Tree-Based Regressors for Predicting Energy Expenditure from Heart Rate in Wearable Devices

Stephan Selinger
Department of Mobility and Energy
University of Applied Sciences Upper Austria
Hagenberg, Austria
email: stephan.selinger@fh-hagenberg.at

Luka Dimitrijević
Department of Mobility and Energy
University of Applied Sciences Upper Austria
Hagenberg, Austria
email: luka.dimitrijevic@students.fh-hagenberg.at

***Abstract*—Estimating energy expenditure from heart rate usually relies on population-based multiple linear regression equations, taking heart rate, age, gender, mass, height and, if available, VO_{2max} into account. In this paper, we show that non-linear models, such as random forests and regression trees are suited for the deployment on memory constrained wearable devices and assess physical activity more accurately than linear regression models. We fitted linear regression models, regression trees, and random forests with data from 892 graded exercise tests on a treadmill with 857 participants and evaluated their performance, as well as memory consumption on a PineTime smartwatch and an Apple Watch. A regression tree with a tree depth of 11 performed the same ($R^2 = 0.825$) as a widely used linear model by Keytel ($R^2 = 0.821$) but does not depend on VO_{2max}, which can be relevant for amateur athletes. The additional memory on the PineTime smartwach needed to store the tree increased the the original firmware size of 390 KiB to 416 KiB. If VO_{2max} is available, then a tree with a depth of 11 achieves a coefficient of 0.877, and the total memory size is 418 KiB.**

***Keywords*—energy expenditure; heart rate; regression tree; random forest regressor; wearable device**

I. Introduction

The importance of physical activity resulting in energy expenditure (EE) [1] for the prevention of non-communicable diseases, such as cardiovascular diseases and type 2 diabetes, as well as the link between exercise and longevity has long been well documented [2] [3] and validated over many decades [4]. The protective effects of exercise also enhances the immune response against bacteria and viruses [5].

Assessing EE as accurately as possible is not only relevant in the global context of health, but also for the automatic generation and adaptation of nutrition plans for athletes [6] and for individuals planning and tracking weight loss [7] [8].

Besides indirect calorimetry, heart rate measurement and accelerometry, or a combination of both, are popular methods for estimating EE [1] [9], and consequently, physical activity. Based on the assumption of a linear relationship between heart rate and oxygen consumption (VO_2), EE can be estimated from heart rate. Such a relationship is obtained by deriving a linear regression equation with EE being the dependent variable, and heart rate, age, gender, body mass, height, etc., the independent ones which can then be used to estimate VO_2 or EE in day-to-day living conditions [1].

Since this relationship is not always linear [10], it seems promising to investigate whether non-linear regression methods, such as random forests and regression trees which, from a computational point of view, are still feasible for the deployment on wearable devices, allow to more accurately predict EE rather than linear regression models.

The remainder of this paper is structured as follows: in Section II we review the related work, in Section III we describe the study design, then we continue with a discussion of results in Section IV, and conclude with Section V, in which we briefly summarize our findings and discuss possible future work.

II. Related Work

Keytel et al. [11] provide two equations for predicting EE based on heart rate, age, gender, body mass, and optionally VO_{2max}. The first equation K_1 takes into account VO_{2max}, with which the results – not surprisingly – have a higher coefficient of determination ($R^2 = 0.821$). The second one, K_2, without a fitness measure might be more universal, however, at the cost of a lower coefficient ($R^2 = 0.737$).

Additionally, taking resting heart rate into account, Charlot et al. [12] achieve a coefficient of determination which is higher ($R^2 = 0.809$) than the Keytel equation without VO_{2max}, but slightly lower than the Keytel equation with VO_{2max}. However, incorporating resting heart rate and real-time running speed resulted in $R^2 = 0.919$. Even using running speed without heart rate outperforms the Keytel equations ($R^2 = 0.913$). However, this obviously limits the applicability to only running activities.

While the previous described approaches rely solely on heart rate and linear regression, Ellis et al. use values from hip and wrist mounted accelerometers and measured heart rate to train a regression forest [13]. In addition, they also perform activity classification. In their evaluation, Ellis et al. focus on performance only and leave the question about tree depth and the number of estimators in a random forest and consequently memory consumption unanswered.

III. Tools and Methods

As a starting point and to define a baseline, we fitted two linear regression equations using *scikit-learn* (version 0.22) [14], a Python library and compared it with the approaches

 ISBN: 978-1-61208-951-5

described by Keytel et al. [11] and Charlot et al. [12]. Subsequently, we trained random forest and decision tree regressors and evaluated their performance, not only in regards to the obtained coefficients of determination and mean absolute error, but also in terms of memory demand. To that end, using an open-source code generation tool, *m2cgen* [15], we generated C-code from the previously trained regressors for the open-source smartwatch *PineTime* [16], which resembles a contemporary wearable device with an ARM Cortex-M4 CPU with 512 KiB of Flash and 64 KiB of RAM, capable of measuring heart rate based on photoplethysmography; other sensors include an accelerometer.

In addition, to investigate which coefficient of determination can be achieved on a fairly unconstrained device, we also deployed the previously trained models to an Apple Watch SE (2020) running watchOS 8.3. The watch is considered a powerful device in comparison to the PineTime because of its 1 GiB of RAM and as 32 GiB of storage capacity and a custom dual core CPU which also contains dedicated hardware which can help streamline machine learning tasks. Like the open-source watch, it is also capable of measuring heart rate with an optical sensor; an accelerometer is present here as well.

A. Participants and Graded Exercise Tests

In our study, we used a publicly available database provided by the Exercise Physiology and Human Performance Lab of the University of Malaga [17] [18] [19]. In addition to other measurements, the database contains heart rate, oxygen consumption, carbon dioxide generation, and treadmill speed from 857 amateur and professional athletes (149 females, 708 males) performing 992 graded exercise tests. After a warm-up period of 5 minutes with 5 km/h, treadmill speed was periodically increased by 0.5 or 1 km/h intervals until exhaustion after which the speed was reduced back to the initial 5 km/h. The demographic characteristics of the participants are given in Table I.

TABLE I
DEMOGRAPHIC CHARACTERISTICS.

Variable	*Range*	*Median*	*Interquartile Range*
Age (years)	10 - 63	27.1	15.2
Body mass (kg)	41 - 135	73.0	14.0
Height (cm)	150 - 203	175.0	10.0
VO_{2max} (ml/kg/min)	22.4 - 86.9	52.4	12.7

B. Fitting the Regressors

Before splitting the data into a training (75 %) and test data set (25 %) we calculated EE in KJ from O_2 and VO_2 according to the Weir formula [20]. Next, with different combinations of the independent variables listed below, we fitted two linear models to predict EE:

- LM_1: heart rate, age, weight, and gender; this is comparable to K_2.
- LM_2: same as LM_1, additionally VO_{2max}; comparable to K_1.

Regression trees were first fitted using the default hyper parameters provided by scikit-learn to determine the maximum tree depth which, as expected, exceeded the available memory on the PineTime. Therefore, we reduced the tree depth to a value that gave the same, or even a slightly better performance than linear model LM_1. From that point on, we increased the tree depth until the available memory was exhausted again. During that process we not only observed the influence of the tree depth based on our test data set, but also on the training data set to assess possible over-training (see Figure 1):

- RT_1: Regression tree with sckikit-learn default hyper parameters.
- RT_2, same as RT_1, but with a tree depth of 6 (equal or better performance than LM_1).
- RT_3, same as RT_1, but with a tree depth of 10 (equal or better performance than LM_2).
- RT_4, same as RT_1, but with a tree depth of 11 (equal or better performance than Keytel's regression equation with VO_{2max} (K_1).
- RT_5, same as RT_1, but with a tree depth of 12 (a deeper tree would exceed the available memory on the PineTime smartwatch).
- RT_6, same as RT_4, but with VO_{2max} as additional feature

As can be seen in Figure 1, a tree depth of more than 20 does not lead to a smaller mean absolute error. Furthermore it becomes visible that at a tree depth of approx. 15, errors obtained with the test data set begin to differ slightly from those obtained with the training data set, which could be a possible indication for over-training.

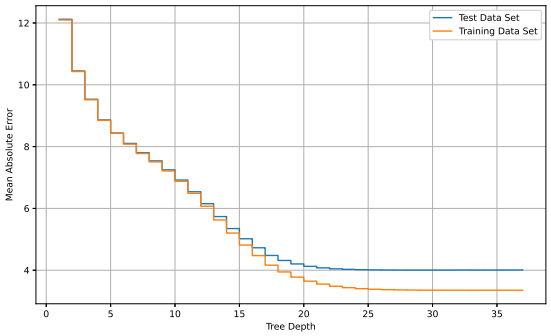

Figure 1. Tree depth vs. mean absolute error.

Along the same lines as with regression trees, we first trained a random forest with default parameters, resulting in a promising performance, yet unmanageable code size for the PineTime smartwatch:

- RF_1: Random forest with scikit-learn default parameters; heart rate, age, weight, and gender.
- RF_2: same as RF_1, tree depth of 6, 10 trees.
- RF_2: same as RF_1, tree depth of 9, 10 trees.

 ISBN: 978-1-61208-951-5

C. Code Generation and Deployment

After the models had been trained using the scikit-learn library inside a Python environment, we used m2cgen to create C code which we then cross-compiled for the C++-based InfiniTime operating system [21] (version 1.8.0) running on the PineTime smartwatch. For regression trees and random forests, m2cgen generates a function `double score(double* input)` which contains sequences of hard-coded if/else statements, therefore just consuming ROM and only little RAM.

Similiar to Sudharsan et al. [22], in an effort to reduce memory demands, all double keywords and as double literals were substituted in the generated code using regular expressions, furthermore instead of floating point numbers, we employed fixed point arithmetic numbers by multiplying with a factor of 1000, resulting in the overall use of integer numbers.

The InfiniTime OS is based on FreeRTOS and therefore employs multiple tasks, one of which is responsible for performing heart rate measurements. We therefore extended the `void HeartRateTask::Work()` method so that whenever a heart rate measurement is available, the `score()` function containing the code for the regressor is called to predict EE. As suggested by [21], we make use of puncover [23] to analyze the mapfile written by the C/C++ compiler (ARM-GCC, version 9-2020-q2-update) and determine the amount of ROM and RAM required by each regressor.

Due to the generous amount of memory, the Apple Watch is capable of running more complex models with more memory available to them. We therefore deployed RT_1 and RF_1 on the Apple Watch. The two regressors were trained using the same Python code as for the PineTime. The m2cgen code generation utility is not necessary, since the watch is programmable directly in Swift and can also use the trained models as resource files. The models need to be converted however, this is achieved using Apple's Core ML Tools library [24] [25]. The conversion is from a scikit-learn (version 0.19.2) trained model to a mlmodel file, which can then be easily integrated. In contrast to the PineTime, where code is executed in place, on the Apple Watch, once the app is executed, the model code runs in RAM to make predictions.

For a detailed analysis of the memory usage on the Apple Watch we used *Instruments* (version 13.0) as another tool inside the Xcode bundle (version 13.1) with which information can be collected regarding but not limited to memory leaks, time profiling, memory allocation statistics etc. We documented the amount of heap memory that was allocated by the watch application needed to perform a prediction, as well as the entire application size.

IV. Results and Discussion

Figure 4 compares the determined coefficients of determination. Our linear regression model LM_1 ($EE = -122.52022356 + 0.53176246hr + 0.26039323a + 0.23666578m + 0.39951689h - 7.88144777g$) is on par with Keytel's second equation (R^2 of 0.735 vs. 0.737) and also LM_2 ($EE = -103.97232241 + 0.54302212hr + 0.34344245a + 0.08775421m + 0.10690366h - 0.14505558g + 0.01021273o$) which perform quite similar (R^2 of 0.805 vs. 0.821) (EE: energy expenditure in KJ; hr: heart rate in beats/minute; a: age in years; m: body mass in kg; h: height in cm; g: gender (0 = male, 1 = female); o: VO_{2max} in ml/kg/min). Also the regression equation with basic parameters (e.g., age, height, mass, bodymass index (BMI), age-predicted HR_{max}) by Charlot et al. [12] ($R^2 = 0.809$) is similar; however, they use resting heart rate as an additional feature.

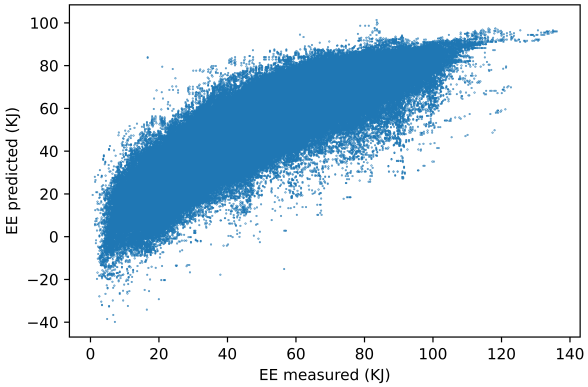

Figure 2. EE vs. predicted EE by LM_2.

RT_2, a regression tree with a depth of 6 exhibits $R^2 = 0.748$ which is similar to Keytel's equation without VO_{2max}.

RT_3 with a tree depth of 10 is comparable to LM_2. A tree size of 11 leads to a coefficient of determination of 0.825 which is almost the same as Keytel's equation K_1. However, both regression trees do not use VO_{2max} and perform the same as linear regression models, they are of particular interest to amateur athletes who do not know their VO_{2max}. Finally, increasing tree depth to 12 leads to $R^2 = 0.840$, which is better than Keytel's first equation, again without using VO_{2max}. Incorporating VO_{2max} into a tree of depth 11 gives an even higher coefficient ($R^2 = 0.877$).

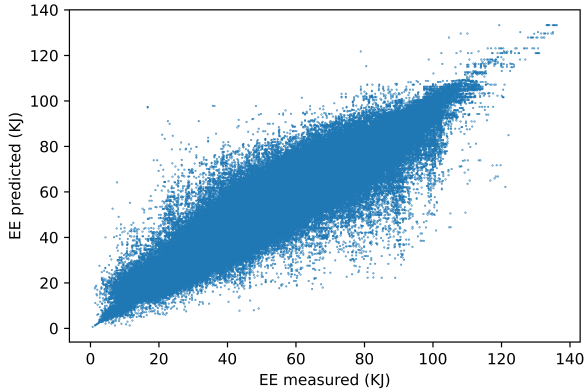

Figure 3. EE vs. predicted EE by DT_1.

The scatter plot in Figure 2 indicates that the linear model underestimates EE for submaximal and maximal efforts and

can result in negative values near resting heart rate – something that is better taken care of by the regression tree model (see Figure 3).

Random forest models, which are also less prone to overfitting, perform best. However, for the PineTime smartwatch the biggest model we could accommodate had a maximum tree depth of 6 and used 10 estimators which led to a coefficient of ($R^2 = 0.800$) which is comparable to RT_3, again with the benefit of not using VO_{2max}. Increasing the number of estimators or tree depth then exceeded the available memory.

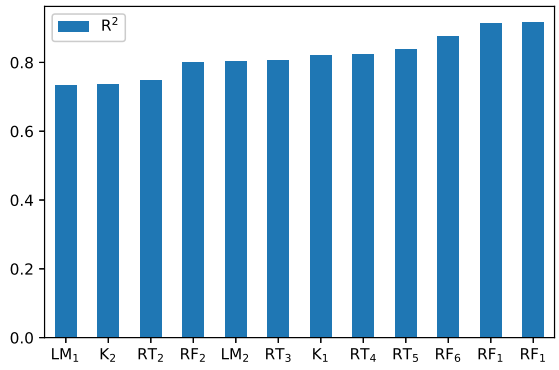

Figure 4. Coefficients of determination.

Table II summarizes the obtained regressor performances, as well as memory needs on the PineTime and Apple Watch. Despite performing more favorably than the linear models, tree-based regressors need more memory. Without any regressor, the code size needed for the heart rate task is 402 Bytes, the rest of the InfiniTime OS occupies 397434 Bytes, which means that 80.9 % of the available flash memory is used (only 480 KiB are available for applications). For the comparisons on the PineTime we used a release build environment, whereas a debug build environment was used for the Apple Watch. On the Apple Watch, the memory needed to hold the tree-based models is around 5.5 MiB, and the size of the application occupies approx. 20 MiB on the nonvolatile storage.

TABLE II
PERFORMANCE AND MEMORY REQUIREMENTS – PINETIME (PT) AND APPLE WATCH (AW).

Model	R^2	*MAE*	*PT ROM (Bytes)*	*PT Flash %*	*AW Model MiB*	*AW Size (MiB)*
LM_1	0.735	8.43	456	81.0	–	–
LM_2	0.805	7.26	446	81.0	–	–
RT_1	0.914	3.51	–	–	5.54	19.3
RT_2	0.748	8.09	1222	81.1	–	–
RT_3	0.807	6.87	14970	83.9	–	–
RT_4	0.825	6.49	28732	86.7	–	–
RT_5	0.840	6.10	54194	91.9	–	–
RT_5	0.877	5.33	30912	87.2	–	–
RF_1	0.917	3.53	–	–	5.21	19.8
RF_2	0.800	7.06	65576	94.2	–	–

V. CONCLUSIONS AND FUTURE WORK

In this paper, we described how to estimate energy expenditure from heart rate with a higher coefficient of determination using tree-based regressors than commonly used linear models. Using data from 892 graded exercise tests we trained various models and selected one which not only performed better than the linear model but also fitted in the flash memory of the open source smartwach PineTime. Our tree-based model does not need to know VO_{2max} but achieves a comparable result as the linear model with VO_{2max} making it especially interesting for amateur athletes. The additional memory on the PineTime smartwach needed to store the tree increased the the original firmware size of 390 KiB to 416 KiB. If VO_{2max} is available, then a tree with a depth of 11 achieves a coefficient of 0.877, and the total memory size is 418 KiB.

When considering the Apple Watch as another amateur athlete tool the memory constraint becomes irrelevant, since the regressors used in this paper can be utilized on the watch with no difficulty and result in an acceptable memory usage of less than 10 MiB.

In contrast to the linear model, our regression tree-based model seems to predict EE for sub-maximal, maximal and light efforts better. However, this still needs to be further investigated by defining limits of agreement and performing an ANOVA. Because the database contains data from treadmill tests only, it is not possible to validate how our model performs in other contexts, e.g., cycling or nordic walking. Consequently, we plan to extend the database in the future.

REFERENCES

[1] A. P. Hills, N. Mokhtar, and N. M. Byrne, "Assessment of physical activity and energy expenditure: An overview of objective measures," *Frontiers in Nutrition*, vol. 1, 2014. [Online]. Available: https://www.frontiersin.org/article/10.3389/fnut.2014.00005

[2] I.-M. Lee, C. cheng Hsieh, and R. S. Paffenbarger, "Exercise Intensity and Longevity in Men: The Harvard Alumni Health Study," *JAMA*, vol. 273, no. 15, pp. 1179–1184, 04 1995.

[3] J. Morris, J. Heady, P. Raffle, C. Roberts, and J. Parks, "Coronary heart-disease and physical activity of work," *The Lancet*, vol. 262, no. 6795, pp. 1053–1057, 1953.

[4] G. D. Batty and I.-M. Lee, "Physical activity and coronary heart disease." *BMJ (Clinical research ed.)*, vol. 328, no. 7448, pp. 1089–1090, 2004.

[5] K. Domaszewska, M. Boraczynski, Y.-Y. Tang, J. Gronek, K. Wochna, T. Boraczynski, D. Wielinski, and P. Gronek], "Protective effects of exercise become especially important for the aging immune system in the covid-19 era," *Aging and disease*, vol. 13, no. 1, pp. 129–143, 2022.

[6] D. Fister, I. Fister, S. Rauter, and I. Fister, "Generating eating plans for athletes using the particle swarm optimization," in *2016 IEEE 17th International Symposium on Computational Intelligence and Informatics (CINTI)*, 2016, pp. 000 193–000 198.

[7] J. Rivera, A. McPherson, J. Hamilton, C. Birken, M. Coons, S. Iyer, A. Agarwal, C. Lalloo, and J. Stinson, "Mobile apps for weight management: A scoping review," *JMIR Mhealth Uhealth*, vol. 4, no. 3, p. e87, Jul 2016.

[8] S. S. Coughlin, "Use of consumer wearable devices to promote physical activity: A review of health intervention studies," *Journal of Environment and Health Sciences*, vol. 2, pp. 1– 6, 2016.

[9] D. Ndahimana and K. Eun-Kyung, "Measurement methods for physical activity and energy expenditure: a review." *Clinical nutrition research*, vol. 6, no. 2, pp. 68–80, 2017.

[10] W. Leonard, "Measuring human energy expenditure: what have we learned from the flex-heart rate method?" *American journal of human biology : the official journal of the Human Biology Council*, vol. 15, pp. 479–89, 07 2003.

 ISBN: 978-1-61208-951-5

[11] L. Keytel, J. Goedecke, T. Noakes, H. Hiiloskorpi, R. Laukkanen, L. van der Merwe, and E. Lambert, "Prediction of energy expenditure from heart rate monitoring during submaximal exercise," *Journal of sports sciences*, vol. 23, pp. 289–97, 04 2005.

[12] K. Charlot, J. Cornolo, R. Borne, J. Brugniaux, J.-P. Richalet, D. Chapelot, and A. Pichon, "Improvement of energy expenditure prediction from heart rate during running," *Physiological measurement*, vol. 35, pp. 253–266, 01 2014.

[13] K. Ellis, J. Kerr, S. Godbole, G. R. G. Lanckriet, D. Wing, and S. J. Marshall, "A random forest classifier for the prediction of energy expenditure and type of physical activity from wrist and hip accelerometers." *Physiological measurement*, vol. 35 11, pp. 2191–203, 2014.

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

[15] "m2cgen (model 2 code generator)," 2022, URL: https://github.com/BayesWitnesses/m2cgen [accessed: 2022-4-1].

[16] "Pinetime)," 2022, URL: https://wiki.pine64.org/wiki/PineTime [accessed: 2022-4-1].

[17] D. Mongin, J. García-Romero, and J. R. Alvero-Cruz, "Treadmill maximal exercise tests from the exercise physiology and human performance lab of the university of malaga (version 1.0.1)," PhysionNet, 2021, URL: https://doi.org/10.1080/15438627.2021.1954513 [accessed: 2022-4-1].

[18] D. Mongin, C. Chabert, D. S. Courvoisier, J. García-Romero, and J. R. Alvero-Cruz, "Heart rate recovery to assess fitness: comparison of different calculation methods in a large cross-sectional study," *Research in Sports Medicine*, vol. 0, no. 0, pp. 1–14, 2021.

[19] A. L. Goldberger, L. A. N. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley, "PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals," *Circulation*, vol. 101, no. 23, pp. e215–e220, 2000 (June 13).

[20] J. B. d. V. Weir, "New methods for calculating metabolic rate with special reference to protein metabolism," *The Journal of Physiology*, vol. 109, no. 1-2, pp. 1–9, 1949.

[21] "Infinitime," 2022, URL: https://infinitime.io/ [accessed: 2022-4-1].

[22] B. Sudharsan, P. Patel, J. G. Breslin, and M. I. Ali, "Ultra-fast machine learning classifier execution on iot devices without sram consumption," in *2021 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops)*, 2021, pp. 316–319.

[23] "puncover," 2022, URL: https://github.com/HBehrens/puncover [accessed: 2022-4-1].

[24] "coremltools," 2017, URL: https://github.com/apple/coremltools [accessed: 2022-4-1].

[25] "coremltools documentation," 2017, URL: https://coremltools.readme.io/docs [accessed: 2022-4-1].

 ISBN: 978-1-61208-951-5