Arrhythmia Detection: Evaluation of Various Machine Learning Approaches

Kafi Khan
2017-2-60-078
2017-2-60-078@std.ewubd.edu

Mayisha Tasneem
2017-2-60-010
2017-2-60-010@ std.ewubd.edu

Tahmina Ahmed Prima
2017-1-60-105
2017-1-60-105@std.ewubd.edu

***Abstract*— Arrhythmia is a quite prevalent heart disease among the worldwide population. The most common way to detect it is by analyzing ECG results manually. Understandably, this technique is tedious and error-prone. As machine learning is taking over the world by storm, it is also feasible to detect arrhythmia using machine learning. In this paper, we have compared the results of some of the most well-founded machine learning algorithms. We found that Extreme Gradient Boosting provides the most reliable narrative to detect Arrhythmia.**

***Keywords*— arrhythmia, disease detection, machine learning, algorithm comparison.**

I. INTRODUCTION

Arrhythmia is a form of heart disease that causes a patient's heart to beat either slower and faster than usual. Should it be left undiagnosed, the patient's heart may fail to supply enough blood to different organs of the body. Moreover, it is a widely prevalent disease among the worldwide population. Approximately, three million people in the US are diagnosed with this disease each year[1]. The most common test used to diagnose an arrhythmia is an electrocardiogram (ECG). Due to the non-stationary nature of the ECG signal and the error-prone human nature, it is rather challenging to use traditional handcraft methods, such as time-based analysis of feature extraction and classification.

Machine Learning (ML) describes a computational approach that uses large amounts of data to enable algorithms to "learn", performing tasks that are difficult to achieve via standard software[2]. In the past few decades, ML created paradigm-shifts throughout various fields such as Business, Healthcare, Education, etc. ML paved the way for unparalleled computational approaches to detect diseases from neurodegenerative diseases[3] to cancer detection[4]. Furthermore, many studies have been conducted to determine arrhythmia using ECG results using various machine learning tools.

In this paper, we want to determine the presence of arrhythmia and identify the class of arrhythmia in a patient using different supervised Machine Learning tools and rectify the performance of those tools using accuracy score and F1 score. To do so, we will be using the Arrhythmia Data Set found in the UCI machine learning repository[5].

II. DATA

The dataset consists of 279 features depicting different attributes that are described in [6]. Furthermore, in the dataset, there are 16 target classes. 1 represents normal, 2 to 15 represents different forms of arrhythmia, and 16 represents uncategorized arrhythmia. Moreover, the dataset consists of missing values. The link for the dataset is [5].

III. METHODOLOGY

The methodology will be divided into six sections.

1. PREDICTING MISSING VALUES AND STANDARDIZING THE DATA

We used Simple Imputer[7] from python's Sklearn library to predict missing values. We replaced the missing value with the median. We noticed that mean yield unreliable values if there are any outliers on the dataset. Median is relatively more reliable in such circumstances. Lastly, we used Standard Scaler from python's Sklearn library to standardize the data.

2. REDUCING FEATURES

The arrhythmia dataset has 279 different features. Which makes it incredibly time-consuming to process. Furthermore, it may also reduce the accuracy of the algorithm. As a result, it might be helpful to use some dimensionality reducing algorithms to reduce the number of features. For all of the algorithms, we are using python's Sklearn library.

A. EXTRA TREES CLASSIFIER

Extra Trees Classifier ranks the importance of a feature in correlation with the target attribute. In doing so, the algorithm can figure out which features have the lowest impact on the target class. Therefore, we can drop the features with the lowest impact.

B. PRINCIPAL COMPONENT ANALYSIS

Principal component analysis (PCA) is a technique for reducing the dimensionality of such datasets, increasing interpretability but at the same time minimizing information loss. It does so by creating new uncorrelated variables that successively maximize variance[8]. We used PCA and took the first 29 primary components, as it offers 95% of the variance in the dataset.

C. ISOmap

Isomap is a nonlinear dimensionality reduction method. It is one of several widely used low-dimensional embedding methods. Isomap is used for computing a quasi-isometric, low-dimensional embedding of a set of high-dimensional data points. The algorithm provides a simple method for estimating the intrinsic geometry of a data manifold based on a rough estimate of each data point's neighbors on the manifold[9]. We used ISOmap and took the first 29 features, as it offers 95% of the variance in the dataset.

3. ALGORITHMS

For the algorithms, we have selected some of the established well-performing algorithms in Machine Learning literature. Namely, Neural Network[11], AdaBoost[12], KNeighborsClassifier[13], Gradient Boosting[14], Naive Bayes[15], Support Vector Classification[16], and Random Forest[17] from Sklearn library and Extreme Gradient Boosting[18] from Xgboost library.

4. GENERALIZATION

As we can presume that, to distinctly classify 16 classes, we need a large dataset. However, the dataset we have only has 450 instances. As a result, it might be difficult to achieve substantially high accuracy. To remedy this, we have generalized the 15 arrhythmia classes into one class, thus, leaving us with only two classes.

5. RECURSIVE FEATURE ELIMINATION

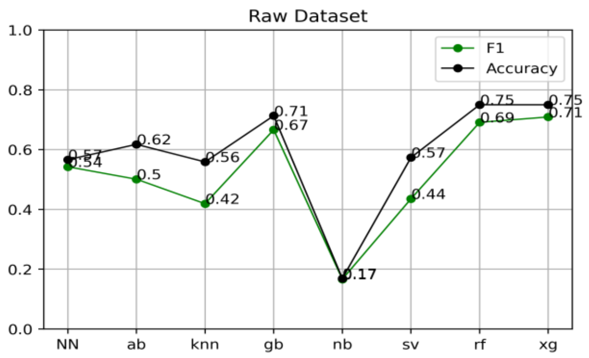
Given an external estimator that assigns weights to features (e.g., the coefficients of a linear model), the goal of recursive feature elimination (RFE) is to select features by recursively considering smaller and smaller sets of features. First, the estimator is trained on the initial set of features and the importance of each feature is obtained either through any specific attribute or callable. Then, the least important features are pruned from the current set of features. That procedure is recursively repeated on the pruned set until the desired number of features to select is eventually reached[10]. We have used Random Forest for RFE and took the first 100 features. RFE can be obtained from the Sklearn library.

6. EVALUATION METRICS

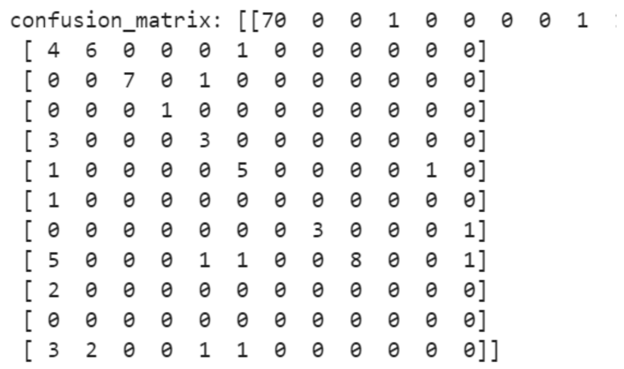
We have used accuracy score, F1 score, and Confusion Matrix to determine the performance of the algorithms.

IV. RESULTS

The graphs in this section contain the accuracy score and F1 score for each algorithm which are denoted as such, Neural Network(nn), AdaBoost(ab), KNeighborsClassifier(knn), Gradient Boosting(gb), Naive Bayes(nb), Support Vector Classification(sv), Random Forest(rf), and Extreme Gradient Boosting(xg).

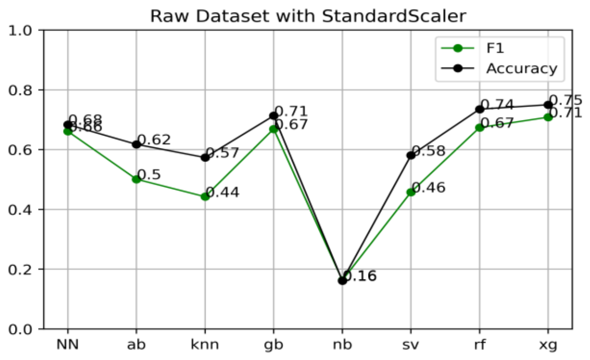


Confusion Matrix for xgboost:



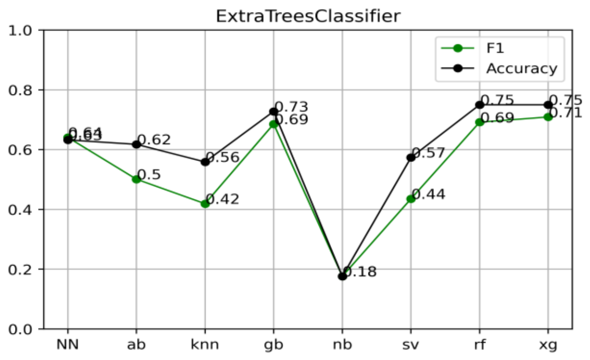
```
confusion_matrix: [[70  0  0  1  0  0  0  0  1
 [ 4  6  0  0  0  1  0  0  0  0  0  0]
 [ 0  0  7  0  1  0  0  0  0  0  0  0]
 [ 0  0  0  1  0  0  0  0  0  0  0  0]
 [ 3  0  0  0  3  0  0  0  0  0  0  0]
 [ 1  0  0  0  0  5  0  0  0  0  1  0]
 [ 1  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  3  0  0  0  1]
 [ 5  0  0  0  1  1  0  0  8  0  0  1]
 [ 2  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0]
 [ 3  2  0  0  1  1  0  0  0  0  0  0]]
```

After running the algorithms on the raw dataset, the ensemble techniques offer quite an accurate prediction.

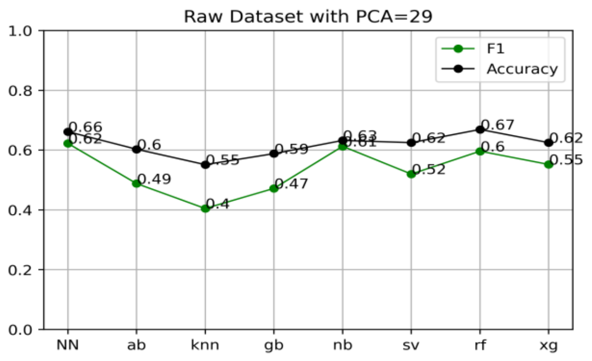


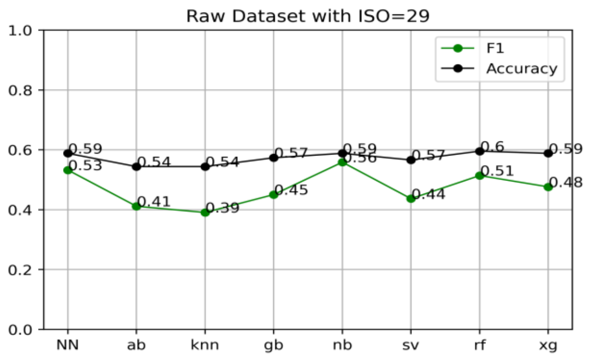
Using StanddardScaler did not seem to have any major effects on the accuracy of the algorithms.

1. DIMENSIONALITY REDUCTION

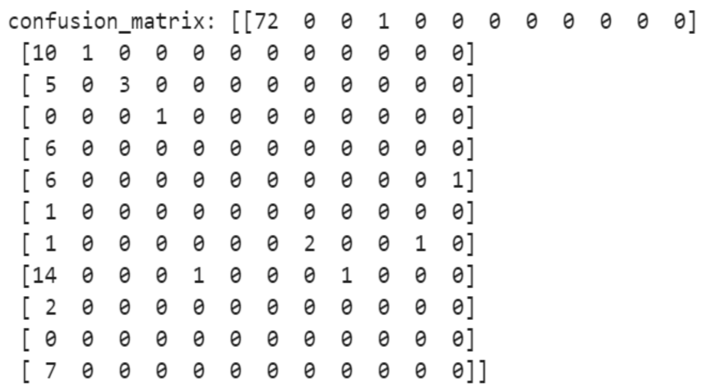


Using ExtraTreesClassifer to reduce dimensionality from 279 to 249 did not put any major stain on the accuracy. However, taking the first 29 components in PCA and ISO, significantly worsened the performance of the ensemble techniques.





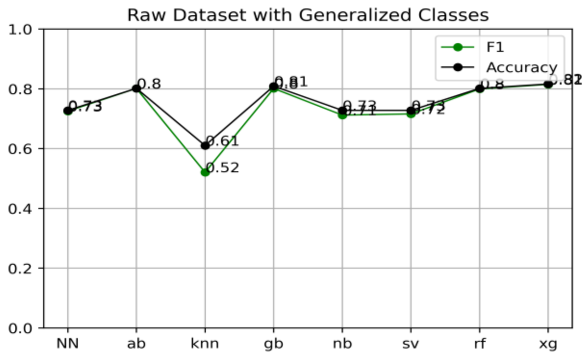
Confusion matrix for xgboost with PCA:



```
confusion_matrix: [[72  0  0  1  0  0  0  0  0  0  0  0]
 [10  1  0  0  0  0  0  0  0  0  0  0]
 [ 5  0  3  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  1  0  0  0  0  0  0  0  0]
 [ 6  0  0  0  0  0  0  0  0  0  0  0]
 [ 6  0  0  0  0  0  0  0  0  0  0  1]
 [ 1  0  0  0  0  0  0  0  0  0  0  0]
 [ 1  0  0  0  0  0  0  2  0  0  1  0]
 [14  0  0  0  1  0  0  0  1  0  0  0]
 [ 2  0  0  0  0  0  0  0  0  0  0  0]
 [ 0  0  0  0  0  0  0  0  0  0  0  0]
 [ 7  0  0  0  0  0  0  0  0  0  0  0]]
```

Furthermore, Naive Bayes performed poorly on the raw dataset, but after using PCA and ISO, the performance of NB increased significantly.

2. GENERALIZATION

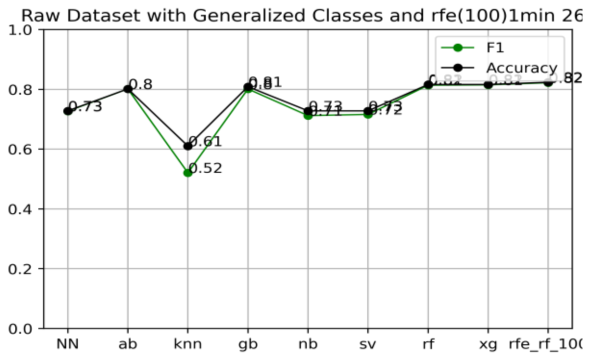


Confusion matrix for xgboost with generalized classes:

```
confusion_matrix: [[64  9]
 [16 47]]
```

Generalization offered a significant increase in the accuracy of the algorithms.

3. RECURSIVE FEATURE ELIMINATION



Confusion matrix for RFE with Random Forest:

```
confusion_matrix: [[65  8]
 [16 47]]
```

Using RFE did provide relatively high accuracy. However, it is incredibly time consuming. For our 450 instances, RFE took around 1 minute and 26 seconds. In contrast, XGBoost only takes around 2 seconds.

V. CONCLUSION

Throughout the entire process, XGBoost performed well quite consistently, closely followed by Random Forest and Gradient Boosting. With this dataset, on the raw dataset, the highest accuracy and F1 score we could achieve is 75% and 71%. Furthermore, using the Extra Trees Classifier provided quite the same accuracy with 30 fewer features, which undoubtedly will reduce the required time for any larger dataset. Moreover, PCA and ISO predictably did not perform well. The main reason being, most of the features serve some purpose in correlation to the classes. Thus, reducing features will understandably reduce the accuracy and F1 score. As the dataset does not have an adequate number of instances to classify 16 classes, it is foreseeable that generalization will provide relatively better accuracy. Here, both XGBoost, Random Forest, and RFE provided 82% accuracy and 82% F1 score. However, XGBoost and Random Forest took only a few seconds to complete the task, in contrast, RFE took 1 minute and 26 seconds. So, for this dataset, we do not believe RFE is a wise choice.
With only 450 instances in the dataset, we could not conduct a thorough analysis of the algorithms. However, it can be stated that for arrhythmia detection Ensamble techniques such as XGBoost, Random Forest, and Gradient Boosting will perform significantly better than other classification algorithms.

VI. RESOURCES

We have formulated three Jupyter Notebooks to conduct this study. The names of the files and link to the files are,

1. Arrhythmia Dataset Classification. Link.
2. Arrhythmia Dataset Classification with PCA and ISO. Link.
3. Arrhythmia Dataset Classification (Generalized Classes). Link.

VII. REFERENCES

1. Heart arrhythmia. (2020, August 09). Retrieved December 30, 2020, from https://www.mayoclinic.org/diseases-conditions/heart-arrhythmia/symptoms-causes/syc-20350668?utm_source=Google
2. Horkoff, J. (2019). Non-Functional Requirements for Machine Learning: Challenges and New Directions. 2019 IEEE 27th International Requirements Engineering Conference (RE). doi:10.1109/re.2019.00050
3. Myszczynska, M., Ojamies, P., Lacoste, A., Neil, D., Saffari, A., Mead, R., . . . Ferraiuolo, L. (2020, July 15). Applications of machine learning to diagnosis and treatment of

neurodegenerative diseases. Retrieved January 05, 2021, from https://www.nature.com/articles/s41582-020-0377-8

4. Savage, N. (2020, March 25). How AI is improving cancer diagnostics. Retrieved January 05, 2021, from https://www.nature.com/articles/d41586-020-00847-2
5. (n.d.). Retrieved January 05, 2021, from https://archive.ics.uci.edu/ml/datasets/arrhythmia
6. H. Altay Guvenir, Burak Acar, Gulsen Demiroz, Ayhan Cekin "A Supervised Machine Learning Algorithm for Arrhythmia Analysis." Proceedings of the Computers in Cardiology Conference, Lund, Sweden, 1997
7. Sklearn.impute.SimpleImputer¶. (n.d.). Retrieved January 05, 2021, from https://scikit-learn.org/stable/modules/generated/sklearn.impute.SimpleImputer.html
8. Jolliffe Ian T. and Cadima Jorge, Ian T. Jolliffe Ian T. Jolliffe College of Engineering, Jolliffe, I., Ian T. Jolliffe College of Engineering, Cadima, J., Jorge Cadima Secção de Matemática (DCEB), &amp; One contribution of 13 to a theme issue 'Adaptive data analysis: theory and applications'. (2016, April 13). Principal component analysis: A review and recent developments. Retrieved January 05, 2021, from https://royalsocietypublishing.org/doi/10.1098/rsta.2015.0202
9. J. B. Tenenbaum, V. de Silva, J. C. Langford, A Global Geometric Framework for Nonlinear Dimensionality Reduction, Science 290, (2000), 2319–2323.
10. Sklearn.feature_selection.RFE¶. (n.d.). Retrieved January 05, 2021, from https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.RFE.html
11. Larry Hardesty | MIT News Office. (n.d.). Explained: Neural networks. Retrieved January 05, 2021, from https://news.mit.edu/2017/explained-neural-networks-deep-learning-0414
12. Kurama, V. (2020, December 22). A Guide To Understanding AdaBoost. Retrieved January 05, 2021, from https://blog.paperspace.com/adaboost-optimizer/
13. Sklearn.neighbors.KNeighborsClassifier¶. (n.d.). Retrieved January 05, 2021, from https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html
14. Sklearn.ensemble.GradientBoostingClassifier¶ (n.d.). Retrieved January 05, 2021, from https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier.html
15. Sunil RayI am a Business Analytics and Intelligence professional with deep experience in the Indian Insurance industry. I have worked for various multi-national Insurance companies in last 7 years. (2020, October 18). Learn Naive Bayes Algorithm: Naive Bayes Classifier Examples. Retrieved January 05, 2021, from https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/
16. Gandhi, R. (2018, July 05). Support Vector Machine - Introduction to Machine Learning Algorithms. Retrieved January 05, 2021, from https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47
17. Yiu, T. (2019, August 14). Understanding Random Forest. Retrieved January 05, 2021, from https://towardsdatascience.com/understanding-random-forest-58381e0602d2
18. Brownlee, J. (2020, August 17). Extreme Gradient Boosting (XGBoost) Ensemble in Python. Retrieved January 05, 2021, from https://machinelearningmastery.com/extreme-gradient-boosting-ensemble-in-python/