JOHANNES KEPLER
UNIVERSITY LINZ

Submitted by

Arnold Ackerlauer

Submitted at

Institute of

Signal Processing

Supervisor

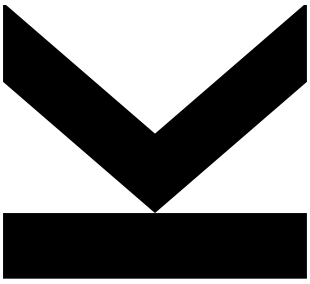
Univ.-Prof. Dr. Mario Huemer

Co-Supervisors

DI Carl Böck

March 2022

Feature Engineering for Spike Sorting



Bachelor Thesis

to obtain the academic degree of

Bachelor of Science

in the Bachelors's Program

Bioinformatics

JOHANNES KEPLER

UNIVERSITY LINZ

Altenbergerstraße 69

4040 Linz, Österreich

www.jku.at

DVR 0093696

Bibliographical Detail

Ackerlauer, A., 2022: Feature Engineering for Spike Sorting. Bachelor Thesis, in English. – 43 p., Institute for Signal Processing University, Linz, Austria

Annotation

Spike sorting is the process of identifying the neuron from which a spike originated from and can be used in various applications such as brain-machine interfaces. This thesis is based on simulated, continuous recordings of single neuronal cells. Using several feature engineering techniques we were able to propose a feature set which can be used for spike sorting.

Declaration

I hereby declare that I have worked on my bachelor's thesis independently and used only the sources listed in the bibliography.

I hereby declare that, in accordance with Article 47b of Act No. 111/1998 in the valid wording, I agree with the publication of my bachelor thesis, in full to be kept in the Faculty of Science archive, in electronic form in a publicly accessible part of the IS STAG database operated by the University of South Bohemia in České Budějovice accessible through its web pages. Further, I agree to the electronic publication of the comments of my supervisor and thesis opponents and the record of the proceedings and results of the thesis defence in accordance with aforementioned Act No. 111/1998. I also agree to the comparison of the text of my thesis with the Theses.cz thesis database operated by the National Registry of University Theses and a plagiarism detection system.

Linz, 03.01.2023

Place, Date

Arnold Ackerlauer

Contents

Abstract

Spike sorting is the clustering of different spikes according to which neuron they originated from. One of the most important steps in spike sorting is feature engineering. Within this thesis we focus on applying feature engineering techniques on simulated, continuous recordings of single neuronal cells in order to propose a feature set, which is a good basis for predicting the neurons of each spike. The steps to get to a proposed feature set included extracting single action potentials from the continuous dataset, extracting features based on the identified action potentials and applying feature selection methods to identify the most promising features. The applied feature selection methods included the investigation of the distribution of feature values between different cluster both visually and using ANOVA. Furthermore, random forest intrinsic feature importance and random forest permutation feature importance were used for determining the best features. Considering the outcomes of these feature selection methods, we found that an ideal feature set consists of the geometric feature positive & negative amplitude as well as positive & negative action potential energy and the first four principal components. The different feature selection methods were not conclusive for the feature spike width. Therefore, we propose to test a feature set with and without that feature when running machine learning-based prediction tasks. The proposed feature set can be used as a basis to do spike sorting in applications such as invasive brain computer interfaces, which is an emerging topic in the past few years . Such applications have the potential to support people with various disabilities in their daily life.

1 Introduction

Applications of neuronal spike sorting have the potential to change the world as we know it. Spike sorting helps researchers in understanding the processes of the human brain and fosters the discovery of treatments for neurological diseases such as Alzheimer's disease or epilepsy [6]. Neuronal spike sorting is also an essential component in brain-machine interfaces [6]. Brain-machine interfaces open the possibility for communication between the brain and the external world, without having to rely on standard communication ways such as muscles or nerves ([16] in [8]). They hold the possibility to support people with various disabilities by providing speech synthesizers [1] and robotic limbs [12] [9].

Spike sorting is essential to gain knowledge of extracellular recordings needed for the previously described applications. Extracellular recordings are recorded by placing electrodes in brain tissue between neurons in order to monitor the extracellular activity of neurons. [15]. The first step of spike sorting is identifying the actual action potentials, as electrodes also pick up other electrical activities apart from the desired neuronal action potentials. These other electrical activities can include physiological activities such as muscular activity or external activities such as cellphone signals [4]. The next step after the action potentials are identified is feature engineering. This can be used to extract relevant information from each action potential. Using these features, machine learning can then be applied to cluster spikes together originating from the same neurons. This is possible, as each neuron produces a certain spike shape different from other neurons [15].

The aim of this thesis is to focus on the step of feature extraction and selection and to give a proposal for a feature set, which is a good basis for predicting the neurons of each spike. The reason for focusing on this part of spike sorting is that it is one of the two major parts in typical approaches of spike sorting, besides the classification of the action potentials [2]. This also allows to give this thesis a good focus.

Our research is based on simulated continuous recordings of single neuronal cells based on real action potentials recorded by Faraut et al. [5]. From this dataset we identified action potentials, as this is a necessary data pre-processing step to perform feature engineering. Features which were extracted based on the dataset include geometric features based on the shape

of the spike, PCA for dimensionality reduction of the raw spike data and NEO coefficients which give an estimate of the energy content at specific points in the action potential. The extracted features were then evaluated using feature selection methods. Finally, we come up with a proposal for a feature set which can then be used for predicting the neurons for each spike.

2 Related Work

2.1 Feature Engineering

Feature engineering is an important part in most machine learning tasks. In general, feature engineering involves finding relations between two or more variables and transforming them into a new variable, a feature. It can also involve transformations of data, such as for example taking the root of the variable. Furthermore, feature engineering can also contain taking averages, or applying other transformations such as fast Fourier transform or wavelet transforms. It is known that feature engineering is a task very difficult to automate, as it requires exploratory data analysis as well as domain knowledge. [10] Therefore, for the application of spike sorting, it is vital to know more about the origin of neuronal spikes.

2.2 Neuronal Spikes

Neuronal spikes originate from neurons, which are, simply put, special cells in the human brain. A neuron consists of a cell body, which is very similar to any other cell. What is unique to neurons is the additional structure around the cell body. The cell body is surrounded by dendrites, which look like wires branching out of the cell body. Furthermore, the axon also originates out of the cell body. It can be described as the tentacle of the neuron which can be used to connect to other neurons. The point where an axon meets with a dendrite of another neuron is called synapse. This is where the neuron can send out action potentials to another neuron in the form of electric energy built up using ion channels at the synapse. [3]

These action potentials can now be measured using electrodes being placed in the vicinity of these neurons. The shape of the measured action potential depends on the shape and structure of the dendrites of the spike. It also depends on how the ion channels are distributed at the synapse. Finally, the direction and distance of the electrode to the neuron also determine the spike shape. ([7] in [15])

2.3 Applications

Making use of domain knowledge regarding neuronal spikes, feature engineering and the other steps involved in spike sorting, new and exciting applications can be developed. One

which gained quite a bit of attention in the media lately is the brain-machine interface developed by Neuralink. This startup founded by Elon Musk aims at creating a brain-machine interface which enables people with neurological disorders to regain motor and sensory functions. They developed a device which contains 3072 electrodes for recording neuronal action potentials. In addition, Neuralink uses a precise robot to place the electrodes of the device in a brain. With this approach they are able to record neural activity spread across different areas of the brain and monitor the brain activity in the form of spikes in real-time. [6] [12] This is a technology trend not limited to Elon Musk's company, but is popular across several different companies. Some have even already achieved first test trials of similar technology in humans. [14]

3 Experimental Setup

3.1 Data Set

The basis for this experiment are simulated, continuous recordings of single neuronal cells. The continuous dataset was created based on actual recordings of action potentials by Faraut et al. [5]. They were then randomly distributed after each other in random time intervals and noise was added. To be able to experiment with different noise levels, we simulated added noise on four different levels, from level 1 with rather little noise to level 4 with much noise. Figure 1 shows a small subset of the data plotted over time. Apart from the continuous level of noise there are several spikes visible.

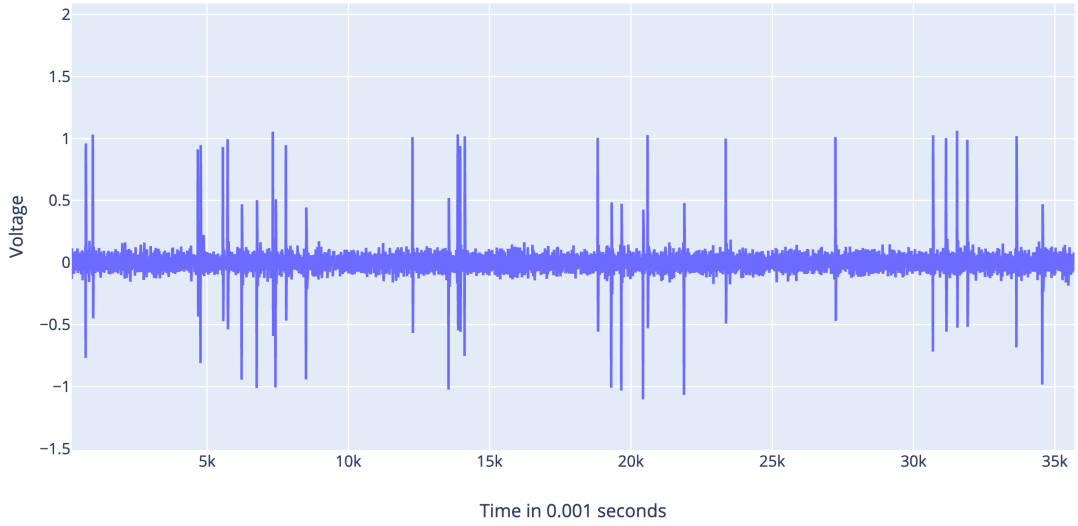


Figure 1: The figure shows a sample of the dataset used for this experiment plotted over time.

3.2 Methods

3.2.1 Data Pre-Processing

The very first step of spike sorting is actually extracting action potentials from the continuous data set. This is vital, as we cannot simply cluster single samples of the data set, we rather want to cluster the whole action potential which consists of several data points.

Identifying Action Potential Peaks In order to extract action potentials, the first step is to identify peaks of action potentials in order then to be able to use a specified range around these peaks, each of which are then defined as action potentials. A primitive approach would be to take all peaks found in the data set, i.e. all samples s at position p (s_p) where $s_{p-1} < s_p$ and $s_{p+1} < s_p$. However, this is obviously not a very precise method, as high-amplitude noise will also be identified as peaks.

Therefore, we used a different approach. As proposed in Quiroga et al. [13] we introduced a threshold which a peak has to at least surpass before being identified as an action potential peak. The formula for this threshold (Thr) as proposed in Quiroga et al. [13] is:

$$\sigma_n = \text{median}\{\frac{\mid x \mid}{0.6745}\} \tag{1}$$

$$\text{Thr} = \text{z}\sigma_n \tag{2}$$

where x are the action potentials and σ_n is an estimation of the standard deviation of the noise [13]. It would also be possible to directly take the standard deviation of the data σ, however, as the dataset contains spikes which are very different to the rest of the data, the standard deviation is not very robust. However, using the median as in the proposed method is more robust to outliers. The variable z is a multiplier of σ_n which then makes up the threshold. As proposed by Quiroga et al. [13] this multiplier could be set to 4.

However, to optimize this parameter according to the noise level found in our data set, we fine tuned this multiplier. In order to achieve this, we tried out different values in the range from 0 to 6, more precisely all the following values:

$$\sum_{z=0}^{60} \frac{z}{10} = \{0, 0.1, 0.2, 0.3, ..., 5.8, 5.9, 6\} \tag{3}$$

Next, we calculated the amount of spikes we would identify using each of the multipliers to set the threshold. Furthermore, we repeated this process for different noise levels 1-4, where ”Noise level 1” has relatively the smallest noise level and ”Noise level 4” relatively the highest noise level. These noise levels were artificially added to the data. Figure 2 shows the amount of spikes being identified for each of the different noise levels and for different thresholds using

the aforementioned values for the multiplier z.

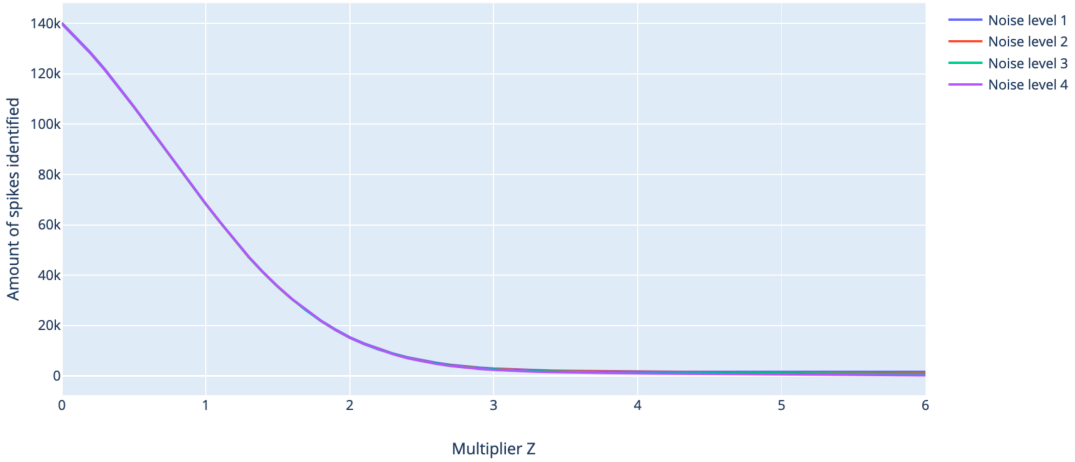


Figure 2: The figure shows the amount of identified spikes at different noise levels for different thresholds each being calculated by using differing multipliers z. As the values for the different noise levels are similar, it becomes hard to distinguish the different noise levels.

As one would expect, there is a clear trend in less peaks being identified as the threshold value increases. More notable is how fast the amount of peaks are decreasing: at the beginning the amount of spikes declines quite rapidly, but this decline levels out approximately at a $z = 3$. From this point on the decline seems to be more stable, however, with the scale of Figure 2 there is not much more we can interpret from it. Therefore, we created another Figure (Figure 3) which shows only a more relevant subset of different z values.

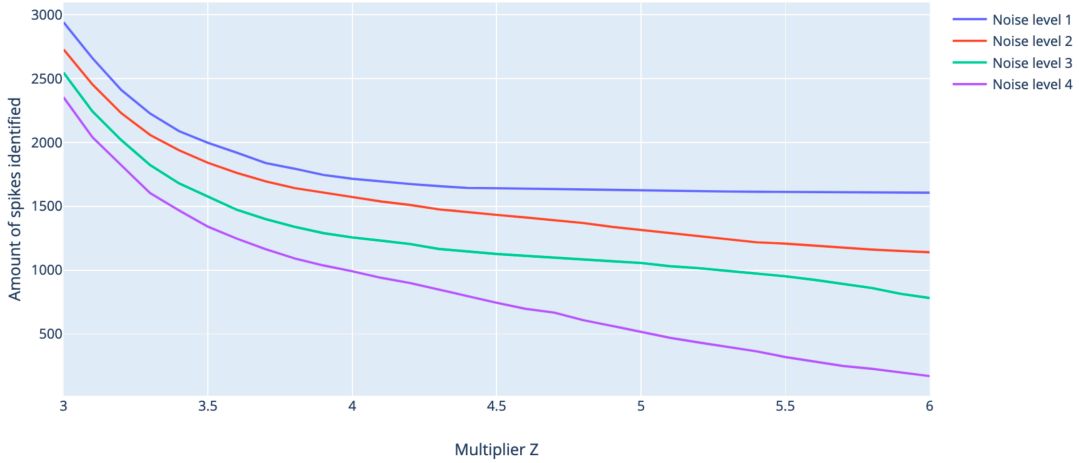
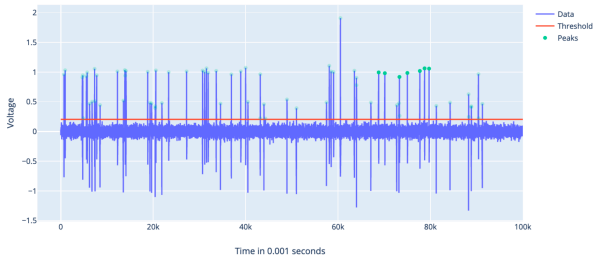


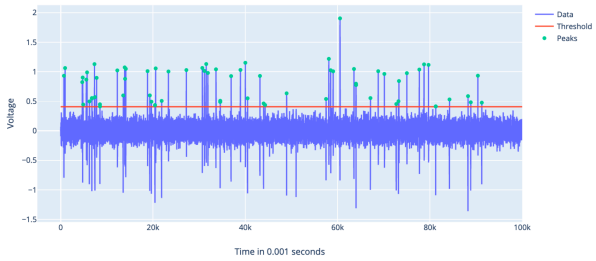
Figure 3: The figure shows the amount of identified spikes at different noise levels for different thresholds each being calculated by using differing multipliers z within a refined range after analysing Figure 2.

Figure 3 gives us a new perspective on the data we already saw in Figure 2. Now, there is a difference observable between the different noise levels and multipliers. There is still a noticeable decrease in amount of spikes identified between $z = 3$ and $z = 4$. However, for Noise level 1 the amount of spikes remained stable for all tested z values after $z = 4$. Yet, for other noise levels the amount of identified spikes still continued to drop even after $z = 4$, but there is a logical explantion for this observation: the higher the noise, the higher the standard deviation of the noise we calculate, the higher the threshold after being multiplied by z and also the fewer spikes being identified.

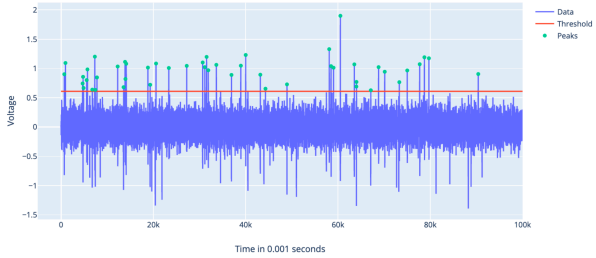
When visually checking the actual data, we noticed that for higher noise levels thresholds with z values greater than 4 seem to cut off peaks which actually are action potentials. Figure 4 shows the thresholds and peaks of $z = 4$, and to us this seems to produce reasonable results. Therefore, in the end we decided to use a z value of 4, as it was actually already recommended by [13]. Nevertheless, we believe this was a topic worth investigating as the threshold could differ for different data and a wrong value could potentially make us miss many action potentials or classify noise as action potentials. This could then drastically influence the quality of features and also a possible classification of the action potentials later on: If we were to take the peaks of noise we would classify something as some action potential even if it is not, if we



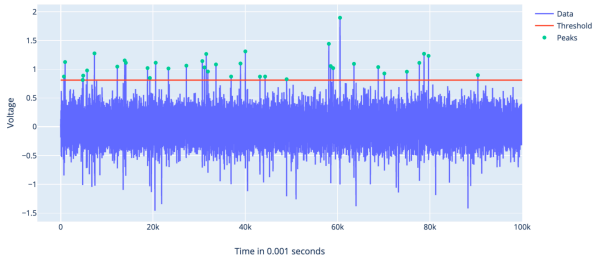
(a) Noise level 1



(b) Noise level 2



(c) Noise level 3



(d) Noise level 4

Figure 4: The figure shows a sample of data with differently simulated noise levels. For each noise level, the threshold was calculated and the peaks were identified according to the calculated threshold.

were to take too few action potentials, we would simply reduce our training data, and perhaps even miss clusters with lower peaks completely. This would significantly reduce the accuracy of any clustering efforts.

Action Potential Extraction After the peaks are identified, we still need to extract the actual action potentials. The basic principle for this is rather simple, as it is just taking a window of some amount of samples before and after the peak. The size of this window directly depends on the sampling rate of the data. This is because the time duration of action potentials remains the same no matter in which sampling rate they are recorded, but the amount of samples in this same time frame changes with the sampling rate. In order to get the best window size, we checked the action potentials visually to identify the ideal size.

Yet, there is still a special case to consider when extracting the action potentials. It is possible that one action potential actually has two peaks which are above the threshold. In this case we only want to extract this action potential once, rather than twice as if we were to do this naively. This can be generalized for a action potential to have n peaks, where we only want to use the highest peak as action potential peak and apply the window considering this peak.

This issue was solved by recursively removing the smallest peak until only one remains per window. Figure 5 shows some action potentials extracted from the data, centered around their peak.

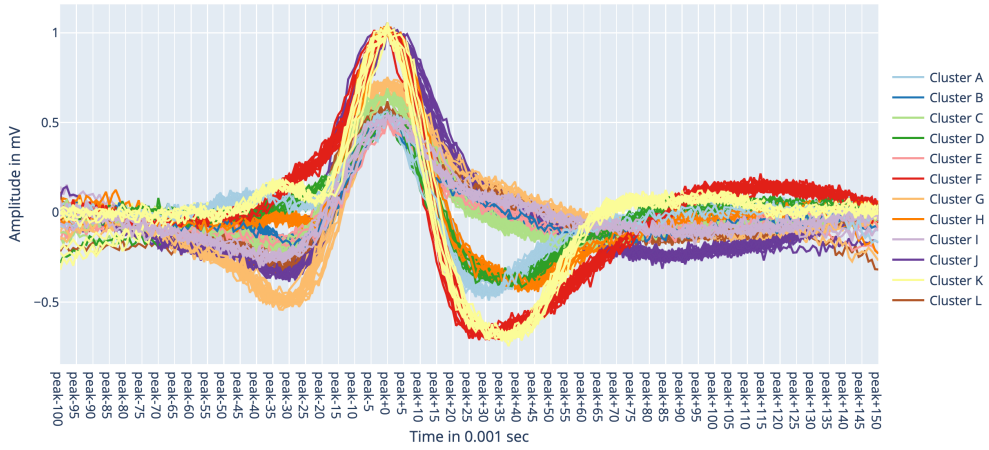


Figure 5: The figure shows a random sub-sample of identified action potentials, centered around the peak and with the respective areas left and right of the peak. Colors are assigned according to the cluster the action potential originates from.

3.2.2 Feature Extraction

The next task in spike sorting is extracting features. This is necessary, as directly using the extracted action potential data as input to a machine learning model can be to memory intensive, as there can be many data points for just one action potential. Also, feature extraction can be used to add new information to the existing data using domain knowledge.

Geometric Features The first subset of features we extracted are geometric features. Geometric features focus solely on the geometric shape of the action potential waveforms and are usually computationally inexpensive as they typically have a deterministic way of calculating them.

Positive Amplitude The feature positive amplitude is the highest voltage found in the action potential waveform, meaning the voltage at the peak of the action potential. The formal definition of the feature f_{PA} for the action potential X is defined in equation 4. Figure 6 displays the calculation visually. As we have previously extracted the action potential with a defined window centered around its peak, the peak is always at the same time interval after the start of the action potential.

$$f_{PA} = \max(X) \tag{4}$$

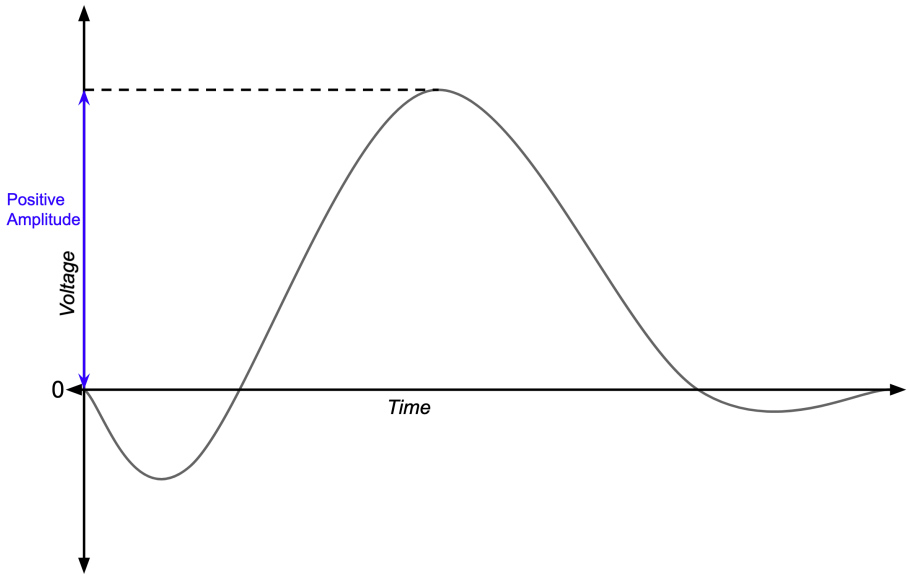


Figure 6: The figure shows a visual explanation of the positive amplitude feature.

Negative Amplitude The feature negative amplitude f_{NA} is the lowest voltage value found in the action potential X. It is comparable to the positive amplitude as described in section 3.2.2. However, the negative amplitude is not always found at the same time interval as it is the case with the positive amplitude. This feature was calculated using equation 5, figure 7 displays this as well.

$$f_{NA} = \min(X) \tag{5}$$

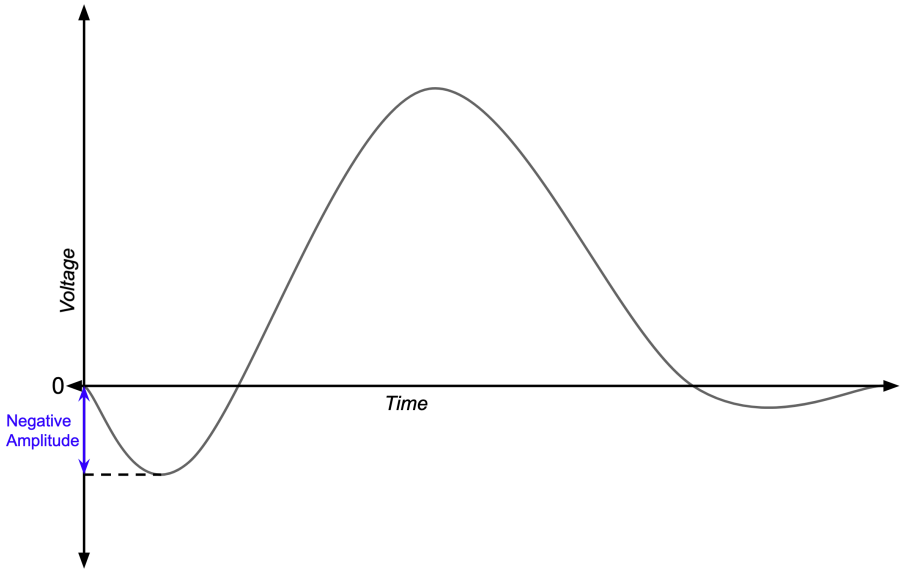


Figure 7: The figure shows a visual explanation of the negative amplitude feature.

Positive Action Potential Energy The feature positive action potential energy f_{PSE} is the action potential energy of only the positive part of the action potential. As the data we are working with is discrete, equation 6 can be used to calculated the feature. Figure 8 displays the calculation in a more visual way.

$$f_{PSE} = \sum_{i=1}^{n} \begin{cases} x_i^2, & \text{if } x \geq 0, \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

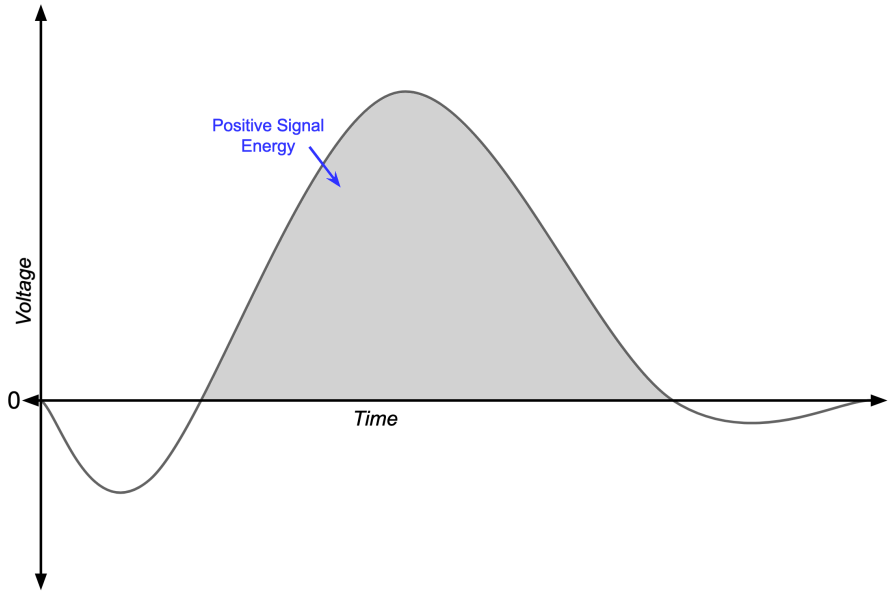


Figure 8: The figure shows a visual explanation of the positive action potential energy. The positive action potential energy is the area shaded grey in this visualization.

Negative Action Potential Energy The feature negative action potential energy f_{NSE} is the energy of the negative parts of the action potential (contrary as described in section 3.2.2). Again, this can be calculated with the sum of the squares, as also defined in equation 7. Figure 9 shows this feature. Note, that the positive action potential energy f_{PSE} and the negative action potential energy f_{NSE} sum up to be the total action potential energy of a action potential.

$$f_{NSE} = \sum_{i=1}^{n} \begin{cases} x_i^2, & \text{if } x \leq 0, \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

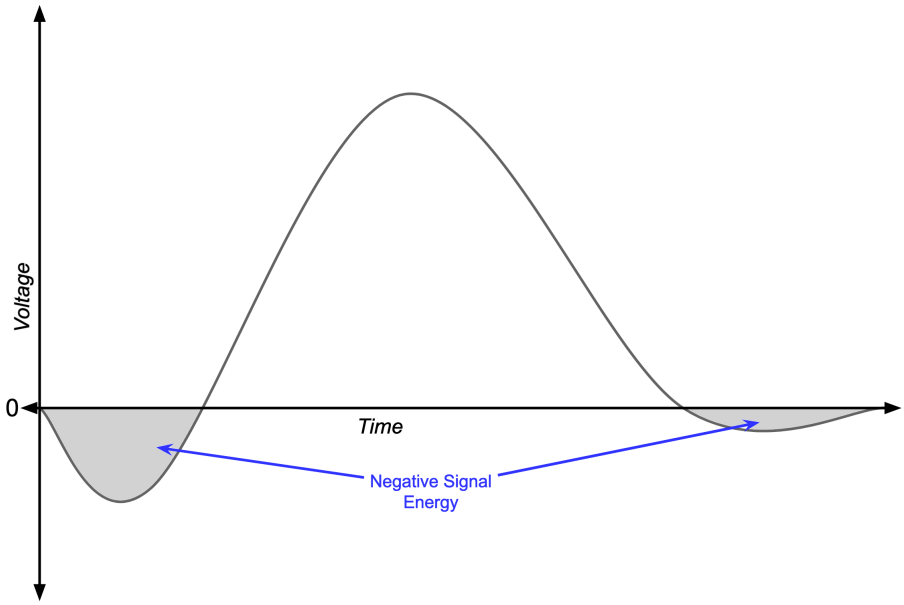


Figure 9: The figure shows a visual explanation of the negative action potential energy. The negative action potential energy is the area shaded grey in this visualization.

Left Spike Angle The feature left spike angle f_{LSA} is calculated in a similar way as described in [2]. First, the point left of the spike with a value closest to 50 percent of the highest point p_{l50} was identified. For the case that there are several such points, the first point to the left of the spike peak was taken. Next, we approximated the tangent for this point. For this, one point further to the left p_{l50-1} and one point further to the right p_{l50+1} was taken. Equation 8 is used calculate the gradient m and equation 9 is used to calculate the y-intercept k. They are then combined using equation 10 to give the tangent line y of the point p_{l50}.

$$m = \frac{p_{l50+1_y} - p_{l50-1_y}}{p_{l50+1_x} - p_{l50-1_x}} \tag{8}$$

$$k = -m * p_{l50-1_x} + p_{l50-1_y} \tag{9}$$

$$y = m * x + k \tag{10}$$

Using the gradient m we are now able to calculate the left spike angle f_{LSA} (which is also

the gradient angle of y). This calculation is done using equation 11 and a visual explanation of the whole calculation can be found in figure 10.

$$f_{LSA} = \begin{cases} 0, & \text{if } m = 0, \\ \arctan(m), & \text{if } m > 0, \\ \arctan(m) + \pi, & m < 0. \end{cases} \tag{11}$$

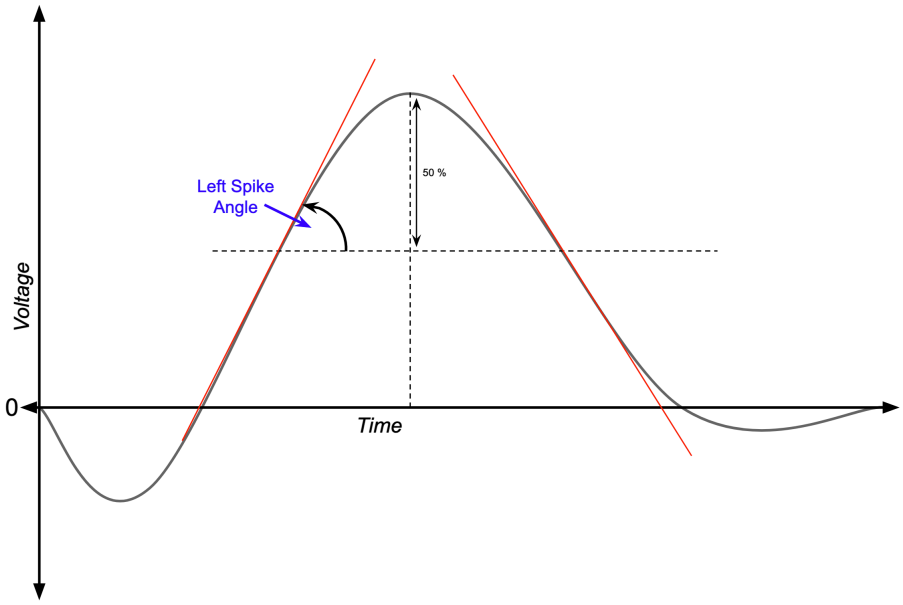


Figure 10: The figure shows a visual explanation of the left spike angle.

Right Spike Angle The feature right spike angle f_{RSA} is calculated in a very similar way as the left spike angle from section 3.2.2. However, for the right spike angle, instead of point p_{l50} we use p_{r50} which is again the point closest to the 50 percent but right of the action potential peak. The further calculations are then the same as already described in the section 3.2.2. Figure 11 is a diagram depicting the whole calculation of f_{RSA}.

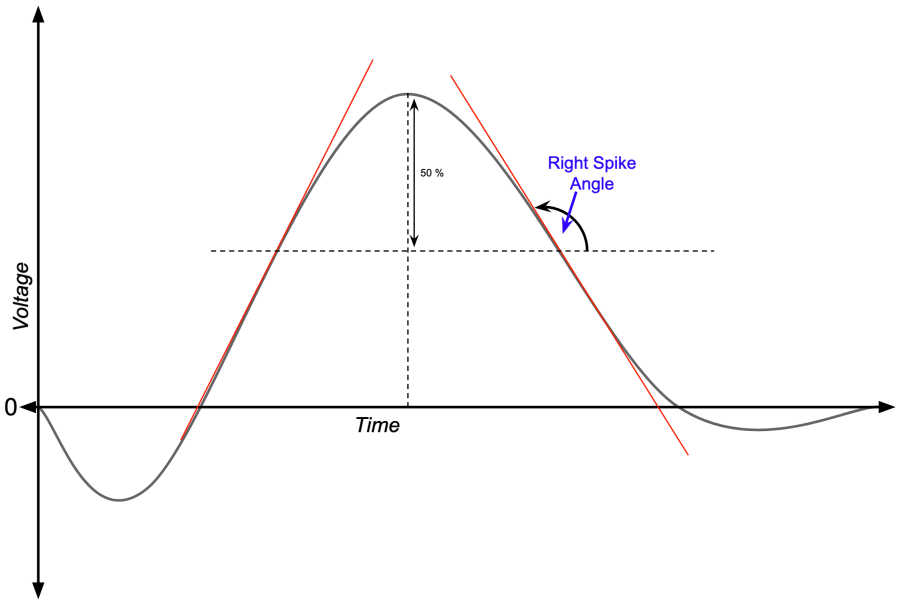


Figure 11: The figure shows a visual explanation of the right spike angle.

Spike Width The spike width f_{SW}. is approximated using the gradient lines of the points p_{l50} and p_{r50}, y_l and y_r respectively. For these two lines the time of the x-intercept is calculated using equation 12.

$$\text{x-intercept} = \frac{k}{m} \tag{12}$$

The difference between the two x-intercepts per each gradient line gives the spike width as defined in equation 13 and visually explained in figure 12.

$$f_{SW} = \text{x-intercept}_r - \text{x-intercept}_l \tag{13}$$

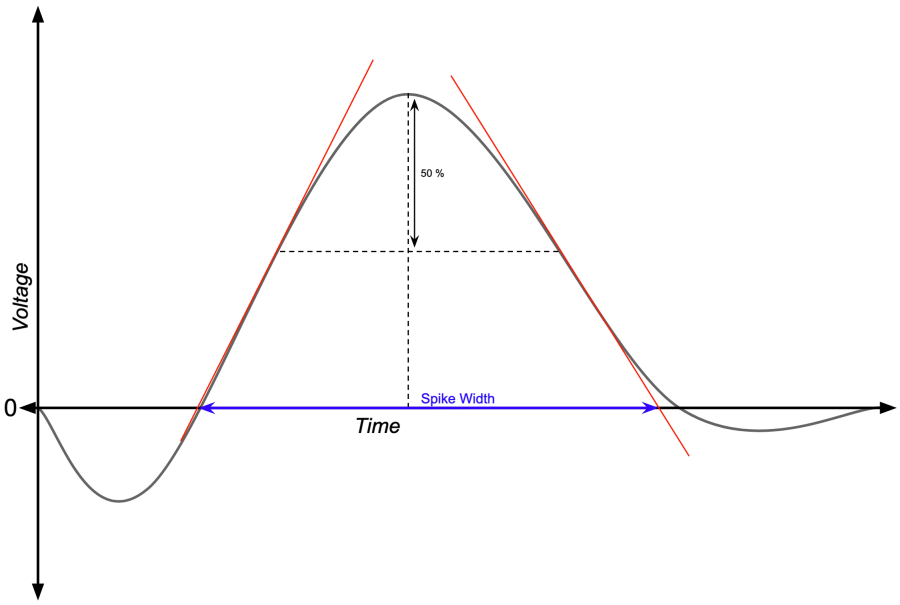


Figure 12: The figure shows a visual explanation of the spike width.

Principal Components Next to the geometric feature, principal component analysis is used for dimensionality reduction and using the extracted principal components as features. PCA is performed for all discrete voltage values of a spike. In order to decide which principal components should be used as features, we looked into the amount of cumulative explained variance. As can be seen in figure 13, there is already a high level of cumulative explained variance reached with less than 10 principal components.

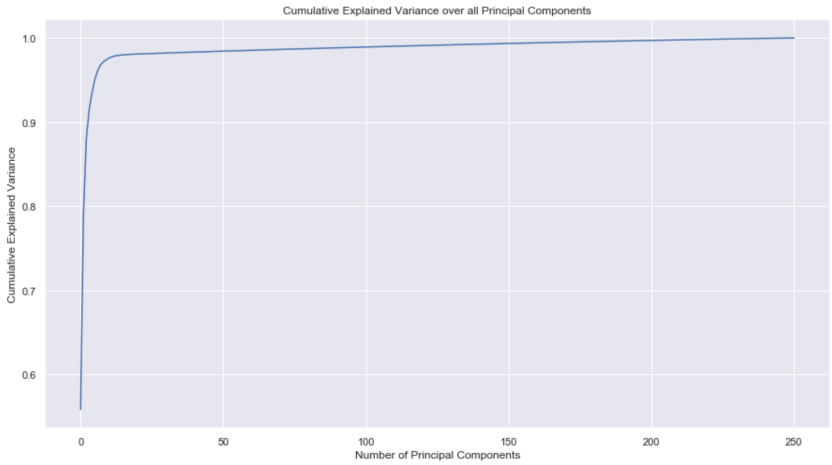


Figure 13: This figure shows the cumulative explained variance over all principal components.

Therefore, we decided to magnify this section of the graph, as can be seen in figure 14. Using this figure, we decided to use 4 principal components, as already a high level of cumulative explained variance is reached (0.91), and the increase per further principal component diminishes.

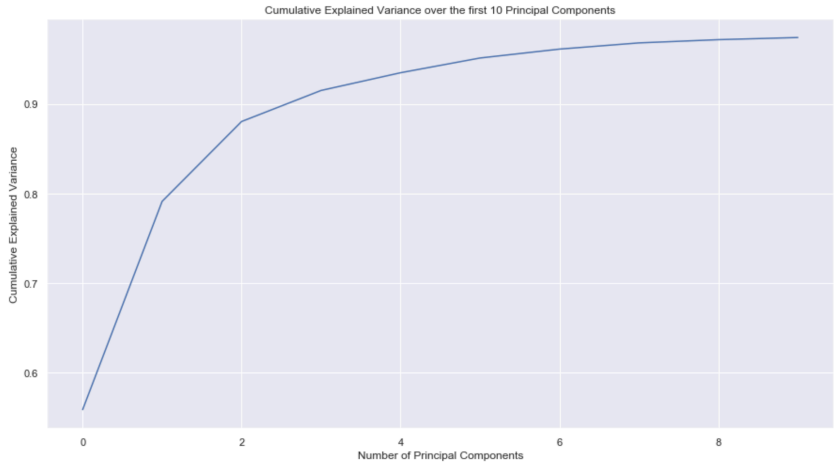


Figure 14: This figure shows the cumulative explained variance for the first 10 principal components.

NEO Coefficients The next two features are calculated using the NEO operator. Using equation 14 taken from [2] it is possible to calculate the NEO operator $\Psi(x[n])$. This gives an estimate of the energy content at a discrete time step n for the action potential $x[n]$ [2]. Similar

to Bestel et. al we calculated the NEO coefficient for the minimum. However, we added an additional feature by calculating the NEO coefficient also for the peak of the action potential.

$$\Psi(x[n]) = x^2[n] - x[n-1] \cdot x[n+1] \tag{14}$$

Feature Extraction Overview All in all, we use 9 calculated features and the first four principal components as features. A summary can be found in the table below. For reproducibility, the code used for extracting the signals and calculating the features is made available publicly as a python package which can be retrieved via: https://github.com/arnold3003/bachelor-thesis-public.git.

Feature Group	Feature Name
Geometric Features	Positive Amplitude
	Negative Amplitude
	Positive Action Potential Energy
	Negative Action Potential Energy
	Left Spike Angle
	Right Spike Angle
	Spike Width
PCA	First four principal components
NEO Coefficient	NEO Coefficient for Action Potential Minimum
	NEO coefficient for Action Potential Maximum

Table 1: A collection of all features which were extracted with their corresponding feature group.

The next step in feature engineering is to perform feature selection and evaluation to identify how well the features are suited to solve machine learning classification tasks, which will be done in the following section (section 3.2.3).

3.2.3 Feature Evaluation and Selection

Feature evaluation and selection is a very important step in every classification task. Among the many benefits it brings, one of them is that it can avoid overfitting by discarding irrelevant features. Furthermore, less features usually mean a reduced runtime of training

and predicting machine learning models and also an increase in machine learning performance can be observed when focusing on relevant features only.

Distribution of Features across different Clusters We started the feature evaluation by investigating the distribution of the features across different clusters. For this we used two different methods: First, we visualized the distributions using parallel boxplots. Although a boxplot only shows the distribution using the quartiles, we decided to still use this approach for comparing the distribution across many clusters. We therefore produced such visualizations, with the cluster on the x-axis and the calculated feature values on the y-axis, per feature and compared how different the distributions are across clusters. Then we decided to use analysis of variance (ANOVA) as a more statistical approach to check if the values of features are distributed differently across different clusters. To achieve this, we did an ANOVA per cluster and visualized the F- statistic and whether it is significant per cluster. We chose a significance level of 0.05.

Nevertheless, there is one limitation when using both of these methods. It is impossible to identify possible interactions between different features. For example, two features could be individually distributed very similarly across different clusters. However, a combination of those two features could end up being different across different clusters and therefore the features could still be valuable to be kept. For this reason, we continued to explore further methods for feature selection as well within this section.

Random Forest Feature Importance Next, we investigated the feature importance when training a random forest. For this, we first trained a random forest with 500 trees based on a training dataset of the dataset which is a sample of the original data set. The training dataset was obtained by using the first 70% of the dataset. Then, we calculated the feature importance of each feature for the random forest by using the intrinsic importance of each feature per tree and the averaging it over the entire forest. In more details, we calculated feature importance as also done by [11]: The total decrease in node impurity is weighted by the likelihood of reaching each node (which is approximated by the percentage of samples in the training data reaching that node). This is done per tree in the random forest. Next, we took the mean over

all trees, which now becomes the mean node impurity (MPI) used as the feature importance. These values were then compared across all features.

However, also this method has drawbacks. It is commonly known that this method does not work well with high cardinality features, which could actually be the case with our data. Furthermore, as this method uses the intrinsic feature importance of one machine learning model (random forest), it might not perform as well for other machine learning models. For example, for a deep neural net simpler features might suffice as it can combine information from several variables well. On the other hand, a simple linear regression model might need other features to best be able to learn (e.g. apply non-linear transformation). Nevertheless, we still believe that this method is applicable for a variety of machine learning models, as the random forest has non-linear interactions and the features selected as important have information necessary to predict the target. Anyhow, we also looked into one more method for feature importance, as described in the next paragraph.

Random Forest Permutation Feature Importance Finally, we also looked into the permutation feature importance when using a random forest machine learning model. Again, we trained a random forest with 500 trees based on the same training dataset as already described above. We then calculated a base accuracy based on the test dataset, which was the remaining 30% of the data. Next, we permuted always one feature randomly and compared the accuracy when predicting based on the permuted dataset.

Also this method has also a similar drawback as the previous method: It is again specific to this one machine learning model. Still, as also written previously, we believe that this method is in general valid also for other machine learning models. For this method the previously described drawback regarding high cardinality features does not apply for this method.

4 Results

In this section, we examine the results of the various feature extraction and selection methods which were applied within this work.

In order to find out the best feature set, we applied several feature selection techniques. Each technique is unique and the results can be interpreted individually. In the end we can the combine the outcomes of each method and give a combined recommendation for the best possible set.

The first technique was visualizing the distribution of the values of each feature for the different clusters using boxplots. Figures 15 and 16 are just two of the visualizations we used, the complete set of visualizations for all features can be found in the appendix in section 7.1. We chose the two figures, because they can be seen representative for visualizations of other features. Figure 15 shows the distribution of the feature ”Negative Action Potential Energy” of each cluster. What can be observed for this feature is that values are very differently spread for each cluster, therefore, making it possible to use the feature to predict the cluster based on the value of the feature. Still some of the clusters have similar and/ or overlapping ranges of values, but a lot of differentiating can already be done just by using this feature.

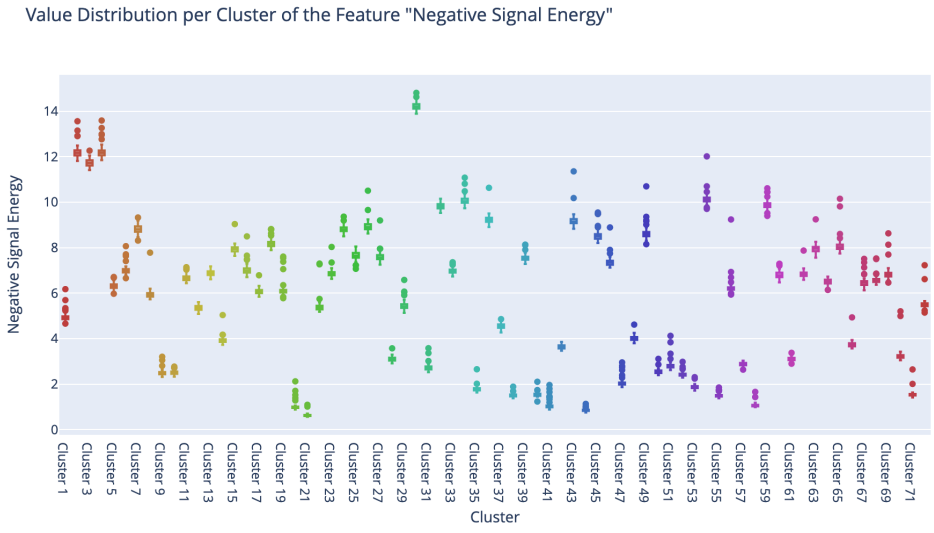


Figure 15: This figure shows a parallel boxplot which displays the distribution of values for the feature ”Negative Signal Energy” for each cluster in the dataset.

This is a different situation when looking at the distribution of the values of the feature ”NEO Coefficient Max” for each cluster in figure 16. In this case, most values are overlapping and no cluster stands out as having a unique distribution of values. Therefore, we believe that this feature is not likely to perform well when being used as input for predicting the clusters. When applying this method to the remaining extracted features, we identified that the following features also had a high overlap across different clusters: left spike angle and NEO coefficient min. The features positive amplitude, right spike angle, spike width and PCA 4 only showed very overlapping values for some of the clusters, while the value distribution was divers for other clusters. The remaining features (negative amplitude, positive action potential energy, PCA 1, PCA 2 and PCA 3) showed very different distributions across clusters, meaning they are most likely to perform well when being used in a prediction task.

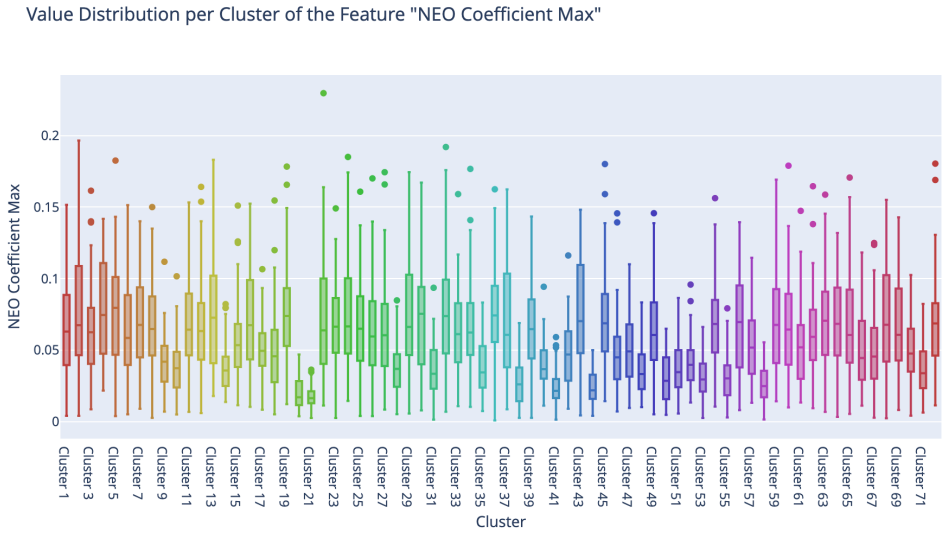


Figure 16: This figure shows a parallel boxplot which displays the distribution of values for the feature ”NEO Coefficient Max” for each cluster in the dataset.

Furthermore, we performed an ANOVA analysis of all features each across the different clusters. Figure 17 shows the results of the ANOVA analysis. As can be seen, the F-statistic of each feature is significant and all are greater than 1 (some are even much greater than 1 - they are cut off as Figure 17 only shows a maximum F-statistic of 10). If an F-statistic is close to 1, one could not that reject the null hypothesis that all clusters have the same mean. As

the F-statistics are greater than 1 and all are significant, we can reject the null hypothesis in this case. This would mean that all features have at least one cluster where the values have a different mean than another cluster, meaning that we can say the the distribution of the values are different. To summarize, from our ANOVA we deduce that all features are relevant and at least one cluster has a different distribution in values.

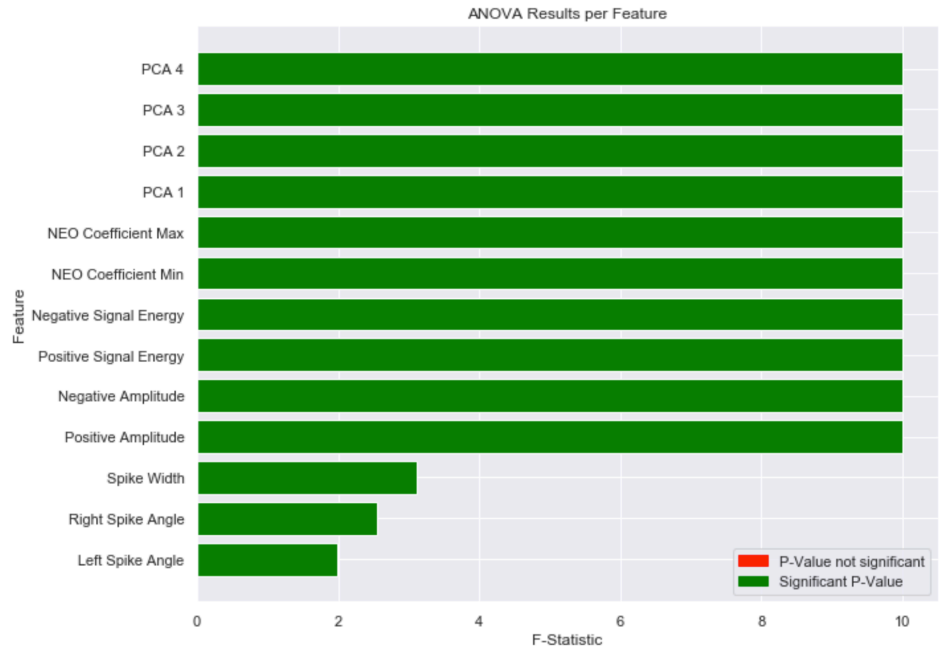


Figure 17: This figure shows a parallel boxplot which displays F-statistic for each feature in the dataset. The color is coded according to whether the p-value indicates significance or not (at a significance level of 0.05). The graph is cut off at a maximim F-statistic of 10.

Of course, when applying those two methods it is important to keep mind its limitations. One important limitation is that we only looked at each feature individually. One the one hand, it would still be possible that a combination of different features holds valuable information, but not the features by themselves. On the other hand, it is not possible to identify if several features have very redundant information and some of those features would not be necessary. Furthermore, using ANOVA we can only deduce that at least one cluster has a different mean value than another cluster. This means that in the extreme case all clusters could have the same mean value except for one cluster - making the feature still not very valuable. Therefore,

we didn't discard any feature just by using those methods, but rather combining the gained knowledge with the results of other methods.

Another method applied for identifying and selecting the best features was random forest feature importance. Figure 18 shows the importance of each feature. As can be seen on this figure, according to the random forest feature importance positive action potential energy and negative action potential energy are the most important action potentials. The 4 PCA features and positive and negative amplitude also have a high importance. The remaining features spike width, right spike angle, left spike angle, NEO coefficient min and NEO coefficient max have rather low importance (and are ranked in this order). Spike width still has the highest importance of these features and as it contains to some extent similar information as the features right spike angle and left spike angle, this might be a good cut-off point to define a feature set to be used for machine learning. However, as also this method has some drawbacks, we looked at yet another method for finding the feature importance.

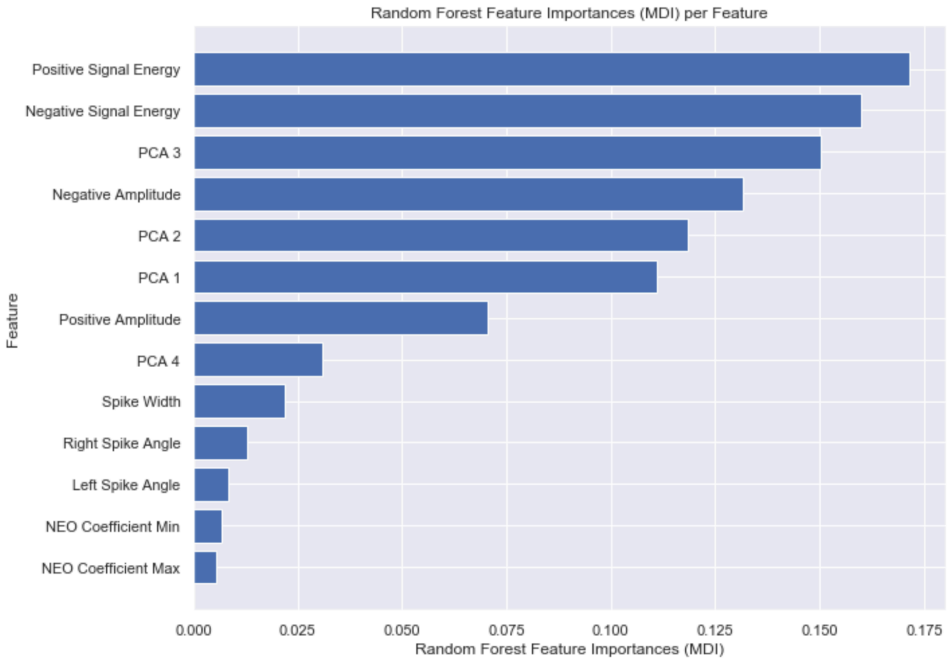


Figure 18: This figure shows the random forest feature importance of each calculated feature.

The final method we looked into was permutation feature importance using a random forest. Figure 19 shows the permutation feature importance across all features. In general, this

method showed a similar result as the previous, as the 8 most important features stayed the same (though not in rank between each other). This means that also the 5 least important features stayed the same (also this time the rank in between these features changed). However, what becomes quite clear from this visualization is that the least 5 important features, starting from the feature spike width, play only a very minor part in getting the correct prediction. Therefore, this method might suggest to cut off the feature to be used right before the spike width.

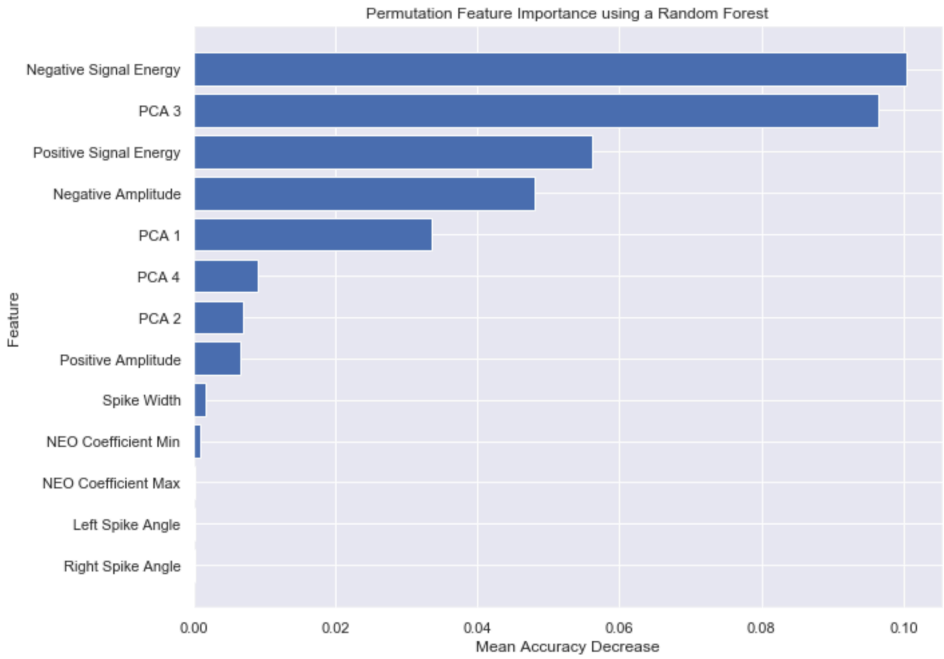


Figure 19: This figure shows the random forest permutation feature importance of each calculated feature.

Table 2 shows an overview of the findings regarding feature selection of this thesis. Feature importance is displayed using two buckets, low for everything below the proposed cut off value and high for everything above the proposed cut off value. The distribution of values is described in a written form. The color of the cell of the table reflects whether a method would be in favor of selecting a feature (green=yes, yellow=maybe and red=no). As ANOVA had the same result for each of the features (at least one cluster is significantly different to another cluster) it was not included in the table.

Feature Name	Distribution of values per cluster	RF Feature Importance	Permutation Feature Importance
Positive Amplitude	Wide spread between some clusters, others very similar	High	High
Negative Amplitude	Wide spread between different clusters	High	High
Positive Action Potential Energy	Wide spread between different clusters	High	High
Negative Action Potential Energy	Wide spread between different clusters	High	High
Left Spike Angle	Very similar distribution between clusters	Low	Low
Right Spike Angle	Wide spread between some clusters, others very similar	Low	Low
Spike Width	Wide spread between some clusters, others very similar	High	Low
PCA - component 1	Wide spread between different clusters	High	High
PCA - component 2	Wide spread between different clusters	High	High
PCA - component 3	Wide spread between different clusters	High	High
PCA - component 4	Wide spread between some clusters, others very similar	High	High
NEO coefficient for spike minimum	Very similar distribution between clusters	Low	Low
NEO coefficient for spike maximum	Very similar distribution between clusters	Low	Low

Table 2: This table shows the results of three of the applied feature selection methods per each feature.

5 Discussion

Within this thesis we aimed to give a proposal of a feature set which can be used for spike sorting. We set a focus onto the feature engineering aspect of spike sorting. For this, we wanted to extract useful features, some based on domain knowledge, and verify the quality of these features using feature selection methods. Finally, combining this information, we wanted to give a proposal for a feature set which can be used to predict which neurons fired given spikes.

After identifying and extracting the action potentials from the dataset, we identified 13 potential features. These features included 7 geometric features (positive amplitude, negative amplitude, positive action potential energy, negative action potential energy, left spike angle, right spike angle and spike width). Furthermore, we also used the first four principal components as features as well as the value of the NEO coefficient at the spike minimum and at the spike maximum.

We then evaluated these 13 features using four different feature evaluation methods. When comparing the results of the different feature selection methods, it becomes clear that there is a trend towards similar features being selected using different methods. Overall, based on our research and analysis we would suggest the following feature set to be used for predicting from which spike a specific neuron originates from:

- Positive amplitude
- Negative amplitude
- Positive action potential energy
- Negative action potential energy
- PCA components 1-4
- Spike width (maybe)

The feature spike width is the only questionable feature in this set, as the analysis wasn't fully conclusive when it comes to this feature (different methods would suggest opposing outcomes). Therefore, we would propose to test a feature set with and without that feature when running a machine learning-based prediction task.

Within this thesis, we were able to come up with a concrete proposal for a feature set to be used for spike sorting of neuronal spikes. This makes it possible to continue the process of spike sorting by fitting a machine learning model on a dataset transformed using these features. In other words, this enables future researchers to continue the process of spike sorting without or with only little domain knowledge, as the part where extensive domain knowledge is needed, feature engineering, is already completed.

We believe that one of the limitations of this thesis is at the same time also one of the main strengths of this thesis: the sharp focus on the feature engineering part of spike sorting. With such a sharp focus, we were able to go into depth and properly explore different aspects of this topic. However, we are aware that feature engineering in itself has rather little value and only becomes valuable when the features are being using in a machine learning application. Therefore, feature engineering can usually not be seen as a completely independent topic as the effects on the machine learning performance should be considered when optimizing the feature set. Nevertheless, we are convinced that a deeper investigation of the feature engineering aspects of spike sorting brings value, as it can easily be combined with the classification aspects of spike sorting. Furthermore, we want to mention that we were not completely blind to the classification aspects of spike sorting. One of our feature selection methods, random forest permutation feature importance, even involves training a machine learning model (random forest) based on test data and then evaluating the performance based on different feature sets.

To summarize, we were able to produce a proposal for a feature set to be used for neuronal spike sorting and recommend this to be used for future applications.

6 Conclusion

The aim of this thesis was to give a proposal for a features set which can be used for fitting a machine learning model to predict the originating neuron of each action potential. We were able to achieve this by first extracting action potentials from a dataset with continuous recordings of single neuronal cell responses. Next, we extracted several features for each action potential and evaluated these features using several feature selection methods. By combining the results of all of these methods, we came up with the proposed feature set.

This proposed feature sets gives the basis for spike sorting to be performed on the dataset. The knowledge gained in spike sorting can then at some point be used for the previously discussed applications in section 2.3. There are still many additional major parts needed for applications such as brain-machine interfaces. These would include all the hardware of actually recording the spikes as well as the hardware for performing the desired task (such as a robotic arm). Also, there needs to be further research done in order to have meaningful predictions and to be able to translate them into actionable information for machines. Yet, we believe that a focused research into one component of spike sorting is still valuable as a wider research aim would have undoubtedly meant less focus on details, considering the limited time with which we performed this research.

Nevertheless, we still faced some limitations even when only considering the narrow field of feature engineering for spike sorting. One of the main limitations was the lack of action potentials with labels of their corresponding neurons. This is why we had to fallback to simulated data. Although the action potentials of this data are based on real action potentials, we still suspect that real data might behave differently and an ideal feature set might look differently. Still, in this case our publicly made available code can be used to reproduce the signal and feature extraction steps based on a different data set.

For future research regarding feature engineering for spike sorting, we would recommend to use a feature set based on real data actually recorded in a human brain and then labelled by experts. Furthermore, we believe that exploring further possible features apart from the ones mentioned within this thesis would be another potential to improve the feature set.

To conclude, within this thesis we were able to come up with a proposed feature set which

can be used for fitting a machine learning model to predict the originating neuron of each action potential. Although we are aware that there are still certain limitation to this work, we believe that the outcome is valuable and can be used for further research into this topic.

List of Figures

List of Tables

References

[1] Gopala K. Anumanchipalli, Josh Chartier, and Edward F. Chang. “Speech synthesis from neural decoding of spoken sentences”. In: *Nature* 568.7753 (Apr. 2019), pp. 493–498. URL: http://www.nature.com/articles/s41586-019-1119-1.

[2] Robert Bestel, Andreas W. Daus, and Christiane Thielemann. “A novel automated spike sorting algorithm with adaptable feature extraction”. In: *Journal of Neuroscience Methods* 211.1 (2012), pp. 168–178.

[3] V. Srinivasa Chakravarthy. *Demystifying the Brain*. 2019.

[4] Marcos Fabietti, Mufti Mahmud, and Ahmad Lotfi. “A Matlab-Based Open-Source Toolbox for Artefact Removal from Extracellular Neuronal Signals”. In: *Mahmud M., Kaiser M.S., Vassanelli S., Dai Q., Zhong N. (eds) Brain Informatics. BI 2021. Lecture Notes in Computer Science, vol 12960.* Vol. 12960 LNAI. 2021, pp. 351–365. URL: https://link.springer.com/10.1007/978-3-030-86993-9_32.

[5] Mailys C. M. Faraut, April A. Carlson, Shannon Sullivan, Oana Tudusciuc, Ian Ross, Chrystal M. Reed, Jeffrey M. Chung, Adam N Mamelak, and Ueli Rutishauser. “Dataset of human medial temporal lobe single neuron activity during declarative memory encoding and recognition”. In: *Scientific Data* 5.1 (Dec. 2018), p. 180010. URL: https://www.nature.com/articles/sdata201810.

[6] Brian Fiani, Taylor Reardon, Benjamin Ayres, David Cline, and Sarah R Sitto. “An Examination of Prospective Uses and Future Directions of Neuralink: The Brain-Machine Interface”. In: *Cureus* 13.3 (2021).

[7] Carl Gold, Darrell A. Henze, and Christof Koch. “Using extracellular action potential recordings to constrain compartmental models”. In: *Journal of Computational Neuroscience* 23.1 (2007), pp. 39–58.

[8] Christoph Guger, Brendan Z. Allison, and Aysegul Gunduz. *Brain-Computer Interface Research*. Ed. by Christoph Guger, Brendan Z. Allison, and Aysegul Gunduz. SpringerBriefs in Electrical and Computer Engineering. Cham: Springer International Publishing, 2021, pp. 1–11. URL: https://link.springer.com/10.1007/978-3-030-79287-9.

[9] Leigh R. Hochberg, Daniel Bacher, Beata Jarosiewicz, Nicolas Y. Masse, John D. Simeral, Joern Vogel, Sami Haddadin, Jie Liu, Sydney S. Cash, Patrick van der Smagt, and John P. Donoghue. “Reach and grasp by people with tetraplegia using a neurally controlled robotic arm”. In: *Nature* 485.7398 (May 2012), pp. 372–375. URL: https://linkinghub.elsevier.com/retrieve/pii/S0140673612618169%20http://www.nature.com/articles/nature11076.

[10] Arthur K. Kordon. *Applying Data Science*. Cham: Springer International Publishing, 2020, p. 273. URL: https://link.springer.com/10.1007/978-3-030-36375-8.

[11] Breiman Leo, Jerome Friedman, Richard Olshen, and Charles Stone. *Classification And Regression Trees*. 1st Editio. New York: Routledge, 1984, p. 368.

[12] Elon Musk. “An integrated brain-machine interface platform with thousands of channels”. In: *Journal of Medical Internet Research* 21.10 (2019), pp. 1–14.

[13] R. Quian Quiroga, Z. Nadasdy, and Y. Ben-Shaul. “Unsupervised spike detection and sorting with wavelets and superparamagnetic clustering”. In: *Neural Computation* 16.8 (2004), pp. 1661–1687.

[14] Adrien B. Rapeaux and Timothy G. Constandinou. “Implantable brain machine interfaces: first-in-human studies, technology challenges and trends”. In: *Current Opinion in Biotechnology* 72 (Dec. 2021), pp. 102–111. URL: https://linkinghub.elsevier.com/retrieve/pii/S095816692100183X.

[15] Hernan Gonzalo Rey, Carlos Pedreira, and Rodrigo Quian Quiroga. “Past, present and future of spike sorting techniques”. In: *Brain Research Bulletin* 119 (2015), pp. 106–117. URL: http://dx.doi.org/10.1016/j.brainresbull.2015.04.007.

[16] Jonathan R. Wolpaw, Niels Birbaumer, Dennis J. McFarland, Gert Pfurtscheller, and Theresa M Vaughan. “Brainâ□□computer interfaces for communication and control”. In: *Clinical Neurophysiology* 113.6 (June 2002), pp. 767–791. URL: https://linkinghub.elsevier.com/retrieve/pii/S1388245702000573.

7 Appendix

7.1 Distribution of Feature Value

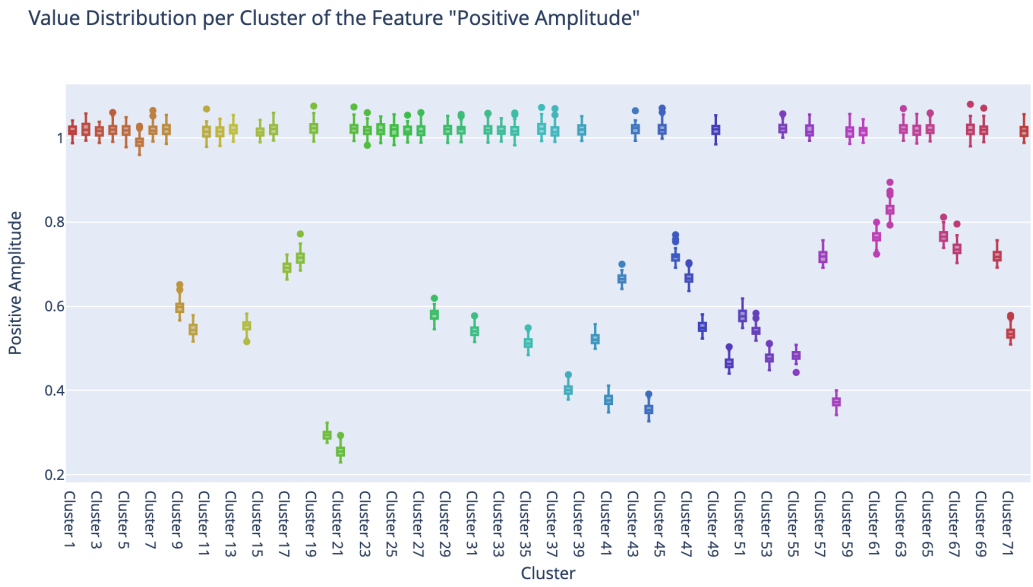


Figure 20: This figure shows a parallel boxplot which displays the distribution of values for the feature ”Positive Amplitude” for each cluster in the dataset.

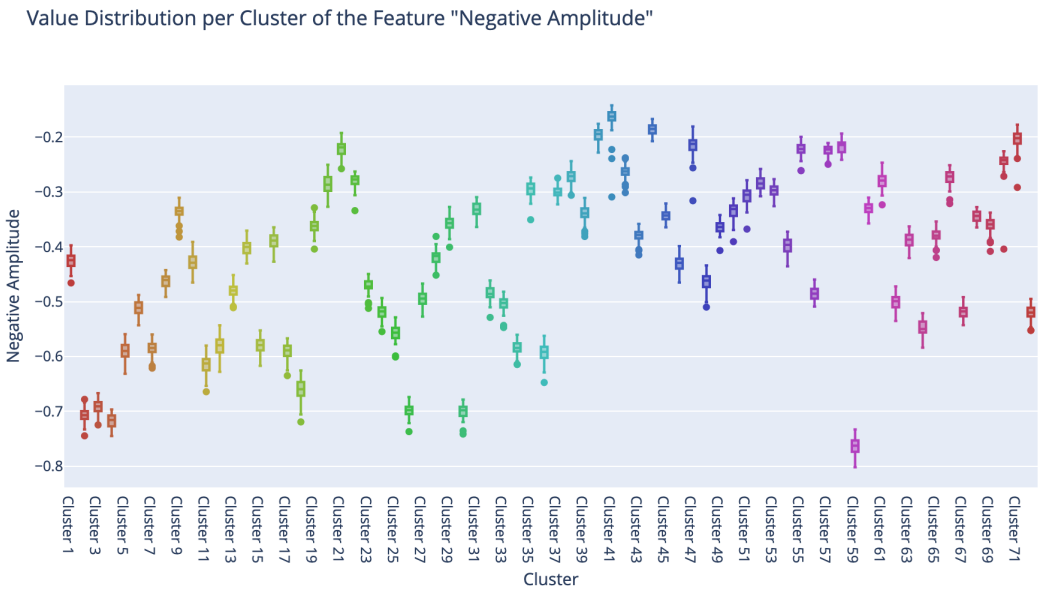


Figure 21: This figure shows a parallel boxplot which displays the distribution of values for the feature ”Negative Amplitude” for each cluster in the dataset.

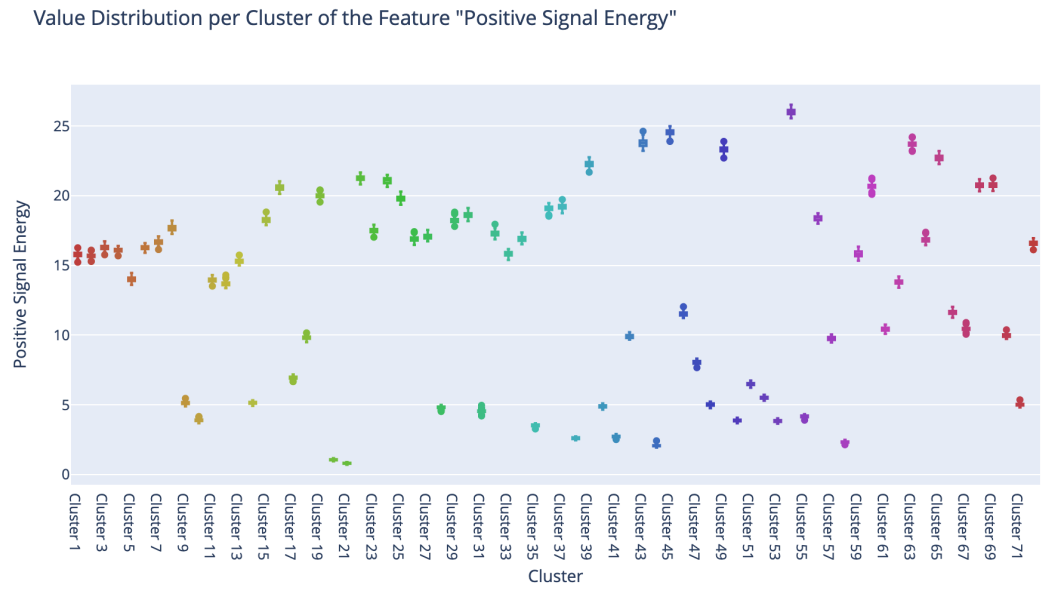


Figure 22: This figure shows a parallel boxplot which displays the distribution of values for the feature ”Positive Signal Energy” for each cluster in the dataset.

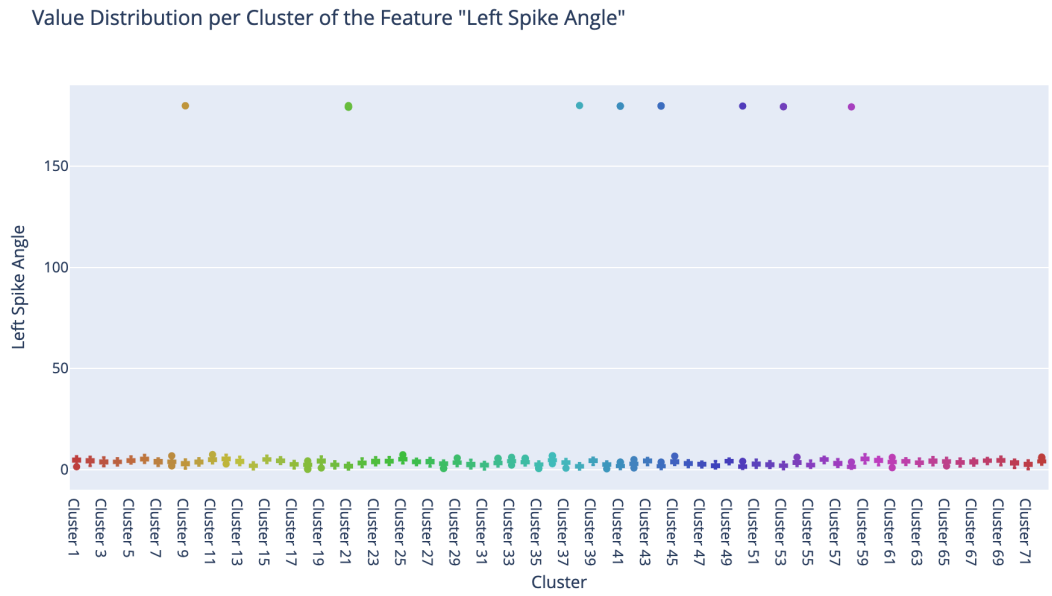


Figure 23: This figure shows a parallel boxplot which displays the distribution of values for the feature ”Left Spike Angle” for each cluster in the dataset.

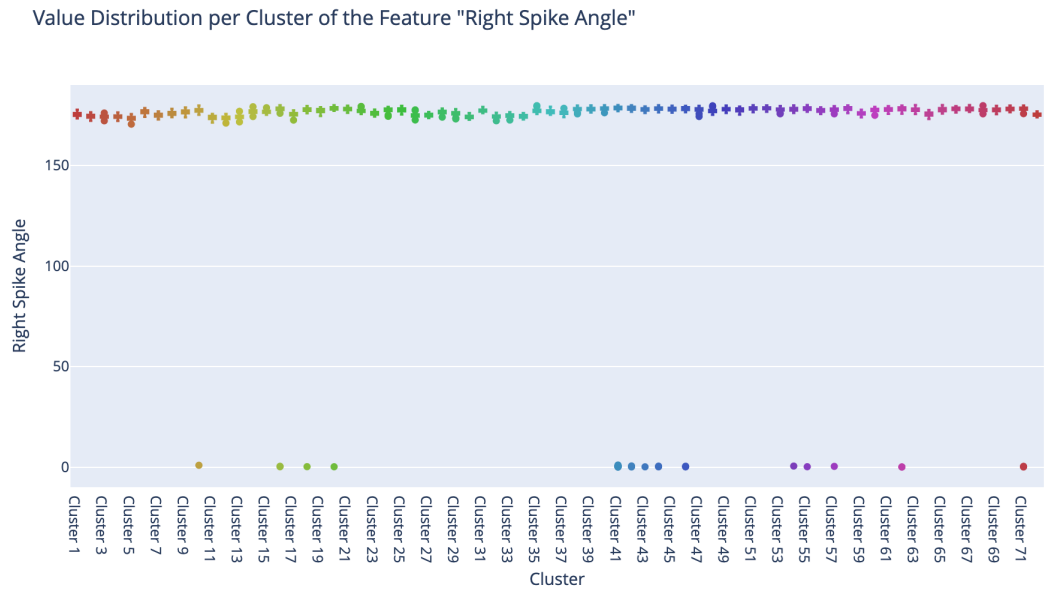


Figure 24: This figure shows a parallel boxplot which displays the distribution of values for the feature ”Right Spike Angle” for each cluster in the dataset.

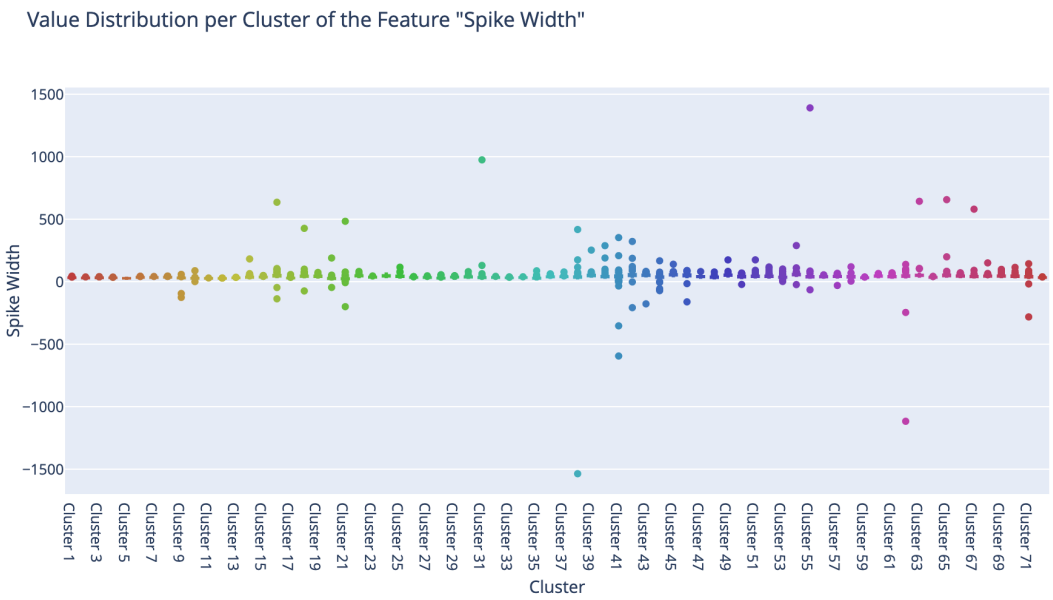


Figure 25: This figure shows a parallel boxplot which displays the distribution of values for the feature ”Spike Width” for each cluster in the dataset.

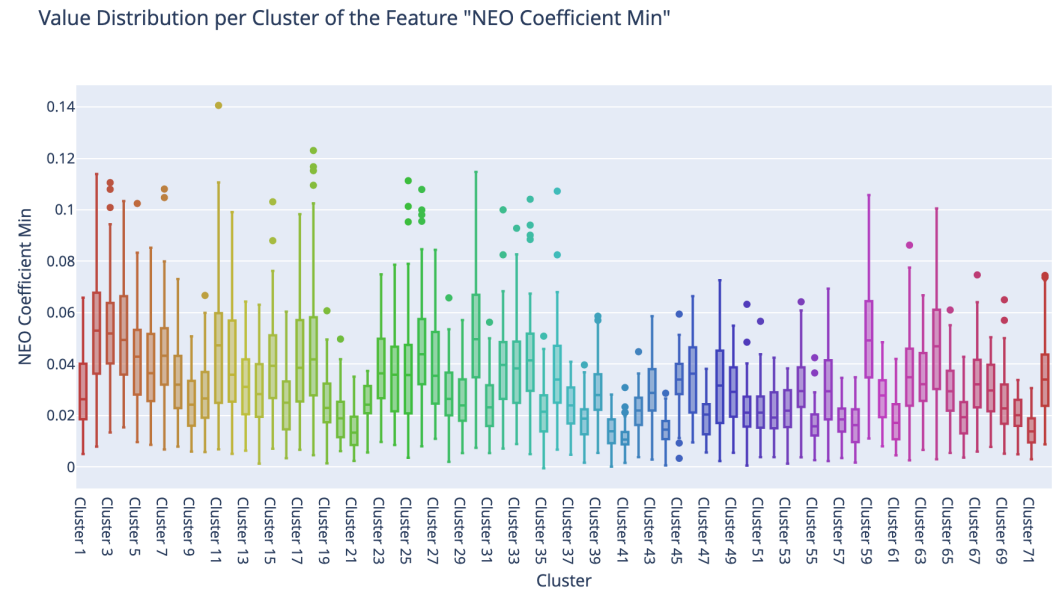


Figure 26: This figure shows a parallel boxplot which displays the distribution of values for the feature ”NEO Coefficient Min” for each cluster in the dataset.

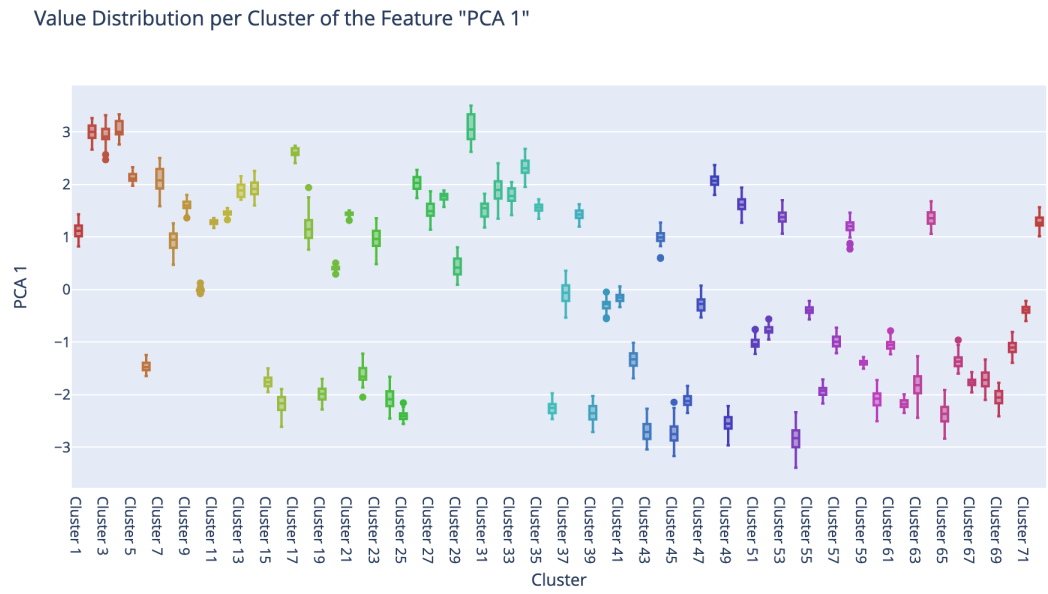


Figure 27: This figure shows a parallel boxplot which displays the distribution of values for the feature ”PCA 1” for each cluster in the dataset.

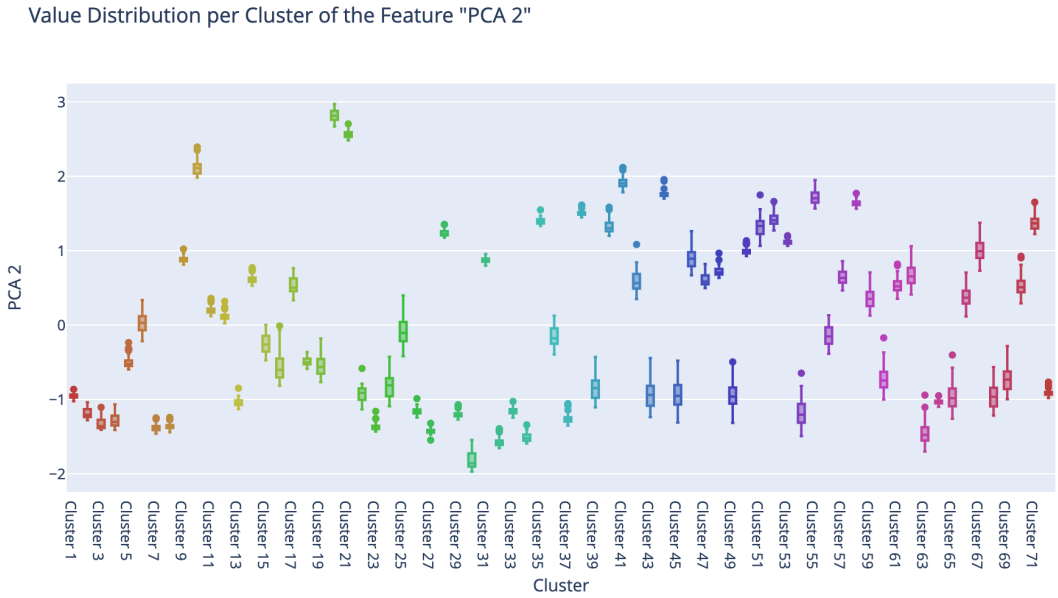


Figure 28: This figure shows a parallel boxplot which displays the distribution of values for the feature ”PCA 2” for each cluster in the dataset.

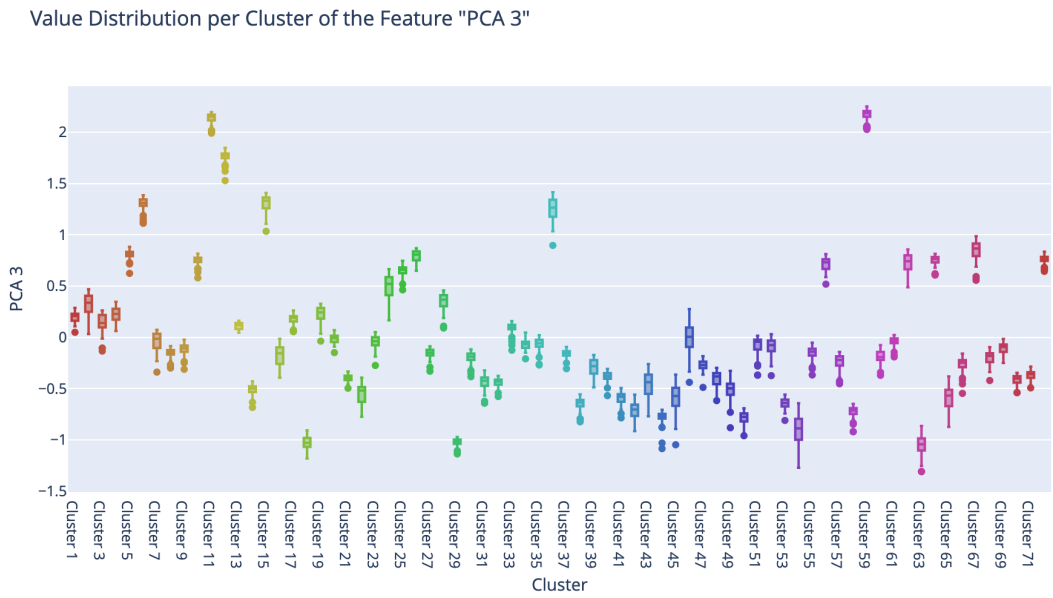


Figure 29: This figure shows a parallel boxplot which displays the distribution of values for the feature ”PCA 3” for each cluster in the dataset.

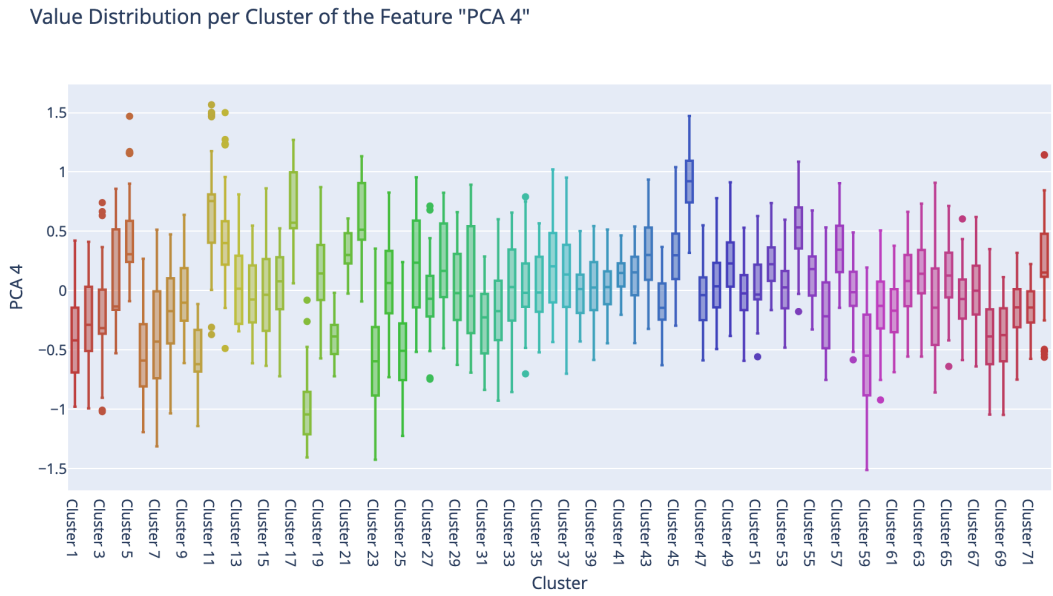


Figure 30: This figure shows a parallel boxplot which displays the distribution of values for the feature ”PCA 4” for each cluster in the dataset.