Context-aware Weighted KNN with Locally Adaptive k

Quarter 2 Project - 03/27/25
Mentor: Dr. Yilmaz

Anieesh Saravanan
Machine Learning 1/2
Thomas Jefferson HS for Science and Technology
2025asaravan@tjhsst.edu

Abstract

Context-aware methods are increasingly implemented to enhance classification algorithms in high-dimensional, noisy environments. Traditional K-Nearest Neighbors (KNN) struggles in these environments because it uses a fixed neighborhood size and equal feature weights, which fail to capture local variations present in complex data such as electrocardiogram (ECG) signals. We propose an improved context-aware weighted KNN algorithm that dynamically adjusts its neighborhood size based on local data density and assigns feature weights derived from both mutual information and Pearson correlation. Experiments on the UCI Arrhythmia dataset reveal that our method enhances class separability and overall classification performance, achieving an accuracy of up to 71.43% with optimized parameters. These results indicate strong potential for future clinical applications and further research in adaptive classification methods.

1 Introduction

Heart arrhythmia, characterized by irregular heartbeats, is a life-threatening medical condition that significantly increases the risks of stroke, heart failure, and sudden cardiac death. Early and accurate diagnoses of arrhythmia are essential for effective intervention, reducing these risks with preventative treatment. However, classifying arrhythmia from ECG signals poses challenges due to high dimensionality and inherent noise in the data.

Traditional algorithms like K-Nearest Neighbors are widely favored for their simplicity and interpretability. Yet, the use of a fixed number of neighbors and uniform feature weighting in standard KNN often fails to capture the subtle variations in ECG signals and other medical datasets, where the underlying structure of the data is essential. For arrhythmia detection, certain signal features like amplitude, temporal intervals, and frequency components vary significantly in relevance to the actual diagnosis.

In our work, the input to the algorithm is a set of numerical features extracted from the UCI Arrhythmia dataset. Each feature represents a patient biometric (height, weight, age, heart rate) or distinct characteristic of the ECG signal (duration, channels, wave existence, etc.). We propose a novel KNN algorithm that dynamically adapts its neighborhood size to the local data density with a specialized weighting scheme to output a patient classification: nonarrhythmia (0) or arrhythmia (1). With these developments, our method is able to model local structures, reduce the detrimental effects of noise, and ultimately improve the reliability of arrhythmia detection.

2 Related Works

Existing literature on arrhythmia classification can be categorized into three broad categories:

1. **Signal Pre-Processing and Fixed-Feature Extraction**

   One common approach involves extensive signal preprocessing techniques. For example, the work titled *Classification of Arrhythmia using KNN-Classifier*[1] employs a discrete wavelet transform (DWT) combined with higher order statistics (HOS) to compress ECG signals into feature vectors, which are then fed into a KNN classifier. This approach obtains very high precision and recall (both around 96.30%), demonstrating the effectiveness of time-frequency domain analysis. However, this methodology requires extensive manual preprocessing and a fixed feature extraction pipeline, severely limiting its adaptability to real-world datasets.

[1] https://www.academia.edu/29971108/Classification_of_Arrhythmia_using_KNN_Classifier

2. **Classifier Comparisons with Feature Selection**
   Another area of research compares multiple classifiers using automated feature selection methods. In *Heart Arrhythmia Classification Using Machine Learning Algorithms*[2], multiple classifiers (SVM, KNN, and Naive Bayes) are evaluated on the UCI Arrhythmia dataset after using the Boruta feature selection technique. Although SVM and Naive Bayes achieved accuracies in the low 70%s, the standard KNN model trails behind with an accuracy of around 62.6%. Additional studies, such as *Classification of Arrhythmia*[3], achieved an accuracy of 66.96% using standard KNN with a specialized selection of features. These studies highlight the necessity of balancing feature selection with the classification itself when dealing with high-dimensional data.

3. **Algorithmic Approaches with Feature Projection**

   Other approaches have focused on automating feature extraction with projection techniques. In *A Supervised Machine Learning Algorithm for Arrhythmia Analysis*[4], the VF15 Algorithm is introduced. It leverages individual feature intervals and majority voting to perform classifications. By representing each feature separately and combining their votes, VF15 automates part of the feature extraction process. Despite the innovative voting technique used, V15 has moderate performance (62% accuracy) and still depends on a supervised training phase that requires sensitive parameter tuning. Building on these ideas, the paper *Identifying Best Feature Subset for Cardiac Arrhythmia Classification*[5] introduces a hybrid method that combines an improved F-score filter with a Sequential Forward Search (SFS) to select a subset of features. Although this approach reduces the feature space and achieves an accuracy of 73.8% using conventional KNN, it is inherently static. It does not adjust to local data variations, and treats features equally with a fixed neighborhood size.

These works suggest that an adaptive approach that considers both local data density and feature relevance could result in significant improvements.

[2] https://ieeexplore.ieee.org/document/10431441
[3] http://www.ijoee.org/uploadfile/2013/1006/20131006034620314.pdf
[4] https://ieeexplore.ieee.org/abstract/document/647926
[5] https://ieeexplore.ieee.org/document/7237188

3 Dataset and Features

Our study utilizes the UCI Arrhythmia dataset, originally compiled by researcher H. Altay Guvenir. It contains 452 instances with 279 features and covers a broad range of clinical ECG measurements. It is particularly applicable due to its comprehensive representation of heart arrhythmia indicators and high-dimensionality, which evaluates the utility of our methods.

1. **Data Preprocessing**

   The dataset contained missing values (marked by "?"), which we imputed using the mean for numerical features and mode for categorical features. We then standardized all numerical features to ensure variables had equal weightage within distance computations used in the KNN algorithm. Additionally, one-hot encoding was applied to all categorical features for distance computations. Given the high dimensionality of the dataset, we applied Principal Component Analysis (PCA) provided by scikit-learn to preserve 95% of the variance. This not only reduced the computational complexity of the model but also prevented the curse of dimensionality by focusing on the most influential features. The dataset distinguishes between 15 different arrhythmia conditions:

   class: Class Codes 01-16

   a. Code 01: Normal
   b. Code 02: Ischemic changes (Coronary Artery Disease)
   c. Code 03: Old Anterior Myocardial Infarction
      …
   p. Code 16: Others

   For the purposes of this study, One-vs-Rest (OvR) was used. The dataset's 15 distinct arrhythmia condition classes were collapsed into a binary classification problem, nonarrhythmia (0) vs arrhythmia (1):

   class:

   a. Code 01: Nonarrhythmia
   b. Code 02-16: Arrhythmia

2. **Dataset Split**

   To prepare the dataset for model training and evaluation, we divided the transformed dataset into two subsets:

   - Training Set (80% of the data): Trains the classification models to learn patterns
   - Testing Set (20% of the data): Gives a final and unbiased evaluation of the model

   This split ensures the model has sufficient data to learn patterns while leaving enough unseen data to evaluate the model holistically. It also maintains the class distribution within subsets with stratification to prevent biases in model evaluation. This split resulted in 361 instances/patients in the training subset and 91 instances/patients in the test subset.

4 Methods

Our proposed method iterates upon the traditional KNN algorithm by integrating dynamic feature weighting and adaptive neighborhood selection.

1. **Feature Weighting**

   Each feature is assigned a weight based on two metrics:

   1. **Mutual Information (MI)**: Captures nonlinear dependencies between each feature and the output
   2. **Absolute Pearson Correlation**: Quantifies linear relationships

   These measures are subsequently normalized and amalgamated using the parameter α ($0 \leq \alpha \leq 1$, default 0.5) as defined by the following:

   $$w = \alpha \times \frac{MI}{\Sigma MI + \epsilon} + (1 - \alpha) \times \frac{|Corr|}{\Sigma |Corr| + \epsilon}$$

   This formula ensures that features that contribute significantly to the separation of classes are emphasized in distance calculations. A small constant ε (default 10^{-5}) is added to avoid division by zero.

2. **Adaptive k Selection**

   Instead of using a fixed neighborhood size *k*, our algorithm determines an adaptive k based on local density. Using a fixed number of r (default 10) neighbors, we compute the median distance to estimate this density. Then, the local density is normalized relative to the global minimum and maximum densities from the training set. The adaptive k is then calculated as the following:

   $$k = k_{min} + \frac{local\ density - minimum\ density}{maximum\ density - minimum\ density + \epsilon} \times (k_{max} - k_{min})$$

   k_{min} defaults to 3 and k_{max} defaults to 15

   This allows the algorithm to properly adjust its sensitivity based on data variability.

3. **Distance Metrics and Voting**

   Two complementary distance metrics are computed between the test instance and its neighbors:

   1. **Weighted Euclidean Distance:** Incorporates feature weights directly.
   2. **Local Mahalanobis Distance**: Uses a locally computed covariance matrix (with regularization value λ_{reg}, default 10^{-3}) to account for interdependencies in features.

These distances are amalgamated in a similar fashion to the feature weighting through the parameter β ($0 \leq \beta \leq 1$, default 0.5) as defined by the following:

$$d = \beta \times d_{euclidean} + (1 - \beta) \times d_{mahalanobis}$$

A self-tuning local scale parameter (σ) is derived as the median of the combined distances. A Gaussian kernel then transforms the distances into weights that influence the final vote. The classification is determined by comparing the weighted vote score against a decision threshold (default 0.0).

The configurability of these metrics allows the algorithm to be as applicable as possible. Future implementations can implement hyperparameter tuning to find an optimal configuration.

5 Results

The classifier was evaluated on the preprocessed UCI Arrhythmia dataset.

Adaptive k Parameters:

1. k_{min} and k_{max}: We set $k_{min} = 3$ and $k_{max} = 15$. The lower bound was chosen to ensure that a minimum number of neighbors are considered, avoiding issues with extreme local noise. The upper bound was limited to prevent excessive smoothing in high-density regions.
2. Local Density Parameter r: We use r = 10 as the number of neighbors for estimating local density. This choice was based on preliminary experiments that showed a good balance between capturing local variability and computational efficiency.
3. Regularization Parameter λ_{reg}: Set to 10^{-3} to handle near-singular covariance matrices in the computation for local Mahalanobis distance.

Feature Weighting Parameters (determined through grid search):

1. Blending Factor α: Set to a balanced 0.5. Determines the balance between Mutual Information (MI) and Absolute Pearson Correlation in computing feature weights.
2. Blending Factor β: Set to a balanced 0.5. Controls the relative influences of weighted Euclidean distance and local Mahalanobis distance in the combined distance metric.

Metrics such as accuracy, precision, recall, and F1 score were used to assess the performance of the algorithm based on the binary classification outcomes.

Our initial configuration (with default parameters) resulted in:

1. Accuracy: **68.13%**
2. Precision: **84.21%**
3. Recall: **38.10%**
4. F1 Score: **52.46%**

Figures 1 and **2** depict the results of the algorithm with the base parameters. These already surpass the results obtained by related works.

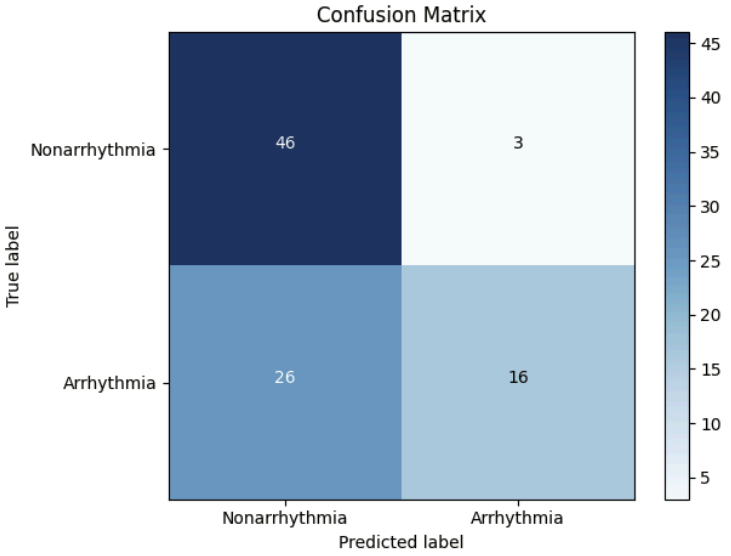

Figure 1: Confusion Matrix with default parameters

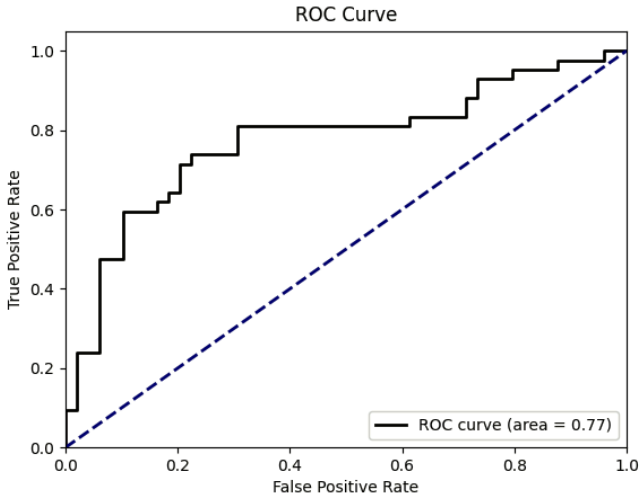

Figure 2: ROC Curve with default parameters

Additional experiments involved sensitivity tests with the parameters defined above. To further investigate the impact of the blending parameters, we performed a grid search (as shown in **Figure 3**) over both α and β values. The below shows a heatmap of accuracy results, where the x-axis represents different values of α and the y-axis represents values of β:

Figure 3: Grid Search Accuracy Heatmap for α and β

Tuning the α and β values marginally improved the results of the proposed KNN model (as shown in **Table 1**). It reached around **71.43% accuracy** on the arrhythmia dataset. This demonstrates this implementation can outperform almost all other models obtained by related works with room for improvement through other parameters. The inherently adaptive nature of the algorithm allows it to address the data, while the weighted distance measures ensure that the most relevant features are emphasized. Due to hardware limitations, it is difficult to perform a more comprehensive grid search for optimal parameters (for reference, the alpha and beta grid search in **Figure 3** had a runtime of around an hour).

	Accuracy (%)	**Precision (%)**	**Recall (%)**	**F1 Score (%)**
Default Parameters	68.13	84.21	38.10	52.46
Optimized Parameters	71.43	86.00	40.50	55.07

Table 1: Comparison between Base Model and α and β tuned Model

The adaptive selection of k allowed the classifier to be more flexible in regions with high variability. In areas where the local density was low, a smaller k preserved and represented intricate structures. In denser regions, a larger k smoothed out noise. The blended distance metric, which combined the weighted Euclidean and Mahalanobis distances, improved the separability of the classes as both contribute complimentary information.

6 Conclusion

Our context-aware weighted KNN algorithm with a locally adaptive neighborhood has demonstrated improved performance over traditional KNN for arrhythmia detection in noisy, high-dimensional datasets. By dynamically selecting the number of neighbors and emphasizing the most relevant features with a blended weighting scheme, the algorithm is able to achieve better class separation and overall accuracy.

7 Future Work

Further exploration is needed to optimize the precision and recall values. Comprehensive hyperparameter tuning using cross-validation can be implemented and the algorithm can be applied to other high-dimensional medical datasets. Due to time and hardware constraints, I was unable to perform a comprehensive grid search. A more thorough grid search over more than two parameters will almost certainly yield improved results. There is significant value in investigating hybrid models that integrate adaptive KNN into deep learning approaches for even greater performance improvements.

8 Contributions

Anieesh Saravanan: Designed and implemented the improved context-aware weighted KNN algorithm; developed the data preprocessing pipeline; conducted experiments; composed report

Dr. Yilmaz: Educated on KNN; provided holistic feedback on report

9 References

Liebman J. (2002). Nonarrhythmia clinical electrocardiography--can it be returned to clinical viability? *Journal of electrocardiology*, *35*(4), 293–298. https://doi.org/10.1054/jelc.2002.36273

University of Rochester Medical Center. (n.d.). *Arrhythmias*. Health Encyclopedia. https://www.urmc.rochester.edu/encyclopedia/content?ContentTypeID=85&ContentID=P00195

Hiremani, V., Devadas, R. M., L, H., S, S., & Pasha, A. (2023). Heart arrhythmia classification using machine learning algorithms. In *Proceedings of the 2nd International Conference on Ambient Intelligence in Health Care (ICAIHC)* (pp. 1–4). IEEE. https://doi.org/10.1109/ICAIHC59020.2023.10431441

Samad, S., Khan, S. A., Haq, A., & Riaz, A. (2014). Classification of arrhythmia. *International Journal of Electrical Energy*, 2(1), 57–61. https://doi.org/10.12720/ijoee.2.1.57-61

Guvenir, H. A., Acar, B., Demiroz, G., & Cekin, A. (1997). A supervised machine learning algorithm for arrhythmia analysis. In *Computers in Cardiology 1997* (pp. 433–436). IEEE. https://doi.org/10.1109/CIC.1997.647926

Niazi, K. A. K., Khan, S. A., Shaukat, A., & Akhtar, M. (2015). Identifying best feature subset for cardiac arrhythmia classification. In *Proceedings of the 2015 Science and Information Conference (SAI)* (pp. 494–499). IEEE. https://doi.org/10.1109/SAI.2015.7237188

mutual_info_classif. (n.d.). scikit-learn. Retrieved March 27, 2025, https://scikit-learn.org/stable/modules/generated/sklearn.feature_selection.mutual_info_classif.html

StandardScaler. (n.d.). scikit-learn. Retrieved March 27, 2025,

https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html

NearestNeighbors. (n.d.). scikit-learn. Retrieved March 27, 2025, from https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.NearestNeighbors.html

SimpleImputer. (n.d.). scikit-learn. Retrieved March 27, 2025, from https://scikit-learn.org/stable/modules/generated/sklearn.impute.SimpleImputer.html

ColumnTransformer. (n.d.). scikit-learn. Retrieved March 27, 2025, from https://scikit-learn.org/stable/modules/generated/sklearn.compose.ColumnTransformer.html

OneHotEncoder. (n.d.). scikit-learn. Retrieved March 27, 2025, from https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.OneHotEncoder.html

Pipeline. (n.d.). scikit-learn. Retrieved March 27, 2025, from https://scikit-learn.org/stable/modules/generated/sklearn.pipeline.Pipeline.html

PCA. (n.d.). scikit-learn. Retrieved March 27, 2025, from https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html