Computing and Informatics, Vol. 40, 2021, 850–879, doi: 10.31577/cai_2021_4_850

COMPARISON OF MANUAL AND AUTOMATED FEATURE ENGINEERING FOR DAILY ACTIVITY CLASSIFICATION IN MENTAL DISORDER DIAGNOSIS

Jakub Adamczyk, Filip Malawski

AGH University of Science and Technology
Faculty of Computer Science, Electronics and Telecommunications
Institute of Computer Science
al. A. Mickiewicza 30, 30-059 Krakow, Poland
e-mail: jadamczyk@student.agh.edu.pl, fmal@agh.edu.pl

Abstract. Motor activity data allows for analysis of complex behavioral patterns, including the diagnosis of mental disorders, such as depression or schizophrenia. However, the classification of actigraphy signals remains a challenge. The main reasons are small datasets and the need for sophisticated feature engineering. The recent development of AutoML approaches allows for automating feature extraction and selection. In this work, we compare automatic and manual feature engineering for applications in mental health. We also analyze classifier evaluation methods for small datasets. The automated approach results in better classification, as measured with several metrics, and in a shorter, cleaner code, providing software engineering advantages.

Keywords: Feature engineering, feature selection, activity classification, time series, mental disorder diagnosis, AutoML, actigraphy, signal processing

Mathematics Subject Classification 2010: 62H30, 62M10, 62P15, 68T10

1 INTRODUCTION

Depression, formally known as Major Depressive Disorder (MDD), and schizophrenia, are among the most common mental disorders [3]. According to the WHO, depression is the single largest cause of disability, affecting 7.5 % of the global population [4]. Schizophrenia, while being less common, results in serious disability and

a very significant risk of suicide, up to a 37-fold increase in comparison to the general population [5]. The common element of those two psychiatric disorders is presence of abnormal activity patterns as one of the distinguishing symptoms [6, 7, 8, 9].

The daily activity behavioral patterns, however, are problematic as a clinical diagnosis tool, since they require constant monitoring for an extended period of time by a medical professional. The recent advancements in Internet of Things (IoT) sensors, especially wearable technologies, can be a solution to this problem [10]. The major advantages of this approach are low cost, convenience to the patient, and providing objective measurements. Actigraph IoT devices, very similar to watches and often combined with watch functionality, can be used to record the continuous 1D signal with the wearer's activity, based on the accelerometers built into them. Data from such sensors has been applied in the mental health domain to support the diagnosis of bipolar disorder [11], ADHD [12], depression [1] and schizophrenia [2]. We focus on the last two since datasets for classification have been published for them.

Statistical approaches have been used in this area in the past, but they can be improved. Some were performed before IoT devices became widely available, and almost all of them were limited to simple statistical analysis, such as calculating linear correlation, performing statistical tests, and calculating p-values, e.g. [6, 8]. They also extracted statistical features but without formal mathematical justification. More recent developments focus on using machine learning to automatically perform diagnosis [13, 14].

From a machine learning perspective, the task is a binary classification of time series, i.e. assign a "healthy" or "sick" class to a given signal. A naive approach would be to feed the data directly to a classifier, but it results in very long, dense vectors of variable length, which is unsuitable for typical classification algorithms. While neural networks such as Gated Recurrent Unit (GRU) or Long-Short Term Memory (LSTM) can work with it, they are known to require a significant amount of data to train, and available datasets are very small. For those reasons, the approach taken by most papers is a classical 2-step process: extract statistical features and use a regular classifier for tabular data [2, 13, 15].

This approach can be performed manually, where researchers choose features themselves. However, this requires extensive knowledge of many areas (e.g. psychiatry, statistics, signal processing), a long trial-and-error phase, and often relies on the subjective decisions of the scientist. Automated ML approaches seek to fix those problems by automating either feature extraction [16] or the entire process of classification [19, 20]. We focus on the former, i.e. automated extraction of meaningful statistical features from signals for classification.

We use Depresjon [1] and Psykose [2] datasets, for diagnosis of depression and schizophrenia, respectively. In both cases, they contain 24 h motor activity signals measured with actigraph watches. The sampling frequency is 32 Hz, movements over 0.05 g are recorded, and for each minute total activity count is stored, so the output is in gravitational acceleration units per minute. Authors perform exploratory data analysis, observing, e.g. an overall lower activity for depressed patients, or con-

siderably lower standard deviation of activity for schizophreniacs, compared to the control group. The important characteristic of both datasets is a very low number of samples, in comparison to the sizes typical for machine learning datasets: 55 in Depresjon and 54 in Psykose. This undermines the statistical assumptions underlying the typical test-validation-training division, that some papers seem to take for granted. We explore this in greater depth, and, similarly to the authors of Psykose, recognize the need for more sophisticated testing approaches, showing that naive measurements lead to grossly overestimated performance estimates.

The research hypothesis for this paper is: a software engineering approach to ML problems in the classification of time series for mental disorder diagnosis using AutoML feature extraction tools gives results on par with the manual approach while requiring less knowledge, work, and time.

The rest of the paper is organized in the following way. In Section 2 we perform an overview of other papers related to time series classification for mental disorder diagnosis. In Section 3 we outline manual and automated approaches to feature engineering, machine learning algorithms used for evaluation, and discuss evaluation methods, problems with the most common approaches, and potential solutions. In Section 4 we provide the results of our experiments and discuss the results. We also show what additional data insights automated methods provide. Section 5 contains the overall conclusions.

2 RELATED WORK

Analysis of accelerometer data for medical applications is also known as actigraphy and has been extensively researched. However, traditional analyses carried out mostly by psychologists or psychiatrists are purely statistical. In [6], just 3 simple features are used: MESOR (mean daily activity), amplitude, and acrophase (timing of the activity peak), and they are compared between depressed patients and controls. While [11] applies machine learning, the main focus is on extracting and comparing simple features from activity signals between conditions and controls, e.g. MESOR, mean activity during 10 most active hours (M10), mean activity during 5 least active hours (L5) or acrophase.

Solutions that apply ML techniques for mental disorder classification from actigraphy data first extract features from activity signals and then use tabular classification algorithms. However, the features extracted are still simple, there are only a few and their choice heavily relies on the researcher's experience. The approach presented in [12] relies on only 4 domain-specific features: L5, diurnal skew (skewness for daylight hours), relative circadian amplitude (amplitude scaled to range $[0, 1]$), and vulnerability index. In [15] only the most basic statistical features are extracted from the signal, such as mean, variance, or interquartile range. Authors of [13] propose feature extraction additionally in the frequency domain, using power spectral density (PSD), calculating, e.g. mean, variance, entropy, or spectral flatness in the frequency domain. This seems quite rare, despite the popularity of spectral features

in signal processing. The method presented in [14] uses just 3 features for depression classification: mean activity level, standard deviation, and proportion of zeros (fraction of minutes with zero activity). All works described above use the Depresjon dataset; for schizophrenia classification, we failed to find any paper apart from the Psykose dataset paper [2], in which baselines are presented, using mean, standard deviation, and proportion of zeros.

Automated feature extraction from time series has been described in papers accompanying software libraries, e.g. tsfresh [16, 29]. Automated techniques have been used for classification of audio data [22] or assessing the fault severity of industrial gearboxes [21]. In computational medicine applications are sparse and focus on EEG or ECG data, focusing exclusively on frequency domain features, based on Fourier or wavelet transforms [24, 23]. To the best of our knowledge, there are no applications of automated feature extraction for mental disorder diagnosis.

The problem of classifier evaluation is well-grounded in the ML community. Multiple techniques are described in [25] and [26], including popular methods such as holdout and cross-validation, but also more sophisticated bootstrap-based algorithms such as 0.632 estimator. The problem of properly measuring classifier performance on mental disorder datasets, because of their small sample size, has been mentioned in [2], but only simple cross-validation is suggested.

To sum up, the existing works concerning the classification of actigraphy time series use only manual feature engineering. Works from other areas suggest that automated feature engineering from time series works well and has great potential in the mental health domain. In this work, we explore this area, using AutoML in the classification of mental disorders based on actigraphy signals.

3 METHODS

This section is organized as follows. In Subsection 3.1, manual feature extraction is described, while in Subsection 3.2 we outline the automated approaches. In Subsection 3.3 we briefly describe the steps between feature extraction and classification phases. In Subsection 3.4, the classification algorithms used are briefly described. In Subsection 3.5, the evaluation techniques are discussed, and they are compared with those used in other works in Subsection 3.6.

3.1 Manual Feature Engineering

To fairly compare manual and automated approaches, a good set of properly discriminative features has to be extracted from the datasets. As both Depresjon and Psykose feature the same kind of signal, with the identical method of measurement, we propose using the same set of features for both. For more details on the datasets see Section 4.1.

Manual feature engineering is highly interpretable and therefore choosing particular features should be properly justified. Moreover, domain knowledge should

be applied to properly preprocess data, e.g. eliminate redundant features. To this end, interpretation has been provided next to all chosen features in 1. It should be noted that we found no such detailed justification for choosing particular features in any related paper, and often there is no justification at all.

We extracted basic statistical features from time domain signals based on previous articles using Depresjon and Psykose datasets [13, 15, 2]. Additionally, since the actigraphy data is a signal, we included frequency domain features [13]. We used Welch's method to estimate the power spectral density (PSD) and extract features from it. In both domains we extracted statistical features such as minimum, maximum, mean, variance, skewness, or interquartile range. Entropy has also been calculated in both domains. We also included the proportion of zeros in the time domain and spectral flatness in the frequency domain.

Justifications provided in Table 1 for features calculated in both domains relate to the time domain. For the frequency domain, it is hard to provide intuitive interpretations. Instead, the reason to include them is that those statistical features provide an overall description of the shape of the power spectral density distribution.

Feature	**Justification**
Minimum	May be higher than 0 for patients with sleep problems, typical for depression and schizophrenia
Maximum	Highest activity, related e.g. to sports or physical activity; high values are more likely for healthy controls
Mean	Measure of overall activity, lower for depressed patients
Median	Similar to mean, but better suited for asymmetrical distributions
Variance	Bipolar depression and schizophrenia may lead to higher variance due to psychotic episodes
Kurtosis	Measures heaviness of tails of distribution, e.g. presence of outliers like mood swings in bipolar depression
Skewness	Distribution may be positively skewed for depressed patients with lower activity
Coefficient of variation	Combines variability (important for detecting mood swings) and mean activity (important for detecting depressive episodes)
Interquartile range (IQR)	Combines Q1 and Q3, describes dispersion around median
Trimmed mean (10 %)	Like mean, but less sensitive to outliers
Entropy	Measure of uncertainty and randomness, may indicate psychotic behavior in schizophrenia
Proportion of zeros	High percentage of zeros, especially during daytime, may indicate hypersomnia (oversleeping) typical for depression
Spectral flatness	Describes how noise-like is the activity, this may indicate randomness typical for psychosis or paranoia in schizophrenia

Table 1. Manually extracted features

Activity signals have a lot of variation and noise, but we are interested in the overall long-term effects to detect the mental disorder. For this reason, we perform experiments with resampling data in the time domain before feature engineering, replacing the signal with the mean activity for each hour, same as in [13]. This should be particularly effective with features based on minimal values since for raw signals the minimum is always 0 (there is always a period when someone does not move).

We perform experiments on 3 subsets for each dataset: full 24 h data, night hours only (from 21:00 to 8:00) and day hours only (from 8:00 to 21:00), similarly to [13]. This is because depression and schizophrenia especially affect sleep patterns [13, 3, 27], which should be more clearly visible in night-only data.

3.2 Automated Feature Engineering

Automated approaches in machine learning are gaining popularity, as they greatly simplify the parts or even the whole pipeline of data processing, from feature extraction to classification and tuning. Here we focus only on automated feature engineering, since automating this part is arguably the most important, while we still retain the flexibility of choosing different approaches to further features processing, selection, and choosing classification algorithms. Additionally, automating this part is relatively easy, compared, e.g. to hyperparameter tuning. The most notable advantages of the automated approach are:

- little domain knowledge is required – many, often sophisticated, feature calculation algorithms are already built-in;
- discovering new features – a wide range of possible features is evaluated, instead of only those defined by a particular expert;
- shorter, cleaner code – particularly important to real-world applications, where code maintenance is a considerable burden;
- quantitative way of feature selection – the subjectivity of feature engineering is replaced with well theoretically grounded, mathematical process.

For Python programming language, the most well-known library for automated feature extraction from time series is tsfresh (Time Series FeatuRe Extraction based on Scalable Hypothesis tests) [16, 29]. Its main goal is to provide an easy-to-use tool for feature extraction and selection using statistical tests.

In the tsfresh library (version 0.18.0), over 70 basic features are extracted in time and frequency domains. They range from simple statistical features such as minimum, maximum, mean, variance, or quantiles, through lesser-known ones like absolute energy, absolute sum of changes and counts below/above mean, to sophisticated features like Lempel-Ziv complexity, wavelet transform or time reversal asymmetry statistic. Many of those basic features are then parametrized (e.g. autocorrelation is calculated for multiple lag values), resulting in about 800 features.

Notably, tsfresh already calculates all features that we chose manually in the previous section, except for proportion of zeros. In this step, functions for calculating custom features can be added, so it is easy to create a half-automated approach, using both manual and automated feature extraction.

After extraction, automated feature selection is performed. In tsfresh "relevant" variable is defined as feature X that is not independent of target variable Y, therefore hypothesis testing is used for feature selection. A nonparametric statistical significance test is performed independently for each feature, measuring predictive power. For binary classification by default this is the Mann-Whitney U test (for continuous features) or the Fisher's exact test (for binary features) [17]. Then the Benjamini-Yekutieli procedure [18] with $\alpha = 0.05$ is used to reject hypotheses (i.e. remove features). It is used for two reasons: it does not assume hypothesis independence and can control the false discovery rate (FDR, equal to α), which would otherwise be a considerable problem with testing such a high number of hypotheses. This way, irrelevant or redundant features are not included in the final set.

Considerable thought has been given to scalability and efficiency. Feature extraction and statistical testing are fully parallelized, and intermediate calculation results are shared and reused. There are 3 profiles for feature extraction settings: minimal (extracting only the dozen most commonly useful features), efficient (extracting about 800 features), and comprehensive (extracting a massive number of features, with almost all possible parameters). This emphasis on efficiency is important even for small datasets in the mental health domain since actigraphy measurements are very long (5–20 days, see Section 4.1) and feature extraction may take a very long time.

3.3 Feature Selection and Processing

After feature extraction (both manual and automated), we additionally perform two steps: univariate feature selection with variance thresholding and standardization (subtracting the mean and dividing by standard deviation). Features with low variance do not discriminate well and therefore are not particularly useful for classification. Standardization, on the other hand, is required by regularized logistic regression using L1/LASSO regularization [26] and by SVM [30].

For feature selection, we scale the data to range $[0, 1]$, to be able to directly compare variances of features with different scales. Then features with variance lower than 0.05 are rejected.

Calculations for those two steps are performed using the training set only. For example, the variance of each feature is calculated using training data, features are selected for elimination and then removed from both training and test data. This ensures that the model is not tuned based on the test set, i.e. we avoid data leakage (using knowledge from test data during training). For details on the procedure for training/test split see Section 3.5.4.

3.4 Classification Algorithms

For evaluating feature extraction approaches, we have chosen 3 popular algorithms: logistic regression (LR) (with ElasticNet regularization [31], i.e. both L1 and L2), Support Vector Machine (SVM) with RBF kernel, and Random Forest (RF).

Those particular algorithms have been chosen since in this way we have one linear classifier and two very different nonlinear classifiers. Additionally, all of them are popular in the classification of medical data: LR due to simplicity, kernelized SVM gives very good results for high-dimensional problems with a low number of samples typical for this domain, and RF tends to give good results with little hyperparameter tuning required [26].

We use grid search for hyperparameter tuning. We tune:

- LR:
  - C – regularization strength, the strength is directly proportional to C; we check values: $[0.001, 0.01, 0.1, 0.5, 1, 2, 5, 10, 25, 50, 100, 500, 1\,000]$.
  - l1_ratio – how much of the regularization is L1 and how much is L2; we check values from 0 to 1, spaced equally by 0.05.
- SVM:
  - C – regularization strength, the strength is inversely proportional to C; it is equal to the squared L2 regularization; we check 50 points spaced evenly on a log scale (base 10) from 10e−3 to 10e3.
  - γ – RBF kernel parameter; we check the same values as for C.

For RF, we set the number of trees to 500 (following [26]) and use the default values for other hyperparameters.

For all 3 algorithms we additionally tune the class weighting:

- equal – no weighting;
- class-balanced – weights inversely proportional to class frequencies;
- subsample-balanced – only for RF, class-balanced separately for each bootstrap sample.

For details on those parameters, please refer to the Scikit-learn documentation [32].

3.5 Performance Evaluation

Estimating the generalization performance of classifiers is a nontrivial task, which has to take into consideration multiple factors, e.g. dataset size and dimensionality, type of classifier, possible effects of outliers, and avoiding data leakage from test data to model training. For this reason, there are multiple methods available, suitable

for different situations. Relevant papers that we have examined used 3 popular and simple approaches to estimating generalization performance: holdout, cross-validation (CV), and bootstrap-based Out-Of-Bag (OOB).

Since the terminology in this area can be ambiguous, we will use the following definitions. The *dataset* is the entirety of data available. It is divided into *training data*, used for training and validating the model, and *test set*, used to estimate the generalization performance of the classifier. The test set is not used until the very end when we test the production-ready classifier on it. Training data is split into *train set* and *validation set*, where the train set is used to calculate the classifier's parameters, and its performance is checked on the validation set, which serves as a basis for hyperparameter tuning and model selection.

3.5.1 Holdout Method

Holdout method has been used in Psykose dataset baselines [2] and in [13]. In this approach, a random subset is chosen as a test set, and the rest is the training data. In [13] it is 30 % of the data, and in [2] experiments are performed for sizes from 10 % to 90 %. The theoretical justification of this approach is that randomly choosing the test set is representative of the future samples and that it follows approximately the same distribution as the test set. However, while this method is popular in machine learning [25], especially in computer vision and natural language processing, it assumes a sufficiently large dataset. This assumption is important both to satisfy the condition of similar distribution of training and test data, and to keep enough data in the training set to train a classifier with high enough capacity, without significant risk of overfitting. This is not the case for Depresjon and Psykose datasets, which have 55 and 54 samples, respectively. Taking 30 % of the data for testing results in only 16–17 samples, which is not nearly enough, especially since we have high-dimensional data with many features, possibly even $d >> n$ (n – number of samples, d – number of dimensions/features). Since the underlying assumptions of this method are not met, we recommend refraining from using it in this domain.

3.5.2 Cross-Validation (CV)

Multiple papers [1, 12, 14, 2] use cross-validation (CV) approach, where the data is divided into k folds, and each fold acts as a test set and the rest of the data is merged into training data. This process is performed for each fold, so the mean test metrics and their standard deviations can be reported. The extreme case of $k = 1$ is called Leave-One-Out CV (LOOCV), where just a single sample is a test set at a given time.

Authors of [1] use 10-fold CV, in [12] 4-fold CV is used, and two works [14, 2] utilize Leave-One-Out CV (LOOCV).

The bias and variance of the error estimate depend on k. LOOCV is an unbiased estimator of the true prediction error [26]; however, it is typically pessimistic [26]. For small datasets such as Depresjon or Psykose high variance may give misleading

results, and following [26] we recommend using 5-fold CV with stratification (due to class imbalance). This number of folds combined with stratification has a few desirable properties (the numbers provided are for Depresjon and Psykose datasets):

- each test set is quite large, compared to the overall size of the dataset (11 samples), therefore each fold will give a reasonable estimate of the classifier performance;
- training set is large enough (44 samples) to train classifiers to recognize both classes;
- both training and test set have enough samples of both classes to allow variability for metrics (such as accuracy or recall): 4–5 patients and 6–7 controls for the test set, 18–19 patients and 25–26 controls for the training set;
- this number of folds is a compromise between bias and variance in estimating classifier performance [26];
- this number of folds is enough to measure how sensitive the classifier is to changes in test set distribution (standard deviation of metrics on test folds).

However, due to the small dataset size, the results obtained with this method may be pessimistic. For hyperparameter tuning *nested cross-validation* can be used, where cross-validation is again used on the training set, so we have "outer" test folds and for each, we use subsequent "inner" validation folds for hyperparameter tuning. For inner folds, there are 2 important possibilities: LOOCV and a high number of folds. LOOCV is unbiased and leaves as much data for training as possible, but results in a very high number of evaluations, which may be problematic even for small datasets if the hyperparameter grid is large. This may be the case for classifiers sensitive to regularization, e.g. SVM. A high number of folds, on the other hand, e.g. 10-fold or 15-fold cross-validation, reduces the estimation variance and greatly reduces the computational complexity.

3.5.3 Bootstrap Methods

In [15] the Out-Of-Bag (OOB) has been used. This method is an example of bootstrap-based methods. Those methods are especially useful for small datasets, where resampling alleviates some of the problems with the small sample number. The classic OOB method generates n bootstrap samples (usually 100–200 [26]); each sample has the same size as the dataset and elements are drawn with replacement. Due to the latter, usually only about 63.2 % points in each sample are unique [25]. Classifier instances are trained on those samples, while the test results are calculated separately for each sample, predicting it using classifiers that did not have it in the test set.

A more sophisticated approach is the 0.632 estimator. There the classifier instances are trained bootstrap samples and the original training set is used as the test set; this A_t estimate is highly optimistic [25, 26]. Another approach is to check the accuracy for each point in the dataset on the bootstrap samples where it was

not included; this A_l estimate is pessimistic. Those two can be averaged, resulting in the 0.632 estimator:

$$A_{0.632} = 0.632 \cdot A_l + 0.368 \cdot A_t.$$

3.5.4 Used Methods and Recommendations

For our experiments, we choose nested cross-validation with stratified 5-fold CV for testing (outer loop), and LOOCV for hyperparameter tuning and validation (inner loop). We optimize hyperparameters for the highest accuracy.

As suggested in [1, 2] we calculate multiple metrics: balanced accuracy [33] (instead of regular accuracy due to class imbalance), precision, recall, specificity, Matthews correlation coefficient, F1 score, and ROC AUC.

We do not use the holdout method because of the arguments in Section 3.5.1.

The nested cross-validation approach has multiple advantages:

- simplicity – especially important in conjunction with the automated feature engineering;
- ability to calculate test mean and standard deviation – provides a measure of performance bounds and classifier's robustness;
- high control over bias and variance of test and validation estimates – number of folds for inner and/or outer CV can be changed for different sized datasets, providing a high degree of control over results;
- it is easy to apply automated feature engineering – it can easily be applied for current training data only.

We refrain from using bootstrap methods since they are problematic to use with automated feature engineering. It is unclear how features should be generated in this case; we present two possible implementations, each of which has some critical drawbacks.

Firstly, we could use the whole dataset to calculate features before generating bootstrap samples. This means that, for example, statistical tests in tsfresh use all samples, therefore have higher statistical power, since all available samples from the dataset are used. This can result in rejecting some features based on this knowledge. However, in bootstrap the same samples are later used as test samples for some bootstrap samples, therefore resulting in data leakage. This is unacceptable since it makes results overly optimistic.

On the other hand, features can be calculated for each bootstrap sample separately. On average, bootstrap samples contain 36.8 % of duplicate samples, which will make automated feature engineering libraries not work correctly, since they assume points uniqueness. For example, in tsfresh, it is assumed that the number of samples n, used for statistical tests, is equal to the dataset size. The higher the n, the higher the statistical power, which has a considerable impact on feature selection. However, since the real number of points is much lower, the results

are wrong. Moreover, the correlations of features would be overall high, since calculation from the same points would result in the same values, which would lead to uninformative features. If the points are made unique before calculating features, on the other hand, the method is no longer bootstrap, since this would be equal to drawing without replacement and with n lower than the original dataset size.

3.6 Performance Evaluation in Related Works

Direct comparison of our results with those of most other papers using Depresjon or Psykose datasets is not possible because our evaluation procedure described in Section 3.5.4 differs from them. In addition, as we describe above, contrary to most other works, we report direct classification scores based on the classification of individual subjects, while the majority of papers classify extracted subsets of data. We argue that our approach is the most sound from the methodological point of view.

The overall goal of the classification of time series in mental health is the diagnosis of an individual patient. This means that in the end, we expect the label healthy/sick to be assigned to each sample that we perform the prediction for. Therefore, we are also ultimately interested in performance metrics calculated from those final predictions per patient. Because of the very small number of samples, this kind of evaluation may be very harsh, especially if a demanding testing procedure with multiple test sets, such as cross-validation, is used. Of course, this does not disqualify approaches where another, larger intermediate dataset is generated to use algorithms requiring a larger amount of data for training. For example, if the original time series is cut into 24-hour parts and each part is classified as either depressed or healthy, then also another method is required to obtain a final, single prediction per patient.

In the original Depresjon paper [1] the baselines are provided only for the task of classifying individual days, i.e. depressed vs non-depressed. No method is presented for individual patient prediction. The authors of [13] propose a classification of individual hours, divided into three subsets: full 24 hours data, only night hours, and only day hours. Metrics are only calculated for each subset, but no method is presented for extracting a final prediction for the patient. In the case of one work [15] the description was so unclear that we could not understand the exact methodology, but based on the number of samples mentioned (about 4 thousand) we assume that authors classified individual hours.

Work [14] performs daily classification, but ultimately outputs the final prediction per subject using majority voting based on individual days' predictions. It uses LOOCV, so it is similar to our approach and we can compare results. However, only mean values of the metrics are reported, without standard deviation.

In the Psykose dataset paper [2], the baselines are also provided only for the daily classification, with no method for individual subject prediction.

4 EXPERIMENTS

4.1 Datasets

We used Depresjon [1] and Psykose [2] datasets. Both datasets have been gathered in the same way: actigraph watches were handed out to patients with the condition and healthy controls, and measurements were gathered continuously for multiple days.

Both datasets are imbalanced in terms of class distribution: Depresjon has 23 conditions and 32 controls, while Psykose has 22 conditions and 32 controls. Length of measurement varies between subjects: between 5 and 20 days for Depresjon and between 8 and 20 days for Psykose. However, for both datasets, the majority of subjects have measurements varying between 12 and 14 days: 39 out of 55 for Depresjon and 46 out of 54 for Psykose. The distribution of lengths of measurement is similar in the condition and control groups. In both datasets there are missing values, but they constitute a very small percentage of data, typically at most 1 hour per patient during the whole measurement duration.

4.2 Data and Code Availability

Datasets can be downloaded from:

- Depresjon: `https://datasets.simula.no/depresjon/`,
- Psykose: from `https://datasets.simula.no/psykose/`.

Our code for the reproduction of the results has been made publicly available at `https://github.com/j-adamczyk/Mental_disorder_TS_feature_engineering`.

4.3 Experimental Setup

All experiments have been conducted for both datasets in the same way. The first step is feature extraction, performed for each time series separately. We first fill missing values with the mean value of a given subject's activity. Then, for manual feature extraction only, we resample the data with a 1-hour frequency. After this, we create 3 subsets from each time series: full 24 h data (whole time series), night data (from 21:00 to 8:00), and day data (from 8:00 to 21:00). Then we extract features and merge the data, gathering rows into tables ready for classification. We used the following feature extraction settings: manual (features described in Table 1), automated with "minimal" tsfresh settings and automated with "efficient" tsfresh settings. Therefore in the end we have 9 tables for each dataset (3 methods, 3 parts for each).

The next step is the classification itself. As mentioned in Section 3.5.4, we use a nested CV, with a 5-fold CV for testing and LOOCV for hyperparameter optimization and model selection. In the outer loop, we calculate the metrics on

test sets, reporting the mean and standard deviation after all 5 folds. In the inner loop, the procedure depends on the experiment.

For manual feature extraction and for automated feature extraction with "minimal" settings we only perform variance thresholding and standardization before optimizing hyperparameters with grid search. For automated feature extraction with "efficient" settings during initial tests we ran into problems with tsfresh's feature selection algorithm. The Benjamini–Yekutieli procedure with FDR set to default 0.05 often results in rejecting all features since the training sets are very small here. Based on tsfresh documentation, we consider two options for dealing with this problem:

- increasing FDR – after variance thresholding, but before standardization we use tsfresh's feature selection based on Benjamini-Yekutieli procedure with increasing FDR, starting with 0.05 and increasing by 0.05 until at least 1 feature "relevant" (according to this procedure) is selected;
- selecting the top N features – instead of using only "relevant" feature we take the table with p-values and take N features with the lowest p-values, i.e. the most important according to the statistical tests.

For the latter, we consider $N = 5$ and $N = 10$ since the initial experiments indicated that those values give good and stable (in terms of standard deviation) results, better than with higher N. This feature selection is performed after variance thresholding but before standardization.

4.4 Results

Results for Depresjon and Psykose are given in the tables below. We report each subset (full 24 h, night, day) for each dataset in a separate table. In each table, for each metric, the best score is marked with bold text. We define the best metric score as the highest mean test value, and in the case of equal means, we choose the one with a lower standard deviation.

We performed 90 experiments in total (2 datasets, 3 subsets each, 5 feature engineering variants, 3 classifiers). Therefore, due to space constraints, we report only the best classifier for each feature variant. They were selected based on both the highest metrics and lowest standard deviation. Typically, there was no ambiguity since one classifier outperformed the other two by 5 % or 10 % on all metrics.

In tables, the first column denotes the feature engineering method: "M" (manual), "AM" (automated, "minimal" settings), "AE" (automated, "efficient" settings). For "AE" we also mark the variant: FDR (increasing FDR), $N = 5$ (top N with $N = 5$), $N = 10$. In the third column, there is information about the type of chosen classifier. Metrics are: balanced accuracy (B. acc.), F1 score (F1), precision, recall, specificity (spec.), ROC AUC (AUC), and Matthews Correlation Coefficient (MCC). All metrics except for MCC range from 0 to 1, whereas MCC ranges from -1 to 1. All numbers have been rounded to 2 decimal places. We denote the standard deviation by the number after the $\pm$ sign.

Method	Clf.	B. Acc.	F1	Precision	Recall	Spec.	AUC	MCC
M	LR	$\mathbf{0.76 \pm 0.16}$	$\mathbf{0.73 \pm 0.18}$	0.73 ± 0.21	$\mathbf{0.77 \pm 0.16}$	0.76 ± 0.25	$\mathbf{0.77 \pm 0.16}$	$\mathbf{0.54 \pm 0.32}$
AM	RF	0.73 ± 0.06	0.68 ± 0.07	0.83 ± 0.15	0.60 ± 0.11	0.87 ± 0.12	0.73 ± 0.06	0.52 ± 0.14
AE, FDR	SVM	0.74 ± 0.14	0.67 ± 0.18	0.77 ± 0.20	0.60 ± 0.17	0.88 ± 0.12	0.74 ± 0.14	0.50 ± 0.29
AE, $N = 5$	SVM	0.68 ± 0.07	0.56 ± 0.12	$\mathbf{0.85 \pm 0.20}$	0.43 ± 0.12	$\mathbf{0.94 \pm 0.08}$	0.68 ± 0.07	0.45 ± 0.17
AE, $N = 10$	RF	0.71 ± 0.09	0.61 ± 0.16	0.81 ± 0.19	0.51 ± 0.19	0.90 ± 0.08	0.71 ± 0.09	0.47 ± 0.18

Table 2. Depresjon, full 24 h data

Method	Clf.	B. Acc.	F1	Precision	Recall	Spec.	AUC	MCC
M	LR	0.67 ± 0.13	0.52 ± 0.28	0.78 ± 0.22	0.49 ± 0.26	0.85 ± 0.16	0.67 ± 0.13	0.36 ± 0.28
AM	RF	$\mathbf{0.74 \pm 0.07}$	$\mathbf{0.69 \pm 0.08}$	0.81 ± 0.17	$\mathbf{0.64 \pm 0.14}$	0.84 ± 0.18	$\mathbf{0.74 \pm 0.07}$	0.52 ± 0.15
AE, FDR	LR	0.68 ± 0.14	0.59 ± 0.22	0.67 ± 0.22	0.62 ± 0.34	0.74 ± 0.13	0.68 ± 0.14	$\mathbf{0.68 \pm 0.14}$
AE, $N = 5$	SVM	0.68 ± 0.07	0.57 ± 0.12	$\mathbf{0.85 \pm 0.20}$	0.43 ± 0.12	$\mathbf{0.94 \pm 0.08}$	0.68 ± 0.07	0.45 ± 0.17
AE, $N = 10$	SVM	0.68 ± 0.08	0.57 ± 0.17	0.83 ± 0.15	0.50 ± 0.25	0.87 ± 0.12	0.68 ± 0.08	0.44 ± 0.12

Table 3. Depresjon, night data

4.5 Discussion

Classifiers that achieved the highest mean metrics results on subsets of datasets are:

- Depresjon:
  - full 24 h data: manual, LR;
  - night data: automated "minimal", RF;
  - day data: automated "efficient", top $N = 5$, RF; the best results.
- Psykose:
  - full 24 h data: automated "efficient", top $N = 10$, LR;
  - night data: automated "efficient", FDR, LR; the best results;
  - day data: automated "efficient", top $N = 5$, RF.

In almost all cases, across almost all metrics, the automated approach outperforms the manual, often by a considerable margin. The only exception is Depresjon full 24 h data, where the manual approach has the highest mean values of the metrics. However, it has very high standard deviations, over 0.16 for all metrics, which makes it highly unreliable. On the other hand, the automated method with "minimal" settings often achieved only slightly lower results, while getting much lower standard deviation, e.g. for balanced accuracy 0.73 ± 0.06 compared to 0.76 ± 0.16, or for AUC 0.73 ± 0.06 compared to 0.77 ± 0.16.

Method	Clf.	B. Acc.	F1	Precision	Recall	Spec.	AUC	MCC
M	LR	0.73 ± 0.13	0.58 ± 0.31	$\mathbf{1.00 \pm 0.00}$	0.46 ± 0.27	$\mathbf{1.00 \pm 0.00}$	0.73 ± 0.13	0.53 ± 0.28
AM	RF	0.72 ± 0.03	0.65 ± 0.07	0.80 ± 0.16	0.60 ± 0.17	0.84 ± 0.15	0.72 ± 0.03	0.49 ± 0.08
AE, FDR	RF	0.74 ± 0.05	0.70 ± 0.04	0.85 ± 0.19	0.64 ± 0.14	0.83 ± 0.21	0.74 ± 0.05	0.54 ± 0.13
AE, $N = 5$	RF	$\mathbf{0.81 \pm 0.06}$	$\mathbf{0.78 \pm 0.10}$	0.87 ± 0.11	$\mathbf{0.76 \pm 0.22}$	0.87 ± 0.12	$\mathbf{0.81 \pm 0.06}$	$\mathbf{0.68 \pm 0.08}$
AE, $N = 10$	RF	0.77 ± 0.03	0.72 ± 0.05	0.89 ± 0.14	0.64 ± 0.14	0.90 ± 0.13	0.77 ± 0.03	0.60 ± 0.07

Table 4. Depresjon, day data

Method	**Clf.**	**B. Acc.**	**F1**	**Precision**	**Recall**	**Spec.**	**AUC**	**MCC**
M	RF	0.80 ± 0.09	0.76 ± 0.11	$\mathbf{0.86 \pm 0.20}$	0.72 ± 0.11	0.88 ± 0.17	0.80 ± 0.09	0.64 ± 0.19
AM	SVM	0.80 ± 0.02	0.75 ± 0.05	0.82 ± 0.09	0.72 ± 0.11	0.88 ± 0.06	0.80 ± 0.02	0.62 ± 0.01
AE, FDR	SVM	0.83 ± 0.07	0.79 ± 0.08	0.83 ± 0.14	0.78 ± 0.13	0.88 ± 0.11	0.83 ± 0.07	0.68 ± 0.14
AE, $N = 5$	SVM	0.81 ± 0.08	0.77 ± 0.10	0.85 ± 0.20	0.73 ± 0.07	$\mathbf{0.89 \pm 0.17}$	0.81 ± 0.08	0.64 ± 0.19
AE, $N = 10$	LR	$\mathbf{0.89 \pm 0.06}$	$\mathbf{0.86 \pm 0.08}$	0.83 ± 0.15	$\mathbf{0.92 \pm 0.10}$	0.85 ± 0.13	$\mathbf{0.89 \pm 0.06}$	$\mathbf{0.77 \pm 0.13}$

Table 5. Psykose, full 24 h data

Method	**Clf.**	**B. Acc.**	**F1**	**Precision**	**Recall**	**Spec.**	**AUC**	**MCC**
M	RF	0.91 ± 0.12	0.88 ± 0.15	$\mathbf{0.92 \pm 0.16}$	0.87 ± 0.17	0.94 ± 0.11	0.91 ± 0.12	0.82 ± 0.23
AM	SVM	0.91 ± 0.06	0.89 ± 0.08	0.89 ± 0.09	0.92 ± 0.16	0.91 ± 0.07	0.91 ± 0.06	0.83 ± 0.10
AE, FDR	LR	$\mathbf{0.93 \pm 0.03}$	$\mathbf{0.92 \pm 0.04}$	0.89 ± 0.09	$\mathbf{0.96 \pm 0.08}$	0.91 ± 0.07	$\mathbf{0.93 \pm 0.03}$	$\mathbf{0.86 \pm 0.07}$
AE, $N = 5$	SVM	0.89 ± 0.07	0.86 ± 0.09	0.89 ± 0.14	0.87 ± 0.17	0.91 ± 0.11	0.89 ± 0.07	0.80 ± 0.12
AE, $N = 10$	SVM	0.88 ± 0.08	0.85 ± 0.09	0.91 ± 0.11	0.82 ± 0.16	$\mathbf{0.94 \pm 0.07}$	0.88 ± 0.08	0.78 ± 0.14

Table 6. Psykose, night data

The lower standard deviation for the automated approach is a clearly visible pattern across all experiments. Sometimes differences are truly large, for example for Depresjon day data, F1 metric, the AE top $N = 5$ method achieved 0.78 ± 0.10, while the manual approach got 0.58 ± 0.31. In practice, using classifiers with a standard deviation higher than 0.1–0.15 carries considerable risk, as they are highly unreliable. The appearance of such high standard deviations in our results also signifies the importance of using nested CV for evaluation.

Results vary considerably between full, night, and day data for both datasets. For example, for best classifiers using Depresjon full 24 h and day data, the results are 5–10 % higher for day data. This makes sense from a psychological point of view, as either circadian, nocturnal, or daily activity patterns may be the most significant. This also means that segmentation into different parts of the day is an important direction for feature extraction from actigraphy data. Such information is relevant not only from a classification point of view but also for purely psychological analysis of behavioral patterns of mental disorders.

For both datasets, the best results were achieved using automated feature engineering. All 4 variants of automated feature engineering are among the best classifiers. This means that in future experiments with this method various scenarios have to be considered. For different datasets and different subsets of data, the most relevant features may differ considerably. While automated methods can and will extract and select those features, the optimal selection method is not obvious.

Interestingly only the LR and RF classifiers achieved the best results. However, the SVM got comparable results, with only a slightly lower mean, and often with very

Method	**Clf.**	**B. Acc.**	**F1**	**Precision**	**Recall**	**Spec.**	**AUC**	**MCC**
M	RF	0.81 ± 0.08	0.78 ± 0.10	0.82 ± 0.18	0.77 ± 0.02	0.85 ± 0.16	0.81 ± 0.08	0.64 ± 0.18
AM	SVM	0.80 ± 0.06	0.76 ± 0.07	0.83 ± 0.14	0.72 ± 0.11	0.88 ± 0.12	0.80 ± 0.06	0.62 ± 0.12
AE, FDR	LR	0.85 ± 0.03	0.83 ± 0.03	$\mathbf{0.92 \pm 0.10}$	0.77 ± 0.02	0.93 ± 0.08	0.85 ± 0.03	0.74 ± 0.09
AE, $N = 5$	RF	$\mathbf{0.89 \pm 0.07}$	$\mathbf{0.87 \pm 0.08}$	0.85 ± 0.12	$\mathbf{0.90 \pm 0.12}$	0.88 ± 0.12	$\mathbf{0.89 \pm 0.07}$	$\mathbf{0.79 \pm 0.13}$
AE, $N = 10$	RF	0.88 ± 0.07	0.86 ± 0.09	0.91 ± 0.11	0.82 ± 0.09	$\mathbf{0.94 \pm 0.08}$	0.88 ± 0.07	0.77 ± 0.14

Table 7. Psykose, day data

low standard deviation. Therefore, all three types should be considered for future experiments on this dataset. The good performance of LR, which is a relatively simple, linear classifier, shows that proper feature engineering allows even simple classification algorithms to achieve excellent results.

Overall results are better for the Psykose dataset, both in terms of higher mean metrics and lower standard deviations, and are in general very good. The best classifier for Psykose (night data, AE FDR, LR) clearly stands out, achieving great results. No metric mean is lower than 0.86, and most are over 0.90; at the same time, standard deviations are low, ranging between 0.03 and 0.09. This may mean that there are very specific behavioral patterns of schizophrenics during their nocturnal activities. However, the usage of such patterns requires deeper analysis of the features used – see Section 4.7 for discussion.

The fact that the best results are achieved with smaller subsets of data is very beneficial from a practical point of view. This means that future subjects may not need to wear the actigraph all the time, just during a part of a day. This makes a diagnosis using this method less cumbersome and data easier to collect.

4.6 Comparison with Other Works

Recalling the results of analysis from Section 3.6, we were able to find only one other paper with results provided per patient [14]. We use a weighted average from paper for each metric, using the SMOTE variant for each classifier (as it was described as the best variant). We also compare accuracy from the paper and balanced accuracy from our results. It should be kept in mind that balanced accuracy is a harsher, but more realistic metric for imbalanced datasets such as Depresjon.

For comparison with our results, we choose the classifiers that achieved the highest mean metrics results on subsets of Depresjon: manual LR (full 24 h data), automated "minimal" RF (night data), automated "efficient" top $N = 5$ RF (day data). Results are gathered in Table 8. The best value for each metric is marked with bold font.

Method	Acc./B. Acc.	F1	Precision	Recall	Spec.	MCC
Garcia-Ceja et al., RF	0.73	0.73	0.73	0.73	0.72	0.44
Garcia-Ceja et al., DNN	0.69	0.69	0.69	0.69	0.65	0.35
Adamczyk & Malawski, M, LR	0.76	0.73	0.85	**0.77**	**0.94**	0.54
Adamczyk & Malawski, AM, RF	0.74	0.69	0.81	0.64	0.84	0.52
Adamczyk & Malawski, AE, $N = 5$, RF	**0.81**	**0.78**	**0.87**	0.76	0.87	**0.68**

Table 8. Comparison of results with [14]

Our approaches gave superior results in all cases. In particular, automated feature engineering with "efficient" settings, using top $N = 5$ and RF have better results for all metrics than those achieved by either RF or DNN from [14]. The version using "minimal" settings and RF also gave results comparable to RF and DNN in that paper. This proves the usefulness of automated feature engineering. The

automated approach is also simpler, requiring no additional intermediate datasets or models.

4.7 Analysis of Extracted Features

Features that are extracted and selected using automated feature engineering do not need to be analyzed separately, they can be just used directly. Results presented in Section 4.4 have been obtained that way. However, we can also use the automated approach to gain additional insights into useful features and include only some of them, or just look into what domains provide the best discriminative qualities.

For such analysis, we count how many times different features have been used for training classifiers using different automated approaches. We count how many times different features have been chosen across all train/test folds (outer 5-fold CV loop). Feature sets used for training classifiers are already after both variance thresholding and feature selection. The only exception is the method using "minimal" settings, where no additional feature selection is used, only variance thresholding. We review all 4 variants of automated feature selection: AM, AE FDR, AE top $N = 5$, AE top $N = 10$. Because of the space constraints, we select only a few example histograms for Depresjon, full 24 h data in Figures 1, 2, 3 and 4. A full collection of plots can be found in the appropriate notebook in the Github repository.

Below we explain the meaning of the less obvious features. The more detailed explanations can be found in tsfresh documentation [34].

1. linear trend intercept – fits the linear regression to the raw values of time series and returns the intercept value;
2. aggregated linear trend intercept, chunk of length N, aggregation function f – similar to the above, but before regression it divides the time series into chunks of size N and aggregates them with function f (min, mean or max), fitting the linear regression to such preprocessed series;
3. count above mean – number of values above mean value of time series;
4. absolute energy – absolute energy of the signal, which is defined as a sum of its squared values;
5. CID – an estimate of the complexity of the time series, based on either raw (unnormalized) or normalized values [35], defined as:

$$CID(x) = \sqrt{\sum_{i=1}^{n-1}(x_i - x_{i-1})^2};$$

6. Lempel-Ziv complexity, N bins – an estimate of the complexity of the time series. The complexity is defined as the number of dictionary entries (or subwords) needed to encode the time series when viewed from left to right. For this, the time series is first binned into N bins. Then Lempel-Ziv compression is used:

signal is converted into sub-words with different prefixes, and the number of sub-words needed for this divided by the length of the time series is the complexity estimate [34];

7. FFT absolute coefficient N – absolute value of coefficient N of 1D Fourier transform;
8. Spectral Welch density coefficient N – coefficient N of the power spectral cross density estimated using Welch's method.

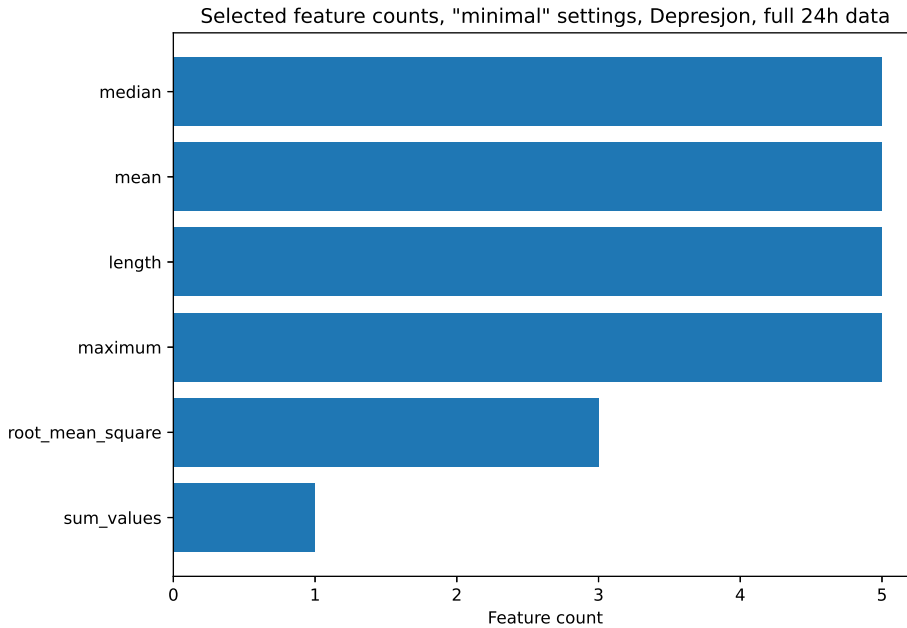

Figure 1. Feature counts, Depresjon, "minimal" settings, full 24 h data

Across all subsets of both datasets, the "minimal" settings features after variance thresholding were: median, mean, length, maximum. Additionally, root mean square was used very often. Since "minimal" settings do not have many features, this is not particularly surprising. Also, many of those features were selected by us in the manual feature engineering and they have quite obvious psychological interpretations. Interestingly the length of the time series was often used. We checked the length distribution for both classes and there are no clear differences, so while this feature can be considered "fair" for usage, it does not have the immediate psychological interpretation. One possible explanation could be that subjects suffering from mental disorders are less disciplined and less inclined to wear the actigraph for more extended periods of time.

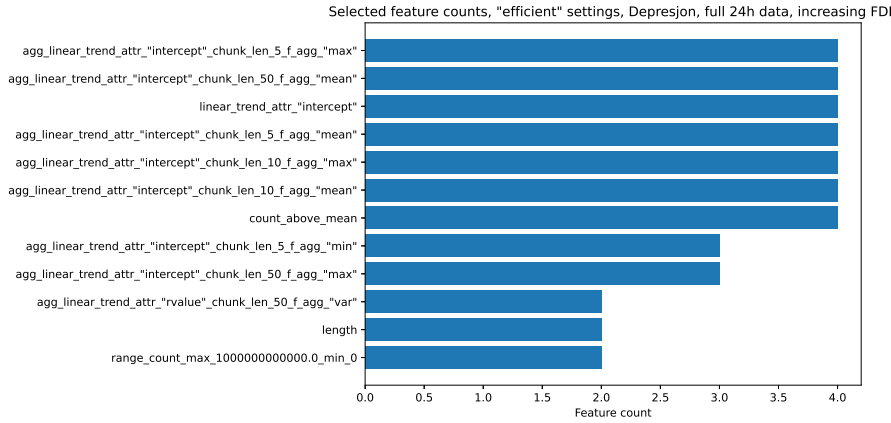

Figure 2. Feature counts, Depresjon, "efficient" settings with increasing FDR, full 24 h data

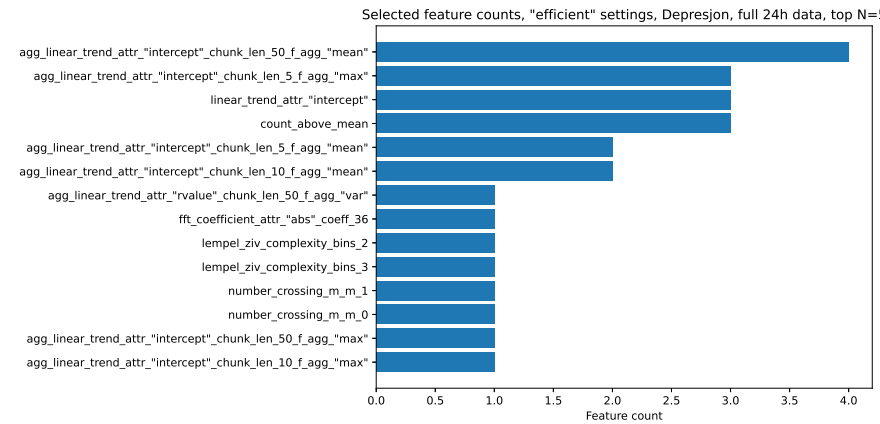

Figure 3. Feature counts, Depresjon, "efficient" settings with top $N = 5$, full 24 h data

For each subset of datasets, the most commonly selected features across three "efficient" variants have been gathered in Table 9. The features selected using automated feature engineering are very different from those used by us in the manual approach. The statistical features were not selected (tsfresh does calculate them and could use them). Instead, we can roughly divide those features into two groups: describing the complexity of the time series and the overall shape.

The complexity-related features include Lempel-Ziv complexity, CID (unnormalized), and spectral Welch density coefficient. The Lempel-Ziv complexity has been selected only for Depresjon and the spectral Welch density coefficient only for Psykose, while CID was used in subsets from both datasets. This means that while

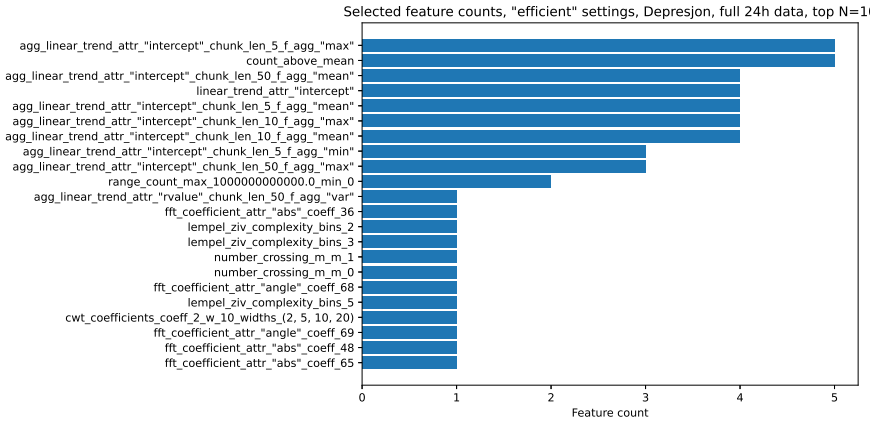

Figure 4. Feature counts, Depresjon, "efficient' settings with top $N = 10$, full 24 h data

the complexity measures are very useful, the exact one to use is not obvious and depends on the particular dataset.

Features concerning the overall shape of the time series are related mostly to fitting the linear regression: on raw data or on aggregated data (often using 2 or 3 aggregation measures). Other features in this group include autocorrelation, partial autocorrelation, and absolute energy. Linear regression intercept has been selected

Dataset	Signal Complexity Features	Signal Shape Features
Depresjon full 24 h	–	aggregated linear trend intercept: len 5 mean and max, len 10 mean and max, len 50 mean; linear trend intercept; count above mean
Depresjon night	absolute energy; CID without normalization; Lempel-Ziv complexity with 2 bins	–
Depresjon day	Lempel-Ziv complexity with 2 and 3 Bins	aggregated linear trend intercept: len 5 mean and max, len 10 mean and max; linear trend intercept
Psykose full 24 h	absolute energy	aggregated linear trend intercept: len 5 min, mean and max, len 10 mean and max, len 50 mean; linear trend intercept
Psykose night	absolute energy; spectral Welch density coefficient 2	autocorrelation with lag 1; partial autocorrelation with lag 1
Psykose day	FFT absolute coefficient 23; spectral Welch density coefficient 2; CID without normalization	aggregated linear trend intercept: len 5 min, mean and max, len 10 min, mean and max, len 50 mean; linear trend intercept

Table 9. Features most commonly extracted with automated feature engineering

overall the most often of all features, and often many types of aggregation have been used before fitting the regression line: min, mean, and max. This means that analysis of the overall tendency in the subject's activity is a well-discriminating feature. The absolute energy aggregates the overall sum of activity, but since it squares the values before summation, it puts more weight on activity peaks. We expect both depressed or schizophrenic patients to have overall lower activity and especially lower activity peaks, so this feature incorporates this knowledge into the model. Autocorrelation and partial autocorrelation, both with lag 1, have been chosen for Psykose night data only. This may extract information about short period abnormalities in the subject's activity, which may relate to psychotic episodes in patients suffering from schizophrenia.

We can conclude that the features extracted automatically are all nonobvious and advanced. For manual extraction, they would require very specific knowledge from statistics, time series analysis, or signal processing, even to know about their existence in some cases (e.g. CID, which is less known than others). Many features were chosen reliably, often in all cases for a given subset or dataset, further implying their importance. Applying automated methods for such feature discovery is a promising way for psychologists or psychiatrists to enrich their analysis. Further work in this area, especially domain interpretation of extracted features, would require the help of medical professionals.

4.8 Software Engineering Aspects

The automated feature engineering turned out to be an easier to implement solution than manual feature engineering. It required less work and domain knowledge. It also resulted in a shorter, cleaner, easier-to-understand and maintain code. Those qualities can be measured qualitatively and quantitatively.

In terms of qualitative analysis, the manual approach required more knowledge and work. Firstly, it required to gain domain knowledge in actigraphy, psychology, and signal processing, so we understood the problem and what features can be extracted, and which ones may be the most useful for this particular problem. Research into practical implementation and writing code for extracting those features also took a considerable amount of time. The manual approach also requires more temporary code for researching various libraries, their APIs, and testing our implementation. While getting comparable results with both the manual and automated approaches is definitely possible, the latter requires considerably less work to achieve them.

Qualitatively we can use code complexity measures to compare manual and automated approaches. We choose three simple code complexity measures: physical Lines Of Code (LOC), LOC including comments, number of functions and classes, and number of function calls.

The Source Lines Of Code (SLOC) measure is one of the most commonly used code complexity measures [36]. However, its definition is not consistent, and there are many types of SLOC measures. Two major ones are physical SLOC (LOC) and

logical SLOC (LLOC). We use the LOC measure, as its definition is less ambiguous [36].

Precise definitions of this measure may vary, however. We make the following assumptions about the code being compared:

- code uses the Python language with PEP8 code formatting standard [37]; in particular, no lines are longer than 80 characters;
- we count only feature engineering related directly to the particular method, not the code common for all methods;
- we include imports from external libraries;
- each function call uses 1 line unless it needs to be broken into multiple lines due to 80 characters length constraint;
- function calls for library functions with multiple arguments may use keyword arguments for readability;
- we do not count empty lines.

We also commented our code in multiple places for readability, understanding complex library calls, and features being calculated. It was necessary for the maintainability of the created ML pipelines. Therefore, we also provide LOC including comments, as this measure also takes into account the logical complexity of code and the need for additional text for human understanding. This is a measure that falls somewhere between LOC and LLOC measures. For docstrings documenting functions and classes, we also follow PEP8, and we only count lines with actual comments (not the lines with triple quotation marks marking the start and the end of the multiline comment).

The number of functions and classes serves as a metric for code maintainability burden. The more objects such as those, the more burden the code maintenance is, especially in larger codebases. We only count classes and not their methods, since all implemented classes use the standard Scikit-learn interface of fit/transform [32] and in practice are used like transforming functions.

We also count the number of function calls required for the feature engineering process. This measure takes into account external library calls, which increase code complexity because code maintainers have to monitor API changes in those libraries. They also typically require external documentation lookups during development, constituting an additional development burden.

The exact lines of code used for comparison can be found in other files in the Github repository linked in Section 4.2. Results are presented in Table 10, with feature engineering methods marked similarly to the tables from Section 4.4, with the lowest (best) metrics marked with bold font.

The automated feature engineering did not meet the advertised "one-line feature engineering". However, the code complexity across all metrics was significantly lower for both automated approaches. The "minimal" settings provided the shortest, least complex code.

Approach	LOC	LOC Incl. Comments	Functions and Classes	Function Calls
M	68	90	6	35
AM	**25**	**12**	**2**	**12**
AE, FDR	47	61	3	21
AE, top N	48	61	3	26

Table 10. Code complexity measures

LOC and LOC include comments for automated approach with "efficient" settings are longer, since they also require feature selection methods, either increasing FDR or selecting top N features. For "minimal" settings, however, there is no such need and therefore both of those metrics are significantly lower than for all other approaches, especially compared to the manual one. Additionally, while the LOC is about 3 times lower compared to the manual feature engineering, the LOC including comments is about 7.5 times lower. The fact that the number of comments is this much lower signifies the simplicity of the code using tsfresh – almost no comments are required since the code boils down to simple reshaping of the input data and calling the feature extraction function.

The manual approach also requires 2-3 times as many functions as the automated one. It should be also noted that the number of custom functions grows linearly in manual feature engineering, as more domain-specific features (not implemented in libraries) need to be extracted. For the automated approach, the number remains constant.

The number of function calls also takes into account individual library calls. For manually extracting features this number is high because extracting each feature requires calling either custom or external library functions and would also linearly grow with the number of features. For automated methods with feature selection, this number is much lower, but still a bit high, since this step requires additional code. For the "minimal' settings, however, the code is really short, consisting of only feature extraction, and the overall number of function calls is about 3 times lower than for the manual approach.

We can therefore conclude that the automated approach has considerable software engineering advantages since it results in shorter, cleaner, easier-to-understand code. This is especially important for psychologists or psychiatrists without high programming expertise. It would also be much less burdensome to maintain in larger codebases and practical deployments, making this an important advantage for practical applications.

4.9 Code Example

For completeness sake, we include the code listing performing automated feature extraction with tsfresh using "minimal" settings. Code for other approaches can be found in the appropriate notebook in the Github repository.

```
import pandas as pd
from tsfresh.feature_extraction.settings \
  import MinimalFCParameters
from tsfresh.transformers import FeatureAugmenter

# ts - pd.DataFrame with time series, tsfresh "flat" format
# X  - pd.DataFrame with extracted features

ids = ts["id"].unique()
X = pd.DataFrame(index=ids)

augmenter = FeatureAugmenter(
  default_fc_parameters=MinimalFCParameters,
  column_id="id",
  column_sort="timestamp",
  column_value="activity",
  chunksize=1, n_jobs=4
)

augmenter.set_timeseries_container(ts)

# extract all features defined in default_fc_parameters
X = augmenter.transform(X)
```

5 SUMMARY

The main objective of this paper was to compare manual and automated feature engineering for the classification of time series in mental disorder diagnosis. For this purpose, we have researched commonly used features and their domain interpretation, and the tsfresh library for automated feature extraction and selection, with different possible approaches to using it. We also performed an analysis of performance evaluation methods, suggesting the best methods for comparing classifiers on small datasets typical for the mental health domain.

The automated approach proved to give very good results on both Depresjon and Psykose datasets. The results were always comparable to the manual methods, but often the metrics were higher or lower standard deviation, meaning a more robust classification. We also compared qualitative metrics of code complexity, showing that using automated feature engineering leads to shorter, cleaner, easier-to-understand and maintain code compared to the manual methods.

We conclude that it is possible to reliably diagnose mental disorders such as depression and schizophrenia based on features extracted from actigraphy signals. For the best classifiers, multiple reported metrics are about 0.8 for Depresjon and 0.9 for Psykose, while having a reasonable standard deviation, about 0.05–0.1. These

results are all achieved using good, highly discriminative features, extracted in an automated way.

There is a lot of possible future work in this domain. The results on both datasets are very good, yet they may be improved. For example, we could conduct further analysis of features discovered with automated methods or consult their interpretation with medical professionals. This may potentially lead to discovering reliable patterns in the activity of depressed or schizophrenic patients, which are visible only using those advanced features. Also, the classification using only those selected features could be conducted to analyze whether it would give better or more reliable results than a purely automated approach.

Our results also signify the importance of feature selection; the methods used here are relatively simple. Instead, more sophisticated methods such as genetic algorithms or dedicated feature selection methods such as Boruta algorithm [38] can be used to select relevant features after extraction with tsfresh. Another path of research is the usage of classifiers working directly on time series, such as kNN with Dynamic Time Warping (DTW) [39] or Time Series Forest [40]. Also, we can change our approach and classify individual days, deploying different strategies for the final classification of the subject, such as majority voting or Borda voting; this way we could also use models requiring more data for training, such as recurrent neural networks (RNNs, especially GRU or bidirectional variants) or neural networks with an attention mechanism.

Therefore, we can conclude that the machine learning methods presented in this work can be used to develop supportive diagnosis tools in psychology or psychiatry, leading to an affordable, non-intrusive way of early diagnosis of mental disorders. Automated feature engineering also makes the software engineering process easier for non-technical professionals like psychologists or psychiatrists.

Acknowledgements

The research presented in this paper has been supported by the funds of the Polish Ministry of Science and Higher Education assigned to AGH University of Science and Technology.

REFERENCES

[1] Garcia-Ceja, E.—Riegler, M.—Jakobsen, P.—Tørresen, J.—Nordgreen, T.—Oedegaard, K. J.—Fasmer, O. B.: Depresjon: A Motor Activity Database of Depression Episodes in Unipolar and Bipolar Patients. Proceedings of the 9th ACM Multimedia Systems Conference (MMSys '18), 2018, pp. 472–477, doi: 10.1145/3204949.3208125.

[2] Jakobsen, P.—Garcia-Ceja, E.—Stabell, L. A.—Oedegaard, K. J.—Berle, J. O.—Thambawita, V.—Hicks, S. A.—Halvorsen, P.—Fasmer, O. B.—Riegler, M. A.: PSYKOSE: A Motor Activity Database

of Patients with Schizophrenia. 2020 IEEE 33rd International Symposium on Computer-Based Medical Systems (CBMS), IEEE, 2020, pp. 303–308, doi: 10.1109/CBMS49503.2020.00064.

[3] World Health Organization: Mental Disorders Facts. Fact Sheets, 2019, available at: https://www.who.int/en/news-room/fact-sheets/detail/mental-disorders.

[4] World Health Organization: Depression and Other Common Mental Disorders: Global Health Estimates. 2017, available at: https://www.who.int/publications/i/item/depression-global-health-estimates.

[5] Hor, K.—Taylor, M.: Suicide and Schizophrenia: A Systematic Review of Rates and Risk Factors. Journal of Psychopharmacology, Vol. 24, 2010, No. 4, pp. 81–90, doi: 10.1177/1359786810385490.

[6] Minaeva, O.—Booij, S. H.—Lamers, F.—Antypa, N.—Schoevers, R. A.—Wichers, M.—Riese, H.: Level and Timing of Physical Activity During Normal Daily Life in Depressed and Non-Depressed Individuals. Translational Psychiatry, Vol. 10, 2020, Art. No. 259, doi: 10.1038/s41398-020-00952-w.

[7] Difrancesco, S.—Lamers, F.—Riese, H.—Merikangas, K. R.—Beekman, A. T. F.—van Hemert, A. M.—Schoevers, R. A.—Penninx, B. W. J. H.: Sleep, Circadian Rhythm, and Physical Activity Patterns in Depressive and Anxiety Disorders: A 2-Week Ambulatory Assessment Study. Depression and Anxiety, Vol. 36, 2019, No. 10, pp. 975–986, doi: 10.1002/da.22949.

[8] Fervaha, G.—Agid, O.—McDonald, K.—Foussias, G.—Remington, G.: Daily Activity Patterns in Remitted First-Episode Schizophrenia. Comprehensive Psychiatry, Vol. 5, 2014, No. 5, pp. 1182–1187, doi: 10.1016/j.comppsych.2014.04.001.

[9] Waters, F.—Sinclair, C.—Rock, D.—Jablensky, A.—Foster, R. G.—Wulff, K.: Daily Variations in Sleep-Wake Patterns and Severity of Psychopathology: A Pilot Study in Community-Dwelling Individuals with Chronic Schizophrenia. Psychiatry Research, Vol. 187, 2011, No. 1-2, pp. 304–306, doi: 10.1016/j.psychres.2011.01.006.

[10] Farahani, B.—Firouzi, F.—Chang, V.—Badaroglu, M.—Constant, N.—Mankodiya, K.: Towards Fog-Driven IoT eHealth: Promises and Challenges of IoT in Medicine and Healthcare. Future Generation Computer Systems, Vol. 78, 2018, Part 2, pp. 659–676, doi: 10.1016/j.future.2017.04.036.

[11] Scott, J.—Murray, G.—Henry, C.—Morken, G.—Scott, E.—Angst, J.—Merikangas, K. R.—Hickie, I. B.: Activation in Bipolar Disorders: A Systematic Review. JAMA Psychiatry, American Medical Association, Vol. 74, 2017, No. 2, pp. 189–196, doi: 10.1001/jamapsychiatry.2016.3459.

[12] Faedda, G. L.—Ohashi, K.—Hernandez, M.—McGreenery, C. E.—Grant, M. C.—Baroni, A.—Polcari, A.—Teicher, M. H.: Actigraph Measures Discriminate Pediatric Bipolar Disorder from Attention-Deficit/Hyperactivity Disorder and Typically Developing Controls. Journal of Child Psychology and Psychiatry, Vol. 57, 2016, No. 6, pp. 706–716, doi: 10.1111/jcpp.12520.

[13] Rodríguez-Ruiz, J. G.—Galván-Tejada, C. E.—Zanella-Calzada, L. A.—Celaya-Padilla, J. M.—Galván-Tejada, J. I.—Gamboa-Rosales, H.—Luna-García, H.—Magallanes-Quintanar, R.—Soto-Murillo, M. A.:

Comparison of Night, Day and 24 h Motor Activity Data for the Classification of Depressive Episodes. Diagnostics, Vol. 10, 2020, No. 3, Art. No. 162, doi: 10.3390/diagnostics10030162.

[14] Garcia-Ceja, E.—Riegler, M.—Jakobsen, P.—Torresen, J.—Nordgreen, T.—Oedegaard, K. J.—Fasmer, O. B.: Motor Activity Based Classification of Depression in Unipolar and Bipolar Patients. 2018 IEEE 31st International Symposium on Computer-Based Medical Systems (CBMS), 2018, pp. 316–321, doi: 10.1109/CBMS.2018.00062.

[15] Zanella Calzada, L. A.—Galván Tejada, C. E.—Chávez-Lamas, N. M.—del Carmen Gracia-Cortés, M.—Magallanes-Quintanar, R.—Celaya Padilla, J. M.—Galván-Tejada, J. I.—Gamboa-Rosales, H.: Feature Extraction in Motor Activity Signal: Towards a Depression Episodes Detection in Unipolar and Bipolar Patients. Diagnostics, Vol. 9, 2019, No. 1, Art. No. 8, doi: 10.3390/diagnostics9010008.

[16] Christ, M.—Braun, N.—Neuffer, J.—Kempa-Liehr, A. W.: Time Series FeatuRe Extraction on Basis of Scalable Hypothesis Tests (tsfresh – A Python Package). Neurocomputing, Vol. 307, 2018, pp. 72–77, doi: 10.1016/j.neucom.2018.03.067.

[17] Corder, G. W.—Foreman, D. I.: Nonparametric Statistics: A Step-by-Step Approach. John Wiley and Sons, 2014.

[18] Benjamini, Y.—Yekutieli, D.: The Control of the False Discovery Rate in Multiple Testing Under Dependency. The Annals of Statistics, Vol. 29, 2001, No. 4, pp. 1165–1188, doi: 10.1214/aos/1013699998.

[19] Olson, R. S.—Moore, J. H.: TPOT: A Tree-Based Pipeline Optimization Tool for Automating Machine Learning. In: Hutter, F., Kotthoff, L., Vanschoren, J. (Eds.): Automated Machine Learning: Methods, Systems, Challenges. Springer, Cham, The Springer Series on Challenges in Machine Learning, 2019, pp. 151–160, doi: 10.1007/978-3-030-05318-5_8.

[20] Feurer, M.—Klein, A.—Eggensperger, K.—Springenberg, J. T.—Blum, M.—Hutter, F.: Auto-Sklearn: Efficient and Robust Automated Machine Learning. In: Hutter, F., Kotthoff, L., Vanschoren, J. (Eds.): Automated Machine Learning: Methods, Systems, Challenges. Springer, Cham, The Springer Series on Challenges in Machine Learning, 2019, pp. 113–134, doi: 10.1007/978-3-030-05318-5_6.

[21] Cabrera, D.—Sancho, F.—Li, C.—Cerrada, M.—Sánchez, R.-V.—Pacheco, F.—de Oliveira, J. V.: Automatic Feature Extraction of Time-Series Applied to Fault Severity Assessment of Helical Gearbox in Stationary and Non-Stationary Speed Operation. Applied Soft Computing, Vol. 58, 2017, pp. 53–64, doi: 10.1016/j.asoc.2017.04.016.

[22] Mierswa, I.—Morik, K.: Automatic Feature Extraction for Classifying Audio Data. Machine Learning, Vol. 58, 2005, pp. 127–149, doi: 10.1007/s10994-005-5824-7.

[23] Sternickel, K.: Automatic Pattern Recognition in ECG Time Series. Computer Methods and Programs in Biomedicine, Vol. 68, 2002, No. 2, pp. 109–115, doi: 10.1016/S0169-2607(01)00168-7.

[24] Samiee, K.—Kovács, P.—Gabbouj, M.: Epileptic Seizure Classification of EEG Time-Series Using Rational Discrete Short-Time Fourier Transform. IEEE Transactions on Biomedical Engineering, Vol. 62, 2015, No. 2, pp. 541–552, doi: 10.1109/TBME.2014.2360101.

[25] Aggarwal, C. C.: Data Mining: The Textbook. Springer, 2015, doi: 10.1007/978-3-319-14142-8.

[26] Hastie, T.—Tibshirani, R.—Friedman, J.: The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer New York, Springer Series in Statistics, 2009, doi: 10.1007/978-0-387-84858-7.

[27] Wulff, K.—Dijk, D.-J.—Middleton, B.—Foster, R. G.—Joyce, E. M.: Sleep and Circadian Rhythm Disruption in Schizophrenia. The British Journal of Psychiatry, Vol. 200, 2012, No. 4, pp. 308–316, doi: 10.1192/bjp.bp.111.096321.

[28] Cohrs, S.: Sleep Disturbances in Patients with Schizophrenia. CNS Drugs, Vol. 22, 2008, No. 11, pp. 939–962, doi: 10.2165/00023210-200822110-00004.

[29] Christ, M.—Kempa-Liehr, A. W.—Feindt, M.: Distributed and Parallel Time Series Feature Extraction for Industrial Big Data Applications. 8th Asian Machine Learning Conference (ACML 2016), 2016, arXiv: 1610.07717.

[30] Support Vector Machines – Scikit-Learn 0.24.2 Documentation. Available at: `https://scikit-learn.org/stable/modules/svm.html`.

[31] Zou, H.—Hastie, T.: Regularization and Variable Selection via the Elastic Net. Journal of the Royal Statistical Society: Series B (Statistical Methodology), Vol. 67, 2005, No. 2, pp. 301–320, doi: 10.1111/j.1467-9868.2005.00503.x.

[32] Pedregosa, F.—Varoquaux, G.—Gramfort, A.—Michel, V.—Thirion, B.—Grisel, O.—Blondel, M.—Prettenhofer, P.—Weiss, R.—Dubourg, V.—Vanderplas, J.—Passos, A.—Cournapeau, D.—Brucher, M.—Perrot, M.—Duchesnay, E.: Scikit-Learn: Machine Learning in Python. Journal of Machine Learning Research, Vol. 12, 2011, pp. 2825–2830, arXiv: 1201.0490.

[33] Balanced Accuracy Metric – Scikit-Learn 0.24.2 Documentation. Available at: `https://scikit-learn.org/stable/modules/generated/sklearn.metrics.balanced_accuracy_score.html`.

[34] Overview on Extracted Features, tsfresh Documentation. Available at: `https://tsfresh.readthedocs.io/en/latest/text/list_of_features.html`.

[35] Batista, G. E.—Keogh, E. J.—Tataw, O. M.—de Souza, V. M. A.: CID: An Efficient Complexity-Invariant Distance for Time Series. Data Mining and Knowledge Discovery, Vol. 28, 2014, No. 3, pp. 634–669, doi: 10.1007/s10618-013-0312-3.

[36] Nguyen, V.—Deeds-Rubin, S.—Tan, T.—Boehm, B.: A SLOC Counting Standard. Cocomo II Forum, Citeseer, 2007, pp. 1–16.

[37] van Rossum, G.—Warsaw, B.—Coghlan, N.: PEP 8 – Style Guide for Python Code. Available at: `https://www.python.org/dev/peps/pep-0008/`.

[38] Kursa, M. B.—Jankowski, A.—Rudnicki, W. R.: Boruta – A System for Feature Selection. Fundamenta Informaticae, Vol. 101, 2010, No. 4, pp. 271–285, doi: 10.3233/FI-2010-288.

[39] Ding, H.—Trajcevski, G.—Scheuermann, P.—Wang, X.—Keogh, E.: Querying and Mining of Time Series Data: Experimental Comparison of Representations and Distance Measures. Proceedings of the VLDB Endowment, VLDB Endowment, Vol. 1, 2008, No. 2, pp. 1542–1552, doi: 10.14778/1454159.1454226.

[40] Deng, H.—Runger, G.—Tuv, E.—Martyanov, V.: A Time Series Forest for Classification and Feature Extraction. Information Sciences, Vol. 239, 2013, pp. 142–153, doi: 10.1016/j.ins.2013.02.030.

Jakub Adamczyk received his B.Sc. degree in computer science from the AGH University of Science and Technology, Kraków, Poland, in 2021. He is currently pursuing his M.Sc. degree in computer science – data science at the Institute of Computer Science, AGH-UST. His research interests include machine learning and data mining, especially related to biomedical applications.

Filip Malawski received his M.Sc. and Ph.D. degrees in computer science from the AGH University of Science and Technology, Kraków, Poland, in 2012 and 2019, respectively. Currently, he is Associate Professor at the Institute of Computer Science, AGH-UST. His main research areas are signal processing, computer vision, machine learning, and biomedical engineering.