2023 IEEE Symposium Series on Computational Intelligence (SSCI)
Mexico City, Mexico. December 5-8, 2023

A game theoretic based k-nearest neighbor approach for binary classification

Rodica Ioana Lung
Center for the Study of Complexity
Babeş-Bolyai University
Cluj-Napoca, Romania
Email: rodica.lung@econ.ubbcluj.ro

Mihai Alexandru Suciu
Center for the Study of Complexity
Babeş-Bolyai University
Cluj-Napoca, Romania
Email: mihai.suciu@ubbcluj.ro

***Abstract*—K-nearest neighbor is one of the simplest and most intuitive binary classification methods providing robust results on a wide range of data. However, classification results can be improved by using a decision method that is capable of assigning, if necessary, the minority label from the list of neighbors of a tested instance. In this paper, we propose using a simple game-theoretic model to assign labels based on the neighbors' information to enhance its performance for binary classification.**

***Index Terms*—kNN, game theory, Nash equilibrium, CMA-ES**

I. Introduction

While the field of machine learning is filled with a plethora of classification methods adjusted for various types of applications, most of them rely on a set of approaches that can be considered classic and that are combined, improved, and adapted in order to yield better results on a meaningful set of data.

In [1], a list of the top 10 algorithms in machine learning is presented, and most current approaches are based on one of the methods presented there. Among them, kNN, the k-nearest neighbor classification method [2] is listed as one of the most simple and elegant approaches. To classify an instance, the decision is made by looking at the most similar elements in the training data-set, making the decision based on some majority rule. Moreover, it has appealing theoretical properties; it is known that if we have $k/N \to 0$, for $N \to \infty$ where N is the total number of training instances, the error of kNN converges to the Bayes error almost surely. [3], [4].

There are three main issues that influence the efficiency of kNN: the number of neighboring nodes k, the similarity measure used to select the neighbors, and the rule that is used to make the decision based on the information provided by the neighbors [1].

In this paper, the use of a new decision rule based on a game theoretic approach is proposed. While the information provided by only k neighbors may not be enough to use a classification model, such as logistic regression or probit [5] successfully, the game theoretic model provides a method to capture interactions among neighboring instances having the same label to enhance the results of the decision process. We denote the kNN variant that employs this method kNN$-g$.

This work was supported by a grant of the Romanian Ministry of Education and Research, CNCS - UEFISCDI, project number PN-III-P4-ID-PCE-2020-2360, within PNCDI III.

II. kNN$-g$

Consider the binary classification problem expressed as follows: we are given a set of data X composed of N instances $x_i \in \mathbb{R}^p$ labeled by $y_i \in \{0,1\}, i = 1,\ldots,N$. $Y = \{y_i\}_{i=1,\ldots,N}$ is the set of labels. We need to find a model that, given a new instance x, assigns to it a label y in a "correct" manner. The "correctness" of labels assigned by the model is typically evaluated by testing some instances with known labels.

A. kNN

Standard kNN assigns class y to a test instance x based on information provided by its k neighbors by using a voting mechanism [6], [7]. If simple majority rule is used then:

$$y = \arg\max_{l} \sum_{(x_i,y_i)\in N_x} \mathbb{I}(y_i = l), \tag{1}$$

where N_x is the neighborhood of x, having k instances defined based on some similarity measure and $\mathbb{I}(\cdot)$ is an indicator function with value one if its argument is true and 0 otherwise. For real data, Euclidean distance is mostly used. The model actually consists in storing the training data for reference and computing N_x during the test phase. The efficiency of kNN is affected by the choice of k, the choice of similarity measure used to define the neighbors of an instance, and that of the voting mechanism.

Thus, some approaches to improve KNN study the optimal number of neighbors k [8]–[10]. Others try to adapt the similarity measure used to compute N_x to the specificity of the data [11], [12]. The approach presented in this paper replaces the voting mechanism with a game theoretic approach in order to predict a class by using only the local information provided by the k neighbors, without the need for feature weights or new parameters.

B. A binary classification game

The game theoretic approach used by kNN$-g$ is derived from the Framework for Optimization and game theory

(FROG) in [13]. Within kNN$-g$ a normal form game $\Gamma(N_x)$ with k players consisting of the instances of the neighborhood N_x is constructed. The strategies of the players consist of choosing their class. Thus the set of pure strategy profiles of the game is $S = \{0, 1\}^k$ and an element $s = (s_1, \ldots, s_k) \in S$ consists of the choices of each player, i.e. if $s_i = 0$ it means that player i has chosen label 0.

The payoff of each player depends on its choice and on the choices of all other players with the same label and is computed as:

$$u_i(s_1, \ldots, s_i, \ldots, s_k) = \left\{ \begin{array}{cc} 1, & s_i = y_i \\ \frac{2T(y_i)}{2T(y_i)+F(y_i)}, & s_i \neq y_i \end{array} \right. , \quad (2)$$

where $T(y_i)$ counts how many players having label y_i have chosen it, and $F(y_i)$ how many players with label y_i have chosen differently. The ratio in the payoff u_i is computed in a similar manner with the F_1 score but without taking into account the actions of the players with different labels in order to decentralize the search for an equilibrium.

Example 1: Consider $N_x = \{x_1, x_2, x_3\}$ with labels $0, 0, 1$. Game $\Gamma(N_x)$ has three players, and a possible situation of the game can be $s = (0, 1, 0)$ in which player 1 chooses label 0, player 2 chooses label 1 and player 3 chooses label 0. In this case, the payoff of player 1 will be equal to 1, as it chooses its correct label, while the payoff of player 2 will be:

$$u_2(s) = u_2(0, 1, 0) = \frac{2 \cdot 1}{2 \cdot 1 + 1} = \frac{2}{3}.$$

Player 3 will have a payoff of 0.

The Nash equilibrium (NE) strategy [14] of a game represents a situation in which no player can improve its payoff by unilateral deviation. The Nash equilibrium of game Γ corresponds to the correct classification of the training data, which, in the case of kNN$-g$ represent the k nodes of a neighborhood N_x.

By considering strategies in mixed form we can make the connection between game Γ and a probabilistic classification model. A mixed strategy profile consists of probability distributions over the strategy set of each player, i.e. the set of labels $\{0, 1\}$. Since we have only two possible values, it is enough to denote by σ_i the probability that a player chooses the label 1. Then a mixed strategy profile of the game $\sigma = (\sigma_1, \ldots, \sigma_k) \in [0, 1]^k$ is evaluated by considering the expected payoff for each player.

The probit classification model [5] uses the standard normal distribution to assign an instance the probability to have a label of 1 (or 0 respectively. The model parameter $\beta \in \mathbb{R}^p$ used to compute the probability:

$$P(y_i = 1|N_x) = \Phi(\beta x), i \in \{1, \ldots, k\} \quad (3)$$

is estimated by maximizing the log-likelihood function

$$\log L(\beta; X, Y) = \sum_{i=1}^{k} [y_i \log \Phi(\beta x_i) + (1 - y_i) \log(1 - \Phi(\beta x_i))], \quad (4)$$

where $\Phi(\cdot)$ represents the cumulative distribution function for the standard normal distribution, and βx represents the dot product of β and x.

If we consider

$$\sigma_i = \sigma_i(\beta) = P(y_i = 1|N_x) = \Phi(\beta x), i \in \{1, \ldots, k\},$$

then, by searching for the Nash equilibrium of Γ in mixed form we can find parameters β for the probit model that provide an approximation of the equilibrium which represents also the correct classification of the training data. In this manner, interactions among players with the same label are captured by the game and further included in the probit model.

To approximate the NE in mixed form the game can be converted into an optimization problem [15] by constructing a function $v(\sigma)$:

$$v(\sigma) = \sum_{i \in \mathcal{N}} (1 - u_i(\sigma))^2. \quad (5)$$

having as optima with value 0 the NEs of the game [13]. The optima of this function can be approximated by any optimization heuristic. In this approach, the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) is used [16].

CMA-ES is a search heuristic that evolves the mean and covariance matrix of a randomly generated population following a normal distribution, designed to solve nonlinear or nonconvex continuous optimization problems. It is highly adaptable and requires as parameters only a starting point for the search and initial value for the standard deviation of the population, which comes also with a recommended value. However, because the optimization function is based on the payoffs u_i in eq. (2), in some instances there are multiple β values that yield a minimum of 0 for the optimized function. Because of that, and also because of the stochastic nature of CMA-ES, the heuristic is run multiple times and the average of optimal β values is used for the prediction of x. Furthermore, since the optimum value of v is known to be 0, the search of CMA-ES is stopped when this value is reached by the fitness value.

The outline of kNN$-g$ is presented in Algorithm 1. For each tested value x the set N_x of neighbors is computed. If all instances in N_x have the same label, there is no need for further computations, that label is assigned to x. Otherwise, parameters β that approximate the NE of game $\Gamma(N_x)$ are computed by running CMA-ES multiple times. Since the values of k are relatively small, there is no computational challenge in running CMA-ES for this game. The m values of β resulting in m runs of CMA-ES are averaged and are used to compute the probability that x has label 1 based on the probit model. If the resulting probability is greater than 0.5 that the label 1 is assigned to x, otherwise it is 0. The probability is stored in order to compute the AUC measure during numerical experiments.

III. Numerical experiments

Numerical experiments are conducted on synthetic datasets generated with different numbers of instances, attributes, and

Algorithm 1 kNN$-g$ outline

```
1: Input: training set $X$ with labels $Y$ and a test instance $(x', y')$;
2: Compute $N_x$, the set of size $k$ of neighbors of $x$;
3: if all $x \in N_x$ have the same label $l$ then
4:     Assign $l$ to $x$, i.e. $y \leftarrow l$
5: else
6:     Find $\beta$ that minimizes $v(\sigma(\beta))$ by applying CMA-ES for $m$ times and averaging results;
7:     Assign to $x$ label $y = 1$ with probability
           $p = P(y = 1|N_x) = \Phi(\beta x)$
       in eq. (3);
8: end if
9: Output: label $y$, probability $p$.
```

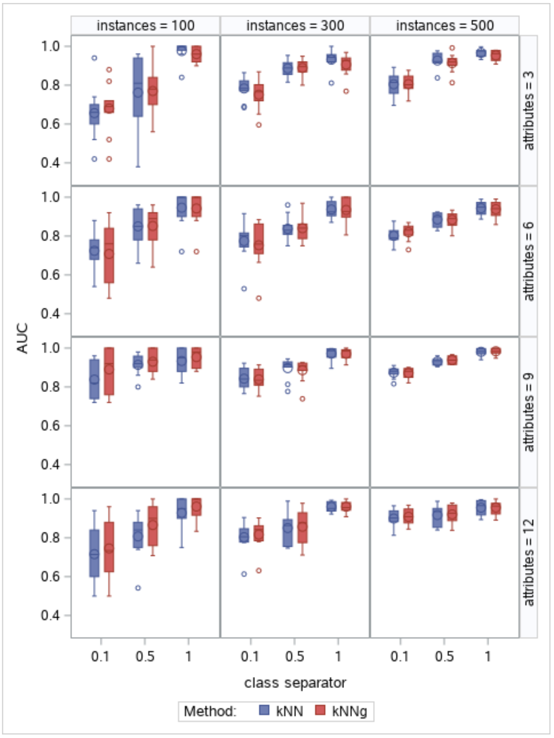

Fig. 1. AUC values for the 10 folds reported by kNN and kNN$-g$ on the synthetic data-sets with 3,6,9, and 12 attributes.

class separators [17], as well as on some real-world data-sets in order to compare the performance of kNN$-g$ with that of the baseline method kNN that uses the majority voting rule. Although better kNN variants may be considered, the purpose of this endeavor is to show the advantages of replacing the decision rule based only on the information provided by the neighbors.

A. Experimental set-up

a) Implementation: Experiments are performed by using the implementation of `KNeighborsClassifier` available in the `scikit-learn` package in `python` [18]. The majority voting rule is replaced with a function that implements the optimization of function v in eq. (5) by using CMA-ES. The CMA-ES `fmin` implementation in `python` [19] is used with default parameters and with the option to stop the search when the objective function reaches 0.

b) Performance measure: The performance of each method is evaluated by using 10-fold cross-validation [7]. For each tested fold the ROC AUC value [20] is reported. Higher values of AUC are considered better. The maximum of 1 is achieved when tested instances are correctly classified. Results reported by kNN and $k$NN$g$ on the 10 folds are compared by using a paired *t-test* with significance level 0.05.

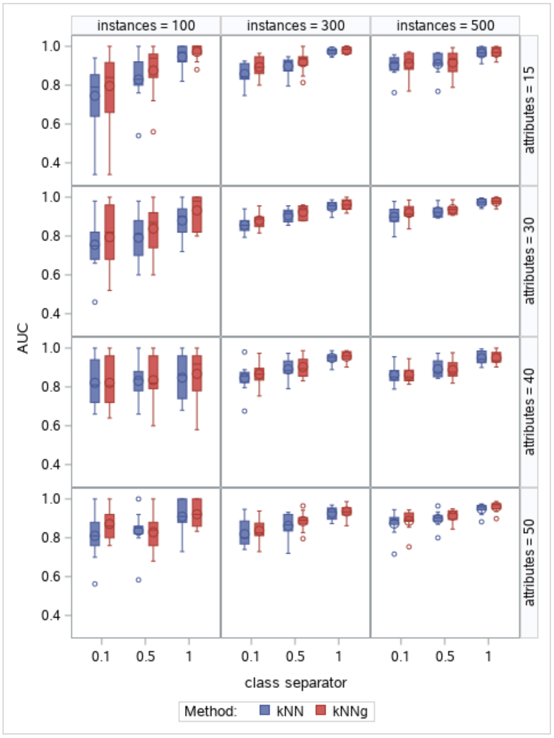

Fig. 2. AUC values for the 10 folds reported by kNN and kNN$-g$ on the synthetic data-sets with 15, 30, 40, and 50 attributes.

B. Synthetic data

Synthetic data-sets are generated by using the `make_classification` function in `scikit-learn` with different number of instances, attributes, and parameters. The tested number of instances are $100, 300,$ and 500, number of attributes are $3, 6, 9, 12, 15, 30, 40,$ and 50 and the class separator parameter was set to $0.1, 0.5,$ and 1, resulting in 72 data-sets of varying difficulty. Smaller values of the class separator parameter generate more difficult data sets with instances with different labels mixed with each other. All experiments on the synthetic data sets are performed with $k = 3$, in order to capture the behavior of kNN$-g$ when as little as possible information is available for the decision process.

Figures 1 and 2 present boxplots of AUC values reported for the 10 folds for each synthetic data-set, grouped by number

TABLE I
AUC values reported by the two methods for the synthetic data-sets for which kNN$-g$ performed significantly better and worse than kNN. Mean and standard deviation over the 10 folds as well as the p value for the paired $ttest$ rejecting the null hypothesis that kNN AUC values are greater than those reported by kNN$-g$.

Instances	Attributes	Class sep.	k	knNN mean ±	std	kNN$-g$ mean ±	std	p
100	9	0.1	3	0.84	0.1	0.89	0.11	0.0
100	12	0.5	3	0.81	0.12	0.87	0.11	0.02
100	12	1.0	3	0.93	0.09	0.96	0.06	0.03
100	15	0.1	3	0.75	0.19	0.8	0.2	0.0
100	15	0.5	3	0.83	0.13	0.88	0.14	0.01
100	15	1.0	3	0.95	0.07	0.97	0.04	0.05
100	30	0.5	3	0.79	0.11	0.84	0.13	0.0
100	30	1.0	3	0.88	0.08	0.93	0.09	0.0
100	50	0.1	3	0.81	0.13	0.87	0.08	0.02
300	15	0.1	3	0.86	0.05	0.89	0.06	0.01
300	15	0.5	3	0.9	0.05	0.92	0.05	0.0
300	15	1.0	3	0.97	0.02	0.98	0.02	0.0
300	30	0.1	3	0.86	0.04	0.88	0.04	0.0
300	30	0.5	3	0.91	0.04	0.92	0.04	0.0
300	30	1.0	3	0.95	0.03	0.96	0.03	0.01
300	40	0.1	3	0.85	0.08	0.87	0.06	0.03
300	40	0.5	3	0.89	0.05	0.9	0.05	0.04
300	50	0.5	3	0.86	0.07	0.89	0.05	0.03
500	6	0.1	3	0.8	0.04	0.82	0.04	0.03
500	9	0.5	3	0.93	0.02	0.94	0.02	0.02
500	9	1.0	3	0.98	0.02	0.98	0.02	0.02
500	12	0.1	3	0.9	0.05	0.91	0.04	0.02
500	15	0.1	3	0.9	0.06	0.91	0.06	0.02
500	30	0.1	3	0.9	0.05	0.92	0.04	0.0
500	30	0.5	3	0.93	0.03	0.94	0.03	0.0
500	30	1.0	3	0.97	0.02	0.98	0.02	0.0
500	50	0.1	3	0.87	0.06	0.89	0.05	0.03
500	50	0.5	3	0.9	0.04	0.91	0.03	0.01
500	50	1.0	3	0.95	0.03	0.96	0.03	0.0
300	3	0.1	3	0.78	0.06	0.75	0.08	1.0
300	3	1.0	3	0.93	0.05	0.91	0.06	1.0
500	3	0.5	3	0.93	0.04	0.91	0.05	0.97
500	3	1.0	3	0.97	0.02	0.95	0.03	0.98

TABLE II
AUC values reported by the two methods for the real data sets. Mean and standard deviation over the 10 folds as well as the p value for the paired $ttest$ rejecting the null hypothesis that kNN AUC values are greater than those reported by kNN$-g$.

Data set	k	knNN mean	± std	kNN$-g$ mean	± std	p
cryoptherapy	3	0.92	0.1	0.95	0.09	0.03 *
cryoptherapy	5	0.91	0.1	0.89	0.07	0.77
cryoptherapy	7	0.92	0.09	0.92	0.11	0.63
immunotherapy	3	0.48	0.21	0.47	0.13	0.56
immunotherapy	5	0.43	0.23	0.53	0.1	0.05*
immunotherapy	7	0.41	0.23	0.61	0.15	0.01*
banknote	3	1	0	1	0	10
banknote	5	1	0	1	0	10
banknote	7	1	0	1	0	10
plrx	3	0.5	0.2	0.55	0.19	0.05 *
plrx	5	0.49	0.2	0.46	0.18	0.8
plrx	7	0.46	0.15	0.47	0.19	0.42

of instances, attributes, and class separators. Out of the 72 tested data-sets, in 29 cases the performance of kNN$-g$ was significantly better than that of kNN. Table I presents the settings, and corresponding AUC and p values. In four out of the 72 tests, the performance of kNN was significantly better, according to the *ttest*, and these settings are presented in the last lines of Table I.

Results show that kNN$-g$ performs better for a higher number of attributes. Standard deviation values are similar, indicating similar behavior of the two methods.

C. Real-world datasets

The following datasets from the UCI Machine Learning repository [21] are presented here: the Cryotherapy Dataset, with 90 instances and 7 attributes [22]; the Immunotherapy Dataset Data Set with 90 instances and 7 attributes [22]; the banknote authentication Data Set with 1372 instances and 4 attributes [21]; and the Planning Relax Data Set (plrx) with 182 instances and 13 attributes [23].

The size of neighborhood k was set to $3, 5$ and 7. Results are presented in Table II. kNN$-g$ improves upon kNN for different values of k - marked with an * in the table - and reports similar results in the rest of the cases. The *banknote* data-set is reported to show that kNN$-g$ does not miss clear results reported by kNN, even if there are instances in which it reports worse results on some folds.

IV. Conclusions and further work

This paper presents a simple game theoretic replacement to the decision-making process of the standard kNN. Numerical results presented show that this approach can improve the performance of kNN, and it may be used for more advanced versions. While the improvement may not be considered spectacular, it shows how we can extract more information than by using simple majority voting, without adding any weights or other conditions, even from three neighbors. The approach can be further improved by considering different payoffs, underlying probabilistic classification models, and optimization methods.

Acknowledgment

This work was supported by a grant of the Romanian Ministry of Education and Research, CNCS - UEFISCDI, project number PN-III-P4-ID-PCE-2020-2360, within PNCDI III.

References

[1] X. Wu, V. Kumar, J. Ross Quinlan, J. Ghosh, Q. Yang, H. Motoda, G. J. McLachlan, A. Ng, B. Liu, P. S. Yu, Z.-H. Zhou, M. Steinbach, D. J. Hand, and D. Steinberg, "Top 10 algorithms in data mining," *Knowl. Inf. Syst.*, vol. 14, no. 1, p. 1–37, Dec. 2007. [Online]. Available: https://doi.org/10.1007/s10115-007-0114-2

[2] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, 1967.

[3] T. Wagner, "Convergence of the nearest neighbor rule," *IEEE Transactions on Information Theory*, vol. 17, no. 5, pp. 566–571, 1971.

[4] L. Rimanic, C. Renggli, B. Li, and C. Zhang, "On convergence of nearest neighbor classifiers over feature transformations," in *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., 2020.

[5] T. Liao, *Interpreting Probability Models*, Thousand Oaks, California, Oct 1994. [Online]. Available: https://methods.sagepub.com/book/interpreting-probability-models

[6] E. Fix, J. Hodges, and U. S. of Aviation Medicine, *Discriminatory Analysis: Nonparametric Discrimination, Consistency Properties*, ser. Discriminatory Analysis: Nonparametric Discrimination, Consistency Properties. USAF School of Aviation Medicine, 1985, no. vol. 1-2. [Online]. Available: https://books.google.ro/books?id=VN07ngEACAAJ

[7] T. Hastie, R. Tibshiran, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2016. [Online]. Available: https://web.stanford.edu/ hastie/ElemStatLearn/

[8] N. García-Pedrajas, J. A. Romero del Castillo, and G. Cerruela-García, "A proposal for local k values for k -nearest neighbor rule," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 2, pp. 470–475, 2017.

[9] S. Zhang, X. Li, M. Zong, X. Zhu, and R. Wang, "Efficient knn classification with different numbers of nearest neighbors," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 5, pp. 1774–1785, 2018.

[10] Z. Pan, Y. Wang, and Y. Pan, "A new locally adaptive k-nearest neighbor algorithm based on discrimination class," *Knowledge-Based Systems*, vol. 204, p. 106185, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0950705120304147

[11] M. Z. Jahromi, E. Parvinnia, and R. John, "A method of learning weighted similarity function to improve the performance of nearest neighbor," *Information Sciences*, vol. 179, no. 17, pp. 2964–2973, 2009, copulas, Measures and Integrals. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0020025509001832

[12] R. Paredes and E. Vidal, "Learning prototypes and distances: A prototype reduction technique based on nearest neighbor error minimization," *Pattern Recognition*, vol. 39, no. 2, pp. 180–188, 2006, part Special Issue: Complexity Reduction. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0031320305002232

[13] R. I. Lung and M.-A. Suciu, "Equilibrium in classification: A new game theoretic approach to supervised learning," in *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion*, ser. GECCO '20. New York, NY, USA: Association for Computing Machinery, 2020, p. 137–138. [Online]. Available: https://doi.org/10.1145/3377929.3390001

[14] M. Maschler, E. Solan, and S. Zamir, *Game Theory*. Cambridge University Press, 2013.

[15] R. D. McKelvey and A. McLennan, "Chapter 2 Computation of equilibria in finite games," ser. Handbook of Computational Economics. Elsevier, 1996, vol. 1, pp. 87 – 142.

[16] N. Hansen, S. Müller, and P. Koumoutsakos, "Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cma-es)," *Evolutionary Computation*, vol. 11, no. 1, pp. 1–18, 2003, cited By 1248.

[17] M. Scholz and T. Wimmer, "A comparison of classification methods across different data complexity scenarios and datasets," *Expert Systems with Applications*, vol. 168, p. 114217, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417420309428

[18] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[19] N. Hansen, yoshihikoueno, ARF1, K. Nozawa, M. Chan, Y. Akimoto, and D. Brockhoff, "Cma-es/pycma: r3.1.0," Jun. 2021. [Online]. Available: https://doi.org/10.5281/zenodo.5002422

[20] T. Fawcett, "An introduction to roc analysis," *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861–874, 2006, rOC Analysis in Pattern Recognition. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S016786550500303X

[21] D. Dua and C. Graff, "UCI machine learning repository," 2017. [Online]. Available: http://archive.ics.uci.edu/ml

[22] F. Khozeimeh, R. Alizadehsani, M. Roshanzamir, A. Khosravi, P. Layegh, and S. Nahavandi, "An expert system for selecting wart treatment method," *Computers in Biology and Medicine*, vol. 81, pp. 167–175, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S001048251730001X

[23] R. Bhatt, "Planning-relax dataset for automatic classification of eeg signals." [Online]. Available: http://archive.ics.uci.edu/ml