electronics

MDPI

Article

Artificial Intelligence Implementation in Internet of Things Embedded System for Real-Time Person Presence in Bed Detection and Sleep Behaviour Monitor

Minh Long Hoang *, Guido Matrella * and Paolo Ciampolini

Department of Engineering and Architecture, University of Parma, 43124 Parma, Italy; paolo.ciampolini@unipr.it
* Correspondence: minhlong.hoang@unipr.it (M.L.H.); guido.matrella@unipr.it (G.M.)

Abstract: This paper works on detecting a person in bed for sleep routine and sleep pattern monitoring based on the Micro-Electro-Mechanical Systems (MEMS) accelerometer and Internet of Things (IoT) embedded system board. This work provides sleep information, patient assessment, and elderly care for patients who live alone via tele-distance to doctors or family members. About 216,000 pieces of acceleration data were collected, including three classes: no person in bed, a static laying position, and a moving state for Artificial Intelligence (AI) application. Six well-known Machine-Learning (ML) algorithms were evaluated with precision, recall, F1-score, and accuracy in the workstation before implementing in the STM32-microcontroller for real-time state classification. The four best algorithms were selected to be programmed into the IoT board and applied for real-time testing. The results demonstrate the high accuracy of the ML performance, more than 99%, and the Classification and Regression Tree algorithm is among the best models with a light code size of 1583 bytes. The smart bed information is sent to the IoT dashboard of Node-RED via a Message Queuing Telemetry broker (MQTT).

Keywords: person on bed detection; sleep pattern; machine learning; artificial intelligence; accelerometer; smart bed

Citation: Hoang, M.L.; Matrella, G.; Ciampolini, P. Artificial Intelligence Implementation in Internet of Things Embedded System for Real-Time Person Presence in Bed Detection and Sleep Behaviour Monitor. *Electronics* **2024**, *13*, 2210. https://doi.org/10.3390/electronics13112210

Academic Editors: Ilaria Sergi and Teodoro Montanaro

Received: 30 April 2024
Revised: 30 May 2024
Accepted: 3 June 2024
Published: 6 June 2024

1. Introduction

The incorporation of cutting-edge technologies such as Micro-Electro-Mechanical Systems (MEMS) accelerometers [1–3], Internet of Things (IoT) frameworks [4–7], and Machine-Learning (ML) [8–10] algorithms has significantly transformed healthcare monitoring, specifically in the field of sleep pattern analysis. Sleeping on time and a deep sleep are key to health maintenance [11]. For restless sleep, like insomnia [12], examples of symptoms shown would be tossing and turning, and trying to get settled and comfortable [13]. The capacity to autonomously identify the existence of an individual on a bed and differentiate between their stationary rest and restless movements carries great importance in diverse healthcare settings. The significance of adequately classifying sleep conditions, such as stationary lying positions or frequent motion, cannot be underestimated. Detecting variations from typical sleep patterns can assist in the early identification of sleep disorders, assess the efficacy of therapies, and reduce potential health hazards linked to inadequate sleep quality. Moreover, for persons with persistent ailments or elderly individuals who are susceptible to falls or disruptions while sleeping, prompt intervention with live monitoring can significantly improve their quality of life and autonomy.

In the article [14], movement activity was extracted from the multichannel ballistocardiography measurements based on Emfit sensor foils placed in the bed mattress, which has a total accuracy of 83%. However, the BCG data's noise, artifacts, or signal distortions can complicate the signal processing algorithms and reduce their accuracy in detecting movement events. Maintaining and calibrating Emfit sensor foils regularly is necessary to ensure their optimal performance over time. Not properly maintaining or calibrating

the sensors may result in sensor readings drifting or a decrease in measurement precision, which can affect the reliability of detecting movement activity. This paper discusses the crucial role of technological improvements in enabling patient assessment and aged care, particularly for persons who live alone. It focuses on using remote sleep pattern monitoring to provide valuable information and actionable insights for healthcare professionals and family members.

Another work of research with bed sensors [15] uses eight pressure sensor signals for movement on bed detection by calculating the respiration amplitude using the Hilbert transform, and then identifying events based on a 20% amplitude reduction from the baseline signal. On the other hand, pressure sensor signals can be susceptible to noise and artifacts from various sources, such as movement unrelated to respiration, environmental vibrations, or sensor malfunctions. These noise sources can affect the accuracy of respiration amplitude calculations and event detection, leading to false positives or false negatives in identifying respiration events.

Therefore, we use the MEMS accelerometer, which has a high sensitivity and accuracy in detecting even subtle movements, allowing for the precise monitoring of movements on the bed, including behaviours during sleep. In addition, the MEMS accelerometer is capable of measuring acceleration in multiple axes, providing comprehensive movement data that can capture movements in different directions and orientations on the bed. The ML implementation based on MEMS accelerometers with IoT systems provides a non-invasive and economical method of collecting uninterrupted sleep-related data in real time. The utilization of ML algorithms for classification tasks is crucial in this context because of their ability to identify complicated structures with a high precision.

Various papers apply ML into accelerations for sleep assessment [16]. In [17], a random forest algorithm is used to classify sleep using wrist-worn accelerometer data. Another article [18] uses ActiGraph wGT3X-BT accelerometers on the user's hip for sleep classification with ML. Work [19] creates a device that monitors the sleep stages with a pulse oximeter and accelerometer placed on the wrist, and then applies ML for sleep monitoring. Another work of research [20] develops deep learning for sleep–wake classification with three types of wearable sensors: Faros sensors, a motion watch, and a head watch. Although these approaches reach a certain accuracy, the wearable accelerometers still cause discomfort to the user during long usage.

The smartwatch is also a popular device for health monitoring. Article [21] states that it is safe to wear a smartwatch, but it needs to be removed for several hours to allow the skin to breathe and prevent the accumulation of bacteria beneath the watch. The illumination emanating from the watch or its strap can occasionally disrupt the natural rhythm of sleep. The smartwatch strap can cause skin irritation and inflammation for some people when tightly wrapped around the wrist or when exposed to residual water or sweat. The notifications from the smartwatch can also disturb sleep. It is difficult for many people, especially older people, to wear a device frequently due to annoyance.

Beside the ML algorithms, Deep-Learning (DL) [22] models also have a good command of classifying the signals, such as the Deep Neural Network (DNN) [23], Long Short-Term Memory (LSTM) [24], Bidirectional LSTM [25], etc. Nevertheless, ML has advantages with a simpler implementation, and ML algorithms typically require less computational resources compared to DL algorithms. This point is especially important in microcontroller applications where resources like memory and processing power are limited. In this case, the dataset is not extremely complicated and ML algorithms perform well with certain datasets. Additionally, pre-trained DL models require more time consumption due to their architecture. Thus, the ML approach is selected to minimize the complexity of the whole system.

This study aims to develop a nonwearable platform which is able to monitor the sleep state without physical contact with the users. In this way, comfort is guaranteed for the users, especially for old people who can forget to equip the wearable device frequently. Furthermore, this work identifies the most efficient models for real-time sleep state classifi-

cation by evaluating various ML algorithms, including the Extra Tree Classifier (ET) [26–29], Logistic Regression (LR) [30–32], Linear Discriminant Analysis (LDA) [33,34], Classification and Regression Trees (CART) [35–40], Support Vector Machines (SVMs) [41,42], and Random Forest (RF) [43–45]. Afterward, the chosen algorithms are implemented in the STM32 embedded system [46] within the IoT framework, allowing a smooth integration into the current healthcare infrastructure for widespread adoption. This research adds to the advancement of sleep pattern monitoring and remote healthcare management by combining MEMS accelerometers, IoT frameworks, and ML algorithms with a microcontroller (MCU) platform. The impacts of our research's results spread throughout the field of sleep research, providing promising possibilities for individualized healthcare interventions and proactive wellness management.

The paper is organized as follows: First, the setup and devices will be depicted, together with the system working principle. In the next section, the ML algorithms will be described briefly; then, it will be their cross-validation and test part. The subsequent content is about the real-time test with the MCU operation and sleep state on the IoT dashboard demonstration. Finally, the conclusion and future work are at the end of the paper.

2. Materials and Methods

In this section, the system architecture and working principle will be shown, together with brief descriptions of the applied ML algorithms.

2.1. Setup and Devices

Figure 1 shows the data acquisition diagram from MCU and MEMS Accelerometer under the IoT bed to the workstation. The STM32 B-L475-IOT01A microcontroller board [47] from the IoT kit has acquired acceleration data using Serial Peripheral Interface (SPI) communication. The data collected by the accelerometer were sent to a workstation for immediate storage in text files. STM32 B-L475-IOT01A was made by STMicroelectronics company which is headquartered in Plan-les-Ouates, Switzerland.

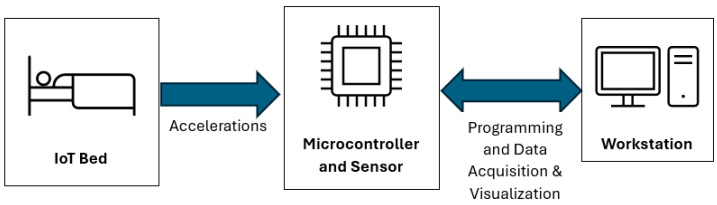

Figure 1. Data acquisition diagram.

For ML model evaluation, these files were trained using Python programming [48], with the packages Scikit-learn [49]. Then, the best models are chosen to implement into MCU via C programming language [50,51] to communicate with the IoT dashboard wirelessly as shown in Figure 2.

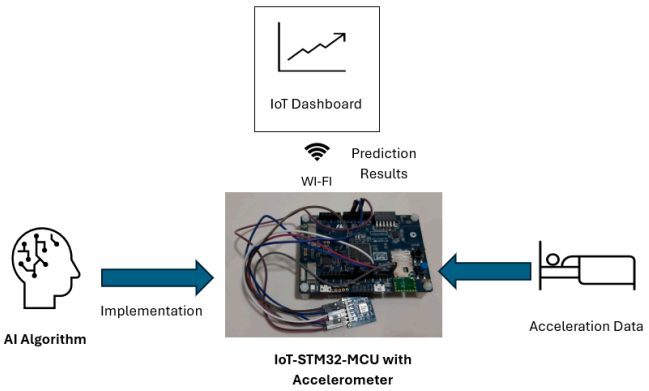

Figure 2. AI implementation in microcontroller (MCU) under IoT communication.

The MEMS accelerometer ADXL355 [52] is concealed within a protective enclosure and positioned accurately beneath the bed frame, as shown in Figure 3, and it is connected to the embedded system platform appropriately. The ADXL355 offers high-resolution measurement capabilities, allowing it to detect even subtle movements during sleep accurately. Its high sensitivity level can provide comprehensive insights into sleep quality and duration. Additionally, for applications like sleep monitoring where the device needs to operate continuously for extended periods, low power consumption is essential to prolong battery life and minimize the need for frequent recharging or replacement. The ADXL355's low power consumption ensures that it can monitor sleep patterns throughout the night without draining the device's battery quickly. ADXL355 was made by Analog Devices Company, which is headquartered in Wilmington, MA, USA.

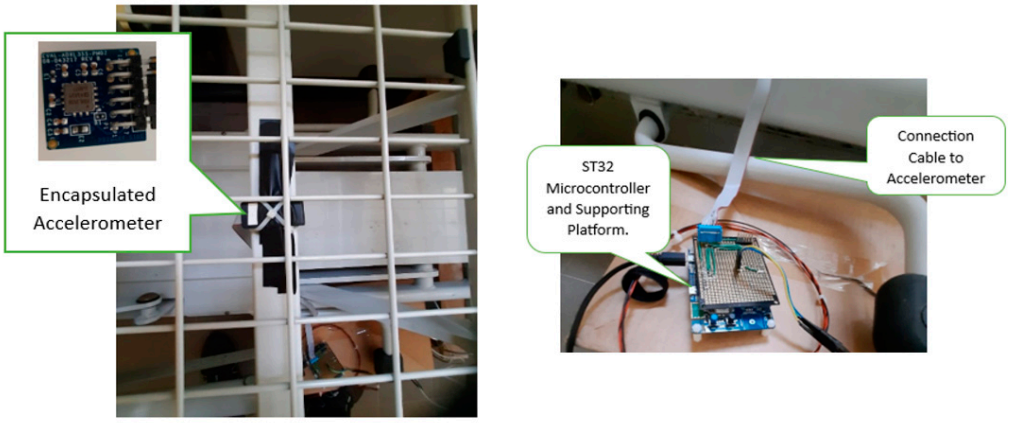

Figure 3. Encapsulated accelerometer and MCU platform under the bed frame.

All the devices were mounted under the bed as shown in Figure 4.

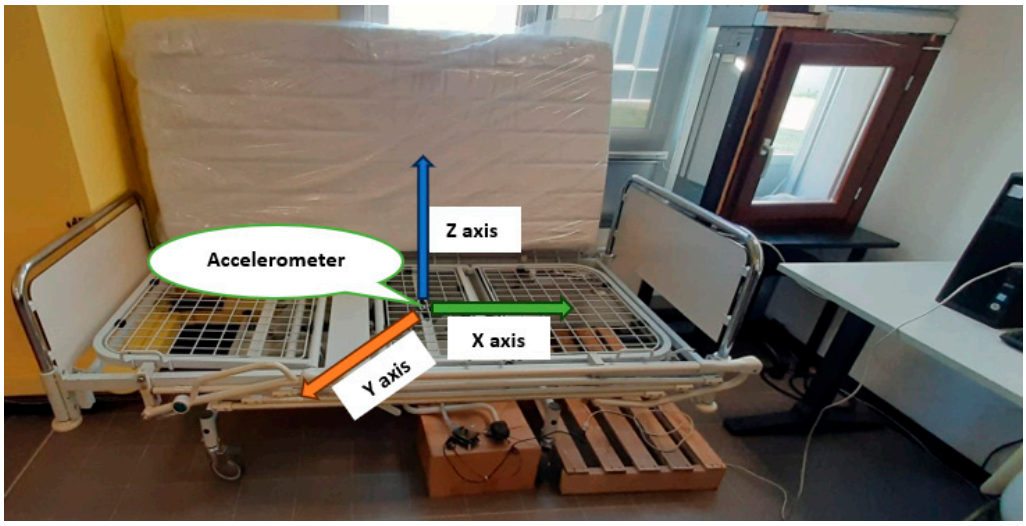

Figure 4. Smart bed under test.

2.2. Raw Data

During the experiment, the user lies on the bed in 4 stationary positions, prone, back, right side, and left side, to collect data for static state. In the dynamic state, the person moves and turns around. Finally, an empty state is carried out where no person is on the bed, and the sensor keeps acquiring the acceleration for data collection. The volunteer participant is more than 30 years old and has a sleeping issue.

Figure 5 shows the raw data in 3 concerned states on all axes: no person in bed; lying static positions (joined data from 4 lying positions), and dynamic motion in bed. It is

complicated to detect a proper threshold to recognize and distinguish these states since the data conduct high variations, and their ranges are not so clearly different, especially in the case of no person in bed and lying in static positions. There are also influences from different sleep positions, which cause the data to be more intricate. Thus, it is necessary to use the ML approach to achieve high classification efficiency.

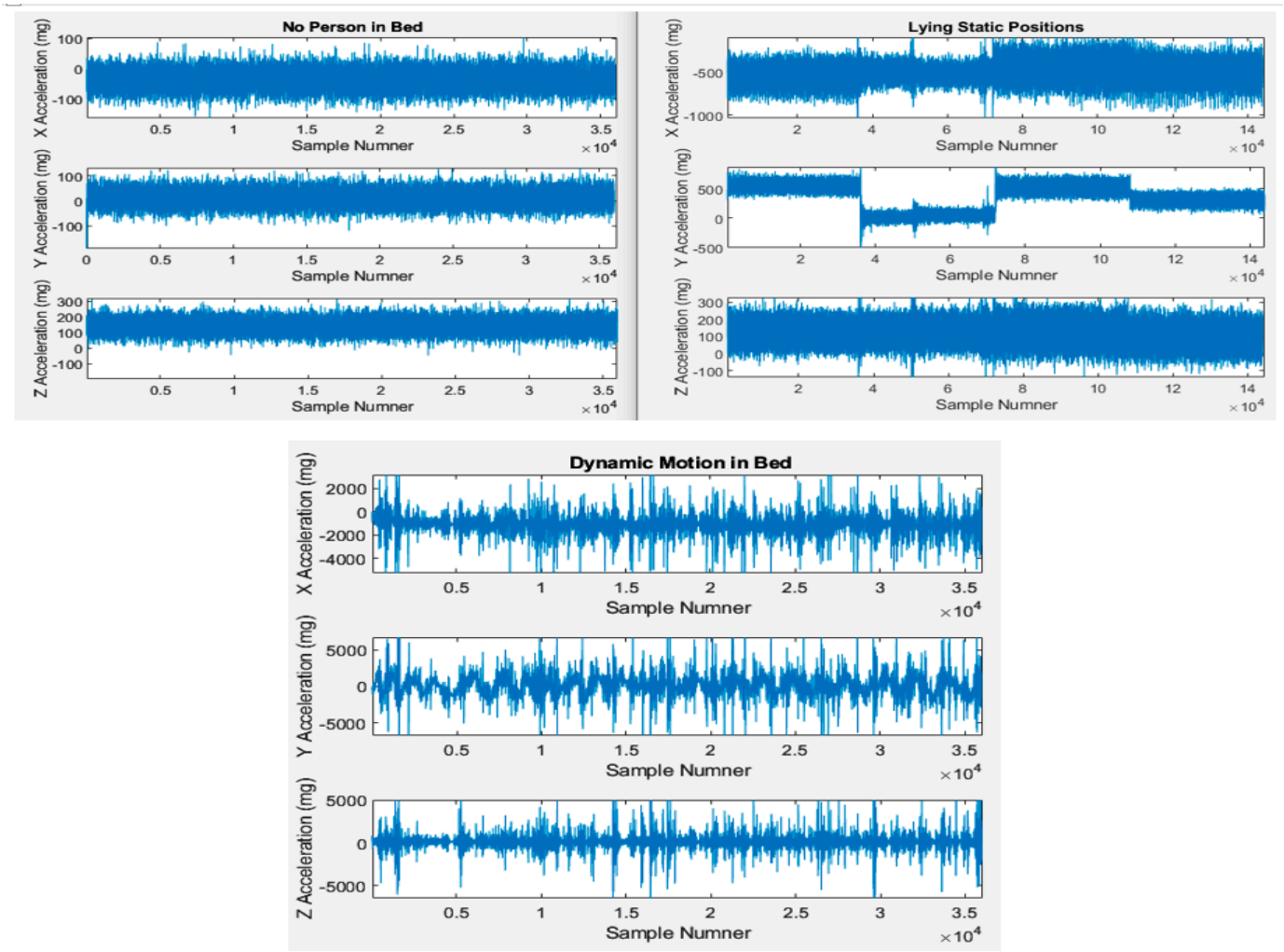

Figure 5. Raw data in 3 states.

2.3. Data Processing and ML features

The output data rate of the device is 200 Sa/s. The absolute difference between 2 consecutive accelerations (Δacc) on the X, Y, and Z axes are calculated as features. There are 216,000 data, containing 1080 windows of 200 samples. The sum of each window is calculated to form the final feature for each axis to classify 3 states: no person, static lying state, and moving state, as described in Table 1. 70% of data were used for training and validation; 30% of the rest is used for testing the ML models. For each second, the ML model will predict the presence of a person in bed and the lying state: static or dynamic state.

Table 1. ML feature and classification.

ML Input Features			Class
ΔXsum	ΔYsum	ΔZsum	• No Person • Static Lying State • Moving State

2.4. ML Algorithms

There are 6 powerful algorithms in consideration for ML classification:

LG is a statistical model that estimates the probability of an instance belonging to a specific class. LG does this by utilizing a logistic function, which is also referred to as the sigmoid function. This function maps any real-valued input to a value within the range of 0 to 1. This transformation is crucial as it enables LR to generate probabilities, which are subsequently utilized for binary classifications. During training, the logistic regression model learns the association between the input features and the target class by estimating coefficients for each feature. These coefficients indicate the individual impact of each attribute on the ultimate forecast. The model iteratively changes these coefficients to minimize the discrepancy between its predictions and the actual class labels in the training data. After being trained, LR utilizes the acquired coefficients to estimate the likelihood that a new instance is part of the positive class. The algorithm calculates a weighted total of the input features, multiplying each feature by its respective coefficient. The total is subsequently inputted into the logistic function to derive the likelihood. LR employs a threshold, usually set at 0.5, to determine the final binary classification based on the likelihood. If the estimated probability exceeds the specified threshold, the instance is categorized as belonging to the positive class; otherwise, it is categorized as belonging to the negative class. LR has a significant benefit in terms of interpretability. LR calculates coefficients for each feature, allowing us to understand the significance and direction of their impact on the projected probability, which assists in the comprehension of how various characteristics contribute to the determination of classification. Nevertheless, LR also possesses its constraints. It presupposes a linear correlation between the characteristics and the logarithm of the probability of the result, which may not consistently be accurate in some scenarios. LR is susceptible to the influence of outliers and multicollinearity among features, which can have a negative impact on the stability and dependability of its predictions.

LDA is a classification approach that identifies optimal linear combinations of characteristics to effectively distinguish between different classes in the data. LDA assumes that the features adhere to a normal (Gaussian) distribution and that the covariance of the features is uniform across all classes. The main objective of LDA is to identify a linear decision boundary that optimizes the distance between different classes and minimizes the variability within each class. In order to accomplish it, LDA initiates the process by creating a model that represents the distribution of features for each class. The algorithm calculates the average and covariance matrix for each class, assuming that the characteristics within each class adhere to a multivariate normal distribution. After modeling the class distributions, LDA computes the class priors, indicating the probability of meeting each class in the dataset. These prior probabilities are usually calculated by determining the relative frequencies of each class in the training data. LDA utilizes the class distributions and priors to create decision boundaries that define specific areas in the feature space for each class. The decision borders are established by optimizing a criterion to maximize the variance ratio across classes to variation within classes. When confronted with new data points, LDA employs the acquired decision boundaries to categorize them according on their feature values. The algorithm utilizes Bayes' theorem to compute the posterior probability of each class for a certain data point, considering both the class distribution and the class priors. The data point is allocated to the class with the greatest posterior probability afterward. LDA is able to handle high-dimensional data by lowering the number of dimensions while retaining most of the class-discriminating information. By mapping the data into a lower-dimensional subspace determined by the linear discriminants, LDA can effectively distinguish between classes, even when the original feature space has a high number of dimensions.

CART is a decision tree technique that iteratively divides the data space into areas according to the values of the features. The CART algorithm constructs a decision tree by dividing the feature space into subsets, each of which is assigned a particular class label.

This process is carried out iteratively and recursively. The CART algorithm commences by utilizing the complete dataset, taking into account all the features and cases. The objective is to identify the characteristic that most effectively divides the data into two groups while maximizing a measure such as Gini impurity or information gain. Gini impurity quantifies the likelihood of incorrectly classifying an instance if it were assigned a random label based on the class distribution within the subset. Information gain measures the decrease in entropy (uncertainty) when splitting is carried out using a specific feature. After selecting the initial feature, CART algorithm splits the dataset into two halves using a threshold value specific to that feature. The algorithm can generate a split for each category when dealing with categorical features. On the other hand, for numerical features, it can utilize a threshold to construct binary splits. The CART algorithm assesses all potential divisions and selects the one that optimizes the criterion, such as minimizing Gini impurity or maximizing information gain. This technique is iteratively applied to each subset, thus generating a tree structure. The process of recursion persists until a specified condition for termination is fulfilled. To ensure the model's generalization and prevent overfitting to the training data, it is important to control the depth of the tree and avoid splitting nodes with insufficient instances. In order to categorize a new instance, the decision tree must be followed from the starting point to the final point, taking into account the characteristics of the instance. At every node, it evaluates the feature value of the instance and compares it to the threshold for splitting. It then proceeds to the appropriate child node based on the comparison. This procedure iterates until it reaches a leaf node, when the projected class for the new instance is determined by selecting the majority class among the instances in that node. The main advantages of CART are its simplicity, interpretability, and capability to handle numerical and categorical data. The resulting decision tree may be readily visible and comprehended, rendering it a valuable instrument for elucidating the underlying decision-making process to stakeholders. Furthermore, CART is resistant to irrelevant characteristics and has the ability to automatically choose the most informative ones for splitting, hence minimizing the necessity for feature engineering.

ET differs from regular decision trees by incorporating an extra layer of randomization in selecting splits instead of evaluating all potential splits for each characteristic at every node. The core principle of ET revolves around the idea of employing randomization while selecting splits. When seeking the most favourable division to separate the samples of a node into two groups, ET generates random splits for each of the randomly chosen features. The randomization process is carried out separately for each feature and is iterated numerous times until a specified maximum number of splits is reached. ET enhances the decision-making process by introducing a greater level of variety and diversity by generating random splits for each characteristic. This point increased the level of randomness and helped mitigate the risk of overfitting by preventing the model from excessively depending on features or patterns in the data. Extra Trees employ a different approach than traditional methods of selecting the best split based on a predetermined criterion. Instead of relying on a single split, ET considers numerous random splits and selects the one that achieves the highest performance. ET mitigates overfitting by employing multiple random splits for each feature, resulting in decision boundaries that are less prone to overfitting the training data, which is especially advantageous when working with datasets that are noisy or have a high number of dimensions, as typical decision trees may have difficulty in properly generalizing. The increased level of unpredictability in the selection of splits enables ET to effectively capture a wider range of patterns and relationships in the data. Enhancing the model's capacity to extrapolate to unfamiliar data enhances its performance on test datasets. ET exhibits greater resilience to outliers and noisy features than conventional decision trees. The technique of randomization reduces the influence of particular data points or attributes that may significantly affect the decision-making process.

SVM is a highly adaptable supervised learning model extensively employed for classification and regression problems. The classification method employed here is distinctive as

it focuses on finding the most suitable hyperplane that successfully distinguishes different classes in the feature space. The objective is to maximize the distance between the hyperplane and the nearest data points, referred to as support vectors. SVM in classification aims to identify the hyperplane that effectively separates the classes and optimizes the margin, which is the distance between the hyperplane and the nearest data points from each class. SVM seeks to obtain superior generalization performance and resilience to data noise by maximizing the margin. The data points located on the margin and closest to the hyperplane are known as support vectors. These support vectors are essential in determining the decision boundary. However, real-world data frequently lack linear separability, indicating that a solitary hyperplane cannot achieve perfect separation between the classes. In such situations, SVM employs kernel functions to transform the input data into a feature space with more dimensions. This transformation increases the probability of finding a hyperplane that can effectively separate the classes. The transformation enables SVM to effectively deal with non-linear decision boundaries by implicitly mapping the input into a higher-dimensional space where a linear separator may be applied. Kernel functions are crucial for SVM's capacity to handle non-linearities in the data. These functions calculate the inner product of data points in the input space, quantifying their similarity. SVM applies a kernel function to the input data, implicitly transforming the data into a feature space of greater dimensions. This transformation enables the possibility of achieving linear separation. In this application, Gaussian Kernel Radial Basis Function captures complex decision boundaries that conduct a non-linearity within the original space of features. SVM's adaptability allows it to capture variety of classification problems.

RF is a technique in ensemble learning that builds numerous decision trees during training and merges their predictions to enhance performance. RF is a fundamental technique in ensemble learning, known for its capacity to improve predictive accuracy by aggregating the outputs of many decision trees. RF addresses the problems of overfitting and high variance commonly associated with single decision trees. It achieves this by utilizing many trees, which introduces variety and helps to minimize these concerns, resulting in more reliable and accurate predictions. RF method functions by constructing several decision trees in the training phase. Every decision tree is trained using a bootstrap sample of the data, which involves training each tree on a randomly chosen subset of the original dataset with replacement. Bootstrap sampling creates variability among the trees, guaranteeing that each tree learns from a slightly distinct viewpoint of the data. In addition, RF offers further unpredictability at each node of every decision tree by examining only a subset of features for splitting. Instead of assessing all features to identify the best split, RF randomly chooses a subset of features and evaluates them to discover the most optimal split. This procedure introduces diversity into the model, preventing it from unnecessarily relying on any specific characteristic and encouraging the examination of other data elements. RF possesses good qualities due to the combination of bootstrapping and random feature selection. RF mitigates the risk of overfitting and enhances generalization performance by training each decision tree on a distinct subset of the data and restricting the number of features examined at each node. The RF guarantees that the combined predictions of the entire forest minimize the errors made by each individual tree. RF is particularly effective in situations where the dataset has a large number of dimensions. The method of randomly selecting features enables RF to prioritize the most informative ones and ignore unnecessary or redundant ones, making it particularly suitable for datasets with a high number of dimensions. RF has the ability to capture complex correlations between features that individual decision trees may overlook, which is achieved by integrating the predictions of numerous decision trees, each trained on a distinct subset of data and features. An ensemble approach allows the RF algorithm to effectively capture and represent complex interactions and non-linear relationships within the data, improving its ability to make accurate predictions.

2.5. Evaluation Metrics

In order to validate the described models, the ML factors of precision, recall, and F1-score were computed. These factors were based on the values of true positive (TPA), false positive (FPA), and false negative (FNA) for class A.

- TP_A refers to the count of predictions made by a classifier that accurately forecast class A;
- FP_A refers to the count of objects that are not part of class A, yet are incorrectly classified as class A;
- FN_A represents the count of objects belonging to class A that are anticipated to belong to a different class;
- Precision measures the proportion of positive class predictions that truly belong to the positive class. The calculation involves adding up the number of true positives for each class and dividing it by the sum of true positives and false positives for all classes:

$$\text{Precision} = \frac{TP_A}{TP_A + FP_A}$$

- Recall is a metric that measures the proportion of positive class predictions produced correctly out of all the positive examples in the dataset. Unlike precision, which only evaluates the accuracy of correctly predicted positive outcomes out of all positive predictions, recall measures the number of positive predictions that were missed. In the context of multiple categorizations, recall is calculated by adding up the number of true positives for each category and dividing it by the sum of true positives and false negatives across all categories:

$$\text{Recall} = \frac{TP_a}{TP_A + FN_A}$$

- F1-score is a unified metric that takes into account both precision and recall, providing a single score. The F-score provides a method to merge precision and recall into a single metric that encompasses both attributes. After obtaining the values for precision and recall in the multiclass classification issue, these two scores can be merged to calculate the F-Measure. Similar to precision and recall, an F-Measure score of 0.0 indicates poor performance, whereas a score of 1.0 represents the best or flawless performance:

$$\text{F1-score} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

- Accuracy is the ratio of correct predictions to the total number of predictions.

$$\text{Accuracy} = \frac{\text{Correct predictions}}{\text{Total predictions}}$$

3. Results and Discussion

This section shows the performances of ML models via cross-validation. The best models were programmed into the microcontroller for further analysis in real time and code size.

3.1. Model Comparison and Selection

K-fold cross-validation is a widely used technique in ML that validates the performance of a predictive model and mitigates the risk of overfitting. The process entails dividing the dataset into K subsets or folds, training, and evaluating the model K times. For each iteration, one specific fold is set aside as the test set, while the rest of the K-1 folds are used for training. The outcomes are calculated by averaging across K iterations,

guaranteeing a more robust and reliable performance estimation. In this instance, the K-fold cross-validation was performed using a value of k equal to 10.

The AI models underwent training on the host computer, which was equipped with an NVIDIA Quadro P620 featuring a Pascal GPU boasting 512 CUDA cores (NVIDIA, Santa Clara, CA, USA). The system is equipped with 2 GB of GDDR5 memory, an Intel Core i7 vPro-10850H Processor operating at a speed of 2.70 GHz, and 32 GB of RAM. The machine is manufactured by Intel, located in Santa Clara, CA, USA.

As shown in Figure 6 and Table 2, ET, LR, CART, and RF have the highest accuracy among all the models. Among these four models, the LR and RF possess the lowest standard deviation (std), showing the small variance between all 10 results. LDA has a weaker prediction capability for this specific case with an accuracy of 0.8, and SVM has a poor accuracy of 0.6. It is essential to achieve more than 0.9 of accuracy for this sleep-monitoring task to handle the proper information, so the four models with an accuracy of 0.99 are selected for a further test procedure.

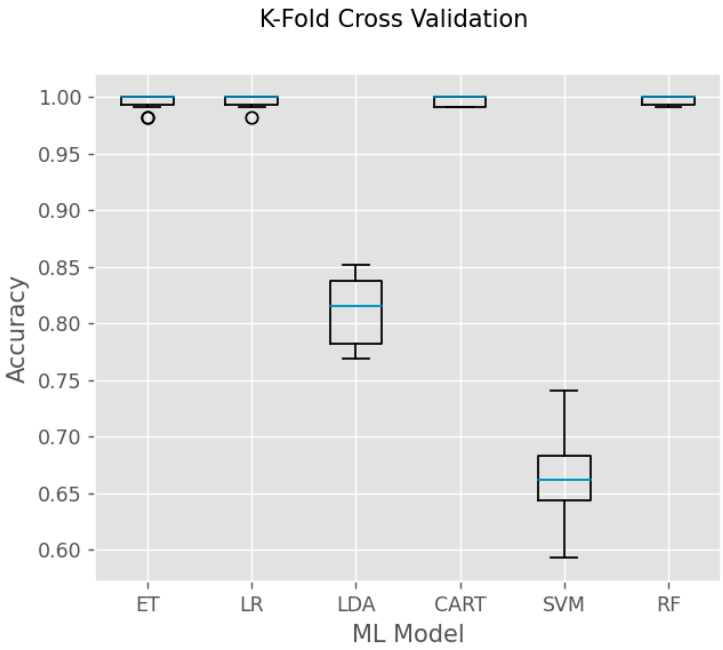

Figure 6. K-fold cross-validation for model evaluations.

Table 2. AI model comparison.

Models	Mean Accuracy	Std	10-Fold Cross Validation Time (s)
ET	0.997	0.007	0.029
LR	0.997	0.004	0.593
LDA	0.811	0.028	0.034
CART	0.997	0.005	0.029
SVM	0.666	0.045	1.307
RF	0.997	0.004	0.194

3.2. Test Process

In this test, 30% of the total data are used for testing the four selected models. Overall, all of them attain highly effective results. As reported in Table 3, all three tree algorithms have the same performance, with the precision, recall, and F1-score being similar. Their predictions are almost perfect, with a small misprediction between the moving state and static position. The moving state has a recall of 0.98, which means that the models correctly identified and classified 98% of the actual instances. Static position has a precision of 0.98,

showing that, out of all instances predicted as a static position by the models, 98% are actually true static position instances.

Table 3. Test result on the selected models.

State	Precision				Recall				F1-Score			
	ET	LR	CART	RF	ET	LR	CART	RF	ET	LR	CART	RF
Moving State	1	1	1	1	0.98	0.96	0.98	0.98	0.99	0.98	0.99	0.99
No Person	1	0.98	1	1	1	0.99	1	1	1	0.99	1	1
Static Position	0.98	0.96	0.98	0.98	1	0.99	1	1	0.99	0.98	0.99	0.99
• Total Accuracy of ET, CART, and RF models: 99.34%												
• Total Accuracy of LG model: 98.01%												

Unlike the tree algorithms, LG has an inferior performance in the test process with the wrong predictions also in the case of no person, which leads to the precision, recall, and F1-score of the other classes not being as good as the ET, CART, and RF algorithms.

3.3. Real-Time Test with STM32-Microcontroller

In this stage, the trained models of the four selected algorithms are implemented in the IoT kit STM32 B-L475-IOT01A microcontroller board. The real-time test with the embedded system includes:

- No person is in bed with 720,000 data;
- A person statically lies on the bed with 1,440,000 data;
- A person moves on the bed with 180,000 data.

Table 4 reports the accuracy and the code size of the ML models. All the models demonstrate a good execution with an accuracy greater than 98%. The LR model has the advantage of the lightest code size, but a lower accuracy than the other models. Since the performance of the other three algorithms are almost the same, the CART model is the most suitable model for the MCU application, with the smallest code size among the tree models.

Table 4. Real-time test with microcontroller.

ML Model	Accuracy (%)	Code Size (Bytes)
ET	99.325	4346
LG	98.05	1480
CART	99.325	1583
RF	99.330	18,224

3.4. IoT Dashboard

With the MQTT [53–56] broker by HiveMQ [57], the Node Red [58–62] received the ML prediction about the monitoring state via Wi-Fi, which can be observed via tele-distance as a result of the IoT technology, as shown in Figure 7. Here, the numeric data means:

- 0: No person on bed;
- 1: Person stays in static position on the bed;
- 2: Person moves on bed.

The dashboard on Figure 7 shows a case where, in the first period, there is no person in bed (0), then the person goes to bed and turns himself to adjust the sleep position, causing a strong motion (2), then lays in a stationary position (1). In this way, the sleep routine of the concerned person will be monitored effectively from a far distance via Wi-Fi based on IoT communication.

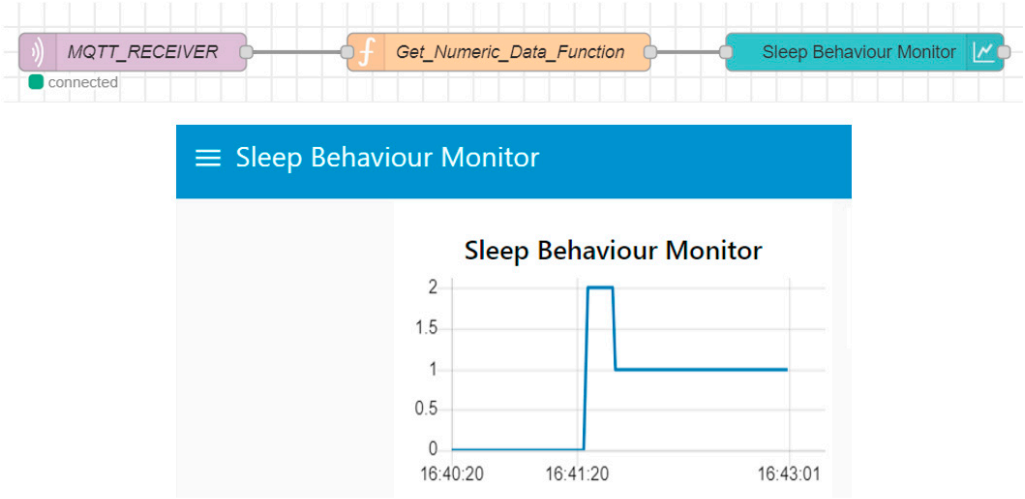

Figure 7. K-Node-RED block connection and IoT dashboard.

4. Discussion

This paper outlines the approaches and results of our study, which entailed gathering and examining more than 216,000 pieces of acceleration data of X, Y, and Z axes points. The difference of two consecutive samples are elaborated, and each of the 200 samples formed a window. These windows are the ML instances for the purpose of training and validating the ML algorithms. The results emphasize the outstanding performance of the chosen Machine-Learning models, reaching accuracy rates that surpass 99% in tests conducted on workstations. The CART technique stood out among the best models in terms of performance, showing both a high accuracy and the lightest code size that is suitable for effective deployment on a IoT microcontroller with limited memory. The monitoring process is carried out via the MQTT protocol and Node-RED development tool.

5. Conclusions

In conclusion, this study demonstrates the significant potential of integrating MEMS accelerometers, IoT frameworks, and Machine-Learning algorithms for automated sleep pattern analysis. By accurately detecting individuals in a bed and classifying their sleep states in real time, this approach offers promising avenues for enhancing patient outcomes, particularly in remote monitoring and elderly care scenarios. The findings underscore the transformative impact of technology-driven healthcare solutions and highlight the importance of continued innovation in this field to realize its full potential in improving healthcare delivery and patient well-being.

Three algorithms, ET, CART, and RF, accomplish the best performances. LG also has a high efficiency with a light code size, although its accuracy is still lower than that of the tree models. This work shows the potential of the automated sleep pattern analysis of a real-time ML model implementation. By remotely monitoring sleep patterns, healthcare professionals can tailor interventions and treatment plans based on individualized data insights, enhancing patient outcomes and fortifying elderly care. Furthermore, the scalability and accessibility afforded by IoT devices promise to democratize access to sleep-monitoring services, ensuring equitable healthcare provision across diverse populations.

As we navigate towards this future, it is imperative to harness the full potential of these technologies to realize their transformative impact on healthcare delivery and improve the lives of individuals worldwide. In future work, more algorithms with more tests will be carried out for the developed platform. Other types of beds and more patients will be tested with the developed system.

Author Contributions: Methodology, M.L.H.; software, M.L.H.; validation, G.M.; formal analysis, M.L.H.; investigation, P.C.; resources, G.M.; writing—original draft, M.L.H.; writing—review and editing, G.M.; supervision, P.C.; project administration, P.C. All authors have read and agreed to the published version of the manuscript.

Funding: This study has been realized with the co-financing of the Ministry of University and Research in the framework of PNC "DARE—Digital lifelong prevention project" (PNC0000002—CUP B53C22006450001). The views and opinions expressed are solely those of the authors and do not necessarily reflect those of the European Union, nor can the European Union be held responsible for them.

Data Availability Statement: The data are contained within the article.

Acknowledgments: The authors would like to thank the University of Parma for supporting this project.

Conflicts of Interest: The authors declare no conflicts of interest.

Notations and Abbreviations

CART	Classification and Regression Trees
ET	Extra Tree Classifier
IoT	Internet of Things
LR	Logistic Regression
LDA	Linear Discriminant Analysis
MCU	Microcontroller
ML	Machine Learning
MQTT	Message Queuing Telemetry broker
RF	Random Forest
SVM	Support Vector Machines

References

1. Hoang, M.L.; Carratu, M.; Ugwiri, M.A.; Paciello, V.; Pietrosanto, A. A New Technique for Optimization of Linear Displacement Measurement Based on MEMS Accelerometer. In Proceedings of the 2020 International Semiconductor Conference (CAS), Sinaia, Romania, 7–9 October 2020. [CrossRef]
2. Hoang, M.L.; Pietrosanto, A. New Artificial Intelligence Approach to Inclination Measurement Based on MEMS Accelerometer. *IEEE Trans. Artif. Intell.* **2021**, *3*, 67–77. [CrossRef]
3. Hoang, M.L.; Pietrosanto, A. An Effective Method on Vibration Immunity for Inclinometer Based on MEMS Accelerometer. In Proceedings of the 2020 International Semiconductor Conference (CAS), Sinaia, Romania, 7–9 October 2020. [CrossRef]
4. Alqahtani, A.; Alsubai, S.; Bhatia, M. IoT-Edge–Cloud-Assisted Intelligent Framework for Controlling Dengue. *IEEE Internet Things J.* **2023**, *11*, 15682–15689. [CrossRef]
5. AbdelHafeez, M.; AbdelRaheem, M. AssIUT IOT: A Remotely Accessible Testbed for Internet of Things. In Proceedings of the IEEE Global Conference on Internet of Things (GCIoT), Alexandria, Egypt, 5–7 December 2018. [CrossRef]
6. Chataut, R.; Phoummalayvane, A.; Akl, R. Unleashing the Power of IoT: A Comprehensive Review of IoT Applications and Future Prospects in Healthcare, Agriculture, Smart Homes, Smart Cities, and Industry 4.0. *Sensors* **2023**, *23*, 7194. [CrossRef] [PubMed]
7. Ullah, I.; Ullah, A.; Sajjad, M. Towards a Hybrid Deep Learning Model for Anomalous Activities Detection in Internet of Things Networks. *IoT* **2021**, *2*, 428–448. [CrossRef]
8. Hosseini, M.-P.; Hosseini, A.; Ahi, K. A Review on Machine Learning for EEG Signal Processing in Bioengineering. *IEEE Rev. Biomed. Eng.* **2021**, *14*, 204–218. [CrossRef] [PubMed]
9. Hoang, M.L.; Nkembi, A.A.; Pham, P.L. Real-Time Risk Assessment Detection for Weak People by Parallel Training Logical Execution of a Supervised Learning System Based on an IoT Wearable MEMS Accelerometer. *Sensors* **2023**, *23*, 1516. [CrossRef] [PubMed]
10. Hoang, M.L.; Delmonte, N. K-Centroid Convergence Clustering Identification in One-Label per Type for Disease Prediction. *IAES Int. J. Artif. Intell.* **2024**, *13*, 1149–1159. [CrossRef]
11. Jara-Díaz, S.R.; Rosales-Salas, J. Time Use: The Role of Sleep. *Transp. Res. Part A Policy Pract.* **2020**, *136*, 1–20. [CrossRef]
12. National Institute on Aging a Good Night's Sleep. Available online: https://www.nia.nih.gov/health/sleep/good-nights-sleep/ (accessed on 25 January 2024).
13. Suni, E.; Wright, H. What Causes Restless Sleep? Available online: https://www.sleepfoundation.org/how-sleep-works/what-causes-restless-sleep (accessed on 25 January 2024).

14. Mendez, M.O.; Matteucci, M.; Cerutti, S.; Bianchi, A.M.; Kortelainen, J.M. Automatic Detection of Sleep Macrostructure Based on Bed Sensors. In Proceedings of the 2009 Annual International Conference of the IEEE Engineering in Medicine and Biology Society, Minneapolis, MN, USA, 3–6 September 2009. [CrossRef]
15. Guerrero, G.; Kortelainen, J.M.; Palacios, E.; Bianchi, A.M.; Tachino, G.; Tenhunen, M.; Mendez, M.O.; van Gils, M. Detection of Sleep-Disordered Breating with Pressure Bed Sensor. In Proceedings of the 2013 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), Osaka, Japan, 3–7 July 2013. [CrossRef]
16. Intongkum, C.; Sasiwat, Y.; Sengchuai, K.; Booranawong, A.; Phukpattaranont, P. Monitoring and Classification of Human Sleep Postures, Seizures, and Falls from Bed Using Three-Axis Acceleration Signals and Machine Learning. *SN Comput. Sci.* **2023**, *5*, 104. [CrossRef]
17. Sundararajan, K.; Georgievska, S.; te Lindert, B.H.W.; Gehrman, P.R.; Ramautar, J.; Mazzotti, D.R.; Sabia, S.; Weedon, M.N.; van Someren, E.J.W.; Ridder, L.; et al. Sleep Classification from Wrist-Worn Accelerometer Data Using Random Forests. *Sci. Rep.* **2021**, *11*, 24. [CrossRef]
18. Kuzik, N.; Spence, J.C.; Carson, V. Machine Learning Sleep Duration Classification in Preschoolers Using Waist-Worn ActiGraphs. *Sleep Med.* **2021**, *78*, 141–148. [CrossRef]
19. Navarro, M.; Nicdao, A.I.; Dela, J.C. Machine Learning Based Sleep Phase Monitoring Using Pulse Oximeter and Accelerometer. In Proceedings of the 5th International Conference on Electrical, Telecommunication and Computer Engineering (ELTICOM), Medan, Indonesia, 15–16 September 2021. [CrossRef]
20. Chen, Z.; Wu, M.; Wu, J.; Ding, J.; Zeng, Z.; Surmacz, K.; Li, X. A Deep Learning Approach for Sleep-Wake Detection from HRV and Accelerometer Data. In Proceedings of the IEEE EMBS International Conference on Biomedical & Health Informatics (BHII), Chicago, IL, USA, 19–22 May 2019. [CrossRef]
21. INTEX Sleep Tracking in Smartwatches—Everything You Need to Know! Available online: https://www.intex.in/blogs/blogs/sleep-tracking-in-smartwatches-everything-you-need-to-know (accessed on 10 December 2023).
22. Miotto, R.; Wang, F.; Wang, S.; Jiang, X.; Dudley, J.T. Deep Learning for Healthcare: Review, Opportunities and Challenges. *Brief. Bioinform.* **2018**, *19*, 1236–1246. [CrossRef] [PubMed]
23. Shamshirband, S.; Fathi, M.; Dehzangi, A.; Chronopoulos, A.T.; Alinejad-Rokny, H. A Review on Deep Learning Approaches in Healthcare Systems: Taxonomies, Challenges, and Open Issues. *J. Biomed. Inform.* **2020**, *113*, 103627. [CrossRef] [PubMed]
24. Shung, D.; Huang, J.; Castro, E.; Tay, J.K.; Simonov, M.; Laine, L.; Batra, R.; Krishnaswamy, S. Neural Network Predicts Need for Red Blood Cell Transfusion for Patients with Acute Gastrointestinal Bleeding Admitted to the Intensive Care Unit. *Sci. Rep.* **2021**, *11*, 8827. [CrossRef] [PubMed]
25. Shi, J.; Ye, M.; Chen, H.; Lu, Y.; Tan, Z.; Fan, Z.; Zhao, J. Enhancing Efficiency and Capacity of Telehealth Services with Intelligent Triage: A Bidirectional LSTM Neural Network Model Employing Character Embedding. *BMC Med. Inform. Decis. Mak.* **2023**, *23*, 269. [CrossRef] [PubMed]
26. Dhananjay, B.; Venkatesh, N.P.; Bhardwaj, A.; Sivaraman, J. Cardiac Signals Classification Based on Extra Trees Model. In Proceedings of the 8th International Conference on Signal Processing and Integrated Networks (SPIN), Greater Noida, India, 26–27 August 2021. [CrossRef]
27. Upadhyay, R.; Tanwar, P.S.; Degadwala, S. Fracture Type Identification Using Extra Tree Classifier. In Proceedings of the 2021 Fifth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), Palladam, India, 11–13 November 2021. [CrossRef]
28. Wahid, N.; Zaidi, A.; Dhiman, G.; Manwal, M.; Soni, D.; Maaliw, R.R. Identification of Coronary Artery Disease Using Extra Tree Classification. Available online: https://ieeexplore.ieee.org/abstract/document/10134338/ (accessed on 13 June 2023).
29. Majumder, A.B.; Gupta, S.; Singh, D.; Acharya, B.; Gerogiannis, V.C.; Kanavos, A.; Pintelas, P. Heart Disease Prediction Using Concatenated Hybrid Ensemble Classifiers. *Algorithms* **2023**, *16*, 538. [CrossRef]
30. Choi, Y.; Boo, Y. Comparing Logistic Regression Models with Alternative Machine Learning Methods to Predict the Risk of Drug Intoxication Mortality. *Int. J. Environ. Res. Public Health* **2020**, *17*, 897. [CrossRef] [PubMed]
31. Prakhar, J.; Haider, M.T.U. Automated Detection of Biases within the Healthcare System Using Clustering and Logistic Re-gression. In Proceedings of the 2023 15th International Conference on Computer and Automation Engineering (ICCAE), Sydney, Australia, 3–5 March 2023.
32. Adil, S.H.; Ebrahim, M.; Raza, K.; Azhar Ali, S.S.; Ahmed Hashmani, M. Liver Patient Classification Using Logistic Regression. Available online: https://ieeexplore.ieee.org/abstract/document/8510581 (accessed on 17 February 2023).
33. Adebiyi, M.O.; Arowolo, M.O.; Mshelia, M.D.; Olugbara, O.O. A Linear Discriminant Analysis and Classification Model for Breast Cancer Diagnosis. *Appl. Sci.* **2022**, *12*, 11455. [CrossRef]
34. Gaudenzi, P.; Nardi, D.; Chiappetta, I.; Atek, S.; Lampani, L.; Pasquali, M.; Sarasini, F.; Tirilló, J.; Valente, T. Sparse sensing detection of impact-induced delaminations in composite laminates. *Compos. Struct.* **2015**, *133*, 1209–1219. [CrossRef]
35. Pathak, S.; Mishra, I.; Swetapadma, A. An Assessment of Decision Tree Based Classification and Regression Algorithms. In Proceedings of the 2018 3rd International Conference on Inventive Computation Technologies (ICICT), Coimbatore, India, 15–16 November 2018. [CrossRef]
36. Pereira, S.; Karia, D. Prediction of Sudden Cardiac Death Using Classification and Regression Tree Model with Coalesced Based ECG and Clinical Data. In Proceedings of the 2018 3rd International Conference on Communication and Electronics Systems (ICCES), Coimbatore, India, 15–16 November 2018.

37. Lemon, S.C.; Roy, J.; Clark, M.A.; Friedmann, P.D.; Rakowski, W. Classification and Regression Tree Analysis in Public Health: Methodological Review and Comparison with Logistic Regression. *Ann. Behav. Med.* **2003**, *26*, 172–181. [CrossRef]
38. Ozcan, M.; Peker, S. A Classification and Regression Tree Algorithm for Heart Disease Modeling and Prediction. *Healthc. Anal.* **2023**, *3*, 100130. [CrossRef]
39. Kuhn, L.; Page, K.; Ward, J.; Worrall-Carter, L. The Process and Utility of Classification and Regression Tree Methodology in Nursing Research. *J. Adv. Nurs.* **2013**, *70*, 1276–1286. [CrossRef]
40. Speybroeck, N. Classification and Regression Trees. *Int. J. Public Health* **2011**, *57*, 243–246. [CrossRef]
41. Scikit-Learn. Support Vector Machine. Available online: https://scikit-learn.org/stable/modules/svm.html (accessed on 24 August 2023).
42. Martinez-Alanis, M.; Bojorges-Valdez, E.; Wessel, N.; Lerma, C. Prediction of Sudden Cardiac Death Risk with a Support Vector Machine Based on Heart Rate Variability and Heartprint Indices. *Sensors* **2020**, *20*, 5483. [CrossRef]
43. Ye, Y.; He, W.; Cheng, Y.; Huang, W.; Zhang, Z. A Robust Random Forest-Based Approach for Heart Rate Monitoring Using Photoplethysmography Signal Contaminated by Intense Motion Artifacts. *Sensors* **2017**, *17*, 385. [CrossRef]
44. Scikit-Learn. Sklearn. Ensemble. RandomForestClassifier. Available online: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html (accessed on 24 August 2023).
45. Mbonyinshuti, F.; Nkurunziza, J.; Niyobuhungiro, J.; Kayitare, E. Application of Random Forest Model to Predict the Demand of Essential Medicines for Noncommunicable Diseases Management in Public Health Facilities. *Pan Afr. Med. J.* **2022**, *42*, 89. [CrossRef]
46. ST STM32 32-Bit Arm Cortex MCUs—STMicroelectronics. Available online: https://www.st.com/en/microcontrollers-microprocessors/stm32-32-bit-arm-cortex-mcus.html (accessed on 25 January 2024).
47. ST B-L475E-IOT01A—STMicroelectronics. Available online: https://www.st.com/en/evaluation-tools/b-l475e-iot01a.html (accessed on 22 September 2023).
48. Python. Available online: https://www.python.org/ (accessed on 21 September 2023).
49. Scikit-Learn: Machine Learning in Python. Available online: https://scikit-learn.org/stable/ (accessed on 1 November 2023).
50. Ritchie Dennis, M.; Kernighan, B.W. *The C Programming Language*; Prentice Hall: Hoboken, NJ, USA, 1988.
51. Ryan, R.R.; Spiller, H. The c Programming Language and a c Compiler. *IBM Syst. J.* **1985**, *24*, 37–48. [CrossRef]
52. ANALOG DEVICES ADXL355 Datasheet and Product Info | Analog Devices. Available online: https://www.analog.com/en/products/adxl355.html#product-documentation (accessed on 5 October 2023).
53. MQTT—The Standard for IoT Messaging. Available online: https://mqtt.org/ (accessed on 20 January 2024).
54. Manowska, A.; Wycisk, A.; Nowrot, A.; Pielot, J. The Use of the MQTT Protocol in Measurement, Monitoring and Control Systems as Part of the Implementation of Energy Management Systems. *Electronics* **2023**, *12*, 17. [CrossRef]
55. D'Ortona, C.; Tarchi, D.; Raffaelli, C. Open-Source MQTT-Based End-to-End IoT System for Smart City Scenarios. *Future Internet* **2022**, *14*, 57. [CrossRef]
56. Shahri, E.; Pedreiras, P.; Almeida, L. Extending MQTT with Real-Time Communication Services Based on SDN. *Sensors* **2022**, *22*, 3162. [CrossRef]
57. HiveMQ MQTT Essentials—All Core Concepts Explained. Available online: https://www.hivemq.com/mqtt/ (accessed on 25 January 2024).
58. OpenJS Foundation Node-RED. Available online: https://nodered.org/ (accessed on 24 September 2023).
59. Torres, D.; Dias, J.P.; Restivo, A.; Ferreira, H.S. Real-Time Feedback in Node-RED for IoT Development: An Empirical Study. Available online: https://ieeexplore.ieee.org/abstract/document/9213544 (accessed on 15 January 2024).
60. Lekic, M.; Gardasevic, G. IoT Sensor Integration to Node-RED Platform. In Proceedings of the 2018 17th International Symposium INFOTEH-JAHORINA (INFOTEH), East Sarajevo, Bosnia and Herzegovina, 21–23 March 2018. [CrossRef]
61. Thomas, L.; Mv, M.K.; Sl, S.D.; Bs, P. Towards Comprehensive Home Automation: Leveraging the IoT, Node-RED, and Wireless Sensor Networks for Enhanced Control and Connectivity. *Eng. Proc.* **2023**, *59*, 173. [CrossRef]
62. Medina-Pérez, A.; Sánchez-Rodríguez, D.; Alonso-González, I. An Internet of Thing Architecture Based on Message Queuing Telemetry Transport Protocol and Node-RED: A Case Study for Monitoring Radon Gas. *Smart Cities* **2021**, *4*, 803–818. [CrossRef]