A One-Class Classification Decision Tree Based on Kernel Density Estimation

Sarah Itani[a,b,*], Fabian Lecron[c], Philippe Fortemps[c]

[a] *Fund for Scientific Research - FNRS (F.R.S.-FNRS), Brussels, Belgium*
[b] *Faculty of Engineering, University of Mons, Department of Mathematics and Operations Research, Mons, Belgium*
[c] *Faculty of Engineering, University of Mons, Department of Engineering Innovation Management, Mons, Belgium*

Abstract

One-class Classification (OCC) is an important field of machine learning which aims at predicting a single class on the basis of its lonely representatives and potentially some additional counter-examples. OCC is thus opposed to traditional classification problems involving two or more classes, and addresses the issue of class unbalance. There is a wide range of one-class models which give satisfaction in terms of performance. But at the time of explainable artificial intelligence, there is an increasing need for interpretable models. The present work advocates a novel one-class model which tackles this challenge. Within a greedy and recursive approach, our proposal for an explainable One-Class decision Tree (OC-Tree) rests on kernel density estimation to split a data subset on the basis of one or several intervals of interest. Thus, the OC-Tree encloses data within hyper-rectangles of interest which can be described by a set of rules. Against state-of-the-art methods such as Cluster Support Vector Data Description (ClusterSVDD), One-Class Support Vector Machine (OCSVM) and isolation Forest (iForest), the OC-Tree performs favorably on a range of benchmark datasets. Furthermore, we propose a real medical application for which the OC-Tree has demonstrated effectiveness, through the ability to tackle inter-

*Corresponding author. University of Mons, Department of Mathematics and Operations Research, Rue de Houdain, 9, 7000 Mons, Belgium.

Email addresses: sarah.itani@umons.ac.be (Sarah Itani), fabian.lecron@umons.ac.be (Fabian Lecron), philippe.fortemps@umons.ac.be (Philippe Fortemps)

pretable medical diagnosis aid based on unbalanced datasets.

Keywords: One-class classification, decision trees, kernel density estimation, explainable artificial intelligence

---

1. Introduction

As precious assets of knowledge extraction, data are massively collected in the fields of industry and research, day by day. Though valuable, the proliferation of data requires attention upon processing. In particular, unbalanced datasets may be hardly addressed through the classical scheme of multi-class prediction. The practice of One-Class Classification (OCC) has been developed within this consideration [1, 2].

Basically, OCC is a problem of **data description** [1]. One-class models are thus trained on the representatives of a given class, in the possible presence of a few counter-examples. Prediction capabilities stem from this description which allows (i) to identify the representatives of the class, i.e., *target* (or *positive*) instances, and (ii) to reject those which do not comply with the description, i.e., *outlier* (or *negative*) instances.

OCC is of major concern in several domains where it may be expensive and/or technically difficult to collect data on a range of behaviors or phenomenons [3]. For example, it may be quite affordable to gather data on the representatives of a given pathology in medicine, or positive operating scenarios of machines in the industry. The related complementary occurrences are, by contrast, scarce and/or expensive to raise [2]. In this respect, **anomaly detection** is one of the most reported applications of OCC: it relates to the detection of events which do not occur frequently. OCC is particularly convenient in this context since the related datasets are usually unbalanced. Also addressed through OCC, **novelty detection** [4] aims to identify new items that have not been observed yet in the study of a phenomenon. In this case, novelties are no more singularities once they are detected: they may be introduced in the initial training set to adjust a given One-Class (OC) model [4].

Beyond the ability to address class unbalance, OCC provides capabilities to **tackle complex data structures**. These consist of difficult, although possibly balanced, datasets [5, 6]. The work of [6] illustrated the interest of OCC in this specific context, for the classification of Autism Spectrum Disorder (ASD) and control (i.e., healthy) subjects. In particular, the study showed that the healthy population is characterized by a significant form of heterogeneity, which is not sufficiently represented by the control sample at hand. Here, transforming the binary classification problem into an OCC task appeared to be convenient. In addition to interesting accuracy, the description of ASD derived from OCC allowed to identify brain patterns related to the neuropathology. In another context, it was shown that **Multi-Class Classification** (MCC) problems may be advantageously converted **into ensembles of OC classifiers** (each describing a class) to handle datasets involving a large number of classes and/or presenting some complexity in data distribution [7].

Though the range of OC models is relatively broad, there is still room for improvement, at least for the two main reasons detailed below.

- Many OCC tasks are patterned on the principles of MCC. Indeed, the literature covers approaches to OCC which are grounded on the artificial generation of negative instances in order to recover a supervised classification mechanism (e.g. [8, 9, 10, 11, 12, 13]). Such a mechanism of transposition can set aside the specific nature of OCC. Indeed, if we refer to the rigorous definition of OCC, the classification task should be oriented towards the *isolation* of the target instances. By contrast, MCC aims at the *separation* of the training instances. Yet isolation and separation are different goals. We also note that some OC methods rely on unsupervised classification approaches which are not properly intended for OCC. For example, clustering-based approaches may fail in the detection of outliers forming clusters [3]. Hence, it is worth breaking away from the usual classification schemes in the design of OC classifiers. This would probably yield models which are more performant, and which fulfill their intended

purpose.

- The advent of explainable Artificial Intelligence (xAI) opens new research horizons for machine learning in encouraging the development of interpretable models [14]. In this respect, the attempts mainly rely on the development of post-hoc systems, made by the combination of a black box and an additional component providing explanations on demand [15]. Interpretable neural networks have been built around this principle [16, 17]. However, these systems do not give a comprehensive picture of the model behavior, and doubts have been expressed on the veracity of their explanations [15]. Here, a great and modern challenge remains the development of models which are inherently interpretable. As highlighted in [15], the design of optimal logical models (e.g., tree-based classifiers) is a research avenue worth considering in this direction.

The present work aims to cope with the above challenges, through our proposal for an interpretable one-class model, called One-Class decision Tree (OC-Tree). Within a greedy and recursive approach, the OC-Tree rests on Kernel Density Estimation (KDE) to split a data subset on the basis of one or several intervals of interest. Thus, the OC-Tree encloses data within hyper-rectangles of interest which can be described by a set of rules. The contributions of our work are exposed below.

(1) Compared to previous adaptations of the decision tree to OCC, our proposal rather focuses on the isolation of the target training instances through a density-based hierarchical process of splitting, in which subdivisions are based on closed intervals of interest.

(2) The model shows favorable performances in comparison to reference methods. The OC-Tree can thus be seen as an interesting interpretable and performant alternative to perform OCC, which complies with the requirements of the modern field of explainable artificial intelligence.

(3) We bring contributions to a real-world application, in applying the OC-

Tree to the diagnosis of Attention Deficit Hyperactivity Disorder (ADHD). Given the recent literature, our model achieved competitive accuracy on the open ADHD-200 collection [18].

The remainder of the paper is organized as follows. Sec. 2 provides a review of the related literature. Sec. 3 describes our algorithm which was assessed in comparison to reference methods according to the experimental protocol presented in Sec. 4. We expose the results in Sec. 5. Then, in Sec. 6, we present a medical case study whose challenging aspects can be appropriately addressed by the OC-Tree. Finally, we discuss and summarize our findings in Sec. 7, before concluding the paper in Sec. 8.

2. Literature review

We present in the following paragraphs a non-exhaustive synthesis of models for OCC listed by category [9, 19, 20, 21, 3].

Boundary-based methods. Boundary-based methods enclose target data within a decision boundary which optimizes a given loss function. The most popular methods in this field are One-Class Support Vector Machine (OCSVM) and Support Vector Data Description (SVDD) [22, 23]. OCSVM aims at finding the hyper-plane that separates the target instances from the origin with the wider margin, while SVDD aims at enclosing these instances within a single hyper-sphere of minimal volume. Far from being contested, the effectiveness of these methods has notably been improved for faster execution [24], increased robustness to noise [25, 26, 27] and optimal hyperparameter selection [20]. Some extensions fit more complex data structures in which the representatives of a class are spread over different groupings in the form of *sub-concepts* that it would be interesting to raise separately [28, 29, 30, 31]. ClusterSVDD [32] achieves such a purpose: this recent method may be seen as a K-means algorithm [33] ruled by the results of distinct SVDD problems.

Distance-based methods. The distance between a given data point and its closest neighbors is a key criterion for distance-based OCC methods. The Local Outlier Factor (LOF) is a common measure in this respect [34, 35, 36, 37]. LOF is computed as the division of the averaged local density of the K nearest neighbors by the local density of the examined instance. Density-Based Spatial Clustering of Applications with Noise (DBSCAN) and its extensions [38, 39, 40, 41] are also part of reference methods for OCC, and more particularly for anomaly detection. DBSCAN identifies core, border and noise points, based on the number and type of neighbors that they have within a given radius. Actually, DBSCAN defines the nature of an instance (inlier/outlier) through its membership to a cluster. K-means rather relies on the distance between an instance and its closest cluster centroid to achieve OCC [42]. A pertinent K-means approach [43] distinguishes *external outliers* which potentially form small and far clusters and *internal outliers* which are located far from their cluster centroid.

Density-based methods. These methods estimate the distribution of the target class. Thresholded at a given level of confidence, this estimation is used to reject any outlier located beyond the decision boundary thus defined. The Gaussian Mixture Model (GMM) and Kernel Density Estimator (KDE) are the most common approaches in this respect. GMMs estimate data distributions based on a limited number (generally inferior to the size of the training set) of Gaussian kernels [44]. KDEs, also called as Parzen estimators, are seen as more accurate ways for the non-parametric estimation of a sample distribution [45]. But KDE loses in performance and readability towards high dimensional samples [12].

Tree-based methods. This category includes decision trees for OCC, which are valued for their interpretability. The methods for tree-based models often rely on the generation of outliers [11, 12]. Hence, these algorithms may associate the target class with a large subspace against the effective one. Indeed, a decision tree is basically built under the hypothesis that the different classes cover the whole domain by their representatives. In a different perspective, the work

of [46] revisits the development of decision trees by orienting the training process towards the isolation of outliers rather than of target instances. The intuition behind the method, called Isolation Forest (iForest), is that outliers are scarce and easily detectable compared to target instances [46]. The outliers can thus be isolated by means of a low number of divisions. An iForest is an ensemble of trees built based on a random choice of attributes and thresholds. For a given instance, if the average path skimmed in the trees is short, the instance is predicted as outlier.

Neural network-based methods. This category includes the extensions of neural networks for OCC, which present quite complex structures and thus tend to be qualified as black boxes [15]. Recently proposed in [47], Deep SVDD is a neural network which implements the SVDD principle, i.e., the transformation of the target instances into a space where they are grouped within a hypersphere of minimal volume. As compared to SVDD or OCSVM, Deep SVDD is advantageously exempted from the definition of any kernel and the resulting necessity to appropriately tune its parameter(s). Moreover, kernel manipulations are computationally expensive [47]. That being said, the Deep SVDD objective function is non-convex, which entails a more complex algorithmic approach for optimization. A similar reasoning was applied to develop one-class neural networks inspired by OCSVM in [48], which were applied successfully for anomaly detection. Other approaches include the generation of artificial negative instances in order to restore a binary classification mode [8, 10].

Ensembling techniques. Ensembling relates to a more sophisticated way of practising OCC, based on models selected from some of the categories listed below. Indeed, certain applications require ensemble approaches to perform effective OCC, i.e., a set of OC classifiers describing the same class, whose decisions need to be reconcilied on a given rule. Here, the way of aggregating the classification outputs over the ensemble requires particular attention for accurate decisions. In this respect, one of the most common strategies is the majority vote rule. A recent work [19] proposed a more flexible strategy

through the computation of a normality score, associated with a cut-off value estimated from the available data. This threshold can be revised on the basis of new incoming data without the need to train new models. The metholodolgy was successfully tested on an ensemble including common OCC models such as OCSVM, LOF and iForest.

The method proposed in the present paper, namely the OC-Tree, tackles OCC through a hybrid methodology where density estimation is considered as part of decision tree induction. The method is intended to combine the benefits of the interpretable decision tree and the intuitive approach to OCC proposed by KDE. Actually, the OC-Tree may be seen as the integration of a multi-dimensional KDE within an intuitive and structured decision scheme, where only the most discriminative attributes are used to perform OCC. Moreover, the OC-Tree is an approach to data description properly speaking as compared to the standard MCC approaches. To illustrate this point, let us consider a toy example proposed in Fig. 1 (left). The latter is processed in two distinct ways:

- *with a multi-class decision tree.* In this case, each Gaussian blob is associated to a distinct class (C_1, in red and C_2, in green). The associated space division is represented in dashed lines.

- *with an OC-Tree.* In this case, the Gaussian blobs are all the representatives of the same Class (called C). The limits of the corresponding hyper-rectangles are represented in continuous lines. The complementary space is the one of Outliers (called O).

As shown, multi- and one-class learning processes lead to different predictive models (Fig. 1, right). Indeed, in the context of a multi-class problem, the class representatives are supposed to share the whole domain in which the attributes take their values. Hence, a decision tree learned with an algorithm like C4.5 [49, 50] proposes a decomposition of the whole space in hyper-rectangles by means of one single attribute. On the opposite, in the context of one-class classification, we propose a learning process looking for target hyper-rectangles that do not

necessarily cover the whole domain in which the attributes take their values, since there may exist outliers to discard.

3. Our proposal

In a *divide and conquer* spirit, the implementation of our one-class tree rests on successive density estimations to raise target areas as hyper-rectangles of interest. We assess the relevance of a subdivision against an information gain criterion adapted to OCC issues proposed by [11].

Let us consider $\chi \subset \mathbb{R}^d$ a hyper-rectangle of dimensions d including target training instances. Let us note $A = \{a_1, a_2, \ldots, a_d\}$ the set of attributes and $X = \{x_1, x_2, \ldots, x_n\}$ the set of instances. The goal of our proposition is the division of the initial hyper-rectangle χ in (non necessarily adjacent) sub-spaces χ_{t_i}, represented by tree nodes t_i, in absence of counter-examples.

Let us denote as A_t the set of eligible attributes for division at a given node t. Thus, $A_t \subseteq A$. We note $A_t = \{a'_1, a'_2, \ldots a'_{l_t}\}$, l_t being the number of eligible attributes at node t, with $l_t \leq d$ accordingly. At each node t, the algorithm searches the attribute $a'_j \in A_t$ which best cuts the initial sub-space χ_t into one or several sub-space(s) χ_{t_i} such that:

$$\chi_{t_i} = \{x \in \chi_t : L_{t_i} \leq x^{a'_j} \leq R_{t_i}\} \tag{1}$$

$x^{a'_j}$ is the value of instance x for attribute a'_j; L_{t_i} and R_{t_i} are respectively the left and right bounds of the closed sub-intervals raised to split the current node t in target nodes t_i, based on attribute a'_j.

For each attribute $a'_j \in A_t$, the algorithm achieves the following steps, at a given node t.

1. Check if the attribute is still eligible and compute the related Kernel Density Estimation (KDE), i.e., an estimation of the probability density function $\hat{f}_j(x)$ based on the available training instances (see Sec. 3.1).

2. Divide the space χ_t, based on the modes of $\hat{f}_j(x)$ (see Sec. 3.2).

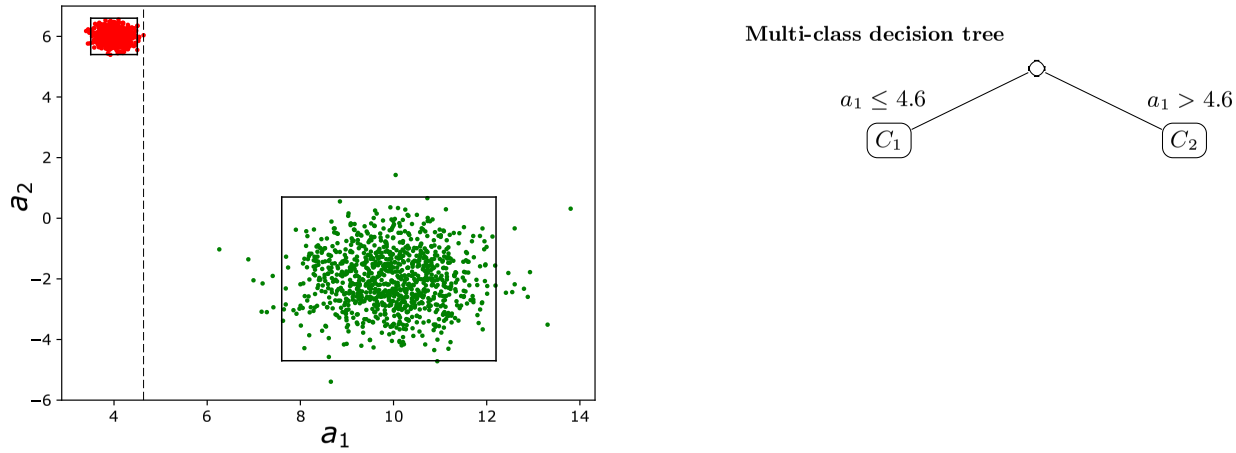

Figure 1: Comparison between a multi-class and our one-class decision tree on an artificial dataset

3. The quality of the division is assessed by the computation of the *impurity* of the resulting nodes deriving from division (see Sec. 3.3).

At each iteration, the attribute that achieves the best purity score is selected to split the current node t in child nodes. If necessary, some branches are pre-pruned in order to preserve the interpretability of the tree (see Sec. 3.4). The algorithm is run recursively; termination occurs under some stopping conditions (see Sec. 3.5).

In the rest of this paper, what we refer to as the *training accuracy* corresponds to the rate of training instances included in target nodes. It follows that, in this context of OCC, the training classification error corresponds to the rate of training instances predicted as outliers by the predictive model.

3.1. Density estimation

In order to identify concentrations of target instances, we have to estimate their distribution over the space, which can be provided by a Kernel Density Estimation (KDE). In particular, our proposal is based on the popular Gaussian kernel [45]:

$$\hat{f}_j(z) = \frac{1}{n_t h_t} \sum_{i=1}^{n_t} K\left(\frac{z - x_i}{h_t}\right) \quad \text{with} \quad K(y) = \frac{1}{\sqrt{2\pi}} \exp \frac{-y^2}{2}$$

where $\hat{f}_j$ is the KDE related to attribute a'_j, $X_t = \{x_1, x_2, \ldots, x_{n_t}\}$ is the set of n_t instances available at node t, K the kernel function and h_t, a parameter called *bandwidth*.

The parameter h_t influences the pace of the resulting function $\hat{f}_j(x)$ [45]. As h_t tends towards zero, $\hat{f}_j(x)$ appears over-shaped while high values of h_t induce a less detailed density estimation. Adaptive methods, such as a least-squares cross-validation, may help setting the bandwidth value [51, 52]. However, such iterative techniques are computationally expensive; their use may be hardly

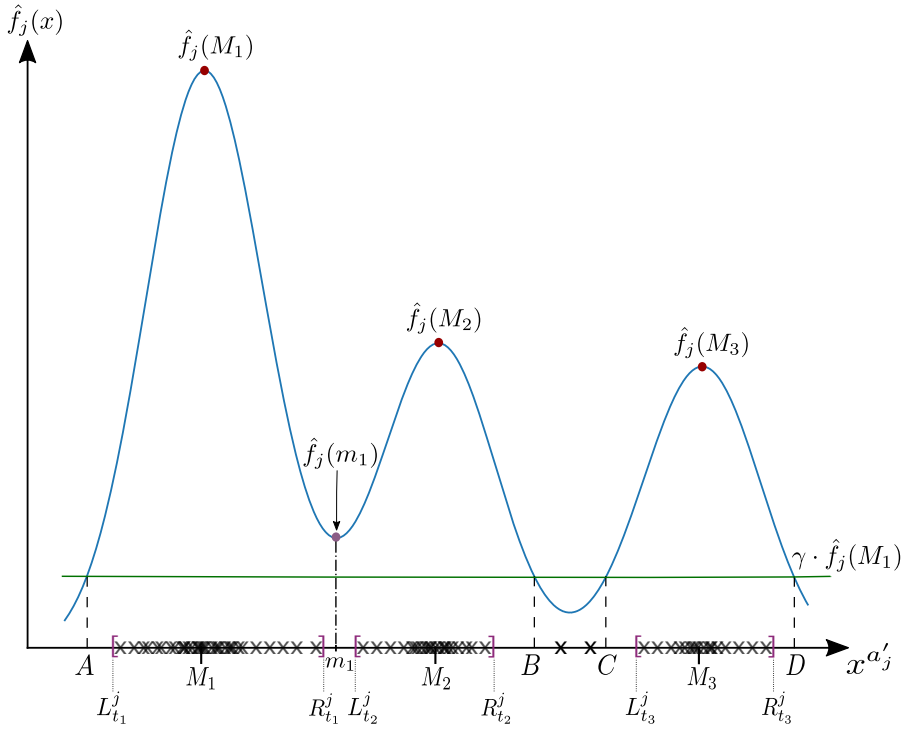

Figure 2: Division mechanism

considered in this context of recursive divisions. Hence, we compute h_t as [45]:

$$h_t = \begin{cases} 0.9 \cdot \min(\hat{\sigma}, IQR/1.34) \cdot n_t^{-1/5} & \text{if } IQR \neq 0 \\ 0.9 \cdot \hat{\sigma} \cdot n_t^{-1/5} & \text{otherwise} \end{cases} \tag{2}$$

where $\hat{\sigma}$ is the standard deviation of the sample X_t and IQR, the associated inter-quartile range. The first relation corresponds to the Silverman's *rule of thumb* [45]. We consider the second relation to address samples with $IQR = 0$, which may reveal very concentrated data, with the potential presence of some singularities that should be eliminated.

3.2. Division

At node t, division is executed based on $\hat{f}_j(x)$, in four steps.

(a) **Clipping KDE (γ)**

$\hat{f}_j(x)$ is thresholded at the level $\gamma \cdot \max_{x \in \chi_t} \hat{f}_j(x)$.
This allows to raise a set of target sub-intervals ${Y_j}^t$.

(b) **Revision (α)**

If $\hat{f}_j(x)$ is k-modal ($k \neq 1$) and $1 \leq |{Y_j}^t| < k$, revision occurs since some modes were not identified. Each sub-interval of ${Y_j}^t$ is thus analyzed: if its image by $\hat{f}_j(x)$ includes at least a significant local minimum, the interval is split in two sub-intervals around this (these) local minimum (minima). The significance of a local minimum is assessed through a parameter α (see below).

(c) **Assessment (β)**

The sub-intervals of ${Y_j}^t$ covering a number of training instances inferior to a quantity $\beta.|T|$ are dropped. This ensures keeping the most significant target nodes.

(d) **Shrinking**

The detected sub-intervals are shrunk in closed intervals in a way to fit the domain strictly covered by the related target training instances, as defined by Eq. 1.

Actually, ${Y_j}^t$ may be updated at the end of steps (b), (c), (d).

If we consider the KDE presented by Fig. 2, (a) results in ${Y_j}^t = \{[A, B]; [C, D]\}$. As the density estimation is 3-modal in this case, a revision of the interval partitioning (b) is launched. It appears there is no need to split the sub-interval $[C, D]$ since the piecewise $\hat{f}_j([C, D])$ includes a single maximum. By contrast, a local minimum is detected in m_1, in the piecewise $\hat{f}_j([A, B])$. The sub-interval $[C, D]$ is thus split into three parts around the local minimum. Concretely, such a split occurs if the local minimum is significant, i.e., sufficiently deep in comparison with both nearby local maxima. In mathematical terms:

$$\hat{f}_j(m_1) \leq \alpha \cdot \min(\hat{f}_j(M_1), \hat{f}_j(M_2)).$$

Thus ${Y_j}^t = \{[A, m_1[; [m_1, C[; [C, D]\}$. Steps (c) and (d) are then launched. The sub-intervals are shrunk around the target training instances (represented by crosses in Fig.2), which results in:

$$ {Y_j}^t = \{[L^j_{t_1}, R^j_{t_1}]; [L^j_{t_2}, R^j_{t_2}]; [L^j_{t_3}, R^j_{t_3}]\}. $$

The complement $\overline{{Y_j}^t}$ represents the set of outlier sub-spaces: it may be represented by a single branch entitled "else".

Except for prior knowledge that would help choosing its value more specifically, there should be no reason to set a high reject threshold β (e.g., $> 2\%$) since the training set is supposed to include a majority of target instances; this would be penalizing with the exclusion of real target nodes as a consequence. An appropriate value for parameter α may be selected by cross-validation; actually, a non-zero value for α (e.g., 0.5) will lead to revision, which appears to be interesting if we want to detect precisely target groupings. Basically, the value of the clipping threshold γ should be low (e.g., 0.05), because it aims at rejecting outliers.

3.3. Impurity decrease computation

At this stage of the algorithm, we have to assess the quality of a division in a particular context, i.e., the absence of representatives for at least a second class. One way to achieve this task is to resort to the physical generation of n'_t outliers in each node [13, 12]; as a result of the division, each child node would include a number of n'_{t_i} instances which would have to be estimated.

The virtual generation of outliers is worth considering as well. In this regard, the work of [11] assumes that each parent node includes uniformly distributed outliers in equal number to that of the target instances, i.e., $n_t = n'_t$. Thus, the number of outliers in each child node may be easily deduced:

$$ n'_{t_i} = n'_t \frac{\mu(\chi_{t_i})}{\mu(\chi_t)} \tag{3} $$

where μ denotes the measure of the hyper-rectangle to which it relates.

Assuming $n_t = n'_t$ may appear counter-intuitive. Indeed, we would naturally be inclined to assume, once and for all, $n = n'$ in the initial root node and to deduce the number of outliers in each child node according to Eq. 3. But throughout the iterations, this would lead to increase the scarcity of the outliers, and thus to their unfair representation in each node. The latter situation corresponds to the well-known effect of the *curse of dimensionality* [53]. This is why the number of outliers in each node t is considered as corresponding to the number n_t of target instances prior to any division [11].

Based on this predictive calculation, the work of [11] gives a proxy for the Gini impurity decrease for OCC. We adapt this result to our proposal where a division may result in more than two child nodes t_i, based on sub-intervals of interest:

$$I_G^{Proxy}(t_i, 1 \leq i \leq r_t) = \sum_{i=1}^{r_t} \frac{n_{t_i} n'_{t_i}}{n_{t_i} + n'_{t_i}}$$

$$\text{with} \quad n'_{t_i} = n'_t \frac{R_{t_i}^{a'_j} - L_{t_i}^{a'_j}}{R_t^{a'_j} - L_t^{a'_j}}$$

where r_t is the total number of target and outlier sub-intervals, included in ${Y_j}^t \cup \overline{{Y_j}^t}$.

3.4. Pre-pruning mechanism

A branch of an OC-Tree is prepruned if there are no more eligible attributes for division. An attribute is not eligible if:

- for this attribute, all the instances have the same value;
- the attribute was already used previously to cut the same target node which was not split in several target nodes in the meantime;
- the computed bandwidth h_t is strictly inferior to the minimum of the difference between two (different) successive values in the set of available instances, i.e., data granularity.

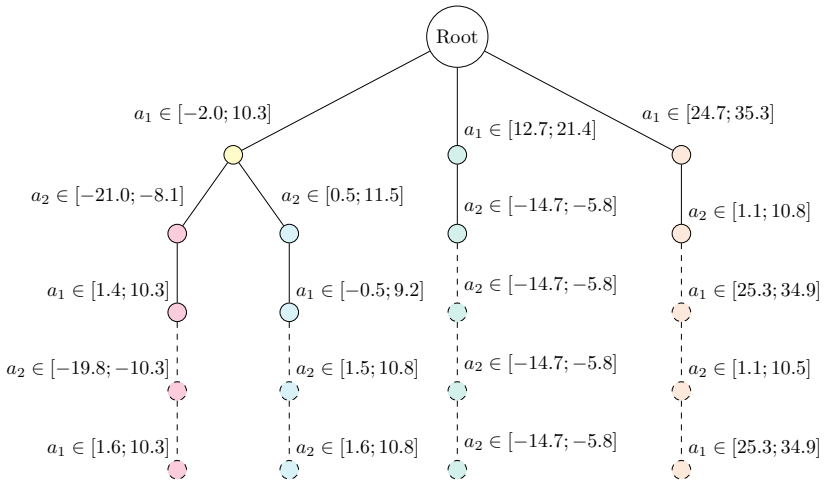

Figure 3: Pre-pruning mechanism

At a given node t, a division based on a non-eligible attribute makes no more sense. Fig. 3 shows a tree trained on two attributes. The nodes in dotted lines are developed in absence of a pre-pruning mechanism; the latter allows to get a shorter and readable decision tree. Note that the branches related to outliers were omitted for the sake of clarity.

The user has basically the choice to keep either the tree as (1) a full predictive model which describes the development that brought to the space division, or (2) the description of the final target hyper-rectangles as a set of sub-intervals of interest regarding the attributes that were used for division.

3.5. Stopping conditions

Let us denote the training accuracy as A_{tr}: it corresponds to the ratio of training instances included in the target nodes. The algorithm stops under some global and local conditions.

- **Globally**, the algorithm is stopped:
  - if A_{tr} remains stable after an iteration in which no additional target node was raised. In this case, the training process reaches a stage

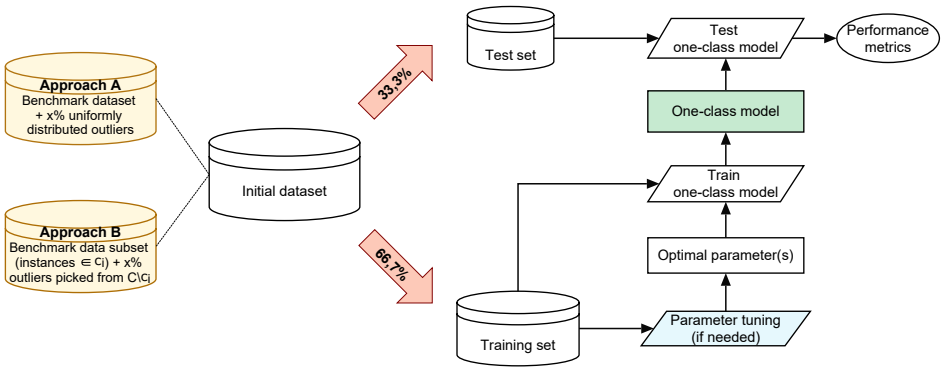

Figure 4: Experimental pipeline

where the target sub-spaces are simply more precisely delimited on the basis of additional attributes, with no further multiplication.

- if $A_{tr} < 1-\nu$, where ν is a parameter corresponding to the fraction of training instances which we tolerate to reject and consider as outliers.

- Divisions may be stopped **locally** if there are compelling reasons to convert a node in a leaf, i.e., when pre-pruning is necessary (see Sec. 3.4).

4. Experimental protocol

Fig. 4 summarizes our experimental protocol which is explained in detail in the following sections.

4.1. Reference methods

We compared the OC-Tree with three reference methods, namely the ClusterSVDD [32], One-Class Support Vector Machine (OCSVM) [22] and Isolation Forest (iForest) [46].

The comparison of the OC-Tree with ClusterSVDD is highly relevant since both methods pursue similar objectives, i.e., enclosing data within one or several hyper-rectangle(s) and hyper-sphere(s) respectively. ClusterSVDD requires that two parameters should be optimized on a dataset: ν and k which constitute respectively, the upper bound on the fraction of instances lying outside the

ClusterSVDD	OC-Tree
• Detects target hyper-sphere(s). • Requires to set the number of hyper-sphere(s) as a parameter. • Relies on two parameters: k, ν_{SVDD}. • Results in a classification model whose predictions are based on the whole set of training attributes.	• Detects target hyper-rectangle(s). • Does not require indications about the number of hyper-rectangle(s) to detect. • Relies on four parameters: $\gamma, \beta, \alpha, \nu$. • Results in a classification model whose predictions are based on a subset of training attributes.

Table 1: Comparison of ClusterSVDD & OC-Tree

decision boundary and the supposed number of clusters. Table 1 exposes a theoretical comparison of the OC-Tree with ClusterSVDD.

OCSVM is a standard OCC method to which a comparison is thus worth considering. We used a Gaussian kernel for this method, and we optimized ν which pursues the same objective as in ClusterSVDD and OC-Tree. Thus, to ensure a fair comparison, we adjusted this parameter in the same way that we did for ClusterSVDD. Finally, a method like iForest provides a relevant benchmark since it is of the same nature than OC-Tree, i.e., a tree-based method, but built in a very different way. Indeed, this ensemble technique aims at the development of decision trees based on a random choice of attributes and thresholds. If the average path length skimmed in the trees is low (resp. high), an instance is predicted as outlier (resp. target). We used the standard parameter settings for this method, since it was shown that the performances are ensured to be quite optimal with such settings [46].

4.2. Benchmark datasets

In absence of benchmark data for OCC, it is standard practice to convert multi-class problems into one-class ones for evaluation purposes. We thus considered a set of benchmark datasets (see Table 2), where each instance belongs to a class c_i among a set of C. The relevance of OC-Tree and of the reference methods on these datasets was assessed in two distinct ways.

	# Classes	# Features	# Instances
Australian	2	14	690
Diabetes	2	8	268
Ionosphere	2	34	351
Iris	3	4	150
Satimage	6	36	4435
Segment	7	19	2310

Table 2: Benchmark datasets [58, 59]

A. All the instances, whatever their class, were considered as the representatives of a same class. We injected in this dataset a certain percentage of additional outliers following a uniform distribution [32]. (**Approach A**)

B. We adopted the *one vs rest* [12] strategy which consists of considering a class $c_i \in C$ as a target one and the others as outliers [54, 13, 12, 55, 56, 57]. In this case, the outliers injected in a given data subset were randomly picked among the representatives of the outlier classes, i.e., $C \backslash c_i$. (**Approach B**)

Whether through approach A or B, the resulting dataset was split in a way that two thirds constituted a training set, while the remaining was kept as a test set.

4.3. Performance metrics

As one-class classification deals with unbalanced datasets, we may hardly consider true positives (or true targets) and true negatives (or true outliers) as equally significant. On this regard, the couple *precision-recall* provides appropriate evaluation metrics [60].

Let us denote as TT (resp. TO), the number of True Targets (resp. True Outliers), i.e., the number of instances correctly detected as targets (resp. outliers); FT (resp. FO) are the number of False Targets (resp. False Outliers) [55]. Precision and recall are defined as follows.

- **Precision** expresses the ratio of instances that were correctly predicted

as target ones to those which were predicted as such.

$$\text{Precision} = \frac{TT}{TT + FT} \tag{4}$$

- **Recall** expresses the ratio of instances that were correctly predicted as target ones to those which are truly target instances.

$$\text{Recall} = \frac{TT}{TT + FO} \tag{5}$$

Precision and recall can be embedded in a single performance indicator, namely the **F1-score** [60].

$$F1 = 2 \cdot \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{6}$$

4.4. Model selection

The reference methods and the OC-Tree need to be adjusted with respect to their parameter(s). For each classifier, **parameter tuning** was achieved through a 10-fold Cross-Validation (10-fold CV), practiced on the training set exclusively. The ranges of parameter values considered for tuning are presented in Table 3. In the case where we had to optimize two parameters, we assessed each possible combination of values for these parameters (grid search). The parameterization which led to the best 10-fold CV performance at the sense of the F1-score (see Eq. 6) was selected to **train the final one-class model** on the whole training set. This model was then **assessed against the test set**.

The tuning procedure is illustrated for the OC-Tree in Fig. 5. A given 10-fold CV run is a set of 10 iterations in which each fold (alternately) is kept for performance testing, while the 9 remaining folds constitute the training subset on which a model is trained. In this case, we had to find the optimal couple of parameters (α^*, ν^*) which yielded the best 10-fold CV performance. There are as many 10-fold CV runs as there are possible combinations of values for parameters α and ν, i.e., 16 runs overall (4 values for α $\times$ 4 values for ν).

Note that the range of values for parameter k, i.e., the number of clusters in

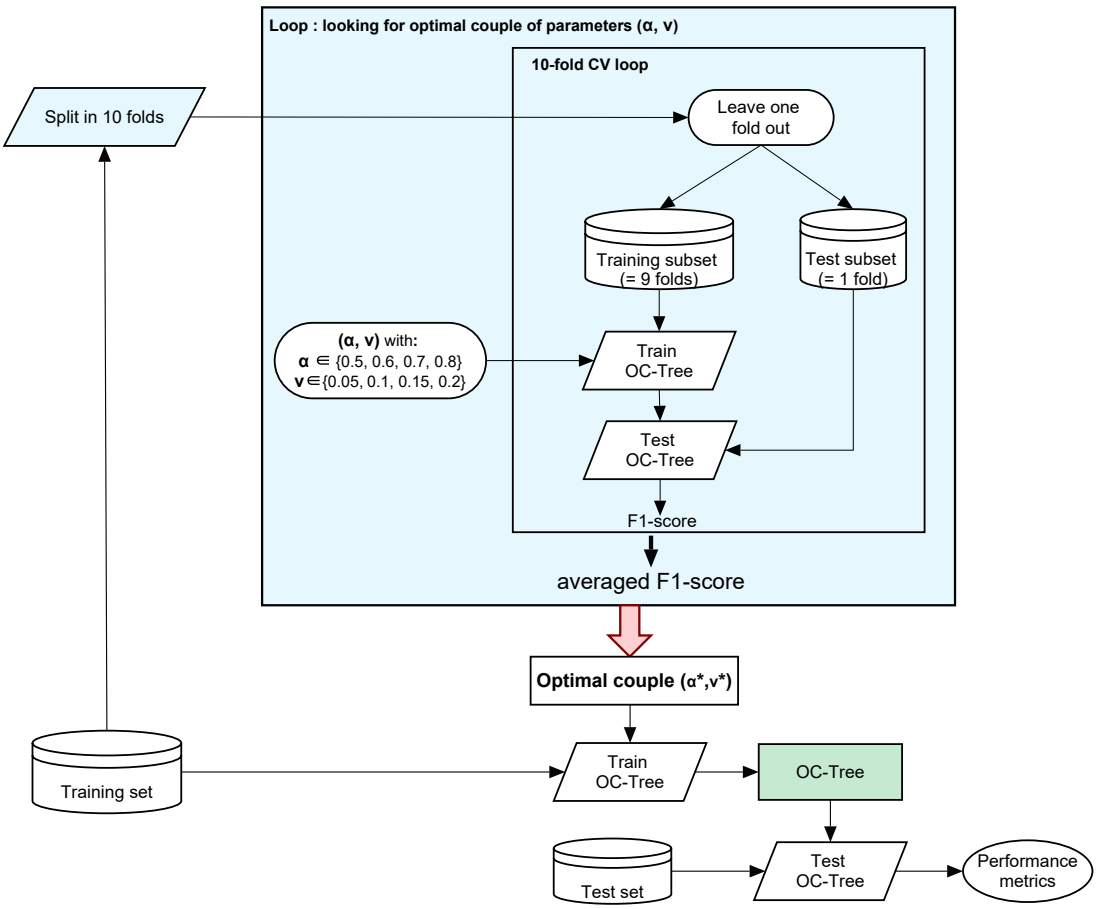

Figure 5: Illustration of parameter tuning for the OC-Tree

Method	Settings
OC-Tree	• $\gamma = 0.05$ • $\alpha = \{0.5, 0.6, 0.7, 0.8\}$ • $\beta = 2\%$ (min. 5 inst./node) • $\nu = \{0.05, 0.1, 0.15, 0.2\}$
ClusterSVDD	• k (see Table 4) • $\nu = \{0.05, 0.1, 0.15, 0.2\}$
OCSVM	$\nu = \{0.05, 0.1, 0.15, 0.2\}$
iForest	Not required

Table 3: Parameter settings

ClusterSVDD, has been differentiated depending on the considered dataset and the approach under which the datasets were addressed, as defined in Sec. 4.2. Some values are suggested in [32]. More particularly in regards to approach B, it appeared to us reasonable to set a range of $[1, 5]$ as possible values for parameter k, regardless of the considered dataset. Indeed, in this case, each class of the multi-class problem is considered for OCC. Thus, intuitively, one would expect that data are concentrated within a small number of target groupings but in the same time, the presence of a single class may reveal a structure of data different from the one observed in the case of a multi-class problem. That is why k may present higher values than those considered with approach A for some datasets.

5. Results

In this section, we first propose to the reader a preliminary experiment on synthetic data, to better understand the scope of the advocated method. We then report the results achieved on benchmark datasets.

	Approach A	Approach B
Australian	$\{1, 2\}$	$\{1, 2, 3, 4, 5\}$
Diabetes	$\{1, 2, 4\}$	
Ionosphere	$\{1, 2\}$	
Iris	$\{1, 2, 3\}$	
Satimage	$\{1, 3, 6, 9\}$ [32]	
Segment	$\{1, 5, 7, 10, 14\}$ [32]	

Table 4: Selected values for parameter k (ClusterSVDD)

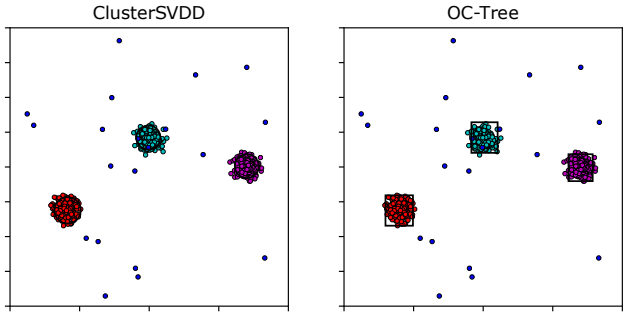

Figure 6: Detection of three Gaussian blobs with 2% of outliers included in the training set

5.1. Preliminary experiment on synthetic data

We propose a first qualitative evaluation of our OC-Tree with ClusterSVDD with respect to the detection of three Gaussian blobs enclosing altogether 1000 instances. The parameter settings are given below.

- OC-Tree : $\gamma = 0.05$, $\alpha = 1$, $\beta = 0\%$, $\nu = 0.1$.
- ClusterSVDD : $k = 3$, $\nu_{SVDD} = 0.1$.

The parameters of OC-Tree were established in a quite penalizing way, in the sense that setting α at 1 means a systematic revision of any division with the risk of decomposing unnecessarily the space covered by the target instances. Moreover, setting β at 0% means no node is dropped; this may potentially lead to small hyper-rectangles to describe the target data.

Additional instances were added to the dataset in the form of uniformly distributed outliers, in proportions of 2% and 5% of the initial training set size

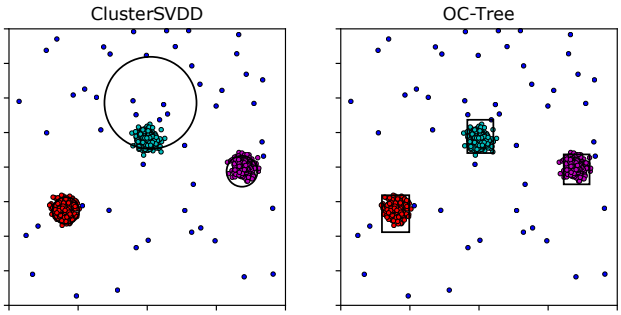

Figure 7: Detection of three Gaussian blobs with 5% of outliers included in the training set

Noise level	ClusterSVDD		OC-Tree	
	Precision	Recall	Precision	Recall
2%	0.998	0.917	0.998	0.985
5%	0.995	0.940	0.999	0.987

Table 5: Performance assessment on artificial data

respectively. The results are proposed in Figs. 6 and 7. Both methods detect the blobs in the form of circles and rectangles respectively. However, it seems that OC-Tree is less sensitive to higher noise levels than ClusterSVDD (see Fig. 7). Table 5 compares the performances of ClusterSVDD and OC-Tree in terms of precision and recall.

5.2. Experiments on benchmark datasets

In the present section, we compare our algorithm to ClusterSVDD, OCSVM, and iForest on benchmark datasets, according to the protocol summarized in Sec. 4. Table 6 on the one hand, and Tables 7, 8 on the other hand summarize the results for the approaches A and B respectively. We report the couple of values Precision – Recall for a validation achieved on the test sets. The results that are marked with an asterisk indicate that the corresponding method outperforms OC-Tree of more than 2% in terms of F1-score. The lines that are succeeded with '(+)' indicates that the OC-Tree achieves a score superior or equal to the other techniques for the considered dataset.

5.2.1. Based on uniformly distributed noise – Approach A

Our first experiment was achieved on the benchmark datasets summarized in Table 2, in which uniformly distributed noise was injected in proportions of 2, 5, 10, 15 % of the initial dataset sizes. It appears that the OC-Tree performs favorably in comparison to the other reference methods. The improvements achieved against iForest may be explained by the fact that the latter method is properly intended for anomaly detection, and may thus have slightly lower performances when the proportion of outliers in the training set is low [46], especially for proportions of 2% and 5%. Moreover, compared to iForest, the OC-Tree seems globally to better handle the *ionosphere* and *satimage* datasets. Actually, the *ionosphere* dataset has a quite diffuse distribution of data along some dimensions, which involves that some normal instances may lie far away from the others. As it is built on a random choice of attributes, the iForest method is likely to detect these instances as outliers. On the opposite, the OC-Tree is built on attributes which concentrate the instances, so the ones lying outside these concentrations may be really perceived as outliers. As regards the *satimage* dataset, the low proportion of outliers in such a high dimensional dataset may have disadvantaged the iForest method, with a difference in terms of F1-score that can reach 5%. As regards the performances of OCSVM, they are in some cases lower than OC-Tree, which may be explained by the fact that OCSVM encloses data within a single boundary and can thus not exactly adjust to the structure of data. Finally, as mentioned previously, ClusterSVDD may be sensitive to noise, which explains why the OC-Tree provides better results in some cases.

5.2.2. Based on the one vs rest strategy – Approach B

In this case, the multi-class problems related to the considered datasets are converted to one-class problems in which the representatives of the other classes are considered as outliers, injected in proportions of 2, 5, 10, 15 % of the one-class dataset sizes. In such a situation, we can expect that reference methods such as OCSVM and iForest perform better since they handle the data of each

Dataset	Noise level	ClusterSVDD	OCSVM	iForest	OC-Tree	
Australian	2%	0.986 - 0.921	0.995 - 0.908	1.000 - 0.926	1.000 - 0.965	(+)
	5%	0.973 - 0.926	0.977 - 0.922	1.000 - 0.922	1.000 - 0.970	(+)
	10%	0.900 - 0.960	0.916 - 0.960	1.000 - 0.991*	0.900 - 0.996	
	15%	0.877 - 0.961	0.882 - 0.935	0.955 - 1.000	1.000 - 0.961	(+)
Diabetes	2%	1.000 - 0.941	1.000 - 0.941	1.000 - 0.874	0.992 - 0.965	(+)
	5%	0.998 - 0.965*	0.988 - 0.957	0.996 - 0.914	0.992 - 0.911	
	10%	0.932 - 0.972	0.975 - 0.933	0.996 - 0.952	0.980 - 0.952	
	15%	0.974 - 0.897	0.974 - 0.901	0.965 - 0.988	0.992 - 0.945	
Ionosphere	2%	0.972 - 0.914	0.972 - 0.897	0.981 - 0.879	1.000 - 1.000	(+)
	5%	0.938 - 0.913	0.937 - 0.904	0.937 - 0.904	1.000 - 1.000	(+)
	10%	0.884 - 0.939	0.880 - 0.904	0.904 - 0.912	0.983 - 1.000	(+)
	15%	0.828 - 0.946	0.824 - 0.920	0.832 - 0.929	0.982 - 1.000	(+)
Iris	2%	1.000 - 0.902	1.000 - 0.961	1.000 - 0.922	1.000 - 0.941	
	5%	0.977 - 0.860	0.980 - 0.960	0.978 - 0.900	0.943 - 1.000	(+)
	10%	0.979 - 0.920	1.000 - 0.940*	0.958 - 0.920	0.978 - 0.900	
	15%	0.902 - 0.920	0.889 - 0.960	0.889 - 0.960	0.862 - 1.000	(+)
Satimage	2%	0.995 - 0.945	0.996 - 0.957	0.996 - 0.890	0.996 - 0.968	(+)
	5%	0.986 - 0.974	0.986 - 0.971	0.984 - 0.914	0.979 - 0.981	(+)
	10%	0.991 - 0.981	0.977 - 0.946	0.952 - 0.922	0.981 - 0.969	
	15%	0.990 - 0.963	0.966 - 0.937	0.907 - 0.934	0.980 - 0.968	
Segment	2%	0.999 - 0.963	0.999 - 0.974	1.000 - 0.912	1.000 - 1.000	(+)
	5%	0.974 - 0.970	0.978 - 0.970	1.000 - 0.940	1.000 - 1.000	(+)
	10%	0.927 - 0.980	0.930 - 0.979	1.000 - 0.991	0.993 - 1.000	(+)
	15%	0.898 - 0.970	0.928 - 0.946	0.976 - 1.000*	0.872 - 0.996	

Table 6: Results (Precision – Recall) - Approach A

class separately. De facto, the OC-Tree shows overall smaller differences in performance.

6. Application to the diagnosis of ADHD

In the previous section, we compared the OC-Tree on benchmark datasets with reference one-class methods, against which it proved to perform favorably. In the present section, we propose a real-world case study in which an algorithm such as the OC-Tree is worth considering. The application is related to the diagnosis of Attention Deficit Hyperactivity Disorder (ADHD).

6.1. Problem statement

ADHD is a neurodevelopmental disorder in children which has been subject to a considerable number of studies, including those conducted on the ADHD-200 collection [61]. This open and free database has been made available since 2012 in order to advance the state of knowledge about ADHD [18].

Dataset	Noise level	ClusterSVDD	OCSVM	iForest	OC-Tree	
Australian (-1)	2%	0.984 - 0.945	0.983 - 0.914	0.991 - 0.875	0.992 - 0.977	(+)
	5%	0.936 - 0.936	0.935 - 0.920	0.959 - 0.928	0.945 - 0.960	(+)
	10%	0.902 - 0.960	0.919 - 0.912	0.941 - 0.896	0.890 - 0.968	(+)
	15%	0.819 - 0.934	0.822 - 0.917	0.886 - 0.901	0.834 - 1.000	(+)
Australian (+1)	2%	0.980 - 0.951	0.980 - 0.951	0.980 - 0.951	0.990 - 0.980	(+)
	5%	0.950 - 0.950	0.950 - 0.941	0.949 - 0.921	0.943 - 0.990	(+)
	10%	0.906 - 0.950	0.932 - 0.950	0.947 - 0.891	0.901 - 0.990	(+)
	15%	0.881 - 0.960	0.872 - 0.950	0.855 - 0.940	0.860 - 0.980	
Diabetes (-1)	2%	0.978 - 0.978*	0.977 - 0.966*	0.976 - 0.910	0.976 - 0.910	
	5%	0.957 - 0.978*	0.956 - 0.967*	0.954 - 0.922*	0.952 - 0.878	
	10%	0.944 - 0.934	0.926 - 0.956	0.928 - 0.846	0.926 - 0.956	(+)
	15%	0.863 - 0.921	0.876 - 0.955	0.874 - 0.933	0.862 - 0.910	
Diabetes (+1)	2%	0.981 - 0.933	0.980 - 0.903	0.980 - 0.879	0.982 - 0.988	(+)
	5%	0.945 - 0.951	0.956 - 0.927	0.955 - 0.909	0.942 - 0.982	(+)
	10%	0.895 - 0.962	0.905 - 0.956	0.909 - 0.938	0.881 - 0.975	
	15%	0.853 - 0.938	0.858 - 0.938	0.871 - 0.919	0.856 - 0.963	(+)
Ionosphere (-1)	2%	0.974 - 0.881	0.967 - 0.690	0.971 - 0.810	0.977 - 1.000	(+)
	5%	0.946 - 0.833	0.935 - 0.690	0.946 - 0.833	0.955 - 1.000	(+)
	10%	0.872 - 0.829	0.857 - 0.732	0.872 - 0.829	0.943 - 0.805	(+)
	15%	0.889 - 0.930	0.861 - 0.721	0.895 - 0.791	0.905 - 0.884	(+)
Ionosphere (+1)	2%	1.000 - 0.960	1.000 - 0.960	1.000 - 0.893	0.986 - 0.973	(+)
	5%	0.973 - 0.973	0.986 - 0.946	0.986 - 0.919	0.947 - 0.959	
	10%	0.972 - 0.932	0.973 - 0.959	0.956 - 0.878	0.973 - 0.973	(+)
	15%	0.920 - 0.958*	0.909 - 0.972	0.920 - 0.958*	0.861 - 0.944	
Iris (1)	2%	1.000 - 1.000*	1.000 - 1.000	1.000 - 0.941*	1.000 - 0.882	
	5%	0.933 - 0.824	1.000 - 0.824	1.000 - 0.882	1.000 - 0.882	(+)
	10%	0.938 - 0.833	0.938 - 0.833	1.000 - 1.000*	1.000 - 0.889	(+)
	15%	0.941 - 0.842	0.941 - 0.842	1.000 - 1.000*	1.000 - 0.895	
Iris (2)	2%	1.000 - 1.000	1.000 - 1.000	1.000 - 0.824	1.000 - 1.000	(+)
	5%	0.944 - 1.000	0.944 - 1.000	0.944 - 1.000	0.944 - 1.000	(+)
	10%	1.000 - 1.000	0.947 - 1.000	1.000 - 1.000	1.000 - 1.000	(+)
	15%	1.000 - 1.000*	0.947 - 0.947	1.000 - 1.000*	0.947 - 0.947	
Iris (3)	2%	1.000 - 0.824	1.000 - 0.824	1.000 - 0.824	1.000 - 1.000	(+)
	5%	0.941 - 0.941*	0.933 - 0.824	0.933 - 0.824	0.938 - 0.882	
	10%	1.000 - 1.000*	1.000 - 0.722	1.000 - 1.000*	1.000 - 0.833	
	15%	0.929 - 0.684	1.000 - 0.789	1.000 - 0.895	1.000 - 0.895	(+)

Table 7: Results (Precision – Recall) - Approach B

Dataset	Noise level	ClusterSVDD	OCSVM	iForest	OC-Tree	
Satimage (1)	2%	0.997 - 0.975	0.991 - 0.955	1.000 - 0.905	0.997 - 0.958	
	5%	0.977 - 0.964	0.980 - 0.972	0.997 - 0.952	0.983 - 0.972	(+)
	10%	0.958 - 0.992	0.955 - 0.986	0.986 - 0.981	0.970 - 0.992	(+)
	15%	0.914 - 0.981	0.914 - 0.978	0.959 - 0.983	0.918 - 0.992	
Satimage (2)	2%	0.980 - 0.942	0.980 - 0.949	0.986 - 0.872	0.979 - 0.910	
	5%	0.994 - 0.981	1.000 - 0.975	1.000 - 0.898	0.980 - 0.955	
	10%	0.910 - 0.987	0.915 - 0.981	0.967 - 0.942	0.937 - 0.968	
	15%	0.925 - 0.948	0.902 - 0.955	0.950 - 0.987*	0.884 - 0.981	
Satimage (3)	2%	0.984 - 0.984	0.984 - 0.981	1.000 - 0.953	0.987 - 0.959	
	5%	0.984 - 0.972	0.984 - 0.966	0.994 - 0.957	0.966 - 0.975	
	10%	0.942 - 0.985	0.944 - 0.988	0.979 - 0.985	0.972 - 0.948	
	15%	0.906 - 0.981	0.917 - 0.985	0.944 - 0.988	0.924 - 0.971	
Satimage (4)	2%	0.984 - 0.933	0.984 - 0.933	0.992 - 0.926	0.984 - 0.933	(+)
	5%	0.969 - 0.941	0.977 - 0.941	0.992 - 0.889	0.964 - 0.985	(+)
	10%	0.907 - 0.948	0.920 - 0.948	0.961 - 0.918	0.917 - 0.985	(+)
	15%	0.869 - 0.940	0.910 - 0.978	0.928 - 0.955	0.905 - 0.993	(+)
Satimage (5)	2%	0.980 - 0.948	0.979 - 0.935	0.993 - 0.869	0.966 - 0.941	
	5%	0.967 - 0.961	0.966 - 0.947	0.993 - 0.882	0.931 - 0.974	
	10%	0.937 - 0.955	0.932 - 0.968	0.942 - 0.929	0.921 - 0.968	
	15%	0.876 - 0.993	0.883 - 0.953	0.898 - 0.940	0.865 - 9.980	
Satimage (6)	2%	0.997 - 0.977	0.994 - 0.977	1.000 - 0.951	0.994 - 0.986	(+)
	5%	0.977 - 0.968	0.977 - 0.971	0.988 - 0.942	0.980 - 0.991	(+)
	10%	0.948 - 0.986	0.950 - 0.986	0.977 - 0.980	0.961 - 0.980	
	15%	0.910 - 0.972	0.931 - 0.952	0.941 - 0.997	0.966 - 0.957	
Segment (1)	2%	0.982 - 0.973*	0.982 - 0.982*	1.000 - 0.855	1.000 - 0.873	
	5%	0.981 - 0.972	0.972 - 0.981	1.000 - 0.906	1.000 - 0.934	
	10%	0.915 - 1.000	0.938 - 0.991	1.000 - 1.000*	0.945 - 0.972	
	15%	0.945 - 0.963	0.938 - 0.981	0.973 - 1.000*	0.919 - 0.953	
Segment (2)	2%	0.991 - 0.955	0.990 - 0.936	1.000 - 0.882	1.000 - 0.927	
	5%	0.981 - 0.972	0.981 - 0.962	1.000 - 0.934	0.955 - 0.991	(+)
	10%	0.910 - 0.944	0.927 - 0.944*	1.000 - 0.972	0.955 - 0.991	
	15%	0.855 - 0.991	0.851 - 0.963*	0.964 - 1.000*	0.990 - 0.897	
Segment (3)	2%	0.981 - 0.918	0.981 - 0.936	0.990 - 0.927	1.000 - 0.982	(+)
	5%	0.950 - 0.896	0.949 - 0.887	0.939 - 0.877	0.962 - 0.943	(+)
	10%	0.909 - 0.935	0.925 - 0.925	0.951 - 0.916	0.904 - 0.972	(+)
	15%	0.862 - 0.935	0.860 - 0.916	0.907 - 0.907	0.866 - 0.963	(+)
Segment (4)	2%	1.000 - 0.945	1.000 - 0.955	0.990 - 0.918	0.991 - 1.000	(+)
	5%	0.952 - 0.934	0.981 - 0.962	0.990 - 0.925	0.981 - 0.991	(+)
	10%	0.937 - 0.972	0.945 - 0.963	0.971 - 0.925	0.922 - 0.991	(+)
	15%	0.898 - 0.991	0.906 - 0.991	0.927 - 0.953	0.869 - 0.991	
Segment (5)	2%	0.981 - 0.945	0.981 - 0.964	0.990 - 0.927	0.991 - 0.964	(+)
	5%	0.952 - 0.934	0.962 - 0.962	0.963 - 0.972	0.955 - 0.991	(+)
	10%	0.917 - 0.925	0.920 - 0.963	0.936 - 0.963	0.921 - 0.981	(+)
	15%	0.864 - 0.953	0.858 - 0.963	0.938 - 0.981*	0.862 - 0.991	
Segment (6)	2%	0.991 - 0.964	0.991 - 0.964	1.000 - 0.909	0.990 - 0.936	
	5%	0.971 - 0.943	0.980 - 0.934	0.990 - 0.925	0.981 - 0.953	(+)
	10%	0.955 - 0.991	0.964 - 0.991	0.981 - 0.991*	0.930 - 0.991	
	15%	0.946 - 0.981	0.946 - 0.981	0.964 - 0.991*	0.898 - 0.991	
Segment (7)	2%	1.000 - 0.918	1.000 - 0.909	1.000 - 0.945	1.000 - 0.982	(+)
	5%	0.970 - 0.925	0.970 - 0.906	1.000 - 0.925	1.000 - 0.972	(+)
	10%	0.909 - 0.935	0.919 - 0.953*	1.000 - 0.981*	0.886 - 0.944	
	15%	0.863 - 0.944	0.871 - 0.944	0.939 - 1.000*	0.875 - 0.981	

Table 8: Results (Precision – Recall) - Approach B

Predicted as ▷	NT	ADHD
NT	3	5
ADHD	1	19

Predicted as ▷	NT	ADHD
NT	4	0
ADHD	5	4

Figure 8: Confusion matrices achieved on the NYU test set for boys (left) and girls (right), based on a multi-class decision tree [63]

The epidemiology of ADHD depends on gender, and evidence suggests that the disorder affects more often boys than girls [62]. Such a gender-differentiated distribution poses some concerns about the development of diagnosis aid models through multi-class classification. Indeed, unbalanced distributions of ADHD and NeuroTypical (NT) subjects are often observed for each gender group in the training sets related to ADHD. This applies to the ADHD-200 collection, and more particularly to the corresponding NYU data subset. The boys' training sample includes approximately twice as many ADHD subjects as NT ones, and the reverse trend is observed in the girls' training sample. Fig. 8 presents the confusion matrices related to the predictions recently achieved on the NYU test set for boys and girls, based on a multi-class decision tree (according to the methodology proposed in [63]). Actually, these results show the effects of class unbalance within each gender group in the training set. Though providing an overall satisfactory predictive accuracy, the final predictive model has a high (resp. low) sensitivity and a low (resp. high) specificity in boys (resp. girls). This bias is among the reasons that explain the limited applicability of such a binary predictive model in the clinical practice setting. The OC-Tree may alleviate this issue. We thus propose to tackle ADHD diagnosis on a gender-differentiated basis, in focusing on the description of the neuropathology with the OC-Tree.

6.2. Data

We consider the preprocessed ADHD-200 collection [64], and focus on the NYU sample. Table 9 presents the distribution of the training and test data, based on the gender and the diagnostic labels. For each subject, the sample includes blood-oxygen-level-dependent signals [65], at resting-state, given a brain

		Girls	Boys
Training set	NT	50	43
	ADHD	25	92
Test set	NT	4	8
	ADHD	9	20

Table 9: Distribution of the NYU sample considered in our study

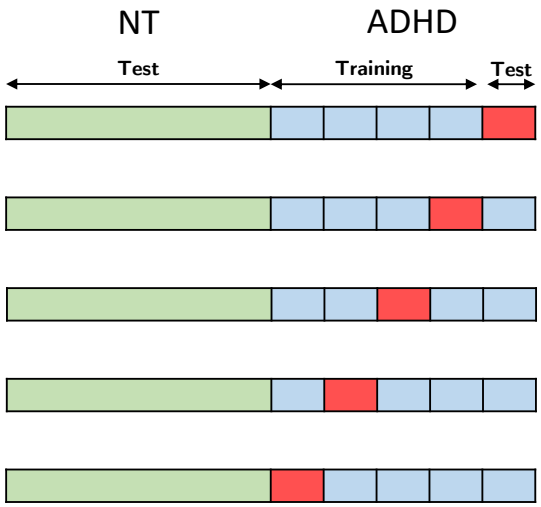

Figure 9: Cross-validation procedure used to tune the OC-Tree for ADHD prediction

parcellation in 90 regions of interest (cf. AAL90 atlas [66]). We considered the variance of the signals as predictors, since they proved to achieve successful predictions [67, 63]. They were computed for brain regions included in two functional systems which were associated to ADHD-related abnormalities in the literature: the limbic system [63] and the Default Mode Network (DMN) [68, 69].

6.3. Tuning and assessment

In this context, a quick visualization of the data shows that the instances are concentrated within a single grouping. Thus, there are no clusters to raise: the models may be reduced to a set of descriptive rules. This means that the parameter α has no influence here. Five values were considered in order to tune parameter $\nu = \{0.05, 0.1, 0.15, 0.2, 0.25\}$. The parameter was tuned through

a 5-fold CV procedure, which is depicted in Fig. 9. The NT subjects of the training set are fully used at each iteration as a test fold in combination to the one extracted from the partitioning of the ADHD training set into 5 folds.

In OCC, performance metrics such as those presented in Sec. 4.3 are generally computed with regards to the target class, i.e. ADHD in this case. However, in the specific case of psychiatric diagnosis, there is a need for cautious predictions, even though that would imply to wrongly predict a subject as neurotypical [70]. In other terms, high specificity and a reasonable level of sensitivity are requirements that a predictive model should meet in this context. We thus propose to assess the model towards its capability to predict NT cases, and thus to compute the metrics with respect to the NT group.

The models which achieve the best F1-score and precision were held as relevant for boys and girls respectively. Indeed, let us recall that our choice to assess the performance of the OC models towards the class of typical controls is motivated by the need to favor high levels of specificity. However, this is an already existing trend in girls, given that there are generally more NT girls than ADHD ones. Thus, to avoid falling into the traps of a somewhat insensitive model and to ensure that ADHD cases are predicted in a reasonable number of situations, we focus on the precision whose maximization is achieved through a minimization of the number of false NT subjects.

6.4. Classification framework

On a gender-differentiated basis, we need to predict a diagnosis based on the activity of brain regions included in the limbic system and/or the DMN. As announced, this is achieved through the practice of OCC, in targeting the ADHD group. For such a purpose, we assessed the relevance of four distinct options presented below.

- **O1**: train the OC-Tree model on the features related to the limbic system.
- **O2**: train the OC-tree model on the features related to the DMN.

- **O3**: train the OC-tree model on features related to both the limbic system and DMN.

- **O4**: constitute an ensemble of OC classifiers by the aggregation of two models trained on the limbic and DMN features separately.

In the case of the fourth option, a subject is diagnosed with ADHD once he/she tests positive with both models. In the other cases, the subject is predicted as disease-free, concerned with the need for a cautious diagnosis [70].

6.5. Final models and performance

In boys, the ensemble strategy as defined by option O4 appeared to be the most successful, with a F1-score of 65.3% on the training set ($\nu = 0.25$). The results were most tightly contested in girls between option O1 and O3, yielding respectively precision rates of 93.6% and 93.3% ($\nu = 0.15$). We selected the latter as a final model since it provides a more detailed description of the pathology than O1, which is based only on two rules. Fig. 10 presents the final confusion matrices for boys and girls.

Tables 10 and 11 present the decision rules (expressed in terms of the logarithm of the variance) related to boys and girls respectively. Note that (L) and (R) denote brain regions included in the Left and Right hemispheres respectively. Our results confirm that the resting-state activity of both the limbic system and the DMN brings some discriminative information for ADHD diagnosis. The mental condition appears to be more complex to describe in boys, and requires the combination of two distinct models. The girls' model is by contrast more minimalist. These important differences between boys and girls in terms of models strengthens our conviction that a gender-differentiated classification is definitely pertinent.

In alleviating the issue of class imbalance within each gender group, we could improve the balance between the diagnostic specificity and sensitivity. If we compare with the confusion matrices presented in Figs. 8 and 10, in boys, the improvement made on specificity (75% against 37.5% previously), was achieved

Predicted as ▷	NT	ADHD
NT	6	2
ADHD	6	14

Predicted as ▷	NT	ADHD
NT	4	0
ADHD	1	8

Figure 10: Confusion matrix achieved by the OC-Tree on the NYU test set for boys (left) and girls (right)

at the expense of the sensitivity (70% against 95% previously). In girls, the sensitivity was doubled without loss of specificity. The overall prediction accuracy was improved as well (78.0% against 73.2%).

7. Discussion

In Sec. 5, we showed that the OC-Tree presents favorable performances in comparison to reference methods such as ClusterSVDD, OCSVM, and iForest, in similar conditions. Depending on the targeted objectives, the OC-Tree may be a wise choice to achieve an OCC task.

The presence of noise in the data may impair the performances of a method like ClusterSVDD, while as a density-based method, the OC-Tree shows more ability to reject such outliers in the data. Moreover, the OC-Tree is developed to be as compact as possible, which constitutes a key to interpretability. Indeed, the predictive model is based on the most discriminative attributes to achieve OCC while ClusterSVDD and OCSVM do not consider such a selection; the corresponding models are computed based on the whole set of training attributes. The OC-Tree also detects automatically the number of groupings related to the class targeted by the classification. This constitutes a significant advantage compared to ClusterSVDD which requires to set the number of possible clusters as an input parameter. As compared to the iForest technique, the OC-Tree is more compact and readable while being able at the same time to perform outlier rejection. Finally, the OC-Tree better fits to the structure of the data as compared to OCSVM, since it allows the detection of sub-concepts of a single class as target groupings.

In Sec. 6, we were interested in a case study related to the diagnosis of

Model 1 - Limbic system		Model 2 - DMN	
Brain region	**log(Var)**	**Brain region**	**log(Var)**
Parahippocampal (L)	[-2.28 ; -1.33]	Angular gyrus (R)	[-2.09 ; -1.29]
Posterior cingulate gyrus (L)	[-1.49 ; -0.81]	Parahippocampal (L)	[-2.24 ; -1.42]
Amygdala (R)	[-2.17 ; -1.48]	Superior frontal gyrus, dorsolateral (L)	[-2.43 ; -1.59]
Superior frontal gyrus, dorsolateral (L)	[-2.43 ; -1.60]	Superior frontal gyrus, medial	[-2.08 ; -1.23]
Hyppocampus (R)	[-2.61 ; -1.81]	Posterior cingulate gyrus (L)	[-1.49 ; -0.81]
Anterior cingulate and paracingulate gyri (R)	[-2.54 ; -1.55]	Angular gyrus (L)	[-1.83 ; -1.12]
Anterior cingulate and paracingulate gyri (L)	[-1.92 ; -1.04]	Superior frontal gyrus, medial (L)	[-1.88 ; -0.97]
Superior frontal gyrus, dorsolateral (R)	[-2.61 ; -1.70]		
Thalamus (R)	[-2.46 ; -1.57]		
Posterior cingulate gyrus (R)	[-1.84 ; -1.03]		

Table 10: Predictive models for boys

Brain region	**log(Var)**
Middle temporal gyrus (R)	[-2.28 ; -1.85]
Thalamus (L)	[-2.43 ; -1.67]
Thalamus (R)	[-2.18 ; -1.40]

Table 11: Predictive model for girls

ADHD. Through this study, we could:

- show the interest of considering the OC-Tree rather than a multi-class decision tree, given the effective availability of the data;
- highlight the advantageous interpretability of the OC-Tree, which is an important characteristic towards a concrete clinical applicability;
- consider one-class ensembles that may help in modeling complex conditions while preserving the interpretability of the predictive framework.

These promising results tend to show that our model may be transposable to medical practice as a diagnosis aid tool.

8. Conclusion & future work

In some applications, the limited availability of data has led to look for alternatives to the usual multi-class classification approaches. The practice of One-Class Classification (OCC) has been considered in this context. This area of machine learning has generated a considerable interest with the development of new methods, some of which were adapted from supervised classification techniques.

In this respect, we proposed a one-class decision tree by completely rethinking the splitting mechanism usually considered in tree-based extensions for OCC. Our proposal for a One-Class decision Tree (OC-Tree) may be actually seen as a compact variant for Kernel Density Estimation (KDE), which only relies on a subset of significant attributes to perform classification. The OC-Tree has shown favorable performances in comparison to reference methods such as Cluster Support Vector Data Description, One-Class Support Vector Machine and Isolation Forest. Our proposal is thus consistent with the objectives of explainable Artificial Intelligence, which enhances both the needs of performance and interpretability. Such qualities are particularly valuable for medical diagnosis, where a balanced representation of the classes is not always ensured. Here, we

could illustrate the benefits of the OC-Tree for the diagnosis of ADHD. We believe that the convenience of the OC-Tree will make it a promising model for future clinical practice.

This work leaves two main perspectives detailed below.

- The parameterization of the KDE remains an open question in regards to the computation of the bandwidth and the use of other kernels. Indeed, on the one hand, our proposal is based on a Gaussian kernel which provides interesting mathematical properties, but the pertinence of other configurations may be studied on a comparative basis. On the other hand, the bandwidth estimation achieved with the Silverman's rule of thumb is quite sensitive to the training set content. The induction of the OC-Tree relies on a pre-pruning mechanism which aims to control this sensitivity. In the future, we would like to develop a mechanism able to address this issue of sensitivity and to increase the accuracy of the estimation. Refining the theoretical bandwidth estimation by cross-validation [51] would be an avenue worth exploring in this respect. This poses several computational challenges, notably in terms of execution time and algorithm complexity.

- It would be interesting to extend the OC-Tree further, in a way to make splits based on the interactions between attributes. Indeed, a previous research [71] showed that it is possible to improve the accuracy and keep the interpretability of generalized additive models built on pairwise interactions. The transposition of this idea to the induction of the OC-Tree requires a full investigation which brings a series of issues. Indeed, beyond interpretability, primary challenges include the identification of meaningful interactions in the context of a recursive mechanism and the design of induction procedures which ensure reasonable execution time.

9. Acknowledgments

Sarah Itani is a research fellow of the Fonds de la Recherche Scientifique - FNRS (F.R.S.- FNRS). The authors would like to thank the anonymous review-

ers for their insightful comments and suggestions to improve the paper.

References

[1] M. M. Moya, M. W. Koch, L. D. Hostetler, One-class classifier networks for target recognition applications, Tech. rep., Sandia National Labs., Albuquerque, NM (United States) (1993).

[2] S. S. Khan, M. G. Madden, A survey of recent trends in one class classification, in: Artificial Intelligence and Cognitive Science: 20th Irish Conference (AICS 2009), Springer, 2009, pp. 188–197.

[3] V. Chandola, A. Banerjee, V. Kumar, Anomaly detection: A survey, ACM computing surveys (CSUR) 41 (3) (2009) 15:1–15:58.

[4] M. Markou, S. Singh, Novelty detection: a review - part 1: statistical approaches, Signal processing 83 (12) (2003) 2481–2497.

[5] T. Kefi-Fatteh, R. Ksantini, M.-B. Kaâniche, A. Bouhoula, A novel incremental one-class support vector machine based on low variance direction, Pattern Recognition 91 (2019) 308–321.

[6] A. Retico, I. Gori, A. Giuliano, F. Muratori, S. Calderoni, One-class support vector machines identify the language and default mode regions as common patterns of structural alterations in young children with autism spectrum disorders, Frontiers in neuroscience 10 (2016) 306.

[7] B. Krawczyk, M. Woźniak, F. Herrera, On the usefulness of one-class classifier ensembles for decomposition of multi-class problems, Pattern Recognition 48 (12) (2015) 3969–3982.

[8] D. Arifoglu, A. Bouchachia, Detection of abnormal behaviour for dementia sufferers using convolutional neural networks, Artificial intelligence in medicine 94 (2019) 88–95.

[9] D. Oosterlinck, D. F. Benoit, P. Baecke, From one-class to two-class classification by incorporating expert knowledge: Novelty detection in human behaviour, European Journal of Operational Research 282 (3) (2020) 1011–1024.

[10] P. Oza, V. Patel, One-class convolutional neural network, IEEE Signal Processing Letters 26 (2) (2018) 277–281.

[11] N. Goix, N. Drougard, R. Brault, M. Chiapino, One class splitting criteria for random forests, in: Asian Conference on Machine Learning, 2017, pp. 343–358.

[12] C. Désir, S. Bernard, C. Petitjean, L. Heutte, One class random forests, Pattern Recognition 46 (12) (2013) 3490–3506.

[13] K. Hempstalk, E. Frank, I. H. Witten, One-class classification by combining density and class probability estimation, in: Machine Learning and Knowledge Discovery in Databases: European Conference (ECML PKDD 2008), Springer, 2008, pp. 505–519.

[14] A. Holzinger, C. Biemann, C. Pattichis, D. Kell, What do we need to build explainable AI systems for the medical domain?, arXiv preprint arXiv:1712.09923 (2017).

[15] C. Rudin, Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead, Nature Machine Intelligence 1 (5) (2019) 206–215.

[16] M. Tsang, Y. Sun, D. Ren, Y. Liu, Can I trust you more? Model-agnostic hierarchical explanations, arXiv preprint arXiv:1812.04801 (2018).

[17] M. T. Ribeiro, S. Singh, C. Guestrin, Why should I trust you? Explaining the predictions of any classifier, in: 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2016, pp. 1135–1144.

[18] M. P. Milham, D. Fair, M. Mennes, S. H. Mostofsky, et al., The ADHD-200 consortium: a model to advance the translational potential of neuroimaging in clinical neuroscience, Frontiers in systems neuroscience 6 (2012) 62.

[19] S. W. Yahaya, A. Lotfi, M. Mahmud, A consensus novelty detection ensemble approach for anomaly detection in activities of daily living, Applied Soft Computing 83 (2019) 105613.

[20] S. Wang, Q. Liu, E. Zhu, F. Porikli, J. Yin, Hyperparameter selection of one-class support vector machine by self-adaptive data shifting, Pattern Recognition 74 (2018) 198–211.

[21] S. S. Khan, M. G. Madden, One-class classification: Taxonomy of study and review of techniques, The Knowledge Engineering Review 29 (03) (2014) 345–374.

[22] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, R. C. Williamson, Estimating the support of a high-dimensional distribution, Neural computation 13 (7) (2001) 1443–1471.

[23] D. Tax, One-class classification: Concept learning in the absence of counter examples, Delft University of Technology (2001) 202.

[24] S. Zheng, A fast iterative algorithm for support vector data description, International Journal of Machine Learning and Cybernetics 10 (5) (2019) 1173–1187.

[25] J. Zhou, W. Fu, Y. Zhang, H. Xiao, J. Xiao, C. Zhang, Fault diagnosis based on a novel weighted support vector data description with fuzzy adaptive threshold decision, Transactions of the Institute of Measurement and Control 40 (1) (2018) 71–79.

[26] J. Yang, T. Deng, R. Sui, An adaptive weighted one-class SVM for robust outlier detection, in: 2015 Chinese Intelligent Systems Conference, Springer, 2016, pp. 475–484.

[27] M. Wu, J. Ye, A small sphere and large margin approach for novelty detection using training data with outliers, IEEE transactions on pattern analysis and machine intelligence 31 (11) (2009) 2088–2092.

[28] V. Barnabé-Lortie, C. Bellinger, N. Japkowicz, Active learning for one-class classification, in: 14th International Conference on Machine Learning and Applications (ICMLA), IEEE, 2015, pp. 390–395.

[29] P. Nguyen, D. Tran, T. Le, T. Hoang, D. Sharma, Multi-sphere support vector data description for brain-computer interface, in: 4th International Conference on Communications and Electronics (ICCE), IEEE, 2012, pp. 318–321.

[30] T. Le, D. Tran, W. Ma, D. Sharma, A theoretical framework for multi-sphere support vector data description, in: 17th International Conference on Neural Information Processing. Models and applications, Springer, 2010, pp. 132–142.

[31] Y. Xiao, B. Liu, L. Cao, X. Wu, C. Zhang, Z. Hao, F. Yang, J. Cao, Multi-sphere support vector data description for outliers detection on multi-distribution data, in: IEEE International Conference on Data Mining Workshops, IEEE, 2009, pp. 82–87.

[32] N. Görnitz, L. A. Lima, K.-R. Müller, M. Kloft, S. Nakajima, Support vector data descriptions and k-means clustering: One class?, IEEE transactions on neural networks and learning systems 29 (9) (2017) 3994–4006.

[33] J. A. Hartigan, J. Hartigan, Clustering algorithms, Vol. 209, Wiley New York, 1975.

[34] S. Chawla, P. Sun, SLOM: a new measure for local spatial outliers, Knowledge and Information Systems 9 (4) (2006) 412–429.

[35] J. X. Yu, W. Qian, H. Lu, A. Zhou, Finding centric local outliers in categorical/numerical spaces, Knowledge and Information Systems 9 (3) (2006) 309–338.

[36] A. L. M. Chiu, A. W. C. Fu, Enhancements on local outlier detection, in: 7th International Database Engineering and Applications Symposium, IEEE, 2003, pp. 298–307.

[37] M. M. Breunig, H.-P. Kriegel, R. T. Ng, J. Sander, LOF: identifying density-based local outliers, in: 2000 ACM Sigmoid International Conference On Management Of Data, Vol. 29, ACM, 2000, pp. 93–104.

[38] K. M. Kumar, A. R. M. Reddy, A fast DBSCAN clustering algorithm by accelerating neighbor searching using Groups method, Pattern Recognition 58 (2016) 39–48.

[39] Y. Zhang, X. Wang, B. Li, W. Chen, T. Wang, K. Lei, Dboost: A Fast Algorithm for DBSCAN-based Clustering on High Dimensional Data, in: Pacific-Asia Conference on Knowledge Discovery and Data Mining, Springer, 2016, pp. 245–256.

[40] T. N. Tran, K. Drab, M. Daszykowski, Revised DBSCAN algorithm to cluster data with dense adjacent clusters, Chemometrics and Intelligent Laboratory Systems 120 (2013) 92–96.

[41] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, et al., A density-based algorithm for discovering clusters in large spatial databases with noise, in: 2nd International Conference on Knowledge Discovery and Data Mining, Vol. 96, 1996, pp. 226–231.

[42] B. Krawczyk, M. Woźniak, B. Cyganek, Clustering-based ensembles for one-class classification, Information Sciences 264 (2014) 182–195.

[43] K.-A. Yoon, O.-S. Kwon, D.-H. Bae, An approach to outlier detection of software measurement data using the k-means clustering method, in: 1st International Symposium on Empirical Software Engineering and Measurement (ESEM), IEEE, 2007, pp. 443–445.

[44] L. Tarassenko, P. Hayton, N. Cerneaz, M. Brady, Novelty detection for the identification of masses in mammograms, in: 1995 Fourth International Conference on Artificial Neural Networks, IET, 1995, pp. 442–447.

[45] B. W. Silverman, Density estimation for statistics and data analysis, Vol. 26, CRC press, 1986.

[46] F. T. Liu, K. M. Ting, Z.-H. Zhou, Isolation forest, in: 8th IEEE International Conference on Data Mining, IEEE, 2008, pp. 413–422.

[47] L. Ruff, R. Vandermeulen, N. Goernitz, et al., Deep one-class classification, in: International Conference on Machine Learning, 2018, pp. 4393–4402.

[48] R. Chalapathy, A. Menon, S. Chawla, Anomaly detection using one-class neural networks, arXiv preprint arXiv:1802.06360v2 (2018).

[49] J. R. Quinlan, Induction of decision trees, Machine learning 1 (1) (1986) 81–106.

[50] J. R. Quinlan, C4. 5: programs for machine learning, San Mateo: Morgan Kaufmann, 1993.

[51] N.-B. Heidenreich, A. Schindler, S. Sperlich, Bandwidth selection for kernel density estimation: a review of fully automatic selectors, AStA Advances in Statistical Analysis 97 (4) (2013) 403–433.

[52] Q. Li, J. S. Racine, Nonparametric econometrics: theory and practice, Princeton University Press, 2007.

[53] R. Bellman, Dynamic Programming, Princeton University Press, Princeton, 1957.

[54] D. Wang, D. S. Yeung, E. C. Tsang, Structured one-class classification, IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 36 (6) (2006) 1283–1295.

[55] D. T. Nguyen, K. J. Cios, Rule-based OneClass-DS learning algorithm, Applied Soft Computing 35 (2015) 267–279.

[56] V. Fragoso, W. Scheirer, J. Hespanha, M. Turk, One-class slab support vector machine, in: 23rd International Conference on Pattern Recognition (ICPR), IEEE, 2016, pp. 420–425.

[57] S. Wang, L. Zhao, E. Zhu, J. Yin, H. Yang, Ensemble one-class extreme learning machine based on overlapping data partition, in: International Conference on Cognitive Systems and Signal Processing, Springer, 2016, pp. 408–416.

[58] M. Lichman, UCI machine learning repository (2013).
URL `http://archive.ics.uci.edu/ml`

[59] C.-C. Chang, C.-J. Lin, LIBSVM: a library for support vector machines, ACM transactions on intelligent systems and technology (TIST) 2 (3) (2011) 27.

[60] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: Machine learning in Python, Journal of Machine Learning Research 12 (2011) 2825–2830.

[61] P. Bellec, C. Chu, F. Chouinard-Decorte, Y. Benhajali, D. S. Margulies, R. C. Craddock, The Neuro Bureau ADHD-200 Preprocessed Repository, Neuroimage 144 (2017) 275–286.

[62] ADHD Institute, ADHD Epidemiology — ADHD Institute, `http://adhd-institute.com/burden-of-adhd/epidemiology/`, [Online; accessed 18-06-2017] (2017).

[63] S. Itani, M. Rossignol, F. Lecron, P. Fortemps, Towards interpretable machine learning models for diagnosis aid: A case study on attention deficit/hyperactivity disorder, PloS one 14 (4) (2019) e0215720.

[64] The Neuro Bureau, NITRC: neurobureau:AthenaPipeline - NITRC Wiki, http://www.nitrc.org/plugins/mwiki/index.php/neurobureau:AthenaPipeline, [Online; accessed 18-06-2017] (2011).

[65] G. Aguirre, E. Zarahn, M. D'esposito, The variability of human, BOLD hemodynamic responses, Neuroimage 8 (4) (1998) 360–369.

[66] N. Tzourio-Mazoyer, B. Landeau, D. Papathanassiou, F. Crivello, O. Etard, N. Delcroix, B. Mazoyer, M. Joliot, Automated anatomical labeling of activations in SPM using a macroscopic anatomical parcellation of the MNI MRI single-subject brain, Neuroimage 15 (1) (2002) 273–289.

[67] S. Itani, F. Lecron, P. Fortemps, A multi-level classification framework for multi-site medical data: Application to the ADHD-200 collection, Expert Systems with Applications 91 (2018) 36 – 45.

[68] L. Tamm, M. E. Narad, T. N. Antonini, K. M. OBrien, L. W. Hawk, J. N. Epstein, Reaction time variability in ADHD: A review, Neurotherapeutics 9 (3) (2012) 500–508.

[69] S. J. Broyd, C. Demanuele, S. Debener, S. K. Helps, C. J. James, E. J. Sonuga-Barke, Default-mode brain dysfunction in mental disorders: a systematic review, Neuroscience & biobehavioral reviews 33 (3) (2009) 279–296.

[70] S. Itani, F. Lecron, P. Fortemps, Specifics of medical data mining for diagnosis aid: A survey, Expert systems with applications 118 (2019) 300 – 314.

[71] Y. Lou, R. Caruana, J. Gehrke, G. Hooker, Accurate intelligible models with pairwise interactions, in: 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2013, pp. 623–631.