Exercise PC Design & Development / Systems and Environments

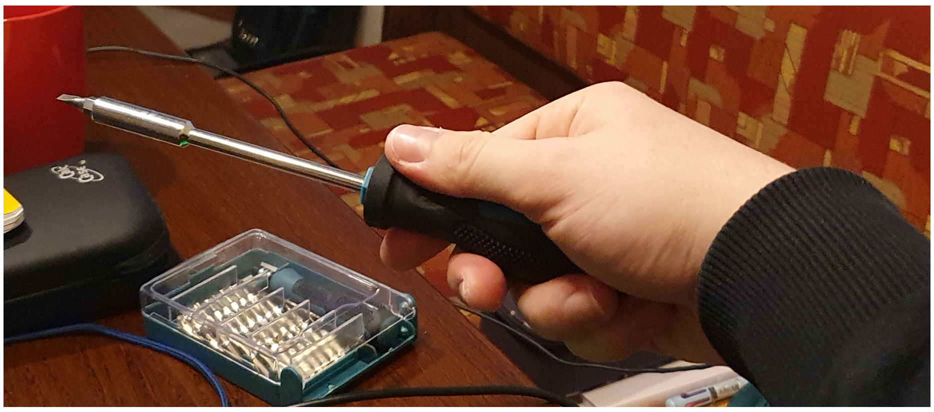
Assignment 3

11908757 Stefan Haslhofer

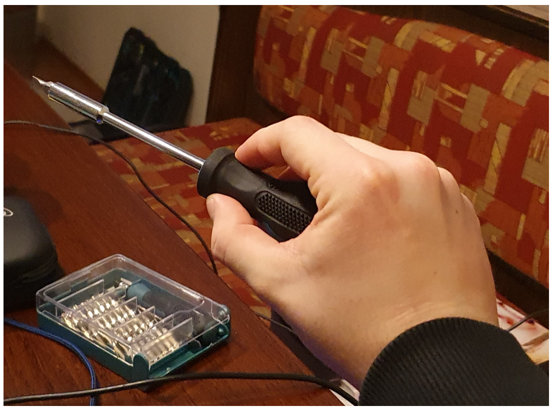
1. Remarks

Each data sample is denoted by an operation (screw/unscrew) and a grip (grip1/grip2). To eliminate possible confusion I appended pictures of each grip that I used:

- Grip 1:



- Grip 2:

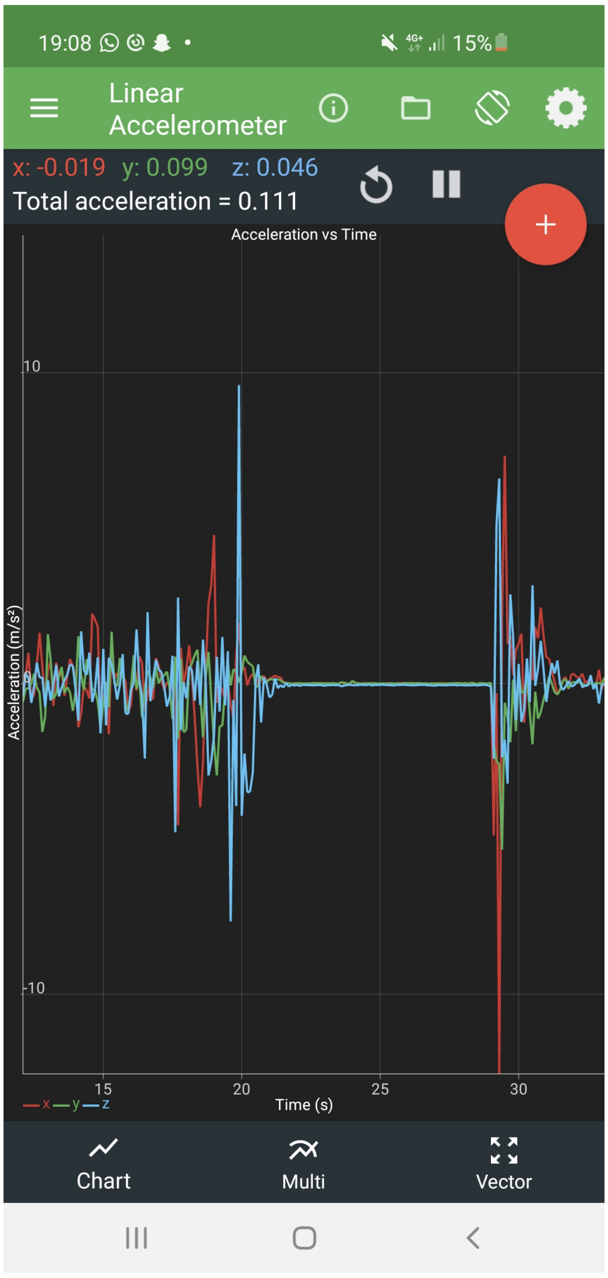


Python code: https://github.com/StefanHaslhofer/PervasiveComputing/tree/main/Assignment3

2. Data recording

To acquire data I taped my mobile phone to my wrist and screwed/unscrewed the same screw six times with each grip.

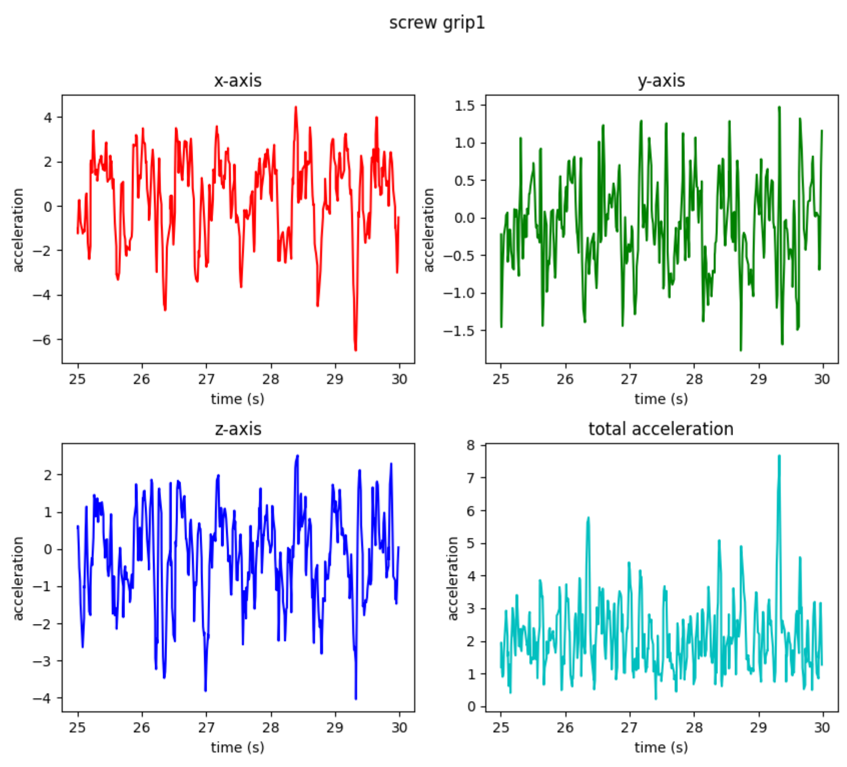
I recorded the acceleration of my wrist over time on my mobile phone with an app called *Physics Toolbox Sensor Suite*. The app allows access to the phone's built-in linear accelerometer and displays acceleration on x, y and z axis as well as a total acceleration value.

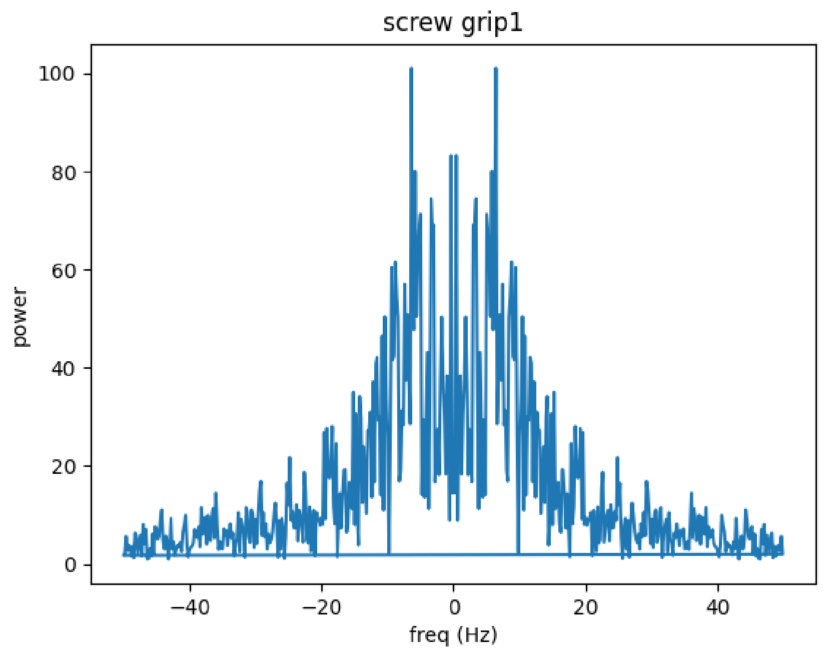


The data can be exported as *csv*. Each row contains a timestamp and all axis values plus the total acceleration.

3. Pre-processing and Segmentation

Similar to assignment 2, I plotted the recordings in the time domain and in the frequency domain for a better understanding of the data. In the previous assignment I concluded that all important frequencies in movements are quite low. Hence, I also only consider low frequencies up to 50Hz for this assignment.
Furthermore, I split the data recordings into windows of 5 seconds to get enough samples. I used a jumping window approach. Windows do not overlap.





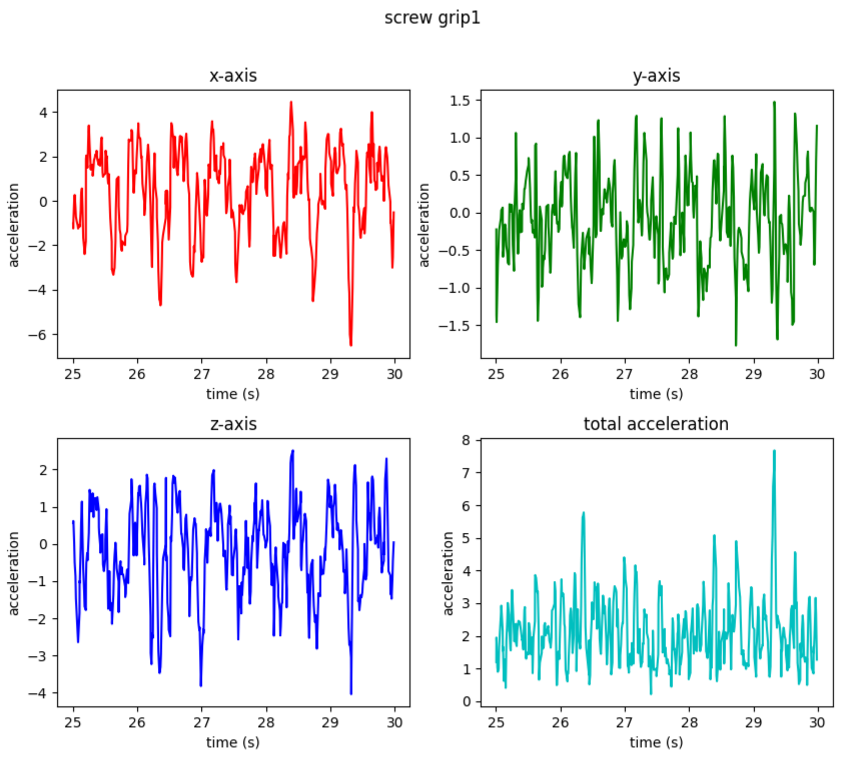
Above you can see two plots of one screw operation using grip 1:

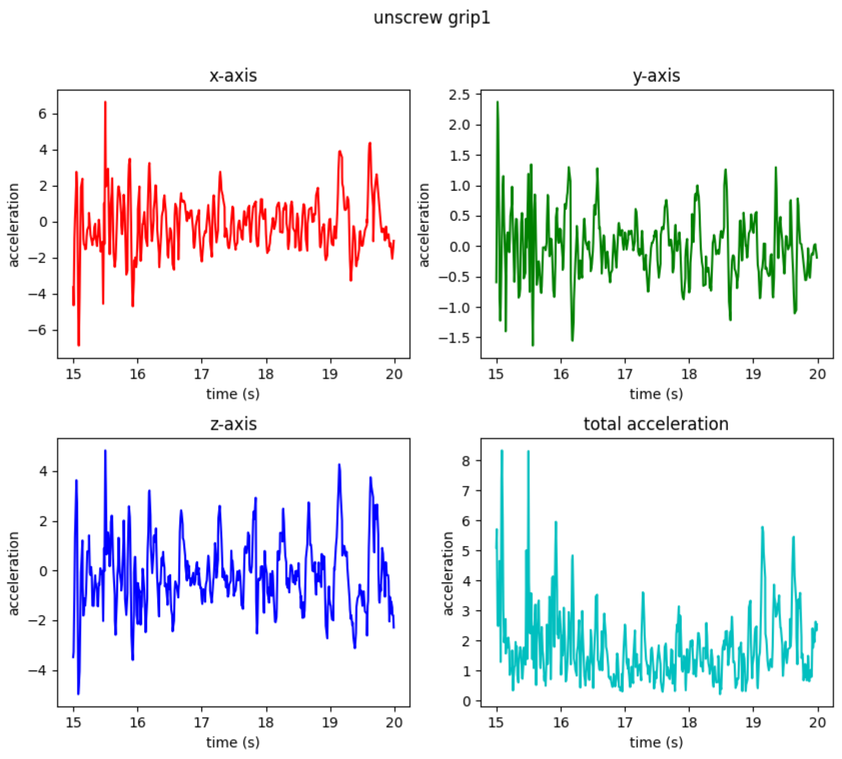
1. x, y and z-axis acceleration plus as well as total acceleration over time
2. frequency composition of total acceleration

4. Feature extraction

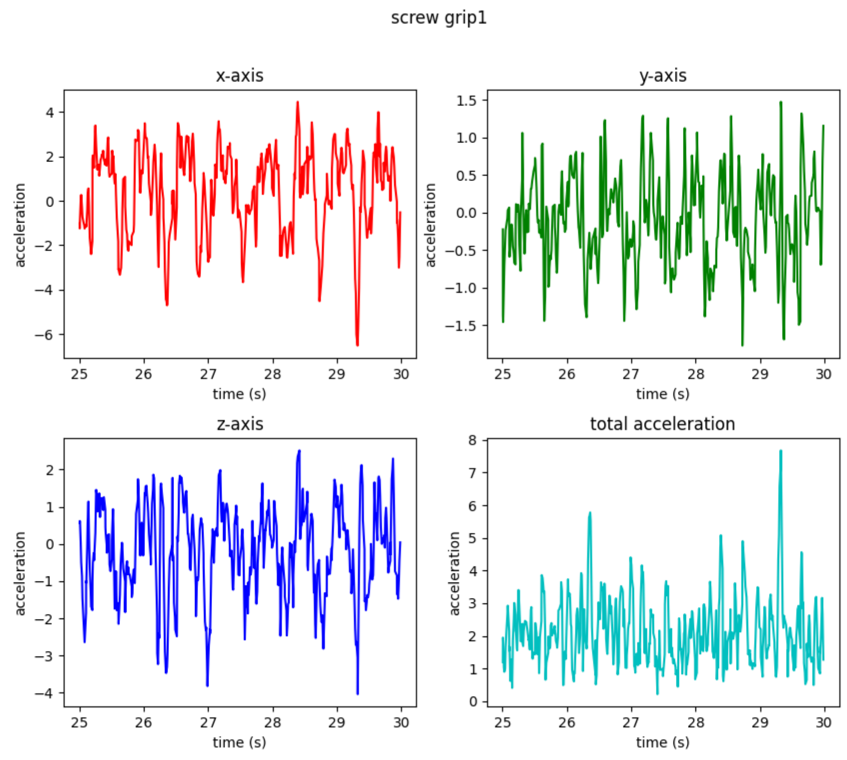
Next, I implemented a python script to extract the features and save it to an *arff*-file

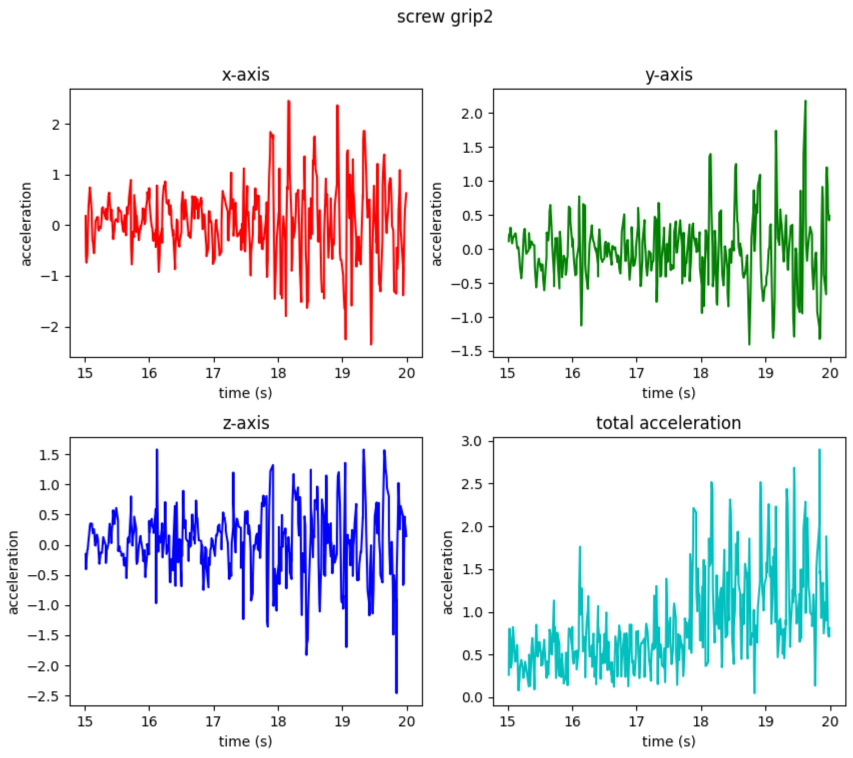
First, I calculated mean and variance for the acceleration of my wrist along all axes as well as for the total wrist acceleration. Looking at the result I noticed that the unscrew operation has a larger acceleration along the z-axis and the screw operation has more acceleration along the x-axis. However, there is no clear difference in the total acceleration of each operation (except some outliers) as can be seen in the two plots below:



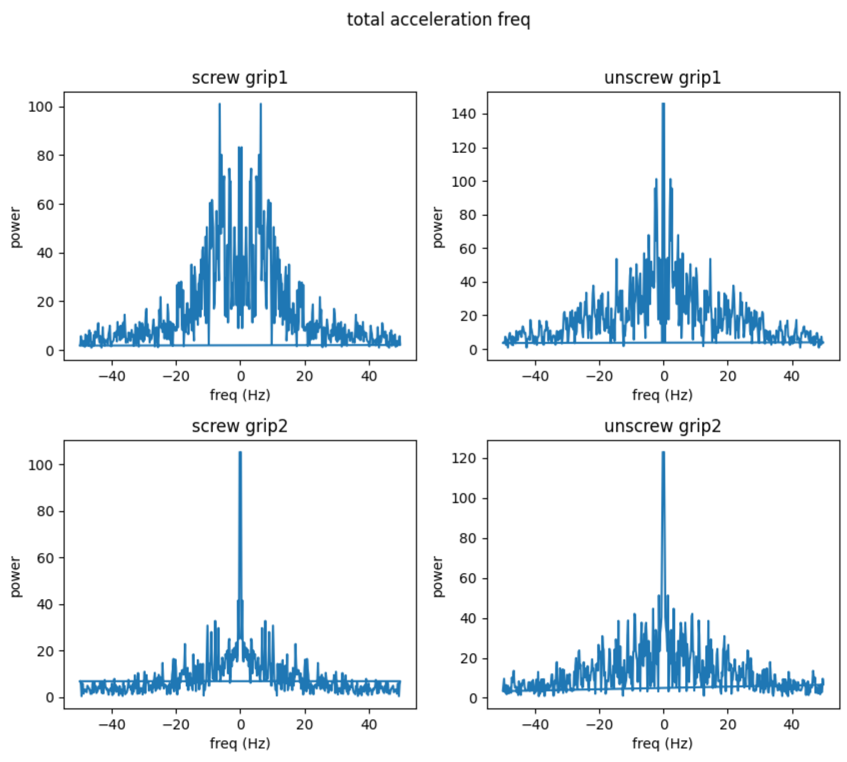


Regarding the grips, there is a clear difference in acceleration for all axes including the total acceleration, which is much higher for grip 1. Following plots visualize the acceleration of my wrist along all axes during the screwing operation for both grips:





Unfortunately, the frequency domain seems not very helpful in this assignment. When comparing the graphs of total acceleration signal frequencies for each operation combined with the grips we can hardly spot any difference except for the screw operation using grip 1.
At least the amplitude seems to be moderately higher for the unscrew operation compared to the screw operation:



The following table lists all extracted features with some remarks:

feature	remark
mean of x-axis	mean acceleration of x-axis seems to be higher for the screw operation and grip 1
mean of y-axis	mean acceleration of y-axis is slightly lower for grip 2
mean of z-axis	mean acceleration of z-axis seems to be higher for the unscrew operation and grip 1
mean of total Acc	mean acceleration seems to be similar for both screwing and unscrewing but higher for grip 1
var of x-axis	variance of wrist acceleration seems to be lower for grip 2
var of y-axis	(see var of x-axis)
var of z-axis	(see var of x-axis)
var of total Acc	(see var of x-axis)
max x-axis Acc	the maximum acceleration seems to be lower for grip 1 but nearly the same for the operation
max y-axis Acc	(see var of x-axis)
max z-axis Acc	(see var of x-axis)
max total Acc	(see var of x-axis)
max x freq energy	maximal amplitude seems to be higher for the unscrew operation
max y freq energy	(see max x freq energy)
max z freq energy	(see max x freq energy)
max total freq energy	(see max x freq energy)

feature	remark
sum of x-axis energy	overall amplitude of frequencies seem to be higher for the unscrew operation
sum of y-axis energy	(see sum of x-axis energy)
sum of z-axis energy	(see sum of x-axis energy)
sum of total energy	(see sum of x-axis energy)

5. Operation classification

Weka was used for classifications.

I figured out that the mean of the z-axis and the mean of the total acceleration had a negative impact on the result (in contradiction to my initial assumption stated in section 4). Therefore, I removed them.

a) J48

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0,830	0,073	0,917	0,830	0,871	0,762	0,866	0,823	screw
	0,927	0,170	0,850	0,927	0,887	0,762	0,866	0,852	unscrew
Weighted Avg.	0,880	0,122	0,883	0,880	0,879	0,762	0,866	0,838	

Parameter tuning:

- C (confidenceFactor): Changing the parameter *C* has hardly any effect on the result. I increased it up to 0.8 and lowered it to 0.01 to see at least some effects but accuracy only gets worse.
- M (minNumObj): Changing the parameter *M* only worsens the result. Similar to parameter *C*, only a significant change led to a change at all.

b) Naïve Bayes

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0,849	0,327	0,714	0,849	0,776	0,529	0,854	0,872	screw
	0,673	0,151	0,822	0,673	0,740	0,529	0,854	0,850	unscrew
Weighted Avg.	0,759	0,237	0,769	0,759	0,758	0,529	0,854	0,861	

Naive bayes has no parameters in *Weka*.

c) kNN

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0,962	0,036	0,962	0,962	0,962	0,926	0,983	0,971	screw
	0,964	0,038	0,964	0,964	0,964	0,926	0,983	0,981	unscrew

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
Weighted Avg.	0,963	0,037	0,963	0,963	0,963	0,926	0,983	0,976	

Parameter tuning:

- k: I increased the parameter *k* (number of nearest neighbors) until the accuracy declined, which was at 3. This allowed me to increase accuracy from 93.52% to 96.3%.
- W: Increasing *W* was not effective. A lower number decreased accuracy drastically, whereas choosing a large number (>=100) led to the initial result.

d) Multilayer perceptron

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0,943	0,036	0,962	0,943	0,952	0,907	0,996	0,996	screw
	0,964	0,057	0,946	0,964	0,955	0,907	0,996	0,996	unscrew
Weighted Avg.	0,954	0,047	0,954	0,954	0,954	0,907	0,996	0,996	

Parameter tuning:

- L (learningRate): A decrease to 0.1 results in an accuracy increase by 0.93%. If I would decrease the parameter even more or increase it the outcome would be negative.
- M (momentum): Setting the parameter to 0.25 improves the ROC- and PRC area slightly by 0.1%.
- N (trainingTime): Changing the parameter *N* has no effect on the result.
- V (validationSetSize): Accuracy falls with larger parameter *V*.
- S (seed): Changing the parameter *S* has no effect on the result.
- E (validationThreshold): Changing the parameter *E* has no effect on the result.

Summary

In summary, the kNN algorithm performed best with an accuracy of 96.3%. It also achieved the lowest FP rate at 3.7%. Nonetheless, the multilayer perceptron has the best ROC- and PRC area. Overall I regard the kNN as the best solution for this particular classification problem.

6. Grip classification

Weka was used for classifications.

I used the Weka visualization to filter out less significant features by hand, which in this classification task is the mean of the y-axis.

a) J48

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0,982	0,058	0,948	0,982	0,965	0,926	0,939	0,907	grip1
	0,942	0,018	0,980	0,942	0,961	0,926	0,939	0,928	grip2

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
Weighted Avg.	0,963	0,039	0,964	0,963	0,963	0,926	0,939	0,917	

Parameter tuning:

- C (confidenceFactor): Changing the parameter *C* only worsens the result.
- M (minNumObj): Decreasing the parameter *M* has no impact on classifier performance. However, increasing it to a value of 5 improved accuracy by 1.85%.

b) Naïve Bayes

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0,982	0,058	0,948	0,982	0,965	0,926	0,992	0,994	grip1
	0,942	0,018	0,980	0,942	0,961	0,926	0,992	0,992	grip2
Weighted Avg.	0,963	0,039	0,964	0,963	0,963	0,926	0,992	0,993	

Naive bayes has no parameters in *Weka*.

c) kNN

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	1,000	0,019	0,982	1,000	0,991	0,982	0,999	0,998	grip1
	0,981	0,000	1,000	0,981	0,990	0,982	0,999	0,998	grip2
Weighted Avg.	0,991	0,010	0,991	0,991	0,991	0,982	0,999	0,998	

Parameter tuning:

- k: I increased the parameter *k* until I reached an accuracy of 99.1%, which was at 2. I could not extract a better result with larger *k*.
- W: Changing *W* only deteriorated the performance.

d) Multilayer perceptron

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0,964	0,019	0,982	0,964	0,973	0,945	0,993	0,994	grip1
	0,981	0,036	0,962	0,981	0,971	0,945	0,993	0,993	grip2
Weighted Avg.	0,972	0,027	0,972	0,972	0,972	0,945	0,993	0,993	

Parameter tuning:

- L (learningRate): A decrease to 0.1 leads to an accuracy increase by 1%. Any increase or further decrease of parameter *L* has an unfavourable effect on the classifier performance.
- M (momentum): Changing the parameter *M* worsens the accuracy.
- N (trainingTime): Increasing the training time worsens the accuracy, decreasing it slightly has no effect.
- V (validationSetSize): Accuracy falls with larger parameter *V*.
- S (seed): After trying various seeds I only achieved to decrease accuracy.
- E (validationThreshold): Changing the parameter *E* has no effect on the result.

Summary

All classifiers perform very well guessing the used grip. However, kNN produces the best outcome with an accuracy of 99.1%. Furthermore, kNN also has the lowest false positive rate at 1% and the largest ROC- and PRC area (99.9% and 99.8%). The multilayer perceptron comes second with an accuracy of 97.2%. Note that j48 and naive bayes have identical accuracies (96.3%) and false positive rates (3.9%).