Anglia Ruskin University

A Novel approach for improving the classification efficiency using GWO for ECG signals.

Principles of Data Mining and Machine Learning

Module Code: MOD007892

Academic Year: 2023/2024

SID: 2224089

Table of Contents

1. Introduction

Cardiovascular disease is a major cause of death globally and a diagnostic challenge for healthcare systems. Arrhythmias, for instance, are difficult to diagnose through conventional methods, which can be time-consuming. (Zaharia et al., 2023) Integrating machine learning approaches can help to classify arrhythmias in ECG signals. This can save valuable time for healthcare professionals, allowing for timely interventions that could save lives.

Machine learning (ML) has become increasingly popular in healthcare, particularly in analyzing Electrocardiogram (ECG) signals. ECG is a vital diagnostic tool for assessing cardiovascular health. This project aims to classify accurately and provide an approach to improving classification efficiency using Gray Wolf Optimizer for the MIT BIH Arrhythmia Dataset. The importance of this task lies in the potential for early detection and classification of cardiac abnormalities, which can lead to timely intervention and improved patient outcomes.

Numerous machine-learning techniques have been developed to detect arrhythmias based on electrocardiograms (ECG). These techniques incorporate manually crafted features, as highlighted in a study by Mohonta, Motin, and Kumar in 2022. Support Vector Machine (SVM), neural networks, and Logistic regression are some of the commonly employed classification algorithms in this context. Optimizing classification accuracy is crucial for handling sensitive data directly impacting human life. Given the significant number of features in the dataset, it's crucial to identify the most pertinent ones for adequate classification. This research effort aims to enhance the accuracy of classification algorithms such as KNN, Logistic Regression, and Naïve Bayes by leveraging the Gray Wolf Optimizer to optimize feature selection.

2. Exploratory Data Analysis and Data Visualization

To develop this code, I used Python as the programming language along with Pandas, Matplotlib, and Scikit-Learn libraries.

Data Loading and Overview of Dataset

0	1	2	3	4	5	6	7	8	9	...	178	179	180	181	182	183	184	185	186	187
0.977941	0.926471	0.681373	0.245098	0.154412	0.191176	0.151961	0.085784	0.058824	0.049020	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.960114	0.863248	0.461538	0.196581	0.094017	0.125356	0.099715	0.088319	0.074074	0.082621	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1.000000	0.659459	0.186486	0.070270	0.070270	0.059459	0.056757	0.043243	0.054054	0.045946	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.925414	0.665746	0.541436	0.276243	0.196133	0.077348	0.071823	0.060773	0.066298	0.058011	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.967136	1.000000	0.830986	0.586854	0.356808	0.248826	0.145540	0.089202	0.117371	0.150235	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

rows × 188 columns

Figure 2.1 Display First 5 rows of MIT BIH Arrythmia Dataset

As Figure 2.1 Display used Panda's library to import dataset and display through it.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 87554 entries, 0 to 87553
Columns: 188 entries, 0 to 187
dtypes: float64(188)
memory usage: 125.6 MB
```

Figure 2.2 Dataset information

According to the Figure 2.1 and Figure 2.2 the dataset consists of 87554 samples and 188 entries. It indicates all the data types can be seen in the float data type.

```
Index(['9.779411554336547852e-01', '9.264705777168273926e-01',
       '6.813725233078002930e-01', '2.450980395078659058e-01',
       '1.544117629528045654e-01', '1.911764740943908691e-01',
       '1.519607901573181152e-01', '8.578431606292724609e-02',
       '5.882352963089942932e-02', '4.901960864663124084e-02',
       ...
       '0.000000000000000000e+00.79', '0.000000000000000000e+00.80',
       '0.000000000000000000e+00.81', '0.000000000000000000e+00.82',
       '0.000000000000000000e+00.83', '0.000000000000000000e+00.84',
       '0.000000000000000000e+00.85', '0.000000000000000000e+00.86',
       '0.000000000000000000e+00.87', '0.000000000000000000e+00.88'],
      dtype='object', length=188)
```

Figure 2.3 Dataset Column Details (Feature)

Figure 2.3 Display the Column headers of the dataset.

```
In [10]:  print(dataset.describe())

       9.779411554336547852e-01  9.264705777168273926e-01  \
count              87553.000000              87553.000000
mean                   0.890359                  0.758158
std                    0.240910                  0.221814
min                    0.000000                  0.000000
25%                    0.921922                  0.682482
50%                    0.991342                  0.826007
75%                    1.000000                  0.910506
max                    1.000000                  1.000000

       6.813725233078002930e-01  2.450980395078659058e-01  \
count              87553.000000              87553.000000
mean                   0.423969                  0.219104
std                    0.227305                  0.206880
min                    0.000000                  0.000000
25%                    0.250965                  0.048458
50%                    0.429467                  0.165992
75%                    0.578767                  0.341727
max                    1.000000                  1.000000

       1.544117629528045654e-01  1.911764740943908691e-01  \
count              87553.000000              87553.000000
mean                   0.201127                  0.210399
std                    0.177058                  0.171910
min                    0.000000                  0.000000
25%                    0.082329                  0.088415
50%                    0.147870                  0.158798
75%                    0.258993                  0.287634
max                    1.000000                  1.000000

       1.519607901573181152e-01  8.578431606292724609e-02  \
count              87553.000000              87553.000000
mean                   0.205809                  0.201774
std                    0.178482                  0.177241
```

Figure 2.4 Dataset Describe function used to describe the data

```
print(dataset.shape)
```

```
(87553, 188)
```

Figure 2.5 Display the Shape of the Dataset

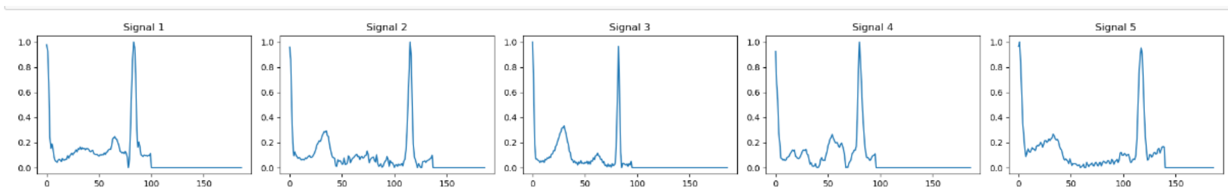


Figure 2.6 Displaying first five ECG samples from the arrythmia dataset.

Display the ECG signal shape of the first 5 samples in the dataset in Figure 2.6.

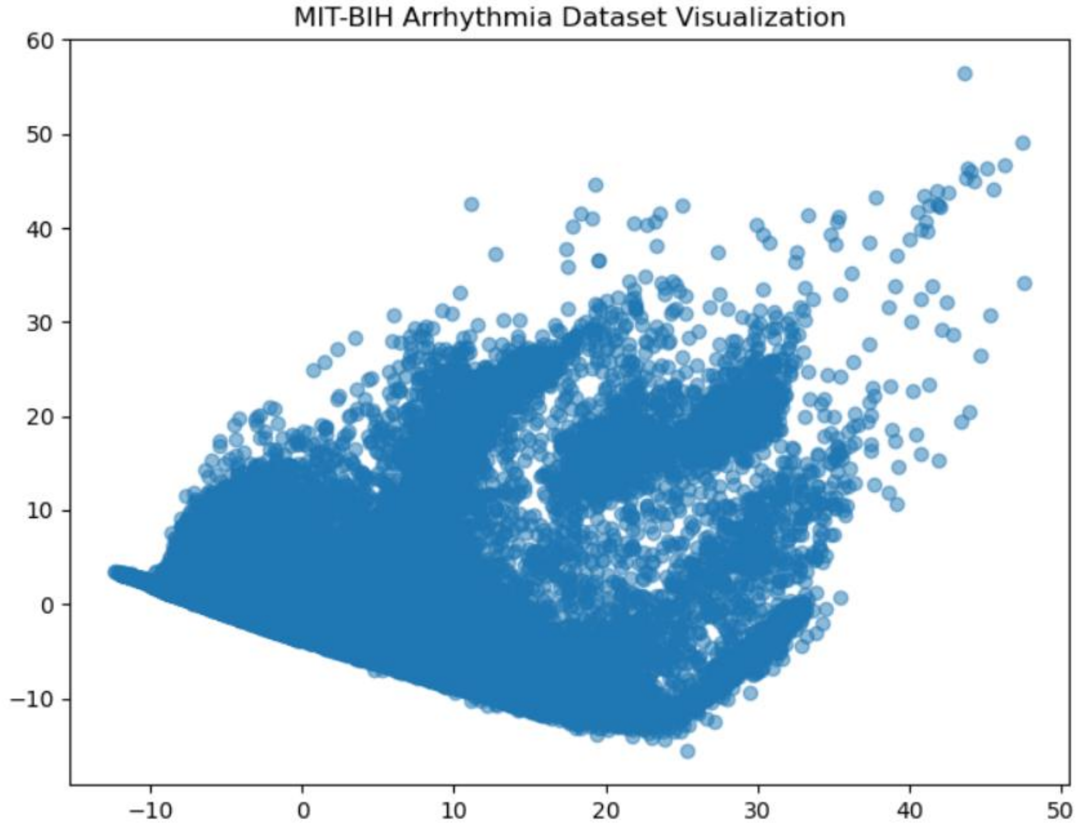


Figure 2.7 Scatter Diagram of MIT BIH Arrythmia Dataset

Figure 2.7 visualizes how the dataset is spread throughout the space.

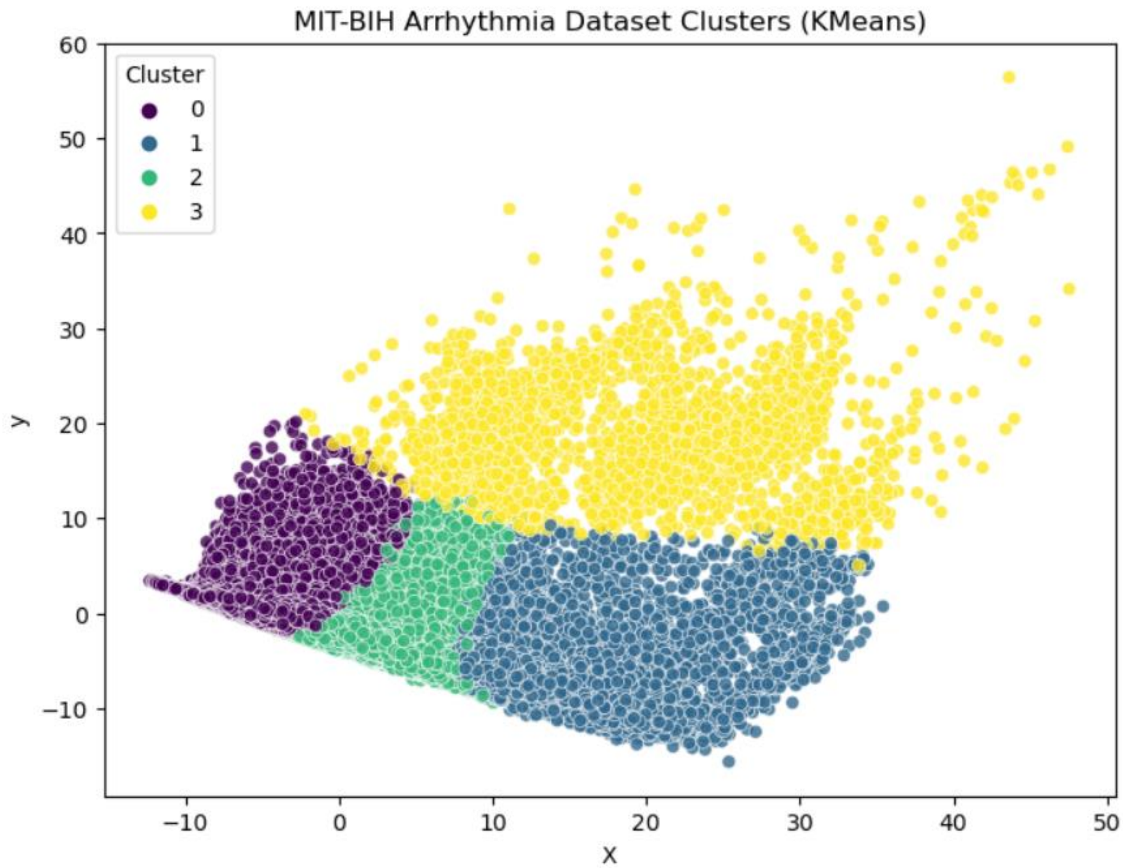


Figure 2.8 Display the clusters in the MIT BIH Arrythmia dataset

In Figure 2.8, K-means clustering was applied to determine the number of clusters present in the dataset. The results suggest that there are only 4 clusters available, but according to the dataset, there are actually 5 clusters present.

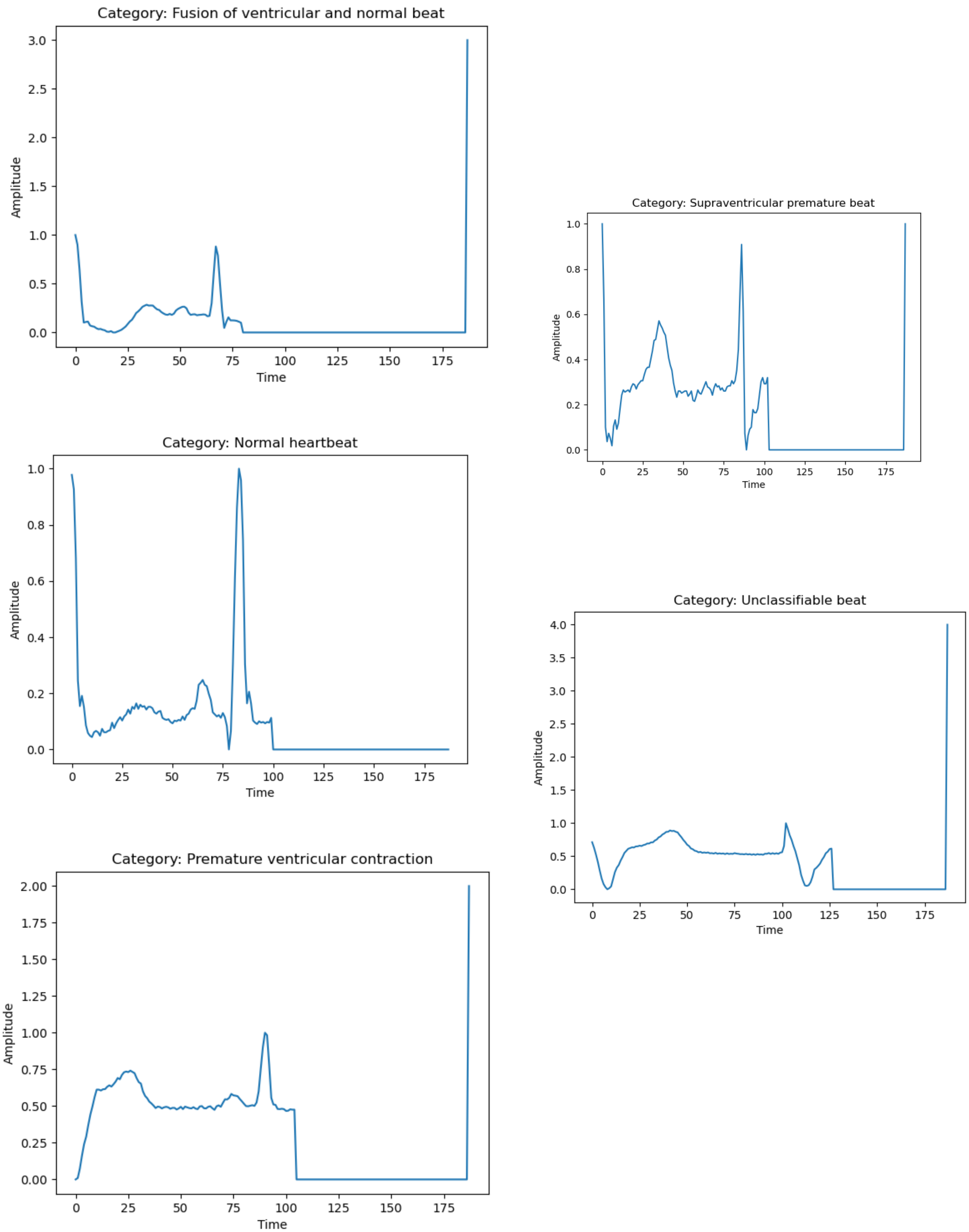


Figure 2.8 Display the five different Signal categoty in the MIT BIH Arrythmia dataset

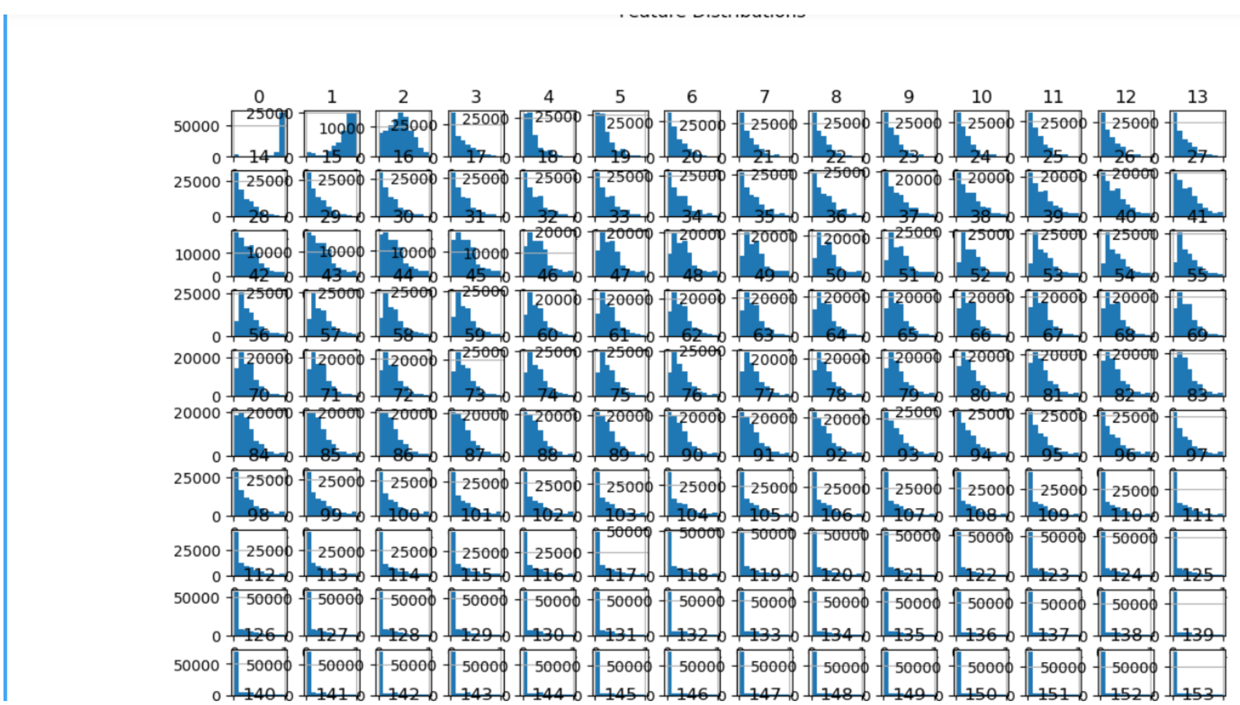


Figure 2.9 Display the Feature distribution of the dataset.

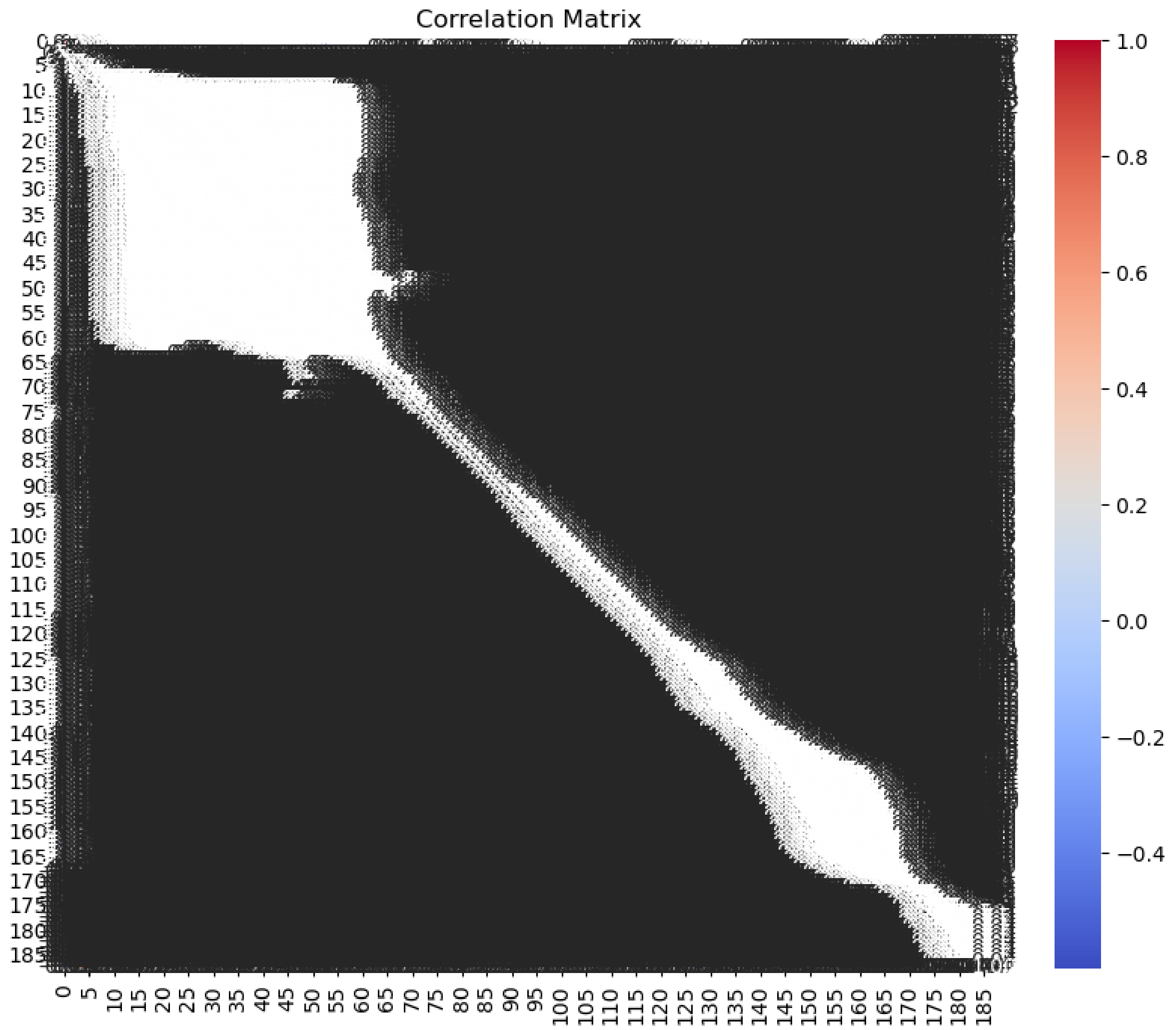


Figure 2.10 Correlation Matrix

```
Category
Normal heartbeat                        72471
Unclassifiable beat                      6431
Premature ventricular contraction        5788
Supraventricular premature beat          2223
Fusion of ventricular and normal beat     641
Name: count, dtype: int64
```

```
Out[18]: <Axes: xlabel='Category'>
```

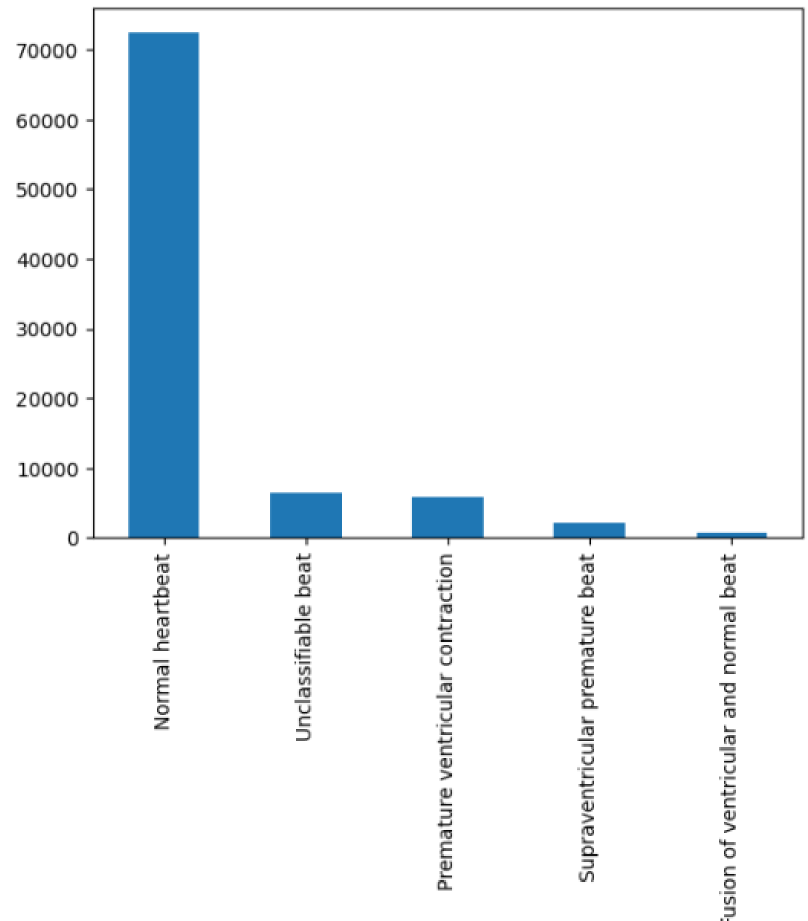


Figure 2.11 The Categories and Number of Data per category (Skewed Dataset)

The figure 2.11 displays the distribution of data across categories. As shown in the figure, there a re 72471 instances of data available in each of the following categories: normal beat (6431), uncl assifiable beat, premature ventricular contraction (5788), supraventricular premature beat (2223), and fusion of ventricular and normal beat (641). However, it is evident that the dataset is highly s kewed, and therefore, needs to be balanced. Figure 2.12 displays the balanced dataset.

```
0.000000000000000000e+00.88
0.0    641
1.0    641
2.0    641
3.0    641
4.0    641
Name: count, dtype: int64
```

```
Out[30]: <Axes: xlabel='0.000000000000000000e+00.88'>
```

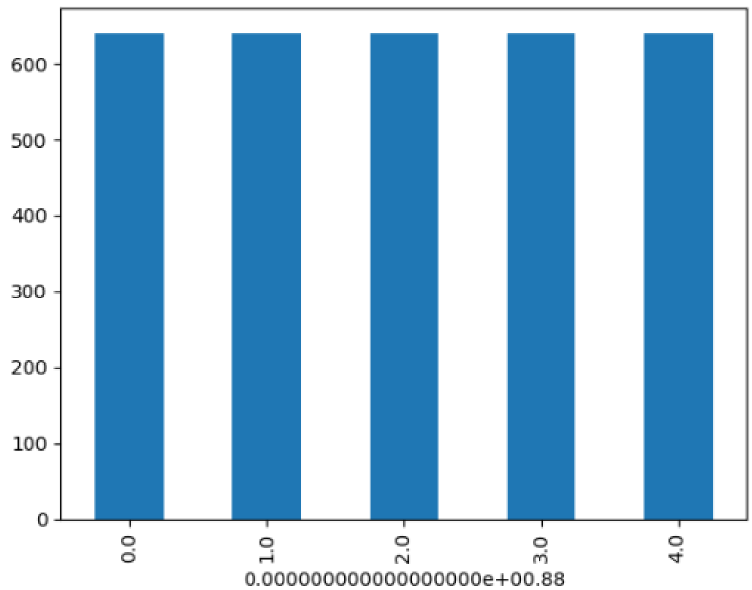


Figure 2.12 The Balanced Dataset after using under-sampling

Classification Reports and Performance Matrix

In below figures will discuss about the performance analysis of used supervised machine learning algorithms through classification reports, confusion matrices, and result evaluations.

```
Random Forest Classifier:
Accuracy: 0.84
Classification Report:
              precision    recall  f1-score   support

         0.0       0.99      0.83      0.90     14579
         1.0       0.30      0.82      0.44       426
         2.0       0.59      0.87      0.70      1112
         3.0       0.12      0.91      0.22       145
         4.0       0.84      0.96      0.90      1249

    accuracy                           0.84     17511
   macro avg       0.57      0.88      0.63     17511
weighted avg       0.93      0.84      0.87     17511


K-Nearest Neighbors Classifier:
Accuracy: 0.74
Classification Report:
              precision    recall  f1-score   support

         0.0       0.98      0.71      0.83     14579
         1.0       0.16      0.80      0.26       426
         2.0       0.50      0.78      0.61      1112
         3.0       0.09      0.94      0.16       145
         4.0       0.77      0.94      0.85      1249

    accuracy                           0.74     17511
   macro avg       0.50      0.83      0.54     17511
weighted avg       0.91      0.74      0.79     17511


Naive Bayes Classifier:
Accuracy: 0.18
Classification Report:
              precision    recall  f1-score   support

         0.0       0.92      0.10      0.19     14579
         1.0       0.19      0.20      0.20       426
         2.0       0.18      0.19      0.18      1112
         3.0       0.02      0.95      0.04       145
         4.0       0.16      0.96      0.27      1249

    accuracy                           0.18     17511
   macro avg       0.30      0.48      0.18     17511
weighted avg       0.80      0.18      0.19     17511
```

Figure 2.13 The Classification reports of Random Forest, KNN,NB algorithms

According to the classification reports in Figure 2.13, Random Forest performed better than the other algorithms.

```
Random Forest Accuracy: 0.950888013248815
Random Forest Classification Report:
              precision    recall  f1-score   support

         0.0       0.95      0.99      0.97     14577
         1.0       0.85      0.36      0.50       418
         2.0       0.92      0.76      0.83      1120
         3.0       0.76      0.19      0.31       152
         4.0       0.97      0.93      0.95      1244

    accuracy                           0.95     17511
   macro avg       0.89      0.64      0.71     17511
weighted avg       0.95      0.95      0.94     17511

K-Nearest Neighbors Accuracy: 0.9516875107075553
K-Nearest Neighbors Classification Report:
              precision    recall  f1-score   support

         0.0       0.96      0.99      0.97     14577
         1.0       0.76      0.39      0.52       418
         2.0       0.90      0.78      0.84      1120
         3.0       0.62      0.32      0.42       152
         4.0       0.97      0.94      0.95      1244

    accuracy                           0.95     17511
   macro avg       0.84      0.68      0.74     17511
weighted avg       0.95      0.95      0.95     17511

Naive Bayes Accuracy: 0.8367883044943178
Naive Bayes Classification Report:
              precision    recall  f1-score   support

         0.0       0.91      0.92      0.91     14577
         1.0       0.00      0.00      0.00       418
         2.0       0.29      0.46      0.35      1120
         3.0       0.00      0.00      0.00       152
         4.0       0.80      0.63      0.71      1244

    accuracy                           0.84     17511
   macro avg       0.40      0.40      0.39     17511
weighted avg       0.83      0.84      0.83     17511
```

Figure 2.14 The Classification reports of Random Forest, KNN,NB algorithms with GWO optimization

Figure 2.14 displays the classification reports alongside the Grey Wolf optimizer. According to this report, Random Forest and KNN perform well.

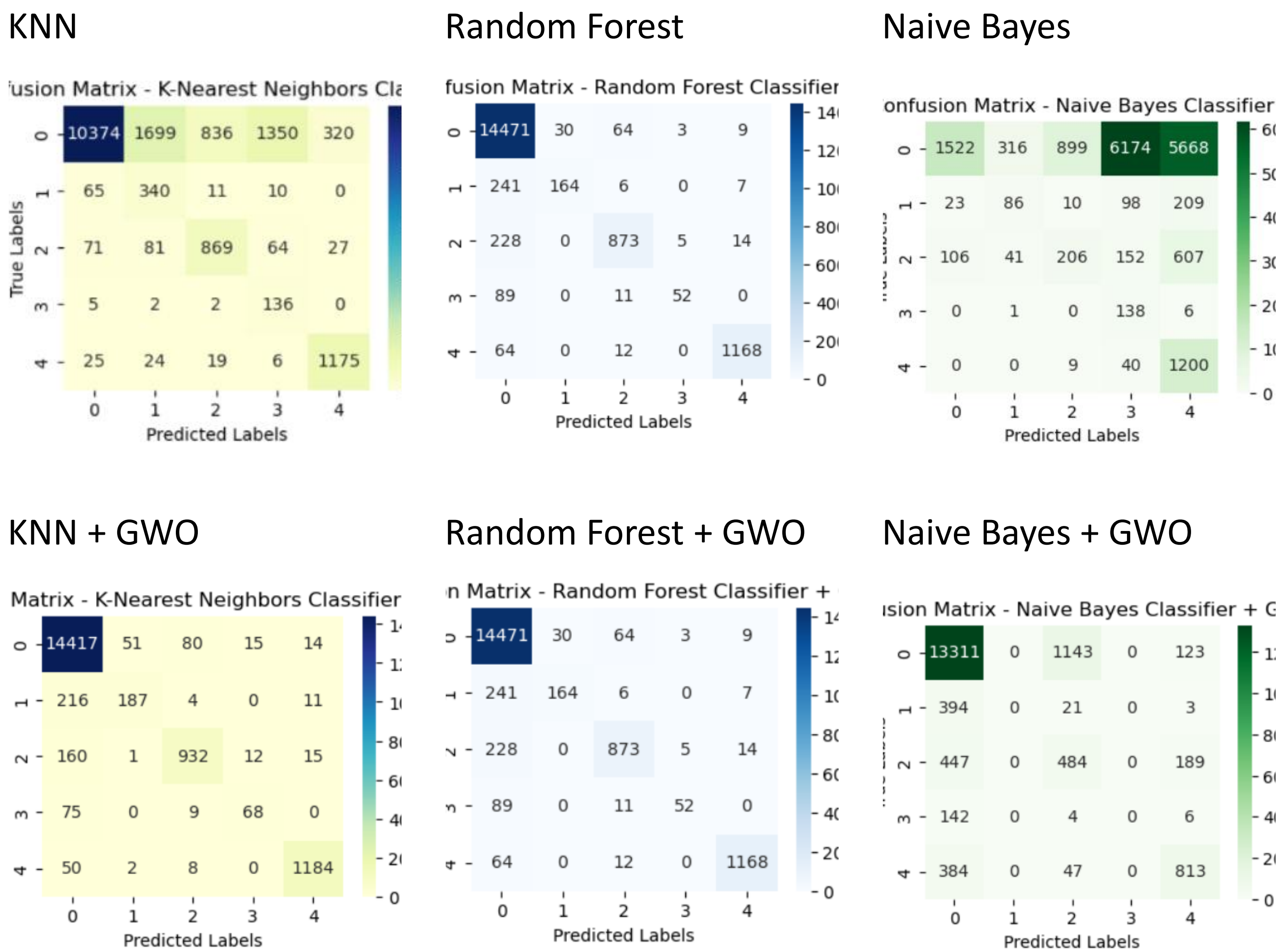
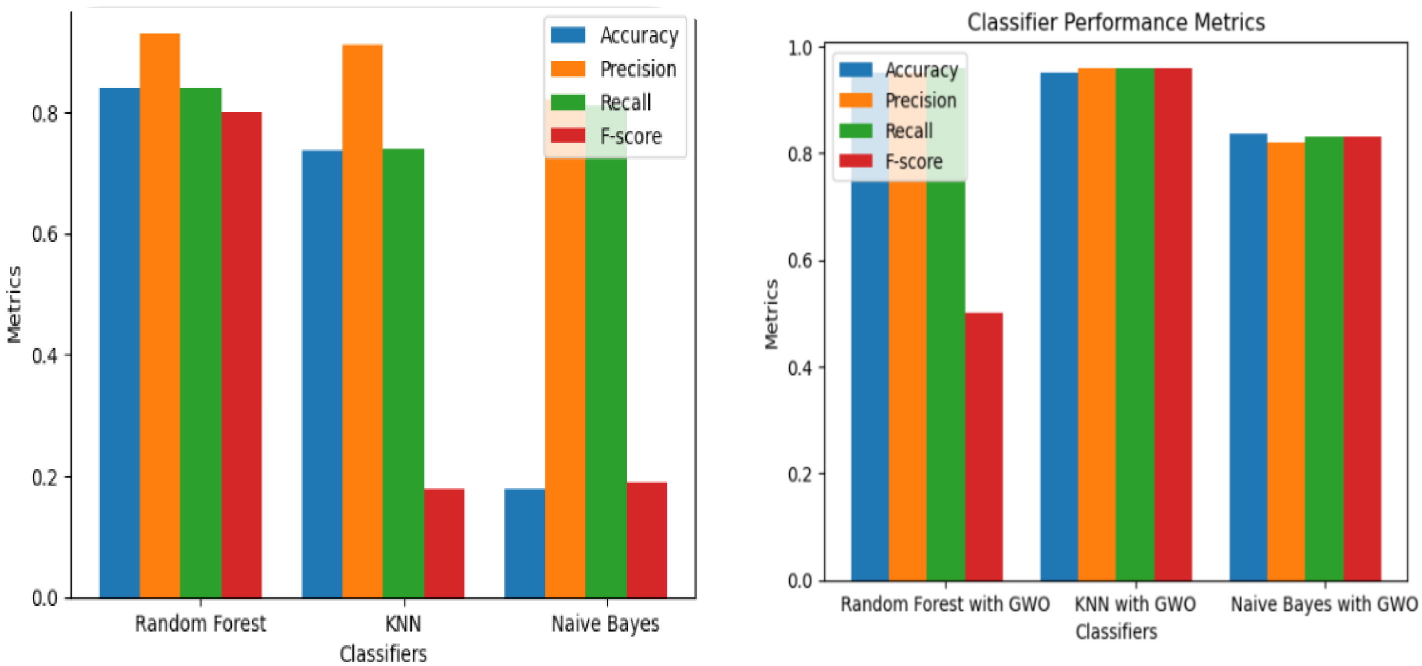


Figure 2.15 The confusion matrix of Random Forest, KNN,NB algorithms with and without GWO



Performance Metrix Without GWO

Performance Metrix With GWO

Figure 2.16 The performance comparison of Random Forest, KNN,NB algorithms with and without GWO

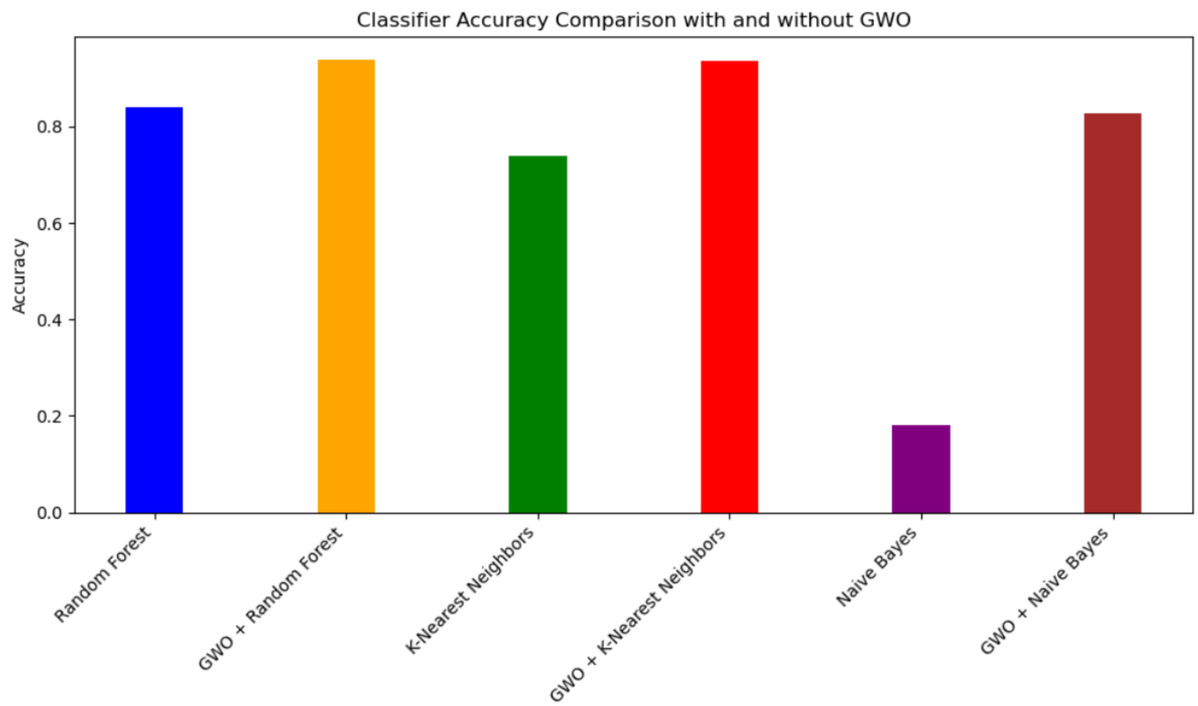


Figure 2.17 The accuracy comparison of Random Forest, KNN,NB algorithms with and without GWO

Based on Figure 2.16, Figure 2.17 the final results show that algorithms with the GWO optimizer yield higher results compared to those without it, which give lower results. Therefore, it can be concluded that the Grey Wolf optimizer performs well for the MIT BIH Arrythmia Dataset and can increase accuracy.

3. Implementation

Figure 3.1 shows a the the process involved in analyzing the performance of arrhythmia classification. This process includes preprocessing, feature extraction and classification, as illustrated in the figure 3.1. After that, classification reports use for a comparative analysis.

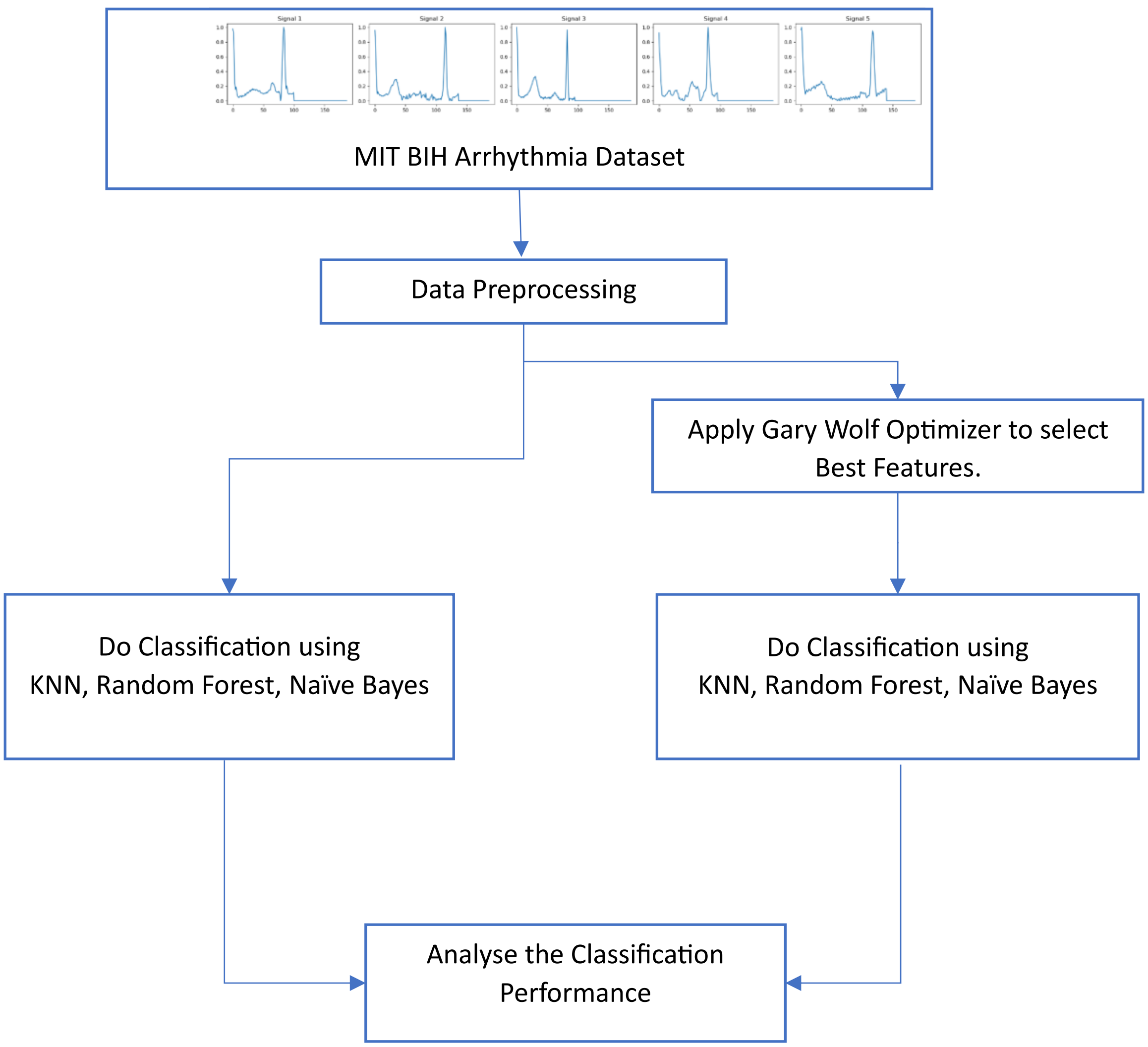


Figure 3.1

This methodology proposes two approaches for classification. The first approach involves KNN, Logistic Regression, and Naïve Bayes for standard classification. The second approach consists of selecting the most suitable features for classification and then performing it. The classification accuracy will be evaluated using classification reports.

A). Dataset

The MIT-BIH Arrhythmia Database is a compilation of 48 half-hour segments of two-channel ambulatory ECG recordings. These recordings were obtained from a research study conducted by the BIH Arrhythmia Laboratory between 1975 and 1979. This ensures that the dataset represents a range of cardiac abnormalities that might not be well-represented in a small random sample. The original dataset comprises 109446 ECG recordings, sampled at 125Hz. Voltage values are recorded at 125 intervals per second.

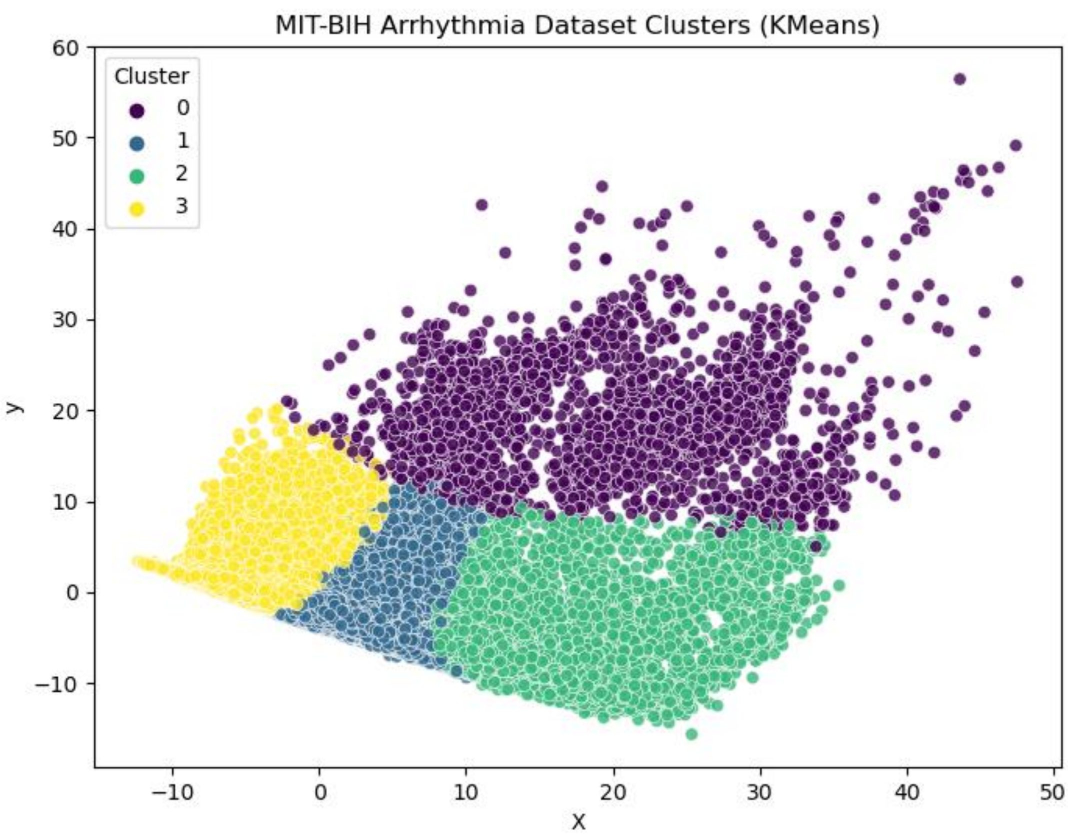


Figure 3.2 Clusters of MIT BIH dataset

The dataset contains five categories, each with a corresponding label representing different heartbeat classes.

- 'N': Normal heartbeat

- 'S': Supraventricular premature beat

- 'V': Premature ventricular contraction

- 'F': Fusion of ventricular and regular beat

- 'Q': Unclassifiable beat

- Classes: ['N': 0, 'S': 1, 'V': 2, 'F': 3, 'Q': 4]

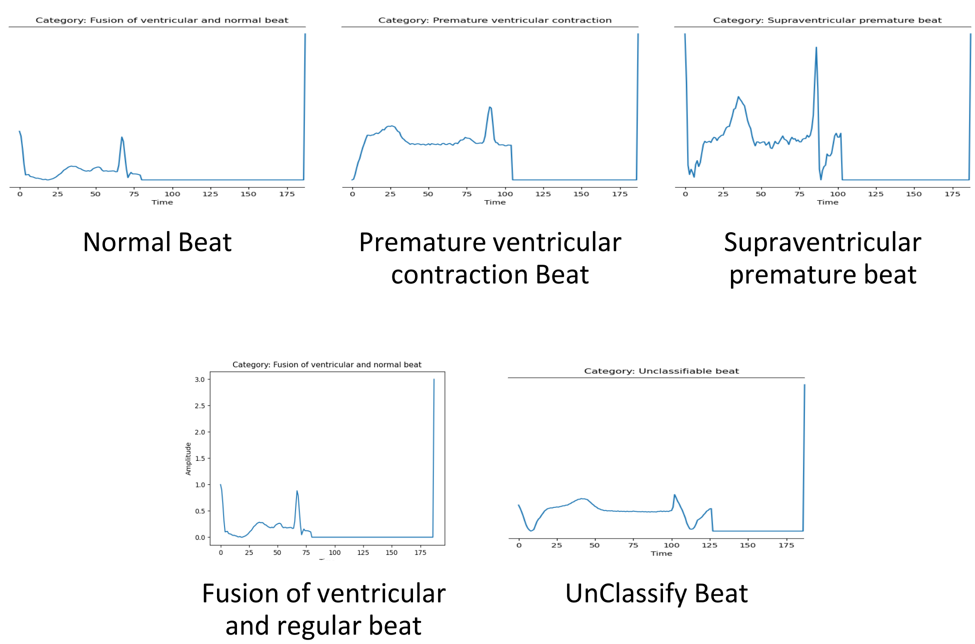


Figure 3.3 Types of Signals in the Dataset

The data was adjusted to a frequency of 360 samples per second for each channel. The measurement system had a resolution of 11 bits and could detect changes as small as 10 millivolts. Each record was reviewed and annotated the group of cardiologists. There are over 100,000 annotations in this dataset. Half of this has been publicly available since September

1999, but the remaining 23 signal files were previously only available on the MIT-BIH Arrhythmia Database CD-ROM.

```
Category
Normal heartbeat                       72471
Unclassifiable beat                     6431
Premature ventricular contraction       5788
Supraventricular premature beat         2223
Fusion of ventricular and normal beat    641
Name: count, dtype: int64
Out[28]: <Axes: xlabel='Category'>
```

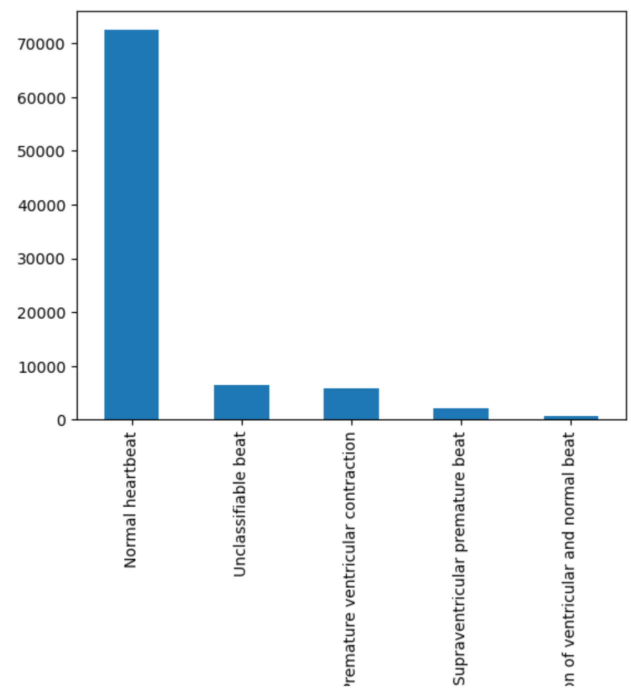


Figure 3.4 MIT BIH dataset

Figure 3.4 shows the number of categories and the amount of data in each category. Based on the data given, it is evident that the dataset needs to be more balanced. This can lead to inaccurate predictions, which is why it is essential to balance the data. The preprocessing step will address this issue.

B). Data Preprocessing

Data preprocessing is a crucial thing in the classification process. Since the dataset is very unbalanced here, apply an under-sampling mechanism.

In under-sampling, the primary objective is to lower down the sample size of the majority class. The aim is to bring the class distribution into a more balanced state. This can be achieved by randomly or strategically removing cases from the majority class or classes until a more equitable representation is achieved. (Brownlee, 2020)

```
0.000000000000000000e+00.88
0.0    641
1.0    641
2.0    641
3.0    641
4.0    641
Name: count, dtype: int64
```

```
Out[29]: <Axes: xlabel='0.000000000000000000e+00.88'>
```

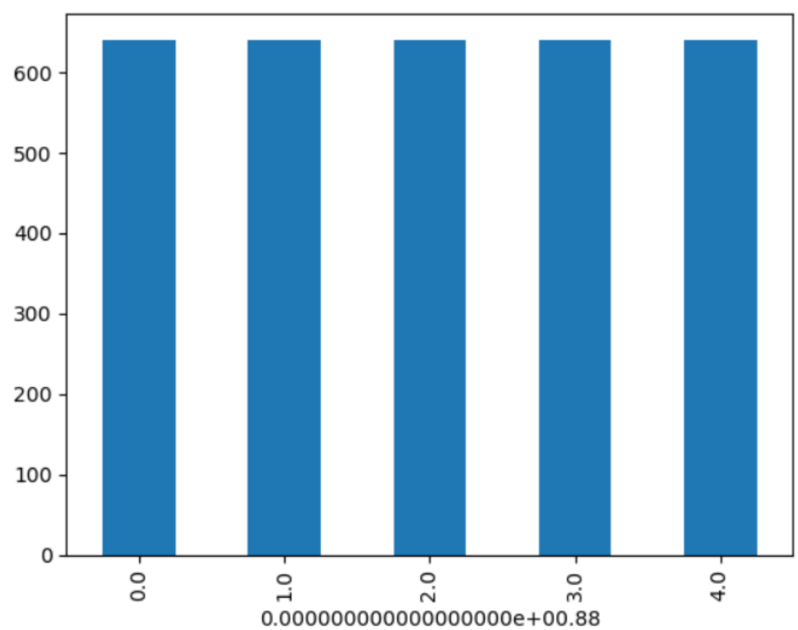


Figure 3.5 Balanced Dataset

C). Classification

Three different classification algorithms were used to implement this project: k-nearest neighbors, Random Forest, and Naïve Bayes. *A detailed explanation of each algorithm will be given below.*

- ### K-nearest neighbors

The K-Nearest Neighbors (KNN) algorithm is a simple and robust machine-learning approach that deals with classification difficulties. It predicts new data points by considering their closest neighbors.

Let's consider a training dataset, denoted as X. This dataset comprises n number of points, and every point is represented a feature vector called d-dimension and denoted as “X_i. The distance calculating using matrix like Euclidean distance.

$$\text{distance}(x, X_i) = \sqrt{\sum_{j=1}^{d}(x_j - X_{i_j})^2}$$

Equation 1: Distance Function

The method identifies the K data points in dataset X that are closest to a new data point, x In these tasks, the algorithm allocate the label y for new X data points and that is most common among its K nearest neighbors. (GeeksforGeeks, 2021).

- ## Random-Forest

Random Forest Regression is a machine learning ensemble technique performing well in regression and classification tasks. It uses multiple decision trees and Bootstrap Aggregating (bagging). The main idea is to combine the predictions from many decision trees to obtain the final output rather than relying solely on the results of individual decision trees. This approach improves the model's robustness and generalization ability, better capturing complex patterns and enhancing overall performance.

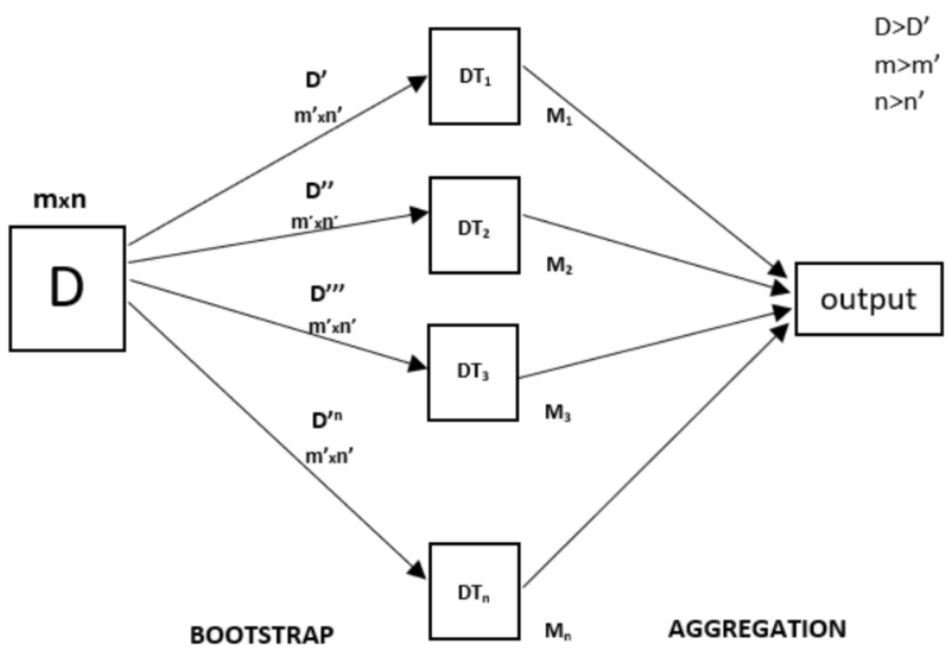


Figure 3.6 Random Forest model work

- Naïve Bayes

Gaussian Naive Bayes (GNB) is a probabilistic classification algorithm based on Bayes' theorem, which provides a formal framework for making predictions by considering the likelihood of events given certain conditions.

The prior probability, P(C), is the likelihood of observing a class without considering any features. In GNB, this is estimated from the training dataset as the ratio of occurrences of each class to the total number of observations.

$$P(C) = \frac{\text{Number of instances of class } C}{\text{Total number of instances}}$$

Equation 2: Prior Probability Calculation Function

The features within each class in GNB are assumed to be normally distributed. Thus, we estimate the mean (μi,C) and standard deviation (σi,C) for each feature Xi within class C.

$$P(X_i|C) = \frac{1}{\sqrt{2\pi\sigma_{i,C}^2}} \exp\left(-\frac{(x_i - \mu_{i,C})^2}{2\sigma_{i,C}^2}\right)$$

Equation 3: Probability of Observing a feature

Equation 3 function shows the probability of observing a feature value xi for a given class C. Using Bayes' theorem, the posterior probability P(C|X), i.e., the probability of class C given the observed features X, can be calculated as follows:

$$P(C|X) = \frac{P(X|C) \cdot P(C)}{P(X)}$$

Equation 4: Probability of Class C

In GNB, the denominator P(X) is computed by marginalizing over all possible classes and acts as a normalization term.

$$P(X) = \sum_C P(X|C) \cdot P(C)$$

Equation 5

Finally, the class with the highest posterior probability is selected as the predicted class for a given observation.

$$\text{predicted class} = \arg\max_C P(C|X)$$

Equation 6: Class Prediction function

- ## Grey Wolf Optimizer (The Novel Approach)

The Grey Wolf Optimizer (GWO) is an optimization algorithm inspired by wolves' social hierarchy and hunting behavior. It aims to mimic their collaborative and hierarchical hunting strategy to optimize complex functions.

Let's denote the position of each grey wolf in the solution space as Xi= [xi1, xi2, ..., xid], where d is the problem's dimensionality. The objective is to optimize the function f(X), where X is the solution vector.

The position of each grey wolf is updated using three main types of wolves: alpha (α), beta (β), and delta (δ) wolves, representing the leader, the sub-leader, and the enforcer, respectively.

The position update for each wolf i is given by:

$$X_i^{t+1} = X_i^t + A \cdot D_i - \beta \cdot X_{\text{alpha}} - \delta \cdot X_{\text{beta}}$$

Equation 7: Position update Functions for each Wolfs

where:

- X_i^t is the current position of wolf i at iteration t,
- A is a coefficient that controls the step size,
- D_i is a random vector in the range $[0, 1]^d$,
- X_{alpha}, X_{beta}, and X_δ are the positions of the alpha, beta, and delta wolves, respectively.

The positions of alpha, beta, and delta wolves are updated based on their fitness values. Wolves with better fitness become leaders and their positions are updated accordingly.

$$
\begin{aligned}
X_{\text{alpha}} &= X_{\text{alpha}}^{t} - \alpha \cdot D_{\text{alpha}} \\
X_{\text{beta}} &= X_{\text{beta}}^{t} - \beta \cdot D_{\text{beta}} \\
X_{\text{delta}} &= X_{\text{delta}}^{t} - \delta \cdot D_{\text{delta}}
\end{aligned}
$$

Equation 8: Positions update for wolves.

The fitness level of each wolf is assessed using an objective function, which can be expressed as Fitnessi = f(Xi).

The coefficient A in the position update equation controls the algorithm's exploration-exploitation balance. A higher A value promotes exploration, while a lower A value enhances exploitation. The algorithm updates wolf positions iteratively until a specified stopping criterion is met, such as reaching a maximum number of iterations or achieving satisfactory fitness. (GeeksforGeeks, 2021).

The MIT-BIH Arrhythmia dataset contains 187 features. Among the various methods used to extract a subset of informative features, implementing the Grey Wolf Optimizer (GWO) is highly effective. It has identified a concise set of 10 golden features that improve computational efficiency by reducing dimensionality and serve as a basis for constructing a highly optimized classification model.

Identifying these ten golden features through GWO is not just a computational achievement but also has profound clinical implications. Each feature encapsulates critical information that can help understand the underlying physiological intricacies contributing to cardiac arrhythmias. These golden features offer clinicians and researchers a succinct and interpretable window into the complex interplay of physiological variables associated with arrhythmia, fostering a deeper understanding of the disease's mechanisms.

The optimum features selected by the GWO are displayed below:

```
Optimum Features Selected by GWO:


['9.264705777168273926e-01', '6.813725233078002930e-01', '1.544117629528045654e-01', '1.911764740943908691e-01', '5.88235296
3089942932e-02', '6.127450987696647644e-02']
```

Figure 3.7 Optimum Features Selected by GWO

4. Result Evaluation and Discussion

To Evaluate the performance, the classification report and confusion matrixes were used. A classification report is a metric used to evaluate the performance of a machine-learning classification model. It assesses the quality of the model by providing a detailed summary of various evaluation metrics, offering insights into its performance across different classes. This report is beneficial in multi-class classification problems. Below, explain the main components of a typical classification report.

Accuracy:

"Accuracy" refers to the degree of correctness exhibited by a model. It is calculated with the help of below equation.

$$Accuracy = \frac{True\ Positive + True\ Negative}{Total\ Instances}$$

Equation 9: Equation for Accuracy

Precision:

Precision measures how accurately the model makes optimistic predictions. It can be calculated using the below function.

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive}$$

Equation 10: Equation for Precision

Recall:

It is computed by dividing the true positives by the sum of true positives and false negatives.

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

Equation 11: Equation for Recall

F1 Score:

The F1-score, a balanced measure of precision and recall, is the harmonic mean of the two. Below function will help to find the F1 score.

$$F1\ Score\ = 2 * \frac{Precisopn * Recall}{Precision + Recall}$$

Equation 12: Equation for F1 Score

The table below displays the performance metrics of the proposed method for KNN, Random Forest, and Naïve Bayes with and without GWO.

Classifiers	Performance Metrix			
	Accuracy	Precision	Recall	F1 Score
KNN	0.74	0.91	0.74	0.79
Random Forest	0.84	0.93	0.84	0.87
Naïve Bayes	0.18	0.80	0.18	0.19
KNN + GWO	0.96	0.96	0.96	0.96
Random Forest + GWO	0.95	0.96	0.96	0.95
Naïve Bayes + GWO	0.83	0.83	0.83	0.83

Table 1: Performance matrixes with KNN, RF, KNN with GWO and Without GWO

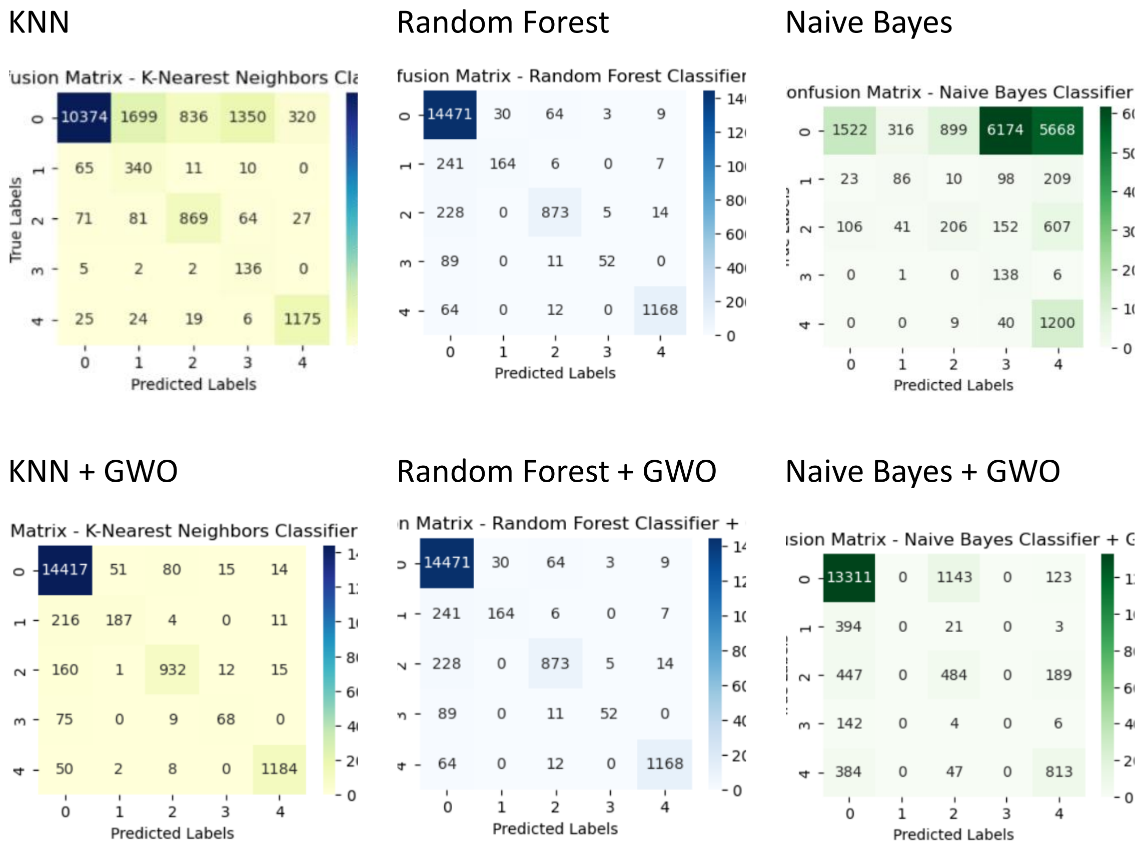


Figure 4.1: Confusion Matrix

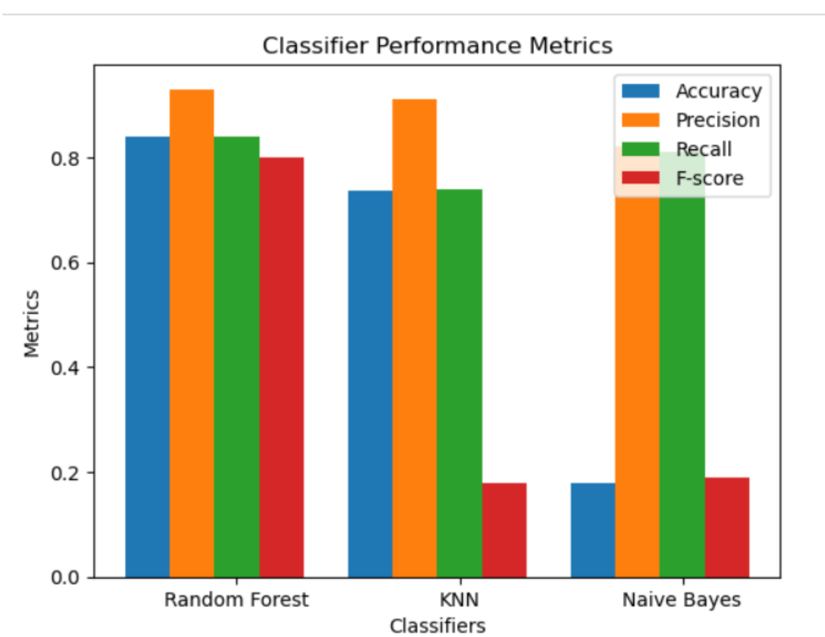


Figure 4.2: RF, KNN, NB Classification Performance Metrix

Based on the information presented in Table 1 and figure 4.2 above, the performance of the NB and KNN models is generally good. However, the Random Forest Classifier outperforms these models in terms of accuracy, precision, recall, and F1 score, making it the best choice overall.

After comparing the implementation of Grey Wolf Optimizer approach by selecting the golden features, the application provided exceptional results for accuracy, precision, recall, and F1 score.

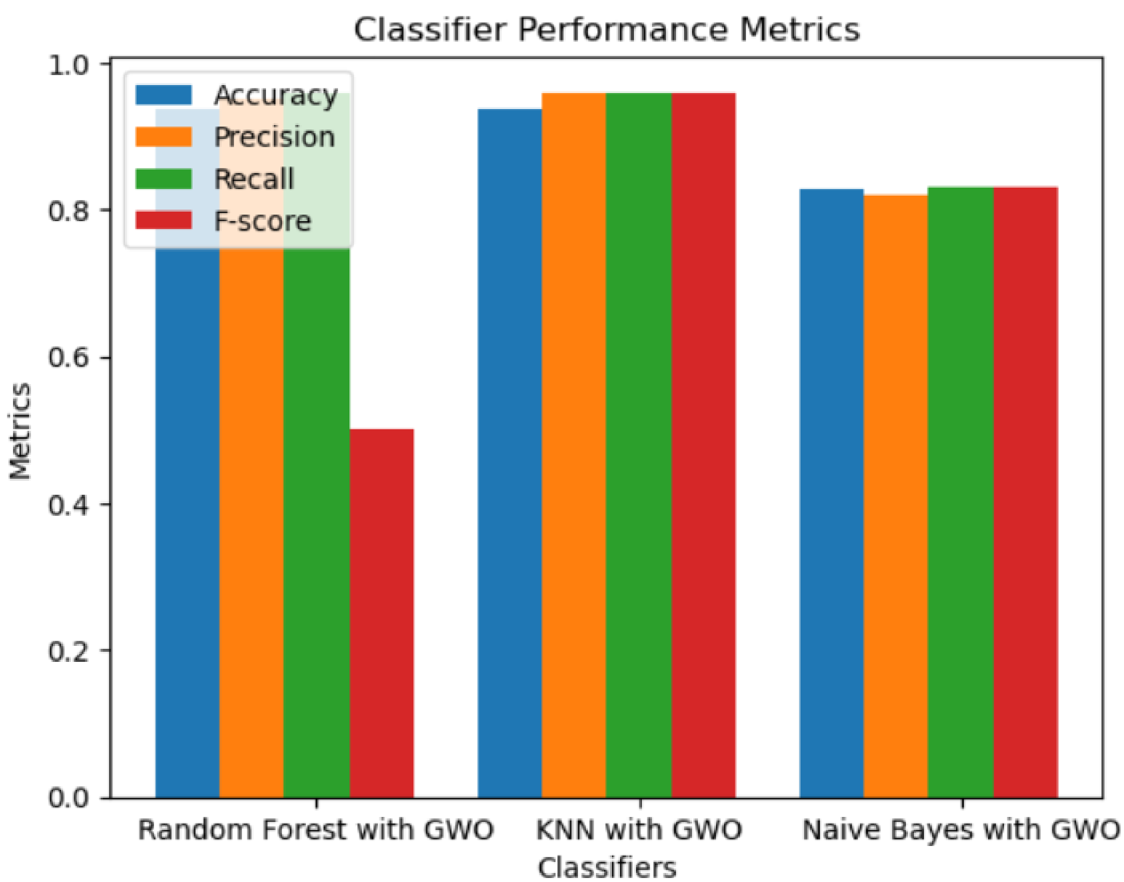


Figure 4.3: Classification Performance with GWO

As shown in the above figure 4.3, all of the classification algorithms provided good classification results for ECG Signals. Therefore, it can be concluded that the provided feature optimization algorithm will help to improve the performance metrics of Random Forest, KNN, and Naïve Bayes algorithms for the domain of ECG Classification.

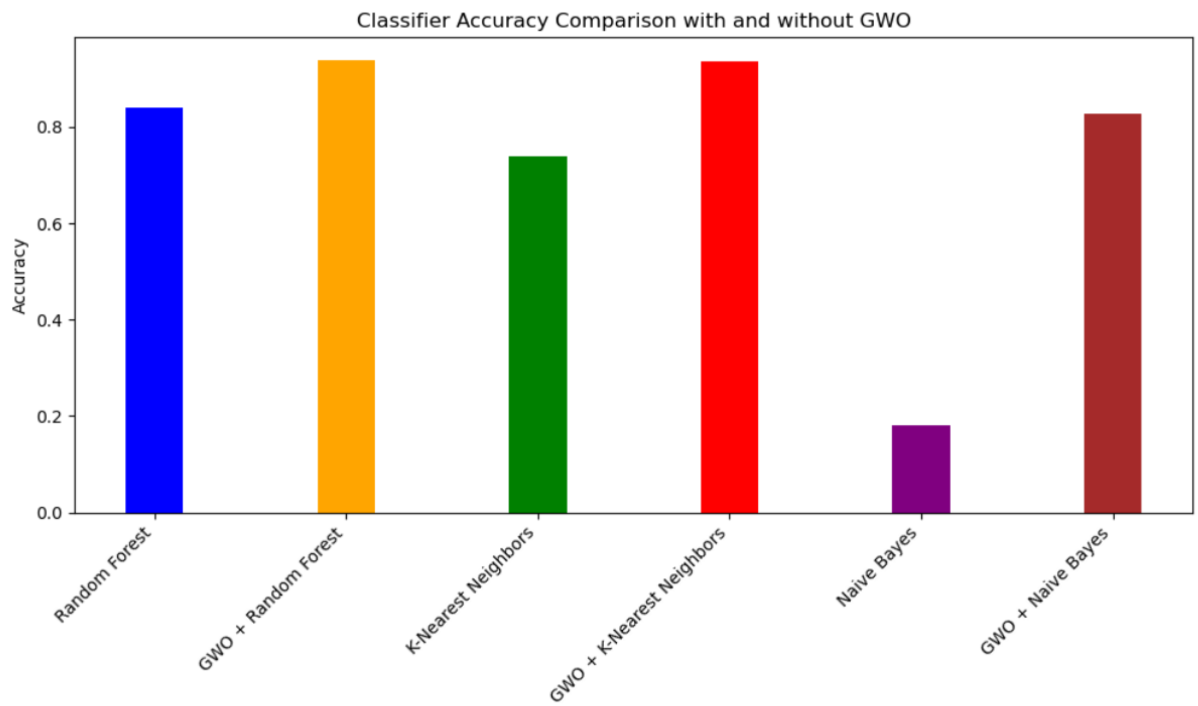


Figure4.4: Classification Accuracy Comparison with GWO and Without GWO

The presented data reveals that among the three classifiers, Random Forest achieved the highest results by 84% accuracy, with a precision, 93%, recall,84%, and F1 score,87%. KNN, on the other hand, obtained an accuracy,74%, precision,91%, recall,74%, and F1 score of 79%. However, Naïve Bayes had the lowest accuracy of 18%, with precision, 80%, recall,18%, and F1 score of 19%. After thoroughly comparing the performance of all three algorithms, it can be concluded that Naïve Bayes had the poorest prediction, while Random Forest gave the best results.

As the previous results were relatively low, this report proposed a method to achieve higher accuracy by optimizing feature selection using the Grey Wolf Optimizer. After applying the Grey Wolf Optimizer, I observed an improvement in classification performance.

After applying the Grey Wolf Optimizer, the accuracy of the Random Forest algorithm was found to be 95%, with 95% precision, 96% recall, and 95% F1 score. Similarly, the K-Nearest

Neighbors (KNN) algorithm achieved an overall accuracy of 95%, with 95% precision, 96% recall, and 95% F1 score. On the other hand, the Naïve Bayes algorithm with optimizer could only achieve an accuracy of 83%, with 83% precision, 84% recall, and 83% F1 score. Based on these results, can conclude that the KNN and Random Forest algorithms perform better with the GWO optimizer, although they have slightly different results.

5. Conclusion

This project outlines an investigation into the detection of cardiac arrhythmia using a new approach that utilizes the Grey Wolf Optimizer to improve the performance of three classifiers. The accurate diagnosis and timely treatment of cardiac arrhythmias are critical, and this investigation explored innovative techniques to improve classification. The initial analysis revealed significant differences in the performance of the classifiers, with Random Forest proving to be the best among KNN and Naive Bayes. However, recognizing the limitations of the original models, the report proposed an innovative solution: using the GWO optimizer for feature selection to enhance the overall classification accuracy.

The application of the GWO optimizer led to significant improvements, particularly for Random Forest and KNN classifiers, achieving an impressive accuracy of 95%. These classifiers also showed high precision, recall, and F1 scores. Although Naive Bayes demonstrated a more modest improvement, its accuracy of 83% highlighted the positive impact of the GWO optimizer. These findings demonstrate the potential of the proposed methodology in improving cardiac arrhythmia detection algorithms. The tailored optimization introduced by GWO demonstrates its capability to refine feature selection, thus empowering classifiers to achieve higher accuracy levels in diagnosing this critical heart condition. The insights from this chapter provide valuable knowledge to the ongoing efforts in cardiac health research and emphasize the importance of exploring innovative techniques to improve diagnostic accuracy.

As a future improvement, this classification can be expanded to include Particle Swarm Optimization and compare it with GWO to find the most accurate results.

6. References

I. Zaharia, T., Danciu, G., Ilie, I., Irina Emilia Nicolae and Septimiu Nechifor (2023). A simplified Approach for Accurate Arrythmia Detection using Automated Machine Learning. doi:https://doi.org/10.1109/atee58038.2023.10108192.

II. Sandipan Chakroborty and Patil, M.A. (2014). Real-time arrhythmia classification for large databases. PubMed. doi:https://doi.org/10.1109/embc.2014.6943873.

III. M. Habijan, I. Galić and A. Pizurica, "Classification of Arrhythmia ECG Signals using Convolutional Neural Network," 2023 30th International Conference on Systems, Signals and Image Processing (IWSSIP), Ohrid, North Macedonia, 2023, pp. 1-5, doi: 10.1109/IWSSIP58668.2023.10180281.

IV. R. Bouchouareb and K. Ferroudji, "Classification of ECG Arrhythmia using Artificial Intelligence techniques (RBF and SVM)," 2022 4th International Conference on Pattern Analysis and Intelligent Systems (PAIS), Oum El Bouaghi, Algeria, 2022, pp. 1-7, doi: 10.1109/PAIS56586.2022.9946873.

V. F. Khalaf, M. I. Owis and I. A. Yassine, "Image features of spectral correlation function for arrhythmia classification," 2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), Milan, Italy, 2015, pp. 5199-5202, doi: 10.1109/EMBC.2015.7319563.

VI. P. Dhal and C. Azad, "A multi-stage multi-objective GWO based feature selection approach for multi-label text classification," 2022 2nd International Conference on Intelligent Technologies (CONIT), Hubli, India, 2022, pp. 1-5, doi: 10.1109/CONIT55038.2022.9847886.

VII. H. Mita J, G. Babu C and G. Shankar M, "Performance Analysis for Arrhythmia Classification using PSO, GWO and SVM," 2021 3rd International Conference on Signal Processing and Communication (ICPSC), Coimbatore, India, 2021, pp. 67-72, doi: 10.1109/ICSPC51351.2021.9451729.

VIII. Zaharia, T., Danciu, G., Ilie, I., Irina Emilia Nicolae and Septimiu Nechifor (2023). A simplified Approach for Accurate Arrythmia Detection using Automated Machine Learning. doi:https://doi.org/10.1109/atee58038.2023.10108192.

IX. Mohonta, S.C., Motin, M.A. and Kumar, D.K. (2022). Electrocardiogram based arrhythmia classification using wavelet transform with deep learning model. Sensing and Bio-Sensing Research, [online] 37, p.100502. doi:https://doi.org/10.1016/j.sbsr.2022.100502.

X. Brownlee, J. (2020). Undersampling Algorithms for Imbalanced Classification. [online] Machine Learning Mastery. Available at: https://machinelearningmastery.com/undersampling-algorithms-for-imbalanced-classification/.

XI. GeeksforGeeks. (2021). Grey wolf optimization - Introduction. [online] Available at: https://www.geeksforgeeks.org/grey-wolf-optimization-introduction/.

XII. Chang, Y.-C., Chang, T.-S. and Lin, T.-L. (2023). A Portable Electrocardiography Arrhythmia Classification Using Time-series Augmented Signals and Convolutional Neural Network. doi:https://doi.org/10.1109/icce-taiwan58799.2023.10226700.