EVALUATION OF CRYOTHERAPY SUCCESS IN WART TREATMENT OF A MACHINE LEARNING ALGORITHM

Asst. Prof. Dr. Aykut Fatih Güven
Yalova University, Faculty of Engineering,
https://orcid.org/0000-0002-1071-9700
Electrical and Electronics Engineer Nermin Oruç
Yalova University, Faculty of Engineering
https://orcid.org/0009-0001-5013-9719
Electrical and Electronics Engineer İrem Salır
Yalova University, Faculty of Engineering
https://orcid.org/0009-0001-3787-8090

ABSTRACT

Warts caused by human papillomavirus (HPV) are common benign skin lesions that can appear on various parts of the body, most frequently on the hands and feet. Cryotherapy, a widely used treatment, involves applying extremely low temperatures to destroy the wart tissue through cold-induced cellular damage. Despite its effectiveness, the outcomes of cryotherapy can vary among patients. This study explored the application of machine learning (ML) models to predict the success of cryotherapy for wart cancer. Using a clinical dataset, five ML algorithms—K-Nearest Neighbors (KNN), Logistic Regression (LR), Naive Bayes (NB), Random Forest (RF), and Support Vector Machine (SVM)—were evaluated to determine their predictive accuracy. Among these models, the RF and SVM models demonstrated the highest predictive performance, significantly outperforming the other models, while the LR model exhibited the lowest accuracy. These findings highlight the potential of ML models, particularly RF and SVM, in providing robust, data-driven predictions of cryotherapy outcomes, paving the way for more personalized and efficient treatment strategies in clinical practice.

Keywords: Warts, Cryotherapy, Machine learning, Treatment outcome, Predictive modeling, Artificial intelligence, Healthcare.

2. INTRODUCTION

2.1. Machine Learning

Machine learning (ML) is a branch of computer science that allows machines to produce human-like solutions to problems or progressively reach solutions by utilizing various data and algorithms. Instead of relying on manually coded rules or algorithms to solve specific problems, ML adopts a data-driven learning approach. It is a crucial tool that enables models to be trained on a dataset and then used to predict new data (Kataria & Singh, 2013).

ML methods can be categorized into two main types: unsupervised learning and supervised learning. Unsupervised learning uses unlabeled data to discover structures and patterns in a dataset (Mahesh, 2020). For example, unsupervised learning can be applied to a customer dataset to identify customer groups with similar shopping habits.

In contrast, supervised learning uses labeled data to learn the relationship between inputs and outputs. In this approach, each input corresponds to a specific output. For instance, to predict the outcome of a treatment, inputs such as a patient's characteristics (age, gender, type of disease, etc.) and outputs like

the treatment result can be used. Supervised learning is commonly applied to classification and regression problems. In classification problems, outputs are usually categorical (e.g., spam or not spam), whereas in regression problems, outputs are continuous values (e.g., house prices).

ML algorithms are widely used in various healthcare fields to identify correct treatment. For example, they can be applied to diagnose diseases, develop treatment plans, select treatment procedures, and understand the recovery process. Although there are numerous studies in the literature on applying ML algorithms to treat certain skin diseases, very few have investigated the use of ML for wart treatment.

In this study, a model was developed using ML algorithms to predict whether warts can be treated with cryotherapy based on a dataset. The most commonly used classification methods are logistic regression (LR), Naive Bayes (NB), Support Vector Machine (SVM), Random Forest (RF), K-Nearest Neighbors (KNN), and Decision Trees (Sharma & Kumar, 2017). The methods employed in this study are KNN, LR, NB, RF, and SVM.

ML algorithms analyze patterns and relationships in a dataset to make data-driven predictions. Each algorithm offers different advantages depending on the target problem. For example, the KNN classifies a new data point based on its nearest neighbors, and the LR uses a probability-based approach to predict outcomes.

In this study, the performance of various ML algorithms was compared to identify the most effective method for treating warts. KNN provides a data-driven classification approach, whereas LR models probability distributions. NB offers a simple and fast classification framework, combines multiple decision trees to enhance prediction accuracy, and SVM constructs a decision boundary to classify data into two categories, identifying the points farthest from the boundary.

The goal of this analysis was to identify the algorithms that performed best on a specific dataset to determine the most effective method for wart treatment. This study provides an important example of how ML methods can be applied in medical practice.

2.2. Warts

Warts are caused by human papillomavirus (HPV), and more than 100 types of HPV have been identified (Al Aboud, & Nigam, 2024). They can appear individually or in groups on any part of the human body, although they are most commonly found on the hands and feet. The appearance of warts can vary depending on the type and location of the virus. Some warts cause hard and thickened bumps on the skin surface, whereas others may be flatter and softer. Moreover, the HPV may not always be visible and can remain dormant in epithelial cells. Several types of skin warts have been identified: common warts, plantar, and flat warts (Plasencia, 2000).

Nongenital skin warts are most commonly seen in children and adolescents, but they can also occur in adults. Warts can spread through direct or indirect contact (Al Aboud, & Nigam, 2024). Factors that increase the risk of warts include using communal showers, working in certain occupations involving handling meat, and having a weakened immune system. Additionally, if the surface of a wart is damaged,

it can become infected, allowing the virus to spread from one area to another. Flat warts are typically found on the backs of the hands and on the legs; they appear as slightly raised, small plaques that are skin-colored or light brown. Common warts form firm, rough, well-defined bumps on the skin, measuring between 1 and 5 mm in diameter (Rowson & Mahy, 1967). They are most frequently found on the backs of the hands. Plantar warts, also known as foot warts, appear on the soles of the feet or under the toes (Plasencia, 2000). Due to their thick and tough appearance, they can be mistaken for calluses. Plantar warts are more common in adolescents and young adults, possibly due to factors like an immature immune system, repeated micro-trauma of the feet during sports, the use of public showers, and excessive sweating (Lye & Miles1951). Walking barefoot can also cause plantar warts.

(a) (b)

Figure 1. (a) Plantar wart (source: https://www.medicabil.com/tr/yazi/ayak-tabanindaki-sigil-nasil-gecer, accessed on 30 May 2024) (b) Common wart (source: https://www.sigiltedavimerkezi.com/elde-sigil.html, accessed on 30 May 2024).

Warts are generally harmless and tend to resolve independently in individuals with healthy immune systems. However, many people seek treatment because the appearance of warts can negatively affect their self-esteem. Various treatment methods are available for warts, including topical caustic acid, cryotherapy, electrocautery, surgical removal, laser treatment, intralesional injection of bleomycin, and immunotherapy with antigens like Candida albicans (C. albicans), purified protein derivatives (PPD), and measles, mumps, and rubella (MMR) antigens (Gibbs et al., 2002). The effectiveness of these treatments varies from one patient to another, and some can be costly. Therefore, it is important to select an appropriate treatment method based on the patient's condition. Among the treatment options, cryotherapy and immunotherapy are frequently used to treat plantar and common warts (Khozeimeh et al., 2017).

2.3. Cryotherapy

The use of cold in medicine has been known since ancient times. In Ancient Greece, Persia, and the Roman Empire, cold remedies such as snow, mixtures of ice and water, and cold water were used to treat a wide range of diseases (Bouzigon et al., 2016). Cryotherapy refers to the local or systemic application of cold for therapeutic purposes (Olson & Stravino,1972). There are various cryotherapy application techniques, including the dipstick technique, spray technique, cryoprobe technique, and intralesional cryotherapy.

Cryotherapy is widely used in many fields. It has numerous benefits, including the relief of soft tissue injuries, postoperative muscle pain, sprains, and swelling; reduction of inflammation; treatment of warts; pain relief; muscle recovery; promotion of weight loss; reducing anxiety and depression; and more.

Cryotherapy is commonly used to treat nongenital warts. The procedure involves freezing and thawing the warts. Liquid nitrogen is commonly used in the treatment, and it is applied via spray or cryoprobe methods. To achieve successful results, treatment should be administered at intervals of three weeks. Although it is an affordable and accessible treatment method, some individuals may experience fear or discomfort during the procedure.

Additionally, cryotherapy can have side effects. Common side effects of cryotherapy include pain, blistering, hypopigmentation, or hyperpigmentation, especially in darker skin tones; tendon or nerve damage from aggressive treatment; and onychodystrophy following the treatment of periungual warts (Leman & Benton, 2000).

Figure 2. Cryotherapy application (source: https://www.gunceldermatoloji.com/tr/kriyoterapi-dondurmasoguk-tedavisi-nedir-nasil-uygulanir, accessed on 30 May 2024).

2.4. Literature Review

Akyol et al. (2018) utilized the same dataset as in this study and conducted a comparative analysis of two widely recognized wart treatment methods: cryotherapy and immunotherapy. The datasets were obtained from the University of California Irvine ML Repository and were collected from patients with plantar and common warts who visited the Dermatology Clinic at Ghaem Hospital in Mashhad, Iran, between January 2013 and February 2015.

The first dataset comprising patients treated with cryotherapy includes seven attributes: patient age, gender, wart type, and treatment duration. The second dataset collected from patients treated with immunotherapy comprises eight attributes, including response to treatment and recovery process.

This study aimed to evaluate the effectiveness of both treatment methods by analyzing the datasets and comparing their performance in reducing the incidence of warts and facilitating patient recovery. The results indicated that both methods were effective, with immunotherapy demonstrating more durable and sustained outcomes. These findings offer valuable insights for clinicians when determining the most appropriate treatment for wart removal (Akyol et al., 2018).

ML algorithms such as Fuzzy Rule, NB, and RF, were employed (Akyol et al., 2018).The best results were achieved with RF, yielding an accuracy of 96.43% and precision of 94.44%. NB and the fuzzy rule followed with lower but notable results.

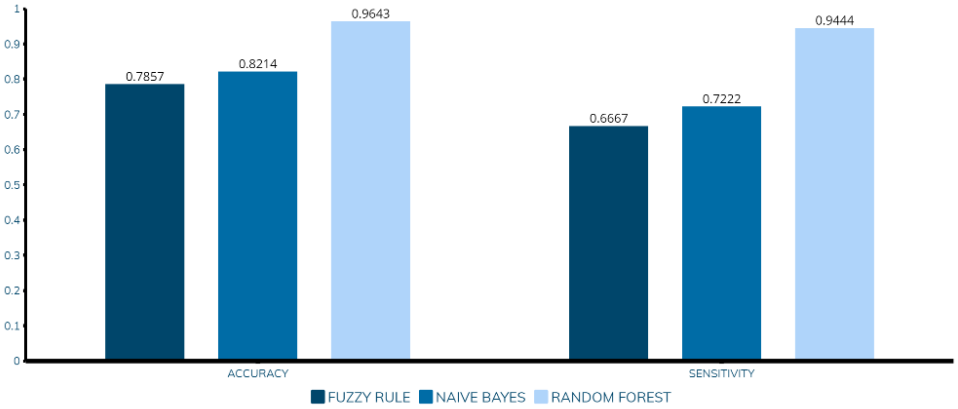

Figure 3. The accuracy and precision of the results reported by Akyol et al. (2018).

In another study, Rahman et al. (2019) developed a classification model to predict the effectiveness of cryotherapy in individual patients. developed a classification model to predict the effectiveness of cryotherapy in individual patients. They emphasized the importance of feature extraction in the data mining process, indicating that classification algorithms perform poorly if irrelevant features are included. Three feature selection methods were examined: wrapper, filter, and embedded methods.

The filter method, specifically, the Fuzzy Entropy and Mutual Information (FEMI) method, was used in this study to rank features. The performance of the classifiers was evaluated using 10-fold cross-validation, ensuring that the classifier was trained on only the training dataset (Yadav & Shukla, 2016). The results demonstrate consistency in the top-ranked features across all ten folds (Rahman et al., 2019).

Rahman et al. (2019) applied KNN, LR, NB, and SVM algorithms. The highest accuracy was observed with KNN 96.78%, while SVM achieved the highest precision 94.58%.

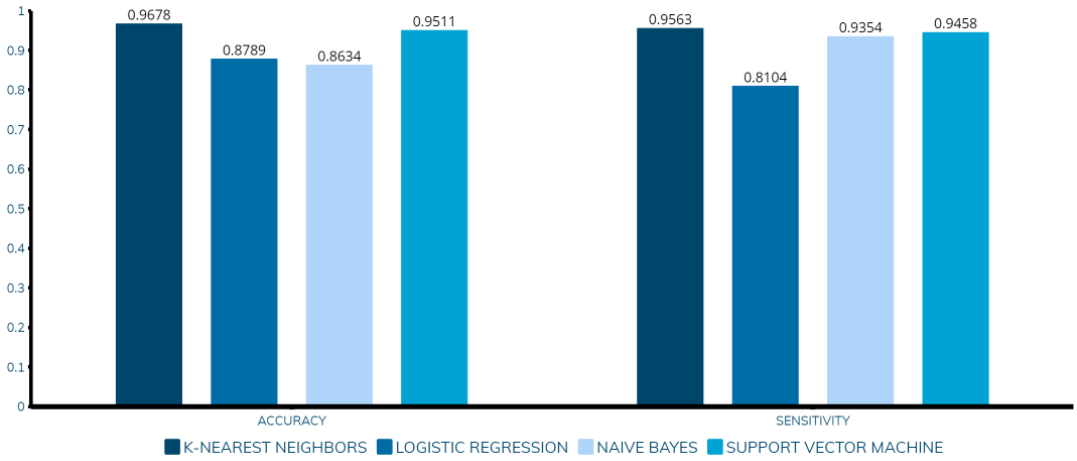

Figure 4. The accuracy and precision results were reported by Rahman et al. (2019).

ML is a critical tool for researchers working in data analysis across various fields (Das et al., 2015). In this study, the accuracy and precision rates of ML algorithms were examined. Accuracy refers to the proportion of instances correctly classified by a classifier, and precision reflects the proportion of true positives among predicted positives.

Akyol et al.'s study (2018) utilized KNN, LR, NB, and SVM. KNN achieved the highest accuracy (96.78%), followed by SVM (95.11%), LR (87.89%), and NB (86.34%). For precision, KNN obtained the highest value (95.63%), followed by SVM (94.58%), NB (93.54%), and LR (81.04%).

Rahman et al.'s study (2019) used NB and RF. RF demonstrated higher accuracy (96.43%) than NB (82.14%). Similarly, RF had a higher precision (94.44%) than NB (72.22%).

These results highlight the variability in performance of different ML algorithms when applied to the same datasets. When selecting an algorithm, factors, such as dataset structure, algorithm complexity, and performance metrics like accuracy and precision, should be considered. These findings can help researchers select the most appropriate algorithm for specific applications by carefully evaluating the characteristics of the dataset and the available algorithm options.

This study proceeds as follows: Section 2 delineates the methodology employed for the proposed system, followed by Section 3, which presents the results and discussion. The study concludes with the conclusions presented in Section 4.

3. Dataset

In this study, data were obtained from the UCI ML Repository. The dataset contains information from 90 different patients and consists of 7 columns. These columns provide details on various factors, such as sex, age, duration of warts, number of warts, type of wart, surface area, and treatment outcome.

This study focused on cryotherapy, an effective treatment for warts. The datasets were collected from patients with plantar and common warts who visited the Dermatology Clinic at Ghaem Hospital in Mashhad, Iran, between January 2013 and February 2015. These datasets include important factors, such as patient response to treatment and recovery process.

The first 10 rows of the 90-row dataset are shown below.

Table 1. First 10 rows of the dataset.

Sex	Age	Time	Number of Warts	Type	Area	Treatment Results
1	35	12	5	1	100	0
1	29	7	5	1	96	1
1	50	8	1	3	132	0
1	32	11.75	7	3	750	0
1	67	9.25	1	1	42	0
1	41	8	2	2	20	1
1	36	11	3	1	8	0
1	59	3.5	3	3	20	0
1	20	4.5	12	1	6	1
2	34	11.25	3	3	150	0

- Sex: The patients' gender is encoded as one for males and 2 for females.
- Age: This dataset includes 90 patients aged 15–67 years.
- Time: This column indicates how long the patients have had warts. The values range from 0 to 12 months.
- Number of Warts: This column shows the number of warts each patient had. The minimum number of warts was 1, and the maximum number was 12.
- Type: This dataset includes three types of warts. Type 1 represents common warts, Type 2 represents plantar warts, and Type 3 represents patients with both types. Out of the 90 patients, 54 had common warts, 9 had plantar warts, and 27 had both.

- Area: This column includes the surface area of the warts in mm². The smallest wart area is 4 mm², while the largest is 750 mm².
- Result of Treatment: This column represents treatment success. A value of 1 indicates successful treatment, and 0 indicates failure. In this cryotherapy study, 48 and 42 patients were successfully treated, while 42 treatments were unsuccessful.

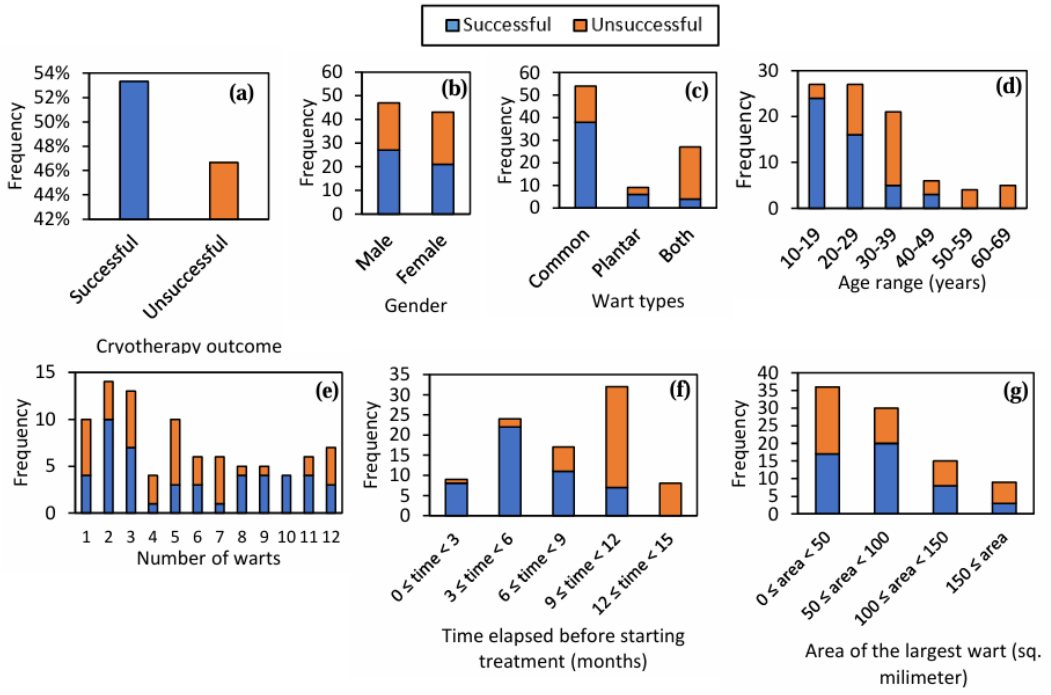

Figure 5. Visualization of the dataset.

Figure 5 visualizes the dataset. The distribution of treatment outcomes showed that 53.3% of treatments were successful and 46.7% were unsuccessful, indicating a balanced dataset. The success rate of cryotherapy was higher in male patients (57.4%) than in female patients (48.8%). However, the treatment was less effective in patients >30 years of age, those with both types of warts (plantar and common), and those who started treatment >9 months after the onset of warts. No clear trends or patterns were observed for the remaining characteristics.

2.1. Data Normalization

Initially, the data were normalized using min-max normalization. Min-max normalization is a crucial preprocessing step for effective ML. The original data is transformed into values between the minimum and maximum, specifically between 0 and 1 (Jain & Bhandare, 2011).

In ML and data analysis, preparing and preprocessing datasets is essential. Min-max normalization is one of these preprocessing steps, and it allows for the scaling of data values. This technique transforms the data into a specific range, typically [0, 1], ensuring that all values fall within the given range.

The formula for min-max normalization is as follows:

$$X_{normalized} = \frac{X - X_{min}}{X_{max} - X_{min}} \quad (1)$$

In Equation (1), $X_{normalized}$ represents the normalized data value, which is obtained after applying the min-max normalization process. The variable X denotes the original data value from the dataset, while X_{min} and X_{max} correspond to the minimum and maximum values in the dataset, respectively. These parameters are crucial for transforming the data values into a scaled range, typically between 0 and 1, and ensure that all features are on the same scale without altering their relative distribution.

This method is widely used when there are differences in the scales of the values in a dataset because it enhances model performance and accelerates the training process. However, this may lead to inaccurate results if there are outliers or data distribution is imbalanced.

Having different scales for values in the dataset can cause some features to have more influence than others, which negatively impacts model training and performance. Min-max normalization helps ensure that all features are within the same range, which allows for more balanced model training. This normalization compresses the data values into a specific range without changing their distribution. As a result, the training process can be sped up, and the algorithm can generate solutions faster.

Not applying min-max normalization can adversely affect model performance. Differences in feature scales can cause the model to be overly sensitive to certain features, potentially leading to incorrect results. In addition, the training process may take longer, and the model convergence may be reduced.

Min-max normalization balances the dataset, resulting in more accurate and reliable model outcomes. It also accelerates the training process, thereby allowing the model to be trained faster. Therefore, min-max normalization is a common practice in ML and data analysis tasks.

Table 2 shows the first 10 rows of the 90-row dataset. As seen in the table, the data were normalized to values between 0 and 1, which ensures consistency across all features for improved performance of the ML models.

Table 2. Dataset normalized between 0 and 1.

Sex	Age	Time	Number of Warts	Type	Area
0	0.384615	1	0.363636	0	0.12868632
0	0.26923	0.574468	0.363636	0	0.12332439
0	0.673076	0.659574	0	1	0.17158176
0	0.326923	0.978723	0.545454	1	1
0	1	0.765957	0	0	0.05093833
0	0	0.659574	0.090909	0	0.02144772
0	0.403846	0.914893	0.090909	0	0.00536193
0	0.846153	0.276595	0.181818	1	0.02144772
0	0.096153	0.361702	1	0	0.00268096
1	0.365384	0.93617	0.181818	1	0.19571045

2.2. Splitting Data into Training and Testing Sets

The concept of train-test refers to splitting the data into two groups: training and testing. In this phase, the holdout method is used, where a portion of the data is used to train the model, while the remaining part is reserved for testing. This method provides insights into how the model performs on real-world data (Li & Zhao, 2010).

The training dataset is a subset of the data used during the model's learning process. The model analyzes these data, identifies patterns, and uses these patterns to make predictions. The training dataset typically comprises the majority of the original dataset and plays a key role in model development.

The testing dataset was used to evaluate how well the model generalized the patterns learned during training. After training the model, its performance was measured on a testing dataset. This helps determine the performance of the model on unseen real-life data.

In this study, the dataset was divided into training and testing sets. During this process, several experiments were conducted to determine the most effective train-test ratio for obtaining optimal results. The results indicate that the best performance was achieved with a 67% training and 33% testing split.

Other ratios used in the experiments included 90% train - 10% test, 80% train - 20% test, 60% train - 40% test, and 50% train - 50% test. However, the highest performance was observed at the 67% training and 33% test ratios.

For the dataset comprising 90 patients, this ratio means that 60 patients (67%) were used for training, while 30 patients (33%) patients were reserved for testing. This balance between training and testing sets ensures that the patterns learned during training can be effectively generalized to the testing set, which results in successful model outcomes.

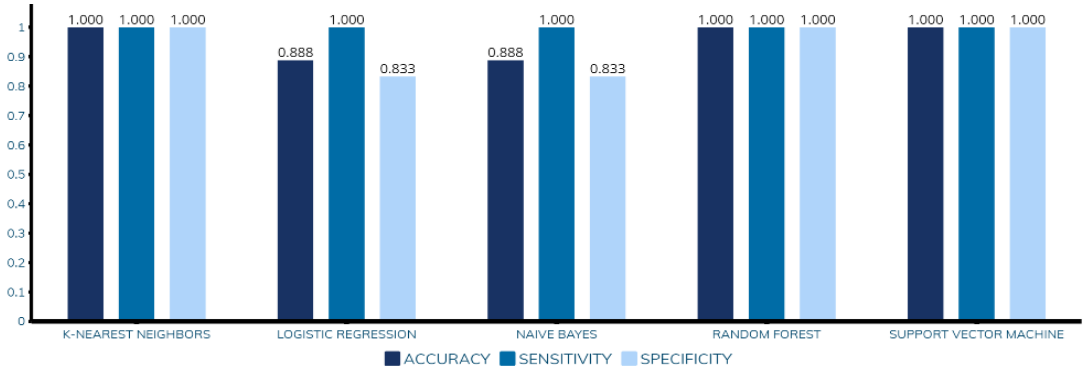

Figure 6. Results obtained when 10% of the data is used for testing and 90% for training.

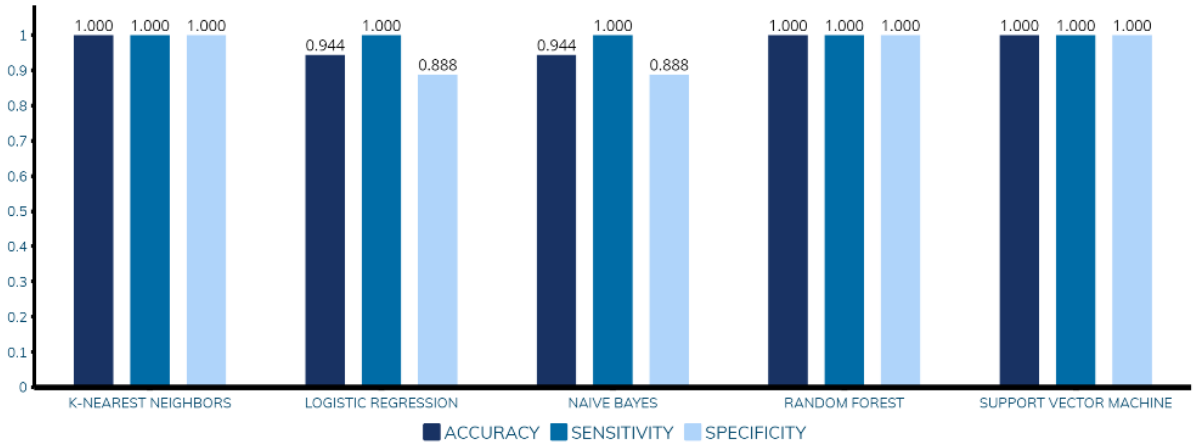

Figure 7. Results obtained when 20% of the data is used for testing and 80% for training.

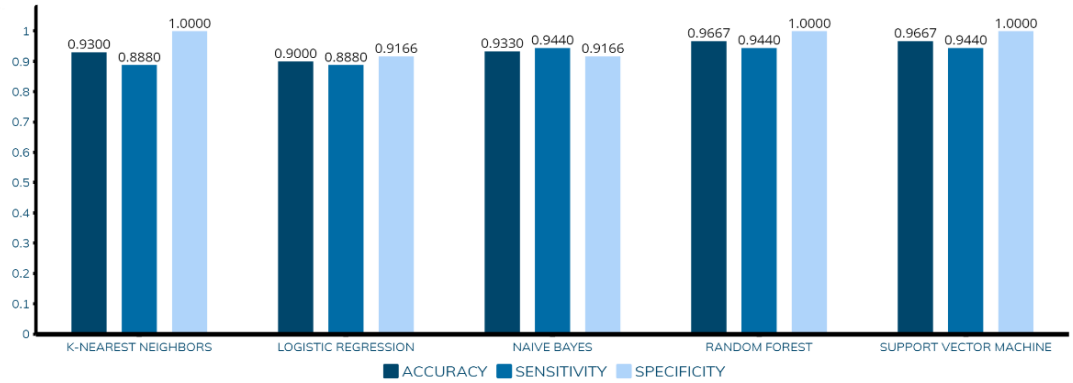

Figure 8. Results obtained when 33% of the data is used for testing and 67% for training.

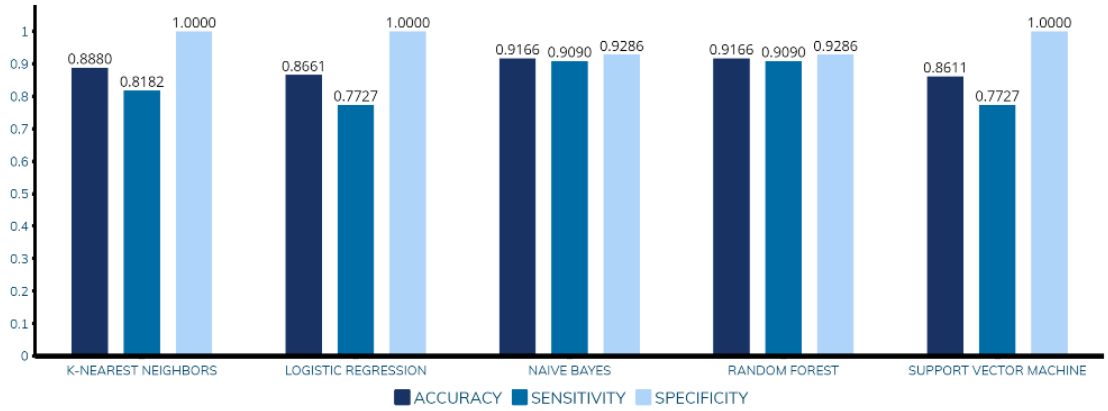

Figure 9. Results obtained when 40% of the data is used for testing and 60% for training.

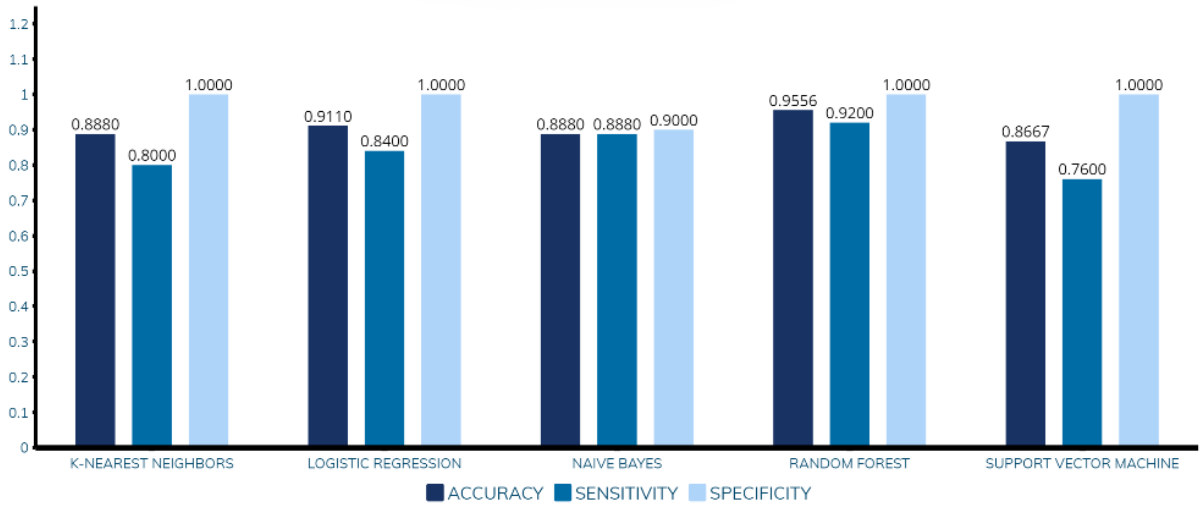

Figure 10. Results obtained when 50% of the data is used for testing and 50% for training.

2.3. Tables Generated from the Data and Their Interpretations

2.3.1. Feature Dataset (X Table)

The X table contains the dataset of patient features used for training and testing the ML model. This includes all patients with a total of 90 rows, where each row represents a single patient. Features include the patient's gender, age, duration of wart presence, number of warts, wart type, and size. These features help the model predict treatment outcomes for patients.

Table 3. Feature dataset (X Table)

Sex	Age	Time	Number of Warts	Type	Area
0	0.384615	1	0.363636	0	0.12868632
0	0.26923	0.574468	0.363636	0	0.12332439
0	0.673076	0.659574	0	1	0.17158176
0	0.326923	0.978723	0.545454	1	1
0	1	0.765957	0	0	0.05093833
0	0	0.659574	0.090909	0	0.02144772
0	0.403846	0.914893	0.090909	0	0.00536193

0	0.846153	0.276595	0.181818	1	0.02144772
0	0.096153	0.361702	1	0	0.00268096
1	0.365384	0.93617	0.181818	1	0.19571045

2.3.2 Outcome Dataset (Y Table)

The Y table contains the patient treatment outcome dataset. This includes all patients with a total of 90 rows, where each row represents the treatment outcome of a patient. Outcomes are encoded as 1 (treated) and 0 (not treated). The model learns from these outputs and uses them to predict whether a patient is being treated.

Table 4. Outcome dataset (Y Table)

y
0
1
0
0
0
1
0
0
1
0

2.3.3 Training Feature Subset (x_train Table)

The x_train table is a subset of the X table used for model training. It comprises 67% of the dataset, which corresponds to 60 randomly selected rows. This dataset was used during the model learning process. The model was trained on x_train to predict treatment outcomes based on features such as gender, age, duration of wart presence, number of warts, wart type, and size. In model training, x_train represents a portion of the X table, and the learning algorithm adjusts the model's internal parameters (weights and coefficients) using this data.

Table 5. Training Feature Subset (x_train Table)

Index	Sex	Age	Time	Number of Warts	Type	Area
51	0	0.01923	0.851063	0.181818	0.5	0.075067
55	0	0.365384	0.893617	0	1	0.121983
65	1	0.5	0.638297	0.363636	0.5	0.021447
89	0	0.076923	0.659574	0.636363	0	0.209115
88	0	0.173076	0.893617	0.818181	0	0.021447
67	0	1	0.297872	0.90909	1	0.021447

70	1	0.115384	0.893617	0.545454	0	0.041554
24	0	0.288461	0.191489	0.090909	0	0.148793
42	0	0	0.276595	0.090909	0	0
83	0	0.480769	0.957446	0.727272	0.5	0.101876

2.3.4 Testing Feature Subset (x_test Table)

The x_test table comprises the remaining 33% of the X table, which equals 30 rows. This dataset was used to evaluate model performance. The model uses the knowledge gained during training to predict the treatment outcomes of patients in the x_test set. This dataset is crucial for assessing the model's performance on unseen data, representing the portion of data not used during training. The x_test dataset helps evaluate the model's generalizability by testing its response to previously unseen data.

Table 6. Testing feature subset (x_test Table)

Index	Sex	Age	Time	Number of Warts	Type	Area
76	0	0.153846	0.851063	0.545454	1	0.091152
46	1	0.153846	0.787234	0.363636	1	0.091152
59	1	0.076923	0.659574	0.727272	0	0.209115
57	1	0.384615	0.702127	0.454545	1	0.128686
48	0	0	0.319148	0.545454	0	0.00268
69	0	0.365384	0.93617	0	1	0.19571
8	0	0.096153	0.361702	1	0	0.00268
14	1	0.038614	0.914893	0.090909	0	0.008042
9	1	0.365384	0.93617	0.181818	0	0.19571
73	1	0	0.148936	0.818181	1	0.088471

2.3.5 Training Outcome Subset (y_train Table)

The y_train table is a subset of the Y table used for model training. It comprises 67% of the dataset, corresponding to 60 randomly selected rows. This dataset was used during the model learning process. The model was trained on y_train to learn the actual treatment outcomes. Typically, y_train represents a portion of the Y table, and this dataset is used to teach the model the target variables (treatment outcomes).

Table 7. Training outcome subset (y_train Table)

y
1
0
1
1
1

0
0
1
1
0

2.3.6 Testing Outcome Subset (y_test Table)

The y_test table comprises the remaining 33% of the Y table. This dataset contains the actual treatment outcomes of patients in the x_test dataset. The model's predictions (Y_pred) were compared to the true values in y_test to evaluate performance.

Table 8. Testing outcome subset (y_test Table)

y
0
0
1
0
1
0
1
0
0
1

2.3.7 Predicted Outcomes (y_pred Table)

The y_pred table contains predictions made by the model on the x_test dataset. The model predicts treatment outcomes for patients in the x_test based on features. These predictions were then used to assess the model's performance by comparing it with the true treatment outcomes (y_test).

Table 9. Predicted Outcomes (y_pred Table)

y
0
0
1
0
1
0
1
0
0
1

4. METHODOLOGY

In this study, data from patients undergoing wart treatment were analyzed. Five different ML algorithms, commonly used in the field of ML analysis were applied to this dataset. The data were divided into two groups: training and testing sets with the purpose of training the model and evaluating its performance, respectively (Mahesh, 2020). The training data (train) comprise the dataset used for constructing and training the ML model, while the testing data (test) are utilized to evaluate the accuracy of the model's predictions.

During the training phase, the machine was presented with the x_train data, which included patient features, and the y_train data, which contained treatment outcomes. The patient features represent the input variables used to understand the influence of different characteristics on treatment outcomes, while the treatment results represent the output variables used to evaluate and improve the model's predictions during training.

Once the training process had been completed, the model was tested for accuracy using the x_test data. The machine was tasked with predicting treatment outcomes based on these input features, which were labeled as y_pred. Subsequently, the actual treatment outcomes were compared against the predicted results to assess the model performance.

The primary objective of this study was to compare the machine's predictions with actual treatment outcomes and evaluate the performance of the applied algorithms based on the analysis of these comparisons. This approach allows assessment of the efficacy and accuracy of ML algorithms in the context of wart treatment, providing valuable insights into their potential application in medical treatment prediction.

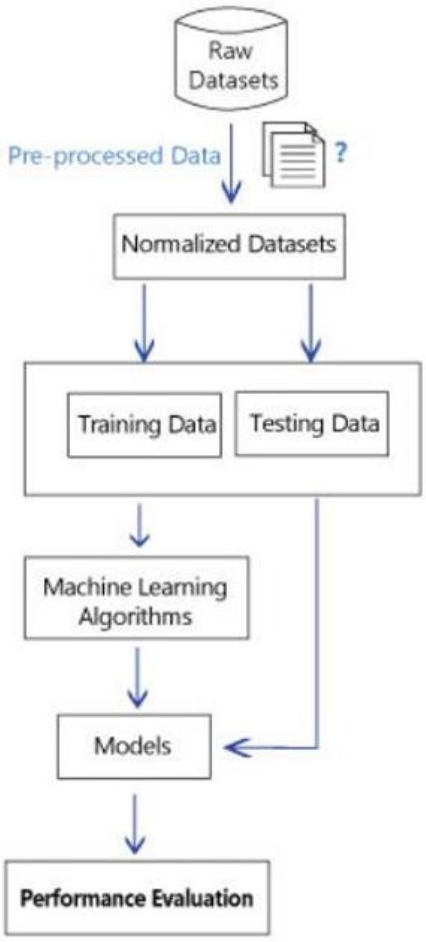

Figure 11. Optimization flowchart.

3.1 Optimization Techniques

In ML, optimizing model performance is critical to ensuring accurate predictions and efficient computation. Optimization techniques aim to minimize or maximize a particular objective function, typically by adjusting the parameters of the model. The algorithms used in this study rely on various optimization techniques to improve prediction accuracy and computational efficiency. These methods also help prevent issues such as overfitting and underfitting, which can severely affect model performance (Güven, 2024a).

The selection of the optimization technique depends on the nature of the problem, the data structure, and the algorithm used. Common techniques include gradient-based methods, evolutionary algorithms, and search-based methods, each of which has its strengths and weaknesses depending on the specific case. By employing these techniques, we aim to find the optimal set of model parameters that best fits the given data.

In this study, five supervised learning algorithms were employed, each benefiting from certain optimization strategies to enhance their performance on the wart treatment dataset. These algorithms are briefly introduced below along with the mathematical formulations and optimization strategies used (Güven, 2024b).

3.1.1 K-Nearest Neighbor Algorithm

The fundamental principle of the KNN method is to classify a new sample by finding its closest neighbors and assigning it to the class most common among these neighbors (Abu Alfeilat et al., 2019). The basic working principle of KNN is described below:

Assuming there are m samples in the dataset:

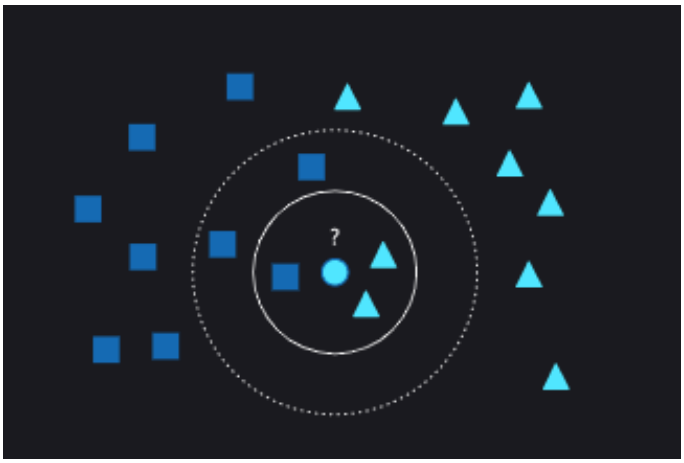

Figure 12. Optimization flowchart of the KNN algorithm (source: https://datakapital.com/blog/makine-ogrenimi-machine-learning-nedir/, accessed on 30 May 2024).

$$D=\{(x1,y1),(x2,y2),...,(xm,ym)\} \quad (2)$$

In this context, xi represents the feature vectors of the samples, while yi denotes the class labels corresponding to these samples. When a new sample (yi) is introduced, the KNN algorithm proceeds in a series of steps as follows:

- Distance Measurement: The distance between the new sample and all other samples in the dataset was calculated. Typically, this distance is measured using a metric such as the Euclidean distance, which is defined as follows:

$$d(x_{yeni}, x_i) = \sqrt{\sum_{j=1}^{n}(x_{yeni,j} - x_{i,j})^2} \quad (3)$$

Here, n represents the number of features, while $x_{yeni,j}$ and $x_{i,j}$ correspond to the j-th feature of the newand existing samplese, respectively.

- Selection of Neighbors: Based on the calculated distances, the k closest samples were selected. Typically, k is an odd number determined by the user.
- Classification: The classes of selected k neighbors are examined, and the majority class is assigned to the new sample.

The KNN algorithm relies on the n_neighbors parameter, which defines the number of neighbors to be considered for classifying or predicting a data point. The parameter n_neighbors determines the extent to which the algorithm generalizes and its sensitivity to specific training data.

In this study, parameter n_neighbors was set to 3. This selection means that the model will either classify or predict based on the labels of the three nearest neighbors. One reason for choosing 3 is that it is typically the default value in KNN algorithms and serves as a good starting point. Additionally, selecting a smaller value simplifies the model, reducing its complexity and shortening both the training and prediction times. The selection of this parameter is problem-specific and depends on the given dataset. For larger datasets or more complex classification problems, experimentation with different n_neighbor values is essential to achieve optimal performance.

In general, the KNN algorithm builds a data-driven classification or regression model, where the n_neighbors parameter is a key factor that influences the complexity and performance of the model. The choice of this parameter significantly affects the model's accuracy, generalizability, and computation time. Therefore, selecting an appropriate 'n_neighbors' value is critical for the success of this study.

```
# KNN Implementation
print("KNN") start = time.time()
# Importing KNeighborsClassifier from sklearn
from sklearn.neighbors import KNeighborsClassifier
# Initializing the KNN classifier with 3 neighbors
KNN = KNeighborsClassifier(n_neighbors=3)
 # Fitting the model to the training data
KNN.fit(x_train, y_train)
# Predicting outcomes using the test set prediction
KNN = KNN.predict(x_test)
 # Outputting accuracy of KNN model on test set
print("KNN accuracy is", KNN.score(x_test, y_test) * 100)
```

Figure 13. KNN code.

3.1.2 Logistic Regression

LR makes predictions by multiplying the input features by a weight set. This result is then passed through an activation function, which is commonly known as the sigmoid function (Bewick et al., 2005). The sigmoid function transforms the input (the sum of the weighted features) to a value between 0 and 1, which can be interpreted as a probability.

IV

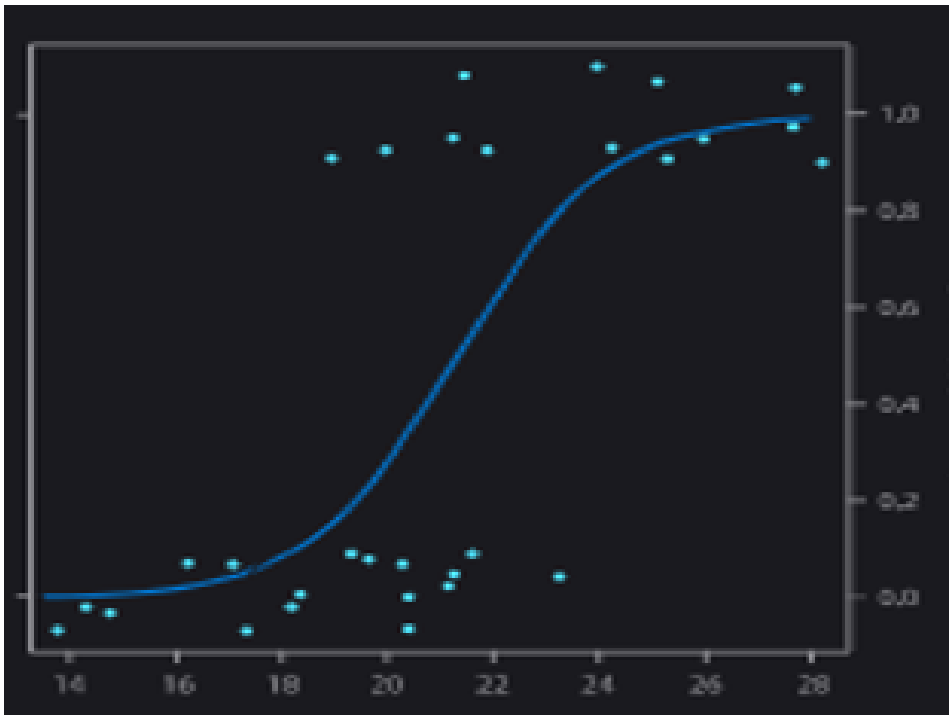

Figure 14. LR Algorithm (source: https://www.mql5.com/tr/articles/10626, accessed on 30 May 2024).

The LR model is typically expressed as follows:

$$P(y=1|x)=\frac{1}{e^{-\left(b_0+b_{1.}x_1+b_{2.}x_2\ldots.+b_n.x_n\right)}} \quad (4)$$

In this formula, $P(y=1|x)$ represents the probability that the class of a given instance is 1, and x denotes the input features. The terms b_0, b_1, b_2 correspond to the model's coefficients, which are also known as weights.

After training the model, it is used to predict the classes of new data points. The model takes the input features, applies the weights and parameters learned during training, and generates a probability value.

3.1.3 Naive Bayes

The proposed method is based on the NB theorem; thus, it is a probability-based classification technique. The algorithm calculates the probabilities for all possible outcomes and classifies the data based on the outcome with the highest probability (Nuthan, & Rohith, 2021). The probability of a class given a specific feature is calculated based solely on the probability of that feature, independent of the presence of other features.

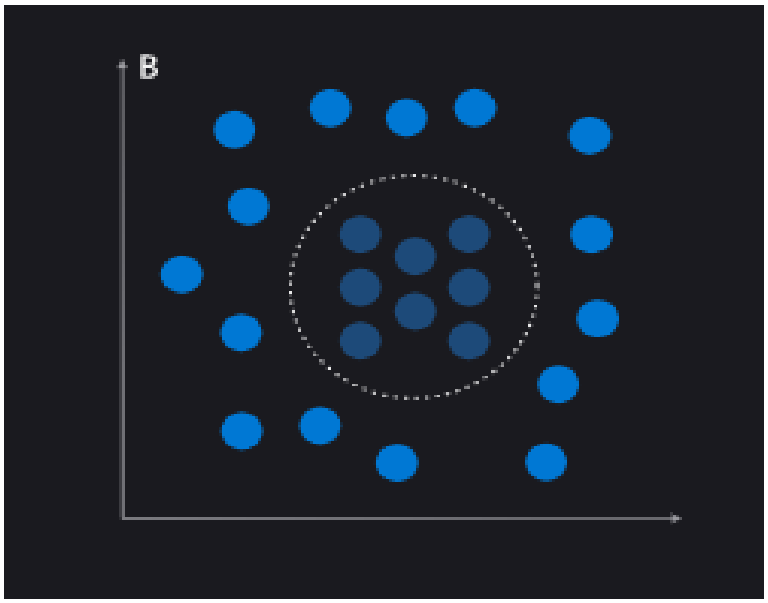

Figure 15. NB Algorithm (https://www.mql5.com/en/articles/12184, accessed on 30.05.2024).

$$P(C|X) = \frac{P(C|X)*P(C)}{P(X)} \quad (5)$$

In this equation, P(C|X) represents the probability that a data point belongs to class C, given the features X. $P(X|C)$ denotes the likelihood of observing the features X given that the data point belongs to class C. $P(C)$ refers to the prior probability of class C, while $P(X)$ represents the overall probability of observing the features X.

3.1.4 Random Forest

The RF algorithm is based on the idea of combining multiple decision trees that operate independently to make collective predictions, with the final prediction being the one with the highest aggregated score (Breiman, 2001). The working principle of the RF is as follows:

- Building Decision Trees: In RF, each decision tree is trained on a random subset of features and samples from the dataset. This allows each tree to view the data from a different perspective.
- Bagging (Bootstrap Aggregating): Each decision tree was trained on a randomly sampled subset of the dataset. This ensures that the trees remain independent of each other and are trained on different subsets of the data.
- Predictions: To predict the class of a data point, each decision tree makes individual predictions. The predictions are then averaged to make a collective prediction.
- Combining Results: The final prediction is obtained by aggregating the predictions of all decision trees. This is a crucial step because the combination of multiple predictions leads to more stable and accurate results.

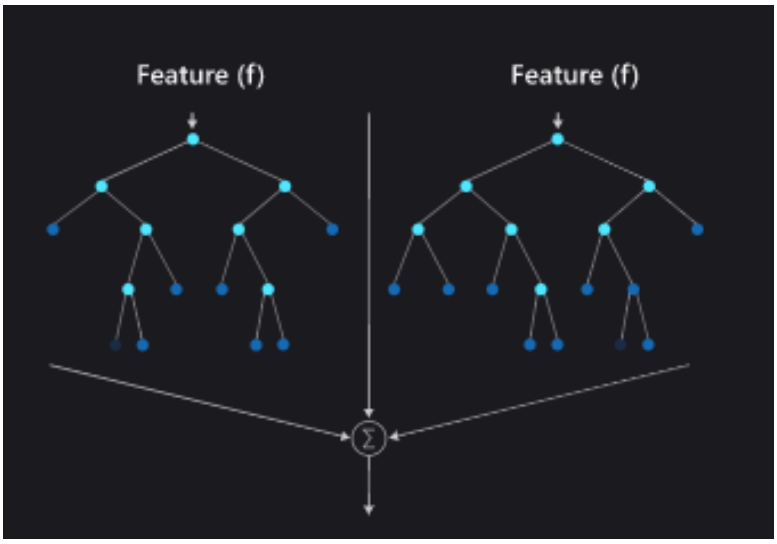

Figure 16. RF Algorithm (https://images.app.goo.gl/W7PoVnUvWU7shjjB7, accessed on 30.05.2024)

The RF algorithm creates a model by combining many decision trees, which is the reason it is referred to as a "forest." Each tree was trained on a subset of the dataset and combined to produce the final model.

The RF algorithm contains two important parameters that determine model complexity: n_estimators and random_state. The n_estimators parameter defines the number of trees in the forest. A larger number of trees make the model more complex and less prone to overfitting. In contrast, random_state controls the randomness of the model. This parameter ensures reproducibility by allowing the same results to be obtained each time the model is run with the same value.

The n_estimators parameter controls the number of trees in the forest. A higher number of trees increases the complexity of the model, allowing it to better adapt to training data. However, too many trees may lead to overfitting, where the model becomes too tailored to the training data, thereby reducing its ability to generalize and perform well on new data.

The random_state parameter controls the randomness by setting a seed. Randomness plays a key role in the RF algorithm because it is used during the random sampling of data and feature selection for tree construction. However, when we seek to obtain the same results across different runs, the random_state parameter ensures consistency. By using the same seed value, we obtain the same results in each run.

The selection of these parameters can significantly affect model performance and should be carefully selected. Optimal parameter values can be determined through trial and error or using cross-validation techniques.

```
# Random Forest Implementation
print("\nRANDOM FOREST")
start = time.time()
# Importing RandomForestClassifier from sklearn
from sklearn.ensemble import RandomForestClassifier
# Initializing RandomForestClassifier with 9 estimators and a random seed
randomf = RandomForestClassifier(n_estimators=9, random_state=1)
# Fitting the model to the training data
```

Figure 17. RF code.

3.1.5 Support Vector Machine

SVMs are used to classify data points by finding a hyperplane that best separates the classes, specifically aiming to identify the hyperplane that provides the maximum margin between classes (Cortes & Vapnik, 1995). The primary goal of SVM is to maximize the margin between classes, thereby ensuring the best possible separation. The margin represents the distance between the hyperplane and the closest data points from both classes.

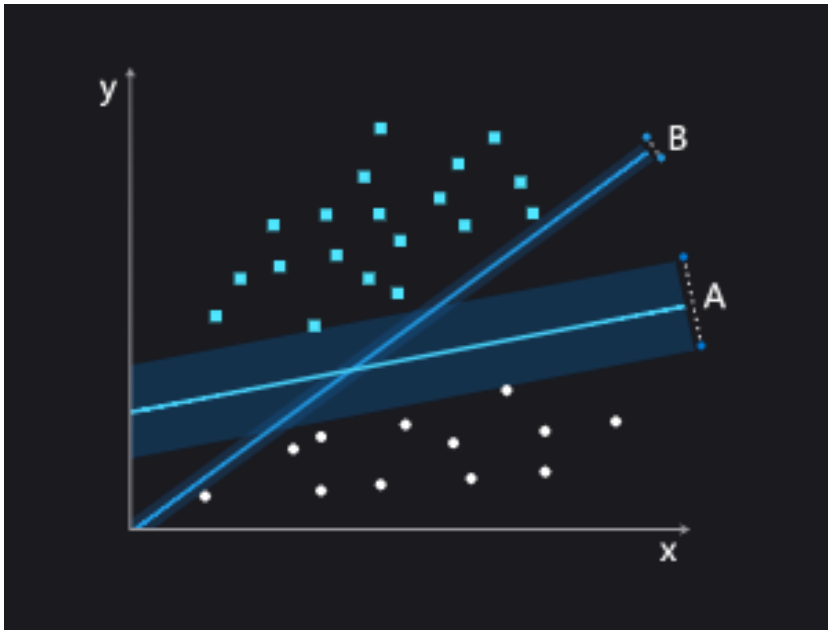

Figure 18. SVM algorithm (https://images.app.goo.gl/YqTuHzgsXRq9zV386, accessed on 30.05.2024).

The working principle of SVM comprises the following steps:

- Separating Data Points: The SVM separates data points by using a hyperplane (a decision boundary). It attempts to find a hyperplane that separates the two classes with the widest possible margin (Cortes & Vapnik, 1995). The position of this hyperplane is chosen such that it provides the maximum separation between the classes.
- Maximizing the Margin: The goal of SVM is to maximize the margin between classes. The margin is defined as the distance between the nearest data points in each class. SVM was used to find the hyperplane that maximizes this margin.
- Finding Support Vectors: SVM uses support vectors to determine the margin between classes. The support vectors are the data points closest to the hyperplane, and they play a key role in defining the margin.

A data point is assigned to a class based on which side of the decision boundary (hyperplane) it lies.

3.2 Confusion Matrix

The confusion matrix is a fundamental metric used to evaluate classification algorithm performance. The proposed model systematically visualizes and analyzes the relationship between actual and predicted classes of a model.

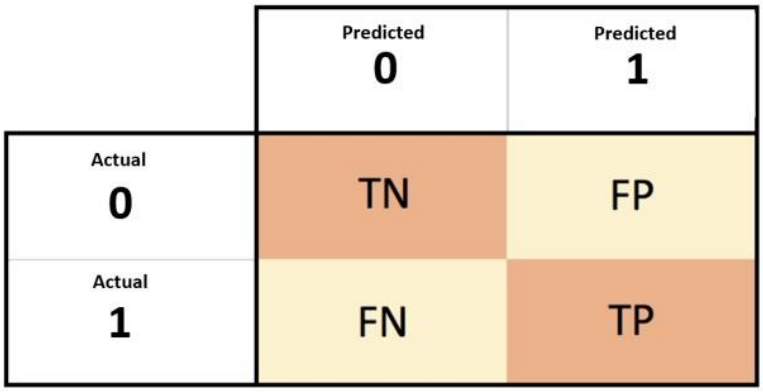

Figure 19. Confusion matrix.

- True Positive (TP): This refers to cases in which the model correctly predicted the positive class. In our study, represents the number of patients for whom the model predicted a successful treatment outcome and who, in reality, experienced a successful treatment outcome.
- False Positive (FP): This refers to cases in which the model incorrectly predicted the positive class. In our study, represents the number of patients for whom the model predicted a successful treatment outcome but who, in reality, did not have a successful treatment outcome.
- True Negative (TN): refers to cases in which the model correctly predicted the negative class. In our study, represents the number of patients for whom the model predicted an unsuccessful treatment outcome and who, in reality, experienced an unsuccessful treatment outcome.
- False Negative (FN): This refers to cases in which the model incorrectly predicted the negative class. In our study, represents the number of patients for whom the model predicted an unsuccessful treatment outcome but who actually experienced a successful treatment outcome.

The confusion matrices obtained in this study are summarized as follows.

Table 10. Confusion matrix results.

Algorithm	True Negative (0,0)	False Positive (0,1)	False Negative (1,0)	True Positive (1,1)
KNN	12	0	2	16
LR	11	1	2	16
NB	11	1	1	17
RF	12	0	1	17
SVM	12	0	1	17

3.2.1 Metrics Calculated from the Confusion Matrix

The confusion matrix provides a comprehensive overview of the model's performance by detailing the relationship between actual and predicted classes. From this matrix, several key performance metrics can be derived to help evaluate the accuracy, sensitivity, and specificity of the model. These metrics are essential to understand how well the model distinguishes between classes, particularly when dealing with classification problems with imbalanced data. The following sections describe these metrics in detail.

3.2.1.1 Accuracy

Accuracy represents the proportion of correctly predicted instances by the model. The accuracy of the classification model is calculated as the ratio of correctly classified instances (true positives and true

negatives) to the total number of instances. It is computed by dividing the sum of true positive (TP) and true negative (TN) predictions by the total number of predictions. Mathematically, it is expressed as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (6)$$

3.2.1.2 Sensitivity

Sensitivity, which is also referred to as recall, measures the proportion of actual positive instances that the model correctly identifies. Here, represents the ratio of true positive predictions to the total number of actual positive instances. Mathematically, it can be expressed as follows:

$$Sensitivity = \frac{TP}{TP + FP} \quad (7)$$

3.2.1.3 Specificity

Specificity measures the model's ability to correctly classify negative instances. Here, represents the ratio of true negative predictions to the total number of actual negative instances. Mathematically, it can be expressed as follows:

$$Specificity = \frac{TN}{TN+FP} \quad (8)$$

Table 11. Accuracy, sensitivity, and specificity values obtained from calculations.

	KNN	LR	NB	RF	SVM
Accuracy	0.9333	0.9	0.9333	0.9667	0.9667
Sensitivity	0.888	0.888	0.9444	0.9444	0.9444
Specificity	1	0.9166	0.9166	1	1

Table 12. Comparison of our results with previous studies.

	Results			Reference 1			Reference 2		
	Accuracy	Sensitivity	Specificity	Accuracy	Sensitivity	Specificity	Accuracy	Sensitivity	Specificity
KNN	0.933	0.888	1	0.9678	0.9563	0.981	-	-	-
LR	0.9	0.888	0.9166	0.8789	0.8104	0.9572	-	-	-
NB	0.933	0.944	0.9166	0.8634	0.9354	0.7809	0.8214	0.722	1
RF	0.9667	0.944	1	-	-	-	0.9643	0.944	1
SVM	0.9667	0.944	1	0.9511	0.9458	0.9572	-	-	-

In this study, five different algorithms were used to measure the accuracy, sensitivity, and specificity metrics: KNN, LR, NB, RF, and SVM. The accuracy values obtained for these methods were 93.33%, 90%, 93.33%, 96.67%, and 96.67%, respectively. These results were derived using a confusion matrix (Table 12). According to the accuracy values, the SVM and RF methods outperformed the other algorithms. In terms of sensitivity and specificity, the NB, RF, and SVM methods yielded the best results. The sensitivity of the NB, RF, and SVM methods was 94.44%, whereas the specificity of the

KNN, RF, and SVM methods was 100%. As a result of all these calculations, the RF and SVM methods produced the best overall results, with accuracy, sensitivity, and specificity values of 96.67%, 94.44%, and 100%, respectively (Table 11). Additionally, the results were compared with two other studies in the literature (Table 12).

In the first study, the following four algorithms were examined: KNN, LR, NB, and SVM. The accuracy values of these methods were 96.78%, 87.89%, 86.34%, and 95.11%, respectively. The sensitivity values were 95.63%, 81.04%, 93.54%, and 94.58%, respectively, with the highest specificity being attributed to the LR and SVM methods 95.72%. In the second study, only the NB and RF algorithms were considered, with accuracy values of 82.14% and 96.43%, respectively. Based on the sensitivity values, the RF method outperformed NB, as RF had a sensitivity of 94.44%, whereas NB had a sensitivity of 72.22%. The specificity of both methods was 100%.

From these comparisons, it was observed that in the first study, the KNN algorithm had the highest accuracy and sensitivity, whereas in the second study, the RF algorithm produced the best results. In this study, however, the RF and SVM methods outperformed the other algorithms with accuracy, sensitivity, and specificity values of 96.67%, 94.44%, and 100%, respectively. Notably, the accuracy of the proposed KNN algorithm was 3.45% higher than that of the previous study. On the other hand, the accuracy of the RF algorithm in the second study was 0.23% lower than that in this study.

3.3 K-Fold Cross Validation

In ML projects, various validation methods are used to evaluate model performance and measure generalizability. One such method is K-fold cross-validation. K-fold cross-validation divides the dataset into K equal parts, where each part is used as the test set in turn, while the remaining K-1 parts are used as the training set. This process was repeated K times, and the overall accuracy of the model was calculated by averaging the accuracy values obtained in each iteration (Yadav & Shukla, 2016).

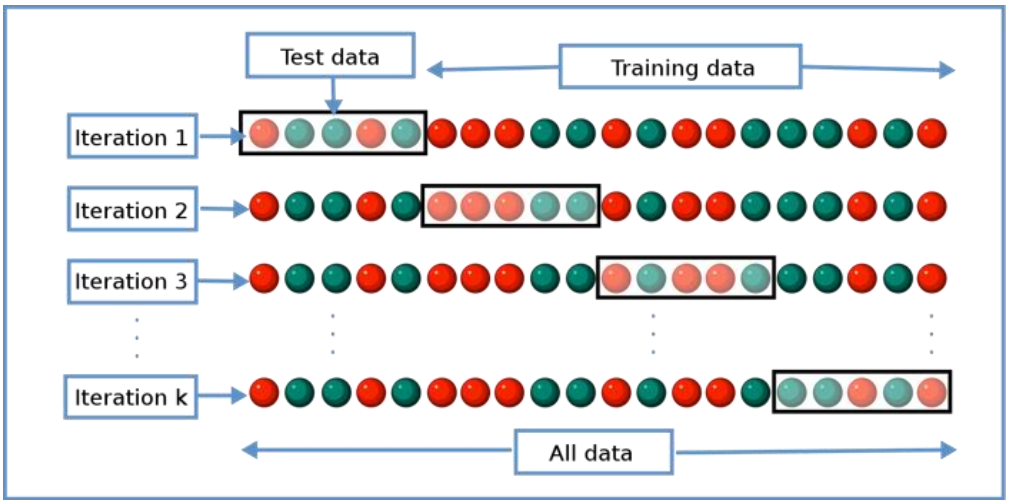

Figure 20. Illustration of K-Fold Cross-Validation.

The K-fold cross-validation method was used to evaluate model performance and reduce the risk of overfitting. This technique allows the entire dataset to be utilized and helps assess how the model performs on different subsets of the data, as well as whether high performance is random. K-fold cross-validation divides the data into different subsets, where the number of folds is defined by the user. In this study, the number of folds was set to 5. The data were divided into 5 parts, and algorithms were applied to each part in turn. A single part was used as the test set, while the remaining parts served as the training set.

The key steps of the K-fold cross-validation are as follows:

- Data Splitting: The dataset was split into K equal parts. In this project, it was split into 5 equal parts.
- Model Training and Testing: For K iterations, each part was used as the test set, and the remaining K-1 parts were used as the training set.
- Performance Measurement: In each iteration, the model is trained and tested, and the accuracy value is recorded.
- Average accuracy: The overall accuracy of the model was calculated by averaging the accuracy values obtained in all iterations (Kohavi, 1995).

Mathematically, it can be expressed as follows:

$$CV_{(K)} = \frac{1}{K}\sum_{i=1}^{k} Accuracy_i \quad (9)$$

In this formula, $CV_{(K)}$ represents the average accuracy obtained from K-fold cross-validation, where K is the number of folds into which the dataset is split, and $Accuracy_i$ refers to the accuracy calculated for the i-th fold.

K-fold cross-validation was used to test the generalizability of the model. This method helps us understand how the model performs on different subsets of the data and reduces the risk of overfitting. Overfitting occurs when the model performs very well on training data but poorly on new data. K-fold cross-validation ensured that the model demonstrated consistent performance on both training and test data (Wong, 2015).

The average accuracy of each iteration calculated during the K-fold cross-validation process is referred to as the K-fold accuracy. The average represents the overall accuracy of the model and indicates how successful it is across different subsets of the data (Refaeilzadeh et al., 2009).

In this study, we used K-fold cross-validation to obtain a more reliable measure of model accuracy. By using this method, we analyzed the performance of our model on different parts of the dataset and calculated the average accuracy. As a result of this process, the accuracy of the proposed model was tested on different parts of the dataset, and the average accuracy value was calculated.

These results demonstrate that the proposed model exhibited consistent performance across different data subsets, demonstrating strong generalizability. The K-fold cross-validation method reduced the risk of overfitting and provided a more reliable performance evaluation of the model (Xu & Goodacre, 2018).

Table 13. K-Fold Cross-Validation accuracy calculation.

Algorithm	1st Iteration	2nd Iteration	3rd Iteration	4th Iteration	5th Iteration	K-Fold Cross-Validation Accuracy
KNN	0.9444	0.888	0.9444	0.9444	0.888	**0.9222**
LR	0.833	0.833	0.778	0.9444	0.778	**0.8333**
NB	0.833	0.833	0.833	0.888	0.833	**0.8444**
RF	0.888	0.833	1.0	0.9444	0.9444	**0.9222**
SVM	0.9444	0.833	0.888	0.9444	0.833	**0.8889**

Using the K-Fold Cross-Validation method, the accuracy values of five different ML algorithms were tested over five iterations. The accuracy of each iteration was calculated, and the average of these values was taken to determine the K-Fold Cross-Validation Accuracy.

For the KNN algorithm, the accuracy values across the five iterations were as follows: 94.44% for the 1st iteration, 88.8% for the 2nd, 94.44% for the 3rd, 94.44% for the 4th, and 88.8% for the 5th. By

averaging these values, the K-fold cross-validation accuracy of the KNN algorithm was calculated as 92.22%. The same approach was applied to the other algorithms, yielding K-Fold Cross-Validation Accuracy values of 83.33% for LR, 84.44% for NB, 92.22% for RF, and 88.89% for SVM.

These results provide valuable insights into each algorithm's performance across different data subsets and their overall generalization capabilities. The KNN and RF algorithms demonstrated the highest performance, with a K-Fold Cross-Validation Accuracy of 92.22%. This suggests that both algorithms are highly consistent in their performance, not only on training data but also on previously unseen data, indicating their strong generalization ability.

The LR and NB algorithms achieved accuracy values of 83.33% and 84.44%, respectively. Although these models performed adequately, their accuracy was notably lower than that of the KNN and RF. The SVM algorithm achieved moderate accuracy (88.89%, performing slightly better than LR and NB but not reaching the level of KNN and RF.

Overall, the findings demonstrate that the KNN and RF algorithms outperform the other algorithms in terms of both accuracy and generalizability. The K-Fold Cross-Validation method was essential for evaluating the model's performance across multiple data partitions, providing a robust measure of each model's reliability. This method is critical in model selection because it prevents overfitting and ensures that the model performs well on both training and test data.

In conclusion, the K-Fold Cross-Validation Accuracy values demonstrate the reliability of the models in this study, and the KNN and RF models are the most effective algorithms. Given their high accuracy and strong generalizability, the KNN and RF models are recommended for tasks requiring robust and reliable ML models.

Figure 21. Comparison of algorithm accuracy and K-Fold Cross-Validation accuracy results.

High K-Fold Cross-Validation accuracy indicates strong model generalization capabilities. In particular, the consistently high performance of the KNN and RF algorithms across both accuracy metrics suggests that these models maintain stable performance on different data subsets and are not prone to overfitting. The proposed RF algorithm achieved the highest standard accuracy (95.56%), indicating excellent performance on a specific test dataset. However, its slightly lower K-Fold Cross-Validation accuracy (0.9222) suggests a potential risk of overfitting to specific datasets.

The SVM algorithm showed better K-fold cross-validation accuracy (0.8889) compared to its standard accuracy (86.67%). This indicates that SVM performed more consistently across different data subsets.

Overall, the KNN and RF algorithms demonstrated the best performance, achieving high values for both accuracy measures. In contrast, the LR and NB algorithms exhibit lower generalization abilities, performing better on certain datasets but showing weaker performance across various subsets.

3.4 Execution Times

The execution times of the algorithms were measured within the code, and the fastest methods were identified as NB and SVM. In contrast, RRF and KNN exhibited the longest processing times. RF generally runs slower than KNN; however, in some instances, KNN required the longest execution time. LR, although slower than NB and SVM, typically produced results faster than RF and KNN. In rare cases, KNN outperformed LR in terms of execution speed.

In summary, the general ranking of execution times was NB > SVM > LR > KNN > RF. Occasionally, the ranking shifted to NB > SVM > LR > RF > KNN, and in rare instances, NB > SVM > KNN > LR > RF was observed. These results highlight the importance of considering execution times when evaluating the performance of ML algorithms.

4. CONCLUSIONS AND RECOMMENDATIONS

This study aimed to predict the outcomes of cryotherapy for wart treatment using ML classification algorithms. The algorithms applied in this study were KNN, LR, NB, RF, and SVM. Among these, RF and SVM were found to have the highest accuracy rates 96.67%.

Compared to the first reference study, the proposed KNN classification method achieved an accuracy rate of 96.78%, which is slightly higher than the accuracy obtained by the KNN method in this study. In the second reference study, the RF algorithm achieved an accuracy of 96.43%, which is slightly lower than the RF accuracy achieved in this research. Based on the results of all classification algorithms, we conclude that RF and SVM are reliable methods for predicting the outcomes of wart treatment with cryotherapy.

In terms of execution time, the SVM method demonstrated faster processing than RF. Therefore, given its faster execution time and high accuracy, SVM may be a more preferable method for real-time applications.

In future research, expanding the dataset would allow for more reliable predictions of the effects of cryotherapy on warts. By increasing the dataset size, the generalizability of the findings could be improved, and the performance of the algorithms could be further enhanced. In addition, exploring other ML algorithms and optimizing model parameters may yield even better prediction accuracy for treatment outcomes.

AUTHORS' CONTRIBUTIONS

All authors contributed equally to the study.

STATEMENT OF INTEREST CONFLICTS

The authors declare no conflict of interest.

STATEMENT OF RESEARCH AND PUBLICATION ETHICS

The author declares that this study complies with Research and Publication Ethics.

REFERENCES

Abu Alfeilat, H. A., Hassanat, A. B., Lasassmeh, O., Tarawneh, A. S., Alhasanat, M. B., & Brini, S. (2019). Effects of Distance Measure Choice on K-Nearest Neighbor Classifier Performance: A Review. Big Data, 7(4), 221-248. doi: 10.1089/big.2018.0175.

Akyol, K., Karacı, A., & Gültepe, Y. (2018, May). A Study on Prediction Success of Machine Learning Algorithms for Wart Treatment. International Conference on Advanced Technologies, Computer Engineering and Science (ICATCES'18), Safranbolu, Turkey.

Al Aboud, A. M., & Nigam, P. K. (2024). Wart. In StatPearls [Internet]. Treasure Island (FL): StatPearls Publishing. Available: https://www.ncbi.nlm.nih.gov/books/NBK441824/

Bewick, V., Cheek, L., & Ball, J. (2005). Statistics review 14: Logistic regression. Critical Care (London, England), 9, 112-118. doi: 10.1186/cc3045.

Bouzigon, R., Grappe, F., Ravier, G., & Dugue, B. (2016). Whole- and partial-body cryostimulation/cryotherapy: current technologies and practical applications. Journal of Thermal Biology. doi: 10.1016/j.jtherbio.2016.08.009.

Breiman, L. (2001). Random Forests. Machine Learning, 45, 5-32. doi: 10.1023/A:1010933404324.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. Machine Learning, 20, 273-297. doi: 10.1007/BF00994018.

Das, S., Dey, A., Pal, A., & Roy, N. (2015). Applications of Artificial Intelligence in Machine Learning: Review and Prospect. International Journal of Computer Applications, 115, 31-41. doi: 10.5120/20182-2402.

Gibbs, S., Harvey, I., Sterling, J., & Stark, R. (2002). Local treatments for cutaneous warts: systematic review. BMJ, 325(7362), 461. doi: 10.1136/bmj.325.7362.461.

Guven AF (2024) Integrating electric vehicles into hybrid microgrids: A stochastic approach to future-ready renewable energy solutions and management. Energy 303:131968. https://doi.org/10.1016/j.energy.2024.131968.

Güven AF (2024) Exploring solar energy systems: A comparative study of optimization algorithms, MPPTs, and controllers. IET Control Theory Appl 18:887–920. https://doi.org/10.1049/cth2.12626.

Jain, Y. K., & Bhandare, S. K. (2011). Min Max Normalization Based Data Perturbation Method for Privacy Protection. International Journal of Computer & Communication Technology, 2(8), 45-50.

Kataria, A., & Singh, M. D. (2013). A review of data classification using k-nearest neighbour algorithm. International Journal of Emerging Technology and Advanced Engineering, 3(6), 354-360.

Khozeimeh, F., Alizadehsani, R., Roshanzamir, M., Khosravi, A., Layegh, P., & Nahavandi, S. (2017). An expert system for selecting wart treatment method. Computer Methods and Programs in Biomedicine, 81, 167-175. doi: 10.1016/j.compbiomed.2017.01.001.

Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. In Proceedings of the 14th International Joint Conference on Artificial Intelligence, 1137-1143.

Leman, J. A., & Benton, E. C. (2000). Verrucas. Guidelines for management. American Journal of Clinical Dermatology, 1(3), 143-149. doi: 10.2165/00128071-200001030-00001.

Li, H., & Zhao, M. (2010). Improving the k-nearest neighbor classifier by the use of expanded min-max normalization. Expert Systems with Applications, 37(7), 4899-4907.

Lye, A., & Miles, J. A. R. (1951). The myrmecia. British Medical Journal, 28, 912-916.

Mahesh, B. (2020). Machine learning algorithms-a review. International Journal of Science and Research (IJSR), 9(1), 381-386.

Nuthan, R., & Rohith, V. (2021). Detection of Skin Cancer Using KNN and Naive Bayes Algorithms. Turkish Online Journal of Qualitative Inquiry, 12(6).

Olson, J. E., & Stravino, V. D. (1972). A review of cryotherapy. Physical Therapy, 52(8), 840-853. doi: 10.1093/ptj/52.8.840.

Plasencia, J. M. (2000). Cutaneous warts: diagnosis and treatment. Primary Care, 27(2), 423-434. doi: 10.1016/s0095-4543(05)70204-9.

Rahman, M. M., Wang, S., Zhou, Y., & Rogers, J. (2019). Predicting the Performance of Cryotherapy for Wart Treatment Using Machine Learning Algorithms.

Refaeilzadeh, P., Tang, L., & Liu, H. (2009). Cross-Validation. In L. Liu & M. T. Özsu (Eds.), Encyclopedia of Database Systems (pp. 532-538). Springer US. doi: 10.1007/978-0-387-39940-9_565.

Rowson, K. E., & Mahy, B. W. (1967). Human papova (wart) virus. Bacteriological Reviews, 31(2), 110-131. doi: 10.1128/br.31.2.110-131.1967.

Sharma, D., & Kumar, N. (2017). A review on machine learning algorithms, tasks and applications. International Journal of Advanced Research in Computer Engineering & Technology (IJARCET), 6(10), 2278-1323.

Wong, T. T. (2015). Performance evaluation of classification algorithms by k-fold and leave-one-out cross-validation. Pattern Recognition, 48(9), 2839-2846. doi: 10.1016/j.patcog.2015.03.009.

Xu, Y., & Goodacre, R. (2018). On splitting training and validation set: a comparative study of cross-validation, bootstrap and systematic sampling for estimating the generalization performance of supervised learning. Journal of Analysis and Testing, 2, 249-262. doi: 10.1007/s41664-018-0068-2

Yadav, S., & Shukla, S. (2016). Analysis of k-fold cross-validation over hold-out validation on colossal datasets for quality classification. In 2016 IEEE 6th International Conference on Advanced Computing (IACC), 78-83. doi: 10.1109/IACC.2016.25.