Mini-Project - ML for Time Series - MVA 2023/2024
Expert Feedback for Anomaly Detection in Time Series

Lucas Haubert lucas.haubert@ens-paris-saclay.fr
Steven Zheng stevenzheng00@gmail.com

February 22, 2024

1 Introduction

1.1 Abstract

Anomaly detection involves identifying data elements within a database that deviate from the norm. Most anomaly detection methods work with datasets that do not have labels. But in certain cases, labels from a portion of the database, possibly derived from expert insights, are accessible and offer valuable guidance for creating the anomaly detector. This paper explores a semi-supervised anomaly detection approach based on support vector machines [1], called "ν-SSVM", and applies it to time series analysis. It examines distinct kernels and parameters to detect anomalies in several uni and multidimensional data sets. The analysis reveals that introducing an expert feedback in an AD task on a time series data set is relevant, and that involving some previous analysis on this specific type of data can improve the performances of the former method.

1.2 The challenge

The purpose of our project is to carry out work parallel to that realized in article [1] : "How to introduce expert feedback in one-class support vector machines for anomaly detection ?" within the framework of the study of time series. The aim of this paper is then to set up a method capable of deciding whether a time series can be considered "normal", using a classification method tested on point clouds in $\mathbb{R}^d$. The results are presented in Section 4.

1.3 Key insights of the mini-project

It is worth pointing out the key points in the organization of this mini-project.

First, the work presented in this research paper is based on the source code [2] for [1], which was used to lay the foundations for our solution and its evaluation. Thus, the methodology presented in Section 2 was already partially implemented, since we extended it in order to carry out our work. Then we estimate that 40% of the code in our work has already been implemented before the mini-project. The difference remains in the adaptation of the method for time series analysis and the evaluation of the influence of the set of parameters.

The work involved in this mini-project is divided into several stages. The first stage involves adapting the method presented in the supporting article to the study of time series, with a view to comparing supervised/unsupervised/semi-supervised learning methods in the context of AD. To do so, a pre-processing step, centered around time series analysis, is settled, in order to translate temporal data into some adequate vectors to feed the AD algorithms. Also, a performance comparison is carried out in order to assign optimal values to the parameters of our method. This makes it possible to extract relevant scores, useful for judging and putting into perspective the method proposed in this paper for time series analysis.

In order to carry out this mini-project in an optimal way, the division of work in the group was as follows. Lucas focused on adapting the anomaly detection method of article [1] to design a method adapted to the study of time series. To achieve this, work was carried out on data pre-processing. On the other hand, Steven focused on evaluating the method, comparing several approaches: with and without pre-processing, with and without adequate feature representation. This allows us to present the results of our work on the last page.

2 Method

SVMs are relevant tools to perform classification for a wide range of type of data. This section summarizes the AD approach developed in [1] in order to introduce its application to time series.

2.1 The ν-SSVM approach

The target can be defined as follows : the user wants to perform AD on a data set made of r labeled data $\{(x_i, y_i)\}_{i=1,\dots,n}$, where the x_i's are vectors in $\mathbb{R}^d$, and the y_i's are equal to -1 for anomalies, and 1 for normal data, and an unlabeled part (vectors x_i only). The idea behind this setting is that AD can be improved with labeled data, annotated by the means of expert feedback. In this context, ν-SSVM provides a relevant solution, using updated SVM methods with few and easy-to-interpret parameters. To do this, the ways of processing the different type of data :

- Labeled data : Use a supervised approach called ν-SVM, which is an extension of default SVM (A.1), that allows some data to violate the mapping constraint. Improvements of this method can be made, in order to avoid overfitting and deal with unlabeled data (which is what we want to inspire from in a semi-supervised setting), that are C-SVM and ν-SVM (A.2 and A.3), the second one bringing more interpretable parameters. The idea of ν-SVM is to introduce a margin ρ and slack variables ξ_i in order to deal with some possible anomalies and give relevant properties to ν.
- Unlabeled data : Use an unsupervised approach called One-Class SVM (OCSVM) (A.4), which considers the data x_i and a feature mapping Φ, and finds the hyperplane furthest from the origin that separates all learning points (minus some possible outliers) (relation with ν in A.4).

In both cases, ν can be seen as an upper bound for the proportion of points x_i that violate the mapping constraint. This remark is a strong motivation for constructing the ν-SSVM method.

$$
\begin{aligned}
\underset{w\in\mathbb{R}^q,\xi\in\mathbb{R}^n,b,\rho_1,\rho_2\in\mathbb{R}}{\arg\min} \quad & \frac{1}{2}\|w\|^2 - r\rho_1 - (n-r)(\rho_2 - b) + \frac{1}{\nu_1}\sum_{i=1}^{r}\xi_i + \frac{1}{\nu_2}\sum_{i=r+1}^{n}\xi_i \\
\text{s.t.} \quad & y_i(w^T\Phi(x_i) + b) \geq \rho_1 - \xi_i, \quad i = 1,\dots,r, \\
& w^T\Phi(x_i) + b \geq \rho_2 - \xi_i, \quad i = r+1,\dots,n, \\
& \rho_1 \geq 0, \rho_2 \geq 0, \xi_i \geq 0, \quad i = 1,\dots,n
\end{aligned}
\tag{ν-SSVM}
$$

Here it is assumed that the bulk of the unlabeled data is normal. Indeed we set $y_i = +1$ for $i > r$ and consider $y \in \mathbb{R}^n$ with the additional values. The approach involves searching for a hyperplane that accommodates occasional misplacements, permitting a certain number of labeled data points to fall on the incorrect side of the boundary and allowing for some of the unlabeled data to be identified as anomalies. In this context, ν_1 can be represented as the trust one can give to the expert feedback, and ν_2 the prior knowledge about the proportion of anomalies in the training set. Note that [1] (Section 3.2.2) proves a boundary property on ν_1, showing that the trust in the expert feedback is controlled by the labeled data, which ensures quality analysis.

This paper focuses on applying this classification method to time series. Now the data x_i's are sequences of data points recorded at successive points in time. Section 2.2 highlights the key points of our method to apply (ν-SSVM) to time series of various shapes / natures, and our way to deepen the analysis of this approach in this new setting.

2.2 Our contributions

Our contribution is in two parts. First, we adapt the method proposed by [1] to the study of time series (2.2.1), which involves a pre-processing phase for the dataset, as well as work on the adequate representation of the time series. Secondly, a calibration and performance evaluation method is proposed (2.2.2) in order to highlight the relevance of the proposed method.

2.2.1 Pre-processing step

The aim of our method is to perform an anomaly detection task on real world data. Consequently, some pre-processing steps are necessary, as in any time series analysis project. After testing the datasets, no interpolation of missing data was necessary. Also, we consider that the presence of some trend in a given time series can be useful to characterize it as an anomaly. For this reason, only denoising and outlier removal steps are considered, as the underlying phenomena is not related to the nature (normal or not) of the sample. To do so, median and average-based approaches are considered to process the entries. Finally, a standardization step (0-mean and 1-standard deviation) is made, in order to feed comparable data to the AD algorithm. (See more details in Appendix B.1).

In a second time, representing the time series in an adequate way may be important, as the semi-supervised SVM is trained and evaluated with vectors in $\mathbb{R}^d$. While time delay embedding method represents the entries as a successive sequence of values, the temporal information is lost when it comes to compute kernels (for example Gaussian kernels that involve $\|x_i - x_j\|_2^2$). It is then relevant to encode the time series in a most clever way. In the case of stationary signals, applying a Discrete Fourier Transform can be relevant, in order to build features vectors. However, dealing with real world data, for example ECGs, is in contradiction with such a stationarity assumption. For this reason, Discrete Wavelet Transform method is used, with Daubechies wavelets, in order to encode the instances in the whole dataset (see Appendix B.2).

2.2.2 Parametrization and evaluation of the method

The hyperparameters for the model are as follows:

- ν_1: Lower and upper bound for the fraction of support vectors and misclassified labeled data, respectively, indicating trust in labels with usual bounds between 0.01 and 0.05 for up to $1-5\%$ label errors.
- ν_2: Lower bound for the fraction of support vectors among unlabeled data, representing prior knowledge of anomalies, varying across datasets.
- γ: RBF kernel parameter in SVM, controlling influence range of training examples, defined by $\gamma = \frac{1}{2\sigma^2}$, affects decision boundary smoothness and overfitting risk.

We used 50% of the labels, a gridsearch was performed on these parameters focusing on balanced accuracy which takes into account the inbalanced betweend normal and anormal data and we compared to the settings as set in the articles. The results are based on the parameters found in the gridsearch.

We also implemented polynomial and sigmoid kernels. However it did not produced good results.

3 Data

The aim of this paper is to perform an anomaly detection task on time series, i.e. to determine whether an whole time series is considered normal or not. In order to carry out such a task, and to evaluate the performance of the underlying process, a wide range of time series data sets need to be selected. To do this, several characteristics of the data must be considered :

- First, it is capital that the data be labeled, so that we can implement our semi-supervision method. It will then be possible to select which data are labelled as well as their proportions.
- Next, the data must be classified into two classes, which explicitly represent the "normal" time series, and the others that need to be detected.
- Also, the time series studied must be sufficiently different, i.e. it is relevant to consider one-dimensional as well as multi-dimensional time series representing signals of different natures (ECG, spectro, motion, etc.) and length.

A large number of datasets, covering all these criteria, are available on the website Time Series Classification [3].

The PhysioNet/CinC Challenge 2016 dataset contains heart sound recordings from healthy individuals and patients with cardiac conditions, classified into normal and abnormal categories. It includes 113 normal and 296 abnormal five-second sound clips, analyzed through spectrograms.

R. Olszewski formatted the dataset for his 2001 thesis at Carnegie Mellon University, comprising electrical activity from heartbeats categorized into normal or indicative of Myocardial Infarction. Anomaly detection models help in identifying the latter for timely medical response.

FTIR spectroscopy with ATR sampling is used in food spectrographs to distinguish authentic strawberries from adulterated or different fruits, aiding in food safety and quality assurance.

The ToeSegmentation dataset from CMU categorizes walking motions as normal or abnormal (simulated impairments) based on motion descriptions. It includes normalized X-axis (ToeSegmentation1) and Y-axis (ToeSegmentation2) motion data for time series analysis.

Dataset	Train Size	Test Size	Length	No Classes	No Dimensions	Type
Heartbeat	204	205	405	2	61	AUDIO
ECG200	100	100	96	2	1	ECG
Strawberry	613	370	235	2	1	SPECTRO
ToeSegmentation2	36	130	343	2	1	MOTION

Table 1: Model Performance Metrics

A visualization of this data is provided in the "Pre_processing" notebook. It is then possible to display the original signal, as well as the pre-processed. We can visually see that applying a denoising window that is too wide tends to significantly smooth our time series, and therefore lose local information that may be relevant in a classification situation. A decent value for such a rolling window is 3.

Also, Heartbeat being a dataset of multivariate time series, it will be processed in an similar way, ie the denoising / outlier removal / discrete wavelet transform steps are made on the univariate time series, that are then concatenated.

4 Results

The approach used to evaluate our method is as follows: First, we consider our time series in a naive way. In other words, we give the ν-SSVM algorithm the values of our time series in the form of a vector of series values. We then present the results of our adaptation of the algorithm.

The evaluation of the naive algorithm is given in Appendix C.1. Note that we are mainly focusing on the balances accuracy metric, as we are performing a classification task on very different classes, in terms of number of elements they contain.

Tables [10] [11] [4] [9] provide the results for the best performance of four machine learning models nuSSVM, OCSVM, NuSVM, and S3VM across various datasets: Heartbeat, ECG, Strawberry, and Toe, with specific preprocessing techniques like db1-DWT for Heartbeat and ECG, and denoising plus outlier removal for the Toe dataset.

- Heartbeat Dataset with db1-DWT: nuSSVM achieves a balanced accuracy of **0.684927**, indicating a good ability to differentiate between normal and abnormal heartbeats.
- ECG Dataset with db1-DWT: Here, the model performs better, with a balanced accuracy of **0.734654**, showing improved efficiency in classifying ECG patterns.
- Strawberry Dataset: nuSSVM's performance drops to a balanced accuracy of **0.560203**, suggesting challenges in handling this particular type of data.
- Toe Dataset with Denoising + Outlier Removal: The balanced accuracy improves to **0.637481**, indicating that preprocessing techniques like denoising and outlier removal enhance the model's classification ability. Notice that the balanced accuracy for the db1-DWT configuration is also close to this one, making the discrete wavelet transform not absurd in that case.

Overall, nuSSVM shows adaptability in semi-supervised settings with partial label access, excelling particularly in medical datasets, but with variability in its precision and recall across different data types.

Now it is relevant to provide some qualitative comments about these results. Our analyses show the relevance of using a wavelet transform. This can be understood as follows: a time series is both a conglomeration of signal record values, and an ordered sequence of elements. It is therefore important to take into account the temporal aspect of our data, in order to use classification techniques created for working on point clouds. This was the key element of the adaptive approach of SVM based-methods.

The performance of the ν-SSVM model across different datasets highlights the challenges of applying SVM techniques to time series data. For example, a balance accuracy of around 0.5 for the Strawberry dataset, in the best possible configuration in terms of pre-processing, indicates the difficulty of an SVM-based algorithm in capturing key information about sequential data, which nevertheless justifies our recourse to feature representation methods (as DWT).

References

[1] Cédric Baudoin Marc Spigai Julien Lesouple and Jean-Yves Tourneret. How to introduce expert feedback in one-class support vector machines for anomaly detection ?, in *Signal Processing*, pp. 108197. 2021.

[2] Laboratoire coopératif TéSA. nu-ssvm. https://www.tesa.prd.fr/logiciels/, 2021.

[3] Time series classification. https://www.timeseriesclassification.com/dataset.php.

A Building the algorithm ν-SSVM

A.1 SVM

SVM consists in considering the labeled dataset $\{(x_i, y_i)\}_{i=1,...,n}$ and find a feature mapping $\Phi : \mathbb{R}^d \to \mathbb{R}^q$, with $q > d$ such that the data set $\{\Phi(x_i)\}_{i=1,...,n}$ is linearly separable (mapping constraint). The corresponding optimization problem is :

$$\begin{array}{ll} \underset{w \in \mathbb{R}^q, b \in \mathbb{R}}{\arg\min} & \frac{1}{2}\|w\|^2 \\ \text{s.t.} & y_i(w^T\Phi(x_i) + b) \geq 1, \quad i = 1, \dots, n \end{array}$$

Solving such a problem is often done by means of kernel tricks, that is considering a function $k : (x_i, x_j) \mapsto \Phi(x_i)^T\Phi(x_j)$ and use support vectors $x_i \in \mathcal{S}_v$ to build the optimal argument w, such that the decision function becomes $x \mapsto f(x) = sign(\sum_{x_i \in \mathcal{S}_v} \alpha_i y_i k(x_i, x) + b)$ for some $\alpha > 0$. One way of analyzing and criticizing the algorithm is to use different kernels, the most classical one being the Gaussian Kernel, defined by :

$$k(x_i, x_j) = \exp(-\frac{1}{2\sigma^2}\|x_i - x_j\|_2^2)$$

This kernel is easy to parameterize (only σ or $\gamma = 1/2\sigma^2$) (Nota bene, one can recall that $\|x_i - x_j\|_2^2$ in fact represents the (squared) euclidean distance between the two time series. Since the time series have the same length and timeline within the same dataset, implementing other "distances" such as the "Dynamic Time Warping" would not be useful).

A.2 C-SVM

In order to allow some variables to get out of their supposed classification zone (for constructing an appropriate anomaly detection setting within the S3VMAD method), C-SVM considers the data set $\{(x_i, y_i)\}_{i=1,...,n}$, as well as some slack variables ξ_i and replace the mapping constraint of SVM by :

$$y_i(w^T\Phi(x_i) + b) \geq 1 - \xi_i, \quad i = 1, \dots, n$$

This leads to the optimization problem :

$$\begin{array}{ll} \underset{w \in \mathbb{R}^q, \xi \in \mathbb{R}^n, b \in \mathbb{R}}{\arg\min} & \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i \\ \text{s.t.} & y_i(w^T\Phi(x_i) + b) \geq 1 - \xi_i, \quad 1 \leq i \leq n, \\ & \xi_i \geq 0, \quad 1 \leq i \leq n \end{array}$$

A.3 ν-SVM

ν-SVM is very close from the previous one. In fact, it is possible to add some new parameter ρ, defined as the margin of the dividing hyperplane, in order to give some interpretation for ν (in fact ν_1 and ν_2) in our semi-supervised AD method. This is translated as :

$$\begin{array}{ll} \underset{w \in \mathbb{R}^q, \xi \in \mathbb{R}^n, \rho \in \mathbb{R}, b \in \mathbb{R}}{\arg\min} & \frac{1}{2}\|w\|^2 - \nu\rho + \frac{1}{n}\sum_{i=1}^{n}\xi_i \\ \text{s.t.} & y_i(w^T\Phi(x_i) + b) \geq \rho - \xi_i, \quad 1 \leq i \leq n, \\ & \xi_i \geq 0, \quad 1 \leq i \leq n, \\ & \rho \geq 0 \end{array}$$

A.4 OCSVM

C-SVM and ν-SVM are supervised method in order to perform classification. On the other hand, according to the provided semi-labeled data set setting, one must consider a method to process the unlabeled data. The one used in the One-Class SVM.

The key point is to introduce slack variables ξ_i, as well as in C-SVM and ν-SVM, in order to build the further semi-supervised method. As stated in Section 2, this approach considers the data x_i and a feature mapping Φ, and finds the hyperplane furthest from the origin that separates all learning points (minus some possible outliers). This is equivalent to the following :

$$\begin{array}{ll} \underset{w\in\mathbb{R}^q,\xi\in\mathbb{R}^n,\rho\in\mathbb{R}}{\arg\min} & \frac{1}{2}\|w\|^2 - \rho + \frac{1}{\nu n}\sum_{i=1}^{n}\xi_i \\ \text{s.t.} & w^T\Phi(x_i) \geq \rho - \xi_i, \quad 1 \leq i \leq n, \\ & \xi_i \geq 0, \quad 1 \leq i \leq n. \end{array}$$

A.5 S3VMAD

The idea of S3VMAD is to perform a trade-off between C-SVM for processing the labeled data, and OCSVM for the unlabeled ones. To do so, we set $y_i = +1$ for $i > r$ and consider $y \in \mathbb{R}^n$ with the additional values. Then, the underlaying optimization problem is given by :

$$\begin{array}{ll} \underset{w\in\mathbb{R}^q,\xi\in\mathbb{R}^n,b\in\mathbb{R}^2}{\arg\min} & \frac{1}{2}\|w\|^2 + \frac{C_0(n-r)}{n-r+1}b + \frac{1}{C_2(r+1)}\sum_{i=1}^{r}\xi_i + \frac{1}{C_1(n-r+1)}\sum_{i=r+1}^{n}\xi_i \\ \text{s.t.} & y_i(w^T\Phi(x_i)+b) \geq 1-\xi_i, \quad i = 1,\dots,r, \\ & w^T\Phi(x_i)+b \geq -\xi_i, \quad i = r+1,\dots,n, \\ & \xi_i \geq 0, \quad i = 1,\dots,n. \end{array}$$

Now why not to use this approach, instead of ν-SSVM ? Article [1] first claims that the parameters C_i are not easy-to-interpret. This is in fact due to the fact that the multiplying factors (penalization errors) in front of the $\sum \xi_i$ in S3VMAD and ν-SSVM are not the same, since S3VMAD involves parameters C_i, but also n and r, that represent the number of data / labeled data. In that sense, parameters C_i suffer from scale sensitivity, making them less easy-to-interpret that ν_1 and ν_2 in ν-SSVM, when we are dealing with different type of time series / expert contribution. Also, the shape of the objective function enforces a margin equal to 1 for labeled data, and to 0 for unlabeled data, which brings an unnecessary constraint about the dividing hyperplane, which is not desirable in an AD task. These are the two main reasons for why [1] focuses on ν-SVM.

B Pre-processing step

B.1 General steps

First of all, it is important that we only perform pre-processing, and not post-processing at the output of our algorithm, since this is an AD method, and the objective is therefore to return a classification value. On the other hand, a pre-processing phase is necessary, to enable the method studied to deliver relevant results.

The main core of the pre-processing task is : outlier removal, denoising and discrete wavelet transformation for feature representation.

The outlier removal step focuses on single points considers as outliers. In order to detect them and remove them, a median approach technique, based on a rolling window is used, reconstruct a more adequate time series. In particular, the transformation is as follows :

$$\hat{x}[n] = median_{-w \leq i \leq +w} x[n-i]$$

where $w = 1$, in order to capture local outlier events.

Then, a denoising step is proposed, based on an average rolling window, with similar definition :

$$\hat{x}[n] = average_{-w \leq i \leq +w} x[n-i]$$

where w is variable, in order to control the trade-off between denoising and information retrieval.

These two processing steps give access to smooth time series data. However, it is important to parameterize the windows in an adequate way, in order to keep relevant information contained in the time series.

Finally, we can specify the need for a standardization step for the input vectors of the algorithm under study. Standardizing (setting the time series to 0-mean and 1-standard deviation) is relevant in the context of SVM algorithm, because the entry vector with the largest variance could dominate the decision criteria, while the other ones might be ignored. This is not what we can expect from an AD method on time series, which should focus on the intrinsic shape of the sequential data. In particular, the use of kernel methods, as the Gaussian one, is relevant with normalized data.

B.2 Discrete Wavelet Transform

The invocation of Discrete Wavelet Transform (DWT) is relevant, in the context of real world time series. Indeed, the idea is to capture some relevant time-frequency features and store them into an adequate vector representation. To do so, and in order to face the variety of time series to study, we chose to work with Daubechies wavelets, in order to represent the entry time series, with a large range of orders (db1, db2, etc). In particular, for capturing features in time series with short and high-frequency events, it is relevant to use wavelets that provide a good time resolution. On the other hand, smoother signal are better represented with wavelets that provide high frequency resolution. Among Daubechies wavelets, the ones of low order (db1, db2) are adapted to the first class of signal described before, while higher orders are adapted to smoother signals.

C About the results

First of all, it is important that we only perform pre-processing, and not post-processing at the output of our algorithm, since this is an AD method, and the objective is therefore to return a classification value. On the other hand, a pre-processing phase is supposed to be necessary, to enable the method studied to deliver relevant results. Also, Discrete Wavelets Transform is an important element of our contribution. This appendix gives the results of our method in different setting, described by the sub-sections.

C.1 Without Pre-Processeing

Model	Accuracy	Balanced Acc	Precision	Recall	AUC	FPR
nuSSVM	0.602941	0.552632	0.753846	0.666667	0.672992	0.561404
OCSVM	0.323529	0.423201	0.591837	0.197279	0.454470	0.350877
NuSVM	0.833333	0.793054	0.884354	0.884354	0.803437	0.298246
S3VM	0.318627	0.414429	0.580000	0.197279	0.476429	0.368421

Table 2: Model Performance on Heartbeat data set

Model	Accuracy	Balanced Acc	Precision	Recall	AUC	FPR
nuSSVM	0.770	0.734485	0.666667	0.626866	0.721692	0.157895
OCSVM	0.740	0.671193	0.659574	0.462687	0.684547	0.120301
NuSVM	0.810	0.786780	0.716418	0.716418	0.788464	0.142857
S3VM	0.705	0.570811	0.785714	0.164179	0.696667	0.022556

Table 3: Model Performance on ECG data set

Model	Accuracy	Balanced Acc	Precision	Recall	AUC	FPR
nuSSVM	0.581892	0.560203	0.425000	0.484330	0.627094	0.363924
OCSVM	0.440488	0.353964	0.076596	0.051282	0.252022	0.343354
NuSVM	0.659207	0.653265	0.518692	0.632479	0.732223	0.325949
S3VM	0.571719	0.455387	0.163462	0.048433	0.609098	0.137658

Table 4: Model Performance on Strawberry data set

Model	Accuracy	Balanced Acc	Precision	Recall	AUC	FPR
nuSSVM	0.771084	0.579109	0.666667	0.190476	0.494048	0.032258
OCSVM	0.632530	0.509985	0.268293	0.261905	0.385945	0.241935
NuSVM	0.879518	0.840630	0.761905	0.761905	0.822389	0.080645
S3VM	0.759036	0.555300	0.600000	0.142857	0.436444	0.032258

Table 5: Model Performance on Toe data set

C.2 With denoising and outlier removal

The idea of this part is to analyse time series in a smoother way. To do so, a denoising method was applied, with as window equal to 3. This is due to the input time series, that we provided with prior regularity and smoothness. Also, a rolling median approach (window equal to 3) was used in order to detect and remove the outlier points.

Model	Accuracy	Balanced Acc	Precision	Recall	AUC	FPR
nuSSVM	0.509804	0.627641	0.898305	0.360544	0.743884	0.105263
OCSVM	0.372549	0.484067	0.693878	0.231293	0.536341	0.263158
NuSVM	0.833333	0.793054	0.884354	0.884354	0.842583	0.298246
S3VM	0.357843	0.468493	0.666667	0.217687	0.558659	0.280702

Table 6: Model Performance on Heartbeat data set with denosing

Model	Accuracy	Balanced Acc	Precision	Recall	AUC	FPR
nuSSVM	0.715	0.622770	0.638889	0.343284	0.540792	0.097744
OCSVM	0.660	0.581416	0.489362	0.343284	0.539333	0.180451
NuSVM	0.810	0.786780	0.716418	0.716418	0.791494	0.142857
S3VM	0.695	0.570699	0.650000	0.194030	0.550892	0.052632

Table 7: Model Performance on ECG data set with denoising

Model	Accuracy	Balanced Acc	Precision	Recall	AUC	FPR
nuSSVM	0.578840	0.544529	0.412742	0.424501	0.620650	0.335443
OCSVM	0.421160	0.333232	0.038136	0.025641	0.207869	0.359177
NuSVM	0.604273	0.564941	0.443787	0.427350	0.671819	0.297468
S3VM	0.583927	0.459181	0.108108	0.022792	0.607951	0.104430

Table 8: Model Performance on Strawberry data set with denoising

Model	Accuracy	Balanced Acc	Precision	Recall	AUC	FPR
nuSSVM	0.728916	0.637481	0.463415	0.452381	0.689132	0.177419
OCSVM	0.656627	0.541859	0.317073	0.309524	0.559908	0.225806
NuSVM	0.849398	0.812596	0.688889	0.738095	0.819892	0.112903
S3VM	0.728916	0.574501	0.440000	0.261905	0.648041	0.112903

Table 9: Model Performance on Toe data set with denoising + outlier removal

C.3 With Discrete Wavelet Transformation

As justified in Section 4, it is relevant to apply a discrete wavelet transform step at the beginning of the approach. The results are here presented for each datasets, with the best choice of wavelet (in terms of balanced accuracy).

Model	Accuracy	Balanced Acc	Precision	Recall	AUC	FPR
nuSSVM	0.607843	0.684927	0.903614	0.510204	0.728249	0.140351
OCSVM	0.323529	0.423201	0.591837	0.197279	0.466404	0.350877
NuSVM	0.833333	0.793054	0.884354	0.884354	0.755221	0.298246
S3VM	0.387255	0.547977	0.843750	0.183673	0.527390	0.087719

Table 10: Model Performance on Heartbeat data set with db1-DWT

Model	Accuracy	Balanced Acc	Precision	Recall	AUC	FPR
nuSSVM	0.785	0.734654	0.722222	0.582090	0.737515	0.112782
OCSVM	0.740	0.671193	0.659574	0.462687	0.674223	0.120301
NuSVM	0.810	0.786780	0.716418	0.716418	0.822803	0.142857
S3VM	0.715	0.578330	0.916667	0.164179	0.711929	0.007519

Table 11: Model Performance on ECG data set with db1-DWT

Model	Accuracy	Balanced Acc	Precision	Recall	AUC	FPR
nuSSVM	0.578840	0.546429	0.414169	0.433048	0.641700	0.340190
OCSVM	0.437436	0.350957	0.072034	0.048433	0.251972	0.346519
NuSVM	0.695829	0.704547	0.556034	0.735043	0.757774	0.325949
S3VM	0.577823	0.460134	0.173469	0.048433	0.625760	0.128165

Table 12: Model Performance on Strawberry data set with db2-DWT

Model	Accuracy	Balanced Acc	Precision	Recall	AUC	FPR
nuSSVM	0.740964	0.566820	0.473684	0.214286	0.493088	0.080645
OCSVM	0.620482	0.494048	0.243902	0.238095	0.368280	0.250000
NuSVM	0.879518	0.840630	0.761905	0.761905	0.852151	0.080645
S3VM	0.759036	0.547427	0.625000	0.119048	0.433756	0.024194

Table 13: Model Performance on Toe data set with db2-DWT