391	0.91154
Classifiers: Naive Bayes								
RAW	0.80132	0.88671	0.80132	0.8936	0.80781	0.89201	0.80795	0.89466
OGWO	0.78146	0.87222	0.76159	0.85666	0.72185	0.83934	0.81457	0.8982
GWO	0.88457	0.886	0.76159	0.86214	0.7947	0.86886	0.7947	0.88317
HPSOGWO	0.7947	0.87964	0.80795	0.8685	0.72848	0.79198	0.76821	0.87416
Classifiers: Random Forest								
RAW	0.83444	0.91861	0.86755	0.93522	0.90066	0.95688	0.96026	0.99205
OGWO	0.82781	0.92259	0.86093	0.93726	0.90728	0.96165	0.95364	0.99434

GWO	0.83444	0.92365	0.8543	0.93982	0.91391	0.95422	0.96026	0.99364
HPSOGWO	0.82781	0.92595	0.86755	0.94406	0.90728	0.96121	0.95364	0.99373

6.2 Hungarian Dataset Output

Table 4 shows the comprehensive output for Hungarian data at 60-40 and 70-30 ratio. The analysis is done on various parameters like specificity, sensitivity, accuracy, AUC, PPV, NPV, FPR, FNR, precision and recall. Highest accuracy and AUC are highlighted in the table. In terms of distinct accuracy and distinguishing, each classifier produces distinct output. If we go through row-wise we get highest accuracy 0.82993 for 60:40 ratio using combination of SVM with Hybrid PSO with GWO. For the same combination we got the highest AUC 0.89241. We also got the highest accuracy 0.63265 for 60:40 ratio with OGWO and radial basis kernel classifier. Also, we got different accuracy for 70:30 ratio such as 0.46939 and 0.84354 for GWO with gaussian kernel and OGWO with Naive Bayes respectively. Similarly, for AUC, we got different AUC such as 0.52609 for 60:40 ratio with GWO and gaussian kernel and also 0.91319 for 70:30 ratio with OGWO and Naive Bayes among all the four classifiers. Negative predictive value is "1" while false negative value is "0" that shows the preciseness of Radial Basis Kernel classifier in GWO optimization technique. False positive ratio is nearly 0.1 for all the classifiers in the ratio 60:40 and 70:30

Table 4: Detailed results of Hungarian dataset for Ratio (60:40) and (70:30)

Ratio	**(60:40)**				**(70:30)**			
Optimizer	**OGWO**	**GWO**	**No Optimization**	**HPSOGWO**	**OGWO**	**GWO**	**No Optimization**	**HPSOGWO**
Classifier: SVM								
Sensitivity	0.7234	0.78723	0.70213	0.81915	0.7766	0.7766	0.80851	0.79787
Specificity	0.86792	0.77358	0.86792	0.84906	0.83019	0.81132	0.81132	0.77358
Accuracy	0.77551	0.78231	0.7619	0.82993	0.79592	0.78912	0.80952	0.78912

PPV	0.90667	0.86047	0.90411	0.90588	0.89024	0.87952	0.88372	0.86207
NPV	0.63889	0.67213	0.62162	0.72581	0.67692	0.67188	0.70492	0.68333
FPR	0.13208	0.22642	0.13208	0.15094	0.16981	0.18868	0.18868	0.22642
FNR	0.2766	0.21277	0.29787	0.18085	0.2234	0.2234	0.19149	0.20213
FDR	0.093333	0.13953	0.09589	0.094118	0.10976	0.12048	0.11628	0.13793
AUC	0.88037	0.82959	0.88017	0.89241	0.88699	0.89944	0.89783	0.85026
Precision	0.90667	0.86047	0.90411	0.90588	0.89024	0.87952	0.88372	0.86207
Recall	0.7234	0.78723	0.70213	0.81915	0.7766	0.7766	0.80851	0.79787
Classifier: SVM with Radial Basis Kernel								
Sensitivity	0.92553	1	0.70213	1	1	0.8617	0.80851	1
Specificity	0.43396	0.20755	0.86792	0.20755	0.39623	0.77358	0.81132	0.39623
Accuracy	0.7483	0.71429	0.7619	0.71429	0.78231	0.82993	0.80952	0.78231
PPV	0.74359	0.69118	0.90411	0.69118	0.74603	0.87097	0.88375	0.74603
NPV	0.76667	1	0.62162	1	1	0.75926	0.70492	1
FPR	0.56604	0.79245	0.13208	0.79245	0.60377	0.22642	0.18868	0.60377
FNR	0.074468	0	0.29787	0	0	0.1383	0.19149	0
FDR	0.25641	0.30882	0.09589	0.30882	0.25397	0.12903	0.11628	0.25397
AUC	0.79647	0.39101	0.88017	0.39101	0.66279	0.87896	0.89783	0.66279
Precision	0.74359	0.69118	0.90411	0.69118	0.74603	0.87097	0.88372	0.74603
Recall	0.92553	1	0.70213	1	1	0.8617	0.80851	1
Classifier: SVM with Gaussian Kernel								
Sensitivity	0.91489	0.042553	0.70213	0.095745	0.010638	0.64894	0.80851	0.095745
Specificity	0.13208	0.9434	0.86792	0.81132	0.9434	0.15094	0.81132	0.81132
Accuracy	0.63265	0.36735	0.7619	0.35374	0.34694	0.46939	0.80952	0.35374
PPV	0.65152	0.57143	0.90411	0.47368	0.25	0.57547	0.88372	0.47368
NPV	0.46667	0.35714	0.62162	0.33594	0.34965	0.19512	0.70492	0.33594
FPR	0.86792	0.056604	0.13208	0.18868	0.056604	0.84906	0.18868	0.18868
FNR	0.085106	0.95745	0.29787	0.90426	0.98936	0.35106	0.19149	0.90426
FDR	0.34848	0.42857	0.09589	0.52632	0.75	0.42453	0.11628	0.52632
AUC	0.42774	0.52609	0.88017	0.30229	0.44099	0.28754	0.89783	0.30229
Precision	0.65152	0.57143	0.90411	0.47368	0.25	0.57547	0.88372	0.47368
Recall	0.91489	0.042553	0.70213	0.095745	0.010638	0.64894	0.80851	0.095745
Classifier: K-Nearest Neighbor								

Sensitivity	0.55319	0.55319	0.70213	0.55319	0.73404	0.73404	0.80851	0.73404
Specificity	0.67925	0.67925	0.86792	0.67925	0.84906	0.84906	0.81132	0.84906
Accuracy	0.59864	0.59864	0.7619	0.59864	0.77551	0.77551	0.80952	0.77551
PPV	0.75362	0.75362	0.90411	0.75362	0.8961	0.8961	0.88372	0.8961
NPV	0.46154	0.46154	0.62162	0.46154	0.64286	0.64286	0.70492	0.64286
FPR	0.32075	0.32075	0.13208	0.32075	0.15094	0.15094	0.18868	0.15094
FNR	0.44681	0.44681	0.29787	0.44681	0.26596	0.26596	0.19149	0.26596
FDR	0.24638	0.24638	0.09589	0.24638	0.1039	0.1039	0.11628	0.1039
AUC	0.61622	0.61622	0.88017	0.61622	0.79155	0.79155	0.89783	0.79155
Precision	0.75362	0.75362	0.90411	0.75362	0.8961	0.8961	0.88372	0.8961
Recall	0.55319	0.55319	0.70213	0.55319	0.73404	0.73404	0.80851	0.73404
Classifier: Naive Bayes								
Sensitivity	0.58511	0.58511	0.70213	0.82979	0.87234	0.7766	0.80851	0.64894
Specificity	0.92453	0.92453	0.86792	0.83019	0.79245	0.84906	0.81132	0.90566
Accuracy	0.70748	0.70748	0.7619	0.82993	0.84354	0.80272	0.80952	0.7415
PPV	0.9322	0.9322	0.90411	0.89655	0.88172	0.90123	0.88372	0.92424
NPV	0.55682	0.55682	0.62162	0.73333	0.77778	0.68182	0.70492	0.59259
FPR	0.075472	0.075472	0.13208	0.16981	0.20755	0.15094	0.18868	0.09434
FNR	0.41489	0.41489	0.29787	0.17021	0.12766	0.2234	0.19149	0.35106
FDR	0.067797	0.067797	0.09589	0.10345	0.11828	0.098765	0.11628	0.075758
AUC	0.84504	0.84504	0.88017	0.90747	0.91319	0.89261	0.89783	0.86772
Precision	0.9322	0.9322	0.90411	0.89655	0.88172	0.90123	0.88372	0.92424
Recall	0.58511	0.58511	0.70213	0.82979	0.87234	0.7766	0.80851	0.64894
Classifier: Random Forest								
Sensitivity	0.31915	0.28723	0.32979	0.32979	0.58511	0.56383	0.57447	0.8617
Specificity	0.96226	0.96226	0.96226	0.96226	0.96226	0.96226	0.96226	0.84906
Accuracy	0.55102	0.53061	0.55782	0.55782	0.72109	0.70748	0.71429	0.85714
PPV	0.9375	0.93103	0.93939	0.93939	0.96491	0.96364	0.96429	0.91011
NPV	0.44348	0.4322	0.44737	0.44737	0.56667	0.55435	0.56044	0.77586
FPR	0.037736	0.037736	0.037736	0.037736	0.037736	0.037736	0.037736	0.15094
FNR	0.68085	0.71277	0.67021	0.67021	0.41489	0.43617	0.42553	0.1383
FDR	0.0625	0.068966	0.060606	0.060606	0.035088	0.036364	0.035714	0.089888
AUC	0.82447	0.84364	0.86491	0.84524	0.90606	0.93456	0.92613	0.92041

Precision	0.9375	0.93103	0.93939	0.93939	0.96491	0.96364	0.96429	0.91011
Recall	0.31915	0.28723	0.32979	0.32979	0.58511	0.56383	0.57447	0.8617

Table 5 shows the detailed output for 80:20 and 90:10 ratio for all the four classifiers and for all the three- optimization technique. Here, raw data is also taken into consideration. Including Accuracy and AUC, different parameters have also been taken into consideration such as: Sensitivity, Specificity, PPV, NPV, FPR, FNR, FDR, Precision and Recall. We got many different accuracy and AUC for each classifier and for each ratio trying many different combinations of classifiers and optimization technique with different ratios. Among all the classifiers and ratios, we get highest accuracy 0.93197 for 90:10 ratio with the combination of KNN classifier and Hybrid PSO with GWO. For AUC, we got highest value 0.98284 for ratio 90:10 with the combination of radial basis kernel and then Hybrid PSO with GWO. Random Forest shows the preciseness with the false positive ratio of 0.0 in the HPSOGWO optimization technique for 90:10 ratio. Sensitivity is near to 1 for both the ratios in Radial Basis Kernel.

Table 5: Detailed results of Hungarian dataset for Ratio (80:20) and (90:10)

Ratio	**(80:20)**				**(90:10)**			
Optimizers	**OGWO**	**GWO**	**No Optimization**	**HPSOGWO**	**OGWO**	**GWO**	**No Optimization**	**HPSOGWO**
Classifier: SVM								
Sensitivity	0.69149	0.78723	0.80851	0.70213	0.81915	0.85106	0.8617	0.85106
Specificity	0.86792	0.81132	0.81132	0.86792	0.86792	0.77358	0.79245	0.77358
Accuracy	0.7551	0.79592	0.80952	0.7619	0.83673	0.82313	0.83673	0.82313
PPV	0.90278	0.88095	0.88372	0.90411	0.91667	0.86957	0.88043	0.86957
NPV	0.61333	0.68254	0.70492	0.62162	0.73016	0.74545	0.76364	0.74545
FPR	0.13208	0.18868	0.18868	0.13208	0.13208	0.22642	0.20755	0.22642
FNR	0.30851	0.21277	0.19149	0.29787	0.18085	0.14894	0.1383	0.14894

FDR	0.097222	0.11905	0.11628	0.09589	0.083333	0.13043	0.11957	0.13043
AUC	0.88117	0.88739	0.89964	0.88077	0.90566	0.93165	0.86469	0.87395
Precision	0.90278	0.88095	0.88372	0.90411	0.91667	0.86957	0.88043	0.86957
Recall	0.69149	0.78723	0.80851	0.70213	0.81915	0.85106	0.8617	0.85106
Classifier: SVM with Radial Basis Kernel								
Sensitivity	1	1	0.80851	0.95745	0.98936	1	0.8617	1
Specificity	0.60377	0.60377	0.81132	0.60377	0.83019	0.79245	0.79245	0.79245
Accuracy	0.85714	0.85714	0.80952	0.82993	0.93197	0.92517	0.83673	0.92517
PPV	0.81739	0.81739	0.88372	0.81081	0.91176	0.89524	0.88043	0.89524
NPV	1	1	0.70492	0.88889	0.97778	1	0.76364	1
FPR	0.39623	0.39623	0.18868	0.39623	0.16981	0.20755	0.20755	0.20755
FNR	0	0	0.19149	0.042553	0.010638	0	0.1383	0
FDR	0.18261	0.18261	0.11628	0.18919	0.088235	0.10476	0.11957	0.10476
AUC	0.86271	0.86271	0.89964	0.91891	0.97411	0.98284	0.91469	0.98284
Precision	0.81739	0.81739	0.88372	0.81081	0.91176	0.89524	0.88043	0.89524
Recall	1	1	0.80851	0.95745	0.98936	1	0.8617	1
Classifier: SVM with Gaussian Kernel								
Sensitivity	0.042553	0.074468	0.80851	0.62766	0.095745	0.095745	0.8617	0.64894
Specificity	0.81132	0.84906	0.81132	0.71698	0.79245	0.79245	0.79245	0.15094
Accuracy	0.31973	0.35374	0.80952	0.90986	0.34694	0.34694	0.55673	0.46939
PPV	0.28571	0.46667	0.88372	0.7973	0.45	0.45	0.88043	0.57547
NPV	0.32331	0.34091	0.70492	0.52055	0.33071	0.33071	0.76364	0.19512
FPR	0.18868	0.15094	0.18868	0.28302	0.20755	0.20755	0.20755	0.84906
FNR	0.95745	0.92553	0.19149	0.37234	0.90426	0.90426	0.1383	0.35106
FDR	0.71429	0.53333	0.11628	0.2027	0.55	0.55	0.11957	0.42453
AUC	0.48866	0.36933	0.89964	0.89437	0.33481	0.33481	0.50469	0.28764
Precision	0.28571	0.46667	0.88372	0.7973	0.45	0.45	0.88043	0.57547
Recall	0.042553	0.074468	0.80851	0.62766	0.095745	0.095745	0.8617	0.64894
Classifier: K-Nearest Neighbor								
Sensitivity	0.80851	0.80851	0.80851	0.80851	0.91489	0.91489	0.8617	0.91489
Specificity	0.88679	0.88679	0.81132	0.88679	0.96226	0.96226	0.79245	0.96226
Accuracy	0.83673	0.83673	0.80952	0.83673	0.93197	0.93197	0.83673	0.93197
PPV	0.92683	0.92683	0.88372	0.92683	0.97727	0.97727	0.88043	0.97727

NPV	0.72308	0.72308	0.70492	0.72308	0.86441	0.86441	0.76364	0.86441
FPR	0.11321	0.11321	0.18868	0.11321	0.037736	0.037736	0.20755	0.037736
FNR	0.19149	0.19149	0.19149	0.19149	0.085106	0.085106	0.1383	0.085106
FDR	0.073171	0.073171	0.11628	0.073171	0.022727	0.022727	0.11957	0.022727
AUC	0.84765	0.84765	0.89964	0.84765	0.93858	0.93858	0.91469	0.93858
Precision	0.92683	0.92683	0.88372	0.92683	0.97727	0.97727	0.88043	0.97727
Recall	0.80851	0.80851	0.80851	0.80851	0.91489	0.91489	0.8617	0.91489
Classifier: Naive Bayes								
Sensitivity	0.73404	0.79787	0.80851	0.79787	0.85106	0.84043	0.8617	0.87234
Specificity	0.88679	0.88679	0.81132	0.88679	0.81132	0.86792	0.79245	0.83019
Accuracy	0.78912	0.82993	0.80952	0.82993	0.83673	0.85034	0.83673	0.85714
PPV	0.92	0.92593	0.88372	0.92593	0.88889	0.9186	0.88043	0.9011
NPV	0.65278	0.71212	0.70492	0.71212	0.75439	0.7541	0.76364	0.78571
FPR	0.11321	0.11321	0.18868	0.11321	0.18868	0.13208	0.20755	0.16981
FNR	0.26596	0.20213	0.19149	0.20213	0.14894	0.15957	0.1383	0.12766
FDR	0.08	0.074074	0.11628	0.074074	0.11111	0.081395	0.11957	0.098901
AUC	0.89181	0.91248	0.89964	0.91248	0.89623	0.91188	0.91469	0.92874
Precision	0.92	0.92593	0.88372	0.92593	0.88889	0.9186	0.88043	0.9011
Recall	0.73404	0.79787	0.80851	0.79787	0.85106	0.84043	0.8617	0.87234
Classifier: Random Forest								
Sensitivity	0.87234	0.67021	0.68085	0.88298	0.92553	0.78723	0.80851	0.88298
Specificity	0.92453	0.96226	0.98113	0.83019	0.84906	0.92453	0.98113	0.98113
Accuracy	0.89116	0.77551	0.78912	0.86395	0.89796	0.83673	0.87075	0.91837
PPV	0.95349	0.96923	0.98462	0.90217	0.91579	0.94872	0.98701	0.9881
NPV	0.80328	0.62195	0.63415	0.8	0.86538	0.71014	0.74286	0.8254
FPR	0.075472	0.037736	0.018868	0.16981	0.15094	0.075472	0.018868	0.018868
FNR	0.12766	0.32979	0.31915	0.11702	0.074468	0.21277	0.19149	0.11702
FDR	0.046512	0.030769	0.015385	0.097826	0.084211	0.051282	0.012987	0.011905
AUC	0.9715	0.92784	0.95173	0.93346	0.9454	0.94811	0.97792	0.98163
Precision	0.95349	0.96923	0.98462	0.90217	0.91579	0.94872	0.98701	0.9881
Recall	0.87234	0.67021	0.68085	0.88298	0.92553	0.78723	0.80851	0.88298

Table 6 shows the Hungarian results taken over Raw data, OGWO, GWO and, HPSOGWO optimization algorithms. Analyzed over the ratio of 90-10, 80-20, 70-30 and 60-40, the data is further classified using various classifiers like KNN, SVM, Radial Basis, Gaussian Kernel, Naïve Bayes and Random Forest. Using SVM, 0.83673 was the highest accuracy noted for 90-10 ratio and 0.93165 was the highest AUC obtained at 90-10 ratio. Using Radial Basis Kernel, the highest accuracy and AUC noted were 0.93197 and 0.98284 respectively for 90-10 ratio. For Gaussian Kernel, again the peak values were observed at 90-10 ratio. The highest noted accuracy and AUC for Gaussian Kernel were 0.90986 and 0.89437 respectively. For KNN, highest accuracy was 0.93197 and AUC was 0.93858, found using HPSOGWO algorithm at 90-10 ratio. Similarly, for Naïve Bayes and Random Forest classifier also, highest accuracy and AUC were observed at 90-10 ratio and HPSOGWO algorithm.

Table 6: Highest Accuracy and AUC values for each classifier with best optimization technique for Hungarian Dataset

Ratio	**60:40**		**70:30**		**80:20**		**90:10**	
Optimizers	**Accuracy**	**AUC**	**Accuracy**	**AUC**	**Accuracy**	**AUC**	**Accuracy**	**AUC**
Classifier: SVM								
RAW	0.7619	0.88017	0.80952	0.89783	0.80952	0.89964	0.83673	0.86469
OGWO	0.77551	0.88037	0.79592	0.88699	0.7551	0.88117	0.83673	0.90566
GWO	0.78231	0.82959	0.78912	0.89944	0.79592	0.88739	0.82313	0.93165
HPSOGWO	0.82993	0.89241	0.78912	0.85026	0.7619	0.88077	0.82313	0.87395
Classifier: Radial Basis Kernel								
RAW	0.7619	0.88017	0.80952	0.89783	0.80952	0.89964	0.83673	0.91469
OGWO	0.7483	0.79647	0.78231	0.66279	0.85714	0.86271	0.93197	0.97411
GWO	0.71429	0.39101	0.82993	0.87896	0.85714	0.86271	0.92517	0.98284
HPSOGWO	0.71429	0.39101	0.78231	0.66279	0.82993	0.91891	0.92517	0.98284
Classifier: Gaussian Kernel								
RAW	0.7619	0.88017	0.80952	0.89783	0.80952	0.88964	0.55673	0.50469
OGWO	0.63265	0.42774	0.34694	0.44099	0.31973	0.48866	0.34694	0.33481

GWO	0.36735	0.52609	0.46939	0.28754	0.35374	0.36933	0.34694	0.33481
HPSOGWO	0.35374	0.30229	0.35374	0.30229	0.90986	0.89437	0.46939	0.28764
Classifier: KNN								
RAW	0.7619	0.88017	0.80952	0.89783	0.80952	0.89964	0.83673	0.91469
OGWO	0.59864	0.61622	0.77551	0.79155	0.83673	0.84765	0.93197	0.93858
GWO	0.59864	0.61622	0.77551	0.79155	0.83673	0.84765	0.93197	0.93858
HPSOGWO	0.59864	0.61622	0.77551	0.79155	0.83673	0.84765	0.93197	0.93858
Classifier: Naive Bayes								
RAW	0.7619	0.88017	0.80952	0.89783	0.80952	0.89964	0.83673	0.91469
OGWO	0.70748	0.84504	0.84354	0.91319	0.78912	0.89181	0.83673	0.89623
GWO	0.70748	0.84504	0.80272	0.89261	0.82993	0.91248	0.85034	0.91188
HPSOGWO	0.82993	0.90747	0.7415	0.86772	0.82993	0.91248	0.85714	0.92874
Classifier: Random Forest								
RAW	0.55782	0.86491	0.71429	0.92613	0.78912	0.95173	0.87075	0.97792
OGWO	0.55102	0.82447	0.72109	0.90606	0.89116	0.9715	0.89796	0.9454
GWO	0.53061	0.84364	0.70748	0.93456	0.77551	0.92784	0.83673	0.94811
HPSOGWO	0.55782	0.84524	0.85714	0.92041	0.86395	0.93346	0.91837	0.98163

6.3 Switzerland Dataset Output

Table 7 displays the detailed analysis of Switzerland data for different training-testing ratios such as 60-40 and 70-30. The analysis is done on the basis of parameters like specificity, sensitivity, precision, recall, AUC, accuracy, FDR, FNR, PPV and NPV. The analysis is done for four different classifiers and three optimization technique. We have tried different combinations of classifiers and optimization technique for each ratio and we got many different values of accuracy and AUC. For accuracy, we got the highest value 0.96721 for 70:30 ratio using radial basis kernel classifier with GWO among all the classifiers. For AUC, we got the highest value 0.89912 for the ratio 70:30 with the combination of radial basis kernel with Hybrid PSO with GWO. We got NaN (Not a Number) values for PPV, FDR and Precision at many places in all the ratio, it happens at the time if the input is beyond the function domain. For example, the logarithm of 0 or the input contains NaN (or unexpected values). In Switzerland Dataset, False positive ratio is 0 for the Random forest for GWO and HPSOGWO which shows the more accuracy.

Table 7: Detailed results of Switzerland dataset for Ratio (60:40) and (70:30)

Ratio	(60:40)				(70:30)			
Optimizers	OGWO	GWO	No Optimization	HPSOGWO	OGWO	GWO	No Optimization	HPSOGWO
Classifier: SVM								
Sensitivity	0	0	0.5	0	0	0	0	0
Specificity	1	1	0.92982	1	1	1	1	1
Accuracy	0.93443	0.93443	0.90164	0.93443	0.93443	0.93443	0.93443	0.93443
PPV	NaN	NaN	0.33333	NaN	NaN	NaN	NaN	NaN
NPV	0.93443	0.93443	0.96364	0.93443	0.93443	0.93443	0.93443	0.93443
FPR	0	0	0.070175	0	0	0	0	0
FNR	1	1	0.5	1	1	1	1	1
FDR	NaN	NaN	0.66667	NaN	NaN	NaN	NaN	NaN
AUC	0.80263	0.80263	0.72807	0.82895	0.57895	0.57895	0.57895	0.57895
Precision	NaN	NaN	0.33333	NaN	NaN	NaN	NaN	NaN
Recall	0	0	0.5	0	0	0	0	0
Classifier: SVM with Radial Basis Kernel								
Sensitivity	0.25	0.25	0.5	0.25	0.5	0.5	0	0.25
Specificity	1	1	0.92982	1	1	1	1	1
Accuracy	0.95082	0.95082	0.90164	0.95082	0.96721	0.96721	0.93443	0.95082
PPV	1	1	0.33333	1	1	1	NaN	1
NPV	0.95	0.95	0.96364	0.95	0.9661	0.9661	0.93443	0.95
FPR	0	0	0.070175	0	0	0	0	0
FNR	0.75	0.75	0.5	0.75	0.5	0.5	1	0.75
FDR	0	0	0.66667	0	0	0	NaN	0
AUC	0.42105	0.42105	0.72807	0.65789	0.71053	0.71053	0.57895	0.89912
Precision	1	1	0.33333	1	1	1	NaN	1
Recall	0.25	0.25	0.5	0.25	0.5	0.5	0	0.25
Classifier: SVM with Gaussian Kernel								
Sensitivity	0	0	0.5	0	0	0	0	0

Specificity	1	1	0.92982	1	1	1	1	1
Accuracy	0.93443	0.93443	0.90164	0.93443	0.93443	0.93443	0.93443	0.93443
PPV	NaN	NaN	0.33333	NaN	NaN	NaN	NaN	NaN
NPV	0.93443	0.93443	0.96364	0.93443	0.93443	0.93443	0.93443	0.93443
FPR	0	0	0.070175	0	0	0	0	0
FNR	1	1	0.5	1	1	1	1	1
FDR	NaN	NaN	0.66667	NaN	NaN	NaN	NaN	NaN
AUC	0.45105	0.45105	0.60789	0.55053	0.55053	0.73789	0.57895	0.56895
Precision	NaN	NaN	0.33333	NaN	NaN	NaN	NaN	NaN
Recall	0	0	0.5	0	0	0	0	0
Classifier: K-Nearest Neighbor								
Sensitivity	0.5	0.5	0.5	0.5	0.5	0.5	0	0.5
Specificity	0.92982	0.92982	0.92982	0.92982	0.94737	0.94737	1	0.94737
Accuracy	0.90164	0.90164	0.90164	0.90164	0.91803	0.91803	0.93443	0.91803
PPV	0.33333	0.33333	0.33333	0.33333	0.4	0.4	NaN	0.4
NPV	0.96364	0.96364	0.96364	0.96364	0.96429	0.96429	0.93443	0.96429
FPR	0.070175	0.070175	0.070175	0.070175	0.052632	0.052632	0	0.052632
FNR	0.5	0.5	0.5	0.5	0.5	0.5	1	0.5
FDR	0.66667	0.66667	0.66667	0.66667	0.6	0.6	NaN	0.6
AUC	0.71491	0.71491	0.72807	0.71491	0.72368	0.72368	0.57895	0.72368
Precision	0.33333	0.33333	0.33333	0.33333	0.4	0.4	NaN	0.4
Recall	0.5	0.5	0.5	0.5	0.5	0.5	0	0.5
Classifier: Naive Bayes								
Sensitivity	0	0	0.5	0	0	0	0	0
Specificity	1	1	0.92982	1	1	1	1	1
Accuracy	0.93443	0.93443	0.90164	0.93443	0.93443	0.93443	0.93443	0.93443
PPV	NaN	NaN	0.33333	NaN	NaN	NaN	NaN	NaN
NPV	0.93443	0.93443	0.96364	0.93443	0.93443	0.93443	0.93443	0.93443
FPR	0	0	0.070175	0	0	0	0	0
FNR	1	1	0.5	1	1	1	1	1
FDR	NaN	NaN	0.66667	NaN	NaN	NaN	NaN	NaN
AUC	0.5614	0.57018	0.72807	0.57018	0.5614	0.5614	0.57895	0.5614
Precision	NaN	NaN	0.33333	NaN	NaN	NaN	NaN	NaN

Recall	0	0	0.5	0	0	0	0	0
Classifier: Random Forest								
Sensitivity	0.25	0.25	0.25	0.25	0	0.25	0.5	0
Specificity	1	0.98246	1	1	1	0.98246	1	1
Accuracy	0.95082	0.93443	0.95082	0.95082	0.93443	0.93443	0.96721	0.93443
PPV	1	0.5	1	1	NaN	0.5	1	NaN
NPV	0.95	0.94915	0.95	0.95	0.93443	0.94915	0.9661	0.93443
FPR	0	0.017544	0	0	0	0.017544	0	0
FNR	0.75	0.75	0.75	0.75	1	0.75	0.5	1
FDR	0	0.5	0	0	NaN	0.5	0	NaN
AUC	0.69956	0.67982	0.69956	0.67105	0.83114	0.85307	0.82237	0.7807
Precision	1	0.5	1	1	NaN	0.5	1	NaN
Recall	0.25	0.25	0.25	0.25	0	0.25	0.5	0

Table 8 shows the detailed output of Switzerland for ratio 80:20 and 90:10 for all the four classifiers and for three optimization technique. Besides, accuracy and AUC, the table also includes many other parameters such as Sensitivity, Specificity, PPV, NPV, FPR, FNR, FDR, Precision and Recall.

For ratio 90:10, we got the highest accuracy 0.99861 by using the combination of random forest and Hybrid PSO with GWO. Similarly, for AUC, after trying many combinations we got the highest value 0.97807 for the ratio 90:10 with radial basis kernel and Hybrid PSO with GWO. False positive ratio is 0 for HPSOGWO for the Random Forest classifier which shows more accuracy.

Table 8: Detailed results of Switzerland dataset for Ratio (80:20) and (90:10)

Ratio	**(80:20)**				**(90:10)**			
Optimizers	**OGWO**	**GWO**	**No Optimization**	**HPSOGWO**	**OGWO**	**GWO**	**No Optimization**	**HPSOGWO**
Classifier: SVM								
Sensitivity	0	0	0	0	0	0	0	0
Specificity	1	1	1	1	1	1	1	1
Accuracy	0.93443	0.93443	0.93443	0.93443	0.93443	0.93443	0.93443	0.93443

PPV	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
NPV	0.93443	0.93443	0.93443	0.93443	0.93443	0.93443	0.93443	0.93443
FPR	0	0	0	0	0	0	0	0
FNR	1	1	1	1	1	1	1	1
FDR	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
AUC	0.47807	0.47807	0.47807	0.47807	0.55702	0.55702	0.55702	0.55702
Precision	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Recall	0	0	0	0	0	0	0	0
Classifier: SVM with Radial Basis Kernel								
Sensitivity	0.5	0.25	0	0.5	0.75	0.75	0	0.75
Specificity	1	1	1	1	1	1	1	1
Accuracy	0.96721	0.95082	0.93443	0.96721	0.98361	0.98361	0.93443	0.98361
PPV	1	1	NaN	1	1	1	NaN	1
NPV	0.9661	0.95	0.93443	0.9661	0.98276	0.98276	0.93443	0.98276
FPR	0	0	0	0	0	0	0	0
FNR	0.5	0.75	1	0.5	0.25	0.25	1	0.25
FDR	0	0	NaN	0	0	0	NaN	0
AUC	0.80702	0.87719	0.47807	0.86842	0.97807	0.97807	0.55702	0.97807
Precision	1	1	NaN	1	1	1	NaN	1
Recall	0.5	0.25	0	0.5	0.75	0.75	0	0.75
Classifier: SVM with Gaussian Kernel								
Sensitivity	0	0	0	0	0	0	0	0
Specificity	1	1	1	1	1	1	1	1
Accuracy	0.93443	0.93443	0.93443	0.93443	0.93443	0.93443	0.93443	0.93443
PPV	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
NPV	0.93443	0.93443	0.93443	0.93443	0.93443	0.93443	0.93443	0.93443
FPR	0	0	0	0	0	0	0	0
FNR	1	1	1	1	1	1	1	1
FDR	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
AUC	0.45105	0.45105	0.47807	0.46895	0.54105	0.54105	0.55702	0.55053
Precision	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Recall	0	0	0	0	0	0	0	0
Classifier: K-Nearest Neighbor								

Sensitivity	0.5	0.5	0	0.5	0.75	0.75	0	0.75
Specificity	1	1	1	1	1	1	1	1
Accuracy	0.96721	0.96721	0.93443	0.96721	0.98361	0.98361	0.93443	0.98361
PPV	1	1	NaN	1	1	1	NaN	1
NPV	0.9661	0.9661	0.93443	0.9661	0.98276	0.98276	0.93443	0.98276
FPR	0	0	0	0	0	0	0	0
FNR	0.5	0.5	1	0.5	0.25	0.25	1	0.25
FDR	0	0	NaN	0	0	0	NaN	0
AUC	0.75	0.75	0.47807	0.75	0.875	0.875	0.55702	0.875
Precision	1	1	NaN	1	1	1	NaN	1
Recall	0.5	0.5	0	0.5	0.75	0.75	0	0.75
Classifier: Naive Bayes								
Sensitivity	0	0	0	0	0	0	0	0
Specificity	1	1	1	1	1	1	1	1
Accuracy	0.93443	0.93443	0.93443	0.93443	0.93443	0.93443	0.93443	0.93443
PPV	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
NPV	0.93443	0.93443	0.93443	0.93443	0.93443	0.93443	0.93443	0.93443
FPR	0	0	0	0	0	0	0	0
FNR	1	1	1	1	1	1	1	1
FDR	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
AUC	0.55702	0.55702	0.47807	0.55702	0.83333	0.83333	0.55702	0.83333
Precision	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Recall	0	0	0	0	0	0	0	0
Classifier: Random Forest								
Sensitivity	0.25	0.5	0.5	0.25	0.25	0.25	0.75	0.25
Specificity	1	1	1	1	0.98246	1	1	1
Accuracy	0.95082	0.96721	0.96721	0.95082	0.93443	0.95082	0.93861	0.99861
PPV	1	1	1	1	0.5	1	1	1
NPV	0.95	0.9661	0.9661	0.95	0.94915	0.95	0.98276	0.95
FPR	0	0	0	0	0.017544	0	0	0
FNR	0.75	0.5	0.5	0.75	0.75	0.75	0.25	0.75
FDR	0	0	0	0	0.5	0	0	0
AUC	0.77851	0.88596	0.87061	0.875	0.94298	0.88158	0.875	0.95175

Precision	1	1	1	1	0.5	1	1	1
Recall	0.25	0.5	0.5	0.25	0.25	0.25	0.75	0.25

Table 9 shows the Switzerland results. For different combination of dataset algorithms and testing-training ratios, different classifiers show different data of accuracy and AUC. For SVM classifier, the highest accuracy obtained was 0.93443 and AUC was 0.82895, both for HPSOGWO. For Radial Basis Kernel, 0.98361 was the highest accuracy and 0.97807 was the highest AUC, obtained for 90-10 ratio. Using Gaussian Kernel, highest accuracy was obtained for GWO and highest AUC was 0.73789 obtained for GWO. Using KNN, 0.98361 was the highest accuracy and 0.99561 was the highest AUC, both obtained for HPSOGWO at 90-10 ratio. Similarly, for Naive Bayes and Random forest also, highest accuracy and AUC were noted at 90-10 ratio.

Table 9: Highest Accuracy and AUC values for each classifier with best optimization technique for Switzerland Dataset

Ratio	**60:40**		**70:30**		**80:20**		**90:10**	
Optimizers	**Accuracy**	**AUC**	**Accuracy**	**AUC**	**Accuracy**	**AUC**	**Accuracy**	**AUC**
Classifier: SVM								
RAW	0.90164	0.72807	0.93443	0.57895	0.93443	0.47807	0.93443	0.55702
OGWO	0.93443	0.80263	0.93443	0.57895	0.93443	0.47807	0.93443	0.55702
GWO	0.93443	0.80263	0.93443	0.57895	0.93443	0.47807	0.93443	0.55702
HPSOGWO	0.93443	0.82895	0.93443	0.57895	0.93443	0.47807	0.93443	0.55702
Classifier: Radial Basis Kernel								
RAW	0.90164	0.72807	0.93443	0.57895	0.93443	0.47807	0.93443	0.55702
OGWO	0.95082	0.42105	0.96721	0.71053	0.96721	0.80702	0.98361	0.97807
GWO	0.95082	0.42105	0.96721	0.71053	0.95082	0.87719	0.98361	0.97807
HPSOGWO	0.95082	0.65789	0.95082	0.89912	0.96721	0.86842	0.98361	0.97807
Classifier: Gaussian Kernel								
RAW	0.90164	0.60789	0.93443	0.57895	0.93443	0.47807	0.93443	0.55702

OGWO	0.93443	0.45105	0.93443	0.55053	0.93443	0.45105	0.93443	0.54105
GWO	0.93443	0.45105	0.93443	0.73789	0.93443	0.45105	0.93443	0.54105
HPSOGWO	0.93443	0.55053	0.93443	0.56895	0.93443	0.46895	0.93443	0.55053
Classifier: KNN								
RAW	0.90164	0.72807	0.93443	0.57895	0.93443	0.47807	0.93443	0.55702
OGWO	0.90164	0.71491	0.91803	0.72368	0.96721	0.75	0.98361	0.875
GWO	0.90164	0.71491	0.91803	0.72368	0.96721	0.75	0.98361	0.875
HPSOGWO	0.90164	0.71491	0.91803	0.72368	0.96721	0.75	0.98361	0.99561
Classifier: Naive Bayes								
RAW	0.90164	0.72807	0.93443	0.57895	0.93443	0.47807	0.93443	0.55702
OGWO	0.93443	0.5614	0.93443	0.5614	0.93443	0.55702	0.93443	0.83333
GWO	0.93443	0.57018	0.93443	0.5614	0.93443	0.55702	0.93443	0.83333
HPSOGWO	0.93443	0.57018	0.93443	0.5614	0.93443	0.55702	0.93443	0.83333
Classifier: Random Forest								
RAW	0.95082	0.69956	0.96721	0.82237	0.96721	0.87061	0.93861	0.875
OGWO	0.95082	0.69956	0.93443	0.83114	0.95082	0.77851	0.93443	0.94298
GWO	0.93443	0.67982	0.93443	0.85307	0.96721	0.88596	0.95082	0.88158
HPSOGWO	0.95082	0.67105	0.93443	0.7807	0.95082	0.875	0.99861	0.95175

Table 10 displays the summary of the Cleveland output using different classifiers and optimizers. After trying many different combinations of all four classifiers with the all the three-optimization technique along with different parameters we got the following results for the best and highest accuracy and AUC. For SVM and Gaussian Kernel classifiers, the best output was classified for GWO and OGWO respectively. However, HPSOGWO was the best optimizer when evaluated using radial basis kernel as well as KNN. For Naive Bayes, best accuracy was noted for GWO and best AUC was noted for OGWO. For Random Forest, best accuracy 0.96026 was obtained for GWO for 90:10 ratio and best AUC 0.99434 was obtained for OGWO for 90:10 ratio. Random Forest gives highest accuracy with the combination of GWO as Random Forest handles large dataset with higher dimensionality, and it won’t overfit the model and GWO need less storage requirement than other techniques and is also easy to implement due to simple structure. If this combination is used in future, it can give effective results.

Table 10: Result Analysis of Cleveland Dataset

Classifier	Optimization Technique	Ratio	Accuracy	AUC
SVM	GWO	(60:40)	0.83457	
	GWO	(60:40)		0.89635
Radial Basis Kernel	HPSOGWO	(90:10)	0.90728	
	HPSOGWO	(90:10)		0.98277
Gaussian kernel	OGWO	(70:30)	0.89781	
	OGWO	(70:30)		0.89478
K-Nearest Neighbor	HPSOGWO	(90:10)	0.91391	
	HPSOGWO	(90:10)		0.91154
Naive Bayes	GWO	(60:40)	0.88457	
	OGWO	(90:10)		0.8982
Random Forest	GWO	(90:10)	0.96026	
	OGWO	(90:10)		0.99434

Table 11 indicates the summary output data for Hungarian. For four different classifiers viz., Gaussian kernel, KNN, Naive Bayes and Random Forest, HPSOGWO was observed to be the most optimum algorithm as best AUC and accuracy were recorded for the same method. For SVM, OGWO displayed the best accuracy and GWO showed the best AUC. For Radial Basis Kernel, best accuracy and AUC were both noted for OGWO and HPSOGWO respectively. Among all the combinations we have tried for the classifiers and optimization technique and also with many combinations of different parameters we obtained the best accuracy for KNN classifier with the value of 0.93197 for the HPSOGWO and 90:10 ratio. In the same manner, we obtained the best value for AUC 0.98284 for Radial Basis Kernel with the HPSOGWO for the 90:10 ratio.

Table 11: Result Analysis for Hungarian Dataset

Classifier	Optimization Technique	Ratio	Accuracy	AUC
SVM	OGWO	(90:10)	0.83673	
	GWO	(90:10)		0.93165

Radial Basis Kernel	OGWO	(90:10)	0.93197	
	HPSOGWO	(90:10)		0.98284
Gaussian kernel	HPSOGWO	(80:20)	0.90986	
	HPSOGWO	(80:20)		0.89437
K-Nearest Neighbor	HPSOGWO	(90:10)	0.93197	
	HPSOGWO	(90:10)		0.93858
Naive Bayes	HPSOGWO	(90:10)	0.85714	
	HPSOGWO	(90:10)		0.92874
Random Forest	HPSOGWO	(90:10)	0.91837	
	HPSOGWO	(90:10)		0.98163

As indicated in Table 12, in the results of Switzerland HPSOGWO gave the best accuracy and AUC output in case of SVM, Radial Basis Kernel, KNN and Random Forest. While, for Naive Bayes, HPSOGWO gave the best accuracy and best AUC was noted for OGWO. For Gaussian Kernel, GWO showed the best accuracy as well as AUC. After trying many different combinations of all the four classifiers with all the three-optimization technique for each ratio we obtained many different values for accuracy and AUC. After analyzing the output table, we obtained the best accuracy with the value of 0.99861 with the combination of Random forest classifier with the HPSOGWO for the ratio 90:10. Same way, when we analyzed the output, we obtained the best AUC 0.99561 with the combination of KNN and the HPSOGWO for the ratio of 90:10.

Table 12: Result Analysis of Switzerland Dataset

Classifier	**Optimization Technique**	**Ratio**	**Accuracy**	**AUC**
SVM	HPSOGWO	(90:10)	0.93443	
	HPSOGWO	(60:40)		0.82895
Radial Basis Kernel	HPSOGWO	(90:10)	0.98361	
	HPSOGWO	(90:10)		0.97807
Gaussian kernel	GWO	(90:10)	0.93443	
	GWO	(70:30)		0.73789
K-Nearest Neighbor	HPSOGWO	(90:10)	0.98361	
	HPSOGWO	(90:10)		0.99561
Naive Bayes	HPSOGWO	(90:10)	0.93443	

	OGWO	(90:10)		0.83333
Random Forest	HPSOGWO	(90:10)	0.99861	
	HPSOGWO	(90:10)		0.95175

6.4 Discussion

Figure 9 indicates the graphical analysis of Cleveland data. Taken over different training-testing ratio, the graph displays the best optimization technique for different ratios. As shown in the graph blue color indicates accuracy and orange color indicates AUC. As shown in the graph, we got highest accuracy 0.96026 using Random Forest with the Grey Wolf Optimizer with the ratio of 90:10. In the same way for AUC, we got highest AUC 0.99434 using Random Forest with Oppositional Learning based Grey Wolf Optimizer with the same ratio of 90:10.

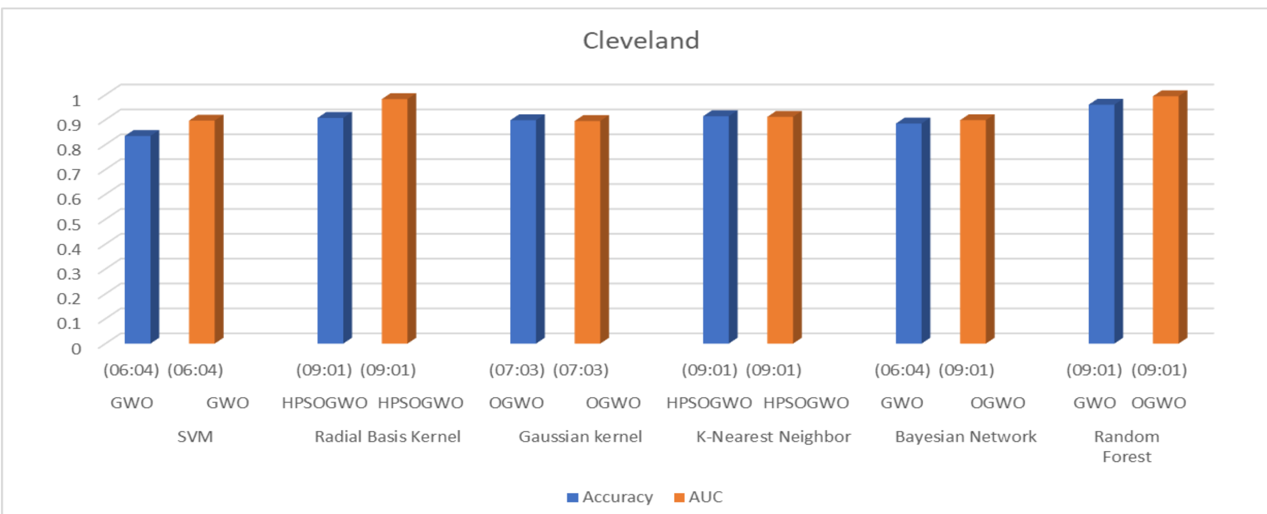

Figure 5: Graphical Analysis of Cleveland Dataset

Figure 10 represents the graph of Hungarian data. The graph represents the best accuracy and AUC values for different training-testing ratios and various optimization algorithms. In the dataset of Hungarian, the highest accuracy we got is 0.93197 using KNN classifier and HPSOGWO optimization technique with the ratio of 90-10. For AUC, we got highest AUC 0.98284 using Radial Basis Kernel classifier and HPSOGWO technique with the ratio of 90-10.

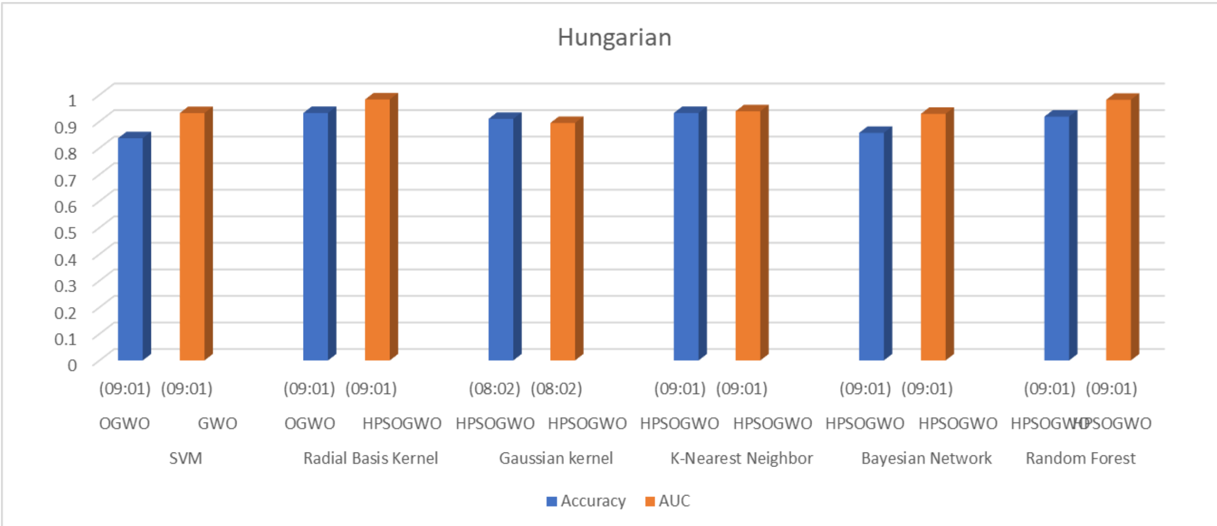

Figure 6: Graphical Analysis of Hungarian Dataset

Figure 11 shows the graph plotted for best accuracy and AUC of Switzerland data. The analysis is done for the datasets of various training-testing ratios. In Switzerland dataset, the highest accuracy 0.99861 is achieved using Random Forest classifier and HPSOGWO optimization technique with the ratio 90-10. For AUC, the highest AUC 0.99561 is achieved using KNN classifier and HPSOGWO optimization technique with the ratio of 90-10.

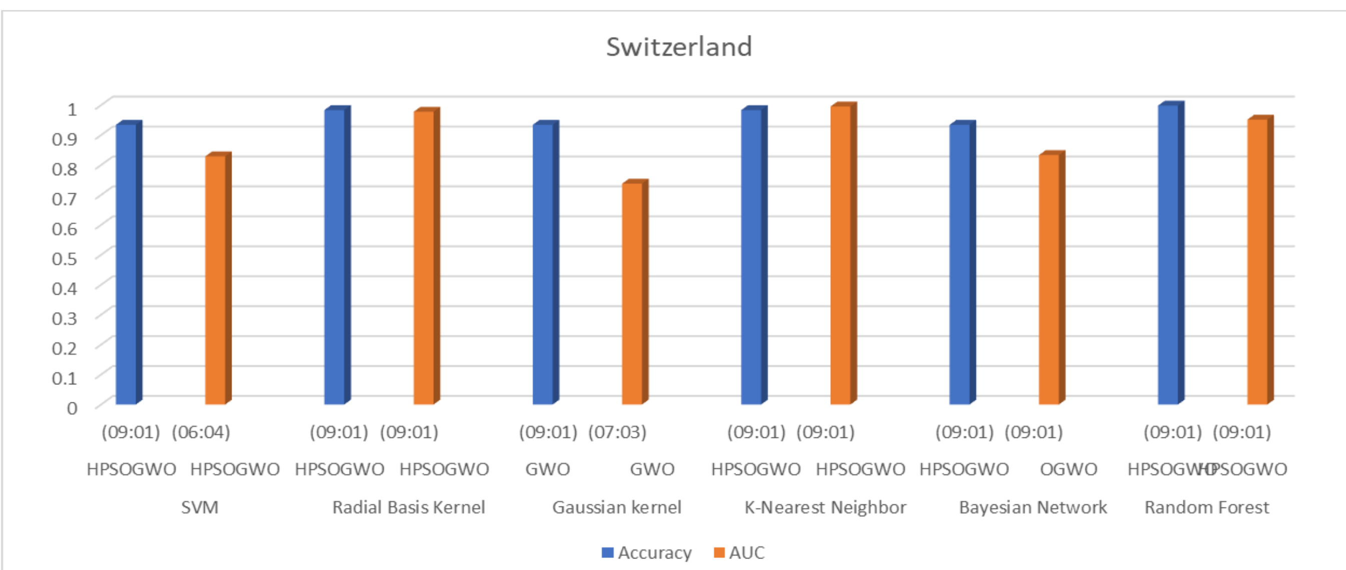

Figure 7: Graphical Analysis of Switzerland Dataset

Figure 12 is the brief of the entire research work presented in this paper. The accuracy for all the three data sets have been taken into consideration and the best of all has been presented in the form of pie chart below. The selected data that is referred data for plotting the pie chart has been mentioned in Table 13. The table reflects the highest values of accuracy for each optimization technique, taking classifiers and ratios into consideration. From the overall analysis, it is clearly seen that in the dataset of Switzerland, highest accuracy of 99.86% was noticed when HPSOGWO was used with the Random Forest classifier with training – testing ratio of 90:10.

Table 13: Combined Results

Optimization Technique	**Accuracy**
GWO/Random Forest - Cleveland (90:10)	0.96026
HPSOGWO/K-Nearest Neighbor - Hungarian (90:10)	0.93197
HPSOGWO/Random Forest - Switzerland (90:10)	0.99861

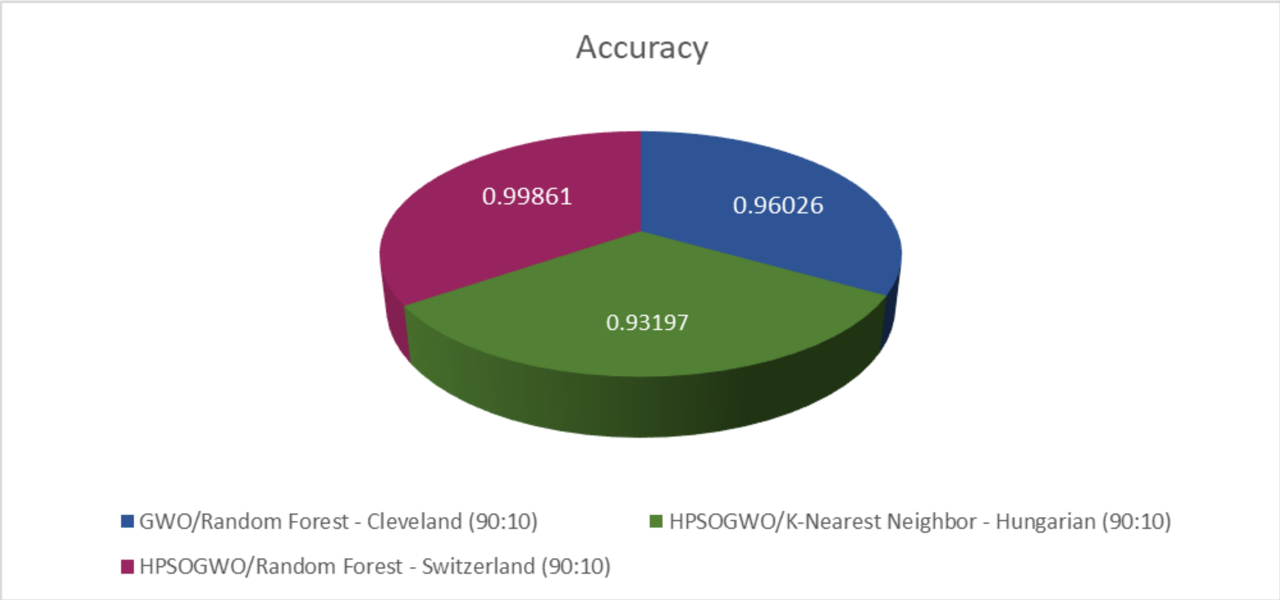

Figure 8: Combined Results Analysis Chart

Chapter 7

Conclusions and Future Work

7.1 Conclusions and Future Work

The hybrid algorithm of PSO with GWO gives the best results for Switzerland and Hungarian datasets. The hybrid algorithm gives an accuracy of 99.86% for Switzerland dataset with Random Forest for training – testing ratio of 90:10. The hybrid algorithm gives an accuracy of 93% for Hungarian dataset with KNN for a training – testing ratio of 90:10. In the case of Cleveland dataset, GWO gives the best accuracy of 96% with Random Forest as compared to the hybrid algorithm which gives an accuracy of 95% for a training – testing ratio of 90:10. Random Forest performed the best in case of Cleveland dataset and Switzerland dataset whereas K – Nearest Neighbor performed best for the Hungarian dataset. In all the three datasets, the training – testing ratio of 90:10 gave us the best trained model. From the above analysis, it is evident that any single classification technique or optimizer cannot be picked up as the best one for a particular dataset. Different combinations of training – testing ratios have different effects on optimization techniques. However, in quite a large number of cases, PSO when hybridized with GWO gave impressive results. Again, this is not the only criterion that serves the outcome. Classifiers also play a major role in generating the results.

In the future, different hybridizations can be tested for different microarray cancer datasets with different classifiers. Different datasets such as Leukemia, Breast Cancer, Lung Cancer, DLBCL can be used with different hybrid optimization techniques such as PSO (Particle Swarm Optimization) can be hybridized with GOA (Grasshopper Optimization Algorithm) for better prediction of cancer microarray data using combination with different classifiers.

References

[1] Rachel Nall, Yamini Ranchod, “What to know about cancer”, 2018.

[2] Laurel J. Kelly, Timothy J. Moynihan “Living with Cancer: Understanding heart cancer”, (2019).

[3] “What is Bioinformatics?”, European Bioinformatics Institute [GB].

[4] “Gene Expression”, Nature Education, 2014.

[5] Dufva M., “Introduction to microarray technology”, 2009, 1-22.

[6] Khushaba, Rami N., Ahmed Al-Ani, and Adel Al-Jumaily. "Feature subset selection using differential evolution and a statistical repair mechanism." Expert Systems with Applications, Vol. 38, No. 9, pp. 11515-11526, 2011.

[7] Moradi Parham, and Mozhgan Gholampour, "A hybrid particle swarm optimization for feature subset selection by integrating a novel local search strategy," Applied Soft Computing, Vol. 43, pp. 117-130, 2016.

[8] Shenkai Gu, Ran Cheng, Yaochu Jin, “Feature selection for high-dimensional classification using a competitive swarm optimizer”, Soft Comput, 07 October 2016.

[9] Shankar K., S. K. Lakshmanaprabu, Deepak Gupta, Andino Maseleno, and Victor Hugo C. de Albuquerque, "Optimal feature-based multi-kernel SVM approach for thyroid disease classification," The Journal of Supercomputing, pp. 1-16, 2018.

[10] https://archive.ics.uci.edu/ml/datasets/heart+Disease

[11] Aijun An, “Classification Methods”, 2005

[12] V. Vapnik. The Nature of Statistical Learning Theory. NY: Springer-Verlag. 1995.

[13] Ebrahim Edriss Ebrahim Ali, Wu Zhi Feng, “Breast Cancer Classification using Support Vector Machine and Neural Network”, International Journal of Science and Research (IJSR), ISSN (Online): 2319-7064

[14] Madhu Sanjeevi, “Support Vector machine with Math”, 2017

[15] Qun Chang Qingcai Chen Xiaolong Wang, “Scaling Gaussian RBF kernel width to improve SVM classification” ITNLP Lab, School of Computer Science and Technology, Harbin Institute of Technology, Harbin, 150001, China.

[16] Ms. Hetal Bhavsar, Dr. Amit Ganatra, “Radial Basis Polynomial Kernel (RBPK): A Generalized Kernel for Support Vector Machine”, International Journal of Computer Science and Information Security (IJCSIS), Vol. 14, No. 4, April 2018.

[17] Vasileios Apostolidis-Afentoulis, Konstantina-Ina Lioufi, “SVM Classification with Linear and RBF Kernels.

[18] Khaled Fawagreh, Mohamed Medhat Gaber & Eyad Elyan, “Random Forests: From early developments to recent advancements”, An open Access Journal, Systems Science & Control Engineering, 2014.

[19] Raul Eulogio, “Introduction to Random Forest”, 12.08.17.

[20] Jason Brownlee, “K-Nearest Neighbors for Machine Learning”, April 15, 2016.

[21] https://www.saedsayad.com/k_nearest_neighbors.htm

[22] Prashant Gupta, “Naïve Bayes in Machine Learning”, Nov 6, 2017.

[23] Jason Brownlee, “Naïve Bayes for Machine Learning”, April 11, 2016.

[24] Rohith Gandhi, “Naïve Bayes Classifier”, May 5,2018.

[25] Xin-She Yang, “Optimization and Metaheuristic Algorithms in Engineering”, in Metaheuristics in Water, Geotechnical and Transport Engineering (Eds. X. S. Yang, A.H. Gandomi, S. Talatahari, A. H. Alavi), Elsevier, (2013), pp. 1-23.

[26] Thomas Weise, “Global Optimization Algorithms Theory and Application”, 2nd Edition Version: 2009-06-26, http://www.it-weise.de/

[27] Xin-She Yang, ”Metaheuristic Optimization”, Cambridge University, 2011

[28] E. Bonabeau, M. Dorigo, and G. Theraulaz, Swarm intelligence: from natural to artificial systems: OUP USA, 1999.

[29] M. Dorigo, M. Birattari, and T. Stutzle, "Ant colony optimization," Computational Intelligence Magazine, IEEE, vol. 1, pp. 28-39, 2006.

[30] J. Kennedy and R. Eberhart, "Particle swarm optimization," in Neural Networks, 1995. Proceedings., IEEE International Conference on, 1995, pp. 1942-1948.

[31] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," Evolutionary Computation, IEEE Transactions on, vol. 1, pp. 67-82, 1997.

[32] Barnali Sahu and Debahuti Mishra, "A Novel Feature Selection Algorithm using Particle Swarm Optimization for Cancer Microarray Data", International Conference on Modeling Optimization and Computing, 2012, 27-31

[33] G. Beni and J. Wang, "Swarm intelligence in cellular robotic systems," in Robots and Biological Systems: Towards a New Bionics, ed: Springer, 1993, pp. 703-712.

[34] B. Basturk and D. Karaboga, "An artificial bee colony (ABC) algorithm for numeric function optimization," in IEEE swarm intelligence symposium, 2006, pp. 12-14.

[35] O. Olorunda and A. P. Engelbrecht, "Measuring exploration/exploitation in particle swarms using swarm diversity," in Evolutionary Computation, 2008. CEC 2008. (IEEE World Congress on Computational Intelligence). IEEE Congress on, 2008, pp. 1128-1134.

[36] E. Alba and B. Dorronsoro, "The exploration/exploitation tradeoff in dynamic cellular genetic algorithms," Evolutionary Computation, IEEE Transactions on, vol. 9, pp. 126-142, 2005.

[37] S. Mirjalili, S. M. Mirjalili, A. Lewis, "Grey Wolf Optimizer", Advances in Engineering Software, vol. 69, pp. 46-61, 2014, DOI: http://dx.doi.org/10.1016/j.advengsoft.2013.12.007.

[38] Jie-Sheng Wang & Shu-Xia-Li. "An Improved Grey Wolf Optimizer Based on Differential Evolution and Elimination Mechanism", a naturesearch journal (2019).

[39] Moumita Pradhan, Provas Kumar Roy, Tandra Pal, "Oppositional based grey wolf optimization algorithm for economic dispatch problem of power system", (2016).

[40] S. Narinder and S. B. Singh, "Hybrid Algorithm of Particle Swarm Optimization and Grey Wolf Optimizer for Improving Convergence Performance", Journal of Applied Mathematics, vol. 2017, pp. 1-15, 2017.

[41] Gupta D., Ahlawat A., "Usability Determination using Multistage Fuzzy System", Procedia Computer Science, ELSEVIER, SCOPUS, DOI: 10.1016/j.procs.2016.02.042.

[42] Gupta D., Ahlawat A. (2016) "Usability Evaluation of Live Auction Portal", International Journal of Control Theory and Applications, SCOPUS. Volume 9, No. 40.

[43] Gupta D., Ahlawat A. (2017). "Usability Prediction of Live Auction using Multistage Fuzzy System", International Journal of Artificial Intelligence and Applications for Smart Devices, Volume 5, No 1.

[44] Gupta D., Ahlawat A., Sagar K. (2017). "Usability Prediction and Ranking of SDLC models using Fuzzy Hierarchical Usability Model", Open Engineering (Central European Journal of Engineering), ESCI, SCOPUS. Volume 7, No. 1.

[45] Gupta D., Khanna A. (2017). "Software Usability Datasets", International Journal of Pure and Applied Mathematics, SCOPUS. Volume 117, No. 15, 1001-1014.

[46] Gupta D., Ahlawat A., Sagar K. (2014). "A critical analysis of a hierarchy-based Usability Model." Contemporary Computing and Informatics (IC3I), 2014 International Conference on. IEEE.

[47] Gupta D., Rodrigues J. J. P. C., Sundaram S., Khanna A., Korotaev V., Albuquerque V. H. C. (2018), "Usability Feature Extraction Using Modified Crow Search Algorithm: A Novel Approach", Neural Computing and Applications, https://doi.org/10.1007/s00521-018-3688-6.

[48] Jason Brownlee, “How to Perform Feature Selection with Machine Learning Data in Weka”, Machine Learning Mastery, 2016.

[49] Gupta D., Ahlawat A. (2017), “Usability Feature Selection via MBBAT: A Novel Approach”, Journal of Computational Sciences, Volume 23, 195-203, Nov 2017.

[50] Jain R., Gupta D., Khanna A. (2018). “Usability Feature Optimization using MWOA”. International Conference on Innovative Computing and Communication (ICICC), Volume 2. [In-Press]

[51] Shankar K., Lakshmanaprabu S. K., Gupta D., Maseleno A., de Albuquerque V. H. C. (2018), "Optimal features-based multi-kernel SVM approach for thyroid disease classification", Journal of Supercomputing, July 2018, https://doi.org/10.1007/s11227-018-2469-4.

[52] Gupta D., Sagar K. (2010) “Remote file synchronization single-round algorithm”, International Journal of Computer Applications, 4(1), 32-36.

[53] Patnaik A., Gupta D. (2010). “Unique identification system”, International Journal of Computer Applications, 7(5).

[54] Xin-She Y. (2009), “Firefly algorithms for multimodal optimization”, in Stochastic Algorithms: Foundations and Applications, SAGA 2009, Lecture Notes in Computer Sciences, Vol. 5792.

[55] Xin-SheY. (2010), “A New Metaheuristic Bat-Inspired Algorithm”, Springer.

[56] Gonzalez-Pardo, Jung J. J., Camacho D. (2017) "ACO-based clustering for Ego Network analysis". Future Generation Computer Systems 66, pp160-170.

[57] Pham D.T., Ghanbarzadeh A., Koç E., Otri S., Rahim S., Zaidi M. (2005). “The Bees Algorithm – A Novel Tool for Complex Optimization”, Manufacturing Engineering Centre, Cardiff University, 2005.

[58] Erik C., Mauricio G., Daniel Z., Marco P. C., Guillermo G. (2012), An Algorithm for Global Optimization Inspired by Collective Animal Behavior, Hindawi Publishing Corporation Discrete Dynamics in Nature and society.

[59] Hussein, A. F., ArunKumar, N., Ramirez-Gonzalez, G., Abdulhay, E., Tavares, J. M. R. S., & de

Albuquerque, V. H. C. (2018). A medical record managing and securing blockchain based system supported by a Genetic Algorithm and Discrete Wavelet Transform. Cognitive Systems Research, 52, 1–11.

[60] Emary E., Yamany W., Hassanein A. E., Snaselc V. (ICCMIT2015), "Multi-Objective Grey-Wolf Optimization for Attribute Reduction," International Conference on Communication, Management and Information Technology.

[61] Vardhana M., Arunkumar N., Abdulhay E, Gustavo R.-G. (2018). “Convolutional neural network for bio- medical image segmentation with hardware acceleration”, Cognitive Systems Research 50: 10-14.