


CUKUROVA UNIVERSITY
ENGINEERING AND ARCHITECTURE FACULTY
DEPARTMENT OF COMPUTER ENGINEERING

GRADUATION THESIS

SUBJECT

Classification with Nature Inspired Optimization Algorithms

By

2015556007 – Koray Aykor

Advisor

Öğr. Gör. Dr. Esin ÜNAL

June – 2021

ADANA

ABSTRACT

The focus of this research is to analyze the test data of individuals with depression disease and to develop a study to diagnose depression disease.

In this thesis study, the target audience is depression disease. The target audience was examined with the test information of the patients taken from the hospital and a data set was prepared with these data. Then, the analysis of the tests performed on the patients was made with this data set and it was aimed to draw a conclusion.

As a result of this thesis, a web environment has been prepared to make the obtained data more orderly and understandable. The images and outputs of the classification and cluster analysis made in the thesis study are shown on this platform. Also, brief information about the algorithms and libraries used in the study is given.

CONTENTS

ICONS AND ABBREVIATIONS

IEEE: Institute of Engineers and Everyone Else

ML: Machine Learning

AI: Artificial Intelligence

KNN: K Nearest Neighborhood

BA: Bat Algorithm

HBA: Hybrid Bat Algorithm

FA: Firefly Algorithm

GWO: Grey Wolf Algorithm

TP: True Positive

TN: True Negative

FP: False Positive

FN: False Negative

FIGURE LIST

CHART LIST

1.OVERVIEW

This project is prepared according to IEEE standard [1]. This document is in content compliance with the IEEE standard 1058-1998 in which the contents of this standard are rearranged and a mapping is provided.

1.1.Project Summary

The complex data set used in the thesis study has been arranged. Classification and optimization analyze were performed in the breast cancer dataset. The optimization algorithms that used are get inspiration from nature. The results obtained were noted for every optimization run.

1.1.1.Purpose, Scope and Objectives

The purpose of the project is to analyze and optimize the data of breast cancer data set. Analyzes were made with series of program on the dataset.

The objectives of the project are to analyze the test results of breast cancer and to give healthcare professionals and doctors a more understandable and regular result.

The scope of the project is to analyze the tests performed on breast cancer patients targeted by the project. As a result of this study, tests on cancer were separated as malignant(dangerous) and benign(harmless). The results of the studies will be displayed in graphs.

1.2.Reasons for Starting the Thesis

One of the reasons for starting this study is the breast cancer is the most common cancer among life. The diagnosis of breast cancer takes long hours manually. ML techniques contribute a lot in the development of such system.

This project has been developed to diagnose breast cancer patients based on the analysis results obtained.

1.3.Contributions of the Thesis

This thesis study will contribute to the health sector. By analyzing the test data of breast cancer patients, it will be easier to diagnose the disease. Also, the tests that are applied to the patients are analyzed and the tests that are not required will not be applied to the patients.

1.4.Order of Thesis

In the first part of the thesis, general information about the purpose of the thesis study, the reasons for its initiation, the contributions of the thesis, similar studies previously conducted and the technologies, languages, tools used in the thesis construction phase are given. In the second part, data sets are preprocessed. The pieces of code used to make the analyzes are described. In the third part, optimization process result gained. In the last part, the result of the thesis study was announced

2.RELATED WORK

One of the reasons for starting this study is the regulation and analysis of test data applied to breast cancer patients. According to the analysis results obtained through this study, it will be easier to diagnose breast cancer patients. As a result of the researchers, various studies in this field stand out. One of them is "Applied Machine Learning to Diagnose Breast Cancer" [2]. Breast cancer is the most common cancer among women and huge reason for increasing death rate in women. The diagnosis of breast cancer takes long hours manually and there is less availability of systems, there is a huge demand to the automatic diagnosis system for early detection of breast cancer. Machine learning and Deep learning techniques contribute a lot in the development of such system. For the classification of benign and malignant tumor we have used classification techniques of machine learning in which the machine is learned from the past data and can predict the category of new input. With respect to the results of accuracy, precision, recall, specificity and False Positive Rate the efficiency of each algorithm is calculated and compared. All algorithms are written in python. The other study is "Evaluation of Machine Learning Classifiers in Breast Cancer Diagnosis"[3]. Breast Cancer is one among the deadliest diseases that threaten women. It affects women in general, but men are also not exceptions to it. Most breast cancers end up fatal except for a few cases. Early diagnosis of the disease helps in successful treatment and cure. Machine learning techniques in the field of medical imaging are increasingly being used in the accurate diagnosis of breast cancer. Machine learning classifiers such as Support Vector Machine and Neural Network are examined in this paper. Breast mammogram images, both normal and pathological images were used in this experiment. The machine learning classifiers were employed to identify the given image as either Benign ore Malignant. Performance of both the classifiers was recorded.

3.MATERIALS AND METHODS

3.1.What's Machine Learning ?

Machine learning (ML) is the study of computer algorithms that improve automatically through experience. It is seen as a subset of artificial intelligence. Machine learning algorithms build a mathematical model based on sample data, known as "training data", in order to make predictions or decisions without being explicitly programmed to do so. Machine learning algorithms are used in a wide variety of applications, such as email filtering and computer vision, where it is difficult or infeasible to develop conventional algorithms to perform the needed tasks. Machine learning is closely related to computational statistics, which focuses on making predictions using computers. The study of mathematical optimization delivers methods, theory and application domains to the field of machine learning. Data mining is a related field of study, focusing on exploratory data analysis through unsupervised learning. In its application across business problems, machine learning is also referred to as predictive analytics [4].

3.1.1.Usage Areas of Machine Learning

3.1.1.1.Machine Learning Speech Recognition

Speech recognition is something of a frontier these days. In a way, the technology is similar to computer vision; it just took more time to figure out how to analyze sound productively. With the emergence of conversational interfaces and mass adoption of virtual assistants - speech recognition turned into a viable business opportunity [5].

3.1.1.2.Computer Vision

Computer Vision is one of the most exciting fields of machine learning use. If text is a more or less raw state of data - images require a different approach. Computer vision algorithm describes image content via matching the features of the images with the features of available samples. The image is broken down to key credentials that are used as reference points. The process looks like this: a photo of a bicycle is recognized as such because the credentials of the sample photo on which the algorithm is trained and the credentials of the input photo correlate. Computer Vision and image recognition, in particular, is widely used throughout different industries [5].

3.1.1.3.Sentiment Analysis

Sentiment Analysis is the next step in the evolution of data analytics platforms. It deals more directly with the way customers interact with the product and express opinions about it.

Sentiment analysis can be used to explore the variety of reactions from the interactions with different kinds of platforms. To do that, the system uses unsupervised machine learning on top of basic recognition procedure [5].

3.1.1.4.Natural Language Processing

Natural Language Processing machine learning algorithms get into the nitty-gritty of the words and extract the stuff of value out of it. And since the text is a raw state of data - it is applied in one form or another practically everywhere.

- NLP applies a broad scope of machine learning algorithms to enable its operation.
- Clustering algorithms are used to explore texts
- Classification algorithms are used to analyze its features

Classification and clustering involve parsing, segmentation, and tagging to construct a model upon which further proceedings are handled.

Regression algorithms are used to determine the proper output sequence upon text generation [5].

3.1.1.5.Service Personalization

Every user loves when the service delivers what the user wants and then some. That's a foundational element of user engagement and a step towards building a strong relationship between the product and its user.

Things can get even better when the said service is tailor-made for the needs and the preferences of the end-users. That's personalization in action, and there is a lot of machine learning involved.

Personalization makes the most of available user data, calculate the possibilities, and turn them into a valuable asset of the business operation [5].

3.1.1.6.Machine learning for Predictive Analytics

When it comes to gaining a competitive advantage with machine learning - Data Analytics is one side of the coin. The other side of the coin is predictive analytics. That's where machine learning comes in full swing.

You see - it is one thing to get the data from different sources in one place, to extract insights and show the thick of it. It is process automation with some fancy tricks. It is an entirely different thing to look at what the future holds and plan your moves accordingly [5].

3.1.1.7.Machine Learning in Data Analytics

Understanding the big picture is a requirement for any company that wants to succeed in a chosen field. Data analytics is one of the preeminent tools that makes it possible.
In essence, data analytics is a three-fold process. It involves:

- gathering data from different sources,
- extracting the valuable insights out of it
- presenting it in a comprehensive manner (i.e., visualizing).

Machine learning algorithms are applied at various stages to secure the efficiency and the accuracy of the process.

- The clustering algorithms are used to explore the data;
- The classification algorithms are used to group data, sift through it and get the gist of it;
- Dimensionality reduction algorithms are used to visualize data, i.e., show it in a coherent form.

Essentially, these data analytics algorithms construct a robust framework for quality decision making. As such, data analytics is used practically in every business aspect of business operation. [5].

3.2.Hyperparameter Optimization

In Machine Learning , hyperparameter optimization or tuning is the problem of choosing a set of optimal hyperparameters for a learning algorithm. A hyperparameter is a parameter whose value is used to control the learning process. By contrast , the values of other parameters are learned.

The same kind of Machine Learning model can require different constraints, weights or learning rates of generalize different data patterns. These measures are called hyperparameters and have to tuned so that the model cam optimally solve the Machine Learning problem. Hyperparameter optimization finds a tuple pf hyperparameters that yields an optimal model which minimizes a predefined loss function on given independent data. The objective function takes a tuple of hyperparameters and returns the associated loss. Cross validation is often used the estimate this generalization performance.

3.2.1.The Bat Optimization Algorithm

Bats are eye-catching animals, and their higher potential of echolocation has engrossed interest of scholars from various arenas. Echolocation mechanism is a kind of sonar: bats, mainly micro-bats, create a loud and short pulse of sound and figure out the distance of an object by using the echo reruns back to their ears. This remarkable positioning method makes bats being able to decide the difference between an obstacle and a prey, allowing them to hunt even in whole darkness.[I]

1: Objective function $f(x)$, $x = (x_1, ..., x_d)^T$
2: Initialize the bat population x_i and v_i for $i = 1 \ldots n$
3: Define pulse frequency $Q_i \in [Q_{min}, Q_{max}]$
4: Initialize pulse rates r_i and the loudness A_i
5: while $(t < T_{max})$ // number of iterations
6: Generate new solutions by adjusting frequency and
7: update velocities and locations/solutions [Eq.(2) to (4)]
8: if$(rand(0,1) > r_i)$
9: Select a solution among the best solutions
10: Generate a local solution around the best solution
11: end if
12: Generate a new solution by flying randomly
13: if$(rand(0,1) < A_i \ and \ f(x_i) < f(x))$
14: Accept the new solutions
15: Increase r_i and reduce A_i
16: end if
17: Rank the bats and find the current best
18: end
19: Postprocess results and visualization

Figure 1:BA Pseudocode

The BA is illustrated in Figure. In this algorithm, the bat behavior is captured into the fitness function of the problem to be solved. It consists of the following components:

- initialization (lines 2-4),
- generation of new solutions (lines 6-7),
- local search (lines 8-11),
- generation of a new solution by flying randomly (lines 12-16) and
- find the current best solution.

3.2.2.The Hybrid Bat Optimization Algorithm

Differential evolution (DE) is a technique for the optimization introduced by Storn and Price in 1995. DE optimizes a problem by maintaining a population of candidate solutions and creates new candidate solutions by combining the existing ones according to its simple formulae, and then keeping whichever candidate solution has the best score or fitness on the optimization problem at hand.[K]

1: Objective function $f(x)$, $x = (x_1, ..., x_d)^T$
2: Initialize the bat population x_i and v_i for $i = 1 \ldots n$
3: Define pulse frequency $Q_i \in [Q_{min}, Q_{max}]$
4: Initialize pulse rates r_i and the loudness A_i
5: while $(t < T_{max})$ // number of iterations
6: Generate new solutions by adjusting frequency and
7: update velocities and locations/solutions [Eq.(2) to (4)]
8: if$(rand(0,1) > r_i)$
9: Select a solution among the best solutions
10: Generate a local solution around the best solution
11: end if
12: Generate a new solution by flying randomly
13: if$(rand(0,1) < A_i \ and \ f(x_i) < f(x))$
14: Accept the new solutions
15: Increase r_i and reduce A_i
16: end if
17: Rank the bats and find the current best
18: end
19: Postprocess results and visualization

Figure 2:HBA Pseudocode

We propose a new BA, called Hybrid Bat Algorithm (HBA). It was obtained by hybridizing the original BA using the DE strategies[K]. The HBA pseudo-code is illustrated in Figure

3.2.3.The Firefly Optimization Algorithm

Firefly algorithm is a bio-inspired metaheuristic algorithm for optimization problems. It was introduced in 2009 at Cambridge University by Yang. The algorithm is inspired by the flashing behavior of fireflies at night. One of the three rules used to construct the algorithm is that all fireflies are unisex, which means any firefly can be attracted to any other brighter one. The second rule is that the brightness of a firefly is determined from the encoded objective function. The last rule is that attractiveness is directly proportional to brightness but decreases with distance, and a firefly will move towards the brighter one, and if there is no brighter one, it will move randomly. [L]

```
1: Begin
2:   Initialisation max iteration, α, βo , γ
3:   Generate initial population
4:   Define the Objective function f(x),
5:   Determine Intensity (I) at cost (x) of each individual determined by f(xi )
6:     While (t < Iter max)
7:           For i=1 to n
8:                 For j=1 to n
9:                       if (I j > Ii )
10:                            Move firefly i towards j in K dimension
11:                      end if
12:                   Evaluate new solutions and update light intensity
13:               end for j
14:               end for i
15:             Rank the fireflies and find the current best
16:          end while
17:       Post process results and visualization
18:End procedure
```

Figure 3:FA Pseudocode

The HBA pseudo-code is illustrated in Figure

3.2.4.The Grey Wolf Optimization Algorithm

The grey wolf optimizer (GWO) is a novel type of swarm intelligence optimization algorithm. An improved grey wolf optimizer (IGWO) with evolution and elimination mechanism was proposed so as to achieve the proper compromise between exploration and exploitation, further accelerate the convergence and increase the optimization accuracy of GWO. The biological evolution and the "survival of the fittest" (SOF) principle of biological updating of nature are added to the basic wolf algorithm[M]

```
1: Begin
2:     Initialize the parameters popsize, maxiter, ub and lb where
3:     popsize: size of population,
4:     maxiter: maximum number of iterations,
5:     ub: upper bound(s) of the variables,
6:     lb: lower bound(s) of the variables;
7:     Generate the initial positions of grey wolves with ub and lb;
8:     Initialize a,A, and C;
9:     Calculate the fitness of each grey wolf;
10:    alpha = the grey wolf with the first maximum fitness;
11:    beta= the grey wolf with the second maximum fitness;
12:    delta= the grey wolf with the third maximum fitness;
13:    While k < maxiter
14:        for i = 1: popsize
15:            Update the position of the current grey wolf
16:        end for
17:    Update a , A and C;
18:    Calculate the fitness of all grey wolves;
19:    Update alpha, beta, and delta;
20: End
```

Figure 4: GWO Pseudocode

The GWO pseudo-code is showed in Figure 4.

3.3.Python

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built-in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms and can be freely distributed [6].

3.4.The Juypter Notebook

The notebook extends the console-based approach to interactive computing in a qualitatively new direction, providing a web-based application suitable for capturing the whole computation process: developing, documenting, and executing code, as well as communicating the results. The Jupyter notebook combines two components:
A web application: A browser-based tool for interactive authoring of documents which combine explanatory text, mathematics, computations and their rich media output.
Notebook documents: A representation of all content visible in the web application, including inputs and outputs of the computations, explanatory text, mathematics, images, and rich media representations of objects. [7].

3.5.Scikit-Learn

Scikit-learn is largely written in Python and uses NumPy extensively for high-performance linear algebra and array operations. Furthermore, some core algorithms are written in Cython to improve performance. Support vector machines are implemented by a Cython wrapper around LIBSVM; logistic regression and linear support vector machines by a

similar wrapper around LIBLINEAR. In such cases, extending these methods with Python may not be possible.

Scikit-learn integrates well with many other Python libraries, such as Matplotlib and plotly for plotting, NumPy for array vectorization,Pandasdataframes, SciPy, and many more. [A]

3.5.1.Nature Inspired Algorithms for Scikit-Learn

Nature inspired algorithms for hyper-parameter tuning of scikit-learn models. This package uses algorithm implementation from NiaPy. The usage is similar to using sklearn's GridSearchCV.[B]

3.5.1.1.NiaPy

Nature inspired algorithms are a very popular tool for solving optimization problems. Numerous variants of nature-inspired algorithms have been developed since the beginning of their era. To prove their versatility, those were tested in various domains on various applications, especially when they are hybridized, modified or adapted. However, implementation of nature-inspired algorithms is sometimes a difficult, complex and tedious task. In order to break this wall, NiaPy is intended for simple and quick use, without spending time for implementing algorithms from scratch.[C]

3.6.Pandas

Pandas is an open-source Python package that is most widely used for data science/data analysis and machine learning tasks. It is built on top of another package named NumPy, which provides support for multi-dimensional arrays. Pandas makes it simple to do many of the time consuming, repetitive tasks associated with working with data, including:

- Data cleansing
- Data fill

- Data normalization
- Merges and joins
- Data visualization
- Statistical analysis
- Data inspection
- Loading and saving data
- And much more

In fact, with Pandas, you can do everything that makes world-leading data scientists vote Pandas as the best data analysis and manipulation tool available.[D]

3.7.Numpy

NumPy is the fundamental package for scientific computing in Python. It is a Python library that provides a multidimensional array object, various derived objects (such as masked arrays and matrices), and an assortment of routines for fast operations on arrays, including mathematical, logical, shape manipulation, sorting, selecting, I/O, discrete Fourier transforms, basic linear algebra, basic statistical operations, random simulation and much more.[E]

3.8.SciPy

SciPy contains modules for optimization, linear algebra, integration, interpolation, special functions, FFT, signal and image processing, ODE solvers and other tasks common in science and engineering. SciPy is also a family of conferences for users and developers of these tools: SciPy (in the United States), EuroSciPy (in Europe) and SciPy.in (in India). Enthought originated the SciPy conference in the United States and continues to sponsor many of the international conferences as well as host the SciPy website.[F]

3.8.Seaborn

Seaborn is a Python library created for enhanced data visualization. It's a very timely and relevant tool for data professionals working today precisely because effective data visualization and communication in general is a particularly essential skill. Being able to bridge the gap between data and insight is hugely valuable, and Seaborn is a tool that fits comfortably in the toolchain of anyone interested in doing just that.[G]

3.9.Matplolib

Matplotlib is a cross-platform, data visualization and graphical plotting library for Python and its numerical extension NumPy. As such, it offers a viable open-source alternative to MATLAB. Developers can also use matplotlib's APIs (Application Programming Interfaces) to embed plots in GUI applications.[H]

4.DATASET USED IN THE PROJECT

4.1.General Information About Data

There are data sets of breast cancer belonging to two classes. One of them is cancer with " malignant "(dangerous) and the other is patients with " benign ". (harmless).

```
Index(['id', 'diagnosis', 'radius_mean', 'texture_mean', 'perimeter_mean',
       'area_mean', 'smoothness_mean', 'compactness_mean', 'concavity_mean',
       'concave points_mean', 'symmetry_mean', 'fractal_dimension_mean',
       'radius_se', 'texture_se', 'perimeter_se', 'area_se', 'smoothness_se',
       'compactness_se', 'concavity_se', 'concave points_se', 'symmetry_se',
       'fractal_dimension_se', 'radius_worst', 'texture_worst',
       'perimeter_worst', 'area_worst', 'smoothness_worst',
       'compactness_worst', 'concavity_worst', 'concave points_worst',
       'symmetry_worst', 'fractal_dimension_worst'],
      dtype='object')
```

Figure 5:Attributes_of_Dataset

Data set has 32 different attributes. In this data set has special attributes about breast cancer.

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave points_mean	symm
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	

Figure 6: Dataset Overview

In the data set some attributes are not necessary about being part of Machine Learning. “id” and “Unnamed: 32” columns are nonessential. Most important attribute is diagnosis. Because it needed for classification operation. Dataset has lot of data in it, checking if there are any missing values.

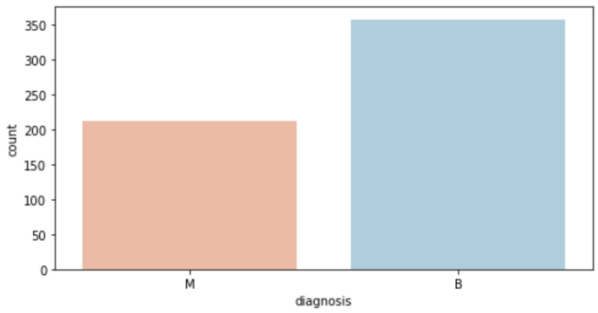
```
id                         0
diagnosis                  0
radius_mean                0
texture_mean               0
perimeter_mean             0
area_mean                  0
smoothness_mean            0
compactness_mean           0
concavity_mean             0
concave points_mean        0
symmetry_mean              0
fractal_dimension_mean     0
radius_se                  0
texture_se                 0
perimeter_se               0
area_se                    0
smoothness_se              0
compactness_se             0
concavity_se               0
concave points_se          0
symmetry_se                0
fractal_dimension_se       0
radius_worst               0
texture_worst              0
perimeter_worst            0
area_worst                 0
smoothness_worst           0
compactness_worst          0
concavity_worst            0
concave points_worst       0
symmetry_worst             0
fractal_dimension_worst    0
dtype: int64
```

Figure 7:Missing Values

In figure 3 we can see that there none missing value Missing value removing step skipped.

4.2.Visaulisation of Data

Our response variable, diagnosis, is categorical and has two classes, 'B' (Benign) and 'M' (Malignant). Checking out the distribution of its classes.




```
Number of cells labeled Benign:  357
Number of cells labeled Malignant :  212

% of cells labeled Benign 62.74 %
% of cells labeled Malignant 37.26 %
```

Figure 8: Diagnosis Distribution

Out of the 569 observations, 357 (or %62.7) have been labeled malignant, while the rest 212 (or %37.3) have been labeled benign. Later when we develop a predictive model and test is on unseen data, we should expect see a similar proportion of labels.

With sense, we could attempt to find some quick insights by analyzing the data in only their perspectives. For instance, we could choose to check out the relationship between the 10 attributes and the “diagnosis” variable by only choosing the “mean” columns.

In figure we can see that radius , texture mean, and diagnosis correlated graphs.

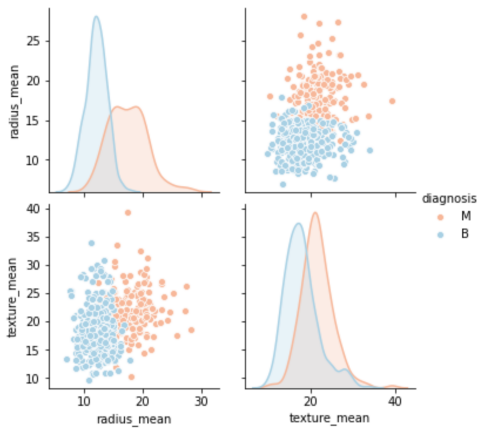


Figure 9:Radius,Texture and Diagnosis Correlated Graph

5.Classification and Optimization

5.1.Random Forest Classification

Random forests are a supervised learning algorithm. It can be used both for classification and regression. It is also the most flexible and easy to use algorithm. A forest is comprised of trees. It is said that the more trees it has, the more robust a forest is. Random forests create decision trees on randomly selected data samples, gets prediction from each tree and selects the best solution by means of voting. It also provides a pretty good indicator of the feature importance.

Random forests have a variety of applications, such as recommendation engines, image classification and feature selection. It can be used to classify loyal loan applicants, identify fraudulent activity and predict diseases. It lies at the base of the Boruta algorithm, which selects important features in a dataset [8].

5.1.2.Creating Model

With help of Scikit-Learn library we can create our model with one function.

```
clf_rf = RandomForestClassifier()
clr_rf = clf_rf.fit(X_train,y_train)
```

Figure 10: Model Function

First, we create our classifier “clr_rf” and define as a Random Forest Classifier. Then we can train our model with train data.

5.1.3.Prediction Result of Random Forest Classifier

First of all, we use %75 percent of dataset to train model other %25 percent used for testing the model’s prediction. When we fit the test selection into model making prediction. For this case our accuracy is %92.3. Let’s have a look at confusion matrix in the figure.

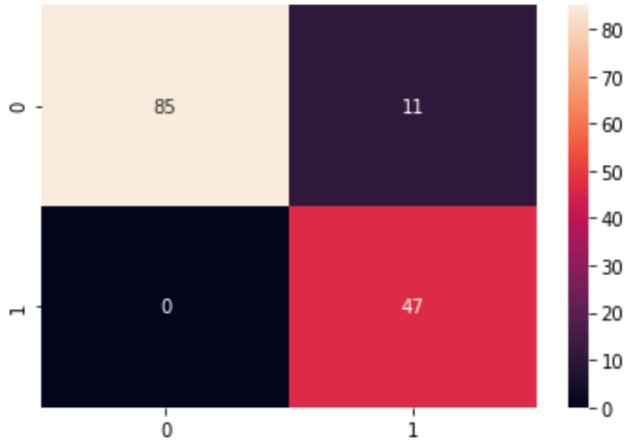


Figure 11:Random Forest Confusion Matrix

Accuracy is almost 95% and as it can be seen in confusion matrix, we make some wrong prediction.

5.2.Hyperparameter Optimization with Nature Inspired Algorithms

All the method that used in this experiment has same attributes. In figure we can see the attributes and their values.

```
param_grid = {
    'n_estimators': range(20, 400, 20),
    'max_depth': range(5, 300, 20),
    'min_samples_split': range(2, 50, 5),
    'max_features': ["auto", "sqrt", "log2"],
}
```

Figure 12:Hyperparameters

All optimizing method trying to find best parameters in the shortest time. All the attributes are used for random forest tree.

Also their Cross Validation ,population size , max gen number , maximum stangnating gene number and run number are same.

- Population size : 50
- Population Max Gene Number : 100
- Maximum Stagnating Gene Number : 8
- Run Number : 6
- Cross Validation: 3

5.2.1.Hyperparameter Optimization With Bat Algorithm

Echolocation works as a type of sonar: bats, mainly microbats, emit a loud and short sound pulse. When they hit an object, after a fraction of time, the echo will return back to their ears. The bat receives and detects the location of the prey in this way.

```
Fitting 3 folds for some of the 8550 candidates, which might total in 25650 fits
CPU times: user 9.51 s, sys: 457 ms, total: 9.97 s
Wall time: 9min 2s
```

Figure 13:BA Process Time

Fitment process took 9 min 2 s. Let's see the best parameters with help of "nia_search.best_params_" function.

```
{'n_estimators': 40,
 'max_depth': 265,
 'min_samples_split': 2,
 'max_features': 'log2'}
```

Figure 14:Best Parameter of BA

Bat algorithms did find the best parameters for this classification.

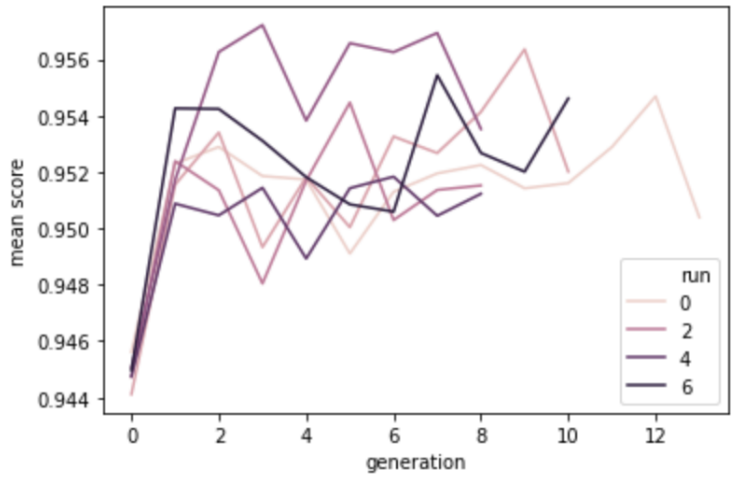


Figure 15:BA Accuracy Graph

We can see on the figure () that 4th run has the best accuracy score. Choosing the most accuracy run is important.

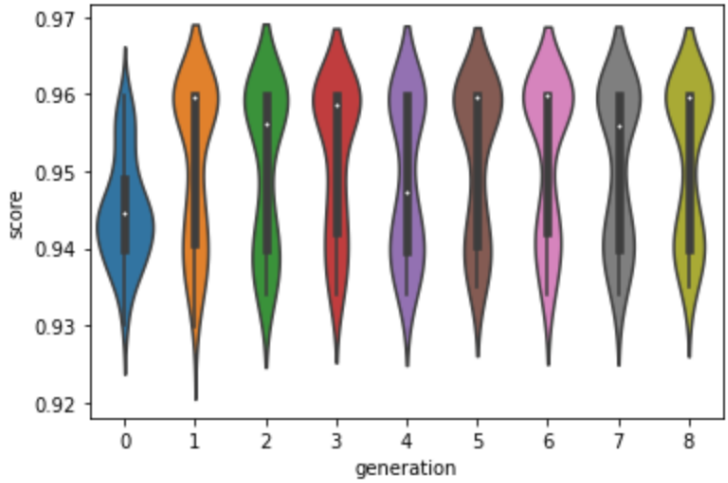


Figure 16:BA Best Run

We can see all the generation population score distrubition. We can see first generation is scored between 0.94 and 0.95 this is random forest search. At the last generation our accucary is goes up to 0.97.Then fit it into model and see the prediction number.

	precision	recall	f1-score	support
B	0.9785	1.0000	0.9891	91
M	1.0000	0.9615	0.9804	52
accuracy			0.9860	143
macro avg	0.9892	0.9808	0.9848	143
weighted avg	0.9863	0.9860	0.9860	143

Figure 17:BA Optimized Results

BA optimization got better precision. Without optimization with Random Forest classification accuracy was %95 . On the other hand Random Forest with Hyperparameter Optimization use of BA accuracy is %98.

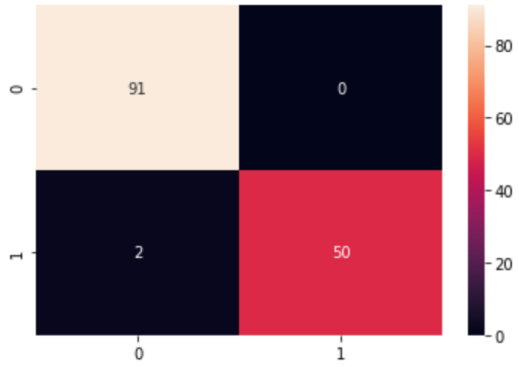


Figure 18:BA Confusion Matrix

Accuracy went up to %98.60 with help of BA optimization.

5.2.2.Hyperparameter Optimization With Hybrid Bat Algorithm

The Hybrid Bat algorithm is hybridized with differential evolution strategies. Besides showing very promising results of the standard benchmark functions, this hybridization also significantly improves the original bat algorithm.

```
Fitting 3 folds for some of the 8550 candidates, which might total in 25650 fits
CPU times: user 17 s, sys: 726 ms, total: 17.7 s
Wall time: 9min 1s
```

Figure 19:HBA Fitting Time

Fitment process took 9 minute 1 second. The best parameters will find with “nia_search.best_params_” function.

```
{'n_estimators': 360,
 'max_depth': 245,
 'min_samples_split': 2,
 'max_features': 'auto'}
```

Figure 20:HBA Best Parameters

Hybrid Bat Algorithm did find the best parameters for this classification.

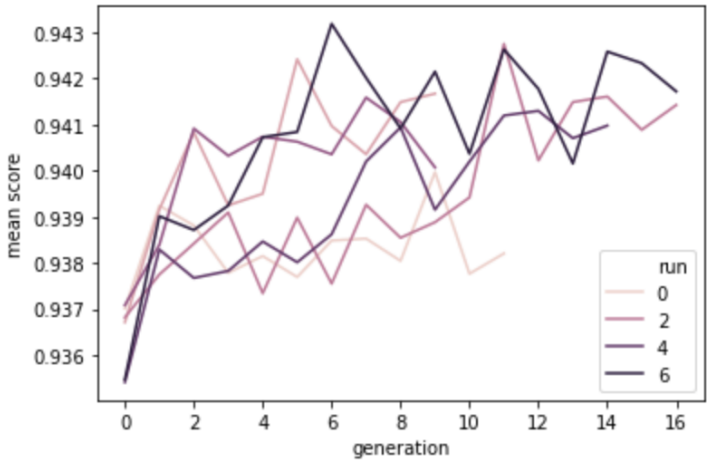


Figure 21:HBA Accuracy Graph

We can see on the figure () that 6th run has the highest accuracy score. Choosing the most succesfull for best accuracy number and genes.

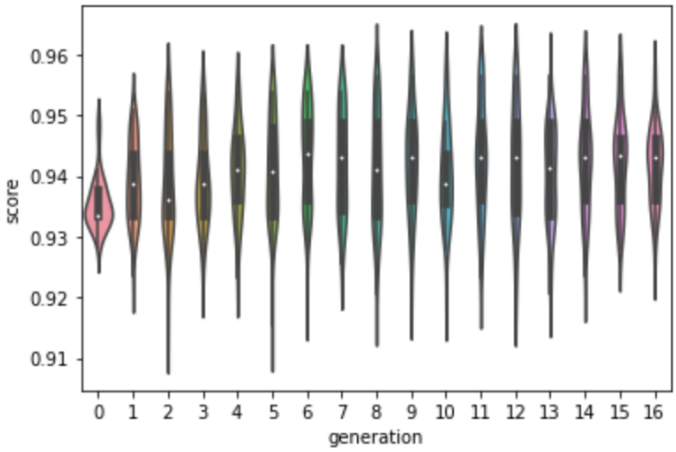


Figure 22: HBA Best Run

We can see all the generation population score distribution. We can see first generation is scored between 0.93 and 0.94 this is random forest search. At he last generation our accuracy is goes up to 0.95. After that best parameters will fit it into model and see the prediction number.

	precision	recall	f1-score	support
B	0.9438	0.9882	0.9655	85
M	0.9815	0.9138	0.9464	58
accuracy			0.9580	143
macro avg	0.9627	0.9510	0.9560	143
weighted avg	0.9591	0.9580	0.9578	143

Figure 23:HBA Optimized Result

HBA optimization got better precision. Without optimization with Random Forest classification accuracy was %95.0 . On the other hand Random Forest with Hyperparameter Optimization use of BA accuracy is %95.8

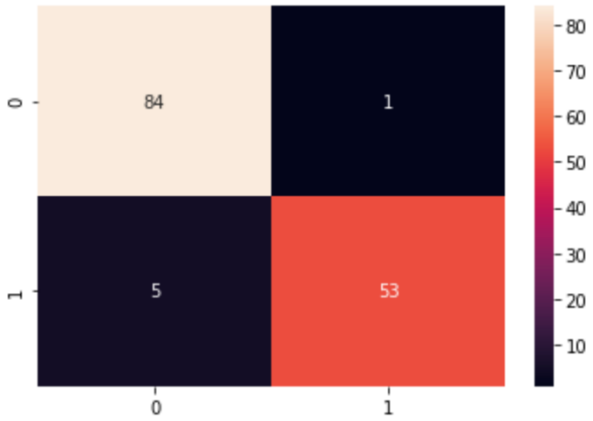


Figure 24:HBA Confusion Matrix

Accuracy went up to %95.80 with help of HBA optimization.

5.2.3.Hyperparameter Optimization With FA

Firefly algorithm is one of the new metaheuristic algorithms for optimization problems. The algorithm is inspired by the flashing behavior of fireflies. In the algorithm, randomly generated solutions will be considered as fireflies, and brightness is assigned depending on their performance on the objective function. One of the rules used to construct the algorithm is, a firefly will be attracted to a brighter firefly, and if there is no brighter firefly, it will move randomly.

```
Fitting 3 folds for some of the 8550 candidates, which might total in 25650 fits
CPU times: user 59.4 s, sys: 1.58 s, total: 1min
Wall time: 18min 1s
```

Figure 25:FA Process Time

Fitment process took 18 minute 1 second. Let's see the best parameters with help of "nia_search.best_params_" function.

```
{'n_estimators': 20,
 'max_depth': 145,
 'min_samples_split': 2,
 'max_features': 'sqrt'}
```

Figure 26:FA Best Parameters

Firefly Algorithm did find the best parameters for this classification.

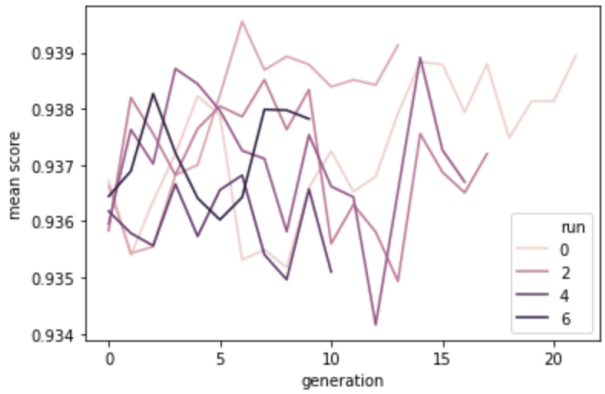


Figure 27:FA Accuracy Graph

In figure we can see 2nd run have the highest accuracy score. Let's have a closer look on 2nd run's generations.

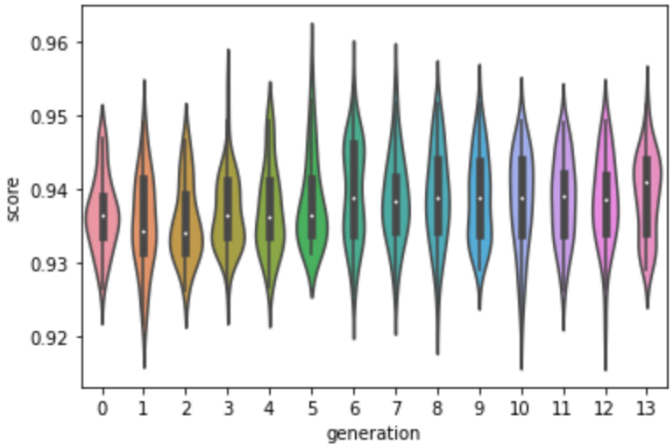


Figure 28:FA Best Run

In Figure28. 13th generation has best density for accuracy process. Next fitting the hyperparameters into model.

	precision	recall	f1-score	support
B	0.9255	1.0000	0.9613	87
M	1.0000	0.8750	0.9333	56
accuracy			0.9510	143
macro avg	0.9628	0.9375	0.9473	143
weighted avg	0.9547	0.9510	0.9504	143

Figure 29:FA Optimized Result

After fitting operation model has %95.10 accuracy.

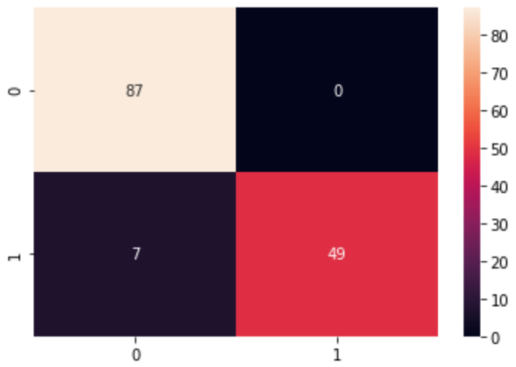


Figure 30:FA Confusion Matrix

Even after optimization there is still some errors in model.

5.2.4.Hyperparameter Optimization With Grey Wolf Optimization

Grey Wolf Optimization algorithm(GWO) is new meta heuristic optimization technology. Its principle is the imitate the behavior of grey wolves in nature to hunt in a cooperative way.

```
Fitting 3 folds for some of the 8550 candidates, which might total in 25650 fits
CPU times: user 12.4 s, sys: 589 ms, total: 13 s
Wall time: 6min 16s
```

Figure 31:GWO Process Time

Fitment process took 6 minute 16 second. The best parameters will find with "nia_search.best_params_" function.

```
{'n_estimators': 40,
 'max_depth': 125,
 'min_samples_split': 2,
 'max_features': 'sqrt'}
```

Figure 32:GWO Best Parameters

The parameters for this optimization run

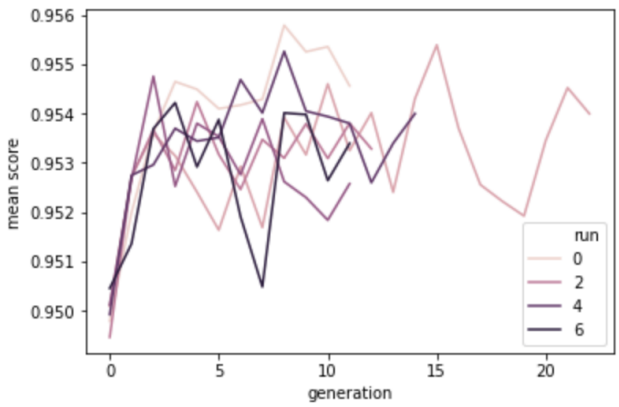


Figure 33:GWO Accuracy Graph

In figure 33 5th run has the best accuracy score. Choosing the best accuracy score is important.

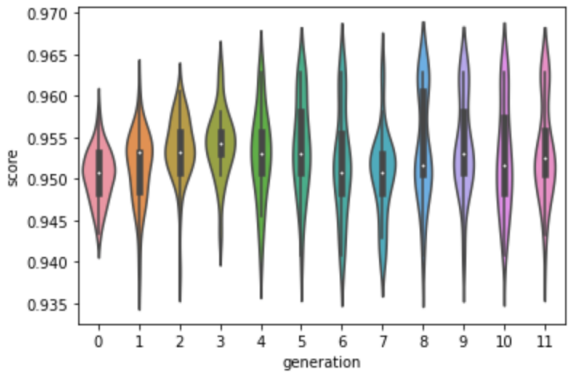


Figure 34:GWO Best Run

In figure 34 , 5th run has 11 generation in it. Generations 1 to 4 is rapidly increasing accuracy score . Best parameter will fit in model for getting best accuracy score.

	precision	recall	f1-score	support
B	0.9684	0.9293	0.9485	99
M	0.8542	0.9318	0.8913	44
accuracy			0.9301	143
macro avg	0.9113	0.9306	0.9199	143
weighted avg	0.9333	0.9301	0.9309	143

Figure 35:GWO Optimized Result

Overall accuracy of model is %93.01 which is better than non optimized model

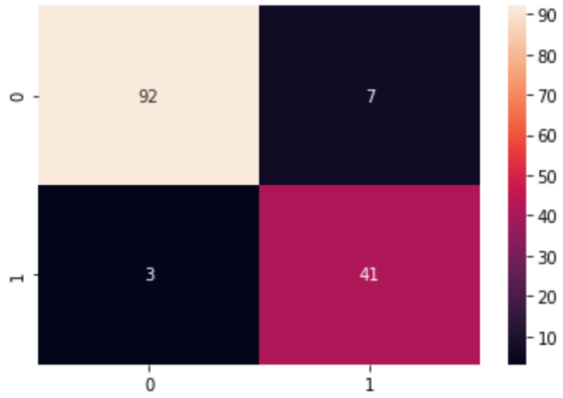


Figure 36:GWO Accuracy Matrix

https://www.researchgate.net/publication/347935440_Applied_Machine_Learning_to_Diagnose_Breast_Cancer

3- https://www.researchgate.net/publication/351090971_Evaluation_Of_Machine_Learning_Classifiers_In_Breast_Cancer_Diagnosis

7- https://jupyter-notebook.readthedocs.io/en/stable/notebook.html

8- random forest

9-

[A]= https://en.wikipedia.org/wiki/Scikit-learn

[B]= https://pypi.org/project/sklearn-nature-inspired-algorithms/#description

[C]= https://github.com/NiaOrg/NiaPy

[D]=https://www.activestate.com/resources/quick-reads/what-is-pandas-in-python-everything-you-need-to-know/

[E]= https://numpy.org/doc/stable/user/whatisnumpy.html

[F]= https://en.wikipedia.org/wiki/SciPy

[G]=https://hub.packtpub.com/what-is-seaborn-and-why-should-you-use-it-for-data-visualization/

[H]=https://www.activestate.com/resources/quick-reads/what-is-matplotlib-in-python-how-to-use-it-for-plotting/

[I]= https://www.researchgate.net/publication/338782603_Bat_algorithm_BA_review_applications_and_modifications

[K]= https://downloads.hindawi.com/journals/tswj/2014/709738.pdf

[L]= https://www.hindawi.com/journals/jam/2012/467631/

[M]= https://www.nature.com/articles/s41598-019-43546-3