# Investigating the role of data augmentation for EEG artifact detection

**Albert Kjøller Jacobsen (s194253)**
**Aron Djurhuus Jacobsen (s194262)**
**Phillip Chavarria Højbjerg (s184984)**

21st of June, 2021

---

---

**02466: Project work - Bachelor of Artificial Intelligence and Data**

---

**GitHub repository with code:** `https://github.com/AronDJacobsen/EEG_epilepsia`

**Abstract**

In the past decade, Artificial Intelligence (AI) has continuously advanced in our modern society, and has time and time again shown the various fields in which it can prove useful. Thus, it comes as no surprise that AI can help save lives through classifications in the medical world as well. A general problem in the medical world is the so-called epilepsy treatment gap, a gap of around 30 million people still suffering from undiagnosed epilepsy, as estimated by the World Health Organization. Treatment of epilepsy is fairly simple and inexpensive, however the difficult part is the diagnosis of epilepsy in the first place, as EEG recordings of the brain are required. Previously, EEG signals have been difficult to record in low and middle income countries, as they require expensive, and immobile equipment, requiring people to travel far and wide for a diagnosis. The company BrainCapture, aims at closing the current gap by developing an accessible solution of cheap EEG caps combined with AI tools for easier and cheaper diagnosis of epilepsy. However, EEG recordings are highly contaminated by noise and unwanted neural activity, known as artifacts, which can cover important neural activity required for epilepsy detection. This study therefore aims at investigating how machine learning (ML) tools and techniques can be used to detect artifacts, by training various models on the Temple University Hospital Artifact (TUAR) data set. Due to the individuality and everchanging state of the human brain, robustness of ML classifiers, in the sense that they should be able to generalize to new unseen subjects, is paramount. For this reason, an approach using various data augmentation techniques to create additional training data was used to cover more of the feature space when training. The effect of these augmentation methods and whether they provide robustness for binary EEG artifact classifiers, were examined through comparison between the balanced accuracy and sensitivity scores of various binary classifiers trained on augmented data, and a control experiment without data augmentation. To examine robustness, stratified cross-validation was applied by splitting the training and test sets based on subjects. The results indicate that data augmentation improves the performance of some machine learning models, however, that there is no significant difference between the best binary classifier using augmented data, if compared to the best binary classifier from the control experiment across all the artifacts in the TUAR data set. Furthermore, predicting using an ensemble learner classifying by majority vote between the 5 most disagreeing of the best performing classifiers, did not show improvement when compared to the control classifiers either. Overall, as no improvement nor decrease in performance were to find for the best evaluated classifiers using augmented data, and as performance was found to be on par with state-of-the-art research, it was argued that the classifiers were quite robust. When combining the best binary classifiers for each artifact into a multi-label classifier, a way to deal with the apparent trade-off between balanced accuracy and sensitivity was suggested, though it is dependent on the use-case.

# Contents

# 1 Introduction

The World Health Organization (WHO) [1] estimates 30 million people with undiagnosed epilepsy in low- and middle-income countries (LMIC), of which 70% can be treated with inexpensive medication if diagnosed - this is referred to as the epilepsy gap. The main diagnostic tool is electroencephalograms (EEG) which previously was of limited accessibility but has now become more accessible to LMIC with i.a. BrainCapture's low-cost EEG caps that provide opportunities of mobile EEG recordings [2]. Yet, EEG signals are typically extremely sensitive to noise with the cortical signals of interest appearing at low frequencies and amplitudes. Besides the neural activity of interest when diagnosing from EEG data, the EEGs are oftenly contaminated by unwanted recorded signals or inference that disrupt the quality of the measured signal. Instances of such activity is referred to as artifacts - a term covering both biologically and technically induced noise, such as eye movement and heart rate or the line frequency and malfunctioning electrodes, respectively. These artifacts have a negative influence on the legibility of an EEG, possibly leading to erroneous interpretation of specific segments of the signal, thereby reducing the opportunity of diagnosis of i.a. epilepsy. Currently, employment of mobile EEG recordings are enrolled in LMIC [3] with a U.S. board-certified technologist and medical students and neurology residents from said countries, being responsible for recording the data after which neurophysiologists analyze the recordings online. As of now it requires familiarity with on-site EEG recordings to correctly detect artifacts and prevent them from negatively impacting diagnosis of EEGs. Meanwhile, brain-signals are incredibly individual, and constantly changing throughout life, making data even harder to analyze. All in all there is a huge potential for closing in on the current epilepsy gap by being able to easily detect contaminating artifacts.

For this reason, this study set out to investigate how machine learning (ML) methods can be used to classify EEG artifacts in order to obtain cleaner EEG data directly from on-site recordings. This aim is approached through a machine learning pipeline, condensed into the following steps, i) preparation of artifact-contaminated EEG data, ii) balancing of unbalanced data, iii) expansion of the training-set through various data augmentation methods, and finally iv) evaluation of the models' abilities to detect artifacts. An illustration of the mentioned pipeline is seen in Figure 1. To successfully evaluate our approach, an assessment of state-of-the-art knowledge on EEG signals, as well as methods used when investigating EEG artifacts in machine learning is essential.

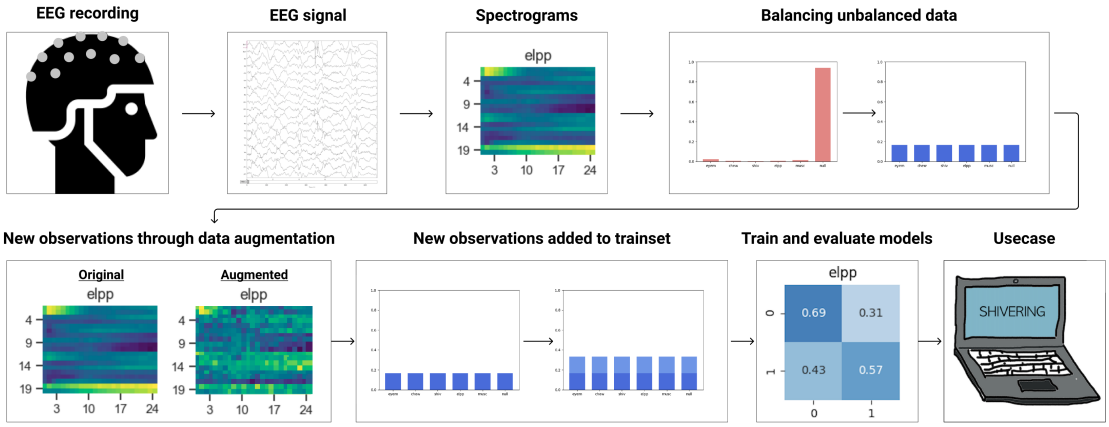



**Figure 1:** Machine Learning pipeline for classifying EEG artifacts.

## 1.1 Introducing EEG data

An EEG is a method used for monitoring cortical activity recorded by measuring voltage fluctuations within neurons. These fluctuations are measured by observing the difference in voltages between electrodes placed - typically non-invasively - at determined positions on the scalp. The placement of the electrodes are setup using specified configurations, that among other things depend on the number of electrodes available in the worn EEG cap. On Figure 2 one of the most popular configurations, as well as the configuration used throughout this project - the 10/20-configuration - is shown as an example of this.

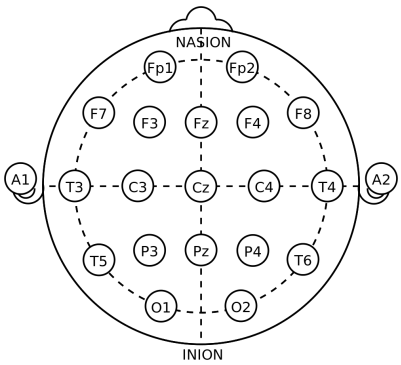



**Figure 2:** Standard 10/20 configuration of EEG electrodes. Illustration taken from Wiki [4].

The continuously measured signals from EEGs can be transformed from the time domain into the frequency domain to obtain knowledge about spectral content of the cortical observations. The spectral content of EEGs are typically divided into frequency bands containing frequencies of the most observed cerebral signals - these are known as the Alpha-, Beta-, Delta-, Gamma-, Mu- and Theta-bands. Most of these - as well as EEG artifacts - are present in bands (ranges) in the lower frequencies ranging from 1-30 Hz [5] [6]. Both the content of EEGs in the frequency domain as well as in waveform of the raw signals can be used when investigating artifacts and neural disorders, i.e. epileptic seizures [7]. This report will focus on artifact detection in the frequency-domain, however.

## 1.2 Automatic detection of EEG artifacts

Manually locating artifacts during the physical setup while recording the EEG in order to obtain a cleaner EEG signal, or analyzing the EEG-signal for artifacts afterwards, can be both difficult and time-consuming for diagnosticians, due to the continuity of the signal [8] and therefore requires experienced EEG annotators to do so. Thus, having a well-functioning automatic method for detecting artifacts in real-time, would be beneficial for the analysis of EEGs and ease the diagnosis of neurological conditions, like epilepsy. Such methods have been approached in former EEG literature in two different manners; i) on-site detection of artifacts used for qualitatively improving the observed EEG signals and ii) off-line analysis used for de-noising EEG signals by removal of artifacts through post-processing. Both of these methods are acknowledged, with the on-site approach resulting in cleaner recorded signals that are more accessible to interpret for clinicians [9], whereas the off-line approach might result in removal of cortical activity of interest, but ease the recording of the signals, and can possibly account for disturbances like ocular activity influencing perceptual and cognitive processes [10].

Regardless of which of the two approaches is chosen, being able to identify artifacts is the cornerstone of obtaining cleaner EEG signals. A widely established way of doing this is by the use of Independent Component Analysis (ICA) [11], which is a blind source separation approach (BSS), that linearly transforms a multivariate signal into subcomponents by minimizing the statistical dependency between these components. ICA has been proved to significantly cancel background noise and separate a mixed signal wherefore it creates a good basis for separating artifacts from desired cortical activity in EEGs [12]. While ICA is an analytical tool applied to EEG signals in the temporal domain, one might benefit of investigating EEGs in the frequency domain as well, i.e. by applying Fourier analyses to investigate the power spectra (amplitude of frequencies) within the EEGs. These are often presented as time-frequency representations (spectrograms) and they are, as previously mentioned, great for analyzing the frequency content of continuous signals. Throughout the literature on EEG artifact detection, feature combinations of ICA-based topographic plots and power spectra based on the Independent Components (IC), as well as plain time-frequency representations are used as features for detecting EEG artifacts. For the sake of clarity, the execution of this project solely considered time-frequency representations as a solution.

As of now multiple analytical approaches ranging from statistically based methods using correlation between channels and IC components [13], to methods using differential energy and derivatives [14], as well as widely ranging machine learning methods have been used to successfully detect artifacts; especially eye movement, eye blinking and muscle contractions [15] [16]. So far, the most prominent results have been found in the field of machine learning. Though there is no unifying

machine learning approach, the use of Support Vector Machines (SVM) applied to IC-based features have been well investigated with sensible results such as accuracies at approx. $90.9 - 95\%$ [15] [17] and approx. $81.4 - 86.2\%$ or $91.6 - 93.5\%$ on a balanced and an unbalanced data set, respectively [18]. The latter mentioned paper written by Radüntz et al. (2017) further applies a Linear Discriminant Analysis (LDA), a Logistic Regression (LR) as well as an Artificial Neural Network (ANN), with best found accuracies between $86.4 - 91.4\%$ and $91.7 - 95.2\%$ on respectively balanced and unbalanced data when using an ANN. Other studies [19] obtain similar results for ANNs with a mean accuracy at approx. 91%.

One of the newer papers on supervised machine learning that benchmark methods on EEG data is the paper by Roy (2019) [20] investigating the open-sourced Temple University Hospital (TUH) Artifact data base, published in 2018. This data base holds more labels, namely eye movement ('eyem'), chewing ('chew'), shivering ('shiv'), electrode pop ('elpp'), muscle contraction ('musc') and null ('null'), than seen in most other literature. The paper by Roy systematically tests a varying palette of machine learning approaches (one from each supervised learning paradigm) such as a LDA, a Random Forest Classifier (RF), a K-Nearest-Neighbours (KNN), Gaussian Naïve-Bayes (GNB) and others. By reporting the weighted F1-score varying between $73.75 - 80.12\%$ and sensitivity scores (noted as $S_{artifact}$) that are highly dependent on the artifacts. The paper demonstrates that there is a great potential of using supervised machine learning approaches on the TUH Artifact data base and that improvement can possibly be made on models classifying the 'shiv' and 'elpp' artifacts. In Table 1 is provided an overview of the results found in the paper by Roy.

**Table 1:** Results found in the benchmark paper on artifact classification by Roy (2019) [20].

| Algorithm | Weighted-F1 | Accuracy | $S_{\text{eyem}}$ | $S_{\text{chew}}$ | $S_{\text{shiv}}$ | $S_{\text{elpp}}$ | $S_{\text{musc}}$ | $S_{\text{null}}$ |
|---|---|---|---|---|---|---|---|---|
| AdaBoost | 0.7375 | 62.57% | 62.51% | 68.63% | 2.31% | 28.30% | 62.88% | 63.17% |
| GaussianNB | 0.7773 | 67.79% | 63.19% | 72.67% | **16.32**% | 13.99% | 43.47% | 69.03% |
| k-NN | 0.7476 | 63.76% | 60.77% | **86.07**% | 5.95% | 26.06% | 48.55% | 64.67% |
| LDA | **0.8012** | **71.43**% | 58.73% | 62.73% | 2.50% | 26.99% | **70.76**% | **72.39**% |
| MLP | 0.7787 | 68.23% | 68.36% | 73.80% | 2.05% | 34.64% | 65.35% | 68.93% |
| Random Forests | 0.7834 | 68.80% | **73.35**% | 80.35% | 3.00% | 35.26% | 67.25% | 69.39% |
| SGD classifier | 0.7887 | 69.57% | 63.06% | 73.61% | 3.10% | 28.79% | 69.01% | 70.36% |
| XGBoost | 0.7996 | 71.19% | 72.38% | 74.08% | 2.75% | **38.75**% | 67.91% | 71.90% |

Other researchers have investigated supervised learning models such as Bayesian deep learning in order to treat the problem of ambiguously labelled EEG data as well as to have a confidence metric for artifact predictions. This has been shown to obtain an accuracy of approximately 95% [16]. The quality of having probabilistic labelled events is suggested to be of great use for clinicians examining off-site removal of EEG artifacts, since complete removal of artifacts might result in removal of important brain signals as well.

In general, many approaches with impressing results have been executed at the state-of-the-art, but in most of the literature well-stated baseline models, as well as the proportion between the majority artifact label and the remaining labels are not elaborated on. The widely used accuracy measure, however, which is highly sensitive to imbalanced data is not preferable compared to other performance metrics, such as the balanced accuracy, sensitivity scores, weighted F-scores or area under the curve scores (AUC). To provide for a comparable and least unbiased estimate (referring to possible imbalance of the test data) of the investigated supervised learners performance, this report will primarily use the balanced accuracy score, as well as the sensitivity score, of six binary artifact classifiers, trained and evaluated on time-frequency representations of EEG data.

## 1.3 Data augmentation techniques for EEG data

Due to the individuality and everchanging state of the human brain, the process of identifying artifacts across multiple individuals and generalizing from data becomes challenging, as it will require vast amounts of data. While the availability of large open-source EEG data sets is continually increasing, the data sets still suffer from a low samples-to-features ratio, making it difficult to create a classifier that is both reliable and robust [21]. Thus, the need of data augmentation becomes apparent. Throughout the last couple of years, multiple papers have described state-

of-the-art data augmentation techniques on EEG-data resulting in various gains of performance [21]. Some of these techniques include Sliding Windows, Synthetic Minority Oversampling Technique (SMOTE), Generative Adversarial Networks (GAN) and Noise Addition. Another technique which has yet to be used in the context of EEG data, but shows very promising results in other classification problems is MixUp [22]. State-of-the-art of these methods will be described in the following section, whereas a description and further details of their application are elaborated in the "Methods"-section.

In a paper by Lashgari et al. (2020) [21], reviewing the techniques and findings of 53 papers about data augmentation in EEG during the past 5 years, they found Sliding Windows to be the most commonly used technique, appearing in 24% of the reviewed papers. One of these is the paper by Mousavi (2019) [23], which used Sliding Windows in a 6-class classification problem, determining sleep stages. The paper reported an accuracy of 93.55% while using Sliding Windows, however it did not mention the accuracy of a model without data augmentation. The study also explored the GAN-method, which obtained an accuracy of 72.33%. It is important to note that the paper used data augmentation as a way to over-sample underrepresented classes, and not a way of generating more data as a whole.

Deepa et al. (2010) [24], proposed using Synthetic Minority Oversampling Technique (SMOTE) on a Brain Computer Interface data set, in order to over-sample a minority-class. The paper reports an accuracy of 60.71% using SMOTE compared to 43.45% without, using a Naïve-Bayes classifier.

Piplani et al. (2018) [25] were able to increase the accuracy of a Passthought Authentication System, which used EEG signals in order to log into devices, from 90.8% to 95.0% using GAN. In another study, Zhang and Liu (2018) [26] proposed using what they called a conditional Deep Convolutional Generative Adversarial Network (cDCGAN) in order to generate further observations for a Brain Computer Interface-related classification problem. Their data augmentation warranted an accuracy increase from 83% to 86%.

In another study by Yin et al. (2017) [27] [28] added Gaussian noise onto the EEG feature vectors, improving the accuracy of their Mental Workload classifier, classifying brain activity, from 76.5% to 85.5%. Similarly, Salama et al. (2018) [29] saw an accuracy-gain from 79.11% to 88.49%, in an emotion recognition classifier, classifying valence and arousal.

MixUp was first described by Zhang et al. (2017) [22]. This paper shows the method outperforming multiple other methods, by improving the generalization error of multiple state-of-the-art models such as ImageNet and CIFAR. The method is also shown to improve the robustness of multiple models, built on various types of data, as well as data with various levels of label corruption. As to our knowledge, this study will be the first to implement MixUp in an EEG-related context, however, since the method works by creating linear-combinations of two observations and their one-hot encoded labels, there is no reason to think it would not work on EEG data as well.

To sum up, all of these five augmentation techniques (Sliding Windows, SMOTE, GAN, Noise Addition and MixUp) have shown improvement in model performance of which reason all of them will be investigated in this report. Some of the methods, namely Sliding Windows and SMOTE, will be investigated in the preprocessing and preparation-part of the pipeline, whereas the remaining three techniques will be investigated as separate methods in the model fitting-part of the pipeline. Finally, the results from the augmented and non-augmented data fitted models will be compared in the evaluation part of the overall pipeline.

## 1.4 Introducing the use-case

The results found from this study are meant to enlighten on how to achieve BrainCapture's goals of ensuring high quality in mobile and remote EEG recordings to help closing the epilepsy treatment gap. Thus, by providing for easier and cheaper EEG recordings that can be made on-site at remote locations in LMIC, BrainCapture aims at making EEGs more accessible. As the goal is to reduce the impact of artifact contamination when recording the EEGs a special focus will be on avoiding false-negative predictions meaning failing to classify artifacts being present and rather be cautious and have false-positive predictions where artifacts are being predicted as present even though they are absent. However, it is naturally not of interest to have too many false-positive predictions as

the technician would then have to correct the test subject too frequently. These considerations will be born in mind throughout this project.

### 1.5 Expectations

This study will focus on EEG artifact classification applicable for on-site detection of EEG artifacts using the open-sourced TUH Artifact data corpus, published in 2018. By creating classifiers which are able to successfully predict the various artifacts, this study aims to be able to help diagnosticians obtain a cleaner recorded EEG signal, without losing useful information in the process, by continuously being able to tell them, and in turn their patient, when an artifact is present. This study aims to solve this by creating a handful of reliable, binary, machine learning classifiers trained on both real and augmented data, that identifies the presence, or absence, of artifacts in a robust manner - meaning being able to detect artifacts on new individuals with data being abnormal compared to the training data due to the everchanging state of the brain - such that our solution can be applied as a means to solve the real-world epilepsy treatment gap. There are two reasons behind the binarization of the classifiers. Firstly, the classifiers should be able to predict the presence of multiple artifacts simultaneously, which normal multi-class classifiers cannot do. Secondly, since some artifacts happen very infrequently, we expect multiple binary classifiers to perform better than a single multi-class classifier. Thereby, this study together with BrainCapture's low-cost EEG caps, should form the basis for an interconnected solution for improving the quality and accessibility of on-site EEG recordings.

This study therefore aims to;

1. Investigate the effect of different data augmentation techniques on multiple binary EEG artifact classifiers performing in line with state-of-the-art in a validation set-up where predictions are made on data from unseen subjects in order to infer on the classifiers robustness.

For these research questions the following hypotheses were created:

1. The performance of classifiers will improve when augmented data is included in the train data. This increase in performance will depend on the type and amount of augmented data and whether a combination of augmentation techniques is used.

2. A committee consisting of the best classifiers (ensemble learning) within each artifact, predicting by majority vote, can obtain a better balanced accuracy than the best individual classifiers, within each artifact, on their own.

## 2 Methods and analyses

This section will describe the technical details necessary for setting up and conducting experiments related to the stated research questions and hypotheses. Descriptions of the data, its processing, technical details about models as well as ethical considerations related to the topic will be elaborated on to ensure transparency for the applied approach and results.

All pre-processing was run on a Windows 10 laptop with an IntelCore i7 processor and model fitting was run on a High Performance Computing (HPC) server at DTU GBAR configured to use Python 3.8. A list of python-packages and the employed version used to run the following experiments on the HPC server can be found in Appendix A (Section 6.1).

### 2.1 Ethical considerations

As EEGs are directly related to activity in the human brain, and the results of analyses on these might influence human lives, it is extremely important to consider benefits and potential flaws of creating Artificial Intelligence (AI) tools on such data. This assessment of ethical considerations related to EEG artifact detection will be based on the four outlined ethical principles in the Ethics Guidelines For Trustworthy AI by the European Commission [30] on the use of AI systems, which can be translated into seven requirements.

Firstly, in order to ensure human autonomy when interacting with the AI solution that is the outcome of this project, the system should only provide known recommendations to the diagnostician, from a formerly constrained list of artifacts, thus only aiding the whole process and ensuring

human-in-command (HIC) approach.

In addition, since the AI system must not cause any harm, the general safety must lie with the diagnostician and the system should optimally provide its level of certainty, since EEG measurements are very sensitive to the environment and must be the center-piece in the evaluation. However, in relation to the scope of this project, we will only focus on the performance (balanced accuracy, sensitivity) of the classifiers when predicting artifacts. The same principles of high accuracy and certainty of predictions apply for further analysis in classifying epilepsy, as a wrong conclusion can greatly affect the well-being of the patient, and must therefore be carefully monitored. Furthermore, it is important that the presence of artifacts, which the technistician will have to adjust, must not put the patient in uncomfortable situations.

The data set used in this project is the TUH EEG Artifact Corpus which is Open Source, given by the Neural Engineering Data Consortium (NEDC), where they have clarified that *"all data is approved for general release"* and that *"they must also be freely sharable"*[1]. This means we expect the individuals to be informed about the data collection process and that the data can not be used to discriminate against them. On the contrary, in an article published by R. W. Thatcher et al. (2016) [31] using a similar setup as this study, shows that they are able to show an inverse relationship between the magnitude of information flow and the IQ, meaning it is important to have a data protection policy to ensure privacy for the persons tested.

When dealing with medical data an important ethical consideration is to determine what should happen if unexpected malignant tendencies are accidentally found when analyzing the data - should the test subject be informed of this or not? In this study, the data set does not withhold enough information to provide actual diagnostics. However, if out-of-distribution issues were observed, it would be of relevance to contact TUH to provide the patient further medical tests if needed. In regards to the TUAR data set, one can discuss whether such open-sourced data might give rise to unethical behaviours such as earning money on something that is freely available, but in this study an assessment in line with NEDC that says that *"the lack of data transparency was limiting innovation"* [2] - meaning without open-source based innovation, one might say that Silicon Valley would not exist.

In continuation hereof it is worth discussing how this study will strive at publishing its results. The optimal final solution creating precise artifact classification, thereby resulting in cheap EEG measures (from BrainCapture's caps) with a low noise ratio might give rise to socioeconomic medical issues, since this could potentially lead to discrimination in favor of epilepsy treatment rather than other diseases. The availability of our research could therefore potentially be discussed, however, in line with the statement from NEDC, we believe that innovation is of extreme importance, and we aim at publishing this study as open-sourced with sufficiently described details to be fully reproducible.

As for creating an AI system that is fair, despite its many interpretations, the system does not contain information about the subjects themselves, though information about the diversity of the dataset is available, in regards to age and gender.

## 2.2 Description of data set

Since 2002, the NEDC has been recording clinical EEG data, now resulting in the largest publicly available corpus of EEG recordings (TUEG) in the world. The corpus consists of more than 30.000 observations, recorded at the Temple University Hospital in Philadelphia - NEDC continues to strive at publishing large quantities of high quality data, well-suited for machine learning [32].

The full EEG corpus is constituted from 16,986 session from 10,874 subjects. Throughout this project we will be looking at version 1.0.0 of the TUH Artifact (TUAR) data set, which is a subset of version 1.1.0 of TUEG. The TUAR data base consists of 310 EDF-files, all containing the standard electrode placements from a 10/20 configuration. The TUAR data base consist of observations from 213 subjects and is therefore divided into folders by a subject ID, which is then subdivided by an ID of the specific recording session. Within each session a text file describing

[1] NEDC, visited 05.03.2021: `https://www.isip.piconepress.com/projects/nedc/html/about_us.shtml`
[2] NEDC, visited 05.03.2021: `https://www.isip.piconepress.com/projects/nedc/html/about_us.shtml`

metadata (such as age and gender) about the subject and session, label-files - either LBL (event-based annotations of artifacts making use of all available artifact categories per channel) or TSE (term-based annotations making use of artifact classes by applying one label to all channels) - based on a start and stop time, as well as at least one EDF-file containing pruned EEG measurements, where the signals are cleaned of uninteresting parts. There are three types of EEGs in the 310 EDF-files in TUAR, determined by the common referencing of the electrodes; 290 uses the Average Reference (AR), 13 uses the Linked Ears Reference (LE) and 7 uses a modified version of AR (named ARa), which is recorded by excluding the auricular channels, i.e. the reference electrodes placed on the ears of the patient. This report will disregard data recorded using the two much less common types of EEG, namely, LE and ARa, as they do not provide much more data, but will require a more thorough pre-processing and interpolation of channels.

Of the 310 EDF-files, there are 259 sessions measured on 213 test subjects with label classes comprised by chewing ('chew'), electrode pop ('elpp'), eye movement ('eyem'), muscle contraction ('musc'), shivering ('shiv') and a label if none of these were present ('null'). This evidently means that some subjects were measured more than once; some even contain more than one EEG observation per session. This is seen from the distribution of the artifact events' labels (thereby excluding the 'null' class defining unidentified annotations) across the data in Figure 2, where the sum of labels exceed the amount of files. Furthermore, it is evident that the distribution of labels is unevenly distributed, which will be accounted for later on.

**Table 2:** Distribution of labels in original data

| | **Total** | chew | elpp | eyem | musc | shiv | **Sum of labels** |
|---|---|---|---|---|---|---|---|
| Files | 310 | 28 | 105 | 177 | 99 | 15 | 424 |

By analysis of the metadata it was found that the subjects have a mean age of 52.3 years with a standard deviation of 17.6 years. Though, it is seen from the distribution of the subjects' age in Figure 3 that the subjects' age ranged from approximately 15-100 years of age. Furthermore, the analysis revealed a minor distortion of the distribution of gender as 54% of the subjects were females and 46% were male. Within these two genders the age were more or less equally distributed, with the female age having a slightly higher mean and standard deviation. These results are presented in Table 3.

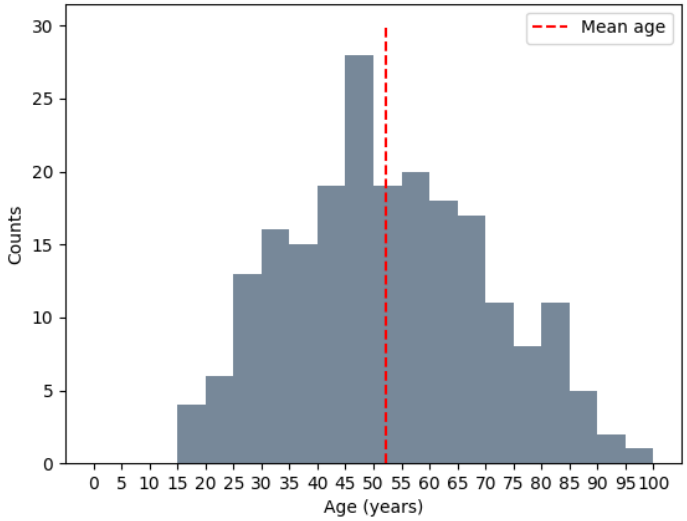



**Figure 3:** Distribution of age for all subjects in intervals of 5 years.

**Table 3:** Descriptive statistics of metadata, $\mu$ = mean, $\sigma$ = standard deviation

| | $\mu$ (years) | $\sigma$ (years) | **Proportion** (%) |
|---|---|---|---|
| All | 52.3 | 17.6 | 100 |
| Female | 53.3 | 18.7 | 54 |
| Male | 51.0 | 16.1 | 46 |

## 2.3 Preprocessing the data

The preprocessing of the data was carried out using modules and modified python scripts from David Enslev Nyrnberg's GitHub repository, DTU_DL_EEG [3], for which access was gained the 19th of Februrary 2021. This approach follows Makoto's Preprocessing Pipeline [33] combined with Early State Preprocessing (PREP) [34]. An intuitive visualization of this combined pipeline can be seen in Figure 4 (with filtering images taken from MathWorks[4] [35].

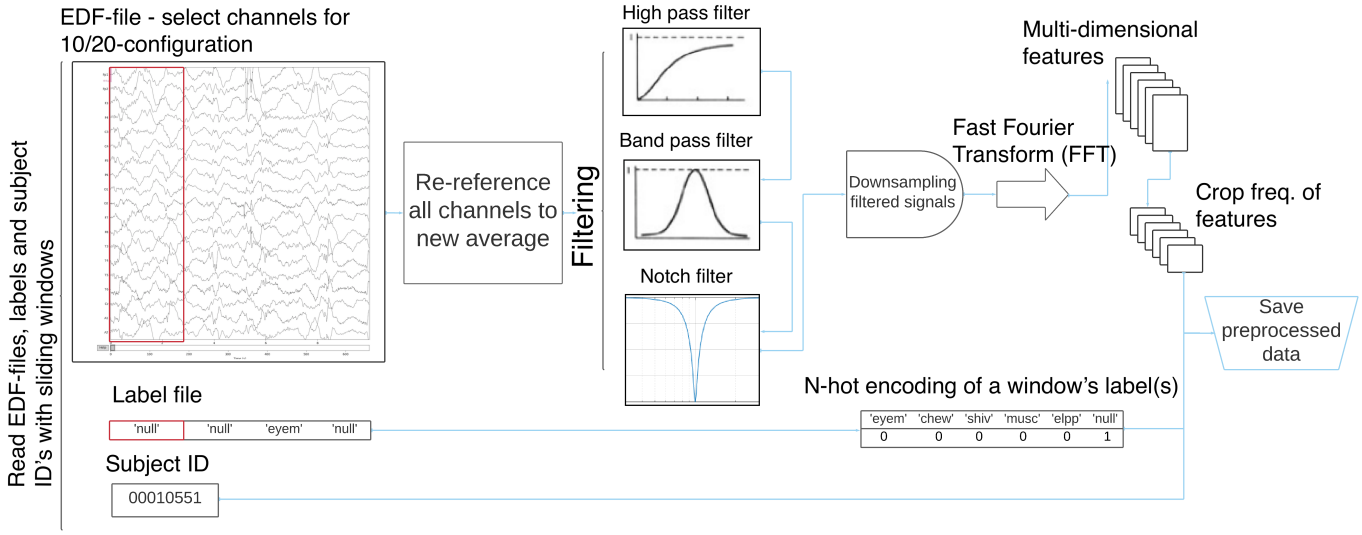



**Figure 4:** Visualization of the preprocessing pipeline.

The signals contained in each EDF-file were loaded, and a high-pass filter with cutoff set to 1 Hz and a band-pass filter ranging from 1 to 40 Hz were applied to the channels in order to remove unimportant frequencies. Furthermore, the US line-frequency of 60 Hz was filtered out using a notch filter, with a width of 5 Hz using "hamming" as the Finite Impulse Response filter (FIR) window design[5]. The EEG-cap setup was specified (standard 1005 position) after which a python module, `mne`, was used to pick the electrodes for the following 19-channel setup using a standard 10/20-configuration. The 19 channels used are; FP1, FP2, F3, F4, C3, C4, P3, P4, 01, 02, F7, F8, T3, T4, T5, T6, Cz and the aural electrodes, A1 and A2. See Figure 2 for specific electrode placements on the scalp. After this, the signals for each channel were re-referenced to average - meaning collecting the channels to a global value/reference, in order to eliminate a bias of spatial position of electrodes on the scalp.

The Sliding Windows technique was applied to the time-representation of the 19-channeled EEG signals to split the data into smaller parts, of increased quantity. The Sliding Windows technique segmented the signals into 1 second windows with a 75% temporal overlap. The temporal overlap should be understood as such that once the 1 second window of the signal is saved as an observation, the sliding window is moved 0.25-seconds to the right in the signal, before the next 1 second window is captured. Figure 5 should serve as a simple illustration of the concept, illustrated on a 1-dimensional discrete vector, whereas this study used the technique on a 19-dimensional array of continuous values. The blue colour indicates the 75% overlap between window 1 and 2.

[3] GitHub, visited 19.02.2021: `https://github.com/DavidEnslevNyrnberg/DTU_DL_EEG`

[4] MathWorks, visited 03.03.2021: `https://se.mathworks.com/help/control/ug/discretizing-a-notch-filter.html;jsessionid=304ae7706b3855cf24eced141996`

[5] MNE, visited 03.03.2021: `https://mne.tools/stable/generated/mne.filter.notch_filter.html`

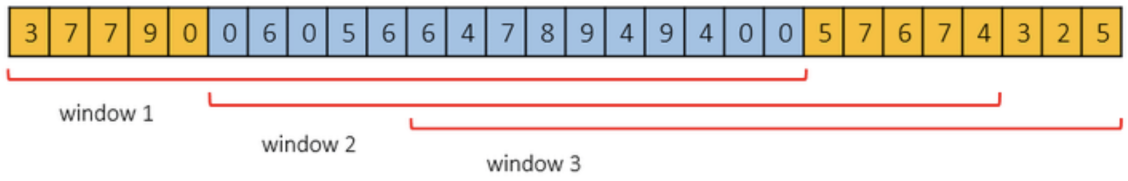



**Figure 5:** Simple visualization of the Sliding Windows technique, where each window indicates a 1-second observation, and the blue color indicates the 75% overlap between windows. [36]

After this, a Fast Fourier Transformation (FFT) was applied, thus downsampling the signals from all respective 19 channels at 150 Hz were done to i) account for aliasing meaning that the sample rate should be above the Nyquist rate determined from the frequency of the notch filter, (2 $\cdot 60 \pm 5$ Hz), and ii) standardize the sample rate of all subjects as different sampling frequencies were used when recording the EEGs. The FFT with overlap at 75% [37] - used to obtain higher resolutions in both the time and frequency axis - was applied to obtain frequency representations (power spectrums divided into frequency bins with a width of 1 Hz) of each of the 19 dimensions per 1 second window, which were cut off at 24 Hz for two reasons; i) most of the interesting waves (see introduction) are found in this lower range and ii) for comparison to the benchmark paper by Roy that uses this cutoff value. The 19 power spectrums belonging to each 1 second window found with this approach were saved in a tuple holding a 19-dimensional Pytorch tensor of power spectrums and a list of labels. Another tensor holding all window-label tensors were saved in a nested dictionary, holding the subject and session ID's as keys.

## 2.4 Description of processed data

The tensors obtained during the preprocessesing were saved as two numpy-pickle matrices; a feature- and a label-matrix respectively. The feature-matrix consisted of 1.345.171 rows, and 475 columns. The rows corresponded to the total number of windows in the entire dataset. During the pre-processing some of the windows in the feature-matrix took NaN-values. These observations were removed, thus a total of 1.344.862 windows remained. As previously mentioned, each of these 1-second windows consisted of 19 channels, and can as such be seen as 19-dimensional power spectrums. Henceforth these input features will be referred to as spectrograms, in line with the naming in the given GitHub repository. Each of the 19 dimensions were made up of 25 frequency bins, ranging from 0 Hz to 24 Hz, thus adding up to a total of 475 values per spectrogram. The label-matrix was of shape $(1.344.862, 6)$, describing the labels of every observed spectrogram in a N-hot encoded setup, since each window could have more than one label. Finally, a vector of length 1.344.862 was created, stating the subject-ID belonging to each spectrogram, which was needed for stratified cross-validation in the evaluation part of the pipeline.

As already seen in the non-processed data, there was an inequality in the distribution of annotated EEG artifacts. This becomes even more evident after labelling the 'null' label as well, since 'null' denotes all 1-second windows with absence of other labelled artifacts. Table 4 and Figure 6 show the number of observations of each label, as well as their proportion when comparing the number of observations to the total sum of windows. Since windows can have multiple artifacts present due to the overlap of windows in the preprocessing, the total proportion of artifacts will exceed 100%. It is evident that the 'null' label is strongly over-represented which will be handled in the "balancing of unbalanced data"-part of the overall pipeline in Figure 1. All in all 18764 windows contained multiple labels, of which 18752 held two different labels, all having 'null' as one of their labels. The remaining 12 multi-labeled windows held three labels, mainly consisting of the 'null', 'eyem' and a third class.

**Table 4:** Observations and percentages of each of the 6 labels. The sum of labels exceed the number of labeled windows, since each window (originally) can have more than one label.

| | eyem | chew | shiv | elpp | musc | null | **windows** | **artifacts** |
|---|---|---|---|---|---|---|---|---|
| Observations | 32585 | 11811 | 5986 | 11945 | 18526 | 1282785 | 1344862 | 1363638 |
| Distribution | 2.43% | 0.88% | 0.45% | 0.89% | 1.38% | 95.4% | 100% | 101.4% |

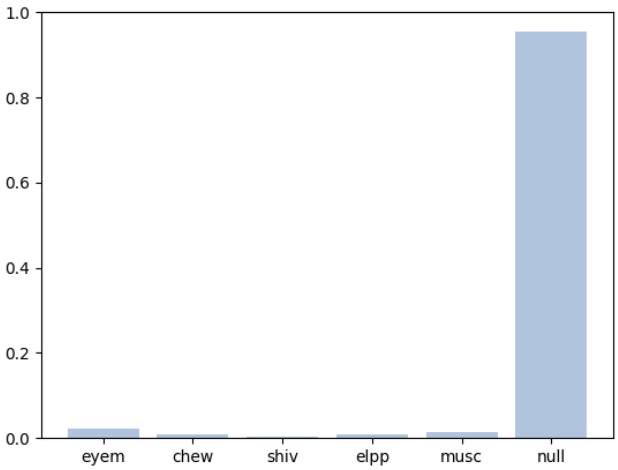



**Figure 6:** Distribution of labels in the preprocessed data

### 2.4.1 Binarization of the multi-label target

As previously mentioned, the observations originally had a multi-label structure (i.e. multiple labels can be present for a single observation), resembling the real world scenario in which multiple artifacts can be present at the same time. For this reason the general machine learning models will not be multi-class, but rather binary classifiers for each artifact independent of the others, resulting in a binarization of the multi-label target vector. A binary classifier will have the classes of present (1) and absent (0) for all artifacts, however, this does not mean that the class of absent is only the label 'null' - it simply signifies the 'absence' of the artifact.

In the cases where 3 artifacts were present at the same time, the whole observation was removed and not considered part of the data set. This choice results in removing 12 observations from the total of 1.344.862 observations. In all of the 18752 observations with 2 artifacts, 'null' was one of the artifacts. Thus, in these cases 'null' was set to 0 (absent). This is to avoid deleting the already low amount of artifacts, and one might further argue that 'null' is often present in some form since most artifacts are rather rapid. After these exclusions, the dataset consists of 1.344.850 observations from a total of 201 subjects.

## 2.5 Data modeling

The following section will explain the data modeling pipeline used in our experiments, as seen below in Figure 7.

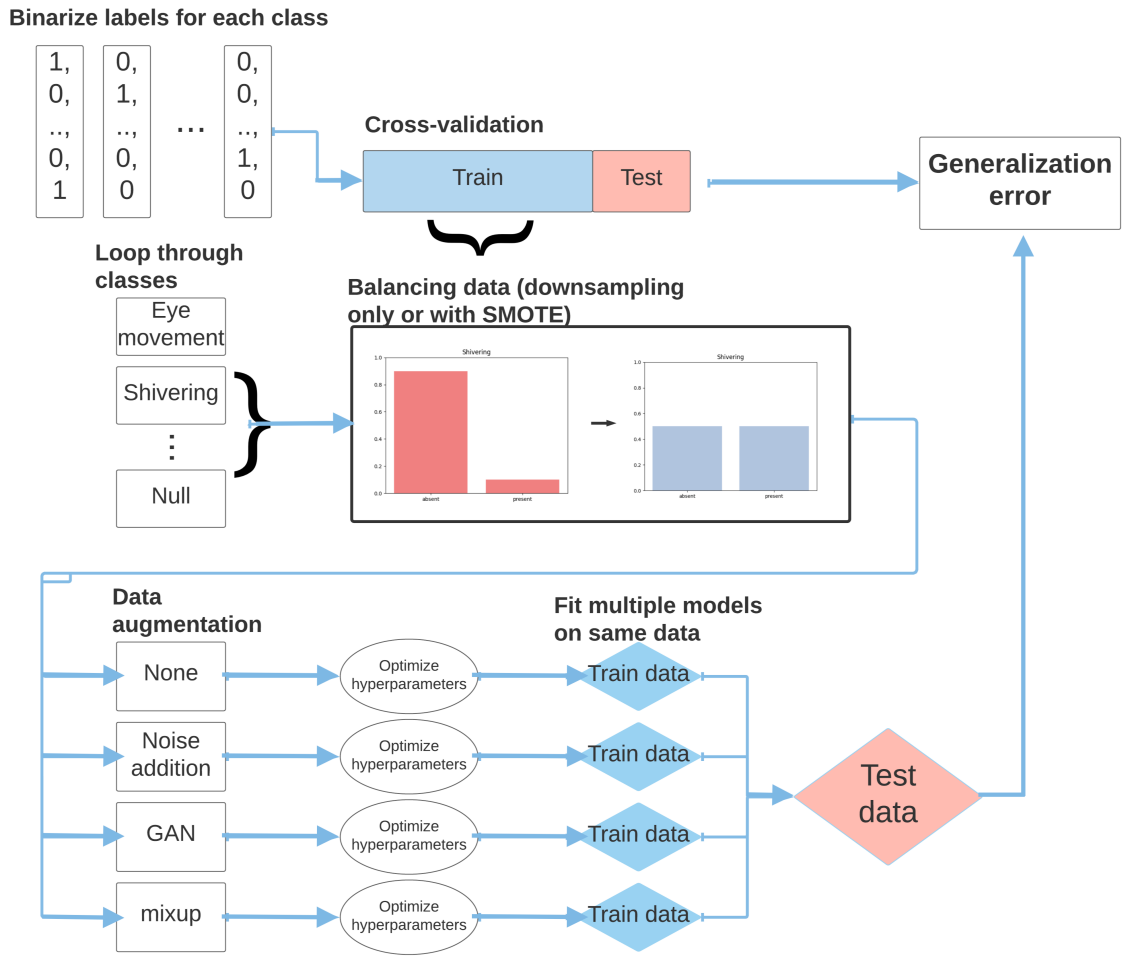



**Figure 7:** Visualization of the data modeling pipeline.

The purpose of this pipeline is to train machine learning models to predict whether or not an artifact is present or not. In general the steps of this pipeline are constructed of; binarizing the training set, balancing the data to train the models optimally, augmenting the data to achieve more robust classifiers and then fitting the models properly to the data. The following sections will explain which techniques are used to perform these steps optimally in this paper.

### 2.5.1 Balancing the data set

As seen in the description of processed data (Figure 6), the data set is highly unbalanced, with the size of the majority class, 'null', being between 20-120 times larger than the other artifacts in the data set. This is a problem since most machine learning algorithms need to train on a balanced data set to perform optimally without biases towards certain classes [6], which makes the general aim of this part of the pipeline, to achieve equal amounts of observations between classes (absent/present). There are several ways to address this problem of imbalance. In this study, the following techniques are used: i) under-sampling of the majority class to the same level as the minority class and ii) a combination of under- and over-sampling, by under-sampling the majority class to a level in which the minority class is over-sampled to, based on a given ratio. In this study this is done by using the SMOTE technique for over-sampling.

When under-sampling observations in the majority class, observations are randomly picked without replacement using the `RandomUnderSampler()` package in the imbalanced-learn library [38], which generally results in under-sampling the absent class of the binary classifiers, except when predicting

[6] PyPi, visited 09.02.2021: `https://pypi.org/project/imbalanced-learn/`

'null', as present is the majority class in this case. Figure 8 illustrates (using principle components) the under-sampling technique on 1000 randomly sampled data points from the unbalanced data set for the artifact 'null' after which under-sampling was carried out to achieve equal number of data points in each class. The principal components basis in the figure is computed from the unbalanced data set.

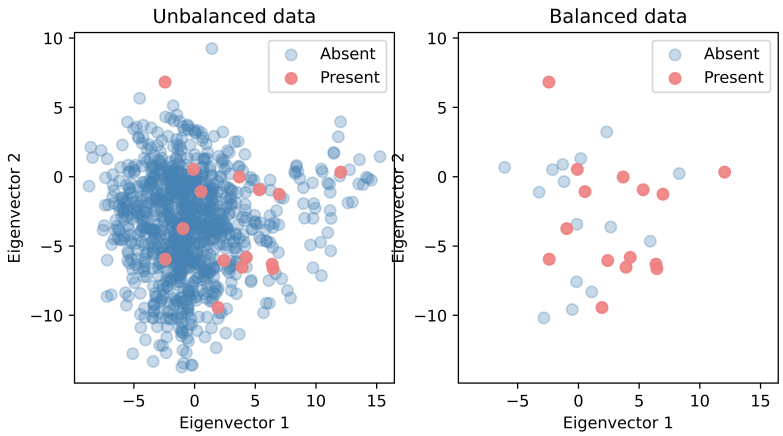



**Figure 8:** Under-sampling of the data set

As seen in Figure 8 the size of balanced data has been significantly reduced compared to the unbalanced data, as the proportions are heavily skewed. This increases the risk of losing useful information about the majority class, which is part of the reason for the second approach. In Figure 9 the over-sampling technique is illustrated by first picking 600 observation at random from the unbalanced 'musc' artifact. To balance the data it is first necessary to define a ratio of which to over-sample the minority class to. Then, the majority class is under-sampled as previously, but to the expected size of the over-sampled minority class. Lastly the data is balanced by oversampling the minority class using SMOTE. This way, the loss of information is expected to be less. In the figure, the principal components are still computed based on the unbalanced data set and the ratio is set to 10 for visual purposes of the approach.

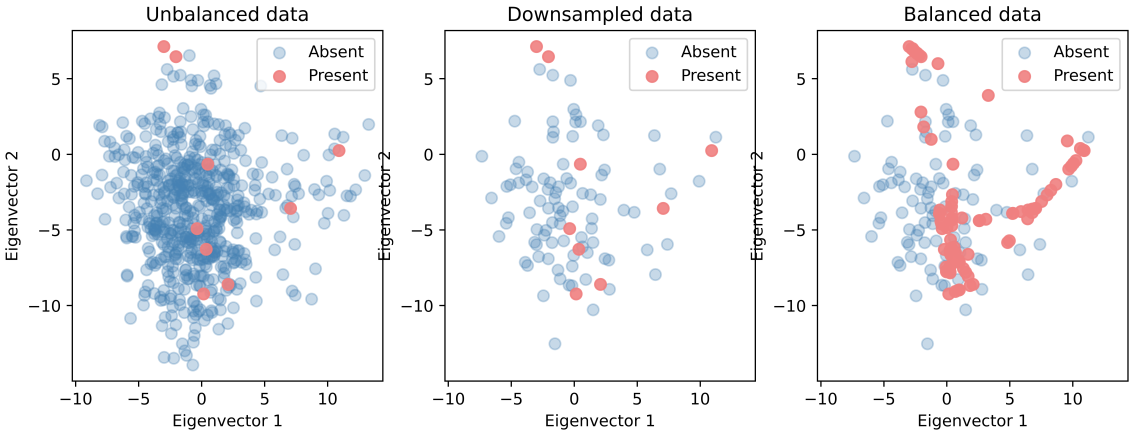



**Figure 9:** Under- and over-sampling of the data set using SMOTE

One might notice that the minority samples in the balanced data in Figure 9 look to follow some kind of linear pattern, and this is due to the over-sampling technique SMOTE - a method used for oversampling underrepresented classes in a classification problem. The technique takes each observation in the underrepresented class, and generates a new observation, at a random point, directly in between the observation and one of its $K$ nearest neighbours. The combination of SMOTE with random under-sampling of the majority class was suggested in the original SMOTE paper [39]. Multiple variations of the SMOTE-technique have been created since it was first introduced, however this study will only be focusing on the original SMOTE variation. The pattern in Figure 9 of the balanced data is therefore due to linear interpolations between a sample

and one of its nearest neighbours. In general, the SMOTE algorithm consisted of three steps when given an oversampling ratio:

1. Selecting minority class instances at random in a set $\mathbf{x} \in \mathbf{X}_{set}$ and finding their respective $K$-nearest neighbours based on the euclidean distance.

2. Based on a sampling rate $N$ (which is the amount of SMOTE data for each instance $\mathbf{x}$), randomly selecting $N$ neighbours $\mathbf{x}_{neigh,\, i \in N}$ out of the $K$-nearest neighbours for each $\mathbf{x}$.

3. Then for each $\mathbf{x}$ and selected neighbour $\mathbf{x}_{neigh,\, i}$ of $\mathbf{x}$, a new synthetic instance $\tilde{\mathbf{x}}$ is generated as a convex combination of the two by,

$$\tilde{\mathbf{x}} = \mathbf{x} + rand(0,\ 1) \cdot |\mathbf{x} - \mathbf{x}_{neigh,\, i}|$$

   where $rand(0,\ 1)$ is a random number between 0 and 1 chosen at random for each of the attributes of an instance[7] [39].

To conclude, these procedures are performed before any model is fitted to the data with the aim that the classifiers achieve a more fair decision boundary.

### 2.5.2 Data augmentation

A major concept in machine learning is the robustness of a model, as it is important that a model is able to predict on new, independent data points. In the case of EEG-measurements, this concept is paramount, as a model must be able to take what it has learned from the data set, and be able to apply it to new patients once the models are trained. In order for a model to be robust, it must be trained on numerous observations, spanning a wide range of the feature space. However, as seen previously, an unfortunate outcome of the balancing of a data set, is the deletion of a vast amount of observations of the majority class. In order to make up for this decrease in the size of the training set, as well as create more robust observations in general, this part of the pipeline experiments with the use of data augmentation, applying it to both classes, i.e. both present and absent.

Further motivation for data augmentation comes from the Vicinal Risk Minimization (VRM) as opposed to Empirical Risk Minimization (ERM). As in, instead of solely fitting a model on the training data (the empirical risk), which typically results in a model memorizing the data instead of generalizing it, one could augment vicinal (neighbouring) examples around the data to represent the training data distribution in a more general way. As shown in the introduction, this approach often leads to an improved generalization.

As previously mentioned, the data augmentation methods used in this step of the pipeline are MixUp, GAN and Noise Addition. In this context one could imply that SMOTE also acts as a data augmentation technique, which is true, however this paper only investigates SMOTE as an over-sampling technique of a minority class and will therefore not be referred to as data augmentation.

**MixUp**
The motivation for choosing MixUp as an augmentation technique is that rather than following the VRM priciple of augmenting neighbouring observation (i.e. that share the same class) one could instead model a vicinal distribution across classes.

The idea of the MixUp-method is thus to construct a new datapoint by blending two randomly drawn observations from the training set, of (potentially) different labels, together as a linear combination. Each observation is weighted by $\lambda$ and $1 - \lambda$, respectively, where $\lambda \in [0, 1]$, is drawn from a Beta-distribution based on an $\alpha$-value recommended to be 0.4 [22]. This creates a higher probability of $\lambda$ being closer to 0 or 1, than anything in between. The label of the augmented datapoint is constructed correspondingly, as a linear combination of two one-hot encoded labels, weighted by $\lambda$ and $1 - \lambda$, respectively. Mathematically, this method is simply written as:

$$\tilde{\mathbf{x}} = \lambda \mathbf{x}_i + (1 - \lambda)\mathbf{x}_j,$$

$$\tilde{\mathbf{y}} = \lambda \mathbf{y}_i + (1 - \lambda)\mathbf{y}_j,$$

[7] GeeksforGeeks, visited 11.02.2021:https://www.geeksforgeeks.org/ml-handling-imbalanced-data-with-smote-and-near-miss-algorithm-in-python/

A simple example of how the method works is shown below in Figure 10.

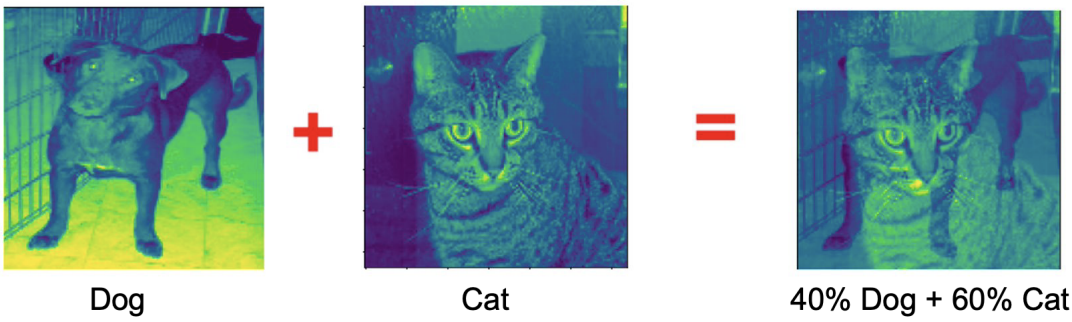



**Figure 10:** General example of augmentation using MixUp [a]

[a]MixUp, visited 08.06.2021:`https://towardsdatascience.com/2-reasons-to-use-mixup-when-training-yor-deep-learning-models-58728f15c559`

Since the augmented data is created by interpolating observations from potentially different classes, a linearity is introduced between the training-examples. This provides smoother decision boundaries and a more robust classifier, as a broader range of possible observations is introduced [22].

Unfortunately, sklearn-models are unable to handle the now fractioned structure of the one-hot encoded labels, meaning a workaround is required. This is a problem not formerly encountered in state-of-the-art, however, in this paper, a workaround will be applied which randomly draws either label by probability equal to their respective $\lambda$ and $1 - \lambda$-values. Thus, the augmented data point will be a linear combination of two randomly selected data points, but the corresponding label will be one of either class, drawn randomly by their probabilities.

In this paper, the alpha-parameter for the Beta distribution was set to 0.4 as recommended.

**GAN**

The Generative Adversarial Network (GAN), is a system consisting of two neural networks, $G$ and $D$ - a generative, and a discriminative neural network respectively. In this setup, G generates new data by taking random noise as input, and generates an output which could be mistaken as an observation taken from the original training set. Meanwhile, D takes as input both original and generated observations, and tries to distinguish the two, by classifying which observations are original, and which have been generated. Throughout this process, both are updated dynamically, with the discriminative network continuously improving its ability to distinguish real from fake, and the generative network continuously improving its ability to fool the discriminative network. In turn, D creates more accurate generated depictions of the original data over time [25]. The point of GAN as a data augmentation method is then to use the trained generative neural network to generate observations, and use these as training data. A simple sketch of the architecture behind GANs can be seen in Figure 11:

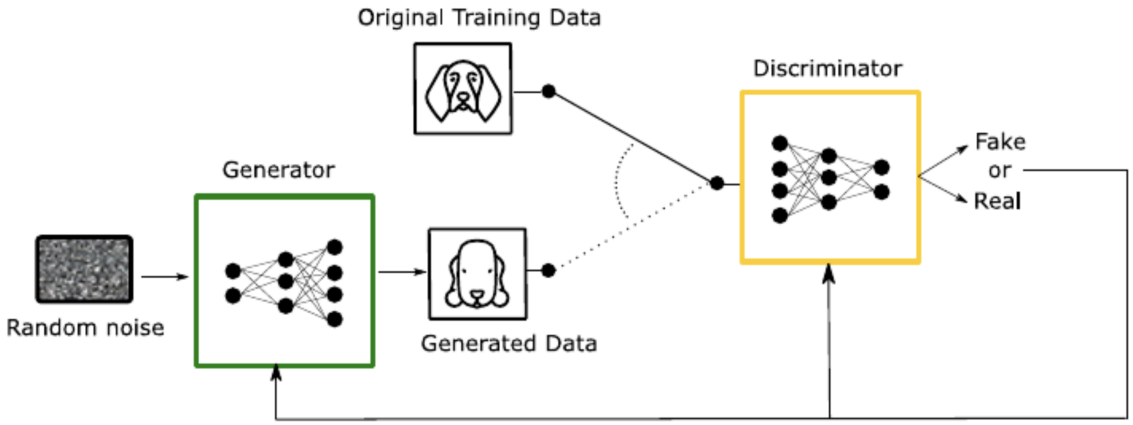



**Figure 11:** Simplification of the architecture behind GANs[40]

In this paper the hyper-parameters for GAN were set to default based on a notebook where they perform GAN on the grey-scale MNIST data set[8], this should to some extent be comparable as MNIST observations are of size 28x28, while this data set is of size 19x25.

**Noise Addition**
The idea behind using Noise Addition as an augmentation technique is to improve the models robustness by blurring the original data, thus forcing the model weights to account for possible noise when recording the data. As is clearly indicated by its name the noise addition technique adds randomly drawn values from a Gaussian distribution to the original data and can thus be applied in both the temporal and frequential domain. This report solely focuses on the temporal aspect of Noise Addition. In connection with EEG data, noise values are drawn from a multivariate Gaussian distribution with zero mean and a covariance matrix determined from the covariance between EEG channels within a specified time-window (1 second, as stated in the preprocessing-section). An advantage of using the covariance between channels rather than a uniform covariance is that it accounts for co-occurring neural activity measured on different channels. In Noise Addition, the importance of covariance between channels is up for optimization which is commonly done by scaling the matrix with a scalar.

By randomly drawing points from the previously mentioned distribution, the added noise signal within a time-window will be more or less equally distributed across the frequency spectrum. This is known as white noise. However, since white noise is equally distributed across all frequency content one could imagine that focusing the added noise to frequency ranges in which neural activity occur would be more optimal for a classifying model as it would still provide for robustness, just mainly do it in the relevant frequency ranges which are used for modelling the problem. This idea is known as colored noise and noise values of this kind are found by applying a low-frequent cut-off filter on the white noise signals drawn from the previously mentioned multivariate Gaussian. The cut-off frequency as well as the covariance scaling factor are up for optimization in Noise Addition.

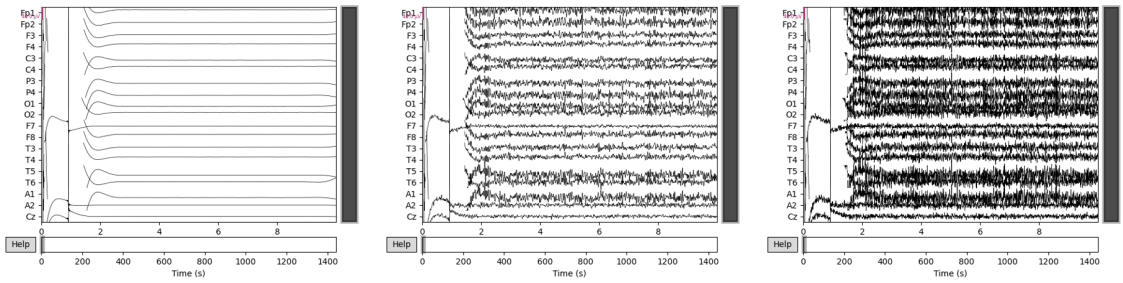



**Figure 12:** Vizualisation of clean signal (left), with added colored noise - cut-off 30 Hz - (middle) and added white noise (right).

In this paper the covariance scaling factor was set to 1 for both white noise and colored noise with colored noise using a cut-off frequency of 30 Hz.

### 2.5.3 Model fitting

As one of the main goals of this paper is to investigate the overall effect of data augmentation on EEG data, it is of importance to report the effect on a wide range of classifiers. This paper will report the obtained effect when training six different supervised learning algorithms, from within six different supervised learning paradigms, as well as a baseline classifier. The classifiers used will be, Logistic Regression (LR), Gaussian Naïve-Bayes (GNB), Random Forest (RF), Linear Discriminant Analysis (LDA), Multilayer Perceptron (MLP) as well as Stochastic Gradient Descent (SGD).

This paper presumes the reader has prior knowledge of LR, GNB, RF as well as SGD. The remaining two, as well as the baseline, however, will now be described in further detail.

**Baseline**
In order to have a comparison ground for the models, a baseline model will be implemented. Usually a baseline model only predicts the majority class, however, as previously touched upon, our

[8]GAN, visited 17.02.2021:`https://github.com/mchablani/deep-learning/blob/master/gan_mnist/Intro_to_GANs_Exercises.ipynb`

metrics are going to evaluate each of the binary classes, and therefore it is desired that baseline in some degree touches upon all the measures in the confusion matrix. Based on this idea and to keep the baseline model simple, its predictions are a permutation of the labels in the test set.

**Multilayer Perceptron**
While the term Multilayer Perceptron might sound unfamiliar to the reader, they will be appeased to hear that it is in fact all too familiar. The Multilayer Perceptron is an ANN, consisting of an input layer, an output layer and hidden layers like any other ANN. The MLP is a fully connected feedforward network, trained using backprobagation, in order to assign weights to the features, but differ from a convolutional neural network (CNN) by not using feature enhancing like convolutional and pooling layers in the network architecture.

**Linear Discrimant Analysis**
The idea behind LDA, is to find a subspace of the feature-space, in which the classes of the original feature-space are separated as much as possible, by finding a linear direction which maximizes the distance between the means of each class, while minimizing the variance within each class. Once the dimensionality has been reduced, predictions are derived through Bayes' rule.

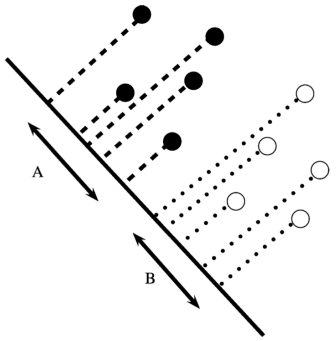



**Figure 13:** The intuition behind LDA [41]

While the idea of dimensionality reduction has been seen before in methods such as Principle Component Analysis (PCA), the difference lies in the fact that PCA, an unsupervised method, finds the directions with maximal variance on the entire data set, while LDA takes the classes into account, trying to separate them as much as possible. The first step is to compute the $d$-dimensional mean vectors, $\mathbf{m}_i$, for each class of the data set. The vector consists of the mean of every feature, for every class. The scatter-matrices are then calculated. First, the scatter-matrix of every class is computed

$$\mathbf{S}_i = \sum_{\mathbf{x} \in D_i}^{n} (\mathbf{x} - \mathbf{m}_i)(\mathbf{x} - \mathbf{m}_i)^T$$

The within-class- and between-class scatter matrices, $S_W$ and $S_B$, are then calculated as

$$\mathbf{S_W} = \sum_{i=1}^{c} \mathbf{S}_i, \qquad \mathbf{S}_B = \sum_{i=1}^{c} N_i (\mathbf{m}_i - \mathbf{m})(\mathbf{m}_i - \mathbf{m})^T$$

Here, $\mathbf{m}$ is the overall mean, and $N_i$ is the size of each class.

Next, the eigenvalues and eigenvectors are found by solving the matrix $\mathbf{S_W}^{-1}\mathbf{S_B}$. They are then sorted by decreasing eigenvalues, and the $k$ most informative eigenpairs are stacked to form a $d \times k$-dimensional eigenvector matrix, $\mathbf{W}$.

In the last step of the dimensionality reduction, our samples, $\mathbf{X}$, are transformed onto the new subspace, by the dot-product between $\mathbf{X}$ and $\mathbf{W}$.[9]

$$\tilde{\mathbf{X}} = \mathbf{X}\mathbf{W}$$

As mentioned, the classification is derived from Bayes rule, on the transformed dataset,

[9]LDA, visited 18.05.2021:`https://sebastianraschka.com/Articles/2014_python_lda.html`

$$P(y = k|x) = \frac{P(\mathbf{x}|y = k)P(y = k)}{P(\mathbf{x})}$$

In LDA, the likelihood, $P(\mathbf{x}|y)$, is given as a multivariate Gaussian.

$$P(\mathbf{x} \mid y = k) = \frac{1}{(2\pi)^{d/2} \left|\Sigma_k\right|^{1/2}} \exp\left(-\frac{1}{2}\left(\mathbf{x} - \mu_k\right)^t \Sigma_k^{-1} \left(x - \mu_k\right)\right)$$

LDA assumes that the covariance-matrices, are equal for all classes, $\Sigma_k = \Sigma$. This assumption does not necessarily hold true with regards to EEG artifacts, however, it is just an assumption. With this assumption, we get the following log-posterior.[10]

$$\log P(y = k \mid \mathbf{x}) = -\frac{1}{2}\left(\mathbf{x} - \mu_k\right)^t \Sigma^{-1} \left(\mathbf{x} - \mu_k\right) + \log P(y = k)$$

Which can now be used to make classifications on new observations.

**Parameter Optimization**

The classifiers all have parameters which should be optimized. In the pipeline, this is handled by using the Hyperopt-library [42], which uses a form of Bayesian Optimization to find the maximum of an objective function on a validation set of the dataset. In order to iterate on the search space when searching for the maximum, a Hyperparameter Optimization Algorithm (HOA) is needed. The default HOA of HyperOpt is the Tree of Parzen Estimator (TPE), and thus the one used in this study. TPE is used to calculate the expected improvement, and works by assigning a threshold to the score of the objective function, then creating two probability distributions. One distribution is made from the scores of the previously queried hyperparameters above the threshold, $g(x)$, the other from the scores below, $l(x)$. The expected improvement is then calculated as the ratio, $\frac{g(x)}{l(x)}$. The next query is then the hyperparameters which maximize the expected improvement, based on the probability distributions.[11]

The hyperparameters optimized for the applied classifiers and their boundaries are seen in Appendix B (Section 6.2).

### 2.5.4 Metrics

Previous sections investigated how the training data can be optimized in order to make the classifiers less biased, however it is worth noticing that the test data will always be left untouched throughout these processes. The reason for not balancing the test set is to evaluate the performance of the models on data that is as close to the real distribution of the data as possible. Consequently, this paper will focus more on other metrics than accuracy, since accuracy will measure more in favor of the majority class and not imply an overall estimation of the performance of the classes. In this report unbalanced test data will be regarded in two ways: i) when measuring the performance of the classifiers in order to optimize them to best fit the training data, and ii) is to have a comparison standpoint between the different augmentation techniques and classifiers. In the case of the first point, the unbalanced test set will be referred to as a validation set, whereas the latter will be referred to as the test set. These two approaches will have two different measures, however the measures originate from the same idea.

As previously mentioned, failing to classify an artifact is of less interest than classifying it too frequently, as it is better for the technician to try to fix an artifact, that in reality might not be present, than the opposite scenario where the artifact passes by. As seen in Figure 14 such a measure is often referenced to as sensitivity (also called recall). The inside of the circle to the left is known as the true positive predictions, TP, the inside of the circle to the right are the false positive predictions, FP, the outside to the left are the false negative predictions, FN, whereas the outside to the right, are the true negative predictions, TN.

[10] LDA, visited 18.05.2021:https://scikit-learn.org/stable/modules/lda_qda.html#mathematical-formulation-of-the-lda-and-qda-classifiers

[11] HyperOpt, visited 13.03.2021:https://towardsdatascience.com/a-conceptual-explanation-of-bayesian-model-based-hyperparameter-optimization-for-machine-learning-b8172278050f

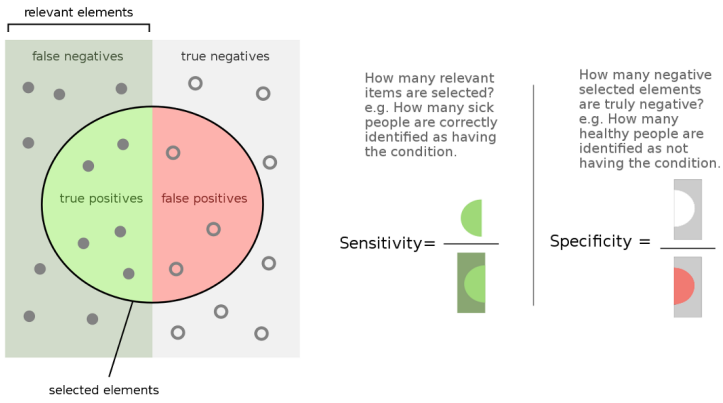



**Figure 14:** Sensitivity and specificity [a]

[a]Metrics, visited 18.05.2021:https://en.wikipedia.org/wiki/File:Sensitivity_and_specificity_1.01.svg

Therefore, the measure of sensitivity is how often a classifier predicts the artifact correctly, out of how many times it was present - referring to how 'sensitive' it is for the positive (present) class.

$$sensitivity = \frac{true\,positives}{true\,positives + false\,negatives}$$

However, sensitivity can be biased in a sense, namely if the artifact only predicts on the artifact being present, the score will be equal to 1. Therefore, sensitivity in combination with another metric could help flip this bias, which is the case with the metrics of the $F_{beta}$ family and balanced accuracy. $F_{beta}$[12] is in this project the metric used for validating a classifier for optimization (approach i) and it combines sensitivity with the metric known as precision:

$$precision = \frac{true\,positives}{true\,positives + false\,positives}$$

Precision reflects how often the classifier predicts an artifact correctly out of how often it predicts the artifact. The score seems to act as a measure of how 'precise' a classifier really is, and could therefore remove the previously mentioned bias of sensitivity. In relation to $F_{beta}$, this score combines sensitivity and precision as the harmonic mean of fractions by,

$$F_\beta = \frac{(1+\beta^2)\cdot precision \cdot sensitivity}{\beta^2 \cdot precision + sensitivity}$$

where the value of the parameter $\beta$ determines the balance between sensitivity and precision. In this study the parameter is set to be $\beta = 2$ for which reason the common name 'F2 score' is used. As a consequence, sensitivity is weighted higher than precision, which is valued in the use-case wherefore it is used for model optimization.

However, the F2 score is rather hard to comprehend and is part of the reason why another metric was used for the test set (approach ii) - namely the balanced accuracy. Balanced accuracy combines sensitivity with the metric known as specificity, which is how often a classifier predicts absent correctly out of how often it predicts absent. Intuitively, this is how sensitive a classifier is on the absent class as seen in Figure 14.

$$specificity = \frac{true\,negatives}{true\,negatives + false\,positives}$$

Balanced accuracy [13] is given as the average of the two, intuitively, the average of how often the present and the absent class is correctly classified, respectively,

$$balanced\,accuracy = \frac{sensitivity + specificity}{2}$$

Therefore by using both balanced accuracy and sensitivity the performance of the classifiers can be compared and evaluated independently of the imbalance of the test set, and relative to the

[12]F-score, visited 18.05.2021:https://machinelearningmastery.com/fbeta-measure-for-machine-learning/

[13]BA, visited 18.05.2021:https://statisticaloddsandends.wordpress.com/2020/01/23/what-is-balanced-accuracy/

use-case of the report. In general, F2 and balanced accuracy are both 1 when there are no Type I and Type II errors, due to the fact that whenever precision and specificity are 1 there are no false positives (Type I) and for sensitivity there are no false negatives (Type II).

### 2.5.5 *p*-value

When statistically testing whether there is a difference between the performance of the classifiers, this study will follow the recommended procedures for comparing classifiers in a $K$-fold cross-validation set-up, presented by Dietterich (1998) [43]. The $p$-values are computed based on a paired t-test of the error rates of the two compared models across the folds. In order to maintain coherency, the error rates are computed as $1 - balanced\ accuracy$ rather than $1 - accuracy$, as was the case in Dietterich. However, as noted in the paper, this approach elevates the probability of making Type I errors, thus detecting a difference between classifiers, when in reality there is none. Furthermore, since $p$-values will be calculated across each augmentation method and artifact, multiple comparisons are made, thus increasing the family-wise error rate - again increasing Type I errors. The increasing risk of Type I errors will therefore be accounted for by adjusting the $p$-values using the Benjamini-Hochberg procedure, where the significance level was set to 5%.

## 2.6 Experiments

The following experiments were carried out through Thinlinc on the GBAR HPC-servers at DTU.

The training of the models followed a general format shown in Figure 15, where the models were evaluated using stratified 5-fold cross validation in order to get a more accurate estimate of the generalization error. Rather than randomly splitting the data, it was stratified based on the 201 individuals - this is to i) ensure predictions are made on unseen data to test robustness and ii) to avoid variance inflation between training and test data, in which the classifiers will hold a bias on some of the variation across individuals. Furthermore, when applying the classifiers in a real-world scenario, they are expected to predict on new unobserved individuals. Since it is 5-fold the ratios in each fold were $\frac{4}{5}$ training and $\frac{1}{5}$ testing. The next step was to optimize the hyperparameters of the classifiers. This optimization was further based on a single split of the training data into a training subset and a validation set of same ratios as before. The training subset was then balanced and HyperOpt was applied to find the optimal hyperparameters of the classifiers. The hyperparameters of the classifiers were optimized by maximizing the F2-score, on the validation split of the training data made from subject ID's through 25 queries of HyperOpt. The classifiers GNB and baseline were disregarded from optimization as they had no parameters to optimize. After optimization the training set was balanced and then evaluated on the test set.

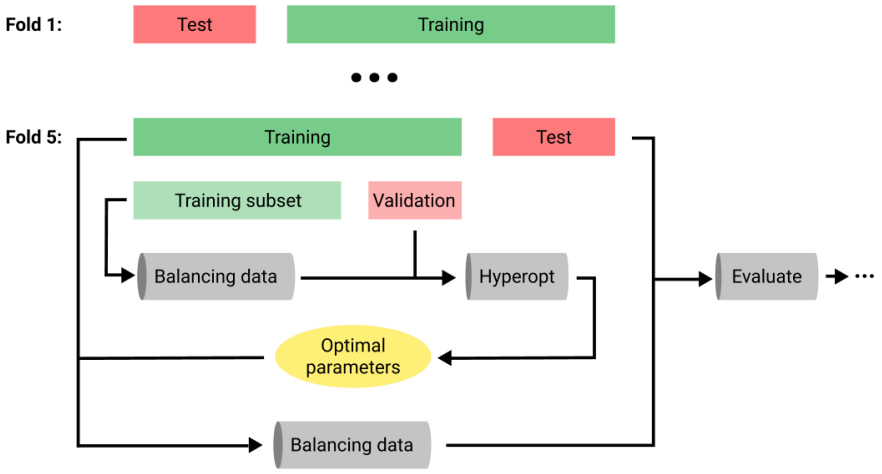



**Figure 15:** 5 fold cross-validation using HyperOpt

In general, the results for each fold were averaged in order to achieve a more general approximation of the estimates.

### 2.6.1 Control experiment - no augmentation

In order to test the effect of data augmentation, a control experiment was needed as a reference for comparison. The general structure of this experiment, was therefore to reproduce some of the benchmarks presented by Roy (2019) [20]. It is worth noting, however, that the reproducibility presented in the paper by Roy is lacking, and therefore the results were expected to differ.

In the control experiment the format followed Figure 15, where the balancing of the data set was done using random under-sampling on the majority class, in each of the binary classifiers. The classifiers were then fitted on the data, and the performance was evaluated. The results were presented as barplots and tables, showing the performance of each of the binary classifiers, measured in both sensitivity and balanced accuracy. Confusion matrices of the best classifier found for each artifact, as well as the average spectrograms for each artifact, independently normalized, were reported as well.

### 2.6.2 Performance improvement experiment

The goal with the performance improvement experiment was to improve the performance found in the control experiment, for which reason two approaches were determined: One experiment i) would compare the performance of the classifiers at various ratios of the various augmentation techniques, while another ii) would predict using an ensemble of classifiers selected from their performance in the previous experiments.

Before conducting either of the experiments, a pilot experiment was performed in order to investigate the model performance when the data was balanced using SMOTE. The pilot experiment investigated SMOTE at ratios 0, 0.5, 1, 1.5, 2, meaning upsampling of the minority class to twice its original size when using a SMOTE ratio of 1. Since the results, seen in Appendix E (Section 6.5), in general indicated that the ratios performed equally well, the augmentation experiment was further split into two by investigating a SMOTE ratio of both 0 and 1. The ratio of 0 acted as a reference, while the ratio of 1 was chosen deliberately, as a ratio with a value being too high might influence the potential gain of the remaining data augmentation techniques (MixUp, GAN and Noise Addition) negatively.

**Augmentation**
The augmentation experiment served as an extension of the control experiment, meaning the general structure of the experiment followed the same pipeline. The balancing of the data set was done using SMOTE ratio 0 and 1, following the pilot experiment. Then, experiments where the classifiers were trained using each of the four augmentation methods - namely MixUp, GAN and Noise Addition of white noise and colored noise at ratios 0.5, 1, 1.5, 2 - were carried out and evaluated.

Results were presented as barplots showing the difference in performance between each of the binary augmentation classifiers, and their corresponding classifier from the control experiment, as well as another barplot, and its corresponding table, showing the performance of the best performing classifiers, for each augmentation technique as well as the control classifier, for each artifact. Finally, $p$-values found from a students t-test, between the best augmentation classifiers, and the best control classifiers, for each artifact from the control experiment, as well as the average spectrograms for each artifact, using every type of augmentation technique, normalized within artifacts.

**Ensemble**
The second of the performance improvement experiments was the ensemble learning experiment. Ensemble learners have multiple advantages to individual classifiers, since e.g. individual models hold a bias of learning in their predictions. When training using an ensemble of classifiers, classifying by majority vote, according to the bias-variance trade-off, this bias will be reduced as the biases of the learners become generalized in the ensemble classifier. Furthermore, if the models chosen for the ensemble have high diversity in their predictions, the majority vote should enhance the overall classification performance [44]. Therefore, the ensemble learners used for the experiments, were created as follows: For each artifact, all previous classifiers, with and without augmentation, were saved and sorted in accordance to highest balanced accuracy. The correlation between the predictions of the 20 best classifiers within each artifact were then calculated and reported, and

the 5 classifiers having the lowest mean correlation with all the remaining classifiers, were chosen as the members of the ensemble learner. This resulted in a total of six ensemble learners - one on each artifact - predicting using the best and least correlated models within their respective artifact-label. It is important to note that one or more of the best models on each artifact from the control experiment could potentially be selected for each artifact's ensemble learner since the ideas of reducing bias and possibly improving the performance compared to the best control model still stand.

Results were presented as the correlation matrix of the 20 best models for each artifact and confusion matrices of the ensemble learner's predictions. Barplots of the performance measured both in balanced accuracy and sensitivity compared to the best model for each artifact label from the control experiment were reported as well.

# 3 Results & Discussion

Since multiple experiments were conducted for this report (i.e. control, improvement and ensemble), the results and discussion sections will be done pairwise for each experiment. As a comment, all performances in the following sections are the averages found through 5-fold-CV, for either balanced accuracy or sensitivity with both having their confidence measure as the average of the standard deviation (SD) across folds. The SD is chosen instead of the standard error of the mean (SEM) as the SD is always larger than the SEM, thus the uncertainty of the mean is rather an overestimation than an underestimation of a classifiers confidence,[14] providing for conservative conclusions on classifier performances.

Table 5 reveals the percentage of each artifact being present in each cross-validation fold.

**Table 5:** Distribution of labels across cross-validation folds for each artifact.

| Fold | eyem | chew | shiv | elpp | musc | null |
|---|---|---|---|---|---|---|
| 1 | 2.24% | 0.99% | 1.57% | 2.12% | 0.84% | 92.23% |
| 2 | 2.72% | 0.17% | 0.35% | 0.50% | 1.28% | 94.98% |
| 3 | 3.05% | 0.47% | 0.01% | 0.87% | 0.68% | 94.92% |
| 4 | 2.57% | 2.90% | 0.16% | 0.34% | 1.36% | 92.67% |
| 5 | 1.68% | 0.09% | 0.15% | 0.57% | 2.54% | 94.98% |

## 3.1 Control experiment - no augmentation

### Results

Figure 16 shows the average standardized spectrogram of each of the six artifacts. The values of the x-axis represent each of the frequencies ranging from 0 Hz to 24 Hz, while the values of the y-axis represent each of the 19 channels shown to the left. The values in the spectrograms are calculated as the average of the five folds average within each artifact. In order to normalize the spectrograms to a common scale between -1 and 1, the min-max normalization is applied to each spectrogram independently. The color illustrates the standardized amplitude of the averaged signal. A yellow color indicates a high amplitude, while blue indicates a low amplitude. In this case, undersampling is only visible on the 'null' label, as the other labels are the minority classes in their respective binary classifiers.

[14] SD & SEM, visited 15.04.2021:`https://www.investopedia.com/ask/answers/042415/what-difference-between-standard-error-means-and-standard-deviation.asp`

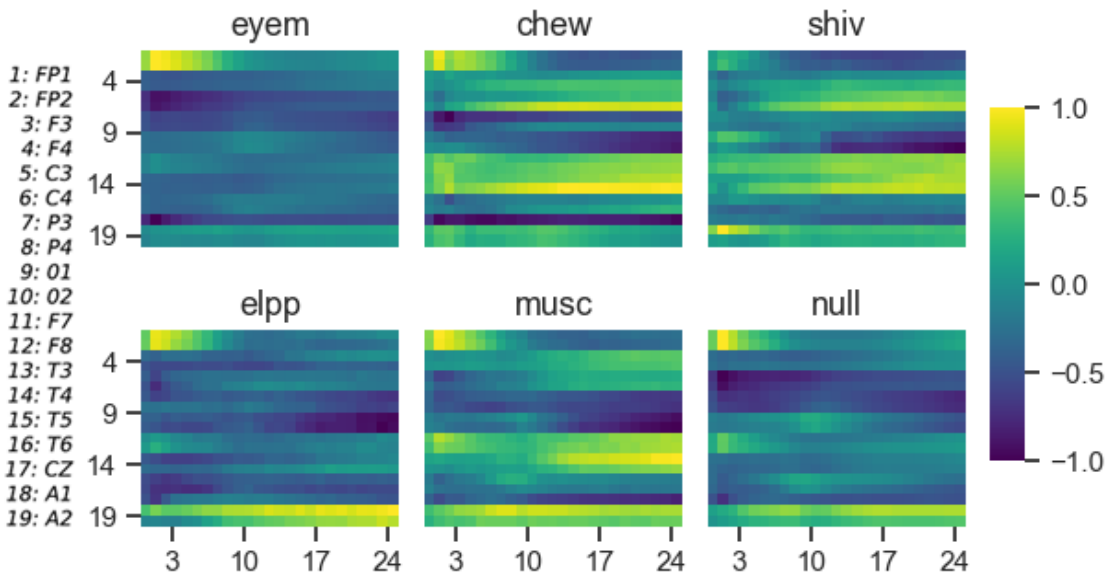



**Figure 16:** Average spectrograms for each artifact. The x-axis of each subplot represents the frequency, ranging from 0 - 24 Hz, while the y-axis represents the channels. The color represents the standardized amplitude of the signal.

Figure 17 shows the average sensitivity and balanced accuracy for all of the binary classifiers in the control experiment, as well as SD. As explained, the SD is chosen instead of the SEM as the SD is always larger than the SEM, thus the uncertainty of the mean is rather an overestimation than an underestimation[15].

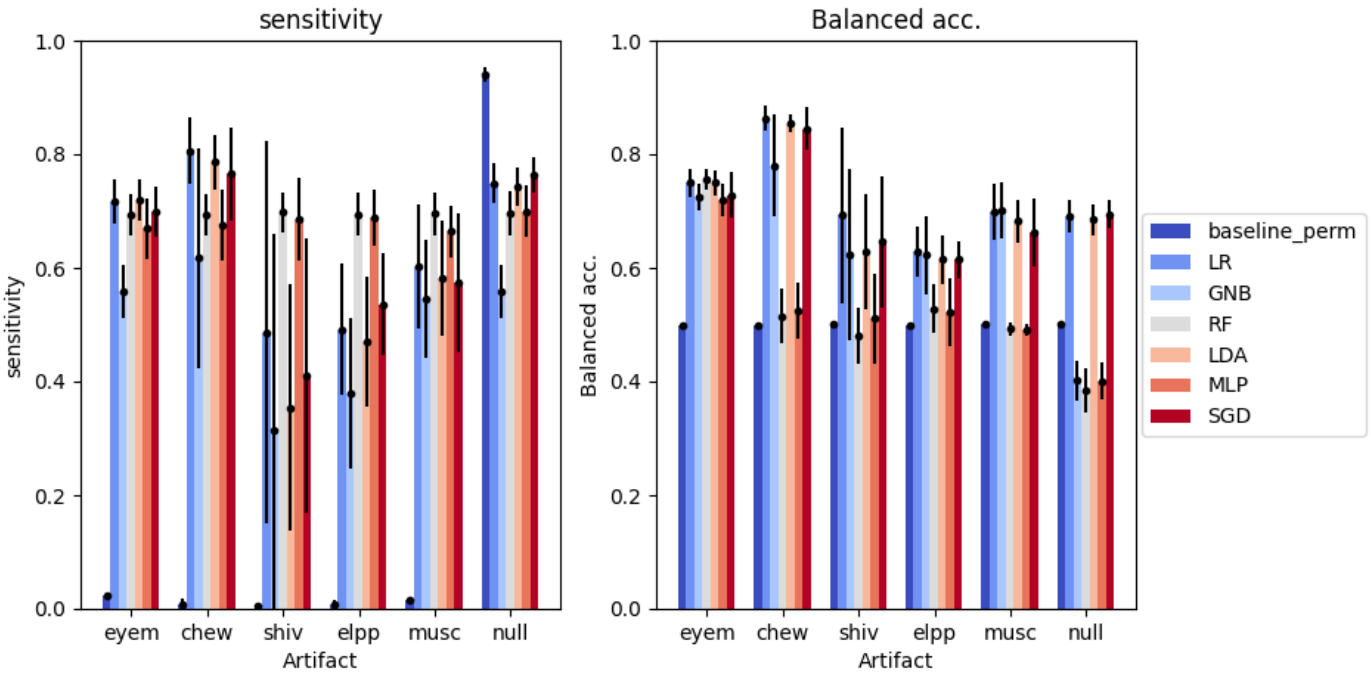



**Figure 17:** Sensitivity and balanced accuracy for all artifacts in the control experiment.

Table 6 & 7 serves as the lookup tables for the previous barplots. They show the average performance for all binary classifiers in the control experiment, measured in sensitivity and balanced accurracy respectively, as well as their standard deviation. The best classifier within each artifact

[15]SD & SEM, visited 15.04.2021:`https://www.investopedia.com/ask/answers/042415/what-difference-between-standard-error-means-and-standard-deviation.asp`

is marked with bold (baseline excluded) for respectively balanced accuracy and sensitivity.

**Table 6:** Mean performance (balanced accuracy) with std. deviation over 5-fold-CV.

| Algorithm | $BAcc._{eyem}$ | $BAcc._{chew}$ | $BAcc._{shiv}$ | $BAcc._{elpp}$ | $BAcc._{musc}$ | $BAcc._{null}$ | Avg. $BAcc.$ |
|---|---|---|---|---|---|---|---|
| LR | 75.03 ± 2.45 | **86.34 ± 2.22** | **69.25 ± 15.51** | **62.94 ± 4.44** | 69.84 ± 4.99 | 69.16 ± 2.84 | **72.09 ± 5.41** |
| GNB | 72.39 ± 2.32 | 78.0 ± 8.95 | 62.28 ± 15.11 | 62.24 ± 6.96 | **70.13 ± 4.84** | 40.23 ± 3.52 | 64.21 ± 6.95 |
| RF | **75.67 ± 1.79** | 51.51 ± 4.86 | 48.03 ± 4.84 | 52.85 ± 4.35 | 49.3 ± 1.13 | 38.38 ± 3.91 | 52.62 ± 3.48 |
| LDA | 74.98 ± 2.2 | 85.48 ± 1.58 | 62.82 ± 10.06 | 61.54 ± 4.27 | 68.24 ± 3.79 | 68.48 ± 2.66 | 70.26 ± 4.09 |
| MLP | 71.85 ± 2.83 | 52.42 ± 4.87 | 51.05 ± 7.99 | 52.27 ± 6.0 | 49.08 ± 1.01 | 40.02 ± 3.25 | 52.78 ± 4.32 |
| SGD | 72.75 ± 4.03 | 84.52 ± 3.82 | 64.59 ± 11.47 | 61.49 ± 3.2 | 66.2 ± 6.01 | **69.44 ± 2.49** | 69.83 ± 5.17 |
| Baseline | 49.99 ± 0.08 | 49.98 ± 0.06 | 50.01 ± 0.09 | 49.99 ± 0.06 | 50.11 ± 0.15 | 50.09 ± 0.07 | 50.03 ± 0.08 |

**Table 7:** Mean performance (sensitivity) with std. deviation over 5-fold-CV.

| Algorithm | $S_{eyem}$ | $S_{chew}$ | $S_{shiv}$ | $S_{elpp}$ | $S_{musc}$ | $S_{null}$ | Avg. $S$ |
|---|---|---|---|---|---|---|---|
| LR | 71.72 ± 3.88 | **80.61 ± 5.82** | 48.69 ± 33.56 | 49.18 ± 11.47 | 60.32 ± 10.94 | 74.85 ± 3.48 | 64.23 ± 11.52 |
| GNB | 55.91 ± 4.72 | 61.75 ± 19.34 | 31.54 ± 34.38 | 37.87 ± 13.31 | 54.62 ± 10.42 | 55.91 ± 4.72 | 49.6 ± 14.48 |
| RF | 69.36 ± 3.66 | 69.39 ± 3.67 | **69.75 ± 3.51** | **69.45 ± 3.77** | **69.5 ± 3.72** | 69.58 ± 3.83 | **69.5 ± 3.69** |
| LDA | **71.88 ± 3.64** | 78.62 ± 4.73 | 35.43 ± 21.77 | 46.99 ± 11.42 | 58.18 ± 10.2 | 74.24 ± 3.46 | 60.89 ± 9.2 |
| MLP | 66.97 ± 5.3 | 67.46 ± 6.28 | 68.48 ± 7.26 | 68.83 ± 4.97 | 66.42 ± 4.55 | 69.93 ± 4.5 | 68.02 ± 5.48 |
| SGD | 69.92 ± 4.43 | 76.55 ± 8.24 | 41.04 ± 24.03 | 53.59 ± 8.9 | 57.29 ± 12.19 | **76.47 ± 3.14** | 62.48 ± 10.15 |
| Baseline | 2.44 ± 0.53 | 0.87 ± 0.94 | 0.48 ± 0.67 | 0.85 ± 0.68 | 1.56 ± 0.62 | 93.97 ± 1.23 | 16.7 ± 0.78 |

Figure 18 shows confusion matrices for the best binary classifier on each artifact trained in the control experiment setting - meaning without SMOTE or augmentation.

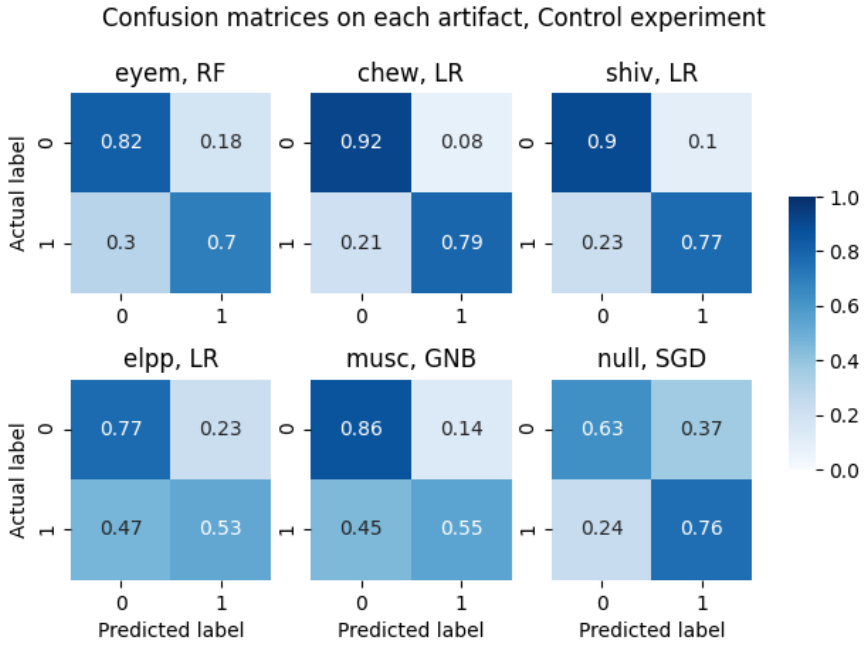



**Figure 18:** Confusion matrix for the best model on each artifact without any augmentation or over-sampling.

### Discussion

The first spectrogram in Figure 16, showing the 'eyem' artifact, clearly shows a much higher energy in the lower frequencies of the first two channels, Fp1 & Fp2, than what is present anywhere else. This is reassuring, as the first two channels represent the frontal channels of the EEG cap, and thus the channels closest to the eyes, as seen on Figure 2. The overall plot seems to suggest that 'eyem', 'elpp' and 'null' are quite similar, while the same thing can be said about 'chew', 'musc' and 'shiv'.

While the spectrograms are similar, there are clear differences to be seen. The difference that can be seen between 'eyem' and 'elpp', is seen in the last two channels, A1 & A2. These channels are the reference electrodes placed on each ear, which might explain the high energy visible during the electrode pop-observations, as it would be reasonable to think that these electrodes might have a higher probability of popping than the rest. The high energy seen at the frontal channels are visible in both artifacts. In 'elpp', this could be explained by contamination from the 'eyem' artifact from small non-detected eye movements, as one could imagine it would be difficult not to move your eyes once the technician fixes the popped electrode. 'null' and 'elpp' however, are almost identical. This could indicate that small eye movements, as well as a higher energy on the reference electrodes are constantly present. 'elpp', however, might be the artifact which makes the least sense to average over, as one could imagine only a single channel would be highlighted in each 'elpp' observation. Thus, 'elpp' and 'null' might look the same, as both are the averages of a lot of very dissimilar observations. As seen in Appendix F (Section 6.6) 'elpp' has a higher SD than all the other artifacts (including 'null') and by looking at some of the incidents of this label, it is evident that it is due to white noise on single channels.

As previously mentioned, 'chew', 'musc' and 'shiv' are very similar. This could easily be explained by the fact that all three are variations of the movement of muscles, and as such would have similar signals. The signal of 'shiv' looks to be slightly smoother, while having a slightly narrower interval of energies, than the other two. This could indicate that shivering consists of less intense muscle movements than the others.

The sensitivities shown in Figure 17 (left), accompanied by Table 7, show that Random Forest is the best classifier for 3 out of 6 artifacts, namely 'shiv', 'elpp' and 'musc'. With sensitivities found to be 69.75, 69.45 and 69.5 respectively. Compared to the benchmarks of the best classifiers presented in the paper by Roy, this is a huge improvement in the classification of the 'shiv' and 'elpp' artifacts, with improvements of 53.52 and 30.7 percent respectively. The performance on 'musc' however, was found to be slightly better in the paper by Roy by 1.26. The remaining artifacts, 'eyem', 'chew' and 'null' were found to have optimal sensitivities, when classified using LDA, LR and SGD, with scores of 71.88, 80.61 and 76.47 respectively. Compared to Roy, only 'null' sees an improvement. The changes in sensitivities compared to Roy are $-1.47$, $-5.46$ and $+4.08$, respectively. As mentioned earlier, the reproducibility in the paper by Roy is lacking, which is why the discrepancies found in comparison to the benchmarks, are not surprising. Though Roy did not mention whether binary or multi-class classifiers were used, it could be inferred that the huge changes found in the classification of 'shiv' and 'elpp' in the paper by Roy are most likely due to the use of multi-class classifiers.

While sensitivity is a great measure when considering the use-case, it becomes biased on an unbalanced test set. As previously mentioned, this was the reasoning behind the balanced accuracy measure, which serves as a more accurate depiction of the overall performance of the model. The balanced accuracies shown in Figure 17 (right), accompanied by Table 6, now show Logistic Regression to be the overall best classifier, outperforming (in the sense of balanced accuracy) the others in 3 out of 6 artifacts, namely 'chew', 'shiv' and 'elpp'. The corresponding balanced accuracies found are 86.34, 69.25 and 69.94. The remaining artifacts, 'eyem', 'musc' and 'null' receive the best balanced accuracy scores through RF, GNB and SGD respectively, with scores of 75.67, 70.13 and 69.44.

As can be seen, sensitivity and balanced accuracy tell quite different stories. Balanced accuracy should be the more informative measure, as a classifier constantly predicting the class present would receive a sensitivity of 100%. However, the sensitivity must not be disregarded, as it is important for the use-case. Thus, both are important measures and to find the best classifier for the use-case, one would have to determine how to trade-off between balanced accuracy and sensitivity in a proper way.

The confusion matrices seen in Figure 18, seem to suggest that the best classifiers are 'eyem', 'chew' and 'shiv', when it comes to predicting both absent and present. However, all classifiers, except 'null', seem to correctly predict absent more frequently than present, which is not what preferred for the use-case. Especially 'elpp' and 'musc' are easily confused, and very often predict present as being absent, whereas the opposite is the case for 'null'.

## 3.2 Performance improvement experiment

### 3.2.1 Augmentation

Figure 19 and 20 are extensions of the average standardized spectrograms, reported during the control experiment, only now the augmented data for each technique is included in the calculation. In Figure 19 the data was balanced using under-sampling, while in Figure 20 SMOTE over-sampling of ratio 1 combined with under-sampling was applied. As opposed to the previous spectrogram (fig. 16), the min-max normalization is applied within each artifact, across augmentation methods in order to best compare the augmentation techniques.

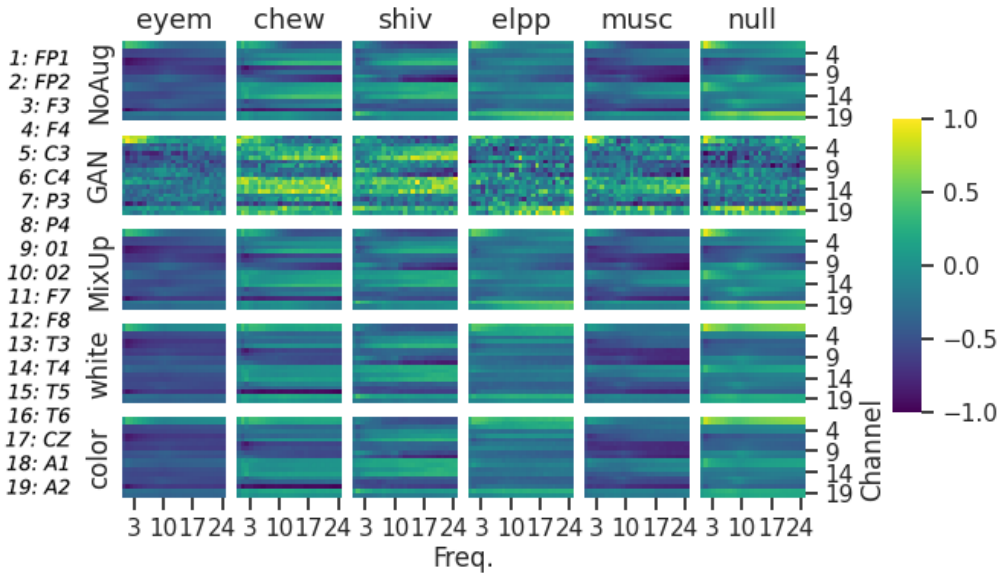



**Figure 19:** Average spectrograms of each artifact, with each method of data augmentation, when undersampling is used to binarily balance each class. The x-axis of each subplot represents the frequency, ranging from 0 - 24 Hz, while the y-axis represents the channels. The color represents the standardized amplitude of the signal.

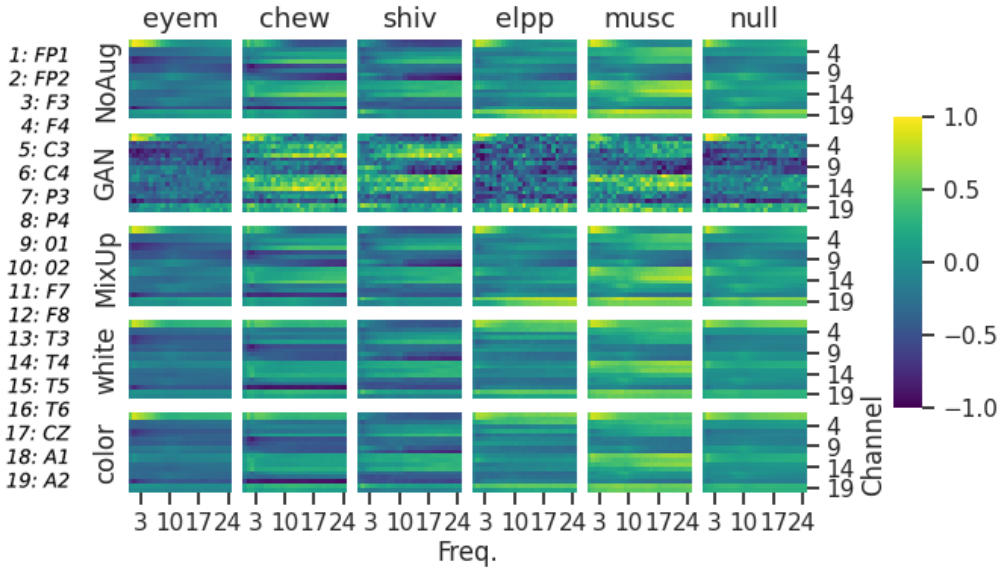



**Figure 20:** Average spectrograms of each artifact, with each method of data augmentation. The x-axis of each subplot represents the frequency, ranging from 0 - 24 Hz, while the y-axis represents the channels. The color represents the standardized amplitude of the signal.

In Figure 21 and 22 subplots of the difference in performance - measured using balanced accuracy -

between the control experiment models and the models trained with data from each augmentation technique with the best found augmentation ratio using no SMOTE and SMOTE with ratio 1 respectively is visualized for all artifacts. The height of the bars are calculated by subtracting the performance of the control models from the augmentation models and the error bars are calculated as the pooled SD of the respective SDs.

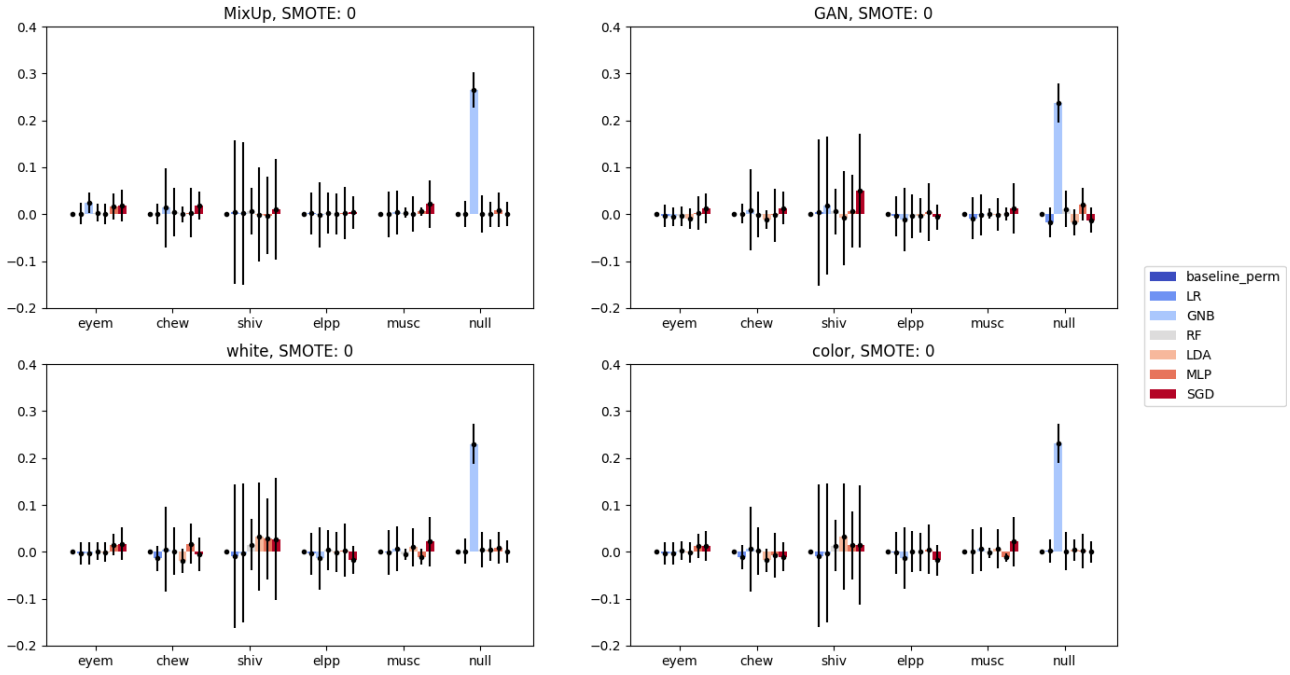



**Figure 21:** Difference in balanced accuracy between models using different augmentation techniques and ratios without SMOTE for balancing.

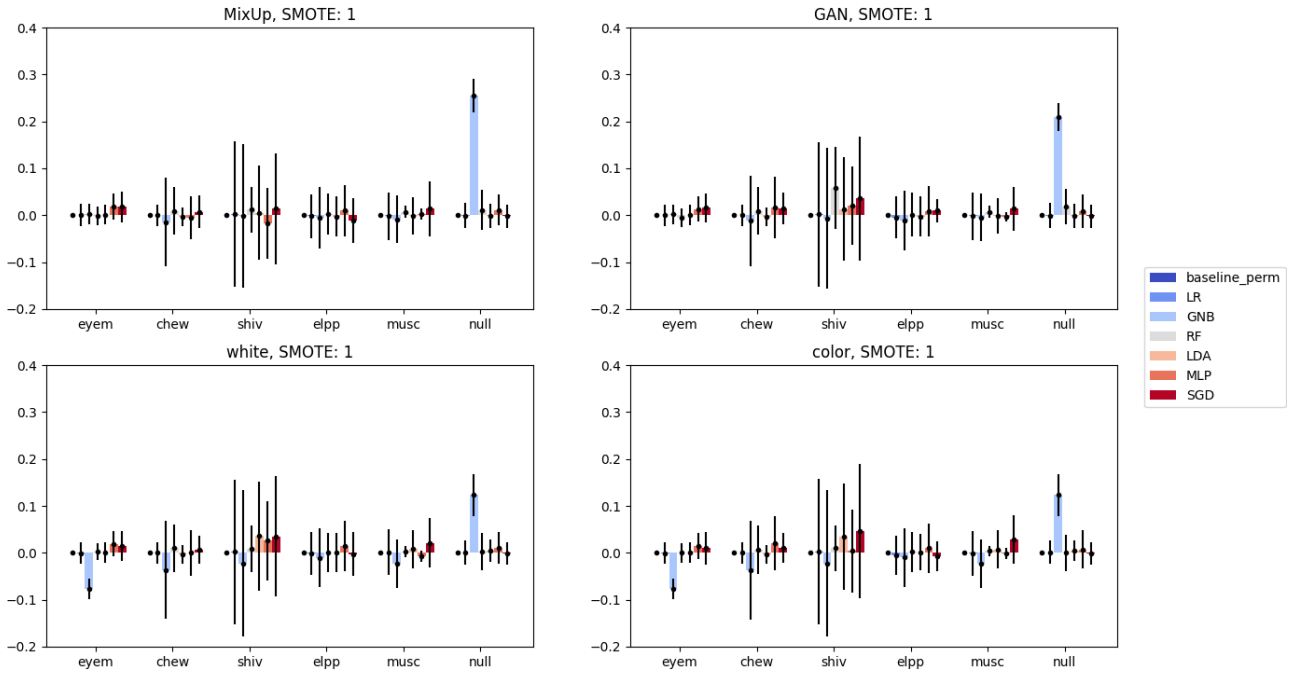



**Figure 22:** Difference in balanced accuracy between models using different augmentation techniques and ratios with SMOTE ratio of 1 for balancing.

In Figure 23 and 24 subplots of the difference in performance - measured using sensitivity - between the control experiment models and the models trained with data from each augmentation technique with the best found augmentation ratio using no SMOTE and SMOTE with ratio 1 respectively is visualized for all artifacts. The height of the bars and the error bars are calculated as previously mentioned.

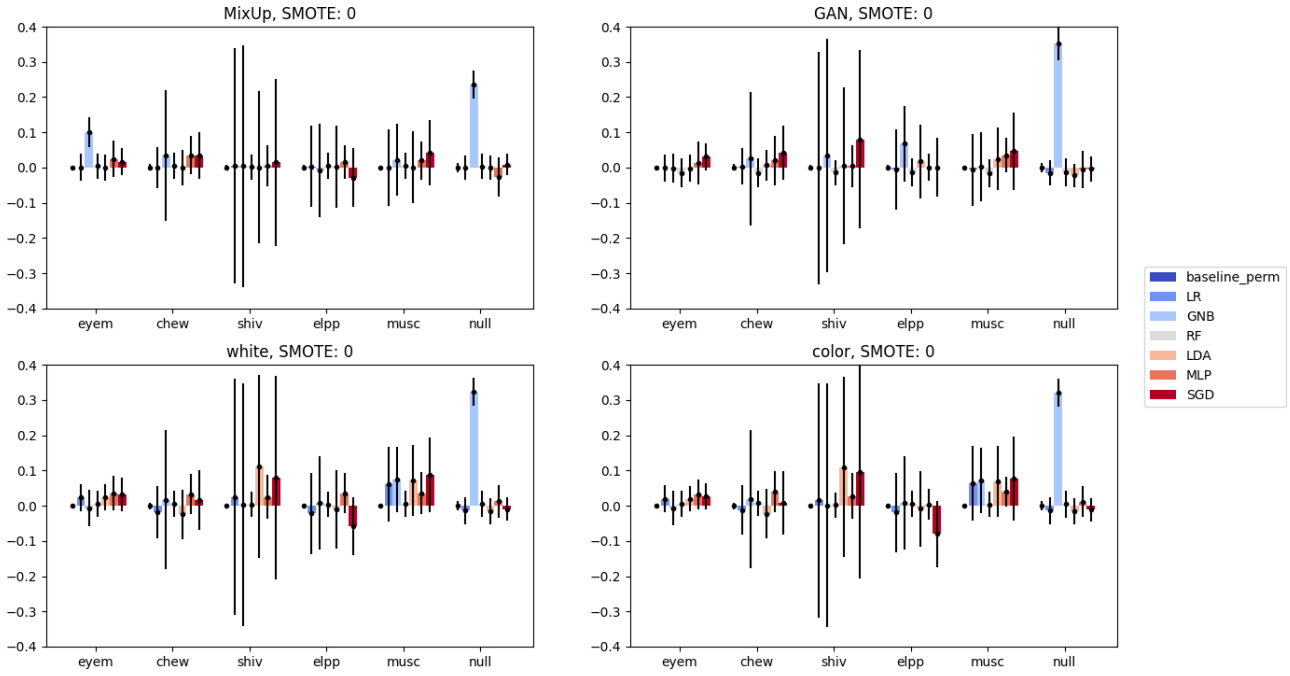



**Figure 23:** Difference in sensitivity between models using different augmentation techniques and ratios without SMOTE for balancing.

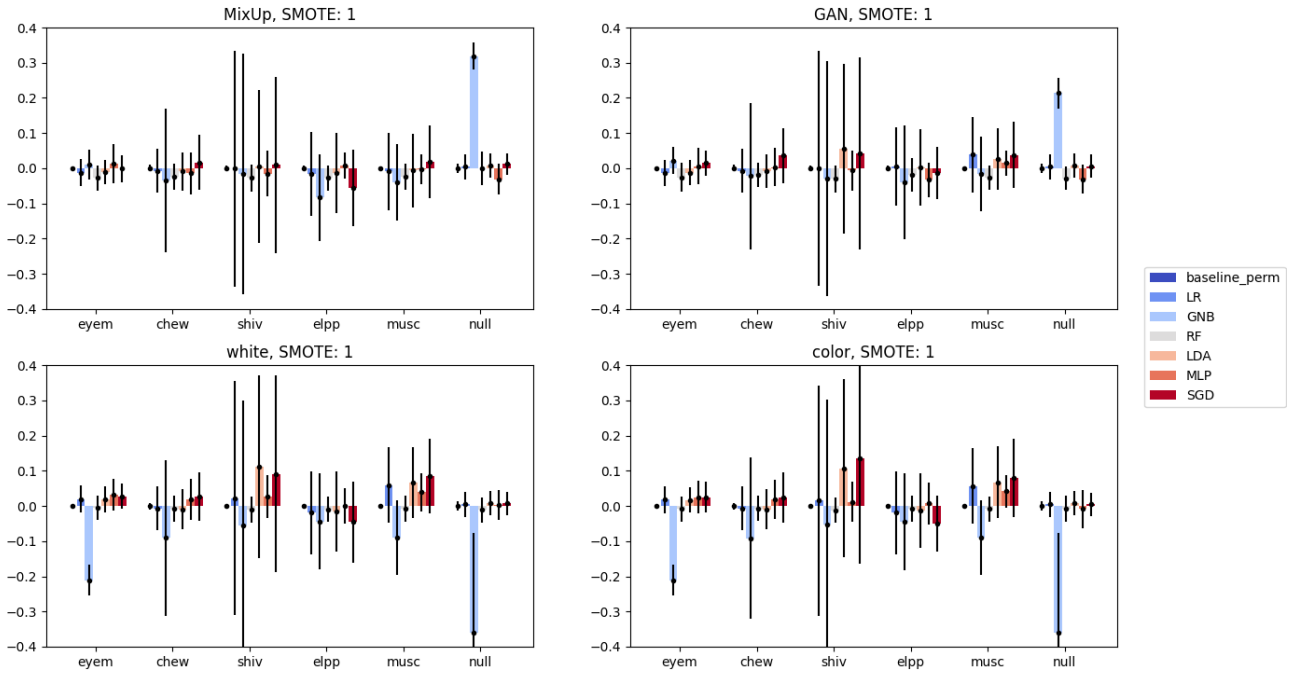



**Figure 24:** Difference in sensitivity between models using different augmentation techniques and ratios with SMOTE ratio of 1 for balancing.

The augmentation ratios for the models used in Figure 21, 22, 23 and 24 as well as tables of their balanced accuracies and sensitivities are reported in Appendix E (Section 6.5).

In Figure 25, the Balanced accuracies of the best performing classifiers using each type of augmentation technique, is visualized along with the best performing model from the control experiment, for all artifacts, regardless of their SMOTE ratios. These help to illustrate whether the best performing classifier uses some sort of data augmentation or not. A list of the 100 best performing classifiers for each artifact, as well as their corresponding augmentation technique, ratios and SMOTE ratios, are reported in Appendix G (Section 6.7). The classifiers are ranked in regards to their balanced accuracies.

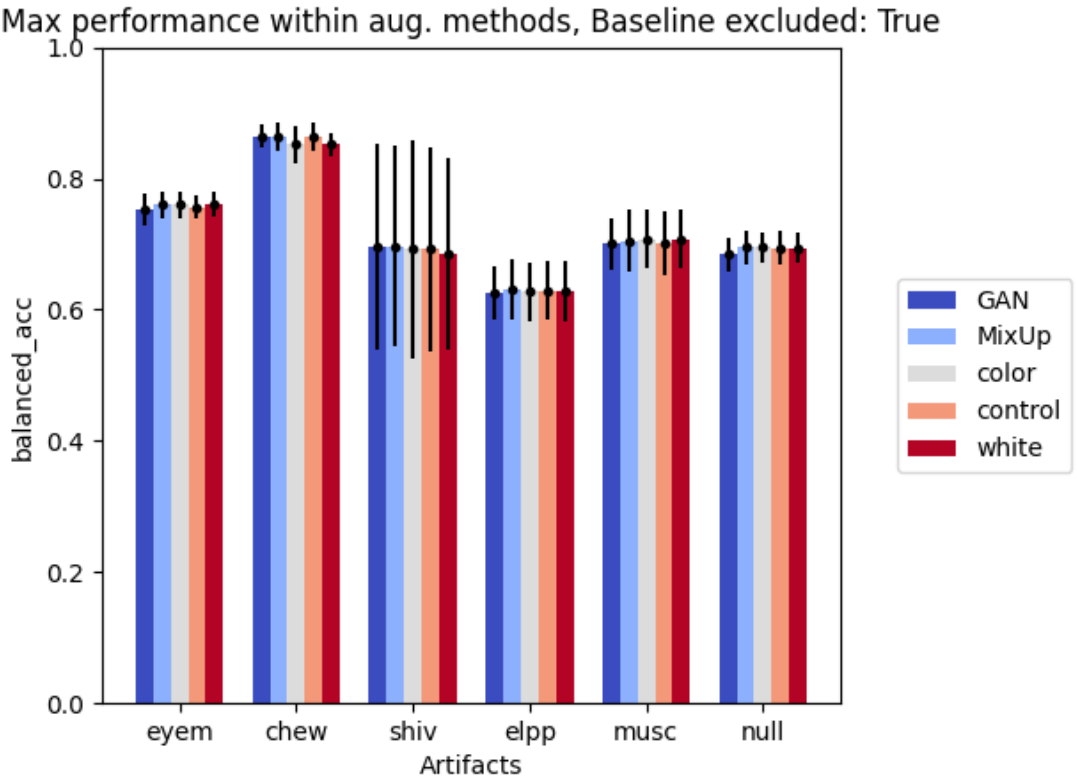



**Figure 25:** Balanced accuracy for the best model (independent of SMOTE ratio) trained using each augmentation technique compared to the best model from control experiment.

**Table 8:** Balanced accuracy on each artifact for the best models on each augmentation technique. The model specifications can be found in the table in Appendix H (Section 6.8)

| | eyem | chew | shiv | elpp | musc | null |
|---|---|---|---|---|---|---|
| Control | 75.67 | 86.34 | 69.25 | 62.94 | 70.13 | 69.44 |
| MixUp | 75.97 | 86.38 | **69.69** | **63.15** | **70.48** | **69.48** |
| GAN | 75.26 | **86.45** | 69.60 | 62.54 | 70.02 | 68.39 |
| colorNoise | 75.96 | 85.20 | 69.20 | 62.77 | 70.73 | 69.46 |
| whiteNoise | **75.98** | 85.16 | 68.48 | 62.80 | 70.79 | 69.45 |

The adjusted $p$-values are shown in Table 9 computed for the best models of each augmentation technique on each artifact, determined from the list of the 100 best classifiers within each artifact, in Appendix G (Section 6.7).

**Table 9:** $p$-values found from a students t-test between the best model in each augmentation technique (left column) and the best control model on each artifact, based on the method presented by Dietterich [43]. The $p$-values marked bold reflect the null-hypotheses which can be rejected.

| Augmentation | eyem | chew | shiv | elpp | musc | null |
|---|---|---|---|---|---|---|
| MixUp | 0.034 | 0.316 | 0.092 | 0.092 | **0.006** | 0.452 |
| GAN | 0.321 | 0.602 | 0.092 | 0.092 | **0.006** | 0.052 |
| whiteNoise | 0.092 | 0.092 | 0.092 | 0.092 | **0.006** | 0.865 |
| colorNoise | **0.013** | 0.224 | 0.099 | 0.092 | **0.006** | 0.861 |

**Discussion**

When comparing the augmentation methods in each column of the spectrograms in Figure 19 and 20, there is a clear tendency of the average artifacts looking very similar to each other. In fact, there are almost no visible differences between the average observations generated through MixUp and Noise Addition, when compared to the original data. Looking closely, the Noise Addition methods seem to be blurring the average spectrograms a bit, as could be expected in a method which adds noise to the signal. The reason why colored noise is not that different from the white noise spectrograms, could be that all spectrograms are cut off at a frequency response of 24 Hz - whereas the cut-off filter used to filter the white noise, in order to make it colored, was set to 30 Hz. If one were to investigate wider frequency ranges of the spectrograms, a clearer difference between the augmentation techniques would be expected, however, one must remember that neural activity transmits low-frequency signals within a range from approximately 0 to 30 Hz.

MixUp also seems to blur the spectrograms, although not as much as Noise Addition. Since MixUp is implemented as a mix of two observations, from different classes, it is reasonable to assume that this would add noise onto the signal as well, thus explaining the blurring effect.

The augmentation technique which stands out the most is GAN. In both plots, GAN seems to have learnt the basics of the artifacts, although the generated observations appear to be more noisy than the others, which is seen by the varying frequency intensities ranging from -1 (dark) to 1 (light) for neighbouring frequencies. This might indicate that the default 100 epochs of which the GAN was trained on, was not enough for the relatively complex features. The reason could also be explained by the fact that GANs generate each observation from random visual noise. The noisier observations in GAN, however, might not be a problem in the case of robustness, as the most important features of the original data seems to have been learnt. Interestingly, GAN seems to have exaggerated the energy present in the artifacts, making the lighter frequencies of the original observations lighter, and the darker frequencies darker, suggesting that the GAN is trained to exaggerate variance within the artifacts. Comparing the spectrograms of each Figure, 19 and 20, i.e. SMOTE ratios of 0 and 1 respectively, it seems as though SMOTE generally increases the energy present in the spectrograms.

PCA plots investigating the resemblance of each augmentation technique, compared to the original data can be found in Appendix D (Section 6.4). These plots indicate, similarly to the spectrograms in Figure 19 and 20, that the augmented observations resemble the original data quite well. Once again GAN seems to have a slight tendency of exaggerating features, while being more noisy in general.

As seen in Figure 21, 22, 23, and 24, most of the best version of each classifier with their respective optimal augmentation ratio, do not seem to improve the classification particularly. Figures 21 and 22, showing the improvement in balanced accuracy with SMOTE ratio 0 and 1 respectively, seem to suggest that most of the best version of each classifier type trained using augmented data do, in fact, improve the classification of the various artifacts, although very minimally. The error bars calculated from the 5-fold-CV, are so large that it is impossible to conclude anything with certainty. Interestingly, the various data augmentation techniques, seem to improve the GNB classifiers prediction of 'null' greatly, by up to 26.5 percent, even though that was the artifact with the most observations to begin with. Comparing the two figures, it seems as though SMOTE makes the balanced accuracy deteriorate quite significantly for the GNB-classifiers in particular, most notably 'eyem' and 'null', while it seems to have no effect on the other classifiers. The deterioration interaction between SMOTE and GNB, could be caused by the linearity in the observations

created by SMOTE, as these might mess up the Gaussianally distributed likelihood of the Gaussian Naive Bayes classifier. All in all, it becomes quite clear, that a combination of SMOTE and other variations of data augmentation, does not further improve the classification of EEG artifacts.

The same tendencies are seen for the sensitivity metric on Figure 23 and 24 - specifically the deterioration of GNB's performance when using a SMOTE ratio of 1. However, in contrast to the balanced accuracy metric, a handful of models show a larger trend of improvement compared to the control experiment based on sensitivity.

As mentioned, the barplots in Figure 25, as well as Table 8, show the balanced accuracy, of the best classifier found for each artifact for each augmentation technique. The plot and the table seem to suggest however, that there are no classifiers using augmentation, which show much of an improvement compared to the best classifiers found in the control experiment. In fact, while MixUp seems to perform best, all seem to be approximately equal. For all artifacts, there is an augmentation technique with a specified augmentation ratio which does minimally improve the performance compared to the best control model, however, when looking at the best within each augmentation method most of them deteriorated their performance compared to the control experiment. The deterioration mainly happened when GAN was used on 'eyem' and 'null', and when both types of Noise Addition was used on 'chew'. The main takeaway, however, is that the best classifiers in the control experiment saw little to no improvement with the addition of augmented data, as the error bars are overlapping in all cases. In Figures 21 and 22, we previously saw that, although it was not much, some of the binary classifiers did see improvement. After seeing Figure 25, however, we can conclude that the classifiers which saw a large improvement were not among the best classifiers from the control experiment to begin with, and although they saw improvement, it was not enough for them to outperform (in the sense of balanced accuracy) the already top performing classifiers. This can most easily be seen in the example of the GNB classifier on the 'null' artifact, which saw a significant improvement of 26.5% when using MixUp. When comparing the classification of 'null' using GNB and the best performing classifier for 'null', SGD, in the control experiment, we see balanced accuracies of 40.23 and 69.44 percent, respectively. Thus we see that even though GNB saw a huge improvement of 26.5 percent, it still was not enough to outperform the SGD classifier, which only saw a maximum improvement of 0.04 percent, using MixUp. Thus, it seems as though data augmentation helps the classifiers which were not performing that well to begin with, but the already well-performing classifiers do not see much of an improvement, in fact, the performance of some deteriorates.

By looking through the list of the top 100 performing classifiers per artifact, regardless of their type of classifier, augmentation technique, ratio or SMOTE ratio, reported in Appendix G (Section 6.7), we see a tendency showing that the best performing models, usually are of the same type of classifier - i.e. the prevalent type of classifier for 'eyem' is RF, for 'chew' it is LR etc. - while the augmentation method, ratio and SMOTE ratio seems to be more or less random. Thus, it seems as though the type of classifier has much more of an influence on the performance, than the augmentation method has, which further proves our comment of the augmentation techniques not improving the classification for the already best classifiers.

One reason why the data augmentation does not improve the classification as expected, might be explained by the spectrogram plots in Figure 19 and 20. As mentioned, the augmented data was incredibly similar to the original data. Thus, the reason that most of the best augmentation versions of each augmentation technique does not improve nor decrease particularly compared to their performance in the control experiment, could simply be that the augmented data resembles the original data too much to be of benefit for training ML classifiers.

While the barplots in Figure 25 seemed to show that there was no significant difference between the augmentation- and the control classifiers respectively, some of the $p$-values shown in Table 9 seem to tell a different story. Here, we see a significant difference between the performance of the best augmentation classifiers and the control classifiers, in all augmentation methods for 'musc', as well as when using colorNoise on 'eyem'. This might seem very strange considering the barplots, where there were no real difference in the means, within the augmentation methods. This can be explained, however, since the $p$-values were calculated over multiple folds, in accordance with the paper by *Dietterich*, this must mean that the difference between the classifiers was significant within each fold. However, the only case where there is a significant difference in the folds, but none when averaging the folds, is in the case where the classifiers would alternate between being

better than the other throughout the folds. Thus in one fold, the augmentation method would be significantly better than the control classifier, however in the next, the control classifier would be significantly better than the augmentation classifier. Thus, the overall means of the two would end up being similar, while the $p$-values would show a significant difference. Based on this discovery, the error bars in the barplots seem to be the more valid measure over a models confidence, and as such we will conclude no significant difference between the classifiers.

As argued in this discussion, none of the best augmentation technique's best models individually perform better than the best control model and it is therefore of interest to examine if the ensemble learner could help improving the performance compared to the control experiment.

### 3.2.2 Ensemble methods

Figure 26 visualizes the correlation matrices over the predictions of the 20 best individual models (potentially including the ones from the control experiment). The model's specifications, such as augmentation technique and ratio, etc. can be found in the PDF in Appendix H (Section ). The lighter the $i$'th row/column is (since the matrix is symmetric), the lower is the mean correlation between the $i$'th best model and the remaining models.

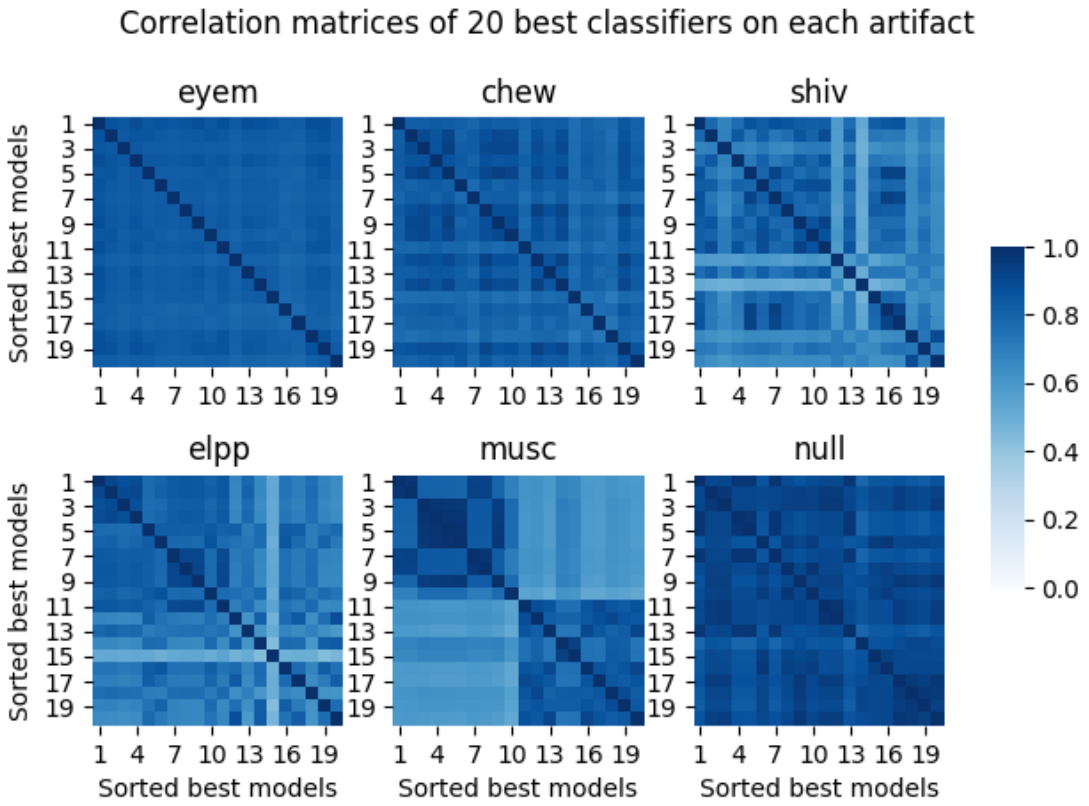



**Figure 26:** Correlation matrices of the 20 best individual classifiers for each artifact. The row and column indices refer to the $i$'th best model in line with the list found in Appendix I (Section 6.9).

Specifications on the 5 classifiers - chosen from the lowest mean correlation to the remaining classifiers - for the ensemble learner on each artifact can be found in Appendix J (Section 6.10). The performance of each of the resulting ensemble learners are plotted against the performance of the best control model for each artifact in Figure 27. The height of the bars denote the classifiers balanced accuracy and the error bar is the mean standard deviation as previously explained.

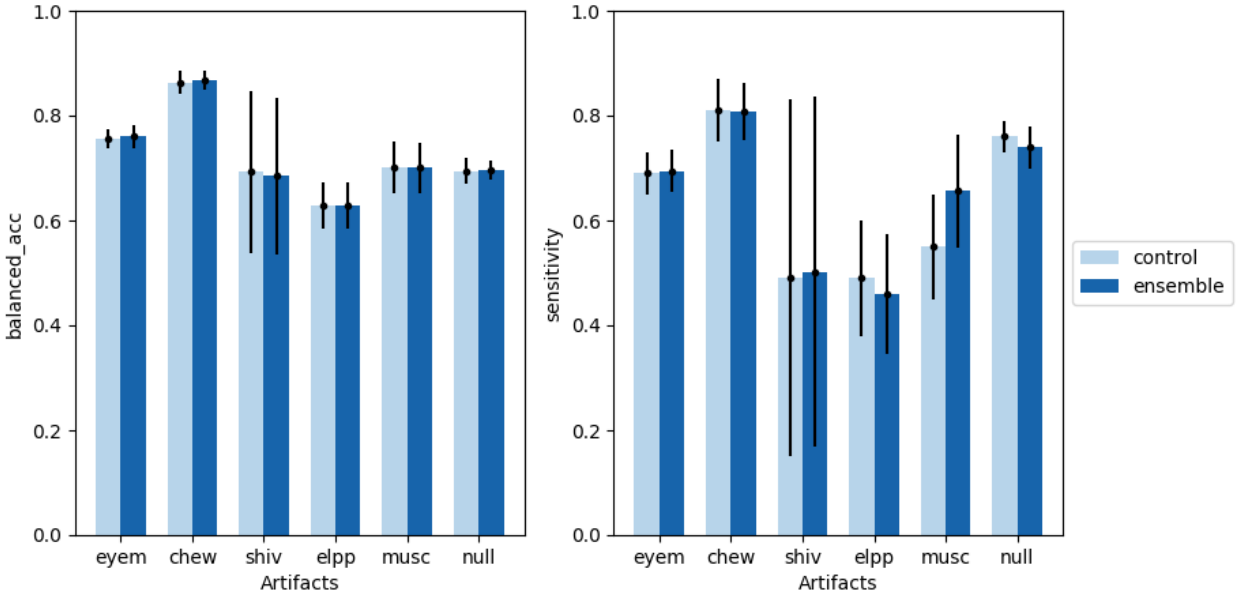



**Figure 27:** Performance of best control classifier on each artifact compared to the respective ensemble learner for both balanced accuracy and sensitivity metric.

The balanced accuracy performance plotted in Figure 27 is reported in Table 11 for the control models and the ensemble learner on each artifact.

**Table 10:** Balanced accuracy on each artifact for the best models following Figure 27.

| | $BAcc._{eyem}$ | $BAcc._{chew}$ | $BAcc._{shiv}$ | $BAcc._{elpp}$ | $BAcc._{musc}$ | $BAcc._{null}$ |
|---|---|---|---|---|---|---|
| Control | 75.67 ± 1.79 | 86.34 ± 2.22 | 69.25 ± 15.51 | 62.94 ± 4.44 | 70.13 ± 4.84 | 69.44 ± 2.49 |
| Ensemble | 76.02 ± 2.25 | 86.67 ± 1.81 | 68.51 ± 14.96 | 62.88 ± 4.51 | 70.03 ± 4.77 | 69.53 ± 1.82 |

The sensitivity performance plotted in Figure 27 is reported in Table 11 for the best control models (based on balanced accuracy) and the ensemble learner on each artifact.

**Table 11:** Sensitivity on each artifact for the best models following Figure 27.

| | $S_{eyem}$ | $S_{chew}$ | $S_{shiv}$ | $S_{elpp}$ | $S_{musc}$ | $S_{null}$ |
|---|---|---|---|---|---|---|
| Control | 69.36 ± 3.66 | 80.61 ± 5.82 | 48.69 ± 33.56 | 49.18 ± 11.47 | 54.62 ± 10.42 | 76.47 ± 3.14 |
| Ensemble | 69.44 ± 4.08 | 80.74 ± 5.37 | 50.22 ± 33.39 | 46.02 ± 11.5 | 65.6 ± 10.73 | 73.93 ± 4.08 |

In Figure 28 the multi-label confusion matrix of the six ensemble learner's predictions on their respective artifact is plotted. The confusion matrices are summed across folds and then standardized. Following the colorbar in the right side of the figure, a darker color signifies a higher percentage of either true-positives, false-negatives, etc..

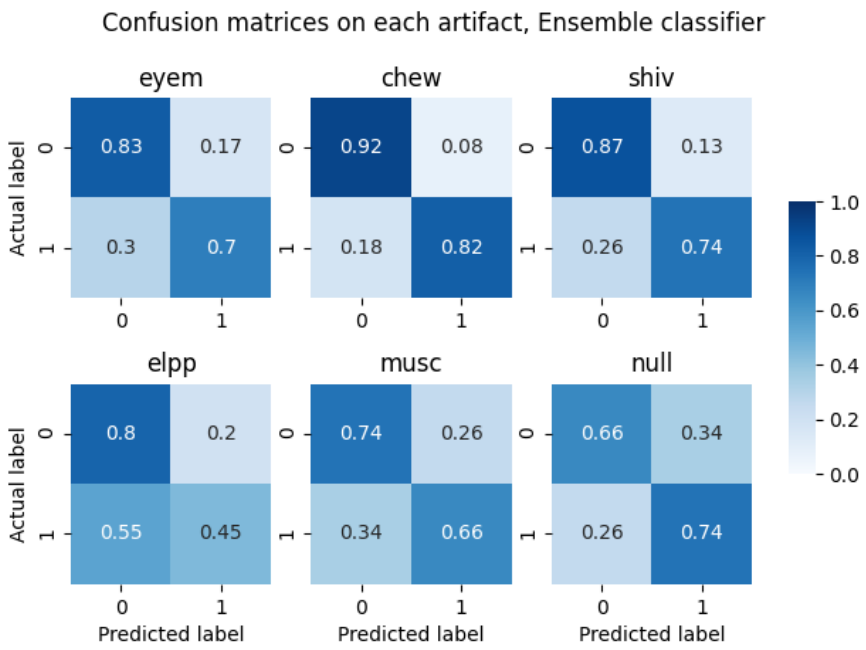



**Figure 28:** Multi-label confusion matrix created from each ensemble learner's prediction on their respective artifact.

### Discussion

By looking at the bars in Figure 27 and the corresponding balanced accuracy values in Table 11, there is not really any indication of difference between the ensemble learner's and the control classifiers as the values of these diverge by less than one percent for all artifacts. At first this might sound peculiar as the models for each ensemble learner were chosen based on low correlation to the other classifiers which intuitively would yield a majority voting classifier performing different than a single classifier. However, there are two potential flaws in this case. First of all, as seen in Figure 26 the correlation between the 20 best classifiers on each artifact is quite high in general. There is some variation on this on the labels 'elpp', 'shiv' and especially 'musc', where 'musc' have several classifiers with low correlation to the remaining classifiers and 'shiv' and 'elpp' only have a one or a couple of models with low correlation to the remaining ones. Thus, since the ensemble learners consist of 5 classifiers, there might simply not be enough models with low correlation between them to improve using an ensemble learner. Another reason for the low variation to the control classifiers could be that the correlation is based on the average correlation to the remaining models, resulting in the two models with lowest average correlation being selected independently from these two model's mutual correlation.

The confusion matrices in Figure 28 reveal that the ensemble learners seem to capture more or less the same tendencies as the best control model on each artifact seen from the confusion matrices of 18 but differ for some of the artifacts. As explained, the balanced accuracy only differs within one percent between the best control model and the ensemble on each artifact but this is not synonymous with the classifiers having the same predictions as is indicated when comparing confusion matrices from Figure 18 to the ones from Figure 28. As is clearly seen in the confusion matrices on the label 'musc', the ensemble learner differ in its predictions by predicting more on the artifact being present than absent in the case of the artifact actually being present or absent. As is further seen in Figure 27 the sensitivity of 'musc' increases to some extend namely as the label is predicted as present more frequently. When considering the use-case one might argue that the ensemble learner in the case of predicting on the 'musc' label would be preferable to the control classifier, as they have more or less the same balanced accuracy of 70% whereas 'musc' has a mean sensitivity approximately 10% higher than the control and thereby would not have the same tendency to overlook when a 'musc' artifact occurs. Contrarily, for the ensemble classifier on the artifact 'null' and 'elpp' this prediction pattern is opposite meaning that more predictions are labelled as the artifact being absent when using an ensemble learner resulting in a lower sensitivity. For the remaining labels, the sensitivity, balanced accuracy and confusion matrices are very similar. In relation to the use-case it is quite easy to choose the better classifier if they perform equally

when considering balanced accuracy but differ in sensitivity as the more sensitive classifier would ensure that less of the actual artifacts are disregarded, thereby helping to enhance online artifact removal.

## 3.3 Overall discussion

In general, the best models from the augmentation and ensemble (Figures 25 & 27) do not indicate a significant increase in performance in neither of the two experiment, and therefore neither of the two research questions in the *Expectations* section can in general be approved nor disapproved. However, these research question were built on the idea of creating a robust model, but evaluating whether or not a model has achieved such a state might not be solely dependent on the performance of the models themselves, as done in this paper. As formerly mentioned in the *Introduction* section, the idea of robustness originated due to the individuality and ever-changing signals of the human brain and the noisiness of the EEG measured signals. Thus training a machine learning model on such uncertainties and skewnesses of the data might make the models biased in some sense, and therefore some form of robustness becomes a necessity. These possible uncertainties of the data have been accounted for by augmenting the data and measuring its effect by looking at the performance of the models compared to no augmentation. However, since the performance of the models do not decrease with the inclusion of the augmented data, and since the augmented data follows the VRM principle, an increase in robustness could be present without an increase in performance when applied in the real-world. However, one could argue that improvement in performance can to some extend be regarded as improvement in robustness, as the experimental set-up accounted for the individuality of EEG recordings by stratifying on the subject ID. If following this thought some of the classifier types with low performance, such as the GNB, can be considered to improve in robustness as it experienced a significant gain in balanced accuracy when trained with augmented data.

In relation to this stratification on subjects (same number of subjects in each fold), the distribution of artifacts across folds is quite varying (Table 5). Thus, this distribution and the results found using this train/test-split might be quite dependent on the random seed chosen. There could be several approaches to fix such variance, like basing the stratification on both subjects and artifacts or making a $5 \times 2$-fold cross-validation to generalize the variance as recommended by Dietterich et. al. [43], and further avoid overlapping training data. On the other hand, SMOTE over-sampling was to avoid the problem of having too few samples, but the technique was applied to each training set individually of the others, and therefore the variance across folds is still present.

Furthermore, there might be an issue with over-sampling the minority class to such a point that the vast majority of observations would consist of augmented data. This could be the case since the classifiers seem to perform better without SMOTE than with SMOTE. In addition, the preprocessing pipeline already includes the augmentation/over-sampling technique Sliding Windows, where a 75% temporal overlap was used. This creates a vast amount of data and might result in redundancy for the augmented data. Another approach and more efficient (in the sense of cheap and effective data collection and labelling) could be to choose data that gives the most informative data points and use augmented data to still provide for robust classifiers. This could be done by using active learning and could prove to be very useful if some artifact data is of low quality and thereby not very representative for the artifact.

This paper did not attempt to optimize the augmentation techniques, i.e. the alpha value in MixUp, the covariance scaling factor and cut-off frequency in Noise Addition and hyperparameters such as epochs, label smoothing and hidden layers in GAN. It is also visible in the spectrograms where GAN is clearly not similar to the original data. Therefore the augmentation techniques can be thought of just being applied and not optimized in this paper, but optimizing the models might have improved the results. Further optimization could be made in relation to the models, especially MLP which had a lot more hyperparameters to optimize than HyperOpt was given. These hyper-parameters could be interesting to investigate, since MLP probably has the highest potential to over-fit the data, and therefore the effect of creating a more robust classifier could be more visible when augmenting the data. But in general, the control models were optimized in the same manner as in the improvement experiment, therefore the augmentation should probably have the same effect.

In relation to the use-case, the models were optimized based on the F2-score, where the general

idea was to give more importance to sensitivity than precision. But as mentioned in the *Metrics* section, the F2-score originates from the $F_{beta}$-score, and therefore the trade-off between sensitivity and precision could be further assessed by adjusting the *beta* value. Similarly, when applying these models in the real world, a trade-off between the false-positives (i.e. sensitivity) and how precise the classifiers are (in this case balanced accuracy), since accounting for and acting on an artifact that is not present is probably at lower cost than letting an artifact pass.

In comparison, the classifiers were able to achieve the same results as the state-of-the-art classifiers reported by Roy (2019) [20], with some of the classifiers obtaining even better results, however the models used at not fully coherent. Even though the reproducibility is lacking, it is worth stressing that the stratification on individuals has a big effect on the results, and it is unclear whether or not Roy used this stratification. Other studies also provide similar results, but a comparison based on accuracy was not desired in this paper, and rather a metric not affected by the unbalanced test set. This paper also only considered spectral content of EEGs, where others research indicated that a combination of temporal and spectral analysis of the signal could be of benefit as well. This opportunity could be worth investigating further.

# 4 Conclusion & perspectives

While data augmentation does seem to be able to improve the performance of some of the presented seven types of classifiers, the results seem to show that this is only the case for the classification methods performing poorly on the original data. While these classifiers showed promising improvements, it was not enough to obtain a balanced accuracy higher than the already top-performing classification methods within each artifact, even though the top-performing classifiers saw little to no improvement with the addition of augmented data. Throughout this report, further experiments have been evaluated - the over-sampling technique, SMOTE, has been tested in combination with other data augmentation techniques, while an ensemble learner, predicting by majority vote, has been tested as well. Unlike the expectations, the results seem to show that these variations provide no further performance improvements either. These findings however, do not mean that the classifiers found are not robust, as the best classifiers found show high performances and even improvements compared to the state-of-the-art, in relation to both sensitivity and balanced accuracy, when evaluated on data from unseen subjects due to the stratification in the cross-validation.

In relation to the use-case, the end-goal is to combine the best binary classifiers for each artifact into a multi-label classifier, able to predict multiple labels at once for the technician to correct when collecting EEG data on-site. This however, would require finding a proper way to determine the trade-off between balanced accuracy and sensitivity, as the results show that these metrics measure the performance of the classifiers quite differently. As balanced accuracy is the most accurate depiction of the performance of the two, one approach would be to pick the binary classifier with the highest sensitivity from a list of the $N$ best classifiers with regards to their balanced accuracy - as the decision should not be made on sensitivity alone, though it is important for the use-case.

However, due to the individuality and ever-changing state of the human brain, it becomes incredibly difficult to form a strict generalized conclusion, as the results would likely vary depending on the subjects and data used when training the models. One could imagine that doing these same investigations on a different data set, consisting of a different population, with different ages, genders and pre-existing conditions, likely would yield completely different results. This could lead to future research in terms of transfer-learning on new data sets, i.e. the data set from BrainCapture. Namely, as the brain is ever-changing and in some points depending on age, it might be of ethical interest to examine whether the artifact detection approach investigated in this project hold a bias towards age.

When comparing the top classifiers found within each artifact, we see a tendency of the classification method (LR, RF or so) having a much higher impact on the performance, than any of the data augmentation techniques or SMOTE does. For some of the artifact labels, an explanation could be that the classifiers already have enough valuable data points, to the point where new augmented observations cannot further improve the classification. One could imagine that augmentation would have a larger impact on performance improvement, if the classifiers were trained on a smaller data set. In relation to the real world scenario, where it is both expensive and time-

consuming to label EEG-data, as the labeling happens off-site, it could be of interest to examine whether an active learning approach would be advantageous in the creation of such a data set, and whether this approach, in combination with data augmentation, potentially could reach similar results to the ones found in this report, while being both cheaper and less time-consuming in terms of data collection and labelling.

# References


[1] *WHO - Epilepsy*. 2019. URL: https://www.who.int/news-room/fact-sheets/detail/epilepsy.

[2] *BrainCapture – Diagnosing Epilepsy*. URL: https://braincapture.dk/.

[3] Jennifer A. Williams et al. "Smartphone EEG and remote online interpretation for children with epilepsy in the Republic of Guinea: Quality, characteristics, and practice implications". In: *Seizure* 71 (2019), pp. 93–99. ISSN: 15322688. DOI: 10.1016/j.seizure.2019.05.025.

[4] *10–20 system (EEG) - Wikipedia*. URL: https://en.wikipedia.org/wiki/10%E2%80%9320_system_(EEG).

[5] *A closer look at EEG — Epilepsy Society*. URL: https://epilepsysociety.org.uk/about-epilepsy/diagnosing-epilepsy/closer-look-eeg.

[6] *Electroencephalography - Wikipedia*. URL: https://en.wikipedia.org/wiki/Electroencephalography#Normal_activity.

[7] *EEG — Epilepsy Foundation*. URL: https://www.epilepsy.com/learn/diagnosis/eeg.

[8] Meysam Golmohammadi et al. "Automatic Analysis of EEGs Using Big Data and Hybrid Deep Learning Architectures". In: *Frontiers in Human Neuroscience* 13 (2019), p. 76. ISSN: 1662-5161. DOI: 10.3389/fnhum.2019.00076. URL: https://www.frontiersin.org/article/10.3389/fnhum.2019.00076/full.

[9] Nathan Stevenson and Anton Tokariev. "Automated EEG analysis for neonatal intensive care". In: *Encyclopedia of Biomedical Engineering*. Vol. 1-3. Elsevier, 2019, pp. 240–257. ISBN: 9780128051443. DOI: 10.1016/B978-0-12-801238-3.10889-X.

[10] Gabriele Gratton, Michael G.H. Coles, and Emanuel Donchin. "A new method for off-line removal of ocular artifact". In: *Electroencephalography and Clinical Neurophysiology* 55.4 (1983), pp. 468–484. ISSN: 00134694. DOI: 10.1016/0013-4694(83)90135-9.

[11] Jose Antonio Urigüen and Begoña Garcia-Zapirain. "EEG artifact removal - State-of-the-art and guidelines". In: *Journal of Neural Engineering* 12.3 (2015), p. 031001. ISSN: 17412552. DOI: 10.1088/1741-2560/12/3/031001. URL: https://iopscience-iop-org.proxy.findit.dtu.dk/article/10.1088/1741-2560/12/3/031001%20https://iopscience-iop-org.proxy.findit.dtu.dk/article/10.1088/1741-2560/12/3/031001/meta.

[12] Lisha Sun, Ying Liu, and Patch J. Beadle. "Independent component analysis of EEG signals". In: *Proceedings of the 2005 IEEE International Workshop on VLSI Design and Video Technology, IWVDVT 2005*. 2005, pp. 293–296. ISBN: 0780390067. DOI: 10.1109/iwvdvt.2005.1504590.

[13] H. Nolan, R. Whelan, and R. B. Reilly. "FASTER: Fully Automated Statistical Thresholding for EEG artifact Rejection". In: *Journal of Neuroscience Methods* 192.1 (2010), pp. 152–162. ISSN: 01650270. DOI: 10.1016/j.jneumeth.2010.07.015.

[14] A. Harati et al. "Improved EEG event classification using differential energy". In: *2015 IEEE Signal Processing in Medicine and Biology Symposium - Proceedings*. Institute of Electrical and Electronics Engineers Inc., 2016. ISBN: 9781509013500. DOI: 10.1109/SPMB.2015.7405421.

[15] Mikhail Tokovarov, Małgorzata Plechawska-Wójcik, and Monika Kaczorowska. "Multi-class classification of EEG spectral data for artifact detection". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Vol. 11509 LNAI. Springer Verlag, 2019, pp. 305–316. ISBN: 9783030209148. DOI: 10.1007/978-3-030-20915-5{_}28.

[16] Sangmin S. Lee, Kiwon Lee, and Guiyeom Kang. "EEG Artifact Removal by Bayesian Deep Learning ICA". In: *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS*. Vol. 2020-July. Institute of Electrical and Electronics Engineers Inc., 2020, pp. 932–935. ISBN: 9781728119908. DOI: 10.1109/EMBC44109.2020.9175785.

[17] David B. Stone et al. "Automatic removal of physiological artifacts in EEG: The optimized fingerprint method for sports science applications". In: *Frontiers in Human Neuroscience* 12 (2018). ISSN: 16625161. DOI: 10.3389/fnhum.2018.00096.

[18] Thea Radüntz et al. "Automated EEG artifact elimination by applying machine learning algorithms to ICA-based features". In: *Journal of Neural Engineering* 14.4 (2017). ISSN: 17412552. DOI: 10.1088/1741-2552/aa69d1.

[19] Guiyeom Kang et al. "T59. EEG artifacts removal using machine learning algorithms and independent component analysis". In: *Clinical Neurophysiology* 129 (2018), e24. ISSN: 13882457. DOI: 10.1016/j.clinph.2018.04.060.

[20] Subhrajit Roy. "Machine Learning for removing EEG artifacts: Setting the benchmark". In: (2019). URL: https://arxiv.org/abs/1903.07825.

[21] Elnaz Lashgari, Dehua Liang, and Uri Maoz. *Data augmentation for deep-learning-based electroencephalography*. 2020. DOI: 10.1016/j.jneumeth.2020.108885.

[22] Hongyi Zhang et al. "mixup: Beyond Empirical Risk Minimization". In: (2017). URL: http://arxiv.org/abs/1710.09412.

[23] Z Mousavi et al. "Deep convolutional neural network for classification of sleep stages from single-channel EEG signals". In: *Journal of Neuroscience Methods* 324 (2019), p. 108312. ISSN: 0165-0270. DOI: https://doi.org/10.1016/j.jneumeth.2019.108312. URL: https://www.sciencedirect.com/science/article/pii/S0165027019301700.

[24] M Deepa. "Investigating the performance improvement by sampling techniques in EEG data". In: 2010.

[25] Tanya Piplani, N Merrill, and J Chuang. "Faking it, Making it: Fooling and Improving Brain-Based Authentication with Generative Adversarial Networks". In: *2018 IEEE 9th International Conference on Biometrics Theory, Applications and Systems (BTAS)* (2018), pp. 1–7.

[26] Qiqi Zhang and Ying Liu. "Improving brain computer interface performance by data augmentation with conditional Deep Convolutional Generative Adversarial Networks". In: (2018). URL: https://arxiv.org/abs/1806.07108.

[27] Z Yin and J Zhang. "Cross-session classification of mental workload levels using EEG and an adaptive deep learning model." In: *Biomedical Signal Processing and ControlBiomedical Signal Processing and Control* 33 (2017), pp. 30–47.

[28] Z Yin and J Zhang. "Cross-subject recognition of operator functional states via EEG and switching deep belief networks with adaptive weights." In: *Neurocomputing* 260 (2017), pp. 349–366.

[29] Elham S Salama et al. "EEG-based emotion recognition using 3D convolutional neural networks". In: *Int. J. Adv. Comput. Sci. Appl* 9.8 (2018), pp. 329–337.

[30] AI HLEG. "Ethics guidelines for trustworthy AI". In: *European Commission* (2019).

[31] R. W. Thatcher et al. "Intelligence and EEG measures of information flow: Efficiency and homeostatic neuroplasticity". In: *Scientific Reports* 6 (2016). ISSN: 20452322. DOI: 10.1038/srep38890.

[32] Iyad Obeid and Joseph Picone. "The Temple University Hospital EEG Data Corpus". In: *Frontiers in Neuroscience* 10 (2016). DOI: 10.3389/fnins.2016.00196.

[33] *Makoto's preprocessing pipeline - SCCN*. URL: https://sccn.ucsd.edu/wiki/Makoto%27s_preprocessing_pipeline.

[34] Nima Bigdely-Shamlo et al. "The PREP pipeline: standardized preprocessing for large-scale EEG analysis". In: *Frontiers in Neuroinformatics* 9 (2015), p. 16. ISSN: 1662-5196. URL: https://www.frontiersin.org/article/10.3389/fninf.2015.00016.

[35] " Jesús Monge-Álvarez (2021). Time domain filtering VS Frequency domain filtering in images (https://www.mathworks.com/matlabcentral/fileexchange/51155-time-domain-filtering-vs-frequency-domain-filtering-in-images), MATLAB Central File Exchange. Retrieved April 5, 2021. " In: ().

[36] Bohdan Myroniv et al. "Analyzing User Emotions via Physiology Signals". In: *Data Science and Pattern Recognition* 2 (2017).

[37] *Overlap Percentage*. URL: https://spectraplus.com/DT_help/overlap_percentage.htm.

[38] Guillaume Lemaître, Fernando Nogueira, and Christos K. Aridas. "Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning". In: *Journal of Machine Learning Research* 18 (2017). ISSN: 15337928.

[39] Nitesh V Chawla et al. *SMOTE: Synthetic Minority Over-sampling Technique*. Tech. rep. 2002, pp. 321–357.

[40] Barrios et al. "Partial Discharge Classification Using Deep Learning Methods—Survey of Recent Progress". In: *Energies* 12 (2019), p. 2485. DOI: 10.3390/en12132485.

[41] Petros Xanthopoulos, Panos M Pardalos, and Theodore B Trafalis. "Linear Discriminant Analysis". eng. In: *Robust Data Mining*. Springer New York, 2012, pp. 27–33. ISBN: 9781441998774, 9781441998781. DOI: 10.1007/978-1-4419-9878-1{_}4,10.1007/978-1-4419-9878-1.

[42] J. Bergstra, D. Yamins, and D. D. Cox. "Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures". In: *30th International Conference on Machine Learning, ICML 2013*. PART 1. 2013.

[43] Thomas G. Dietterich. "Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms". In: *Neural Computation* 10.7 (1998). ISSN: 08997667. DOI: 10.1162/089976698300017197.

[44] Sara Akodad et al. "Ensemble Learning Approaches Based on Covariance Pooling of CNN Features for High Resolution Remote Sensing Scene Classification". In: *Remote Sensing* 12 (2020). DOI: 10.3390/rs12203292.

# 5 Abbreviations

**AI** - Artificial Intelligence
**ANN** - Artificial Neural Network
**AR** - Average Reference
**ARa** - Average Reference (modified)
**AUC** - Area Under the Curve
**BSS** - Blind Source Separation
**CNN** - Convolutional Neural Network
**CV** - Cross-Validation
**D** - Discriminative Network
**EDF** - European Data Format (file)
**EEG** - Electroencephalography
**ERM** - Empirical Risk Minimization
**FFT** - Fast Fourier Transform
**FIR** - Finite Impulse Response
**FN** - False Negative
**FP** - False Positive
**G** - Generative Network
**GAN** - Generative Adversarial Network
**GNB** - Gaussian Naïve Bayes
**HIC** - Human In Command
**HOA** - Hyperparameter Optimization Algorithm
**HPC** - High-Performing Computer
**IC** - Independent Component
**ICA** - Independent Component Analysis
**KNN** - K-Nearest Neighbours
**LDA** - Linear Discriminant Analysis
**LE** - Linked Ear Reference
**LMIC** - Low and Middle Income Countries
**ML** - Machine Learning
**NEDC** - Neural Engineering Data Consortium
**PCA** - Principle Component Analysis
**PREP** - Early Stage Preprocessing
**RF** - Random Forest
**SD** - Standard Deviation
**SEM** - Standard Error of the Mean
**SMOTE** - Synthetic Minority Oversampling Technique
**SVM** - Support Vector Machine
**TN** - True Negative
**TP** - True Positive
**TPE** - Tree of Parzen Estimator
**TUAR** - TUH EEG Artifact Corpus
**TUEG** - TUH EEG Corpus
**TUH** - Temple University Hospital
**VRM** - Vicinal Risk Minimizarion
**WHO** - World Health Organization
**cDCGAN** - conditional Deep Convolutional Generative Adversarial Network

# 6 Appendix

## 6.1 A - List of dependencies used on DTU GBAR (hpc)

The packages and version of the modules in the requirements.txt-file installed at DTU GBAR virtual environment for this project.

- `matplotlib =3.3.4`
- `numpy =1.19.5`
- `tensorflow =2.4.1`
- `torch =1.8.0`
- `sklearn =0.0`
- `scikit-learn =0.24.1`
- `imblearn =0.0`
- `pandas =1.2.1`
- `mne =0.22.0`
- `scipy =1.6.0`
- `hyperopt =0.2.5`

## 6.2 B - List of hyperparameter for the classifiers

The hyperparameters for the various models, and the ranges of which they are optimized within.

- LR
  - `C`, floats in range $[log(0.00001), log(0.2)]$
- GNB
  - No parameters to optimize
- RF
  - `n_estimators`, integers in range $[1, 151]$
  - `criterion`, either of $['gini', 'entropy']$
  - `max_depth`, integers in range $[1, 76]$
- LDA
  - `solver`, either of $['svd', 'lsqr', 'eigen']$
- MLP
  - `hidden_layer_sizes`, integers in range $[1, 151]$
  - `learning_rate`, either of $['constant', 'adaptive']$
  - `alpha`, floats in range $[log(0.00001), log(0.01)]$
- SGD
  - `alpha`, floats in range $[log(0.00001), log(1)]$

## 6.3 C - Results from the SMOTE pilot experiment

### SMOTE = 0.5

**Table 12:** Mean performance (balanced accuracy) with std. deviation over 5-fold-CV.

| Algorithm | $BAcc._{eyem}$ | $BAcc._{chew}$ | $BAcc._{shiv}$ | $BAcc._{elpp}$ | $BAcc._{musc}$ | $BAcc._{null}$ | Avg. $BAcc.$ |
|---|---|---|---|---|---|---|---|
| baseline_perm | 49.99 ± 0.08 | 49.98 ± 0.06 | 50.01 ± 0.09 | 49.99 ± 0.06 | 50.11 ± 0.15 | 50.09 ± 0.07 | 50.03 ± 0.08 |
| LR | 75.0 ± 2.34 | 86.52 ± 2.05 | 68.53 ± 14.55 | 62.97 ± 4.85 | 69.62 ± 5.02 | 69.17 ± 2.7 | 71.97 ± 5.25 |
| GNB | 66.95 ± 2.52 | 75.9 ± 10.46 | 60.3 ± 16.43 | 61.31 ± 6.11 | 66.29 ± 5.76 | 43.04 ± 2.04 | 62.3 ± 7.22 |
| RF | 75.56 ± 1.69 | 51.88 ± 4.94 | 48.92 ± 4.61 | 52.71 ± 4.37 | 49.46 ± 1.19 | 38.37 ± 4.08 | 52.82 ± 3.48 |
| LDA | 75.02 ± 2.16 | 85.39 ± 1.87 | 63.21 ± 10.21 | 61.42 ± 4.22 | 68.11 ± 3.88 | 68.43 ± 2.7 | 70.26 ± 4.17 |
| MLP | 71.13 ± 3.88 | 50.11 ± 5.33 | 51.51 ± 6.81 | 51.59 ± 5.47 | 48.75 ± 1.37 | 40.13 ± 3.8 | 52.2 ± 4.44 |
| SGD | 74.52 ± 2.41 | 84.55 ± 3.45 | 63.56 ± 11.67 | 58.75 ± 4.19 | 67.52 ± 4.97 | 69.17 ± 2.49 | 69.68 ± 4.86 |

**Table 13:** Mean performance (Sensitivity) with std. deviation over 5-fold-CV.

| Algorithm | $S_{eyem}$ | $S_{chew}$ | $S_{shiv}$ | $S_{elpp}$ | $S_{musc}$ | $S_{null}$ | Avg. $S$ |
|---|---|---|---|---|---|---|---|
| baseline_perm | 2.44 ± 0.53 | 0.87 ± 0.94 | 0.48 ± 0.67 | 0.85 ± 0.68 | 1.56 ± 0.62 | 93.97 ± 1.23 | 16.7 ± 0.78 |
| LR | 70.8 ± 3.7 | 80.5 ± 5.86 | 46.95 ± 32.34 | 48.32 ± 12.33 | 59.67 ± 11.02 | 75.18 ± 3.56 | 63.57 ± 11.47 |
| GNB | 40.39 ± 5.01 | 56.62 ± 22.16 | 26.35 ± 36.27 | 31.16 ± 11.13 | 40.7 ± 12.58 | 40.39 ± 5.01 | 39.27 ± 15.36 |
| RF | 67.27 ± 3.33 | 67.32 ± 3.42 | 67.38 ± 3.62 | 67.24 ± 3.22 | 67.23 ± 3.56 | 67.0 ± 3.15 | 67.24 ± 3.38 |
| LDA | 71.05 ± 3.57 | 78.23 ± 5.42 | 35.68 ± 22.03 | 46.02 ± 11.49 | 57.7 ± 10.35 | 74.8 ± 3.44 | 60.58 ± 9.38 |
| MLP | 63.6 ± 7.68 | 64.4 ± 5.84 | 67.71 ± 6.14 | 63.41 ± 4.94 | 67.45 ± 4.75 | 66.57 ± 2.76 | 65.52 ± 5.35 |
| SGD | 70.73 ± 2.55 | 76.73 ± 8.16 | 37.48 ± 23.27 | 45.87 ± 8.31 | 57.2 ± 10.85 | 77.39 ± 3.08 | 60.9 ± 9.37 |

### SMOTE = 1

**Table 14:** Mean performance (balanced accuracy) with std. deviation over 5-fold-CV.

| Algorithm | $BAcc._{eyem}$ | $BAcc._{chew}$ | $BAcc._{shiv}$ | $BAcc._{elpp}$ | $BAcc._{musc}$ | $BAcc._{null}$ | Avg. $BAcc.$ |
|---|---|---|---|---|---|---|---|
| baseline_perm | 49.99 ± 0.08 | 49.98 ± 0.06 | 50.01 ± 0.09 | 49.99 ± 0.06 | 50.11 ± 0.15 | 50.09 ± 0.07 | 50.03 ± 0.08 |
| LR | 75.01 ± 2.23 | 86.18 ± 2.14 | 69.46 ± 15.27 | 62.35 ± 4.88 | 69.55 ± 5.02 | 69.1 ± 2.62 | 71.94 ± 5.36 |
| GNB | 64.78 ± 2.0 | 73.95 ± 12.17 | 59.81 ± 16.64 | 60.38 ± 5.3 | 62.59 ± 5.26 | 44.0 ± 1.95 | 60.92 ± 7.22 |
| RF | 75.04 ± 1.76 | 52.31 ± 5.22 | 48.96 ± 4.65 | 52.43 ± 4.45 | 49.89 ± 1.31 | 38.51 ± 4.0 | 52.86 ± 3.56 |
| LDA | 75.01 ± 2.08 | 85.17 ± 2.38 | 63.2 ± 9.96 | 61.26 ± 4.17 | 68.05 ± 3.84 | 68.3 ± 2.65 | 70.17 ± 4.18 |
| MLP | 73.43 ± 2.73 | 50.51 ± 2.77 | 48.19 ± 6.17 | 52.69 ± 5.33 | 47.95 ± 0.79 | 38.62 ± 3.44 | 51.9 ± 3.54 |
| SGD | 73.82 ± 1.77 | 86.29 ± 2.33 | 60.68 ± 9.05 | 61.29 ± 4.35 | 68.73 ± 4.37 | 69.17 ± 2.43 | 70.0 ± 4.05 |

**Table 15:** Mean performance (Sensitivity) with std. deviation over 5-fold-CV.

| Algorithm | $S_{eyem}$ | $S_{chew}$ | $S_{shiv}$ | $S_{elpp}$ | $S_{musc}$ | $S_{null}$ | Avg. $S$ |
|---|---|---|---|---|---|---|---|
| baseline_perm | 2.44 ± 0.53 | 0.87 ± 0.94 | 0.48 ± 0.67 | 0.85 ± 0.68 | 1.56 ± 0.62 | 93.97 ± 1.23 | 16.7 ± 0.78 |
| LR | 70.36 ± 3.61 | 79.61 ± 6.23 | 48.42 ± 33.54 | 47.24 ± 12.21 | 59.32 ± 11.12 | 75.31 ± 3.65 | 63.38 ± 11.73 |
| GNB | 34.79 ± 4.08 | 51.89 ± 25.4 | 24.47 ± 36.13 | 25.73 ± 10.18 | 30.9 ± 11.33 | 34.79 ± 4.08 | 33.76 ± 15.2 |
| RF | 66.13 ± 3.9 | 66.07 ± 4.0 | 66.31 ± 4.22 | 66.13 ± 4.21 | 66.78 ± 3.88 | 65.92 ± 3.76 | 66.22 ± 4.0 |
| LDA | 70.59 ± 3.41 | 77.68 ± 6.51 | 35.48 ± 21.5 | 45.4 ± 11.26 | 57.34 ± 10.29 | 75.01 ± 3.49 | 60.25 ± 9.41 |
| MLP | 67.18 ± 4.46 | 62.15 ± 3.87 | 65.61 ± 4.57 | 67.73 ± 4.5 | 65.81 ± 2.12 | 68.74 ± 4.2 | 66.2 ± 3.95 |
| SGD | 69.24 ± 3.45 | 80.09 ± 5.37 | 32.47 ± 21.77 | 48.56 ± 11.76 | 59.18 ± 9.42 | 76.57 ± 3.93 | 61.02 ± 9.28 |

### SMOTE = 1.5

**Table 16:** Mean performance (balanced accuracy) with std. deviation over 5-fold-CV.

| Algorithm | $BAcc._{eyem}$ | $BAcc._{chew}$ | $BAcc._{shiv}$ | $BAcc._{elpp}$ | $BAcc._{musc}$ | $BAcc._{null}$ | Avg. $BAcc.$ |
|---|---|---|---|---|---|---|---|
| baseline_perm | 49.99 ± 0.08 | 49.98 ± 0.06 | 50.01 ± 0.09 | 49.99 ± 0.06 | 50.11 ± 0.15 | 50.09 ± 0.07 | 50.03 ± 0.08 |
| LR | 75.01 ± 2.28 | 86.03 ± 2.59 | 69.46 ± 15.19 | 62.57 ± 4.65 | 69.37 ± 5.03 | 69.06 ± 2.57 | 71.92 ± 5.39 |
| GNB | 60.25 ± 1.22 | 72.6 ± 12.42 | 59.51 ± 16.35 | 60.0 ± 4.84 | 61.25 ± 4.03 | 45.91 ± 1.47 | 59.92 ± 6.72 |
| RF | 75.36 ± 2.01 | 53.9 ± 6.44 | 48.23 ± 4.08 | 52.34 ± 4.46 | 49.46 ± 1.1 | 38.27 ± 3.78 | 52.93 ± 3.64 |
| LDA | 75.01 ± 2.08 | 84.93 ± 2.43 | 63.13 ± 9.87 | 61.49 ± 3.99 | 68.02 ± 3.84 | 68.32 ± 2.6 | 70.15 ± 4.14 |
| MLP | 72.29 ± 3.75 | 51.67 ± 3.64 | 48.61 ± 5.44 | 51.72 ± 4.82 | 48.32 ± 1.06 | 39.97 ± 4.31 | 52.1 ± 3.84 |
| SGD | 73.89 ± 2.69 | 85.48 ± 2.31 | 62.56 ± 11.79 | 58.72 ± 6.7 | 67.97 ± 4.45 | 69.24 ± 2.46 | 69.64 ± 5.07 |

**Table 17:** Mean performance (Sensitivity) with std. deviation over 5-fold-CV.

| Algorithm | $S_{eyem}$ | $S_{chew}$ | $S_{shiv}$ | $S_{elpp}$ | $S_{musc}$ | $S_{null}$ | Avg. $S$ |
|---|---|---|---|---|---|---|---|
| baseline_perm | 2.44 ± 0.53 | 0.87 ± 0.94 | 0.48 ± 0.67 | 0.85 ± 0.68 | 1.56 ± 0.62 | 93.97 ± 1.23 | 16.7 ± 0.78 |
| LR | 70.12 ± 3.69 | 79.12 ± 7.25 | 48.45 ± 33.46 | 47.43 ± 11.85 | 59.02 ± 11.04 | 75.39 ± 3.64 | 63.26 ± 11.82 |
| GNB | 23.75 ± 2.98 | 48.46 ± 25.63 | 23.39 ± 35.09 | 22.39 ± 9.48 | 27.64 ± 8.28 | 23.75 ± 2.98 | 28.23 ± 14.07 |
| RF | 65.96 ± 3.74 | 65.96 ± 3.62 | 66.31 ± 3.81 | 65.66 ± 4.01 | 65.85 ± 3.7 | 65.59 ± 3.55 | 65.89 ± 3.74 |
| LDA | 70.38 ± 3.43 | 77.15 ± 6.59 | 35.2 ± 21.27 | 45.6 ± 10.92 | 57.19 ± 10.18 | 75.15 ± 3.5 | 60.11 ± 9.32 |
| MLP | 63.51 ± 7.71 | 64.32 ± 4.57 | 66.88 ± 5.37 | 63.78 ± 2.1 | 64.08 ± 4.56 | 65.98 ± 4.39 | 64.76 ± 4.78 |
| SGD | 69.05 ± 4.01 | 78.07 ± 6.0 | 36.0 ± 24.18 | 43.65 ± 12.5 | 58.67 ± 9.85 | 76.86 ± 3.52 | 60.38 ± 10.01 |

## SMOTE = 2

**Table 18:** Mean performance (balanced accuracy) with std. deviation over 5-fold-CV.

| Algorithm | $BAcc._{eyem}$ | $BAcc._{chew}$ | $BAcc._{shiv}$ | $BAcc._{elpp}$ | $BAcc._{musc}$ | $BAcc._{null}$ | Avg. $BAcc.$ |
|---|---|---|---|---|---|---|---|
| baseline_perm | 49.99 ± 0.08 | 49.98 ± 0.06 | 50.01 ± 0.09 | 49.99 ± 0.06 | 50.11 ± 0.15 | 50.09 ± 0.07 | 50.03 ± 0.08 |
| LR | 75.0 ± 2.27 | 85.88 ± 2.66 | 69.49 ± 15.23 | 62.48 ± 4.75 | 69.46 ± 5.09 | 69.01 ± 2.57 | 71.89 ± 5.43 |
| GNB | 55.56 ± 0.88 | 71.42 ± 12.36 | 59.31 ± 16.09 | 59.07 ± 4.26 | 61.41 ± 4.24 | 47.76 ± 0.93 | 59.09 ± 6.46 |
| RF | 75.27 ± 1.96 | 52.67 ± 5.5 | 50.05 ± 5.69 | 52.54 ± 4.6 | 49.78 ± 1.1 | 38.3 ± 3.88 | 53.1 ± 3.79 |
| LDA | 75.02 ± 2.07 | 84.87 ± 2.47 | 63.19 ± 9.96 | 61.43 ± 4.06 | 67.94 ± 3.87 | 68.27 ± 2.64 | 70.12 ± 4.18 |
| MLP | 72.48 ± 2.17 | 53.27 ± 4.34 | 50.27 ± 8.6 | 52.62 ± 4.83 | 48.7 ± 1.53 | 39.81 ± 4.07 | 52.86 ± 4.26 |
| SGD | 74.32 ± 1.78 | 85.03 ± 3.08 | 64.6 ± 10.53 | 58.09 ± 4.4 | 68.3 ± 4.13 | 69.19 ± 2.49 | 69.92 ± 4.4 |

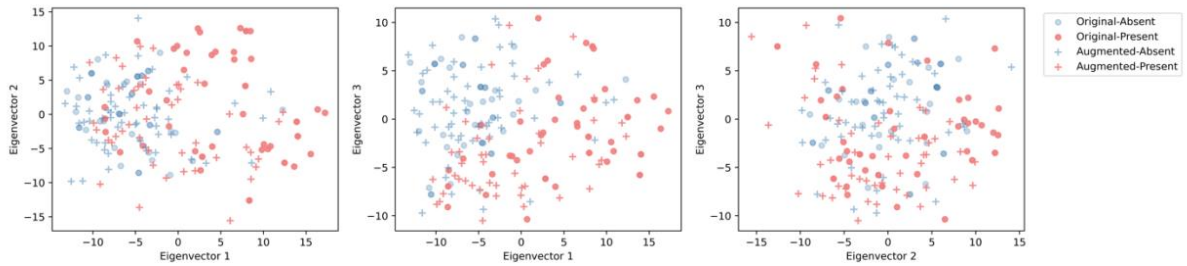
**Table 19:** Mean performance (Sensitivity) with std. deviation over 5-fold-CV.

| Algorithm | $S_{eyem}$ | $S_{chew}$ | $S_{shiv}$ | $S_{elpp}$ | $S_{musc}$ | $S_{null}$ | Avg. $S$ |
|---|---|---|---|---|---|---|---|
| baseline_perm | 2.44 ± 0.53 | 0.87 ± 0.94 | 0.48 ± 0.67 | 0.85 ± 0.68 | 1.56 ± 0.62 | 93.97 ± 1.23 | 16.7 ± 0.78 |
| LR | 70.02 ± 3.59 | 78.93 ± 7.4 | 48.4 ± 33.41 | 47.21 ± 11.89 | 58.73 ± 11.05 | 75.41 ± 3.7 | 63.12 ± 11.84 |
| GNB | 12.68 ± 2.21 | 45.59 ± 25.29 | 22.68 ± 34.3 | 19.51 ± 8.58 | 28.1 ± 8.25 | 12.68 ± 2.21 | 23.54 ± 13.47 |
| RF | 65.46 ± 3.89 | 65.79 ± 3.78 | 65.48 ± 4.02 | 65.77 ± 3.92 | 65.28 ± 3.73 | 65.78 ± 3.58 | 65.59 ± 3.82 |
| LDA | 70.25 ± 3.36 | 76.98 ± 6.64 | 35.25 ± 21.5 | 45.33 ± 10.99 | 56.94 ± 10.17 | 75.24 ± 3.57 | 60.0 ± 9.37 |
| MLP | 64.81 ± 4.69 | 66.06 ± 5.17 | 66.74 ± 4.01 | 62.52 ± 6.56 | 67.17 ± 4.12 | 63.49 ± 6.78 | 65.13 ± 5.22 |
| SGD | 69.45 ± 2.97 | 77.1 ± 7.16 | 39.03 ± 22.6 | 44.91 ± 10.26 | 58.57 ± 9.12 | 77.13 ± 3.36 | 61.03 ± 9.24 |

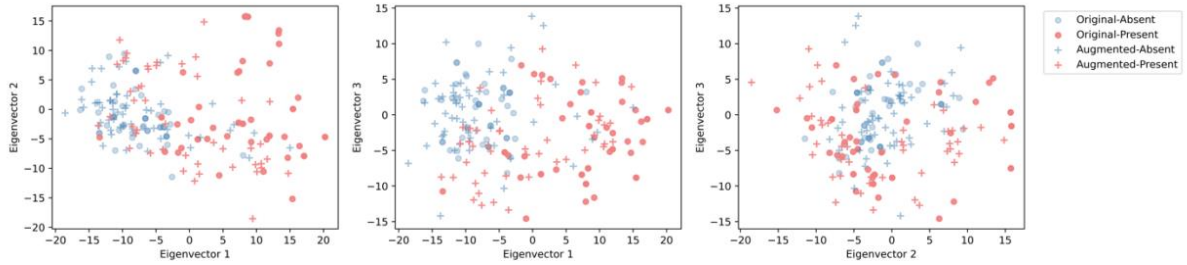
## 6.4 D - PCA plots

# Artifact: chew

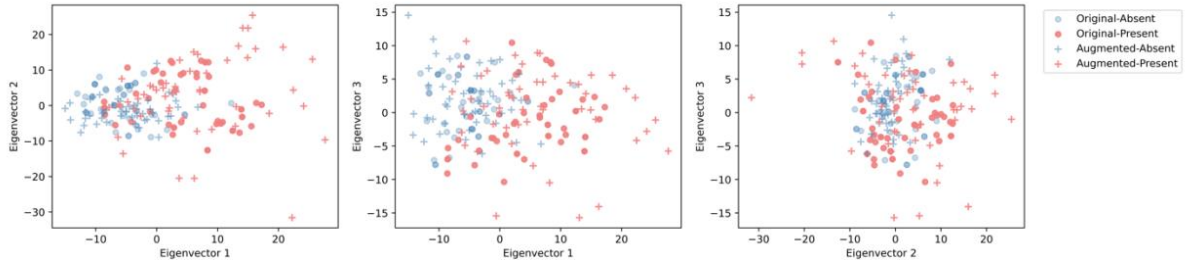
## method: Colored noise - under-sampling



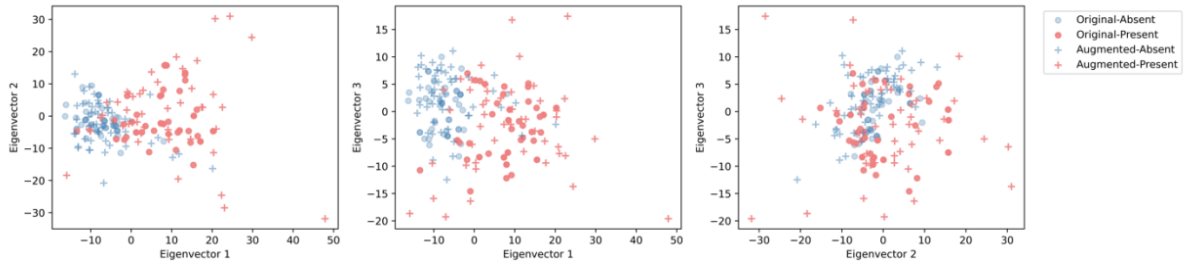

## method: Colored noise - over-sampling



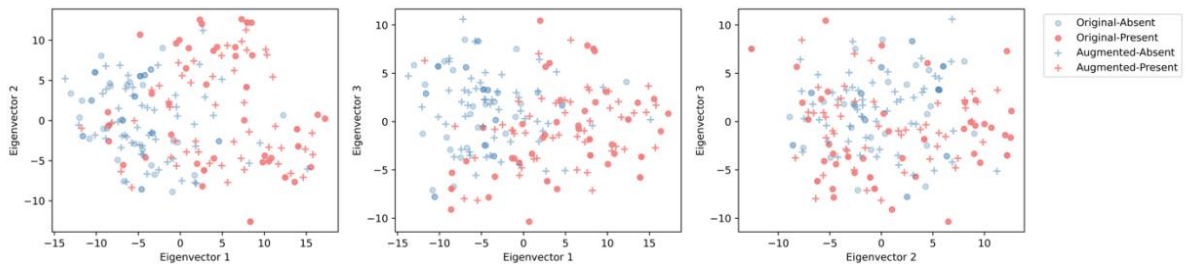

## method: GAN - under-sampling



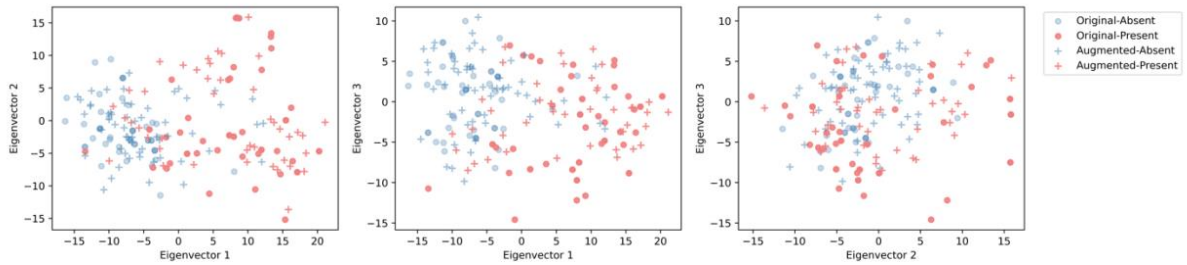

## method: GAN - over-sampling



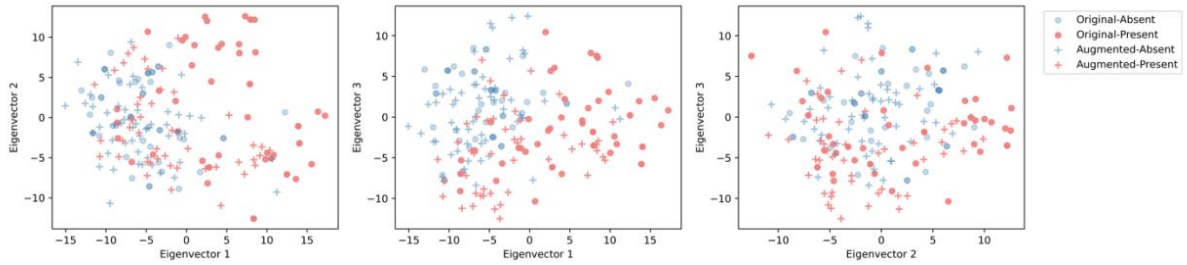
method: MixUp - under-sampling



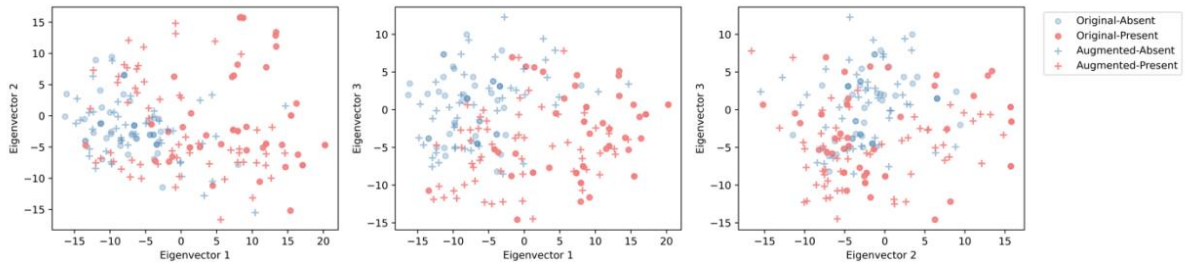

method: MixUp - over-sampling




method: White - under-sampling

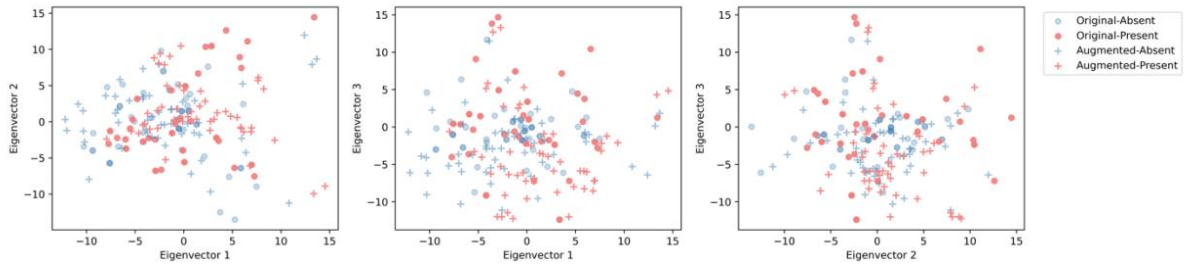



method: White - over-sampling

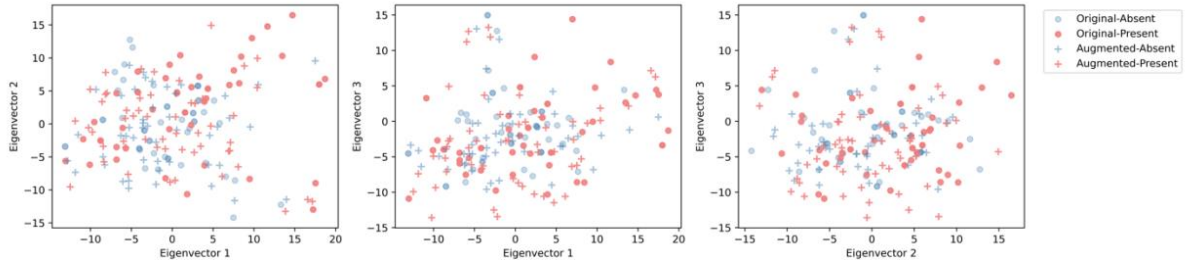



# Artifact: elpp

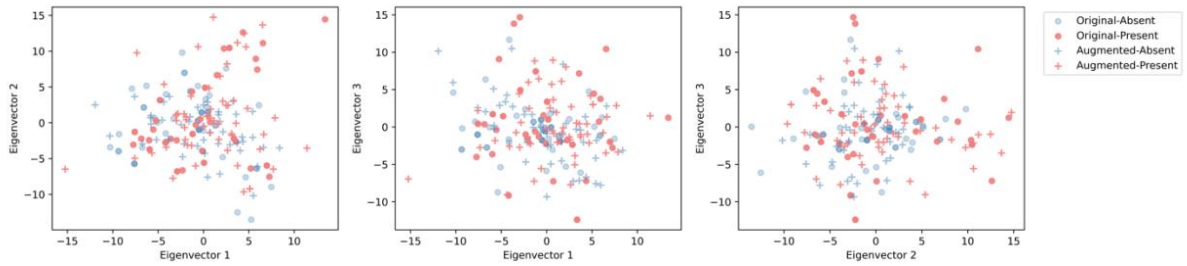
method: Colored noise - under-sampling



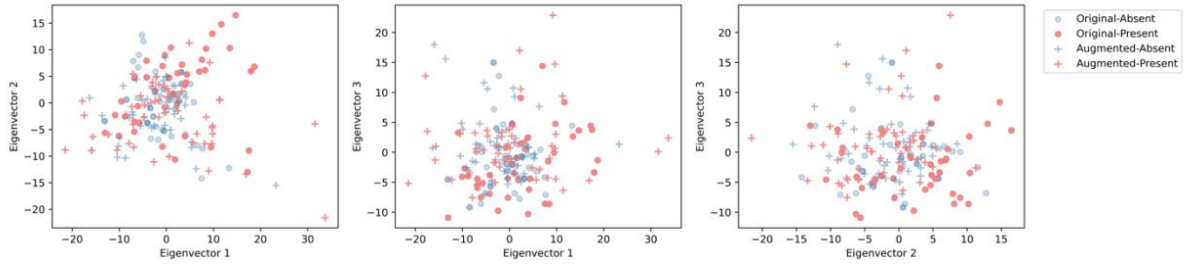

method: Colored noise - over-sampling



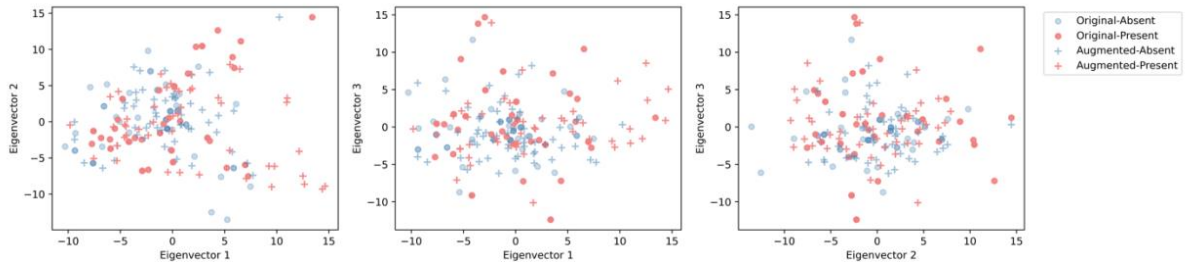

method: GAN - under-sampling



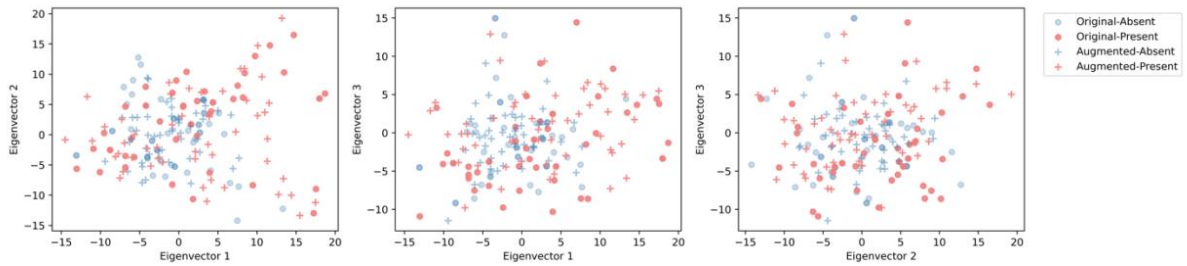

method: GAN - over-sampling



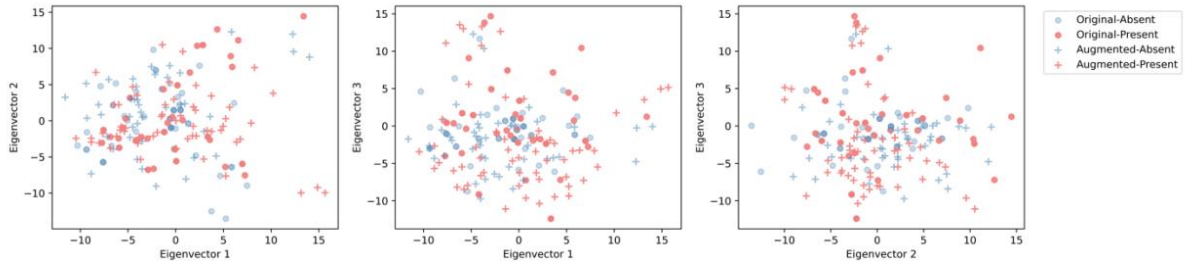

method: MixUp - under-sampling



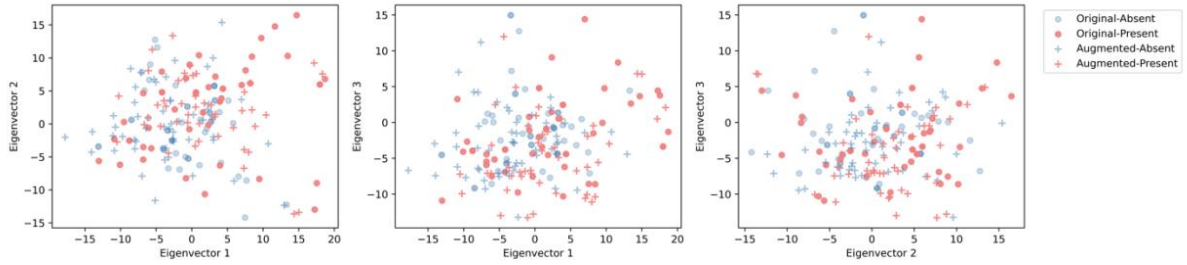

method: MixUp - over-sampling




method: White - under-sampling

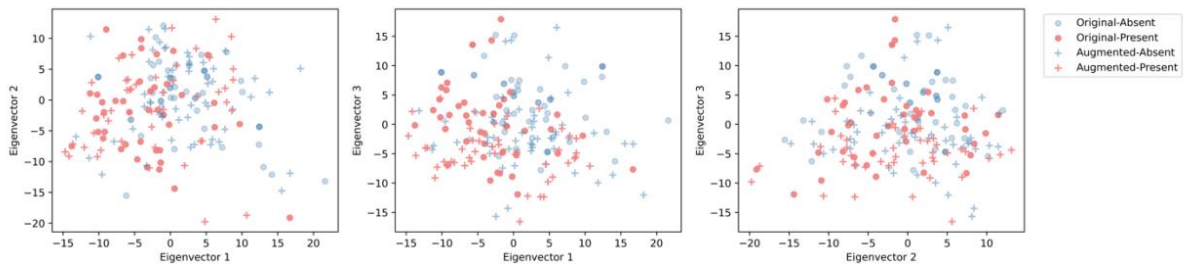



method: White - over-sampling

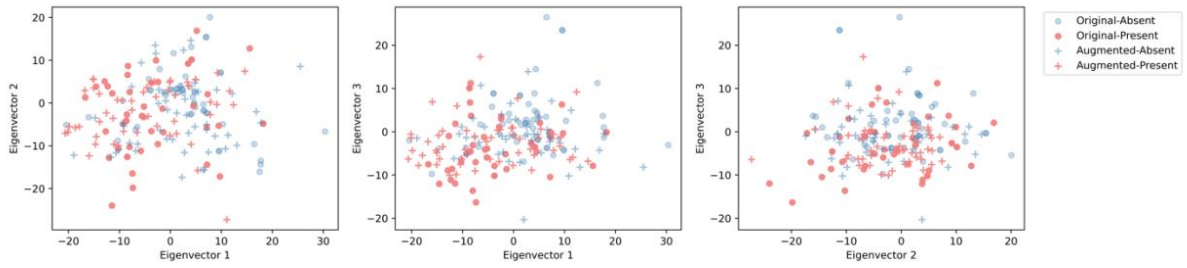



## Artifact: eyem

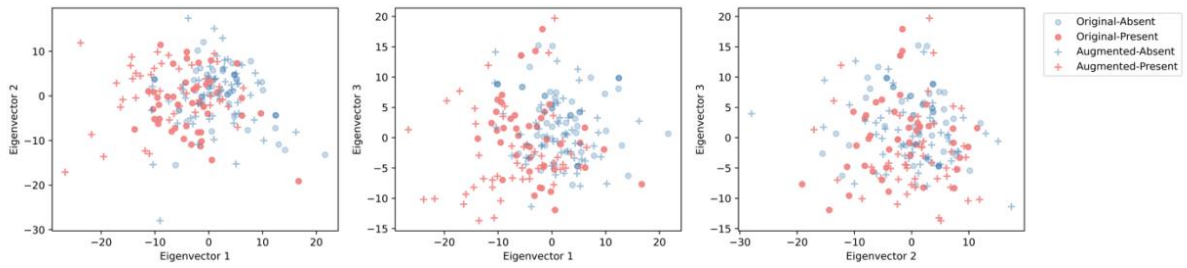
method: Colored noise - under-sampling



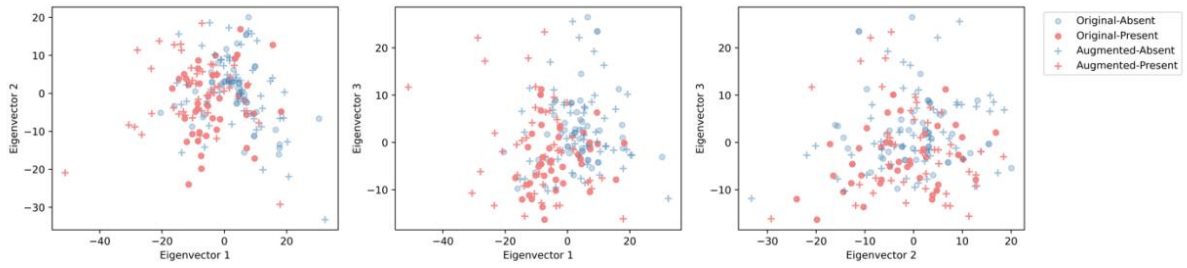

method: Colored noise - over-sampling



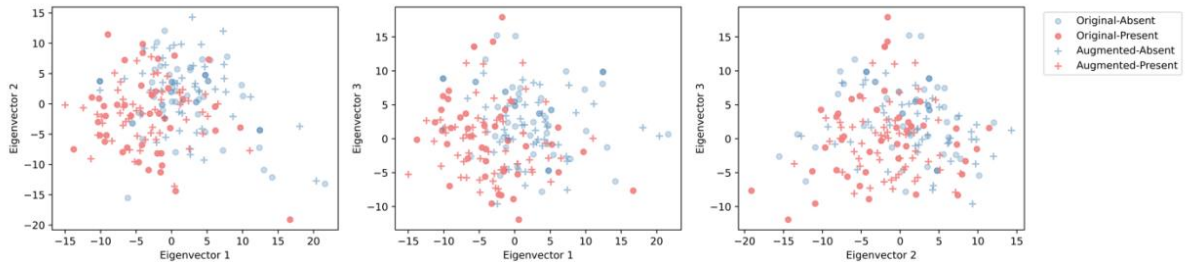

method: GAN - under-sampling



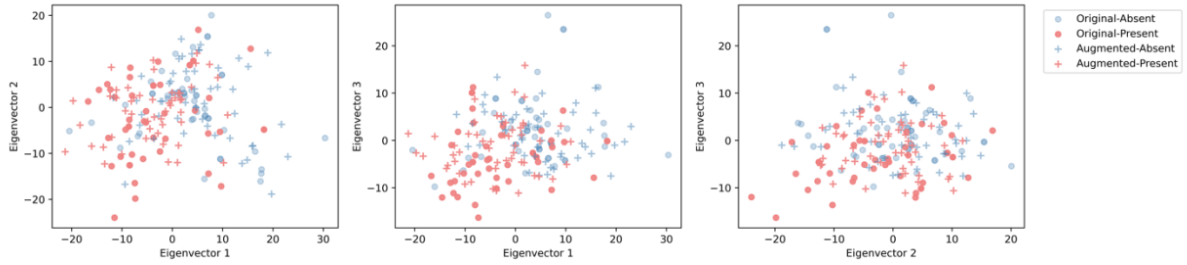

method: GAN - over-sampling



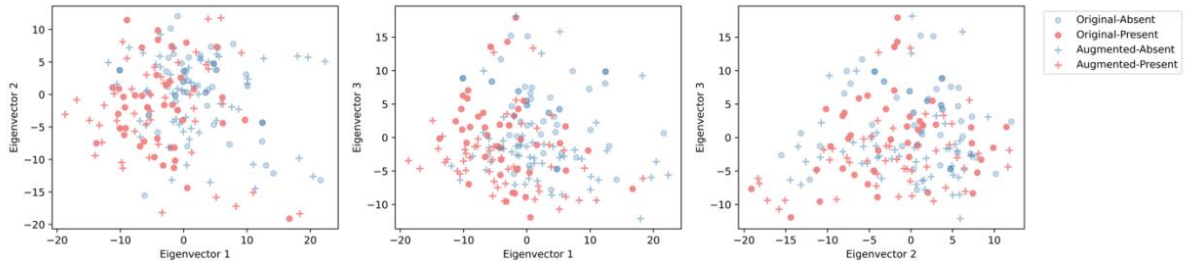

method: MixUp - under-sampling



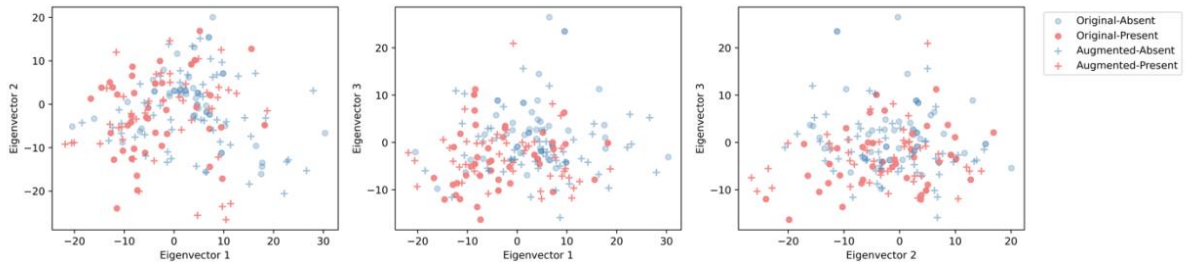

method: MixUp - over-sampling




method: White - under-sampling

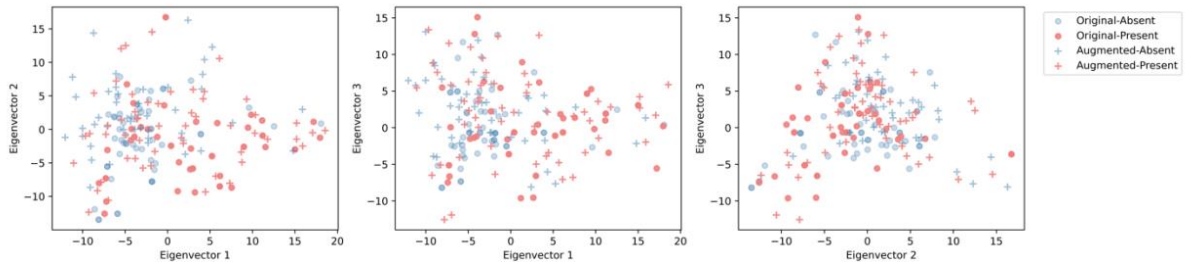



method: White - over-sampling

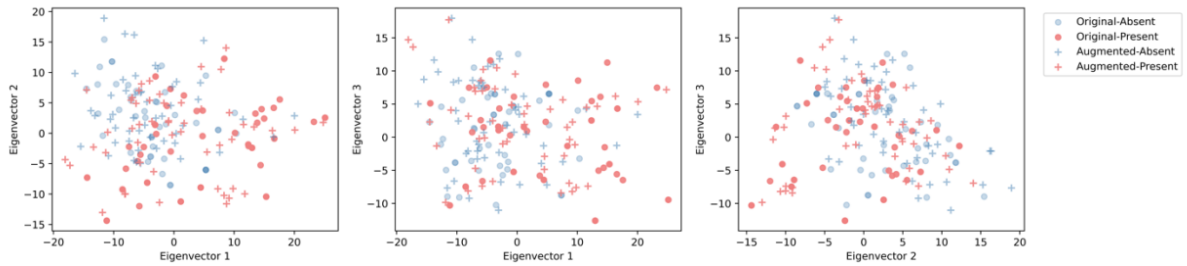



# Artifact: musc

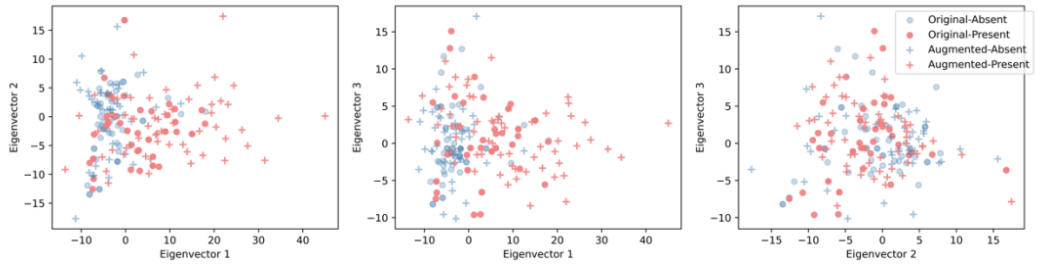
method: Colored noise - under-sampling



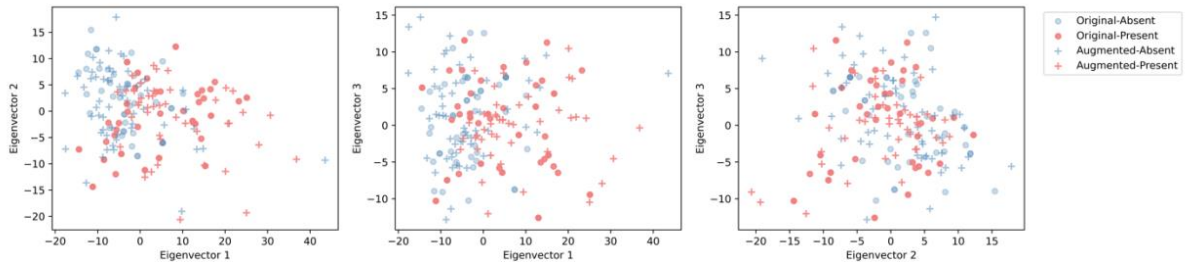

method: Colored noise - over-sampling



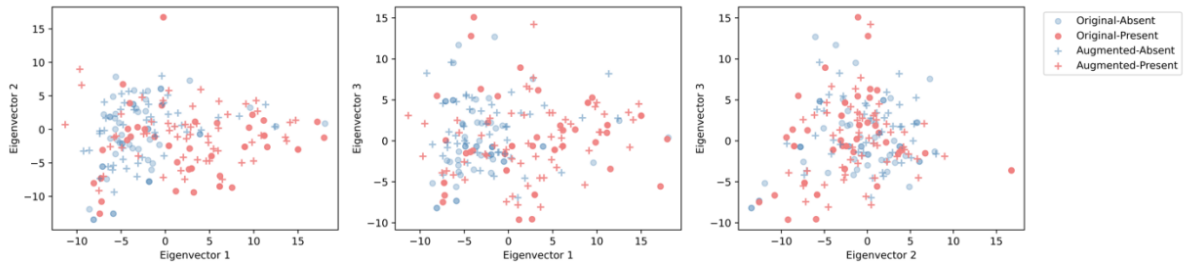

method: GAN - under-sampling



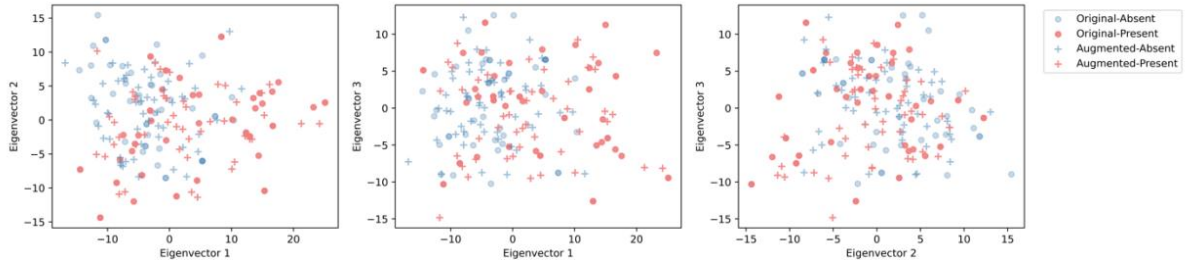

method: GAN - over-sampling



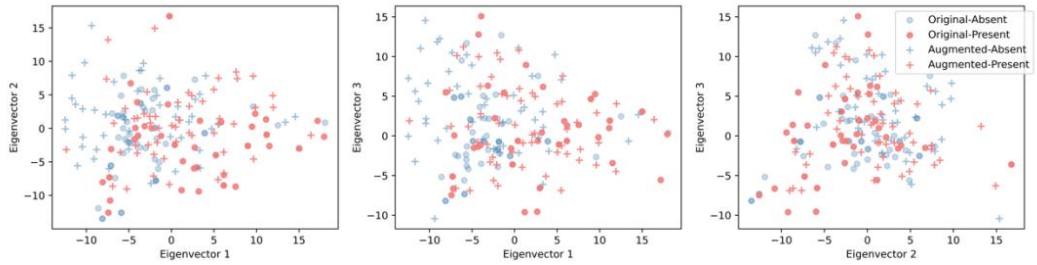

method: MixUp - under-sampling



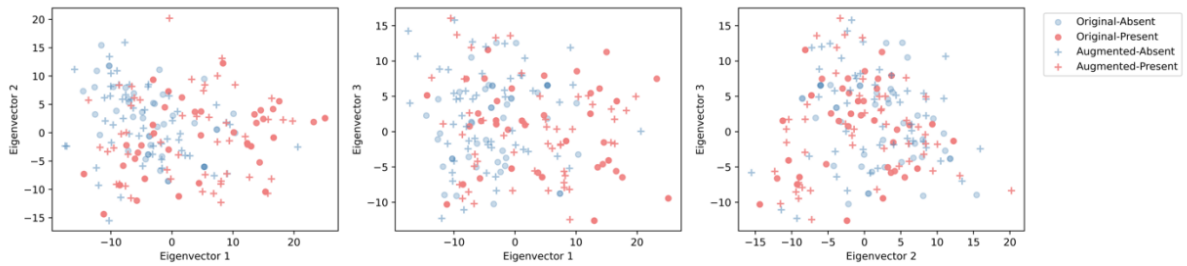

method: MixUp - over-sampling




method: White - under-sampling

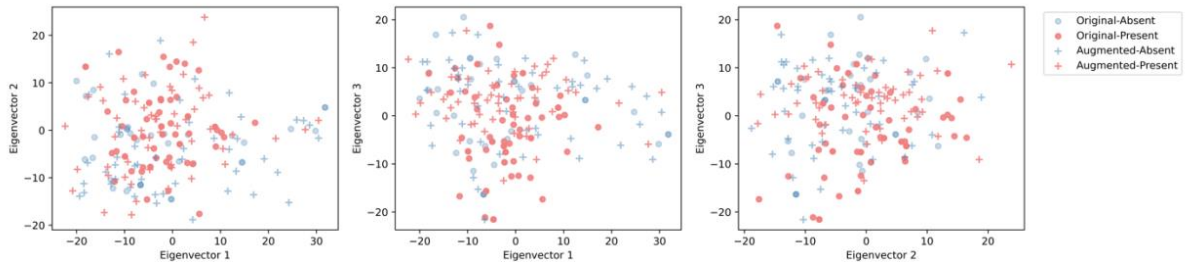



method: White - over-sampling

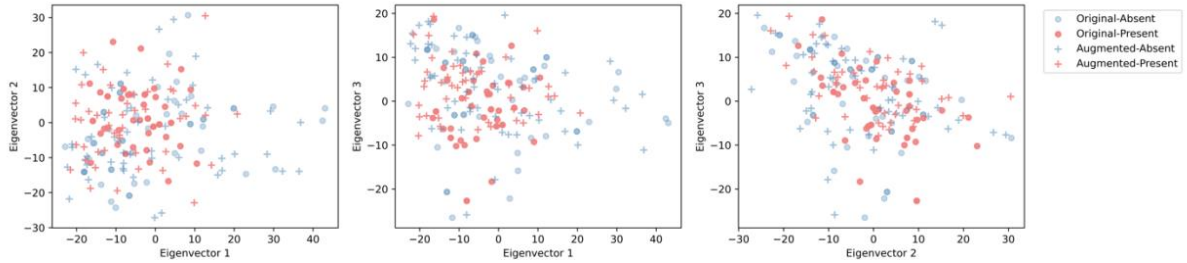



# Artifact: null

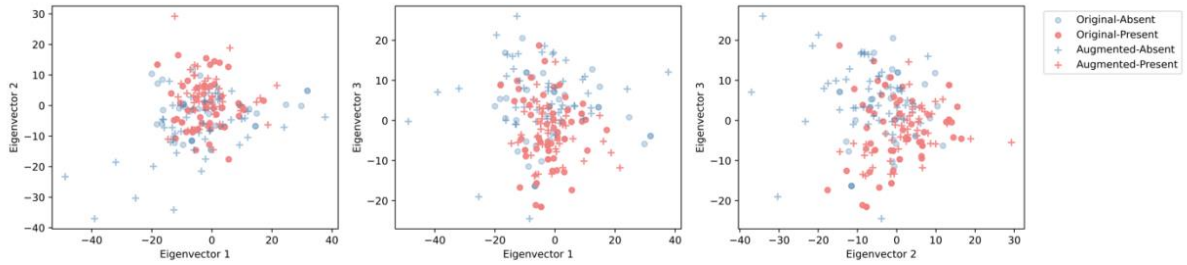
method: Colored noise - under-sampling



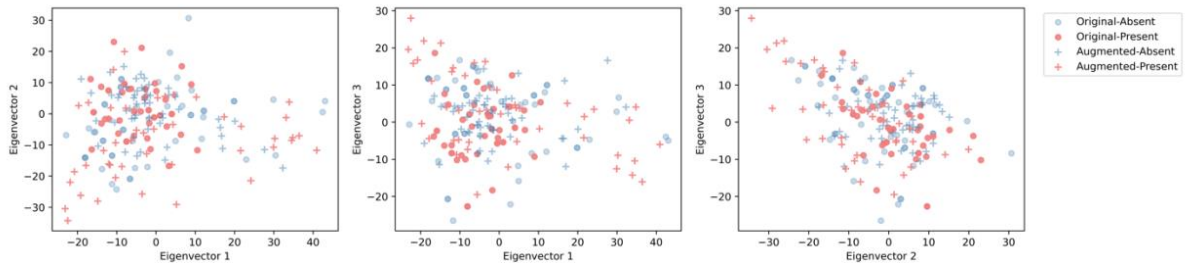

method: Colored noise - over-sampling



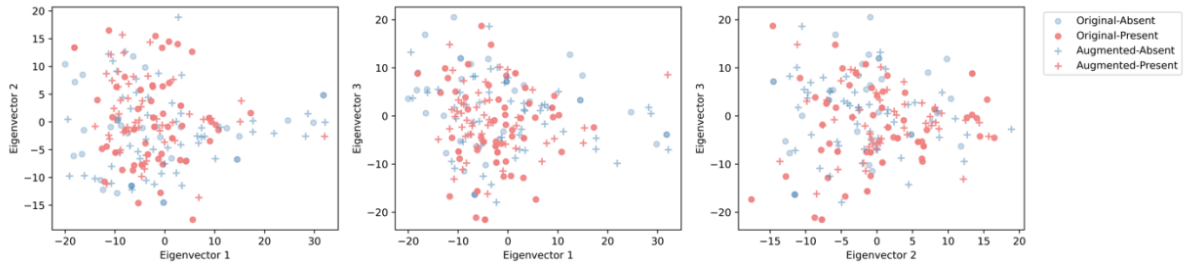

method: GAN - under-sampling



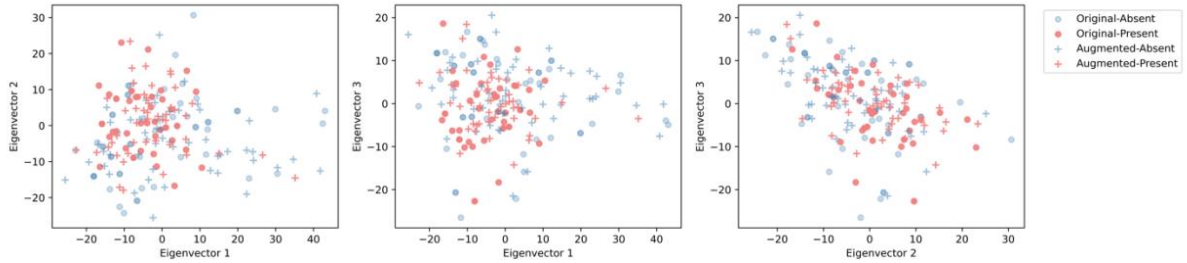

method: GAN - over-sampling



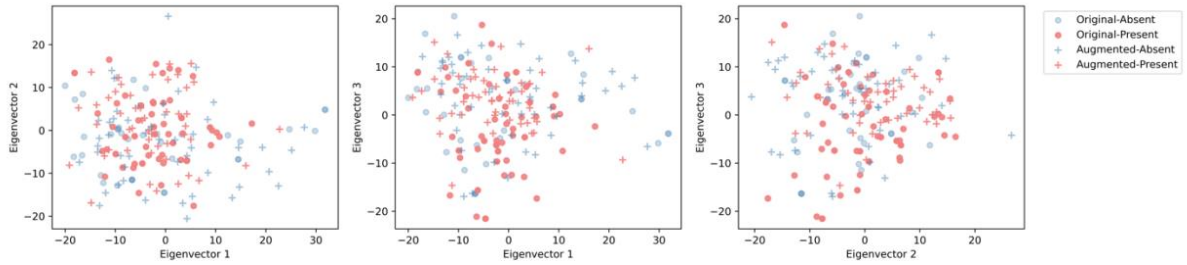

method: MixUp - under-sampling



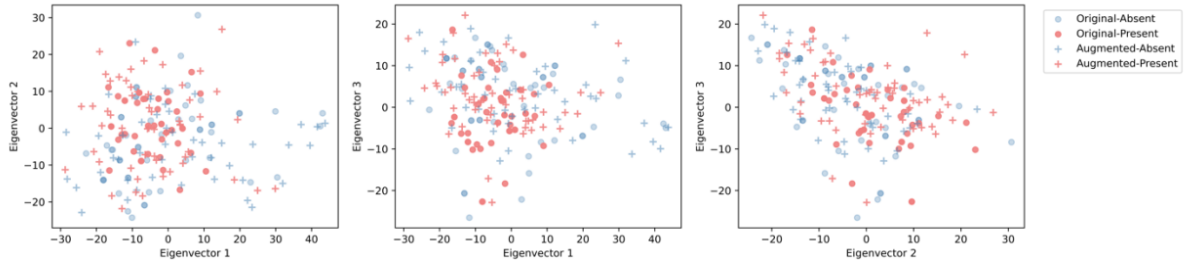

method: MixUp - over-sampling




method: White - under-sampling

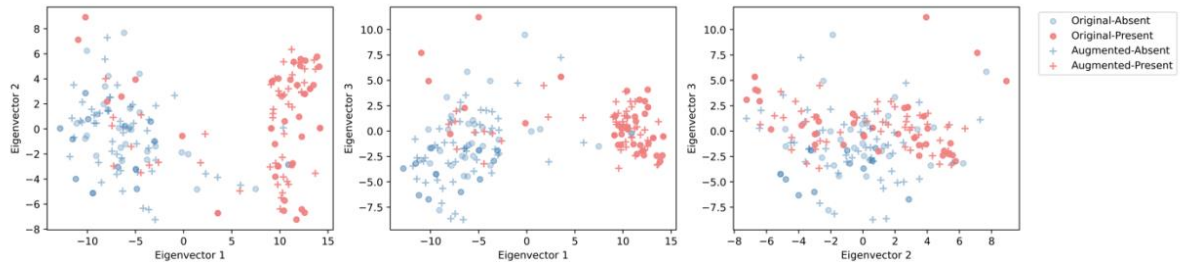



method: White - over-sampling

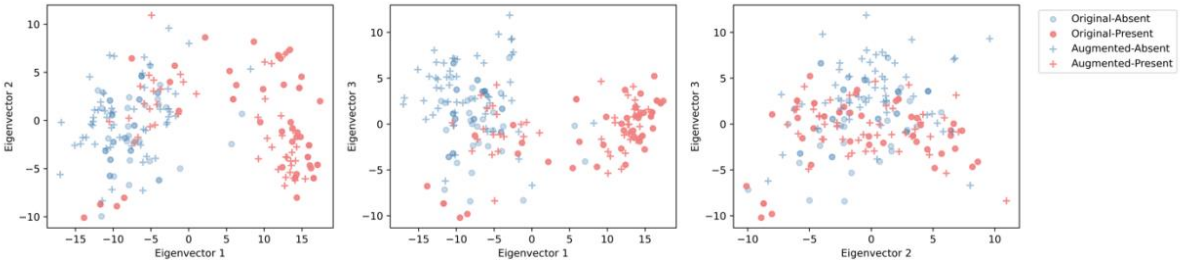



# Artifact: shiv

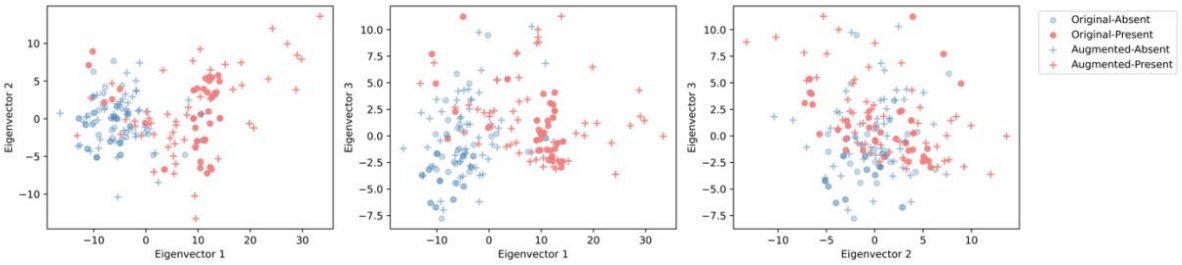
method: Colored noise - under-sampling



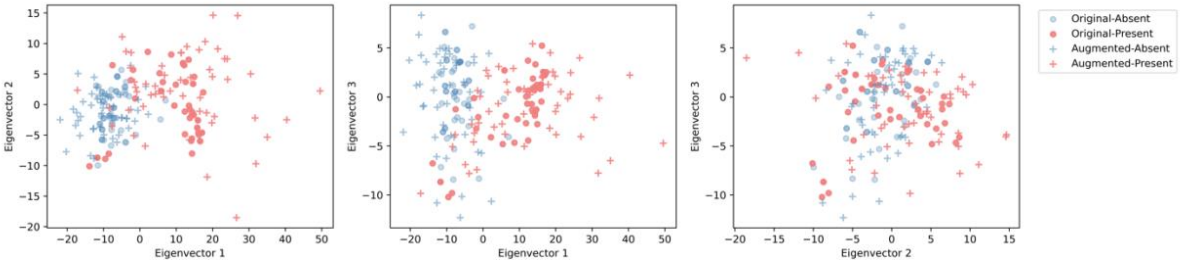

method: Colored noise - over-sampling



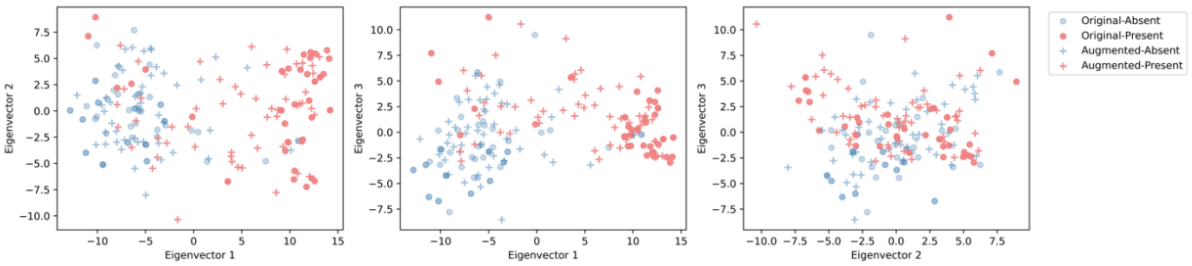

method: GAN - under-sampling



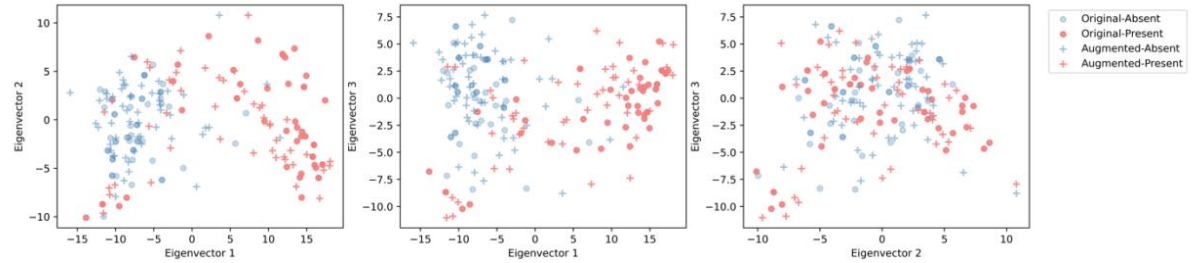

method: GAN - over-sampling



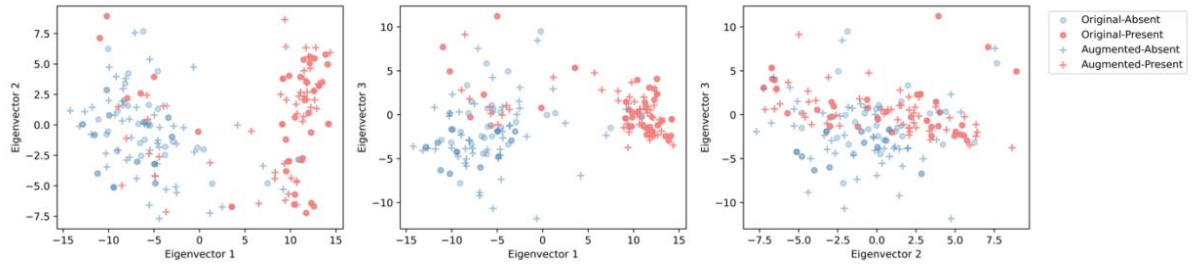

method: MixUp - under-sampling



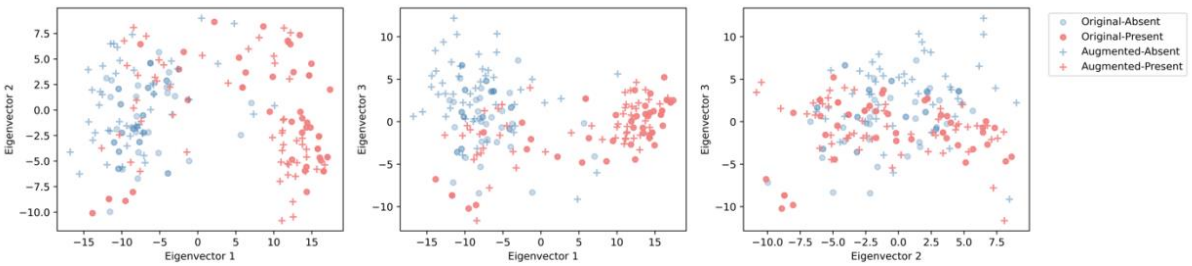

method: MixUp - over-sampling




method: White - under-sampling




method: White - over-sampling



## 6.5 E - Results from Improvement experiment

### SMOTE = 0, Balanced accuracy

### Balanced accuracy

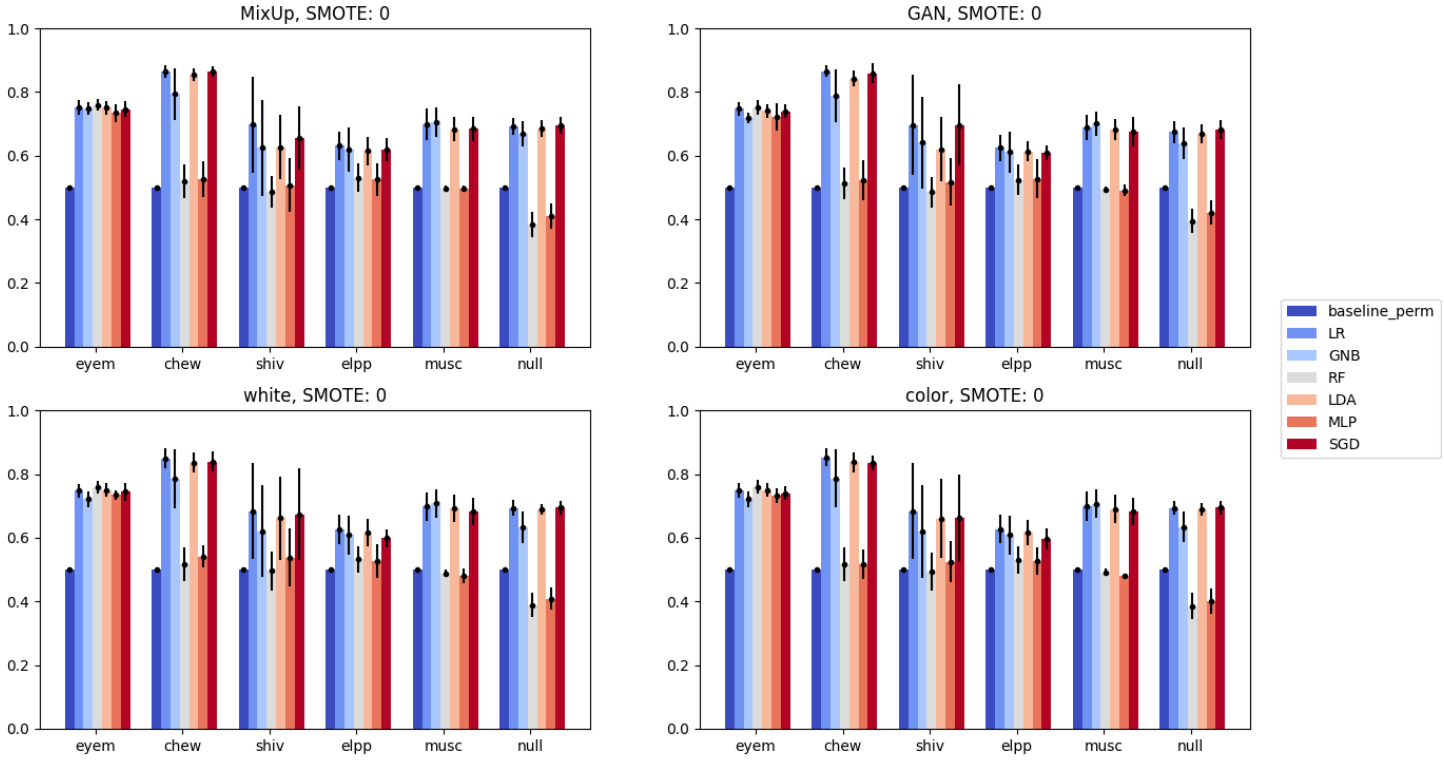



**Figure 29**

### MixUp

**Table 20:** Mean performance (balanced accuracy) with std. deviation over 5-fold-CV

| Algorithm | $BAcc._{eyem}$ | $BAcc._{chew}$ | $BAcc._{shiv}$ | $BAcc._{elpp}$ | $BAcc._{musc}$ | $BAcc._{null}$ | Avg. $BAcc.$ |
|---|---|---|---|---|---|---|---|
| baseline_perm | 49.99 ± 0.08 | 49.98 ± 0.06 | 50.01 ± 0.09 | 49.99 ± 0.06 | 50.11 ± 0.15 | 50.09 ± 0.07 | 50.03 ± 0.08 |
| LR | 75.14 ± 2.25 | 86.38 ± 2.12 | 69.69 ± 15.21 | 63.15 ± 4.52 | 69.83 ± 4.84 | 69.28 ± 2.7 | 72.24 ± 5.27 |
| GNB | 74.82 ± 1.96 | 79.34 ± 8.11 | 62.43 ± 15.14 | 62.02 ± 7.0 | 70.48 ± 4.67 | 66.73 ± 3.98 | 69.3 ± 6.81 |
| RF | 75.97 ± 1.97 | 51.99 ± 5.41 | 48.7 ± 4.97 | 53.11 ± 4.39 | 49.63 ± 1.11 | 38.34 ± 4.06 | 52.96 ± 3.65 |
| LDA | 74.99 ± 2.16 | 85.45 ± 1.9 | 62.7 ± 9.99 | 61.52 ± 4.45 | 68.24 ± 3.78 | 68.46 ± 2.68 | 70.23 ± 4.16 |
| MLP | 73.39 ± 2.73 | 52.73 ± 5.63 | 50.72 ± 8.49 | 52.55 ± 5.19 | 49.64 ± 0.86 | 40.94 ± 3.95 | 53.33 ± 4.48 |
| SGD | 74.54 ± 2.52 | 86.33 ± 1.73 | 65.61 ± 9.94 | 61.84 ± 3.72 | 68.38 ± 3.86 | 69.48 ± 2.57 | 71.03 ± 4.06 |

**Table 21:** Best augmentation ratios belonging to balanced accuracy-scores in **??**

| | baseline_perm | LR | GNB | RF | LDA | MLP | SGD |
|---|---|---|---|---|---|---|---|
| eyem | 0.5 | 0.5 | 0.5 | 1.0 | 0.5 | 0.5 | 1.5 |
| chew | 0.5 | 2.0 | 0.5 | 2.0 | 2.0 | 2.0 | 0.5 |
| shiv | 0.5 | 2.0 | 0.5 | 1.5 | 1.0 | 2.0 | 0.5 |
| elpp | 0.5 | 0.5 | 0.5 | 0.5 | 1.5 | 2.0 | 1.5 |
| musc | 0.5 | 1.0 | 0.5 | 1.5 | 1.0 | 1.0 | 2.0 |
| null | 0.5 | 1.0 | 0.5 | 1.5 | 1.0 | 1.0 | 1.0 |

### GAN

**Table 22:** Mean performance (balanced accuracy) with std. deviation over 5-fold-CV

| Algorithm | $BAcc._{eyem}$ | $BAcc._{chew}$ | $BAcc._{shiv}$ | $BAcc._{elpp}$ | $BAcc._{musc}$ | $BAcc._{null}$ | Avg. $BAcc.$ |
|---|---|---|---|---|---|---|---|
| baseline_perm | 49.99 ± 0.08 | 49.98 ± 0.06 | 50.01 ± 0.09 | 49.99 ± 0.06 | 50.11 ± 0.15 | 50.09 ± 0.07 | 50.03 ± 0.08 |
| LR | 74.68 ± 2.24 | 86.45 ± 1.82 | 69.6 ± 15.68 | 62.54 ± 4.12 | 68.97 ± 3.98 | 67.39 ± 3.56 | 71.6 ± 5.23 |
| GNB | 71.85 ± 1.79 | 78.92 ± 8.3 | 64.14 ± 14.42 | 61.14 ± 6.51 | 70.02 ± 3.95 | 63.94 ± 4.88 | 68.34 ± 6.64 |
| RF | 75.26 ± 2.33 | 51.37 ± 4.89 | 48.58 ± 4.78 | 52.44 ± 4.94 | 49.41 ± 1.05 | 39.44 ± 3.86 | 52.75 ± 3.64 |
| LDA | 74.03 ± 2.12 | 84.26 ± 2.34 | 62.03 ± 10.09 | 61.27 ± 3.12 | 68.14 ± 3.24 | 66.72 ± 3.0 | 69.41 ± 3.98 |
| MLP | 72.09 ± 4.29 | 52.23 ± 6.33 | 51.75 ± 7.49 | 52.76 ± 6.25 | 49.11 ± 1.69 | 42.14 ± 3.71 | 53.35 ± 4.96 |
| SGD | 73.95 ± 2.08 | 85.84 ± 3.17 | 69.58 ± 12.81 | 60.87 ± 2.19 | 67.43 ± 4.69 | 68.13 ± 2.9 | 70.97 ± 4.64 |

**Table 23:** Best augmentation ratios belonging to balanced accuracy-scores in **??**

| | baseline_perm | LR | GNB | RF | LDA | MLP | SGD |
|---|---|---|---|---|---|---|---|
| eyem | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 2.0 | 0.5 |
| chew | 0.5 | 1.0 | 0.5 | 1.5 | 0.5 | 1.5 | 0.5 |
| shiv | 0.5 | 0.5 | 0.5 | 1.5 | 1.0 | 2.0 | 1.5 |
| elpp | 0.5 | 0.5 | 0.5 | 1.5 | 1.5 | 1.5 | 1.0 |
| musc | 0.5 | 0.5 | 0.5 | 2.0 | 0.5 | 0.5 | 1.5 |
| null | 0.5 | 0.5 | 0.5 | 2.0 | 0.5 | 0.5 | 0.5 |

### White Noise

**Table 24:** Mean performance (balanced accuracy) with std. deviation over 5-fold-CV

| Algorithm | $BAcc._{eyem}$ | $BAcc._{chew}$ | $BAcc._{shiv}$ | $BAcc._{elpp}$ | $BAcc._{musc}$ | $BAcc._{null}$ | Avg. $BAcc.$ |
|---|---|---|---|---|---|---|---|
| baseline_perm | 49.99 ± 0.08 | 49.98 ± 0.06 | 50.01 ± 0.09 | 49.99 ± 0.06 | 50.11 ± 0.15 | 50.09 ± 0.07 | 50.03 ± 0.08 |
| LR | 74.72 ± 2.24 | 84.93 ± 3.17 | 68.37 ± 15.01 | 62.6 ± 4.49 | 69.74 ± 4.62 | 69.33 ± 2.48 | 71.61 ± 5.34 |
| GNB | 72.14 ± 2.42 | 78.57 ± 9.18 | 62.04 ± 14.41 | 60.84 ± 6.22 | 70.79 ± 4.53 | 63.26 ± 4.88 | 67.94 ± 6.94 |
| RF | 75.81 ± 2.07 | 51.64 ± 5.27 | 49.59 ± 6.04 | 53.27 ± 4.18 | 48.79 ± 1.11 | 38.86 ± 3.72 | 52.99 ± 3.73 |
| LDA | 74.96 ± 2.12 | 83.56 ± 3.2 | 66.11 ± 12.93 | 61.52 ± 4.33 | 69.26 ± 4.38 | 68.89 ± 1.77 | 70.72 ± 4.79 |
| MLP | 73.4 ± 1.58 | 54.15 ± 3.45 | 53.82 ± 9.11 | 52.62 ± 5.21 | 48.03 ± 2.22 | 40.9 ± 3.45 | 53.82 ± 4.17 |
| SGD | 74.47 ± 2.76 | 83.94 ± 3.25 | 67.31 ± 14.43 | 59.83 ± 2.82 | 68.37 ± 4.33 | 69.45 ± 2.27 | 70.56 ± 4.98 |

**Table 25:** Best augmentation ratios belonging to balanced accuracy-scores in **??**

| | baseline_perm | LR | GNB | RF | LDA | MLP | SGD |
|---|---|---|---|---|---|---|---|
| eyem | 0.5 | 0.5 | 0.5 | 1.5 | 0.5 | 1.0 | 0.5 |
| chew | 0.5 | 0.5 | 0.5 | 2.0 | 0.5 | 2.0 | 1.5 |
| shiv | 0.5 | 2.0 | 0.5 | 1.5 | 2.0 | 1.0 | 2.0 |
| elpp | 0.5 | 0.5 | 2.0 | 1.0 | 1.0 | 0.5 | 0.5 |
| musc | 0.5 | 1.5 | 2.0 | 2.0 | 1.5 | 1.5 | 1.0 |
| null | 0.5 | 0.5 | 0.5 | 2.0 | 1.5 | 2.0 | 0.5 |

### Colored Noise

**Table 26:** Mean performance (balanced accuracy) with std. deviation over 5-fold-CV

| Algorithm | $BAcc._{eyem}$ | $BAcc._{chew}$ | $BAcc._{shiv}$ | $BAcc._{elpp}$ | $BAcc._{musc}$ | $BAcc._{null}$ | Avg. $BAcc.$ |
|---|---|---|---|---|---|---|---|
| baseline_perm | 49.99 ± 0.08 | 49.98 ± 0.06 | 50.01 ± 0.09 | 49.99 ± 0.06 | 50.11 ± 0.15 | 50.09 ± 0.07 | 50.03 ± 0.08 |
| LR | 74.77 ± 2.29 | 85.2 ± 2.82 | 68.38 ± 14.99 | 62.77 ± 4.43 | 69.86 ± 4.6 | 69.37 ± 2.1 | 71.73 ± 5.21 |
| GNB | 72.16 ± 2.44 | 78.63 ± 9.07 | 62.0 ± 14.47 | 60.88 ± 6.21 | 70.73 ± 4.46 | 63.33 ± 4.8 | 67.96 ± 6.91 |
| RF | 75.96 ± 2.12 | 51.72 ± 5.32 | 49.38 ± 5.9 | 52.94 ± 4.25 | 49.12 ± 1.15 | 38.53 ± 4.12 | 52.94 ± 3.81 |
| LDA | 74.94 ± 2.19 | 83.67 ± 3.21 | 66.09 ± 12.56 | 61.58 ± 3.94 | 68.94 ± 4.45 | 68.88 ± 1.97 | 70.68 ± 4.72 |
| MLP | 73.15 ± 2.32 | 51.72 ± 4.66 | 52.45 ± 6.54 | 52.84 ± 4.31 | 47.89 ± 0.91 | 40.21 ± 3.93 | 53.04 ± 3.78 |
| SGD | 74.02 ± 2.11 | 83.49 ± 2.18 | 66.11 ± 13.77 | 59.71 ± 3.38 | 68.41 ± 4.3 | 69.46 ± 2.19 | 70.2 ± 4.66 |

**Table 27:** Best augmentation ratios belonging to balanced accuracy-scores in **??**

| | baseline_perm | LR | GNB | RF | LDA | MLP | SGD |
|---|---|---|---|---|---|---|---|
| eyem | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 1.0 | 1.5 |
| chew | 0.5 | 0.5 | 0.5 | 2.0 | 0.5 | 1.0 | 1.0 |
| shiv | 0.5 | 0.5 | 0.5 | 1.5 | 2.0 | 2.0 | 2.0 |
| elpp | 0.5 | 0.5 | 2.0 | 0.5 | 0.5 | 2.0 | 1.5 |
| musc | 0.5 | 2.0 | 2.0 | 2.0 | 1.5 | 2.0 | 1.0 |
| null | 0.5 | 1.0 | 0.5 | 2.0 | 1.0 | 1.5 | 0.5 |

### SMOTE = 0, Sensitivity

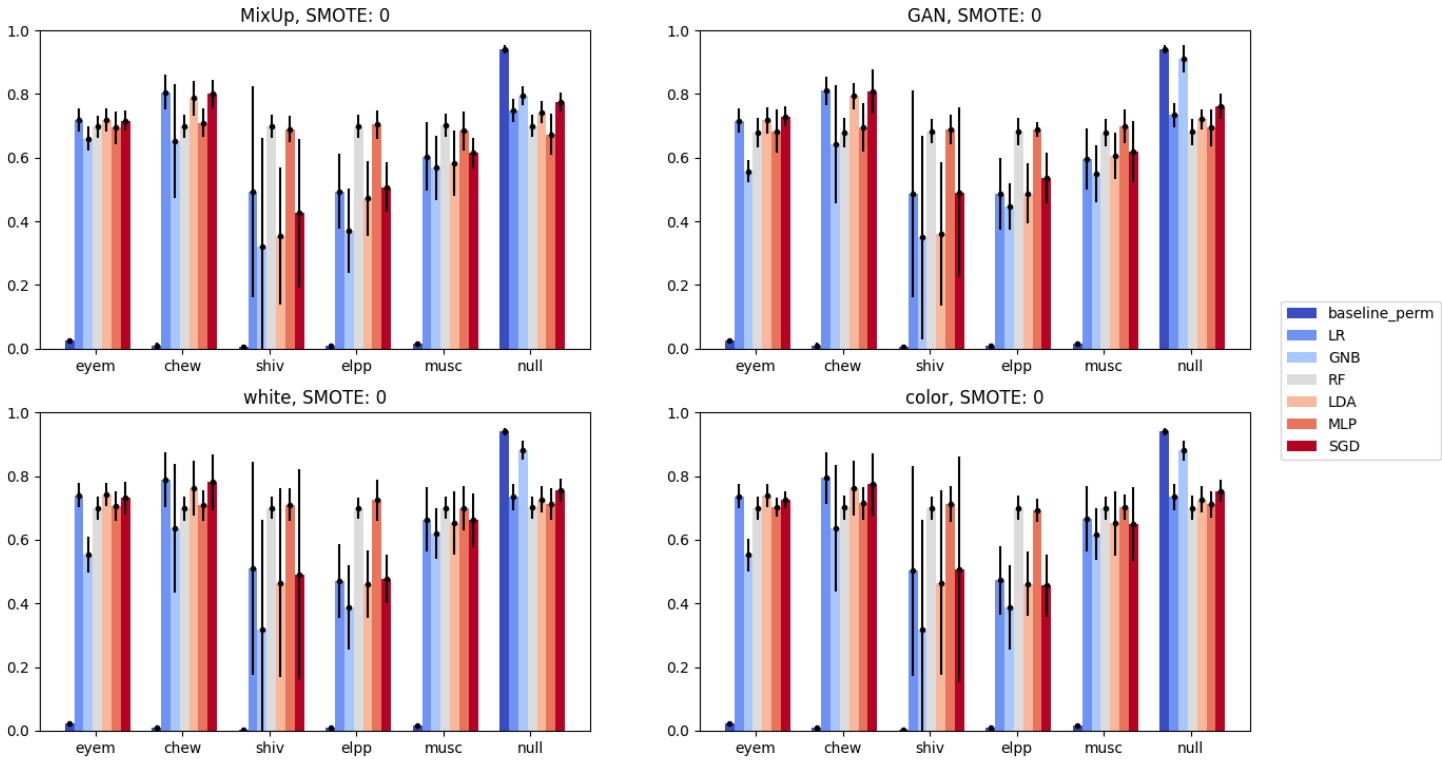



**Figure 30**

### MixUp

**Table 28:** Mean performance (sensitivity) with std. deviation over 5-fold-CV

| Algorithm | $BAcc._{eyem}$ | $BAcc._{chew}$ | $BAcc._{shiv}$ | $BAcc._{elpp}$ | $BAcc._{musc}$ | $BAcc._{null}$ | Avg. $BAcc.$ |
|---|---|---|---|---|---|---|---|
| baseline_perm | 2.44 ± 0.53 | 0.87 ± 0.94 | 0.48 ± 0.67 | 0.85 ± 0.68 | 1.56 ± 0.62 | 93.97 ± 1.23 | 16.7 ± 0.78 |
| LR | 71.78 ± 3.68 | 80.6 ± 5.61 | 49.24 ± 33.22 | 49.46 ± 11.61 | 60.29 ± 10.73 | 74.82 ± 3.55 | 64.36 ± 11.4 |
| GNB | 66.04 ± 3.83 | 65.2 ± 17.88 | 31.97 ± 34.16 | 37.14 ± 13.24 | 56.8 ± 10.02 | 79.47 ± 2.85 | 56.1 ± 13.66 |
| RF | 69.78 ± 3.48 | 69.85 ± 3.6 | 69.89 ± 3.66 | 69.87 ± 3.56 | 70.08 ± 3.69 | 69.94 ± 3.55 | 69.9 ± 3.59 |
| LDA | 71.85 ± 3.55 | 78.67 ± 5.38 | 35.51 ± 21.53 | 47.21 ± 11.73 | 58.24 ± 10.19 | 74.26 ± 3.48 | 60.96 ± 9.31 |
| MLP | 69.37 ± 5.02 | 70.99 ± 4.36 | 68.97 ± 4.07 | 70.36 ± 4.4 | 68.4 ± 6.05 | 67.3 ± 6.5 | 69.23 ± 5.07 |
| SGD | 71.61 ± 3.23 | 79.97 ± 4.55 | 42.52 ± 23.39 | 50.74 ± 7.74 | 61.43 ± 4.87 | 77.35 ± 3.02 | 63.94 ± 7.8 |

**Table 29:** Best augmentation ratios belonging to sensitivity-scores in 34

| | baseline_perm | LR | GNB | RF | LDA | MLP | SGD |
|---|---|---|---|---|---|---|---|
| eyem | 0.5 | 0.5 | 0.5 | 1.0 | 0.5 | 0.5 | 1.5 |
| chew | 0.5 | 1.5 | 0.5 | 2.0 | 2.0 | 2.0 | 0.5 |
| shiv | 0.5 | 2.0 | 0.5 | 2.0 | 1.0 | 1.0 | 2.0 |
| elpp | 0.5 | 0.5 | 0.5 | 2.0 | 1.5 | 1.0 | 1.0 |
| musc | 0.5 | 1.0 | 0.5 | 2.0 | 1.0 | 1.0 | 1.0 |
| null | 0.5 | 1.5 | 2.0 | 1.5 | 1.5 | 1.5 | 1.5 |

### GAN

**Table 30:** Mean performance (sensitivity) with std. deviation over 5-fold-CV

| Algorithm | $BAcc._{eyem}$ | $BAcc._{chew}$ | $BAcc._{shiv}$ | $BAcc._{elpp}$ | $BAcc._{musc}$ | $BAcc._{null}$ | Avg. $BAcc.$ |
|---|---|---|---|---|---|---|---|
| baseline_perm | 2.44 ± 0.53 | 0.87 ± 0.94 | 0.48 ± 0.67 | 0.85 ± 0.68 | 1.56 ± 0.62 | 93.97 ± 1.23 | 16.7 ± 0.78 |
| LR | 71.63 ± 3.77 | 80.98 ± 4.42 | 48.54 ± 32.46 | 48.59 ± 11.27 | 59.64 ± 9.66 | 73.35 ± 3.81 | 63.79 ± 10.9 |
| GNB | 55.72 ± 3.58 | 64.28 ± 18.59 | 34.91 ± 31.89 | 44.69 ± 7.28 | 54.85 ± 8.98 | 91.02 ± 4.35 | 57.58 ± 12.44 |
| RF | 67.87 ± 4.64 | 67.89 ± 4.62 | 68.31 ± 3.78 | 68.11 ± 4.24 | 67.96 ± 4.33 | 68.16 ± 4.12 | 68.05 ± 4.29 |
| LDA | 71.71 ± 4.16 | 79.32 ± 4.07 | 35.94 ± 22.53 | 48.78 ± 9.47 | 60.6 ± 7.24 | 72.0 ± 3.07 | 61.39 ± 8.42 |
| MLP | 68.32 ± 6.68 | 69.47 ± 7.72 | 68.86 ± 4.53 | 68.86 ± 2.28 | 69.89 ± 5.21 | 69.41 ± 5.88 | 69.14 ± 5.38 |
| SGD | 72.97 ± 3.26 | 80.84 ± 7.0 | 49.09 ± 26.73 | 53.69 ± 7.89 | 61.92 ± 9.46 | 76.08 ± 3.92 | 65.76 ± 9.71 |

**Table 31:** Best augmentation ratios belonging to sensitivity-scores in 34

| | baseline_perm | LR | GNB | RF | LDA | MLP | SGD |
|---|---|---|---|---|---|---|---|
| eyem | 0.5 | 1.5 | 0.5 | 0.5 | 0.5 | 2.0 | 2.0 |
| chew | 0.5 | 1.5 | 0.5 | 0.5 | 1.5 | 1.5 | 1.0 |
| shiv | 0.5 | 2.0 | 0.5 | 0.5 | 1.0 | 1.0 | 1.5 |
| elpp | 0.5 | 0.5 | 2.0 | 0.5 | 1.5 | 1.0 | 0.5 |
| musc | 0.5 | 0.5 | 0.5 | 0.5 | 2.0 | 0.5 | 2.0 |
| null | 0.5 | 2.0 | 2.0 | 0.5 | 0.5 | 2.0 | 0.5 |

### White Noise

**Table 32:** Mean performance (sensitivity) with std. deviation over 5-fold-CV

| Algorithm | $BAcc._{eyem}$ | $BAcc._{chew}$ | $BAcc._{shiv}$ | $BAcc._{elpp}$ | $BAcc._{musc}$ | $BAcc._{null}$ | Avg. $BAcc.$ |
|---|---|---|---|---|---|---|---|
| baseline_perm | 2.44 ± 0.53 | 0.87 ± 0.94 | 0.48 ± 0.67 | 0.85 ± 0.68 | 1.56 ± 0.62 | 93.97 ± 1.23 | 16.7 ± 0.78 |
| LR | 74.03 ± 3.85 | 78.77 ± 8.68 | 51.13 ± 33.47 | 47.03 ± 11.63 | 66.42 ± 10.17 | 73.45 ± 4.1 | 65.14 ± 11.98 |
| GNB | 55.27 ± 5.59 | 63.49 ± 20.19 | 31.79 ± 34.5 | 38.73 ± 13.13 | 61.96 ± 8.0 | 88.18 ± 3.07 | 56.57 ± 14.08 |
| RF | 69.92 ± 3.74 | 69.85 ± 3.79 | 70.07 ± 3.64 | 69.86 ± 3.4 | 70.03 ± 3.49 | 70.12 ± 3.52 | 69.98 ± 3.6 |
| LDA | 74.24 ± 3.71 | 76.13 ± 8.53 | 46.55 ± 29.49 | 45.97 ± 10.65 | 65.37 ± 9.95 | 72.69 ± 4.1 | 63.49 ± 11.07 |
| MLP | 70.5 ± 4.56 | 70.75 ± 4.88 | 70.97 ± 5.12 | 72.41 ± 6.39 | 69.96 ± 7.03 | 71.15 ± 4.9 | 70.96 ± 5.48 |
| SGD | 73.07 ± 5.08 | 78.08 ± 8.72 | 49.13 ± 33.07 | 47.75 ± 7.66 | 66.11 ± 8.54 | 75.56 ± 3.54 | 64.95 ± 11.1 |

**Table 33:** Best augmentation ratios belonging to sensitivity-scores in 34

| | baseline_perm | LR | GNB | RF | LDA | MLP | SGD |
|---|---|---|---|---|---|---|---|
| eyem | 0.5 | 2.0 | 1.0 | 1.5 | 2.0 | 2.0 | 2.0 |
| chew | 0.5 | 0.5 | 0.5 | 1.5 | 0.5 | 2.0 | 1.5 |
| shiv | 0.5 | 2.0 | 0.5 | 0.5 | 2.0 | 2.0 | 2.0 |
| elpp | 0.5 | 1.0 | 0.5 | 2.0 | 1.0 | 2.0 | 0.5 |
| musc | 0.5 | 2.0 | 2.0 | 1.0 | 2.0 | 2.0 | 2.0 |
| null | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 |

### Colored Noise

**Table 34:** Mean performance (sensitivity) with std. deviation over 5-fold-CV

| Algorithm | $BAcc._{eyem}$ | $BAcc._{chew}$ | $BAcc._{shiv}$ | $BAcc._{elpp}$ | $BAcc._{musc}$ | $BAcc._{null}$ | Avg. $BAcc.$ |
|---|---|---|---|---|---|---|---|
| baseline_perm | 2.44 ± 0.53 | 0.87 ± 0.94 | 0.48 ± 0.67 | 0.85 ± 0.68 | 1.56 ± 0.62 | 93.97 ± 1.23 | 16.7 ± 0.78 |
| LR | 73.68 ± 3.7 | 79.36 ± 8.06 | 50.2 ± 32.96 | 47.24 ± 10.89 | 66.69 ± 10.28 | 73.47 ± 4.07 | 65.11 ± 11.66 |
| GNB | 55.2 ± 5.15 | 63.61 ± 19.96 | 31.69 ± 34.57 | 38.72 ± 13.33 | 61.76 ± 8.19 | 88.11 ± 3.13 | 56.52 ± 14.06 |
| RF | 69.91 ± 3.79 | 70.1 ± 3.67 | 69.92 ± 3.8 | 70.05 ± 3.72 | 69.91 ± 3.66 | 70.03 ± 3.76 | 69.99 ± 3.73 |
| LDA | 73.87 ± 3.72 | 76.34 ± 8.65 | 46.4 ± 28.99 | 46.17 ± 10.12 | 65.13 ± 10.08 | 72.71 ± 4.02 | 63.44 ± 10.93 |
| MLP | 70.25 ± 2.92 | 71.48 ± 5.19 | 71.21 ± 5.52 | 69.17 ± 3.62 | 70.34 ± 3.92 | 71.11 ± 4.21 | 70.59 ± 4.23 |
| SGD | 72.59 ± 2.78 | 77.43 ± 9.83 | 50.56 ± 35.43 | 45.58 ± 9.89 | 64.98 ± 11.51 | 75.35 ± 3.43 | 64.42 ± 12.15 |

**Table 35:** Best augmentation ratios belonging to sensitivity-scores in 34

| | baseline_perm | LR | GNB | RF | LDA | MLP | SGD |
|---|---|---|---|---|---|---|---|
| eyem | 0.5 | 2.0 | 0.5 | 0.5 | 2.0 | 1.0 | 1.5 |
| chew | 0.5 | 0.5 | 0.5 | 2.0 | 0.5 | 2.0 | 2.0 |
| shiv | 0.5 | 2.0 | 0.5 | 0.5 | 2.0 | 2.0 | 1.5 |
| elpp | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 1.0 | 1.5 |
| musc | 0.5 | 2.0 | 2.0 | 0.5 | 2.0 | 1.5 | 1.5 |
| null | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 |

**SMOTE = 1, Balanced accuracy**

**Balanced accuracy**

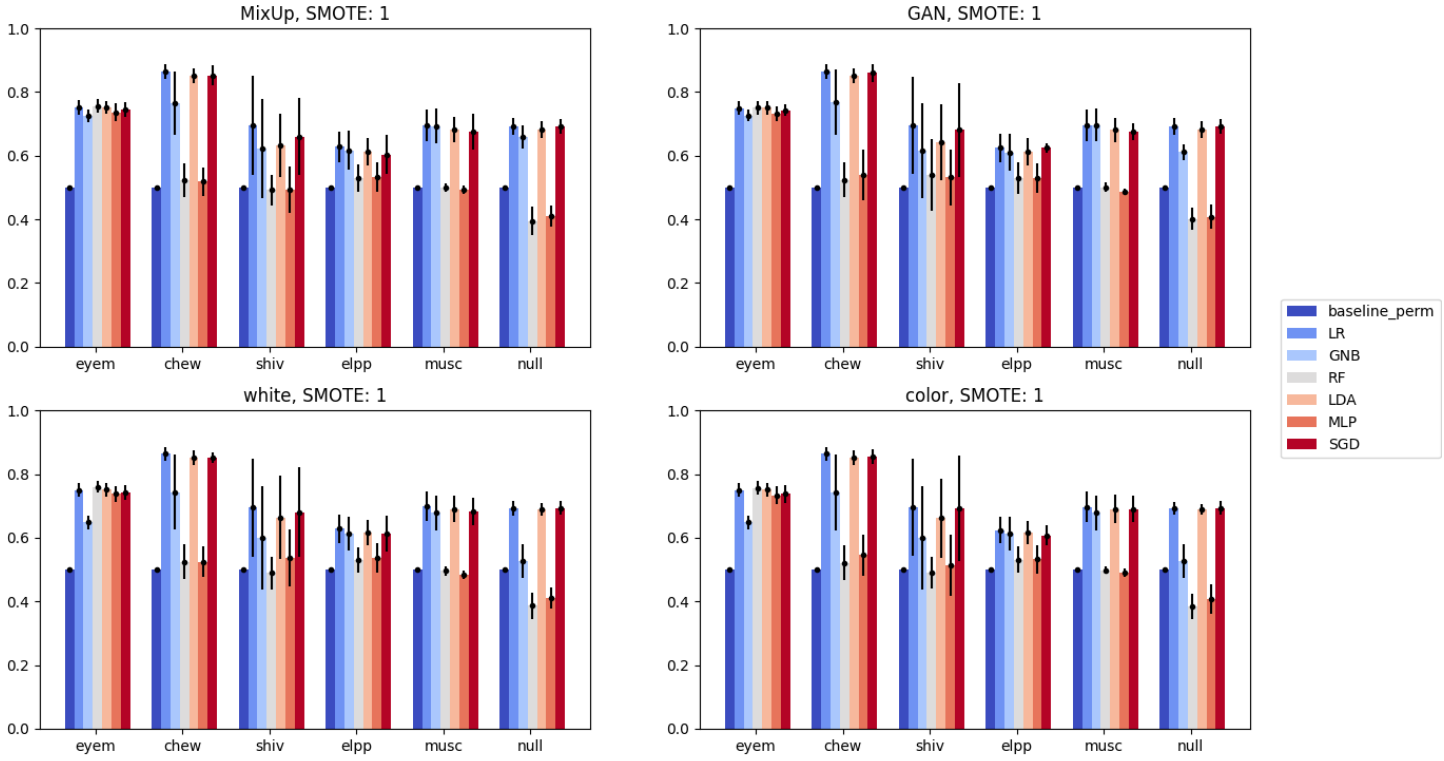



**Figure 31**

**MixUp**

**Table 36:** Mean performance (balanced accuracy) with std. deviation over 5-fold-CV

| Algorithm | $BAcc._{eyem}$ | $BAcc._{chew}$ | $BAcc._{shiv}$ | $BAcc._{elpp}$ | $BAcc._{musc}$ | $BAcc._{null}$ | Avg. $BAcc.$ |
|---|---|---|---|---|---|---|---|
| baseline_perm | 49.99 ± 0.08 | 49.98 ± 0.06 | 50.01 ± 0.09 | 49.99 ± 0.06 | 50.11 ± 0.15 | 50.09 ± 0.07 | 50.03 ± 0.08 |
| LR | 75.08 ± 2.25 | 86.21 ± 2.35 | 69.5 ± 15.54 | 62.76 ± 4.9 | 69.54 ± 5.14 | 69.06 ± 2.55 | 72.03 ± 5.45 |
| GNB | 72.58 ± 1.99 | 76.5 ± 9.96 | 62.14 ± 15.6 | 61.65 ± 6.06 | 69.28 ± 5.4 | 65.8 ± 3.65 | 67.99 ± 7.11 |
| RF | 75.52 ± 2.16 | 52.43 ± 5.29 | 49.18 ± 4.83 | 52.66 ± 4.56 | 49.99 ± 1.36 | 39.52 ± 4.62 | 53.22 ± 3.8 |
| LDA | 75.08 ± 2.07 | 85.11 ± 2.25 | 63.32 ± 9.99 | 61.28 ± 4.19 | 68.09 ± 4.03 | 68.31 ± 2.64 | 70.2 ± 4.2 |
| MLP | 73.63 ± 2.73 | 51.82 ± 4.36 | 49.28 ± 7.2 | 53.25 ± 4.74 | 48.7 ± 1.37 | 41.09 ± 3.3 | 52.96 ± 3.95 |
| SGD | 74.52 ± 2.21 | 85.07 ± 2.3 | 65.98 ± 12.17 | 60.32 ± 6.05 | 67.6 ± 5.67 | 69.24 ± 2.43 | 70.46 ± 5.14 |

**Table 37:** Best augmentation ratios belonging to balanced accuracy-scores in 36

| | baseline_perm | LR | GNB | RF | LDA | MLP | SGD |
|---|---|---|---|---|---|---|---|
| eyem | 0.5 | 2.0 | 0.5 | 0.5 | 1.5 | 0.5 | 1.5 |
| chew | 0.5 | 0.5 | 0.5 | 2.0 | 0.5 | 2.0 | 1.5 |
| shiv | 0.5 | 2.0 | 0.5 | 0.5 | 1.0 | 2.0 | 1.5 |
| elpp | 0.5 | 0.5 | 0.5 | 1.5 | 2.0 | 1.5 | 1.0 |
| musc | 0.5 | 0.5 | 0.5 | 0.5 | 2.0 | 1.0 | 1.0 |
| null | 0.5 | 0.5 | 0.5 | 1.0 | 2.0 | 0.5 | 0.5 |

**GAN**

**Table 38:** Mean performance (balanced accuracy) with std. deviation over 5-fold-CV

| Algorithm | $BAcc._{eyem}$ | $BAcc._{chew}$ | $BAcc._{shiv}$ | $BAcc._{elpp}$ | $BAcc._{musc}$ | $BAcc._{null}$ | Avg. $BAcc.$ |
|---|---|---|---|---|---|---|---|
| baseline_perm | 49.99 ± 0.08 | 49.98 ± 0.06 | 50.01 ± 0.09 | 49.99 ± 0.06 | 50.11 ± 0.15 | 50.09 ± 0.07 | 50.03 ± 0.08 |
| LR | 74.55 ± 2.14 | 85.99 ± 2.57 | 69.42 ± 15.25 | 62.42 ± 4.57 | 68.88 ± 4.49 | 68.39 ± 2.58 | 71.61 ± 5.27 |
| GNB | 72.57 ± 1.79 | 76.76 ± 10.24 | 61.57 ± 14.91 | 61.01 ± 5.71 | 69.67 ± 5.13 | 61.1 ± 2.37 | 67.11 ± 6.69 |
| RF | 74.26 ± 2.33 | 51.98 ± 4.91 | 53.85 ± 11.25 | 52.96 ± 5.0 | 50.03 ± 1.51 | 40.18 ± 3.62 | 53.88 ± 4.77 |
| LDA | 73.57 ± 1.93 | 83.55 ± 2.64 | 64.14 ± 11.98 | 60.66 ± 4.06 | 67.88 ± 3.21 | 66.55 ± 3.14 | 69.39 ± 4.49 |
| MLP | 73.09 ± 1.87 | 54.0 ± 7.85 | 53.14 ± 8.66 | 53.08 ± 4.62 | 48.62 ± 0.92 | 40.79 ± 3.88 | 53.79 ± 4.63 |
| SGD | 73.86 ± 2.34 | 85.94 ± 2.89 | 68.12 ± 14.73 | 62.48 ± 1.57 | 67.58 ± 2.74 | 67.61 ± 2.66 | 70.93 ± 4.49 |

**Table 39:** Best augmentation ratios belonging to balanced accuracy-scores in 38

| | baseline_perm | LR | GNB | RF | LDA | MLP | SGD |
|---|---|---|---|---|---|---|---|
| eyem | 0.5 | 0.5 | 0.5 | 1.0 | 0.5 | 2 | 0.5 |
| chew | 0.5 | 0.5 | 2.0 | 0.5 | 1.0 | 1 | 1.0 |
| shiv | 0.5 | 0.5 | 0.5 | 2.0 | 1.5 | 1 | 1.0 |
| elpp | 0.5 | 0.5 | 0.5 | 2.0 | 0.5 | 2 | 0.5 |
| musc | 0.5 | 2.0 | 0.5 | 0.5 | 0.5 | 2 | 1.0 |
| null | 0.5 | 0.5 | 0.5 | 1.5 | 0.5 | 1 | 1.0 |

### White Noise

**Table 40:** Mean performance (balanced accuracy) with std. deviation over 5-fold-CV

| Algorithm | $BAcc._{eyem}$ | $BAcc._{chew}$ | $BAcc._{shiv}$ | $BAcc._{elpp}$ | $BAcc._{musc}$ | $BAcc._{null}$ | Avg. $BAcc.$ |
|---|---|---|---|---|---|---|---|
| baseline_perm | 49.99 ± 0.08 | 49.98 ± 0.06 | 50.01 ± 0.09 | 49.99 ± 0.06 | 50.11 ± 0.15 | 50.09 ± 0.07 | 50.03 ± 0.08 |
| LR | 74.92 ± 2.23 | 84.59 ± 3.27 | 68.48 ± 14.59 | 62.8 ± 4.59 | 69.95 ± 4.72 | 69.27 ± 2.28 | 71.67 ± 5.28 |
| GNB | 61.59 ± 1.43 | 74.31 ± 11.74 | 60.05 ± 16.2 | 61.18 ± 5.35 | 67.77 ± 5.4 | 52.61 ± 5.21 | 62.92 ± 7.56 |
| RF | 75.98 ± 1.89 | 52.26 ± 5.44 | 48.95 ± 5.11 | 52.96 ± 3.92 | 49.59 ± 1.39 | 38.66 ± 4.12 | 53.07 ± 3.64 |
| LDA | 75.05 ± 2.19 | 83.08 ± 3.99 | 66.41 ± 12.98 | 61.63 ± 3.9 | 69.03 ± 4.25 | 68.91 ± 2.0 | 70.68 ± 4.88 |
| MLP | 73.8 ± 2.54 | 52.45 ± 4.83 | 53.46 ± 9.25 | 53.79 ± 4.59 | 48.29 ± 1.61 | 40.45 ± 4.18 | 53.71 ± 4.5 |
| SGD | 74.23 ± 2.19 | 85.16 ± 1.71 | 68.08 ± 14.03 | 61.26 ± 5.73 | 68.33 ± 4.25 | 69.32 ± 2.11 | 71.06 ± 5.0 |

**Table 41:** Best augmentation ratios belonging to balanced accuracy-scores in 40

| | baseline_perm | LR | GNB | RF | LDA | MLP | SGD |
|---|---|---|---|---|---|---|---|
| eyem | 0.5 | 0.5 | 0.5 | 1.5 | 0.5 | 1.0 | 1.0 |
| chew | 0.5 | 0.5 | 1.0 | 0.5 | 0.5 | 1.5 | 0.5 |
| shiv | 0.5 | 2.0 | 0.5 | 0.5 | 2.0 | 1.5 | 2.0 |
| elpp | 0.5 | 0.5 | 0.5 | 1.0 | 0.5 | 0.5 | 0.5 |
| musc | 0.5 | 1.5 | 0.5 | 0.5 | 1.5 | 0.5 | 2.0 |
| null | 0.5 | 0.5 | 1.5 | 2.0 | 1.0 | 2.0 | 0.5 |

### Color Noise

**Table 42:** Mean performance (balanced accuracy) with std. deviation over 5-fold-CV

| Algorithm | $BAcc._{eyem}$ | $BAcc._{chew}$ | $BAcc._{shiv}$ | $BAcc._{elpp}$ | $BAcc._{musc}$ | $BAcc._{null}$ | Avg. $BAcc.$ |
|---|---|---|---|---|---|---|---|
| baseline_perm | 49.99 ± 0.08 | 49.98 ± 0.06 | 50.01 ± 0.09 | 49.99 ± 0.06 | 50.11 ± 0.15 | 50.09 ± 0.07 | 50.03 ± 0.08 |
| LR | 74.93 ± 2.21 | 84.26 ± 3.97 | 68.1 ± 14.24 | 62.45 ± 4.0 | 69.72 ± 4.71 | 69.32 ± 2.01 | 71.46 ± 5.19 |
| GNB | 61.65 ± 1.51 | 74.24 ± 11.86 | 60.06 ± 16.19 | 61.24 ± 5.36 | 67.77 ± 5.4 | 52.61 ± 5.21 | 62.93 ± 7.59 |
| RF | 75.68 ± 2.28 | 52.13 ± 5.19 | 48.94 ± 5.0 | 53.09 ± 4.21 | 49.33 ± 1.09 | 38.37 ± 3.98 | 52.92 ± 3.62 |
| LDA | 75.1 ± 2.2 | 83.08 ± 3.94 | 66.23 ± 12.39 | 61.68 ± 3.54 | 69.0 ± 4.45 | 68.94 ± 1.65 | 70.67 ± 4.69 |
| MLP | 73.39 ± 2.72 | 53.04 ± 3.11 | 51.45 ± 9.6 | 53.29 ± 4.45 | 48.54 ± 0.82 | 40.8 ± 4.65 | 53.42 ± 4.23 |
| SGD | 73.77 ± 2.75 | 83.87 ± 3.85 | 69.2 ± 16.6 | 60.8 ± 3.15 | 69.07 ± 4.16 | 69.32 ± 2.15 | 71.0 ± 5.44 |

**Table 43:** Best augmentation ratios belonging to balanced accuracy-scores in 42

| | baseline_perm | LR | GNB | RF | LDA | MLP | SGD |
|---|---|---|---|---|---|---|---|
| eyem | 0.5 | 0.5 | 0.5 | 2.0 | 0.5 | 0.5 | 0.5 |
| chew | 0.5 | 0.5 | 1.0 | 1.0 | 0.5 | 0.5 | 0.5 |
| shiv | 0.5 | 1.0 | 0.5 | 1.0 | 2.0 | 0.5 | 1.0 |
| elpp | 0.5 | 1.0 | 0.5 | 0.5 | 0.5 | 1.5 | 1.5 |
| musc | 0.5 | 2.0 | 0.5 | 0.5 | 2.0 | 1.5 | 1.5 |
| null | 0.5 | 1.0 | 1.5 | 0.5 | 2.0 | 1.5 | 0.5 |

### SMOTE = 1, Sensitivity

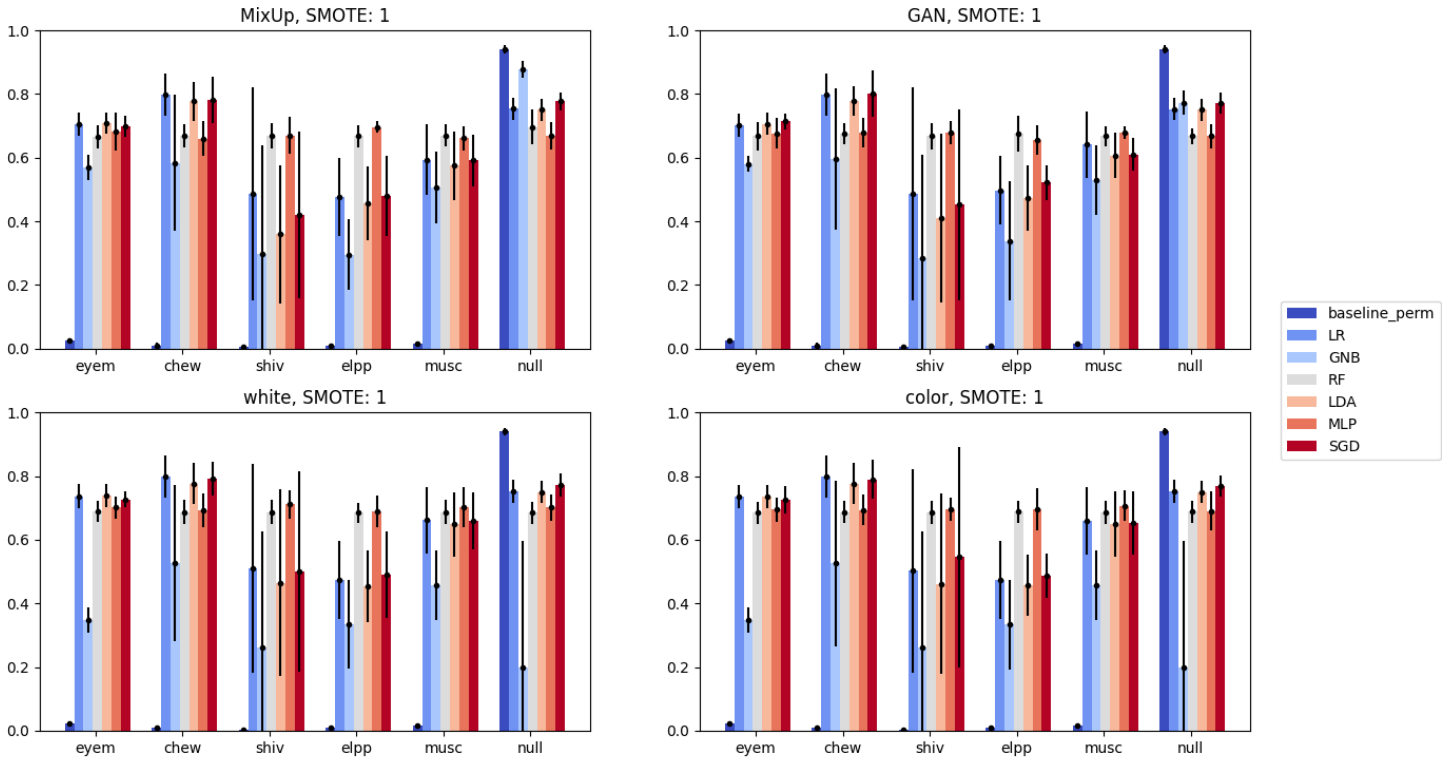



**Figure 32**

### MixUp

**Table 44:** Mean performance (sensitivity) with std. deviation over 5-fold-CV

| Algorithm | $BAcc._{eyem}$ | $BAcc._{chew}$ | $BAcc._{shiv}$ | $BAcc._{elpp}$ | $BAcc._{musc}$ | $BAcc._{null}$ | Avg. $BAcc.$ |
|---|---|---|---|---|---|---|---|
| baseline_perm | 2.44 ± 0.53 | 0.87 ± 0.94 | 0.48 ± 0.67 | 0.85 ± 0.68 | 1.56 ± 0.62 | 93.97 ± 1.23 | 16.7 ± 0.78 |
| LR | 70.5 ± 3.64 | 79.78 ± 6.62 | 48.55 ± 33.47 | 47.6 ± 12.33 | 59.08 ± 11.41 | 75.32 ± 3.64 | 63.47 ± 11.85 |
| GNB | 56.98 ± 3.9 | 58.39 ± 21.41 | 29.86 ± 33.88 | 29.56 ± 11.21 | 50.63 ± 11.34 | 87.79 ± 2.61 | 52.2 ± 14.06 |
| RF | 66.6 ± 3.64 | 66.89 ± 3.79 | 66.96 ± 3.92 | 66.7 ± 3.47 | 67.03 ± 3.64 | 69.64 ± 5.5 | 67.3 ± 3.99 |
| LDA | 70.83 ± 3.32 | 77.68 ± 6.23 | 35.93 ± 21.64 | 45.69 ± 11.48 | 57.49 ± 10.69 | 75.02 ± 3.47 | 60.44 ± 9.47 |
| MLP | 68.19 ± 5.85 | 65.99 ± 5.47 | 66.99 ± 5.81 | 69.66 ± 1.77 | 66.14 ± 3.84 | 66.23 ± 5.58 | 67.2 ± 4.72 |
| SGD | 69.84 ± 3.22 | 77.86 ± 5.01 | 41.99 ± 26.07 | 47.99 ± 12.56 | 57.76 ± 10.26 | 77.73 ± 2.88 | 62.2 ± 10.0 |

**Table 45:** Best augmentation ratios belonging to sensitivity-scores in 44

| | baseline_perm | LR | GNB | RF | LDA | MLP | SGD |
|---|---|---|---|---|---|---|---|
| eyem | 0.5 | 1.0 | 0.5 | 1.5 | 1.5 | 0.5 | 0.5 |
| chew | 0.5 | 0.5 | 0.5 | 2.0 | 0.5 | 1.5 | 1.5 |
| shiv | 0.5 | 0.5 | 0.5 | 2.0 | 1.0 | 1.0 | 1.5 |
| elpp | 0.5 | 2.0 | 0.5 | 1.5 | 2.0 | 0.5 | 1.0 |
| musc | 0.5 | 0.5 | 0.5 | 0.5 | 2.0 | 1.5 | 1.5 |
| null | 0.5 | 0.5 | 2.0 | 1.0 | 0.5 | 1.5 | 1.0 |

### GAN

**Table 46:** Mean performance (sensitivity) with std. deviation over 5-fold-CV

| Algorithm | $BAcc._{eyem}$ | $BAcc._{chew}$ | $BAcc._{shiv}$ | $BAcc._{elpp}$ | $BAcc._{musc}$ | $BAcc._{null}$ | Avg. $BAcc.$ |
|---|---|---|---|---|---|---|---|
| baseline_perm | 2.44 ± 0.53 | 0.87 ± 0.94 | 0.48 ± 0.67 | 0.85 ± 0.68 | 1.56 ± 0.62 | 93.97 ± 1.23 | 16.7 ± 0.78 |
| LR | 70.15 ± 3.7 | 79.74 ± 6.68 | 48.67 ± 33.48 | 49.79 ± 10.8 | 64.15 ± 10.4 | 74.18 ± 3.79 | 64.45 ± 11.48 |
| GNB | 58.09 ± 2.59 | 59.53 ± 22.17 | 28.57 ± 32.22 | 33.87 ± 18.63 | 52.94 ± 10.91 | 77.25 ± 3.89 | 51.71 ± 15.07 |
| RF | 66.79 ± 4.44 | 67.61 ± 3.23 | 66.7 ± 4.2 | 67.62 ± 5.66 | 66.85 ± 3.14 | 66.74 ± 2.46 | 67.05 ± 3.86 |
| LDA | 70.26 ± 3.65 | 77.83 ± 4.63 | 40.97 ± 26.45 | 47.16 ± 10.26 | 60.73 ± 7.05 | 72.65 ± 3.86 | 61.6 ± 9.32 |
| MLP | 67.03 ± 2.33 | 67.77 ± 4.58 | 64.66 ± 5.67 | 65.51 ± 4.68 | 67.86 ± 2.09 | 66.82 ± 3.84 | 66.61 ± 3.86 |
| SGD | 71.4 ± 2.49 | 80.11 ± 7.36 | 45.26 ± 30.0 | 52.17 ± 5.45 | 61.04 ± 5.19 | 74.35 ± 3.56 | 64.06 ± 9.01 |

**Table 47:** Best augmentation ratios belonging to sensitivity-scores in 46

| | baseline_perm | LR | GNB | RF | LDA | MLP | SGD |
|---|---|---|---|---|---|---|---|
| eyem | 0.5 | 0.5 | 0.5 | 0.5 | 0.5 | 2 | 2.0 |
| chew | 0.5 | 1.5 | 2.0 | 1.0 | 1.5 | 1 | 1.0 |
| shiv | 0.5 | 0.5 | 0.5 | 1.0 | 1.5 | 2 | 1.0 |
| elpp | 0.5 | 1.5 | 1.5 | 1.0 | 1.5 | 1 | 0.5 |
| musc | 0.5 | 2.0 | 0.5 | 1.0 | 2.0 | 2 | 1.0 |
| null | 0.5 | 0.5 | 0.5 | 1.0 | 0.5 | 2 | 1.0 |

### White Noise

**Table 48:** Mean performance (sensitivity) with std. deviation over 5-fold-CV

| Algorithm | $BAcc._{eyem}$ | $BAcc._{chew}$ | $BAcc._{shiv}$ | $BAcc._{elpp}$ | $BAcc._{musc}$ | $BAcc._{null}$ | Avg. $BAcc.$ |
|---|---|---|---|---|---|---|---|
| baseline_perm | 2.44 ± 0.53 | 0.87 ± 0.94 | 0.48 ± 0.67 | 0.85 ± 0.68 | 1.56 ± 0.62 | 93.97 ± 1.23 | 16.7 ± 0.78 |
| LR | 73.68 ± 3.8 | 77.92 ± 9.11 | 50.95 ± 32.81 | 46.73 ± 11.25 | 66.19 ± 10.51 | 73.72 ± 4.21 | 64.87 ± 11.95 |
| GNB | 27.11 ± 3.3 | 52.7 ± 24.54 | 26.1 ± 36.6 | 33.49 ± 14.0 | 45.62 ± 10.98 | 19.13 ± 38.27 | 34.02 ± 21.28 |
| RF | 68.81 ± 3.36 | 68.66 ± 3.76 | 68.65 ± 3.82 | 68.51 ± 3.2 | 68.7 ± 3.75 | 68.42 ± 3.54 | 68.62 ± 3.57 |
| LDA | 73.8 ± 3.71 | 74.95 ± 10.23 | 46.54 ± 29.44 | 45.39 ± 10.12 | 64.79 ± 10.08 | 73.36 ± 4.15 | 63.14 ± 11.29 |
| MLP | 70.17 ± 3.55 | 69.28 ± 5.25 | 71.13 ± 4.49 | 68.95 ± 5.06 | 70.32 ± 6.28 | 70.18 ± 4.13 | 70.0 ± 4.79 |
| SGD | 72.68 ± 2.55 | 79.21 ± 5.4 | 50.12 ± 31.49 | 49.0 ± 13.7 | 65.83 ± 8.93 | 76.21 ± 4.11 | 65.51 ± 11.03 |

**Table 49:** Best augmentation ratios belonging to sensitivity-scores in 48

| | baseline_perm | LR | GNB | RF | LDA | MLP | SGD |
|---|---|---|---|---|---|---|---|
| eyem | 0.5 | 2.0 | 0.5 | 1.5 | 2.0 | 2.0 | 1.0 |
| chew | 0.5 | 0.5 | 1.0 | 2.0 | 0.5 | 0.5 | 0.5 |
| shiv | 0.5 | 2.0 | 0.5 | 2.0 | 2.0 | 1.0 | 2.0 |
| elpp | 0.5 | 0.5 | 1.5 | 1.5 | 0.5 | 0.5 | 0.5 |
| musc | 0.5 | 2.0 | 0.5 | 2.0 | 2.0 | 1.5 | 2.0 |
| null | 0.5 | 0.5 | 0.5 | 1.5 | 0.5 | 1.0 | 0.5 |

### Color Noise

**Table 50:** Mean performance (sensitivity) with std. deviation over 5-fold-CV

| Algorithm | $BAcc._{eyem}$ | $BAcc._{chew}$ | $BAcc._{shiv}$ | $BAcc._{elpp}$ | $BAcc._{musc}$ | $BAcc._{null}$ | Avg. $BAcc.$ |
|---|---|---|---|---|---|---|---|
| baseline_perm | 2.44 ± 0.53 | 0.87 ± 0.94 | 0.48 ± 0.67 | 0.85 ± 0.68 | 1.56 ± 0.62 | 93.97 ± 1.23 | 16.7 ± 0.78 |
| LR | 73.48 ± 3.68 | 77.18 ± 10.45 | 50.21 ± 31.86 | 46.44 ± 10.89 | 66.0 ± 10.65 | 73.76 ± 4.2 | 64.51 ± 11.96 |
| GNB | 27.23 ± 3.41 | 52.61 ± 26.06 | 26.14 ± 36.62 | 33.34 ± 14.11 | 45.6 ± 10.97 | 19.14 ± 38.28 | 34.01 ± 21.58 |
| RF | 68.47 ± 3.58 | 68.69 ± 3.56 | 68.57 ± 3.6 | 68.76 ± 3.56 | 68.64 ± 3.6 | 68.75 ± 3.49 | 68.65 ± 3.56 |
| LDA | 73.61 ± 3.58 | 74.96 ± 10.1 | 46.14 ± 28.33 | 45.69 ± 9.55 | 64.93 ± 10.36 | 73.42 ± 4.1 | 63.12 ± 11.0 |
| MLP | 69.46 ± 3.88 | 69.33 ± 4.83 | 69.62 ± 3.63 | 69.54 ± 6.74 | 70.6 ± 4.77 | 69.04 ± 6.02 | 69.6 ± 4.98 |
| SGD | 72.45 ± 4.32 | 76.87 ± 9.69 | 54.53 ± 34.73 | 48.62 ± 7.02 | 65.24 ± 9.94 | 76.29 ± 4.15 | 65.67 ± 11.64 |

**Table 51:** Best augmentation ratios belonging to sensitivity-scores in 50

| | baseline_perm | LR | GNB | RF | LDA | MLP | SGD |
|---|---|---|---|---|---|---|---|
| eyem | 0.5 | 2.0 | 0.5 | 1.5 | 2.0 | 2.0 | 2.0 |
| chew | 0.5 | 0.5 | 1.5 | 2.0 | 1.0 | 2.0 | 0.5 |
| shiv | 0.5 | 2.0 | 0.5 | 2.0 | 2.0 | 0.5 | 2.0 |
| elpp | 0.5 | 0.5 | 1.5 | 2.0 | 0.5 | 1.5 | 1.5 |
| musc | 0.5 | 2.0 | 0.5 | 1.5 | 2.0 | 1.5 | 2.0 |
| null | 0.5 | 0.5 | 0.5 | 2.0 | 0.5 | 2.0 | 0.5 |

### 6.6 F - Standard Deviation of spectrograms + dingle spectrograms of 'elpp'

Standard deviation of the spectrograms in the control experiment:

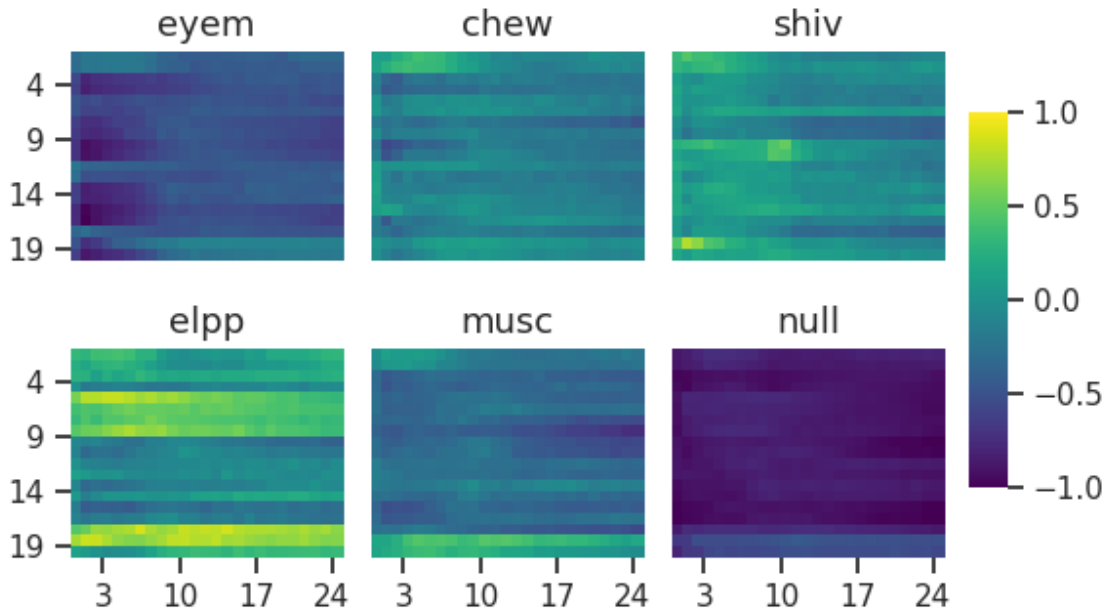



**Figure 33**

Standard deviation of the spectrograms in the improvement experiment, with SMOTE:

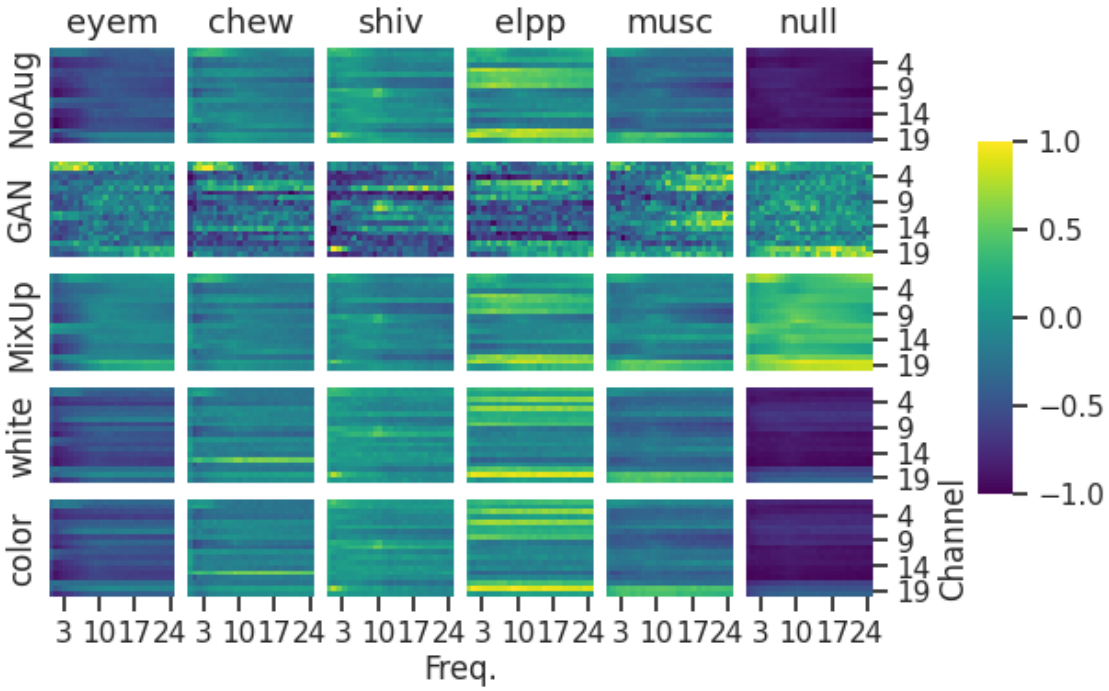



**Figure 34**

Standard deviation of the spectrograms in the improvement experiment, without SMOTE:

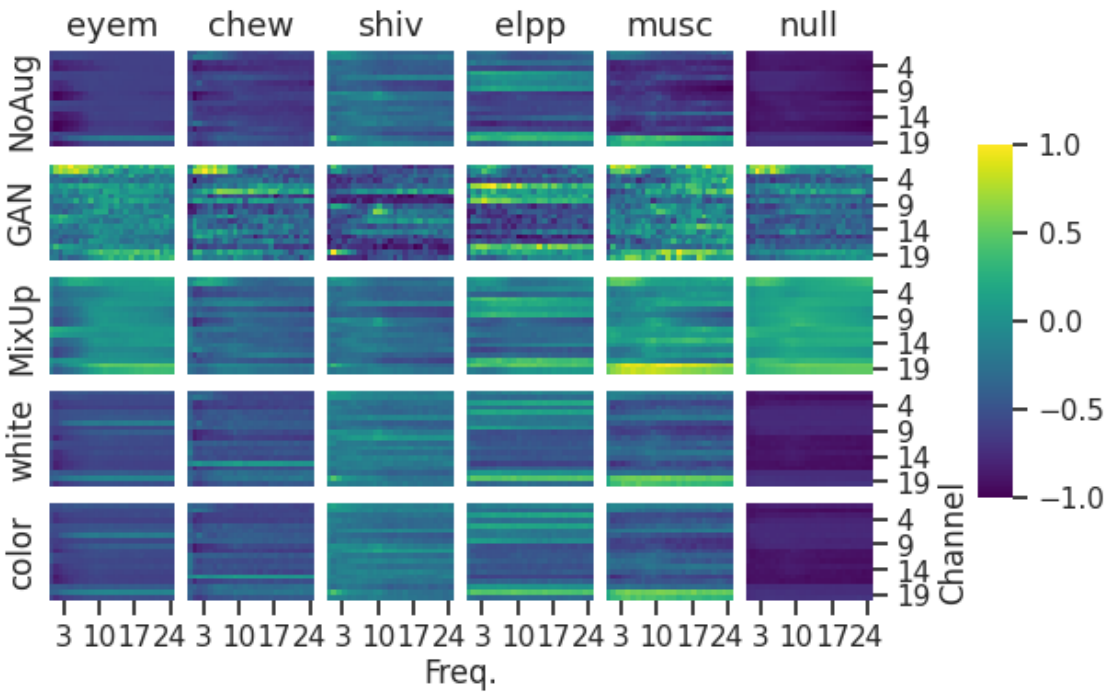



**Figure 35**

Six instances of 'elpp', it is clear that the electrodes popping are quite random. Also explained by the standard deviation.

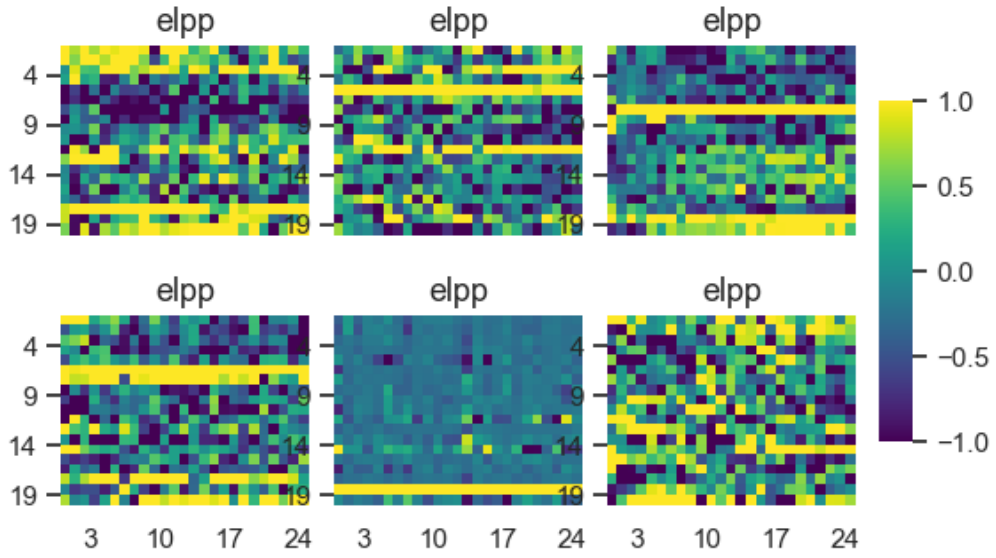



**Figure 36**

## 6.7 G - Python console output (PDF) of the 100 best models ranked by balanced accuracy

```
"C:\Users\Albert Kjøller\AppData\Local\Programs\Python\Python38\python.exe" "C:\Program Files\JetBrains\PyCharm 2020.3.4\plugins\python\helpers\pydev\pydevd.py" --multiproc --qt-support=auto --client 127.0.0.1 --port 53006 --file "C:/Users/Albert Kjøller/Documents/GitHub/EEG_epilepsia/evaluation/ensemble_results.py"
Connected to pydev debugger (build 203.7717.65)
----------------------------------------------------------------------------------
Created object for experiment: _ensemble_experiment

orderedPredictions_100balanced_acc_foldsTrue.npy loaded

------------------------------------------------------------------------------------------
Artifact: eyem

1)  RF      (SMOTE 1): white        (1.5) = 0.7598470063712564
2)  RF      (SMOTE 0): MixUp        (1.0) = 0.7597052281147808
3)  RF      (SMOTE 0): color        (0.5) = 0.7595652115969178
4)  RF      (SMOTE 1): white        (1.0) = 0.7590293015651215
5)  RF      (SMOTE 0): MixUp        (1.5) = 0.7582840990759143
6)  RF      (SMOTE 0): white        (1.5) = 0.7581002903960485
7)  RF      (SMOTE 0): color        (1.5) = 757602096549428
8)  RF      (SMOTE 0): color        (1.0) = 0.7572142547681888
9)  RF      (SMOTE 1): color        (2.0) = 0.7567710644242869
10) RF      (SMOTE 0): control      (0.0) = 0.7567331201381862
11) RF      (SMOTE 1): color        (1.0) = 0.7565487756650048
12) RF      (SMOTE 0): color        (2.0) = 0.7564164943605605
13) RF      (SMOTE 1): white        (2.0) = 0.7563538272146888
14) RF      (SMOTE 1): color        (1.5) = 0.7562706917576204
15) RF      (SMOTE 0): white        (2.0) = 0.7561579820947238
16) RF      (SMOTE 0): MixUp        (0.5) = 755831872562968
17) RF      (SMOTE 0): white        (1.0) = 0.7557018605079536
18) RF      (SMOTE 1): color        (0.5) = 0.7555146607373292
19) RF      (SMOTE 1): MixUp        (0.5) = 755170657460716
20) RF      (SMOTE 0): GAN      (0.5) = 0.7525796978410668
21) LR      (SMOTE 0): MixUp        (0.5) = 0.7514075905272499
22) RF      (SMOTE 0): white        (0.5) = 0.7513906475781461
23) RF      (SMOTE 0): GAN      (1.5) = 0.7513397740212329
24) LDA     (SMOTE 1): color        (0.5) = 0.7509850233010805
25) LDA     (SMOTE 1): MixUp        (1.5) = 0.7507834982222455
26) LR      (SMOTE 1): MixUp        (2.0) = 0.7507770385832854
27) LR      (SMOTE 0): MixUp        (1.0) = 0.7506147416805391
28) LDA     (SMOTE 1): white        (0.5) = 0.7504952218535246
29) RF      (SMOTE 1): MixUp        (1.5) = 0.7504771878107921
30) LR      (SMOTE 0): control      (0.0) = 0.7502690479869896
31) LR      (SMOTE 1): MixUp        (1.0) = 0.7501647516253448
32) LDA     (SMOTE 1): MixUp        (1.0) = 0.7500801938112168
33) LDA     (SMOTE 1): MixUp        (0.5) = 0.7500009713990018
34) LDA     (SMOTE 1): white        (1.0) = 0.7499777258299176
35) LDA     (SMOTE 1): MixUp        (2.0) = 0.7499151046205779
36) LDA     (SMOTE 0): MixUp        (0.5) = 0.7498753287340968
37) LR      (SMOTE 1): MixUp        (1.5) = 0.749844469850496
38) LDA     (SMOTE 0): control          (0.0) = 0.7498124630057288
39) RF      (SMOTE 0): MixUp        (2.0) = 0.7496093921489527
40) LR      (SMOTE 1): MixUp        (0.5) = 0.7495753595673914
41) LDA     (SMOTE 0): white        (0.5) = 0.7495512887998006
```

```
 52 42) LR      (SMOTE 0): MixUp        (2.0) = 0.7493969109601336
 53 43) LDA     (SMOTE 0): color        (0.5) = 0.7493555802202273
 54 44) LR      (SMOTE 1): color        (0.5) = 0.749315378294822
 55 45) LR      (SMOTE 1): white        (0.5) = 0.7492298390705852
 56 46) LR      (SMOTE 0): MixUp        (1.5) = 0.7491867359165549
 57 47) LDA     (SMOTE 0): MixUp        (1.0) = 0.749065811467914
 58 48) LDA     (SMOTE 1): color        (1.0) = 0.7490129587008988
 59 49) LDA     (SMOTE 0): MixUp        (1.5) = 0.7487901280300063
 60 50) LR      (SMOTE 1): white        (1.0) = 0.7486267894995758
 61 51) LDA     (SMOTE 0): MixUp        (2.0) = 0.7482142552274804
 62 52) LDA     (SMOTE 1): white        (1.5) = 0.7482076516053157
 63 53) GNB     (SMOTE 0): MixUp        (0.5) = 0.7481966401551186
 64 54) LDA     (SMOTE 0): white        (1.0) = 0.7478592095741206
 65 55) LDA     (SMOTE 0): color        (1.0) = 0.7478219772458786
 66 56) LR      (SMOTE 0): color        (0.5) = 0.7477303308245263
 67 57) LDA     (SMOTE 1): color        (1.5) = 0.74769997244688
 68 58) RF      (SMOTE 1): white        (0.5) = 0.747506843424197
 69 59) LR      (SMOTE 0): white        (0.5) = 0.7472304095951752
 70 60) LR      (SMOTE 1): color        (1.0) = 0.7471802242156584
 71 61) LDA     (SMOTE 1): white        (2.0) = 0.7471631589205292
 72 62) LR      (SMOTE 0): GAN      (0.5) = 0.7468359828536725
 73 63) LR      (SMOTE 0): white        (1.0) = 0.746709318048827
 74 64) LDA     (SMOTE 0): white        (1.5) = 0.7466552902590152
 75 65) LDA     (SMOTE 1): color        (2.0) = 0.7464999202928844
 76 66) LR      (SMOTE 0): GAN      (1.5) = 0.7464869149994312
 77 67) LR      (SMOTE 0): GAN      (2.0) = 0.7464584835407648
 78 68) LR      (SMOTE 1): white        (1.5) = 0.7464258490457312
 79 69) GNB     (SMOTE 0): MixUp        (1.0) = 0.7463471811628741
 80 70) LDA     (SMOTE 0): color        (1.5) = 0.7463467771140067
 81 71) LR      (SMOTE 1): color        (1.5) = 0.7463001142971221
 82 72) LR      (SMOTE 0): color        (1.0) = 0.7461157551954761
 83 73) LR      (SMOTE 1): white        (2.0) = 0.7456362520657611
 84 74) LDA     (SMOTE 0): white        (2.0) = 0.7456122299347822
 85 75) LR      (SMOTE 1): GAN      (0.5) = 0.745540207630993
 86 76) SGD     (SMOTE 0): MixUp        (1.5) = 0.7454406865403638
 87 77) SGD     (SMOTE 1): MixUp        (1.5) = 0.7452307115760884
 88 78) LR      (SMOTE 0): GAN      (1.0) = 0.7451240238908804
 89 79) LDA     (SMOTE 0): color        (2.0) = 0.7449086480594295
 90 80) GNB     (SMOTE 0): MixUp        (1.5) = 0.7447544114904897
 91 81) SGD     (SMOTE 1): MixUp        (1.0) = 0.7447480288281783
 92 82) SGD     (SMOTE 0): white        (0.5) = 0.7447014837879429
 93 83) LR      (SMOTE 1): color        (2.0) = 0.7445122666610391
 94 84) LR      (SMOTE 0): white        (1.5) = 0.7444258229933128
 95 85) LR      (SMOTE 0): color        (1.5) = 0.7442045350093968
 96 86) RF      (SMOTE 1): MixUp        (2.0) = 0.7440404348358673
 97 87) GNB     (SMOTE 0): MixUp        (2.0) = 0.7438462522442852
 98 88) LR      (SMOTE 0): white        (2.0) = 0.7433146923302463
 99 89) SGD     (SMOTE 1): MixUp        (0.5) = 0.7431082954006286
100 90) RF      (SMOTE 1): GAN      (1.0) = 0.742625621015949
101 91) LR      (SMOTE 0): color        (2.0) = 0.7424443233261332
102 92) SGD     (SMOTE 1): white        (1.0) = 0.7422787939107072
103 93) SGD     (SMOTE 0): MixUp        (0.5) = 0.7421358030731209
104 94) SGD     (SMOTE 1): white        (0.5) = 0.7407187805032243
105 95) LDA     (SMOTE 0): GAN          (0.5) = 0.7402767572510862
106 96) SGD     (SMOTE 0): color        (1.5) = 0.7401691738595506
107 97) RF      (SMOTE 1): GAN      (0.5) = 0.7395584406895584
108 98) SGD     (SMOTE 0): GAN          (0.5) = 0.7395258916400453
```

```
99) LR      (SMOTE 1): GAN      (1.5) = 0.7395223009800229
100)    RF      (SMOTE 1): GAN      (1.0) = 0.7390185454259365

-------------------------------------------------------------------------------
Artifact: chew

1)  LR      (SMOTE 0): GAN      (1.0) = 0.8645131766448457
2)  LR      (SMOTE 0): MixUp        (2.0) = 0.8638269597932723
3)  LR      (SMOTE 0): MixUp        (1.5) = 0.8636275775886251
4)  LR      (SMOTE 0): GAN      (0.5) = 0.8636255950386641
5)  LR      (SMOTE 0): control      (0.0) = 0.8634163134161114
6)  SGD     (SMOTE 0): MixUp        (0.5) = 0.8632892874550567
7)  LR      (SMOTE 0): GAN      (1.5) = 0.86220998407032
8)  LR      (SMOTE 1): MixUp        (0.5) = 0.8621376716750477
9)  LR      (SMOTE 0): MixUp        (0.5) = 0.8617078068716648
10) LR      (SMOTE 0): MixUp        (1.0) = 0.8612120113351616
11) LR      (SMOTE 0): GAN      (2.0) = 0.8605063346819086
12) LR      (SMOTE 1): MixUp        (2.0) = 0.8602363540255654
13) LR      (SMOTE 1): GAN      (0.5) = 0.8598622721661219
14) LR      (SMOTE 1): MixUp        (1.5) = 0.859426779918391
15) SGD     (SMOTE 1): GAN          (1.0) = 0.8593930431557745
16) LR      (SMOTE 1): GAN      (1.0) = 0.8585229141937532
17) SGD     (SMOTE 0): GAN          (0.5) = 0.8583848762108512
18) LR      (SMOTE 1): GAN      (1.5) = 0.8583715665742503
19) LR      (SMOTE 1): MixUp        (1.0) = 0.858230076548711
20) SGD     (SMOTE 1): GAN          (0.5) = 0.8581035985905816
21) SGD     (SMOTE 0): GAN          (1.0) = 0.8566911026308057
22) LDA     (SMOTE 0): control          (0.0) = 0.8547892349760453
23) LR      (SMOTE 1): GAN      (2.0) = 0.8547494348950562
24) LDA     (SMOTE 0): MixUp        (2.0) = 0.8544865615949945
25) LDA     (SMOTE 0): MixUp        (1.5) = 0.8540584524662659
26) LDA     (SMOTE 0): MixUp        (1.0) = 0.853680430876339
27) LDA     (SMOTE 0): MixUp        (0.5) = 0.8533157319560793
28) LR      (SMOTE 0): color        (0.5) = 0.8519551245049414
29) SGD     (SMOTE 0): GAN          (1.5) = 0.8518511850149928
30) SGD     (SMOTE 1): white        (0.5) = 0.8515964492286965
31) LDA     (SMOTE 1): MixUp        (0.5) = 0.8510752260747478
32) SGD     (SMOTE 1): MixUp        (1.5) = 0.8506823124593689
33) LDA     (SMOTE 1): MixUp        (2.0) = 0.8503390088060325
34) LDA     (SMOTE 1): MixUp        (1.5) = 0.8502838066972126
35) LR      (SMOTE 0): white        (0.5) = 0.8493199572174641
36) SGD     (SMOTE 0): GAN          (2.0) = 0.8483442198842607
37) SGD     (SMOTE 0): MixUp        (2.0) = 0.8479571272945045
38) SGD     (SMOTE 1): GAN          (2.0) = 0.8474225349796264
39) LDA     (SMOTE 1): MixUp        (1.0) = 0.8472295040288824
40) LR      (SMOTE 1): white        (0.5) = 0.8459157788185598
41) SGD     (SMOTE 0): control          (0.0) = 0.8452065744393507
42) SGD     (SMOTE 0): MixUp        (1.0) = 0.8449239459753846
43) LR      (SMOTE 0): white        (1.0) = 0.843747489655204
44) SGD     (SMOTE 1): MixUp        (0.5) = 0.8433598265968438
45) SGD     (SMOTE 0): MixUp        (1.5) = 0.8430515995226076
46) LR      (SMOTE 1): color        (0.5) = 0.8425538015810007
47) LDA     (SMOTE 0): GAN          (0.5) = 0.8425537216128609
48) LDA     (SMOTE 0): GAN          (1.5) = 0.8420970236667916
49) LDA     (SMOTE 0): GAN          (1.0) = 0.8407253117739973
50) LR      (SMOTE 0): color        (1.0) = 0.8407119756537955
```

```
165 51) LR      (SMOTE 1): white        (1.0) = 0.8402794252301199
166 52) SGD     (SMOTE 0): white        (1.5) = 0.8394207474655284
167 53) SGD     (SMOTE 1): GAN          (1.5) = 0.8392943860870965
168 54) SGD     (SMOTE 1): color        (0.5) = 0.8387467950973531
169 55) LR      (SMOTE 0): white        (1.5) = 0.8383713335936225
170 56) SGD     (SMOTE 0): white        (0.5) = 0.8372508163254327
171 57) LR      (SMOTE 1): color        (1.0) = 0.8370552532968748
172 58) LDA     (SMOTE 0): color        (0.5) = 0.8367216749282594
173 59) LDA     (SMOTE 0): white        (0.5) = 0.8355738509917325
174 60) LDA     (SMOTE 1): GAN          (1.0) = 0.8354551480235699
175 61) SGD     (SMOTE 0): color        (1.0) = 0.8349219673106063
176 62) LDA     (SMOTE 1): GAN          (1.5) = 0.8346891908962419
177 63) LR      (SMOTE 1): white        (1.5) = 0.8342735683000653
178 64) LR      (SMOTE 0): white        (2.0) = 0.8334399793432686
179 65) SGD     (SMOTE 0): white        (1.0) = 0.8327709312192155
180 66) SGD     (SMOTE 1): white        (1.5) = 0.8324884313182643
181 67) SGD     (SMOTE 0): color        (2.0) = 831961280130902
182 68) SGD     (SMOTE 0): color        (0.5) = 0.8318780438469411
183 69) LDA     (SMOTE 1): color        (0.5) = 0.8308349701907096
184 70) LDA     (SMOTE 1): white        (0.5) = 83075652558591
185 71) LR      (SMOTE 1): white        (2.0) = 0.8304761703653124
186 72) LR      (SMOTE 0): color        (2.0) = 0.8304746053473094
187 73) LR      (SMOTE 0): color        (1.5) = 0.8299516911826561
188 74) LDA     (SMOTE 1): GAN          (2.0) = 0.8299482574015503
189 75) SGD     (SMOTE 0): color        (1.5) = 0.8296744030844408
190 76) LR      (SMOTE 1): color        (1.5) = 0.8291962505681895
191 77) LDA     (SMOTE 1): GAN          (0.5) = 0.8289406786132046
192 78) LDA     (SMOTE 0): color        (1.0) = 0.8280508277013034
193 79) LDA     (SMOTE 0): GAN          (2.0) = 0.8255558867650287
194 80) SGD     (SMOTE 1): color        (1.5) = 0.8254862560427144
195 81) SGD     (SMOTE 1): MixUp        (2.0) = 825474884777187
196 82) LDA     (SMOTE 0): white        (1.0) = 0.8253992773355776
197 83) LR      (SMOTE 1): color        (2.0) = 0.8253548560567954
198 84) LDA     (SMOTE 1): color        (1.0) = 0.8251855006480628
199 85) SGD     (SMOTE 1): color        (1.0) = 0.8247044656972535
200 86) SGD     (SMOTE 1): white        (2.0) = 0.8242533591139736
201 87) LDA     (SMOTE 1): white        (1.0) = 0.8242177933419533
202 88) SGD     (SMOTE 1): color        (2.0) = 8229383213880231
203 89) LDA     (SMOTE 0): white        (1.5) = 8211332170646409
204 90) LDA     (SMOTE 0): white        (2.0) = 8209621960464901
205 91) LDA     (SMOTE 0): color        (2.0) = 820641510200193
206 92) LDA     (SMOTE 1): white        (1.5) = 819416300809964
207 93) LDA     (SMOTE 0): color        (1.5) = 8173275763323999
208 94) LDA     (SMOTE 1): color        (1.5) = 8168663466948365
209 95) LDA     (SMOTE 1): color        (2.0) = 8145206659885466
210 96) LDA     (SMOTE 1): white        (2.0) = 8126487880157661
211 97) SGD     (SMOTE 0): white        (2.0) = 8107076098533413
212 98) SGD     (SMOTE 1): MixUp        (1.0) = 8038685561575285
213 99) SGD     (SMOTE 1): white        (1.0) = 800810830772542
214 100)    GNB     (SMOTE 0): white        (0.5) = 7933989251390504
215
216 ------------------------------------------------------------------------
    ------------------------------
217 Artifact: shiv
218
219 1)  LR      (SMOTE 0): MixUp        (2.0) = 6968703333660956
220 2)  LR      (SMOTE 0): GAN      (0.5) = 6959733970956717
```

```
221 3)  SGD     (SMOTE 0): GAN          (1.5) = 0.6958360959992921
222 4)  LR      (SMOTE 0): GAN      (1.0) = 0.6957765940596937
223 5)  LR      (SMOTE 1): MixUp        (2.0) = 0.6949792692323143
224 6)  LR      (SMOTE 0): MixUp        (1.0) = 0.6948590037117202
225 7)  LR      (SMOTE 1): MixUp        (0.5) = 0.6946677387912208
226 8)  LR      (SMOTE 1): GAN      (0.5) = 0.6941653677037531
227 9)  LR      (SMOTE 0): MixUp        (0.5) = 0.6937825110699635
228 10) LR      (SMOTE 0): MixUp        (1.5) = 0.6927416250203455
229 11) LR      (SMOTE 0): control      (0.0) = 0.6924678380916449
230 12) SGD     (SMOTE 1): color        (1.0) = 0.6919841996323252
231 13) LR      (SMOTE 0): GAN      (2.0) = 0.6904925344615449
232 14) SGD     (SMOTE 1): color        (2.0) = 0.6882956400967165
233 15) LR      (SMOTE 1): GAN      (1.0) = 0.6880603287445288
234 16) LR      (SMOTE 1): MixUp        (1.0) = 0.6873089937372467
235 17) LR      (SMOTE 1): MixUp        (1.5) = 0.6869518807742068
236 18) LR      (SMOTE 1): white        (2.0) = 0.6847617793229464
237 19) LR      (SMOTE 0): color        (0.5) = 0.6837850904234962
238 20) LR      (SMOTE 0): white        (2.0) = 0.6837120422815585
239 21) LR      (SMOTE 0): white        (0.5) = 0.683505204753898
240 22) SGD     (SMOTE 1): color        (0.5) = 0.6824016575813706
241 23) SGD     (SMOTE 1): GAN          (1.0) = 0.6812173868828932
242 24) LR      (SMOTE 1): color        (1.0) = 0.6810413318431241
243 25) SGD     (SMOTE 1): white        (2.0) = 0.6808229940026423
244 26) LR      (SMOTE 0): GAN      (1.5) = 0.6803284501462148
245 27) LR      (SMOTE 0): white        (1.0) = 0.6800079881771942
246 28) LR      (SMOTE 0): color        (1.0) = 0.6799531939711175
247 29) LR      (SMOTE 0): color        (1.5) = 0.6799097575370416
248 30) LR      (SMOTE 1): color        (1.5) = 0.6776222151458309
249 31) LR      (SMOTE 1): white        (1.0) = 0.6776064931445149
250 32) LR      (SMOTE 1): color        (2.0) = 0.6770698794920444
251 33) LR      (SMOTE 1): white        (0.5) = 0.6767680283522803
252 34) LR      (SMOTE 0): color        (2.0) = 0.6761459886300948
253 35) LR      (SMOTE 1): GAN      (1.5) = 0.6757931754124951
254 36) LR      (SMOTE 1): white        (1.5) = 0.675173641223076
255 37) LR      (SMOTE 1): color        (0.5) = 0.6750749927385007
256 38) LR      (SMOTE 1): GAN      (2.0) = 0.6736201117771363
257 39) SGD     (SMOTE 0): white        (2.0) = 0.6730666417505622
258 40) LR      (SMOTE 0): white        (1.5) = 0.6712644918779118
259 41) SGD     (SMOTE 1): white        (1.5) = 0.6687735190729445
260 42) SGD     (SMOTE 1): color        (1.5) = 0.6672335296252689
261 43) SGD     (SMOTE 0): white        (0.5) = 0.6670843919986595
262 44) LDA     (SMOTE 1): white        (2.0) = 0.6640517299988922
263 45) LDA     (SMOTE 1): color        (2.0) = 0.6622852690492009
264 46) LDA     (SMOTE 1): white        (1.5) = 0.6617256241744105
265 47) SGD     (SMOTE 0): color        (2.0) = 0.6610839559190069
266 48) LDA     (SMOTE 0): white        (2.0) = 0.6610521161075562
267 49) LDA     (SMOTE 0): color        (2.0) = 0.6609476228031237
268 50) SGD     (SMOTE 0): color        (1.0) = 0.6607057639448277
269 51) SGD     (SMOTE 1): MixUp        (1.5) = 0.6598296851510254
270 52) SGD     (SMOTE 0): color        (0.5) = 0.6596888072123683
271 53) SGD     (SMOTE 0): white        (1.5) = 0.6592728037417587
272 54) LDA     (SMOTE 1): color        (1.5) = 0.658665395818693
273 55) SGD     (SMOTE 0): color        (1.5) = 0.6581217303302853
274 56) LDA     (SMOTE 0): color        (1.5) = 0.6574663938670734
275 57) SGD     (SMOTE 0): MixUp        (0.5) = 0.6560772021305719
276 58) LDA     (SMOTE 1): color        (1.0) = 0.6556577080521944
277 59) SGD     (SMOTE 0): white        (1.0) = 0.6547682167777584
```

```
278 60) SGD     (SMOTE 1): GAN          (0.5) = 0.6529547427398058
279 61) LDA     (SMOTE 0): white        (1.0) = 0.6526101432986132
280 62) SGD     (SMOTE 0): MixUp        (2.0) = 0.6525739336023044
281 63) SGD     (SMOTE 0): GAN          (2.0) = 0.6518375474292879
282 64) LDA     (SMOTE 1): color        (0.5) = 0.651635653203071
283 65) LDA     (SMOTE 1): white        (1.0) = 0.6512628805211136
284 66) SGD     (SMOTE 1): white        (1.0) = 0.6510865977273578
285 67) LDA     (SMOTE 0): color        (1.0) = 0.651032711889966
286 68) LDA     (SMOTE 0): white        (1.5) = 0.6496367285728907
287 69) LDA     (SMOTE 1): white        (0.5) = 0.6476059891597399
288 70) SGD     (SMOTE 0): control          (0.0) = 0.645903452037234
289 71) LDA     (SMOTE 0): white        (0.5) = 0.645714680542918
290 72) SGD     (SMOTE 1): white        (0.5) = 0.6436695932505925
291 73) LDA     (SMOTE 0): color        (0.5) = 0.6422048178050251
292 74) GNB     (SMOTE 0): GAN          (0.5) = 0.6414402727611253
293 75) LDA     (SMOTE 1): GAN          (1.5) = 0.6414130256855844
294 76) GNB     (SMOTE 0): GAN          (1.0) = 0.6410663431439843
295 77) SGD     (SMOTE 0): GAN          (0.5) = 0.6385338148027857
296 78) SGD     (SMOTE 1): MixUp        (2.0) = 0.636012388203417
297 79) LDA     (SMOTE 1): GAN          (2.0) = 0.6338474934242042
298 80) LDA     (SMOTE 1): MixUp        (1.0) = 0.6332073340203817
299 81) SGD     (SMOTE 1): MixUp        (0.5) = 0.6316321564586886
300 82) GNB     (SMOTE 0): GAN          (2.0) = 0.6311353259419944
301 83) LDA     (SMOTE 1): MixUp        (1.5) = 0.6308241166619214
302 84) LDA     (SMOTE 1): GAN          (0.5) = 0.6303539456937702
303 85) LDA     (SMOTE 1): MixUp        (0.5) = 0.6290915295478318
304 86) LDA     (SMOTE 1): MixUp        (2.0) = 0.6286230012432135
305 87) LDA     (SMOTE 0): control          (0.0) = 0.6281821546609219
306 88) LDA     (SMOTE 0): MixUp        (1.0) = 0.6269690358929861
307 89) LDA     (SMOTE 0): MixUp        (1.5) = 0.6268520446915545
308 90) SGD     (SMOTE 1): GAN          (1.5) = 0.6259974832070505
309 91) LDA     (SMOTE 0): MixUp        (0.5) = 0.6248365684473071
310 92) LDA     (SMOTE 1): GAN          (1.0) = 0.6246881190088174
311 93) SGD     (SMOTE 0): MixUp        (1.0) = 0.6246491205695441
312 94) GNB     (SMOTE 0): MixUp        (0.5) = 0.6242629876578529
313 95) GNB     (SMOTE 0): MixUp        (1.5) = 0.6240907559173823
314 96) GNB     (SMOTE 0): MixUp        (2.0) = 0.6240286539045029
315 97) GNB     (SMOTE 0): MixUp        (1.0) = 0.6239447214640057
316 98) SGD     (SMOTE 1): GAN          (2.0) = 0.6233918005229926
317 99) GNB     (SMOTE 0): control          (0.0) = 0.6227875408218527
318 100)    LDA     (SMOTE 0): control          (2.0) = 0.
    6225536950603127
319
320 ------------------------------------------------------------------
    ------------------------------
321 Artifact: elpp
322
323 1)  LR      (SMOTE 0): MixUp        (0.5) = 0.6314662118221585
324 2)  LR      (SMOTE 0): control      (0.0) = 0.629425774685428
325 3)  LR      (SMOTE 0): MixUp        (1.5) = 0.62837824556863
326 4)  LR      (SMOTE 0): MixUp        (1.0) = 0.6280177402403585
327 5)  LR      (SMOTE 1): white        (0.5) = 0.6279982198221812
328 6)  LR      (SMOTE 0): color        (0.5) = 0.6277090756544543
329 7)  LR      (SMOTE 1): MixUp        (0.5) = 0.627576468182765
330 8)  LR      (SMOTE 1): MixUp        (1.5) = 0.6271413258398125
331 9)  LR      (SMOTE 1): MixUp        (2.0) = 0.6269816140332429
332 10) LR      (SMOTE 0): white        (0.5) = 0.6260295521703261
```

```
333 11) LR      (SMOTE 1): MixUp        (1.0) = 0.6260124659324358
334 12) LR      (SMOTE 1): white        (1.0) = 0.6256558483063245
335 13) LR      (SMOTE 0): GAN      (0.5) = 0.6254257715211055
336 14) LR      (SMOTE 0): white        (2.0) = 0.6251503163644502
337 15) SGD     (SMOTE 1): GAN          (0.5) = 0.6247668299425324
338 16) LR      (SMOTE 1): color        (1.0) = 0.6245425262365416
339 17) LR      (SMOTE 0): white        (1.0) = 0.6243955805788786
340 18) LR      (SMOTE 1): GAN      (0.5) = 0.6242119252896977
341 19) LR      (SMOTE 0): color        (1.5) = 0.624142991638885
342 20) LR      (SMOTE 1): white        (1.5) = 0.6239201579697609
343 21) LR      (SMOTE 0): color        (1.0) = 0.6238094648423788
344 22) LR      (SMOTE 1): color        (0.5) = 0.6226840667253543
345 23) GNB     (SMOTE 0): control          (0.0) = 0.622380054478122
346 24) LR      (SMOTE 0): white        (1.5) = 0.6217202512381036
347 25) LR      (SMOTE 1): white        (2.0) = 0.6207262134832304
348 26) LR      (SMOTE 0): color        (2.0) = 0.6207170242258027
349 27) GNB     (SMOTE 0): MixUp        (0.5) = 0.6202187938868182
350 28) GNB     (SMOTE 0): MixUp        (1.5) = 0.6201579144503477
351 29) LR      (SMOTE 1): GAN      (1.5) = 0.6195835086048677
352 30) GNB     (SMOTE 0): MixUp        (2.0) = 0.6187493829350743
353 31) SGD     (SMOTE 0): MixUp        (1.5) = 0.6184263389590308
354 32) LR      (SMOTE 1): color        (1.5) = 0.6181990188456501
355 33) LR      (SMOTE 0): MixUp        (2.0) = 0.6175081569799684
356 34) GNB     (SMOTE 0): MixUp        (1.0) = 0.6174246293335504
357 35) LR      (SMOTE 1): color        (2.0) = 0.6171555606250138
358 36) LDA     (SMOTE 1): color        (0.5) = 0.6167638367277857
359 37) GNB     (SMOTE 1): MixUp        (0.5) = 0.6165338987353943
360 38) LDA     (SMOTE 1): white        (0.5) = 0.6162545968583906
361 39) LDA     (SMOTE 0): color        (0.5) = 0.615836164974491
362 40) LDA     (SMOTE 0): control          (0.0) = 0.6153977398343339
363 41) LDA     (SMOTE 0): MixUp        (1.5) = 0.6151758987971132
364 42) LDA     (SMOTE 0): white        (1.0) = 0.6151664156990154
365 43) LDA     (SMOTE 0): white        (0.5) = 0.6151135167637675
366 44) LDA     (SMOTE 0): color        (1.0) = 0.6150963630754746
367 45) SGD     (SMOTE 0): control          (0.0) = 0.614881710902867
368 46) LDA     (SMOTE 1): white        (1.0) = 0.6146653629824033
369 47) LDA     (SMOTE 1): color        (1.0) = 0.6144332154089505
370 48) LDA     (SMOTE 1): white        (1.5) = 0.6140126629658093
371 49) LDA     (SMOTE 0): MixUp        (0.5) = 0.6140052199018559
372 50) LDA     (SMOTE 0): MixUp        (1.0) = 0.6134850596018794
373 51) LDA     (SMOTE 0): white        (2.0) = 0.6133641242026142
374 52) LDA     (SMOTE 1): color        (1.5) = 0.6129315479598147
375 53) LDA     (SMOTE 1): MixUp        (2.0) = 0.6127645639367693
376 54) LDA     (SMOTE 0): GAN          (1.5) = 0.612700025559014
377 55) SGD     (SMOTE 1): white        (0.5) = 0.6126287270834775
378 56) GNB     (SMOTE 1): color        (0.5) = 0.612434506530277
379 57) LDA     (SMOTE 1): MixUp        (1.0) = 0.6122667021617655
380 58) LDA     (SMOTE 0): color        (1.5) = 0.6122590725987228
381 59) GNB     (SMOTE 1): MixUp        (1.0) = 0.6121457293673106
382 60) LDA     (SMOTE 1): MixUp        (1.5) = 0.6118664982012767
383 61) GNB     (SMOTE 1): white        (0.5) = 0.6118373472265166
384 62) LR      (SMOTE 0): GAN      (1.0) = 0.6116580549106472
385 63) LDA     (SMOTE 0): color        (2.0) = 0.6115772012588511
386 64) LDA     (SMOTE 1): MixUp        (0.5) = 0.6115581800457076
387 65) GNB     (SMOTE 0): GAN          (0.5) = 0.6113751826897189
388 66) LDA     (SMOTE 1): white        (2.0) = 0.6113185491933779
389 67) LDA     (SMOTE 0): white        (1.5) = 0.6104221587412454
```

```
390 68) SGD     (SMOTE 1): GAN          (1.5) = 0.6101670672386825
391 69) GNB     (SMOTE 1): GAN          (0.5) = 0.6100679131718592
392 70) LDA     (SMOTE 0): MixUp        (2.0) = 0.6098291591591337
393 71) LDA     (SMOTE 1): color        (2.0) = 0.6097954792881859
394 72) GNB     (SMOTE 1): MixUp        (1.5) = 0.6096943116229978
395 73) LDA     (SMOTE 0): GAN          (1.0) = 0.6092154258446971
396 74) GNB     (SMOTE 0): color        (2.0) = 0.608754459170339
397 75) SGD     (SMOTE 0): GAN          (1.0) = 0.6086904403619393
398 76) GNB     (SMOTE 0): white        (2.0) = 0.6084205072182896
399 77) LDA     (SMOTE 0): GAN          (0.5) = 0.6084149460671802
400 78) SGD     (SMOTE 0): MixUp        (1.0) = 0.6084060664038652
401 79) GNB     (SMOTE 1): MixUp        (2.0) = 0.6083029218123274
402 80) SGD     (SMOTE 1): color        (1.5) = 0.6079528570013347
403 81) SGD     (SMOTE 1): GAN          (2.0) = 0.607658878152858
404 82) LR      (SMOTE 1): GAN      (1.0) = 0.6067126308504065
405 83) LDA     (SMOTE 1): GAN          (0.5) = 0.6065532848413889
406 84) LR      (SMOTE 0): GAN      (2.0) = 0.606412036001273
407 85) GNB     (SMOTE 0): color        (1.5) = 0.6053654979506357
408 86) GNB     (SMOTE 0): white        (1.5) = 0.6045665673572029
409 87) SGD     (SMOTE 0): MixUp        (0.5) = 0.6044740166370784
410 88) SGD     (SMOTE 0): GAN          (0.5) = 0.6042112101104247
411 89) SGD     (SMOTE 1): MixUp        (1.0) = 0.6032016508209757
412 90) GNB     (SMOTE 0): white        (0.5) = 0.602554358123031
413 91) GNB     (SMOTE 0): color        (0.5) = 0.6025341586593107
414 92) LR      (SMOTE 0): GAN      (1.5) = 0.6023994995862807
415 93) SGD     (SMOTE 1): white        (1.0) = 0.6021184651625053
416 94) SGD     (SMOTE 1): white        (2.0) = 0.6017949791595673
417 95) GNB     (SMOTE 1): GAN          (1.5) = 0.6016294463078322
418 96) LDA     (SMOTE 0): GAN          (2.0) = 0.601401049441457
419 97) GNB     (SMOTE 0): color        (1.0) = 0.6013677589523632
420 98) GNB     (SMOTE 0): white        (1.0) = 0.6008440255858827
421 99) GNB     (SMOTE 0): GAN          (1.5) = 0.6007609759546539
422 100)    SGD     (SMOTE 1): GAN          (1.5) = 0.6002228570848787
423
424 ------------------------------------------------------------------------
    ------------------------------
425 Artifact: musc
426
427 1)  GNB     (SMOTE 0): white        (2.0) = 0.707925263645252
428 2)  GNB     (SMOTE 0): color        (2.0) = 0.7072809870838184
429 3)  GNB     (SMOTE 0): MixUp        (0.5) = 0.7047887422818743
430 4)  GNB     (SMOTE 0): MixUp        (1.0) = 0.7044766234715352
431 5)  GNB     (SMOTE 0): MixUp        (1.5) = 0.7034231587570454
432 6)  GNB     (SMOTE 0): MixUp        (2.0) = 0.7030142432656827
433 7)  GNB     (SMOTE 0): color        (1.5) = 0.7027217192001343
434 8)  GNB     (SMOTE 0): white        (1.5) = 0.7025738693862043
435 9)  GNB     (SMOTE 0): control          (0.0) = 0.7013020155471354
436 10) GNB     (SMOTE 0): GAN          (0.5) = 0.7001863019259652
437 11) LR      (SMOTE 1): white        (1.5) = 0.6995248449812387
438 12) LR      (SMOTE 1): white        (1.0) = 0.6994753128983392
439 13) LR      (SMOTE 0): color        (2.0) = 0.6986288916172387
440 14) LR      (SMOTE 0): control      (0.0) = 0.6983861925288618
441 15) LR      (SMOTE 0): MixUp        (1.0) = 0.6982625766476634
442 16) LR      (SMOTE 0): color        (1.5) = 0.6977935804023521
443 17) LR      (SMOTE 0): white        (1.5) = 0.6974498805983405
444 18) LR      (SMOTE 1): white        (0.5) = 0.6974316766313808
445 19) LR      (SMOTE 0): color        (0.5) = 0.6972823572685204
```

```
446 20) LR      (SMOTE 1): color        (2.0) = 0.6972409610702351
447 21) LR      (SMOTE 1): color        (1.5) = 0.6972346197923942
448 22) LR      (SMOTE 1): color        (0.5) = 0.6971608905788336
449 23) LR      (SMOTE 0): white        (0.5) = 0.697122437956671
450 24) LR      (SMOTE 1): white        (2.0) = 0.6967779851082628
451 25) LR      (SMOTE 0): white        (2.0) = 0.6967542013352584
452 26) LR      (SMOTE 0): color        (1.0) = 0.6967200458604325
453 27) GNB     (SMOTE 1): GAN          (0.5) = 0.6966784841661784
454 28) LR      (SMOTE 0): MixUp        (0.5) = 0.6964529523545928
455 29) LR      (SMOTE 1): color        (1.0) = 0.6963929118341015
456 30) LR      (SMOTE 0): white        (1.0) = 0.6961884721401141
457 31) GNB     (SMOTE 0): white        (0.5) = 0.6959105025764502
458 32) LR      (SMOTE 0): MixUp        (1.5) = 0.6956808568952082
459 33) LR      (SMOTE 0): MixUp        (2.0) = 0.6956679028308222
460 34) GNB     (SMOTE 0): color        (0.5) = 0.6955707937651927
461 35) LR      (SMOTE 1): MixUp        (0.5) = 0.6953966987118849
462 36) LR      (SMOTE 1): MixUp        (1.5) = 0.69386637559046
463 37) LR      (SMOTE 1): MixUp        (2.0) = 0.6936199396682602
464 38) GNB     (SMOTE 0): color        (1.0) = 0.6929325018766226
465 39) GNB     (SMOTE 0): white        (1.0) = 0.6928350910798613
466 40) GNB     (SMOTE 1): MixUp        (0.5) = 0.692763663468899
467 41) LDA     (SMOTE 0): white        (1.5) = 0.6925635948208797
468 42) LR      (SMOTE 1): MixUp        (1.0) = 0.6916020517905045
469 43) SGD     (SMOTE 1): color        (1.5) = 0.6906749411559412
470 44) LDA     (SMOTE 1): white        (1.5) = 0.6903439314087267
471 45) LDA     (SMOTE 1): color        (2.0) = 0.6899567054606142
472 46) LR      (SMOTE 0): GAN      (0.5) = 0.6897444919669296
473 47) LDA     (SMOTE 1): color        (1.5) = 0.6897164279057133
474 48) GNB     (SMOTE 1): GAN          (1.0) = 0.6895707569557183
475 49) LDA     (SMOTE 0): color        (1.5) = 0.6894091464774534
476 50) LDA     (SMOTE 0): color        (2.0) = 0.6890329763698853
477 51) LDA     (SMOTE 1): color        (1.0) = 0.6889595216300585
478 52) LR      (SMOTE 1): GAN      (2.0) = 0.6887771616684487
479 53) LDA     (SMOTE 0): white        (1.0) = 0.6886779799244085
480 54) LDA     (SMOTE 0): white        (2.0) = 0.6885641961666984
481 55) LDA     (SMOTE 1): white        (1.0) = 0.6884975826367337
482 56) LDA     (SMOTE 1): color        (0.5) = 0.688317878721912
483 57) LDA     (SMOTE 1): white        (0.5) = 0.6882167549021789
484 58) LDA     (SMOTE 1): white        (2.0) = 0.688008417553242
485 59) LDA     (SMOTE 0): white        (0.5) = 0.6879464183263082
486 60) GNB     (SMOTE 1): MixUp        (1.0) = 0.6876245112313751
487 61) LDA     (SMOTE 0): color        (0.5) = 0.6873571638064557
488 62) LR      (SMOTE 0): GAN      (2.0) = 0.6864521282302156
489 63) LDA     (SMOTE 0): color        (1.0) = 0.68643663120399
490 64) GNB     (SMOTE 0): GAN          (1.5) = 0.68582459941748
491 65) GNB     (SMOTE 1): MixUp        (1.5) = 0.685287241330699
492 66) LR      (SMOTE 0): GAN      (1.0) = 0.685075431975666
493 67) SGD     (SMOTE 0): color        (1.0) = 0.6840622088336058
494 68) SGD     (SMOTE 0): MixUp        (2.0) = 0.6838225843448669
495 69) SGD     (SMOTE 0): white        (1.0) = 0.6836669405664695
496 70) SGD     (SMOTE 1): white        (2.0) = 0.6832940573726762
497 71) LR      (SMOTE 1): GAN      (1.5) = 0.6830350262038288
498 72) GNB     (SMOTE 1): MixUp        (2.0) = 0.683026375987432
499 73) LR      (SMOTE 1): GAN      (0.5) = 0.6826023579706453
500 74) LDA     (SMOTE 0): MixUp        (1.0) = 0.6824401926158566
501 75) LDA     (SMOTE 0): control          (0.0) = 0.6823929653322833
502 76) LR      (SMOTE 0): GAN      (1.5) = 0.6823899772550011
```

```
503 77) SGD     (SMOTE 1): white        (1.5) = 0.6823787048190336
504 78) SGD     (SMOTE 1): color        (2.0) = 0.6823146674556605
505 79) SGD     (SMOTE 0): color        (1.5) = 0.6822024538041715
506 80) LDA     (SMOTE 0): MixUp        (0.5) = 0.6820516736752594
507 81) LDA     (SMOTE 0): MixUp        (1.5) = 0.6819770657772992
508 82) LDA     (SMOTE 0): MixUp        (2.0) = 0.6815305431456368
509 83) LDA     (SMOTE 0): GAN          (0.5) = 0.6814431036559203
510 84) GNB     (SMOTE 1): GAN          (1.5) = 0.6810488735227563
511 85) LDA     (SMOTE 1): MixUp        (2.0) = 0.6809058011245546
512 86) LDA     (SMOTE 1): MixUp        (0.5) = 0.6808364977924213
513 87) LDA     (SMOTE 1): MixUp        (1.0) = 0.6795255710408903
514 88) LDA     (SMOTE 1): MixUp        (1.5) = 0.6794038025899216
515 89) SGD     (SMOTE 1): white        (1.0) = 0.6792043390535111
516 90) LDA     (SMOTE 1): GAN          (0.5) = 0.6788360798459527
517 91) SGD     (SMOTE 1): white        (0.5) = 0.6785302315448108
518 92) GNB     (SMOTE 1): color        (0.5) = 0.6777320743138382
519 93) GNB     (SMOTE 1): white        (0.5) = 0.6776554390992395
520 94) GNB     (SMOTE 0): GAN          (1.0) = 0.6772538999436586
521 95) LR      (SMOTE 1): GAN      (1.0) = 0.6771675360300506
522 96) SGD     (SMOTE 0): white        (2.0) = 0.6764596021618237
523 97) SGD     (SMOTE 1): MixUp        (1.0) = 0.6759993943289555
524 98) SGD     (SMOTE 1): GAN          (1.0) = 0.6758378388875533
525 99) SGD     (SMOTE 0): color        (0.5) = 0.675536546099018
526 100)    SGD     (SMOTE 0): color        (1.0) = 0.6751316854944361
527
528 ------------------------------------------------------------------------
    ------------------------------
529 Artifact: null
530
531 1)  SGD     (SMOTE 0): MixUp        (1.0) = 0.6948380234795876
532 2)  SGD     (SMOTE 0): color        (0.5) = 0.6945578233643006
533 3)  SGD     (SMOTE 0): white        (0.5) = 0.6944557559985787
534 4)  SGD     (SMOTE 0): control          (0.0) = 0.6944242757652339
535 5)  SGD     (SMOTE 0): MixUp        (0.5) = 0.6941389352092433
536 6)  LR      (SMOTE 0): color        (1.0) = 0.6936687943023908
537 7)  SGD     (SMOTE 0): MixUp        (2.0) = 0.6936145230486664
538 8)  LR      (SMOTE 0): color        (0.5) = 0.6935678631796854
539 9)  SGD     (SMOTE 0): white        (1.0) = 0.6933589750706729
540 10) LR      (SMOTE 0): white        (0.5) = 0.6933309164530792
541 11) SGD     (SMOTE 1): color        (0.5) = 0.6932495000049348
542 12) SGD     (SMOTE 1): white        (0.5) = 0.693239550824131
543 13) SGD     (SMOTE 0): MixUp        (1.5) = 0.6932292957230187
544 14) SGD     (SMOTE 1): color        (2.0) = 0.6932062152062144
545 15) LR      (SMOTE 1): color        (1.0) = 0.6931695453813127
546 16) LR      (SMOTE 0): white        (1.0) = 0.6931250424547268
547 17) SGD     (SMOTE 1): color        (1.0) = 0.6931041377207571
548 18) SGD     (SMOTE 1): color        (1.5) = 0.6931030522090967
549 19) SGD     (SMOTE 0): color        (2.0) = 0.6930244332450493
550 20) SGD     (SMOTE 1): white        (1.0) = 0.693023437369144
551 21) LR      (SMOTE 1): color        (0.5) = 0.6929596166187011
552 22) LR      (SMOTE 0): color        (1.5) = 0.6929484062516161
553 23) SGD     (SMOTE 0): color        (1.0) = 0.6929128411971158
554 24) LR      (SMOTE 0): MixUp        (1.0) = 0.6928112846669836
555 25) SGD     (SMOTE 0): color        (1.5) = 0.6927950867923841
556 26) LR      (SMOTE 1): white        (0.5) = 0.6927021938172666
557 27) LR      (SMOTE 0): color        (2.0) = 0.6926626809308496
558 28) LR      (SMOTE 1): white        (1.0) = 0.6925650990200855
```

```
559 29) LR      (SMOTE 0): white        (1.5) = 0.6925611774999375
560 30) SGD     (SMOTE 1): white        (2.0) = 0.6925413100923411
561 31) LR      (SMOTE 1): white        (1.5) = 0.6925245379426922
562 32) SGD     (SMOTE 1): MixUp        (0.5) = 0.69244581341601
563 33) LR      (SMOTE 1): color        (1.5) = 0.6923390586851904
564 34) SGD     (SMOTE 1): MixUp        (1.5) = 0.6923261739947426
565 35) SGD     (SMOTE 1): white        (1.5) = 0.6923004687279726
566 36) LR      (SMOTE 0): MixUp        (0.5) = 0.6922230533228442
567 37) SGD     (SMOTE 1): MixUp        (2.0) = 0.6921685901788244
568 38) LR      (SMOTE 1): color        (2.0) = 0.6921595421362337
569 39) LR      (SMOTE 0): white        (2.0) = 0.6921529973646605
570 40) LR      (SMOTE 0): MixUp        (1.5) = 0.6921128280162362
571 41) SGD     (SMOTE 0): white        (1.5) = 0.6918989004412109
572 42) SGD     (SMOTE 0): white        (2.0) = 0.691865325873791
573 43) SGD     (SMOTE 1): MixUp        (1.0) = 0.6918040446601751
574 44) LR      (SMOTE 1): white        (2.0) = 0.6917825591020851
575 45) LR      (SMOTE 0): control      (0.0) = 0.6915563590823696
576 46) LR      (SMOTE 0): MixUp        (2.0) = 0.6915528140549538
577 47) LR      (SMOTE 1): MixUp        (0.5) = 0.6906253646702953
578 48) LR      (SMOTE 1): MixUp        (1.5) = 0.6900647278326002
579 49) LR      (SMOTE 1): MixUp        (1.0) = 0.690033825775901
580 50) LR      (SMOTE 1): MixUp        (2.0) = 0.689820050975827
581 51) LDA     (SMOTE 1): color        (2.0) = 0.6893731643450615
582 52) LDA     (SMOTE 1): white        (1.0) = 0.6891259670250585
583 53) LDA     (SMOTE 1): color        (1.0) = 0.6890553799085009
584 54) LDA     (SMOTE 0): white        (1.5) = 0.688854018368853
585 55) LDA     (SMOTE 0): color        (1.0) = 0.6888285025379818
586 56) LDA     (SMOTE 0): color        (1.5) = 0.6888076986971641
587 57) LDA     (SMOTE 1): white        (2.0) = 0.6887643109698691
588 58) LDA     (SMOTE 1): color        (1.5) = 0.6887126396770282
589 59) LDA     (SMOTE 0): color        (0.5) = 0.6887042979715051
590 60) LDA     (SMOTE 1): white        (1.5) = 0.6887032222855897
591 61) LDA     (SMOTE 1): color        (0.5) = 0.6885888630137734
592 62) LDA     (SMOTE 0): color        (2.0) = 0.6885286001682613
593 63) LDA     (SMOTE 0): white        (1.0) = 0.6884677682621749
594 64) LDA     (SMOTE 1): white        (0.5) = 0.6883687263071882
595 65) LDA     (SMOTE 0): white        (0.5) = 0.6883158537554884
596 66) LDA     (SMOTE 0): white        (2.0) = 0.6878633169681876
597 67) LDA     (SMOTE 0): control          (0.0) = 0.6847668183613904
598 68) LDA     (SMOTE 0): MixUp        (1.0) = 0.6846193430216961
599 69) LDA     (SMOTE 0): MixUp        (0.5) = 0.6845947092886387
600 70) LDA     (SMOTE 0): MixUp        (1.5) = 0.6841947352991196
601 71) LR      (SMOTE 1): GAN      (0.5) = 0.6839056992567343
602 72) LDA     (SMOTE 0): MixUp        (2.0) = 0.6837074776589344
603 73) LDA     (SMOTE 1): MixUp        (2.0) = 0.6830999527356347
604 74) LDA     (SMOTE 1): MixUp        (1.0) = 0.6829840654722119
605 75) LR      (SMOTE 1): GAN      (1.0) = 0.6829271855374064
606 76) LDA     (SMOTE 1): MixUp        (0.5) = 0.682739782709036
607 77) LDA     (SMOTE 1): MixUp        (1.5) = 0.6826869595465681
608 78) LR      (SMOTE 1): GAN      (1.5) = 0.6818747318380944
609 79) SGD     (SMOTE 0): GAN          (0.5) = 0.6813204508540123
610 80) LR      (SMOTE 1): GAN      (2.0) = 0.678799690960509
611 81) SGD     (SMOTE 0): GAN          (1.0) = 0.6764602601188003
612 82) SGD     (SMOTE 1): GAN          (1.0) = 0.6761045004920406
613 83) LR      (SMOTE 0): GAN      (0.5) = 0.6739183772698464
614 84) SGD     (SMOTE 1): GAN          (1.5) = 0.6737002636598859
615 85) LR      (SMOTE 0): GAN      (2.0) = 0.6735546302870826
```

```
616 86) LR      (SMOTE 0): GAN      (1.5) = 0.6725978936062218
617 87) SGD     (SMOTE 1): GAN          (2.0) = 0.6713281147974924
618 88) SGD     (SMOTE 0): GAN          (2.0) = 0.6713227559395667
619 89) SGD     (SMOTE 1): GAN          (0.5) = 0.6700458149969825
620 90) LR      (SMOTE 0): GAN      (1.0) = 0.6675893057698812
621 91) GNB     (SMOTE 0): MixUp        (0.5) = 0.6673373212212628
622 92) LDA     (SMOTE 0): GAN          (0.5) = 0.6672165114687523
623 93) GNB     (SMOTE 0): MixUp        (1.5) = 0.6670638048675649
624 94) GNB     (SMOTE 0): MixUp        (2.0) = 0.6670542456655228
625 95) GNB     (SMOTE 0): MixUp        (1.0) = 0.667000313815576
626 96) SGD     (SMOTE 0): GAN          (1.5) = 0.6656712670449336
627 97) LDA     (SMOTE 1): GAN          (0.5) = 0.6655356906059204
628 98) LDA     (SMOTE 0): GAN          (1.0) = 0.6631676968643232
629 99) GNB     (SMOTE 1): MixUp        (0.5) = 0.6579996748759556
630 100)    LDA     (SMOTE 1): MixUp        (1.0) = 0.6578136639835783
631
```

## 6.8 H - Specifications of the best classifiers within each augmentation technique and artifact

**Table 52:** Augmentation ratios and model specifications of the best classifier within each artifact. The baseline model is excluded. An entry in the Table is builded as; SMOTE ratio, model, Aug. ratio.

| | eyem | chew | shiv | elpp | musc | null |
|---|---|---|---|---|---|---|
| Control | 0, RF 0.0 | 0, LR, 0.0 | 0, LR, 0.0 | 0, LR, 0.0 | 0, GNB, 0.0 | 0, SGD, 0 |
| MixUp | 0, RF, 1.0 | 0, LR, 2.0 | 0, LR, 2.0 | 0, LR, 0.5 | 0, GNB, 0.5 | 0, SGD, 1.0 |
| GAN | 0, RF, 0.5 | 0, LR, 1.0 | 0, LR, 0.5 | 0, LR, 0.5 | 0, GNB, 0.5 | 1, LR, 0.5 |
| colorNoise | 0, RF, 0.5 | 0, LR, 0.5 | 1, SGD, 1.0 | 0, LR, 0.5 | 0, GNB, 2.0 | 0, SGD, 0.5 |
| whiteNoise | 1, RF, 1.5 | 1, SGD, 0.5 | 1, LR, 2.0 | 1, LR, 0.5 | 0, GNB, 2.0 | 0, SGD, 0.5 |

## 6.9 I - Tables of models in ensemble classifiers

**Table 53:** Models used in the ensemble classifier for the artifact 'eyem'. The choice of models are determined as the 5 models with least mean correlation with all the other models within the overall 20 best classifiers. An intuition of the correlation can be seen from the correlation matrices in Figure 26

| | Model | Aug. tech. | Aug. ratio | SMOTE ratio |
|---|---|---|---|---|
| Classifier 16 | RF | MixUp | 0.5 | 0 |
| Classifier 17 | RF | whiteNoise | 1 | 0 |
| Classifier 20 | RF | GAN | 0.5 | 0 |
| Classifier 12 | RF | colorNoise | 2 | 0 |
| Classifier 15 | RF | whiteNoise | 2 | 0 |

**Table 54:** Models used in the ensemble classifier for the artifact 'chew'. The choice of models are determined as the 5 models with least mean correlation with all the other models within the overall 20 best classifiers. An intuition of the correlation can be seen from the correlation matrices in Figure 26

| | Model | Aug. tech. | Aug. ratio | SMOTE ratio |
|---|---|---|---|---|
| Classifier 15 | SGD | GAN | 1 | 1 |
| Classifier 18 | LR | GAN | 1.5 | 1 |
| Classifier 11 | LR | GAN | 2 | 0 |
| Classifier 17 | SGD | GAN | 0.5 | 0 |
| Classifier 7 | LR | GAN | 1.5 | 0 |

**Table 55:** Models used in the ensemble classifier for the artifact 'shiv'. The choice of models are determined as the 5 models with least mean correlation with all the other models within the overall 20 best classifiers. An intuition of the correlation can be seen from the correlation matrices in Figure 26.

| | Model | Aug. tech. | Aug. ratio | SMOTE ratio |
|---|---|---|---|---|
| Classifier 14 | SGD | colorNoise | 2 | 1 |
| Classifier 12 | SGD | colorNoise | 1 | 1 |
| Classifier 3 | SGD | GAN | 1.5 | 0 |
| Classifier 18 | LR | whiteNoise | 2 | 1 |
| Classifier 20 | LR | whiteNoise | 2 | 0 |

**Table 56:** Models used in the ensemble classifier for the artifact 'elpp'. The choice of models are determined as the 5 models with least mean correlation with all the other models within the overall 20 best classifiers. An intuition of the correlation can be seen from the correlation matrices in Figure 26.

| | Model | Aug. tech. | Aug. ratio | SMOTE ratio |
|---|---|---|---|---|
| Classifier 15 | SGD | GAN | 0.5 | 1 |
| Classifier 20 | LR | whiteNoise | 1.5 | 1 |
| Classifier 18 | LR | GAN | 0.5 | 1 |
| Classifier 14 | LR | whiteNoise | 2 | 0 |
| Classifier 19 | LR | colorNoise | 1.5 | 0 |

**Table 57:** Models used in the ensemble classifier for the artifact 'musc'. The choice of models are determined as the 5 models with least mean correlation with all the other models within the overall 20 best classifiers. An intuition of the correlation can be seen from the correlation matrices in Figure 26.

| | Model | Aug. tech. | Aug. ratio | SMOTE ratio |
|---|---|---|---|---|
| Classifier 10 | GNB | GAN | 0.5 | 0 |
| Classifier 13 | LR | colorNoise | 2 | 0 |
| Classifier 17 | LR | whiteNoise | 1.5 | 0 |
| Classifier 20 | LR | colorNoise | 2 | 1 |
| Classifier 16 | LR | colorNoise | 1.5 | 0 |

**Table 58:** Models used in the ensemble classifier for the artifact 'null'. The choice of models are determined as the 5 models with least mean correlation with all the other models within the overall 20 best classifiers. An intuition of the correlation can be seen from the correlation matrices in Figure 26.

| | Model | Aug. tech. | Aug. ratio | SMOTE ratio |
|---|---|---|---|---|
| Classifier 14 | SGD | colorNoise | 2 | 1 |
| Classifier 13 | SGD | MixUp | 1.5 | 0 |
| Classifier 1 | SGD | MixUp | 1 | 0 |
| Classifier 15 | LR | colorNoise | 1 | 1 |
| Classifier 19 | SGD | colorNoise | 2 | 0 |

## 6.10 J - Correlation matrices between model predictions

Below the correlation matrices of the 20 best classifiers on each label is visualized. The matrices are symmetric and the green rows indicate the models with lowest mean correlation. Specifications on these models are found in Appendix G (6.7)

### Eyemovement

**Table 59:** Correlation matrix of the 20 best performing classifiers on the eyemovement label.

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1.00 | 0.87 | 0.85 | 0.89 | 0.85 | 0.86 | 0.85 | 0.86 | 0.88 | 0.85 | 0.88 | 0.84 | 0.88 | 0.87 | 0.86 | 0.82 | 0.83 | 0.88 | 0.88 | 0.83 |
| 2 | 0.87 | 1.00 | 0.84 | 0.86 | 0.86 | 0.84 | 0.83 | 0.84 | 0.85 | 0.86 | 0.85 | 0.82 | 0.84 | 0.84 | 0.82 | 0.83 | 0.81 | 0.85 | 0.87 | 0.83 |
| 3 | 0.85 | 0.84 | 1.00 | 0.85 | 0.83 | 0.83 | 0.83 | 0.83 | 0.83 | 0.83 | 0.83 | 0.81 | 0.83 | 0.82 | 0.82 | 0.80 | 0.80 | 0.83 | 0.84 | 0.81 |
| 4 | 0.89 | 0.86 | 0.85 | 1.00 | 0.85 | 0.85 | 0.84 | 0.85 | 0.86 | 0.84 | 0.86 | 0.82 | 0.86 | 0.85 | 0.83 | 0.81 | 0.82 | 0.86 | 0.88 | 0.82 |
| 5 | 0.85 | 0.86 | 0.83 | 0.85 | 1.00 | 0.82 | 0.82 | 0.83 | 0.83 | 0.84 | 0.83 | 0.81 | 0.83 | 0.83 | 0.81 | 0.81 | 0.80 | 0.83 | 0.86 | 0.82 |
| 6 | 0.86 | 0.84 | 0.83 | 0.85 | 0.82 | 1.00 | 0.83 | 0.84 | 0.84 | 0.82 | 0.84 | 0.82 | 0.83 | 0.83 | 0.83 | 0.80 | 0.81 | 0.84 | 0.83 | 0.80 |
| 7 | 0.85 | 0.83 | 0.83 | 0.84 | 0.82 | 0.83 | 1.00 | 0.83 | 0.83 | 0.81 | 0.83 | 0.81 | 0.83 | 0.83 | 0.82 | 0.79 | 0.80 | 0.83 | 0.83 | 0.80 |
| 8 | 0.86 | 0.84 | 0.83 | 0.85 | 0.83 | 0.84 | 0.83 | 1.00 | 0.84 | 0.82 | 0.84 | 0.82 | 0.83 | 0.83 | 0.83 | 0.80 | 0.81 | 0.84 | 0.84 | 0.81 |
| 9 | 0.88 | 0.85 | 0.83 | 0.86 | 0.83 | 0.84 | 0.83 | 0.84 | 1.00 | 0.83 | 0.85 | 0.82 | 0.86 | 0.85 | 0.83 | 0.80 | 0.81 | 0.85 | 0.86 | 0.81 |
| 10 | 0.85 | 0.86 | 0.83 | 0.84 | 0.84 | 0.82 | 0.81 | 0.82 | 0.83 | 1.00 | 0.82 | 0.80 | 0.82 | 0.81 | 0.80 | 0.81 | 0.80 | 0.83 | 0.85 | 0.81 |
| 11 | 0.88 | 0.85 | 0.83 | 0.86 | 0.83 | 0.84 | 0.83 | 0.84 | 0.85 | 0.82 | 1.00 | 0.82 | 0.85 | 0.85 | 0.83 | 0.80 | 0.81 | 0.86 | 0.86 | 0.81 |
| 12 | 0.84 | 0.82 | 0.81 | 0.82 | 0.81 | 0.82 | 0.81 | 0.82 | 0.82 | 0.80 | 0.82 | 1.00 | 0.81 | 0.81 | 0.81 | 0.78 | 0.79 | 0.81 | 0.81 | 0.79 |
| 13 | 0.88 | 0.84 | 0.83 | 0.86 | 0.83 | 0.83 | 0.83 | 0.83 | 0.86 | 0.82 | 0.85 | 0.81 | 1.00 | 0.84 | 0.83 | 0.79 | 0.80 | 0.84 | 0.85 | 0.81 |
| 14 | 0.87 | 0.84 | 0.82 | 0.85 | 0.83 | 0.83 | 0.83 | 0.83 | 0.85 | 0.81 | 0.85 | 0.81 | 0.84 | 1.00 | 0.83 | 0.79 | 0.80 | 0.84 | 0.84 | 0.80 |
| 15 | 0.86 | 0.82 | 0.82 | 0.83 | 0.81 | 0.83 | 0.82 | 0.83 | 0.83 | 0.80 | 0.83 | 0.81 | 0.83 | 0.83 | 1.00 | 0.78 | 0.80 | 0.83 | 0.82 | 0.79 |
| 16 | 0.82 | 0.83 | 0.80 | 0.81 | 0.81 | 0.80 | 0.79 | 0.80 | 0.80 | 0.81 | 0.80 | 0.78 | 0.79 | 0.79 | 0.78 | 1.00 | 0.78 | 0.81 | 0.82 | 0.79 |
| 17 | 0.83 | 0.81 | 0.80 | 0.82 | 0.80 | 0.81 | 0.80 | 0.81 | 0.81 | 0.80 | 0.81 | 0.79 | 0.80 | 0.80 | 0.80 | 0.78 | 1.00 | 0.81 | 0.81 | 0.78 |
| 18 | 0.88 | 0.85 | 0.83 | 0.86 | 0.83 | 0.84 | 0.83 | 0.84 | 0.85 | 0.83 | 0.86 | 0.81 | 0.84 | 0.84 | 0.83 | 0.81 | 0.81 | 1.00 | 0.87 | 0.82 |
| 19 | 0.88 | 0.87 | 0.84 | 0.88 | 0.86 | 0.83 | 0.83 | 0.84 | 0.86 | 0.85 | 0.86 | 0.81 | 0.85 | 0.84 | 0.82 | 0.82 | 0.81 | 0.87 | 1.00 | 0.84 |
| 20 | 0.83 | 0.83 | 0.81 | 0.82 | 0.82 | 0.80 | 0.80 | 0.81 | 0.81 | 0.81 | 0.81 | 0.79 | 0.81 | 0.80 | 0.79 | 0.79 | 0.78 | 0.82 | 0.84 | 1.00 |

### Chewing

**Table 60:** Correlation matrix of the 20 best performing classifiers on the chewing label.

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1.00 | 0.83 | 0.83 | 0.84 | 0.83 | 0.81 | 0.79 | 0.84 | 0.82 | 0.83 | 0.80 | 0.84 | 0.82 | 0.83 | 0.76 | 0.81 | 0.78 | 0.77 | 0.85 | 0.81 |
| 2 | 0.83 | 1.00 | 0.90 | 0.86 | 0.92 | 0.80 | 0.83 | 0.88 | 0.91 | 0.90 | 0.79 | 0.88 | 0.83 | 0.89 | 0.76 | 0.81 | 0.80 | 0.75 | 0.86 | 0.80 |
| 3 | 0.83 | 0.90 | 1.00 | 0.87 | 0.94 | 0.83 | 0.83 | 0.90 | 0.91 | 0.91 | 0.79 | 0.89 | 0.84 | 0.90 | 0.77 | 0.82 | 0.81 | 0.77 | 0.88 | 0.82 |
| 4 | 0.84 | 0.86 | 0.87 | 1.00 | 0.87 | 0.84 | 0.80 | 0.88 | 0.85 | 0.86 | 0.82 | 0.87 | 0.85 | 0.86 | 0.77 | 0.84 | 0.80 | 0.79 | 0.88 | 0.83 |
| 5 | 0.83 | 0.92 | 0.94 | 0.87 | 1.00 | 0.82 | 0.84 | 0.91 | 0.94 | 0.92 | 0.79 | 0.90 | 0.85 | 0.91 | 0.77 | 0.83 | 0.80 | 0.77 | 0.88 | 0.81 |
| 6 | 0.81 | 0.80 | 0.83 | 0.84 | 0.82 | 1.00 | 0.78 | 0.85 | 0.80 | 0.83 | 0.79 | 0.85 | 0.82 | 0.83 | 0.78 | 0.79 | 0.82 | 0.77 | 0.86 | 0.85 |
| 7 | 0.79 | 0.83 | 0.83 | 0.80 | 0.84 | 0.78 | 1.00 | 0.82 | 0.83 | 0.82 | 0.76 | 0.83 | 0.78 | 0.83 | 0.75 | 0.77 | 0.77 | 0.73 | 0.81 | 0.77 |
| 8 | 0.84 | 0.88 | 0.90 | 0.88 | 0.91 | 0.85 | 0.82 | 1.00 | 0.88 | 0.88 | 0.79 | 0.93 | 0.88 | 0.92 | 0.78 | 0.84 | 0.81 | 0.80 | 0.92 | 0.85 |
| 9 | 0.82 | 0.91 | 0.91 | 0.85 | 0.94 | 0.80 | 0.83 | 0.88 | 1.00 | 0.90 | 0.78 | 0.88 | 0.83 | 0.89 | 0.75 | 0.81 | 0.79 | 0.76 | 0.86 | 0.80 |
| 10 | 0.83 | 0.90 | 0.91 | 0.86 | 0.92 | 0.83 | 0.82 | 0.88 | 0.90 | 1.00 | 0.80 | 0.88 | 0.83 | 0.90 | 0.77 | 0.82 | 0.80 | 0.76 | 0.88 | 0.82 |
| 11 | 0.80 | 0.79 | 0.79 | 0.82 | 0.79 | 0.79 | 0.76 | 0.79 | 0.78 | 0.80 | 1.00 | 0.80 | 0.80 | 0.79 | 0.73 | 0.80 | 0.75 | 0.76 | 0.82 | 0.80 |
| 12 | 0.84 | 0.88 | 0.89 | 0.87 | 0.90 | 0.85 | 0.83 | 0.93 | 0.88 | 0.88 | 0.80 | 1.00 | 0.87 | 0.94 | 0.79 | 0.84 | 0.82 | 0.79 | 0.93 | 0.85 |
| 13 | 0.82 | 0.83 | 0.84 | 0.85 | 0.85 | 0.82 | 0.78 | 0.88 | 0.83 | 0.83 | 0.80 | 0.87 | 1.00 | 0.85 | 0.76 | 0.83 | 0.78 | 0.79 | 0.88 | 0.83 |
| 14 | 0.83 | 0.89 | 0.90 | 0.86 | 0.91 | 0.83 | 0.83 | 0.92 | 0.89 | 0.90 | 0.79 | 0.94 | 0.85 | 1.00 | 0.78 | 0.83 | 0.81 | 0.77 | 0.91 | 0.84 |
| 15 | 0.76 | 0.76 | 0.77 | 0.77 | 0.77 | 0.78 | 0.75 | 0.78 | 0.75 | 0.77 | 0.73 | 0.79 | 0.76 | 0.78 | 1.00 | 0.75 | 0.76 | 0.71 | 0.78 | 0.77 |
| 16 | 0.81 | 0.81 | 0.82 | 0.84 | 0.83 | 0.79 | 0.77 | 0.84 | 0.81 | 0.82 | 0.80 | 0.84 | 0.83 | 0.83 | 0.75 | 1.00 | 0.76 | 0.78 | 0.86 | 0.81 |
| 17 | 0.78 | 0.80 | 0.81 | 0.80 | 0.80 | 0.82 | 0.77 | 0.81 | 0.79 | 0.80 | 0.75 | 0.82 | 0.78 | 0.81 | 0.76 | 0.76 | 1.00 | 0.73 | 0.81 | 0.80 |
| 18 | 0.77 | 0.75 | 0.77 | 0.79 | 0.77 | 0.77 | 0.73 | 0.80 | 0.76 | 0.76 | 0.76 | 0.79 | 0.79 | 0.77 | 0.71 | 0.78 | 0.73 | 1.00 | 0.80 | 0.78 |
| 19 | 0.85 | 0.86 | 0.88 | 0.88 | 0.88 | 0.86 | 0.81 | 0.92 | 0.86 | 0.88 | 0.82 | 0.93 | 0.88 | 0.91 | 0.78 | 0.86 | 0.81 | 0.80 | 1.00 | 0.87 |
| 20 | 0.81 | 0.80 | 0.82 | 0.83 | 0.81 | 0.85 | 0.77 | 0.85 | 0.80 | 0.82 | 0.80 | 0.85 | 0.83 | 0.84 | 0.77 | 0.81 | 0.80 | 0.78 | 0.87 | 1.00 |

### Shivering

**Table 61:** Correlation matrix of the 20 best performing classifiers on the shivering label.

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1.00 | 0.79 | 0.68 | 0.76 | 0.87 | 0.83 | 0.87 | 0.86 | 0.83 | 0.82 | 0.83 | 0.59 | 0.77 | 0.53 | 0.82 | 0.85 | 0.87 | 0.66 | 0.72 | 0.67 |
| 2 | 0.79 | 1.00 | 0.69 | 0.84 | 0.73 | 0.82 | 0.73 | 0.76 | 0.85 | 0.80 | 0.89 | 0.58 | 0.83 | 0.50 | 0.73 | 0.72 | 0.72 | 0.63 | 0.71 | 0.64 |
| 3 | 0.68 | 0.69 | 1.00 | 0.69 | 0.67 | 0.71 | 0.66 | 0.65 | 0.71 | 0.71 | 0.71 | 0.57 | 0.70 | 0.49 | 0.63 | 0.65 | 0.66 | 0.60 | 0.67 | 0.60 |
| 4 | 0.76 | 0.84 | 0.69 | 1.00 | 0.72 | 0.81 | 0.72 | 0.75 | 0.84 | 0.79 | 0.86 | 0.57 | 0.82 | 0.50 | 0.73 | 0.71 | 0.71 | 0.63 | 0.71 | 0.63 |
| 5 | 0.87 | 0.73 | 0.67 | 0.72 | 1.00 | 0.79 | 0.93 | 0.83 | 0.76 | 0.81 | 0.76 | 0.61 | 0.73 | 0.53 | 0.80 | 0.93 | 0.93 | 0.66 | 0.74 | 0.65 |
| 6 | 0.83 | 0.82 | 0.71 | 0.81 | 0.79 | 1.00 | 0.82 | 0.78 | 0.87 | 0.89 | 0.89 | 0.60 | 0.82 | 0.53 | 0.75 | 0.80 | 0.79 | 0.66 | 0.75 | 0.66 |
| 7 | 0.87 | 0.73 | 0.66 | 0.72 | 0.93 | 0.82 | 1.00 | 0.84 | 0.79 | 0.84 | 0.78 | 0.59 | 0.74 | 0.52 | 0.81 | 0.93 | 0.92 | 0.65 | 0.73 | 0.64 |
| 8 | 0.86 | 0.76 | 0.65 | 0.75 | 0.83 | 0.78 | 0.84 | 1.00 | 0.79 | 0.77 | 0.80 | 0.56 | 0.75 | 0.51 | 0.83 | 0.83 | 0.82 | 0.64 | 0.69 | 0.65 |
| 9 | 0.83 | 0.85 | 0.71 | 0.84 | 0.76 | 0.87 | 0.79 | 0.79 | 1.00 | 0.86 | 0.93 | 0.59 | 0.83 | 0.51 | 0.76 | 0.77 | 0.76 | 0.64 | 0.74 | 0.65 |
| 10 | 0.82 | 0.80 | 0.71 | 0.79 | 0.81 | 0.89 | 0.84 | 0.77 | 0.86 | 1.00 | 0.87 | 0.61 | 0.80 | 0.53 | 0.74 | 0.81 | 0.81 | 0.66 | 0.77 | 0.65 |
| 11 | 0.83 | 0.89 | 0.71 | 0.86 | 0.76 | 0.89 | 0.78 | 0.80 | 0.93 | 0.87 | 1.00 | 0.59 | 0.86 | 0.51 | 0.77 | 0.76 | 0.76 | 0.64 | 0.74 | 0.65 |
| 12 | 0.59 | 0.58 | 0.57 | 0.57 | 0.61 | 0.60 | 0.59 | 0.56 | 0.59 | 0.61 | 0.59 | 1.00 | 0.59 | 0.68 | 0.54 | 0.59 | 0.61 | 0.71 | 0.72 | 0.70 |
| 13 | 0.77 | 0.83 | 0.70 | 0.82 | 0.73 | 0.82 | 0.74 | 0.75 | 0.83 | 0.80 | 0.86 | 0.59 | 1.00 | 0.51 | 0.73 | 0.73 | 0.72 | 0.64 | 0.72 | 0.65 |
| 14 | 0.53 | 0.50 | 0.49 | 0.50 | 0.53 | 0.53 | 0.52 | 0.51 | 0.51 | 0.53 | 0.51 | 0.68 | 0.51 | 1.00 | 0.48 | 0.52 | 0.54 | 0.69 | 0.64 | 0.69 |
| 15 | 0.82 | 0.73 | 0.63 | 0.73 | 0.80 | 0.75 | 0.81 | 0.83 | 0.76 | 0.74 | 0.77 | 0.54 | 0.73 | 0.48 | 1.00 | 0.79 | 0.78 | 0.62 | 0.66 | 0.63 |
| 16 | 0.85 | 0.72 | 0.65 | 0.71 | 0.93 | 0.80 | 0.93 | 0.83 | 0.77 | 0.81 | 0.76 | 0.59 | 0.73 | 0.52 | 0.79 | 1.00 | 0.93 | 0.64 | 0.73 | 0.63 |
| 17 | 0.87 | 0.72 | 0.66 | 0.71 | 0.93 | 0.79 | 0.92 | 0.82 | 0.76 | 0.81 | 0.76 | 0.61 | 0.72 | 0.54 | 0.78 | 0.93 | 1.00 | 0.66 | 0.75 | 0.65 |
| 18 | 0.66 | 0.63 | 0.60 | 0.63 | 0.66 | 0.66 | 0.65 | 0.64 | 0.64 | 0.66 | 0.64 | 0.71 | 0.64 | 0.69 | 0.62 | 0.64 | 0.66 | 1.00 | 0.73 | 0.90 |
| 19 | 0.72 | 0.71 | 0.67 | 0.71 | 0.74 | 0.75 | 0.73 | 0.69 | 0.74 | 0.77 | 0.74 | 0.72 | 0.72 | 0.64 | 0.66 | 0.73 | 0.75 | 0.73 | 1.00 | 0.71 |
| 20 | 0.67 | 0.64 | 0.60 | 0.63 | 0.65 | 0.66 | 0.64 | 0.65 | 0.65 | 0.65 | 0.65 | 0.70 | 0.65 | 0.69 | 0.63 | 0.63 | 0.65 | 0.90 | 0.71 | 1.00 |

### Electrode pop

**Table 62:** Correlation matrix of the 20 best performing classifiers on the electrode pop label.

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1.00 | 0.92 | 0.88 | 0.89 | 0.78 | 0.80 | 0.84 | 0.85 | 0.84 | 0.81 | 0.85 | 0.68 | 0.79 | 0.66 | 0.52 | 0.74 | 0.70 | 0.76 | 0.68 | 0.65 |
| 2 | 0.92 | 1.00 | 0.90 | 0.91 | 0.78 | 0.81 | 0.84 | 0.85 | 0.85 | 0.82 | 0.85 | 0.67 | 0.80 | 0.66 | 0.53 | 0.74 | 0.70 | 0.77 | 0.69 | 0.64 |
| 3 | 0.88 | 0.90 | 1.00 | 0.88 | 0.76 | 0.78 | 0.82 | 0.83 | 0.83 | 0.79 | 0.83 | 0.66 | 0.78 | 0.64 | 0.52 | 0.72 | 0.69 | 0.76 | 0.66 | 0.64 |
| 4 | 0.89 | 0.91 | 0.88 | 1.00 | 0.77 | 0.80 | 0.83 | 0.84 | 0.84 | 0.80 | 0.84 | 0.67 | 0.78 | 0.65 | 0.52 | 0.73 | 0.70 | 0.76 | 0.68 | 0.64 |
| 5 | 0.78 | 0.78 | 0.76 | 0.77 | 1.00 | 0.83 | 0.82 | 0.82 | 0.82 | 0.84 | 0.81 | 0.82 | 0.71 | 0.77 | 0.52 | 0.83 | 0.83 | 0.71 | 0.74 | 0.78 |
| 6 | 0.80 | 0.81 | 0.78 | 0.80 | 0.83 | 1.00 | 0.79 | 0.79 | 0.78 | 0.87 | 0.77 | 0.75 | 0.73 | 0.76 | 0.50 | 0.85 | 0.77 | 0.71 | 0.80 | 0.72 |
| 7 | 0.84 | 0.84 | 0.82 | 0.83 | 0.82 | 0.79 | 1.00 | 0.92 | 0.91 | 0.78 | 0.90 | 0.72 | 0.74 | 0.67 | 0.52 | 0.76 | 0.72 | 0.76 | 0.68 | 0.68 |
| 8 | 0.85 | 0.85 | 0.83 | 0.84 | 0.82 | 0.79 | 0.92 | 1.00 | 0.92 | 0.78 | 0.91 | 0.71 | 0.75 | 0.66 | 0.53 | 0.76 | 0.71 | 0.77 | 0.67 | 0.67 |
| 9 | 0.84 | 0.85 | 0.83 | 0.84 | 0.82 | 0.78 | 0.91 | 0.92 | 1.00 | 0.78 | 0.91 | 0.71 | 0.76 | 0.65 | 0.54 | 0.75 | 0.71 | 0.76 | 0.67 | 0.67 |
| 10 | 0.81 | 0.82 | 0.79 | 0.80 | 0.84 | 0.87 | 0.78 | 0.78 | 0.78 | 1.00 | 0.78 | 0.76 | 0.73 | 0.78 | 0.51 | 0.82 | 0.79 | 0.72 | 0.77 | 0.74 |
| 11 | 0.85 | 0.85 | 0.83 | 0.84 | 0.81 | 0.77 | 0.90 | 0.91 | 0.91 | 0.78 | 1.00 | 0.70 | 0.75 | 0.65 | 0.54 | 0.74 | 0.71 | 0.78 | 0.66 | 0.67 |
| 12 | 0.68 | 0.67 | 0.66 | 0.67 | 0.82 | 0.75 | 0.72 | 0.71 | 0.71 | 0.76 | 0.70 | 1.00 | 0.62 | 0.81 | 0.48 | 0.77 | 0.84 | 0.64 | 0.76 | 0.89 |
| 13 | 0.79 | 0.80 | 0.78 | 0.78 | 0.71 | 0.73 | 0.74 | 0.75 | 0.76 | 0.73 | 0.75 | 0.62 | 1.00 | 0.61 | 0.51 | 0.68 | 0.65 | 0.69 | 0.63 | 0.60 |
| 14 | 0.66 | 0.66 | 0.64 | 0.65 | 0.77 | 0.76 | 0.67 | 0.66 | 0.65 | 0.78 | 0.65 | 0.81 | 0.61 | 1.00 | 0.44 | 0.78 | 0.77 | 0.60 | 0.83 | 0.83 |
| 15 | 0.52 | 0.53 | 0.52 | 0.52 | 0.52 | 0.50 | 0.52 | 0.53 | 0.54 | 0.51 | 0.54 | 0.48 | 0.51 | 0.44 | 1.00 | 0.49 | 0.50 | 0.50 | 0.44 | 0.47 |
| 16 | 0.74 | 0.74 | 0.72 | 0.73 | 0.83 | 0.85 | 0.76 | 0.76 | 0.75 | 0.82 | 0.74 | 0.77 | 0.68 | 0.78 | 0.49 | 1.00 | 0.78 | 0.68 | 0.82 | 0.75 |
| 17 | 0.70 | 0.70 | 0.69 | 0.70 | 0.83 | 0.77 | 0.72 | 0.71 | 0.71 | 0.79 | 0.71 | 0.84 | 0.65 | 0.77 | 0.50 | 0.78 | 1.00 | 0.65 | 0.71 | 0.82 |
| 18 | 0.76 | 0.77 | 0.76 | 0.76 | 0.71 | 0.71 | 0.76 | 0.77 | 0.76 | 0.72 | 0.78 | 0.64 | 0.69 | 0.60 | 0.50 | 0.68 | 0.65 | 1.00 | 0.62 | 0.61 |
| 19 | 0.68 | 0.69 | 0.66 | 0.68 | 0.74 | 0.80 | 0.68 | 0.67 | 0.67 | 0.77 | 0.66 | 0.76 | 0.63 | 0.83 | 0.44 | 0.82 | 0.71 | 0.62 | 1.00 | 0.76 |
| 20 | 0.65 | 0.64 | 0.64 | 0.64 | 0.78 | 0.72 | 0.68 | 0.67 | 0.67 | 0.74 | 0.67 | 0.89 | 0.60 | 0.83 | 0.47 | 0.75 | 0.82 | 0.61 | 0.76 | 1.00 |

### Muscle movement

**Table 63:** Correlation matrix of the 20 best performing classifiers on the muscle movement label.

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1.00 | 0.98 | 0.81 | 0.81 | 0.80 | 0.80 | 0.93 | 0.93 | 0.80 | 0.69 | 0.62 | 0.62 | 0.59 | 0.68 | 0.66 | 0.59 | 0.59 | 0.61 | 0.60 | 0.59 |
| 2 | 0.98 | 1.00 | 0.81 | 0.81 | 0.80 | 0.80 | 0.93 | 0.93 | 0.80 | 0.69 | 0.62 | 0.62 | 0.60 | 0.68 | 0.66 | 0.60 | 0.59 | 0.61 | 0.60 | 0.60 |
| 3 | 0.81 | 0.81 | 1.00 | 0.98 | 0.97 | 0.97 | 0.85 | 0.85 | 0.94 | 0.77 | 0.61 | 0.62 | 0.59 | 0.70 | 0.68 | 0.59 | 0.59 | 0.63 | 0.61 | 0.59 |
| 4 | 0.81 | 0.81 | 0.98 | 1.00 | 0.98 | 0.98 | 0.84 | 0.84 | 0.96 | 0.77 | 0.60 | 0.62 | 0.58 | 0.70 | 0.68 | 0.59 | 0.58 | 0.62 | 0.61 | 0.58 |
| 5 | 0.80 | 0.80 | 0.97 | 0.98 | 1.00 | 0.99 | 0.84 | 0.84 | 0.96 | 0.77 | 0.60 | 0.62 | 0.58 | 0.69 | 0.68 | 0.59 | 0.58 | 0.62 | 0.61 | 0.58 |
| 6 | 0.80 | 0.80 | 0.97 | 0.98 | 0.99 | 1.00 | 0.84 | 0.84 | 0.97 | 0.78 | 0.60 | 0.62 | 0.58 | 0.69 | 0.68 | 0.58 | 0.58 | 0.62 | 0.61 | 0.58 |
| 7 | 0.93 | 0.93 | 0.85 | 0.84 | 0.84 | 0.84 | 1.00 | 0.98 | 0.83 | 0.71 | 0.62 | 0.62 | 0.59 | 0.68 | 0.66 | 0.60 | 0.59 | 0.62 | 0.61 | 0.60 |
| 8 | 0.93 | 0.93 | 0.85 | 0.84 | 0.84 | 0.84 | 0.98 | 1.00 | 0.83 | 0.71 | 0.62 | 0.62 | 0.59 | 0.68 | 0.66 | 0.59 | 0.59 | 0.62 | 0.60 | 0.59 |
| 9 | 0.80 | 0.80 | 0.94 | 0.96 | 0.96 | 0.97 | 0.83 | 0.83 | 1.00 | 0.77 | 0.60 | 0.61 | 0.57 | 0.69 | 0.67 | 0.58 | 0.57 | 0.62 | 0.60 | 0.58 |
| 10 | 0.69 | 0.69 | 0.77 | 0.77 | 0.77 | 0.78 | 0.71 | 0.71 | 0.77 | 1.00 | 0.54 | 0.56 | 0.52 | 0.62 | 0.61 | 0.53 | 0.53 | 0.56 | 0.55 | 0.53 |
| 11 | 0.62 | 0.62 | 0.61 | 0.60 | 0.60 | 0.60 | 0.62 | 0.62 | 0.60 | 0.54 | 1.00 | 0.89 | 0.86 | 0.75 | 0.76 | 0.87 | 0.90 | 0.86 | 0.83 | 0.87 |
| 12 | 0.62 | 0.62 | 0.62 | 0.62 | 0.62 | 0.62 | 0.62 | 0.62 | 0.61 | 0.56 | 0.89 | 1.00 | 0.82 | 0.80 | 0.80 | 0.85 | 0.88 | 0.91 | 0.87 | 0.84 |
| 13 | 0.59 | 0.60 | 0.59 | 0.58 | 0.58 | 0.58 | 0.59 | 0.59 | 0.57 | 0.52 | 0.86 | 0.82 | 1.00 | 0.73 | 0.73 | 0.91 | 0.85 | 0.81 | 0.83 | 0.92 |
| 14 | 0.68 | 0.68 | 0.70 | 0.70 | 0.69 | 0.69 | 0.68 | 0.68 | 0.69 | 0.62 | 0.75 | 0.80 | 0.73 | 1.00 | 0.94 | 0.75 | 0.74 | 0.83 | 0.82 | 0.73 |
| 15 | 0.66 | 0.66 | 0.68 | 0.68 | 0.68 | 0.68 | 0.66 | 0.66 | 0.67 | 0.61 | 0.76 | 0.80 | 0.73 | 0.94 | 1.00 | 0.75 | 0.75 | 0.84 | 0.83 | 0.74 |
| 16 | 0.59 | 0.60 | 0.59 | 0.59 | 0.59 | 0.58 | 0.60 | 0.59 | 0.58 | 0.53 | 0.87 | 0.85 | 0.91 | 0.75 | 0.75 | 1.00 | 0.87 | 0.84 | 0.86 | 0.92 |
| 17 | 0.59 | 0.59 | 0.59 | 0.58 | 0.58 | 0.58 | 0.59 | 0.59 | 0.57 | 0.53 | 0.90 | 0.88 | 0.85 | 0.74 | 0.75 | 0.87 | 1.00 | 0.85 | 0.83 | 0.86 |
| 18 | 0.61 | 0.61 | 0.63 | 0.62 | 0.62 | 0.62 | 0.62 | 0.62 | 0.62 | 0.56 | 0.86 | 0.91 | 0.81 | 0.83 | 0.84 | 0.84 | 0.85 | 1.00 | 0.90 | 0.83 |
| 19 | 0.60 | 0.60 | 0.61 | 0.61 | 0.61 | 0.61 | 0.61 | 0.60 | 0.60 | 0.55 | 0.83 | 0.87 | 0.83 | 0.82 | 0.83 | 0.86 | 0.83 | 0.90 | 1.00 | 0.85 |
| 20 | 0.59 | 0.60 | 0.59 | 0.58 | 0.58 | 0.58 | 0.60 | 0.59 | 0.58 | 0.53 | 0.87 | 0.84 | 0.92 | 0.73 | 0.74 | 0.92 | 0.86 | 0.83 | 0.85 | 1.00 |

### Null

**Table 64:** Correlation matrix of the 20 best performing classifiers on the null label.

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1.00 | 0.91 | 0.92 | 0.98 | 0.97 | 0.86 | 0.97 | 0.90 | 0.87 | 0.89 | 0.91 | 0.91 | 0.97 | 0.79 | 0.85 | 0.85 | 0.87 | 0.85 | 0.83 | 0.87 |
| 2 | 0.91 | 1.00 | 0.97 | 0.92 | 0.92 | 0.91 | 0.92 | 0.93 | 0.94 | 0.93 | 0.96 | 0.95 | 0.91 | 0.86 | 0.90 | 0.90 | 0.94 | 0.92 | 0.91 | 0.94 |
| 3 | 0.92 | 0.97 | 1.00 | 0.92 | 0.92 | 0.90 | 0.92 | 0.93 | 0.94 | 0.93 | 0.96 | 0.96 | 0.91 | 0.85 | 0.90 | 0.90 | 0.94 | 0.92 | 0.90 | 0.94 |
| 4 | 0.98 | 0.92 | 0.92 | 1.00 | 0.98 | 0.86 | 0.98 | 0.90 | 0.87 | 0.90 | 0.91 | 0.91 | 0.97 | 0.79 | 0.85 | 0.86 | 0.87 | 0.85 | 0.83 | 0.87 |
| 5 | 0.97 | 0.92 | 0.92 | 0.98 | 1.00 | 0.86 | 0.97 | 0.90 | 0.87 | 0.90 | 0.91 | 0.91 | 0.97 | 0.80 | 0.85 | 0.86 | 0.87 | 0.85 | 0.83 | 0.87 |
| 6 | 0.86 | 0.91 | 0.90 | 0.86 | 0.86 | 1.00 | 0.86 | 0.95 | 0.92 | 0.94 | 0.89 | 0.89 | 0.85 | 0.87 | 0.95 | 0.97 | 0.91 | 0.92 | 0.92 | 0.91 |
| 7 | 0.97 | 0.92 | 0.92 | 0.98 | 0.97 | 0.86 | 1.00 | 0.90 | 0.87 | 0.90 | 0.91 | 0.91 | 0.97 | 0.79 | 0.85 | 0.86 | 0.87 | 0.85 | 0.83 | 0.87 |
| 8 | 0.90 | 0.93 | 0.93 | 0.90 | 0.90 | 0.95 | 0.90 | 1.00 | 0.92 | 0.98 | 0.91 | 0.91 | 0.89 | 0.85 | 0.93 | 0.94 | 0.91 | 0.90 | 0.89 | 0.91 |
| 9 | 0.87 | 0.94 | 0.94 | 0.87 | 0.87 | 0.92 | 0.87 | 0.92 | 1.00 | 0.92 | 0.93 | 0.94 | 0.87 | 0.89 | 0.91 | 0.93 | 0.96 | 0.95 | 0.94 | 0.97 |
| 10 | 0.89 | 0.93 | 0.93 | 0.90 | 0.90 | 0.94 | 0.90 | 0.98 | 0.92 | 1.00 | 0.91 | 0.91 | 0.89 | 0.85 | 0.92 | 0.94 | 0.91 | 0.90 | 0.89 | 0.91 |
| 11 | 0.91 | 0.96 | 0.96 | 0.91 | 0.91 | 0.89 | 0.91 | 0.91 | 0.93 | 0.91 | 1.00 | 0.98 | 0.91 | 0.86 | 0.89 | 0.89 | 0.95 | 0.92 | 0.90 | 0.94 |
| 12 | 0.91 | 0.95 | 0.96 | 0.91 | 0.91 | 0.89 | 0.91 | 0.91 | 0.94 | 0.91 | 0.98 | 1.00 | 0.91 | 0.86 | 0.89 | 0.89 | 0.95 | 0.92 | 0.90 | 0.95 |
| 13 | 0.97 | 0.91 | 0.91 | 0.97 | 0.97 | 0.85 | 0.97 | 0.89 | 0.87 | 0.89 | 0.91 | 0.91 | 1.00 | 0.79 | 0.84 | 0.85 | 0.87 | 0.84 | 0.83 | 0.87 |
| 14 | 0.79 | 0.86 | 0.85 | 0.79 | 0.80 | 0.87 | 0.79 | 0.85 | 0.89 | 0.85 | 0.86 | 0.86 | 0.79 | 1.00 | 0.87 | 0.87 | 0.89 | 0.92 | 0.92 | 0.89 |
| 15 | 0.85 | 0.90 | 0.90 | 0.85 | 0.85 | 0.95 | 0.85 | 0.93 | 0.91 | 0.92 | 0.89 | 0.89 | 0.84 | 0.87 | 1.00 | 0.94 | 0.91 | 0.91 | 0.91 | 0.90 |
| 16 | 0.85 | 0.90 | 0.90 | 0.86 | 0.86 | 0.97 | 0.86 | 0.94 | 0.93 | 0.94 | 0.89 | 0.89 | 0.85 | 0.87 | 0.94 | 1.00 | 0.91 | 0.91 | 0.91 | 0.91 |
| 17 | 0.87 | 0.94 | 0.94 | 0.87 | 0.87 | 0.91 | 0.87 | 0.91 | 0.96 | 0.91 | 0.95 | 0.95 | 0.87 | 0.89 | 0.91 | 0.91 | 1.00 | 0.96 | 0.94 | 0.97 |
| 18 | 0.85 | 0.92 | 0.92 | 0.85 | 0.85 | 0.92 | 0.85 | 0.90 | 0.95 | 0.90 | 0.92 | 0.92 | 0.84 | 0.92 | 0.91 | 0.91 | 0.96 | 1.00 | 0.97 | 0.96 |
| 19 | 0.83 | 0.91 | 0.90 | 0.83 | 0.83 | 0.92 | 0.83 | 0.89 | 0.94 | 0.89 | 0.90 | 0.90 | 0.83 | 0.92 | 0.91 | 0.91 | 0.94 | 0.97 | 1.00 | 0.94 |
| 20 | 0.87 | 0.94 | 0.94 | 0.87 | 0.87 | 0.91 | 0.87 | 0.91 | 0.97 | 0.91 | 0.94 | 0.95 | 0.87 | 0.89 | 0.90 | 0.91 | 0.97 | 0.96 | 0.94 | 1.00 |