When does oversampling techniques become efficient for Imbalanced Time Series Classification

Dorian Joubaud[a], Sylvain Kubler[a], Quentin Lao[a], Maxime Cordy[a], Yves Le Traon[a]

[a] *Interdisciplinary Centre for Security, Reliability & Trust (SnT), University of Luxembourg, Luxembourg*

Abstract

Class imbalance is very common in data analysis, it occurs when the proportion of classes among the data are different. With the success of deep learning in many domains such as computer vision, natural language processing, and more recently with time series, the requirement of a large amount of balance data is mandatory. The imbalance leads classifiers to overestimate the majority classes due to their proportion. Thus, data belonging to minority classes may be misclassified. Imbalanced data is common to many real-world problems, such as in the medical field with the prediction of rare but important diseases or the detection of anomalies in the manufacturing field. There are many solutions to overcome this problem. In particular, oversampling, which consists in generating new data to balance the minority classes. Several strategies exist to oversample data such as duplicating existing data (or random oversampling), using data augmentation techniques or other approaches such as SMOTE which is very popular.

Although advanced data augmentation techniques such as generative models are increasingly studied, basic data augmentation techniques remain widely used because of their ease of use and speed. We empirically evaluate the impact of balancing time series data sets using data augmentation on classification performances considering standard and neural networks classification models. Our results suggest that certain data augmentation techniques are more effective when applied to data with specific characteristics and within particular domains.

Keywords: Imbalanced Time Series Classification (ITSC), Machine Learning, Data Augmentation, Oversampling, Balancing data

1. Introduction

Classification tasks are among the most popular in data analysis. Supervised learning is most often used as a method to determine whether a data belongs to a particular class. The main idea of this approach is to produce a function based on the training data, with a view to predict to which class the studied data belongs. This means that the success of using of a learning classification algorithm depends largely on the selection of the data in the training database.

With the success of deep learning in many domains such as computer vision, natural language processing and more recently with time series, the requirement of a large amount of balance data is mandatory. Indeed issues like data scarcity and data imbalance can lead these models to overfit and thus affect the classification performances. Class imbalance in the training data leads classifiers to over-estimate the class associated with the majority group due to its increased proportion. As a result, data belonging to the minority group might be misclassified. However, it is often impossible to create large balanced data sets from real data. This problem is related to many real-life applications such as medical diagnosis with electrocardiogram (ECG) data analysis; fraud detection in banking operations, network intrusion detection networks, risk management, the prediction of technical equipment failures in manufacturing field with sensor data.

A popular solution to remedy the lack of data is data augmentation. It aims at creating synthetic data from the input space. However, most studies use data augmentation to increase the samples of the train set with the same distribution [1]. It has been shown that increasing the amount of data in time series datasets can drastically improve the accuracy of a deep learning model

*Corresponding author

Email addresses: dorian.joubaud@uni.lu (Dorian Joubaud), sylvain.kubler@uni.lu (Sylvain Kubler), (Quentin Lao), maxime.cordy@uni.lu (Maxime Cordy), yves.letraon@uni.lu (Yves Le Traon)

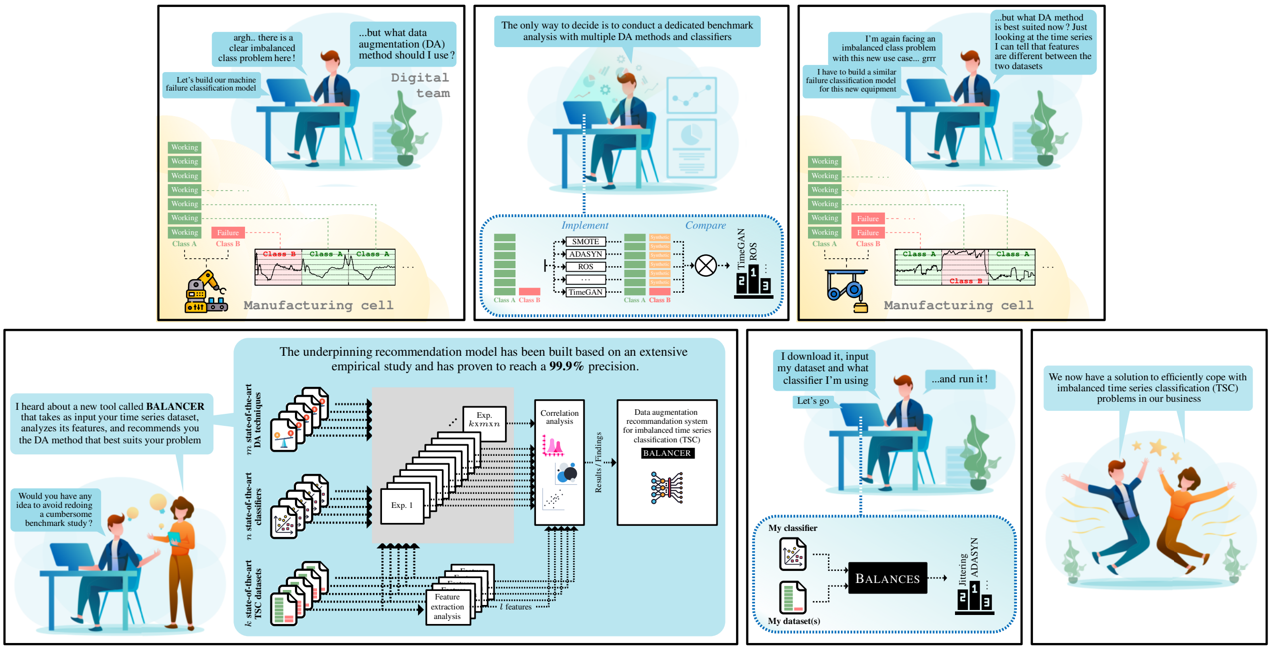


Figure 1: Comics

while having a slight negative impact on some datasets in the worst case.[] Although minority classes can benefit from more samples, majority class samples also increase, and thus problems related to imbalanced data persist. Balancing class distribution becomes crucial when dealing with imbalanced datasets in classification tasks. Imbalanced data can impede model performance by causing overfitting to the majority class, resulting in misclassification of minority class instances. In high-stakes scenarios, such as detecting medical anomalies or fraud, achieving a balance in class representation through techniques like data augmentation or resampling becomes imperative. This empowers the model to allocate sufficient attention to critical yet rare cases, ultimately enhancing the performance and reliability of classification outcomes.

In the literature, studies on balancing techniques focus mainly on benchmarking these techniques through global evaluations across various datasets. However, there is a noticeable gap in intensive investigations that dive into the specific conditions under which certain techniques perform better based on the inherent characteristics of individual input datasets. Such in-depth analyzes could shed light on the optimal strategies for addressing class imbalance in diverse scenarios, guiding practitioners to choose the most suitable approach depending on factors like dataset size, class imbalance, and dataset variance.

In this study, we empirically explore the effects of balancing time series datasets on classifier performance. This investigation covers a diverse set of classifiers, ranging from traditional machine learning algorithms to deep neural networks. To address class imbalance, we employ both fundamental and advanced data augmentation techniques sourced from existing literature. Additionally, we delve into the intrinsic characteristics of the time series datasets. Taking into account factors such as the number of data, data variance, or class imbalance levels, we discern the optimal data augmentation method for specific dataset profiles. Finally, our study culminates in the creation of a decision support tool. This tool uses both the characteristics of the dataset and the empirical findings to provide tailored recommendations for selecting appropriate data augmentation techniques. It also provides an estimate of the potential improvement of classifier performance through the inclusion of synthetic data.

This paper is organized as follows: Section 2 presents a comprehensive review of the existing literature concerning oversampling with data augmentation techniques to address data imbalance in time series datasets. In Section 3, we examine the current practices and approaches used to balance time series data, with a particular focus on how most studies compare various data augmentation techniques. Finally, Section 4 outlines our proposed methodology to conduct a comprehensive analysis of the suitability of data augmentation techniques in the context of time series data. In addition, we provide a description of a decision support tool designed to assist practitioners in selecting the most appropriate data augmentation technique for their input data based on the intrinsic characteristics of the time series.

2. Related Work

2.1. Imbalance data handling

2.1.1. Strategies

Two main approaches can be distinguished to address imbalanced data. The first corresponds to cost-sensitive or algorithm-level strategies [The foundations of cost-sensitive learning]. It aims to adjust the training procedure of classic classification algorithms to accommodate data imbalance and lessen the detrimental impact on minority classes classification performance. Every misclassification error does not have the same cost depending on the occurrence of each class. Data with fewer occurrences have a much greater weight in the training.

The second relies on data-level methods. It aims to artificially modify the training set to reduce data imbalance. One can find three methods to do so: undersampling, oversampling, and hybridsampling. Undersampling [] involves removing a certain amount of samples from the dominant classes. Removing data may potentially lead to a loss of information, particularly when the original training dataset is not sufficiently large, and thus may reduce the effectiveness of classification. However, undersampling remains a fast and efficient approach to balance the dataset, as it only utilizes a subset of the majority class for training. Oversampling involves increasing the number of samples in the minority classes. But the manner in which this is achieved raises questions. On the one hand, the synthesized data should closely resemble the original data, and on the other hand, overly similar samples may lead to overfitting issues. Finally, hybridsampling combines both oversampling and undersampling approaches. In this paper, our focus is specifically on oversampling methods for time-series classification.

2.1.2. Oversampling

One naive approach to oversampling, referred to as "Random Oversampling"[?], involves randomly selecting samples from the minority class and duplicating them. At first, the field of computer vision provides numerous more sophisticated oversampling methods []. They employ data transformation i.e. modifying existing data such as geometric transformations (e.g. flipping, rotating, jittering, cropping) and photometric transformations (e.g variations in colorimetry, contrast, exposure, white balance). Instead of transforming existing data, a widely used method called Synthetic Minority Oversampling TEchnique (SMOTE) [] involves combining existing data to create new samples. SMOTE generates new instances by interpolating existing neighbor samples selected through K-Nearest Neighbors. This technique has inspired several variants, including Borderline-SMOTE[], SVM-SMOTE[], ADASYN[], and others.

The lack of a standard procedure for data augmentation in time-series recognition is a major limitation. While data augmentation is a widely used technique in image recognition, X pointed out that the absence of a standard procedure for data augmentation in time-series recognition is a major obstacle. *"while data augmentation is a common practice in image recognition with neural networks, it is not established as a standard procedure for time series recognition."*. In fact, more advanced tools such as [2] provide an impactful insight on which data augmentation techniques is the best considering the input data. It uses reinforcement learning to learn the best augmentations policy from the data, treating the generalization of the target model as a reward signal. More recently, NVIDIA developed the NVIDIA Data Loading Library (DALI) facilitates image augmentation during deep learning training by offering automatic augmentation policies, such as AutoAugment, which enable probabilistic, policy-driven, random selection, and application of image transformations, and can be easily integrated and customized within the data processing pipeline through a straightforward API, enhancing computational efficiency and flexibility in training workflows.
Although many techniques in computer vision can be adapted for time-series classification, there is a need for specific techniques tailored to time-series data. Iwana et al.[1] provide a comprehensive taxonomy of such techniques in their survey, including Jittering, Scaling, and introducing more specialized techniques for time series, such as Time Warping. The requirement of minority classes oversampling in imbalanced time-series classification (ITSC) has spurred the development of adaptable data augmentation techniques, originally employed in computer vision, to be applied to time series such as TSMOTE [] and DTW-SMOTE [].
More recently, advanced techniques that exhibit remarkable performance in image processing have been intro-

duced, albeit at a higher computational cost: generative models. Variational Auto-Encoders (VAEs) [] aim to directly estimate the probability distribution of the data, while Generative Adversarial Networks (GANs) [] indirectly approximate the probability distribution by training two neural networks to compete against each other, one generating synthetic data and the other discriminating between synthetic and real data. Additionally, a class of score-based diffusion models, pioneered by Song and Ermon [], has emerged, outperforming GANs. In the context of time series, new architectures and variations of generative models, such as TS-GANs [] for GANs and Schrodinger bridge [] for diffusion models, have been proposed. Notably, generative models differ from the techniques mentioned earlier, as they aim to estimate the data probability distribution in order to generate new data that conforms to this distribution, rather than modifying existing data.

Although advanced data augmentation techniques are increasingly studied [Time-series Generative Adversarial Networks] [GENERATIVE ADVERSARIAL NETWORKS IN TIME SERIES: A SURVEY AND TAXONOMY], basic data augmentation techniques remain largely used due to their ease of use as well as their poor resource blow. The application of these techniques, which are primarily used to address the issues of data scarcity and incompleteness in deep learning model training, can also serve as a potential remedy for class imbalance by oversampling the underrepresented classes.

2.2. *Time Series Classification (TSC) features*

Utilizing features that characterize time series datasets offers valuable advantages in enhancing analysis tasks such as classification. Indeed, extracting relevant features can help to gain deeper insights. Moreover, given the abundance of data augmentation techniques available in the literature for time series data augmentation, it sounds interesting to know which techniques is more suitable to what kind of input data based on their caracteristics. In fact, [1] already conducted comparative comparative assessments of data augmentation methods using different neural network classifiers. Its results turned out to show that the choice of classifier is important depending on the input data. In other words, the impact of a data augmentation method may vary depending on the choice of classifier. To do so, it extracted 6 properties as well as the application domain of the input dataset. However, this previous research only considered neural network-based classifiers, a limited set of data augmentation methods and analyze big scale characteristics of time series dataset. We believe that using additional features that capture more finer properties of the input time series data help us to get more insight.

To this end *hctsa* (highly comparative time series analysis) toolbox [3] propose an architecture to extract over 7,700 time series features. However the relevance of all features as well as the computational cost of such features make it difficult to use with real-world data. In order to reduce redundancy of informations and the computation time, [3] introduce *catch22* (CAnonical Time-series CHaracteristics), a pipeline to reduce all the set of features to a subset of 22 the most useful and complementary characteristics for classification.

2.3. *Benchmarking studies of oversampling methods & Paper positioning*

Time series data introduces unique challenges due to its high dimensionality and temporal nature. They exhibit strong correlations between neighboring attributes, making the generation of consistent synthetic data particularly complex. This complexity partly accounts for the limited availability of data generation techniques for time series compared to image data. Sophisitized methods have been studied to augment time-series for oversampling purpose while preserving the internal structure of the data.

2.3.1. *Time Series augmentation for oversampling*

Several sophisticated methods have been suggested to oversampling minority classes while preserving the internal structure of the data. One such approach is the Mahalanobis distance-based sampling technique, which generates synthetic data that align with the covariance structure of the minority class, as proposed in [4]. However, it is not effective for datasets with a large number of dimensions, such as time series. Recently, [5] proposed a new temporal oriented smote variant T-SMOTE. It aims to generate more samples near the class border for minority classes by synthesizing samples along the line segment between a given minority data and its temporal neighbor. This paper realizes a comparison between T-SMOTE and 8 state-of-the-art oversampling techniques (SMOTE, Bordeline-SMOTE, ADASYN, MWSMOTE, MBS[6], INOS and MBO [7]). Overall, T-SMOTE outperforms all other techniques on 10 well-known datasets from the literature using a LSTM-based model.

However, many of these studies tend to focus on a broad comparison of new techniques with existing methods, emphasizing overall performance metrics. They often lack a detailed analysis specifying the particular scenarios or conditions in which a newly proposed

Table 1: Models and Data Augmentation Techniques

Model	**Description**
Multilayer Perceptrons (MLPs)	Basic neural network architectures consisting of multiple layers of interconnected nodes.
Random Forest (RF)	Ensemble model that combines multiple decision trees.
TimeSeries Random Forest (TS-RF)	Variation of Random Forest designed for time series data.
Dynamic Time Warping K-nearest neighbors (DTW-KNN)	Classifier using DTW distance metric for time series data.
ROCKET Kernel	Kernel-based algorithm using random convolutional kernels for time series data.
Shapelet	Classifier extracting discriminative subsequences from time series data.
Data Augmentation techniques	
Random Oversampling (ROS)	Method to address class imbalance by duplicating data from the minority class.
Jittering	Transformation involving adding noise to the time series.
Time warping	Transformation involving distorting the time series in the temporal environment.
Synthetic Minority Oversampling TEchnique (SMOTE)	Method to generate new minority samples by interpolating existing neighbors.
Adaptive synthetic (ADASYN)	Variant of SMOTE generating new data predominantly at class boundaries.
DTW-SMOTE	Variant of SMOTE using DTW distance for time series comparison.
TimeGAN	Data Augmentation method based on Generative Adversarial Networks (GANs) for time series data.
Schrodinger Bridge	Generative model for time series based on entropic interpolation through optimal transport.

technique may be more suitable or effective. By not providing these nuanced insights, the studies may not fully illuminate the unique advantages or applicability of the new methods in specific use cases or data conditions.

2.3.2. *Comparison between oversampling method with tabular*

Among tabular data oversampling techniques, studies have investigated the impact of different oversampling techniques on classifier performance with tabular data. It has been shown that, in general, oversampling seems to be a feasible solution to slightly improve the performance of classifiers [8]. It allows in particular to reduce the classification errors on the minority class. In addition, these studies suggest that some techniques are overall better than others. [9] performed an empirical analysis of 85 SMOTE variants over 104 datasets. They categorize oversampling techniques based on their main operating principle (e.g. noise removal, dimension reduction, or density-based) and characterize the datasets by three main attributes: The Imbalance Ratio, the number of samples in the minority class, and the total number of samples. Their conclusion emphasized that oversampling is a reliable solution to improve imbalance classification performance, that more advanced techniques do not necessarily perform better than basic smote, and that some techniques globally perform better than others. But even if this categorization permits the identification of which techniques perform the best in which scenario, it only considers class distribution attributes without considering data-based attributes. [10] performs the same type of experimentation on 96 synthetically generated imbalanced binary datasets. Each generated dataset has its own characteristics: size, clusters by class, number of features, and imbalance ratio. 4 oversampling techniques are studied using 5 different classifiers. This study shows that although performance is generally better with oversampling, the intrinsic characteristics of each dataset significantly impact the performance obtained.

The reviewed studies employ data generation techniques to oversample datasets in order to achieve a balanced distribution of classes. However, these studies tend to focus on a limited range of ”basic” oversampling techniques, as well as a short list of classifier.

2.3.3. *Paper positionning*

The intrinsic relationship between data features and the appropriateness of various data augmentation techniques, particularly in oversampling, is not extensively explored in the existing literature. Findings from studies that focus on tabular data often imply that no single oversampling technique universally excels across all datasets [9]. The effectiveness of a technique can depend on the unique characteristics inherent in a dataset, influencing its capacity to enhance classification performance. When it comes to time-series data, the complexity further complicates the applicability of augmentation techniques, limiting the usability of certain methods. Hence, there is a pressing necessity to deepen our understanding of the conditions under which various augmentation techniques, ranging from basic to sophisticated, can be most effectively employed for oversampling within diverse datasets.

Consequently, there is a need to analyze and determine which oversampling techniques are more suitable for specific datasets, based on their intrinsic characteristics. In this paper, we address this gap by presenting a comprehensive study investigating the impact of oversampling on classification performance, taking into account the characteristics of the dataset, including more advanced features such as cath22. Through this study, we aim to use a machine learning model that can predict the expected output performance from a classifier based on the input dataset features. This model acts like a decision support system, assisting AI practitioners in effectively oversampling their time-series datasets using data augmentation techniques, particularly when faced with class imbalance.

3. Data balancing: current way of practice

In a real-life context where a developer is faced with imbalanced data, it comes down to the basic scenario of completely balancing the dataset with popular data augmentation techniques. This scenario is consistent with the studies presented above. Each technique is compared for its overall performance. The user simply chooses the one he thinks is most effective, without worrying about whether it really fits his input data.

First, we adopt this systematic approach to assess the performance of 6 time series classification with 6 data augmentation techniques across multiple datasets. This allows us to rank the techniques based on their overall performance.

The table in 1 presents the 6 selected models for time series classification, alongside the 6 popular data augmentation techniques specifically designed for time series data. Detailed descriptions of each model and the corresponding data augmentation parameters can be found in Appendix .1.2.

To carry out the experiment, we first split the imbalanced dataset into a training set and a testing set. Initially, we train the classifier on raw data without any augmentation and evaluate its performance on the test set. Next, we completely balance the training set using a data augmentation method. We then train the classifier model on the new balanced/augmented dataset and assess its performance on the same testing set. To ensure reliable results, we repeat this process 20 times and take the mean performances.

Based on our initial experiment, we have successfully evaluated the performance of various techniques when paired with different classifiers. In particular, we have observed that the effectiveness of a specific data augmentation method can vary depending on the classifier employed.

The graph depicted in 3 illustrates that employing jittering to address data imbalances tends to yield positive results when used in conjunction with neural networks. However, its impact appears to be less pronounced when applied to kernel classifiers.

Furthermore, we observed that when using a specific classifier, one method can produce mixed results. In 2, we use green to represent a positive improvement in the classification performance with augmentation and red to indicate a negative impact for the KERNEL classifier. We notice that D_4 datasets and its modified version see that while its performance to be degraded when oversample with jittering and time warping techniques, using SMOTE tends to improve the performances. Identicaly, dataset D_{22} and its modified version benefit from oversampling with all techniques except with jiterring.

Based on our preliminary results, it is challenging to conclude on the effectiveness of data augmentation techniques to achieve balance in our datasets. Our findings suggest that the outcomes tend to be highly dependent on the specific choice of classifier, but also of the dataset. This variability underscores the importance of considering these factors when using data augmentation. Further investigation and analysis are required to fully understand the intricate relations between data augmentation, classifiers, and datasets in the context of achieving effective data balance.

4. Data balancing decision support for TSC

4.1. Data Acquisition, Balancing Classification

In this experimental study, we are dealing with an imbalanced dataset and employing a classifier model. The main objective is to assess the impact of different data augmentation methods on the performance of the classifier. We aim to compare the classifier's performance when using data augmentation versus when not using it and thus get the performance improvement when using data augmentation. In order to extract relevant informations, we need to gather a significant number of experiments with various type of data.

We generate new datasets based on a modification of one of the characteristics of the initial datasets from the UCR time series: imbalance. We have chosen to focus essentially on a variation of this feature, as it corresponds to the entry point from which a user will want to balance his datasets. However, it could be interesting to

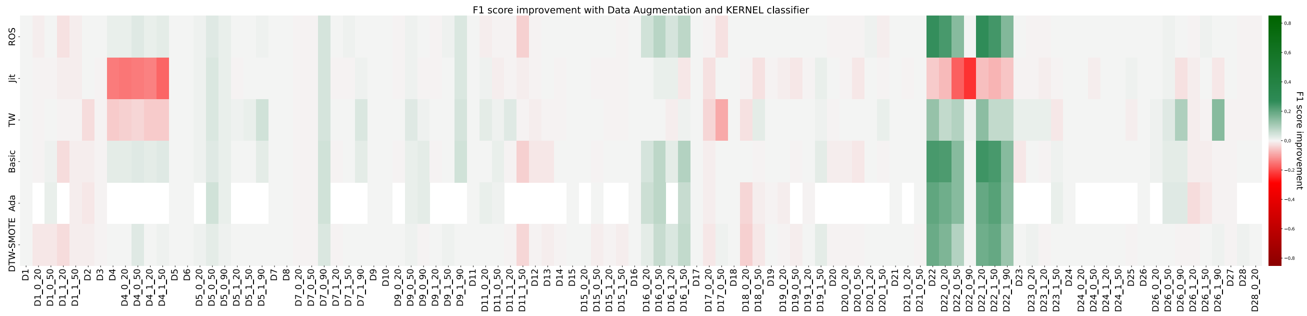


Figure 2: F1 score improvement after balancing for the KERNEL classifier
Green indicates improvement in performance; red indicates degradation in performance.
White indicate that the oversampling method cannot be applied in the scenario

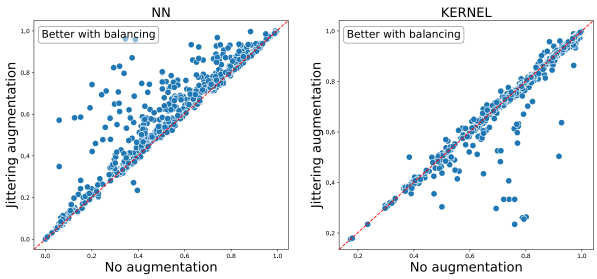


Figure 3: Neural Network and Kernel Classification performances with Jittering augmentation for balancing

play with other features. To do so, we artificially create a minority class by randomly removing $k \in \{20, 50, 90\}$ % of data belonging to a class. To guarantee that we have enough data to proceed with data augmentation, we only keep datasets with artificially generated minority classes that have more than two elements. For reasons of clarity, we denote our datasets from the UCR Time Series D_i with $i \in [\![1, ..., n]\!]$. Furthermore, in the case of a modified dataset, we denote $D_{i\_j\_k}$ as the D_i data set whose class j is reduced by k %. A matching table of the new names and the original names can be found in Appendix .1.2.

4.2. *Dataset Features Extraction*

To gain deeper insights into the effectiveness of each data augmentation technique, we extract relevant characteristics from the dataset and analyze their influence on the performance of the classifier. This analysis helps us to determine the relationship between data augmentation techniques and the dataset. In other words, which data augmentation techniques are more suitable for a given dataset based on its inherent characteristics.

To obtain a solid base of features that characterize our datasets, we first refer to *hctsa* toolbox [2]. The hctsa framework incorporates a diverse range of algorithms to extract 7658 features from time series data. These encompass fundamental statistics, linear correlations, stationarity measures, entropy computations, and techniques from nonlinear time series analysis. It also involves evaluations of linear and nonlinear model parameters, fits, and predictive capabilities. In particular, hctsa integrates methods from various disciplines, including physics, making it a comprehensive tool for time series analysis. However, the extensive number of features contributes to the computational complexity, and the similarity among several of these features diminishes the significance of certain ones.

Catch22 [3] introduces a method to infer small sets of time series features that ensure strong classification performance and minimize redundancy between features.

To achieve this, it first removes features that are sensitive to mean and variance, as well as features that return special values. Then, it scores the performance of individual remaining features across classification tasks using a decision tree model and a given number of cross-validation for each classification task. Each feature is ranked based on their performance score. The objective is to cluster the subset of the β-highest performing using the Pearson correlation distance $d = 1 - r$, with r the Pearson correlation coefficient between feature scores. Finally, they selected a single feature to represent each cluster based on their score and their interpretability according to them. In the end, *catch22* extract 22 features that approximate the classification performances from the initial features to 90% and compute much faster. A more concrete description of each feature can be found at [11].

Secondly, we refer to [1] who extracted 6 characteristics at the data set level. Its objective was to empirically analyze which data augmentation techniques are most suitable to which deep learning model when it comes

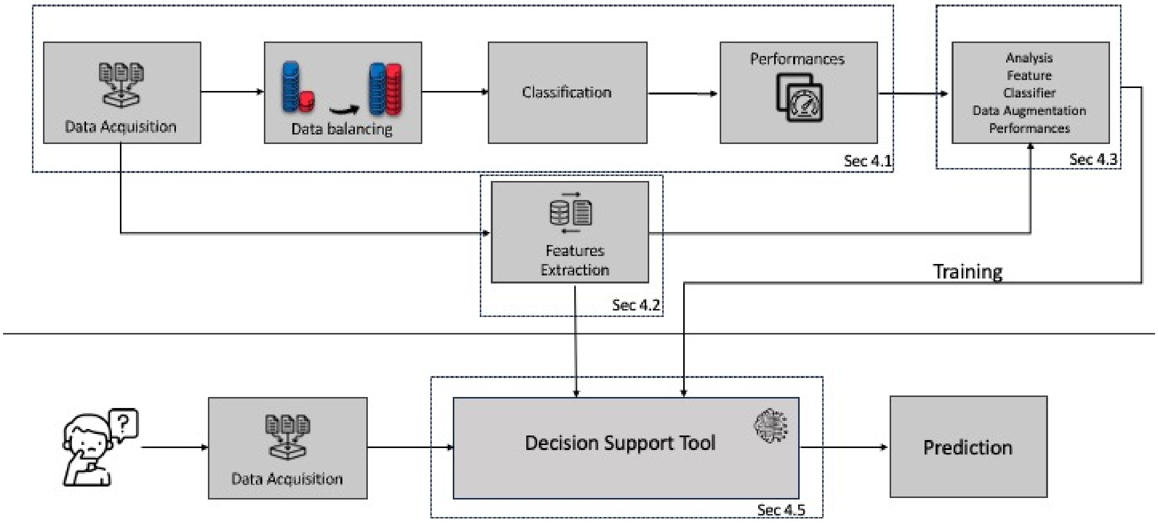


Figure 4: Caption

to classification. We further explore this direction by incorporating additional characteristics at the dataset level. One of these additional features we take into account is class separability. We believe that a clear distinction between classes allows greater flexibility in applying data transformations, as well as improved differentiation in learning patterns among classes using learning techniques. Conversely, when class boundaries are unclear, it results in less flexibility for generating accurate new data through transformations. To estimate the class separability we refer to [12], who initially introduced the Gaussian Bhattacharyya Coefficient (GBC) with the idea of measuring the overlap between target classes in the feature space of machine learning model for transferability purposes. Here we use this metric to measure ovelap directly on the input dataset.

Gaussian Bhattacharyya Coefficient. The Gaussian Bhattacharyya coefficient measures the overlap between two multivariate Gaussian distributions.
Let μ_{c_i} and μ_{c_i} (resp. Σ_{c_i} and Σ_{c_i}) be the mean vector (resp. covariance matrix) of class c_i and c_j. We denote $\Sigma = \frac{\Sigma_{c_j}+\Sigma_{c_i}}{2}$ and the Gaussian Bhattacharyya Coefficient between c_i and c_j as:

$$\mathrm{GBC}\left(c_i, c_j\right) = \sqrt{\frac{\sqrt{|\Sigma_{c_i}||\Sigma_{c_j}|}}{|\Sigma|}} \exp\left(-\frac{1}{8}{}^t\left(\mu_{c_i} - \mu_{c_j}\right)\Sigma^{-1}\left(\mu_{c_i} - \mu_{c_j}\right)\right)$$

Finally, we have :

$$GBC = \sum_{i<j} GBC(c_i, c_j)$$

A GBC value equal to zero corresponds to a dataset with completely distinct classes.

We gather all the data-level and dataset-level features used in Table 2.

4.3. Decision Support Model

Exploring the intricate relationships between dataset features and classification performances when harmonizing with data augmentation can be challenging. Furthermore, it can be difficult for practitioners to discern which technique is most appropriate for each feature and to formulate a comprehensive classification of the techniques to be implemented. Indeed, complex relationships can exist across multiple features.

To address this issue, we examine data pairs (X_i, Y_i), where X_i symbolizes the input feature vector corresponding to the i^{th} dataset D_i over the all n datasets, consisting of 29 unique characteristics denoted as $(x_{i1}, x_{i2}, \ldots, x_{i35})$. The 29 features, x_{ij}, encapsulate specific information as per Table 2.

Y_i denotes the output performance vector corresponding to the i^{th} sample, composed of 42 features for each of the scenario classifier / data augmentation.

Table 2: Time series features extracted from [1] and *catch22*

Feature	**Description**
Dataset Level	
`Dataset size` (DS)	Number of time series in the input dataset
`Length` (L)	Length of time series in the input dataset
`Average Patterns per Class` (APC)	Average count of data occurrences in each class
`Dataset Variance` (DV)	Average variance of each component of time series in the dataset
`Intra-class Variance` (IV)	Average variance per class (element-wise)
`Class Imbalance` (ID)	Measures the distribution of classes. A larger value indicates a non-equitable distribution, 0 indicates an equitable distribution
`Gaussian Bhattacharyya Coefficient` (GBC)	Measure of the class overlapping
Data Level	
Distribution	
`DN_HistogramMode_5` (DN5)	Most dominant z-score range based on the highest count of a 5-bin histogram
`DN_HistogramMode_10` (DN10)	Most dominant z-score range based on the highest count of a 10-bin histogram
Temporal Statistics	
`SB_BinaryState_Mean_Longest` (SB)	Longest period of consecutive values above the mean
`DN_OutlierInclude_p_001_mdrmd` (DNOp)	Time intervals between successive extreme events above the mean
`DN_OutlierInclude_n_001_mdrmd` (DNOn)	Time intervals between successive extreme events below the mean
Linear autocorrelation	
`CO_f1ecac` (COf1)	First 1/e crossing of autocorrelation function
`CO_FirstMin_ac` (COfi)	First minimum of autocorrelation function
`SP_Summaries_welch_rect_area_5_1` (SPa)	Total power in lowest fifth of frequencies in the Fourier power spectrum
`SP_Summaries_welch_rect_centroid` (SPc)	Centroid of the Fourier power spectrum
`FC_LocalSimple_mean3_stderr` (FCs)	Mean error from a rolling 3-sample mean forecasting
Nonlinear autocorrelation	
`CO_trev_1_num` (COt)	Time-reversibility statistic, $(x_{t+1} - x_t)^3$
`CO_HistogramAMI_even_2_5` (COh)	Automutual information, $m = 2, \tau = 5$
`IN_AutoMutualInfoStats_40_gaussian_fmmi` (IN)	First minimum of the automutual information function
Successive differences	
`MD_hrv_classic_pnn40` (MD)	Proportion of successive differences exceeding 0.04
`SB_BinaryStats_diff_longstretch0` (SBb)	Longest period of successive incremental decreases
`SB_MotifThree_quantile_hh` (SBm)	Shannon entropy of two successive letters in equiprobable 3-letter symbolization
`FC_LocalSImple_mean1_tauresrat` (FCm)	Change in correlation length after iterative differencing
`CO_Embed2_Dist_tau_d_expfit_meandiff` (COE)	Exponential fit to successive distances in 2-d embedding space
Fluctuation Analysis	
`SC_FluctAnal_2_dfa_50_1_2_logi_prop_r1` (FCd)	Proportion of slower timescale fluctuations that scale with DFA (50% sampling)
`SC_FluctAnal_2_rsrangefit_50_1_logi_prop_r1` (FCr)	Proportion of slower timescale fluctuations that scale with linearly rescaled range fits
Others	
`SB_TransitionMatrix_3ac_sumdiagcov` (SBT)	Trace of covariance of transition matrix between symbols in 3-letter alphabet
`PD_PeriodicityWang_th0_01` (PD)	Periodicity measure of (Wang et al.)

Transition from time-series data to tabular data for the purpose of characterizing datasets. This shift in perspective leads us into the realm of tabular data regression models, with the goal of predicting the estimated performance of our time-series classification models. In the study conducted by Borisov et al. in 2021 [13], a comprehensive benchmarking was carried out to assess the efficacy of various models for tabular data, ranging from gradient boosting models to deep neural networks. The findings of this investigation indicate that, despite significant advances in deep learning within the field of machine learning, gradient boosting models such as CatBoost [14], XGBoost [15] or LightGBM [16] continue to exhibit superior performance compared to all other state-of-the-art models. Gradient Boosting constructs a model in a stage-wise fashion and is renowned for optimizing an arbitrary differentiable loss function, allowing for enhanced predictive accuracy and model generalization across diverse dataset features.

We train a Gradient Boosting model, to predict Y_i based on X_i, thereby capturing the intricate relationships between the characteristics of the data set and the performance after balancing. This model allows us to delve into the complex interplay between different features and classification performances, providing a deeper understanding of the impact and efficacy of various data augmentation techniques.

To evaluate the performance of our model, we employ two metrics to compare the predicted values $\hat{Y}$ against the actual values Y . Since our initial problem is a regression task, we first use the Root Mean Square Error (RMSE) defined as:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}$$

Furthermore, to determine the effectiveness of recommending the optimal technique, we rank each value in Y from the best to the worst, generating a ranked list, denoted Y_r. For example, if $Y = [0.12, 0.15, 0.99, 0.56]$, then the corresponding ranking list would be $Y_r = [4, 3, 1, 2]$.

We denote Y_r as the actual ranked list and $\hat{Y}_r$ as the predicted ranked list. We employ Kendall's Tau, τ, to assess the agreement between the rankings of Y_r and $\hat{Y}_r$. It is computed as follows:

$$\tau(Y_r, \hat{Y}_r) = \frac{C - D}{\sqrt{(C + T_{Y_r})(C + T_{\hat{Y}_r})}}$$

Here, C represents the number of concordant pairs between Y_r and $\hat{Y}_r$, D represents the number of discordant pairs between Y_r and $\hat{Y}_r$, T_{Y_r} is the number of pairs tied in Y_r and $T_{\hat{Y}_r}$ is the number of pairs tied in $\hat{Y}_r$. The value of τ ranges from -1 to 1; a value of 1 implies perfect concordance, -1 implies perfect discordance, and 0 denotes no correlation between the rankings.

5. In practice

In this section we explore the experimental results of our study. Specifically, in 5.1 we present in details the experimental parameters that we used, in 5.2 we discuss the correlation analysis between the characteristics and the improvement after and before balancing :

$$\Delta_{F1} = F1_{\text{after}} - F1_{\text{before}}$$

It turns out that there is a mixed type of relationship based on the chosen classifier and data augmentation techniques. These multivariable complex relationship between features, classifier, data augmentation techniques used, and performances lead to development of a machine learning model that can predict which scenario is the best for which data. We tackle this in 5.2.2. After validation of our model and we used eXplainable Artificial Intelligence (XAI) techniques to determine if the model deeply understand the relations extracted from the correlation analysis and if other relations rise.

5.1. Experimental Settings

We performed the preliminary experiments presented in 4 on the **XXX** datasets from the UCR time series repository and the modified siblings presented in 4.1.

We extracted the caractecrizing features from 2 for each datasets.

5.2. Exp results

5.2.1. Correlation analysis

Correlations analysis plots is shown in 5 for classifiers, all the other plots can be found in Appendix. Each row corresponds to the correlation between Δ_{F1} and the extracted dataset features for a given classifier (e.g. DTW-Neighbors for the first row).

Correlations are computed using the Spearman's rank correlation coefficient. We employ Spearman's rank correlation coefficient for the reasons detailed in [1] (can handle skewed variable distribution and robust to outliers) as well as its capacity to catch non-linear relationship between variables. To do so, it consists of estimating the correlation coefficient between the rank of the variables observations.The relationships between dataset features and Δ_{F1} seem to be significantly impacted by the combination of data augmentation techniques and classifiers used. Unique correlations appear when certain classifiers are paired with specific data augmentation techniques, each affecting various features distinctively.

Firstly, with the MLP classifier, the correlations across all data augmentation techniques appear fairly consistent. This could be attributed to the neural network architecture's ability to discern the intrinsic information of the input data, allowing it to learn essential information regardless of the applied transformations. However, there are noteworthy correlations with certain features. Features such as DNO, ID, and COh tend to be positively correlated with all techniques, while L and PD generally show a negative correlation. With respect to scenarios involving other classifiers, there are pervasive correlations, but there are also intrinsic correlations emerging with specific augmentation techniques. For instance, Random oversampling (ROS) has a positive relationship with DNOp, except when it is used in conjunction with the MLP classifier. This effect is quite subtle. In the case of Jittering, it shows significant behaviors with certain features; MD and SB tend to be positively and negatively correlated with Δ_{F1}, respectively, when used with RandomForest and TS-RandomForest. Additionally, with the KERNEL classifier, unique correlations appear: DNOn is negatively correlated, while DN10 and DN5 show positive correlations. This pattern is intriguing because other techniques do not exhibit similar behavior. Time warping augmentation exhibits a distinctive behavior. For all classifiers, except MLP, it presents a higher positive correlation compared to other techniques. Specifically, it is the only technique that reveals significant correlations with certain features (DN5 and DN10) when applied with the DTW-NN classifier. SMOTE and ADASYN exhibit similar trends, particularly showing negative correlations with DN5 and DN10 when paired with the kernel classifier. As for the DTW-SMOTE technique, its correlations are somewhat comparable to SMOTE and ADASYN, but with lower values, except when used with the KERNEL classifier, where elevated values are observed.

This correlation analysis sheds light on the relationships between Δ_{F1}, dataset features, models and the applied oversampling techniques. It is crucial to recognize that these observed correlations may not encapsulate the full complexity of the interactions involved. The analysis might not be exhaustive enough to recommend a definitive data augmentation technique, as these methods come with their subtleties, influencing the dataset and classifiers beyond what the correlation metrics can convey.

5.2.2. *Model building & training/testing*

We created a relevant dataset using the results of the experiment described in 3. To find the best hyperparameters for our model, we conducted a grid search (learning rate : 0.04321, max depth : 7, n estimators : 477, gamma : 0, subsample : 0.5713, colsample bytree: 0.6081, reg alpha : 0.1, min child weight: 5, scale pos weight : 1). The dataset is divided into training and testing sets with an 80/20 split. To gain a more comprehensive understanding of the model's effectiveness, we applied a 10-fold cross-validation.

The results obtained from the test set are summarized in Table 4. Predictions were categorically grouped per model to facilitate a systematic ranking among the various data augmentation techniques. The global mean of Kendall's tau was used to compare the predicted optimal data augmentation techniques with their actual optimal technique. Additionally, both the Mean Squared Error (MSE) and variance were computed for each grouped prediction. The estimated performance prediction, when employing data augmentation, was found to be notably commendable. Furthermore, considering that all data augmentation techniques can result in closely related performance scores in certain scenarios, Kendall's tau serves to indicate the capability of the model to accurately rank the different data augmentation techniques. Evidently, the model demonstrates considerable ranking efficacy in all scenarios. Specifically, the model has a remarkable ability to decide which data augmentation technique should be used for oversampling in conjunction with various models such as Random Forest, Kernel, MLP, and DTW-NEIGHBORS. This is evidenced by a strong Kendall's tau value exceeding 0.7, illustrating the model's robust and relevant augmentation ranking capabilities. For TS-Random Forest and Shapelet models, the model faces slight challenges, reflected by a modest Kendall's tau value above 0.4. Despite this, the model still maintains an acceptable level of effectiveness in ranking the augmentation techniques. More detailed results can be found in the appendix.

5.3. *Feature Importance*

One of the key benefits of employing gradient-boosting methods lies in their interpretability. Determining the importance of features in these models is a straightforward process. In their work, [17] introduced SHAP, a unified approach that improves our understanding of model predictions. SHAP enables us to precisely

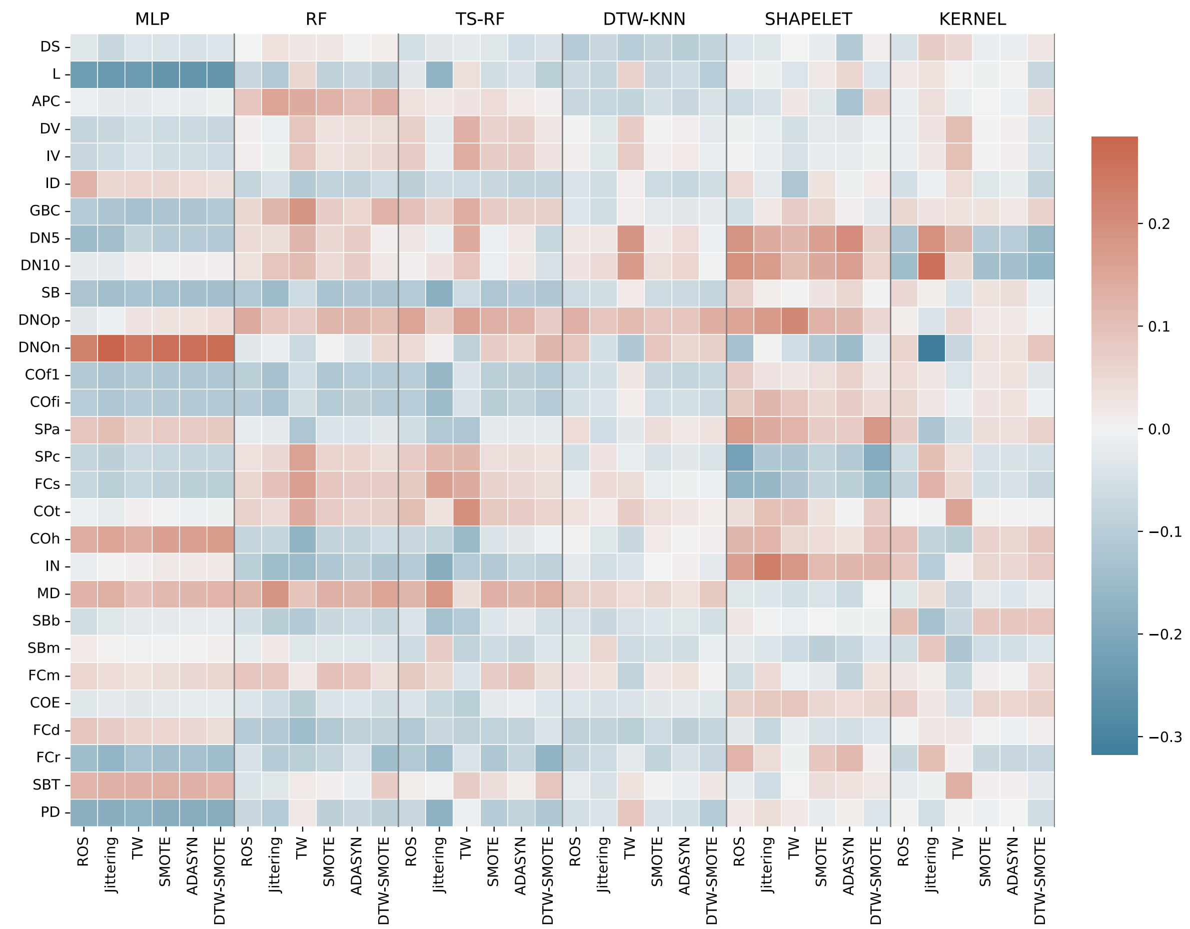


Figure 5: Correlation between features and Δ_{F1}

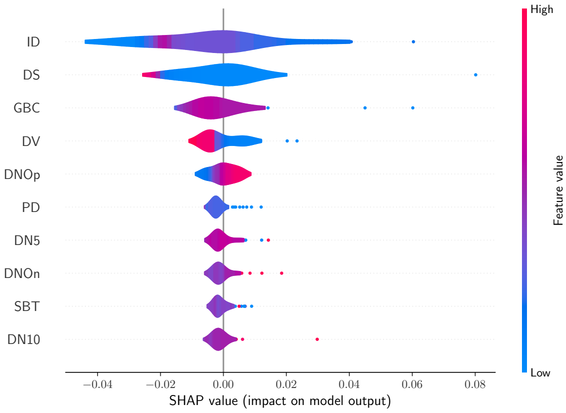
Model	Kendall's Tau	MSE
Random Forest	0.73 ± 0.08	1e−2 ± 3e−4
TS Random Forest	0.46 ± 0.06	1e−2 ± 5e−4
Kernel	0.79 ± 0.04	3e−3 ± 5e−5
Shapelet	0.43 ± 0.11	3e−2 ± 2e−5
MLP	0.91 ± 0.01	1e−2 ± 4e−6
DTW-NEIGBORS	0.77 ± 0.07	1e−2 ± 1e−3

Table 3: Comparative Analysis of Model performance on Test set using Kendall's Tau and MSE

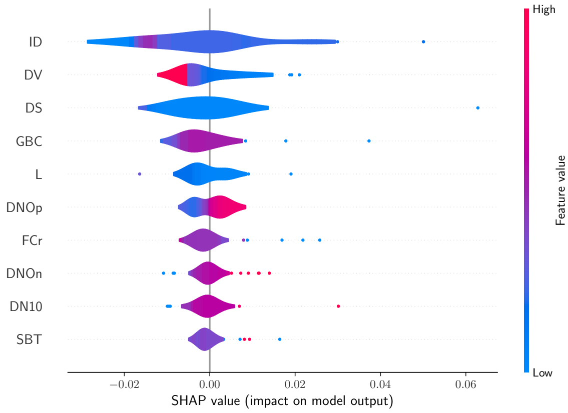
assess which features carry the most significant weight in a model's predictions, offering concrete insights into their impact. Therefore, we use eXplainable Artificial Intelligence (XAI) techniques to gain insight into the inherent connections between features and the classification performance learned by our model. This approach enables us to determine whether BALANCER captures the same relationships as identified in our earlier study. Furthermore, it empowers us to investigate whether the model uncovers previously undiscovered, deeper connections.

In 6, a summary plot illustrates the 10 most important characteristics of time series datasets given to each of the data augmentation techniques. Features are methodically ranked in ascending order, with the most critical feature prominently displayed at the top. The x-axis delineates the impact, either positive or negative, of the corresponding characteristic values. A shade of red indicates a high value of the feature, while blue signifies a low value. This visual representation assists in comprehending the significance and influence of each feature, facilitating an informed evaluation of the data augmentation techniques in the context of the respective time-series datasets.

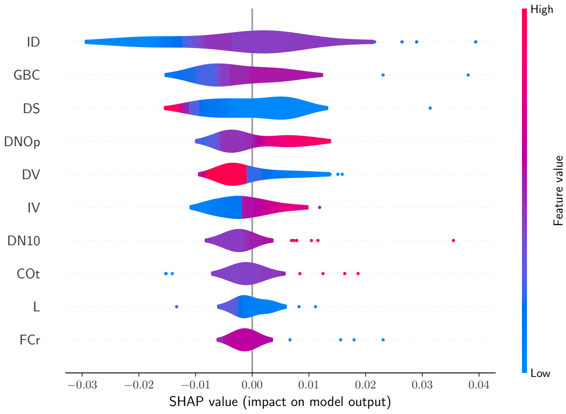
SHAP values have revealed notable trends in our model, showing the various effects of individual features in forecasting results when applied for balancing purposes through multiple data augmentation techniques. Notable characteristics like ID, Battacharyya (GBC), dataset variance (DV), dataset size (DS), and DNOp are consistently found to be highly significant. A careful examination reveals subtle connections between data augmentation strategies and certain characteristics. The ID feature dominates consistently, always securing the top ranking position in all techniques, affirming its universal influence. On the contrary, features



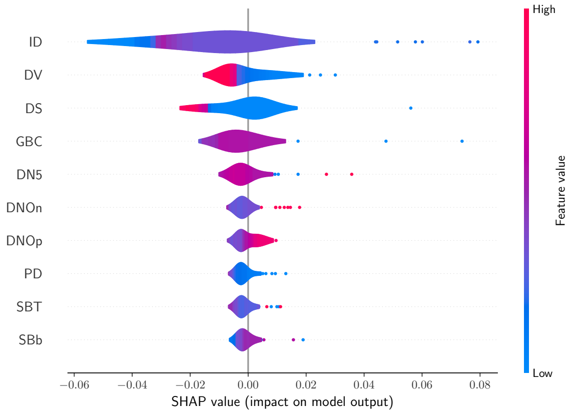
((a)) ROS



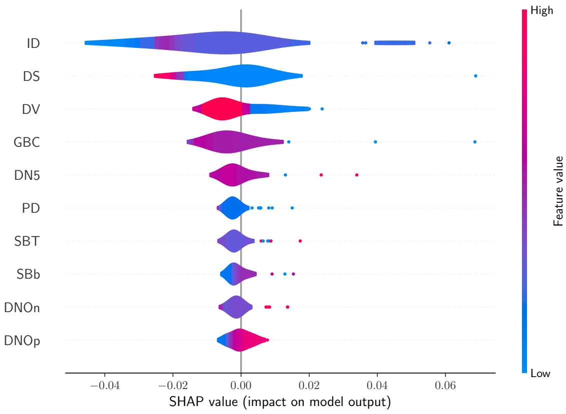
((b)) Jittering



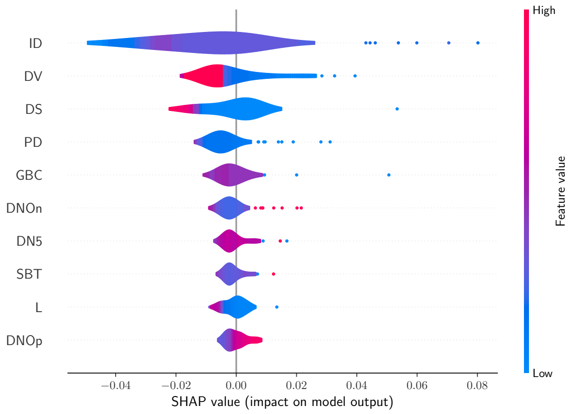
((c)) Time Warping



((d)) SMOTE



((e)) ADASYN



((f)) DTW-SMOTE

Figure 6: Mean TOP 10 shapeley values from (a) Random Oversampling, (b) Jittering, (c) Time Warping, (d) SMOTE, (e) ADASYN and (f) DTW-SMOTE

such as GBC, DN_HistogramMode_10/5(DN10, DN5), DV, PD, and length (L) demonstrate variability in their significance, revealing that their impact is contextually dependent on the applied data augmentation technique.

This exploration delineates four behaviors among the top ten critical features:

Ascendant Features: These enhance model performance when values are higher and, inversely, diminish performance when values are lower.

Descendant Features: These improve performance when values are lower, while higher values tend to reduce performance.

Uniform Features: These features maintain a consistent type of value, predominantly high or low.

Complex Features: Exhibiting intricate relationships, these features do not conform to straightforward influences on performance.

Examining the ROS technique reveal DNOp as an ascendant feature, where larger temporal intervals between extreme events relative to the mean enhance model performance. On the contrary, features such as DS and DV emerge as descendant features, suggesting that larger and more volatile datasets could diminish the efficacy of the ROS technique. Moreover, several features, including GBC, DN5, DNOn, SB_TransitionMatrix_3ac_sumdiagcov (SBT), and DN10, manifest themselves as complex features, highlighting significant, though mixed, influences on the model.

Using the Jittering technique, features such as DNOp continue to assert themselves as ascendant features, increasing Δ_{F1} when the time gap between extreme occurrences is considerable. Unique to Jittering, DS and L appear as uniform features, where smaller datasets and shorter time series optimally influence model performance.

Time Warping reflects DNOp as a predominant ascendant feature. Meanwhile, DV and DS emerge as descendant features, displaying an adverse effect on performance with larger and more variable datasets.

A pattern of feature importance is evident when considering SMOTE, ADASYN, and DTW-SMOTE, although there are slight variations. DNOp is consistently the most influential feature, leading to improved performance with these techniques. On the contrary, DS and DV typically act as descendant features, where larger or more variable datasets seem to hinder Δ_{F1}

The results demonstrate the intricate nature of determining clear connections between the characteristics of a dataset and the enhancement of performance. They accentuate the multifaceted interactions among features, affirming that while individual features maintain intrinsic significance, their collective interplay profoundly influences model behavior. Using a Gradient Boosting model aims to capture theses complex relationship, it can be complicated to represent them in a single dimension.

Top Important Feature	Correlations	SHAP
ROS	DNop	5e−3 ± 2e−6
Jittering	0.75 ± 0.06	6e−3 ± 5e−6
Time Warping	0.76 ± 0.06	5e−3 ± 2e−6
SMOTE	0.77 ± 0.03	7e−3 ± 1e−5
ADASYN	0.73 ± 0.06	6e−3 ± 4e−6
DTW-SMOTE	0.76 ± 0.03	6e−3 ± 5e−6

Table 4: Comparative Analysis of Model performance on Test set using Kendall's Tau and MSE

5.4. Comparison between correlations and feature importance

The empirical analysis allows us to identify meaningful connections between features and the improvements made possible by different data augmentation techniques based on correlation. Additionally, our model shows varying results when trying to determine which data augmentation technique is most suitable. Examining the knowledge from our model and comparing it to the dependencies found in the empirical study not only verifies some previously discovered relationships but also gives a deeper understanding of the essential dataset features when practitioners want to use data augmentation on an imbalanced time-series dataset. When comparing the correlation from Figure .5 with feature importance from Figure 6, intesting insight emerge. First, many highly correlated features for a given augmentation technique are present as the most important feature related to the augmentation technique for BALANCER (e.g. DNOp when used with ROS; L, DNOn and SBT when used with Jittering; SBb when used with SMOTE & ADASYN). Moreover, an intriguing observation emerges from comparing features with complex behaviors with SHAP value importance against their correlation coefficients. They seem to have minimal correlations. This notable observation underscores the limitation of relying solely on simple correlation analyses, as they might not fully unveil the multifaceted relationships that certain features share with others.

Such a phenomenon emphasizes the need to go beyond basic correlation analyses. Using machine learning models ability to capture these nuanced relationships accentuates the need to employ more sophisticated analytical approaches to unravel the complexity of interactions amongst features using XAI techniques such as SHAP. In the case of this study to enravel the intrinsic link existing between datasets features, oversampling statesgies through several state-of-the-art data augmentation techniques and the link improvement of performance.

5.5. Mise a disposition de nos modèles pour un praticioneur

The prediction model discussed in this paper offers an accurate way to estimate the performance improvements achieved when balancing a time series dataset for classification tasks. It has been extensively trained on a diverse set of 800 example datasets that encompass various scenarios, including different classifiers and data augmentation techniques. Additionally, it has the ability to reliably rank these techniques based on the F1 score achieved after balancing.

A primary advantage of this model lies in its ability to assist practitioners in selecting the most suitable data augmentation technique, thereby saving them valuable time. The model can quickly generate predictions within a matter of seconds, helping practitioners make informed decisions about the optimal data augmentation approach to use with the aim of balancing their dataset.

References

[1] B. K. Iwana, S. Uchida, An empirical survey of data augmentation for time series classification with neural networks (????). URL: https://arxiv.org/abs/2007.15951. doi:10.48550/ARXIV.2007.15951.

[2] E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, Q. V. Le, AutoAugment: Learning augmentation policies from data (????). URL: https://arxiv.org/abs/1805.09501. doi:10.48550/ARXIV.1805.09501, publisher: arXiv Version Number: 3.

[3] C. H. Lubba, S. S. Sethi, P. Knaute, S. R. Schultz, B. D. Fulcher, N. S. Jones, catch22: CAnonical time-series CHaracteristics: Selected through highly comparative time-series analysis 33 (????) 1821–1852. URL: http://link.springer.com/10.1007/s10618-019-00647-x. doi:10.1007/s10618-019-00647-x.

[4] L. Abdi, S. Hashemi, To combat multi-class imbalanced problems by means of over-sampling techniques 28 (????) 238–251. URL: https://ieeexplore.ieee.org/document/7163639/. doi:10.1109/TKDE.2015.2458858.

[5] P. Zhao, C. Luo, B. Qiao, L. Wang, S. Rajmohan, Q. Lin, D. Zhang, T-SMOTE: Temporal-oriented synthetic minority oversampling technique for imbalanced time series classification, in: Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, International Joint Conferences on Artificial Intelligence Organization, ????, pp. 2406–2412. URL: https://www.ijcai.org/proceedings/2022/334. doi:10.24963/ijcai.2022/334.

[6] C.-L. Liu, P.-Y. Hsieh, Model-based synthetic sampling for imbalanced data 32 (????) 1543–1556. URL: https://ieeexplore.ieee.org/document/8668459/. doi:10.1109/TKDE.2019.2905559.

[7] Z. Gong, H. Chen, Model-based oversampling for imbalanced sequence classification, in: Proceedings of the 25th ACM International on Conference on Information and Knowledge Management, ACM, ????, pp. 1009–1018. URL: https://dl.acm.org/doi/10.1145/2983323.2983784. doi:10.1145/2983323.2983784.

[8] A. Gosain, S. Sardana, Handling class imbalance problem using oversampling techniques: A review, in: 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), IEEE, ????, pp. 79–85. URL: http://ieeexplore.ieee.org/document/8125820/. doi:10.1109/ICACCI.2017.8125820.

[9] G. Kovács, An empirical comparison and evaluation of minority oversampling techniques on a large number of imbalanced datasets 83 (????) 105662. URL: https://linkinghub.elsevier.com/retrieve/pii/S1568494619304429. doi:10.1016/j.asoc.2019.105662.

[10] S. J. Dattagupta, A performance comparison of oversampling methods for data generation in imbalanced learning tasks, ???? URL: http://hdl.handle.net/10362/31307.

[11] catch22-features desciption, ???? URL: https://feature-based-time-series-analys.gitbook.io/catch22-features/feature-overview-table.

[12] M. Pándy, A. Agostinelli, J. Uijlings, V. Ferrari, T. Mensink, Transferability estimation using bhattacharyya class separability (????). URL: https://arxiv.org/abs/2111.12780. doi:10.48550/ARXIV.2111.12780, publisher: arXiv Version Number: 3.

[13] V. Borisov, T. Leemann, K. Seßler, J. Haug, M. Pawelczyk, G. Kasneci, Deep neural networks and tabular data: A survey (????). URL: https://arxiv.org/abs/2110.01889. doi:10.48550/ARXIV.2110.01889, publisher: arXiv Version Number: 3.

[14] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, A. Gulin, CatBoost: unbiased boosting with categorical features

(????). URL: https://arxiv.org/abs/1706.09516. doi:10.48550/ARXIV.1706.09516, publisher: arXiv Version Number: 5.

[15] T. Chen, C. Guestrin, XGBoost: A scalable tree boosting system (????). URL: https://arxiv.org/abs/1603.02754. doi:10.48550/ARXIV.1603.02754, publisher: arXiv Version Number: 3.

[16] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, T.-Y. Liu, LightGBM: A highly efficient gradient boosting decision tree, in: I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, R. Garnett (Eds.), Advances in neural information processing systems, volume 30, Curran Associates, Inc., ???? URL: https://proceedings.neurips.cc/paper_files/paper/2017/file/6449f44a102fde848669bdd9eb6b76fa-Paper.pdf.

[17] S. Lundberg, S.-I. Lee, A unified approach to interpreting model predictions (????). URL: https://arxiv.org/abs/1705.07874. doi:10.48550/ARXIV.1705.07874, publisher: arXiv Version Number: 2.

Measure imbalance & measure perf

In order to quantify the imbalance of a dataset, we refer to the distribution of the different classes inherent to it. Let's consider a dataset with K classes. We denote $(c_1, ..., c_K)$ the K classes and $\boldsymbol{\zeta} = (\zeta_1, ..., \zeta_K)$ their distributions. Thus each ζ_i estimates the probability of each class c_i by simply determining the frequency of class c_i in the data set. Formally

$$\zeta_i = \frac{1}{n}\sum_{k=1}^{n} \mathbb{1}(Y_k = c_k)$$

where n is the number of data in the dataset and Y_k the label of the k th data. Class c_i is said to be a minority class when $\zeta_i < \frac{1}{K}$. Reciprocally c_i is said to be a majority class when $\zeta_i > \frac{1}{K}$. In the case of a fully balanced dataset, we note $\boldsymbol{\zeta_{eq}} = e = (\frac{1}{K}, ..., \frac{1}{K})$

This description of the class distribution seems to be a good choice to compare datasets between them. However, their analysis can be quite tedious in problems with a large number of number of classes, or when studying datasets with a different number of classes.

In the literature, many measures exist to quantify the imbalance of a dataset. These measures are used to compare datasets and describe how unbalanced they are. One of the most use measure is Imbalance Ratio (IR) [] corresponding to the ratio between the number of elements of the most majority class and the number of elements of the most minority class

$$IR(\boldsymbol{\zeta}) = \frac{max_i \zeta_i}{min_i \zeta_i}$$

This measure is relevant to study data sets with two classes since it is injective, i.e. each distribution of two classes has a different measure. Thus each possible scenario can be found from an IR measure. However, the measure becomes obsolete for $K > 2$ (loss of injectivity and difficulty to estimate the order of magnitude of the measures).

Imbalance Degree or (ID) [] is based on the idea of extending the injectivity of IR for any K and having a defined interval of values for the measure. By subdividing the space of possible distributions into m subspaces corresponding to distributions with m minority classes, this measure gives values in the interval $(m-1; m]$. Moreover, this measure allows to choose between different metrics or divergences d_Δ to compare distributions.

$$ID(\zeta) = \frac{d_\Delta(\zeta, e)}{d_\Delta(\iota_m, e)} + (m-1)$$

where m is the number of minority classes, d_Δ is the chosen distance/divergence and i_m is the distribution showing exactly m minority classes with the largest distance from e. In practice this measure is complicated to use because of the interval in which the measure lies. It is only relevant to compare two distributions if they have the same number of minority classes.

KL ?

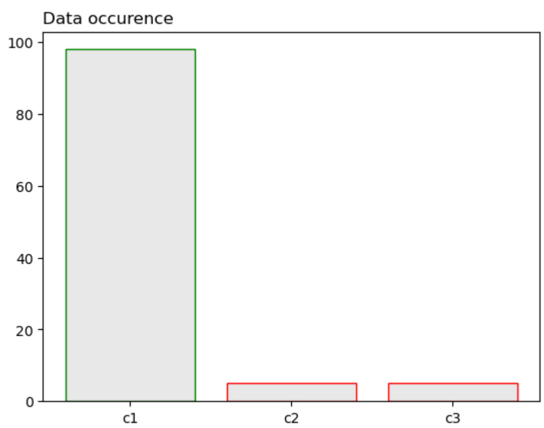


Figure .7: Imbalanced dataset distribution

The imbalance data sorely impacts classical classification methodes. This is due to the fact that the majority of classification algorithms aim at maximizing accuracy and minimizing overall error.

Considering an example of classification with a very imbalanced dataset with three classes such that $\zeta = (0.98, 0.01, 0.01)$. Let us consider a bad classifier which classifies all data as belonging to class 0. We then obtain 98% accuracy. This measure indicates that the classifier is good even though no class 2 and 3 data have been well classified.

It is therefore necessary to introduce measures that provide more accurate information on each information on each class.

Precision is the ratio of the number of data correctly assigned to a class with to total number of data assigned to class A. It measures the correctness of a classifier. Low precision indicates a high number of false positives.

Recall is the ratio of the number of data correctly assigned to class A to the total number of data actually belonging to class A. It measures the completeness of a classifier. A low sensitivity indicates a high number of false negatives.

$$\text{F1-score} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

F1-score computes harmonic mean of precision and recall by class. The scores of each class are averaged to obtain a single measure call macro F1-score.

$$\text{G-mean} = \sqrt{\text{Precision} * \text{Recall}}$$

The geometric mean (G-mean) measures the balance between precision and recall. For binary classification G-mean is the squared root of the product of the precision and recall. For multi-class problems it is a higher root of the product of recall for each class. The scores for each class are averaged for the scores of each class to obtain a single measure.

$$MCC = \frac{TP * TN - FP * FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TP + FN)}}$$

where

- TP: True Positives
- FN: False Negatives
- FP: False Positives
- TN: True Negatives

Matthew's Correlation Coefficient (MCC) is a correlation coefficient between actual and predicted between real and predicted classes. The value varies from -1 to +1 with a value of +1 representing a perfect prediction, 0 being no better than a random prediction and -1 the worst possible prediction.

Models

Multilayer Perceptrons (MLPs) are basic neural network architectures consisting of multiple layers of interconnected nodes. For benchmarking, we used an MLP with a specific architecture: time series input, followed by a dense layer with 64 neurons, rectified linear unit (ReLU) activation, dropout regularization (rate = 0.1), and classification output. We train the MLP for 100 epochs with a batch size of 32.

The **Random Forest** (RF) classifier is an ensemble model that combines multiple decision trees. For our benchmarking tests, we employ a Random Forest with 130 decision trees and a maximum depth of 50. The remaining hyperparameters are set to their default values.

The **TimeSeries Random Forest** (TS-RF) is a variation of the Random Forest algorithm specifically designed for time series data. It extracts time series features, such as mean and standard deviation, before constructing the ensemble of decision trees. We keep the same hyperparameters as Random Forest.

The **Dynamic Time Warping *K*-nearest neighbors** (DTW-KNN) classifier assigns a time series to the majority class among its nearest neighbors in the feature space, using the Dynamic Time Warping (DTW) distance metric. DTW measures the similarity between two time series, considering possible distortions in their temporal alignment. DTW is more appropriate than the Euclidian distance in the context of time series.

The **ROCKET Kernel** classifier is a kernel-based algorithm that uses random convolutional kernels to find important patterns in time series data. It analyzes segments of the data and converts them into a high-dimensional representation. For the benchmarking, we utilize 10,000 random convolutional kernels.

The **Shapelet** classifier extracts discriminative subsequences, called shapelets, from time series data. The classifier selects the most relevant shapelets and uses them for classification.

Data augmentation techniques

Appendix .1. Data generation methods

Many data generation techniques exist for the time series. These techniques can be divided into two main types of approaches: (i) basic techniques grouping together simple transformations in the temporal and frequency domain and (ii) advanced techniques grouping together learning methods and generative models.

Since data augmentation relies on generating new synthetic data from initial data, we make the assumption that all new generated data belong to the same class

Table .5: Spearman Correlation and p-value

	Default		ROS		Jittering		TimeWarping		Basic		ADASYN		DTW-SMOTE	
	Corr.	Pvalues	Corr.	Pvalues	Corr.	Pvalues	Corr.	Pvalues	Corr.	Pvalues	Corr.	Pvalues	Corr.	Pvalues
Length	-0.225	0.0	-0.324	0.0	-0.35	0.0	-0.355	0.0	-0.37	0.0	-0.373	0.0	-0.378	0.0
Dataset size	-0.042	0.33	-0.048	0.259	-0.065	0.128	-0.065	0.129	-0.063	0.14	-0.067	0.114	-0.063	0.142
Avg label size	0.117	0.006	0.126	0.003	0.122	0.004	0.117	0.006	0.124	0.004	0.121	0.005	0.126	0.003
Dataset variance	0.063	0.143	-0.058	0.177	-0.117	0.006	-0.107	0.012	-0.133	0.002	-0.135	0.002	-0.139	0.001
Intra-class variance	0.26	0.0	0.148	0.0	0.096	0.024	0.098	0.022	0.076	0.076	0.075	0.078	0.072	0.094
Bhattacharyya	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan
DN_HistogramMode_5	0.231	0.0	0.185	0.0	0.174	0.0	0.192	0.0	0.185	0.0	0.186	0.0	0.182	0.0
DN_HistogramMode_10	0.235	0.0	0.22	0.0	0.214	0.0	0.223	0.0	0.221	0.0	0.222	0.0	0.223	0.0
CO_f1ecac	-0.184	0.0	-0.215	0.0	-0.217	0.0	-0.214	0.0	-0.212	0.0	-0.213	0.0	-0.212	0.0
CO_FirstMin_ac	-0.136	0.001	-0.16	0.0	-0.162	0.0	-0.158	0.0	-0.156	0.0	-0.157	0.0	-0.155	0.0
CO_HistogramAMI_even_2_5	-0.067	0.119	-0.01	0.812	0.018	0.682	0.012	0.778	0.029	0.501	0.03	0.48	0.038	0.373
CO_trev_1_num	0.119	0.005	0.127	0.003	0.115	0.007	0.131	0.002	0.124	0.004	0.125	0.003	0.129	0.002
MD_hrv_classic_pnn40	0.206	0.0	0.242	0.0	0.253	0.0	0.239	0.0	0.247	0.0	0.249	0.0	0.253	0.0
SB_BinaryStats_mean_longstretch1	-0.122	0.004	-0.161	0.0	-0.165	0.0	-0.163	0.0	-0.165	0.0	-0.167	0.0	-0.168	0.0
SB_TransitionMatrix_3ac_sumdiagcov	-0.225	0.0	-0.164	0.0	-0.158	0.0	-0.152	0.0	-0.143	0.001	-0.145	0.001	-0.147	0.001
PD_PeriodicityWang_th0_01	-0.097	0.023	-0.187	0.0	-0.207	0.0	-0.204	0.0	-0.222	0.0	-0.224	0.0	-0.228	0.0
CO_Embed2_Dist_tau_d_expfit_meandiff	-0.057	0.181	-0.022	0.607	0.005	0.906	-0.004	0.916	0.011	0.799	0.014	0.751	0.022	0.613
IN_AutoMutualInfoStats_40_gaussian_fmmi	-0.114	0.007	-0.106	0.013	-0.092	0.03	-0.093	0.03	-0.083	0.051	-0.084	0.049	-0.08	0.061
FC_LocalSimple_mean1_tauresrat	0.045	0.294	0.06	0.162	0.054	0.206	0.049	0.253	0.049	0.252	0.053	0.213	0.057	0.179
DN_OutlierInclude_p_001_mdrmd	0.007	0.862	-0.013	0.758	-0.008	0.846	0.014	0.752	0.013	0.766	0.016	0.712	0.02	0.643
DN_OutlierInclude_n_001_mdrmd	0.003	0.937	0.073	0.088	0.111	0.009	0.1	0.02	0.114	0.007	0.114	0.008	0.121	0.005
SP_Summaries_welch_rect_area_5_1	-0.111	0.009	-0.101	0.018	-0.081	0.058	-0.091	0.032	-0.082	0.055	-0.082	0.056	-0.076	0.076
SB_BinaryStats_diff_longstretch0	-0.049	0.25	-0.046	0.287	-0.022	0.612	-0.031	0.463	-0.021	0.626	-0.019	0.654	-0.014	0.751
SB_MotifThree_quantile_hh	0.116	0.006	0.128	0.003	0.125	0.003	0.123	0.004	0.122	0.004	0.122	0.004	0.123	0.004
SC_FluctAnal_2_rsrangefit_50_1_logi_prop_r1	0.083	0.052	0.052	0.223	0.041	0.335	0.053	0.217	0.049	0.251	0.049	0.255	0.046	0.284
SC_FluctAnal_2_dfa_50_1_2_logi_prop_r1	-0.047	0.269	0.018	0.673	0.024	0.569	0.022	0.61	0.025	0.551	0.023	0.588	0.021	0.62
SP_Summaries_welch_rect_centroid	0.164	0.0	0.148	0.001	0.125	0.003	0.123	0.004	0.112	0.009	0.112	0.009	0.107	0.012
FC_LocalSimple_mean3_stderr	0.143	0.001	0.124	0.003	0.107	0.012	0.111	0.009	0.1	0.019	0.099	0.02	0.094	0.028
ID	-0.158	0.0	-0.105	0.014	-0.115	0.007	-0.116	0.007	-0.109	0.011	-0.114	0.008	-0.116	0.007

Model	Random Oversampling	Jittering	Time Warping	SMOTE	ADASYN	DTW-SMOTE
Random Forest	4.75e−3 ± 3e−6	5.02e−3 ± 8e−6	4.44e−3 ± 2e−6	6.44e−3 ± 6e−6	1.127e−2 ± 2.1e−5	5.46e−3 ± 9e−6
TS Random Forest	4.69e−3 ± 7e−6	4.47e−3 ± 3e−6	6.24e−3 ± 6e−6	1.037e−2 ± 1.5e−5	5.59e−3 ± 9e−6	3.82e−3 ± 2e−6
Kernel	5.03e−3 ± 4e−6	5.68e−3 ± 4e−6	1.335e−2 ± 2.2e−5	5.41e−3 ± 7e−6	4.38e−3 ± 4e−6	5.31e−3 ± 8e−6
Shapelet	5.21e−3 ± 7e−6	4.62e−3 ± 6e−6	6.99e−3 ± 1e−5	5.39e−3 ± 4e−6	6.91e−3 ± 7e−6	1.213e−2 ± 1.4e−5
MLP	1.014e−2 ± 1.6e−5	5.99e−3 ± 8e−6	4.18e−3 ± 3e−6	4.73e−3 ± 6e−6	6.12e−3 ± 1e−5	6.37e−3 ± 1e−5
DTW-Neighbors	9.42e−3 ± 2.3e−5	1.224e−2 ± 2.9e−5	6.93e−3 ± 5e−6	5.9e−3 ± 5e−6	5.98e−3 ± 8e−6	4.79e−3 ± 4e−6

Table .6: 10-Fold Validation Mean Squared Error per scenario model / data augmentation technique

Model	Random Oversampling	Jittering	Time Warping	SMOTE	ADASYN	DTW-SMOTE
Random Forest	5.18e−3 ± 1.77e−4	3.78e−3 ± 1.18e−4	4.76e−3 ± 2.53e−4	6.67e−3 ± 4.52e−4	8.4e−3 ± 3.3e−4	6.04e−3 ± 2.98e−4
TS Random Forest	3.94e−3 ± 2.11e−4	4.27e−3 ± 1.96e−4	7.07e−3 ± 4.84e−4	1.076e−2 ± 3.45e−4	5.97e−3 ± 3.2e−4	5.58e−3 ± 2.32e−4
Kernel	4.6e−3 ± 2.34e−4	6.83e−3 ± 5.31e−4	8.55e−3 ± 3.92e−4	5.89e−3 ± 2.61e−4	5.85e−3 ± 3.08e−4	4.42e−3 ± 1.98e−4
Shapelet	6.64e−3 ± 4.82e−4	5.18e−3 ± 2.33e−4	5.74e−3 ± 3.27e−4	4.17e−3 ± 1.61e−4	7.4e−3 ± 3.65e−4	9.75e−3 ± 5.5e−4
MLP	7.55e−3 ± 2.74e−4	5.53e−3 ± 2.04e−4	6.09e−3 ± 3.06e−4	4.17e−3 ± 1.95e−4	4.73e−3 ± 2.2e−4	8.06e−3 ± 6.12e−4
DTW-Neighbors	8.24e−3 ± 3.94e−4	8.51e−3 ± 2.85e−4	8.05e−3 ± 4.4e−4	6.47e−3 ± 2.66e−4	4.18e−3 ± 1.4e−4	4.55e−3 ± 2.2e−4

Table .7: Test set Mean Squared Error per scenario model / data augmentation technique

as the initial data. This hypothesis will be discussed later. Moreover, we consider only univariate time series to keep things simple first. In the rest of the paper, a (univariate) time series of length t is defined as a finite sequence $\boldsymbol{X} := (x_1, \ldots, x_t) \in \mathbb{R}^t$.

Appendix .1.1. Basic techniques

We take six popular basic techniques to generate synthetic data.

Random Oversampling (ROS) consists of randomly duplicating data from the minority class to the minority class to rebalance the dataset. It is a simple and straightforward method to address class imbalance but naive as it does not add any variations to the synthetic samples.

Jittering is one of the most basic transformations. It involves adding noise to the time series. Let $\boldsymbol{X} := (x_1, ..., x_t)$ be a real time series. Then, jittering is able to create a new sample $\boldsymbol{X'} = (x_1 + \varepsilon_1, ..., x_t + \varepsilon_t)$ where $\varepsilon_i \sim \mathcal{N}(0, \sigma^2)$ are independent and identically distributed (iid). The standard deviation σ is a hyperparameter that determines the amplitude of the added noise. In practice, σ^2 is set to 0.03. []

Time warping acts the same way as Jittering and consists of slightly distorting the initial data . However, the distortion takes place in the temporal environment. Let $\boldsymbol{X} := (x_1, ..., x_t)$ be a real time series. Then we have $\boldsymbol{X'} = (x_{\tau(1)}, ..., x_{\tau(t)})$ where $\tau : t \rightarrow \tau(t)$ is the warping function which distorts the time steps using a *smooth* function. This function is defined by a Hermite cubic spline, which allows to generate a polynomial interpolating a function between points $u = (u_1, ..., u_I)$ such that each u_i are independent and identically distributed with $u_i \sim \mathcal{N}(\mu, \sigma^2)$. Thus, the time steps of the series present a smooth transition between stretches and contractions.This technique requires solving a complex optimization problem and can be be computationally time consuming.

Synthetic Minority Oversampling TEchnique (SMOTE) is one of the most popular approaches. It creates new minority individuals that look like the others, without being strictly identical, and densifies the population of minority individuals more homogeneously[]. SMOTE generates new instances by interpolating existing neighbor samples selected through the K-Nearest Neighbors algorithm based on Euclidian distance. To generate new samples, a time series $\boldsymbol{X_0}$ is uniformly chosen from the minority classes. Let $\boldsymbol{X_1}, \ldots, \boldsymbol{X_K}$ be its K nearest neighbors of the same class. SMOTE generates K new samples as follows :

$$\begin{aligned} \boldsymbol{X'_1} &:= \boldsymbol{X_0} + \lambda_1(\boldsymbol{X_1} - \boldsymbol{X_0}) \\ &\vdots \\ \boldsymbol{X'_K} &:= \boldsymbol{X_0} + \lambda_K(\boldsymbol{X_K} - \boldsymbol{X_0}) \end{aligned}$$

where $\lambda_i \sim \mathcal{U}([0,1])$ iid.

Adaptive synthetic (ADASYN) is a variant of SMOTE where new data are generated predominantly at the class boundaries, where classification is most challenging. Unlike SMOTE, which uniformly selects a data point $\boldsymbol{X_0}$, ADASYN tends to choose data points in dense neighborhoods with samples of the majority class.

DTW-SMOTE is also a variant of SMOTE that uses the DTW distance instead. The use of Euclidean distance in SMOTE may not be suitable for comparing time series. For example, a cosine wave and a sine wave are likely considered different based on the Euclidean norm, even though they are intuitively similar because they only differ with a temporal shift. Thus, it is relevant to use DTW instead which measures the similarity between two time series, taking into account temporal shifts.

Appendix .1.2. Advanced techniques

TimeGAN [J Yoon, D Jarrett, M Schaar. Time-series Generative Adversarial Networks. NeurIPS 2019 ; 2019.] is a Data Augmentation method based on Generative Adversarial Networks (GANs). GANs involve training two neural networks to compete against each other, one generating synthetic data and the other discriminating between synthetic and real data. TimeGAN is specifically designed for time series data. It incorporates additional embedding components to capture the temporal characteristics of the time series.

Schrodinger Bridge is a recent generative model for time series based that belongs to the category of score-based models [M Hamdouche, P Henry-Labordere, H Pham. Generative modeling for time series via Schrödinger bridge. ; 2023]. It involves the entropic interpolation through optimal transport between a reference probability measure on path space and a target measure that aligns with the joint data distribution of the time series. [A VOIR SI ON GARDE CA, PARCE QUE C'EST PAS OUF]

Matching names table

New_name	Initial_name
D1	ACSF1
D1_0_20	ACSF1_RUS_0_20
D1_0_50	ACSF1_RUS_0_50
D1_1_20	ACSF1_RUS_1_20
D1_1_50	ACSF1_RUS_1_50
D2	AconityMINIPrinterLargeEq
D3	AconityMINIPrinterSmallEq
D4	Adiac
D4_0_20	Adiac_RUS_0_20
D4_0_50	Adiac_RUS_0_50
D4_1_20	Adiac_RUS_1_20
D4_1_50	Adiac_RUS_1_50
D5	AllGestureWiimoteX
D6	AllGestureWiimoteXEq
D5_0_20	AllGestureWiimoteX_RUS_0_20
D5_0_50	AllGestureWiimoteX_RUS_0_50
D5_0_90	AllGestureWiimoteX_RUS_0_90
D5_1_20	AllGestureWiimoteX_RUS_1_20
D5_1_50	AllGestureWiimoteX_RUS_1_50
D5_1_90	AllGestureWiimoteX_RUS_1_90
D7	AllGestureWiimoteY
D8	AllGestureWiimoteYEq
D7_0_20	AllGestureWiimoteY_RUS_0_20
D7_0_50	AllGestureWiimoteY_RUS_0_50
D7_0_90	AllGestureWiimoteY_RUS_0_90
D7_1_20	AllGestureWiimoteY_RUS_1_20
D7_1_50	AllGestureWiimoteY_RUS_1_50
D7_1_90	AllGestureWiimoteY_RUS_1_90
D9	AllGestureWiimoteZ
D10	AllGestureWiimoteZEq
D9_0_20	AllGestureWiimoteZ_RUS_0_20
D9_0_50	AllGestureWiimoteZ_RUS_0_50
D9_0_90	AllGestureWiimoteZ_RUS_0_90
D9_1_20	AllGestureWiimoteZ_RUS_1_20
D9_1_50	AllGestureWiimoteZ_RUS_1_50
D9_1_90	AllGestureWiimoteZ_RUS_1_90
D11	ArrowHead
D11_0_20	ArrowHead_RUS_0_20
D11_0_50	ArrowHead_RUS_0_50
D11_1_20	ArrowHead_RUS_1_20
D11_1_50	ArrowHead_RUS_1_50
D12	BME
D12_0_20	BME_RUS_0_20
D12_0_50	BME_RUS_0_50
D12_1_20	BME_RUS_1_20
D12_1_50	BME_RUS_1_50
D13	Beef
D13_0_20	Beef_RUS_0_20
D13_0_50	Beef_RUS_0_50
D13_1_20	Beef_RUS_1_20
D13_1_50	Beef_RUS_1_50
D14	BeetleFly
D14_0_20	BeetleFly_RUS_0_20
D14_0_50	BeetleFly_RUS_0_50
D15	BirdChicken
D15_0_20	BirdChicken_RUS_0_20
D15_0_50	BirdChicken_RUS_0_50
D16	CBF
D16_0_20	CBF_RUS_0_20
D16_0_50	CBF_RUS_0_50
D16_1_20	CBF_RUS_1_20
D16_1_50	CBF_RUS_1_50
D17	Car
D17_0_20	Car_RUS_0_20
D17_0_50	Car_RUS_0_50
D17_1_20	Car_RUS_1_20
D17_1_50	Car_RUS_1_50
D18	Chinatown
D18_0_20	Chinatown_RUS_0_20
D18_0_50	Chinatown_RUS_0_50
New_name	Initial_name
D19	ChlorineConcentration
D19_0_20	ChlorineConcentration_RUS_0_20
D19_0_50	ChlorineConcentration_RUS_0_50
D19_0_90	ChlorineConcentration_RUS_0_90
D19_1_20	ChlorineConcentration_RUS_1_20
D19_1_50	ChlorineConcentration_RUS_1_50
D19_1_90	ChlorineConcentration_RUS_1_90
D20	CinCECGTorso
D20_0_20	CinCECGTorso_RUS_0_20
D20_1_20	CinCECGTorso_RUS_1_20
D20_1_50	CinCECGTorso_RUS_1_50
D21	Coffee
D21_0_20	Coffee_RUS_0_20
D21_0_50	Coffee_RUS_0_50
D21_1_20	Coffee_RUS_1_20
D21_1_50	Coffee_RUS_1_50
D22	Colposcopy
D23	Computers
D23_0_20	Computers_RUS_0_20
D23_0_50	Computers_RUS_0_50
D23_0_90	Computers_RUS_0_90
D23_1_20	Computers_RUS_1_20
D23_1_50	Computers_RUS_1_50
D23_1_90	Computers_RUS_1_90
D24	Covid3MonthDiscrete
D25	CricketX
D25_0_20	CricketX_RUS_0_20
D25_0_50	CricketX_RUS_0_50
D25_0_90	CricketX_RUS_0_90
D25_1_20	CricketX_RUS_1_20
D25_1_50	CricketX_RUS_1_50
D25_1_90	CricketX_RUS_1_90
D26	CricketY
D26_0_20	CricketY_RUS_0_20
D26_0_50	CricketY_RUS_0_50
D26_0_90	CricketY_RUS_0_90
D26_1_20	CricketY_RUS_1_20
D26_1_50	CricketY_RUS_1_50
D26_1_90	CricketY_RUS_1_90
D27	CricketZ
D27_0_20	CricketZ_RUS_0_20
D27_0_50	CricketZ_RUS_0_50
D27_0_90	CricketZ_RUS_0_90
D27_1_20	CricketZ_RUS_1_20
D27_1_50	CricketZ_RUS_1_50
D27_1_90	CricketZ_RUS_1_90
D28	Crop
D28_0_20	Crop_RUS_0_20
D28_0_50	Crop_RUS_0_50
D28_0_90	Crop_RUS_0_90
D28_1_20	Crop_RUS_1_20
D28_1_50	Crop_RUS_1_50
D28_1_90	Crop_RUS_1_90
D29	DistalPhalanxOutlineAgeGroup
D29_0_20	DistalPhalanxOutlineAgeGroup_RUS_0_20
D29_0_50	DistalPhalanxOutlineAgeGroup_RUS_0_50
D29_0_90	DistalPhalanxOutlineAgeGroup_RUS_0_90
D29_1_20	DistalPhalanxOutlineAgeGroup_RUS_1_20
D29_1_50	DistalPhalanxOutlineAgeGroup_RUS_1_50
D29_1_90	DistalPhalanxOutlineAgeGroup_RUS_1_90
D30	DistalPhalanxOutlineCorrect
D30_0_20	DistalPhalanxOutlineCorrect_RUS_0_20
D30_0_50	DistalPhalanxOutlineCorrect_RUS_0_50
D30_0_90	DistalPhalanxOutlineCorrect_RUS_0_90
D30_1_20	DistalPhalanxOutlineCorrect_RUS_1_20
D30_1_50	DistalPhalanxOutlineCorrect_RUS_1_50
D30_1_90	DistalPhalanxOutlineCorrect_RUS_1_90
D31	DistalPhalanxTW
D31_0_20	DistalPhalanxTW_RUS_0_20
D31_0_50	DistalPhalanxTW_RUS_0_50
D31_0_90	DistalPhalanxTW_RUS_0_90
D31_1_20	DistalPhalanxTW_RUS_1_20
D31_1_50	DistalPhalanxTW_RUS_1_50
D31_1_90	DistalPhalanxTW_RUS_1_90
D32	DodgerLoopDay
D33	DodgerLoopDayNmv
D32_0_20	DodgerLoopDay_RUS_0_20
D32_0_50	DodgerLoopDay_RUS_0_50
D32_1_20	DodgerLoopDay_RUS_1_20
D32_1_50	DodgerLoopDay_RUS_1_50
D34	DodgerLoopGame
D34_0_20	DodgerLoopGame_RUS_0_20
D34_0_50	DodgerLoopGame_RUS_0_50
D35	DodgerLoopWeekend
D36	DodgerLoopWeekendNmv
D35_0_20	DodgerLoopWeekend_RUS_0_20
D35_0_50	DodgerLoopWeekend_RUS_0_50
D37	ECG200
D37_0_20	ECG200_RUS_0_20
D37_0_50	ECG200_RUS_0_50
D37_0_90	ECG200_RUS_0_90
D37_1_20	ECG200_RUS_1_20
D37_1_50	ECG200_RUS_1_50
D37_1_90	ECG200_RUS_1_90
D38	ECGFiveDays
D38_0_20	ECGFiveDays_RUS_0_20
D38_1_20	ECGFiveDays_RUS_1_20
D38_1_50	ECGFiveDays_RUS_1_50
D39	EOGHorizontalSignal
D39_0_20	EOGHorizontalSignal_RUS_0_20
D39_0_50	EOGHorizontalSignal_RUS_0_50
D39_0_90	EOGHorizontalSignal_RUS_0_90
D39_1_20	EOGHorizontalSignal_RUS_1_20
D39_1_50	EOGHorizontalSignal_RUS_1_50
D39_1_90	EOGHorizontalSignal_RUS_1_90
D40	EOGVerticalSignal
D40_0_20	EOGVerticalSignal_RUS_0_20
D40_0_50	EOGVerticalSignal_RUS_0_50
D40_0_90	EOGVerticalSignal_RUS_0_90
D40_1_20	EOGVerticalSignal_RUS_1_20
D40_1_50	EOGVerticalSignal_RUS_1_50
D40_1_90	EOGVerticalSignal_RUS_1_90
D41	Earthquakes
D41_0_20	Earthquakes_RUS_0_20
D41_0_50	Earthquakes_RUS_0_50
D41_0_90	Earthquakes_RUS_0_90
D41_1_20	Earthquakes_RUS_1_20
D41_1_50	Earthquakes_RUS_1_50
D41_1_90	Earthquakes_RUS_1_90
D42	ElectricDeviceDetection
D43	ElectricDevices
D43_0_20	ElectricDevices_RUS_0_20
D43_0_50	ElectricDevices_RUS_0_50
D43_0_90	ElectricDevices_RUS_0_90
D43_1_20	ElectricDevices_RUS_1_20
D43_1_50	ElectricDevices_RUS_1_50
D43_1_90	ElectricDevices_RUS_1_90
D44	EthanolLevel

D44_0_20	EthanolLevel_RUS_0_20
D44_0_50	EthanolLevel_RUS_0_50
D44_0_90	EthanolLevel_RUS_0_90
D44_1_20	EthanolLevel_RUS_1_20
D44_1_50	EthanolLevel_RUS_1_50
D44_1_90	EthanolLevel_RUS_1_90
D45	FaceAll
D45_0_20	FaceAll_RUS_0_20
D45_0_50	FaceAll_RUS_0_50
D45_0_90	FaceAll_RUS_0_90
D45_1_20	FaceAll_RUS_1_20
D45_1_50	FaceAll_RUS_1_50
D45_1_90	FaceAll_RUS_1_90
D46	FaceFour
D46_0_20	FaceFour_RUS_0_20
D46_0_50	FaceFour_RUS_0_50
D46_1_20	FaceFour_RUS_1_20
D46_1_50	FaceFour_RUS_1_50
D47	FacesUCR
D47_0_20	FacesUCR_RUS_0_20
D47_0_50	FacesUCR_RUS_0_50
D47_1_20	FacesUCR_RUS_1_20
D47_1_50	FacesUCR_RUS_1_50
D48	Fish
D48_0_20	Fish_RUS_0_20
D48_0_50	Fish_RUS_0_50
D48_0_90	Fish_RUS_0_90
D48_1_20	Fish_RUS_1_20
D48_1_50	Fish_RUS_1_50
D48_1_90	Fish_RUS_1_90
D49	FloodModeling1Discrete
D50	FloodModeling2Discrete
D51	FloodModeling3Discrete
D52	FordA
D52_0_20	FordA_RUS_0_20
D52_0_50	FordA_RUS_0_50
D52_0_90	FordA_RUS_0_90
D52_1_20	FordA_RUS_1_20
D52_1_50	FordA_RUS_1_50
D52_1_90	FordA_RUS_1_90
D53	FordB
D53_0_20	FordB_RUS_0_20
D53_0_50	FordB_RUS_0_50
D53_0_90	FordB_RUS_0_90
D53_1_20	FordB_RUS_1_20
D53_1_50	FordB_RUS_1_50
D53_1_90	FordB_RUS_1_90
D54	FreezerRegularTrain
D54_0_20	FreezerRegularTrain_RUS_0_20
D54_0_50	FreezerRegularTrain_RUS_0_50
D54_0_90	FreezerRegularTrain_RUS_0_90
D54_1_20	FreezerRegularTrain_RUS_1_20
D54_1_50	FreezerRegularTrain_RUS_1_50
D54_1_90	FreezerRegularTrain_RUS_1_90
D55	FreezerSmallTrain
D55_0_20	FreezerSmallTrain_RUS_0_20
D55_0_50	FreezerSmallTrain_RUS_0_50
D55_1_20	FreezerSmallTrain_RUS_1_20
D55_1_50	FreezerSmallTrain_RUS_1_50
D56	GestureMidAirD1
D57	GestureMidAirD1Eq
D56_0_20	GestureMidAirD1_RUS_0_20
D56_0_50	GestureMidAirD1_RUS_0_50
D56_1_20	GestureMidAirD1_RUS_1_20
D56_1_50	GestureMidAirD1_RUS_1_50
D58	GestureMidAirD2
D59	GestureMidAirD2Eq
D58_0_20	GestureMidAirD2_RUS_0_20
D58_0_50	GestureMidAirD2_RUS_0_50
D58_1_20	GestureMidAirD2_RUS_1_20
D58_1_50	GestureMidAirD2_RUS_1_50
D60	GestureMidAirD3
D61	GestureMidAirD3Eq
D60_0_20	GestureMidAirD3_RUS_0_20
D60_0_50	GestureMidAirD3_RUS_0_50
D60_1_20	GestureMidAirD3_RUS_1_20
D60_1_50	GestureMidAirD3_RUS_1_50
D62	GesturePebbleZ1
D63	GesturePebbleZ1Eq
D62_0_20	GesturePebbleZ1_RUS_0_20
D62_0_50	GesturePebbleZ1_RUS_0_50
D62_1_20	GesturePebbleZ1_RUS_1_20
D62_1_50	GesturePebbleZ1_RUS_1_50
D64	GesturePebbleZ2
D65	GesturePebbleZ2Eq
D64_0_20	GesturePebbleZ2_RUS_0_20
D64_0_50	GesturePebbleZ2_RUS_0_50
D64_0_90	GesturePebbleZ2_RUS_0_90
D64_1_20	GesturePebbleZ2_RUS_1_20
D64_1_50	GesturePebbleZ2_RUS_1_50
D64_1_90	GesturePebbleZ2_RUS_1_90
D66	GunPoint
D67	GunPointAgeSpan
D67_0_20	GunPointAgeSpan_RUS_0_20
D67_0_50	GunPointAgeSpan_RUS_0_50
D67_0_90	GunPointAgeSpan_RUS_0_90
D67_1_20	GunPointAgeSpan_RUS_1_20
D67_1_50	GunPointAgeSpan_RUS_1_50
D67_1_90	GunPointAgeSpan_RUS_1_90
D68	GunPointMaleVersusFemale
D68_0_20	GunPointMaleVersusFemale_RUS_0_20
D68_0_50	GunPointMaleVersusFemale_RUS_0_50

D68_0_90	GunPointMaleVersusFemale_RUS_0_90
D68_1_20	GunPointMaleVersusFemale_RUS_1_20
D68_1_50	GunPointMaleVersusFemale_RUS_1_50
D68_1_90	GunPointMaleVersusFemale_RUS_1_90
D69	GunPointOldVersusYoung
D69_0_20	GunPointOldVersusYoung_RUS_0_20
D69_0_50	GunPointOldVersusYoung_RUS_0_50
D69_0_90	GunPointOldVersusYoung_RUS_0_90
D69_1_20	GunPointOldVersusYoung_RUS_1_20
D69_1_50	GunPointOldVersusYoung_RUS_1_50
D69_1_90	GunPointOldVersusYoung_RUS_1_90
D66_0_20	GunPoint_RUS_0_20
D66_0_50	GunPoint_RUS_0_50
D66_0_90	GunPoint_RUS_0_90
D66_1_20	GunPoint_RUS_1_20
D66_1_50	GunPoint_RUS_1_50
D66_1_90	GunPoint_RUS_1_90
D70	Ham
D70_0_20	Ham_RUS_0_20
D70_0_50	Ham_RUS_0_50
D70_0_90	Ham_RUS_0_90
D70_1_20	Ham_RUS_1_20
D70_1_50	Ham_RUS_1_50
D70_1_90	Ham_RUS_1_90
D71	HandOutlines
D71_0_20	HandOutlines_RUS_0_20
D71_0_50	HandOutlines_RUS_0_50
D71_0_90	HandOutlines_RUS_0_90
D71_1_20	HandOutlines_RUS_1_20
D71_1_50	HandOutlines_RUS_1_50
D71_1_90	HandOutlines_RUS_1_90
D72	Haptics
D72_0_20	Haptics_RUS_0_20
D72_0_50	Haptics_RUS_0_50
D72_1_20	Haptics_RUS_1_20
D72_1_50	Haptics_RUS_1_50
D72_1_90	Haptics_RUS_1_90
D73	Herring
D73_0_20	Herring_RUS_0_20
D73_0_50	Herring_RUS_0_50
D73_0_90	Herring_RUS_0_90
D73_1_20	Herring_RUS_1_20
D73_1_50	Herring_RUS_1_50
D73_1_90	Herring_RUS_1_90
D74	HouseTwenty
D74_0_20	HouseTwenty_RUS_0_20
D74_0_50	HouseTwenty_RUS_0_50
D74_1_20	HouseTwenty_RUS_1_20
D74_1_50	HouseTwenty_RUS_1_50
D75	InlineSkate
D75_0_20	InlineSkate_RUS_0_20
D75_0_50	InlineSkate_RUS_0_50
D75_1_20	InlineSkate_RUS_1_20
D75_1_50	InlineSkate_RUS_1_50
D76	InsectEPGRegularTrain
D76_0_20	InsectEPGRegularTrain_RUS_0_20
D76_0_50	InsectEPGRegularTrain_RUS_0_50
D76_0_90	InsectEPGRegularTrain_RUS_0_90
D76_1_20	InsectEPGRegularTrain_RUS_1_20
D76_1_50	InsectEPGRegularTrain_RUS_1_50
D76_1_90	InsectEPGRegularTrain_RUS_1_90
D77	InsectEPGSmallTrain
D77_0_20	InsectEPGSmallTrain_RUS_0_20
D77_0_50	InsectEPGSmallTrain_RUS_0_50
D77_1_20	InsectEPGSmallTrain_RUS_1_20
D78	InsectWingbeatSound
D78_0_20	InsectWingbeatSound_RUS_0_20
D78_0_50	InsectWingbeatSound_RUS_0_50
D78_1_20	InsectWingbeatSound_RUS_1_20
D78_1_50	InsectWingbeatSound_RUS_1_50
D79	ItalyPowerDemand
D79_0_20	ItalyPowerDemand_RUS_0_20
D79_0_50	ItalyPowerDemand_RUS_0_50
D79_0_90	ItalyPowerDemand_RUS_0_90
D79_1_20	ItalyPowerDemand_RUS_1_20
D79_1_50	ItalyPowerDemand_RUS_1_50
D79_1_90	ItalyPowerDemand_RUS_1_90
D80	KeplerLightCurves
D81	LargeKitchenAppliances
D81_0_20	LargeKitchenAppliances_RUS_0_20
D81_0_50	LargeKitchenAppliances_RUS_0_50
D81_0_90	LargeKitchenAppliances_RUS_0_90
D81_1_20	LargeKitchenAppliances_RUS_1_20
D81_1_50	LargeKitchenAppliances_RUS_1_50
D81_1_90	LargeKitchenAppliances_RUS_1_90
D82	Lightning2
D82_0_20	Lightning2_RUS_0_20
D82_0_50	Lightning2_RUS_0_50
D82_1_20	Lightning2_RUS_1_20
D82_1_50	Lightning2_RUS_1_50
D82_1_90	Lightning2_RUS_1_90
D83	Lightning7
D83_0_20	Lightning7_RUS_0_20
D83_0_50	Lightning7_RUS_0_50
D83_1_20	Lightning7_RUS_1_20
D83_1_50	Lightning7_RUS_1_50
D84	Meat
D84_0_20	Meat_RUS_0_20
D84_0_50	Meat_RUS_0_50
D84_1_20	Meat_RUS_1_20
D84_1_50	Meat_RUS_1_50
D85	MedicalImages

D85_0_20	MedicalImages_RUS_0_20
D85_0_50	MedicalImages_RUS_0_50
D85_0_90	MedicalImages_RUS_0_90
D85_1_20	MedicalImages_RUS_1_20
D85_1_50	MedicalImages_RUS_1_50
D86	MelbournePedestrian
D87	MelbournePedestrianNmv
D86_0_20	MelbournePedestrian_RUS_0_20
D86_0_50	MelbournePedestrian_RUS_0_50
D86_0_90	MelbournePedestrian_RUS_0_90
D86_1_20	MelbournePedestrian_RUS_1_20
D86_1_50	MelbournePedestrian_RUS_1_50
D86_1_90	MelbournePedestrian_RUS_1_90
D88	MiddlePhalanxOutlineAgeGroup
D88_0_20	MiddlePhalanxOutlineAgeGroup_RUS_0_20
D88_0_50	MiddlePhalanxOutlineAgeGroup_RUS_0_50
D88_0_90	MiddlePhalanxOutlineAgeGroup_RUS_0_90
D88_1_20	MiddlePhalanxOutlineAgeGroup_RUS_1_20
D88_1_50	MiddlePhalanxOutlineAgeGroup_RUS_1_50
D88_1_90	MiddlePhalanxOutlineAgeGroup_RUS_1_90
D89	MiddlePhalanxOutlineCorrect
D89_0_20	MiddlePhalanxOutlineCorrect_RUS_0_20
D89_0_50	MiddlePhalanxOutlineCorrect_RUS_0_50
D89_0_90	MiddlePhalanxOutlineCorrect_RUS_0_90
D89_1_20	MiddlePhalanxOutlineCorrect_RUS_1_20
D89_1_50	MiddlePhalanxOutlineCorrect_RUS_1_50
D89_1_90	MiddlePhalanxOutlineCorrect_RUS_1_90
D90	MiddlePhalanxTW
D90_0_20	MiddlePhalanxTW_RUS_0_20
D90_0_50	MiddlePhalanxTW_RUS_0_50
D90_0_90	MiddlePhalanxTW_RUS_0_90
D90_1_20	MiddlePhalanxTW_RUS_1_20
D90_1_50	MiddlePhalanxTW_RUS_1_50
D90_1_90	MiddlePhalanxTW_RUS_1_90
D91	MixedShapesRegularTrain
D91_0_20	MixedShapesRegularTrain_RUS_0_20
D91_0_50	MixedShapesRegularTrain_RUS_0_50
D91_0_90	MixedShapesRegularTrain_RUS_0_90
D91_1_20	MixedShapesRegularTrain_RUS_1_20
D91_1_50	MixedShapesRegularTrain_RUS_1_50
D91_1_90	MixedShapesRegularTrain_RUS_1_90
D92	MixedShapesSmallTrain
D92_0_20	MixedShapesSmallTrain_RUS_0_20
D92_0_50	MixedShapesSmallTrain_RUS_0_50
D92_1_20	MixedShapesSmallTrain_RUS_1_20
D92_1_50	MixedShapesSmallTrain_RUS_1_50
D93	MoteStrain
D93_1_20	MoteStrain_RUS_1_20
D93_1_50	MoteStrain_RUS_1_50
D94	OSULeaf
D94_0_20	OSULeaf_RUS_0_20
D94_0_50	OSULeaf_RUS_0_50
D94_0_90	OSULeaf_RUS_0_90
D94_1_20	OSULeaf_RUS_1_20
D94_1_50	OSULeaf_RUS_1_50
D94_1_90	OSULeaf_RUS_1_90
D95	OliveOil
D95_0_20	OliveOil_RUS_0_20
D95_1_20	OliveOil_RUS_1_20
D95_1_50	OliveOil_RUS_1_50
D96	PLAIDEq
D97	PhalangesOutlinesCorrect
D97_0_20	PhalangesOutlinesCorrect_RUS_0_20
D97_0_50	PhalangesOutlinesCorrect_RUS_0_50
D97_0_90	PhalangesOutlinesCorrect_RUS_0_90
D97_1_20	PhalangesOutlinesCorrect_RUS_1_20
D97_1_50	PhalangesOutlinesCorrect_RUS_1_50
D97_1_90	PhalangesOutlinesCorrect_RUS_1_90
D98	PickupGestureWiimoteZ
D99	PickupGestureWiimoteZEq
D98_0_20	PickupGestureWiimoteZ_RUS_0_20
D98_1_20	PickupGestureWiimoteZ_RUS_1_20
D98_1_50	PickupGestureWiimoteZ_RUS_1_50
D100	PigAirwayPressure
D101	PigArtPressure
D102	PigCVP
D103	Plane
D103_0_20	Plane_RUS_0_20
D103_0_50	Plane_RUS_0_50
D103_1_20	Plane_RUS_1_20
D103_1_50	Plane_RUS_1_50
D104	PowerCons
D104_0_20	PowerCons_RUS_0_20
D104_0_50	PowerCons_RUS_0_50
D104_0_90	PowerCons_RUS_0_90
D104_1_20	PowerCons_RUS_1_20
D104_1_50	PowerCons_RUS_1_50
D104_1_90	PowerCons_RUS_1_90
D105	ProximalPhalanxOutlineAgeGroup
D105_0_20	ProximalPhalanxOutlineAgeGroup_RUS_0_20
D105_0_50	ProximalPhalanxOutlineAgeGroup_RUS_0_50
D105_0_90	ProximalPhalanxOutlineAgeGroup_RUS_0_90
D105_1_20	ProximalPhalanxOutlineAgeGroup_RUS_1_20
D105_1_50	ProximalPhalanxOutlineAgeGroup_RUS_1_50
D105_1_90	ProximalPhalanxOutlineAgeGroup_RUS_1_90
D106	ProximalPhalanxOutlineCorrect
D106_0_20	ProximalPhalanxOutlineCorrect_RUS_0_20
D106_0_50	ProximalPhalanxOutlineCorrect_RUS_0_50
D106_0_90	ProximalPhalanxOutlineCorrect_RUS_0_90
D106_1_20	ProximalPhalanxOutlineCorrect_RUS_1_20
D106_1_50	ProximalPhalanxOutlineCorrect_RUS_1_50
D106_1_90	ProximalPhalanxOutlineCorrect_RUS_1_90
D107	ProximalPhalanxTW
D107_0_20	ProximalPhalanxTW_RUS_0_20
D107_0_50	ProximalPhalanxTW_RUS_0_50
D107_1_20	ProximalPhalanxTW_RUS_1_20
D107_1_50	ProximalPhalanxTW_RUS_1_50
D107_1_90	ProximalPhalanxTW_RUS_1_90
D108	RefrigerationDevices
D108_0_20	RefrigerationDevices_RUS_0_20
D108_0_50	RefrigerationDevices_RUS_0_50
D108_0_90	RefrigerationDevices_RUS_0_90
D108_1_20	RefrigerationDevices_RUS_1_20
D108_1_50	RefrigerationDevices_RUS_1_50
D108_1_90	RefrigerationDevices_RUS_1_90
D109	Rock
D109_0_20	Rock_RUS_0_20
D109_1_20	Rock_RUS_1_20
D109_1_50	Rock_RUS_1_50
D110	ScreenType
D110_0_20	ScreenType_RUS_0_20
D110_0_50	ScreenType_RUS_0_50
D110_0_90	ScreenType_RUS_0_90
D110_1_20	ScreenType_RUS_1_20
D110_1_50	ScreenType_RUS_1_50
D110_1_90	ScreenType_RUS_1_90
D111	SemgHandGenderCh2
D111_0_20	SemgHandGenderCh2_RUS_0_20
D111_0_50	SemgHandGenderCh2_RUS_0_50
D111_0_90	SemgHandGenderCh2_RUS_0_90
D111_1_20	SemgHandGenderCh2_RUS_1_20
D111_1_50	SemgHandGenderCh2_RUS_1_50
D111_1_90	SemgHandGenderCh2_RUS_1_90
D112	SemgHandMovementCh2
D112_0_20	SemgHandMovementCh2_RUS_0_20
D112_0_50	SemgHandMovementCh2_RUS_0_50
D112_0_90	SemgHandMovementCh2_RUS_0_90
D112_1_20	SemgHandMovementCh2_RUS_1_20
D112_1_50	SemgHandMovementCh2_RUS_1_50
D112_1_90	SemgHandMovementCh2_RUS_1_90
D113	SemgHandSubjectCh2
D113_0_20	SemgHandSubjectCh2_RUS_0_20
D113_0_50	SemgHandSubjectCh2_RUS_0_50
D113_0_90	SemgHandSubjectCh2_RUS_0_90
D113_1_20	SemgHandSubjectCh2_RUS_1_20
D113_1_50	SemgHandSubjectCh2_RUS_1_50
D113_1_90	SemgHandSubjectCh2_RUS_1_90
D114	ShakeGestureWiimoteZ
D115	ShakeGestureWiimoteZEq
D114_0_20	ShakeGestureWiimoteZ_RUS_0_20
D114_1_20	ShakeGestureWiimoteZ_RUS_1_20
D114_1_50	ShakeGestureWiimoteZ_RUS_1_50
D116	ShapeletSim
D116_0_20	ShapeletSim_RUS_0_20
D116_0_50	ShapeletSim_RUS_0_50
D117	ShapesAll
D117_0_20	ShapesAll_RUS_0_20
D117_0_50	ShapesAll_RUS_0_50
D117_1_20	ShapesAll_RUS_1_20
D117_1_50	ShapesAll_RUS_1_50
D118	SmallKitchenAppliances
D118_0_20	SmallKitchenAppliances_RUS_0_20
D118_0_50	SmallKitchenAppliances_RUS_0_50
D118_0_90	SmallKitchenAppliances_RUS_0_90
D118_1_20	SmallKitchenAppliances_RUS_1_20
D118_1_50	SmallKitchenAppliances_RUS_1_50
D118_1_90	SmallKitchenAppliances_RUS_1_90
D119	SmoothSubspace
D119_0_20	SmoothSubspace_RUS_0_20
D119_0_50	SmoothSubspace_RUS_0_50
D119_0_90	SmoothSubspace_RUS_0_90
D119_1_20	SmoothSubspace_RUS_1_20
D119_1_50	SmoothSubspace_RUS_1_50
D119_1_90	SmoothSubspace_RUS_1_90
D120	SonyAIBORobotSurface1
D120_0_20	SonyAIBORobotSurface1_RUS_0_20
D120_0_50	SonyAIBORobotSurface1_RUS_0_50
D120_1_20	SonyAIBORobotSurface1_RUS_1_20
D121	SonyAIBORobotSurface2
D121_0_20	SonyAIBORobotSurface2_RUS_0_20
D121_0_50	SonyAIBORobotSurface2_RUS_0_50
D121_1_20	SonyAIBORobotSurface2_RUS_1_20
D121_1_50	SonyAIBORobotSurface2_RUS_1_50
D122	StarLightCurves
D122_0_20	StarLightCurves_RUS_0_20
D122_0_50	StarLightCurves_RUS_0_50
D122_0_90	StarLightCurves_RUS_0_90
D122_1_20	StarLightCurves_RUS_1_20
D122_1_50	StarLightCurves_RUS_1_50
D122_1_90	StarLightCurves_RUS_1_90
D123	Strawberry
D123_0_20	Strawberry_RUS_0_20
D123_0_50	Strawberry_RUS_0_50
D123_0_90	Strawberry_RUS_0_90
D123_1_20	Strawberry_RUS_1_20
D123_1_50	Strawberry_RUS_1_50
D123_1_90	Strawberry_RUS_1_90
D124	SwedishLeaf
D124_0_20	SwedishLeaf_RUS_0_20
D124_0_50	SwedishLeaf_RUS_0_50
D124_0_90	SwedishLeaf_RUS_0_90
D124_1_20	SwedishLeaf_RUS_1_20
D124_1_50	SwedishLeaf_RUS_1_50
D124_1_90	SwedishLeaf_RUS_1_90

D125	Symbols
D125_0_20	Symbols_RUS_0_20
D125_0_50	Symbols_RUS_0_50
D125_1_20	Symbols_RUS_1_20
D126	SyntheticControl
D126_0_20	SyntheticControl_RUS_0_20
D126_0_50	SyntheticControl_RUS_0_50
D126_0_90	SyntheticControl_RUS_0_90
D126_1_20	SyntheticControl_RUS_1_20
D126_1_50	SyntheticControl_RUS_1_50
D126_1_90	SyntheticControl_RUS_1_90
D127	ToeSegmentation1
D127_0_20	ToeSegmentation1_RUS_0_20
D127_0_50	ToeSegmentation1_RUS_0_50
D127_1_20	ToeSegmentation1_RUS_1_20
D127_1_50	ToeSegmentation1_RUS_1_50
D128	ToeSegmentation2
D128_0_20	ToeSegmentation2_RUS_0_20
D128_0_50	ToeSegmentation2_RUS_0_50
D128_1_20	ToeSegmentation2_RUS_1_20
D128_1_50	ToeSegmentation2_RUS_1_50
D129	Trace
D129_0_20	Trace_RUS_0_20
D129_0_50	Trace_RUS_0_50
D129_0_90	Trace_RUS_0_90
D129_1_20	Trace_RUS_1_20
D129_1_50	Trace_RUS_1_50
D129_1_90	Trace_RUS_1_90
D130	TwoLeadECG
D130_0_20	TwoLeadECG_RUS_0_20
D130_0_50	TwoLeadECG_RUS_0_50
D130_1_20	TwoLeadECG_RUS_1_20
D130_1_50	TwoLeadECG_RUS_1_50
D131	TwoPatterns
D131_0_20	TwoPatterns_RUS_0_20
D131_0_50	TwoPatterns_RUS_0_50
D131_0_90	TwoPatterns_RUS_0_90
D131_1_20	TwoPatterns_RUS_1_20
D131_1_50	TwoPatterns_RUS_1_50
D131_1_90	TwoPatterns_RUS_1_90
D132	UMD
D132_0_20	UMD_RUS_0_20
D132_0_50	UMD_RUS_0_50
D132_1_20	UMD_RUS_1_20
D132_1_50	UMD_RUS_1_50
D133	UWaveGestureLibraryAll
D133_0_20	UWaveGestureLibraryAll_RUS_0_20
D133_0_50	UWaveGestureLibraryAll_RUS_0_50
D133_0_90	UWaveGestureLibraryAll_RUS_0_90
D133_1_20	UWaveGestureLibraryAll_RUS_1_20
D133_1_50	UWaveGestureLibraryAll_RUS_1_50
D133_1_90	UWaveGestureLibraryAll_RUS_1_90
D134	UWaveGestureLibraryX
D134_0_20	UWaveGestureLibraryX_RUS_0_20
D134_0_50	UWaveGestureLibraryX_RUS_0_50
D134_0_90	UWaveGestureLibraryX_RUS_0_90
D134_1_20	UWaveGestureLibraryX_RUS_1_20
D134_1_50	UWaveGestureLibraryX_RUS_1_50
D134_1_90	UWaveGestureLibraryX_RUS_1_90
D135	UWaveGestureLibraryY
D135_0_20	UWaveGestureLibraryY_RUS_0_20
D135_0_50	UWaveGestureLibraryY_RUS_0_50
D135_0_90	UWaveGestureLibraryY_RUS_0_90
D135_1_20	UWaveGestureLibraryY_RUS_1_20
D135_1_50	UWaveGestureLibraryY_RUS_1_50
D135_1_90	UWaveGestureLibraryY_RUS_1_90
D136	UWaveGestureLibraryZ
D136_0_20	UWaveGestureLibraryZ_RUS_0_20
D136_0_50	UWaveGestureLibraryZ_RUS_0_50
D136_0_90	UWaveGestureLibraryZ_RUS_0_90
D136_1_20	UWaveGestureLibraryZ_RUS_1_20
D136_1_50	UWaveGestureLibraryZ_RUS_1_50
D136_1_90	UWaveGestureLibraryZ_RUS_1_90
D137	Wafer
D137_0_20	Wafer_RUS_0_20
D137_0_50	Wafer_RUS_0_50
D137_0_90	Wafer_RUS_0_90
D137_1_20	Wafer_RUS_1_20
D137_1_50	Wafer_RUS_1_50
D137_1_90	Wafer_RUS_1_90
D138	Wine
D138_0_20	Wine_RUS_0_20
D138_0_50	Wine_RUS_0_50
D138_0_90	Wine_RUS_0_90
D138_1_20	Wine_RUS_1_20
D138_1_50	Wine_RUS_1_50
D138_1_90	Wine_RUS_1_90
D139	Worms
D140	WormsTwoClass
D140_0_20	WormsTwoClass_RUS_0_20
D140_0_50	WormsTwoClass_RUS_0_50
D140_0_90	WormsTwoClass_RUS_0_90
D140_1_20	WormsTwoClass_RUS_1_20
D140_1_50	WormsTwoClass_RUS_1_50
D140_1_90	WormsTwoClass_RUS_1_90
D139_0_20	Worms_RUS_0_20
D139_0_50	Worms_RUS_0_50
D139_0_90	Worms_RUS_0_90
D139_1_20	Worms_RUS_1_20
D139_1_50	Worms_RUS_1_50
D139_1_90	Worms_RUS_1_90
D141	Yoga
D141_0_20	Yoga_RUS_0_20
D141_0_50	Yoga_RUS_0_50
D141_0_90	Yoga_RUS_0_90
D141_1_20	Yoga_RUS_1_20
D141_1_50	Yoga_RUS_1_50
D141_1_90	Yoga_RUS_1_90