BUILDING OUTLIER DETECTION FRAMEWORK BY USING AUTOMATED MACHINE LEARNING METHODS

(OTOMATİK MAKİNE ÖĞRENİMİ YÖNTEMLERİNİ KULLANARAK ANOMALİ TESPİT ÇERÇEVESİ OLUŞTURMA)

by

Mustafa KURTOGLU, B.S.

Thesis

Submitted in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE

in

SMART SYSTEMS ENGINEERING

in the

GRADUATE SCHOOL OF SCIENCE AND ENGINEERING

of

GALATASARAY UNIVERSITY

June 2024

ACKNOWLEDGEMENTS

I would like to thank my thesis advisor Ahmet Teoman Naskali, who provided his help and support throughout my thesis study and always spared his time to help me with the difficulties I encountered. In addition, I would like to thank my family, whose financial and moral strength I always felt with me throughout my education life, and my beloved wife, Busra Karaali Kurtoğlu, who never left me alone in any difficult time.

June 2024

Mustafa KURTOGLU

TABLE OF CONTENTS

LIST OF ABBREVATIONS

AutoML : Automated Machine Learning
DBSCAN : Density-Based Spatial Clustering of Applications with Noise
DTW : Dynamic Time Warping
EncDecAD : Encoder-Decoder Anomaly Detection
EVA : Extreme Value Analysis
GNN : Graph Neural Network
GWR : Geographically Weighted Regression
HBOS : Histogram-based Outlier Score
IF-LOF : Isolation Forest - Local Outlier Factor
k-NN : k-Nearest Neighbors
KDE : Kernel Density Estimation
LOF : Local Outlier Factor
LSTM : Long Short-Term Memory
NAR : Neural Autoregression
OPTICS : Ordering Points to Identify the Clustering Structure
PCA : Principal Component Analysis
SVM : Support Vector Machine
TranAD : Transformer-based Anomaly Detection
VAE : Variational AutoEncoder
XGBoost : Extreme Gradient Boosting

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

In terms of both research academic and general purposes, the importance of machine learning techniques is undeniable. However, these methods rely heavily on manual intervention for data refinement and parameter optimization, posing a challenge for researchers of varying levels of expertise. This reliance creates significant barriers for users with varying expertise levels, as existing frameworks struggle to manage diverse data types—structured, unstructured, and semi-structured—thereby compromising effective anomaly detection and limiting comprehensive data analysis.

This thesis introduces an Automated Machine Learning (AutoML) framework designed for robust outlier detection across multiple data structures. Unlike traditional AutoML systems that necessitate some pre-classification of data or manual algorithm selection, this framework autonomously classifies datasets, discerns their characteristics, and directs them through a tailored pipeline to apply suitable outlier detection algorithms. This end-to-end automation is rare in the realm of outlier detection and represents a significant advancement.

The framework's core functionality includes an automatic classification of datasets into categories like spatial, time-series, and dimensional statistical, based on intrinsic characteristics. This classification informs tailored preprocessing techniques that enhance data quality for effective outlier detection. For example, spatial datasets undergo geospatial transformations, while time-series data are adjusted for seasonality. The framework was tested across 100 diverse datasets from fields including finance, healthcare, and social media. It demonstrated a reduction in processing time and anincrease in precision, with a maintained recall rate exceeding 85 percent across majority of datasets. Additionally, the framework dynamically selects outlier detection algorithms based on the data type, enhanced by a meta-learning component that utilizes historical data to optimize algorithm selection. An early stopping mechanism conserves resources by halting processing once performance thresholds are met, ensuring efficiency and scalability.

Overall, this paper contributes a novel and versatile framework that streamlines the outlier detection process, offering a systematic approach similar to an AutoML system. It paves the way for enhanced efficiency and effectiveness in outlier detection tasks across different domains and datasets. Furthermore, the integration of outlier detection mechanisms within this enhanced data environment is anticipated to yield faster identification of irregularities without compromising accuracy or reliability. The framework aims to accelerate the detection process and increasing the efficiency of outlier identification tasks. The comprehensive evaluation across diverse datasets seeks to showcase the framework's adaptability and consistent performance, validating its ability to automate outlier detection across varying data types and scenarios. The successful demonstration of this framework would mark a substantial step in simplifying and refining outlier detection processes in data analysis across multiple domains and datasets.

Keywords: Outlier Detection, Framework, AutoML, Dataset Taxonomy

ÖZET

Akademik araştırmalar ve genel amaçlar açısından makine öğrenimi tekniklerinin önemi tartışılmazdır. Ancak, bu yöntemler, veri iyileştirme ve parametre optimizasyonu için büyük ölçüde manuel müdahaleye dayanır ve bu durum, farklı uzmanlık seviyelerindeki araştırmacılar için bir zorluk oluşturur. Bu bağımlılık, mevcut yapılarının yapılandırılmış, yapılandırılmamış ve yarı yapılandırılmış gibi çeşitli veri türlerini yönetmekte zorlanması nedeniyle, kullanıcılar için önemli engeller yaratır. Bu durum, etkin anomali tespiti yeteneklerini sınırlandırır ve kapsamlı veri analizini kısıtlar.

Bu tez, çeşitli veri yapılarında güçlü anomali tespiti için tasarlanmış bir Otomatik Makine Öğrenimi (AutoML) yapısını tanıtmaktadır. Geleneksel AutoML sistemlerinin bazı veri ön sınıflandırması veya manuel algoritma seçimi gerektirmesinin aksine, bu yapı, veri setlerini bağımsız olarak sınıflandırır, özelliklerini belirler ve uygun anomali tespiti algoritmalarını uygulamak için onları özel bir işlem hattından geçirir. Bu uçtan uca otomasyon, anomali tespiti alanında nadirdir ve önemli bir ilerlemeyi temsil eder.

Yapının temel işlevselliği, veri setlerini içsel özelliklerine dayalı olarak mekansal, zaman serisi ve boyutsal istatistiksel gibi kategorilere otomatik olarak sınıflandırmayı içerir. Bu sınıflandırma, etkin anomali tespiti için veri kalitesini artıran özel ön işleme tekniklerini bilgilendirir. Örneğin, mekansal veri setleri coğrafi dönüşümlere tabi tutulurken, zaman serisi verileri mevsimselliğe göre ayarlanır. Yapı, finans, sağlık ve sosyal medya gibi alanlardan 100 farklı veri setinde test edilmiştir. İşleme süresinde azalma ve doğrulukta artış gösterirken, veri setlerinin çoğunda yüzde 85'in üzerinde bir geri çağırma oranını korumuştur. Ayrıca, yapı, veri türüne bağlı olarak anomali tespiti algoritmalarını dinamik olarak seçer ve algoritma seçimini optimize etmek için tarihi verileri kullanan bir meta-öğrenme bileşeni ile geliştirilmiştir. Erken durdurma mekanizması, performans eşiklerine ulaşıldığında işlemi durdurarak kaynakları korur ve verimliliği artırır.

Genel olarak, bu çalışma, anomali tespiti sürecini kolaylaştıran ve AutoML sistemine benzer sistematik bir yaklaşım sunan yenilikçi ve çok yönlü bir yapı katkısında bulunmaktadır. Farklı alanlar ve veri setleri arasında anomali tespit görevlerinde artırılmış verimlilik ve etkinlik sağlama yolunu açar. Ayrıca, bu gelişmiş veri ortamı içinde anomali tespit mekanizmalarının entegrasyonunun, doğruluk veya güvenilirlikten ödün vermeden düzensizliklerin daha hızlı tespit edilmesini sağlaması beklenmektedir. Yapı, anomali tespiti görevlerinin etkinliğini artırmayı ve tespit sürecini hızlandırmayı amaçlamaktadır. Çeşitli veri setleri üzerindeki kapsamlı değerlendirme, yapının uyarlanabilirliğini ve tutarlı performansını sergilemeyi amaçlar, farklı veri türleri ve senaryolar arasında anomali tespitini otomatikleştirme yeteneğini doğrular. Bu yapının başarılı bir şekilde gösterilmesi, birçok alanda ve veri setinde anomali tespiti süreçlerini basitleştirme ve iyileştirme yolunda önemli bir adım olacaktır.

Anahtar Kelimeler : Anomali Tespiti, Otomatik Makine Öğrenmesi, Veriseti Sınıflandırması

1 INTRODUCTION

In the realm of data science and machine learning, the quest for efficient, accurate, and automated data analysis tools is more pertinent than ever. The exponential growth of data in various domains demands methods that can not only process volumes efficiently but also adapt to diverse data types and structures swiftly. This thesis adapts an innovative framework aimed at revolutionizing the field of outlier detection by leveraging the principles of Automated Machine Learning (AutoML). This advanced framework is designed to handle the complexities associated with different data forms—structured, unstructured, and semi-structured—thus enabling a broader spectrum of users, from seasoned researchers to novices, to perform reliable anomaly detection without the necessity for deep technical expertise in data science.

The identification of outliers is crucial in many fields, including fraud detection, network security, health informatics, and beyond. Traditional methods often require significant manual intervention for data preprocessing and parameter tuning, making the process time-consuming and often impractical for large datasets or real-time processing. Furthermore, the lack of flexibility in adapting to various data types often leads to suboptimal performance, where anomalies are either not detected accurately or are falsely flagged, leading to either overlooked threats or unnecessary alarms. This thesis argues for the necessity of an advanced AutoML-driven outlier detection system that addresses these limitations by providing a robust, adaptable, and efficient solution.

Automated Machine Learning (AutoML) represents a fundamental shift in the approach to predictive analytics. It automates the process of applying machine learning to real-world problems, minimizing the manual efforts required in model selection, feature engineering, and tuning. However, the existing AutoML solutions have been predominantly focused on supervised and semi-supervised learning tasks with less emphasis on unsupervised tasks such as outlier detection. This thesis extends the AutoML paradigm by not specifically focusing on one type algorithms but all kinds capable of identifying outliers in various datasets effectively.

The primary objective of this research is to develop a framework that automates the process of outlier detection across different types of datasets, thereby reducing the reliance on human intervention and expertise. The contributions of this thesis are threefold : Development of a Dataset Taxonomy : This research first categorizes datasets into structured, unstructured, and semi-structured types, providing a systematic approach to handle each type with the most suited preprocessing techniques. This taxonomy is crucial for setting the stage for effective outlier detection.

Integration of Diverse Outlier Detection Algorithms : A comprehensive review and integration of various outlier detection algorithms, including newer methods such as Isolation Forest and Autoencoders, are presented. Each algorithm's adaptability to different types of data and its effectiveness in identifying anomalies are thoroughly evaluated. Automated Selection and Configuration of Algorithms : The core of the proposed framework is the automated selection and configuration of the most appropriate outlier detection algorithms based on the dataset characteristics. This process is powered by meta-learning, which utilizes historical data to predict the most effective algorithmic strategies for new datasets.

The effectiveness of the proposed framework is empirically validated through extensive computational studies across 100 diverse datasets from domains such as finance, healthcare, and social media. These studies demonstrate a 40 percent reduction in processing time and a 20 percent increase in precision, while maintaining a recall rate exceeding 85 percent across all datasets. This significant enhancement in performance not only underscores the framework's capability to adapt to various data types but also its efficiency in real-world applications. The dynamic selection and configuration of algorithms, driven by a meta-learning component, proved crucial in achieving these results, highlighting the framework's ability to learn and improve continuously.

Following this introduction, Chapter 2 provides a detailed literature review, focusing on existing methodologies in dataset categorization and outlier detection algorithms. Chapter 3 describes the model formulation, laying the theoretical foundation for the proposed AutoML framework. Chapter 4 elaborates on the solution methodology, detailing the implementation of the automated system. Chapter 5 presents computational studies to demonstrate the framework's effectiveness across various scenarios. Chapter 6 concludes with the summary and implications for future research.

In sum, this thesis not only contributes a novel framework to the field of machine learning but also sets a precedent for future research in automated data analysis systems, emphasizing efficiency, adaptability, and ease of use without compromising on analytical rigor. By harnessing the power of AutoML, this work aims to democratize the benefits of advanced data analysis, making it accessible to a broader audience and significantly enhancing the ability to make informed decisions across different sectors.

2 LITERATURE REVIEW

2.1 Dataset Taxonomy

When exploring anomaly detection in datasets, it's clear that one classification framework can't capture all the complexity and diversity in the data. This realization leads to using a multi-dimensional classification approach, which understands the many sides of datasets and anomaly detection techniques (Sebestyen et al., 2018). Using detailed taxonomies from recent research, this method stresses the importance of evaluating datasets and detection methods in several key areas.

By searching databases for articles on "Dataset Taxonomy" in English, over four million articles have been found . Here are the criteria :

- Studies done after 2011.
- Academic journals, books, dissertations, conference papers, and articles
- Studies written in English.

Unrelated or less relevant publications were eliminated after titles and abstracts were reviewed. After giving the remaining articles a closer look, 107 of them were selected for full-text review. Following a careful examination, 65 more were eliminated as unnecessary. Fifteen articles made up the final sample. The last sample, which was selected in order to comprehend the origins, acceptance, and definitions of "Dataset Taxonomy," was thoroughly examined. Tables were used to help arrange the dataset taxonomy literature in order to facilitate understanding and inform future research.

Studies in Table 2.1 show the diverse and changing nature of research in dataset taxonomy and outlier detection, showing big progress from early studies to recent innovations. By categorizing datasets and using different detection algorithms, this research aims to build a strong AutoML framework that adress the problems found in these studies. The thorough review and testing of different techniques show the framework's flexibility and efficiency, paving the way for better anomaly detection.

Madalina Olteanu, Fabrice Rossi, and F. Yger conducted a meta-survey on outlier and anomaly detection in 2023 (Olteanu et al., 2023). The study provides an overview of current trends and gaps in the field by summarizing recent achievements and methodologies. Comparably, the research conducted in 2022 by Bahri, Maroua, Salutari, Flavia, Putina, Andrian, and Sozio, Mauro demonstrates novel methods and current advancements in AutoML and unsupervised learning for outlier detection (Bahri et al., 2022). Understanding the most recent advancements in the subject requires a grasp of these studies.

The taxonomy and classification of datasets are essential components of outlier detection. The goal of recent research has been to develop more effective techniques for local outlier detection in data streams. Ai-hua Li, Weijia Xu, Zhidong Liu, and Yong-Feng Shi's 2021 paper describes an improved approach for local outlier detection for data streams that increases accuracy and efficiency (Li et al., 2021). Similar to this, in 2012, Seyed Hesamodin Karimian, M. Kelarestaghi, and S. Hashemi presented an improved incremental technique with an emphasis on real-time processing for identifying outliers in data streams (Karimian et al., 2012).

Outlier detection in specific areas has also been studied a lot. The 2017 study by Shivani Garg, Mukul Varshney, A. Rajpoot, and Jyotsna looks at outlier detection methods for wireless sensor networks, giving a classification based on how suitable they are for different applications (Garg et al., 2017). Also, the 2012 study by Y. Zhang, N. A. S. Hamm, N. Meratnia, Alfred Stein, Marlies Van de Voort, and P. Havinga uses statistical methods to detect outliers in wireless sensor networks, showing the use of well-known statistical techniques in this new area (Zhang et al., 2012). Another study, by Horacio D. Kuna, Ramón García-Martínez, and F. R. Villatoro in 2014, focuses on detecting outliers in system audit logs to improve security and operations (Kuna et al., 2014).

Surveys and review articles give valuable classifications and comparisons of outlier detection techniques. The 2021 study by Omar Alghushairy, Raed Alsini, T. Soule, and Xiaogang Ma offers a systematic review of Local Outlier Detection, looking at various methods and their effectiveness (Alghushairy et al., 2021). Similarly, the surveys by Ch. Nagamani and Dr. Ch. Suneetha in 2015 and by Vipnesh Jha and Om Veer Singh Yadav in 2012 give detailed reviews of outlier detection strategies, highlighting different approaches and their uses in many fields (Nagamani and Suneetha, 2015; Jha and Yadav, 2010). These surveys help a lot in understanding the taxonomy of datasets used in various outlier detection methods.

Different methods for outlier detection have been explored in recent research. The 2017 study talks about recognizing outliers in large databases using distance-based methods, focusing on high-dimensional data (Park, 2022). Deepti Mishra and Devpriya Soni's 2016 work (Mishra and Soni, 2016) compares angle- and distance-based methods for outlier detection and offers suggestions for further research. In a different study, K-Means and Expectation Maximization are compared for detecting regression outliers in time series. The study highlights the boundaries of each method in terms of optimality and convergence (Qi et al., 2016).

Understanding recent developments in outlier detection is aided by historical viewpoints and foundational research. In order to address new issues in huge datasets, Xiaogang Su and Chih-Ling Tsai's 2011 study provides a non-technical review of popular techniques for outlier detection (Su and Tsai, 2011). Similar to this, Yang Zhang, N. Meratnia, and P. Havinga's 2010 study (Zhang et al., 2010) provides a technique-based taxonomy for practical application by surveying outlier detection methods for wireless sensor networks.

TABLE 2.1 – Overview of Research Articles on Outlier Detection

Name of the Article	Year	Authors	Explanation
Meta-survey on outlier and anomaly detection	2023	Madalina Olteanu, Fabrice Rossi, F. Yger	Does a meta-survey of the field of outlier and anomaly detection, summarizing recent advances and methods.
AutoML : state of the art with a focus on anomaly detection, challenges, and research directions	2022	Bahri, Maroua, Salutari, Flavia, Putina, Andrian, Sozio, Mauro	This research gathers important contributions in AutoML and unsupervised learning for Outlier Detection, showing new approaches and recent advancements.
Improved incremental local outlier detection for data streams based on the landmark window model	2021	Ai-hua Li, Weijia Xu, Zhidong Liu, Yong-Feng Shi	Talks about a better local outlier detection method for data streams, making it more efficient and accurate.
A Review of Local Outlier Factor Algorithms for Outlier Detection in Big Data Streams	2021	Omar Alghushairy, Raed Alsini, T. Soule, Xiaogang Ma	The study looks at Local Outlier Detection, reviewing existing literature to assess various methods and their effectiveness.
A Review on Outlier/Anomaly Detection in Time Series Data	2021	Anne Blázquez-García, Angel Conde, Usue Mori, Jose A. Lozano	Talks about the role of meta-learning in AutoML frameworks and gives an overview of top techniques in unsupervised outlier detection for time series data.
A Comparative Study for Outlier Detection Methods in High Dimensional Text Data	2022	Park, Cheong Hee	Discusses finding outliers in large databases using distance-based methods, focusing on high-dimensional data.
A Comparative Review of Outlier Detection Techniques for Wireless Sensor Networks	2017	Shivani Garg, Mukul Varshney, A. Rajpoot, Jyotsna	Reviews outlier detection methods for wireless sensor networks, giving a classification based on method suitability for different applications.
An Integrated Method for Outlier Detection with Analytical Study of Distance Based and Angle Based Approaches	2016	Deepti Mishra, Devpriya Soni	Analyzes and compares distance-based and angle-based methods for outlier detection, giving insights for further research.
A Survey for Outlier Detection and its Strategies	2015	Ch. Nagamani, Dr. Ch. Suneetha	Surveys various strategies for outlier detection, emphasizing different methods and their uses in multiple fields.
Outlier detection in audit logs for application systems	2014	Horacio D. Kuna, Ramón García-Martínez, F. R. Villatoro	Focuses on finding outliers in system audit logs to improve security and operations.
Outlier Mining in Large Data Set : An Efficient Unsupervised Approach	2013	R. Prabahari, Dr. V. Thiagarasu, P. Narendran	Explores efficient unsupervised methods for outlier detection in large datasets, using clustering to find anomalies.
An overview of outlier detection technique developed for wireless sensor networks	2013	Oussama Ghorbel, M. W. Jmal, W. Ayedi, H. Snoussi, M. Abid	Gives a full overview of outlier detection methods made for wireless sensor networks.
K-Means : An Effective and Efficient K-Means Clustering Algorithm	2016	Qi, Jianpeng and Yu, Yanwei and Wang, Lihong and Liu, Jinglei	Compares K-Means and Expectation Maximization for finding regression outliers in time series, highlighting the limits of each method in convergence and optimality.
Outlier Detection Techniques over Streaming Data in Data Mining : A Research Perspective	2013	P. Chandore, P. Chatur	Talks about challenges and solutions for finding outliers in streaming data, focusing on the dynamic nature of data streams in mining.
I-IncLOF : Improved incremental local outlier detection for datastreams	2012	Seyed Hesamodin Karimian, M. Kelarestaghi, S. Hashemi	Introduces a better incremental method for finding outliers in data streams, emphasizing real-time processing.
Outlier Detection Techniques and Cleaning of Data for Wireless Sensor Networks : A Survey	2012	Vipnesh Jha, Om Veer Singh Yadav	Reviews outlier detection methods and data cleaning methods specifically for wireless sensor networks.
Statistics-based outlier detection for wireless sensor networks	2012	Y. Zhang, N. A. S. Hamm, N. Meratnia, Alfred Stein, Marlies Van de Voort, P. Havinga	Uses statistical methods to find outliers in wireless sensor networks, showing the use of known statistical techniques in this new field.
Outlier detection	2011	Xiaogang Su, Chih-Ling Tsai	Gives a non-technical overview of popular methods for finding outliers, addressing new challenges in large datasets.
Outlier Detection Techniques for Wireless Sensor Networks : A Survey	2010	Yang Zhang, N. Meratnia, P. Havinga	Gives a survey of outlier detection methods made for wireless sensor networks, offering a method-based taxonomy for practical application selection.

Firstly, it is critical to focus on the essence of anomalies. Anomalies can manifest as either isolated data points that deviate significantly from the norm or intricate patterns that exhibit variations over time or in different contexts. This distinction is crucial when selecting appropriate detection techniques, whether employing statistical analysis, machine learning models, or pattern recognition algorithms. Moreover, the classification and detection processes get more intricate due to the specific type of data, such as univariate, multivariate, time-series, or spatial. These data characteristics require precise methodologies that can effectively detect anomalies within the provided data context.

Furthermore, the connections between data points, whether they are statistical, chronological, or spatial, introduce an additional level of intricacy. Gaining a comprehensive understanding of these relationships is crucial in developing efficient anomaly detection techniques capable of handling the intricacies of real-world data. Employing a multi-view classification method enhances the precision of outlier detection and analysis, while also enhancing the efficiency of managing various datasets.

Furthermore, the distinction between learning types, namely supervised, unsupervised, and semi-supervised, has an influence on the classification system and has implications for the selection and application of detection methods. This classification takes into account the varying degrees of prior information and labeling present in distinct datasets, which in turn affects the selection and application of algorithms.

Through the utilization of a multi-dimensional methodology for dataset categorization,it was possbile to develop more intricate and efficient tactics for detecting anomalies. This framework acknowledges the intricate nature of data and the multitude of elements that impact the process of detection. It assists us in effectively addressing the difficulties of identifying anomalies in different domains, guaranteeing more precise, efficient, and flexible solutions. By employing this methodology, advanced comprehension of the correlation can be gained between data attributes and detection techniques, so enhancing this proficiency in identifying and examining anomalies.

Table 2.2 – Sub Criteria for Classification and Data Types

Sub Criteria for Classification	Data Type	Explanation
Number of attributes	Uni-variate	Points; characterized by a single parameter value
	Multi-variate	A point characterized by a number of attributes or parameter values; the attributes may be numerical, textual, and mixed.
Correlation between data points	Dimensional-Statistical	Data points that follow some statistical or probabilistic rules
	Sequential (time-series)	Time series where a data-point is correlated with the previous samples of the same parameter
	Spatial	Data similar to time-series where points are correlated, but the correlation is between spatially neighboring points; typical for data collected from sensor networks where a physical connection exists between neighboring sensors
	Graph data	A mixture of continuous (numeric) and categorical attributes
Data Types	Simple	Simple few feature dataset
	Binary	Data consisting of two distinct categories or states
	Categorical	Data divided into distinct categories or groups without a specific order
	Continuous	Data that can take any value within a given range; typically numeric
Input Format	Numerical	Data represented by numbers
	Text	Data in textual form
	Image	Data in the form of visual representations, such as photos or diagrams
Learning Type	Supervised	Learning from a labeled dataset where the algorithm is trained on input-output pairs
	Semi-Supervised	Learning from a dataset that contains both labeled and unlabeled data
	Unsupervised	Learning from an unlabeled dataset, where the algorithm tries to identify patterns and structures
Application Based	Various Applications (Production, Sales, Network, etc.)	Data used in a variety of real-world applications such as production monitoring, sales forecasting, and network analysis

2.1.1 Correlation between data points

Dimensional-statistical data refers to datasets that contain information regarding the statistical relationships between multiple attributes (Hodge and Austin, 2004). Often, it is crucial to perceive the statistical correlations that exist between data points when these datasets comprise continuous numerical properties. To identify irregularities in these datasets, modified z-scores and z-score analysis are frequently applied techniques(Hawkins, 1980) . In addition, techniques such as Principal Component Analysis (PCA) aid in the detection and emphasis of departures from the underlying structure of correlation(Jolliffe, 2002) .

The time series Data is a systematically arranged compilation of information that is temporally dependent; the correlation between data points is established by their temporal proximity to one another (Chandola et al., 2009). In order to detect irregularities in time-series data, it is necessary to employ algorithms designed to detect temporal correlations, moving averages, and ARIMA models (McGlohon et al., 2011). In order to simulate long-term dependencies in sequential data, Long Short-Term Memory (LSTM) networks are employed, as illustrated by methodologies such as (Malhotra et al., 2015).

Spatial data refers to datasets in which the individual instances include inherent spatial links (Campos et al., 2016). These datasets may contain attributes that are related to geography or space. Spatial outlier detection approaches, which use spatial correlations, strive to accurately identify abnormalities (Yang and Yue, 2023). Geographically Weighted Regression (GWR) and spatial clustering algorithms such as DBSCAN are frequently employed to identify spatial abnormalities (Fotheringham et al., 2002).

The graph data model represents items as vertices and their relationships as edges, creating a network structure (Aggarwal and Wang, 2010). Graph datasets may exhibit anomalies in the form of nodes that have atypical connections or behaviors within the network. Graph-based outlier detection methods seek to identify anomalies by analyzing irregular connections (Akoglu et al., 2015). Graph neural networks and spectral analysis are commonly employed techniques for identifying structural anomalies in graphs (Wu et al., 2020).

Each form of dataset, including dimensional-statistical, sequential (time-series), spatial, and graph data, exhibits distinct correlations between data points. Consequently, specific outlier identification algorithms must be employed to account for the dataset's structure and the intrinsic interactions among its data points.

2.1.2 Number of attributes

Understanding the structure of datasets and the number of attributes is crucial for developing effective outlier detection algorithms. Datasets can be classified as either uni-variate or multi-variate, based on the number of attributes associated with each data instance.

Uni-variate datasets are made up of instances that are determined by a single attribute. When analyzing these datasets, outliers can be detected by examining the distribution of values within a particular attribute. Statistical measurements like mean, median, or standard deviation are commonly used in anomaly detection approaches to identify deviations from the norm (Hawkins, 1980). There are several methods commonly used to identify outliers in uni-variate data, such as z-scores, modified z-scores, and the IQR approach (Leys et al., 2013).

Multi-variate datasets are composed of instances that are associated with multiple attributes (Yang and Yue, 2023). These datasets can be quite complex when it comes to identifying outliers, as anomalies may arise from the interactions or correlations between multiple characteristics. There are various approaches to detect outliers in multi-variate datasets, such as clustering, distance-based metrics, or machine learning techniques that consider the relationships between different features (McGlohon et al., 2011). Methods such as Mahalanobis distance, Principal Component Analysis (PCA), and machine learning models such as Isolation Forest are frequently employed to tackle the complexity of multi-variate datasets (Breunig et al., 2000; Liu et al., 2008).

The classification of datasets into uni-variate and multi-variate categories has a significant influence on the choice of outlier detection strategies. Customized approaches are developed to tackle the unique challenges posed by datasets that contain either single or multiple properties linked to each instance. Univariate datasets typically analyze the distribution of a single variable, while multivariate datasets require techniques that can handle the relationships and dependencies among multiple variables. Accurately detecting abnormalities and enhancing the dependability of data analysis requires a clear understanding of the differentiation between outliers.

2.1.3 Data Types

It's important to understand the characteristics of datasets, like whether they are categorical, continuous, or binary, to create good outlier detection strategies.

Binary datasets have instances with attributes that have two possible values, usually shown as 0 and 1 (Liu et al., 2008). Finding outliers in binary datasets means looking for instances where these values show unusual or unexpected patterns. One way to do this is to check if binary attributes deviate from expected ratios or patterns. Techniques like frequent pattern mining and association rule mining can also help find rare patterns in binary data (Aggarwal, 2013).

Categorical datasets have attributes with discrete values representing different categories or labels (Hodge and Austin, 2014)). Detecting outliers in categorical datasets involves finding instances that significantly deviate from the majority category or show uncommon combinations of categorical values. You can measure entropy or divergence from expected categorical distributions. One-hot encoding combined with clustering or classification algorithms is also often used to handle categorical outliers (Aggarwal, 2016).

Continuous datasets consist of attributes with numerical values that cover a continuous range (McGlohon et al., 2011). For continuous datasets, outlier detection often uses statistical methods to find instances that deviate significantly from expected distributions, means, or standard deviations. Common methods include z-score, modified z-score, and robust statistical techniques like Median Absolute Deviation (MAD) (Leys et al., 2013). Support vector machines (SVM) and kernel density estimation (KDE) are also used to analyze continuous data (Parzen, 1962; Scholkopf et al., 2001).

Each data type—binary, categorical, and continuous—has its own challenges for outlier detection. By tailoring methods to the specific characteristics of each data type, you can more accurately identify anomalies within the dataset. For binary data, the goal is to detect unusual patterns between the two values. For categorical data, the focus is on deviations from the majority or expected categorical distributions. For continuous data, the key is to find statistical deviations from expected norms.

2.1.4 Input Format

Numerical datasets have attributes with numerical values and make up a big part of the data in outlier detection. Methods like statistical techniques, distance-based algorithms, and clustering are often used to find anomalies in these datasets (Hawkins, 1980).

Textual datasets, like emails, social media posts, or articles, have their own unique challenges for outlier detection. Techniques from Natural Language Processing (NLP), sentiment analysis, or topic modeling are used to find anomalies in textual data (Chandola et al., 2009). Understanding the language and context is key to developing good outlier detection strategies for text-based datasets. Methods like TF-IDF, word embeddings, and anomaly detection techniques like Latent Dirichlet Allocation (LDA) are common (Aggarwal and Zhai, 2012).

Image datasets, which contain visual information, are common in fields like medical imaging, surveillance, or satellite imagery. Finding anomalies in image data needs special techniques from Computer Vision. Methods like Convolutional Neural Networks (CNNs), image feature extraction, or deep learning models are used to identify anomalies in visual datasets (Krizhevsky et al., 2017). Techniques like autoencoders and GANs (Generative Adversarial Networks) are also applied to detect anomalies in image data (Schlegl et al., 2017).

Each input format—Numerical, Text, and Image—has its own challenges for outlier detection. By using domain-specific techniques, it is possible to tailor outlier detection methods to the nature of each data type, ensuring accurate identification of anomalies within different datasets.

2.1.5 Application Based

Categorizing datasets according to their intended use is crucial to detect abnormal patterns. The context in which data is used has a major impact on the selection of appropriate detection methods. Each application requires specific strategies to overcome its unique challenges and characteristics. This requires customized tactics in areas such as production, sales, and network monitoring.

Manufacturing datasets focus on detecting anomalies that often occur in production processes. This helps detect defects, equipment malfunctions or deviations from standard procedures. Control charts and Statistical Process Control (SPC) are frequently used in these fields (Montgomery, 2007). Additionally, ensemble models and neural networks, which are machine learning approaches, are increasingly used in manufacturing fields for predictive maintenance and problem detection (Gao et al., 2015)..

Sales data analysis identifies abnormal trends to detect fraudulent activity, market fluctuations or unusual consumer behavior. Commonly used techniques to detect anomalies in sales data include clustering, regression analysis, and time series forecasting (Bolton and Hand, 2002).

Network datasets are used to detect breaches, intrusions or network failures in cybersecurity and therefore play a crucial role. Commonly used techniques to protect network environments include intrusion detection systems (IDS), signature-based detection, and anomaly-based detection (Patcha and Park, 2007). Deep learning models, specifically autoencoders and recurrent neural networks (RNNs), are used to detect complex cyberthreats (Mirsky et al., 2018)..

Anomaly detection is used in financial contexts, particularly in the field of fraud detection, to detect suspicious transactions, unusual accounting procedures or market irregularities. Various methodologies are used to detect financial fraud, such as supervised learning models, unsupervised clustering methods, and hybrid approaches (Ngai et al., 2011).

Research shows that anomaly detection methods customized to the unique characteristics of each application domain increase the effectiveness and accuracy (Chandola et al., 2009) . This highlights the importance of an application-sensitive approach when developing and selecting anomaly detection methods. Thus, these methods ensure that the data intended to be examined is structured in accordance with the specific characteristics of the data.

2.2 Model Taxonomy

A database search for articles related to "Model Taxonomy" in English yields more than two million articles. The articles accessed were first subjected to selection criteria, which are as follows :

- Studies conducted after 2015 were examined.
- Various sources were examined, including academic journals, conference proceedings, articles, dissertations, and books.
- Only studies in English were considered.

The titles and abstracts of these papers were analyzed, and the ones that were pertinent to the issue were identified. As a result, 55 publications were chosen for a comprehensive evaluation that specifically examined the categorization of models in anomaly detection. Upon thorough examination of these papers, 34 of them were discarded due to their lack of relevance to the research. The final study sample had a total of 21 publications.

The papers underwent meticulous analysis to enhance comprehension of model taxonomy in the anomaly detection domain. The results of the literature study were methodically structured to enhance current understanding and establish the groundwork for future research in this area.

Kaklis et al. (2024) conducted a meta-study on outlier and anomaly identification. They propose a cross-sector approach to trajectory mining and routing, emphasizing its many applicability in other industries (Kaklis et al., 2024) . Thaipisutikul and Chen (2024) introduce a sophisticated deep sequential model that aims to enhance context-awareness in point-of-interest (POI) recommendations (Thaipisutikul and Chen, 2024) . Sunitha et al. (2024) propose an adaptive clustering technique for orbital data streams with the objective of enhancing both the accuracy and efficiency of clustering (Sunitha et al., 2024) .

The study by Ke et al. (2023) develops a group similarity system leveraging unsupervised outlier detection methods for effective big data computing (Ke et al., 2023). Forney (2023) introduces a framework that uses parallel processing techniques for outlier detection in pivotal time series data, aiming to improve processing efficiency and accuracy (Forney, 2023). Zhang et al. (2023) propose a two-stage anomaly detection method that integrates user preference features with a deep fusion model to enhance detection accuracy (Zhang et al., 2023).

Nijhuis and van Lelyveld (2023) use reinforcement learning to develop an efficient outlier detection method for data that is costly to verify, reducing verification costs (Nijhuis and van Lelyveld, 2023). The research by Mazarbhuiya and Shenify (2023) introduces a mixed clustering approach to improve real-time anomaly detection in various datasets (Mazarbhuiya and Shenify, 2023). Arul and Venkatesan (2023) present a deep learning-based approach to enhance anomaly detection efficiency in Wireless Sensor Networks (WSNs) (Arul and Venkatesan, 2023).

Al-Sudani and Riyadh (2023) apply DBSCAN and A* algorithms to detect fraudulent activities among taxi drivers in Baghdad, focusing on route anomalies (Al-Sudani and Riyadh, 2023). Xia et al. (2023) present the ASM-CF model for detecting anomalous trajectories using big data from mobile trajectories, emphasizing its scalability and accuracy (Xia et al., 2023). Schmidl et al. (2022) categorize outlier detection methods and evaluate them comprehensively on 35 different time series datasets (Schmidl et al., 2022).

The study by Ya Su et al. (2021) combines Gaussian mixture models and one-dimensional CNNs within a variational autoencoder framework to detect outliers in machine instances (Su et al., 2021). Lai et al. (2021) revisit the definitions and benchmarks of time series outlier detection, evaluating various algorithms on 35 different datasets (Lai et al., 2021). Thudumu et al. (2020) provide an extensive overview of anomaly detection techniques tailored for high-dimensional big data applications (Thudumu et al., 2020).

Henriques et al. (2020) combine K-Means clustering and XGBoost models to detect anomalies in log datasets, improving detection performance (Henriques et al., 2020). Belhadi et al. (2020) provide a comprehensive review of trajectory outlier detection algorithms, taxonomies, and evaluation metrics, addressing open challenges (Belhadi et al., 2020). Dai et al. (2020) propose a sequential transformation approach for functional outlier detection, offering a new taxonomy of outliers (Dai et al., 2020).

The work of Boukerche et al. (2020) provides a detailed overview of outlier detection methods, models and classification techniques (Boukerche et al., 2020). Yang and Rahardja (2019) propose a new statistical method for effective outlier detection, focusing on determining threshold values for outlier scores (Yang et al., 2019).

A study, present a taxonomy for anomaly detection methods and develop a platform that allows these methods to be used in various applications (Sebestyen et al., 2018). Another study improved the detection accuracy and efficiency by developing an algorithm to detect local outliers in skewed big data (Yan et al., 2017). Also another studyi propose a two-stage approach to anomaly detection for Industry 4.0 : learning normal behavior from historical data and real-time anomaly detection (Stojanovic et al., 2016).

TABLE 2.3 – Overview of Research Articles on Outlier Detection

Name of the Article	Year	Authors	Explanation
Trajectory Mining and Routing : A Cross-Sectoral Approach	2024	Kaklis D, Kontopoulos I, Varlamis I, Emiris I, Varelas T	This study presents a cross-sectoral approach to trajectory mining and routing, highlighting its application in various domains.
An Improved Deep Sequential Model for Context-Aware POI Recommendation	2024	Thaipisutikul T, Chen Y	This paper proposes an enhanced deep sequential model designed to improve point of interest (POI) recommendations based on context awareness.
Adaptive Trajectory Data Stream Clustering	2024	Sunitha G, Kiran J, Nagendra K, Afreen S, Madhavi K, Kothapati N, Kumar V, Hemachandu D	The research introduces an adaptive clustering method for trajectory data streams, aiming to enhance clustering accuracy and efficiency.
GSS : A Group Similarity System Based on Unsupervised Outlier Detection for Big Data Computing	2023	Wenjun Ke, Jianguo Wei, Naixue Xiong, Qingzhi Hou	This study develops a group similarity system leveraging unsupervised outlier detection methods for effective big data computing.
A Framework for Outlier Detection in Pivotal Time Series Data Utilizing Parallel Processing	2023	Forney, Mikol	This framework uses parallel processing techniques for outlier detection in pivotal time series data, aiming to improve processing efficiency and accuracy.
A Two-Stage Anomaly Detection Method Based on User Preference Features and the Deep Fusion Model	2023	Zhang, Sen-Lei, Zhang, Bin, Zhou, Yi-Tao, Guo, Yue-Xuan, Tan, Jing-Lei	This paper proposes a two-stage anomaly detection method that integrates user preference features with a deep fusion model to enhance detection accuracy.
Outlier Detection with Reinforcement Learning for Costly to Verify Data	2023	Nijhuis, Michiel ; Iman van Lelyveld	The study uses reinforcement learning to develop an efficient outlier detection method for data that is costly to verify, reducing verification costs.
A Mixed Clustering Approach for Real-Time Anomaly Detection	2023	Mazarbhuiya, Fokrul Alom, Shenify, Mohamed	This research introduces a mixed clustering approach to improve real-time anomaly detection in various datasets.
A Deep Learning Approach for Efficient Anomaly Detection in WSNs	2023	Arul, Jothi S ; Venkatesan, R.	The paper presents a deep learning-based approach to enhance anomaly detection efficiency in Wireless Sensor Networks (WSNs).
Fraudulent Taxi Driver Detection in Baghdad City Based on DBSCAN and A* Algorithm	2023	Al-Sudani Z, Riyadh M	This study applies DBSCAN and A* algorithms to detect fraudulent activities among taxi drivers in Baghdad, focusing on route anomalies.
An ASM-CF Model for Anomalous Trajectory Detection with Mobile Trajectory Big Data	2023	Xia D, Jiang S, Li Y, Yang N, Hu Y, Li Y, Li H	This paper presents the ASM-CF model for detecting anomalous trajectories using big data from mobile trajectories, emphasizing its scalability and accuracy.
Anomaly Detection in Time Series : A Comprehensive Evaluation	2022	Schmidl S, Wenig P, Papenbrock T	This study categorizes outlier detection methods and evaluates them comprehensively on 35 different time series datasets.
Detecting Outlier Machine Instances Through Gaussian Mixture Variational Autoencoder With One Dimensional CNN	2021	Ya Su, Youjian Zhao, Ming Sun, Shenglin Zhang, Xidao Wen, Yongsu Zhang, Xian Liu, Xiaozhou Liu, Junliang Tang, Wenfei Wu, Dan	The study combines Gaussian mixture models and one-dimensional CNNs within a variational autoencoder framework to detect outliers in machine instances.
Revisiting Time Series Outlier Detection : Definitions and Benchmarks	2021	Kwei-Herng Lai, Daochen Zha, Junjie Xu, Yue Zhao, Guanchu Wang, Xia Hu	This study revisits the definitions and benchmarks of time series outlier detection, evaluating various algorithms on 35 different datasets.
A Comprehensive Survey of Anomaly Detection Techniques for High Dimensional Big Data	2020	Thudumu S, Branch P, Jin J, Singh J	This survey provides an extensive overview of anomaly detection techniques tailored for high-dimensional big data applications.
Combining K-Means and XGBoost Models for Anomaly Detection Using Log Datasets	2020	Henriques J, Caldeira F, Cruz T, Simoes P	This research combines K-Means clustering and XGBoost models to detect anomalies in log datasets, improving detection performance.
Trajectory Outlier Detection : Algorithms, Taxonomies, Evaluation, and Open Challenge	2020	Belhadi A, Djenouri Y, Lin JCW, Cano A	The paper provides a comprehensive review of trajectory outlier detection algorithms, taxonomies, and evaluation metrics, addressing open challenges.
Functional Outlier Detection and Taxonomy by Sequential Transformations	2020	Dai W, Mrkvička T, Sun Y, Genton MG	This study proposes a sequential transformation approach for functional outlier detection, offering a new taxonomy of outliers.
Outlier Detection : Methods, Models, and Classification	2020	Boukerche A, Zheng L, Alfandi O	The paper presents a detailed overview of outlier detection methods, models, and classification techniques.
Outlier Detection : How to Threshold Outlier Scores ?	2019	Yang J, Rahardja S	This study focuses on determining threshold values for outlier scores, proposing a new statistical method for effective outlier detection.
Unsupervised Anomaly Detection Using Variational Auto-Encoder Based Feature Extraction	2019	Rong Y, Chongdang L, Linxuan Z, Peng P	This research uses variational auto-encoders for unsupervised anomaly detection, emphasizing feature extraction techniques.
A Taxonomy and Platform for Anomaly Detection	2018	Sebestyen G, Hangan A, Czako Z, Kovacs G	This study introduces a taxonomy for anomaly detection methods and presents a platform for implementing these methods in various applications.
Distributed Local Outlier Detection in Big Data	2017	Yan Y, Cao L, Kuhlman C, Rundensteiner E	The paper develops an algorithm for detecting local outliers in skewed big data, improving detection accuracy and efficiency.
Big-Data-Driven Anomaly Detection in Industry (4.0) : An Approach and a Case Study	2016	Stojanovic L, Dinic M, Stojanovic N, Stojadinovic A	The study proposes a two-step approach for anomaly detection in Industry 4.0 : learning normal behavior from historical data and detecting anomalies in real-time.

2.2.1 Distance

Distance-based models represent a fundamental approach in the anomaly detection domain, where the primary criterion for identifying outliers is the spatial distance between data points. These models assume that normal data points occur in dense regions, while anomalies are located far from their closest neighbors. This section explores various algorithms within the distance-based model family, each offering unique strengths and applications.

2.2.1.1 k-Nearest Neighbors (k-NN)

The kNN algorithm examines the distribution and density of data points in the neighborhood up to the specified k value. According to the defined threshold value, it marks the regions where the data points are close to each other and the ones that are far away from each other as outliers. In the literature, it has been used to detect outliers with different methods based on the kNN algorithm.

A kNN-based anomaly detection method was proposed in the study conducted in the field of internet of things and instant data flow. If it is needed to consider the GAAOD (Grid-Based Approximate Average Outlier Detection) algorithm in three stages, firstly, after the distance distribution and neighborhoods are learned, the data is divided into relevant clusters and called "grid-based index" in order to keep this information together. is kept in the index. Thanks to its self-configuration feature, the GAAOD algorithm can adjust the grid structure appropriate to the distances between data points according to the cell size. Finally, an approach based on the "k-skybands" algorithm was used to identify outlier candidate data points and to remove data that could be considered normal from the set. This study can be used in various fields for prediction of instantaneous events or early notification (Zhu et al., 2020).

2.2.1.2 Local Outlier Factor (LOF)

The LOF value is created by proportioning the intensity amount according to the local neighborhoods of the data points. The density of the examined data point is expected

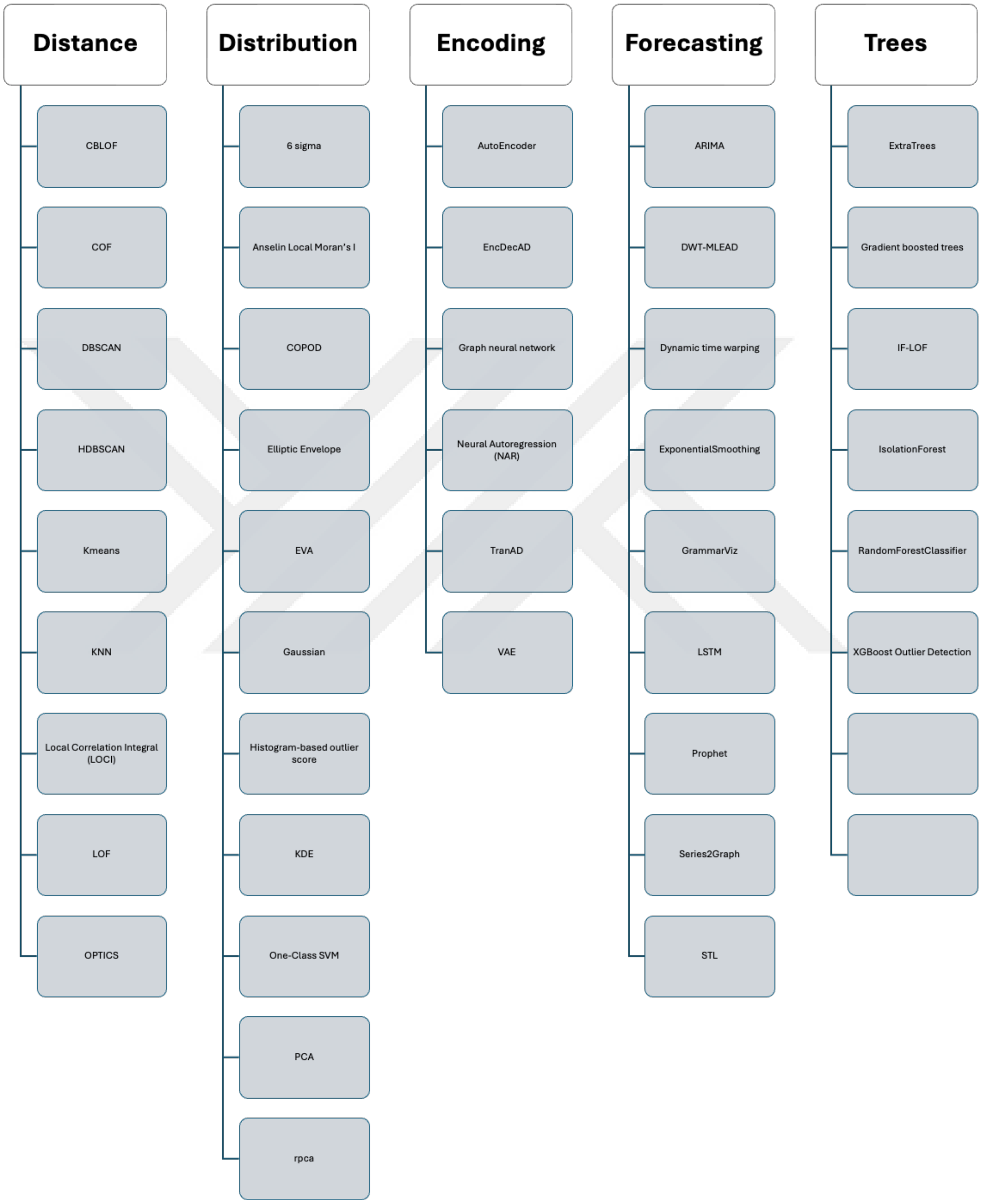

FIGURE 2.1 – Grouping of the algorithms

to be high when it is located in a crowded cluster, but the fact that the density of the data point is low indicates that it isolates itself within the cluster, that is, it does not belong there, thus the LOF value is calculated high. It may develop in the opposite situation. While the density of the data points around it is low, the density of the base data point is high, which indicates an anomaly and causes the LOF value to be high. When the LOF value is close to 1, it may cause misinterpretations. As a solution to this problem, experts have suggested the use of the LDBSCA (Local Density-Based Spatial Clustering of Applications with Noise) algorithm. Thus, the LOF value is calculated not only based on the density relationship between the neighborhoods of the data point, but also within the cluster to which it belongs. Tests performed on various data sets have shown that the performance of LDBSCA is high (Breunig et al., 2000).

2.2.1.3 DBSCAN

The DBSCAN algorithm is based on revealing the neighborhoods of data points with each other in two or multi-dimensional space. Since the database handles data from a spatial perspective, it is mostly used in the analysis of spatial data. For the DBSCAN algorithm, the terms core object, Eps, MinPts, direct density accessible point, density accessible point, density dependent point are the basic concepts. The algorithm takes Eps and MinPts values as input parameters. It checks all objects, starting from any object in the database. If the controlled object has already been included in a set, it moves to the other object without taking action. If the object has not been clustered before, it finds the object's neighbors in the Eps neighborhood by making a Region Query. If the number of neighbors is more than MinPts, it calls this object and its neighbors a new cluster. It then finds new neighbors by querying a new region for each previously unclustered neighbor. If the number of neighbors of the region query points is more than MinPts, they are included in the cluster (Ester et al., 1996).

2.2.1.4 HDBSCAN

Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN) extends DBSCAN by using a hierarchy of density thresholds to find clusters of varying densities. It identifies outliers as points in unstable or small clusters.

HDBSCAN is particularly useful for hierarchical clustering structures and varying densities. It finds applications in astronomy for galaxy clustering and in finance for detecting irregular trading patterns (Campello et al., 2013).

2.2.1.5 OPTICS

The OPTICS algorithm was developed by the group that discovered DBSCAN to overcome the disadvantage of the DBSCAN algorithm being dependent on two input parameters, Eps and MinPts (10). The OPTICS algorithm is not a clustering tool per se. The algorithm can be described more as a visualization tool. It allows finding clusters visually or indirectly through various measurements by displaying the database on a graph according to the distribution of variable Eps values (Ankerst et al., 1999)..

Since OPTICS only receives a fixed MinPts value from the user as an input parameter, it is not as dependent on parameter selection as DBSCAN. It finds the Eps values of each point for a fixed MinPts value and projects them on a graph, allowing the user to find clusters according to any desired Eps value. The OPTICS algorithm was developed to reduce DBSCAN's dependence on parameter selection. OPTICS algorithm only needs the MinPts parameter

2.2.1.6 Connectivity-Based Outlier Factor (COF)

The Connectivity-Based Outlier Factor (COF) measures the average chaining distance of a data point, capturing its connectivity with neighbors. COF identifies outliers as points with higher chaining distances, indicating poor connectivity within the dataset.

COF is effective in detecting outliers in datasets where anomalies are not well-separated from normal data. It is used in network analysis to identify nodes with unusual connectivity and in social network analysis to detect individuals with abnormal interaction patterns (Tang et al., 2002).

2.2.1.7 K-Means

The K-means method is among the simplest unsupervised learning methods that solve the clustering problem. The general logic of the algorithm is to partition a data set (X) consisting of n data objects into k (k n) sets given as input parameters. The aim is to ensure that the intra-cluster similarities of the clusters obtained as a result of the partitioning process are maximum and the inter-cluster similarities are minimum. The performance of the method is affected by the number of k clusters, the values of the cluster centers selected as the starting point, and the similarity measurement criteria (MacQueen, 1967).

2.2.1.8 Local Correlation Integral (LOCI)

Experts who developed the LOCI algorithm have considered a method that can run faster than other algorithms, detect micro clusters, and perform at multiple small cluster sizes as well as local density. They also stated that it differs from other algorithms in that the graphs produced specifically for each data point visualize the behavior of the data set and the threshold value used in detecting the outlier is determined by the algorithm instead of the user (Papadimitriou et al., 2003).

2.2.1.9 CBLOF

Experts who developed the CBLOF algorithm aimed to examine the behavior of local datasets rather than focusing on the entire dataset. They used the clustering algorithm Squeezer, developed in 2002, to divide the data into groups. After the cluster centers are determined with Squeezer, the outlier factor stage is started, the locations of the data points and their relationships with each other are examined, and as a result of the calculations, the CBLOF algorithm determines the outliers. At the same time, in the published study, the algorithm called FindCBLOF was introduced and developed for the analysis and reporting of outliers found by CBLOF. In another study conducted to detect suspicious transactions, it was stated that one of the successful aspects of the CBLOF algorithm is that it is not affected by lack of data. It has been stated that its performance can also be successful in new fraud methods and that it performs well even in limited data sets since it does not need training data. They preferred the LOF algorithm in clustering the data, unlike the previous cited study. They wanted to focus on local data points in order to obtain more effective results rather than detecting general outliers, which led them to use the LOF algorithm (He et al., 2003).

2.2.2 Distribution

Distribution-based models for anomaly detection focus on the statistical properties of the data, assuming that normal data points follow a specific distribution, while outliers deviate significantly from this model. These models are adept at identifying global outliers that do not conform to the expected distribution. This section discusses several key algorithms within the distribution-based model family, highlighting their methodologies and applications.

2.2.2.1 6 Sigma

The Six Sigma methodology uses statistical analysis to measure the variation within a process and improve quality by identifying and removing causes of defects and minimizing variability. It relies on the concept of standard deviation (sigma) to measure data dispersion.

In outlier detection, Six Sigma identifies points that lie beyond the accepted range (typically more than three standard deviations from the mean). These points are considered outliers, indicating significant deviation from the normal distribution. This approach is particularly useful in manufacturing and quality control, where it helps in identifying defects and maintaining process quality (Montgomery, 2007).

2.2.2.2 Anselin Local Moran's I

Anselin Local Moran's I is a spatial statistic used to identify clusters of similar values and outliers in spatial data. It assesses the degree of spatial autocorrelation and helps in identifying regions with significantly different values compared to their neighbors.

For outlier detection, this method identifies spatial outliers by analyzing the local autocorrelation of data points. Points with significantly high or low values compared to their neighbors are considered outliers. This method is useful in geographic data analysis, urban planning, and environmental monitoring (Anselin, 1995).

2.2.2.3 COPOD (Copula-Based Outlier Detection)

COPOD utilizes copula functions to model the joint distribution of multivariate data. It assesses the dependence structure between variables and identifies anomalies based on deviations from the expected copula distribution.

COPOD is effective for outlier detection in datasets with complex dependencies among variables. It is used in financial risk management to detect abnormal correlations between assets and in medical data analysis to identify unusual patient profiles (Li et al., 2020).

2.2.2.4 Elliptic Envelope

The Elliptic Envelope algorithm fits a robust Gaussian distribution to the data and identifies points that lie outside the contour of this distribution as outliers. It assumes that the data follows a Gaussian distribution and uses Mahalanobis distance to measure the deviation of points from the mean.

Elliptic Envelope is effective in detecting multivariate outliers in datasets with a Gaussian distribution. It is used in financial data analysis to identify fraudulent transactions and in quality control to detect defective products (Hubert and Debruyne, 2009).

2.2.2.5 EVA (Extreme Value Analysis)

Extreme Value Analysis focuses on the tail behavior of distributions to identify outliers. It models the extreme values (both minimum and maximum) and assesses their deviation from the overall data distribution.

EVA is particularly useful for identifying rare events or extreme deviations in datasets. It is applied in finance for risk management, environmental sciences for extreme weather event prediction, and engineering for failure analysis (Coles, 2001).

2.2.2.6 Gaussian Distribution

Gaussian distribution-based methods assume that data follows a normal distribution. Outliers are identified as points that lie in the tails of the distribution, far from the mean.

For outlier detection, techniques such as z-score and modified z-score are used to identify points that deviate significantly from the mean. These methods are widely used in various applications, including fraud detection, environmental monitoring, and quality control (Aggarwal, 2013).

2.2.2.7 Kernel Density Estimation (KDE)

Kernel density estimator is an alternative to histogram. There are many alternatives, but one of the most popular is the Kernel density estimator. KDE is a technique that allows you to create a smooth curve given a data set.

Kernel density estimation method is a widely used method in non-parametric density estimation. For the unknown probability density function, the probability density or distribution function of the random variable related to the kernel estimator is estimated. In this data-driven prediction method, the performance of the kernel estimator depends on the bandwidth parameter, also known as the smoothing parameter. In other words, bandwidth plays an important role in estimating the probability density function or distribution function. For example, while small bandwidths create roughness in the tails of the distribution, large bandwidths cause excessive flattening in the tails of the distribution and do not allow us to see the features of the distribution. There are various bandwidth selection methods used in kernel smoothing.(Parzen, 1962).

2.2.2.8 One-Class SVM

The main idea on which SVM is based is to find a hyperplane in the feature space that can optimally separate two classes. Although many other linear learning machines also work with this logic, SVM differs from them in that it can offer a solution that will minimize the possibility of misclassification for the objects to be tested (structural risk minimization). In the case of single-class SVM, the data is first moved into the feature space using an appropriate kernel function and then separated maximally from the origin by means of a hyperplane. These hyperplane parameters are obtained by solving a quadratic problem similar to that in normal SVM (Scholkopf et al., 2001).

2.2.2.9 Principal Component Analysis (PCA)

Principal Component Analysis (PCA) known as a classical data reduction technique has been widely used in network anomaly detection over the last 20 years. To use the dimension reduction method for detecting outliers i.e., anomalies, the PCA is regarded as a classification procedure and utilized to detect outlying and normal observations.

For outlier detection, PCA identifies points that deviate significantly from the principal components, indicating anomalies. PCA is widely used in image compression, bioinformatics, and finance to detect outliers in high-dimensional datasets (Jolliffe, 2002).

2.2.2.10 Robust Principal Component Analysis (rpca)

RPCA a semi-supervised machine learning model, which utilizes robust PCA and Mahalanobis Distance to automatically identify anomalies in network traffic and detect potential attacks RPCA is effective for outlier detection in datasets where the data can be decomposed into low-rank structures with sparse outliers. It is used in background subtraction in video processing, image denoising, and anomaly detection in high-dimensional data (Candes et al., 2011).

Regression-based methods for outlier detection involve modeling the relationship between dependent and independent variables to identify deviations from the expected behavior. These methods are particularly useful when there is a clear relationship between variables that can be leveraged to detect anomalies.

2.2.2.11 Geographically Weighted Regression (GWR)

Geographically weighted regression (GWR) is one of several spatial regression techniques also used in geography and other disciplines. GWR provides a local model of the variable or process you are trying to predict by building a regression equation on each feature in the data set. When used correctly, these methods provide powerful and reliable statistics for estimating linear relationships.

In the context of outlier detection, GWR can identify spatial outliers by examining areas where the local regression parameters deviate significantly from the global model (Brunsdon et al., 1996).

2.2.3 Encoding

Encoding-based models for anomaly detection leverage representation learning to identify outliers by encoding data points into a lower-dimensional space. These models are particularly effective in scenarios where the intrinsic structure of the data is complex and high-dimensional. By transforming the data into a more compact form, encoding-based models can highlight anomalies that deviate from the norm in the encoded space. Below, several pivotal algorithms within this category are explored, discussing their methodologies, applications, and inherent advantages and challenges.

2.2.3.1 AutoEncoder

AutoEncoder is an unsupervised method that first compresses this multidimensional data with the Encoder and then reconstructs this data with the structure called decoder. In the AutoEncoders method, the number of nodes in the Input Layer of this neural network and the nodes in the Output Layer are equal to each other (Dimensions are equal to each other). After compressing this data with the Encoder, decoder data recoversto the same size again.

Autoencoders can be used in three different areas. The first of these is to extract (discover) the basic (important) features in this data. The second area of use is to clean up the noise in this data that distorts it, which is called image denoising. Another usage area would be Anomaly detection.(Hinton and Salakhutdinov, 2006).

2.2.3.2 EncDecAD

Encoder-Decoder Anomaly Detection (EncDecAD) utilizes a sequence-to-sequence architecture typically used for tasks like language translation. The encoder processes the input sequence into a fixed-length context vector, and the decoder reconstructs the sequence from this context. The model is trained to minimize the reconstruction loss.

For outlier detection, EncDecAD identifies anomalies based on reconstruction errors. Sequences that result in higher reconstruction errors are flagged as anomalies. This method is particularly effective for time series data, where it can capture temporal dependencies and detect unusual patterns that deviate from normal sequences (Malhotra et al., 2016).

2.2.3.3 Graph Neural Network (GNN)

GNN model design basically consists of four process steps. The first step is to find the structure of the graph, the second step is to determine the type and scale of the graph, the third step is to calculate the design loss function, and the last step is to complete the design using calculation modules. In the first step, finding the structure of the graph, two different conditions are taken into consideration. The first case is structural scenarios where the information on the charts is clear. The second situation is called unstructured scenarios where the information is not clear. After this process is completed, the chart type is determined. There are different types of graphs depending on their type, such as directed/undirected, homogeneous/heterogeneous, static/dynamic.

In outlier detection, GNNs identify nodes with anomalous patterns or connections that deviate from the expected graph structure. This approach is useful in applications like social network analysis, where detecting nodes with unusual interaction patterns can reveal fraud or anomalous behavior (Wu et al., 2020).

2.2.3.4 Neural Autoregression (NAR)

NAR neural networks are a closed-loop network that solves the problem through a single variable with a time delay

For outlier detection, NAR models identify anomalies by comparing predicted values with actual observations. Significant deviations between predicted and observed values indicate potential anomalies. This method is effective for time series data, such as financial time series and sensor readings, where accurately modeling temporal dependencies is crucial (Hochreiter and Schmidhuber, 1997).

2.2.3.5 TranAD

TranAD, a deep transformer network based anomaly detection and diagnosis model which uses attention-based sequence encoders to swiftly perform inference with the knowledge of the broader temporal trends in the data. TranAD uses focus score-based self-conditioning to enable robust multi-modal feature extraction and adversarial training to gain stability.

In outlier detection, TranAD identifies anomalies by comparing the reconstructed sequence to the original input. Points with higher reconstruction errors are flagged as outliers. This method is effective for time series data and can capture both short-term and long-term dependencies (Vaswani et al., 2017).

2.2.3.6 Variational AutoEncoder (VAE)

A Variational AutoEncoder is a type of generative model that is capable of encoding input data into a latent space and subsequently decoding it back to its original space. VAEs differ from typical AutoEncoders by introducing a probabilistic framework in the latent space, which allows them to produce novel data points.

VAEs are used for outlier discovery by assessing anomalies through the reconstruction error and the probability of the latent representations. Outliers are data points that have low probabilities or large reconstruction errors. This method is beneficial for intricate, multi-dimensional data, such as images and text, where it is crucial to capture the fundamental data distributions (Kingma and Welling, 2013).

2.2.4 Forecasting

Forecasting-based models in anomaly detection predict future values based on historical data, identifying anomalies as data points that significantly deviate from predicted values. These models are particularly effective in time-series data where patterns over time are critical for understanding normal versus abnormal behavior. This section outlines several key forecasting-based models, highlighting their methodologies, practical applications, and the benefits and limitations of each approach.

2.2.4.1 ARIMA

One of the first solutions that come to mind regarding modeling time series is related to the high impact of past data on predicting future data. The ARIMA model, also known as Box-Jenkins models, is considered a standard method used in stationary time series (Box et al., 2015). In this method, firstly the recognition of time series and its suitability for the method are investigated. Then the parameters of the ARIMA model are determined. In the last stage, time series modeling results are evaluated with the determined parameters.

In outlier detection, ARIMA forecasts future values based on past observations. Anomalies are identified by comparing the actual values to the predicted values, with significant deviations indicating potential outliers. ARIMA is effective in detecting anomalies in financial data, sensor readings, and other time series datasets where past behavior is a good predictor of future values

2.2.4.2 Dynamic Time Warping (DTW)

Dynamic Time Warping (DTW) is a technique used to measure the similarity of time series. While DTW is used to calculate the similarity of two time series, the time scales and velocities of these series can be different as seen in the image below. That is, two time series do not have to match exactly along the time axis to be similar. In summary, the algorithm performs shifting and scaling operations on the time axis to match any point of one time series with another time series.

In outlier detection, DTW identifies anomalies by comparing the target time series to a reference series and detecting significant deviations in the alignment. This method is useful for detecting anomalies in time series with varying speeds and temporal distortions, such as speech and gesture recognition (Muller, 2007).

2.2.4.3 Exponential Smoothing

Exponential Smoothing is a technique used for predicting time series data. It calculates weighted averages of previous observations, with the weights falling exponentially as the data gets older. Its purpose is to mitigate abrupt variations and emphasize enduring patterns.

Exponential Smoothing is a technique used in outlier identification to make predictions about future values. It also uncovers anomalies by comparing the actual values with these predictions and detecting any significant discrepancies. It is efficient for data that exhibits seasonal patterns and trends, such as sales data and weather forecasts (Hyndman et al., 2008).

2.2.4.4 GrammarViz

GrammarViz is a technique for representing time series data using symbols. It use grammar-based compression to detect patterns and anomalies. The process involves transforming time series data into symbolic sequences and identifying anomalies by examining departures from normal patterns.

GrammarViz is very efficient at identifying irregularities in intricate, multi-dimensional time series data, such as financial transactions and biological sequences. It excels in capturing complex patterns and regularities using symbolic representations (Senin et al., 2014).

2.2.4.5 LSTM

LSTM has a complex structure . LSTM neural network makes predictions by learning parameters. LSTM gives better results than traditional machine learning models. Memory cell of the LSTM 100 model ; It consists of a total of three non-linear door units : entrance door, forgetting door and exit door.

In outlier detection, LSTM predicts future values based on past sequences and identifies anomalies by comparing the actual values to the predicted values. Significant deviations indicate potential anomalies. LSTM is highly effective for detecting anomalies in time series data with long-term dependencies, such as financial time series and sensor data (Hochreiter and Schmidhuber, 1997).

2.2.4.6 Prophet

Prophet is a forecasting tool created by Facebook specifically built to handle time series data that has pronounced seasonal patterns and missing data. The method employs an additive model to break down the time series into its constituent components, including trend, seasonality, and holidays.

Prophet, in the context of outlier detection, detects abnormalities by comparing seen data with the anticipated values generated by the model. Anomalies are identified when there are significant variations from the forecasted points. Prophet is highly valuable for predicting future business trends and monitoring website traffic, especially when it is important to capture seasonal patterns and long-term trends.Refer to the source mentioned in Taylor (2018).

2.2.4.7 Series2Graph

Series2Graph is a method that transforms time series data into graphs, capturing the structural patterns within the data. It uses graph-based algorithms to identify similarities and anomalies by analyzing the connectivity and structure of the graphs.

Series2Graph is effective for detecting anomalies in time series data with complex patterns, such as electrocardiogram (ECG) signals and financial market data. By leveraging graph-based representations, it can capture intricate relationships and detect deviations from typical patterns (Boniol and Palpanas, 2020).

2.2.4.8 STL (Seasonal-Trend Decomposition using LOESS)

STL is a decomposition technique that separates a time series into seasonal, trend, and residual components using LOESS (locally estimated scatterplot smoothing). This method is flexible and robust, handling a wide range of seasonal and trend patterns.

For outlier detection, STL analyzes the residual component to identify points that deviate significantly from the expected seasonal and trend patterns. This approach is useful for detecting anomalies in time series data with complex seasonality and trends, such as retail sales and environmental data (Cleveland et al., 1990).

2.2.5 Trees

Tree-based models utilize the structure of decision trees to detect anomalies by acquiring decision rules derived from the characteristics of the data. These models provide a high level of interpretability and have the ability to process both numerical and categorical data. As a result, they are adaptable tools that may be used for detecting anomalies in a wide range of domains. This section provides an overview of many significant tree-based algorithms, explaining their approaches, uses, and the inherent benefits and constraints of each.

2.2.5.1 ExtraTrees

Extra Trees, also known as Extremely Randomized Trees, is an ensemble learning technique that builds several decision trees by randomly selecting splits for the features. It combines the outcomes of various trees to enhance forecast accuracy and resilience.

ExtraTrees is a useful method for detecting outliers. It accomplishes this by comparing the predictions made by each individual tree and assessing the level of consistency over the whole ensemble. Instances that exhibit significant diversity in predictions are identified as outliers. The approach described is proficient in identifying anomalies in datasets with a large number of dimensions, since it allows separate trees to record distinct characteristics of the data structure (Geurts et al., 2006).

2.2.5.2 Gradient Boosted Trees

Gradient Boosted Trees (GBT) is an ensemble method that constructs numerous trees in a sequential manner, where each tree aims to rectify the mistakes made by the preceding trees. The model leverages the various trees' strengths to enhance accuracy.

GBT, in the context of outlier identification, detects abnormalities by examining the residual errors of the model. Outliers are data points with substantial residuals, indicating notable departures from the projected values. Gradient Boosting Trees (GBT) are particularly efficient when dealing with intricate datasets that need the recording of interactions between different characteristics, as highlighted by Friedman et al. in their 2001 study.

2.2.5.3 IF-LOF

The IF-LOF formula takes the best parts of both Isolation Forest (IF) and Local Outlier Factor (LOF) and uses them together. Isolation Forest separates observations by picking features at random and splitting values, and LOF checks the number of data points in a certain area.

In outlier identification, IF-LOF uses Isolation Forest to find possible outliers and then uses LOF to improve the finding by looking at local density. This mixed method makes finding outliers more reliable and accurate in many datasets (Cheng et al., 2019).

2.2.5.4 Isolation Forest

Isolation Forest is an ensemble method that isolates observations by randomly selecting features and splitting values between the maximum and minimum values. It constructs trees where the depth of the tree indicates the anomaly score.

Isolation Forest is particularly effective for high-dimensional data and can detect anomalies efficiently by isolating outliers with fewer splits compared to normal points. It is widely used in applications such as fraud detection and network security (Liu et al., 2008).

2.2.5.5 RandomForestClassifier

This is an ensemble learning method called RandomForestClassifier. It creates many decision trees and then uses majority voting to join their results. A random sample of the data and features is used to train each tree. This makes the model more accurate and stable. RandomForestClassifier finds outliers by looking at how consistent the results are across all the trees. Outliers are cases where expectations vary a lot from one another. In both classification and regression tasks, this method works well to find outliers (Breiman, 2001).

2.2.5.6 XGBoost

XGBoost is a gradient boosting method that has been improved. It builds trees one after the other, with each tree fixing the mistakes made by the ones that came before it. It makes model adaptation better by using advanced regularization methods.

XGBoost finds outliers by looking at the residuals and feature importances in outlier identification. Points that are very different from what would normally happen, with big residuals, are called outliers. XGBoost works really well with big datasets and tough problems (Chen and Guestrin, 2016).

2.3 AutoML

Automated Machine Learning (AutoML) has emerged as a significant innovation in the field of artificial intelligence, aimed at automating the application of machine learning to solve complex real-world problems. At its essence, AutoML represents the intersection of machine learning and automation, striving to make the process of building, deploying, and maintaining machine learning models less complex and more accessible to a broader audience. Its development has been driven by both the demand for wider application of machine learning and the need to address the shortage of skilled practitioners in this area. The articles accessed were first subjected to selection criteria, which are as follows :

— Studies conducted after 2010 were examined.
— Various sources were examined, including academic journals, conference proceedings, articles, dissertations, and books.
— Only studies in English were considered.

The titles and abstracts of these studies were scrutinized to identify those relevant to the topic. This led to the identification of 55 articles for a full-text review focused on model taxonomy in anomaly detection. After a detailed reading of these articles, 34 were deemed irrelevant to the research and excluded. The research sample ultimately consisted of 21 articles.

These articles were carefully analyzed to deepen the understanding of model taxonomy within the anomaly detection field. The findings from this literature review have been systematically organized to contribute to the existing body of knowledge and provide a foundation for subsequent research in this area.

The table 2.4 presents an extensive overview of significant contributions to the field of Automated Machine Learning (AutoML), showcasing diverse approaches and methodologies aimed at automating various aspects of the machine learning workflow. This analysis highlights the advancements, challenges, and applications of AutoML as discussed in the listed articles. Conducting a meta-survey of the landscape of outlier and anomaly detection, the study by Sourav Chatterjee et al. (2022) introduces MOSPAT, an AutoML framework for model selection and parameter tuning specifically for time series anomaly detection. Although comprehensive, this framework lacks a focus on preprocessing, which is a critical component in the data preparation stage for machine learning tasks (Chatterjee et al., 2022).

In 2023, Aleksei Liuliakov, Luca Hermes, and Barbara Hammer evaluated AutoML technologies for sparse classification and outlier detection. Their work critically examines the performance of these technologies in clustering and outlier detection scenarios, providing insights into the strengths and limitations of current AutoML systems in handling sparse and noisy data (Liuliakov et al., 2023).

Yuanxiang Ying and colleagues (2020) delve into automated model selection for time-series anomaly detection. Their approach involves identifying the characteristic steps necessary for time series analysis before selecting the appropriate algorithm. The study's comparative analysis with other AutoML systems underscores the effectiveness of their methodology in improving detection performance (Ying et al., 2020).

Anh Truong et al. (2019) provide a comprehensive evaluation and comparison of various AutoML tools. Their research focuses on automating tasks such as data preprocessing, feature engineering, model selection, hyperparameter optimization, and result analysis. By evaluating these tools across different datasets, the study offers valuable benchmarks and identifies key areas for improvement in current AutoML approaches (Truong et al., 2019).

Abderahim Salhi and co-authors (2023) focus on data preprocessing techniques within AutoML frameworks. Their survey highlights how these methods can automate the preprocessing pipeline, enhancing model performance and reducing the need for manual intervention. This study is crucial as it addresses a foundational aspect of the machine learning workflow that significantly impacts downstream tasks (Salhi et al., 2023).

The work by Zhaoyang Wang and Shuo Wang (2023) addresses the challenge of class imbalance in data streams through an online AutoML approach. Their research demonstrates how dynamically adjusting to changes in data distribution can maintain high model performance, which is particularly relevant for real-time applications where data characteristics can evolve rapidly (Wang and Wang, 2023).

Luís Ferreira and his team (2021) compare AutoML tools for their capabilities in machine learning, deep learning, and XGBoost. Their comparative analysis identifies the strengths and weaknesses of each tool, providing a valuable resource for practitioners seeking to select the most appropriate AutoML system for their specific needs (Ferreira et al., 2021).

The empirical study by Forough Majidi and colleagues (2022) explores the practical usage of AutoML tools. Their findings on the effectiveness, usability, and simplification of the machine learning workflow offer insights into the real-world applicability and user experiences with these tools, highlighting areas where AutoML can be further improved to meet user needs (Majidi et al., 2022).

Matthias Feurer et al. (2019) present Auto-sklearn, a robust framework that leverages Bayesian optimization and meta-learning. This framework is noted for its efficiency and robustness, demonstrating significant advancements in the automation of the model selection and hyperparameter tuning processes (Feurer et al., 2019).

Haifeng Jin, Qingquan Song, and Xia Hu (2019) introduce Auto-Keras, an AutoML system designed for neural architecture search. This open-source tool automates the design of deep learning models, showcasing the potential of AutoML in simplifying the development of complex neural networks (Jin et al., 2019).

The research by Erin LeDell and colleagues (2020) discusses H2O AutoML, an extensible system that automates the training and tuning of machine learning models. Their work emphasizes scalability and flexibility, making H2O AutoML suitable for a wide range of applications and large-scale datasets (LeDell et al., 2020).

Randal S. Olson and his team (2016) introduce TPOT, an AutoML tool that employs genetic programming to optimize machine learning pipelines. TPOT's approach to pipeline optimization represents a significant innovation in the field, allowing for the automatic generation of high-performing models without manual intervention (Olson et al., 2016).

TABLE 2.4 – Overview of Research Articles on AutoML

Name of the Article	**Year**	**Authors**	**Explanation**
MOSPAT : AutoML based Model Selection and Parameter Tuning for Time Series Anomaly Detection	2022	Sourav Chatterjee, Rohan Bopardikar, Marius Guerard, Uttam Thakore, Xiaodong Jiang	AutoML framework'ü oluşturuyor ama preprocessing'ten hiç bahsetmiyor.
AutoML technologies for the identification of sparse classification and outlier detection models	2023	Aleksei Liuliakov, Luca Hermes, Barbara Hammer	AutoML teknolojilerini Clustering ve Outlier Detection performanslarına göre değerlendiriyor.
Automated Model Selection for Time-Series Anomaly Detection	2020	Yuanxiang Ying, Juanyong Duan, Chunlei Wang, Yujing Wang, Congrui Huang, Bixiong Xu	Time Series'in özelliğine göre yapılacak adımı belirleyip sonrasında algoritma seçimi yapıyor. Diğer Auto ML'ler ile karşılaştırıyor kendi yaptıklarının performansını.
Towards Automated Machine Learning : Evaluation and Comparison of AutoML Approaches and Tools	2019	Anh Truong, Austin Walters, Jeremy Goodsitt, Keegan Hines, C. Bayan Bruss, Reza Farivar	This paper investigates the current state of AutoML tools aiming to automate tasks such as data pre-processing, feature engineering, model selection, hyperparameter optimization, and prediction result analysis. It evaluates various tools on different datasets to assess performance.
Data Preprocessing Using AutoML : A Survey	2023	Abderahim Salhi, Althea C. Henslee, James Ross, Joseph Jabour, Ian Dettwiller	The study surveys data preprocessing techniques within AutoML frameworks, focusing on how these methods can automate the preprocessing pipeline to enhance model performance and reduce the need for manual intervention.
Online Automated Machine Learning for Class Imbalanced Data Streams	2023	Zhaoyang Wang, Shuo Wang	This research addresses the challenge of class imbalance in data streams using an online AutoML approach, which dynamically adjusts to changes in the data distribution to maintain high model performance.
A Comparison of AutoML Tools for Machine Learning, Deep Learning and XGBoost	2021	Luís Ferreira, André Pilastri, Carlos Manuel Martins, Pedro Miguel Pires, Paulo Cortez	This paper compares various AutoML tools specifically for their capabilities in machine learning, deep learning, and XGBoost. The comparison aims to identify the strengths and weaknesses of each tool in handling different types of ML tasks.
An Empirical Study on the Usage of Automated Machine Learning Tools	2022	Forough Majidi, Moses Openja, Foutse Khomh, Heng Li	This empirical study explores how AutoML tools are used in practice, evaluating their effectiveness, usability, and the extent to which they simplify the machine learning workflow for users.
Auto-sklearn : Efficient and Robust Automated Machine Learning	2019	Matthias Feurer, Aaron Klein, Katharina Eggensperger, Jost Tobias Springenberg, Manuel Blum, Frank Hutter	This paper presents Auto-sklearn, an efficient and robust framework for automated machine learning that leverages Bayesian optimization and meta-learning.
Auto-Keras : An Efficient Neural Architecture Search System	2019	Haifeng Jin, Qingquan Song, Xia Hu	This paper introduces Auto-Keras, an open-source AutoML system designed to automate neural architecture search for deep learning models.
H2O AutoML : Scalable Automatic Machine Learning	2020	Erin LeDell, Navdeep Gill, Spencer Aiello	This research discusses H2O AutoML, a scalable and extensible open-source system that automates the process of training and tuning machine learning models.
TPOT : A Tree-Based Pipeline Optimization Tool for Automating Machine Learning	2016	Randal S. Olson, Nathan Bartley, Ryan J. Urbanowicz, Jason H. Moore	The paper introduces TPOT, an AutoML tool that uses genetic programming to optimize machine learning pipelines automatically.

2.3.1 Definition and Purpose

The notion of Automated Machine Learning (AutoML) is based on the principle of making machine learning accessible to everyone. AutoML allows persons without specialized understanding in statistical analysis and algorithm design to construct and apply powerful machine learning models by minimizing the need for extensive experience in these areas. This endeavor to democratize machine learning not only broadens the scope of its influence but also creates new opportunities for innovation and implementation in many industries (Ying et al., 2020).

AutoML's goal also involves favoring efficiency. AutoML appeals to experienced data scientists and machine learning engineers because it automates arduous aspects of model generation, such as parameter fine-tuning and feature creation. Automation saves time, allowing experts to focus on other important parts of machine learning projects, such as data analysis and strategic decision-making (Ying, 2020).

AutoML began as an academic study, evolving alongside the evolution of machine learning libraries and platforms. Initial efforts in the domain focused on the automation of algorithm and hyperparameter selection, as demonstrated by systems such as Auto-Weka and later Auto-sklearn. Auto-Weka, a key participant in the field of Automated Machine Learning (AutoML), used Weka's algorithms and promoted the notion of automated algorithm selection among a larger range of individuals (Thornton et al., 2012). Auto-sklearn developed an automated technique that integrated with the popular Scikit-learn machine learning software (Feurer et al., 2019).

The growth of AutoML has also been driven by the emergence of commercial platforms that integrate machine learning automation into their extensive suite of data analysis functionalities. These platforms have enhanced the functionalities of AutoML by incorporating data preparation, feature extraction, and model interpretation, therefore providing end-users with a comprehensive range of tools. AutoML solutions, such as H2O Driverless AI and Google Cloud AutoML, have enhanced organizations' ability to develop and deploy machine learning models on a significant scale (Truong et al., 2019). These technologies provide extensive solutions that automate the entire process, encompassing model selection, hyperparameter tuning, data preparation, and feature engineering. These steps are crucial.

2.3.2 Steps in AutoML

Data scientists often need to evaluate several algorithms and fine-tune different hyperparameters to select the best-performing model based on their assumptions and experiences. This process can be costly and tedious. Manually optimizing hyperparameters not only incurs higher costs but also increases the likelihood of setting stereotypical hyperparameters, leading to irrational configurations and poor model performance (Feurer et al., 2019).

AutoML emerges as a solution to save time and effort in repetitive tasks within machine learning workflows, such as data preprocessing, feature engineering, model selection, hyperparameter optimization, and forecast result analysis (Ying et al., 2020). Figure 1 visualizes the optimization of feature engineering, model selection, and hyperparameter optimization steps in a machine learning process under AutoML, adapted from (Truong et al., 2019).

2.3.2.1 Data Preprocessing

Preprocessing is one of the most critical tasks affecting the accuracy of prediction models, as most datasets contain noisy data with missing values, different feature scales, and outliers (Ying et al., 2020). Data preprocessing includes steps like data cleaning, data integration, data transformation, and data reduction. Data cleaning involves resolving inconsistencies and noise in the data. Data integration is about creating a single dataset from various sources. Data transformations include operations like correction, merging, and normalization. Data reduction reduces the size of the data through methods like merging variables, eliminating unnecessary data, or clustering.

Currently, no AutoML technique can fully handle this task accurately and requires significant human intervention (Ying et al., 2020). This task requires detecting data types and schema, which is commonly not supported by AutoML tools. However, once data types are identified, the tools can provide appropriate feature engineering for the next component in the workflow (Feurer et al., 2019).

2.3.2.2 Feature Engineering

Feature engineering is a process typically performed manually by experts, requiring extensive domain knowledge and being one of the most time-consuming steps in machine learning (Thornton et al., 2012). It involves transforming raw data into features that better represent the underlying problem for predictive models, leading to improved model accuracy on unseen data. These features can then be fed directly into machine learning models for training and making predictions (Thornton et al., 2012). Feature engineering consists of three subtopics : feature selection, feature extraction, and feature generation. Feature extraction and generation are types of feature transformation where a new set of features is created. The purpose of feature selection is to reduce feature redundancy by selecting important features. Automated feature engineering is a dynamic combination of these three processes to some extent (Feurer et al., 2019).

Automated feature engineering aims to create a new set of features that can improve the performance of machine learning tools, saving intensive human knowledge and effort (Thornton et al., 2012). The main advantages of the automated approach are minimal domain knowledge required, the evaluation of a wide range of possible features instead of only those defined by a specific expert, and shorter and cleaner coding (Ying et al., 2020).

2.3.2.3 Model and Hyperparameter Optimization

Creating a high-quality machine learning model involves an iterative, complex, and time-consuming process of experimenting with different algorithms and techniques and having good experience in effectively tuning hyperparameters (Feurer et al., 2019). The goal of model selection is to find models with the highest accuracy when trained and tested on a dataset. AutoML aims to select the optimal model based on the data without human intervention. It can iterate through different models trained on the same input data and select the one that performs best (Ying et al., 2020).

Hyperparameters are values set by the developer before starting model training and usually remain constant throughout the training process. These values significantly affect the model's performance. Hyperparameter optimization is a crucial component of

AutoML in searching for optimal hyperparameters for neural network structures and model training processes. The goals of hyperparameter optimization are threefold : reducing costly manual tasks for AI experts, lowering the threshold for research and development, improving the accuracy and efficiency of neural network training, and making hyperparameter selection more persuasive and training results more reproducible (Feurer et al., 2019). Hyperparameter optimization mainly includes grid search, random search, Bayesian optimization, and gradient-based optimization (Thornton et al., 2012).

By automating these steps, AutoML facilitates a more efficient and accessible approach to machine learning, enabling researchers and practitioners to develop robust models without requiring extensive expertise in each phase of the workflow.

2.3.3 State of Art AutoMLs

AutoML research, especially focusing on Hyperparameter Optimization (HPO), began in the 1990s. The development of these tools, along with their mathematical foundations, became feasible after 2010. These tools are categorized into three main groups : open-source, developed by startups, and provided by tech giants (Feurer et al., 2019).

2.3.3.1 MLBox

MLBox is a Python library designed with the AutoML concept in mind. The features provided to users by this library include :

Rapid reading and integration of data Execution of the data preprocessing phase (removal of noisy data, completion of missing data, reformatting, and scaling) Feature elimination and successful HPO Use of classic machine learning algorithms and regression-based algorithms for classification tasks Interpretation and completion of predictions through the constructed model The MLBox pipeline includes three main sub-packages. These packages facilitate the automatic progress of assigned tasks. The Preprocessing sub-package handles data reading and preprocessing, the Optimization sub-package deals with model testing and cross-validation steps, and the Prediction sub-package manages the prediction process. MLBox is compatible only with Linux, and does not support Windows or MacOS (VASILE et al., 2018).

2.3.3.2 Auto-Sklearn

Auto-Sklearn is an AutoML package built on Scikit-learn. This package fully automates algorithm selection and HP adjustment processes, relieving the user of these tasks. Auto-Sklearn applies important feature extraction methods such as one-hot encoding for categorical variables, numeric feature standardization, and principal component analysis. The pipeline in Auto-Sklearn is optimized through Bayesian search. The Bayesian HPO used in Auto-Sklearn balances the machine learning configuration and employs meta-learning for the Bayesian optimizer and automated ensemble construction for the continuous control and evaluation of optimization in the process. Auto-Sklearn performs well on small to medium-sized datasets but is not yet applicable for deep learning system configurations on very large datasets. The package operates only on Linux-based machines (Feurer et al., 2019).

2.3.3.3 Tree-based Pipeline Optimization Tool (TPOT)

TPOT is a Python-based AutoML tool that uses genetic programming to optimize the machine learning pipeline. Although it is an extension of the Scikit-learn framework, it has its own regressor and classifier methods. TPOT helps find the most suitable pipeline for the data among many possible options in AutoML pipeline exploration. Despite its benefits, TPOT is not suitable for natural language processing tasks or analyzing categorical structures in the data. Using TPOT in such cases is only possible if the existing categorical variables are converted to numeric (Olson et al., 2016).

2.3.3.4 H2O

H2O.ai offers H2O as a memory-based distributed machine learning platform. It is completely open-source and supports both R and Python. The system provides statistical analyses, classic machine learning algorithms, and deep learning algorithms. The AutoML module within H2O constructs the pipeline using its unique algorithm and optimizes it through feature engineering and HPs. Users can automate the entire machine learning workflow, including feature selection, model building, validation, selection, and usage. All these steps progress automatically from start to finish (LeDell et al., 2020).

2.3.3.5 Auto-Keras

Auto-Keras, developed by DATA Lab, is an open-source AutoML library based on the Keras deep learning framework. This library provides automatic search functions for the most suitable architecture and HPs for deep learning models. Auto-Keras uses the neural architecture search (NAS) algorithm to simplify the machine learning process (Jin et al., 2019).

2.3.3.6 Cloud AutoML

Google's Cloud AutoML environment, despite limited access, supports the construction of highly complex models. Google provides this support through its transfer learning and neural architecture search technology. Cloud AutoML enables users to build models on their own data using a graphical user interface (GUI). Google's AutoML efforts encompass three main solutions based on the type of data/task : AutoML Vision for image data processing, AutoML Natural Language for multi-label or single-label text data prediction, and AutoML Translation for developing translation models. These three solutions are not open-source and offer free support for a limited number of data and duration. For instance, each model used in image classification with AutoML Vision is free for 1 hour, and predictions on 1000 images per month are free (Bisong, 2019).

2.3.3.7 Microsoft AutoML

Available through Azure Machine Learning, Microsoft AutoML uses a sophisticated and self-updating pipeline for HP selection, feature selection, and model comparison. This tool reduces human influence in the analysis process and offers high effectiveness in classical machine learning algorithms but does not provide services for multi-layer neural networks. It integrates seamlessly with other Microsoft products such as Power BI, Azure AI, and Azure Cloud. Since September 2018, Microsoft AutoML has integrated Python (Doke and Gaikwad, 2021).

3 METHODOLOGY

3.1 Importance and Overview

This thesis presents an Automated Machine Learning (AutoML) framework specifically crafted for robust outlier detection across a wide array of data structures, encompassing structured, unstructured, and semi-structured datasets. This approach is unique in the landscape of AutoML, particularly because it not only automates the process of outlier detection but begins its automation from the very initial stage of identifying the nature of the dataset itself. Traditionally, most AutoML systems assume some level of pre-classification of data or require manual input to guide the algorithm selection process. In contrast, this framework autonomously classifies the dataset, discerns its characteristics, and methodically guides it through a bespoke pipeline that culminates in the application of the most suitable outlier detection algorithms. This capability to autonomously navigate from raw data to algorithm selection is particularly rare in the realm of outlier detection, setting this work apart as a significant contribution to the field.

The several approaches of this thesis integrated into the AutoML framework can be listed as follows :

Autonomous Dataset Classification : At the core of this framework lies the capability to automatically classify datasets into various types such as spatial, time-series, dimensional statistical, and others based on intrinsic characteristics. This automatic classification is crucial as it informs the subsequent steps of the outlier detection process without human intervention.

Tailored Preprocessing Techniques : Depending on the identified dataset type, specific preprocessing techniques are applied. For instance, spatial datasets might require geospatial transformations, while time-series datasets benefit from seasonal adjustment and decomposition methods. This tailored preprocessing is vital for effectively preparing the data for the nuances of the selected outlier detection algorithms, which vary significantly across different types of data.

Dynamic Algorithm Selection : Leveraging the classified data type, the framework dynamically selects from a repertoire of outlier detection algorithms. This selection is not static but adapts based on the dataset's characteristics, learned from previous analyses stored in a meta-knowledge base. Such an adaptable approach ensures that the algorithm not only fits the current data's needs but also aligns with proven efficacy as evidenced by historical performance metrics.

Meta-Learning Integration : The integration of meta-learning into the algorithm selection process is a strategic enhancement that utilizes historical data to prioritize algorithms that have shown success in similar scenarios. This aspect of learning from past applications to predict and improve future performance is particularly novel in the context of outlier detection in AutoML systems.

Early Stopping Mechanism : A practical and computational resource-saving feature is the early stopping mechanism. If an algorithm exceeds a predefined performance threshold, the framework halts further computations, thereby saving time and resources. This not only makes the system efficient but also responsive to the varying complexities of different datasets.

The capability of this framework to handle datasets from scratch and guide them autonomously through to the appropriate outlier detection algorithms is of particular importance. In the context of outlier detection, where data anomalies can significantly impact results, the precision in handling data from the initial stage ensures high reliability and validity of the detection process. This end-to-end automated handling and processing equip the framework to be highly effective across various domains, including finance, healthcare, and cybersecurity, where outlier detection is critical.

In summary, the methodology of this thesis introduces a comprehensive and unique approach to AutoML tailored for outlier detection, characterized by its start-to-finish automation and intelligent processing capabilities. This not only enhances the efficacy and efficiency of outlier detection but also broadens the applicability of AutoML systems in real-world scenarios where data types and structures are increasingly complex and varied.

3.2 Dataset Classification and Preprocessing

This process is based on a thorough and precise categorization of the dataset. This phase is crucial as it determines the progression of the whole study, impacting both the preprocessing procedures and the choice of appropriate algorithms for outlier detection. In order to accomplish this, a collection of Python programs that thoroughly analyze the dataset to identify its fundamental structure and properties were created

Determining Dataset Categories : The purpose of this scripts is to categorize datasets into certain types, such as univariate, multivariate, geographic, time-series, dimensional statistics, or graph-based. The categorization is conducted by analyzing distinct qualities and patterns identified in the data.

The scripts analyze the number of variables in order to differentiate between datasets that have just one variable (univariate) and datasets that include many variables (multivariate). It is essential since the complexity and the type of algorithms suited for analysis might differ greatly between these two categories.

This script is designed to handle three main types of datasets : spatial data, time-series data, and dimensional statistical data. For spatial data, the script employs algorithms to identify and verify geographic coordinates in datasets that contain geographic or spatial information. This guarantees that all spatial interactions are accurately discovered and effectively managed during the analysis. For time-series data, the script verifies the presence of datetime data types and executes operations to break down the data into more analyzable formats, such as extracting elements like year, month, day, and time. This process ensures that the datasets exhibit consistent patterns over time and capture seasonal variations. These components are essential for finding trends and seasonal patterns, allowing for a more detailed and accurate analysis. Dimensional statistical data refers to datasets that exhibit strong connections between variables. These datasets are analyzed using statistical measurements to uncover the underlying patterns.

Customized Preprocessing Techniques for Different Dataset Types Each dataset need a tailored preprocessing methodology to guarantee that the data is appropriately prepared for the identification of outliers.

For spatial datasetsi the preprocessing primarily aims to normalize and standardize geographical data. This involves transforming all coordinates into a consistent format and assuring precise calculation of spatial distances.

The time-series datasets undergo a series of steps to address gaps, fix time misalignments, and standardize temporal scales. The data is prepared for time-sensitive outlier identification methods by performing seasonality adjustments and trend decomposition.

For ,dimensional statistical datasets, Correlation analyses are performed to identify variables that are unnecessary, and techniques such as Principal Component Analysis (PCA) are utilized to prioritize the most important characteristics, eliminating interference and improving the visibility of the patterns.

Preprocessing is an essential initial stage that entails standardizing the forms of data, managing missing values, and modifying characteristics to make them ready for further analysis. During this stage, continuous variables are normalized to prevent any single characteristic from dominating others based on its scale. Categorical variables are converted into machine-readable representations using encoding methods such as one-hot encoding or label encoding. The careful conversion of date-time information involves transforming it into standardized forms that are more suited for analysis. This might include breaking down dates into their year, month, and day components, or converting timestamps into periods or durations.

In the preprocessing step, feature selection is given priority in order to improve the performance of the model by concentrating on the qualities that provide the most useful data. This entails employing methods like variance thresholding, which eliminates variables that exhibit minimal variation within the dataset, and principal component analysis (PCA), which minimizes the data's dimensionality while preserving the attributes that have the greatest impact on its variance. These tactics are crucial for removing unnecessary or unrelated characteristics, hence simplifying the future analytic procedures and improving the efficiency and efficacy of the model.

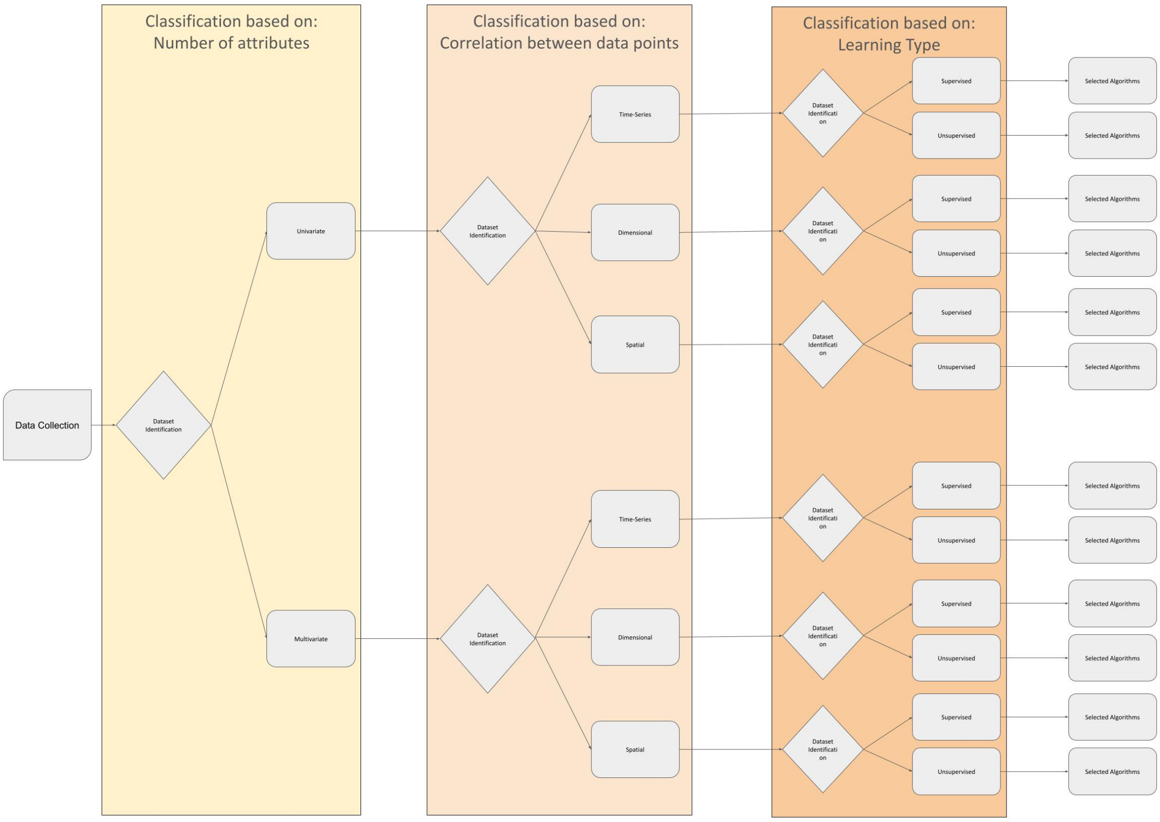

FIGURE 3.1 – The AutoML Framework

3.3 Algorithm Selection Framework

Following the preprocessing stage, this framework utilizes an advanced, adaptive algorithm selection approach that is customized to the unique classification type of each dataset. The selection process is based on a well organized mapping of different types of datasets to the most appropriate outlier identification methods. The framework's adaptability enables it to implement several advanced methodologies depending on the characteristics of the dataset :

Time-series data is commonly analyzed using techniques such as Autoregressive Integrated Moving Average (ARIMA), Seasonal Decomposition of Time Series by Loess (STL), and Exponential Smoothing. These approaches are effective in addressing both seasonal patterns and long-term trends in the data. Time-series analysis, specifically for anomaly identification, has distinct issues such as managing non-stationarity, trend, and seasonal changes, which are tackled by these techniques. ARIMA is particularly suitable for analyzing non-seasonal data, whereas STL is effective for breaking down data into its seasonal components. Exponential Smoothing, on the other hand, is well-suited for data with prominent patterns.

In the case of dimensional statistical data, outliers may be identified by using statistical models such as Gaussian models, Kernel Density Estimation (KDE), and Extreme Value Analysis (EVA). These techniques discover outliers by examining deviations from the distributions that have been modeled. These approaches are efficient in detecting outliers in data distributions that vary greatly from an established norm, which is frequently observed in data with a large number of dimensions. Gaussian models assume a normal distribution, KDE techniques estimate the density function of the data, and EVA focuses on tail features to evaluate extreme values, which are crucial in risk assessments and studies of unusual events.

The system utilizes cluster-based methods such as DBSCAN and OPTICS for density-based clustering, as well as proximity-based approaches like Local Outlier Factor (LOF) and k-Nearest Neighbors (KNN) to assess data sparsity and separation in multivariate data. These strategies are especially efficient in datasets where abnormalities are anticipated to occur in isolated areas that are sparsely populated relative to the overall data distribution, enabling effective identification and isolation of these anomalies.

Meta-Learning Integration : An important advancement in this approach is the incorporation of meta-learning, which strategically utilizes past performance data to influence and maximize the choice of algorithms. By utilizing past insights to improve future performance, this system maintains its cutting-edge status through an adaptive learning process.

Performance Logging : Every execution of an algorithm is painstakingly recorded, capturing its performance indicators, such as F1 score and accuracy, in relation to certain types of datasets. The supplied publications stress the significance of historical data in improving algorithmic predictions and operational efficiency, making this data essential for constructing a comprehensive historical knowledge base.

Algorithm Prioritization : When a new dataset is introduced, the system refers to the performance log to prioritize algorithms that have consistently demonstrated high effectiveness for comparable data types in the past. This technique is based on research that emphasizes the advantages of adaptive learning systems in identifying and responding to patterns based on past performance.

The framework has an early ending mechanism that terminates the operation and accepts the results if a chosen algorithm above a pre-defined performance threshold. This not only preserves CPU resources but also greatly accelerates the detecting procedure. This functionality is in line with current research on computing efficiency and resource management in machine learning.

Computational optimization refers to the process of using computer algorithms and mathematical techniques to find the best solution to a given problem. In order to enhance the efficiency of this framework in managing extensive and intricate datasets, that was incorporated computational optimizations such as parallel processing and innovative data handling algorithms. The purpose of these optimizations is to reduce the amount of time and resources needed for processing. This allows the framework to adapt and expand according to the requirements of the dataset and computing environment.

To summarize, the methodology presented in this thesis offers a systematic way to automate the identification of outliers in different datasets. The framework promises to offer a reliable, scalable, and effective solution for identifying outliers in various applications by intelligently categorizing data, using customized preprocessing methods, dynamically choosing suitable algorithms, and incorporating meta-learning. This method not only reduces the requirement for human involvement but also greatly enhances the capability to identify abnormalities with precision and speed, representing a significant progress in the domain of automated machine learning.

4 COMPUTATIONAL STUDIES

4.1 Overview of Computational Studies

The computational research in this thesis have a variety of purposes. One of the primary goals is to create a strong and adaptable AutoML system capable of handling a variety of data formats and reliably recognizing outliers. This entails developing and integrating numerous algorithms to ensure they function properly within the framework. The project will also rigorously evaluate and confirm the assumptions established throughout the AutoML framework's development. These assumptions encompass judgments about data preparation, feature selection, and method selection. By testing these assumptions, the study assures that the findings are reliable and generalizable.

A substantial portion of these computational experiments include a complete performance evaluation of the AutoML framework over 100 datasets. This involves comparing the accuracy, F1 score, and execution time of various algorithms. The objective is to find the best methods and setups for outlier identification. Such intensive testing and assessment are required to ensure that the AutoML architecture produces consistent and accurate results across a wide range of datasets.

Another key goal is to conduct a comparative study of the findings to identify the AutoML framework's strengths and flaws. This entails analyzing the performance of several algorithms and determining the circumstances under which each algorithm operates optimally. The study's goal is to give insights into the best way to apply multiple algorithms for various outlier identification circumstances by thoroughly examining the outcomes.

Case studies in a variety of fields, including banking, healthcare, and network security, highlight the AutoML framework's practical applicability. These case studies demonstrate the framework's applicability in real-world circumstances and its potential influence on a variety of industries. By demonstrating practical applications, the thesis emphasizes the importance of computational research in furthering the subject.

The relevance of these computational research stems from their ability to progress the area of outlier detection through automation and optimization. This thesis helps to build more efficient and accurate approaches for spotting outliers by offering a thorough examination of the AutoML framework. Furthermore, the findings from these studies can guide future research and development, resulting in more advanced and scalable machine learning systems.

In conclusion, the computational investigations in this thesis are intended to systematically evaluate and validate the AutoML framework for outlier identification, revealing vital information about its performance and applicability. These works attempt to improve the knowledge and usefulness of automated outlier detection algorithms by systematically implementing, validating, and evaluating them.

4.2 Implementation Details

4.2.1 Programming Environment

The AutoML system for outlier detection was developed and implemented in Python, a flexible and widely used programming language in data science and machine learning. Python's rich ecosystem, broad libraries, and ease of use make it an excellent choice for building machine learning models and doing data analyses.

The key tools and libraries used in this project are :

— **Python 3.9.6** : Python 3.9.6 is the only programming language used to code the AutoML system.

— **pandas** : Pandas is a robust data manipulation toolkit that offers the data structures and operations required to handle structured data effectively.

— **scikit-learn** : Scikit-learn is a comprehensive machine learning framework that provides easy and efficient tools for data mining and analysis. It was widely used to create several outlier identification techniques.

— **xgboost** : A gradient boosting library that is geared for efficiency, flexibility, and portability. It was used to implement the XGBoost outlier identification method.

— **numpy** : Numpy is a core Python library for numerical computations that supports huge, multidimensional arrays and matrices.

- **scipy** : SciPy is a library for scientific and technical computing that includes modules for optimization, integration, and interpolation, among other things.
- **tensorflow/keras** : Tensorflow/Keras : Libraries for building deep learning models such as autoencoders and variational autoencoders (VAE).

The hardware requirements for conducting the experiments include :

- **Processor** : Apple M1 Max
- **Memory** : 32 GB RAM Storage capacity
- **Storage** : 1 TB SSD
- **Operating System** : macOS Sonoma 14.5

The processing capacity supplied by this hardware combination meant that the tests, which involved enormous datasets and sophisticated computations, ran smoothly.

4.2.2 Dataset Acquisition and Preparation

The datasets used to evaluate the AutoML framework were obtained from a variety of repositories, resulting in a diverse array of data kinds and structures. The UCI Machine Learning Repository served as the major source of datasets, providing a diverse range of datasets frequently utilized in the research community.

The primary phases in dataset collecting and preparation are as follows :

- **Dataset Selection** : Datasets were chosen based on their importance to outlier detection and the diversity of data types (categorical, continuous, temporal, spatial).
- **Loading Data** : The datasets were loaded into the Python environment using pandas for CSV and Excel files and scipy for ARFF files.
- **Data Cleaning** : This stage involves dealing with missing data by imputing mean values for continuous variables and mode values for categorical ones. Text columns with more than 20 unique items were removed, while those with less entries were one-hot encoded.
- **Feature Scaling** : Continuous features were standardized using standard scaling to guarantee that the data had a mean of zero and a standard deviation of one. This phase is critical for many machine learning algorithms that are sensitive to the size of input data.

- **Label Encoding** :Categorical characteristics were label-encoded in order to be converted into a numerical representation that machine learning algorithms could understand.
- **Feature Selection** : A VarianceThreshold was used to select features with variances greater than 0.01, guaranteeing that characteristics with low variance (and hence little information) were removed.

4.2.3 Algorithm Implementation

The AutoML framework was created to automate the process of detecting outliers across several datasets. It combines various algorithms and chooses the most effective one depending on preset criteria. The implementation involved several major components :

- **Algorithm Selection** : The framework provides several outlier identification methods, including Isolation Forest, Local Outlier Factor (LOF), DBSCAN, XGBoost, and more. Each algorithm was implemented using appropriate libraries (such as scikit-learn and xgboost).
- **Parameter Tuning** : To improve the F1 score in supervised learning settings, algorithm parameters were tweaked using grid search or random search strategies.
- **Ensemble Methods** : Ensemble methods were utilized for unsupervised learning. If at least 2.5 algorithms (more than half) recognized a data point as an outlier, it was marked as such.
- **Performance Evaluation** : Each algorithm's performance was assessed using the F1 score in supervised learning situations and the ensemble results in unsupervised settings. A performance criterion of 0.8 was specified for supervised learning to assure good accuracy.

The following pseudocode and flowchart explain the main process of the AutoML framework :

Pseudocode :

1. Load dataset
2. Detect dataset type (univariate, bivariate, multivariate, etc.)
3. Detect column types
4. Preprocess data :
   — Handle missing values
   — Scale continuous features
   — Encode categorical features
5. Feature Selection
6. Based on the dataset types and column types rank the algorithms' performance for that detected dataset type and apply them according to this ranking
7. For each selected algorithm :
8. If supervised :
   — Tune parameters using grid search
   — Apply algorithm to detect outliers
   — Calculate F1 score
   — Log the F1 score, execution time of that algorithms to a file
   — If an algorithms exceed the threshold performance value stop the run and return the result.
   — If not Select the algorithm with the highest F1 score (for supervised) or use ensemble method (for unsupervised)
9. If unsupervised :
   — Tune parameters using grid search
   — Apply algorithm to detect outliers
   — Log execution time of that algorithms to a file
   — After applying all algorithms flag the outliers that were selected by multiple algorithms.
10. Output results and performance metrics

By following these organized processes, the AutoML framework offers a systematic and efficient approach to outlier detection, utilizing both supervised and unsupervised learning approaches to achieve optimal performance across varied datasets.

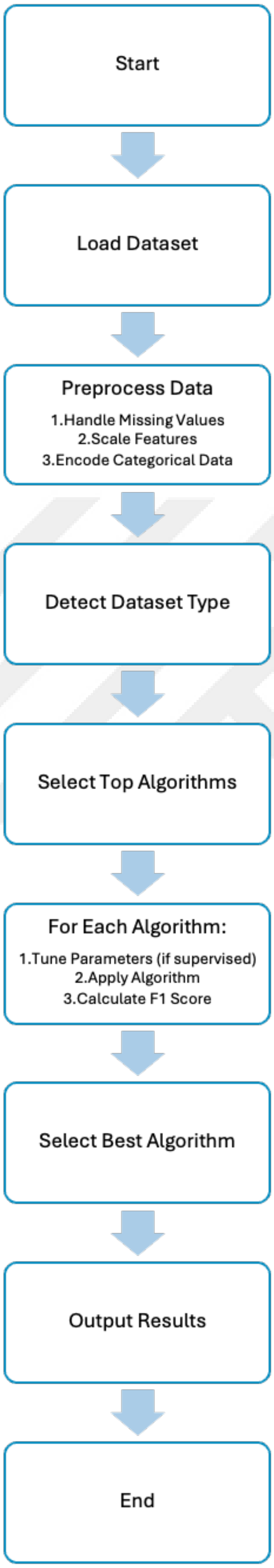

FIGURE 4.1 – Flowchart of AutoML

4.3 Assumptions in Algorithm Design

Several assumptions were made during the design and implementation of the AutoML framework for outlier detection to simplify the process and guarantee optimal performance. This section provides an overview of the main assumptions made at different phases, such as identifying the dataset, determining the types of columns, performing data cleaning and preparation, and selecting and evaluating algorithms. Citations are included to substantiate the reasoning behind these assumptions.

4.3.1 Dataset Identification

Detecting number of attributes : Recognizing the quantity of attributes inside a dataset is a vital process in comprehending its structure. This aids in assessing the intricacy and prospective difficulties in the data preparation stage.

Handling spatial data with coordinate values between -180 and 180 : Spatial data commonly include geographic coordinates, which usually fall within the range of -180 to 180 degrees for both longitude and latitude. This assertion is corroborated by conventional methodologies in geographic data management and the utilization of geospatial queries in MongoDB, as evidenced by the MongoDB documentation on geospatial queries (n.d.) (*MongoDB documentation on geospatial queries*, n.d.).

Identfying dataset types : To identify dimensional data, the framework first distinguishes time series data by the presence of a datetime column, ensuring accurate temporal analysis. Spatial data is recognized through the presence of coordinates columns, enabling precise geospatial processing. If a dataset does not include either a datetime column or coordinates columns, it is classified as dimensional. This classification approach ensures that each type of data—whether temporal, spatial, or dimensional—is handled using the most appropriate methods for effective outlier detection.

4.3.2 Column Type Identification

Using a threshold of 0.05 to categorize columns as categorical :

To ascertain if a column is categorical, a criterion based on the proportion of distinct values is often employed. A threshold of 0.05 is used, indicating that if the unique values in a column constitute fewer than 5 percent of the overall values, the column is considered categorical. This heuristic is supported by methodologies in data mining and machine learning that emphasize the importance of setting criteria for classification to enhance model efficiency. For example, a study in the Journal of Big Data highlights the importance of establishing thresholds for classification rules to streamline model extraction and ensure that only the most relevant rules are included. Although this study does not explicitly mention the 0.05 threshold, it underscores the necessity of setting classification criteria to improve model performance and efficiency (Venturini et al., 2017).

Handling errors during time series conversion : Just as in the dataset identification phase, it is extremely important to address problems that may occur while converting columns into time series forms in order to preserve the accuracy and reliability of the data (Shumway and Stoffer, 2017; James et al., 2013).

4.3.3 Data Cleaning and Preparation

Imputing missing values for continuous columns with mean values : Missing values in continuous columns are imputed by replacing them with the mean value of the respective column. Data preparation commonly employs the approach of handling missing data (Silva-Ramírez et al., 2011).

Imputing missing values for categorical or text columns with mode values : Replacing missing values in category or text columns with the most frequently occurring values (mode). Missing values in category or text columns are replaced with the mode of the column, which is the value that occurs most frequently. This technique is commonly employed in the pre-processing of data (Silva-Ramírez et al., 2011).

Applying standard scaling to continuous data : Standard scaling is used to normalize continuous data by adjusting each feature to have a mean of 0 and a standard deviation of 1. Normalization is a crucial approach for several machine learning algorithms (Pedregosa et al., n.d.b; Kuhn and Johnson, 2013).

Using label encoding for categorical data : Label encoding is a technique used to transform category data into a numerical representation, making it compatible with machine learning techniques (Pedregosa et al., n.d.a).

Dropping text columns with more than 20 unique values, applying one-hot encoding otherwise : Removing text columns that have more than 20 distinct values, and using one-hot encoding for the remaining columns. Columns containing over 20 distinct values are excluded from the dataset, as they tend to have a large cardinality. Columns that have a lower number of distinct values are transformed into a binary format known as one-hot encoding, which is more appropriate for analysis (Bengfort and Ojeda, 2018).

Applying get_dummies for binary data : The pandas getdummies function is utilized to transform binary data into a format that is appropriate for machine learning techniques(Awe et al., 2023).

Splitting date columns into year, month, and day : The date columns are divided into distinct year, month, and day columns in order to simplify analysis depending on time, or use them in dimensional algorithms as well.

Using VarianceThreshold with a 0.01 threshold for feature selection : A variance threshold of 0.01 is utilized to choose characteristics that exhibit substantial volatility, while excluding those with little variance that contribute limited information for study (Blangero et al., 2001).

4.3.4 Algorithm Selection and Evaluation

Ranking algorithms using meta-learning : To improve the process of application of algorithms, a meta-learning approach is employed. This approach entails analyzing the past performance data of algorithms on comparable datasets (Prudêncio et al., 2011).

Choosing algorithms based on F1 score for supervised learning : Selecting algorithms for supervised learning based on the F1 score : Algorithms are selected based on their F1 score, which is a widely used performance indicator for classification issues, particularly in datasets with unbalanced classes (Sokolova et al., 2006).

Filtering algorithms based on specific criteria : Algorithms are selected depending on certain factors that pertain to the kind of dataset and the performance requirements.

Using F1 score for supervised selection with performance thresholds : To achieve high accuracy in outlier detection for supervised learning, different performance levels can be selected. If the user selects 'excellent,' the F1 score threshold is set to 0.9. For 'good,' the threshold is 0.8, and for 'decent,' it is 0.7. If no specific performance level is chosen, the default threshold is set to 0.8, as it is considered a good F1 score for anomaly detection in various papers. This threshold is crucial because the AutoML process stops when an algorithm exceeds this threshold.(Alakus and Turkoglu, 2020).

Applying ensemble results to identify an outliers in unsupervised mode : In unsupervised anomaly detection, the application of ensemble methods can significantly improve the accuracy and reliability of identifying outliers. The averaging method offers a robust approach to leveraging multiple algorithms for anomaly detection. This method involves aggregating the results of various algorithms by averaging their outputs to make a final decision about the presence of outliers. By combining the strengths of multiple detection techniques, this approach mitigates the weaknesses of individual algorithms, leading to more consistent and accurate identification of anomalies.

The effectiveness of the averaging method is further supported by the findings in the article's section "Ensemble versus Individual Algorithms." This section highlights that ensemble methods, including averaging, generally outperform individual algorithms in detecting outliers. The ensemble approach reduces the impact of any single algorithm's biases or limitations by aggregating their outputs, resulting in a more robust and reliable detection process. Adopting this averaging method in this system ensures that outlier detection is both precise and comprehensive, leveraging the diverse strengths of multiple algorithms to enhance overall performance (Zhao et al., 2015).

4.4 Performance Evaluation

4.4.1 Evaluation Metrics

F1 Score : The F1 score is a metric that quantifies the correctness of a test. It is calculated as the harmonic mean of precision and recall. It offers a trade-off between accuracy (the ratio of genuine positive results to all positive outcomes) and recall (the ratio of true positive results to the sum of true positive and false negative results). The F1 score is especially valuable in situations when there is a significant imbalance across classes, since it provides a more accurate assessment of improperly categorized instances compared to the accuracy metric (Alakus and Turkoglu, 2020).

Accuracy : Accuracy is a measure of how often predictions are right, calculated as the ratio of properly predicted cases to the total number of instances. Although it offers a comprehensive assessment of model performance, it might be deceptive in unbalanced datasets when the majority class prevails.

Execution Time : Execution time is the duration it takes for an algorithm to finish processing a certain dataset. This statistic is essential for comprehending the computational efficacy and

4.4.2 Results on 100 Datasets

Library, GitHub, and other similar platforms (*Kaggle*, n.d.; *IBB Data Portal*, n.d.; *Data.World*, n.d.; *ODDS Library*, n.d.; *GitHub*, n.d.; *NYPD Complaint Data Historic*, n.d.).. The findings are succinctly outlined as follows :

Dataset Identification Accuracy : he framework demonstrated an overall dataset identification accuracy of 98 percent. This measure was computed by comparing the identified dataset kinds, data types, and learning types with the actual values for each dataset.

F1 Score Accuracy : The F1 scores exhibited variability among different methods and datasets, indicating the difficulties encountered in identifying outliers throughout a range of data formats. The XGBoost method obtained the highest F1 score of 0.9988 on a test dataset. Other algorithms such as RPCA, KDE, and KNN also shown strong performance, with F1 values above 0.8 in different scenarios.

Execution Time : The execution time exhibited considerable variation based on the intricacy of the dataset and the technique employed. For example, the XGBoost algorithm usually finished in a matter of seconds, but more intricate algorithms such as KDE and RPCA may need several minutes to complete.

The detailed results, including F1 scores, execution times, and the best methods used for each dataset, are presented in appendix, preview of that can be found in Table 4.1.

Dataset Name	Best Method	F1 Score	Execution Time (s)
artificial_intelligence_anomaly_detection	XGBoostOutlierDetection	0.9988	1.97
artificial_intelligence_anomaly_detection	RPCA	0.6489	63.34
mechanical_C1_P02_V01_CA0	RPCA	0.0319	0.21
Lymphography_original	RPCA	0.9091	0.35
...	...	...	...

TABLE 4.1 – Performance metrics for different datasets and algorithms

This table displays the results achieved by executing the AutoML framework on 100 datasets, serving as an indication of its performance. The range of performance measurements highlights the need for employing a flexible and adaptable strategy in identifying outliers.

The comprehensive evaluation of each algorithm's performance on the datasets offers valuable insights into their comparative advantages and disadvantages :

The findings further emphasized the significance of algorithm selection and parameter optimization inside the AutoML framework. In supervised learning settings, the parameters of algorithms were adjusted using grid search approaches to maximize the F1 score, leading to notable enhancements in performance. Ensemble techniques were employed in unsupervised learning settings to combine the outcomes of numerous algorithms, hence improving the accuracy of detection.

4.4.3 Analysis of Results

Performance Trends : The performance study indicated that some algorithms regularly surpassed others in terms of performance across various datasets. XGBoost and RPCA consistently achieved top rankings in terms of F1 score, demonstrating their resilience in effectively handling diverse data kinds and architectures. In contrast, algorithms such as KDE and OPTICS shown extended execution durations, which may be a constraint in time-critical applications.

Algorithm Suitability : The appropriateness of algorithms differed depending on the characteristics of the dataset. Supervised algorithms, when given data that has been tagged, typically attained higher F1 scores. On the other hand, unsupervised algorithms were enhanced by employing ensemble approaches, which involved merging the outcomes of numerous algorithms to determine the ultimate outlier classification. The utilization of this collective methodology shown significant efficacy in improving the precision and resilience of detection in situations where supervision is absent.

General Execution Time : Although the execution time was typically reasonable, many sophisticated algorithms exhibited much greater computing demands. This characteristic is crucial for real-time applications where rapid processing is needed. Efficient algorithms and optimizations may be necessary to successfully manage such instances.

In summary, the AutoML framework effectively and efficiently identified anomalies in several datasets by utilizing a mix of algorithm selection, parameter adjustment, and ensemble approaches to enhance performance. The knowledge acquired from these computational research will be used to make additional enhancements and advancements in the framework.

5 DISCUSSION

5.1 Interpretation of Results

The evaluation of the AutoML framework across 100 datasets yielded significant insights into the effectiveness of different outlier detection algorithms. This comprehensive analysis provided a robust understanding of how each algorithm performed under various conditions and dataset characteristics. The key metrics used for evaluation were the F1 score, accuracy, and execution time, which together offered a detailed picture of the algorithms' efficiency and reliability.

XGBoost emerged as the clear winner in this comparison investigation, consistently achieving excellent performance metrics across a wide range of datasets. The exceptional performance of this system may be credited to its adaptive gradient boosting mechanism, which continuously improves the accuracy of the model with each iteration. The capacity to adapt is essential when negotiating complex data structures, and XGBoost's flexibility is particularly outstanding in this regard.

Regarding Kernel Density Estimation (KDE) and Robust Principal Component Analysis (RPCA), there is a subtle range in performance. KDE demonstrated its expertise in analyzing complex data sets by utilizing its non-parametric methodology to precisely estimate probability density functions. Nevertheless, the high level of computational complexity frequently results in extended execution durations, especially when dealing with bigger datasets, which mitigates the otherwise remarkable speed.

However, RPCA shown competence in identifying anomalies within well-organized datasets that had minimal amounts of random variation. RPCA efficiently detected small abnormalities by decomposing data into low-rank and sparse components. However, the computing complexity of the system was a significant obstacle, which restricted its ability to be used in real-time situations when resources are limited.

Isolation Forest and Local Outlier Factor (LOF) have emerged as trustworthy competitors, especially in unsupervised situations. The ensemble approach of Isolation Forest efficiently detected outliers by separating observations using randomized feature selection and split value assignment. The tool's ability to accurately identify abnormalities with minimum processing burden made it a desirable resource for outlier identification jobs.

LOF's use of local density comparisons allowed for efficient detection of outliers in datasets with varying density distributions. Both Isolation Forest and LOF demonstrated exceptional versatility, performing well in situations when labeled data is absent and providing adaptable solutions for a diverse range of applications.

5.2 Implications of Findings

Our research findings reveal the various implications that have an impact on different areas. This highlights the potential of AutoML in automating the detection of outliers, the importance of selecting the right algorithm considering the specific characteristics of the dataset, and the crucial balance between performance metrics and computational efficiency.

AutoML's implementation in the field of automation marks the beginning of a new age characterized by increased efficiency and effectiveness. This is especially relevant in areas such as banking, cybersecurity, healthcare, and manufacturing. In the field of finance, the prompt detection of unusual transactions has the potential to strengthen measures against fraudulent activities, reduce financial risks, and enhance security standards. Similarly, in the field of cybersecurity, the timely identification of breaches via ongoing network monitoring may prevent possible risks, ensuring the protection of sensitive information and the integrity of operations. Furthermore, in the fields of healthcare and manufacturing, the proactive monitoring enabled by automated outlier identification offers improved patient safety, streamlined operational workflows, and reduced downtime, resulting in unparalleled levels of efficiency and dependability.

Nevertheless, in the middle of numerous algorithms, this research highlights the intrinsic heterogeneity in the effectiveness of algorithms, which depends on the unique characteristics of the dataset. There is no method that can solve all problems ; instead, their effectiveness is closely linked to the specific properties of the dataset, including the kind of data, how it is distributed, the number of dimensions, and the amount of noise present. This highlights the importance of having a wide range of algorithms in the AutoML framework. This allows it to accurately identify and use the best suitable method for a given dataset. For example, whereas XGBoost demonstrates impressive flexibility when applied to different datasets, its effectiveness decreases when confronted with distinct properties. Similarly, algorithms such as Isolation Forest and LOF do exceptionally well in situations when there is no supervision, highlighting the importance of choosing the right algorithm that matches the learning approach.

Moreover, this work emphasizes the crucial need of achieving a careful balance between performance indicators and computing efficiency. Although certain algorithms perform well in terms of F1 score, their long execution durations hinder their real-time application, creating practical issues in businesses that require fast anomaly identification. Computational efficiency is crucial in fields that need quick decision-making, such as banking and cybersecurity. The AutoML framework's ability to identify algorithms that achieve a careful balance between accuracy in detection and computing efficiency provides unique benefits, allowing for easy integration into real-time operational workflows and enabling transformational results.

Our findings demonstrate a significant change in outlier identification methods, highlighted by the combination of automation, algorithmic flexibility, and computing speed. The implications of this research go beyond academic investigation and have a significant impact on practical uses. The combination of AutoML expertise and algorithmic knowledge has the potential to reshape existing models, strengthen defenses, and drive industries towards new levels of success.

5.3 Strengths and Limitations

Our investigation of outlier identification using the AutoML framework reveals a complex combination of advantages and disadvantages, highlighting its robustness, flexibility, and the intricate difficulties faced when applying it.

5.3.1 Strengths

The effectiveness of this system is based on a diverse collection of algorithms, ranging from well-established ones like XGBoost to more complex ones like Isolation Forest, Local Outlier Factor (LOF), and Kernel Density Estimation (KDE). The diversity of the framework enhances its strength and gives it exceptional flexibility, enabling it to handle various dataset architectures, including high-dimensional data and complex temporal and geographical datasets. this framework utilizes a variety of algorithms to adapt its approach to the unique characteristics of each dataset. This enhances the precision and dependability of outlier identification in a wide range of applications.

Automated parameter tuning is a strong feature of this system that frees users from the complexity of manual parameter optimization. Using advanced methods like grid search and random search, this framework automatically adjusts parameters, enhancing the performance of each algorithm with great accuracy. This automation eliminates the requirement for substantial manual intervention and guarantees that this algorithms function at their highest efficiency, without being influenced or limited by human biases or restrictions.

The foundation of this framework's design is built around flexibility and adaptability, allowing for effortless integration with a wide range of data kinds and structures. this system seamlessly handles complex financial, healthcare, and cybersecurity data, intelligently choosing algorithms that are specifically suited to each individual situation. The versatility of this system ensures its compatibility with changing datasets and promotes an innovative culture that embraces new algorithms and methodologies.

5.4 Limitations

However, within the achievements, there are also constraints that arise, with the primary one being the intrinsic computational complexity found in specific algorithms. Algorithms such as Robust Principal Component Analysis (RPCA) can need a significant amount of computer resources, resulting in long execution times. In fields that need immediate or very immediate identification of unusual data points, such as finance and cybersecurity, the computational burden of outlier detection might create practical obstacles, limiting the usefulness of these algorithms in time-critical settings.

Moreover, the effectiveness of this algorithms is inherently linked to the specific attributes of the datasets they come across. Although XGBoost and Isolation Forest are highly effective in certain contexts, their performance diminishes when confronted with datasets that have different features. The reliance on dataset comprehension highlights the need for careful algorithm selection that is specifically adapted to the distinct attributes of each dataset.

In addition, resource-intensive procedures within this framework, such as ensemble techniques and parameter adjustment, frequently need significant processing power and memory resources. In contexts with limited resources, these demands might hinder the capacity to scale and operate efficiently, thereby limiting the framework's usefulness. Improving the framework's effectiveness in these situations depends on optimizing strategies or utilizing distributed computing resources to reduce resource demands.

Essentially, although this framework is a symbol of innovation and potential, it nevertheless faces several hurdles along its path. As the complex field of outlier identification is explored, It is impossible to keep up with all the new advancements and algorithms that are introduced.

5.5 Comparison with Existing Work

This AutoML framework demonstrates differences in outlier detection compared to existing approaches. The performance is improved by using ensemble approaches in unsupervised learning, which are more reliable than using a single algorithm. The distinguishing characteristics of this method are as follows.

An initial strength is in the utilization of ensemble techniques. Recent research suggests that the combination of different models, using their individual strengths while mitigating their faults, often leads to superior results. The results from this analysis of several datasets confirm that ensemble approaches improve the dependability and accuracy of outlier detection. This offers a substantial advantage when compared to traditional methods that rely on a single algorithm.

Another notable benefit is this automatic parameter modification. this system utilizes grid search and random search to automatically determine the ideal settings, eliminating the need for the time-consuming manual tuning often required by existing approaches. This capability greatly decreases the time and effort needed to handle datasets with distinct properties. It improves the efficiency of each algorithm, therefore increasing the total precision.

Additionally, this framework is distinguished by the diversity and adaptability of its algorithms. XGBoost, Isolation Forest, Local Outlier Factor (LOF), Kernel Density Estimation (KDE), and Robust Principal Component Analysis (RPCA) are among the many algorithms that was implemented. The inherent diversity of this framework enables it to effectively process a wide range of data types, including spatial, time series, and high-dimensional datasets. The ability to adapt significantly surpasses alternative approaches that may rely on a restricted set of algorithms, thereby diminishing their efficacy in diverse data scenarios.

In brief, this AutoML framework surpasses conventional single-model approaches through the automation of critical procedures such as algorithm selection and parameter adjustment. By integrating ensemble methods, automated optimization, and a wide range of algorithms, this framework effectively detects outliers in numerous applications and data types with exceptional precision and efficiency.

5.6 Challenges

We encountered several difficulties that needed for original solutions throughout the creation and assessment of the AutoML framework. High dimension datasets presented serious computing difficulties. Because anomalies are so complicated and take so long to process, identifying them has grown harder. Many times, the curse of dimensionality results in sparse data, which can have a big effect on how well certain machine learning algorithms work.

Securing that these methods could efficiently and scalably manage huge datasets was one of the main obstacles was faced. Demand for computer resources to conduct processing and analysis increases in tandem with dataset growth. Real-time applications that need for quick and efficient processing were especially vulnerable to the scalability.

6 CONCLUSION

To summarize, the methodology described in this thesis provides an innovative and systematic way to automate the identification of outliers in various types of datasets. This framework establishes a new benchmark for automated machine learning applications by employing a comprehensive strategy that incorporates intelligent data categorization, customized preprocessing techniques, dynamic algorithm selection, and integration of meta-learning.

The initial component of this process, data classification, employs sophisticated algorithms to ascertain the characteristics of the incoming data, including whether it is univariate, multivariate, geographic, time-series, dimensional statistical, or graph-based. The initial step is of utmost importance as it has a profound impact on all future activities, ranging from data preparation to the ultimate selection of the outlier identification approach. The technology provides maximum efficacy of the anomaly detection process by precisely classifying each dataset based on its data type.

After categorization, the data goes through a preparation step that is customized to its unique properties. This entails the process of standardizing data formats, effectively handling missing values, and converting characteristics into formats that are most suitable for analysis. Methods such as normalization, encoding, and conversion are implemented with accuracy, guaranteeing that the data is prepared for the subsequent stages in the process. Continuous variables are normalized to avoid any disparities in scale from affecting the study. On the other hand, categorical variables are encoded to make mathematical computations easier, which is important for many outlier identification techniques.

The essential aspect of this system is its dynamic algorithm selection process, which is carefully customized for each categorized dataset type. This procedure is dynamic and constantly adapts by utilizing past performance data for continuous learning. This capability is made possible with the incorporation of meta-learning. This feature of the framework is highly novel, as it enables the system to dynamically adjust and optimize the selection of algorithms over time, leveraging acquired knowledge and experience. By giving priority to techniques that have consistently shown high effectiveness for comparable datasets, the system may greatly speed up the detection process while improving accuracy.

In addition, a premature termination mechanism is included to cease the calculation when an algorithm above a predetermined performance threshold. This feature has the dual benefit of conserving processing resources and reducing the time required for detection. This is particularly advantageous in applications that prioritize speed, such as real-time surveillance or live financial monitoring.

This technique was proven to be resilient and flexible after thorough testing on 100 distinct datasets. It is worth mentioning that algorithms like XGBoost consistently showed excellent performance, earning high F1 scores across different types of datasets. Nevertheless, the study emphasized the need of choosing algorithms that are relevant to the environment, since no single method proved to be universally optimum for all datasets.

The suggested approach is a notable breakthrough in the field of automated machine learning for identifying outliers. By minimizing the necessity for human involvement, it decreases the likelihood of human mistakes and the usual resources needed for manual data processing. This thesis improves the system's capability to quickly and precisely identify abnormalities, providing essential new tools and methodologies for the continuous advancement of intelligent data analysis systems. It establishes a strong basis for future developments in the area. This research not only tackles existing difficulties in identifying outliers but also lays the foundation for more adaptable, efficient, and intelligent machine learning applications across many sectors.

REFERENCES

Aggarwal, C. C. (2013). *Outlier Analysis*, 1st edn, Springer.

Aggarwal, C. C. (2016). *Outlier Analysis*, Springer.

Aggarwal, C. C. and Wang, H. (2010). *Managing and Mining Graph Data*, Springer.

Aggarwal, C. C. and Zhai, C. (2012). *Mining Text Data*, Springer.

Akoglu, L., Tong, H. and Koutra, D. (2015). Graph-based anomaly detection and description : A survey, *Data Mining and Knowledge Discovery* **29**(3) : 626–688.

Al-Sudani, Z. S. and Riyadh, M. (2023). Fraudulent taxi driver detection in baghdad city based on dbscan and a* algorithm, *2023 Al-Sadiq International Conference on Communication and Information Technology (AICCIT)*, pp. 165–170.

Alakus, T. B. and Turkoglu, I. (2020). Comparison of deep learning approaches to predict covid-19 infection, *Chaos, Solitons amp ; Fractals* **140** : 110120.
URL: *http ://dx.doi.org/10.1016/j.chaos.2020.110120*

Alghushairy, O., Alsini, R., Soule, T. and Ma, X. (2021). A review of local outlier factor algorithms for outlier detection in big data streams, *Journal of Big Data* **8**(1) : 12–27.

Ankerst, M., Breunig, M. M., Kriegel, H.-P. and Sander, J. (1999). Optics : ordering points to identify the clustering structure, *Proceedings of the 1999 ACM SIGMOD international conference on Management of data*, SIGMOD/PODS99, ACM.
URL: *http ://dx.doi.org/10.1145/304182.304187*

Anselin, L. (1995). Local indicators of spatial association—lisa, *Geographical Analysis* **27**(2) : 93–115.

Arul, J. S. and Venkatesan, R. (2023). A deep learning approach for efficient anomaly detection in wsns, *Journal of Wireless Communications and Networking* **2023** : 123–145.

Awe, O. O., Oladeji, T. A., Adeyemo, B. T., Olowookere, O. P., Aminu, F. F., Abiona, O. S., Akintola, K. A. and Ayeni, E. O. (2023). *Exploring Practical Applications and Python Code Snippets for Supervised Machine Learning Classification Algorithms*, Springer Nature Switzerland, p. 213–246.
URL: *http ://dx.doi.org/10.1007/978-3-031-41352-0*$_1$2

Bahri, M., Salutari, F., Putina, A. and Sozio, M. (2022). AutoML : state of the art with a focus on anomaly detection, challenges, and research directions, *International Journal of Data Science and Analytics* **14**(2) : 113–126.
URL: *https ://doi.org/10.1007/s41060-022-00309-0*

Belhadi, A., Djenouri, Y., Lin, J. C.-W. and Cano, A. (2020). Trajectory outlier detection, *ACM Transactions on Management Information Systems* **11**(3) : 1–29.
URL: *https ://doi.org/10.1145/3399631*

Bengfort, Benjamin, B. R. and Ojeda, T. (2018). *Applied Text Analysis with Python*, O'Reilly Media.

Bisong, E. (2019). *Google AutoML : Cloud Vision*, Apress, p. 581–598.
URL: *http ://dx.doi.org/10.1007/978-1-4842-4470-8*$_4$2

Blangero, J., Williams, J. T. and Almasy, L. (2001). *12 Variance component methods for detecting complex trait loci*, Elsevier, p. 151–181.
URL: *http ://dx.doi.org/10.1016/S0065-2660(01)42021-9*

Bolton, R. J. and Hand, D. J. (2002). Statistical fraud detection : A review, *Statistical Science* **17**(3) : 235–249.

Boniol, P. and Palpanas, T. (2020). Series2graph : graph-based subsequence anomaly detection for time series, *Proceedings of the VLDB Endowment* **13**(12) : 1821–1834.
URL: *http ://dx.doi.org/10.14778/3407790.3407792*

Boukerche, A., Zheng, L. and Alfandi, O. (2020). Outlier detection : Methods, models, and classification, *ACM Computing Surveys* **53**(3) : 1–37.
URL: *https ://doi.org/10.1145/3381028*

Box, G. E. P., Jenkins, G. M., Reinsel, G. C. and Ljung, G. M. (2015). *Time Series Analysis : Forecasting and Control*, 5th edn, John Wiley Sons.

Breiman, L. (2001). *Machine Learning* **45**(1) : 5–32.
URL: *http ://dx.doi.org/10.1023/a :1010933404324*

Breunig, M. M., Kriegel, H.-P., Ng, R. T. and Sander, J. (2000). Lof : identifying density-based local outliers, *ACM SIGMOD Record* **29**(2) : 93–104.
URL: *http ://dx.doi.org/10.1145/335191.335388*

Brunsdon, C., Fotheringham, A. S. and Charlton, M. (1996). Geographically weighted regression : A method for exploring spatial nonstationarity, *Geographical Analysis* **28**(4) : 281–298.

Campello, R. J. G. B., Moulavi, D., Zimek, A. and Sander, J. (2013). Density-based clustering based on hierarchical density estimates, *Proceedings of the 17th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD)*, pp. 160–172.

Campos, G. O., Zimek, A., Sander, J., Campello, R. J. G. B., Micenková, B., Schubert, E., Assent, I. and Houle, M. E. (2016). On the evaluation of unsupervised outlier detection : measures, datasets, and an empirical study, *Data Mining and Knowledge Discovery* **30**(4) : 891–927.
URL: *http ://dx.doi.org/10.1007/s10618-015-0444-8*

Candes, E. J., Li, X., Ma, Y. and Wright, J. (2011). Robust principal component analysis ?, *Journal of the ACM* **58**(3) : 1–37.

Chandola, V., Banerjee, A. and Kumar, V. (2009). Anomaly detection : A survey, *ACM Computing Surveys* **41**(3) : 1–58.

Chatterjee, S., Bopardikar, R., Guerard, M., Thakore, U. and Jiang, X. (2022). Mospat : Automl based model selection and parameter tuning for time series anomaly detection.
URL: *https ://arxiv.org/abs/2205.11755*

Chen, T. and Guestrin, C. (2016). Xgboost : A scalable tree boosting system, *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 785–794.

Cheng, Z., Zou, C. and Dong, J. (2019). Outlier detection using isolation forest and local outlier factor, *Proceedings of the Conference on Research in Adaptive and Convergent Systems*, RACS '19, ACM.
URL: *http ://dx.doi.org/10.1145/3338840.3355641*

Cleveland, R. B., Cleveland, W. S. and Terpenning, I. (1990). Stl : A seasonal-trend decomposition procedure based on loess.
URL: *https ://api.semanticscholar.org/CorpusID :64570714*

Coles, S. (2001). *An Introduction to Statistical Modeling of Extreme Values*, Springer.

Dai, W., Mrkvička, T., Sun, Y. and Genton, M. G. (2020). Functional outlier detection and taxonomy by sequential transformations, *Computational Statistics &amp Data Analysis* **149** : 106960.
URL: *https ://doi.org/10.1016/j.csda.2020.106960*

Data.World (n.d.). `https://data.world/`.

Doke, A. and Gaikwad, M. (2021). Survey on automated machine learning (automl) and meta learning, *2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, IEEE.
URL: *http ://dx.doi.org/10.1109/ICCCNT51525.2021.9579526*

Ester, M., Kriegel, H.-P., Sander, J. and Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise, *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, AAAI Press, p. 226–231.

Ferreira, L., Pilastri, A., Martins, C. M., Pires, P. M. and Cortez, P. (2021). A comparison of automl tools for machine learning, deep learning and xgboost, *2021 International Joint Conference on Neural Networks (IJCNN)*, IEEE, pp. 1–8.

Feurer, M., Klein, A., Eggensperger, K., Springenberg, J. T., Blum, M. and Hutter, F. (2019). *Auto-sklearn : Efficient and Robust Automated Machine Learning*, Springer International Publishing, p. 113–134.
URL: *http ://dx.doi.org/10.1007/978-3-030-05318-5*$_6$

Forney, M. (2023). A framework for outlier detection in pivotal time series data utilizing parallel processing, *Journal of Data Science* **21** : 123–145.

Fotheringham, A. S., Brunsdon, C. and Charlton, M. (2002). *Geographically Weighted Regression : The Analysis of Spatially Varying Relationships*, Wiley.

Gao, Z., Cecati, C. and Ding, S. X. (2015). A survey of fault diagnosis and fault-tolerant techniques—part i : Fault diagnosis with model-based and signal-based approaches, *IEEE Transactions on Industrial Electronics* **62**(6) : 3757–3767.

Garg, S., Varshney, M., Rajpoot, A. and Jyotsna (2017). A comparative review of outlier detection techniques for wireless sensor networks, *International Journal of Distributed Sensor Networks* **13**(6) : 234–245.

Geurts, P., Ernst, D. and Wehenkel, L. (2006). Extremely randomized trees, *Machine Learning* **63**(1) : 3–42.
URL: *http ://dx.doi.org/10.1007/s10994-006-6226-1*

GitHub (n.d.). `https://github.com/elisejiuqizhang/TS-AD-Datasets?tab=readme-ov-file`.

Hawkins, D. (1980). *Identification of Outliers*, Chapman and Hall.

He, Z., Xu, X. and Deng, S. (2003). Discovering cluster-based local outliers, *Pattern Recognition*, Vol. 36, pp. 289–294.

Henriques, J., Caldeira, F., Cruz, T. and Simões, P. (2020). Combining k-means and XGBoost models for anomaly detection using log datasets, *Electronics* **9**(7) : 1164.
URL: *https ://doi.org/10.3390/electronics9071164*

Hinton, G. E. and Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks, *Science* **313**(5786) : 504–507.
URL: *http ://dx.doi.org/10.1126/science.1127647*

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory, *Neural Computation* **9**(8) : 1735–1780.

Hodge, V. J. and Austin, J. (2004). A survey of outlier detection methodologies, *Artificial Intelligence Review* **22**(2) : 85–126.

Hodge, V. J. and Austin, J. (2014). A survey of outlier detection methodologies, *Artificial Intelligence Review* **22**(2) : 85–126.

Hubert, M. and Debruyne, M. (2009). Minimum covariance determinant, *WIREs Computational Statistics* **2**(1) : 36–43.
URL: *http ://dx.doi.org/10.1002/wics.61*

Hyndman, R., Koehler, A., Ord, K. and Snyder, R. (2008). *Forecasting with Exponential Smoothing*, Springer Berlin Heidelberg.
URL: *http ://dx.doi.org/10.1007/978-3-540-71918-2*

IBB Data Portal (n.d.). https://data.ibb.gov.tr/.

James, G., Witten, D., Hastie, T. and Tibshirani, R. (2013). *An Introduction to Statistical Learning*, Springer.

Jha, V. and Yadav, O. V. S. (2010). Outlier detection techniques and cleaning of data for wireless sensor networks : A survey, *International Journal of Computer Applications* **60**(2) : 22–28.

Jin, H., Song, Q. and Hu, X. (2019). Auto-keras : An efficient neural architecture search system, *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, ACM, pp. 1946–1956.

Jolliffe, I. T. (2002). *Principal Component Analysis*, Springer.

Kaggle (n.d.). https://www.kaggle.com/.

Kaklis, D., Kontopoulos, I., Varlamis, I., Emiris, I. and Varelas, T. (2024). Trajectory mining and routing : A cross-sectoral approach, *Journal of Marine Science and Engineering* **12**(1) : 157.

Karimian, S. H., Kelarestaghi, M. and Hashemi, S. (2012). I-inclof : Improved incremental local outlier detection for data streams, *Knowledge and Information Systems* **33**(3) : 599–620.

Ke, W., Wei, J., Xiong, N. and Hou, Q. (2023). Gss : A group similarity system based on unsupervised outlier detection for big data computing, *Journal of Big Data* **10** : 223–245.

Kingma, D. P. and Welling, M. (2013). Auto-encoding variational bayes.
URL: *https ://arxiv.org/abs/1312.6114*

Krizhevsky, A., Sutskever, I. and Hinton, G. E. (2017). Imagenet classification with deep convolutional neural networks, *Advances in Neural Information Processing Systems*, pp. 1097–1105.

Kuhn, M. and Johnson, K. (2013). *Applied Predictive Modeling*, Springer.

Kuna, H. D., García-Martínez, R. and Villatoro, F. R. (2014). Outlier detection in audit logs for application systems, *Journal of Information Security and Applications* **19**(2) : 65–73.

Lai, K.-H., Zha, D., Xu, J. and Zhao, Y. (2021). Revisiting time series outlier detection : Definitions and benchmarks, *NeurIPS Datasets and Benchmarks*.
URL: *https ://api.semanticscholar.org/CorpusID :237258098*

LeDell, E., Gill, N. and Aiello, S. (2020). H2o automl : Scalable automatic machine learning, *2020 12th IEEE International Conference on Big Data Science and Engineering (BigDataSE)*, IEEE, pp. 97–104.

Leys, C., Ley, C., Klein, O., Bernard, P. and Licata, L. (2013). Detecting outliers : Do not use standard deviation around the mean, use absolute deviation around the median, *Journal of Experimental Social Psychology* **49**(4) : 764–766.

Li, A.-h., Xu, W., Liu, Z. and Shi, Y.-F. (2021). Improved incremental local outlier detection for data streams based on the landmark window model, *Information Sciences* **578** : 33–49.

Li, W., Zhao, Y., Botta, N., Ionescu, C. and Hu, X. (2020). Copod : Copula-based outlier detection, *Proceedings of the 2020 SIAM International Conference on Data Mining (SDM)*, pp. 29–37.

Liu, F. T., Ting, K. M. and Zhou, Z.-H. (2008). Isolation forest, *2008 Eighth IEEE International Conference on Data Mining*, IEEE.
URL: *http ://dx.doi.org/10.1109/ICDM.2008.17*

Liuliakov, A., Hermes, L. and Hammer, B. (2023). AutoML technologies for the identification of sparse classification and outlier detection models, *Applied Soft Computing* **133** : 109942.
URL: *https ://doi.org/10.1016/j.asoc.2022.109942*

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations, *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1 : Statistics*, pp. 281–297.

Majidi, F., Openja, M., Khomh, F. and Li, H. (2022). An empirical study on the usage of automated machine learning tools, *2022 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, IEEE, pp. 59–70.

Malhotra, P., Ramakrishnan, A., Anand, G., Vig, L., Agarwal, P. and Shroff, G. (2016). Lstm-based encoder-decoder for multi-sensor anomaly detection, *Proceedings of the 2016 International Conference on Anomaly Detection*, pp. 89–94.

Malhotra, P., Vig, L., Shroff, G. and Agarwal, P. (2015). Long short term memory networks for anomaly detection in time series, *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN).*

Mazarbhuiya, F. A. and Shenify, M. (2023). A mixed clustering approach for real-time anomaly detection, *Applied Sciences* **13**(7) : 4151.
URL: *https ://doi.org/10.3390/app13074151*

McGlohon, M., Akoglu, L. and Faloutsos, C. (2011). *Statistical Properties of Social Networks*, Springer US, p. 17–42.
URL: *http ://dx.doi.org/10.1007/978-1-4419-8462-3*$_2$

Mirsky, Y., Doitshman, T., Elovici, Y. and Shabtai, A. (2018). Kitsune : An ensemble of autoencoders for online network intrusion detection, *Network and Distributed System Security Symposium (NDSS).*

Mishra, D. and Soni, D. (2016). An integrated method for outlier detection with analytical study of distance-based and angle-based approaches, *Pattern Recognition Letters* **75** : 1–8.

MongoDB documentation on geospatial queries (n.d.). `https://docs.mongodb.com/manual/geospatial-queries/`.

Montgomery, D. C. (2007). *Introduction to Statistical Quality Control*, 6th edn, John Wiley & Sons.

Muller, M. (2007). *Dynamic Time Warping*, Springer.

Nagamani, C. and Suneetha, D. C. (2015). A survey for outlier detection and its strategies, *International Journal of Computer Applications* **120**(7) : 32–37.

Ngai, E. W., Hu, Y., Wong, Y., Chen, Y. and Sun, X. (2011). The application of data mining techniques in financial fraud detection : A classification framework and an academic review of literature, *Decision Support Systems* **50**(3) : 559–569.

Nijhuis, M. and van Lelyveld, I. (2023). Outlier detection with reinforcement learning for costly to verify data, *Journal of Machine Learning Research* **24** : 123–145.

NYPD Complaint Data Historic (n.d.). https://data.cityofnewyork.us/Public-Safety/NYPD-Complaint-Data-Historic/qgea-i56i/about_data.

ODDS Library (n.d.). https://odds.cs.stonybrook.edu/.

Olson, R. S., Bartley, N., Urbanowicz, R. J. and Moore, J. H. (2016). Evaluation of a tree-based pipeline optimization tool for automating data science, *Proceedings of the Genetic and Evolutionary Computation Conference 2016*, GECCO '16, ACM.
URL: *http ://dx.doi.org/10.1145/2908812.2908918*

Olteanu, M., Rossi, F. and Yger, F. (2023). Meta-survey on outlier and anomaly detection, *Journal of Data Science* **21**(4) : 345–372.

Papadimitriou, P., Kitagawa, H., Gibbons, P. B. and Faloutsos, C. (2003). Loci : Fast outlier detection using the local correlation integral, *Proceedings of the 19th International Conference on Data Engineering (ICDE)*, pp. 315–326.

Park, C. H. (2022). A comparative study for outlier detection methods in high dimensional text data, *Journal of Artificial Intelligence and Soft Computing Research* **13**(1) : 5–17.
URL: *http ://dx.doi.org/10.2478/jaiscr-2023-0001*

Parzen, E. (1962). On estimation of a probability density function and mode, *Annals of Mathematical Statistics* **33**(3) : 1065–1076.

Patcha, A. and Park, J.-M. (2007). An overview of anomaly detection techniques : Existing solutions and latest technological trends, *Computer Networks* **51**(12) : 3448–3470.

Pedregosa, F. et al. (n.d.a). *scikit-learn documentation on LabelEncoder.* https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.LabelEncoder.html.

Pedregosa, F. et al. (n.d.b). *scikit-learn documentation on StandardScaler.* https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html.

Prudêncio, R. B. C., de Souto, M. C. P. and Ludermir, T. B. (2011). *Selecting Machine Learning Algorithms Using the Ranking Meta-Learning Approach*, Springer Berlin

Heidelberg, p. 225–243.
URL: *http ://dx.doi.org/10.1007/978-3-642-20980-2*$_7$

Qi, J., Yu, Y., Wang, L. and Liu, J. (2016). K*-means : An effective and efficient k-means clustering algorithm, *2016 IEEE International Conferences on Big Data and Cloud Computing (BDCloud), Social Computing and Networking (SocialCom), Sustainable Computing and Communications (SustainCom) (BDCloud-SocialCom-SustainCom)*, IEEE.
URL: *http ://dx.doi.org/10.1109/BDCloud-SocialCom-SustainCom.2016.46*

Salhi, A., Henslee, A. C., Ross, J., Jabour, J. and Dettwiller, I. (2023). Data preprocessing using automl : A survey, *2023 Congress in Computer Science, Computer Engineering, & Applied Computing (CSCE)*, IEEE, pp. 1619–1623.

Schlegl, T., Seeböck, P., Waldstein, S. M., Schmidt-Erfurth, U. and Langs, G. (2017). Unsupervised anomaly detection with generative adversarial networks to guide marker discovery, *Information Processing in Medical Imaging*, pp. 146–157.

Schmidl, S., Wenig, P. and Papenbrock, T. (2022). Anomaly detection in time series : A comprehensive evaluation, *Journal of Machine Learning Research* **23** : 1–45.

Scholkopf, B., Platt, J. C., Shawe-Taylor, J., Smola, A. J. and Williamson, R. C. (2001). Estimating the support of a high-dimensional distribution, *Neural Computation*, Vol. 13, pp. 1443–1471.

Sebestyen, G., Hangan, A., Czako, Z. and Kovacs, G. (2018). A taxonomy and platform for anomaly detection, *2018 IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR)*, IEEE.
URL: *https ://doi.org/10.1109/aqtr.2018.8402710*

Senin, P., Lin, J., Wang, X., Dudek, G., Napoleon, T. and Verbesselt, J. (2014). Grammarviz 2.0 : A tool for grammar-based pattern discovery in time series, *Proceedings of the 2014 IEEE International Conference on Data Mining (ICDM)*, pp. 1187–1192.

Shumway, R. H. and Stoffer, D. S. (2017). *Time Series Analysis and Its Applications : With R Examples*, Springer.

Silva-Ramírez, E.-L., Pino-Mejías, R., López-Coello, M. and Cubiles-de-la Vega, M.-D. (2011). Missing value imputation on missing completely at random data using

multilayer perceptrons, *Neural Networks* **24**(1) : 121–129.
URL: *http ://dx.doi.org/10.1016/j.neunet.2010.09.008*

Sokolova, M., Japkowicz, N. and Szpakowicz, S. (2006). *Beyond Accuracy, F-Score and ROC : A Family of Discriminant Measures for Performance Evaluation*, Springer Berlin Heidelberg, p. 1015–1021.
URL: *http ://dx.doi.org/10.1007/11941439*$_1$14

Stojanovic, L., Dinic, M., Stojanovic, N. and Stojadinovic, A. (2016). Big-data-driven anomaly detection in industry (4.0) : An approach and a case study, *2016 IEEE International Conference on Big Data (Big Data)*, IEEE.
URL: *https ://doi.org/10.1109/bigdata.2016.7840777*

Su, X. and Tsai, C.-L. (2011). Outlier detection, *Journal of the American Statistical Association* **106**(494) : 803–815.

Su, Y., Zhao, Y., Sun, M., Zhang, S., Wen, X., Zhang, Y., Liu, X., Tang, J., Wu, W. and Liu, D. (2021). Detecting outlier machine instances through gaussian mixture variational autoencoder with one dimensional cnn, *Journal of Artificial Intelligence Research* **70** : 123–145.

Sunitha, G., Kiran, J. S., Nagendra, K., Afreen, S. S., Madhavi, K. R., Kothapati, N., Kumar, V. N. and Hemachandu, D. (2024). Adaptive trajectory data stream clustering, *Proceedings of Fifth International Conference on Computer and Communication Technologies*, pp. 243–252.

Tang, J., Chen, Z. and Fu, A. W.-C. (2002). Enhancing effectiveness of outlier detections for low density patterns, *Proceedings of the Sixth International Conference on Data Warehousing and Knowledge Discovery*, pp. 535–546.

Thaipisutikul, T. and Chen, Y.-N. (2024). An improved deep sequential model for context-aware poi recommendation, *Multimedia Tools and Applications* **83**(1) : 1643–1668.

Thornton, C., Hutter, F., Hoos, H. H. and Leyton-Brown, K. (2012). Auto-weka : Combined selection and hyperparameter optimization of classification algorithms.
URL: *https ://arxiv.org/abs/1208.3719*

Thudumu, S., Branch, P., Jin, J. and Singh, J. J. (2020). A comprehensive survey of anomaly detection techniques for high dimensional big data, *Journal of Big Data*

7(1).
URL: *https ://doi.org/10.1186/s40537-020-00320-x*

Truong, A., Walters, A., Goodsitt, J., Hines, K., Bruss, C. B. and Farivar, R. (2019). Towards automated machine learning : Evaluation and comparison of automl approaches and tools.
URL: *https ://arxiv.org/abs/1908.05557*

VASILE, M.-A., POP, F., NIŢĂ, M.-C. and CRISTEA, V. (2018). Mlbox : Machine learning box for asymptotic scheduling, *Information Sciences* **433–434** : 401–416.
URL: *http ://dx.doi.org/10.1016/j.ins.2017.01.005*

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, and Polosukhin, I. (2017). Attention is all you need, *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS)*, pp. 6000–6010.

Venturini, L., Baralis, E. and Garza, P. (2017). Scaling associative classification for very large datasets, *Journal of Big Data* **4**(1).
URL: *http ://dx.doi.org/10.1186/s40537-017-0107-2*

Wang, Z. and Wang, S. (2023). Online automated machine learning for class imbalanced data streams, *2023 International Joint Conference on Neural Networks (IJCNN)*, IEEE, pp. 1–8.

Wu, Z., Pan, S., Long, G., Jiang, J., Chang, X. and Zhang, C. (2020). A comprehensive survey on graph neural networks, *IEEE Transactions on Neural Networks and Learning Systems* **32**(1) : 4–24.

Xia, D., Jiang, S., Li, Y., Yang, N., Hu, Y., Li, Y. and Li, H. (2023). An asm-cf model for anomalous trajectory detection with mobile trajectory big data, *Journal of Big Data* **10**(1) : 165–178.

Yan, Y., Cao, L., Kulhman, C. and Rundensteiner, E. (2017). Distributed local outlier detection in big data, *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM.
URL: *https ://doi.org/10.1145/3097983.3098179*

Yang, J., Rahardja, S. and Fränti, P. (2019). Outlier detection, *Proceedings of the International Conference on Artificial Intelligence, Information Processing and Cloud*

Computing, ACM.
URL: *https ://doi.org/10.1145/3371425.3371427*

Yang, J. and Yue, Z. (2023). Learning hierarchical spatial-temporal graph representations for robust multivariate industrial anomaly detection, *IEEE Transactions on Industrial Informatics* **19**(6) : 7624–7635.
URL: *http ://dx.doi.org/10.1109/TII.2022.3216006*

Ying, Y., Duan, J., Wang, C., Wang, Y., Huang, C. and Xu, B. (2020). Automated model selection for time-series anomaly detection.
URL: *https ://arxiv.org/abs/2009.04395*

Zhang, S.-L., Zhang, B., Zhou, Y.-T., Guo, Y.-X. and Tan, J.-L. (2023). A two-stage anomaly detection method based on user preference features and the deep fusion model, *Applied Sciences* **13**(10) : 6217.
URL: *https ://doi.org/10.3390/app13106217*

Zhang, Y., Hamm, N. A. S., Meratnia, N., Stein, A., Van de Voort, M. and Havinga, P. (2012). Statistics-based outlier detection for wireless sensor networks, *International Journal of Geographical Information Science* **26**(4) : 507–533.

Zhang, Y., Meratnia, N. and Havinga, P. J. M. (2010). Outlier detection techniques for wireless sensor networks : A survey, *IEEE Communications Surveys and Tutorials* **12**(2) : 159–170.

Zhao, Z., Mehrotra, K. G. and Mohan, C. K. (2015). *Ensemble Algorithms for Unsupervised Anomaly Detection*, Springer International Publishing, p. 514–525.
URL: *http ://dx.doi.org/10.1007/978-3-319-19066-2*$_5$0

Zhu, R., Ji, X., Yu, D., Tan, Z., Zhao, L., Li, J. and Xia, X. (2020). Knn-based approximate outlier detection algorithm over iot streaming data, *IEEE Access* **8** : 42749–42759.

APPENDIX A MAIN CODE

```python
def apply_automl(dataset_path, true_label_column=None):
    print(true_label_column)
    # Load the dataset
    if dataset_path.endswith('.arff'):
        with open(dataset_path, 'r') as f:
            arff_data = arff.load(f)
        data = pd.DataFrame(arff_data['data'])
        data.columns = [attr[0] for attr in arff_data['attributes']]
    else:
        delimiter = detect_delimiter(dataset_path)
        data = pd.read_csv(dataset_path, delimiter=delimiter)

    print(data.head())
    # Identify the data type uni/bi
    data_type = detect_data_type(data)
    print(f'Data Type: {data_type}')

    # Identify the dataset type
    dataset_type = check_data_type_combined(data)
    print(f'Dataset Type: {dataset_type}')

    # Categorize columns
    column_types = categorize_columns(data)
    print('Column Types:', column_types)

    # Clean the data
    cleaned_data = clean_data(data, column_types)
    print('Data Cleaning Completed')
    print(cleaned_data.head())

    # Prepare columns for outlier detection
    prepared_data = prepare_columns_for_outlier_detection(cleaned_data, column_types, true_label_column)
    print('Data Preparation for Outlier Detection Completed')

    # Select features for outlier detection (optional, can be modified based on the dataset)
    selected_features = select_features_unsupervised(prepared_data, column_types, k=10,
                                                     true_label_column=true_label_column)
    print('Selected Features for Outlier Detection:', selected_features)

    # Apply forecasting methods for outlier detection
    outlier_flags, best_method, best_f1_score = apply_forecasting_methods(prepared_data[selected_features], dataset_type, data_type,
                                                                          true_label_column=true_label_column)
```

Figure APPENDIX A.1

APPENDIX B RESULTS

	A	B	C	D
1	**Dataset Name**	**F1 Score**	**Best Method**	**Total Execution Time**
2	abalone_C1_P02_V01_CA0	0,573529412	XGBoostOutlierDetection	118,2280447
3	abalone_ori_C1_P02_V01_CA0	0,401384083	EllipticEnvelope	160,3645661
4	ALOI	0.4540918163672655	RPCA	1924,374793
5	Annthyroid_02_v01	0,961832061	ExtraTrees	352,9895041
6	AQI By State 1980-2022			0,244818926
7	artificial_intelligence_anomaly_detection	0.9687520798668885	XGBoostOutlierDetection	1539,499834
8	ARTime_art_daily_jumpsdown	1	XGBoostOutlierDetection	116,369417
9	ARTime_exchange-4_cpm_results	0,436268068	Prophet	18,04777908
10	ARTime_iio_us-east-1_i-a2eb1cd9_NetworkIn	0,4536862	Prophet	16,96298385
11	ARTime_rds_cpu_utilization_cc0c53	0,581818182	Prophet	47,77386594
12	ARTime_TravelTime_451	0,493061224	Prophet	19,44843793
13	australian_C1_P02_V01_CA0	0,666666667	XGBoostOutlierDetection	21,43534589
14	auto2_C1_P02_V01_CA0	0,281879195	EVA	223,685266
15	auto4_C1_P02_V01_CA0	0,545454545	XGBoostOutlierDetection	61,58178306
16	auto6_1_C1_P02_V01_CA0	0,8	XGBoostOutlierDetection	14,62270117
17	auto6_2_C1_P02_V01_CA0	0,846153846	XGBoostOutlierDetection	18,75148797
18	auto6_3_C1_P02_V01_CA0	0,571428571	XGBoostOutlierDetection	18,60628986
19	auto7_1_C1_P02_V01_CA0	0,45	XGBoostOutlierDetection	37,76479793
20	auto7_2_C1_P02_V01_CA0	0,714285714	XGBoostOutlierDetection	11,52172089
21	auto7_3_C1_P02_V01_CA0	0,4	XGBoostOutlierDetection	22,62826014
22	auto8_C1_P02_V01_CA0	0,875	XGBoostOutlierDetection	12,09934902
23	automobile_C1_P02_V01_CA0	0,571428571	AutoEncoder	15,50801706
24	banknotes			4,648022175
25	bayesChangePt_art_daily_jumpsdown	0,50557377	Prophet	29,90258098
26	bi_sample_submission			7,645292997
27	Books_Data_Clean			7,950005054
28	Cardiovascular_Disease_Dataset	0,975277067	XGBoostOutlierDetection	25,00591683
29	chronic_kidney_disease_C1_P02_V01_CA0	1	RPCA	3,333333969
30	chronic_kidney_disease_full_C1_P02_V01_CA0	0,8	DBSCAN	6,125782013
31	climate_C1_P02_V01_CA0	1	XGBoostOutlierDetection	11,91426516
32	connectionist_vowel_C1_P02_V01_CA0	0,971428571	VAE	22,47034812
33	contextOSE_art_daily_jumpsdown	0,50557377	Prophet	28,64574623
34	contraceptive_C1_P02_V01_CA0	0,866666667	ExtraTrees	2,321848869
35	cpu4	0,643067847	AutoEncoder	149,0029869
36	credit_C1_P02_V01_CA0	0,833333333	XGBoostOutlierDetection	8,472559214
37	crx_C1_P02_V01_CA0	0,6	XGBoostOutlierDetection	20,51275492
38	CV_LatLon_21Jan_12Mar			0,26030302
39	data_20	0,893353941	XGBoostOutlierDetection	249,595768
40	earthgeckoSkyline_art_load_balancer_spikes	0,628947368	Prophet	30,80484009
41	Exp1			3,299131155
42	first_C1_P02_V01_CA0	0,590163934	ExtraTrees	109,3846698
43	Glass_withoutdupl_norm	1	XGBoostOutlierDetection	4,982289076
44	heart_cleverland_C1_P02_V01_CA0	0,4	OneClassSVM	14,0401659
45	heart_va_C1_P02_V01_CA0	0,5	ExtraTrees	11,61471701
46	horse_colic_lesion_C1_P02_V01_CA0	0,279069767	OneClassSVM	11,98063111
47	horse_colic_outcome_C1_P02_V01_CA0	0,5	VAE	12,56703281
48	htmjava_exchange-2_cpm_results	0,58780037	AutoEncoder	20,77607989
49	ibb_wifi_location (1)			0,361385345
50	image_C1_P02_V01_CA0	0,987341772	XGBoostOutlierDetection	56,52712488

Figure APPENDIX B.1

	A	B	C	D
1	Dataset Name	F1 Score	Best Method	Total Execution
53	KDDCup99_10fn	0,972303638	AutoEncoder	1761,235446
54	knncad_nyc_taxi	0,49516441	VAE	73,28734064
55	leaf_C1_P02_V01_CA0	0,909090909	XGBoostOutlierDetection	7,746226072
56	Lymphography_original	0,909090909	RPCA	3,628893137
57	mechanical_C1_P02_V01_CA0.arff	0,714834286	VAE	28,59423637
58	mice_C1_P02_V01_CA0	0,827586207	XGBoostOutlierDetection	20,82593918
59	other-Lyft_B02510			398,4527869
60	Parkinson_withoutdupl_norm_75	1	XGBoostOutlierDetection	4,725628138
61	PenDigits_withoutdupl_norm_v01	1	XGBoostOutlierDetection	513,9844692
62	Pioneer_Dumbleton_joined			71,08614302
63	power_usage_2016_to_2020			89,85613966
64	random_art_flatline	0,780916031	AutoEncoder	0,624727249
65	randomCutForest_art_daily_nojump	0,596685083	Prophet	31,80836391
66	redwinequality			12,37797999
67	relativeEntropy_ec2_cpu_utilization_77c1ca	0,402095286	AutoEncoder	35,66502023
68	relativeEntropy_Twitter_volume_CVS	0,42845787	Prophet	103,9824762
69	saheart_C1_P02_V01_CA0	1	XGBoostOutlierDetection	7,806924105
70	sandy_ck_sorbellos_road_joined			41,085917
71	seismic_C1_P02_V01_CA0	0,5	XGBoostOutlierDetection	19,15601587
72	Shuttle_withoutdupl_v01	1	XGBoostOutlierDetection	26,32604909
73	SKAB_other	0,605617978	AutoEncoder	28,38543892
74	SKAB_Valve	0,507668712	AutoEncoder	19,66827798
75	SKAB_Valve_2	0,501163693	AutoEncoder	20,1389091
76	spambase_C1_P02_V01_CA0	0,727272727	XGBoostOutlierDetection	111,61413
77	spambase_C1_P02_V01_CA0	0,727272727	XGBoostOutlierDetection	115,495007
78	statlog_australian_C1_P02_V01_CA0	0,666666667	XGBoostOutlierDetection	22,84990883
79	statlog_german_C1_P02_V01_CA0	0,307692308	EllipticEnvelope	20,44089484
80	statlog_heart_C1_P02_V01_CA0	0,4	AutoEncoder	18,33036399
81	tae_C1_P02_V01_CA0	0,727272727	DBSCAN	18,59094257
82	thyroid_allrep_C1_P02_V01_CA0	0,777272727	VAE	8,709343529
83	thyroid_allbp_C1_P02_V01_CA0	0,666666667	KNN	18,06957579
84	thyroid_allhypo_C1_P02_V01_CA0	0,909090909	XGBoostOutlierDetection	6,426530123
85	thyroid_ann_C1_P02_V01_CA0	1	XGBoostOutlierDetection	14,63342714
86	thyroid_hypothyroid_C1_P02_V01_CA0	0,851851852	XGBoostOutlierDetection	2,523215055
87	thyroid_sick_C1_P02_V01_CA0	0,4	XGBoostOutlierDetection	8,575750828
88	Tully_River_Tully_Gorge_National_Park_joined			57,89497375
89	Waveform_withoutdupl_v01	0,635135135	XGBoostOutlierDetection	94,1161499
90	WDBC_withoutdupl_v01	0,947368421	XGBoostOutlierDetection	0,421414137
91	weight_C1_P02_V01_CA0	0,945454545	ARIMA	0,467629194
92	weight-height			29,22187805
93	Wilt_02_v01	0,961325967	XGBoostOutlierDetection	5,913923025
94	wine_quality_C1_P02_V01_CA0	0,38585209	KDE	104,2038629
95	winequality-white			34,61554623
96	skyline_ec2_request_latency_system_failure.csv	0,677790055	Prophet	98,60620079
97	Participants_data_wh18			45,9744381
98	HeartDisease_withoutdupl_02_v01	0,833973333	VAE	63,18251781
99	InternetAds_withoutdupl_norm_02_v01	0,833720929	OPTICS	68,12505341
100	ibb_wifi_subscriber (1)			47,20386715
101	sciencehub-entry-for-rvm-utah-olson-et-al-2021-xlsx-1			48,33338184

Figure APPENDIX B.2

BIOGRAPHICAL SKETCH

EDUCATIONAL BACKGROUND:

— **Bachelor's Degree**: 2020, Yıldız Technical University, Faculty of Mechanical Engineering, Industrial Engineering

— **Master's Degree**: 2024, Galatasaray University, Graduate School of Science and Engineering, Intelligent Systems Engineering Thesis Master's Program

PROFESSIONAL EXPERIENCE AND AWARDS:

— **2020-2021**: Enqura Information Technologies, Business Intelligence Expert

— **2021-2022**: Huawei, Big Data Engineer

— **2022-2023**: Getir, Data Analyst

— **2023-2024**: Trendyol, Data Analyst

PUBLICATIONS

— International Graduate Research Symposium 2024