IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Traitement du Signal
Vol. 43, No. 2, April, 2026, pp. 687-695
Journal homepage: http://iieta.org/journals/ts

Signal-Driven Energy Expenditure Estimation Using Wearable Physiological Sensors: A Comparative Study with Edge-Oriented Deployment Analysis

Ruikai Chen

College of Sports and Health, Sanming University, Sanming 365004, China

Corresponding Author Email: chenruikai9988@163.com

https://doi.org/10.18280/ts.430211

Received: 22 October 2025
Revised: 9 February 2026
Accepted: 19 February 2026
Available online: 30 April 2026

Keywords:
physiological signal processing, wearable sensors, energy expenditure estimation, machine learning, edge computing, IoT systems, feature modeling

ABSTRACT

Energy expenditure (EE) estimation based on wearable physiological signals has become an essential component of intelligent health monitoring systems. However, the deployment of such models on resource-constrained devices requires a careful balance between signal representation and computational efficiency. This study investigates EE estimation using multi-source physiological signals, including heart rate, body temperature, and activity duration, collected in a campus Internet of Things (IoT) environment. A signal-oriented modeling framework is adopted, in which sensor signals are standardized prior to regression modeling. Nine representative machine learning algorithms are evaluated from both predictive and deployment perspectives. The results show that ensemble models achieve the highest predictive accuracy, with Random Forest reaching an R^2 of 0.9973 and a mean absolute error of 3.52 kcal. In contrast, linear regression achieves comparable accuracy (R^2 = 0.9963) while requiring significantly lower computational resources, with a model size below 1 KB and microsecond-level inference latency. Further analysis incorporating computational complexity and robustness to signal perturbations demonstrates that model selection should be guided not only by accuracy but also by signal characteristics and deployment constraints. The findings provide practical insight into the design of lightweight, signal-driven EE estimation systems for wearable and edge-computing environments.

1. INTRODUCTION

With the rapid development of Internet of Things (IoT) technology and wearable devices, sensor-driven human activity monitoring and health management have become integral components of smart campus infrastructure [1, 2]. Wearable devices such as smartbands and sports watches enable real-time acquisition of physiological signals, including heart rate, body temperature, and motion acceleration, providing a data foundation for accurate estimation of exercise energy expenditure (EE) [3]. Accurate EE estimation is essential for evidence-based physical activity guidance, personalized exercise prescription, and quantitative support for institutional health management decisions.

In campus IoT scenarios, wearable devices are typically characterized by limited computational resources, strict power constraints, and restricted storage capacity. Consequently, the edge-deployment capability of machine learning models becomes a critical factor determining system practicality [4]. Traditional EE estimation relies primarily on physiology-based regression formulae (e.g., the Keytel equation) [5], which are computationally simple but limited in accuracy and unable to accommodate inter-individual variability and diverse exercise modalities. In recent years, machine learning methods have demonstrated significant advantages in EE prediction owing to their powerful nonlinear fitting capabilities [6]; however, existing studies predominantly focus on single algorithms or limited algorithm comparisons and lack systematic evaluation oriented toward edge deployment. Furthermore, the No Free Lunch theorem [7] implies that no single algorithm universally dominates across all problem domains, underscoring the necessity of task-specific empirical comparison.

The main contributions of this study are as follows: (1) A systematic evaluation of nine mainstream machine learning regression algorithms for EE prediction from wearable sensor signals; (2) A five-dimensional edge-deployment evaluation framework encompassing prediction accuracy, inference latency, model size, computational complexity, and noise robustness; (3) Statistical significance analysis of algorithm rankings via the Friedman test and Nemenyi post-hoc test; (4) Deployment-specific algorithm recommendations under varying resource constraints. Additionally, a supplementary classification experiment comparing eight algorithms on a physical fitness grade assessment task is presented.

2. RELATED WORK

2.1 Exercise energy expenditure estimation

Exercise EE estimation is a central problem in exercise science and health management. Traditional approaches include: (1) physiology-model-based formulae exploiting the

linear relationship between heart rate and oxygen consumption [5]; (2) accelerometer-based activity intensity classification [8]; and (3) multi-sensor fusion methods for metabolic equivalent (MET) estimation [9]. Although computationally simple and interpretable, these methods exhibit limited accuracy and generalizability.

Advances in wearable sensor technology and machine learning have catalyzed a growing body of research applying ML algorithms to EE prediction. O'Driscoll et al. [6] compared the validity and generalizability of Random Forest, Gradient Boosting, neural networks, and linear regression for EE prediction using wrist-worn device data, finding that while ML methods showed promise, they did not consistently outperform linear regression. Altini et al. [10] investigated the effect of sensor placement and number on EE estimation accuracy using body-worn accelerometers. Staudenmayer et al. [11] employed an artificial neural network to estimate EE and identify physical activity types from accelerometer data. Montoye et al. [12] demonstrated that raw accelerometer data combined with machine learning can yield accurate free-living EE predictions. Zhu et al. [13] applied deep learning (convolutional neural networks) to wearable sensor data for EE estimation, while Cvetković et al. [14] proposed a real-time activity monitoring and EE estimation algorithm fusing wristband and smartphone sensor signals. However, the aforementioned studies are largely confined to comparisons of a few algorithms and lack systematic edge-deployment evaluation.

2.2 Edge computing and lightweight machine learning

Edge computing is a paradigm that migrates computational tasks from the cloud to network-edge devices, effectively reducing inference latency, protecting data privacy, and decreasing bandwidth requirements [4]. On wearable devices, where microcontrollers offer severely constrained compute and storage resources (e.g., ARM Cortex-M4 with 256 KB Flash and 64 KB RAM), machine learning models must satisfy stringent low-latency, small-footprint, and low-FLOPs deployment constraints [15].

The emergence of TinyML (tiny machine learning) has propelled lightweight model deployment on edge devices [15, 16]. Existing research indicates that, for low-dimensional sensor data, classical ML algorithms (e.g., linear regression, decision trees) can maintain competitive accuracy while satisfying edge-device resource constraints. Shwartz-Ziv and Armon [17] demonstrated in a systematic comparison that tree-ensemble models generally outperform deep learning models on tabular data with lower hyperparameter tuning costs. These findings motivate the present study's focus on classical ML algorithm comparison from an edge-deployment practicality perspective.

2.3 Machine learning in physical fitness assessment

Machine learning applications in student physical fitness assessment have attracted increasing attention. Prior studies have employed Random Forest [18], support vector machines, and other classifiers for health-grade evaluation of physical fitness test data, achieving promising results. However, existing work predominantly adopts single-algorithm or limited-algorithm comparisons, lacking comprehensive analysis spanning multiple algorithm families (linear models, tree models, ensemble methods, kernel methods, and distance-based models). As the No Free Lunch theorem [7] suggests, empirical evaluation across diverse algorithm types is essential. The supplementary classification experiment in this study addresses this gap.

3. METHODOLOGY

3.1 Data sources and description

3.1.1 Exercise energy expenditure dataset (Primary experiment)

The primary experiment uses an exercise EE dataset comprising 80 exercise monitoring records that simulate physiological signals and motion parameters collected by campus wearable devices (smartbands). The dataset was originally sourced from a public Kaggle repository and subsampled for rapid experimental validation. It contains eight feature variables and one target variable (calories burned), as described in Table 1.

Table 1. Description of variables in the exercise energy expenditure (EE) dataset

Variable	Description	Type	Unit
Duration	Exercise duration	Sensor signal	min
Heart_Rate	Mean heart rate	Sensor signal	bpm
Body_Temp	Body temperature	Sensor signal	°C
Age	Age	Demographic	years
Height	Height	Demographic	cm
Weight	Weight	Demographic	kg
Gender	Gender	Demographic	M/F
BMI	Body mass index (derived)	Derived	kg/m²
Calories	Calories burned (target)	Target	kcal

Table 2 presents the descriptive statistics of sensor signals and the target variable. Exercise duration ranges from 25 to 60 minutes, heart rate from 105 to 155 bpm, body temperature from 37.2 to 37.8 ℃, and caloric expenditure from 132.45 to 438.83 kcal. The coefficients of variation (CV) indicate substantial inter-individual variability in exercise duration and caloric expenditure (CV > 25%), whereas body temperature exhibits minimal dispersion (CV = 0.46%).

Table 2. Descriptive statistics of sensor signals and target variable

Variable	Mean	Std	Min	Max	Range	CV (%)
Duration	40.63	10.24	25.0	60.0	35.0	25.21
Heart_Rate	129.68	14.50	105.0	155.0	50.0	11.18
Body_Temp	37.51	0.17	37.2	37.8	0.60	0.46
Calories	259.97	92.27	132.45	438.83	306.38	35.49

Note: CV = Coefficients of variation

3.1.2 Physical fitness dataset (Supplementary experiment)

The supplementary classification experiment uses a physical fitness test dataset containing 80 student records with 10 feature variables and one target variable (fitness grade: A/B/C/D). Features include basic anthropometric measures (age, gender, height, weight, body fat percentage, BMI) and physical performance indicators (grip strength, sit-and-reach, sit-ups, standing long jump).

3.2 Data preprocessing

The data preprocessing pipeline comprises: (1) Data cleaning—verification and handling of missing values and outliers to ensure data integrity; (2) Feature engineering—computation of the derived BMI feature (BMI = weight/height²) and label encoding of the gender variable (Male = 1, Female = 0); (3) Feature standardization—Z-score normalization via StandardScaler for algorithms requiring standardized inputs (SVR, KNN); (4) Data partitioning—a 75%/25% train–test split (random_state = 42 for reproducibility), with stratified sampling applied to the classification task to preserve class proportions.

3.3 Algorithm selection

3.3.1 Regression algorithms (Primary experiment)

Nine regression algorithms spanning five major algorithm families were selected to ensure a comprehensive and representative comparison. These include: Linear Regression and its regularized variants Ridge [19] and Lasso [20] from the linear model family; Decision Tree Regression based on recursive partitioning [21]; Random Forest employing Breiman's bagging ensemble strategy [18]; Gradient Boosting Regression based on the sequential boosting framework of Friedman [22], which was further extended in scalable systems such as XGBoost [23]; AdaBoost based on the adaptive boosting algorithm of Freund and Schapire [24]; Support Vector Regression (SVR) based on the structural risk minimization principle [25]; and K-Nearest Neighbors (KNN) Regression [26]. The specific hyperparameter settings are listed in Table 3.

Table 3. Regression algorithms and hyperparameter settings

Algorithm	Family	Key Hyperparameters
Linear Regression	Linear model	Default
Ridge Regression	Regularized linear	α = 1.0
Lasso Regression	Regularized linear	α = 1.0
Decision Tree	Tree model	max_depth = 10
Random Forest	Ensemble (Bagging)	n_estimators = 100, max_depth = 10
Gradient Boosting	Ensemble (Boosting)	n_estimators = 100, max_depth = 5
AdaBoost	Ensemble (Boosting)	n_estimators = 50
SVR	Kernel method	kernel = RBF
KNN	Distance-based	n_neighbors = 5

Note: SVR = Support Vector Regression; KNN = K-Nearest Neighbors

3.3.2 Classification algorithms (Supplementary experiment)

Eight classification algorithms were selected for the supplementary experiment: Decision Tree, Random Forest, Gradient Boosting, AdaBoost, Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Naïve Bayes, and Logistic Regression, covering tree models, ensemble methods, kernel methods, distance-based models, probabilistic models, and linear models.

3.4 Evaluation framework

3.4.1 Prediction accuracy metrics

Four metrics are employed to evaluate regression prediction accuracy. Let y_i denote the actual value of the i-th sample, $\hat{y}_i$ the predicted value, $\bar{y}$ the mean of actual values, and n the number of test samples.

The coefficient of determination R^2 quantifies the proportion of variance explained by the model (range $(-\infty, 1]$, higher is better):

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \quad (1)$$

The root mean square error (RMSE) reflects the average magnitude of prediction deviations:

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \quad (2)$$

The mean absolute error (MAE) represents the average absolute prediction error (unit: kcal):

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \quad (3)$$

The mean absolute percentage error (MAPE) captures relative prediction accuracy in percentage form:

$$\mathrm{MAPE} = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \quad (4)$$

Classification metrics include accuracy, weighted F1-score, and 5-fold cross-validation score.

3.4.2 Edge-deployment metrics

Three computational-efficiency metrics are introduced to address wearable-device edge-deployment requirements: (1) Inference latency—the mean time for single-sample inference (μs), measured as the average of 1,000 repeated inferences; (2) Model size—the serialized model size (KB) via Python pickle, reflecting deployment feasibility on storage-constrained devices; (3) Theoretical FLOPs—the number of floating-point operations per single-sample inference, estimated from each algorithm's computational principles.

3.4.3 Noise robustness evaluation

Wearable sensors inevitably suffer from noise interference during actual use. Seven noise conditions (including baseline) are designed to systematically assess performance degradation under sensor noise perturbation: Gaussian noise on heart rate (σ = 5, 10 bpm), uniform noise on body temperature (±0.3, ±0.5 °C), and their combinations.

3.4.4 Statistical significance testing

To ensure the statistical reliability of algorithm-comparison conclusions, the Friedman test is employed to determine whether significant performance differences exist among multiple algorithms. If the Friedman test rejects the null hypothesis ($p < 0.05$), the Nemenyi post-hoc test is subsequently applied to compute the critical difference (CD), and a CD diagram is used to visualize statistical significance groupings [27, 28]. Cross-validation adopts a 5 × 3 repeated K-fold strategy (15 folds total) to enhance statistical robustness [29].

3.5 Experimental environment

All experiments were conducted in Python 3.8 using scikit-learn 0.24+, pandas, NumPy, Matplotlib, and SciPy on

standard CPU hardware (no GPU acceleration required) [30]. All random seeds were fixed at 42 to ensure reproducibility.

4. EXPERIMENTAL RESULTS AND ANALYSIS

4.1 Correlation analysis

Figure 1 presents the Pearson correlation coefficient matrix between feature variables and the target variable (Calories). Exercise duration (Duration) exhibits the strongest correlation with caloric expenditure ($r > 0.9$), followed by heart rate (Heart_Rate, $r > 0.8$) and body temperature (Body_Temp, moderate positive correlation). Demographic features (Age, Height, Weight, BMI, Gender) show relatively weaker correlations but still provide valuable individual difference information. The strong linear correlations provide an explanatory basis for the strong performance of linear models.

4.2 Regression algorithm comparison (Primary experiment)

4.2.1 Prediction accuracy

Table 4 presents the comprehensive performance comparison of nine regression algorithms on the exercise EE prediction task, sorted by descending R^2 score (Figure 2).

Random Forest Regression achieves the best prediction accuracy across all metrics: $R^2 = 0.9973$, MAE = 3.52 kcal, MAPE = 1.37%. For an exercise scenario with a mean expenditure of 260 kcal, the prediction error is less than 3.5 kcal. Gradient Boosting follows closely ($R^2 = 0.9965$), while Linear Regression ranks third with $R^2 = 0.9963$—only 0.001 below Random Forest. Notably, the strong performance of simple Linear Regression is attributable to the pronounced linear correlations between EE and input features, particularly exercise duration and heart rate.

SVR exhibits substantially inferior performance ($R^2 = 0.34$, MAPE = 22.14%), primarily due to: (1) unoptimized default RBF kernel hyperparameters; (2) the low feature dimensionality and strong linear correlations being ill-suited to kernel-based nonlinear mapping; and (3) SVR's high sensitivity to hyperparameters under small-sample conditions.

4.2.2 Computational complexity and edge-deployment analysis

Table 5 presents the theoretical FLOPs estimates for each algorithm (Figure 3). Linear models (Linear Regression, Ridge, Lasso) have the lowest FLOPs (17), equal to $2p + 1$ floating-point operations ($p = 8$ features), making them suitable for deployment on extremely resource-constrained microcontrollers. Random Forest and Gradient Boosting, owing to their large number of constituent trees, incur 719 and 600 FLOPs, respectively, yet these remain acceptable for modern embedded processors.

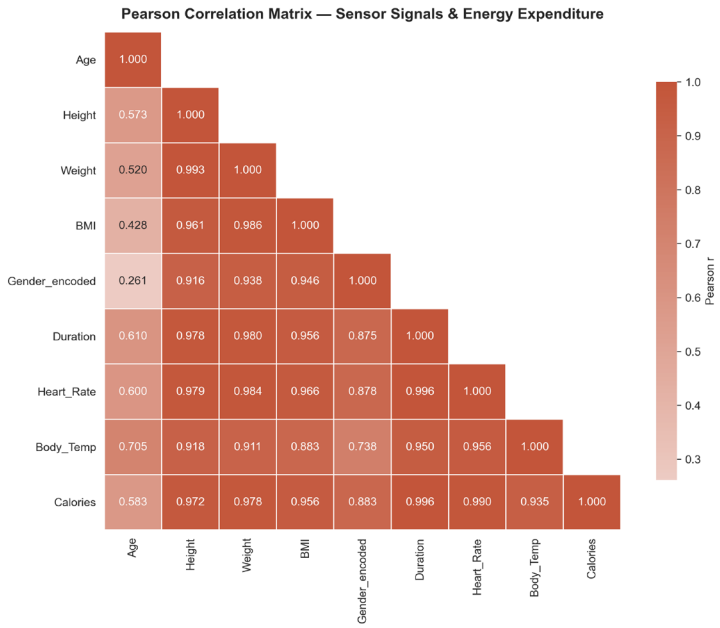

Figure 1. Pearson correlation coefficient heatmap

Table 4. Comprehensive performance comparison of regression algorithms

Algorithm	R^2	RMSE	MAE (kcal)	MAPE (%)	Train (s)	Latency (μs)	Size (KB)
Random Forest	0.9973	4.21	3.52	1.37	0.138	110115	505.3
Gradient Boosting	0.9965	4.79	4.20	1.68	0.012	42.3	386.5
Linear Regression	0.9963	4.88	4.13	1.74	0.005	14.0	0.66
Decision Tree	0.9931	6.69	5.41	2.24	0.001	23.6	8.39
Ridge	0.9912	7.57	6.13	2.43	0.002	15.6	0.61
Lasso	0.9900	8.07	6.96	2.95	0.002	19.8	0.71
AdaBoost	0.9881	8.78	6.81	2.54	0.023	938.7	64.1
K-Nearest Neighbors (KNN)	0.9707	13.80	10.95	4.15	0.001	146.4	9.87
Support Vector Regression (SVR)	0.3400	65.47	53.58	22.14	0.010	26.4	5.62

Note: RMSE = Root mean square error; MAE = Mean absolute error; MAPE = Mean absolute percentage error

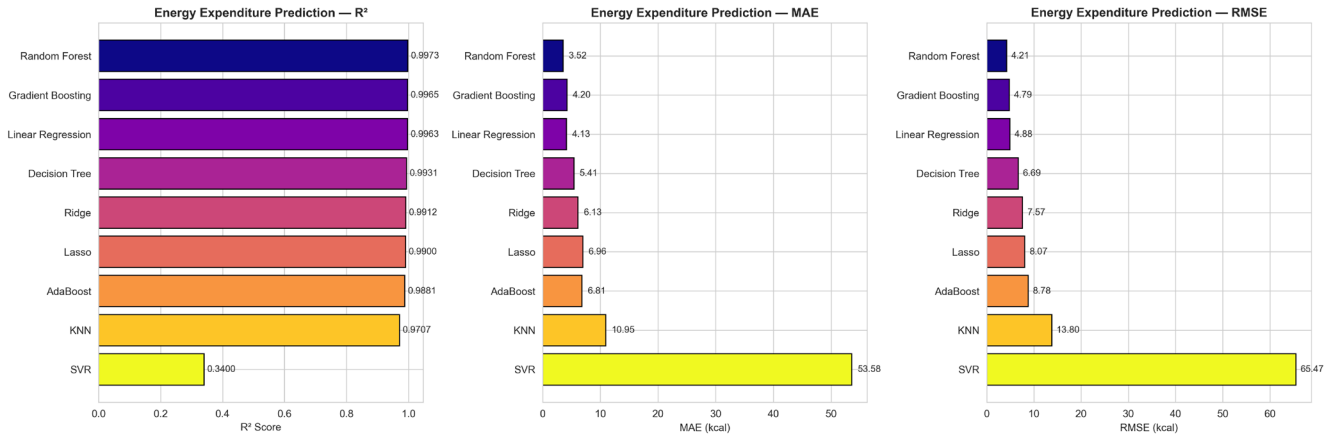

Figure 2. Regression algorithm performance comparison (R^2, mean absolute error (MAE), root mean square error (RMSE))

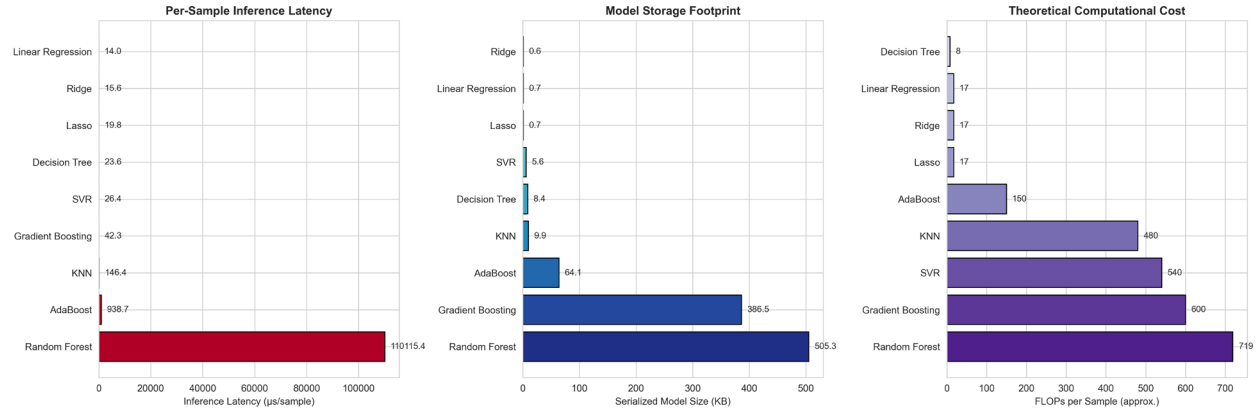

Figure 3. Computational complexity comparison (latency, model size, FLOPs)

From an edge-deployment perspective, Linear Regression is the optimal on-device model: model size = 0.66 KB, inference latency = 14 µs, FLOPs = 17, while still achieving $R^2 = 0.9963$. By contrast, Random Forest, despite its highest accuracy, requires 505.3 KB and 110 ms inference time, making it more suitable for server-side deployment. Gradient Boosting strikes a favorable balance between accuracy and computational efficiency (R^2 = 0.9965, 386.5 KB, 42 µs), appropriate for gateway-level edge devices.

4.3 Sensor noise robustness analysis

Table 6 presents the MAE variation across different sensor noise conditions (Figure 4). Under the most severe noise condition (HR ±10 bpm + Temp ±0.5 °C), performance degradation varies substantially among algorithms.

Key findings include: (1) Random Forest exhibits strong robustness under single-noise conditions, with MAE increasing controllably (baseline 3.52 → worst-case 8.81 kcal, ≈ 2.5×); (2) Linear Regression is sensitive to heart-rate noise (MAE rises from 4.13 to 14.95 under HR ±10 bpm) because heart rate, as a strongly correlated feature, directly amplifies prediction error when perturbed; (3) AdaBoost displays counter-intuitive stability under high noise, attributable to its ensemble of multiple weak learners providing a noise-buffering mechanism; (4) SVR and KNN exhibit poor noise robustness and are unsuitable for high-noise sensor environments.

Table 5. Computational complexity and deployment metrics

Algorithm	FLOPs	Formula	Latency (µs)	Size (KB)
Linear Regression	17	2p+1	14.0	0.66
Ridge	17	2p+1	15.6	0.61
Lasso	17	2p+1	19.8	0.71
Decision Tree	8	depth	23.6	8.39
AdaBoost	150	B×avg_depth	938.7	64.1
KNN	480	n_train×p	146.4	9.87
SVR	540	n_sv×(p+1)	26.4	5.62
Gradient Boosting	600	B×(avg_depth+1)	42.3	386.5
Random Forest	719	B×avg_depth	110115	505.3

Table 6. Sensor noise robustness analysis — mean absolute error (MAE) (kcal)

Noise Condition	Linear	RF	GBRT	AdaBoost	SVR	KNN
Baseline (no noise)	4.13	3.52	4.20	6.81	53.58	10.95
HR ± 5 bpm	7.78	4.50	5.44	6.13	53.71	10.61
HR ± 10 bpm	14.95	9.97	10.35	8.49	53.92	12.59
Temp ± 0.3 ℃	4.26	4.20	4.87	6.99	55.84	13.97
Temp ± 0.5 ℃	4.12	3.96	4.37	7.19	56.94	20.97
HR ± 5 + Temp ± 0.3	7.39	7.08	7.54	8.71	54.37	13.69
HR ± 10 + Temp ± 0.5	16.59	8.81	10.15	7.57	58.01	20.67

Note: Support Vector Regression (SVR); KNN: K-Nearest Neighbors

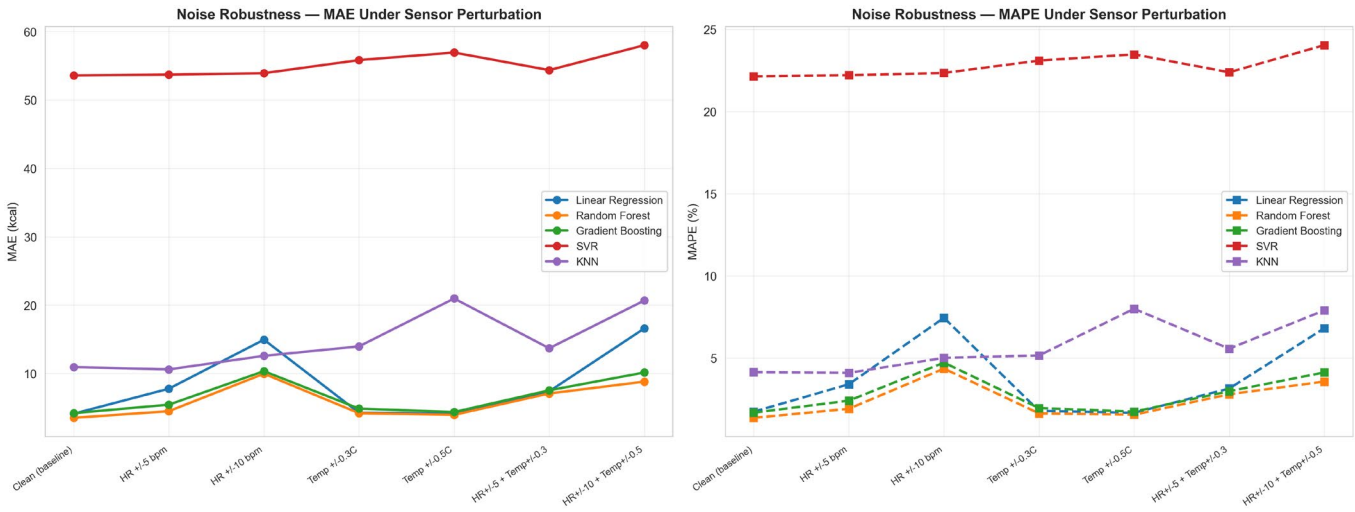

Figure 4. Sensor noise robustness analysis (mean absolute error (MAE) and mean absolute percentage error (MAPE) vs. noise condition)

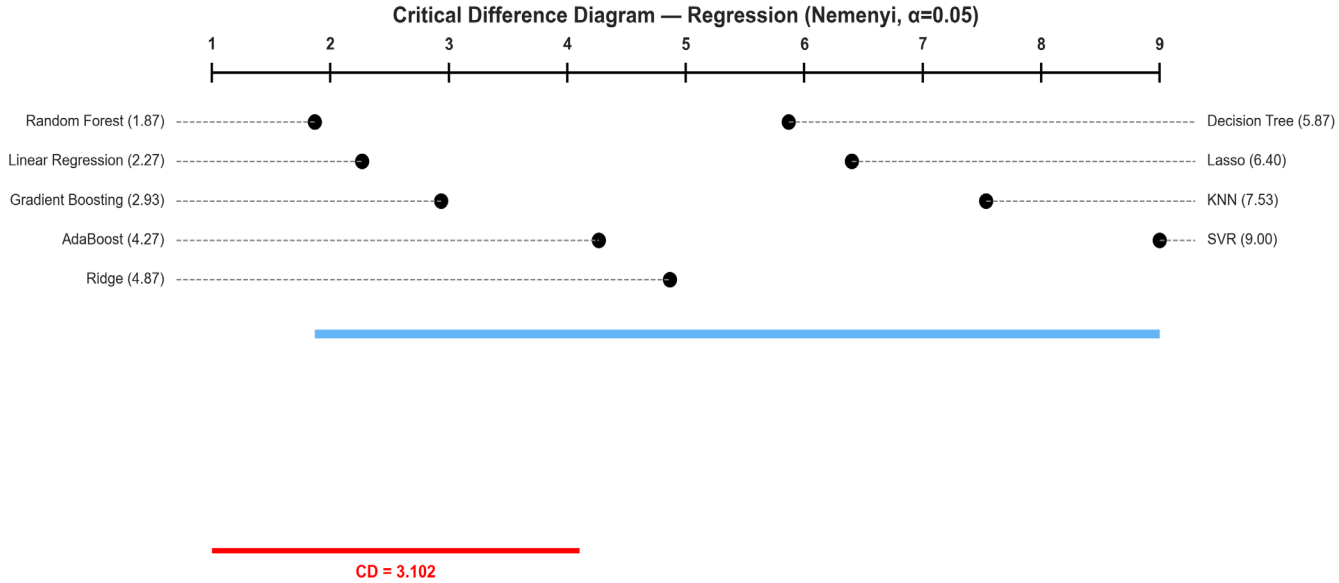

Figure 5. Critical difference diagram (Nemenyi post-hoc test, α = 0.05, CD = 3.10)

4.4 Statistical significance testing

Based on the MAE results from 15-fold (5 × 3 repeated) cross-validation, the Friedman test statistic is $\chi^2 = 94.49$ ($p = 5.69 \times 10^{-17}$), far below the significance level $\alpha = 0.05$, firmly rejecting the null hypothesis that all algorithms perform equally and confirming statistically significant performance differences among the nine algorithms (Figure 5).

Table 7. Friedman–Nemenyi test: Algorithm mean rank

Rank	Algorithm	Mean Rank
1	Random Forest	1.87
2	Linear Regression	2.27
3	Gradient Boosting	2.93
4	AdaBoost	4.27
5	Ridge	4.87
6	Decision Tree	5.87
7	Lasso	6.40
8	KNN	7.53
9	SVR	9.00

The Nemenyi post-hoc test yields a critical difference CD = 3.10 ($q_\alpha = 3.102$, k = 9, N = 15). Table 7 presents the mean-rank-based algorithm rankings. Random Forest achieves the best mean rank (1.87), followed by Linear Regression (2.27) and Gradient Boosting (2.93); the rank differences among these three do not exceed the CD value, indicating no statistically significant difference. SVR ranks last (mean rank = 9.0) and differs significantly from all top-four algorithms.

4.5 Supplementary experiment: Physical fitness classification

Table 8 presents the performance comparison of eight classification algorithms on the physical fitness grade assessment task (Figure 6). Random Forest achieves 100% test-set accuracy and an F1-score of 1.0, with a 5-fold cross-validation score of 96.25%, indicating strong generalization capability. KNN and Logistic Regression tie for second place at 95% accuracy. Naïve Bayes performs poorly (50% accuracy), as the substantial inter-feature correlations in physical fitness indicators violate its conditional independence assumption (Figure 7).

Table 8. Classification algorithm comparison (fitness grade assessment)

Algorithm	Accuracy	F1	CV Score	Std	Train (s)
Random Forest	1.0000	1.0000	0.9625	0.0500	0.119
KNN	0.9500	0.9491	0.8733	0.0670	0.000
Logistic Reg.	0.9500	0.9500	0.9250	0.0729	0.011
Decision Tree	0.9000	0.9000	0.9742	0.0317	0.001
Gradient Boost.	0.9000	0.9000	0.9625	0.0306	0.060
AdaBoost	0.9000	0.9000	0.9367	0.0685	0.027
SVM	0.9000	0.8946	0.6925	0.0220	0.001
Naïve Bayes	0.5000	0.4583	0.6008	0.0818	0.000

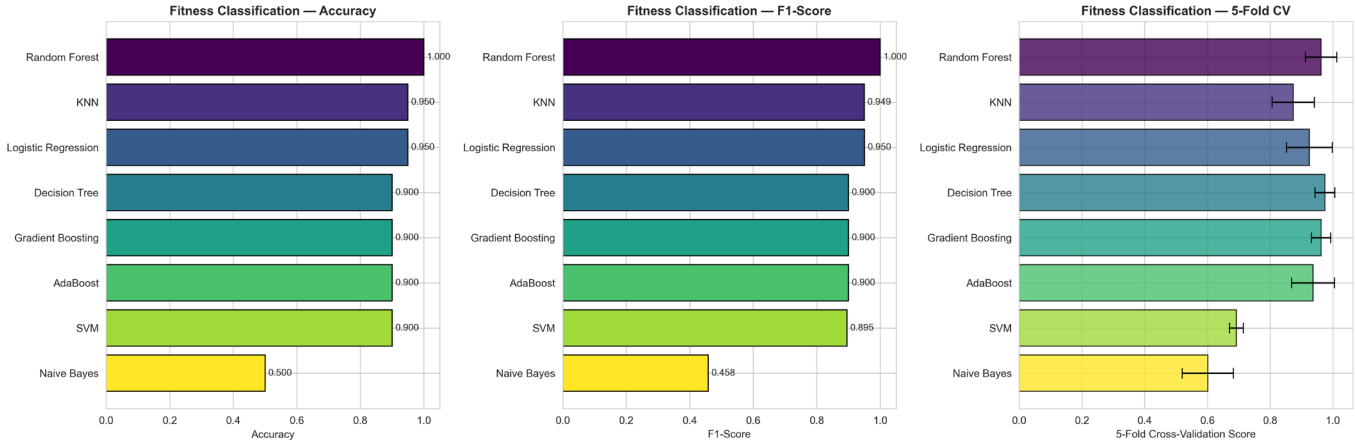

Figure 6. Classification algorithm comparison (accuracy, F1, CV score)

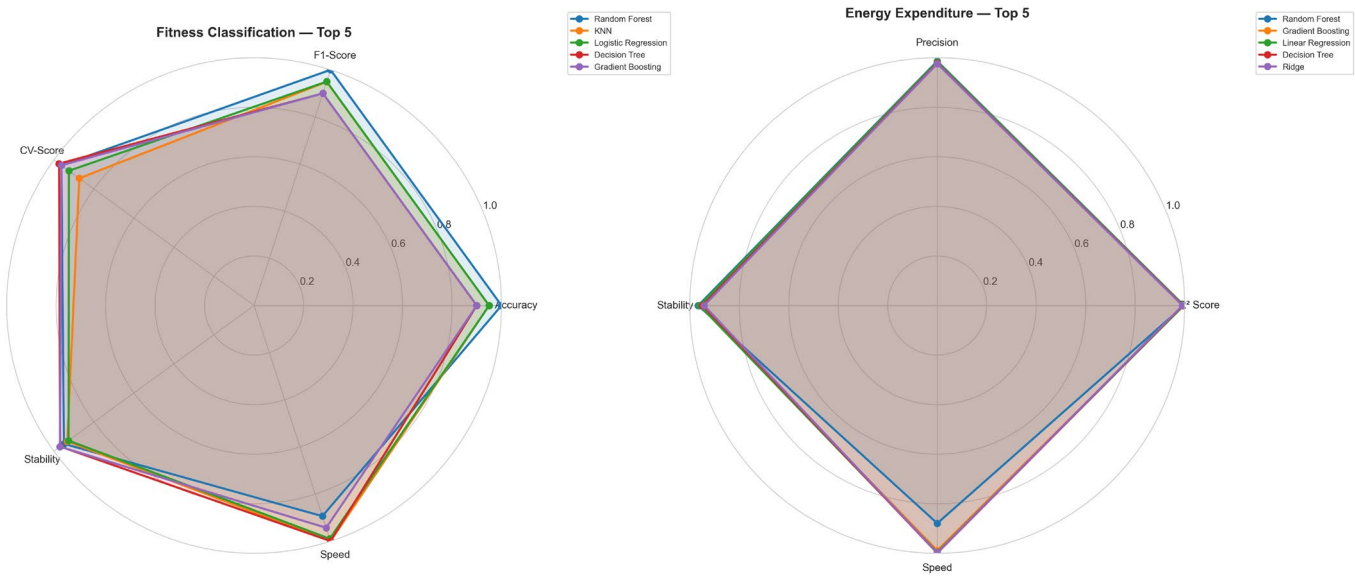

Figure 7. Comprehensive radar plots (left: classification top-5; right: regression top-5)

5. DISCUSSION

5.1 Key findings

(1) Ensemble learning algorithms excel in prediction accuracy. Random Forest, Gradient Boosting, and AdaBoost achieve R^2 = 0.9973, 0.9965, and 0.9881, respectively, in the regression task, occupying the top three positions (excluding SVR). In the classification task, Random Forest attains 100% accuracy. Bagging (Random Forest) reduces prediction variance through averaging multiple decision trees [18], while Boosting (Gradient Boosting) minimizes prediction bias via sequential optimization [22], both demonstrating powerful learning capacity.

(2) Simple linear models offer outstanding edge-deployment advantages. Linear Regression achieves R^2 = 0.9963 in the regression task—only 0.001 below Random Forest—yet its model size is 1/766th of Random Forest's (0.66 KB vs. 505.3 KB), inference latency is 1/7,865th (14 μs vs. 110 ms), and FLOPs are 1/42nd. This finding has significant practical implications for campus IoT system engineering: under the stringent resource constraints of wearable devices, Linear Regression is the preferred on-device EE prediction model.

(3) Algorithm selection should consider multiple dimensions. Table 9 summarizes deployment-specific

algorithm recommendations. Pursuing prediction accuracy alone may lead to unnecessary complexity; practical engineering requires balancing accuracy, speed, model size, robustness, and interpretability [27].

Table 9. Deployment-specific algorithm recommendations

Deployment Scenario	Recommended	Key Advantage	Resources
Wearable edge device	Linear Regression	$R^2 = 0.9963$, 0.66 KB, 14 μs	Very low
Smart gateway / edge server	Gradient Boosting	$R^2 = 0.9965$, 42 μs	Medium
Cloud / server	Random Forest	$R^2 = 0.9973$, highest accuracy	High
High-noise sensor env.	AdaBoost	Strong noise robustness	Medium
Model interpretability req.	Linear Reg. / Decision Tree	Coefficients / rules interpretable	Low

5.2 Comparison with prior work

In contrast to Zhu et al. [13], who employed only a deep learning approach, the present study encompasses nine algorithms spanning five algorithm families, providing a more comprehensive performance benchmark. Compared with the ML-based EE comparison by O'Driscoll et al. [6], this study not only examines prediction accuracy but also reveals that Linear Regression achieves comparable accuracy with markedly superior computational efficiency in edge-deployment scenarios. Furthermore, this study is among the first to incorporate edge-deployment metrics (inference latency, model size, FLOPs) and sensor noise robustness into the algorithm evaluation framework, providing more complete decision support for real-world campus IoT deployment.

5.3 Limitations and future directions

This study has several limitations: (1) The sample size is limited (80 records), and the generalizability of conclusions to large-scale datasets requires further validation; (2) The data originates from a subsampled public Kaggle dataset and may exhibit distributional differences from real campus scenarios; (3) All algorithms employ default or lightly tuned hyperparameters; systematic hyperparameter optimization (e.g., grid search, Bayesian optimization) could further improve certain algorithms' performance; (4) Deep learning methods (e.g., LSTM, Transformer) are not included in the comparison; (5) The noise models consider only Gaussian and uniform noise, without accounting for motion artifacts or other complex noise patterns.

Future research directions include: (1) Validation on larger-scale real-world campus exercise monitoring data; (2) Integration of TinyML toolchains (e.g., TensorFlow Lite Micro) for on-device deployment benchmarking on real microcontroller hardware; (3) Exploration of temporal deep learning models (LSTM, TCN) for continuous EE estimation; (4) Federated learning frameworks for privacy-preserving cross-institutional collaborative modeling; (5) Explainable AI techniques (SHAP, LIME) to analyze model decision mechanisms and provide interpretable feature-contribution information for exercise prescription.

6. CONCLUSION

Targeting the demand for real-time EE estimation in campus IoT wearable health monitoring systems, this study systematically evaluates nine machine learning regression algorithms and constructs a five-dimensional edge-deployment evaluation framework. The principal conclusions are:

(1) Random Forest Regression achieves the highest prediction accuracy ($R^2 = 0.9973$, MAE = 3.52 kcal, MAPE = 1.37%), suitable for computationally abundant cloud/server-side deployment. In the supplementary classification task, Random Forest also leads with 100% accuracy.

(2) Linear Regression is the optimal choice for edge deployment. While maintaining $R^2 = 0.9963$, its model size is only 0.66 KB, inference latency is 14 μs, and FLOPs are only 17, fully satisfying the stringent resource constraints of wearable devices.

(3) The Friedman test ($\chi^2 = 94.49$, $p < 0.001$) confirms statistically significant differences among algorithms. The Nemenyi post-hoc test (CD = 3.10) shows no significant difference among the top three algorithms (Random Forest, Linear Regression, Gradient Boosting), providing statistical support for the strategy of substituting complex ensemble models with Linear Regression on edge devices.

(4) Noise robustness analysis reveals that Random Forest maintains good stability under moderate noise, Linear Regression is sensitive to heart-rate noise, and AdaBoost exhibits unexpected robustness under high noise. Deployment decisions should account for the sensor accuracy class.

This study provides systematic empirical evidence for edge-intelligent algorithm selection in campus IoT wearable devices, offering practical reference value for advancing smart campus development and intelligent student health management.

REFERENCES

[1] Gubbi, J., Buyya, R., Marusic, S., Palaniswami, M. (2013). Internet of Things (IoT): A vision, architectural elements, and future directions. Future Generation Computer Systems, 29(7): 1645-1660. https://doi.org/10.1016/j.future.2013.01.010

[2] Islam, S.R., Kwak, D., Kabir, M.H., Hossain, M., Kwak, K.S. (2015). The internet of things for health care: A comprehensive survey. IEEE Access, 3: 678-708. https://doi.org/10.1109/ACCESS.2015.2437951

[3] Patel, S., Park, H., Bonato, P., Chan, L., Rodgers, M. (2012). A review of wearable sensors and systems with application in rehabilitation. Journal of NeuroEngineering and Rehabilitation, 9(1): 21. https://doi.org/10.1186/1743-0003-9-21

[4] Shi, W., Cao, J., Zhang, Q., Li, Y., Xu, L. (2016). Edge computing: Vision and challenges. IEEE Internet of Things Journal, 3(5): 637-646. https://doi.org/10.1109/jiot.2016.2579198

[5] Keytel, L.R., Goedecke, J.H., Noakes, T.D., Hiiloskorpi, H., Laukkanen, R., van der Merwe, L., Lambert, E.V. (2005). Prediction of energy expenditure from heart rate monitoring during submaximal exercise. Journal of Sports Sciences, 23(3): 289-297. https://doi.org/10.1080/02640410470001730089

[6] O'DRiscoll, R., Turicchi, J., Hopkins, M., Duarte, C., Horgan, G.W., Finlayson, G., Stubbs, R.J. (2021).

Comparison of the validity and generalizability of machine learning algorithms for the prediction of energy expenditure: Validation study. JMIR mHealth and uHealth, 9(8): e23938. https://doi.org/10.2196/23938

[7] Wolpert, D., Macready, W. (1997). No free lunch theorems for optimization. IEEE Transactions on Evolutionary Computation, 1(1): 67-82. https://doi.org/10.1109/4235.585893

[8] Freedson, P.S., Melanson, E., Sirard, J. (1998). Calibration of the computer science and applications, Inc. accelerometer. Medicine & Science in Sports & Exercise, 30(5): 777-781. https://doi.org/10.1097/00005768-199805000-00021

[9] Ainsworth, B.E., Haskell, W.L., Herrmann, S.D., Meckes, N., Bassett, D.R., Tudor-Locke, C., Leon, A.S. (2011). 2011 compendium of physical activities: A second update of codes and MET values. Medicine & Science in Sports & Exercise, 43(8): 1575-1581. https://doi.org/10.1249/MSS.0b013e31821ece12

[10] Altini, M., Penders, J., Vullers, R., Amft, O. (2014). Estimating energy expenditure using body-worn accelerometers: A comparison of methods, Sensors Number and Positioning. IEEE Journal of Biomedical and Health Informatics, 19(1): 219-226. https://doi.org/10.1109/jbhi.2014.2313039

[11] Staudenmayer, J., Pober, D., Crouter, S., Bassett, D., Freedson, P. (2009). An artificial neural network to estimate physical activity energy expenditure and identify physical activity type from an accelerometer. Journal of Applied Physiology, 107(4): 1300-1307. https://doi.org/10.1152/japplphysiol.00465.2009

[12] Montoye, A.H., Mudd, L.M., Biswas, S., Pfeiffer, K.A. (2015). Energy expenditure prediction using raw accelerometer data in simulated free living. Medicine & Science in Sports & Exercise, 47(8): 1735-1746. https://doi.org/10.1249/MSS.0000000000000597

[13] Zhu, J., Pande, A., Mohapatra, P., Han, J.J. (2015). Using deep learning for energy expenditure estimation with wearable sensors. In 2015 17th International Conference on E-health Networking, Application & Services (HealthCom), Boston, MA, USA, pp. 501-506. https://doi.org/10.1109/HealthCom.2015.7454554

[14] Cvetković, B., Szeklicki, R., Janko, V., Lutomski, P., Luštrek, M. (2018). Real-time activity monitoring with a wristband and a smartphone. Information Fusion, 43: 77-93. https://doi.org/10.1016/j.inffus.2017.05.004

[15] Warden, P., Situnayake, D. (2019). TinyML: Machine Learning with TensorFlow Lite on Arduino and Ultra-Low-Power Microcontrollers. O'Reilly Media.

[16] Banbury, C.R., Reddi, V.J., Lam, M., Fu, W., Fazel, A., Holleman, J., Yadav, P. (2020). Benchmarking TinyML systems: Challenges and direction. arXiv preprint arXiv:2003.04821. https://doi.org/10.48550/arXiv.2003.04821

[17] Shwartz-Ziv, R., Armon, A. (2022). Tabular data: Deep learning is not all you need. Information Fusion, 81: 84-90. https://doi.org/10.1016/j.inffus.2021.11.011

[18] Breiman, L. (2001). Random forests. Machine Learning, 45(1): 5-32. https://doi.org/10.1023/A:1010933404324

[19] Hoerl, A.E., Kennard, R.W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. Technometrics, 12(1): 55-67. https://doi.org/10.1080/00401706.1970.10488634

[20] Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society Series B: Statistical Methodology, 58(1): 267-288. https://doi.org/10.1111/j.2517-6161.1996.tb02080.x

[21] Quinlan, J.R. (1986). Induction of decision trees. Machine Learning, 1(1): 81-106. https://doi.org/10.1007/BF00116251

[22] Friedman, J.H. (2001). Greedy function approximation: A gradient boosting machine. The Annals of Statistics, 29(5): 1189-1232. https://doi.org/10.1214/aos/1013203451

[23] Chen, T., Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 785-794. https://doi.org/10.1145/2939672.2939785

[24] Freund, Y., Schapire, R.E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. Journal of Computer and System Sciences, 55(1): 119-139. https://doi.org/10.1006/jcss.1997.1504

[25] Cortes, C., Vapnik, V. (1995). Support-vector networks. Machine Learning, 20(3): 273-297. https://doi.org/10.1007/BF00994018

[26] Cover, T., Hart, P. (1967). Nearest neighbor pattern classification. IEEE Transactions on Information Theory, 13(1): 21-27. https://doi.org/10.1109/tit.1967.1053964

[27] Hastie, T., Tibshirani, R., Friedman, J. (2009). The Elements of Statistical Learning: Data Mining, Inference, and Prediction. 2nd ed., Springer, p. 567.

[28] Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research, 7: 1-30.

[29] Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. Appears in the International Joint Conference on Artificial Intelligence, 14(2): 1137-1145.

[30] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12: 2825-2830.