Al-Farouk journal of sciences
Volume (2), Issue (1) (March, 2026)
ISSN: 3135-2359

مجلة الفاروق للعلوم
المجلد (2) العدد 1 مارس 2026
ردمد: 3135-2359

K-Nearest Neighbors for Cardiac Catheterization

TAHANI MUFTAH ABDULSALAM
Computer Science
University of Benghazi
tahani.kasih@uob.edu.ly

Kaled Milad
Computer Science
University of Benghazi
Khaled.milad@uob.edu.ly

Nouri Bader Mahjoub Mohammed
Computer Science
University of Benghazi
Nouri.mohammed@uob.edu.ly

SUAD MOHAMMED MOHAMMED
Computer Science
University of Benghazi
Suad.aljali@uob.edu.ly

تاريخ الاستلام: 2026/01/08 تاريخ المراجعة 17 / 2 / 2026 تاريخ القبول: 2026/03/10- تاريخ النشر: 19/ 03/ 2026

Abstract—Cardiovascular disease (CVD) is the term for conditions affecting the heart and blood arteries. These conditions have been linked to the risk of heart attacks, strokes, and angina. Globally, CVDs are the primary cause of illness and mortality. Cardiac catheterization is a diagnostic and treatment tool for CVDs. The modeling of cardiac catheterization for binary classification can benefit from ML. This study looks into the classification of patients who require monitoring using K-Nearest Neighbors. utilizing a dataset of roughly 814 individuals that was acquired from the Benghazi Heart Disease Center. The research's objective is to build a binary classifier that can predict whether or not a patient needs to be monitored by a physician following a catheterization operation. We discover that, with only minor influences on recall, the K-Nearest Neighbors model achieves an accuracy of 87.09% in identifying the critical patients.

Keywords – Machine Learning (ML), Cardiovascular Disease (CVD), K-Nearest Neighbors (KNN).

The World Health Organization estimates that CVDs, particularly heart attacks and strokes, cause 17 million deaths annually worldwide. especially heart attacks and strokes, is estimated to be 17 million. CVDs affect both men and women almost equally. Even though heart attacks and strokes are among the leading causes of mortality worldwide, 80% of these illnesses' untimely deaths could be prevented by reducing the use of tobacco, eating an unhealthy diet, and engaging in physical inactivity [1], [2]. The American Heart Association/American College of Cardiology (ACC/AHA), which estimates the future risk of CVD, recommends basic methods for risk assessment based on age, smoking, diabetes, cholesterol, hypertension, and cholesterol[3]. There are none of the symptoms of CVD in the early stages of life. The lethal complexity of CVD has surfaced in middle-aged and adults [4].

18% of patients at the Benghazi Heart Disease Center have cardiac catheterization[5]. This study aims to find a suitable machine learning technique to assess whether patients require monitoring after a catheter operation. In this study, the necessary follow-up after a cardiac catheterization operation was predicted using the K-nearest neighbors classification method based on prediction accuracy, precision, recall, and the F1 score.

KNN is easily implementable since it is accurate and simple to use. It can also quickly adjust to new training data and has a small number of hyperparameters[6]

I. Literature Review

Several ML techniques have been used by Jose et al. [7]. to improve the accuracy of coronary artery disease (CAD) prediction by performing a comparative analysis of the 918 cases from the UCI and 11 characteristics. To make sure the test set accuracy is higher than the training set accuracy, they employed a cross-validation approach. They discover that the accuracy of LR is 79% higher than that of other classifiers.

Krstic and Krstic [8] established a model based on clinical data that was effective in forecasting the chances of dying from heart failure. The collection consists of the medical records of 299 patients with 12 clinical characteristics. The model, which was created using RapidMiner, had an accuracy rating of 79.19%. The model's 86.7% AUC value indicates that it can predict mortality risk.

Ngew et al.[9] used information from the National CVD-PCI registry to conduct a retrospective cohort analysis. They separated the 28,007 patient data sets into two groups: testing (3,598) and training (24,409). Four prediction models were produced by them: LR, RF, SVM, and artificial neural networks (ANN). The best method for predicting mortality LR (AUC 0.820).

A hybrid ML model (MLMI) developed by Chattopadhyay and Subhagata [10] is intended to assess the risk of myocardial infarction (MI) in individuals suffering from cardiovascular diseases (CVD). The model classifies patients into high-risk (HR) and not-high-risk (NHR) groups using clustering via the Gaussian mixture model (GMM). It facilitates the best possible care strategies for illnesses by helping to prioritize high-risk patients. To calculate MIRS, the FFNN architecture was created with one input layer, one hidden layer, and one output layer. To reduce research bias, three additional medical professionals validate the MIRS values. To assess the performance of the MLMI model, its sensitivity, specificity, precision, F1 scores, and accuracy are calculated. The suggested model exhibits an average accuracy and precision of 77.33% with small input data, although its sensitivity and F1 score are 100% and 88%, respectively.

Building a ML model to forecast the 10-year risk of CVD was another goal of Dahia and Szabo [11]. Numerous ML models, such as KNN, SVM, RF, Decision Trees (DT), LR, and ANN, were trained using the UCI Heart dataset Cross Validation (CV), normalization methods, and hyperparameter tuning were all implemented to enhance model performance. They provide the findings as well as compare them with traditional classifiers. The best-performing models were the RF and Ada models, with an AUC-ROC of 0.8734.

To figure out how to refine classification models for heart disease prediction based on key performance metrics (Accuracy, Precision, Recall, F-1 score, and AUC curve), this article evaluated the variations between ML and DL using a standard dataset from UCI ML databases of heart illness [12].

Although some studies use large datasets, others use smaller ones. Additionally, some studies use certain methods, such as t-test statistics for continuous features and chi-square statistics for categorical features, to identify significant features. For evaluating a classification model's effectiveness, certain research studies rely on one or two performance metrics, whereas various studies use all of them.

II. Proposal Model

classify and predict patients with cardiovascular diseases. As shown in Fig.1, the proposed model comprises three essential processes: data collection and preparation, classification model construction, and classification model evaluation. These key processes have procedures explain how to build this system.

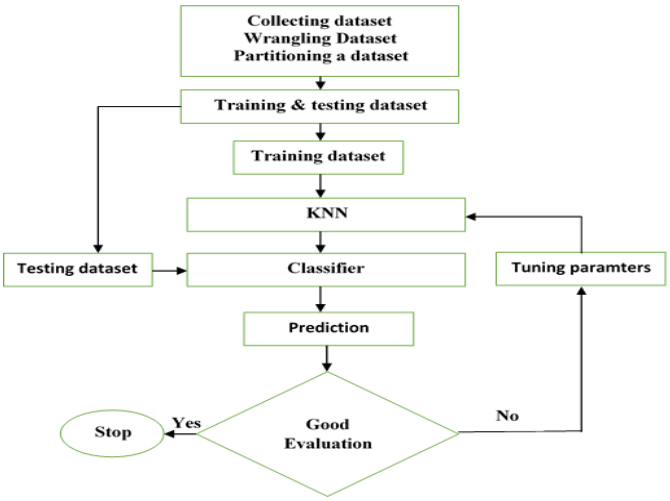


Fig. 1. Block diagram for proposed model.

III. DATASET

It's important to select an accurate dataset set for the learning algorithm[13]. The dataset contains some issues, such as duplicate data and missing or incorrect values. Data preprocessing is a critical step in ML that involves transforming raw data into a suitable format for analysis and modeling[14].

Throughout the collection of the dataset. We gathered information on about 814 patients whose statistics were incomplete or inaccurate. After preparing, the dataset has 21 features and around 772 patients. The dataset has been used to construct training and testing datasets. 80% of our data (651 patients) will be used to train our learning algorithm, and the remaining 20% (163 patients) will be used to test it to ensure that our learning algorithm accurately predicts whether or not a patient needs to follow up.

Gender [F,M], address, stages of age [adults, children, and pediatrics], myocardial Infarction (MI) [+,-], hypertension (HTN) [+,-], diabetes (DM) [+,-], chest pain (Angina) [+,-], smoker [+,-], ischemic heart disease (IHD) [+,-], procedure, ANA system [General, Local], recommendation: labeled classes [not follow up, follow Up], and age [1, 88].

IV. K-NEAREST NEIGHBORS

KNN is an incredibly simple and effective ML algorithm [13], [14], [15]. Unlike other techniques, such logistic regression, K-NN does not actually learn from the data and encodes these learnings in parameters or weights. K-NN uses instance based or lazy learning, which essentially involves saving or memorizing all of the training examples and the relevant classes. The algorithm for class prediction uses a majority vote of the K-nearest points to determine the associated class when given a test sample. That's where the term" K-nearest neighbors" came from[16]. Fig. 2 illustrates how the triangle class name of a new data point (?) is determined by a majority vote among its five nearest neighbors.

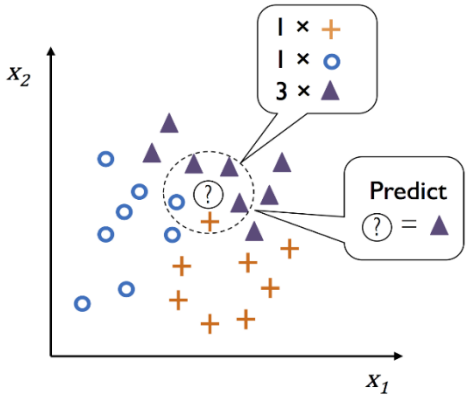


Fig. 2. majority voting among its five nearest neighbors.

The distance, which is simply a generalization of the Euclidean and Manhattan distances, is written as:

$$d(x^i, x^j) = \sqrt[p]{|x^i - x^j|^p} \dots\dots\dots\dots\dots\dots (1)$$

At p=2, it becomes the Euclidean distance, while at p=1, it becomes the Manhattan distance, while k represents the number of neighbors required for each sample [13], [14].

V. THE BIAS-VARIANCE TRADEOFF

Occasionally, we discover that the model performs poorly on the validation data and is unable to generate correct predictions. A model that is too sophisticated to generalize on the validation dataset or one that is too simple to accurately model the data could be the cause of the poor performance. Whereas the model in the first instance has a high variance and leads to overfitting, the model in the second instance has a high bias and produces underfitting [16].

Generally, when we have limited training sets, we choose to utilize a biased model. Insufficient data will cause a high variance model to overfit. When K is equal to the number of training samples, the model is the most biased. Following that, every new data point will have the same label and predicted as the class that makes up the majority. In contrast, the few nearest neighbors within a smaller radius are a better choice to consult when we have a good amount of data, as it is more likely that they will belong to the same class as our new sample [17].

A. Learning Curve:

The learning curve is frequently employed for identifying underfitting and overfitting in classifiers [13], [18]. The learning curve displays the classifier's performance on the validation and training sets through different sized subsets of the training set. In cases when the validation and training curves diverge, the classifier becomes prone to overfitting. When more of the enormous training data was added, the two curves would continue to converge[13].

The variations in the accuracy of several training samples are depicted in Fig. 3. This is precisely what happened in Fig.3, as both curves steadily ascended to the high point and then stabilized. The bias and variance are hence minimal. is the most appealing for a company in that situation. We can add new data without affecting the classifier's complexity. More new instances added will likely aid in the convergence of these curves and assist overcome overfitting.

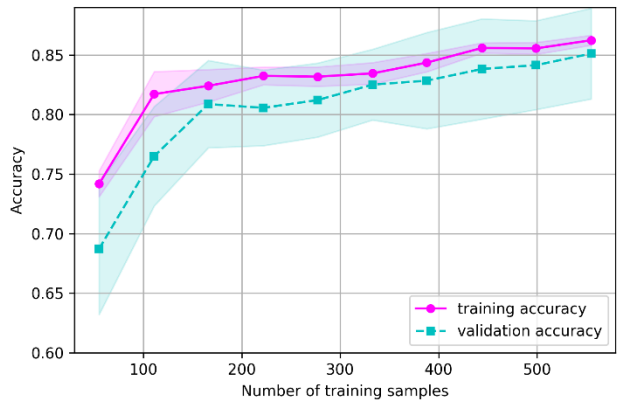


Fig.3. Learning Curve for KNN.

classifier complexity. As a result, the classifier exhibits overfitting when the training and validation curves diverge and underfitting when they converge at low accuracy[19].

Fig. 4 indicates that when the hyperparameter is applied to a number of neighbors (K), the accuracy changes simultaneously. As illustrated in Fig. 4, the training curve was gradually decreased relative to the rises in the validation curve. As k rises, the two curves converge. Additionally, the training and validation's accuracy is suitable and satisfactory.

B. Validation Curve

Validation curves are an effective method for resolving overfitting and underfitting problems in models, therefore enhancing their performance. Validation curves resemble learning curves, with the exception that instead of showing the training and test accuracies as functions of sample size, we alter the values of the model parameters, such as a number of neighbors (K) and distance metric [14]. The validation curve illustrates the relationship between accuracy and

Table 1. The best parameters set discovered by the grid search

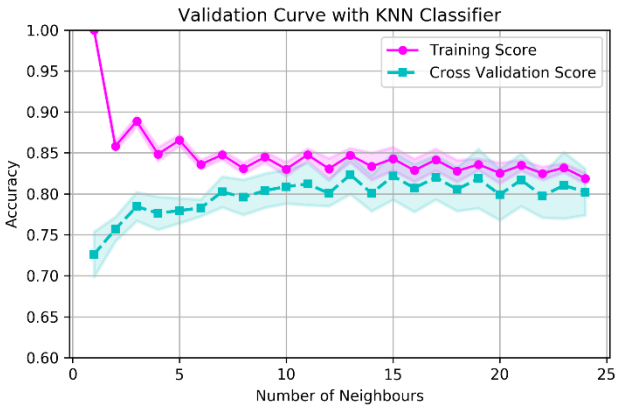


Fig.4 Validation Curve for KNN.

VI. TUNING WITH GRID SEARCH

The procedure of grid search is pretty easy, where we define a list of values for several parameters and the computer estimates the classifier performance for each collection of values to get the optimal collection of values [20]. The number of neighbors and Distance metric are the most significant hyperparameters for KNN. For the number of neighbors, the range of 1 to 20 is suitable[21].

The most effective classifiers on the validation set (Val. set) with the best hyperparameters are shown in TABEL 1. The grid search indicates that the KNN classifier with (k=13 and p=1) is the best one overall. Its performance on the validation set was approximately 87.09%, However, when using (k=11 and p=1), the validation accuracy of the KNN classifiers is around 86.45%. Furthermore, at p=2, the k-nearest neighbors classification performance is

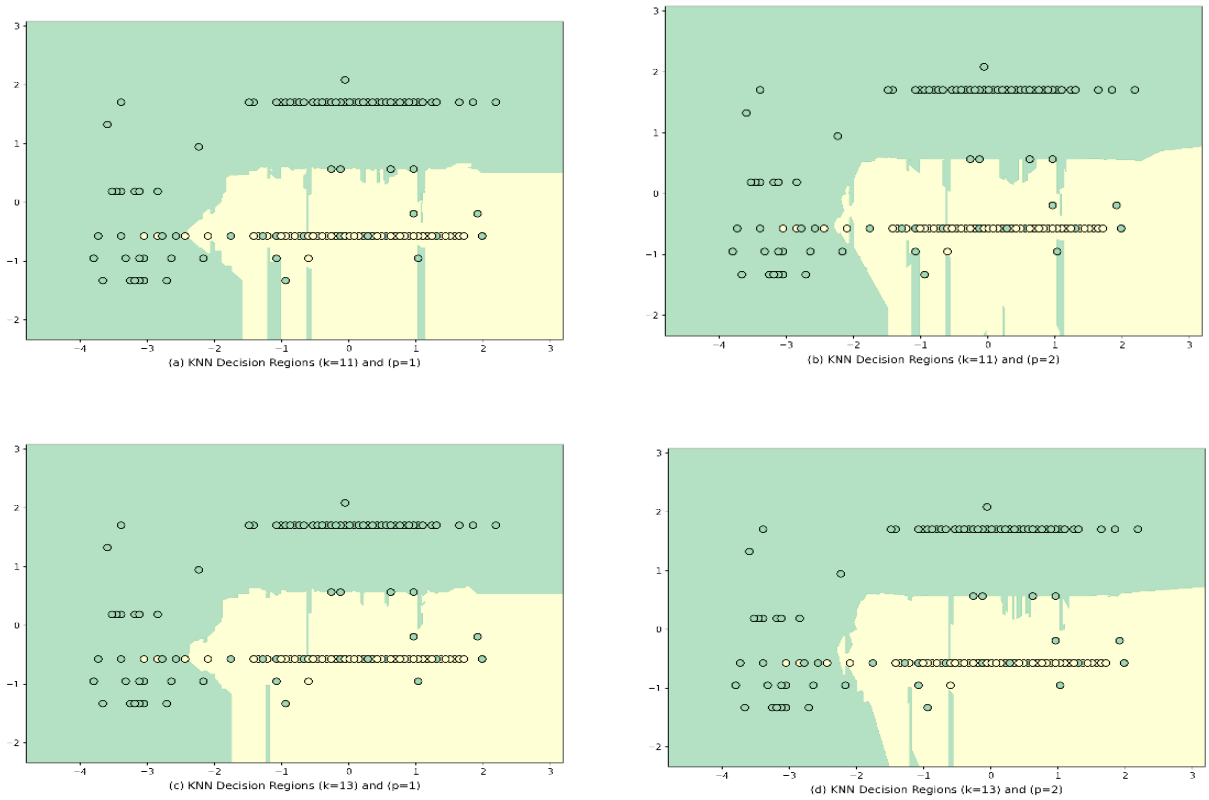


Fig. 3. Decision boundaries of KNN

approximately 85.80%. Consequently, for k=13 and p=1, the K-nearest neighbors classifier performs optimally.

VII. DECISION BOUNDARY

The decision boundary for a variety of k, it has an inversed effect on the classifier's flexibility. Test points that are adjacent to one other may readily discover points from both classes in their neighborhood when k is tiny, creating an inconsistent decision boundary that leads to overfitting (low bias but very high variance); Greater robustness is brought about by higher values of k; this means that the decision boundary is more robust to variations in the training dataset and produces a more regular decision boundary, which eventually leads to a straight line for extremely large k (low variance but high bias)[22].

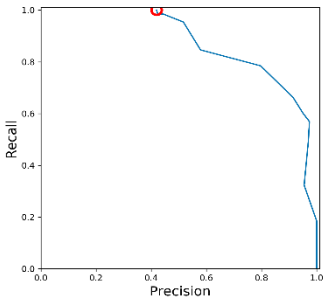
In this procedure, selecting the parameter K is highly important. Fig. 5 illustrates how the value" K" affects the choice limits. The two classes are divided by the following boundaries, which have different values of K. The decision boundary is less constrained when k = 11 has an accuracy score that is marginally less than 87.09% as shown in Fig. 5(a). It is advised to select a larger number for k because these are tied for the highest score. This results in a more regular decision boundary that eventually tends, for big k=13 as illustrated in Fig. 5(c). Fig. 5 shows that boosting the value of p from 1 to 2 when moving from left to right. A p = 1 results in less restrictive decision boundaries, as shown in Fig. (a), (b), whereas a p = 2 results in more restrictive decision boundaries, as shown in Fig.5(c), (d).

When p=1, it means that the classifier has a low complexity, whereas p=2 means that the classifier has a higher complexity. As a result, the classifier is susceptible to overfitting and cannot generalize to the validation set.

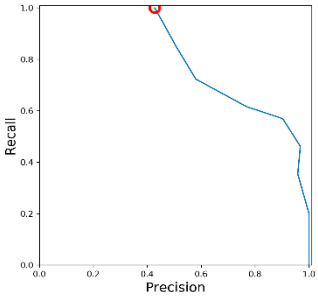
VIII. EVALUATION OF KNN CLASSIFIERS

A. Precision and recall curve

When a curve remains in the upper-right corner, it indicates that the classifier is perfect, with precision and recall both high at the exact same threshold. In generally, one of the most essential tasks is to calculate high precision and high recall for a class using the precise same threshold [23]. A further technique for finding a suitable precision/recall tradeoff is to plot precision versus recall. However, it is dependent on the type of fields [13].



(a) KNN with p=1



(b) KNN with p=2

Fig. 4. Precision-Recall Curves of KNNs

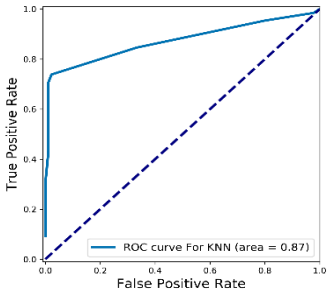
The precision-recall curve illustrates how classifiers naturally trade off recall and precision. Stated differently, do we identify every patient who is sick, and if we do, is the patient truly sick? Someone we miss may pass away from a disease left untreated. The resources would be burdened if we discovered that everyone was sick [24]. Fig. 6 shows the recall vs precision trade-off to find the classifier's optimal performance at the default threshold. As demonstrated in Fig. 6(a), the classifier has high precision and low recall, while it appears that classifier (b) is similar to classifier (a), as shown in Fig. 6(b). Both classifiers' recall has dropped. It implies that the classifiers who classified some patients are not required to follow patients, despite the fact that they should. The position denoted by the red circle corresponds to the decision function's default threshold. When the predict method is called, the precision-recall trade-off is chosen. This means we can get a recall of 1.00 while maintaining a precision of around 0.60.

A. Receiver Operating Characteristic

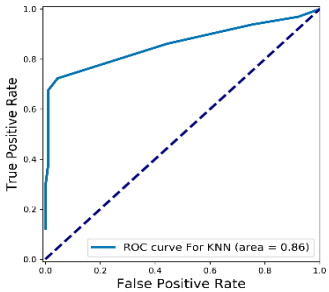
The most significant matrix for visualizing a binary classifier's performance is the ROC. The area under the curve (AUC) is a convenient technique for figuring out a binary classifier's performance in a given value [25], [26]. A great classifier would quickly ascend into the upper left corner of the plot with a true positive rate (TPR) of one and a false positive rate (FPR) of zero. The ROC (AUC) can provide information about the classifier's performance.[26].

The area under the classifier's ROC curve is depicted in Fig.7. We can compare two classifiers based on the ROC AUC (area). The ROC AUC score of a perfect classifier is one. We can observe that the KNN classifier with p=1 performs significantly better than the KNN classifier with p=2. We can observe in Fig. 7(a) that the KNN classifier with p=1 has the highest ROC AUC score of 0.87, which is the closest to 1. While the ROC AUC score for the KNN classifier with p=2 is 0.86, indicating that the classifier has a ROC AUC score that is distant from one, as seen in Fig. 7(b).

Table 2. The classifiers with their metric measures.



(a) ROC for the KNN with p=1



(b) ROC for the KNN with p=2

Fig. 5. ROCs for KNNs.

I. DISCUSSION AND RESULTS

Table 2 gathers the optimal results for each algorithm, including Accuracy, Precision, Recall, and F1-Score. The research findings and the literature review are compared in this study. The classifiers with their metric measures.

KNN performs superiorly to the other models, with an accuracy of 87.09%. KNN performs higher compared to some performance metrics of other related works, with comparable precision, recall, and F1-score of 97.87%, 70.76%, and 82.14%. As well, the study finds that Rahman[12] and KNN (p=2) accuracy scores are roughly comparable, at almost 85%. According to the study's Chattopadhyay, a classifier attained a less accurate accuracy range of 77.3%, which is thought to be the lowest accuracy in previous studies. The accuracy of the studies of Dahia and Rahman is higher than that of other previous studies, at 85.4% and 86.9%, respectively. In contrast, Table 1 demonstrates that the accuracy of Ngew et al.[9], Krstic[8], and Roje et al.[7] is higher than that of earlier studies, ranging from 79% to 83%.

All of our experiments had the highest precision when compared to previous studies, with a special case of Dahia's study, which has higher precision than the KNN (p=2) classifier. Chattopadhyay had the highest recall and the highest F1 score (77.3% accuracy, 77.3% precision, 100% recall, and 88% F1 score), although it was less accurate and precise than our experiments and other studies. Dahia et al.'s findings are more than sufficient. 86.9% F1-score,

87.7% precision, 86.9% recall, and 86.9% accuracy. All the recall values for the examinations carried out in this study and the results of other studies are acceptable when taking into account a specific case of two occurrences involving the KNN classifier and Ngew's study, which had recall values of 67.69% and 64.7% that were below the intended values. In the end, the KKN (p=1) classifier consistently outperforms the other models, with an accuracy of 87.09% and a precision of 97.87%.

Classifier	*Accuracy*	*precision*	*recall*	*F1-score*
Roje et al[7]	79%	80%	80%	80%
Krstic [8]	80%	80%	80%	-
Ngew et al.[9]	83.3%	84%	64.7%	-
Chattopadhyay[10]	77.3%	77.3%	100%	88%
Dahia[11]	86.9%	87.7%	86.9%	86.9%
Rahman[12]	85.4%	82.7%	83.3%	85.7%
KNN(p=1)	87.09%	97.87%	70.76%	82.14%
KNN(p=2)	85.80%	97.77%	67.69%	80.00%

II. CONCLUSION

KNN is a wonderful ML technique for learning about the importance of the data that we provide and constructing classifiers capable of predicting new data, such as diagnoses. The KNN is one of the ML algorithms that performs classification. In this work, KNN is utilized to develop two classifiers that determine whether a patient should be followed up on. The KNN(p=1) has a performance accuracy rate of 87.09%, a precision rate of 97.87%, a recall rate of 70.76%, and an F1-score rate of 82.14%. KNN(p=2) is the worst KNN, however recall percentage is paid more attention in the field of cardiac disease than other percentages. The poorest KNN is the KNN (p=2). As a result, the KNN (p=1) performs much better than the KNN(p=2).

I. REFERENCE

[1] K. Kwakye and E. Dadzie, “Machine Learning-Based Classification Algorithms for the Prediction of Coronary Heart Diseases,” Dec. 02, 2021, *arXiv*: arXiv:2112.01503. doi: 10.48550/arXiv.2112.01503.

[2] T. L. Ayalew, K. E. Haile, M. G. Feleke, B. T. Zewudie, and T. Y. Chichiabellu, “A systematic review and meta-analysis of cardiovascular diseases and associated factors among diabetes mellitus patients in Ethiopia,” *BMC Cardiovasc. Disord.*, vol. 23, no. 1, p. 413, Aug. 2023, doi: 10.1186/s12872-023-03443-0.

[3] S. F. Weng, J. Reps, J. Kai, J. M. Garibaldi, and N. Qureshi, “Can machine-learning improve cardiovascular risk prediction using routine clinical data?,” *PloS One*, vol. 12, no. 4, p. e0174944, 2017, doi: 10.1371/journal.pone.0174944.

[4] M. J. Gaikwad, P. S. Asole, and L. S. Bitla, “Effective Study of Machine Learning Algorithms for Heart Disease Prediction,” in *2022 2nd International Conference on Power Electronics & IoT Applications in Renewable Energy and its Control (PARC)*, Jan. 2022, pp. 1–6. doi: 10.1109/PARC52418.2022.9726613.

[5] H. Kutrani and S. Eltalhi, “Cardiac Catheterization Procedure Prediction Using Machine Learning and Data Mining Techniques,” vol. 21, pp. 86–92, Jan. 2019, doi: 10.9790/0661-2101018692.

[6] N. KUMAR, *Machine Learning for Beginner’s*. Niranjan Kumar, 2023.

[7] R. Jose, A. Thomas, J. Guo, R. Steinberg, and M. Toma, “Evaluating machine learning models for prediction of coronary artery disease,” *Glob. Transl. Med.*, vol. 3, p. 2669, Mar. 2024, doi: 10.36922/gtm.2669.

[8] M. Krstić and L. Krstić, “A logistic regression-based model for predicting heart failure mortality,” *J. Eng. Manag. Compet.*, vol. 15, pp. 57–64, Jan. 2025, doi: 10.5937/JEMC2501057K.

[9] K. Ngew, H. Tay, and ahmad khairuddin bin mohamed yusof, “Development and validation of a predictive models for predicting the cardiac events within one year for patients underwent percutaneous coronary intervention procedure at IJN,” *BMC Cardiovasc. Disord.*, vol. 23, Nov. 2023, doi: 10.1186/s12872-023-03536-w.

[10] S. Chattopadhyay, “MLMI: A Machine Learning Model for Estimating Risk of Myocardial Infarction,” *Artif. Intell. Evol.*, pp. 11–23, Jan. 2024, doi: 10.37256/aie.5120243714.

[11] S. Dahia and C. Szabo, “Implementing Machine Learning to Predict the 10-Year Risk of Cardiovascular Disease,” *Qeios*, vol. 5, Nov. 2023, doi: 10.32388/1SVUCI.2.

[12] S. Rahman, M. M. Hasan, and A. K. Sarkar, “Machine Learning and Deep Neural Network Techniques for Heart Disease Prediction,” in *2022 25th International Conference on Computer and Information Technology (ICCIT)*, Dec. 2022, pp. 1086–1091. doi: 10.1109/ICCIT57492.2022.10055902.
[13] A. Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow*. O’Reilly Media, Inc., 2022.
[14] S. Raschka, Y. Liu, V. Mirjalili, and D. Dzhulgakov, *Machine Learning with PyTorch and Scikit-Learn: Develop Machine Learning and Deep Learning Models with Python*. Packt Publishing, 2022.
[15] P. Gupta and N. K. Sehgal, *Introduction to Machine Learning in the Cloud with Python: Concepts and Practices*. Springer International Publishing, 2021.
[16] B. Johnston and I. Mathur, *Applied Supervised Learning with Python: Use Scikit-Learn to Build Predictive Models from Real-world Datasets and Prepare Yourself for the Future of Machine Learning*. Packt Publishing, 2019.
[17] T. Amr, *Hands-On Machine Learning with scikit-learn and Scientific Python Toolkits | Data | eBook*, Frist. Packt Publishing Ltd, 2020. Accessed: Sep. 24, 2025. [Online]. Available: https://www.packtpub.com/en-us/product/hands-on-machine-learning-with-scikit-learn-and-scientific-python-toolkits-9781838823580
[18] S. Raschka and V. Mirjalili, *Python Machine Learning: Perform Python Machine Learning and Deep Learning with Python, Scikit-learn, and TensorFlow*. Packt Publishing, 2017.
[19] A. Ng, *Machine Learning Yearning*. GitHub; eBook (Draft, 2018); eBook (MIT Licensed), 2018.
[20] S. Raschka and V. Mirjalili, *Python Machine Learning: Machine Learning and Deep Learning with Python, scikit-learn, and TensorFlow 2*. Packt Publishing, 2019.
[21] H. Tatsat, Puri, Lookabaugh Sahil, Brad, *Machine Learning and Data Science Blueprints for Finance*, Frist Edition. O’Reilly Media, Incorporated, 2020.
[22] A. Coluccia, *Adaptive Radar Detection: Model-Based, Data-Driven and Hybrid Approaches*. Artech House, 2022.
[23] M. Swamynathan, *Mastering Machine Learning with Python in Six Steps: A Practical Implementation Guide to Predictive Data Analytics Using Python*. Apress, 2019.
[24] R. Kumar and B. Auffarth, *Artificial Intelligence with Python Cookbook: Proven recipes for applying AI algorithms and deep learning techniques using TensorFlow 2.x and PyTorch 1.6*. Packt Publishing Ltd, 2020.
[25] W. Richert, *Building Machine Learning Systems with Python*. Packt Publishing Ltd, 2013.
[26] L. Buitinck *et al.*, “API design for machine learning software: experiences from the scikit-learn project,” Sep. 01, 2013, *arXiv*: arXiv:1309.0238. doi: 10.48550/arXiv.1309.0238.