technische universität
dortmund

Master Thesis

From Data to Equations: Evolutionary Mathematical Function Learning for Interpretable Outlier Detection

MD Maruf Hossain

October 10, 2025

Supervisors:
Prof. Dr. Emmanuel Müller
Dr. Simon Klüttermann

Technical University of Dortmund
Department of Computer Science
Chair of Data Science and Data Engineering
https://ls9-www.cs.tu-dortmund.de/

Contents

1. Introduction

1.1. Motivation and Background

Anomaly detection is an essential task in many real-world domains, including fraud detection in finance [49], predictive maintenance in industrial systems [36], fault diagnosis in engineering [3], and cybersecurity [2]. Detecting anomalies enables the prevention of financial loss, early intervention in system failures, and improved reliability in safety-critical environments. However, a key challenge is to develop anomaly detection methods that achieve high performance while remaining interpretable, since many applications require transparent explanations for anomalies.

Traditional unsupervised anomaly detection approaches, such as distance-based and density-based methods [22, 5], and boundary-based methods like the One-Class SVM (OCSVM) [51] offer conceptual simplicity and some interpretability in low-dimensional spaces. However, they degrade in performance with increasing dimensionality and typically provide little insight into the mechanisms generating anomalies. Ensemble methods such as the Isolation Forest [31] and its extensions [16, 61] achieve scalability and robustness through recursive partitioning strategies, while more recent deep ensemble methods such as DEAN [21] reach state-of-the-art performance. Yet, these approaches remain largely heuristic or black-box in nature, limiting their usefulness in domains where explanation is as important as prediction.

In parallel, symbolic regression (SR) has emerged as a powerful paradigm for interpretable machine learning. SR aims to discover explicit mathematical equations that describe relationships in data [41]. Unlike traditional regression, which assumes a fixed functional form, SR simultaneously evolves both structure and parameters. Genetic programming (GP), introduced by Koza in the early 1990s [24], became the foundational method for SR and demonstrated remarkable flexibility, even rediscovering natural laws from experimental data [50]. Despite its expressiveness, GP-based SR suffers from inefficiency, excessive expression complexity [33], and limited ability to incorporate prior knowledge [11]. To address these limitations, probabilistic approaches such as Bayesian Symbolic Regression (BSR) [20] and hybrid systems like AI Feynman [56] were developed, incorporating priors and physics inspired simplifications to improve efficiency. More recently, neural guided and reinforcement learning based methods such as Deep Symbolic Regression (DSR) [37] and Unified Deep Symbolic Regression (uDSR) [26] have pushed SR benchmarks forward, but at the cost of increased computational complexity.

The intersection of symbolic regression and anomaly detection remains relatively underexplored. Domain specific applications have demonstrated promise, as seen in the combination

of symbolic regression with sparse regression to explain orbital anomalies [35], integrated with information theory for fault detection in engineering systems [48], and applied to fraud detection using symbolic classification [58]. However, these methods often rely on prior domain knowledge, labeled data, or density-based assumptions [7], limiting their applicability in general unsupervised anomaly detection.

This thesis is motivated by the importance of a general purpose, interpretable, unsupervised anomaly detection framework that balances predictive performance, scalability, and transparency. By extracting symbolic expressions directly from raw unlabelled data using evolutionary algorithms, anomalies can be detected not just as deviations in embeddings or heuristic scores but as breaches of explicit human readable equations. This approach seeks to bridge the gap between black-box performance driven anomaly detection and interpretable, equation based modeling, offering a new path from data to equations.

1.2. Thesis Structure

As a starting point, Section 2 provides the related works relevant to this thesis. It introduces symbolic regression and unsupervised anomaly detection, before discussing the existing work that combines symbolic regression with anomaly detection tasks. The section concludes with a statement of how this thesis contributes to the literature.

Section 3 presents the anomaly detection algorithms considered in this work. It begins by defining anomalies and outlining their different types, followed by a review of some algorithms, including K-Nearest Neighbor, Support Vector Machine, Isolation Forest, and DEAN. These algorithms are used as competitive algorithms because they serve as baselines for comparison with the proposed method.

Section 4 describes the methodology and toy experiment used to motivate and test the proposed method. It starts with the design rationale and data generation process, then introduces the custom loss function and optimization procedure for symbolic regression. The section reports results and insights from the toy experiment, and proceeds to describe the proposed ensemble method. This includes the motivation of the ensemble strategy, feature chunking with symbolic function learning, parameter tuning and scoring, and score aggregation. It also presents the pseudocode of the proposed *SYRAN* algorithm to provide a clear overview of its workflow and concludes with a summary of the novelty of the method.

Section 5 presents the experiments with real-world data and their outcomes. It begins with the experimental setup, where the datasets are described and the evaluation metrics are introduced. The section then examines the influence of hyperparameters, focusing on complexity, chunk size, and the number of chunks, and provides a heatmap analysis to illustrate their interactions. Performance evaluation is discussed through a detailed case study on a particular dataset, accuracy comparisons across datasets, and statistical validation using a critical difference diagram. The section also presents the extraction of symbolic equations from real-world datasets and their biomedical interpretation. Furthermore, it includes an extended analysis with

a higher number of chunks to investigate the scalability and robustness of the proposed method. The section concludes with a summary of the experimental findings.

This thesis concludes with Section 6, which summarizes the main results, discusses the contributions and limitations of the work, and outlines directions for future work.

2. Related Works

This chapter reviews existing research relevant to this thesis in three main areas: (i) symbolic regression, (ii) unsupervised anomaly detection, and (iii) the use of symbolic regression for anomaly detection. The goal is to situate the contribution of this work within the broader scientific context.

2.1. Symbolic Regression

Symbolic regression (SR) is a machine learning approach that extracts mathematical expressions from datasets to characterize the relationships between variables [41]. Unlike traditional regression, which assumes a fixed model class, SR simultaneously discovers both the structure and parameters of equations. Its potential for interpretability and scientific discovery has made it a prominent area of research over the past three decades [34].

Traditionally, symbolic regression (SR) has been closely associated with genetic programming (GP), introduced by Koza [24] in the early 1990s. In this paradigm, candidate expressions are represented as trees and evolved over generations using mutation and crossover [24]. GP-based methods are highly flexible and can rediscover natural laws from raw data, as demonstrated by Schmidt and Lipson's Eureqa system, which gained attention for automatically distilling natural laws from experimental observations [50]. However, GP approaches often suffer from inefficiency in memory usage, excessive expression complexity, which Luke and Panait [33] referred to as "bloat", and the limited mechanisms for incorporating prior knowledge [11].

To address these limitations, probabilistic approaches were introduced. Jin et al. [20] proposed Bayesian Symbolic Regression (BSR), which uses Markov Chain Monte Carlo (MCMC) sampling over symbolic expression trees. This Bayesian formulation enables the incorporation of prior knowledge, yields more concise models, and reduces memory consumption compared to conventional GP. Similar motivations underpinned hybrid methods such as AI Feynman by Udrescu and Tegmark [56], which combine neural networks with physics-inspired simplification strategies to accelerate the discovery of interpretable laws from scientific data. These works serve as a motivation for this thesis to develop a fully data-driven symbolic regression framework that mitigates excessive expression complexity and memory usage, without relying on any prior knowledge.

Another milestone was the shift toward deep learning based methods. Petersen et al. [37] introduced Deep Symbolic Regression (DSR), which leverages reinforcement learning with a risk-seeking policy gradient to explicitly optimize for high-quality expressions rather than average-

case performance. Their method outperforms classical systems such as Eureqa on standard symbolic regression benchmarks. Building further, Landajuela et al. [26] proposed a Unified Deep Symbolic Regression (uDSR) framework that integrates five paradigms into a modular system, which includes recursive problem simplification, neural-guided search, large-scale pre-training, genetic programming, and linear modeling. Evaluated on the SRBench benchmark, uDSR demonstrated strong accuracy and expression recovery, but at the cost of high computational demand and complexity.

In summary, symbolic regression has evolved from genetic programming to probabilistic, deep learning, and hybrid approaches. These advances have improved accuracy, interpretability, and benchmarking practices, yet computational efficiency, scalability, and the handling of high-dimensional data remain open challenges. Moreover, most research has focused on general equation discovery or physics-inspired modeling, leaving domain-specific applications such as unsupervised anomaly detection relatively unexplored, to the best of my knowledge, which motivates the development of this thesis.

2.2. Unsupervised Anomaly Detection

Early work on unsupervised anomaly detection was driven by distance and density based methods. Knorr et al. [22] introduced a distance-based framework, where points in sparse regions were flagged as anomalies, followed by Ramaswamy et al. [43], who proposed efficient algorithms based on k-nearest neighbors (kNN). Breunig et al. [5] extended this paradigm with the Local Outlier Factor (LOF), which detects anomalies relative to local density variations. While these approaches are intuitive and interpretable, they become computationally expensive in high-dimensional settings and are sensitive to parameter selection. This thesis addresses these challenges, such as being computationally expensive in a high-dimensional environment.

Boundary-based methods gained prominence with the introduction of the One-Class SVM (OCSVM) by Schölkopf et al. [51]. OCSVM estimates the support of the underlying data distribution by learning a function f that is positive on the support S of the data and negative elsewhere, effectively separating high-density regions of the input space from the rest. This approach generalizes the support vector framework to the case of unlabeled data. While OCSVM effectively identifies high-density regions, it does not produce a human-readable model of the data. This limitation motivates our exploration of symbolic regression for unsupervised anomaly detection, which can learn interpretable functions capturing the mechanisms generating anomalies.

A major advancement in anomaly detection came with ensemble-based approaches, particularly the Isolation Forest (iForest) introduced by Liu et al. [31]. Unlike distance or density based methods, iForest explicitly isolates anomalies by recursively partitioning the feature space using random isolation trees (iTrees). Anomalies, being few and distinct, are typically separated closer to the root of the trees, while normal points require deeper partitions. This design, combined with subsampling, allows iForest to achieve linear-time complexity with low memory require-

ments, making it suitable for large, high-dimensional datasets. Extensions such as the Extended Isolation Forest (EIF) by Hariri et al. [16] and OptIForest by Xiang et al. [61] further improved robustness to data heterogeneity and noise. EIF addresses artifacts in anomaly scores by using hyperplanes with random slopes, enhancing stability across the feature space, while OptIForest optimizes tree structures and incorporates clustering-based learning to hash, improving isolation quality. Despite these advancements, both EIF and OptIForest remain heuristic methods and do not provide explicit insights into the generative mechanisms of anomalies. This highlights a key motivation for this thesis, which is to not only improve anomaly detection performance but also ensure interpretability and robustness by handling high-dimensional features through a chunked sampling strategy.

More recently, deep ensemble methods have emerged. Klüttermann et al. [21] proposed DEAN (Deep Ensemble Anomaly Detection), which formalizes surrogate anomaly detection axioms and leverages neural representations within an ensemble framework. DEAN demonstrates state-of-the-art results across diverse domains, but like other deep methods, it inherits the challenges of limited interpretability and high computational demand.

To assess such trade-offs, Han et al. [15] introduced ADBench, a comprehensive benchmark of 30 anomaly detection algorithms evaluated across 57 real-world datasets. The study emphasizes persistent shortcomings across methods, including sensitivity to noise, difficulty in hyperparameter tuning, and lack of interpretability.

In summary, while unsupervised anomaly detection has demonstrated strong performance across multiple domains, existing methods often face limitations in interpretability, robustness, and scalability. These challenges motivate our exploration of symbolic regression for unsupervised anomaly detection as a potential solution.

2.3. Symbolic Regression for Anomaly Detection

Recent research has begun exploring the intersection of symbolic regression and anomaly detection. These approaches aim to leverage the interpretability of symbolic models while maintaining anomaly detection performance.

Manzi and Vasile [35] proposed a hybrid approach combining symbolic regression with sparse regression to reconstruct orbital anomalies. Their method explains deviations in orbital dynamics caused by unmodeled natural incidents or orbital maneuvers. However, the approach is domain-specific, relying on assumptions about orbital dynamics. This motivates the present work to develop a general-purpose, unsupervised symbolic regression framework for anomaly detection that does not rely on domain-specific assumptions.

In engineering systems, Safikou et al. [48] integrated symbolic regression with information theory for fault detection and remaining useful life estimation. While interpretable, the model depends on physical assumptions specific to engineering contexts. Similarly, Visbeek et al. [58] applied deep symbolic classification to fraud detection, discovering interpretable expressions robust to class imbalance. However, this framework focuses on labeled data and classification,

whereas anomaly detection often operates in unlabeled settings. To address this challenge, this thesis operates in unlabeled settings to extract a meaningful equation from the raw data.

Cao et al. [7] introduced a one-class anomaly detection method combining kernel density estimation (KDE) with GP-based symbolic regression. Although computationally efficient, it depends on predefined density functions and thresholding mechanisms.

In summary, symbolic regression has shown potential for anomaly detection, however, existing methods have some limitations. Some are domain-specific, relying on physical models or system-specific assumptions such as orbital dynamics or engineering sensor models. Some methods rely on labeled data, which makes them less suitable when only unlabeled data is available. Others need predefined density functions or thresholds, which can limit their use on new datasets. Overall, there remains a gap for general-purpose, unsupervised symbolic regression approaches that can directly learn interpretable mathematical functions from raw, unlabeled data.

2.4. Thesis Contribution

This thesis proposes a general-purpose, unsupervised anomaly detection framework based on symbolic regression. Unlike previous work, our method evolves symbolic expressions directly from raw unlabeled data without relying on physical models, labeled anomalies, or density-based assumptions. The approach aims to balance accuracy, interpretability, and scalability, thereby addressing key limitations identified in the existing methods.

3. Anomaly Detection Algorithms

3.1. Anomalies and Anomaly Detection

In many datasets, most data points follow some trend or pattern and can be grouped as normal behavior. However, there are some data points that do not follow this pattern. These points, which differ significantly from the normal data points, are referred to as anomalies [9].

Figure 3.1 shows an example of how normal samples or anomalies could be distributed in a dataset. It is a simple illustration of anomalies. In this figure, we can see that there are two regions of normal data points, which are denoted as N_1 and N_2. It can also be seen that there are two more data points, represented as O_1 and O_2, and a small region including some data points named as O_3. It is clear that O_1 data point, O_2 data point , and the region O_3 of some data points are far from the normal data points regions, namely, N_1 and N_2. Since they do not follow the normal behavior pattern and are far from normal data points, hence they are referred to as anomalies.

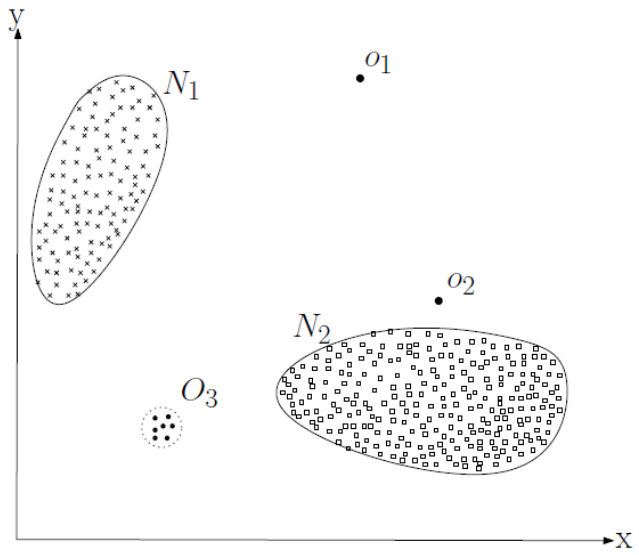


Figure 3.1.: A simple illustration of anomalies in a 2-D dataset [9]

The process of detecting these anomalies is known as anomaly detection and the methods that are used for anomaly detection are known as anomaly detection techniques. Using anomaly

detection, we identify the abnormal data from a dataset [18].

3.2. Types of Anomalies

If we would like to detect the outliers in a data set, it is very crucial that we understand the nature of anomalies. According to Chandola et al. [9], Ahmed and Mahmood [1], and Yang et al. [63], anomalies can be classified into three distinct categories, namely point anomalies, collective anomalies, and contextual anomalies.

Point Anomalies

A point anomaly is defined as an occurrence where one observation or data point differs significantly from the normal data points. It is the simplest type of anomaly and refers to a single anomalous point. A simple example of an outlier in a two-dimensional data set can be seen in Figure 3.1. In this figure, the points O_1 and O_2 are anomalies and can be denoted as point anomalies. Besides, O_3 is also an anomaly since it is far from the normal data point regions. Therefore, the points in O_3 region can also be considered as point anomalies. Let us assume a real-life example where a person uses ten litres of car fuel on a daily basis. If the usage of fuel increases to sixty litres from ten litres in a random single day, then it is considered a point anomaly.

Contextual Anomalies

When an observation or a particular data point acts as an anomaly in a specific context, then it is referred to as a contextual anomaly. It can also be denoted as a conditional anomaly.

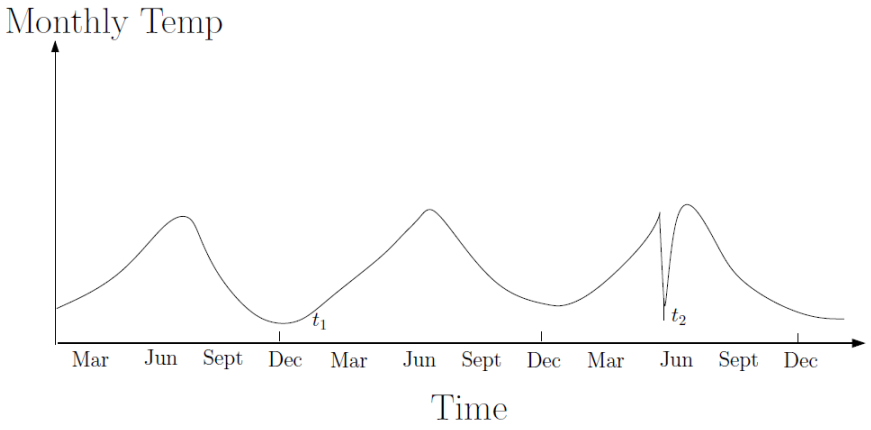


Figure 3.2.: An example of contextual anomaly [9]

Let us consider a real-life example where people are highly likely to spend more money at the biggest festivals like Christmas or New Year. During that period, if the expenditure goes high

on a credit card, it may not be considered an anomaly in this festival context, but it is considered an anomaly during the non-festive times or months.

Figure 3.2 presents a further illustration of a contextual anomaly within temperature time series, with the x-axis representing "Time" and the y-axis representing "Monthly Temperature". In x-axis, we can see the time on a monthly basis throughout the year. In general, we all know that the temperature is lower in winter and higher in summer. Lets say the temperature in December (at time t_1) is around $30°F$, which is normal in winter. But if the temperature in June (at time t_2) is also around $30°F$, which is not normal in summer, then it can be considered as a contextual anomaly.

Collective Anomalies

When a collection of observations or a group of data points acts as an anomaly with respect to the entire data set, it is denoted as a collective anomaly. It is worth noting that a single observation or a data point in a collective anomaly may not be considered as an anomaly, but a collective occurrence of them can be considered as an anomaly.

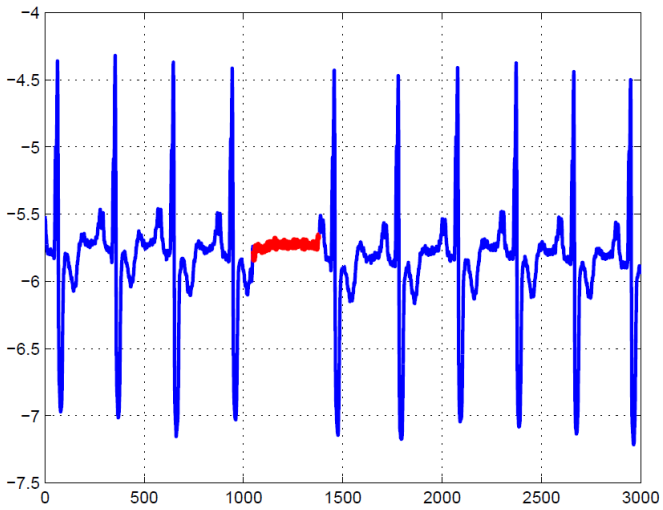


Figure 3.3.: An example of collective anomaly [9]

Figure 3.3 shows an example of collective anomaly by considering the output of a human electrocardiogram or ECG. The red-marked region in this figure refers to the anomaly. It is clear that the presence of low values for a longer time indicates the abnormalities, whereas if a single data point is low, it is not considered as an anomaly. The collective presence of observations makes it an anomaly rather than a single observation.

3.3. Competitor Algorithms

In this section, we briefly discuss several algorithms that serve as baselines for comparison with our proposed method. These competitor algorithms are widely used in anomaly detection and provide a strong foundation for performance evaluation.

3.3.1. K-Nearest Neighbor (KNN)

Traditional supervised methods for anomaly detection rely on labeled data points. In the case of unlabeled data, these methods become ineffective [32, 46]. These challenges can be effectively addressed using unsupervised anomaly detection techniques, such as K-Nearest Neighbor (KNN). Since KNN is performed using the distance of a data point to its k-nearest neighbors, based on the usage of this distance measure, KNN can be used in either distance-based or density-based ways to detect anomalies [60].

Distance-Based Anomaly Detection

In most datasets, anomalies are not higher in numbers, and typically appear in regions where they would not normally be expected. In a dataset, the group of normal data points are always remained close to each other whereas the distance is higher between anomalies and these normal data points. This idea has been used as the key concept behind the distance-based anomaly detection method. When looking at normal data points in the same neighborhood, they are very close to each other. But when it comes to anomalies, they are farther apart, and they tend to be found in areas with a low probability. Using this concept, Knorr and Ng first came up with a distance-based anomaly detection approach in 1998 [23].

There are a couple of distance based equations that are generally used in KNN to find the anomalies or distances between or among the data points. A general form of these equations is the Minkowski distance, which defines the distance of order p between two d-dimensional samples $\mathrm{x}_i = (x_{i1}, x_{i2}, \ldots, x_{id})$ and $\mathrm{x}_j = (x_{j1}, x_{j2}, \ldots, x_{jd})$ as

$$\mathrm{dist}(\mathrm{x}_i, \mathrm{x}_j) = \left(\sum_{k=1}^{d} |x_{ik} - x_{jk}|^p \right)^{\frac{1}{p}}. \tag{3.1}$$

Different distance measures can be obtained by changing the value of p in Equation (3.1). For example, setting $p = 1$ gives the Manhattan distance [13]:

$$\mathrm{dist}_{Manhattan}(\mathrm{x}_i, \mathrm{x}_j) = \sum_{k=1}^{d} \left| x_{ik} - x_{jk} \right|. \tag{3.2}$$

When $p = 2$, the formula reduces to the Euclidean distance [53]:

$$\mathrm{dist}_{Euclidean}(\mathrm{x}_i, \mathrm{x}_j) = \sqrt{\sum_{k=1}^{d} \left(x_{ik} - x_{jk}\right)^2}. \tag{3.3}$$

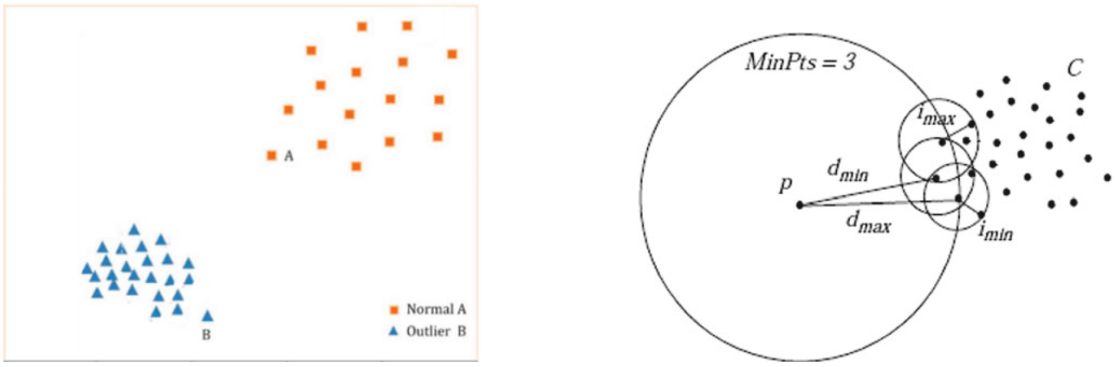
Finally, if p approaches infinity, the Chebyshev distance is obtained [39]:

$$\mathrm{dist}_{\mathrm{Chebyshev}}(\mathrm{x}_i, \mathrm{x}_j) = \max_k(|x_{ik} - x_{jk}|). \tag{3.4}$$

Let us assume a point x and its distance from its k^{th} nearest neighbor is denoted by $D^k(x)$. If we calculate the distance using one of the distance measure equations, like Manhattan distance, or Euclidean distance etc, we can then come up with a rank based on the distances. An outlier can be determined using this distance measure $D^k(x)$. The data points that have the highest distance from x can be considered as outliers. The data points that have the smallest distance from x are known to be normal data points [43].

Density-Based Anomaly Detection

In anomaly detection, we always try to find the data points that are far from the normal data points. Distance-based anomaly detection tries to find these anomalies based on the distances from the respective data point to other data points. It also works very well when the data are grouped into clusters and have a similar density. However, in real world cases, the datasets does not always follow the above pattern, hence it becomes difficult for distance based anomaly detection to detect anomalies correctly.



(3.4(a)) A local outlier

(3.4(b)) An example of local outlier factor

Figure 3.4.: Illustrations of local outlier factor [60]

A general scenario of this problem can be demonstrated in Figure 3.4(a), where it can easily be seen that data point B can not be detected as an outlier using distance based anomaly detection but it is an outlier within its local area. Hence in global point of view, B is considered as normal whereas in local area it can easily be seen as an anomaly.

In order to address this issue, a local outlier factor (LOF) was proposed in 2000 by Breunig et al. [5], where LOF is nothing but a density based outlier score for each data point, which is illustrated in Figure 3.4(b).

Consider a positive integer k and an object x. The k-distance neighorhood of this object x can be defines as:

$$N_k(x) = \{q \in D \setminus \{x\} \mid d(x,q) \leq k - \text{distance}(x)\}. \tag{3.5}$$

Consider another object $o \in D$, and if it is in the k-nearest neighborhood of x, then the local reachability distance can be defined as follows [10]:

$$\text{reach-dist}_k(x,o) = \max\{k - \text{distance}(o), d(x,o)\}. \tag{3.6}$$

Considering $|N_k(x)|$ being the cardinality of $N_k(x)$, the local reachability density of x is defined as the reciprocal of the reachability distance derived from the k-nearest neighbors.

$$lrd_k(x) = \frac{1}{\frac{\sum_{o \in N_k(x)} \text{reach\_disp}_k(o)}{|N_k(x)|}}. \tag{3.7}$$

The local outlier factor of x is determined using the following equation:

$$lof_k(x) = \frac{\sum_{o \in N_k(x)} \frac{lrd_k(o)}{lrd_k(x)}}{|N_k(x)|}. \tag{3.8}$$

The local reachability density of an object shows us how close together its points are. The density is low if the point is far away from its neighbors. The local outlier factor determines the density of a particular point or object in relation to the density of its knn. Any data point or object x is considered to be an anomaly if x is in low density area but its neighbors are in high density areas. Hence, a high value of LOF indicates a more isolated point or object whereas low LOF value refers to a data point or object that is close to its neighbors [60].

3.3.2. One Class Support Vector Machine(OC-SVM)

One-class support vector machine or OC-SVM, is an extension of the standard binary support vector machine proposed by Schölkopf et al. [51]. This method is one of the popular methods for anomaly detection. A general approach of OC-SVM is to use the non-linear kernel function to map data from the input space to the feature space. In the feature space, the vectors that have been mapped are referred to as image vectors [8, 12, 59].

Let us consider the symbol $\mathbf{\Phi}(\mathrm{x}_i)$ represents an image x_i in the feature space, where the symbol $\mathbf{\Phi}(.)$ is assumed to be unknown.

The Gaussian kernel $K(\mathrm{x}_i, \mathrm{x}_j) = \exp\left(-\|\mathrm{x}_i - \mathrm{x}_j\|^2/(2\sigma^2)\right)$, however, is an example of a kernel function that may be used to determine the inner product of the image $\mathbf{\Phi}(\mathrm{x}_i)$, where both (i, j) range from 1 to l.

By solving the following quadratic equation, a one-class SVM algorithm maps the training samples into a feature space via a kernel and identifies a hyperplane that optimally separates the data from the origin [51].

$$\min_{\mathrm{w},\rho} \quad \frac{1}{2}\mathrm{w}^T\mathrm{w} - \rho + \frac{1}{vl}\sum_{i=1}^{l}\xi_i \quad . \tag{3.9}$$

$$\text{s.t.} \quad \mathrm{w}^T\mathbf{\Phi}(\mathrm{x}_i) \geq \rho - \xi_i, \quad \xi_i \geq 0, \quad \forall i = 1, 2, \ldots, l. \tag{3.10}$$

where, i varies from 1 to l, T denotes the transpose of matrices or vectors, and the parameter $v \in (0, 1]$ establishes an upper limit and a lower limit on the number of outliers and support vectors respectively. This determines the trade-off between permitting outliers and making the model more complex.

Let us introduce positive multipliers, denoted as α_i and β_i, both of which are non-negative, meaning they are greater than or equal to 0. Using these multipliers, the Lagrangian equation can be written as follows:

$$\begin{aligned} L(\mathrm{w}, \boldsymbol{\xi}, \rho, \boldsymbol{\alpha}, \boldsymbol{\beta}) = \frac{1}{2}\mathrm{w}^T\mathrm{w} - \rho + \frac{1}{vl}\sum_{i=1}^{l}\xi_i \quad - \sum_{i=1}^{l}\alpha_i\left(\mathrm{w}^T\mathbf{\Phi}(\mathrm{x}_i) - \rho + \xi_i\right) \\ - \sum_{i=1}^{l}\beta_i\xi_i. \end{aligned} \tag{3.11}$$

By taking the derivatives of the Equation (3.10) w.r.t the variables w, ξ, $\boldsymbol{\rho}$, and setting them to zero, the resulting equations are below:

$$\mathrm{w} = \sum_{i=1}^{l}\alpha_i\mathbf{\Phi}(\mathrm{x}_i). \tag{3.12}$$

$$\alpha_i = \frac{1}{vl} - \beta_i \quad \text{and} \quad \sum_{i=1}^{l}\alpha_i = 1. \tag{3.13}$$

By putting the Equation (3.12), and Equation (3.13) into the Equation (3.11), we get the following dual form:

$$\begin{aligned} &\min_{\boldsymbol{\alpha}} \quad \boldsymbol{\alpha}^T\boldsymbol{Q}\boldsymbol{\alpha} \\ &\text{s.t.} \quad 0 \leq \alpha_i \leq \frac{1}{vl}, \qquad \sum_{i=1}^{l}\alpha_i = 1 \end{aligned} \tag{3.14}$$

where, i ranges from 1 to l, $\boldsymbol{\alpha}$ is a vector form and can also be written as $(\alpha_1, \alpha_2, ..., \alpha_l)$, and $\boldsymbol{Q}$ is the kernel matrix, which can be denoted as $Q(i, j) = K(\mathrm{x}_i, \mathrm{x}_j)$ as well.

All samples $\{\mathrm{x}_i | \alpha_i > 0\}$ are defined as support vectors where i ranges from $1, 2, ...l$ and using these support vectors (SVs), the kernel function $f(\mathrm{x})$ can be written as follows:

$$f(\mathrm{x}) = \mathrm{sgn}\left(\sum_{i \in SVs} \alpha_i K\left(\mathrm{x}_i, \mathrm{x}\right) - \rho\right). \tag{3.15}$$

The variable ρ in Equation (3.15) can be defined using the following equation:

$$\rho = \mathrm{w}^T \Phi\left(\mathrm{x}_i\right) = \sum_{j \in \left\{j | \alpha_j \neq 0\right\}} \alpha_j K\left(\mathrm{x}_i, \mathrm{x}_j\right). \tag{3.16}$$

where α_i represents a non-zero Lagrange multiplier and $\mathrm{x_i}$ denotes a sample associated with α_i.

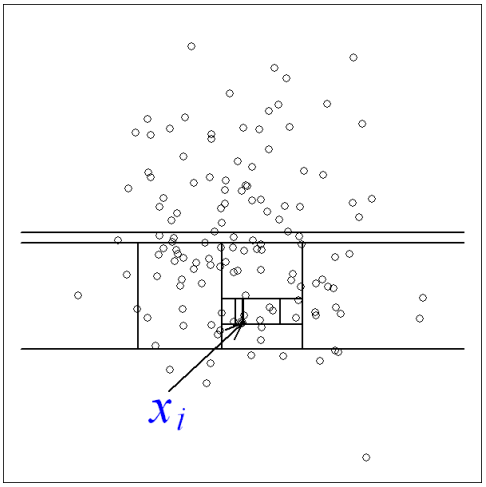
The function $f(\mathrm{x})$ returns either 1 or -1. For a given sample x, the function $f(\mathrm{x})$ will return an output of 1 if the sample is a target, otherwise it will give an output of -1, which indicates that the sample is an outlier.

3.3.3. Isolation forest (Iforest)

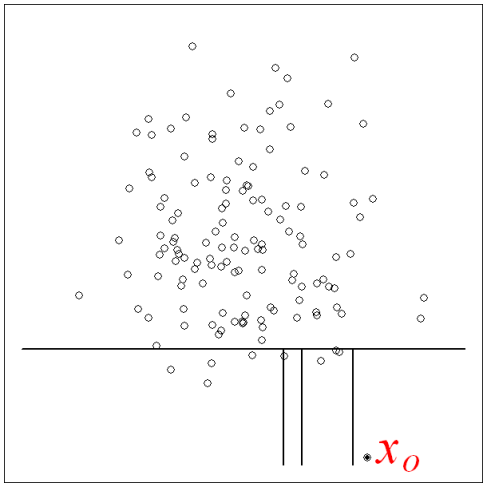
The idea of isolating denotes the process of segregating an object or instance from other objects or instances. Isolation forest is an unsupervised anomaly detection algorithm that detects outliers by isolating them from the ensemble of binary trees by generating multiple isolation trees, and anomalies are identified by their respective anomaly scores and smaller path lengths.

The idea behind the isolation is that, in general, the number of anomaly data points is few or not high in numbers, and they usually requires a shorter path lengths in the forest of collective random trees [30].

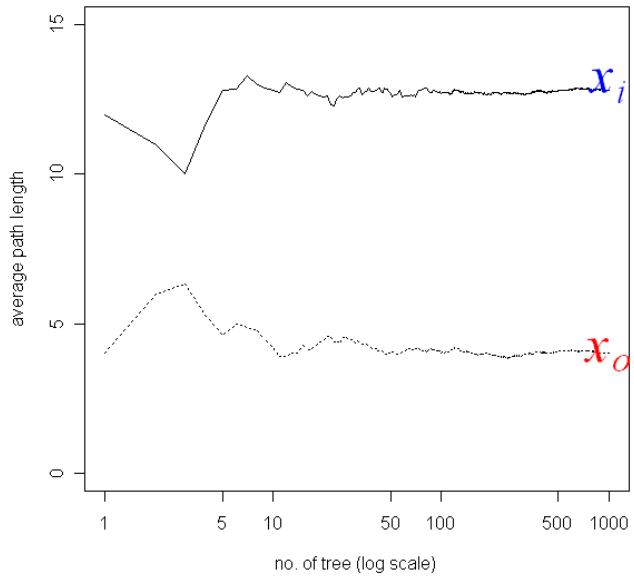
Figures 3.5(a) and 3.5(b), depicts the idea of isolation. Assuming x_i and x_o as normal data point and anomaly, respectively. It can clearly be seen that to isolate x_i, the path length is higher, whereas the path length is shorter while isolating x_o. In this given example, partitions are created through the random selection of an attribute, followed by the random selection of a split value within the range defined by the maximum and minimum values of the chosen attribute. The illustration of recursive partitioning as a tree structure signifies that the amount of partitions required to isolate a point is equivalent to the path length from the root node to a leaf node. Here it can be seen that the path for x_i is higher than the path length of x_o.



(3.5(a)) Isolating x_i



(3.5(b)) Isolating x_o



(3.5(c)) Average path lengths coverage

Figure 3.5.: Combined figures illustrating isolation forest concepts [31]

The individual trees have different partition sets as each of these partitions are generated randomly. The path lengths over a number of trees are averaged to determine the expected path length. Figure 3.5(c) illustrates the concept of average path length, and we can say that while increasing the number of trees, the average path length for x_i and x_o converge at some point.

Isolation Trees

Let us consider that T is a node in an isolation tree. It can be a leaf, which means it doesn't have any child node, or it can be an internal node, which means it does have child node. Considering T_l as left child node, and T_r as right child node, T can be split into two parts. Let us consider a feature q and a split value p. Based on q and p, the internal node T divides into two parts and the condition $q < p$ is used to decide whether a data point should go towards the left child node T_l or the right child node T_r [62].

Let us assume a sample X with n data points $x_1, \ldots, x_n$ from a d-variate distribution [27]. An isolation tree (iTree) is constructed using these data points by recursively dividing the sample through random selection of a feature q and a split value p. The recursive partitioning process continues until one of the stopping conditions is reached, namely when only a single data point remains in a node ($|X| = 1$), when the tree reaches a maximum depth or height limit, or when all data points in a node have identical values such that further splitting is not possible.

Each node in an iTree consists of either zero or two child nodes which indicates that it is a proper binary tree. If we assume that all the data points are distinct, then when an iTree is fully grown, each data point ends up in its leaf node, where the number of leaf nodes, internal nodes, and total number of nodes are n, $n-1$, and $2n-1$, respectively. Therefore, it is evident that the memory usage increases in a linear manner as n increases.

Now, in order to detect anomalies, we can proceed in a way that all the data points are sorted based on either their path lengths or anomaly scores. In general, anomaly points are shorter in path length, hence making them top of the ranked list.

Assuming a data point x and its corresponding path length $h(x)$. In an iTree, x traverses a specific number of edges, beginning at the root node and progressing to the leaf node. The total number of edges that x traverses is known as the path length of x. Using $h(x)$, we have to derive the anomaly score so that we may detect the anomalies.

Isolation Tree (iTree)	**Binary Search Tree (BST)**
Constructed as proper binary trees	Constructed as proper binary trees
Traversal ends at the leaf node	Traversal stops when the search fails
No concept of key matching (not applicable)	Traversal may end in a successful search

Table 3.1.: List of similarities in Isolation Trees and Binary Search Trees

In Table 3.1, it can easily be seen that there are some similarities in iTree and BST in terms of structure. In the case of iTree, traversal ends at the leaf node, whereas it ends in an unsuccessful search in the BST. Therefore, we may say that the unsuccessful search in the BST is equivalent to the average $h(x)$ in iTree, and to estimate it, the analysis from the BST can be used. Assuming a dataset consisting of n data points, the average path length of unsuccessful searches in the BST can be determined using the following equation from Section 10.3.3 [40].

$$c(n) = 2H(n-1) - (2(n-1)/n). \tag{3.17}$$

here $H(i)$ denotes the harmonic number and can be determined using $\ln(i)+$ Eulers constant. Assuming s as the anomaly score, which can be calculated for an instance x by using the following formula:

$$s(x, n) = 2^{-\frac{E(h(x))}{c(n)}}. \tag{3.18}$$

where $E(h(x))$ indicates the average of $h(x)$. From Equation (3.18), different outcomes can be observed. When $E(h(x))$ tends to $c(n)$, the anomaly score s approaches 0.5. When $E(h(x))$ approaches 0, the anomaly score s moves toward 1. Finally, when $E(h(x))$ tends to $n-1$, the anomaly score s gets closer to 0.

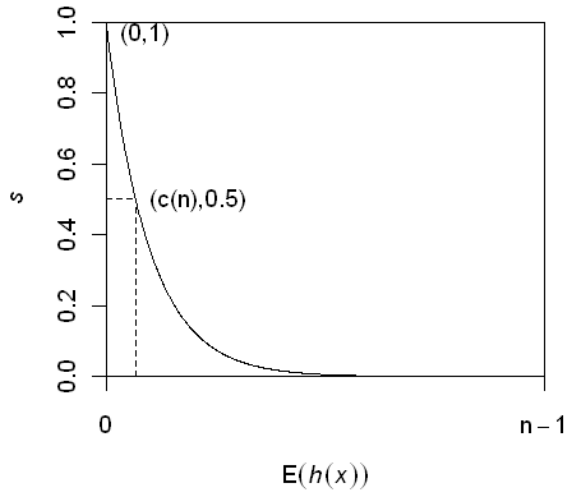


Figure 3.6.: Relationship between $E(h(x))$ and s [31]

Figure 3.6 shows the relationship between the expected path length $E(h(x))$ and the anomaly score s where for the values of $0 < E(h(x)) \leq (n-1)$, we get $0 < s \leq 1$.

Since the anomaly score s is obtained from Equation 3.18, it can be interpreted in the following way. When the anomaly score is around 1, it indicates the presence of anomalies. When the score is close to 0 or smaller than 0.5, it corresponds to normal data points. When the score is approximately 0.5, it indicates that there are no clear or distinct anomalies.

3.3.4. DEAN

Deep Ensemble Anomaly Detection, which is also known as DEAN, is an ensemble-based anomaly detection method that combines many simple deep neural networks into an ensemble in order to improve performance, as well as robustness, interpretability, and scalability [21]. This method

suggests some attributes, namely, Scalability, Depth, Variability, and Consistency, and focuses on them in order to find a deep ensemble method.

The loss function used in DEAN method is defined as follows:

$$\mathcal{L}_{\text{DEAN}} = (f(x_{\text{train}}) - 1)^2 \tag{3.19}$$

where each model is trained to get a constant output value of 1. The idea behind this approach is to detect the outliers. If the function $f(x)$ returns an output value which is close to 1, meaning $f(x) \approx 1$, then the samples should be considered as normal, whereas if the output of $f(x)$ deviates far from 1, then the samples are considered as anomalies [21].

The outlier score for each sample is calculated using the following equation:

$$\text{Score} = |(f(x_{test}) - q)| \tag{3.20}$$

where q indicates the mean value of $f(x_{\text{train}})$.

During the optimization process, DEAN uses relu as an activation function even though it may sometimes lead to certain issues with vanishing gradients. In order to address these issues, it is also suggested to use elu activation function. Additionally, to get a more stable and meaningful output, no activation function is used in the final layer.

DEAN uses the feature bagging technique to train multiple submodels with different initializations. The goal is not to optimize each submodel in order to get a higher accuracy, instead, it focuses on the ensemble's collective behavior of such submodels that can eventually perform better.

In order to get the ensemble score, the following equation is used:

$$F(x) = \frac{1}{n} \sqrt[Power]{\sum_{i=1}^{n} f_i^{Power}(x)} \tag{3.21}$$

which averages the score of each model. If the score is higher, it means that the deviation from the normal behavior is also higher. Therefore, it has a higher possibility of being an anomaly.

3.3.5. Additional Competitor Algorithms

Along with these four algorithms mentioned above, we compare our method against 11 additional competitor algorithms. These Include LOF [5], COF [55], DeepSVDD [47], PCA [6], HBOS [14], COPOD [28], ECOD [29], CBLOF [17], LODA [38], DAGMM [64], and SOD [25].

4. Methodology and Toy Experiment

We introduce a controlled toy experiment to establish an appropriate basis for the symbolic regression approach proposed in this thesis. This experiment aims to evaluate whether the proposed method could successfully rediscover a known mathematical relationship from synthetic data using an evolutionary search strategy.

4.1. Motivation and Design

Symbolic regression is most suitable for tasks that involve extracting an interpretable and simple functional relationship from data. In real-world scenarios, the ground truth is not always known, making it hard to tell if a learned function is correct. To address this issue and ensure our method would work, we created a toy experiment in which the underlying function is predetermined but hidden from the optimizer.

We introduced two continuous variables, a and b, in this toy experiment, where b is defined deterministically as a function of a:

$$b = a^2. \tag{4.1}$$

This relationship is suitable for an initial test of our method, since it is simple yet non-linear. This equation has been instructed to search for a function $f(a, b)$ such that:

$$f(a, b) \approx 1. \tag{4.2}$$

We keep the underlying assumption that if the algorithm identifies the function:

$$f(a, b) = \frac{b}{a^2}, \tag{4.3}$$

then the desired condition $f(a, b) = 1$ will hold true whenever $b = a^2$. In order to find a normalized function that evaluates to a constant value, we simplify the goal of finding the hidden dependency.

4.2. Data Generation

To create a consistent and informative dataset, we sampled 100 linearly spaced values for a from the interval $[-5, 5]$. We used the quadratic equation $b = a^2$ to find the value of b for each value of a. This structured dataset was the main dataset for training.

Together with these structured pairings, we also produced random points $(a_{\text{rand}}, b_{\text{rand}})$ that were independent of the structured function and evenly sampled from the same range. In order to evaluate whether the determined function truly depends on both variables in a meaningful way and to keep the process separate from converging to trivial constant functions, this random sample was included.

4.3. Loss Function Design

The custom loss function is an important part of this experiment, which incorporates three essential goals: (1) how close the output is to the target value; (2) how well it works with data other than training data; and (3) how easy it is to understand the learned function.

The loss function is defined as:

$$\text{Loss} = \frac{L_1}{|L_1 + L_2|} + 0.1 \cdot \log(1 + \log(1 + \text{complexity})). \tag{4.4}$$

Where:

- $L_1 = \frac{1}{N} \sum_{i=1}^{N} |f(a_i, b_i) - 1|$, tells us how far the function is from 1 on average across the whole dataset.
- $L_2 = -\frac{1}{N} \sum_{i=1}^{N} |f(a_{\text{rand}}, b_{\text{rand}}) - 1|$ penalizes the function if it behaves similarly on random, potentially incorrect data. This enables finding non-trivial functions that are not constant across the domain.
- The complexity term allows more interpretable solutions by penalizing too deep or large expressions. In this case, the depth of the expression tree and the number of operations are used to determine complexity.

This loss structure ensures the optimization of functions, achieving accurate results on known data while maintaining simplicity and generalizability beyond the training samples.

4.4. Optimization Process

We used an evolutionary algorithm, a search method based on natural selection, to find the best-fitting function. The algorithm incorporates a group of randomly generated candidate functions instead of trying to make the perfect function from the beginning. Over many iterations, these functions get better through two key operations: crossover, which takes parts of two functions and makes new ones, and mutation, which makes small random changes. At the end of each round, the most probable candidates, those who do best on a set of evaluation criteria, are chosen to move on to the next round. This process will continue until a function that is accurate enough and easy to understand is found.

At the beginning of the search process, the algorithm generated a random set of candidate functions using a predefined library of mathematical operations. This library includes basic mathematical operations, including addition and multiplication, trigonometric functions such as sine and cosine, and additional nonlinear components. In each iteration, two existing population functions were selected and combined via crossover or modified through mutation to generate new candidate expressions. Each new function is evaluated using the loss function, with the most effective ones being retained or modified to generate the next round of candidates, thus enabling the population to evolve towards improved solutions gradually.

The search process was conducted for $30,000$ iterations, selected based on empirical findings, to establish a compromise between convergence and maintaining an acceptable computational cost. We observed the error values and complexity of the candidate functions at every stage of the optimization.

4.5. Results and Interpretation

At the end of the optimization process, the algorithm identified the following function:

$$f(a, b) = \frac{b}{a^2}. \tag{4.5}$$

This expression returns a value of 1 whenever the original data relationship $b = a^2$ holds, which confirms that the underlying pattern was correctly discovered. The function achieved one of the lowest loss values during the search while maintaining a minimal structural complexity, making it accurate and simple to read.

This result is important for several reasons. First, it demonstrates that the symbolic regression system can successfully extract a meaningful and correct mathematical equation from data even without prior knowledge of the form of the function. Second, we can see that there is a good trade-off between how accurate the loss function is and how easy it is to interpret. Finally, it illustrates how well-designed evolutionary approaches can be a strong tool for finding accurate and interpretable models.

4.6. Conclusion and Insights

The toy experiment illustrates the successful implementation of the symbolic regression approach. This indicates that evolutionary optimization, combined with an efficient loss function, can effectively identify mathematical functions from accurate and interpretable data.

Though the configuration was kept intentionally simple, it highlighted some key features of how the model operates, such as avoiding constant solutions, preferring simpler expressions, and finding the appropriate equation structure. These results provide confidence that the approach is feasible and beneficial for more complex problems, which will be addressed in the following sections of this thesis.

4.7. Proposed Ensemble Method (SYRAN)

In the previous section, we demonstrated a toy experiment in order to evaluate whether our proposed *SYmbolic Regression for unsupervised ANomaly detection (SYRAN)* method can discover a mathematical relationship from two-dimensional synthetic data. Since it worked successfully with the toy experiment, we now extend this similar approach to more dimensional real-world benchmark datasets.

The objective is to extend our method to higher-dimensional data and derive a mathematical equation for anomaly detection while maintaining the key advantages of simplicity, interpretability, and generalizability. To achieve our objective, we employ an ensemble strategy that involves subdividing high-dimensional data into smaller chunks. This allows us to apply our symbolic regression approach on low-dimensional chunks and then combine the results to detect outliers in a reliable and scalable way.

4.7.1. Motivation of the Ensemble Strategy

As the number of features increases, the search space for possible equations also becomes large, hence making it very difficult to find accurate and interpretable models. Therefore, it is not feasible to apply the symbolic regression directly to the higher-dimensional data. To address this challenge, we adopt a strategy inspired by ensemble learning methods, which divide the high-dimensional data into smaller chunks and combine the results to improve performance. In particular, our approach is inspired by the DEAN method [21], which also employs ensembles to improve anomaly detection performance. Our main idea is straightforward. Since our method worked successfully on two features, we can implement it to a higher number of features just by simply dividing them into smaller chunks, where each chunk will be treated independently. We will then apply the symbolic regression to learn a small mathematical function that captures a local pattern, and combine all these small models to form an overall anomaly score.

4.7.2. Feature Chunking and Symbolic Function Learning

We begin the process by dividing the entire set of input features into multiple random *feature chunks*, each of which contains a small number of features (for example, three features per chunk). Let's assume a dataset with N number of features, which is divided into M number of chunks. For simplicity, we assume each chunk has an equal size that contains $k = \frac{N}{M}$ features, and the data in each chunk mostly corresponds to a consistent but hidden pattern, similar to the toy experiment.

Earlier in our toy experiment, we discovered a function $f(a, b) \approx 1$. Similar to that, in our current ensemble technique for high-dimensional data, each chunk is used to train one symbolic regression model. This gives us the opportunity to focus on simpler relationships within a small group of features, hence making the whole learning process simpler and easier to understand.

We apply a symbolic regression for each chunk to learn a function $f(x_1, x_2, \ldots, x_k) \approx 1$ that,

on normal (non-anomalous) training data, provides output values that are approximately 1. The notation $x_1, x_2, \ldots, x_k$ denotes that a chunk has a total of k features. We apply the same custom loss function as indicated in Equation (4.4). This loss function in Equation (4.4) also ensures that each of the learned functions is accurate, simple, and easy to understand.

4.7.3. Alpha Parameter Tuning and Scoring

For each chunk with k number of features, a symbolic function $f(x_1, x_2, \ldots, x_k)$ is learned and for each function, we evaluate new test sample data by determining how much the output of the function deviates from 1. The absolute deviation $|f(x) - 1|$ allows us to find the anomaly by simply measuring the deviation.

We need to convert this deviation into a probabilistic anomaly score, which is determined by the following equation:

$$s = \sigma\left(\alpha \cdot |f(x) - 1|\right) \tag{4.6}$$

where $\sigma(z)$ is defined using the equation $\frac{1}{1+e^{-z}}$. This sigmoid function is scaled by a factor α, and is calculated using the following equation:

$$\alpha = \frac{1}{\mathbb{E}_{x \sim \text{train}}[|f(x) - 1|]} \tag{4.7}$$

meaning the average deviation over the training data. This scaling factor α plays an important role by preventing the sigmoid function from saturating, which could otherwise result in all values collapsing to 0 or 1. By normalizing with the expected deviation, the scores are kept well distributed, ensuring that the anomaly measure remains informative. This transformation converts the raw deviations into the scores in the interval $(0, 1)$, where values close to 1 indicate more normal behaviour and values close to 0 indicate potential anomalies.

4.7.4. Score Aggregation

Each learned symbolic function produces an anomaly score. If there are M number of chunks, then there would be M number of anomaly scores. To aggregate the predictions from multiple chunks, we simply average the scores across all models. The score aggregation is calculated using the following equation:

$$S = \frac{1}{M} \sum_{i=1}^{M} s_i \tag{4.8}$$

where s_i denotes the anomaly score from the i-th chunk and M denotes the total number of chunks.

This ensemble approach enhances robustness by capturing a variety of patterns across multiple feature chunks, as well as preventing overfitting to specific data points and reducing noise. We also observe the cumulative average ROC AUC scores across chunks to assess the improvement of performance with the addition of more symbolic models in the ensemble.

4.7.5. Pseudocode of SYRAN

To clearly present the working mechanism of the proposed approach, we provide the pseudocode of *SYRAN* in Algorithm 1. The method operates by dividing the feature space into smaller chunks, applying symbolic regression within each chunk to learn expressions, and then aggregating the results to compute anomaly scores.

Algorithm 1 SYRAN: SYmbolic Regression for unsupervised ANomaly detection

Require: training data x_{train} (assumed normal), test data x_{test} (unlabeled)
Require: hyperparameters: complexity $\in \{0.01, 0.05, 0.10, 0.20, 0.30, 0.40\}$, chunk size $\in \{3, 4, 5, 6, 7\}$, number of chunks $\in \{5, 10, 15, 20, 25, 30\}$, iterations $= 30000$
Ensure: anomaly scores prediction and ROC–AUC performance

$(x_{train}, x_{test}) \leftarrow \textsc{Normalize}(x_{train}, x_{test})$
Partition feature space into num_chunk subsets of size chunk_size
for $i = 1$ to num_chunk **do**
    Initialize random symbolic expression f_i
    for $t = 1$ to 30000 **do**
        Evaluate f_i with loss = reconstruction error + complexity penalty
        Update f_i with mutation and crossover
    end for
    Compute scaling factor: $\alpha_i \leftarrow \dfrac{1}{\mathbb{E}_{x \in x_{train}}[\,|f_i(x) - 1|\,]}$
    Compute anomaly score for each test instance x: $s_i(x) \leftarrow \sigma\big(\alpha_i \cdot |f_i(x) - 1|\big)$
end for
Aggregate scores into final prediction: $S(x) \leftarrow \dfrac{1}{\text{num\_chunk}} \sum_{i=1}^{\text{num\_chunk}} s_i(x)$
return prediction $S(x)$

In the above pseudocode, $\sigma(z) = \dfrac{1}{1 + e^{-z}}$ denotes the sigmoid function, which normalizes the chunk-wise anomaly scores before aggregation. The algorithm highlights the role of the three hyperparameters, namely, complexity, chunk size, and number of chunks, along with the iterative search process of $30,000$ iterations for symbolic function discovery. This step-by-step formalization provides a transparent view of how anomaly scores are generated.

4.7.6. Contribution of SYRAN

SYRAN allows us to extend the symbolic regression from the dataset with two features to a complex and real world dataset with a higher number of features. These features are subdivided into smaller chunks. Each symbolic model produces an anomaly score and all the scores from each chunk are aggregated using the ensemble technique which ensures robustness, interpretability and more accurate predictions. This approach allows us to directly extend the success of the toy experiment into real-world problems for anomaly detection.

5. Experiments with Real-World Data

5.1. Experimental Setup

5.1.1. Dataset Descriptions

This research includes 15 datasets that are publicly available to evaluate the performance of the proposed method. They are drawn from the ADBench benchmark suite [15], which provides a comprehensive collection of anomaly detection datasets, and chosen to represent a range of real-world scenarios across various domains, including healthcare, biochemical, and categorical classification problems. To ensure computational feasibility, we chose only datasets with no more than 30 features and that could complete each set of hyperparameter evaluations within a reasonable time limit (approximately 40 hours). Each dataset consists of normal and anomalous instances, allowing for an in-depth model performance analysis, and is provided in `.npz` format, containing a training feature set (x), a test feature set (tx), and the corresponding test labels (ty). Out of these fifteen datasets, we explain five in detail below to provide an overview of their properties and relevance.

breastw

The *breastw* dataset comprises 478 test samples and 205 training samples, each defined by nine numerical features. These features are extracted from images of fine needle aspirates of breast tissue and describe characteristics such as texture and shape. The dataset is important to biomedical research and is frequently used in binary classification tasks that differentiate between benign and malignant cases.

Lymphography

The *Lymphography* dataset comprises 136 training samples and 12 test samples, each described by 18 categorical features. It is applicable to the diagnosis of diseases that impact the lymphatic system. Due to its categorical nature and small size, this dataset is valuable for evaluating models on rare class detection and classification tasks in discrete feature spaces.

wine

The *wine* dataset consists of 109 training samples and 20 test samples, each of which is defined by 13 features. Chemical measurements of various wine types, such as acidity, alcohol content, and

other physical or chemical properties, determine the features. This data set is frequently utilized in classification tasks concerning wine quality or wine type. It is suitable for evaluating baseline models and discovering their behavior on more straightforward datasets due to its smaller size and lower dimensionality.

cardio

The *cardio* dataset has $1,479$ training samples and 352 test examples. Each sample is described by 21 features. This dataset is often used to predict whether someone has or is likely to get heart-related diseases, and these features are linked to indicators of cardiovascular health. Since it is larger and has more dimensions than wine, this dataset makes a more difficult and realistic classification task.

WBC

The *WBC* dataset consists of 203 training samples and 20 test samples, each of which is defined by 9 features. It is relevant to biomedical applications such as cell classification and hematological diagnosis. This data set allows us to evaluate the performance of the model in smaller datasets with fewer characteristics.

Additional Datasets

Along with these five datasets mentioned above, we evaluate our method on 10 additional datasets. These Include *Hepatitis*, *yeast*, *vowels*, *Stamps*, *Pima*, *PageBlocks*, *pendigits*, *glass*, *cover*, and *Waveform*.

5.1.2. Evaluation Metrics

To measure performance criteria such as accuracy, ROC score etc, a confusion matrix is very useful. A confusion matrix consists of four cells, namely, true positive (TP), false negative (FN), true negative (TN), and false positive (FP) [52]. TP indicates the number of predictions where the model correctly predicts the positive class. TN represents the number of predictions where the model correctly predicts the negative class. FP is the number of predictions in which the model predicts a class as positive when the class is actually negative. FN denotes the number of predictions where the model predicts a class as negative when the actual class is positive [52].

Accuracy, true positive rate (TPR), and false positive rate (FPR) can be calculated using the following formulas [52]:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}. \tag{5.1}$$

$$TPR = \frac{TP}{TP + FN}. \tag{5.2}$$

$$FPR = \frac{FP}{TN + FP}. \tag{5.3}$$

Although accuracy is widely used as a performance measure, it is less informative in anomaly detection since anomalies are typically rare, and accuracy can be biased by the majority (normal) class. For this reason, we do not rely on accuracy in this work. Instead, we focus on the receiver operating characteristic (ROC) score, also known as the area under the curve (AUC), which is a popular metric for evaluating the performance of anomaly detection and classification models. The ROC curve is obtained by plotting the true positive rate (TPR) against the false positive rate (FPR) across different decision thresholds. The ROC score measures the area under the curve, with higher values indicating stronger discriminative performance. The ROC score lies between 0 and 1, where a value of 0.5 corresponds to random guessing, values below 0.5 indicate performance worse than random guessing, and a value of 1 represents perfect classification ability [4].

5.2. Hyperparameter Understanding

This section focuses on tuning the three hyperparameters employed in this study, namely complexity, chunk size, and number of chunks (`num_chunk`). These hyperparameters define the behavior of the proposed method as illustrated in Algorithm 1. Specifically, the complexity parameter controls the penalty on the symbolic expressions to balance accuracy and interpretability, the chunk size determines the number of features included in each subset or chunk, and the number of chunks specifies how many such subsets are created. For each parameter, multiple candidate values were considered: complexity $\in 0.05, 0.01, 0.10, 0.20, 0.30, 0.40$, chunk size $\in 3, 4, 5, 6, 7$, and number of chunks $\in 5, 10, 15, 20, 25, 30$. The Cartesian product of these sets yields a total of 180 distinct hyperparameter configurations. For example, for the complexity parameter, each of the six values is evaluated in combination with five chunk sizes and six numbers of chunks ($6 \times 5 \times 6 = 180$). Similarly, chunk size and number of chunks are each tested under the same 180 different configurations.

For each configuration, the method was evaluated on 15 datasets from the AdBench benchmark [15]. The performance metric used for hyperparameter selection was the average ROC score across all datasets. To determine the best setting for each hyperparameter, results were aggregated by fixing one parameter value and averaging over all combinations of the remaining two. The parameter value that achieved the highest average ROC score was selected.

5.2.1. Complexity

The complexity parameter plays an important role in shaping the symbolic expressions generated by the algorithm, directly influencing both interpretability and predictive performance. Lower weights of the complexity parameter allow the model to produce highly elaborate expressions consisting of multiple layers of exponentials, trigonometric functions, and composite structures. While such equations may capture subtle nonlinearities, they are often difficult to

interpret. Increasing the weight of this parameter gradually penalizes such complexity, forcing the model to generate shorter and more human-readable equations. This behaviour highlights the trade-off between accuracy gains that may arise from complex expressions and the clarity of simpler, interpretable equations.

To illustrate this trade-off, we analyze the *breastw* dataset and report the symbolic expressions obtained at different settings of the complexity parameter (Table 5.1). At very low weights (e.g., 0.01 or 0.05), the algorithm produces highly complex nested expressions such as $(\exp(\text{auto}_{00})+g)^c$, which are expressive but difficult to understand. With moderate weights (0.10 or 0.20), the expressions become more compact and manageable, while still retaining informative structure (e.g., $f^{\exp(h)}$ or a^{3f}). At high weights (0.30 or 0.40), the symbolic regression turns into extremely simple forms, such as f or b, which are easy to interpret but overly simplistic.

Complexity Weight	Equation 1	Equation 2
0.01	$(\exp(\text{auto}_{00}) + g)^c$	$a^{\cos(d)/c^2}$
0.05	$e^{f^{(c-\text{auto}_1)}}$	$i^{\exp(g)}$
0.10	$f^{\exp(h)}$	a^{3f}
0.20	$a^{\exp(c)}$	$g^{\exp(i)}$
0.30	b^3	0.9999^d
0.40	f	b

Table 5.1.: Effect of complexity parameter on symbolic expressions on the *breastw* dataset.

In addition to interpretability, we also measured predictive performance across all datasets to determine the impact of the complexity parameter on anomaly detection accuracy. For each of the six tested complexity values, the method was performed under all combinations of chunk size and number of chunks, yielding $5 \times 6 = 30$ configurations per complexity level. Each configuration was evaluated on 15 datasets from the AdBench benchmark [15] using the ROC AUC as the performance metric. The resulting scores were then averaged first across all datasets and then across all configurations to obtain the mean ROC AUC per complexity setting.

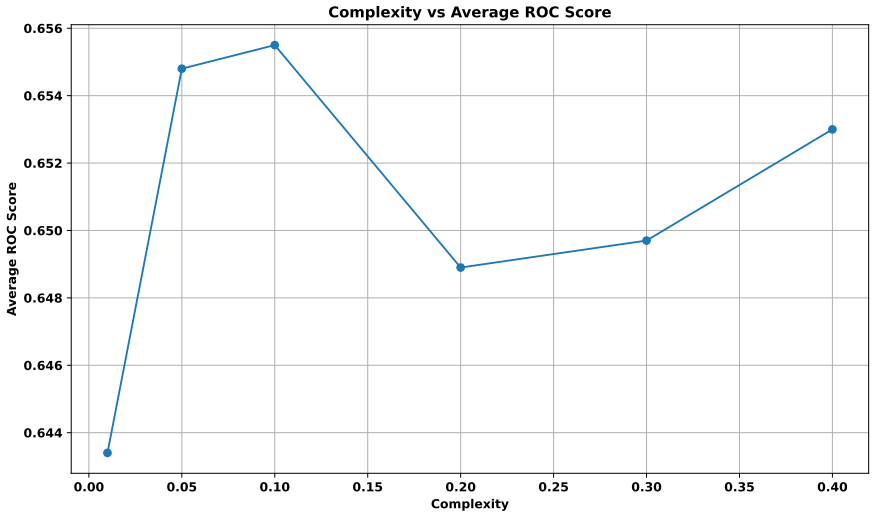


Figure 5.1.: Complexity vs ROC Scores for all datasets.

Figure 5.1 shows the relationship between the complexity parameter and the average ROC AUC scores. The results indicate that the average ROC AUC ranges only from about 0.643 to 0.656 across all complexity values, suggesting that the parameter has minimal effect on predictive performance. Performance is relatively stable, and differences across settings fall within the variability observed across datasets. Thus, while 0.10 shows the numerically highest mean score (0.655), this improvement is not dramatic compared to other values. At the same time, the highest weight (0.40) still provides similarly high scores, but the resulting equations are overly simplistic, limiting their explanatory power.

Overall, this analysis shows that the complexity parameter regulates the balance between interpretability and performance. Very low values encourage overly complicated expressions that are hard to read, whereas very high values collapse the symbolic models into trivial forms with limited insight. Based on both interpretability and performance, a middle-range complexity value of **0.10** was selected as the optimal setting for subsequent experiments. However, it should be noted that predictive performance was relatively stable across the tested range, so this choice is guided more by the interpretability–complexity trade-off than by substantial differences in ROC AUC. Nevertheless, it is important to acknowledge a limitation of our method. While the parameter effectively controls symbolic complexity, it does not guarantee that the resulting expressions at the extremes are meaningful. At high weights, the method risks producing equations that, although interpretable, may not capture informative relationships.

5.2.2. Chunk Size

The chunk size parameter was tuned over five candidate values, with each value corresponding to 36 different combinations with complexity and number of chunks.

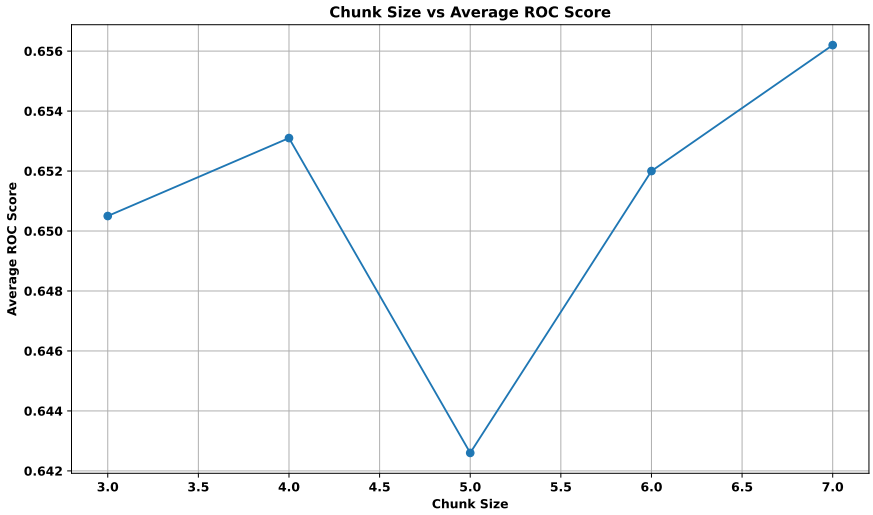


Figure 5.2.: Chunk Size vs ROC Scores for all datasets.

Figure 5.2 presents the relationship between chunk size and the average ROC score across the 15 datasets. The figure shows that after the initial increase, the average ROC score drops at chunk size 5 but then increases again, reaching its highest value at chunk size 7 with an average ROC AUC of 0.655. Consequently, the chunk size 7 was selected as the optimal setting for subsequent experiments.

5.2.3. Number of Chunks

The number of chunks was varied across six values, ranging from 5 to 30 in increments of 5. Each value corresponds to 30 different combinations with chunk size and complexity.

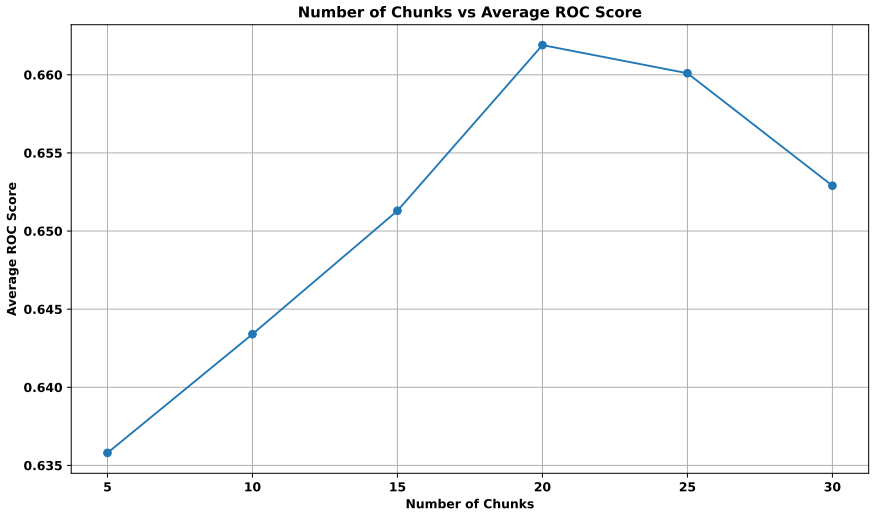


Figure 5.3.: Number of Chunks vs ROC Scores for all datasets.

Figure 5.3 shows the relationship between the number of chunks and the corresponding average ROC score. The results indicate that performance improves steadily up to 20 chunks, where it reaches its peak value of approximately 0.662. Beyond this point, performance declines, with the score decreasing to around 0.653 at 30 chunks. Therefore, the optimal number of chunks selected for subsequent experiments is **20**.

5.2.4. Heatmap Analysis of Hyperparameters

Fixed Chunk Size

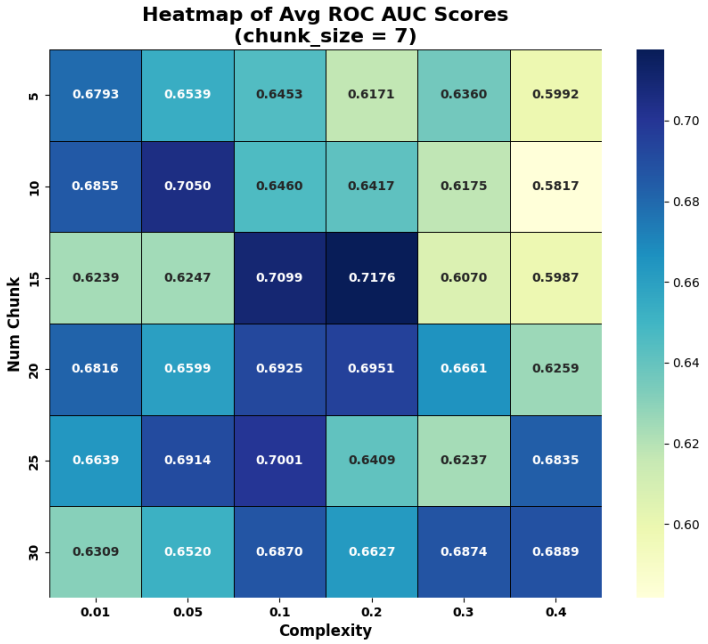


Figure 5.4.: Heatmap of Average ROC AUC Scores Across Number of Chunks and Complexity with Chunk Size 7.

Figure 5.4 presents the heatmap of average ROC scores for different combinations of the number of chunks and complexity, with the chunk size fixed at 7. The color intensity in the heatmap indicates the magnitude of average ROC scores for each combinations, with darker shades representing higher ROC scores. From the heatmap, we can see that the model performance varies across both the number of chunks and complexity levels. Overall, the scores range between 0.58 and 0.72, indicating moderate discriminative performance across settings.

Figure 5.4 shows that when complexity values are low $(0.01 - 0.05)$, it results in stable but only moderately high performance, while increasing complexity values to $(0.1 - 0.2)$ improves the results considerably, particularly when the number of chunks is set between 15 and 20. On the other hand, with high complexity values $(0.3 - 0.4)$, we can see a decrease in the performance, likely due to overfitting or decreased generalization capability.

A similar trend is observed for the number of chunks. Using very few chunks results in lower ROC scores, whereas increasing the number of chunks to $(15 - 20)$ provides stronger and more consistent performance, particularly at moderate complexity. Increasing the number of chunks beyond 25 shows stable but not significantly higher performance, indicating comparatively lower performance from further increasing this parameter.

The highest ROC score of 0.7176 is achieved with 15 chunks and a complexity of 0.2, indicating that optimal performance can be obtained by having balanced moderate settings of com-

plexity and chunk number, rather than extreme parameter values when chunk size is fixed. For completeness and to evaluate robustness across chunk sizes, additional heatmaps for alternative chunk sizes $(3, 4, 5, \text{and } 6)$ are presented in Appendix A (Figure A.1, A.2, A.3, and A.4).

Fixed Number of Chunks

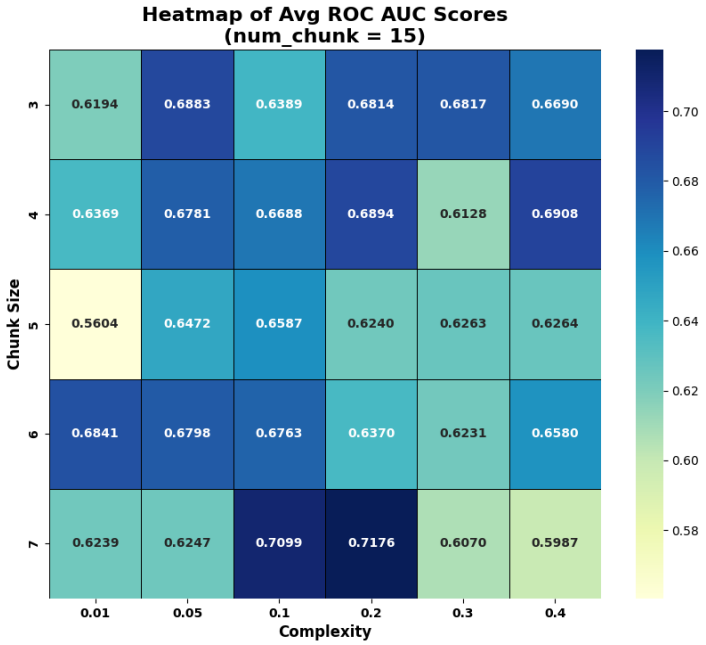


Figure 5.5.: Heatmap of Average ROC AUC Scores Across Chunk Size and Complexity with Number of Chunks 15.

Figure 5.5 presents the heatmap of average ROC AUC scores for various combinations of chunk size and complexity, with the number of chunks fixed at 15. The color intensity in the heatmap signifies the average ROC AUC for each setting, with darker shades representing higher scores. From the heatmap, it is evident that model performance varies across both chunk size and complexity. Overall, scores range from 0.56 to 0.72, indicating moderate discriminative ability across these configurations.

At low complexity values $(0.01 - 0.05)$, performance tends to be stable but only moderately high for all chunk sizes. Increasing complexity to the moderate range $(0.1 - 0.2)$ improves results considerably for most chunk sizes, with chunk size 7 and complexity 0.2 achieving the highest ROC score of 0.7176. In contrast, higher complexity values $(0.3-0.4)$ often correspond with reduced performance, which is likely attributable to overfitting or a loss of generalization capability.

Chunk size also demonstrates a notable effect on ROC scores. Smaller chunk sizes $(3$ or $4)$ produce moderate to strong performance, particularly when paired with moderate complexity.

However, chunk size 5 generally yields lower ROC values, particularly at low complexity. Increasing chunk size to 6 and 7 enhances model performance, with chunk size 7 at moderate complexity providing the peak value.

In summary, the heatmap indicates that optimal performance is achieved with a balanced selection of high chunk size (7) and moderate complexity (0.2), as extreme values for the complexity parameter does not improve and may even degrade performance. Additional heatmaps exploring alternative numbers of chunks $(5, 10, 20, 25,$ and $30)$ are provided in Appendix A (see Figures A.5, A.6, A.7, A.8, and A.9) to further illustrate this behavior.

Fixed Complexity

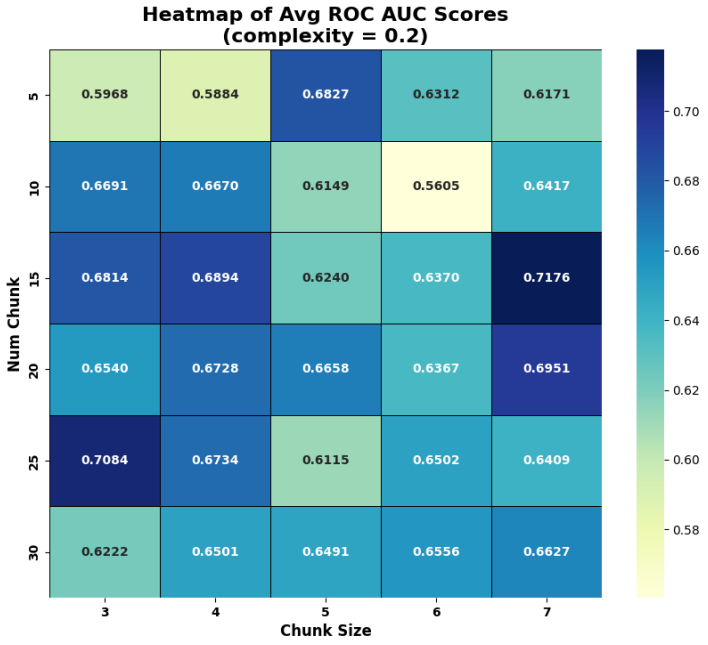


Figure 5.6.: Heatmap of Average ROC AUC Scores Across Chunk Size and Number of Chunks with Complexity 0.2.

Figure 5.6 presents the heatmap of average ROC AUC scores for different combinations of chunk size and number of chunks, with complexity fixed at 0.2. The color intensity represents the magnitude of the ROC AUC scores for each parameter pairing, where darker shades indicate higher performance. The figure shows that model discrimination capabilities varies as both chunk size and number of chunks change, with scores ranging approximately from 0.56 to 0.72, suggesting moderate overall performance.

Analysis of the heatmap shows that models with small chunk sizes $(3$ or $4)$ generally result in consistently high ROC AUC values across various numbers of chunks, especially in the mid-range $(15 - 25)$. Chunk sizes 6 and 7 also exhibit strong performance, particularly for number

of chunks $(15 - 20)$, where chunk size 7 and number of chunks 15 achieve the highest AUC of 0.7176. In contrast, chunk size 5 tends to produce lower AUCs, especially for small or large numbers of chunks.

With respect to the number of chunks, using lower number of chunks (5 or 10) often results in moderate performance, while mid-range values $(15 - 25)$ yield higher and more consistent ROC AUC scores, regardless of chunk size. The highest scores are observed for chunk sizes $4, 6$, and 7 with number of chunks set to $(15 - 25)$. Increasing the number of chunks beyond 25 maintains performance stability but does not further increase ROC AUC.

Overall, the highest average ROC AUC scores are obtained with a balanced setting of large chunk size and mid-range numbers of chunks $(15-25)$ under fixed complexity. To ensure completeness and evaluate robustness across complexity settings, additional heatmaps for alternative complexity values $(0.01, 0.05, 0.1, 0.3,$ and $0.4)$ are provided in Appendix A (Figure A.10, A.11, A.12, A.13, and A.14).

Conclusion

Although some of the trends are already visible in a single heatmap, presenting all three perspectives which are fixed chunk size, fixed number of chunks, and fixed complexity, is essential for a complete interpretation. Each of these views captures a different slice of the hyperparameter space and highlights how performance varies when specific parameters are controlled. Fixing only one hyperparameter does not show how performance changes when that parameter itself is varied. Moreover, showing all heatmaps allows for cross-validation of observed patterns and emphasizes that hyperparameters do not act in isolation but interact to influence performance. As interpretability is a central theme of this work, including all three heatmaps ensures transparency in the experimental exploration and provides the reader with a clear and robust understanding of the parameter effects without requiring reliance on the Appendix alone.

It is also important to note that while the heatmaps suggest a local optimum around complexity 0.2, chunk size 7, and 15 chunks, the global optimum identified earlier in Sections 5.2.1, 5.2.2, and 5.2.3 (complexity 0.1, chunk size 7, and 20 chunks) was determined using the average ROC scores across all combinations. This approach, based on mean performance across datasets, is more reliable, since a single high-scoring configuration (such as the one observed at 0.2, 7, and 15) may be sensitive to random initialization or seed variation. Averaging across combinations provides a more robust and stable criterion for hyperparameter selection. In this way, the heatmaps are used to illustrate interaction effects and highlight promising regions of the hyperparameter space, while the line-chart analyses ensure that the final reported settings reflect consistent and generalizable performance.

5.3. Performance Evaluation

To evaluate the performance of our proposed *SYRAN (SYmbolic Regression for unsupervised ANomaly detection)* algorithm, we compare it with 16 anomaly detection methods, namely, *DEAN*, *PCA*, *OCSVM*, *LOF*, *CBLOF*, *COF*, *HBOS*, *KNN*, *SOD*, *COPOD*, *ECOD*, *DeepSVDD*, *DAGMM*, *LODA*, and

IForest. The results of these algorithms are taken from the Unsupervised Surrogate Anomaly Detection study by Klüttermann et al. [21].

5.3.1. Performance Analysis on the breastw Dataset

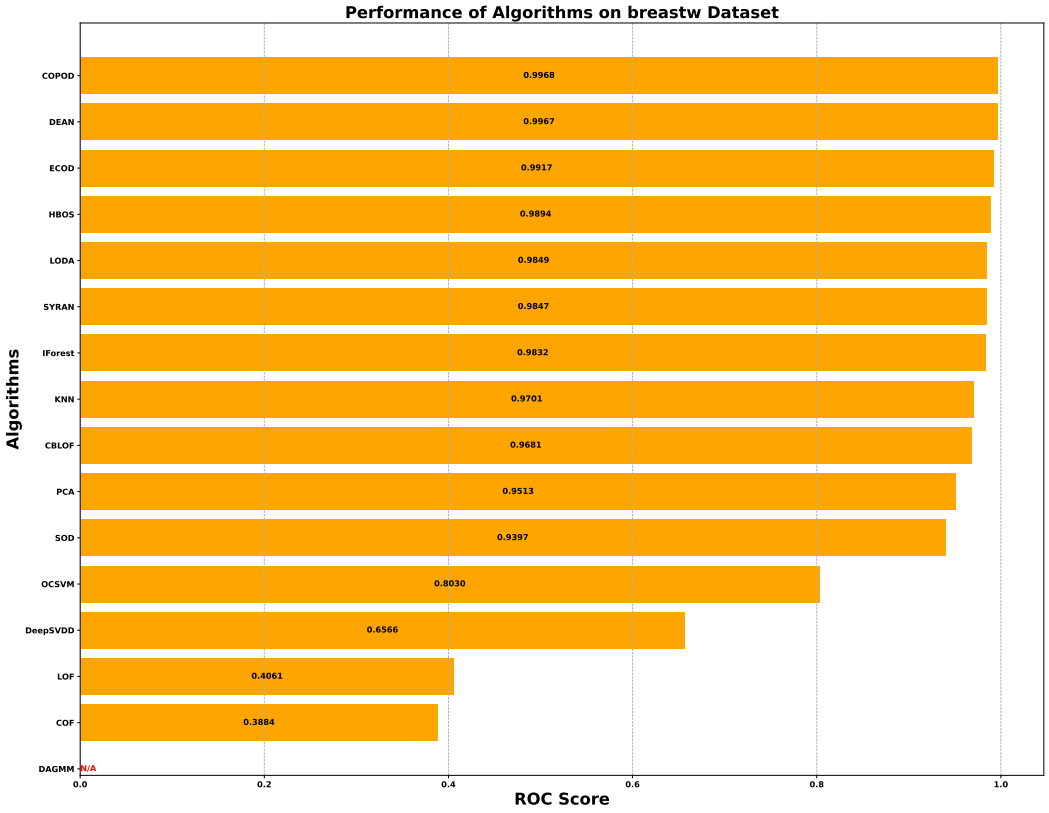


Figure 5.7.: Accuracy comparison of different algorithms on the breastw dataset.

Figure 5.7 shows the ROC AUC scores of various competitor algorithms along with the proposed *SYRAN* method, evaluated on the *breastw* dataset. *SYRAN* achieves a ROC score of 0.9847, which shows strong performance and places it among the best-performing algorithms for this *breastw* dataset.

The highest scores are obtained by *COPOD* (0.9968) and *DEAN* (0.9967), closely followed by *ECOD*, *HBOS*, and *LODA*, all of which achieve ROC scores above 0.98. The performance of *SYRAN* falls within this top group, indicating that the proposed approach is comparable to state-of-the-art methods in anomaly detection.

Traditional algorithms such as *IForest*, *KNN*, and *CBLOF* also perform well, with slightly lower ROC AUC scores ranging from 0.95 to 0.97. In contrast, methods like *SOD*, *OCSVM*, *DeepSVDD*,

LOF, and *COF* show moderate to poor performance, as reflected by substantially lower ROC AUC values. *DAGMM* failed to produce a valid score.

Overall, the comparison demonstrates that *SYRAN* is both reliable and competitive. It outperforms many traditional methods and achieves performance that is very close to the strongest modern algorithms on the *breastw* benchmark.

5.3.2. Accuracy Comparison Across Datasets

Algorithm	Hepatitis	Lymphography	PageBlocks	Pima	Stamps	WBC	Waveform
SYRAN	0.6302	0.8889	0.8219	0.5254	0.6051	0.9900	0.5713
DEAN	0.4615	**1.0000**	0.8587	0.6754	0.8512	0.9900	0.7449
PCA	0.7595	0.9982	0.9064	0.7077	0.9147	0.9820	0.6548
OCSVM	0.6775	0.9954	0.8876	0.6692	0.8386	0.9903	0.5629
LOF	0.3802	0.8986	0.7590	0.6571	0.5126	0.5417	0.7332
CBLOF	0.6640	0.9983	0.8504	0.7142	0.6818	**0.9946**	0.7242
COF	0.4145	0.9085	0.7265	0.6105	0.5381	0.6090	0.7256
HBOS	0.7985	0.9949	0.8058	0.7107	0.9073	0.9872	0.6877
KNN	0.5276	0.5591	0.8194	**0.7343**	0.6861	0.9056	0.7378
SOD	0.6817	0.7249	0.7775	0.6125	0.7326	0.9460	0.6857
COPOD	**0.8205**	0.9948	0.8805	0.6910	**0.9340**	0.9911	**0.7503**
ECOD	0.7967	0.9952	**0.9092**	0.5154	0.9141	0.9911	0.7325
DeepSVDD	0.5096	0.3229	0.5777	0.5103	0.5584	0.5550	0.5447
DAGMM	0.5480	0.7211	0.8961	0.5592	0.8888	-	0.4935
LODA	0.6487	0.8555	0.8334	0.6593	0.8718	0.9691	0.6013
IForest	0.6975	0.9981	0.8957	0.7287	0.9121	0.9901	0.7147

Table 5.2.: ROC comparison of different algorithms across the first 7 datasets.

Tables 5.2 and 5.3 illustrate the ROC scores of sixteen anomaly detection algorithms across 15 benchmark datasets. The results show that algorithmic performance varies considerably based on the dataset characteristics. Methods such as *DEAN*, *IForest*, *COPOD*, *PCA*, *ECOD*, and *HBOS* consistently show strong performance, often resulting in ROC values above 0.98 on datasets like *Lymphography*, and *WBC*, and get ROC scores above 0.90 on the *pendigits* dataset.

In contrast, several algorithms show mixed performance across datasets. For instance, *SYRAN* performs very well on *WBC* (0.9900) and *breastw* (0.9847), and achieves a competitive score on *wine* (0.8800). However, it shows lower performance on more complex datasets such as *Pima* (0.5254), *Waveform* (0.5713), and *yeast* (0.5618). Similarly, algorithms like *LOF* and *COF* generally produce modest scores, often below 0.80, which means they can not easily adapt to different types of data distributions.

Algorithm	breastw	cardio	cover	glass	pendigits	vowels	wine	yeast
SYRAN	0.9847	0.7711	0.5018	0.6790	0.3306	0.6452	0.8800	**0.5618**
DEAN	0.9967	0.9274	0.5000	**0.9012**	**0.9821**	0.9616	0.8800	0.5445
PCA	0.9513	**0.9555**	**0.9373**	0.6629	0.9373	0.6529	0.8437	0.4115
OCSVM	0.8030	0.9391	0.9262	0.3536	0.9375	0.6159	0.7307	0.4100
LOF	0.4061	0.6633	0.8458	0.6920	0.4799	0.9312	0.3774	0.4531
CBLOF	0.9681	0.8993	0.8930	0.8294	0.9040	0.8992	0.2586	0.4485
COF	0.3884	0.7141	0.7691	0.7224	0.4507	0.9404	0.4444	0.4448
HBOS	0.9894	0.8467	0.8024	0.7723	0.9304	0.7221	**0.9136**	0.3964
KNN	0.9701	0.7664	0.8597	0.8229	0.7295	**0.9726**	0.4498	0.3906
SOD	0.9397	0.7325	0.7446	0.7336	0.6629	0.9265	0.4611	0.4246
COPOD	**0.9968**	0.9235	0.8864	0.7243	0.9068	0.5315	0.8865	0.3699
ECOD	0.9917	0.9444	0.9342	0.7570	0.9122	0.4581	0.7134	0.3961
DeepSVDD	0.6566	0.5896	0.4620	0.4749	0.3992	0.5249	0.5952	0.4792
DAGMM	-	0.7501	0.8989	0.7609	0.6422	0.6058	0.6170	0.4111
LODA	0.9849	0.9034	0.9234	0.7313	0.8910	0.7036	0.9012	0.4458
IForest	0.9832	0.9319	0.8674	0.7713	0.9476	0.7394	0.8037	0.3776

Table 5.3.: ROC comparison of different algorithms across the last 8 datasets.

Some traditional algorithms, such as *DAGMM* and *DeepSVDD*, tend to underperform compare to other algorithms. Their ROC scores frequently remain below 0.70, with a few exceptions, for example, *DAGMM* achieves above 0.88 only on *PageBlocks*, *cover*, and *Stamps*. This indicates that these models are less suitable for various anomaly detection tasks.

The most challenging dataset appears to be *yeast*, where nearly all algorithms perform poorly, with ROC scores typically below 0.60. *SYRAN* outperforms all other algorithms on this dataset, while strong algorithms such as *DEAN* and *IForest* record relatively low values 0.5445 and 0.3776 respectively, highlighting the difficulty posed by noisy or overlapping data structures.

Overall, no single algorithm dominates across all datasets. Instead, performance strongly depends on the nature of the dataset, emphasizing the importance of evaluating anomaly detection algorithms over a wide range of benchmarks. While *SYRAN* achieves competitive results in several domains, its relative performance varies, underlining both its potential and its limitations.

5.3.3. Critical Difference (CD) Diagram

To further evaluate the performance of the anomaly detection algorithms presented in Tables 5.2 and 5.3, we conduct a statistical analysis and visualize the results using a critical difference (CD) plot.

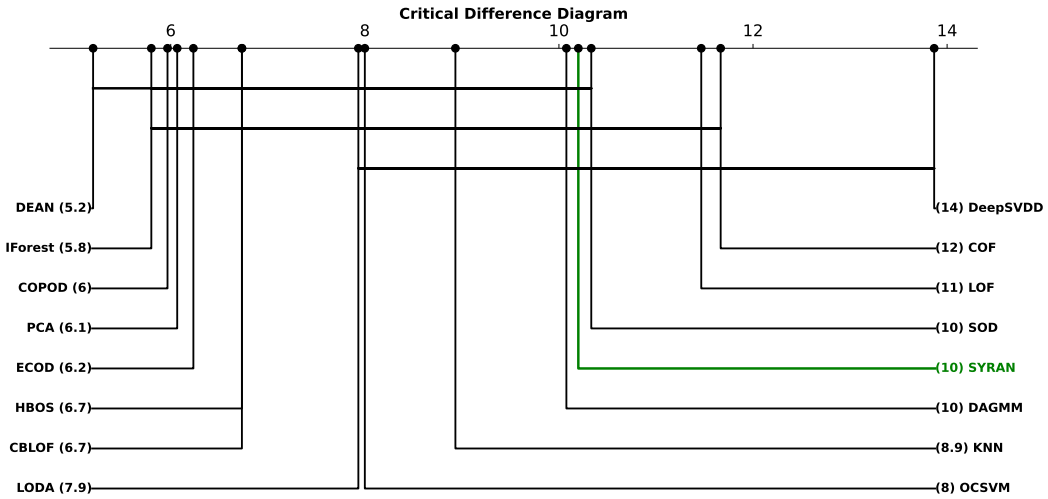


Figure 5.8.: Critical Difference (CD) diagram comparing the AUC-ROC performance using fifteen datasets. SYRAN is highlighted in green while other anomaly detection algorithms are in black.

Figure 5.8 presents the CD diagram, which ranks algorithms based on their average performance across all fifteen benchmark datasets. In this ranking, lower values indicate stronger overall performance. A horizontal line connects groups of algorithms that are not statistically significantly different from each other.

The results indicate that *DEAN* achieves the best overall ranking, followed by *IForest* and algorithms such as *COPOD*, *PCA*, and *ECOD*. In comparison, *SYRAN* achieves a mid-level position, ranking after *DAGMM* and before *SOD*. While *SYRAN* is not among the top-ranked methods, it remains competitive, outperforming algorithms such as *LOF*, *COF*, and *DeepSVDD*, which are the lower-ranked algorithms for these 15 datasets.

These findings suggest that while *SYRAN* does not occupy the very top position in the ranking, no algorithm is statistically significantly better than *SYRAN* according to the CD analysis. This result highlights that *SYRAN* performs on par with strong baselines such as *DEAN* and *IForest*, while additionally offering the important advantage of interpretability through symbolic representations. In this sense, the CD plot demonstrates that *SYRAN* achieves a balanced trade-off. It provides competitive predictive performance while simultaneously enabling transparent and interpretable anomaly explanations, a feature not available in most other methods.

5.4. Equations from Real-World Datasets

In interpretable machine learning, the true value of a model lies not just in its predictive performance, but in its ability to extract meaningful patterns in real-world data. In this section, we demonstrate how our *SYRAN* algorithm can extract clear, human-readable equations from a

medical dataset named *breastw* (Wisconsin Breast Cancer Dataset).

This dataset consists of 9 cytological features, namely, *clump thickness*, *uniformity of cell size*, *uniformity of cell shape*, *marginal adhesion*, *single epithelial cell size*, *bare nuclei*, *bland chromatin*, *normal nucleoli*, and *mitoses*. Accurately distinguishing malignant from benign tumors is a challenging task that demands models that are both precise and understandable, particularly in a clinical context. For this demonstration, we use the optimal complexity value of 0.10, as identified in Section 5.2.1, which offers a balanced trade-off between interpretability and predictive performance.

5.4.1. Extraction of Symbolic Equation

Here are three example equations produced by the algorithm, each involving a single feature raised to an exponential power. These demonstrate how the approach balances predictive accuracy (AUC-ROC) and formula simplicity for interpretable results.

$$
\begin{aligned}
S_1 &= f^{\exp(h)}, \\
S_2 &= a^{3f}, \\
S_3 &= b^3,
\end{aligned}
$$

where a denotes *clump thickness*, b denotes *uniformity of cell size*, f denotes *bare nuclei*, and h denotes *normal nucleoli*. These equations capture nonlinear, exponential interactions among key cytological features that are known to correlate with malignancy. For example, in S_1, higher values of bare nuclei combined with elevated nucleoli indicate more aggressive tumors, and the exponential form highlights how risk can increase sharply rather than linearly.

5.4.2. Biomedical Interpretation

Beyond performance metrics, the key advantage of these symbolic equations is interpretability. The interpretability of these equations allows us to directly understand how individual cytological features contribute to malignancy risk. Unlike complex models such as random forests or deep neural networks, which are difficult to explain, these formulas are transparent, which gives the advantage that clinicians can understand directly how features like clump thickness or nucleoli contribute to malignancy risk. For instance, S_1 highlights the interaction between *bare nuclei* and *nucleoli*, capturing how nuclear abnormalities amplify malignancy risk. Similarly, S_3 indicates that as ununiformity of cell size increases, malignancy risk grows rapidly, highlighting potential threshold effects that can guide diagnostic decisions.

Interpretation of the Symbolic Regression Equation $S_1 = (f^{\exp(h)})$
The equation $S_1 = (f^{\exp(h)})$ models the *bare nuclei* feature (f) raised to an exponential power determined by *normal nucleoli* (h), yielding an exponential growth term that reflects their combined diagnostic influence in malignant cases.

From a cytological perspective, bare nuclei are isolated nuclei stripped of their cytoplasmic envelope. In benign breast aspirates, these typically represent myoepithelial cells and are seen as uniform, bipolar structures within cohesive epithelial clusters [54, 57, 19]. By contrast, in malignant aspirates such as invasive ductal carcinoma, an increased number of naked tumor nuclei has been reported, reflecting tumor cell dissociation and loss of adhesion [45].

The exponent $\exp(h)$ introduces marked sensitivity to *nucleolar prominence*. Nucleoli are key sites of ribosomal biogenesis, and enlargement or conspicuity of nucleoli is a recognized cytological feature of malignancy, associating with increased protein synthesis and cell proliferation. In this formulation, as h increases, the effective power on f escalates dramatically (e.g., for $h = 5$, $\exp(5) \approx 148$; for $h = 10$, $\exp(10) \approx 22{,}026$). This formulation, therefore, captures a synergistic relationship in which larger and more conspicuous nucleoli, often seen in high-grade tumors, significantly amplify the diagnostic impact of bare nuclei. This interpretation is consistent with established grading systems, where both nucleolar prominence and nuclear morphology contribute significantly to the discrimination of benign and malignant lesions [45].

It is important to note that a small value of S_1 does not solely determine whether the tissue is normal. Instead, S_1 needs to be 1 to show that the cytology is normal. This indicates that a low value of f combined with a high value of h results in an abnormal S_1, suggesting malignancy due to the exponential amplification. In contrast, a low value of f with a negative or very low value of h yields S_1 approaching 1, which indicates normal tissue. This indicates that negative or very low nucleolar prominence effectively normalizes the impact of bare nuclei, aligning with benign cytological features.

In summary, equation S_1 acts as a proliferation amplifier. When nucleolar prominence (h) is negative or very low, S_1 remains close to 1, reflecting normal cytology regardless of bare nuclei levels. However, as nucleoli become increasingly prominent (positive h), the effect of bare nuclei (f) grows nonlinearly, indicating aggressive tumors. This provides an interpretable quantitative link for pathologists, where nucleolar hyperactivity magnifies nuclear fragility, serving as a surrogate marker of malignant transformation.

Interpretation of the Symbolic Regression Equation $S_2 = a^{3f}$

The equation $S_2 = a^{3f}$ captures a synergistic interaction between clump thickness (a) and bare nuclei (f), two critical cytological features of breast tumors. In the equation, clump thickness is raised to a power proportional to bare nuclei, scaled by a factor of 3. For example, if $f = 2$ (benign-like), the exponent is 6, producing a strong effect (e.g., $a = 7$ yields $7^6 \approx 117,649$); if $f = 7$ (malignant), the exponent becomes 21, yielding an extreme output (e.g., $7^{21} \approx 5.6 \times 10^{17}$). This exponential relationship suggests that bare nuclei amplify the malignancy risk associated with loose tumor clumps, reflecting a biological synergy where fragile cells (high f) worsen the invasive potential of dissociated clumps (high a).

The equation S_2 enhances the ability of our SR model to distinguish benign from malignant tumors. Its nonlinear structure amplifies the combined effect of clump thickness and bare nuclei, creating a clear threshold for malignancy risk. For instance, low values (e.g., $a = 2$, $f = 1$,

$S_2 = 2^3 = 8$) indicate benign tumors, while high values (e.g., $a = 7$, $f = 5$, $S_2 \approx 7^{15}$) indicate malignant cases. This aligns with clinical grading, where dissociated clumps and fragile nuclei indicate high-grade carcinomas [42]. The simplicity of this equation lies in the single term involving two features, which makes it interpretable for pathologists, who can use it as a risk score in FNA diagnostics, complementing traditional scoring systems.

In summary, the equation S_2 is a biologically meaningful and interpretable component of our SR model. It quantifies the synergistic interaction between clump thickness and bare nuclei, reflecting how cellular dyshesion and fragility drive tumor invasiveness in breast cancer. For pathologists, this offers an interpretable signal, such as cohesive clumps with few bare nuclei suggest benignity, while dissociated clumps combined with abundant bare nuclei predict malignancy with high sensitivity.

Interpretation of the Symbolic Regression Equation $S_3 = b^3$

The equation $S_3 = b^3$ captures the nonlinear escalation of malignancy risk associated with uniformity of cell size, a key cytological feature in breast cancer diagnostics. In cytological literature, benign cells are described as showing uniformity in size and shape in both cellular and nuclear dimensions, whereas malignant or dysplastic changes are associated with loss of this uniformity, or pleomorphism [44].

The cubic transformation in S_3 amplifies the effect of cell size variation. For example, a benign-like score of $b = 2$ yields $S_3 = 8$, while a malignant score of $b = 8$ produces $S_3 = 512$, and $b = 10$ yields $S_3 = 1000$. This cubic nonlinearity reflects a biological threshold where moderate size variation has minimal impact, but extreme variation indicates aggressive tumor behavior. Cytological grading systems, such as Robinson's, assign higher scores to increased cell size variation, correlating with histological grade and poor prognosis [42].

Equation S_3 strengthens the ability of our SR model to distinguish benign from malignant tumors. Its cubic structure creates a clear threshold for malignancy risk where low values (e.g., $b = 2$, $S_3 = 8$) indicate benign tumors, while high values (e.g., $b = 7$, $S_3 = 343$) indicate malignant cases. This aligns with clinical practice, where increased cell size variation is a diagnostic marker for high-grade carcinomas [42]. The simplicity of this equation lies in the single term with one feature, which makes it highly interpretable for pathologists, who can use S_3 as a standalone risk score in FNA diagnostics.

In summary, equation S_3 is a biologically meaningful and interpretable component of our SR model. It quantifies the nonlinear escalation of malignancy risk driven by cell size variation, reflecting the anaplastic changes in aggressive breast tumors. For pathologists, this offers an interpretable signal, such as uniform cell sizes (low b) suggest benignity, while pronounced variation in cell size (high b) indicates malignancy.

5.4.3. Conclusion

By extracting these equations from real-world data, our method provides a transparent and interpretable alternative to traditional black-box models. Researchers and clinicians can use such equations not only for predictive tasks but also for gaining mechanistic insights into the

factors driving malignancy, thereby supporting explainable and trustworthy decision-making in medical contexts.

Overall, these equations show that interpretable symbolic regression can transform raw biomedical data into understandable, reliable knowledge that supports decision-making in high-stakes settings like oncology.

5.5. Extended Analysis with Higher Number of Chunks

To further investigate whether we can improve the average performance of the proposed method, we extended the hyperparameter space by increasing the number of chunks to larger values of 50, 100, and 200. In contrast to the previous experiments, where the number of chunks ranged from 5 to 30, this setting explores whether a higher number of chunks can provide performance benefits. To balance computational feasibility, the chunk size was reduced to values of 3, 4, 5, and 6, removing chunk size 7 from consideration. This experiment was conducted on 12 different sets of datasets, selected based on their ability to complete within the available computational resources, with these new hyperparameter settings.

5.5.1. Effect of Higher Number of Chunks

The relationship between the number of chunks and the average ROC AUC score is illustrated in Figure 5.9. As shown, performance improves gradually as the number of chunks increases. Moving from 50 to 200 chunks raises the average ROC AUC from approximately 0.69 to nearly 0.77. This indicates that dividing features into a larger number of chunks enables the symbolic ensemble to capture richer and more detailed relationships in the data, which improves anomaly detection capability.

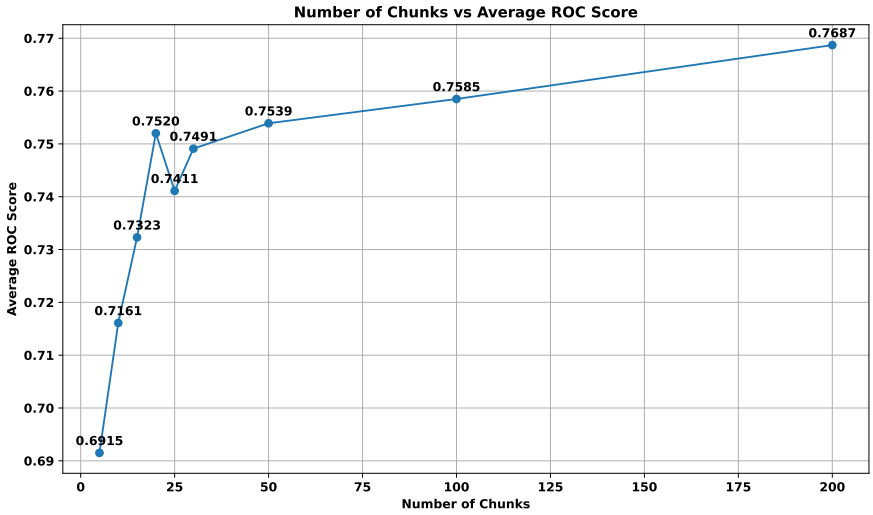


Figure 5.9.: Complexity vs ROC Scores for twelve datasets with a higher number of chunks.

However, the results also suggest diminishing returns beyond a certain point. While the jump from 50 to 100 chunks yields a substantial improvement, the gains from 100 to 200 chunks, although positive, are comparatively smaller. This indicates that, although larger numbers of chunks are beneficial, they come at the cost of higher computational effort and only moderate accuracy improvements beyond a certain point.

5.5.2. Comparative Performance Evaluation

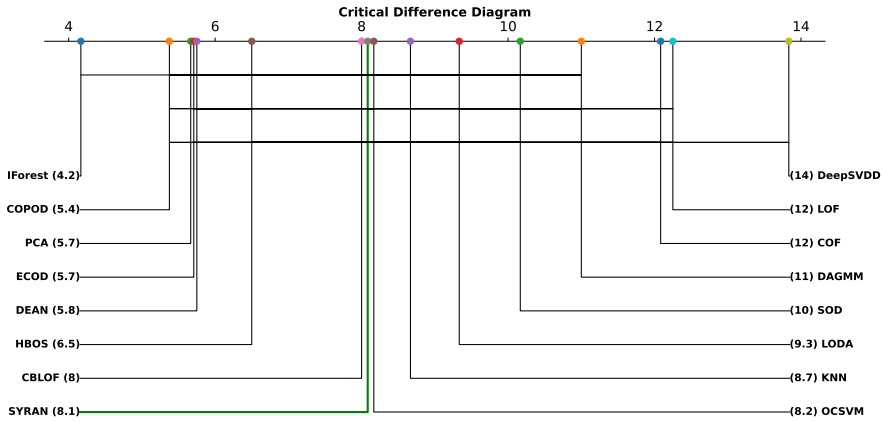


Figure 5.10.: Critical Difference (CD) diagram comparing the AUC-ROC performance using twelve datasets with a higher number of chunks. SYRAN is highlighted in green.

In order to illustrate these findings, we used a CD plot to further evaluate the extended chunk configuration against established anomaly detection baselines, as illustrated in Figure 5.10. The diagram ranks the algorithms based on their average ROC AUC across the 12 datasets, where a lower rank value indicates stronger overall performance. Algorithms connected by a horizontal line are not statistically significantly different from each other.

The results show that *IForest* (4.2) achieves the best ranking, followed closely by *COPOD* (5.4), *PCA* (5.7), *ECOD* (5.7), and *DEAN* (5.8). These algorithms form the leading group, consistently delivering high predictive accuracy. In comparison, *SYRAN* achieves an average rank of 8.1, which places it in the mid-tier but still within the equivalence group that includes established baselines such as *CBLOF* (8.0), *OCSVM* (8.2), and *KNN* (8.7). Importantly, *SYRAN* outperforms several methods such as *LODA* (9.3), *SOD* (10), *DAGMM* (11), *COF* (12), *LOF* (12), and *DeepSVDD* (14), all of which are positioned at the lower end of the ranking.

Compared to the earlier CD analysis with smaller chunk configurations (Figure 5.8), the high-chunk setting shows a relative improvement for *SYRAN*. While it does not surpass the very top-performing methods, its competitive placement within the same statistical group as strong baselines indicates that the algorithm scales effectively with larger chunk settings. This suggests that although *SYRAN* requires more computational resources, it still offers a good balance between interpretability and performance, and remains competitive even under more demanding hyperparameter configurations.

5.5.3. Conclusion

In summary, this extended analysis shows that increasing the number of chunks improves the performance of *SYRAN* by allowing the model to capture more detailed feature interactions. While higher numbers of chunks $(50, 100, 200)$ provide consistent improvements, they also come with higher computational demands, making it necessary to balance accuracy against efficiency. Moreover, this behaviour shows a clear trade-off between interpretability and performance. As the number of chunks increases, the ensemble produces a larger number of symbolic equations, which enhances predictive power but makes the overall model structure more complex and less interpretable. These results highlight that *SYRAN* can adapt to different computational costs. When resources are sufficient, a higher number of chunks delivers near state-of-the-art accuracy, whereas moderate settings already provide competitive performance with reduced complexity and cost. Further increasing the number of chunks could potentially lead to even better results, though at the expense of additional computational effort.

6. Conclusion and Future Work

This thesis proposed *SYRAN (SYmbolic Regression for unsupervised ANomaly detection)*, a novel symbolic regression based approach for interpretable unsupervised anomaly detection. The method is designed to address a key challenge in anomaly detection by achieving accurate detection while maintaining human interpretability. By using evolutionary mathematical function learning together with an ensemble framework based on feature chunking, *SYRAN* transforms raw data into transparent equations that can be directly evaluated and understood by researchers and domain experts.

The experiments on 15 real-world datasets showed that *SYRAN* is competitive with established anomaly detection algorithms. It performed particularly strongly on biomedical datasets such as *breastw* and *WBC*, where it achieved ROC scores close to or above 0.98, and it was able to produce symbolic equations that carried meaningful biomedical interpretations. On some datasets, such as *Pima*, *Waveform*, and *yeast*, the performance was lower, indicating the difficulty of learning robust symbolic functions under noisy or high-dimensional conditions. The study of hyperparameters further showed that moderate complexity, larger chunk sizes, and a balanced number of chunks in the range of $15 - 20$ often yielded the best results. Compared with methods like *DEAN*, *IForest*, and *COPOD*, *SYRAN* achieved a middle-level overall ranking. Although it does not reach state-of-the-art performance compared to the strongest algorithms, it is also not significantly worse, and its main strength remains the interpretability of the learned functions.

The main contribution of this work lies in showing that symbolic regression can be adapted to unsupervised anomaly detection in a way that balances interpretability and accuracy. The systematic exploration of hyperparameters provided new insights into how design choices affect performance, while the extraction of symbolic equations from real-world biomedical data illustrated the possibility of translating raw features into domain-relevant knowledge. The extended analysis with a higher number of chunks shows that dividing features into finer partitions can improve detection performance. As the number of chunks increases, the average ROC AUC also increases gradually, indicating that the ensemble structure of *SYRAN* benefits from more detailed symbolic representations when sufficient computational resources are available. These results further support the idea that interpretable symbolic regression can not only make predictions but also help uncover meaningful patterns in complex domains.

At the same time, several limitations were identified. Symbolic regression is computationally demanding, especially in high-dimensional settings, and the performance of *SYRAN* varies across datasets. Moreover, not all generated equations were meaningful or interpretable: in several cases, the output included trivial or nonsensical expressions (such as constants (0.9999^e),

single-variable terms (f), or overly complex constructs like $e^{|\sqrt{(\text{auto}_0^{-1})}|}$. Such results highlight the inherent challenge of balancing search flexibility with interpretability. The chosen hyperparameters, while carefully tuned, should also be regarded as approximations rather than optimal configurations, as no guarantee exists that they represent the best possible combinations.

Future research should focus on improving scalability, for example through parallelized search strategies or hybrid models that combine symbolic regression with deep representation learning. More adaptive ensemble strategies could be developed to improve robustness by weighting symbolic functions according to their diversity or reliability. Extending the method to semi-supervised settings, where limited labels are available, may further enhance its practical value. Finally, applying *SYRAN* to real-world domains such as fraud detection, predictive maintenance, or clinical diagnostics would provide practical validation of its interpretability and relevance in decision-making contexts.

In conclusion, this thesis demonstrates that interpretable anomaly detection is feasible without significant losses in performance. By turning data into equations, *SYRAN* shows that symbolic regression can bridge the gap between accuracy and transparency in anomaly detection. At the same time, it shows the challenges and imperfections that accompany symbolic modeling, pointing toward the need for more trustworthy and domain-adaptable methods that not only predict but also explain.

Bibliography

[1] M. Ahmed and A.N. Mahmood. 2015. Novel Approach for Network Traffic Pattern Analysis using Clustering-based Collective Anomaly Detection. *Annals of Data Science* 2 (2015), 111–130. doi:10.1007/s40745-015-0035-y

[2] Mohiuddin Ahmed, Abdun Naser Mahmood, and Jiankun Hu. 2016. A survey of network anomaly detection techniques. *J. Netw. Comput. Appl.* 60, C (Jan. 2016), 19–31. doi:10.1016/j.jnca.2015.11.016

[3] Andrea Borghesi, Andrea Bartolini, Michele Lombardi, Michela Milano, and Luca Benini. 2019. Anomaly Detection Using Autoencoders in High Performance Computing Systems. *Proceedings of the AAAI Conference on Artificial Intelligence*, 9428–9433. doi:10.1609/aaai.v33i01.33019428

[4] Andrew P. Bradley. 1997. The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition* 30, 7 (1997), 1145–1159. doi:10.1016/S0031-3203(96)00142-2

[5] Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, and Jörg Sander. 2000. LOF: identifying density-based local outliers. *SIGMOD Rec.* 29, 2 (May 2000), 93–104. doi:10.1145/335191.335388

[6] Christian Callegari, Loris Gazzarrini, Stefano Giordano, Michele Pagano, and Teresa Pepe. 2011. A Novel PCA-Based Network Anomaly Detection. In *2011 IEEE International Conference on Communications (ICC)*. 1–5. doi:10.1109/icc.2011.5962595

[7] Van Loi Cao, Miguel Nicolau, and James Mcdermott. 2016. One-Class Classification for Anomaly Detection with Kernel Density Estimation and Genetic Programming, Vol. 9594. 3–18. doi:10.1007/978-3-319-30668-1_1

[8] Raghavendra Chalapathy, Aditya Krishna Menon, and Sanjay Chawla. 2018. Anomaly Detection using One-Class Neural Networks. *CoRR* (2018). arXiv:1802.06360 http://arxiv.org/abs/1802.06360

[9] Varun Chandola, Arindam Banerjee, and Vipin Kumar. 2009. Anomaly Detection: A Survey. *ACM Comput. Surv.* 41 (07 2009). doi:10.1145/1541880.1541882

[10] Shuyan Chen, Wei Wang, and Henk van Zuylen. 2010. A comparison of outlier detection algorithms for ITS data. *Expert Syst. Appl.* 37 (03 2010), 1169–1178. doi:10.1016/j.eswa.2009.06.008

[11] Vipul K. Dabhi and Sanjay Chaudhary. 2012. A Survey on Techniques of Improving Generalization Ability of Genetic Programming Solutions. *CoRR* abs/1211.1119 (2012). http://arxiv.org/abs/1211.1119

[12] Sarah M. Erfani, Sutharshan Rajasegarar, Shanika Karunasekera, and Christopher Leckie. 2016. High-dimensional and large-scale anomaly detection using a linear one-class SVM with deep learning. *Pattern Recognition* 58 (2016), 121–134. doi:10.1016/j.patcog.2016.03.028

[13] Xing Gao and Guilin Li. 2020. A KNN Model Based on Manhattan Distance to Identify the SNARE Proteins. *IEEE Access* 8 (2020), 112922–112931. doi:10.1109/ACCESS.2020.3003086

[14] Markus Goldstein and Andreas Dengel. 2012. Histogram-based Outlier Score (HBOS): A fast Unsupervised Anomaly Detection Algorithm.

[15] Songqiao Han, Xiyang Hu, Hailiang Huang, Mingqi Jiang, and Yue Zhao. 2022. ADBench: Anomaly Detection Benchmark. https://arxiv.org/abs/2206.09426

[16] Sahand Hariri, Matias Carrasco Kind, and Robert J. Brunner. 2018. Extended Isolation Forest. *arXiv preprint arXiv:1811.02141* (2018).

[17] Zengyou He, Xiaofei Xu, and Shengchun Deng. 2003. Discovering cluster-based local outliers. *Pattern Recognition Letters* 24, 9 (2003), 1641–1650. doi:10.1016/S0167-8655(03)00003-5

[18] Chengqiang Huang. 2018. *Featured anomaly detection methods and applications.* Ph. D. Dissertation. University of Exeter, Devon, UK.

[19] S. Jain and R. Sharma. 2013. Benign Pairs: A Significant Entity in Aspiration Cytology. *PMC* (2013). https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3855199/

[20] Ying Jin, Weilin Fu, Jian Kang, Jiadong Guo, and Jian Guo. 2020. Bayesian Symbolic Regression. arXiv:1910.08892 [stat.ME] https://arxiv.org/abs/1910.08892

[21] Simon Klüttermann, Tim Katzke, and Emmanuel Müller. 2025. Unsupervised Surrogate Anomaly Detection. https://arxiv.org/abs/2504.20733 arXiv preprint arXiv:2504.20733.

[22] Edwin Knorr, Raymond Ng, and Vladimir Tucakov. 2000. Distance-Based Outliers: Algorithms and Applications. *The VLDB Journal* 8 (02 2000), 237–253. doi:10.1007/s007780050006

[23] Edwin M. Knorr and Raymond T. Ng. 1998. Algorithms for Mining Distance-Based Outliers in Large Datasets. In *Proceedings of the 24rd International Conference on Very Large Data Bases (VLDB '98)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 392–403.

[24] John R. Koza. 1994. Genetic programming as a means for programming computers by natural selection. *Statistical Computing* 4 (1994), 87–112. doi:10.1007/BF00175355

[25] Hans-Peter Kriegel, Peer Kröger, Erich Schubert, and Arthur Zimek. 2009. Outlier detection in axis-parallel subspaces of high dimensional data. In *Advances in Knowledge Discovery and Data Mining*, T. Theeramunkong, B. Kijsirikul, N. Cercone, and T.B. Ho (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 831–838.

[26] Mikel Landajuela, Chak Shing Lee, Jiachen Yang, Ruben Glatt, Claudio P. Santiago, Ignacio Aravena, Terrell Mundhenk, Garrett Mulcahy, and Brenden K. Petersen. 2022. A Unified Framework for Deep Symbolic Regression. In *Advances in Neural Information Processing Systems (NeurIPS)*, Vol. 35. 33985–33998. https://proceedings.neurips.cc/paper_files/paper/2022/file/dbca58f35bddc6e4003b2dd80e42f838-Paper-Conference.pdf

[27] Julien Lesouple, Cédric Baudoin, M. Spigai, and Jean-Yves Tourneret. 2021. Generalized Isolation Forest for Anomaly Detection. *Pattern Recognition Letters* 149 (06 2021). doi:10.1016/j.patrec.2021.05.022

[28] Zheng Li, Yue Zhao, Nicola Botta, Cezar Ionescu, and Xiyang Hu. 2020. COPOD: Copula-Based Outlier Detection. In *2020 IEEE International Conference on Data Mining (ICDM)*. 1118–1123. doi:10.1109/ICDM50108.2020.00135

[29] Zheng Li, Yue Zhao, Xiyang Hu, Nicola Botta, Cezar Ionescu, and George H. Chen. 2022. ECOD: Unsupervised Outlier Detection Using Empirical Cumulative Distribution Functions. *CoRR* abs/2201.00382 (2022). https://arxiv.org/abs/2201.00382

[30] Fei Tony Liu, Kai Ting, and Zhi-Hua Zhou. 2012. Isolation-Based Anomaly Detection. *ACM Transactions on Knowledge Discovery From Data - TKDD* 6 (03 2012), 1–39. doi:10.1145/2133360.2133363

[31] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. 2008. Isolation Forest. In *2008 Eighth IEEE International Conference on Data Mining*. 413–422. doi:10.1109/ICDM.2008.17

[32] Cewu Lu, Jianping Shi, and Jiaya Jia. 2013. Abnormal Event Detection at 150 FPS in MATLAB. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2720–2727. doi:10.1109/ICCV.2013.338

[33] Sean Luke and Liviu Panait. 2006. A Comparison of Bloat Control Methods for Genetic Programming. *Evolutionary Computation* 14, 3 (2006), 309–344. doi:10.1162/evco.2006.14.3.309

[34] Nour Makke and Sanjay Chawla. 2024. Interpretable scientific discovery with symbolic regression: a review. *Artificial Intelligence Review* 57, 1 (Jan. 2024). doi:10.1007/s10462-023-10622-0

[35] Matteo Manzi and Massimiliano Vasile. 2020. Orbital Anomaly Reconstruction Using Deep Symbolic Regression. In *Proceedings of the 71st International Astronautical Congress (IAC)*. Dubai, UAE. https://strathprints.strath.ac.uk/74268/

[36] Michael Nsor. 2024. Predictive Maintenance Using Machine Learning for Engineering Systems Through Real-Time Sensor Data and Anomaly Detection Models. *International Journal of Research Publication and Reviews* 5 (2024), 5167–5183. doi:10.55248/gengpi.6.0725.2541

[37] Brenden K. Petersen, Mikel Landajuela, T. Nathan Mundhenk, Claudio P. Santiago, Soo K. Kim, and Joanne T. Kim. 2021. Deep symbolic regression: Recovering mathematical expressions from data via risk-seeking policy gradients. arXiv:1912.04871 [cs.LG] https://arxiv.org/abs/1912.04871

[38] Tomáš Pevný. 2016. Loda: Lightweight on-line detector of anomalies. *Machine Learning* 102, 2 (2016), 275–304. doi:10.1007/s10994-015-5521-0

[39] VB Prasath, HAA Alfeilat, O Lasassmeh, and ABA Hassanat. 2017. Distance and Similarity Measures Effect on the Performance of K-Nearest Neighbor Classifier–A. *arXiv preprint arXiv:1708.04321* (2017).

[40] B. R. Preiss. 1999. *Data Structures and Algorithms with Object-Oriented Design Patterns in Java.* Wiley.

[41] Yousef A. Radwan, Gabriel Kronberger, and Stephan Winkler. 2024. A Comparison of Recent Algorithms for Symbolic Regression to Genetic Programming. https://arxiv.org/abs/2406.03585

[42] Koshalya Rajendran, Muthu Sudalaimuthu, and Shivashekar Ganapathy. 2022. Cytological Grading of Breast Carcinomas and Its Prognostic Implications. *Cureus* 14 (2022). doi:10.7759/cureus.29385

[43] Sridhar Ramaswamy, Rajeev Rastogi, and Kyuseok Shim. 2000. Efficient Algorithms for Mining Outliers from Large Data Sets. In *SIGMOD 2000 - Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data (SIGMOD 2000 - Proceedings of the*

2000 ACM SIGMOD International Conference on Management of Data). Association for Computing Machinery, Inc, 427–438. doi:10.1145/342009.335437 Publisher Copyright: © ACM 2000; 2000 ACM SIGMOD International Conference on Management of Data, SIGMOD 2000 ; Conference date: 15-05-2000 Through 18-05-2000.

[44] Rose E. Raskin. 2010. CHAPTER 2 - General Categories of Cytologic Interpretation. In *Canine and Feline Cytology (Second Edition)* (second edition ed.), Rose E. Raskin and Denny J. Meyer (Eds.). W.B. Saunders, Saint Louis, 15–25. doi:10.1016/B978-141604985-2.50007-4

[45] I.A. Robinson, G. McKee, A. Nicholson, P.A. Jackson, M.G. Cook, J. D'Arcy, and M.W. Kissin. 1994. Prognostic value of cytological grading of fine-needle aspirates from breast carcinomas. *The Lancet* 343, 8903 (1994), 947–949. doi:10.1016/S0140-6736(94)90066-3 Originally published as Volume 1, Issue 8903.

[46] R. Rodrigues, N. Bhargava, R. Velmurugan, and S. Chaudhuri. 2020. Multi-timescale Trajectory Prediction for Abnormal Human Activity Detection. In *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*. Snowmass, CO, USA, 2615–2623. doi:10.1109/WACV45572.2020.9093633

[47] Lukas Ruff, Robert Vandermeulen, Nico Görnitz, Lucas Deecke, Shoaib Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. 2018. Deep One-Class Classification. In *Proceedings of the 35th International Conference on Machine Learning (ICML)*. PMLR, 4393–4402.

[48] Efi Safikou, Krishna Pattipati, and George Bollas. 2025. Fault Diagnosis and Prognosis With Inferential Sensors: A Hybrid Approach Integrating Symbolic Regression and Information Theory. *IEEE Access* PP (2025), 1–1. doi:10.1109/ACCESS.2025.3564467

[49] Yusuf Sahin and Ekrem Duman. 2011. Detecting Credit Card Fraud by Decision Trees and Support Vector Machines. *IMECS 2011 - International MultiConference of Engineers and Computer Scientists 2011* 1 (March 2011), 442–447.

[50] Michael Schmidt and Hod Lipson. 2009. Distilling Free-Form Natural Laws from Experimental Data. *Science* 324, 5923 (2009), 81–85. doi:10.1126/science.1165893

[51] Bernhard Schölkopf, John C. Platt, John Shawe-Taylor, Alex J. Smola, and Robert C. Williamson. 2001. Estimating the Support of a High-Dimensional Distribution. *Neural Computation* 13, 7 (July 2001), 1443–1471. doi:10.1162/089976601750264965

[52] G. T. Chandra Sekhar, H. S. Behera, Janmenjoy Nayak, Bighnaraj Naik, and Danilo Pelusi. 2021. *Intelligent Computing in Control and Communication*. Lecture Notes in Electrical Engineering, Vol. 702. Springer Singapore. doi:10.1007/978-981-15-8439-8

[53] Aman Singh and Babita Pandey. 2016. An euclidean distance based KNN computational method for assessing degree of liver damage. In *2016 International Conference on Inventive Computation Technologies (ICICT)*, Vol. 1. 1–4. doi:10.1109/INVENTIVE.2016.7823222

[54] N. Sneige. 1999. Cytopathology of the breast: benign and malignant neoplastic conditions. *Journal of Clinical Pathology* 52, 5 (1999), 321–338.

[55] Jian Tang, Zhixiang Chen, Ada W.-C. Fu, and David W. Cheung. 2002. Enhancing effectiveness of outlier detections for low density patterns. In *Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*. Springer, 535–548.

[56] Silviu-Marian Udrescu and Max Tegmark. 2020. AI Feynman: A physics-inspired method for symbolic regression. *Science Advances* 6, 16 (2020). doi:10.1126/sciadv.aay2631

[57] S. Varma and M. Agarwal. 2011. Fine Needle Aspiration Cytology of the Breast. *PMC* (2011). https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3108472/

[58] Samantha Visbeek, Erman Acar, and Floris den Hengst. 2023. Explainable Fraud Detection with Deep Symbolic Classification. arXiv:2312.00586 [cs.LG] https://arxiv.org/abs/2312.00586

[59] Siqi Wang, Qiang Liu, En Zhu, Fatih Porikli, and Jianping Yin. 2017. Hyperparameter Selection of One-class Support Vector Machine by Self-adaptive Data Shifting. *Pattern Recognition* 74 (09 2017). doi:10.1016/j.patcog.2017.09.012

[60] Xiaochun Wang. 2024. *Anomaly Detection in Video Surveillance* (1 ed.). Springer Singapore. doi:10.1007/978-981-97-3023-0 Published: 07 August 2024.

[61] Haolong Xiang, Xuyun Zhang, Hongsheng Hu, Lianyong Qi, Wanchun Dou, Mark Dras, Amin Beheshti, and Xiaolong Xu. 2023. OptIForest: Optimal Isolation Forest for Anomaly Detection. 2379–2387. doi:10.24963/ijcai.2023/264

[62] Dong Xu, Yanjun Wang, Yulong Meng, and Ziying Zhang. 2017. An Improved Data Anomaly Detection Method Based on Isolation Forest. In *2017 10th International Symposium on Computational Intelligence and Design (ISCID)*, Vol. 2. 287–291. doi:10.1109/ISCID.2017.202

[63] X.S. Yang, S. Sherratt, N. Dey, and A. Joshi. 2021. *Proceedings of Sixth International Congress on Information and Communication Technology: ICICT 2021, London, Volume 4*. Springer Nature Singapore. https://link.springer.com/book/10.1007/978-981-16-2102-4?page=3

[64] Bo Zong, Qi Song, Martin Renqiang Min, Wei Cheng, Cristian Lumezanu, Dae ki Cho, and Haifeng Chen. 2018. Deep Autoencoding Gaussian Mixture Model for Unsupervised

Anomaly Detection. In *International Conference on Learning Representations*. https://api.semanticscholar.org/CorpusID:51805340

A. Additional Figures

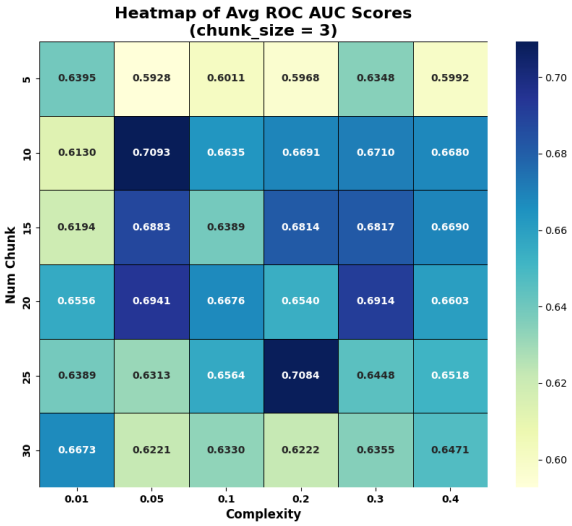


Figure A.1.: Heatmap of Average ROC AUC Scores Across Number of Chunks and Complexity with Chunk Size 3.

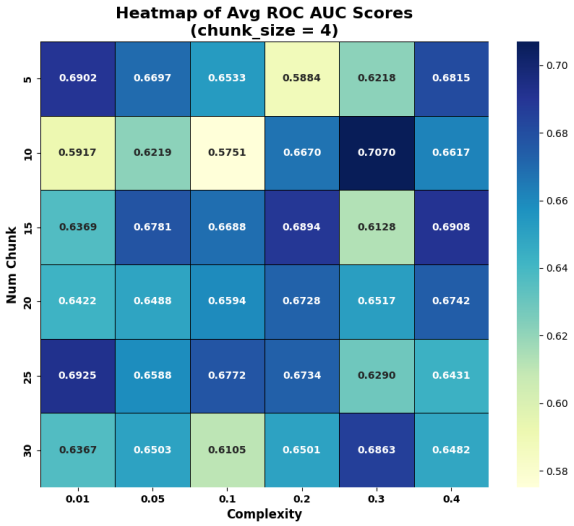


Figure A.2.: Heatmap of Average ROC AUC Scores Across Number of Chunks and Complexity with Chunk Size 4.

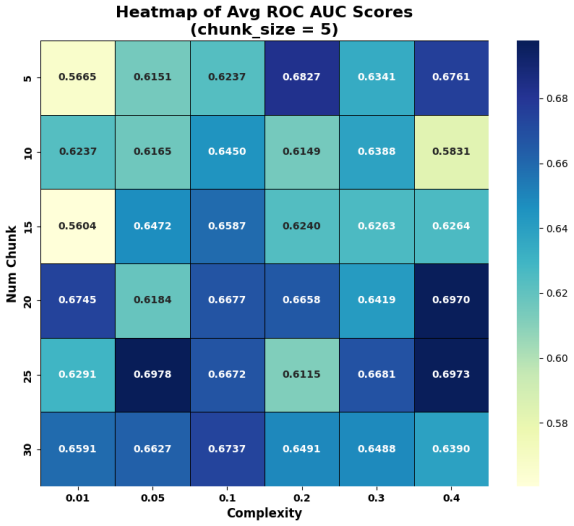


Figure A.3.: Heatmap of Average ROC AUC Scores Across Number of Chunks and Complexity with Chunk Size 5.

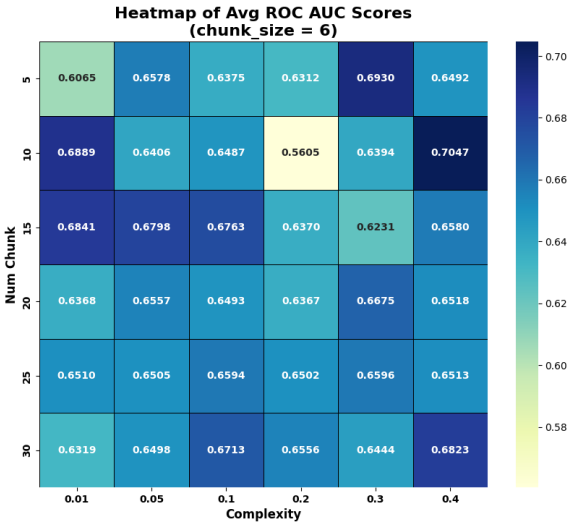


Figure A.4.: Heatmap of Average ROC AUC Scores Across Number of Chunks and Complexity with Chunk Size 6.

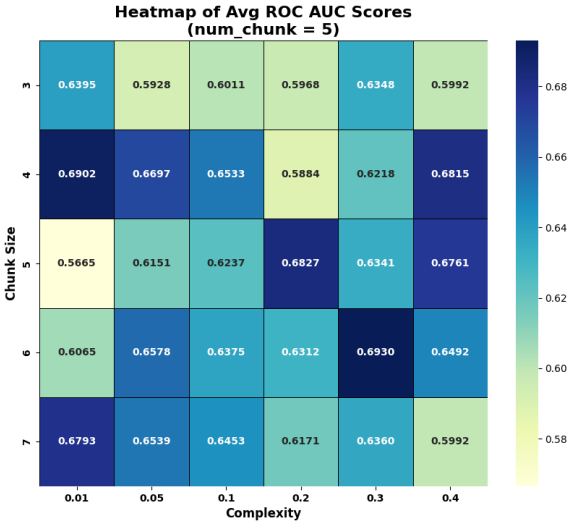


Figure A.5.: Heatmap of Average ROC AUC Scores Across Chunk Size and Complexity with Number of Chunks 5.

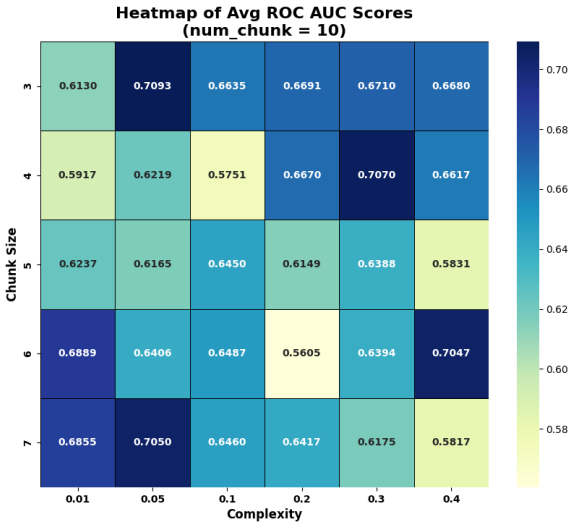


Figure A.6.: Heatmap of Average ROC AUC Scores Across Chunk Size and Complexity with Number of Chunks 10.

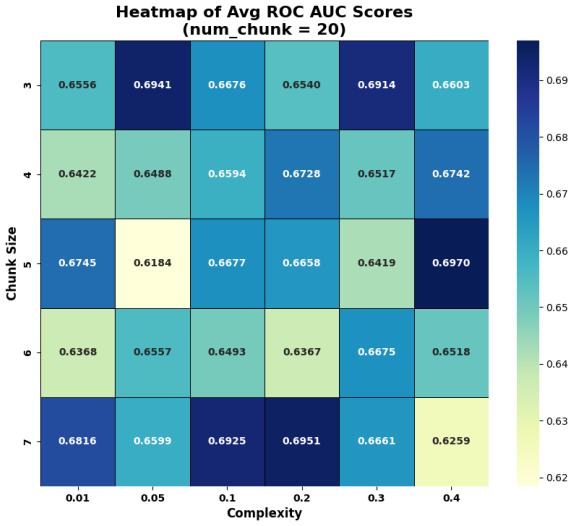


Figure A.7.: Heatmap of Average ROC AUC Scores Across Chunk Size and Complexity with Number of Chunks 20.

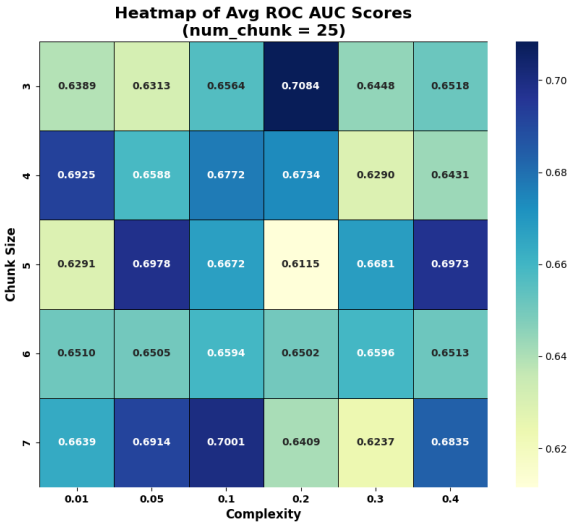


Figure A.8.: Heatmap of Average ROC AUC Scores Across Chunk Size and Complexity with Number of Chunks 25.

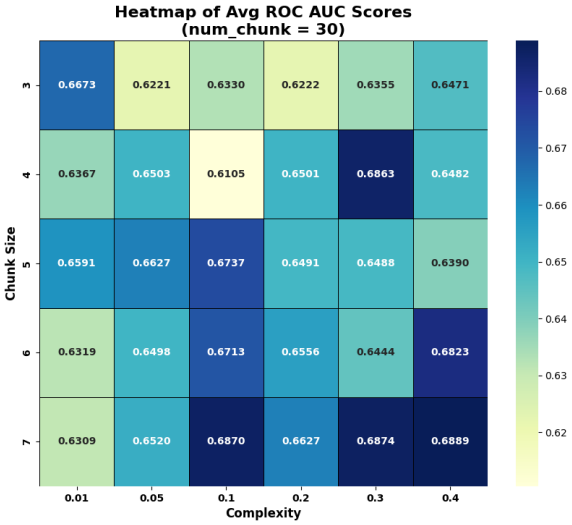


Figure A.9.: Heatmap of Average ROC AUC Scores Across Chunk Size and Complexity with Number of Chunks 30.

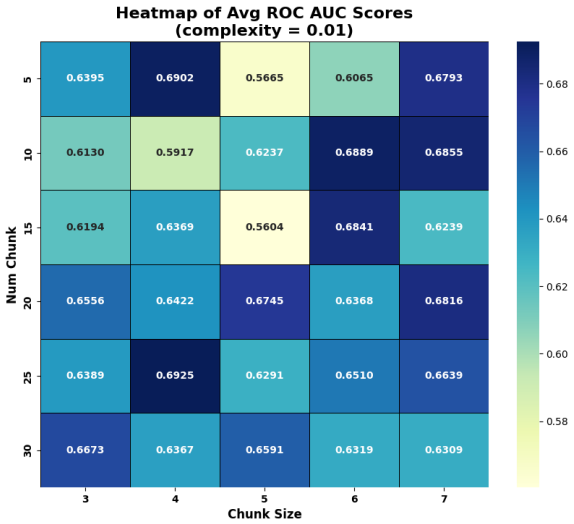


Figure A.10.: Heatmap of Average ROC AUC Scores Across Number of Chunks and Chunk Size with Complexity 0.01.

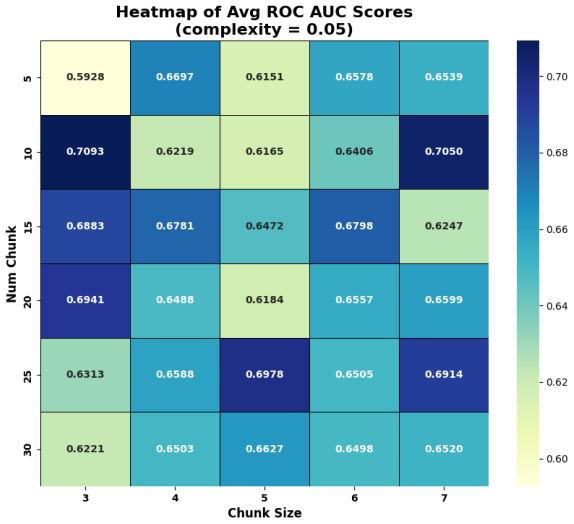


Figure A.11.: Heatmap of Average ROC AUC Scores Across Number of Chunks and Chunk Size with Complexity 0.05.

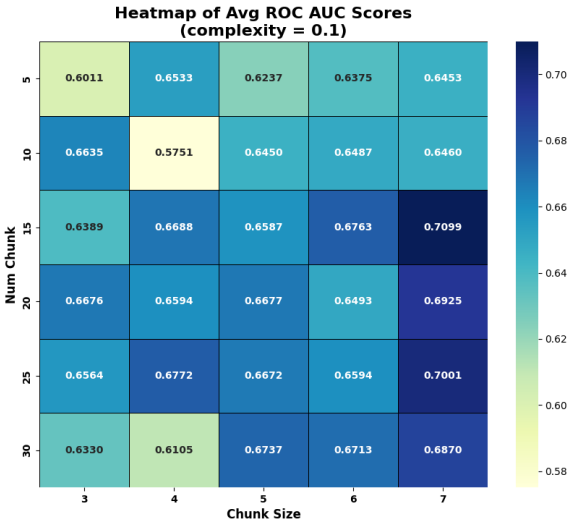


Figure A.12.: Heatmap of Average ROC AUC Scores Across Number of Chunks and Chunk Size with Complexity 0.1.

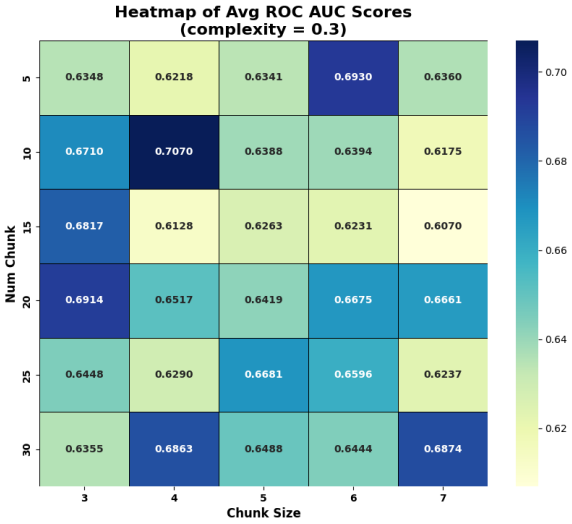


Figure A.13.: Heatmap of Average ROC AUC Scores Across Number of Chunks and Chunk Size with Complexity 0.3.

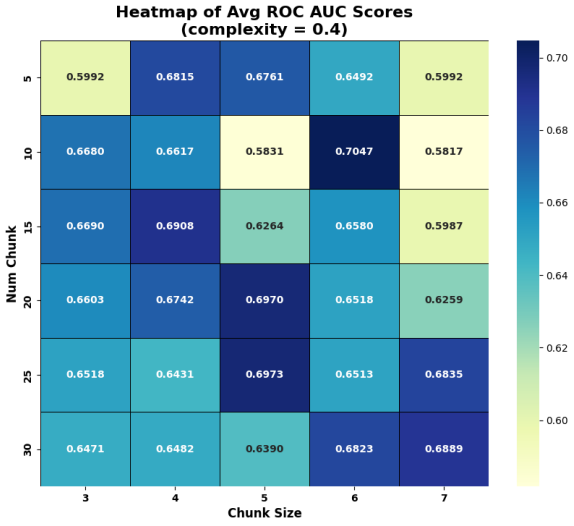


Figure A.14.: Heatmap of Average ROC AUC Scores Across Number of Chunks and Chunk Size with Complexity 0.4.

Eidesstattliche Versicherung

(Affidavit)

Hossain, Md Maruf

Name, Vorname
(surname, first name)

230155

Matrikelnummer
(student ID number)

☐ Bachelorarbeit
(Bachelor's thesis)

☑ Masterarbeit
(Master's thesis)

Titel
(Title)

From Data to Equations: Evolutionary Mathematical Function Learning for Interpretable Outlier Detection

Ich versichere hiermit an Eides statt, dass ich die vorliegende Abschlussarbeit mit dem oben genannten Titel selbstständig und ohne unzulässige fremde Hilfe erbracht habe. Ich habe keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie wörtliche und sinngemäße Zitate kenntlich gemacht. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

I declare in lieu of oath that I have completed the present thesis with the above-mentioned title independently and without any unauthorized assistance. I have not used any other sources or aids than the ones listed and have documented quotations and paraphrases as such. The thesis in its current or similar version has not been submitted to an auditing institution before.

Dortmund, 06-10-2025

Ort, Datum
(place, date)

MdMarufHossain

Unterschrift
(signature)

Belehrung:
Wer vorsätzlich gegen eine die Täuschung über Prüfungsleistungen betreffende Regelung einer Hochschulprüfungsordnung verstößt, handelt ordnungswidrig. Die Ordnungswidrigkeit kann mit einer Geldbuße von bis zu 50.000,00 € geahndet werden. Zuständige Verwaltungsbehörde für die Verfolgung und Ahndung von Ordnungswidrigkeiten ist der Kanzler/die Kanzlerin der Technischen Universität Dortmund. Im Falle eines mehrfachen oder sonstigen schwerwiegenden Täuschungsversuches kann der Prüfling zudem exmatrikuliert werden. (§ 63 Abs. 5 Hochschulgesetz - HG -).

Die Abgabe einer falschen Versicherung an Eides statt wird mit Freiheitsstrafe bis zu 3 Jahren oder mit Geldstrafe bestraft.

Die Technische Universität Dortmund wird ggf. elektronische Vergleichswerkzeuge (wie z.B. die Software „turnitin") zur Überprüfung von Ordnungswidrigkeiten in Prüfungsverfahren nutzen.

Die oben stehende Belehrung habe ich zur Kenntnis genommen:

Official notification:
Any person who intentionally breaches any regulation of university examination regulations relating to deception in examination performance is acting improperly. This offense can be punished with a fine of up to EUR 50,000.00. The competent administrative authority for the pursuit and prosecution of offenses of this type is the Chancellor of TU Dortmund University. In the case of multiple or other serious attempts at deception, the examinee can also be unenrolled, Section 63 (5) North Rhine-Westphalia Higher Education Act (*Hochschulgesetz, HG*).

The submission of a false affidavit will be punished with a prison sentence of up to three years or a fine.

As may be necessary, TU Dortmund University will make use of electronic plagiarism-prevention tools (e.g. the "turnitin" service) in order to monitor violations during the examination procedures.

I have taken note of the above official notification:*

Dortmund, 06-10-2025

Ort, Datum
(place, date)

MdMarufHossain

Unterschrift
(signature)

***Please be aware that solely the German version of the affidavit ("Eidesstattliche Versicherung") for the Bachelor's/ Master's thesis is the official and legally binding version.**

Ergänzung zur Eidesstattlichen Versicherung (Affidavit) hinsichtlich des Einsatzes von generativen IT-/KI-gestützten Sprachmodellen

Hossain, Md Maruf
Name, Vorname
(surname, first name)

230155
Matrikelnummer
(student ID number)

☐ Bachelorarbeit
(Bachelor's thesis)

☑ Masterarbeit
(Master's thesis)

Titel
(Title)

From Data to Equations: Evolutionary Mathematical Function Learning for Interpretable Outlier Detection

In Ergänzung zur abgegebenen eidesstattlichen Versicherung (Affidavit) versichere ich, dass ich bei der Erstellung der vorliegenden Abschlussarbeit betreut von Prof. Dr. Emmanuel Müller (Erstbetreuer) und Dr. Simon Klüttermann (Zweitbetreuer) im Falle des Einsatzes von generativen IT-/KI-gestützten Sprachmodellen diese Werkzeuge in der Übersicht verwendeter Hilfsmittel mit ihrem Produktnamen und der Art des Hilfsmittels, der Art der Verwendung und ggf. den jeweiligen Prompts vollständig aufgeführt habe (siehe beispielhaft nachfolgende Tabelle). Erlaubte Einsatzzwecke sind unter anderem Optimierung selbstverfasster Texte hinsichtlich Grammatik, Rechtschreibung und Schreibstil, Brainstorming bei der Forschungsfrage oder Unterstützung bei der Literaturrecherche, die die Eigenständigkeit bei der Bearbeitung der Abschlussarbeit nicht einschränken. Ferner versichere ich, dass über die genannten Beispiele hinausgehende Verwendung von IT-/KI-gestützten Sprachmodellen mit dem/der Betreuer:in der Arbeit abgesprochen wurde.
Auf die Belehrung hinsichtlich der Folgen von Täuschungsversuchen und der Abgabe einer falschen Versicherung an Eides statt wird auf die eidesstattliche Versicherung und die Prüfungsordnung verwiesen. Diese Belehrung habe ich zur Kenntnis genommen.

In addition to the affidavit submitted, I hereby affirm that I have used generative IT-/AI-supported language models in the preparation of this thesis under the supervision of Prof. Dr. Emmanuel Müller (first supervisor) and Dr. Simon Klüttermann (second supervisor) and that I have listed these tools in full in the overview of tools used with their product name and the type of tool, the type of use and, if applicable, the respective prompts (see table below as an example). Permitted uses include optimization of self-authored texts with regard to grammar, spelling and writing style, brainstorming on the research question or support in literature research that does not restrict independence in the processing of the thesis. Furthermore, I assure that the use of IT-/AI-supported language models beyond the examples mentioned has been discussed with the supervisor(s) in the thesis.
Please refer to the affidavit and the examination regulations for information on the consequences of attempted cheating and making a false declaration in lieu of an oath. I have taken note of these instructions.

Verwendete IT-/KI-gestützte Sprachmodelle/IT/AI-supported language models used

Produktname/ Productname	Art des Hilfsmittels/Type of tool	Art der Verwendung/Type of use	Prompt	Textstelle(n)/ Text passage(s)
z.B./e.g. ChatGPT	KI-Textgenerator/AI text generator	Brainstorming zur Forschungsfrage/on the research question	Mögliche Fragen zum Thema/Possible questions on the topic	-
z.B./e.g. DeepL, Write	KI-Schreibassistent/AI writing assistant	Optimierung des Kapitels XY, z.B. Rechtschreibung, Grammatik etc./Optimization of chapter XY, e.g. spelling, grammar etc.		z.B. Abschnitt 1.2, 1.3, 2.3-2.5 und 3.1/e.g. sections 1.2, 1.3, 2.3-2.5 and 3.1

Dortmund, 06-10-2025
Ort, Datum
(place, date)

MdMarufHossain
Unterschrift
(signature)

Product Name	**Type of Application**	**Exemplary Prompt**
ChatGPT and Grok	Code Debugging	Why is this function giving me an error? Help me to find the error.
ChatGPT	Code Reformatting	Combine these code chunks into a single code.
ChatGPT and Grok	Report Writing	Would you assist me with interpreting the meaning of this statement? Please assist me in identifying any mistakes in grammar or spelling. Rephrase this sentence in a simple and easy to understand structure.
ChatGPT	BibTeX	Create a standard Bib entry for this Paper.

Table A.1.: Examples of prompts to support Affidavit