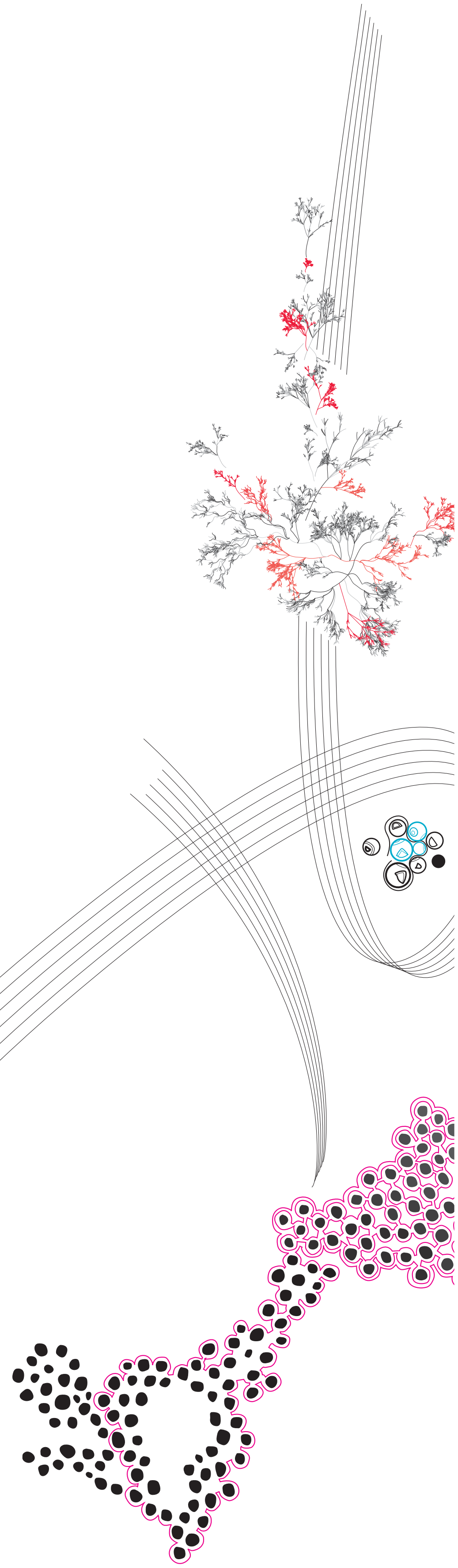
BSc Thesis Applied Mathematics & Technical Computer Science



On the effects of smoothing on machine learning performance in fatigue detection using sensor data

Stan Verschuren

Supervisor: Katharina Proksch, Dennis Reidsma

February, 2023

Department of Applied Mathematics
Faculty of Electrical Engineering,
Mathematics and Computer Science

UNIVERSITY OF TWENTE.

Preface

I want to thank my supervisors Katharina Proksch and Dennis Reidsma for their valuable feedback and sharp comments, their assistance and the cookies. I also want to thank the people in the Bachelor Project Support Group for helping me organize and stay positive during the creation of this paper.

On the effects of smoothing on machine learning performance in fatigue detection using sensor data

Stan. P. Verschuren

February, 2023

Abstract

Fatigue detection during a running session is important to decrease the risk of injury. Machine learning in combination with wearable sensor technology is a viable option to detect it, and can be used to provide live feedback to the runner. The raw data is often smoothed to decrease the noise, but the effects of this smoothing can be drastically different, depending on the setup of the sensors and data processing. In this paper, we compare the difference in performance of a machine learning algorithm using smoothed data against non-smoothed data in order to examine the influence of different smoothing setups on the detection of fatigue. This experiment used raw acceleration data from 7 sensors placed on the body during a fatiguing treadmill run. On that data, we applied Butterworth low-pass filters with different orders and cutoff frequencies, and for each sensor, the magnitude of the acceleration was calculated. The data was then segmented into strides and four features were extracted per sensor. Then, twelve combinations of sensors were tested by comparing the performance, in terms of accuracy, of a random forest with the smoothed dataset and the non-smoothed dataset using the 5x2 cross validation t-test. We found that the smoothing could increase or decrease the accuracy by 15% when using a single sensor as input data for the random forest. Moreover, the smoothing did not increase or lower the performance in a significant way ($\alpha = 0.05$) when multiple sensors are used, independent of the configuration of the cutoff frequency and order. Additionally, the accuracy increased in general the more sensors are used. *Keywords*: smoothing, sensors, machine learning,

Butterworth, filter, running, fatigue, detection, random forest, 5x2 cross validation

1 Introduction

Running has risen in popularity as a recreational sport all across Europe [15]. Consequently, the number of people who get a running-related injury is high [4]. There can be a lot of factors that contribute to this, and one of those factors is fatigue [17]. It changes the way we move in multiple ways [7], and so it is something to keep track of during a run.

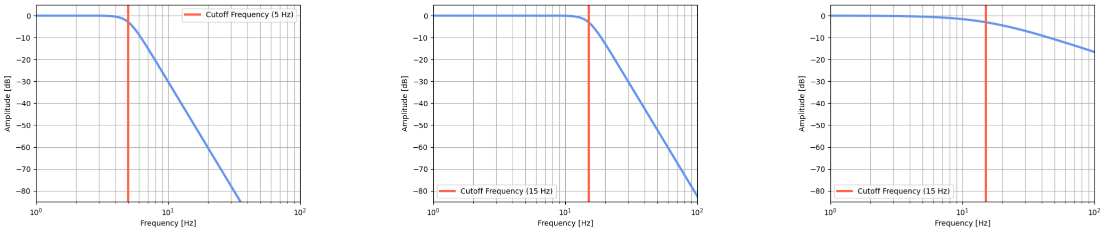
A means to do so is gait analysis which can be used to determine if someone is exerted. Gait analysis studies the characteristics of the steps someone makes, for example knee angle, foot height or leg swing acceleration. These measurements can be extracted using wearable sensor technology [16]. However, researchers try to minimize the amount of sensors used to reduce the intrusiveness towards the runners [13].

Machine learning is an advancing technique that can detect fatigue with promising performance [3, 11, 18]. When fitted, it can detect a fatigued state faster than a human analyser can and might be able to pick up on more subtle changes in movement, and thus can give live feedback during training. The wearable sensors can provide live data to the algorithm to detect the state of the runner, which can be used to give the runner a training suggestion in real time, for example to take a break.

In traditional gait analysis, the data would be processed into features which can be used to make a judgement about fatigue, like acceleration data and consequently joint reaction forces [6]. However, this method is prone to noise. If a noisy signal is processed into features, then the features can be inaccurate [6]. Therefore, a smoothing procedure is often applied in practice before handling the data [6, 9].

Machine learning algorithms can be reasonably resistant to noise, for example the random forest algorithm [2]. However, it is not uncommon to apply some smoothing process before providing the data to the machine learning algorithm [3, 18]. Smoothing removes the noise present in the sensor data, but it might also remove real data. This data can be beneficial when detecting fatigue, and a machine learning algorithm might be able to pick up on it. Additionally, There exists no general consensus on how to configure the smoothing procedures. The questions in this article are thus whether the smoothing process is needed at all for a machine learning algorithm to detect fatigue accurately, and, if so, how important is the configuration of the smoothing for the performance?

One smoothing procedure which is dominant in the field of bio-mechanics is the Butterworth low-pass filter [1, 3, 12, 14, 18]. A Butterworth low-pass filter reduces parts of a signal as function of frequency. There are two parameters to be set for a Butterworth low-pass filter, the order and the cutoff frequency. The cutoff frequency determines at which frequency the filter starts to reduce the signal. If the cutoff frequency is set to 15 Hz, all frequencies higher than 15 Hz will be reduced, leaving the ones below 15 nearly untouched. However, the reduction is applied gradually from 15 Hz onward. How steep the reduction occurs, is determined by the order of the filter. A higher order means steeper reduction, (see Figure 1 for a visual example on what these two parameters do).



(A) (B) (C)

Figure 1: Comparison of frequency responses with different order or cutoff frequency. (a) frequency response when the order is set to 5 and the cutoff frequency to 5 Hz; (b) frequency response when the order is set to 5 and the cutoff frequency to 15 Hz; (c) frequency response when the order is set to 1 and the cutoff frequency to 15 Hz.

In this article, we will use the random forest algorithm as machine learning algorithm to test our hypotheses. This algorithm is present in different other studies regarding fatigue detection with machine learning [3, 11, 18], and is robust with respect to noise [2]. A random forest is a machine learning algorithm that takes majority votes from a collection of randomized decision trees. A decision tree is a binary tree with condition checks as nodes and a final outcome on the leaf nodes. This way, a decision can be made based on certain conditions. In machine learning, the split nodes in decision trees are established by comparing how effective the split is (this will be explained in further detail in the next section), and predictions are made by iterating through the tree. A small example of a decision tree can be seen in Figure 2.

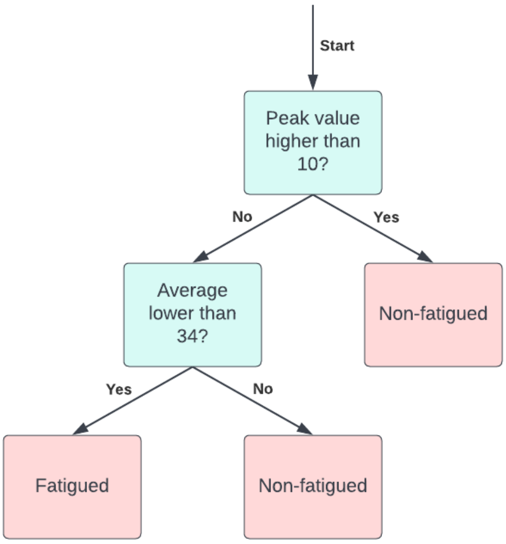


Figure 2: An example decision tree for predicting the fatigued state of a runner based on the peak value and average acceleration of a sensor. The red nodes are the final outcomes whereas the green nodes are the condition checks.

Each decision tree in random forest is trained using a new bootstrapped dataset with a set sample size. This is a set of randomly picked entries from the original set, with replacement. Additionally, only a subset of features is chosen at random when determining the optimal split for each decision tree in the forest. These measures provides a randomness factor to each tree, making them distinct from each other but not unrelated.

The goal of this article is to analyse the effects of the cutoff frequency and the order of a Butterworth low-pass filter on the detection of a fatigued and non-fatigued state based on sensor data using a random forest algorithm. We will look into the performance of the random forests and see if there is a difference between algorithms trained on smoothed or raw data. This will be done using a specialized t-test called the 5x2 cross validation t-test, as proposed by Dietterich in his paper "Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms" [8]. Furthermore, this difference, when statistically significant, can tell us more about the importance of the configuration of the smoothing procedure.

2 Methods

For our numerical experiments we use Python v3.10.7 and the libraries Pandas v1.5.1 for data manipulation, Numpy 1.24.4 for general mathematics, Scipy v1.9.2 for signal processing, Scikit-learn v1.1.3 for the machine learning and Matplotlib v3.6.1 for creating the plots and graphs.

2.1 Data

The raw data used for this experiment was gathered using several IMU sensors attached on multiple different places on the body during a fatiguing run. The sensors were set to measure with a rate of 240 Hz. To mitigate errors and noise from external factors like wind, the experiment was done indoors on a treadmill. The sensors were placed on the pelvis, the upper and lower parts of both the legs, and on both feet (see Figure 3).

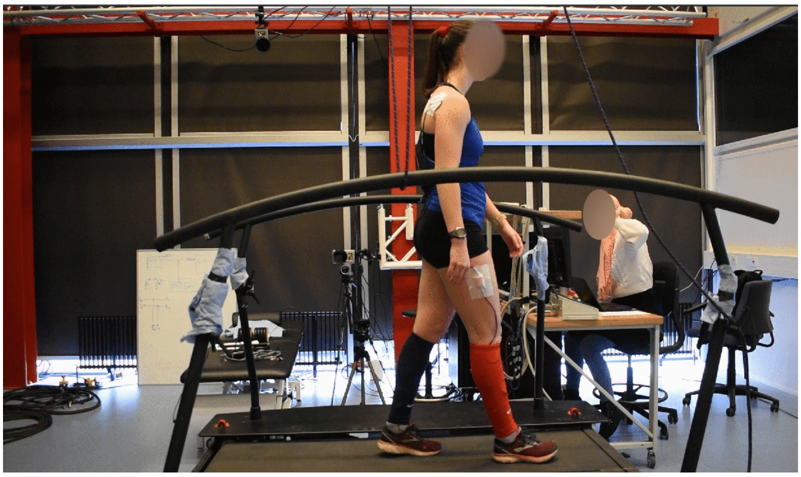


FIGURE 3: caption.

All sensors reported the acceleration in all 3 spacial directions. A sample of the raw sensor data of the left lower leg can be seen in Figure 4.

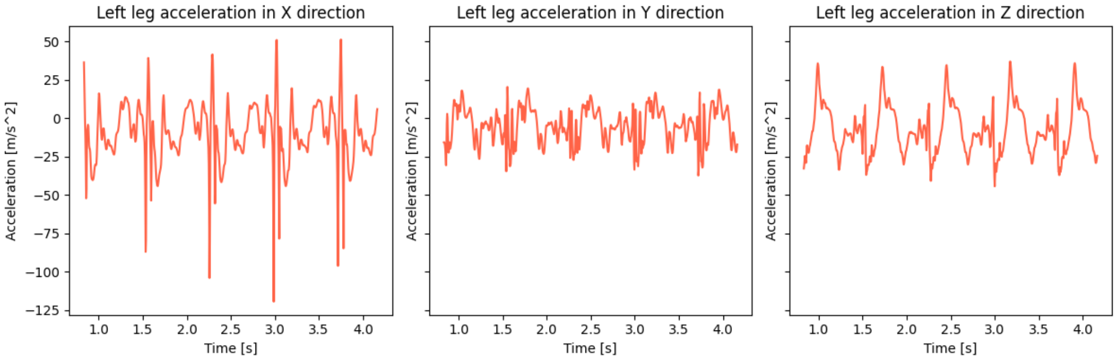


FIGURE 4: A sample of the recorded acceleration in all three spacial directions from the sensor on the left lower foot.

The data was directly read from the sensors, without any processing beforehand. After loading in the data, certain data features, like the sampling rate, were determined.

2.2 Smoothing

First the smoothing is applied to the sensor data. This is done normally to remove the noise from the sensors and to reduce the error in the processing to come. We use a Butterworth low-pass filter to achieve this smoothing [5].

Definition 1 (Butterworth (zero-phase) low-pass filter)**.** *The Butterworth low-pass filter is a filter that is applied in the frequency domain. Let f_c be the cutoff frequency and o be the order of the filter. Then the frequency response r to the filter is defined as*

$$r = \frac{1}{\sqrt{1 - \frac{f}{f_c}^{2o}}}.$$

However, the resulting signal will be shifted in the time domain, and so the smoothed signal does not fit exactly on the original signal. This is called a phase shift. A Butterworth zero-phase low-pass filter mitigates this problem. It applies the smoothing twice, first forwards, then again backwards. This does not change the resulting signal smoothing-wise, but realigns the smoothed signal with the original signal.

For every combination of parameters, cutoff frequency and order, a Butterworth zero-phase low-pass filter set with that combination was applied on the sensor data separately. The filter was applied using the forward-backward digital filter included in the 'signal' module of the Scipy library. See Figure 5 for a sample of our filtered data.

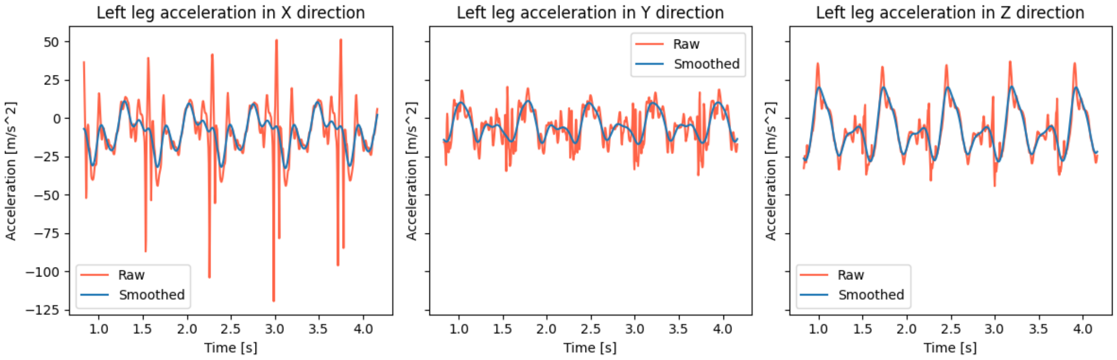


FIGURE 5: A sample of the smoothed acceleration of the left lower leg acceleration in all three spacial directions compared to the raw data using a Butterworth zero-phase low-pass filter. The filter is configured with an order of 3 and a cutoff frequency of 6 Hz.

The order was set between 1 and 5, and the frequency was set to range between 1 and 20 Hz. Both these parameters have an infinite domain to choose a range from, and for completeness, all those values need to be evaluated. However, due to practical limitations, we can only evaluate a smaller range. The order range from 1 to 5 was set with the mindset that in other papers, no order was set higher than 5 [3, 9, 12, 18], and the general recommendation is an order of 4, since filters with higher order introduce unwanted oscillations [6]. The frequency range set from 1 to 20 in increments of 1 was set this way with a similar mindset. The papers mentioned above had their frequencies set in the range between 5 and 15. We widened the range from 1 to 20, to include more edge information, but kept the set reasonably small for performance reasons. The frequency of the strides is also determined to be 1.4 Hz (discussed later on), showing that the range we chose fits in the order of magnitude of the data. This results in a set of 100 datasets, each with a different smoothing applied.

2.3 Data processing

After the smoothing, the data needs to be translated into meaningful signals. The data was first trimmed on both ends to exclude the calibration periods before and after the measurement run, and to remove the distortion at the edges caused by the low-pass filter. After that, we calculated the magnitude of the total acceleration for every sensor, (see definition 2).

Definition 2 (Magnitude of acceleration)**.** *Let $a_{t,X}$, $a_{t,Y}$ and $a_{t,Z}$ be the acceleration at time t measured in the three spacial directions X, Y and Z respectively. Then the magnitude of acceleration at time t is defined by*

$$a_t = \sqrt{{a_{t,X}}^2 + {a_{t,Y}}^2 + {a_{t,Z}}^2}.$$

This was done to keep the data as raw as possible, but to be independent on how the sensor axes are setup. This means that this can be calculated by every setup with similar sensor placements, without the need for exact directions. This technique, calculating the magnitude of the acceleration of a sensor was also done by Wang et al. in their paper "Fatigue Detection in Running with Inertial Measurement Unit and Machine Learning" [18]. A sample of the calculated signals can be seen in Figure 6.

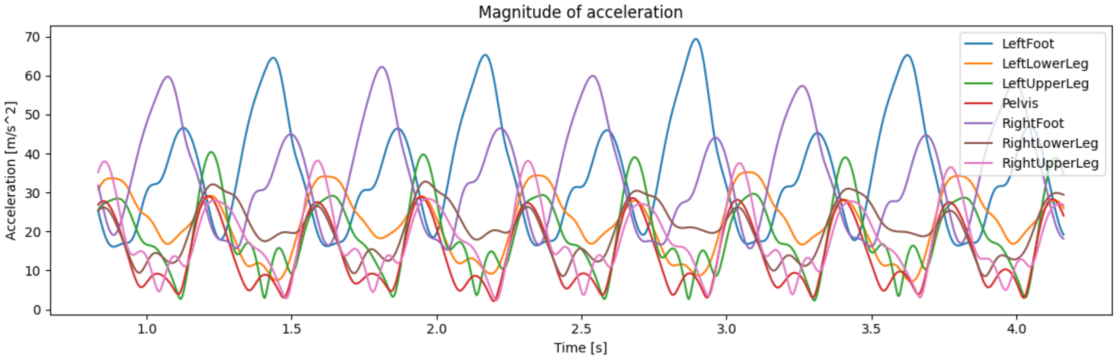


Figure 6: A sample of the magnitude of the acceleration of all sensors, calculated with the smoothed data configured with an order of 3 and a cutoff frequency of 6 Hz.

2.4 Stride segmentation

The sensors have measured the run on a sample rate of 240 Hz. But since a run is a made of repeating strides, the detection of fatigue can best be based on features of a stride, for example by peak values of acceleration of a leg during each stride. Therefore, a stride segmentation needs to be done.

We will define a stride as the cyclic repeating movement of one leg during a run. That is, a stride starts when a foot touches the ground, and ends when the same foot touches the ground again. To find these start and end points, one can look at the sudden peaks of acceleration upwards in one of the legs, since the contact with the ground exerts a sudden force upwards. To cut up the data, we used a peak finding algorithm from the signal module in the Scipy library. This peak finding algorithm was set to find the peaks in the upwards acceleration of the right lower leg. These points were indexed and used as splitting points for the segmentation. The strides were approximately 0.7 seconds long and had a frequency of approximately 1.4 Hz. The first two hundred strides will be labeled as non-fatigued, whereas the last two hundred strides will be labeled as fatigued. The strides in between are not used. See Figure 7 for a visualization of the first 30 and last 30 strides.

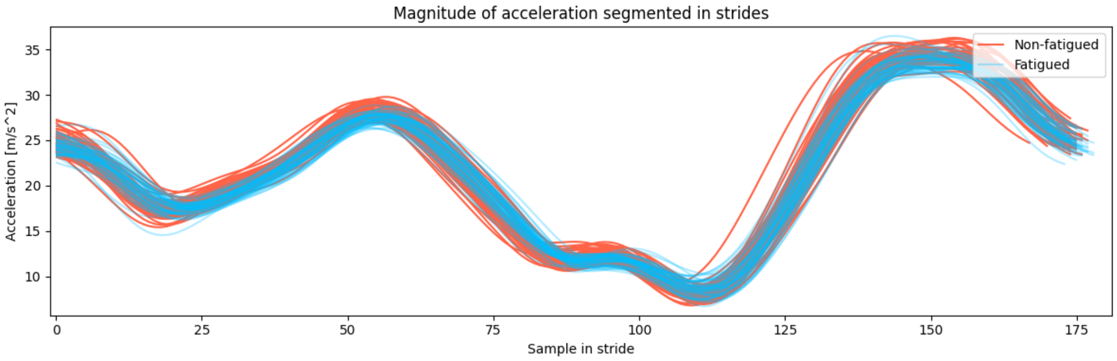


FIGURE 7: The magnitude of the acceleration of the left lower leg for the first and last 30 strides, calculated with the smoothed data configured with an order of 3 and a cutoff frequency of 6 Hz.

2.5 Feature extraction

In this experiment, we extracted several statistical features from the acceleration signals per stride. We collected the area under the curve, the minimum value, the maximum value, and the difference between the minimum and maximum value. These features were calculated for every sensor. The features of the left lower leg are shown in Figure 8

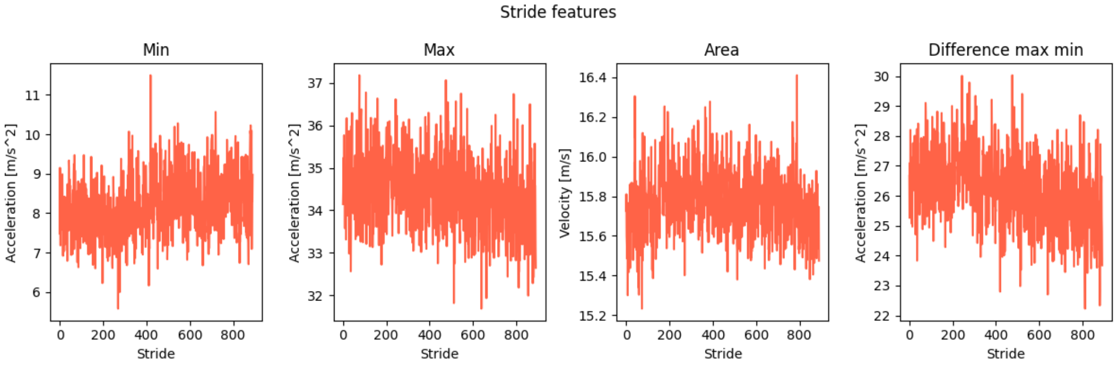


FIGURE 8: The minimum, maximum, area under the curve (velocity) and difference between minimum and minimum of the acceleration of the left lower leg, extracted per stride, calculated with the smoothed data configured with an order of 3 and a cutoff frequency of 6 Hz.

With these features, multiple datasets were created by joining features from different sensors. First, a dataset was created for every sensor on its own. Secondly, both upper leg, both lower leg and both feet sensors were each joined to create a dataset, as well as all sensors on the legs combined. Finally, all sensors were joined to create the largest dataset with all calculated features. This results in 12 datasets, summarized in the list below. These datasets were used separately as input for the machine learning algorithm.

- Pelvis
- Left upper leg
- Right upper leg
- Left lower leg
- Right lower leg
- Left foot
- Right foot
- Left and right upper leg
- Left and right lower leg
- Left and right foot
- Left and right upper and lower leg
- All sensors

2.6 Random forest training

For the classification, a random forest classifier was used. The random forest was created using the random forest classifier from the Sci-kit learn library, and was set up as follows:

- **The amount of decision trees per random forest**: 100. A random forest consists of multiple randomized decision trees. These trees all get an input to predict, and a majority vote is taken to determine the prediction of the random forest.
- **The number of random features to be considered during the determination of a best split**: the square root of the total amount of features. During the training of a decision tree, a feature and a threshold for that feature are determined to split the current (sub)set of data on. To determine the best split, every threshold for a feature in a randomly selected subset of features is compared to each other with a scoring formula. The size of this subset of features is set to be the square root of the total amount of features.
- **The scoring formula to determine the best split with**: The Gini impurity. When selecting the best split during training of a decision tree, the features are compared to each other via a certain scoring mechanism. This mechanism uses the Gini impurity. The scoring mechanism works as follows:

  Definition 3 (Gini impurity)**.** *Let C be a set of classes and let D be a set of samples belonging to a class $c \in C$. Define n_D to be the total number of samples in D, and define $n_{D,c}$ to be the number of samples in D belonging to class c. Then the Gini impurity of dataset D is defined as*

  $$G(D) = 1 - \sum_{c \in C} (\frac{n_{D,c}}{n_D})^2.$$

  Suppose the dataset D is split in two sets, D_1 and D_2, then the Gini impurity of that split is defined as the weighted average of the two split datasets:

  $$G_{\text{split}}(D_1, D_2) = \frac{n_{D_1}}{n_D} G(D_1) + \frac{n_{D_2}}{n_D} G(D_2).$$

  The split that provides the lowest Gini impurity is set as the best split.

- **The number of samples a node can have to be automatically considered as a leaf node**: 2. This determines when to categorize a node as a leaf node. A node can also be set as a leaf node when all samples are of the same class during training.
- **The amount of samples to take when creating a bootstrapped dataset**: The total amount of samples in the original dataset. When creating the decision trees during training, each tree is trained on a randomly sampled dataset from the original

dataset, with replacement. This sampled dataset is called a bootstrapped dataset. The amount of samples to take is set to the same amount of samples the original dataset has.

2.7 Performance evaluation

For every sensor combination created, we will compare the performance of the algorithm with smoothed data against non-smoothed data in terms of accuracy. Accuracy is the amount of correctly predicted states divided by the total amount of predicted states during training, (see definition 4).

Definition 4 (Accuracy)**.** *Let D be a dataset with N entries, each having a class associated with it. If a classification algorithm predicts the class of each entry, then the accuracy of that algorithm is defined by*

$$ACC = \frac{\#(\text{correctly predicted entries})}{N}$$

For every combination of smoothing parameters, we will compare the performance in terms of accuracy using a specialized hypothesis test, called the 5x2 cross validation test as proposed by Dietterich [8].

2.7.1 5x2 cross validation

This test is meant to compare two different machine learning algorithms against each other, trained on the same dataset. However, we want to compare the same algorithm against itself trained on two different datasets. We can reason that, in our case, this test is still applicable if we consider the data processing procedure, including the smoothing step, to be part of the algorithm. This way, the dataset we start with is identical, and the algorithms are different. We will test the difference between the smoothed and non-smoothed versions using the following hypothesis test:

Hypothesis.
H_0: The average accuracy of the algorithm using smoothed data is the same as the average accuracy of the algorithm using non-smoothed data.
H_1: The average accuracy of the algorithm using smoothed data is different from the average accuracy of the algorithm using non-smoothed data.
α: 0.05

The null hypothesis for each of these tests is that the accuracy of both algorithms are the same, while the alternative hypothesis is that they are not. We will accept the null hypothesis if the 5x2 cross validation t-test produces a p-value above 0.05, and we will reject the null hypothesis if the p-value is below 0.05. Let us first define the original test, taken from Dietterich [8], and then propose the alterations that we want to make to use for our case.

1. **Definition**: Let D be the dataset that the machine learning algorithms need to be tested on, let A and B be the two algorithms you want to compare, and let $k \in \{1, 2, 3, 4, 5\}$.

2. **Split**: For every k, randomly partition the dataset D into two equally sized sets, $D_k^{(1)}$ and $D_k^{(2)}$.

3. **2 Fold cross validation**: These two partitions will be used as folds in a two fold cross validation. This gives us 4 accuracy scores for every k, $a_k^{(1)}(A)$ and $a_k^{(1)}(B)$ (accuracy of A and B trained on set $D_k^{(1)}$), and $a_k^{(2)}(A)$ and $a_k^{(2)}(B)$ (accuracy of A and B trained on set $D_k^{(2)}$).

4. **Difference**: Now the difference between scores on a partition for every k can be calculated: $a_k^{(1)} = a_k^{(1)}(A) - a_k^{(1)}(B)$ and $a_k^{(2)} = a_k^{(2)}(A) - a_k^{(2)}(B)$.

5. **Mean and variance**: Using these variables, we can compute a mean: $\bar{a}_k = (a_k^{(2)} + a_k^{(1)})/2$, and a variance: $s_k^2 = (a_k^{(1)} - \bar{a}_k)^2 + (a_k^{(2)} - \bar{a}_k)^2$.

6. **T statistic**: From these we can define a t statistic to test on:

$$t = \frac{a_1^{(1)}}{\sqrt{\frac{1}{5}\sum_{k=1}^{5} s_k^2}} \tag{1}$$

Under the null hypothesis, that is, the mean accuracy of both algorithms is identical, this t is approximately distributed like a Student's t distribution with 5 degrees of freedom [8]. To get the final accuracy of the algorithm, we take the average of all the means $\bar{a}_k$.

Since we have two versions of a dataset and one algorithm, we need to change the definition of some of these variables. we can switch D for an algorithm and A and B for datasets, while the rest stays the same. The changes are as follows:

1. **Definition**: Let A and B be similar but differently smoothed datasets and let D be a machine learning algorithm.

2. **Split**: Define $A_k^{(1)}$ and $A_k^{(2)}$ to be the two equally sized partitions for every k, similarly for B.

3. **2 Fold cross validation**: Then, using those partitions in a two fold cross validation, the four accuracy scores are $a_k^{(1)}(A)$ and $a_k^{(2)}(A)$ (accuracy of D trained on set $A_k^{(1)}$ and $A_k^{(2)}$) and $a_k^{(1)}(B)$ and $a_k^{(2)}(B)$ (accuracy of D trained on set $B_k^{(1)}$ and $B_k^{(2)}$).

4. **Difference**: With these we can calculate the difference in scores with the same formula as the original: $a_k^{(1)} = a_k^{(1)}(A) - a_k^{(1)}(B)$ and $a_k^{(2)} = a_k^{(2)}(A) - a_k^{(2)}(B)$.

The rest of the definitions stays the same.

2.7.2 t-test

Using the t calculated as in (1), we can perform a two-tailed Student's t-test with 5 degrees of freedom. The null hypothesis states that the accuracy of both methods are identical, while the alternative hypothesis states they are not identical. We will use an alpha of 0.05. The probability density can be viewed in Figure 9.

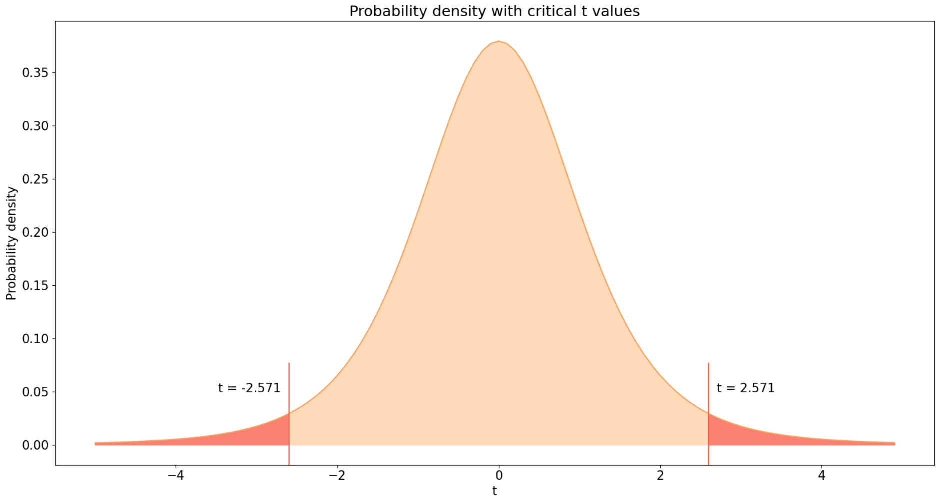


FIGURE 9: The probability density function of a Student's t-test with 5 degrees of freedom, highlighting the critical values of t when the alpha is set to 0.05.

The p-value was calculated by $p = 2\big(1 - F_t(|t|)\big)$, where F_t is the cumulative distribution function of t. Using these p-values, we can see which differences are significant and which are not. This can show if there is a difference, and if so, where it is most important.

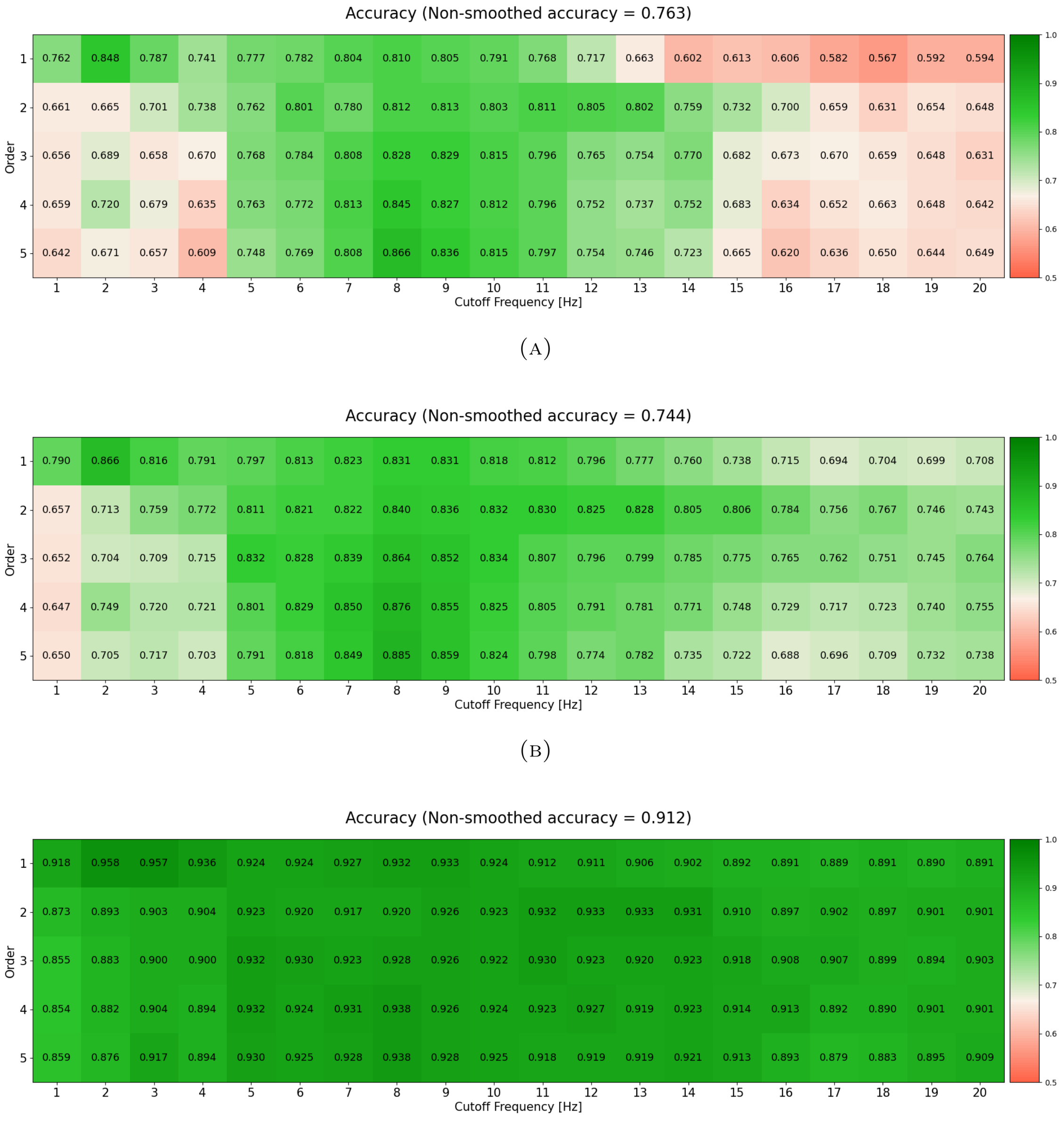
3 Results

For every set of sensors determined earlier, we created a set of tables. Each table is made with the cutoff frequency of the smoothing filter as columns, and the order as rows. This way, each cell represents a random forest, trained on data with the respective smoothing applied.

For every subsequent type of table, we will only show the tables for the left upper leg sensor, the table for the two upper leg sensors together and the table with all sensors combined. The other sensor combinations are deferred to the appendix for the sake of a clear presentation of the results.

3.1 Accuracy

The tables shown in Figure 10 below show the accuracy values of the random forests for each smoothing configuration. The accuracy of the random forest with the non-smoothed data is listed on top of each table as reference. These values were collected from the 10 tests during the 5x2 cross validation described in the methods. Notice that an overall increase in accuracy can be recognized when using more sensors, which indicates that using more sensors provides better performance overall. We can also observe a band in the cutoff frequency where the accuracy is higher than outside on the lower sensor tables, with the exception to the pelvis and both feet. This indicates that there might be a correlation between the cutoff frequency of the filter and the performance of the random forest. However, the band is different for each sensor, so no direct link can be made, proving interesting for further research.



Accuracy (Non-smoothed accuracy = 0.763)

Order	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.762	0.848	0.787	0.741	0.777	0.782	0.804	0.810	0.805	0.791	0.768	0.717	0.663	0.602	0.613	0.606	0.582	0.567	0.592	0.594
2	0.661	0.665	0.701	0.738	0.762	0.801	0.780	0.812	0.813	0.803	0.811	0.805	0.802	0.759	0.732	0.700	0.659	0.631	0.654	0.648
3	0.656	0.689	0.658	0.670	0.768	0.784	0.808	0.828	0.829	0.815	0.796	0.765	0.754	0.770	0.682	0.673	0.670	0.659	0.648	0.631
4	0.659	0.720	0.679	0.635	0.763	0.772	0.813	0.845	0.827	0.812	0.796	0.752	0.737	0.752	0.683	0.634	0.652	0.663	0.648	0.642
5	0.642	0.671	0.657	0.609	0.748	0.769	0.808	0.866	0.836	0.815	0.797	0.754	0.746	0.723	0.665	0.620	0.636	0.650	0.644	0.649

(A)

Accuracy (Non-smoothed accuracy = 0.744)

Order	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.790	0.866	0.816	0.791	0.797	0.813	0.823	0.831	0.831	0.818	0.812	0.796	0.777	0.760	0.738	0.715	0.694	0.704	0.699	0.708
2	0.657	0.713	0.759	0.772	0.811	0.821	0.822	0.840	0.836	0.832	0.830	0.825	0.828	0.805	0.806	0.784	0.756	0.767	0.746	0.743
3	0.652	0.704	0.709	0.715	0.832	0.828	0.839	0.864	0.852	0.834	0.807	0.796	0.799	0.785	0.775	0.765	0.762	0.751	0.745	0.764
4	0.647	0.749	0.720	0.721	0.801	0.829	0.850	0.876	0.855	0.825	0.805	0.791	0.781	0.771	0.748	0.729	0.717	0.723	0.740	0.755
5	0.650	0.705	0.717	0.703	0.791	0.818	0.849	0.885	0.859	0.824	0.798	0.774	0.782	0.735	0.722	0.688	0.696	0.709	0.732	0.738

(B)

Accuracy (Non-smoothed accuracy = 0.912)

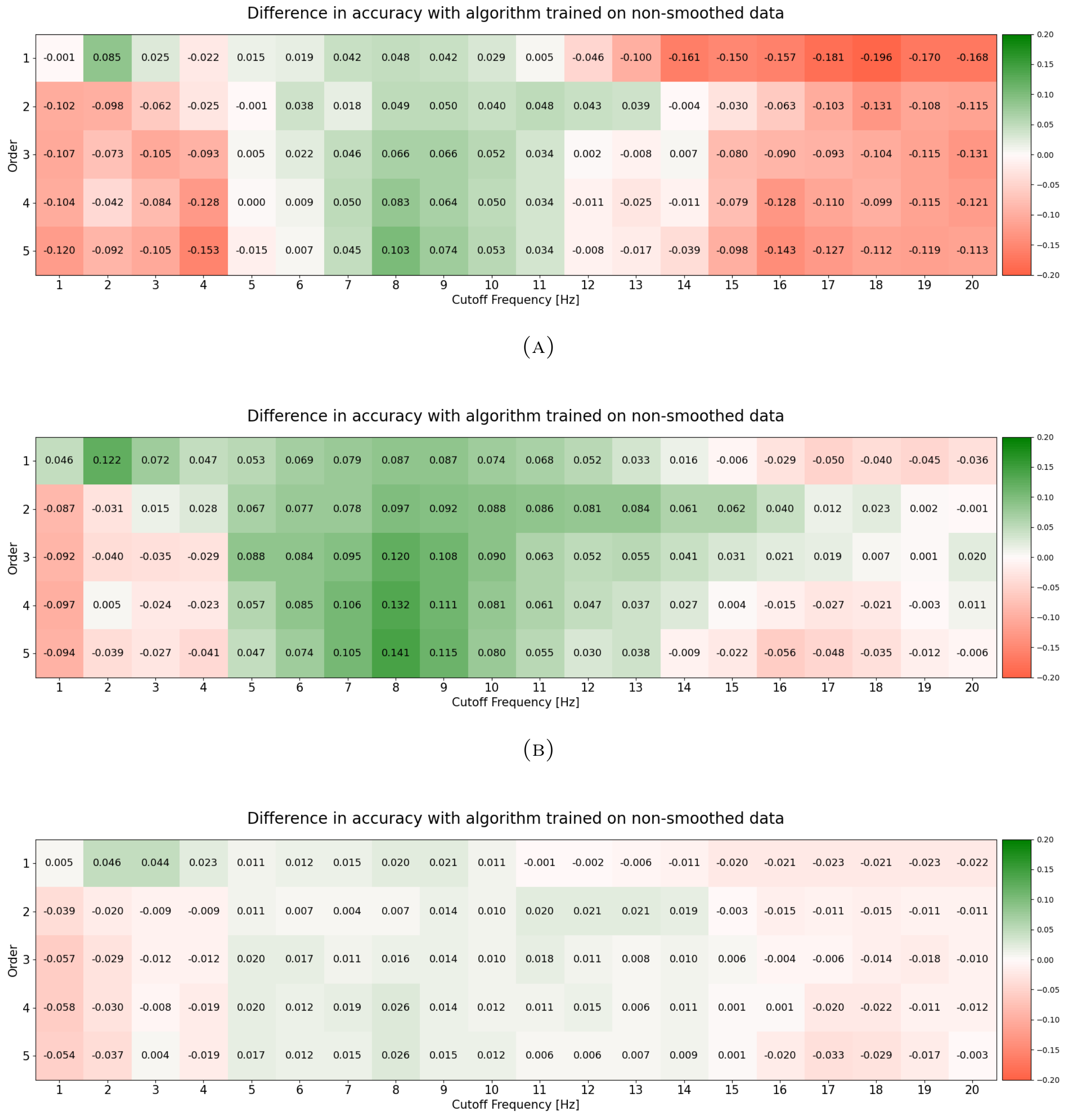
Order	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.918	0.958	0.957	0.936	0.924	0.924	0.927	0.932	0.933	0.924	0.912	0.911	0.906	0.902	0.892	0.891	0.889	0.891	0.890	0.891
2	0.873	0.893	0.903	0.904	0.923	0.920	0.917	0.920	0.926	0.923	0.932	0.933	0.933	0.931	0.910	0.897	0.902	0.897	0.901	0.901
3	0.855	0.883	0.900	0.900	0.932	0.930	0.923	0.928	0.926	0.922	0.930	0.923	0.920	0.923	0.918	0.908	0.907	0.899	0.894	0.903
4	0.854	0.882	0.904	0.894	0.932	0.924	0.931	0.938	0.926	0.924	0.923	0.927	0.919	0.923	0.914	0.913	0.892	0.890	0.901	0.901
5	0.859	0.876	0.917	0.894	0.930	0.925	0.928	0.938	0.928	0.925	0.918	0.919	0.919	0.921	0.913	0.893	0.879	0.883	0.895	0.909

(C)

FIGURE 10: The accuracy scores of the random forests trained on data with differently configured Butterworth low-pass filters using different combinations of sensors. (a) Using the sensor on the left upper leg, (b) Using the sensors on both upper legs combined, (c) Using all sensors.

3.2 Difference in accuracy

The tables shown in Figure 11 show the difference in accuracy of the random forests for each smoothing configuration and the random forest trained on the non-smoothed data. A positive value means that the algorithm with smoothed data performs better than without smoothing. Positive values are shown in green, whereas negative values are shown in red. In a similar way as before, we can observe a band in the cutoff frequency where the accuracy difference is positive in all tables, except for the left foot (see Figure B.6 in the appendix). However, the difference seems to shrink the more sensors are used. This indicates that the smoothing has less of an impact on the performance if more sensors are used, and that there might indeed be a correlation between cutoff frequency.



Order	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	-0.001	0.085	0.025	-0.022	0.015	0.019	0.042	0.048	0.042	0.029	0.005	-0.046	-0.100	-0.161	-0.150	-0.157	-0.181	-0.196	-0.170	-0.168
2	-0.102	-0.098	-0.062	-0.025	-0.001	0.038	0.018	0.049	0.050	0.040	0.048	0.043	0.039	-0.004	-0.030	-0.063	-0.103	-0.131	-0.108	-0.115
3	-0.107	-0.073	-0.105	-0.093	0.005	0.022	0.046	0.066	0.066	0.052	0.034	0.002	-0.008	0.007	-0.080	-0.090	-0.093	-0.104	-0.115	-0.131
4	-0.104	-0.042	-0.084	-0.128	0.000	0.009	0.050	0.083	0.064	0.050	0.034	-0.011	-0.025	-0.011	-0.079	-0.128	-0.110	-0.099	-0.115	-0.121
5	-0.120	-0.092	-0.105	-0.153	-0.015	0.007	0.045	0.103	0.074	0.053	0.034	-0.008	-0.017	-0.039	-0.098	-0.143	-0.127	-0.112	-0.119	-0.113

(a)

Order	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.046	0.122	0.072	0.047	0.053	0.069	0.079	0.087	0.087	0.074	0.068	0.052	0.033	0.016	-0.006	-0.029	-0.050	-0.040	-0.045	-0.036
2	-0.087	-0.031	0.015	0.028	0.067	0.077	0.078	0.097	0.092	0.088	0.086	0.081	0.084	0.061	0.062	0.040	0.012	0.023	0.002	-0.001
3	-0.092	-0.040	-0.035	-0.029	0.088	0.084	0.095	0.120	0.108	0.090	0.063	0.052	0.055	0.041	0.031	0.021	0.019	0.007	0.001	0.020
4	-0.097	0.005	-0.024	-0.023	0.057	0.085	0.106	0.132	0.111	0.081	0.061	0.047	0.037	0.027	0.004	-0.015	-0.027	-0.021	-0.003	0.011
5	-0.094	-0.039	-0.027	-0.041	0.047	0.074	0.105	0.141	0.115	0.080	0.055	0.030	0.038	-0.009	-0.022	-0.056	-0.048	-0.035	-0.012	-0.006

(b)

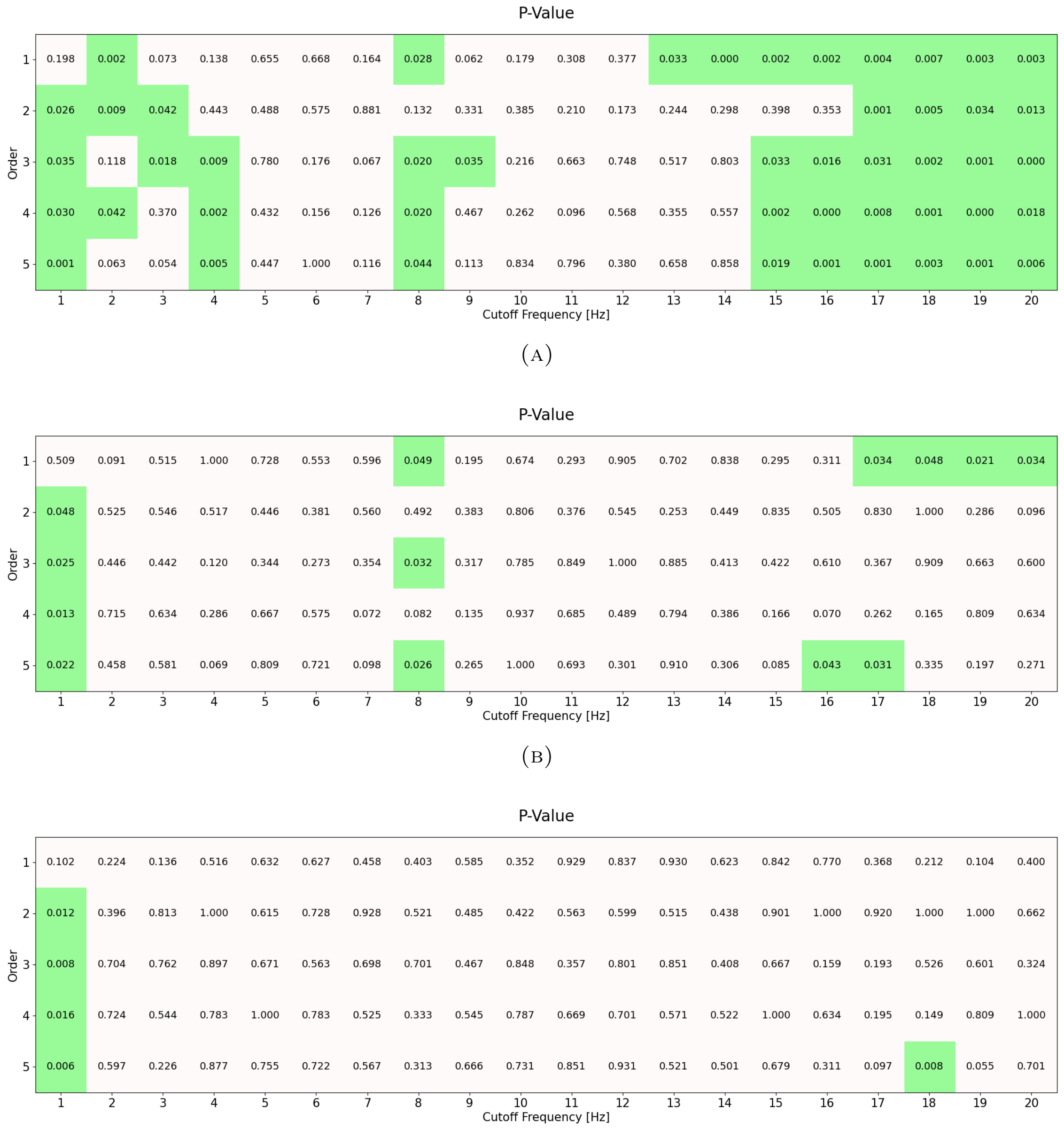
Order	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.005	0.046	0.044	0.023	0.011	0.012	0.015	0.020	0.021	0.011	-0.001	-0.002	-0.006	-0.011	-0.020	-0.021	-0.023	-0.021	-0.023	-0.022
2	-0.039	-0.020	-0.009	-0.009	0.011	0.007	0.004	0.007	0.014	0.010	0.020	0.021	0.021	0.019	-0.003	-0.015	-0.011	-0.015	-0.011	-0.011
3	-0.057	-0.029	-0.012	-0.012	0.020	0.017	0.011	0.016	0.014	0.010	0.018	0.011	0.008	0.010	0.006	-0.004	-0.006	-0.014	-0.018	-0.010
4	-0.058	-0.030	-0.008	-0.019	0.020	0.012	0.019	0.026	0.014	0.012	0.011	0.015	0.006	0.011	0.001	0.001	-0.020	-0.022	-0.011	-0.012
5	-0.054	-0.037	0.004	-0.019	0.017	0.012	0.015	0.026	0.015	0.012	0.006	0.006	0.007	0.009	0.001	-0.020	-0.033	-0.029	-0.017	-0.003

(c)

Figure 11: The difference in accuracy scores between the random forests on smoothed and non-smoothed data using different combinations of sensors. (a) Using the sensor on the left upper leg, (b) Using the sensors on both upper legs combined, (c) Using all sensors.

3.3 P-Value

These tables show the calculated p-value for each comparison between smoothed and non-smoothed data. A p-value below our α value, 0.05, is considered significant, and is thus colored green here. With this α, we need to expect a total of 5 false positives, but clear regions of significant values can be seen throughout. Notice also that the number of cells with a significant p-value decreases with more sensors used. This suggests a decrease in importance of the smoothing with more sensors.



Order	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.198	0.002	0.073	0.138	0.655	0.668	0.164	0.028	0.062	0.179	0.308	0.377	0.033	0.000	0.002	0.002	0.004	0.007	0.003	0.003
2	0.026	0.009	0.042	0.443	0.488	0.575	0.881	0.132	0.331	0.385	0.210	0.173	0.244	0.298	0.398	0.353	0.001	0.005	0.034	0.013
3	0.035	0.118	0.018	0.009	0.780	0.176	0.067	0.020	0.035	0.216	0.663	0.748	0.517	0.803	0.033	0.016	0.031	0.002	0.001	0.000
4	0.030	0.042	0.370	0.002	0.432	0.156	0.126	0.020	0.467	0.262	0.096	0.568	0.355	0.557	0.002	0.000	0.008	0.001	0.000	0.018
5	0.001	0.063	0.054	0.005	0.447	1.000	0.116	0.044	0.113	0.834	0.796	0.380	0.658	0.858	0.019	0.001	0.001	0.003	0.001	0.006

(A)

Order	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.509	0.091	0.515	1.000	0.728	0.553	0.596	0.049	0.195	0.674	0.293	0.905	0.702	0.838	0.295	0.311	0.034	0.048	0.021	0.034
2	0.048	0.525	0.546	0.517	0.446	0.381	0.560	0.492	0.383	0.806	0.376	0.545	0.253	0.449	0.835	0.505	0.830	1.000	0.286	0.096
3	0.025	0.446	0.442	0.120	0.344	0.273	0.354	0.032	0.317	0.785	0.849	1.000	0.885	0.413	0.422	0.610	0.367	0.909	0.663	0.600
4	0.013	0.715	0.634	0.286	0.667	0.575	0.072	0.082	0.135	0.937	0.685	0.489	0.794	0.386	0.166	0.070	0.262	0.165	0.809	0.634
5	0.022	0.458	0.581	0.069	0.809	0.721	0.098	0.026	0.265	1.000	0.693	0.301	0.910	0.306	0.085	0.043	0.031	0.335	0.197	0.271

(B)

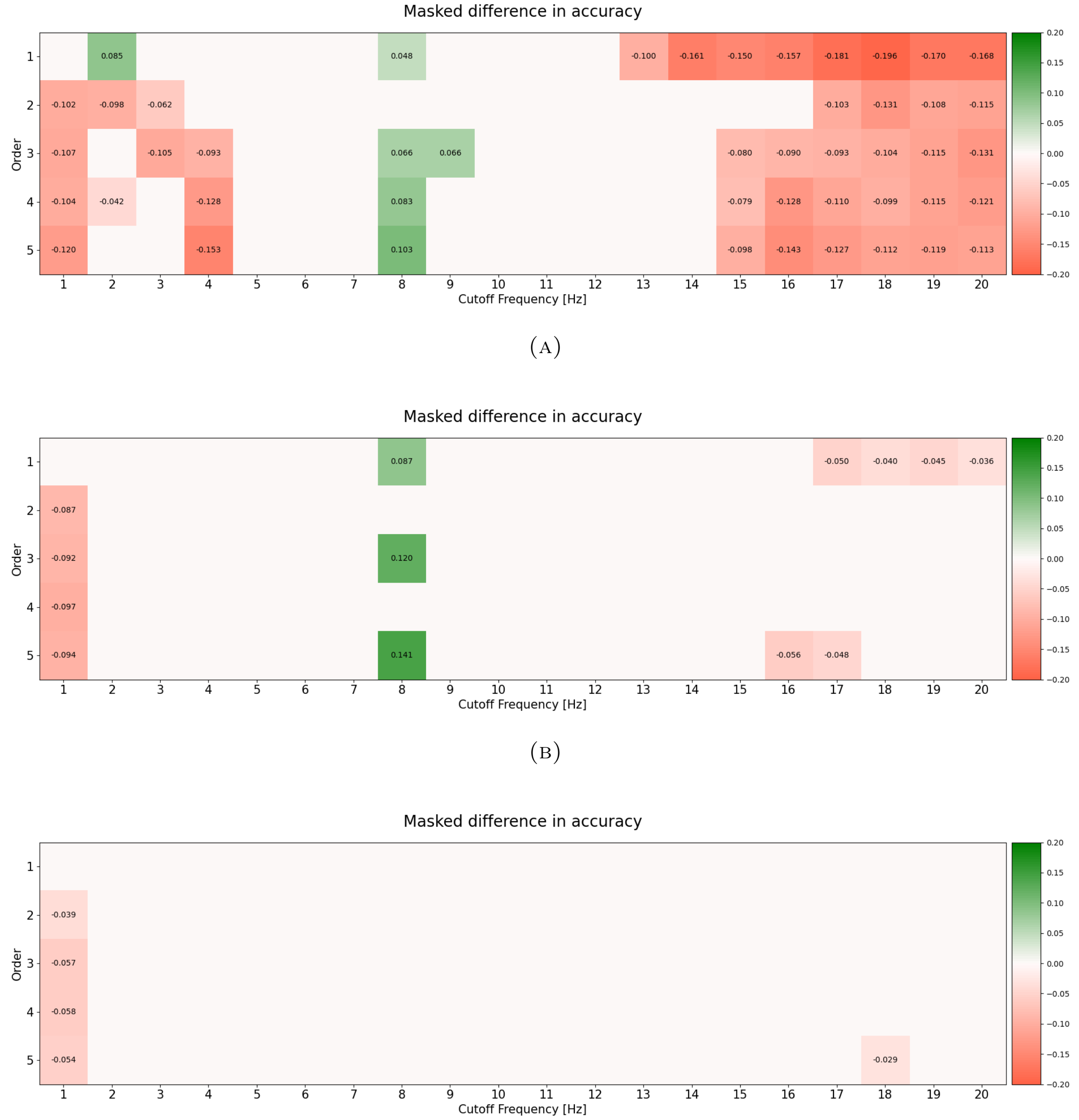
Order	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.102	0.224	0.136	0.516	0.632	0.627	0.458	0.403	0.585	0.352	0.929	0.837	0.930	0.623	0.842	0.770	0.368	0.212	0.104	0.400
2	0.012	0.396	0.813	1.000	0.615	0.728	0.928	0.521	0.485	0.422	0.563	0.599	0.515	0.438	0.901	1.000	0.920	1.000	1.000	0.662
3	0.008	0.704	0.762	0.897	0.671	0.563	0.698	0.701	0.467	0.848	0.357	0.801	0.851	0.408	0.667	0.159	0.193	0.526	0.601	0.324
4	0.016	0.724	0.544	0.783	1.000	0.783	0.525	0.333	0.545	0.787	0.669	0.701	0.571	0.522	1.000	0.634	0.195	0.149	0.809	1.000
5	0.006	0.597	0.226	0.877	0.755	0.722	0.567	0.313	0.666	0.731	0.851	0.931	0.521	0.501	0.679	0.311	0.097	0.008	0.055	0.701

(C)

Figure 12: The p-value of the 5x2 cross validation t-test between random forests trained on smoothed and non-smoothed data using different combinations of sensors. (a) Using the sensor on the left upper leg, (b) Using the sensors on both upper legs combined, (c) Using all sensors.

3.4 Masked difference

These tables show the difference in accuracy again, but only the differences that are significant enough are shown. That is, only the difference is shown when the p-value for the corresponding cell is below our α value of 0.05. We observe negative regions on lower and higher cutoff frequencies, and positive regions in the middle, but only in the tables using few sensors. This suggests that there is a correlation between the cutoff frequency and the performance. However, since the regions are shifted for each sensor, further research needs to be done to find the causation.



(A)

(B)

(C)

Figure 13: The difference in accuracy scores between the random forests on smoothed and non-smoothed data but masked by the significant p-values of below our α of 0.05 using different combinations of sensors. (a) Using the sensor on the left upper leg, (b) Using the sensors on both upper legs combined, (c) Using all sensors.

4 Discussion

First, we will look at the accuracy tables.

Observation 1. The accuracy increases overall when we use more sensors.
The table showing the results using all sensors has the best accuracy scores.

This is not to be unexpected, since there is more information to base a prediction on. The more data we use, the better the basis of the predictions can be. However, providing the algorithm with too much data increases the risk of providing data that is not relevant to the classification. The algorithm will try to find a correlation where there is none. Therefore, it is important to know what data to provide to the machine learning algorithm.

Secondly, we will look at the other three types of tables. Since these tables are closely related to each other, we will discuss them together.

Observation 2. The differences between the smoothed and non-smoothed versions, and their significance, decreases with increasing number of sensors.
The cutoff frequency and order have little to no influence on the performance when enough data is used. Also, the p-values of the differences get smaller the more sensors are used, and the difference in performance that is significant, is negative in all smoothing configurations.

This implies that the smoothing setup is less important and decreases the performance as well when using more sensors.

Observation 3. The cutoff frequency and order do have a high influence on how well the algorithm performs when using just a few sensors.
The setup of the cutoff frequency and order in some tables can mean a difference in performance of more than 30%, for example in the left upper leg. Therefore, the configuration of the cutoff frequency and order for smoothing is highly important when using single sensors. This complements our observations on the tables with a high sensor count.

However, it must be noted that the configuration of the smoothing influences the accuracy differently for every sensor. The areas with a positive difference in for example the left lower leg are a negative difference in the right upper leg when we look at the significant differences. There is general configuration of the cutoff frequency and the order that provides a performance increase for all sensors.

This means that the setup of the smoothing, when using a low sensor count, is dependent on the placement of the sensor, and it implies that the algorithm, is prone to small errors in the setup of the experiment. This is an important observation, since the use of fewer sensors is gaining more popularity [13].

Observation 4. There seems to be a tendency for a middle band of cutoff frequencies to appear in most sensors where the performance increases. For example, there is a general negative difference in performance when using a low cutoff frequency. Nearly all tables have a negative difference in performance in the lower frequency zone. A lower cutoff frequency means that more data is removed, and the signal might get reduced too much by the filter.

This suggests there might be a systematic pattern for the optimization of the smoothing setup.

5 Conclusions

The smoothing, done by a Butterworth low-pass filter, of raw sensor data has a different importance when detecting fatigue with a random forest classifier, depending on the amount of sensors you use.

First of all, the accuracy increases the more sensors are used. The smoothing has little to no significant influence on the performance of the machine learning algorithm when using multiple sensors on different places on the body.

However, the smoothing configuration plays a crucial role in the performance of the algorithm when only a few or single sensors are used. Since there is little data to work with, the algorithm is prone to small errors, and thus the smoothing needs to be setup correctly. The configuration of the cutoff frequency and order for the low-pass filter can create a difference in accuracy of at least 30%.

No apparent optimal way of configuring the cutoff frequency and order for every single sensor is found in this experiment, but the results suggest that there is a systematic pattern in the way the cutoff frequency influences the performance. These patterns are an interesting topic for further analysis, and the methods used in this paper can be helpful in comparing and determining them.

6 Limitations and directions for further research

The different resulting tables for each single sensor is unexplained by this paper. It might be due to an unseen correlation or random interference. Since the usage of less sensors is gaining popularity in research and the world of business [13], growing our understanding about these results, and their differences, is essential.

The results of this study suggest there is a systematic pattern in the optimization of the smoothing when using few sensors. For each individual sensor, there seems to be a range in the cutoff frequencies where there is a significant performance increase when smoothing is applied. The lower cutoff frequencies have a general negative impact on the performance. Our conjecture is that the low cutoff frequency reduces the signal too much, removing too much real data compared to noise. Our method of comparing the performance of an algorithm based on different datasets is not limited to smoothed versus non-smoothed datasets, and might be useful in comparing two differently smoothed datasets as well.

Our next step would be to see if a systematic pattern can be found in the optimization of the smoothing parameters when using individual sensors, but with a different smoothing algorithm. We choose the Butterworth low-pass filter for this experiment due to its popularity in the field of bio-mechanics, but it is not the only smoothing technique. Some examples are local regression and a moving average. A moving average might be a good option to use when analyzing the analytical effects of smoothing due to its simple structure. This might prove easier to use when investigating the systematic patterns. Research towards the effects of other smoothing methods on machine learning performance might give more insight in the actual role of smoothing itself.

A similar argument can be given for the type of machine learning. Another algorithm that is prominent in the bio-mechanics field is the Support vector machine [10]. The way this algorithm reacts to smoothing might be different from a random forest, and further research needs to be done to understand how. However, given the high values of accuracy that our random forests achieved, we conclude that using a random forest is a viable option to detect fatigue nonetheless.

The detection of fatigue in this experiment was done with the magnitude of the acceleration of each sensor. However, there are countless other measures in bio-mechanics to detect fatigue with, for example the knee angle. But these measures require deeper understanding of the human body. A vast amount of research can be done if finding the optimal measurement for detecting fatigue. Additionally, the goal of this paper was to find the effects of smoothing on the performance of machine learning algorithms, not to find the optimal way of detecting fatigue. Finding an optimal measurement to detect fatigue did not fit the scope of this paper. Moreover, we wanted to keep the data as unprocessed as possible. Combining the acceleration from all three directions provides us with a measure for acceleration again. Also, Wang et al. used the same measure in their paper "Fatigue Detection in Running with Inertial Measurement Unit and Machine Learning" [18]. We therefore choose the magnitude of the acceleration of each sensor as our measurement signal.

Similarly, the choice of features can be further researched in the same way. This paper extracted four features from the strides, the minimum value, the maximum value, the difference between those two and the area under the curve. As represented by Figure 8, a visual representation of the features can give an indication to relevant features, which directed us in using the chosen features. A set of features were extracted from the sensors, and features were removed from the set if their removal did not reduce the accuracy greatly.

In this paper, the smoothing was applied directly on the raw sensor data. This is generally done this way, and it seemed logical to keep this up for this experiment. However, one could argue that the smoothing needs to be applied after the calculation of the magnitude of the acceleration, since that is supposed to be a smooth continuous signal as well. The placement of the application of smoothing might be a good subject for further research.

The machine learning algorithm used in this experiment was provided by the sci-kit learn library for python. However, a custom random forest classifier was made from scratch. We could influence every aspect of this classifier and know exactly what is going on. This algorithm was used initially, but was not fully optimized due to the time constraints and it not being our highest priority. This caused the custom algorithm to be drastically slower than the library version. Since our experiment required use to train a random forest at least 12000 times (12 sensor combinations, 20 cutoff frequencies and 5 orders per combination, 10 trainings for each 5x2 cross validation t-test for each smoothing configuration), we could not afford to use our custom algorithm and thus opted for the use of a library. The codebase for this project has ben published online[1].

Finally, we used data from a single run from one subject. We justify this by the fact that we focused on the effects of smoothing on the performance, not the overall performance of the algorithm. To find the differences, we need to use the same data, and thus the same run and subject. However, the results might be unique to the subject. It might be possible that the measurement signals or the features used for this experiment are not sufficient to use on a different subject on a different run. Therefore, extending the experiment to a bigger pool of datasets is an interesting direction for further research.

References

[1] V. Abolins, K. Nesenbergs, and E. Bernans. On improving gait analysis data: heel induced force plate noise removal and cut-off frequency selection for Butterworth filter. In *Proceedings of the 9th International Conference on Signal Processing Systems*, pages 215–219, Nov. 2017.

[1] `https://gitlab.utwente.nl/s2364727/bachelor-project`

[2] L. Breiman. Random Forests. *Machine Learning*, 45(1):5–32, Oct. 2001.

[3] C. Buckley, M. O'Reilly, D. Whelan, A. V. Farrell, L. Clark, V. Longo, M. Gilchrist, and B. Caulfield. Binary classification of running fatigue using a single inertial measurement unit. In *2017 IEEE 14th International Conference on Wearable and Implantable Body Sensor Networks (BSN)*, pages 197–201, May 2017.

[4] I. Buist, S. W. Bredeweg, K. A. P. M. Lemmink, W. van Mechelen, and R. L. Diercks. Predictors of running-related injuries in novice runners enrolled in a systematic training program: A prospective cohort study. *The American Journal of Sports Medicine*, 38(2):273–280, Feb. 2010.

[5] S. Butterworth. On the theory of filter amplifiers. *Wireless Engineer*, 7(6):536–541, 1930.

[6] G. Christodoulakis, K. Busawon, N. Caplan, and S. Stewart. On the filtering and smoothing of biomechanical data. In *2010 7th International Symposium on Communication Systems, Networks & Digital Signal Processing (CSNDSP 2010)*, pages 512–516, July 2010.

[7] T. A. Dierks, I. S. Davis, and J. Hamill. The effects of running in an exerted state on lower extremity kinematics and joint timing. *Journal of Biomechanics*, 43(15):2993–2998, Nov. 2010.

[8] T. G. Dietterich. Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms. *Neural Computation*, 10(7):1895–1923, Sept. 1998.

[9] P. Gulde and J. Hermsdörfer. A Comparison of Smoothing and Filtering Approaches Using Simulated Kinematic Data of Human Movements. In *Proceedings of the 11th International Symposium on Computer Science in Sport (IACSS 2017)*, pages 97–102, 01 2018.

[10] E. Halilaj, A. Rajagopal, M. Fiterau, J. L. Hicks, T. J. Hastie, and S. L. Delp. Machine learning in human movement biomechanics: Best practices, common pitfalls, and new opportunities. *Journal of Biomechanics*, 81:1–11, Nov. 2018.

[11] Y. Halkiadakis, H. M. Alzakerin, and K. D. Morgan. Classification Model for Discriminating Trunk Fatigue During Running. In *2021 43rd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, pages 4546–4549, Nov. 2021.

[12] Y. Hutabarat, D. Owaki, and M. Hayashibe. Seamless Temporal Gait Evaluation during Walking and Running Using Two IMU Sensors. In *2021 43rd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, pages 6835–6840, Nov. 2021.

[13] L. Marotta, J. H. Buurke, B.-J. F. van Beijnum, and J. Reenalda. Towards Machine Learning-Based Detection of Running-Induced Fatigue in Real-World Scenarios: Evaluation of IMU Sensor Configurations to Reduce Intrusiveness. *Sensors*, 21(10):3451, Jan. 2021.

[14] K. J. O'Donovan, B. R. Greene, D. McGrath, R. O'Neill, A. Burns, and B. Caulfield. SHIMMER: A new tool for temporal gait analysis. In *2009 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pages 3826–3829, Sept. 2009.

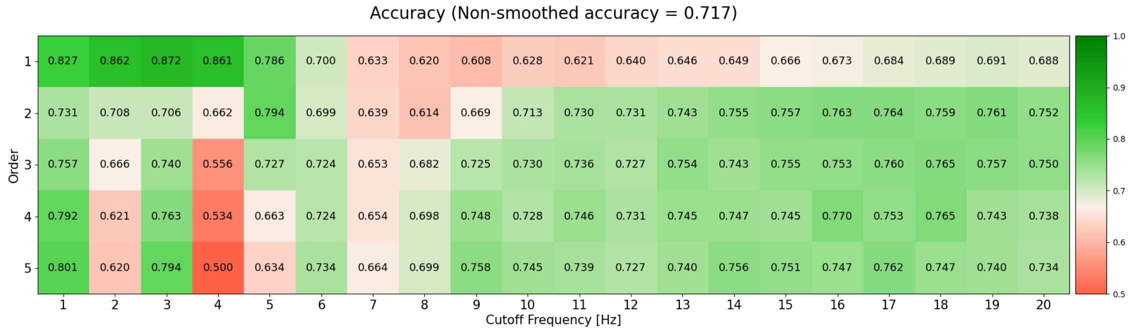
[15] J. Scheerder, K. Breedveld, and J. Borgers. Who Is Doing a Run with the Running Boom? In *Running across Europe: The Rise and Size of One of the Largest Sport Markets*, pages 1–27. Palgrave Macmillan UK, London, 2015.

[16] C. Strohrmann, H. Harms, C. Kappeler-Setz, and G. Troster. Monitoring Kinematic Changes With Fatigue in Running Using Body-Worn Sensors. *IEEE Transactions on Information Technology in Biomedicine*, 16(5):983–990, Sept. 2012.

[17] N. Tam, D. R. Coetzee, S. Ahmed, R. P. Lamberts, Y. Albertus-Kajee, and R. Tucker. Acute fatigue negatively affects risk factors for injury in trained but not well-trained habitually shod runners when running barefoot. *European Journal of Sport Science*, 17(9):1220–1229, Oct. 2017.

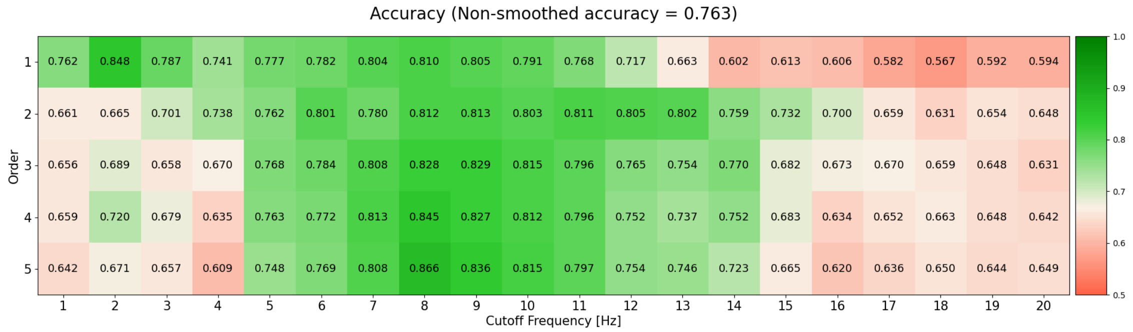
[18] G. Wang, X. Mao, Q. Zhang, and A. Lu. Fatigue Detection in Running with Inertial Measurement Unit and Machine Learning. In *2022 10th International Conference on Bioinformatics and Computational Biology (ICBCB)*, pages 85–90, May 2022.

A Accuracy tables



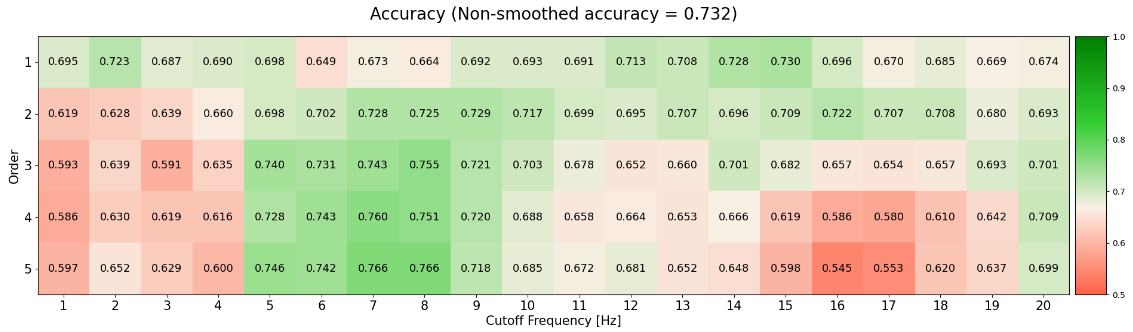
Order	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.827	0.862	0.872	0.861	0.786	0.700	0.633	0.620	0.608	0.628	0.621	0.640	0.646	0.649	0.666	0.673	0.684	0.689	0.691	0.688
2	0.731	0.708	0.706	0.662	0.794	0.699	0.639	0.614	0.669	0.713	0.730	0.731	0.743	0.755	0.757	0.763	0.764	0.759	0.761	0.752
3	0.757	0.666	0.740	0.556	0.727	0.724	0.653	0.682	0.725	0.730	0.736	0.727	0.754	0.743	0.755	0.753	0.760	0.765	0.757	0.750
4	0.792	0.621	0.763	0.534	0.663	0.724	0.654	0.698	0.748	0.728	0.746	0.731	0.745	0.747	0.745	0.770	0.753	0.765	0.743	0.738
5	0.801	0.620	0.794	0.500	0.634	0.734	0.664	0.699	0.758	0.745	0.739	0.727	0.740	0.756	0.751	0.747	0.762	0.747	0.740	0.734

Figure A.1: The accuracy scores of the random forests trained on data with differently configured Butterworth low-pass filters using the sensor on the pelvis.



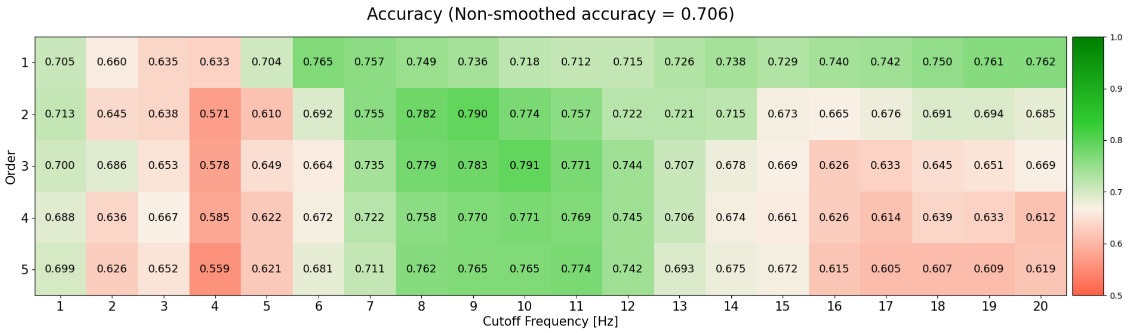
Order	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.762	0.848	0.787	0.741	0.777	0.782	0.804	0.810	0.805	0.791	0.768	0.717	0.663	0.602	0.613	0.606	0.582	0.567	0.592	0.594
2	0.661	0.665	0.701	0.738	0.762	0.801	0.780	0.812	0.813	0.803	0.811	0.805	0.802	0.759	0.732	0.700	0.659	0.631	0.654	0.648
3	0.656	0.689	0.658	0.670	0.768	0.784	0.808	0.828	0.829	0.815	0.796	0.765	0.754	0.770	0.682	0.673	0.670	0.659	0.648	0.631
4	0.659	0.720	0.679	0.635	0.763	0.772	0.813	0.845	0.827	0.812	0.796	0.752	0.737	0.752	0.683	0.634	0.652	0.663	0.648	0.642
5	0.642	0.671	0.657	0.609	0.748	0.769	0.808	0.866	0.836	0.815	0.797	0.754	0.746	0.723	0.665	0.620	0.636	0.650	0.644	0.649

Figure A.2: The accuracy scores of the random forests trained on data with differently configured Butterworth low-pass filters using the sensor on the left upper leg.



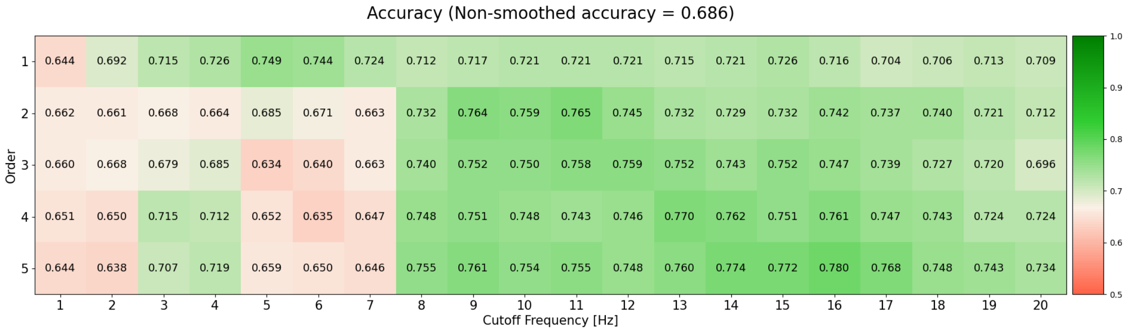
Order	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.695	0.723	0.687	0.690	0.698	0.649	0.673	0.664	0.692	0.693	0.691	0.713	0.708	0.728	0.730	0.696	0.670	0.685	0.669	0.674
2	0.619	0.628	0.639	0.660	0.698	0.702	0.728	0.725	0.729	0.717	0.699	0.695	0.707	0.696	0.709	0.722	0.707	0.708	0.680	0.693
3	0.593	0.639	0.591	0.635	0.740	0.731	0.743	0.755	0.721	0.703	0.678	0.652	0.660	0.701	0.682	0.657	0.654	0.657	0.693	0.701
4	0.586	0.630	0.619	0.616	0.728	0.743	0.760	0.751	0.720	0.688	0.658	0.664	0.653	0.666	0.619	0.586	0.580	0.610	0.642	0.709
5	0.597	0.652	0.629	0.600	0.746	0.742	0.766	0.766	0.718	0.685	0.672	0.681	0.652	0.648	0.598	0.545	0.553	0.620	0.637	0.699

Figure A.3: The accuracy scores of the random forests trained on data with differently configured Butterworth low-pass filters using the sensor on the right upper leg.



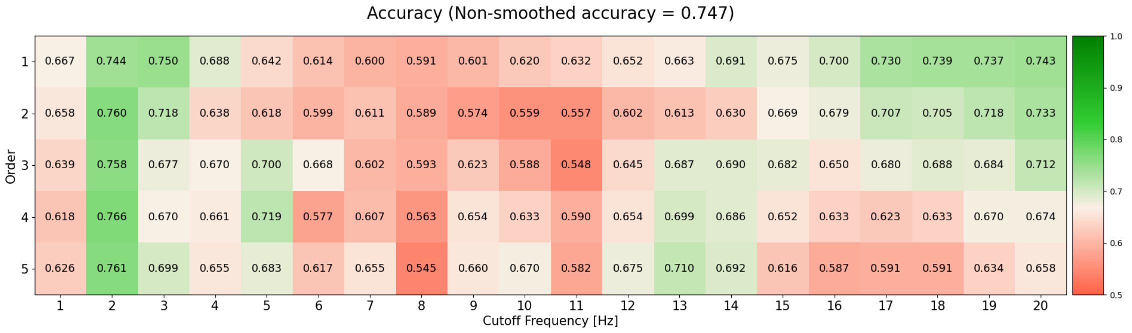
Order	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.705	0.660	0.635	0.633	0.704	0.765	0.757	0.749	0.736	0.718	0.712	0.715	0.726	0.738	0.729	0.740	0.742	0.750	0.761	0.762
2	0.713	0.645	0.638	0.571	0.610	0.692	0.755	0.782	0.790	0.774	0.757	0.722	0.721	0.715	0.673	0.665	0.676	0.691	0.694	0.685
3	0.700	0.686	0.653	0.578	0.649	0.664	0.735	0.779	0.783	0.791	0.771	0.744	0.707	0.678	0.669	0.626	0.633	0.645	0.651	0.669
4	0.688	0.636	0.667	0.585	0.622	0.672	0.722	0.758	0.770	0.771	0.769	0.745	0.706	0.674	0.661	0.626	0.614	0.639	0.633	0.612
5	0.699	0.626	0.652	0.559	0.621	0.681	0.711	0.762	0.765	0.765	0.774	0.742	0.693	0.675	0.672	0.615	0.605	0.607	0.609	0.619

FIGURE A.4: The accuracy scores of the random forests trained on data with differently configured Butterworth low-pass filters using the sensor on the left lower leg.



Order	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.644	0.692	0.715	0.726	0.749	0.744	0.724	0.712	0.717	0.721	0.721	0.721	0.715	0.721	0.726	0.716	0.704	0.706	0.713	0.709
2	0.662	0.661	0.668	0.664	0.685	0.671	0.663	0.732	0.764	0.759	0.765	0.745	0.732	0.729	0.732	0.742	0.737	0.740	0.721	0.712
3	0.660	0.668	0.679	0.685	0.634	0.640	0.663	0.740	0.752	0.750	0.758	0.759	0.752	0.743	0.752	0.747	0.739	0.727	0.720	0.696
4	0.651	0.650	0.715	0.712	0.652	0.635	0.647	0.748	0.751	0.748	0.743	0.746	0.770	0.762	0.751	0.761	0.747	0.743	0.724	0.724
5	0.644	0.638	0.707	0.719	0.659	0.650	0.646	0.755	0.761	0.754	0.755	0.748	0.760	0.774	0.772	0.780	0.768	0.748	0.743	0.734

FIGURE A.5: The accuracy scores of the random forests trained on data with differently configured Butterworth low-pass filters using the sensor on the right lower leg.



Order	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.667	0.744	0.750	0.688	0.642	0.614	0.600	0.591	0.601	0.620	0.632	0.652	0.663	0.691	0.675	0.700	0.730	0.739	0.737	0.743
2	0.658	0.760	0.718	0.638	0.618	0.599	0.611	0.589	0.574	0.559	0.557	0.602	0.613	0.630	0.669	0.679	0.707	0.705	0.718	0.733
3	0.639	0.758	0.677	0.670	0.700	0.668	0.602	0.593	0.623	0.588	0.548	0.645	0.687	0.690	0.682	0.650	0.680	0.688	0.684	0.712
4	0.618	0.766	0.670	0.661	0.719	0.577	0.607	0.563	0.654	0.633	0.590	0.654	0.699	0.686	0.652	0.633	0.623	0.633	0.670	0.674
5	0.626	0.761	0.699	0.655	0.683	0.617	0.655	0.545	0.660	0.670	0.582	0.675	0.710	0.692	0.616	0.587	0.591	0.591	0.634	0.658

FIGURE A.6: The accuracy scores of the random forests trained on data with differently configured Butterworth low-pass filters using the sensor on the left foot.

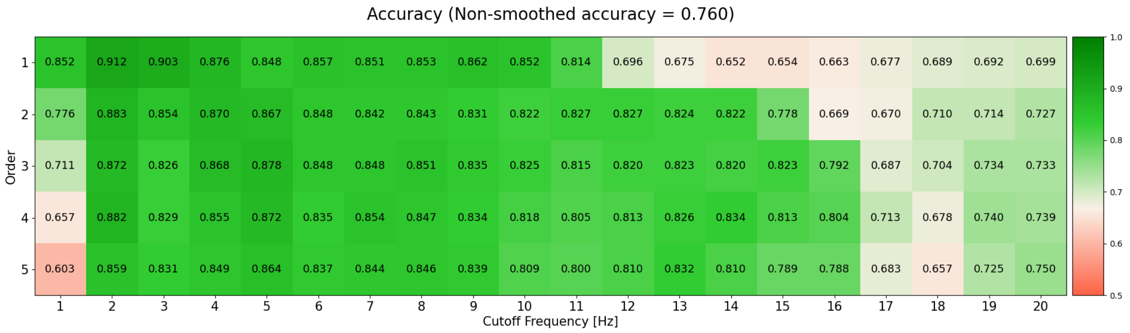


FIGURE A.7: The accuracy scores of the random forests trained on data with differently configured Butterworth low-pass filters using the sensor on the right foot.

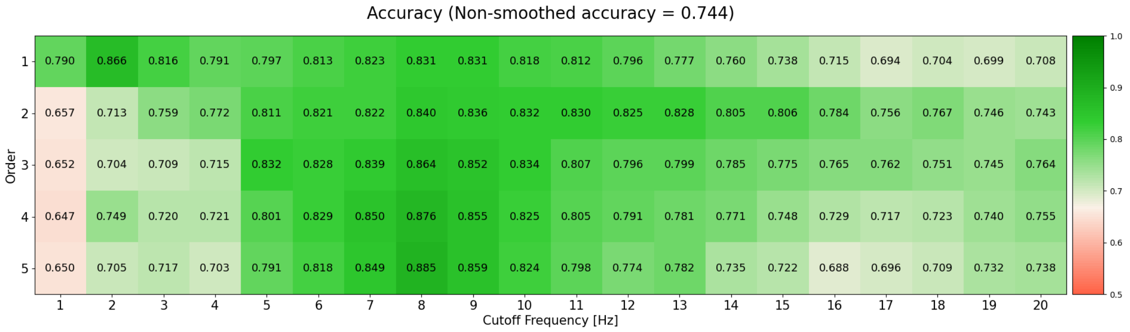


FIGURE A.8: The accuracy scores of the random forests trained on data with differently configured Butterworth low-pass filters using the sensors on both the upper legs.

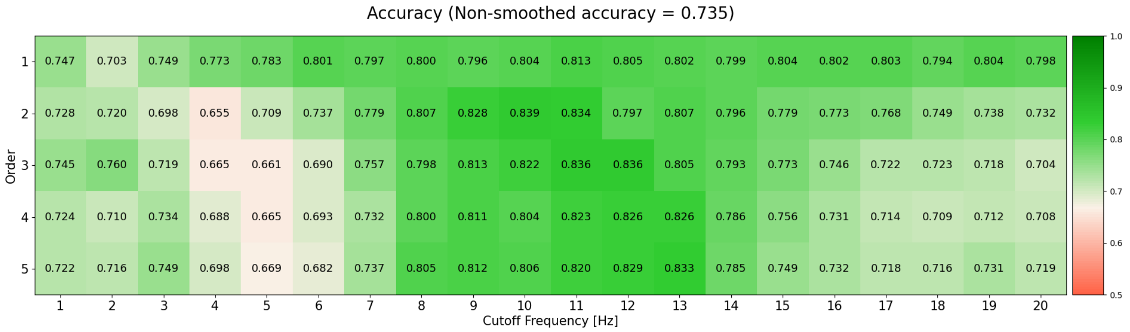
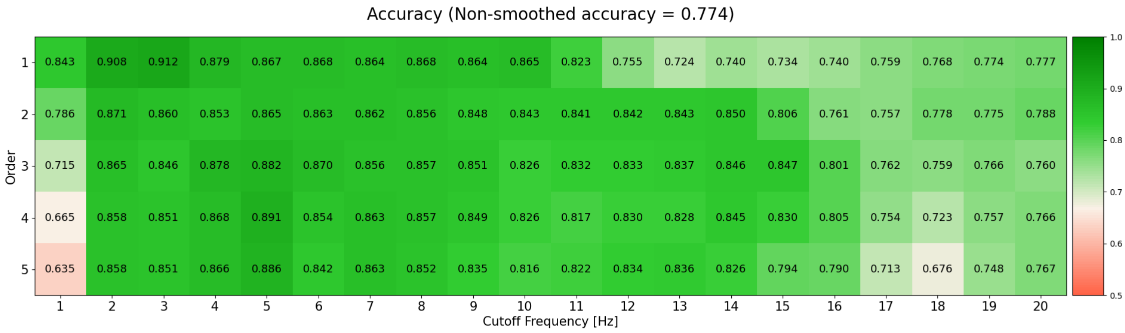
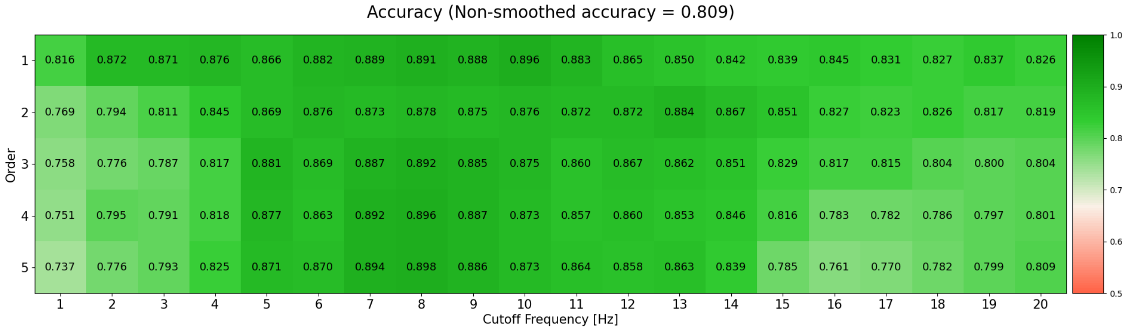


FIGURE A.9: The accuracy scores of the random forests trained on data with differently configured Butterworth low-pass filters using the sensors on both the lower legs.



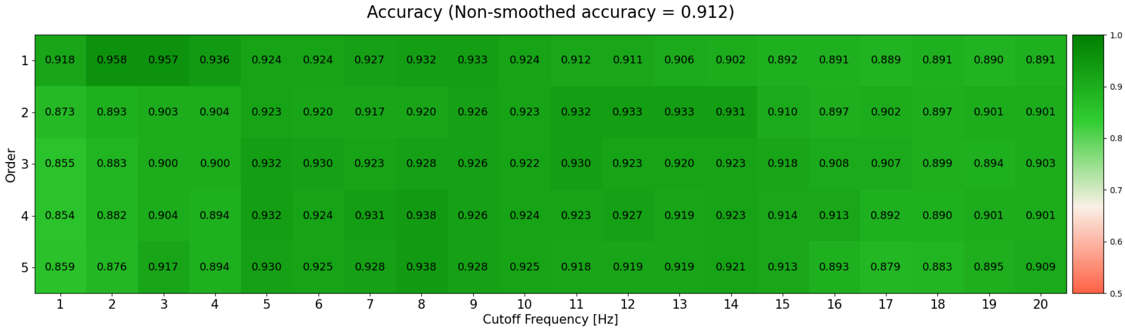
Order	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.843	0.908	0.912	0.879	0.867	0.868	0.864	0.868	0.864	0.865	0.823	0.755	0.724	0.740	0.734	0.740	0.759	0.768	0.774	0.777
2	0.786	0.871	0.860	0.853	0.865	0.863	0.862	0.856	0.848	0.843	0.841	0.842	0.843	0.850	0.806	0.761	0.757	0.778	0.775	0.788
3	0.715	0.865	0.846	0.878	0.882	0.870	0.856	0.857	0.851	0.826	0.832	0.833	0.837	0.846	0.847	0.801	0.762	0.759	0.766	0.760
4	0.665	0.858	0.851	0.868	0.891	0.854	0.863	0.857	0.849	0.826	0.817	0.830	0.828	0.845	0.830	0.805	0.754	0.723	0.757	0.766
5	0.635	0.858	0.851	0.866	0.886	0.842	0.863	0.852	0.835	0.816	0.822	0.834	0.836	0.826	0.794	0.790	0.713	0.676	0.748	0.767

Figure A.10: The accuracy scores of the random forests trained on data with differently configured Butterworth low-pass filters using the sensors on both feet.



Order	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.816	0.872	0.871	0.876	0.866	0.882	0.889	0.891	0.888	0.896	0.883	0.865	0.850	0.842	0.839	0.845	0.831	0.827	0.837	0.826
2	0.769	0.794	0.811	0.845	0.869	0.876	0.873	0.878	0.875	0.876	0.872	0.872	0.884	0.867	0.851	0.827	0.823	0.826	0.817	0.819
3	0.758	0.776	0.787	0.817	0.881	0.869	0.887	0.892	0.885	0.875	0.860	0.867	0.862	0.851	0.829	0.817	0.815	0.804	0.800	0.804
4	0.751	0.795	0.791	0.818	0.877	0.863	0.892	0.896	0.887	0.873	0.857	0.860	0.853	0.846	0.816	0.783	0.782	0.786	0.797	0.801
5	0.737	0.776	0.793	0.825	0.871	0.870	0.894	0.898	0.886	0.873	0.864	0.858	0.863	0.839	0.785	0.761	0.770	0.782	0.799	0.809

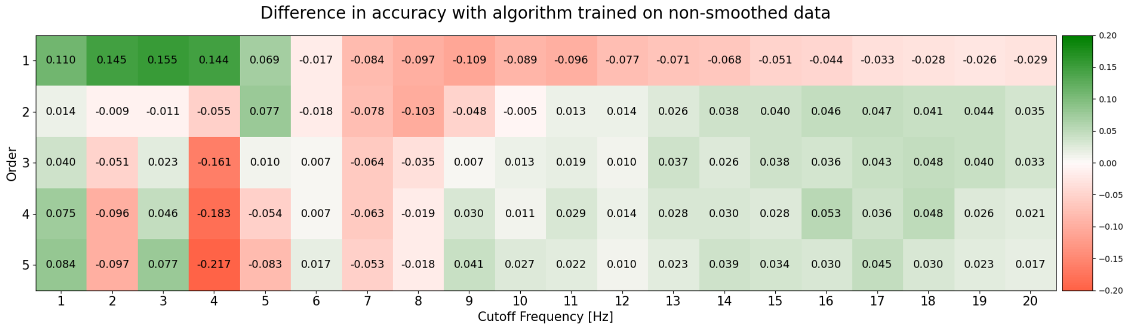
Figure A.11: The accuracy scores of the random forests trained on data with differently configured Butterworth low-pass filters using the 4 sensors on both upper and lower legs.



Order	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.918	0.958	0.957	0.936	0.924	0.924	0.927	0.932	0.933	0.924	0.912	0.911	0.906	0.902	0.892	0.891	0.889	0.891	0.890	0.891
2	0.873	0.893	0.903	0.904	0.923	0.920	0.917	0.920	0.926	0.923	0.932	0.933	0.933	0.931	0.910	0.897	0.902	0.897	0.901	0.901
3	0.855	0.883	0.900	0.900	0.932	0.930	0.923	0.928	0.926	0.922	0.930	0.923	0.920	0.923	0.918	0.908	0.907	0.899	0.894	0.903
4	0.854	0.882	0.904	0.894	0.932	0.924	0.931	0.938	0.926	0.924	0.923	0.927	0.919	0.923	0.914	0.913	0.892	0.890	0.901	0.901
5	0.859	0.876	0.917	0.894	0.930	0.925	0.928	0.938	0.928	0.925	0.918	0.919	0.919	0.921	0.913	0.893	0.879	0.883	0.895	0.909

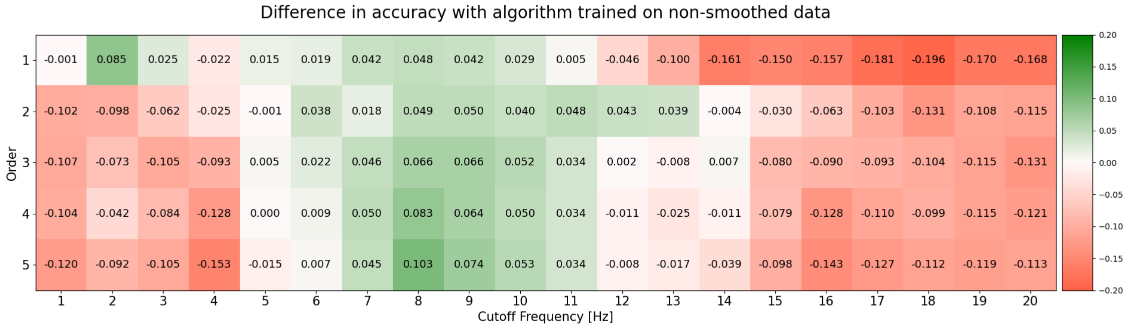
Figure A.12: The accuracy scores of the random forests trained on data with differently configured Butterworth low-pass filters using all sensors.

B Accuracy difference tables



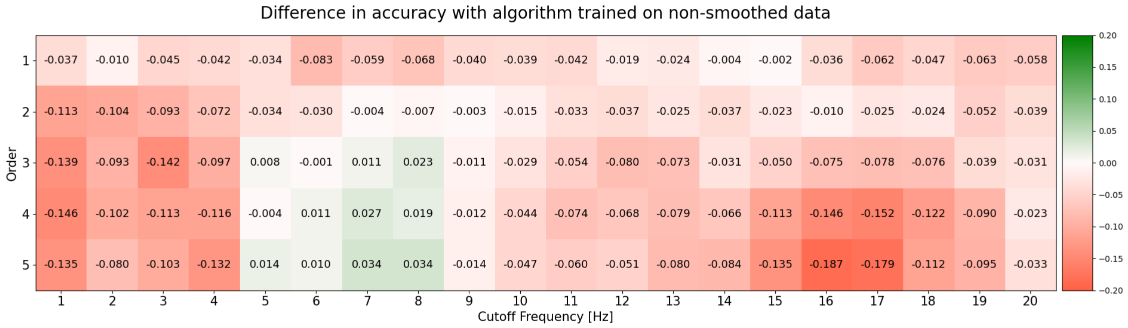
Order	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.110	0.145	0.155	0.144	0.069	-0.017	-0.084	-0.097	-0.109	-0.089	-0.096	-0.077	-0.071	-0.068	-0.051	-0.044	-0.033	-0.028	-0.026	-0.029
2	0.014	-0.009	-0.011	-0.055	0.077	-0.018	-0.078	-0.103	-0.048	-0.005	0.013	0.014	0.026	0.038	0.040	0.046	0.047	0.041	0.044	0.035
3	0.040	-0.051	0.023	-0.161	0.010	0.007	-0.064	-0.035	0.007	0.013	0.019	0.010	0.037	0.026	0.038	0.036	0.043	0.048	0.040	0.033
4	0.075	-0.096	0.046	-0.183	-0.054	0.007	-0.063	-0.019	0.030	0.011	0.029	0.014	0.028	0.030	0.028	0.053	0.036	0.048	0.026	0.021
5	0.084	-0.097	0.077	-0.217	-0.083	0.017	-0.053	-0.018	0.041	0.027	0.022	0.010	0.023	0.039	0.034	0.030	0.045	0.030	0.023	0.017

Figure B.1: The difference in accuracy scores between the random forests on smoothed and non-smoothed data using the sensor on the pelvis.



Order	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	-0.001	0.085	0.025	-0.022	0.015	0.019	0.042	0.048	0.042	0.029	0.005	-0.046	-0.100	-0.161	-0.150	-0.157	-0.181	-0.196	-0.170	-0.168
2	-0.102	-0.098	-0.062	-0.025	-0.001	0.038	0.018	0.049	0.050	0.040	0.048	0.043	0.039	-0.004	-0.030	-0.063	-0.103	-0.131	-0.108	-0.115
3	-0.107	-0.073	-0.105	-0.093	0.005	0.022	0.046	0.066	0.066	0.052	0.034	0.002	-0.008	0.007	-0.080	-0.090	-0.093	-0.104	-0.115	-0.131
4	-0.104	-0.042	-0.084	-0.128	0.000	0.009	0.050	0.083	0.064	0.050	0.034	-0.011	-0.025	-0.011	-0.079	-0.128	-0.110	-0.099	-0.115	-0.121
5	-0.120	-0.092	-0.105	-0.153	-0.015	0.007	0.045	0.103	0.074	0.053	0.034	-0.008	-0.017	-0.039	-0.098	-0.143	-0.127	-0.112	-0.119	-0.113

Figure B.2: The difference in accuracy scores between the random forests on smoothed and non-smoothed data using the sensor on the left upper leg.



Order	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	-0.037	-0.010	-0.045	-0.042	-0.034	-0.083	-0.059	-0.068	-0.040	-0.039	-0.042	-0.019	-0.024	-0.004	-0.002	-0.036	-0.062	-0.047	-0.063	-0.058
2	-0.113	-0.104	-0.093	-0.072	-0.034	-0.030	-0.004	-0.007	-0.003	-0.015	-0.033	-0.037	-0.025	-0.037	-0.023	-0.010	-0.025	-0.024	-0.052	-0.039
3	-0.139	-0.093	-0.142	-0.097	0.008	-0.001	0.011	0.023	-0.011	-0.029	-0.054	-0.080	-0.073	-0.031	-0.050	-0.075	-0.078	-0.076	-0.039	-0.031
4	-0.146	-0.102	-0.113	-0.116	-0.004	0.011	0.027	0.019	-0.012	-0.044	-0.074	-0.068	-0.079	-0.066	-0.113	-0.146	-0.152	-0.122	-0.090	-0.023
5	-0.135	-0.080	-0.103	-0.132	0.014	0.010	0.034	0.034	-0.014	-0.047	-0.060	-0.051	-0.080	-0.084	-0.135	-0.187	-0.179	-0.112	-0.095	-0.033

Figure B.3: The difference in accuracy scores between the random forests on smoothed and non-smoothed data using the sensor on the right upper leg.

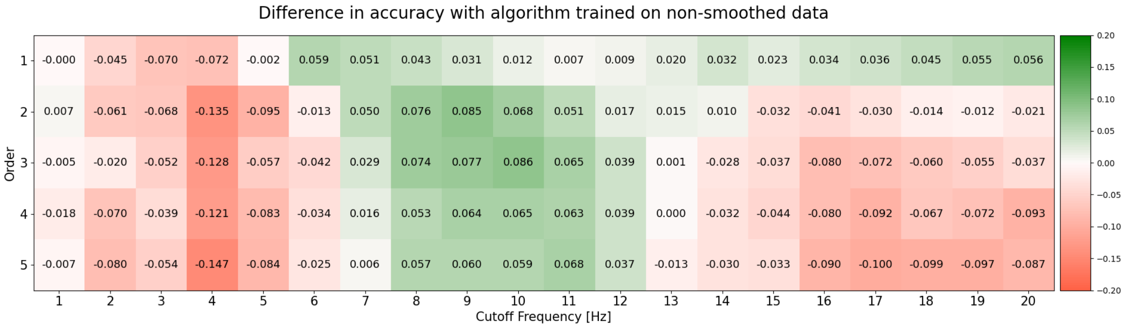


Figure B.4: The difference in accuracy scores between the random forests on smoothed and non-smoothed data using the sensor on the left lower leg.

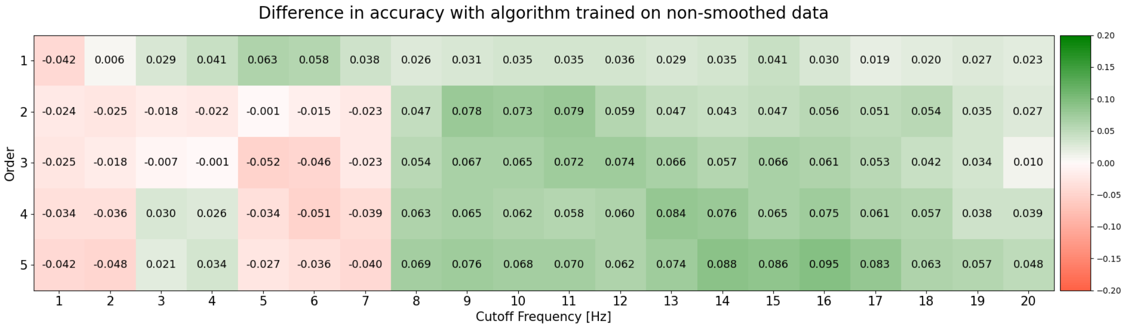


Figure B.5: The difference in accuracy scores between the random forests on smoothed and non-smoothed data using the sensor on the right lower leg.

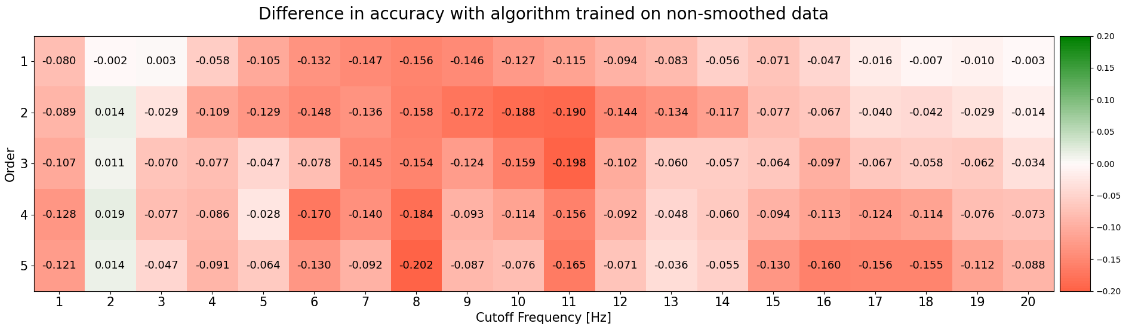


Figure B.6: The difference in accuracy scores between the random forests on smoothed and non-smoothed data using the sensor on the left foot.

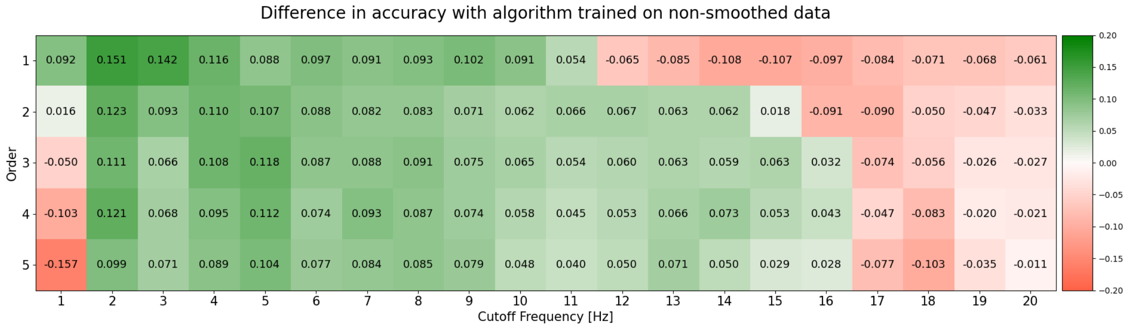


FIGURE B.7: The difference in accuracy scores between the random forests on smoothed and non-smoothed data using the sensor on the right foot.

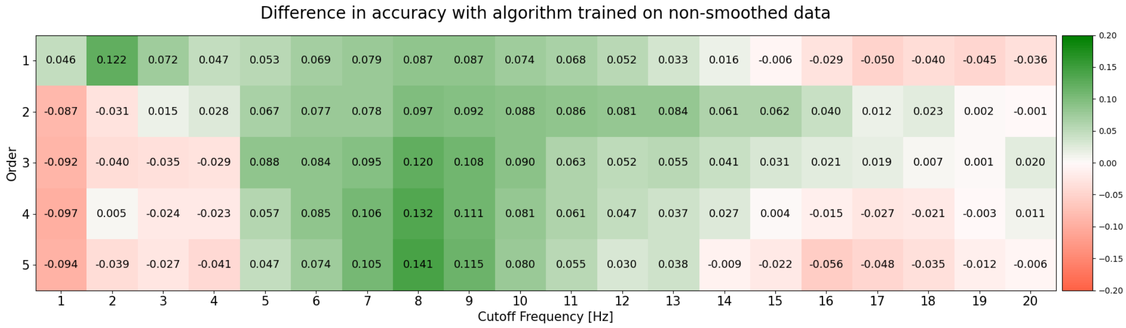


FIGURE B.8: The difference in accuracy scores between the random forests on smoothed and non-smoothed data using the sensors on both the upper legs.

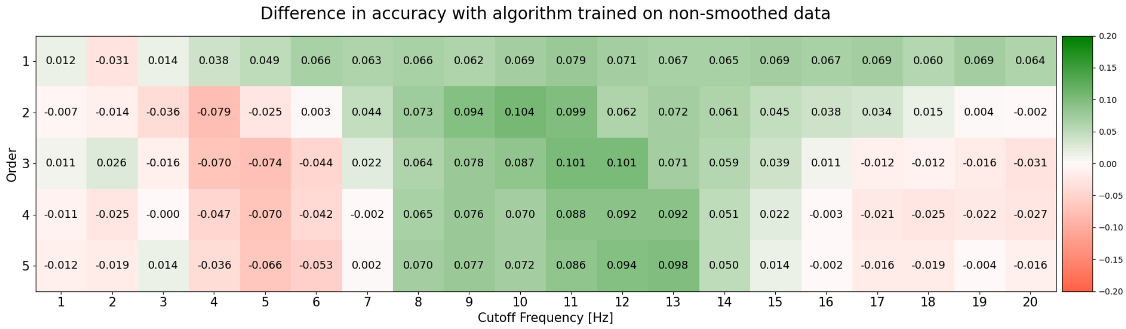


FIGURE B.9: The difference in accuracy scores between the random forests on smoothed and non-smoothed data using the sensors on both the lower legs.

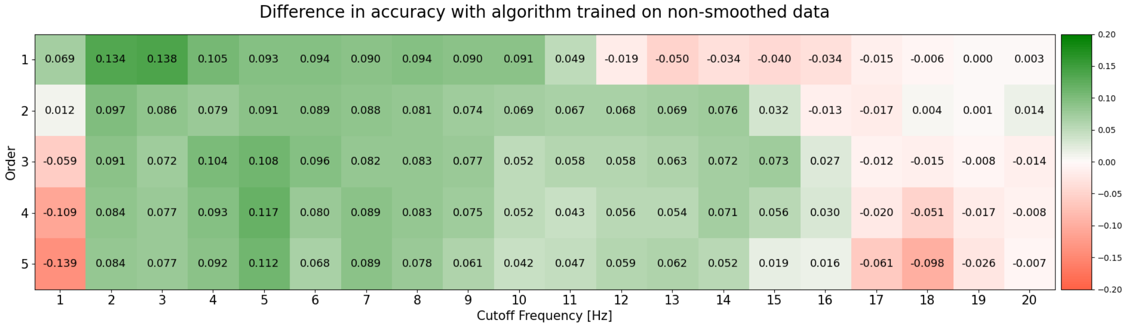


FIGURE B.10: The difference in accuracy scores between the random forests on smoothed and non-smoothed data using the sensors on both feet.

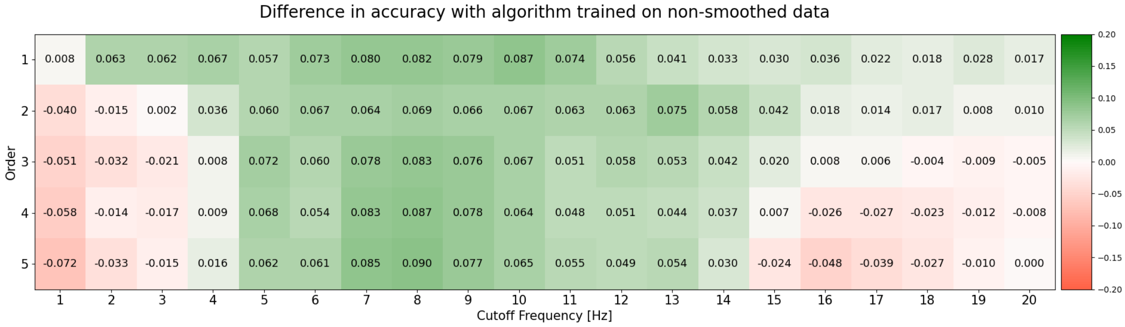


FIGURE B.11: The difference in accuracy scores between the random forests on smoothed and non-smoothed data using the 4 sensors on both upper and lower legs.

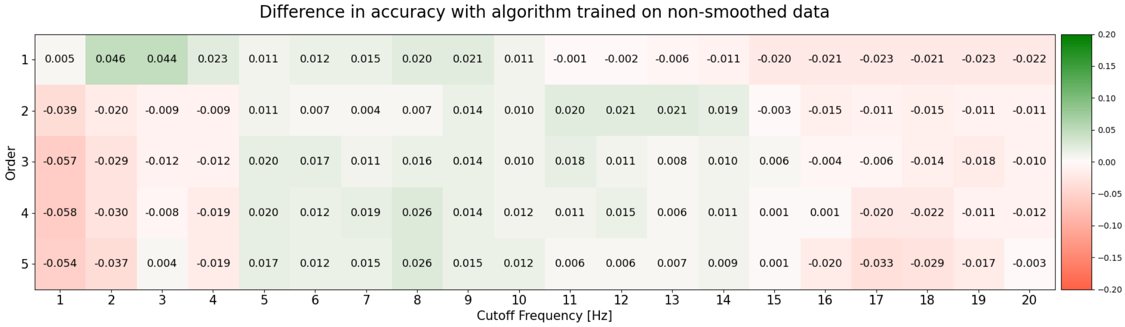
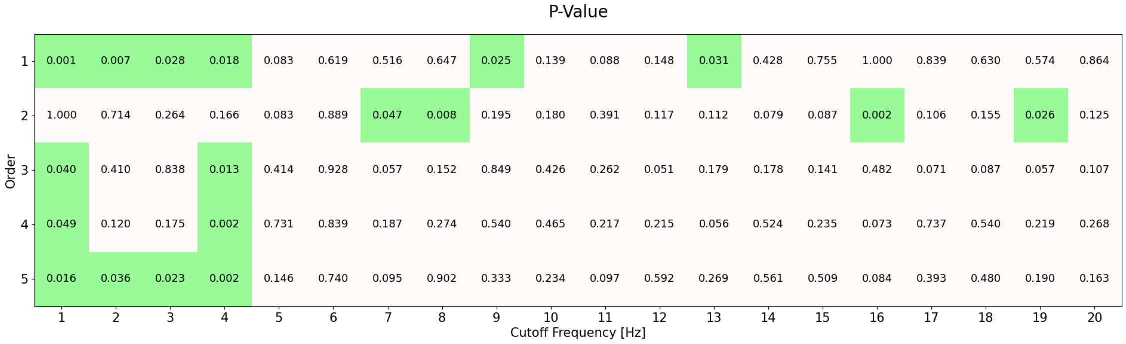


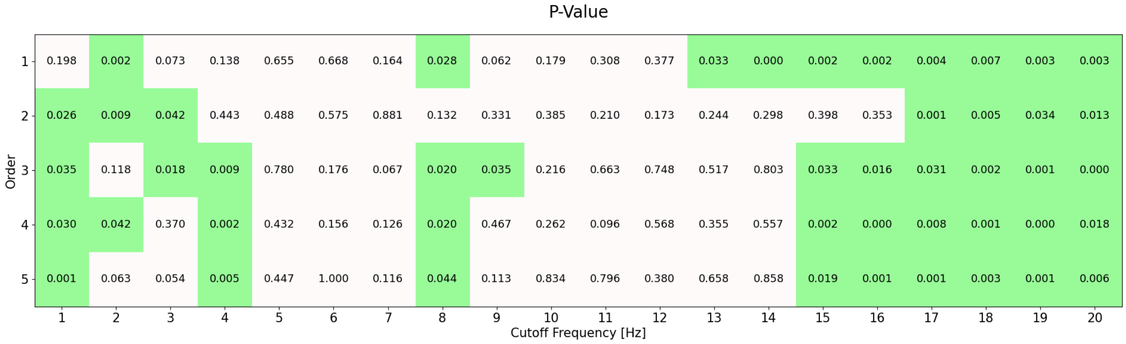
FIGURE B.12: The difference in accuracy scores between the random forests on smoothed and non-smoothed data using all sensors.

C P-Value tables



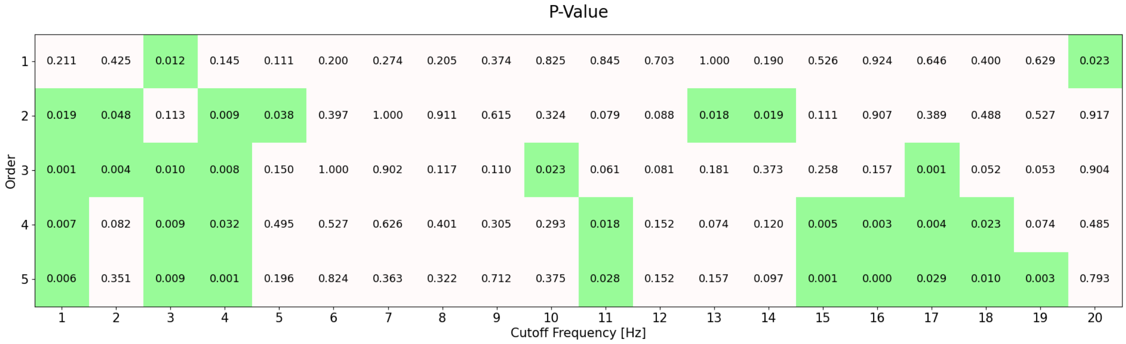
Order	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.001	0.007	0.028	0.018	0.083	0.619	0.516	0.647	0.025	0.139	0.088	0.148	0.031	0.428	0.755	1.000	0.839	0.630	0.574	0.864
2	1.000	0.714	0.264	0.166	0.083	0.889	0.047	0.008	0.195	0.180	0.391	0.117	0.112	0.079	0.087	0.002	0.106	0.155	0.026	0.125
3	0.040	0.410	0.838	0.013	0.414	0.928	0.057	0.152	0.849	0.426	0.262	0.051	0.179	0.178	0.141	0.482	0.071	0.087	0.057	0.107
4	0.049	0.120	0.175	0.002	0.731	0.839	0.187	0.274	0.540	0.465	0.217	0.215	0.056	0.524	0.235	0.073	0.737	0.540	0.219	0.268
5	0.016	0.036	0.023	0.002	0.146	0.740	0.095	0.902	0.333	0.234	0.097	0.592	0.269	0.561	0.509	0.084	0.393	0.480	0.190	0.163

Figure C.1: The p-value of the 5x2 cross validation t-test between random forests trained on smoothed and non-smoothed data using the sensor on the pelvis.



Order	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.198	0.002	0.073	0.138	0.655	0.668	0.164	0.028	0.062	0.179	0.308	0.377	0.033	0.000	0.002	0.002	0.004	0.007	0.003	0.003
2	0.026	0.009	0.042	0.443	0.488	0.575	0.881	0.132	0.331	0.385	0.210	0.173	0.244	0.298	0.398	0.353	0.001	0.005	0.034	0.013
3	0.035	0.118	0.018	0.009	0.780	0.176	0.067	0.020	0.035	0.216	0.663	0.748	0.517	0.803	0.033	0.016	0.031	0.002	0.001	0.000
4	0.030	0.042	0.370	0.002	0.432	0.156	0.126	0.020	0.467	0.262	0.096	0.568	0.355	0.557	0.002	0.000	0.008	0.001	0.000	0.018
5	0.001	0.063	0.054	0.005	0.447	1.000	0.116	0.044	0.113	0.834	0.796	0.380	0.658	0.858	0.019	0.001	0.001	0.003	0.001	0.006

Figure C.2: The p-value of the 5x2 cross validation t-test between random forests trained on smoothed and non-smoothed data using the sensor on the left upper leg.



Order	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.211	0.425	0.012	0.145	0.111	0.200	0.274	0.205	0.374	0.825	0.845	0.703	1.000	0.190	0.526	0.924	0.646	0.400	0.629	0.023
2	0.019	0.048	0.113	0.009	0.038	0.397	1.000	0.911	0.615	0.324	0.079	0.088	0.018	0.019	0.111	0.907	0.389	0.488	0.527	0.917
3	0.001	0.004	0.010	0.008	0.150	1.000	0.902	0.117	0.110	0.023	0.061	0.081	0.181	0.373	0.258	0.157	0.001	0.052	0.053	0.904
4	0.007	0.082	0.009	0.032	0.495	0.527	0.626	0.401	0.305	0.293	0.018	0.152	0.074	0.120	0.005	0.003	0.004	0.023	0.074	0.485
5	0.006	0.351	0.009	0.001	0.196	0.824	0.363	0.322	0.712	0.375	0.028	0.152	0.157	0.097	0.001	0.000	0.029	0.010	0.003	0.793

Figure C.3: The p-value of the 5x2 cross validation t-test between random forests trained on smoothed and non-smoothed data using the sensor on the right upper leg.

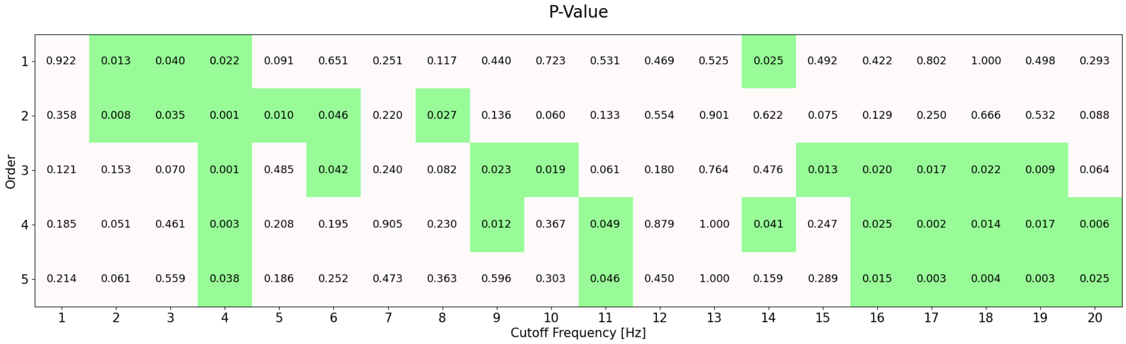


Figure C.4: The p-value of the 5x2 cross validation t-test between random forests trained on smoothed and non-smoothed data using the sensor on the left lower leg.

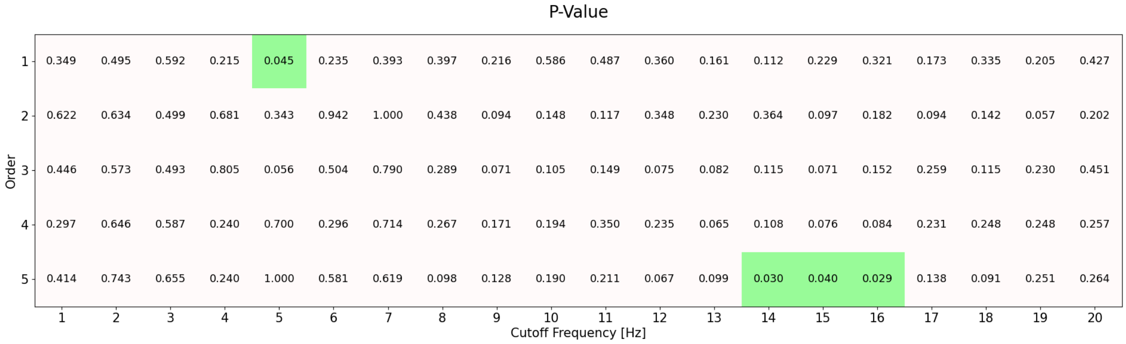


Figure C.5: The p-value of the 5x2 cross validation t-test between random forests trained on smoothed and non-smoothed data using the sensor on the right lower leg.

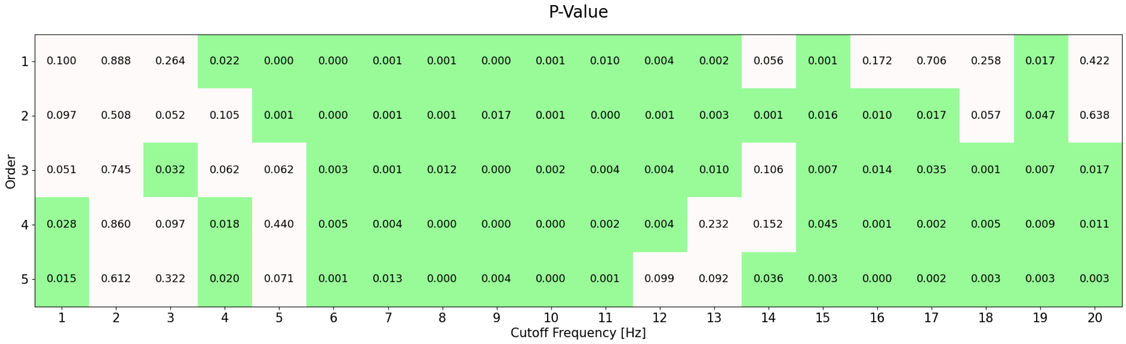
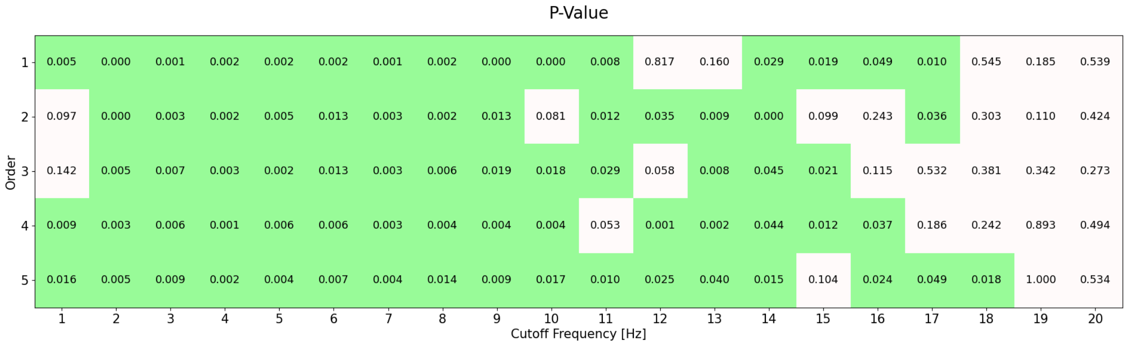
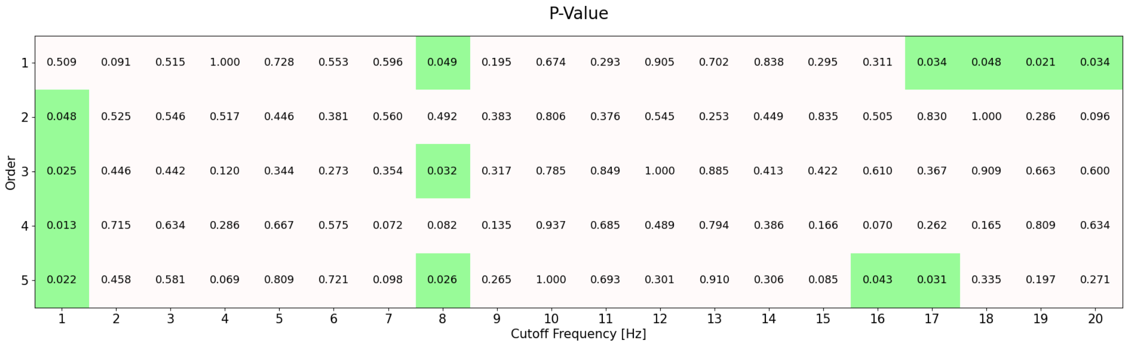


Figure C.6: The p-value of the 5x2 cross validation t-test between random forests trained on smoothed and non-smoothed data using the sensor on the left foot.



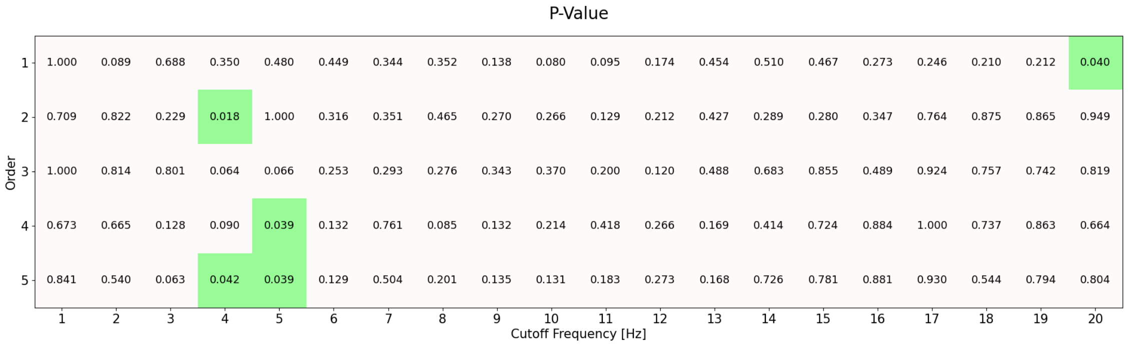
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.005	0.000	0.001	0.002	0.002	0.002	0.001	0.002	0.000	0.000	0.008	0.817	0.160	0.029	0.019	0.049	0.010	0.545	0.185	0.539
2	0.097	0.000	0.003	0.002	0.005	0.013	0.003	0.002	0.013	0.081	0.012	0.035	0.009	0.000	0.099	0.243	0.036	0.303	0.110	0.424
3	0.142	0.005	0.007	0.003	0.002	0.013	0.003	0.006	0.019	0.018	0.029	0.058	0.008	0.045	0.021	0.115	0.532	0.381	0.342	0.273
4	0.009	0.003	0.006	0.001	0.006	0.006	0.003	0.004	0.004	0.004	0.053	0.001	0.002	0.044	0.012	0.037	0.186	0.242	0.893	0.494
5	0.016	0.005	0.009	0.002	0.004	0.007	0.004	0.014	0.009	0.017	0.010	0.025	0.040	0.015	0.104	0.024	0.049	0.018	1.000	0.534

FIGURE C.7: The p-value of the 5x2 cross validation t-test between random forests trained on smoothed and non-smoothed data using the sensor on the right foot.



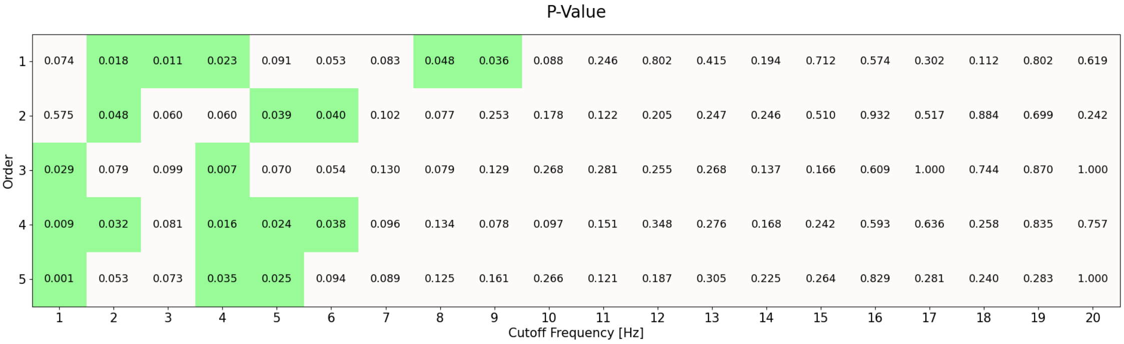
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.509	0.091	0.515	1.000	0.728	0.553	0.596	0.049	0.195	0.674	0.293	0.905	0.702	0.838	0.295	0.311	0.034	0.048	0.021	0.034
2	0.048	0.525	0.546	0.517	0.446	0.381	0.560	0.492	0.383	0.806	0.376	0.545	0.253	0.449	0.835	0.505	0.830	1.000	0.286	0.096
3	0.025	0.446	0.442	0.120	0.344	0.273	0.354	0.032	0.317	0.785	0.849	1.000	0.885	0.413	0.422	0.610	0.367	0.909	0.663	0.600
4	0.013	0.715	0.634	0.286	0.667	0.575	0.072	0.082	0.135	0.937	0.685	0.489	0.794	0.386	0.166	0.070	0.262	0.165	0.809	0.634
5	0.022	0.458	0.581	0.069	0.809	0.721	0.098	0.026	0.265	1.000	0.693	0.301	0.910	0.306	0.085	0.043	0.031	0.335	0.197	0.271

FIGURE C.8: The p-value of the 5x2 cross validation t-test between random forests trained on smoothed and non-smoothed data using the sensors on both the upper legs.



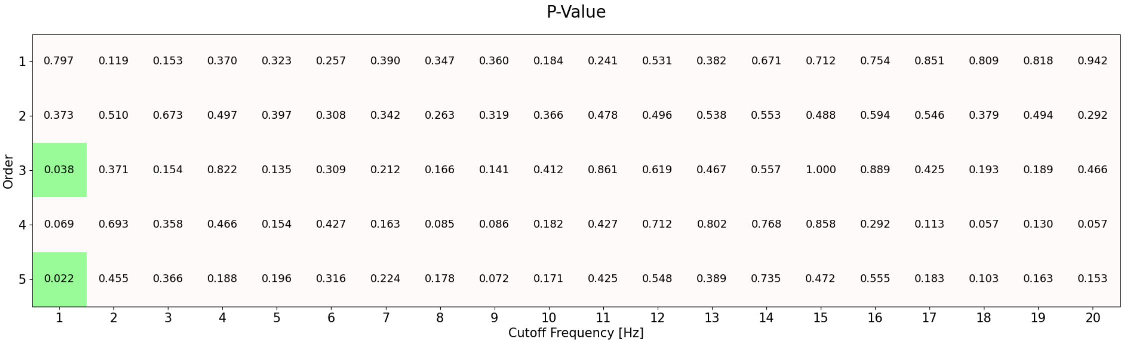
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	1.000	0.089	0.688	0.350	0.480	0.449	0.344	0.352	0.138	0.080	0.095	0.174	0.454	0.510	0.467	0.273	0.246	0.210	0.212	0.040
2	0.709	0.822	0.229	0.018	1.000	0.316	0.351	0.465	0.270	0.266	0.129	0.212	0.427	0.289	0.280	0.347	0.764	0.875	0.865	0.949
3	1.000	0.814	0.801	0.064	0.066	0.253	0.293	0.276	0.343	0.370	0.200	0.120	0.488	0.683	0.855	0.489	0.924	0.757	0.742	0.819
4	0.673	0.665	0.128	0.090	0.039	0.132	0.761	0.085	0.132	0.214	0.418	0.266	0.169	0.414	0.724	0.884	1.000	0.737	0.863	0.664
5	0.841	0.540	0.063	0.042	0.039	0.129	0.504	0.201	0.135	0.131	0.183	0.273	0.168	0.726	0.781	0.881	0.930	0.544	0.794	0.804

FIGURE C.9: The p-value of the 5x2 cross validation t-test between random forests trained on smoothed and non-smoothed data using the sensors on both the lower legs.



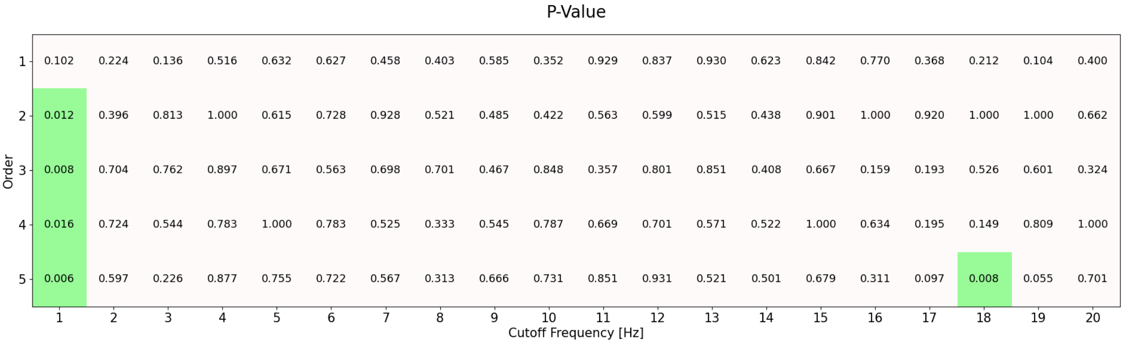
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.074	0.018	0.011	0.023	0.091	0.053	0.083	0.048	0.036	0.088	0.246	0.802	0.415	0.194	0.712	0.574	0.302	0.112	0.802	0.619
2	0.575	0.048	0.060	0.060	0.039	0.040	0.102	0.077	0.253	0.178	0.122	0.205	0.247	0.246	0.510	0.932	0.517	0.884	0.699	0.242
3	0.029	0.079	0.099	0.007	0.070	0.054	0.130	0.079	0.129	0.268	0.281	0.255	0.268	0.137	0.166	0.609	1.000	0.744	0.870	1.000
4	0.009	0.032	0.081	0.016	0.024	0.038	0.096	0.134	0.078	0.097	0.151	0.348	0.276	0.168	0.242	0.593	0.636	0.258	0.835	0.757
5	0.001	0.053	0.073	0.035	0.025	0.094	0.089	0.125	0.161	0.266	0.121	0.187	0.305	0.225	0.264	0.829	0.281	0.240	0.283	1.000

FIGURE C.10: The p-value of the 5x2 cross validation t-test between random forests trained on smoothed and non-smoothed data using the sensors on both feet.



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.797	0.119	0.153	0.370	0.323	0.257	0.390	0.347	0.360	0.184	0.241	0.531	0.382	0.671	0.712	0.754	0.851	0.809	0.818	0.942
2	0.373	0.510	0.673	0.497	0.397	0.308	0.342	0.263	0.319	0.366	0.478	0.496	0.538	0.553	0.488	0.594	0.546	0.379	0.494	0.292
3	0.038	0.371	0.154	0.822	0.135	0.309	0.212	0.166	0.141	0.412	0.861	0.619	0.467	0.557	1.000	0.889	0.425	0.193	0.189	0.466
4	0.069	0.693	0.358	0.466	0.154	0.427	0.163	0.085	0.086	0.182	0.427	0.712	0.802	0.768	0.858	0.292	0.113	0.057	0.130	0.057
5	0.022	0.455	0.366	0.188	0.196	0.316	0.224	0.178	0.072	0.171	0.425	0.548	0.389	0.735	0.472	0.555	0.183	0.103	0.163	0.153

FIGURE C.11: The p-value of the 5x2 cross validation t-test between random forests trained on smoothed and non-smoothed data using the 4 sensors on both upper and lower legs.



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0.102	0.224	0.136	0.516	0.632	0.627	0.458	0.403	0.585	0.352	0.929	0.837	0.930	0.623	0.842	0.770	0.368	0.212	0.104	0.400
2	0.012	0.396	0.813	1.000	0.615	0.728	0.928	0.521	0.485	0.422	0.563	0.599	0.515	0.438	0.901	1.000	0.920	1.000	1.000	0.662
3	0.008	0.704	0.762	0.897	0.671	0.563	0.698	0.701	0.467	0.848	0.357	0.801	0.851	0.408	0.667	0.159	0.193	0.526	0.601	0.324
4	0.016	0.724	0.544	0.783	1.000	0.783	0.525	0.333	0.545	0.787	0.669	0.701	0.571	0.522	1.000	0.634	0.195	0.149	0.809	1.000
5	0.006	0.597	0.226	0.877	0.755	0.722	0.567	0.313	0.666	0.731	0.851	0.931	0.521	0.501	0.679	0.311	0.097	0.008	0.055	0.701

FIGURE C.12: The p-value of the 5x2 cross validation t-test between random forests trained on smoothed and non-smoothed data using all sensors.

D Masked accuracy difference tables

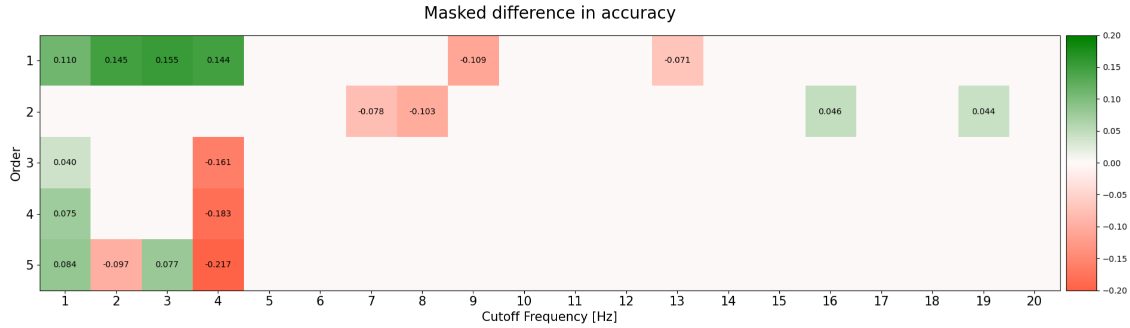


Figure D.1: The difference in accuracy scores between the random forests on smoothed and non-smoothed data but masked by the significant p-values of below our alpha of 0.05 using the sensor on the pelvis.

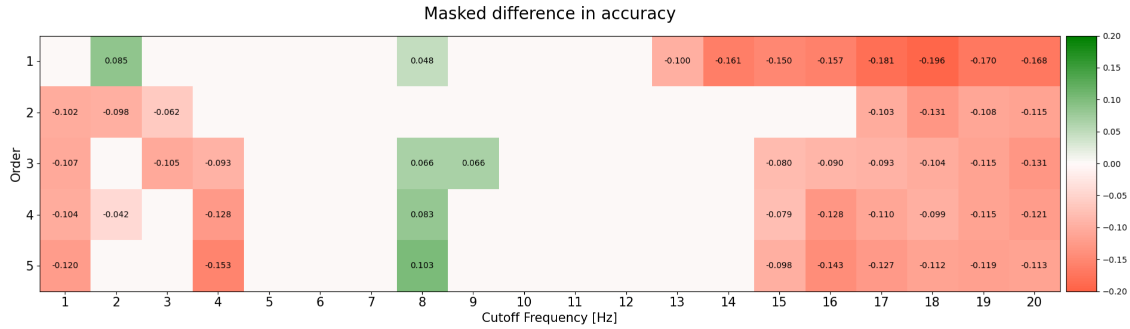


Figure D.2: The difference in accuracy scores between the random forests on smoothed and non-smoothed data but masked by the significant p-values of below our alpha of 0.05 using the sensor on the left upper leg.

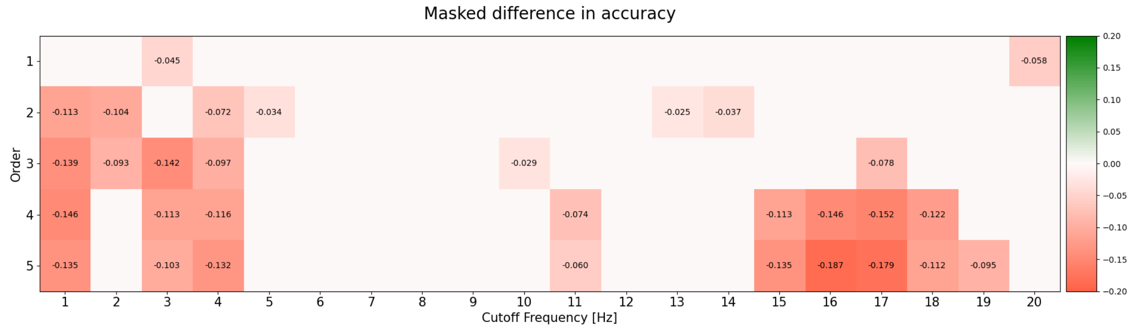


Figure D.3: The difference in accuracy scores between the random forests on smoothed and non-smoothed data but masked by the significant p-values of below our alpha of 0.05 using the sensor on the right upper leg.

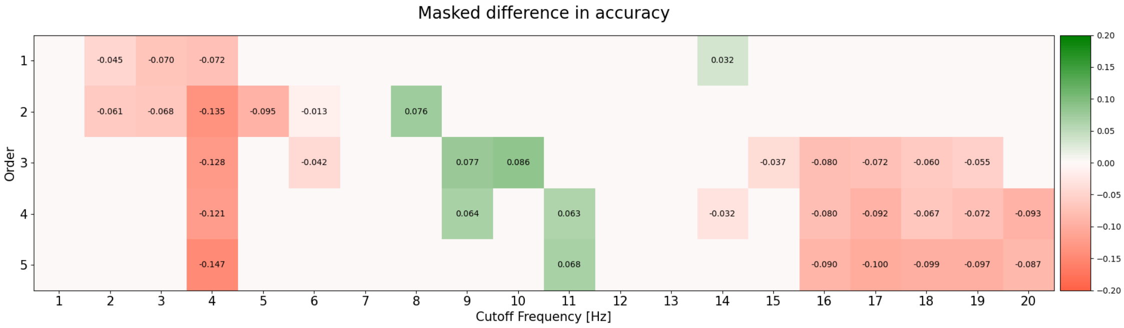


Figure D.4: The difference in accuracy scores between the random forests on smoothed and non-smoothed data but masked by the significant p-values of below our alpha of 0.05 using the sensor on the left lower leg.

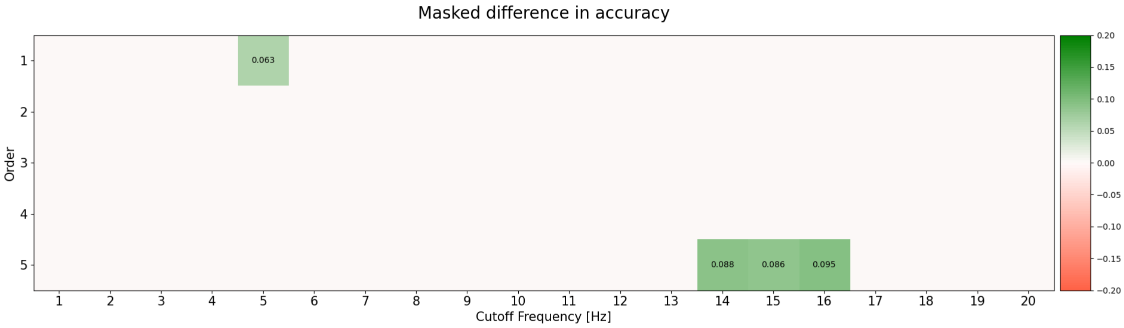


Figure D.5: The difference in accuracy scores between the random forests on smoothed and non-smoothed data but masked by the significant p-values of below our alpha of 0.05 using the sensor on the right lower leg.

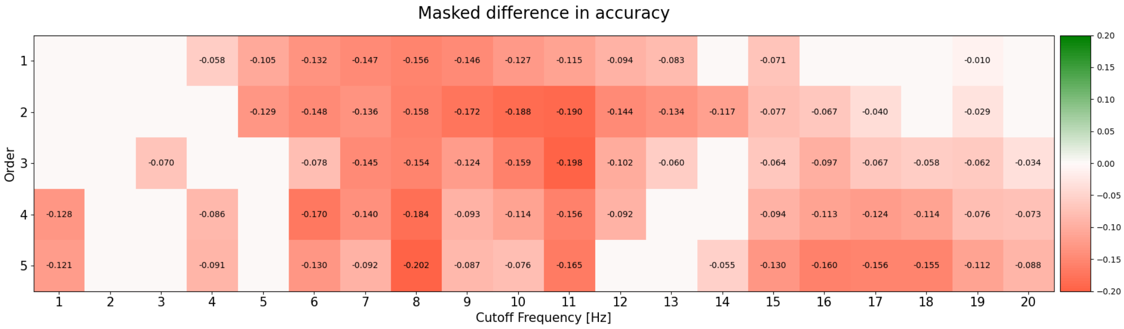


Figure D.6: The difference in accuracy scores between the random forests on smoothed and non-smoothed data but masked by the significant p-values of below our alpha of 0.05 using the sensor on the left foot.

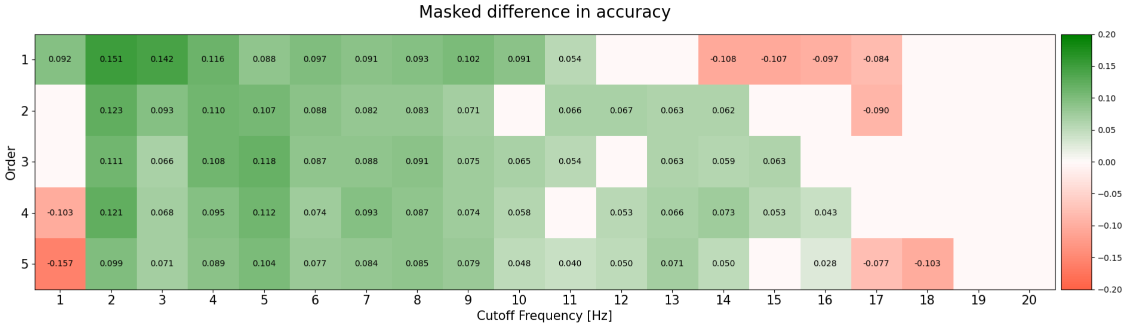


FIGURE D.7: The difference in accuracy scores between the random forests on smoothed and non-smoothed data but masked by the significant p-values of below our alpha of 0.05 using the sensor on the right foot.

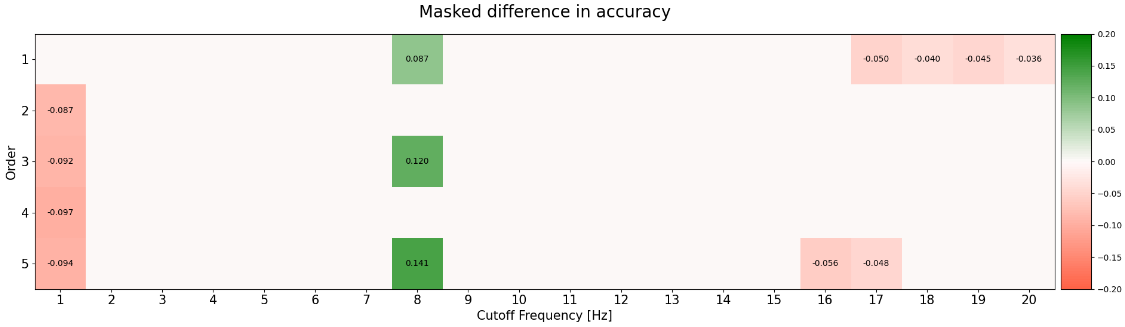


FIGURE D.8: The difference in accuracy scores between the random forests on smoothed and non-smoothed data but masked by the significant p-values of below our alpha of 0.05 using the sensors on both the upper legs.

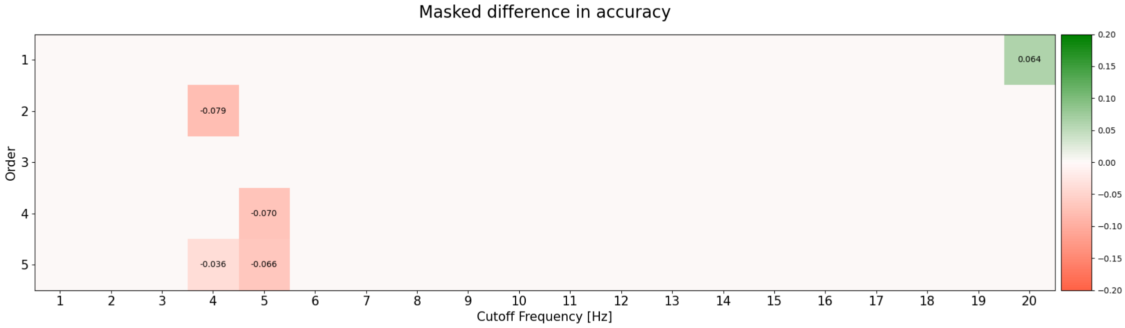


FIGURE D.9: The difference in accuracy scores between the random forests on smoothed and non-smoothed data but masked by the significant p-values of below our alpha of 0.05 using the sensors on both the lower legs.

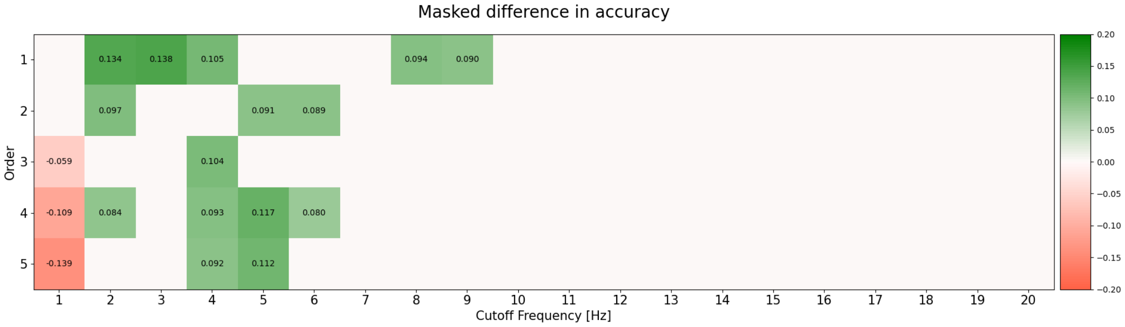


FIGURE D.10: The difference in accuracy scores between the random forests on smoothed and non-smoothed data but masked by the significant p-values of below our alpha of 0.05 using the sensors on both feet.

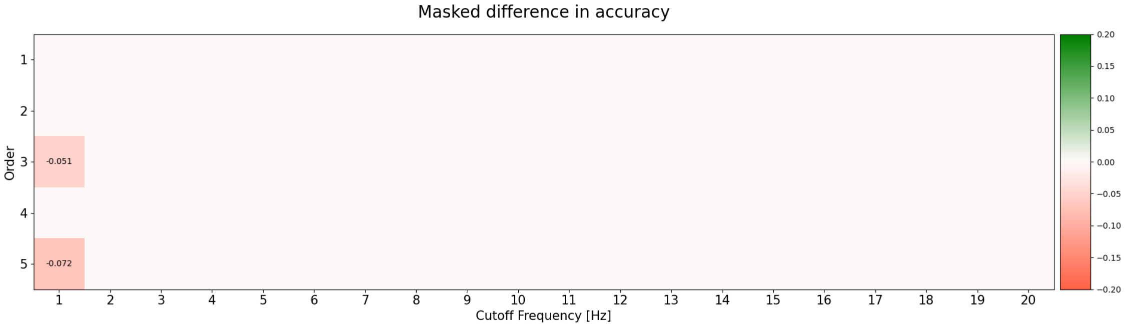


FIGURE D.11: The difference in accuracy scores between the random forests on smoothed and non-smoothed data but masked by the significant p-values of below our alpha of 0.05 using the 4 sensors on both upper and lower legs.

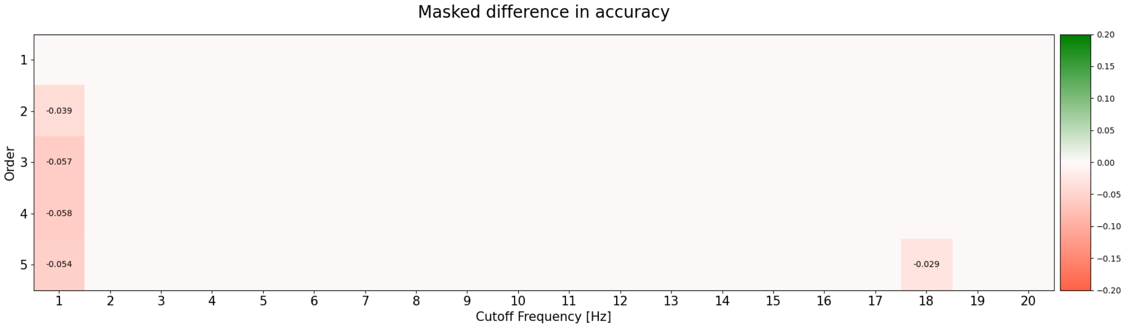


FIGURE D.12: The difference in accuracy scores between the random forests on smoothed and non-smoothed data but masked by the significant p-values of below our alpha of 0.05 using all sensors.