JOURNAL OF SOFT COMPUTING AND DATA MINING
e-ISSN: 2716-621X

JSCDM

Vol. 5 No. 2 (2024) 1-14
https://publisher.uthm.edu.my/ojs/index.php/jscdm

An Efficient MCD-OSVM Model for Outlier Detection in IoT-Based Smart Energy Management Systems

Parh Yong Wong[1], Nayef Abdulwahab Mohammed Alduais[1]*, Hairulnizam Bin Mahdin[1], Abdul-Malik H. Y. Saad[2], Antar Shaddad Hamed Abdul-Qawy[3], Abdullah B Nasser[4] and Waheed Ali H.M.Ghanem[5]

[1] *Faculty of Computer Science and Information Technology (FSKTM), Universiti Tun Hussein Onn Malaysia (UTHM), Parit Raja, Batu Pahat, Johor 86400, MALAYSIA*
[2] *Faculty of Engineering, University of Buraimi, Al Buraimi 512, OMAN*
[3] *Department of Mathematics and Computer Science, Faculty of Science, SUMAIT University, Zanzibar 1933, TANZANIA*
[4] *School of Technology and Innovation, University of Vaasa, 65200 Vaasa, FINLAND*
[5] *Faculty of Computer Science and Mathematics, Universiti Malaysia Terengganu, Kuala Terengganu, Terengganu 21030, MALAYSIA*

*Corresponding Author: nayef@uthm.edu.my
DOI: https://doi.org/10.30880/jscdm.2024.05.02.001

Article Info

Received: 28 May 2024
Accepted: 6 December 2024
Available online: 18 December 2024

Keywords

Internet of Things (IoT), smart energy management system, outlier detection techniques, Minimum Covariance Determinant (MCD), One-class Support Vector Machine (OSVM).

Abstract

As Information, Communication, and Sensor Technologies (ICST) continue to evolve, data-driven innovations like the Internet of Things (IoT) and Smart Technologies, including Smart Energy Management Systems (SEMS), have become increasingly prevalent worldwide. Ensuring data quality is crucial for the effective implementation of IoT-based SEMS, as poor data management in these critical systems can significantly impact the quality of life for millions and potentially lead to severe disruptions and damage at a national level. In this research, an efficient One-class Support Vector Machine (OSVM) model is developed by deploying the Minimum Covariance Determinant (MCD) model at the data pre-processing phase to clean the training data This allow a better trained OSVM model that can be used for the outlier detection. The comparison between the efficient MCD-OSVM model and the base OSVM model, both based on the same original model, highlights a key difference in the training phase: the proposed model was trained with cleaned data using the MCD method, while the base OSVM model used the original, uncleaned data. Cleaning the dataset with an efficient method such as MCD improves the accuracy of OSVM model, an increase of 13.21% in average accuracy, while only increase the operation time 9.5 seconds, although the overall operation time can be further reduced as it is also found a cleaner training dataset will indirectly improve the execution time of OSVM models by allowing it to run on a lower NU parameter value.

1. Introduction

Thanks to ongoing advancements in communication and information technology, the once Sci-Fi concept of Smart Technologies has become a reality in recent years. The Smart Energy Management System (SEMS) exemplifies the seamless integration of traditional manual systems with cutting-edge technologies, designed to enhance human life. SEMS is one of the first intelligent applications to gain significant attention and be fully realized. When combined with Internet-of-Things (IoT) technology, modern SEMS can perform a wide range of functions under optimal conditions, including managing and controlling electrical appliances, automating energy distribution through grids and power stations, managing renewable energy sources, intelligent metering, and outsourcing energy. However, it's crucial to recognize that data-driven fields like IoT-based SEMS are vulnerable to data quality issues, which can severely impact their effectiveness [1].

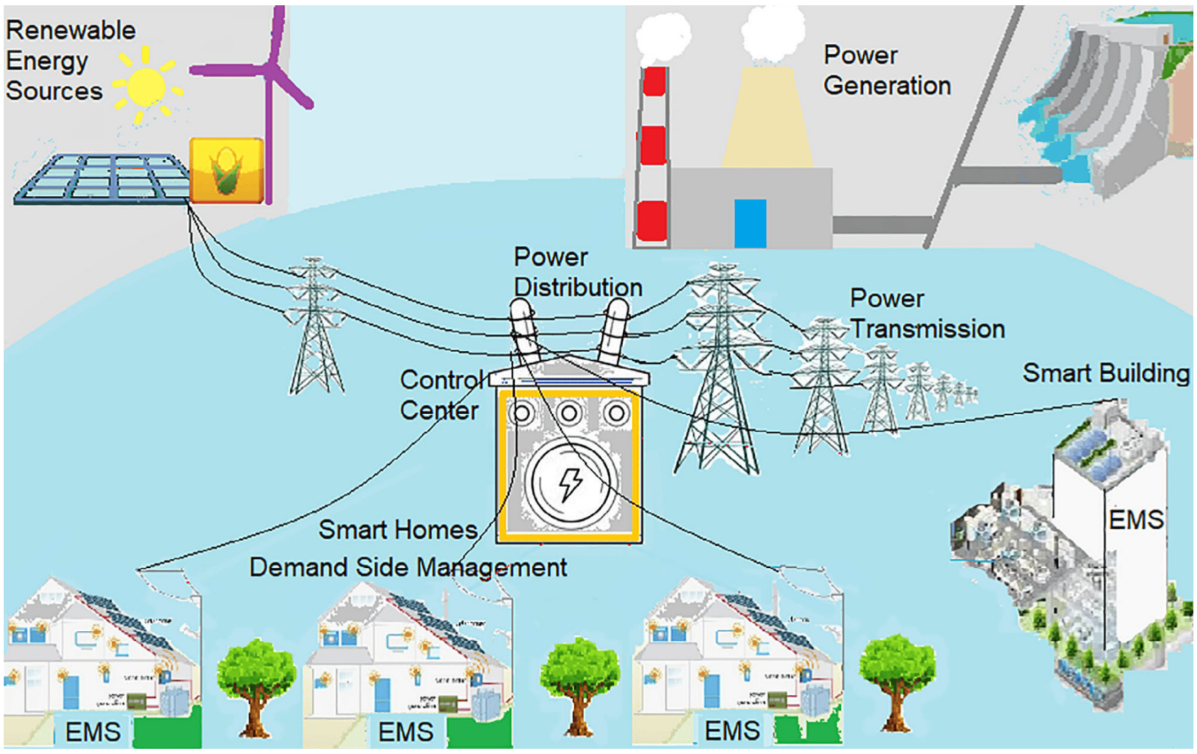


Fig 1 *Concept of IoT-based smart energy management systems [2]*

Figure 1 above illustrates the concept of an IoT-based Smart Energy Management System, managed mainly with a Smart Grid System. Each of these applications relies on data from various sources, primarily sensor-collected data, for full functionality. Therefore, data quality is crucial for ensuring these systems operate efficiently and effectively. Data quality refers to how well data meets the needs of a system or user, fulfilling their requirements for different purposes [3]. In a modern data-driven environment, the importance of data quality is encapsulated in the concept of "Garbage In, Garbage Out." This concept highlights that when AI technologies are trained on poor-quality data, they produce flawed AI models with inaccurate decision-making capabilities [4]. While this idea is often associated with AI, it equally applies to IoT-enabled SEMS. Poor data quality in these systems can lead to either excessive or insufficient energy distribution within electrical grids, potentially causing severe issues like widespread blackouts and damaged electrical appliances. Therefore, identifying and addressing data quality problems is essential when working with IoT-based SEMS.

A method for addressing data quality issues like outliers is to utilize Machine Learning (ML)-based models to rapidly analyze and detect anomalies in datasets [5]. This research runs a detailed analysis on a hybrid ML model proposed by the same researchers behind this research, where both Minimum Covariance Determinant (MCD) and One-class Support Vector Machine (OSVM) are used in outlier detection. Since OSVM relies on clean data for training, MCD is deployed during the data cleaning process, which should perform way better than manual and normal statistical methods for cleaning the absurd and out-of-range data in the training dataset. With the clean dataset obtained, the training data will be used for the OSVM model's training, and once the training is done, for the purpose of proving the effectiveness of the proposed model, a baseline model trained with a manually cleaned dataset will be deployed, too. Both the models will then be used for outlier detection with a manually injected dataset under the same settings.

The remaining sections of this paper are organized as follows: Section 2 provides a Literature Review, covering various aspects of the paper and highlighting related works. Section 3 outlines the Methodology, detailing the analysis process, datasets, and procedures used. Section 4 focuses on the Implementation, offering key information on how the Methodology was applied in this research. Section 5 presents the Results and Discussion, where the testing outcomes are thoroughly analysed and discussed. Finally, Section 6 offers a conclusion to the analysis and addresses the limitations of this paper.

2. Literature Review

This section provides an overview of the machine learning models used, along with references to various sources. Additionally, it discusses related works, highlighting the research gaps addressed in this analysis.

2.1 Minimum Covariance Determinant (MCD)

Highly robust and resistant to outlying observation, Minimum Covariance Determinant or MCD is one of the first affine and equivariant estimators for datasets with multivariate location and scattering points.

Figure 2 presents a sample scatter plot of MCD, a technique that has been crucial for outlier detection since its inception in 1984. However, its popularity significantly increased after Rousseeuw and Van Driessen introduced the FAST-MCD algorithm in 1999, which made its computation more efficient. MCD becomes especially useful when working with higher-dimensional datasets or those containing multiple variables, making visualization more challenging. According to research by Hubert, M., & Debruyne, M., the theorem for the computationally efficient FAST-MCD is as follows [6]:

Matrices $\{x_1,\ldots,x_n\}$ and let $H_1 \subset \{1,\ldots,n\}$ be an h-subset, that is $|H1| = h$. Put $\hat{\mu}_1$ and $\hat{\sum}_1$ the empirical mean and covariance matrices of the data in H_1. If $\det(\hat{\sum}_1) \neq 0$ define the relative distances:

$$d_1(i) := \sqrt{(x_i - \hat{\mu}_1)^t \hat{\sum}_1^{-1} (x_i - \hat{\mu}_1)} \qquad for\ i = 1,\ldots,n \tag{1}$$

Now take H2 such that

$$\{d_1(i); i \in H_2\} := \{(d_1)_{1:n},\ldots,(d_1)_{h:n}\}\ where\ (d_1)_{1:n} \leq (d_1)_{2:n} \leq \cdots \leq (d_1)_{h:n} \tag{2}$$

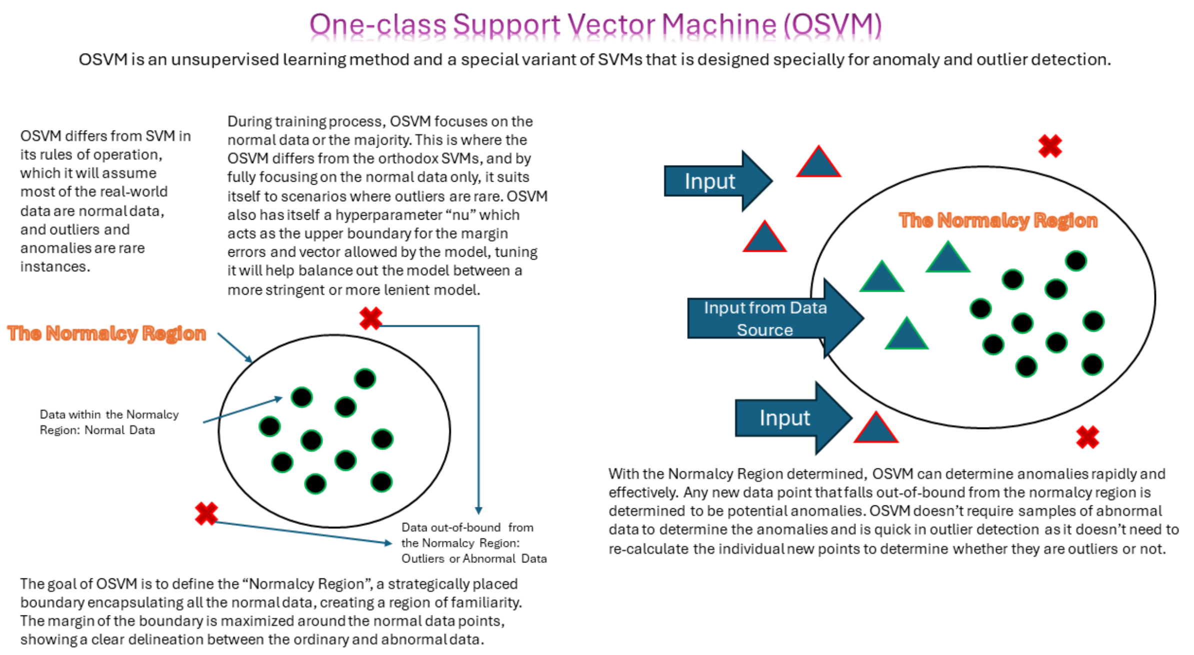
These are the ordered distances, and compute $\hat{\mu}_2$ and $\hat{\sum}_2$ based on H_2. Then

$$det\ (\hat{\sum}_2) \leq det\ ((\hat{\sum}_1)\ with\ equality\ if\ and\ only\ if\ \hat{\mu}_2 = \hat{\mu}_1\ and\ \hat{\sum}_2 = \hat{\sum}_1 \tag{3}$$

MCD, also known as robust Mahalanobis Distance (MD), is more effective in handling cellwise outliers—outliers in individual cells of a multivariate dataset that do not affect other cells, which may still contain valuable data [7]. Compared to MD and other statistical outlier detection methods, the standard MCD is computationally intensive. Therefore, the development of FAST-MCD offers a more efficient alternative for use in programming environments [8]. Like MD, FAST-MCD operates by first calculating a pattern from the input and output of the training datasets, which is then applied to the testing datasets to identify outliers.

2.2 One-class Support Vector Machine (OSVM)

The One-Class Support Vector Machine (OSVM or OCSVM), an unsupervised learning method introduced by Scholkopf, is a specialized variant of Support Vector Machines (SVM) designed for one-class classification tasks. This makes it particularly effective for detecting anomalies and outliers. The method works by training exclusively on normal data to establish a boundary, often referred to as the "Normalcy Region." After training, any new data point that falls outside this boundary is classified as an outlier or abnormal data point [9-11].


One-class Support Vector Machine (OSVM)

OSVM is an unsupervised learning method and a special variant of SVMs that is designed specially for anomaly and outlier detection.

OSVM differs from SVM in its rules of operation, which it will assume most of the real-world data are normal data, and outliers and anomalies are rare instances.

During training process, OSVM focuses on the normal data or the majority. This is where the OSVM differs from the orthodox SVMs, and by fully focusing on the normal data only, it suits itself to scenarios where outliers are rare. OSVM also has itself a hyperparameter "nu" which acts as the upper boundary for the margin errors and vector allowed by the model, tuning it will help balance out the model between a more stringent or more lenient model.

The goal of OSVM is to define the "Normalcy Region", a strategically placed boundary encapsulating all the normal data, creating a region of familiarity. The margin of the boundary is maximized around the normal data points, showing a clear delineation between the ordinary and abnormal data.

With the Normalcy Region determined, OSVM can determine anomalies rapidly and effectively. Any new data point that falls out-of-bound from the normalcy region is determined to be potential anomalies. OSVM doesn't require samples of abnormal data to determine the anomalies and is quick in outlier detection as it doesn't need to re-calculate the individual new points to determine whether they are outliers or not.

The Normalcy Region
Data within the Normalcy Region: Normal Data
Data out-of-bound from the Normalcy Region: Outliers or Abnormal Data
Input
The Normalcy Region
Input from Data Source
Input

Fig. 2 *OSVM model*

The OSVM algorithm is designed to address one-class classification problems by distinguishing target instances (normal data) from outliers or anomalies. Its core principle is to create a hyperplane ⟨w, z⟩ − ρ in a feature space F, determined by a kernel function. This is achieved by mapping the training instances into F and positioning the hyperplane at the maximum margin from the origin. In this context, \(w\) represents the normal vector of the hyperplane, while \(ρ\) is its bias, indicating the distance between the hyperplane and the origin.

To separate the data from the origin, the authors propose to solve the following optimization problem:

$$\begin{gathered}\min_{w,\rho,\zeta}\frac{1}{2}\|w\|^2+\frac{1}{vN}\sum_i \zeta i-p\\ \text{s. t}\langle w,\phi(xi)\rangle\rho-\zeta i,\forall i\\ \text{and}\zeta i\geq 0,\forall i,(1)\end{gathered} \tag{4}$$

With ν ∈ (0,1]. The ζi are slack variables that are used to model the separation errors. ϕ(.) is a non-linear projection of the data into a high-dimensional space. It is evaluated through a kernel function κ by the dot product κ(xi,xj)=⟨ϕ(xi),ϕ(xj)⟩. To ensure separation of the target data from the origin, Scholkopf et al. used OSVM only with the Gaussian RBF kernel as the data, in this case, are projected in the surface of a half of a unit radius hypersphere that is centered on the origin of the projection space. However, OSVM itself comes with several different kernels suited for different purposes; thus, choosing the suitable one is an important step in ensuring an accurate OSVM model [9, 12, 13].

With ν ∈ (0,1]. The ζi are slack variables used to account for separation errors. ϕ(.) represents a non-linear mapping of the data into a higher-dimensional space, and it is evaluated through a kernel function κ by the dot product κ(xi,xj)=⟨ϕ(xi),ϕ(xj)⟩. To ensure the separation of target data from the origin, Scholkopf et al. applied OSVM with the Gaussian RBF kernel, projecting the data onto the surface of a unit-radius hypersphere centred at the origin in the projection space. However, OSVM offers multiple kernels suited for different tasks, making the selection of an appropriate kernel essential for building an accurate model [9], [12], [13].

The main strength of OSVM lies in its focused approach, where any data point falling outside the defined Normalcy Region is flagged as an outlier. This allows OSVM to be trained exclusively on a clean dataset, free of abnormal data. Additionally, the identification process is efficient and faster than many other outlier detection models, as it does not require recalculating and comparing new data points with existing ones—an often computationally expensive task for large datasets. However, this also increases the risk of false positives, and OSVM is highly sensitive to outliers during the training phase. To mitigate these drawbacks and enhance its performance, careful tuning of the model's configuration is necessary, often involving trial and error. It is also vital to use a thoroughly cleaned dataset during training to achieve optimal results [9], [13].

2.3 Related Works

The issue of data quality in IoT-based smart environments has not been overlooked, as numerous researchers have conducted studies and proposed various methods for outlier detection in IoT settings. Table 1 below provides a summary of key details from related research. These studies employ different AI and ML-based outlier detection models within IoT-enabled smart environments or Smart Environmental Monitoring Systems (SEMSs). These works serve as important references and guidance for this analysis, particularly when selecting appropriate models and conducting testing.

Table 1 *Summary of related works/research*

REF.	APPLICATIONS	OUTLIER DETECTION METHOD	DATASET(S) USED	Methods Used for Training Data Cleaning
[14]	Anomaly Detection in Power Consumption Data in Smart Buildings	One-class Support Vector named UAD-OCSVM and SL-based Anomaly Detection based on Micromoments (SAD-M2)	Real-world Collected Dataset from Qatar University's Energy Lab	None, synthetic data is used.
[15]	Anomaly Pattern Detection for Energy Theft Detection in Advanced Metering Infrastructure	Support Vector Machine (SVM)	CER Smart Metering Project	Manual Cleaning
[16]	Anomaly Detection in Industrial IoT (IIoT) for Smart Environments	Graph Neural Networks (GNNs)	Many datasets are used in the original research, kindly refer to the original research for references	None, synthetic data is used.
[17]	Real-time Anomaly Detection in Smart Meter Data through Distributed Fog Computing Architecture and Multi-layered ML Methods	Linear Regression (LR), Support Vector Regression (SVR), Random Forest Regression (RFR), and Gradient Boosting Regression (GBR)	UCI Machine Learning Repository.	Manual Cleaning
[18]	Multi-attributes Monitoring and Anomaly Detection for Power-limited IoT devices through Deep Reinforcement Learning (DRL) algorithm and Unsupervised Learning (UL) reward	Deep Reinforcement Learning (DRL) and UL Reward	Real-time Collected Datasets by the Authors	None
[19]	DDoS Detection through Deployment of Outlier Exposure (OE)-based	Outlier Exposure-Based Cross-Silo	Simulated DDoS Attacks by the Authors as Real-time	None

	Cross Sile Federated Learning at the Edge	Federated Learning, FedOE	Training and Testing Data	
[20]	Anomaly Detection for IoT Devices with Mean-shift & Local Outlier Factor (LOF)-based Ensemble ML Technique	Mean-Shift & LOF-based Ensemble ML	UNSW-NB15 Network Dataset	Statistical Method
[21]	Outlier Detection for IoT-enabled Indoor Localization through Various ML methods	Isolation Forest (iForest), SVM, KNN, and Random Forest (RF)	UCI Open Repository Datasets	Manual & Principal Component Analysis (PCA)
[22]	Comparative Analysis of Several Outlier Detection Methods on Several IoT-based SEMS Sensor-collected Datasets	Local Outlier Factor (LOF), K-Nearest Neighbour with Mahalanobis Distance as Kernel (KNN(MD)), Minimum Covariance Determinant (MCD)	[23 - 25]	Statistical Method, Manual Cleaning

The most significant gap between the proposed model and these existing models is the datasets used and the data pre-processing phase. Most of the existing models, including most of the referred works either assumes the training datasets are clean by default, thus continued without any further data pre-processing; or only uses statistical methods such as Z-score and Standard Deviations in combination with manual analysis and cleaning. In this limited research, there is another research that does use data cleaning method other than statistical and manual cleaning methods is using Principal Component Analysis (PCA), which differs a lot from the method used for this proposed model, in functionality, specialties and workflow. The other difference that matters a lot is the datasets used for this proposed model, which are three different datasets from different data sources of NGO and Government-funded or operated research from different countries, which is a lot of difference from the more often used open repository from universities and real time data streams. The datasets used by this proposed research are more realistic and accurate-to-presentation as they are obtained from and used for different IoT-based SEMS systems which are still currently deployed and operational.

This research also serves as a continuation of the author's research on different Outlier Detection Methods including LOF, KNN(MD) and MCD on the 3 datasets used in this research too. The main differences between this research and the previous published research are the improved data pre-processing phase using one of the analyzed outlier detection methods, the MCD to clean the training datasets before it is used for the training of the Efficient MCD-OSVM Model.

3. Methodology

This section clarifies the Methodology used for this research. It includes the flow of conducting the analysis, the procedure for preparing and preprocessing the datasets, training and testing the supervised ML models, and steps for evaluating the datasets' performance.

3.1 Workflow of Proposed Efficient MCD-OSVM Model

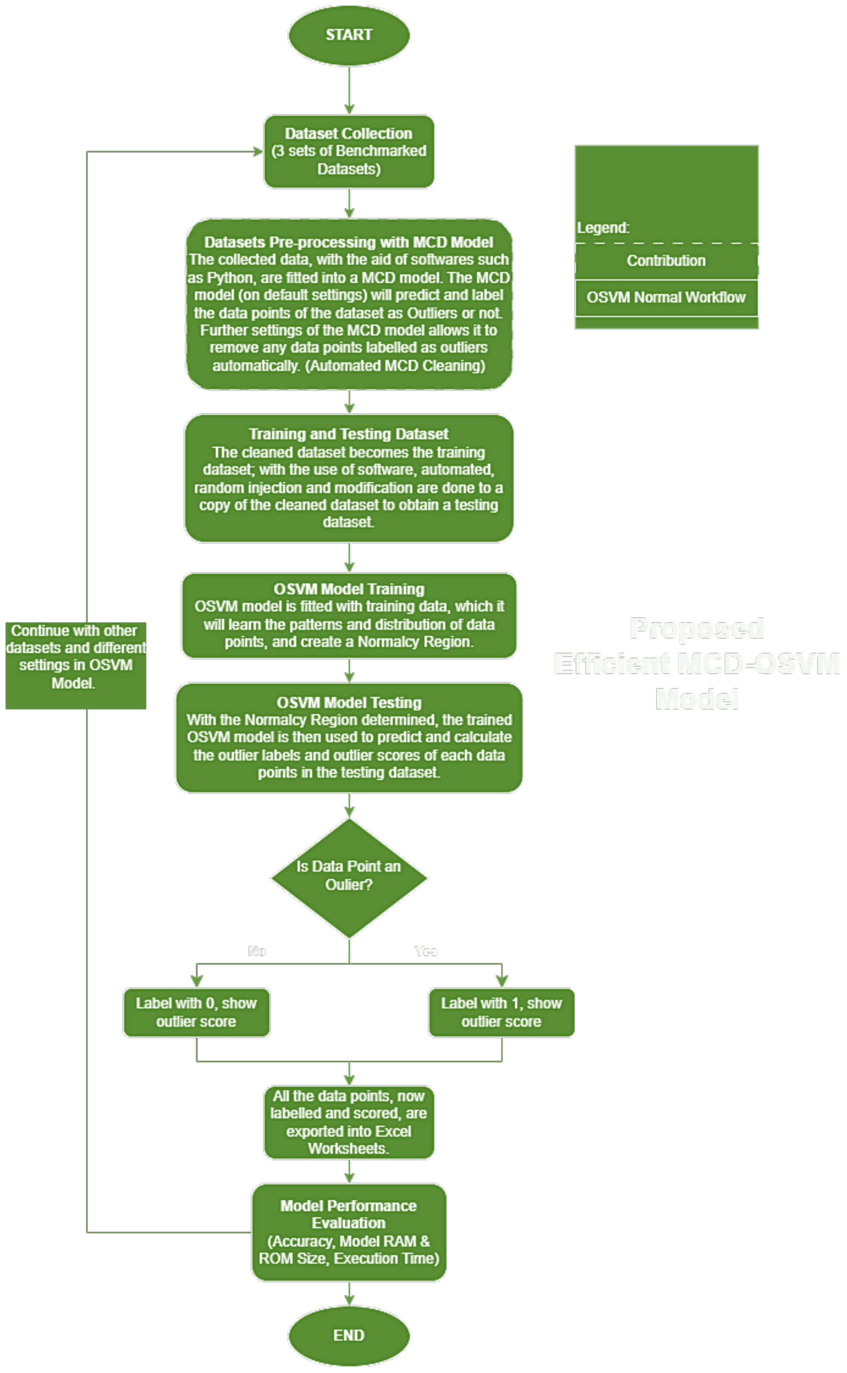

START
Dataset Collection (3 sets of Benchmarked Datasets)
Datasets Pre-processing with MCD Model The collected data, with the aid of softwares such as Python, are fitted into a MCD model. The MCD model (on default settings) will predict and label the data points of the dataset as Outliers or not. Further settings of the MCD model allows it to remove any data points labelled as outliers automatically. (Automated MCD Cleaning)
Training and Testing Dataset The cleaned dataset becomes the training dataset; with the use of software, automated, random injection and modification are done to a copy of the cleaned dataset to obtain a testing dataset.
OSVM Model Training OSVM model is fitted with training data, which it will learn the patterns and distribution of data points, and create a Normalcy Region.
OSVM Model Testing With the Normalcy Region determined, the trained OSVM model is then used to predict and calculate the outlier labels and outlier scores of each data points in the testing dataset.
Is Data Point an Oulier?
No
Yes
Label with 0, show outlier score
Label with 1, show outlier score
All the data points, now labelled and scored, are exported into Excel Worksheets.
Model Performance Evaluation (Accuracy, Model RAM & ROM Size, Execution Time)
END
Continue with other datasets and different settings in OSVM Model.
Legend:
Contribution
OSVM Normal Workflow
Proposed Efficient MCD-OSVM Model

Fig 3 *Workflow diagram of efficient MCD-OSVM model*

Figure 3 is the Proposed Efficient MCD-OSVM Model workflow in this research. As observed from the diagrams, the main difference in workflow in the baseline and proposed models comes from the data pre-processing process. The basic model requires manual cleaning after the statistical evaluation is done on the collected dataset. The manual cleaning includes removing data points containing extreme, out-of-range, empty, and invalid data values and outliers, all based on the statistical evaluation results. On the other hand, the Proposed Efficient MCD-OSVM Model uses the MCD model for data cleaning during the data pre-processing phase. The process of cleaning datasets using the PYOD MCD model is a simple 2-step process: fitting the dataset into the MCD model and letting the MCD model predict and label the possible outliers in the dataset. With some additional coding, the MCD model can remove the detected outliers fully and automatically.

As for the other steps, it is generally the same: collecting suitable datasets and proceed to data pre-processing; generating a copy of the cleaned datasets (which becomes the training dataset for the OSVM model), inject and modifying the data points through the use of software for random results (injected datasets become the testing dataset); fit the training dataset into the OSVM model to allow it to learn from training dataset and determine the Normalcy Region; feed the testing dataset once the model is trained and a Normalcy Region is obtained, the OSVM model will quickly and accurately predict and label whether the data points are outliers or not; the result is the testing dataset labeled and scored for outlier that is exported into an Excel Workbook.

Since this is a comparative analysis between the baseline and proposed model, it is also important to evaluate each model's performance. The main points of the models being evaluated include their average accuracy, the average total ROM and RAM usage, and, lastly, the average total execution time.

3.2 Dataset Requirements & Preprocessing

This section outlines the specific requirements for the datasets used, along with the preprocessing steps applied. An important consideration in selecting suitable datasets for research is ensuring their relevance to the research topic and objectives. Since this research focuses on sensor-collected datasets in IoT-based SEMS, the following dataset criteria have been established:

- The datasets are multivariate numerical datasets (containing various types of data) relevant to SEMS research across different use cases.
- The datasets must be collected via sensors.
- Any datasets sourced from the internet should be benchmarked or associated with published SEMS research.

Table 2 provides a summary of the key information about the datasets used, including the research titles or applications and their source citations. The Implementation section discusses further details on the datasets, though overall, they are deemed fully appropriate for the research purposes.

Table 2 *Basic summary of used datasets*

Dataset	Title	Source
Dataset 1	Data for Short-Term Prediction of CO2 Concentration Based on A Wireless Sensor Network	[23]
Dataset 2	QCAT Smart Office Environment	[24]
Dataset 3	ROBOD, Room-Level Occupancy and Building Operation Dataset	[25]

As stated before, the main difference between the basic OSVM model and the proposed MCD+OSVM model is observed in the data cleaning process. The baseline model uses manual cleaning based on statistical evaluation, while the proposed model uses the MCD model for the cleaning process.

There are five steps of pre-processing of the datasets to be done when manual cleaning is used. The first step is removing unwanted data types (non-numerical and unrelated) in each dataset, leaving only humidity, temperature, light intensity, CO2 count, and occupancy in numbers. Being the most common data types present in most of the existing smart energy management systems as essential, the relationship of humidity, temperature, and light intensity when taking light intensity as the primary data type is as follows: humidity is inversely proportional to light intensity. In contrast, temperature increases when light intensity increases. With this theorem established, for simplicity (more data types equal more dimensions equal to higher complexity) and accuracy in detecting outliers (more data types require more sensor types and amount, leading to higher error rate), this research only keeps these 5 data types. The second step is to remove the incomplete/missing data values, ensuring the process of calculation and evaluation during the outlier detection steps. After removing the empty and invalid data, the third step is to remove the absurdly huge or tiny values in the datasets. With these steps, a “clean” dataset, or a dataset without any data problems such as outliers and out-of-range data, can be obtained. The fourth step is to analyse the datasets statistically to ensure that the probability of any anomalies existing is reduced to the minimum. The statistics include the count (number of data points in each dataset), mean, standard deviation, min, first quartile (25%), second quartile (50%), third quartile (75%) and max of each data type in the datasets. After obtaining a clean dataset, manually injection of outliers into the datasets are done. This is the fifth and critical step in testing the accuracy of the data, where the SL models should be able to detect the outliers in the datasets. The percentage of outliers to the original data used is 1% (0.01) for each data type.

The process of cleaning the dataset is much easier and more efficient when the MCD model is used, as the process of determining the outliers and removing them is done automatically by the MCD model. Using the Python engine, any data point containing invalid and empty values and any data column not considered in outlier detection is removed. The resulting dataset is then fitted into the MCD model. the model is then prompted to predict and label the data points as outliers or not. Once the labeling is done, based on the labels, data points deemed as outliers are removed, and the output is saved.

The resulting outputs From both models are the statistical evaluation of the uncleaned and cleaned datasets for viewing the properties. The results are shown in the tables below:

Table 3 *Statistical evaluation of the uncleaned datasets used by the OSVM model*

	count	mean	std	min	25%	50%	75%	max
				CO2 Datasets (Clean)				
TemperatureCelcius	23369	34.53259018	2.617976325	26.7	33.1	34.8	35.6	50
HumidityPercent	23369	50.36288502	7.208279539	32.4	46.9	50.7	53.1	65.4
LightIntensityLux	23369	26.56620737	9.807386189	2	19	27	32	412
CO2PPM	23369	161.0327357	109.5104872	2	82	166	215	1141
				QCAT Datasets (Clean)				
TemperatureCelcius	25840	25.22397059	2.243481794	21.91	23.65	24.76	25.99	35.72
HumidityPercent	25840	62.26200515	8.239760582	38.048	56.825	62.352	68.257	82.616
LightIntensityLux	25840	17.34333026	28.67318121	0	0	0	35.189	99.976
				ROBOD Datasets (Clean)				
TemperatureCelcius	24999	26.564304	1.460293804	20.44275856	25.76966667	26.84166718	27.56866646	33.99517059
HumidityPercent	24999	69.26420058	9.852598492	45.3851738	61.03533364	68.65862274	78.06383515	93.0039978
LightIntensityLux	24999	39.09897512	60.20112511	0	0	1.840000033	72.88666534	2015.545044
CO2PPM	24999	468.0395156	67.99594538	400	427.1666565	447.2666626	486.8643646	1489.357178
OccupantsCount	24999	1.457778311	3.784874294	0	0	0	1	38

Table 4 *Statistical evaluation on the cleaned datasets processed by proposed efficient MCD+OSVM model*

	count	mean	std	min	25%	50%	75%	max
				CO2 Dataset (Clean)				
TemperatureCelcius	21037	34.1883348	2.2913706	27.6	33.1	34.7	35.6	39.2
HumidityPercent	21037	51.4640662	6.202333715	36.1	49	51.1	53.5	65.4
LightIntensityLux	21037	26.2740267	7.257266058	2	19	27	32	57
CO2PPM	21037	154.856491	87.20869085	2	82	167	215	590
				QCAT Dataset (Clean)				
TemperatureCelcius	23258	24.6961927	1.498109126	21.91	23.52	24.54	25.65	28.85
HumidityPercent	23258	63.6819576	7.177136901	46.077	58.441	64.03	68.736	82.616
LightIntensityLux	23258	13.9713116	25.21810199	0	0	0	20.409	97.289
				ROBOD Dataset (Clean)				

TemperatureCelcius	22499	26.5238671	1.442086206	20.4427586	25.76466751	26.79733276	27.5170002	32.67733383
HumidityPercent	22499	69.2147192	9.890156411	45.3851738	60.89814377	68.57172394	78.13199997	93.0039978
LightIntensityLux	22499	36.353507	50.67159607	0	0	2.920000076	72.45948029	208.8200073
CO2PPM	22499	462.230113	52.21688603	400	426.758606	446.1666565	483.0833435	738.1333618
OccupantsCount	22499	0.65887373	1.396579261	0	0	0	0	6

From the tables, in general, cleaning the datasets before it is used for training is a crucial step in any AI and ML models' training, including OSVM models. Taking the standard deviation of each dataset, which is an important determinant of how closely packed the data points are to each other in a dataset, and is used in Boxplot and Z-Score statistical methods for outlier detections, the MCD cleaned dataset has an overall lower standard deviation in all the training dataset's features, with exception of the Relative Humidity (%) where the cleaned dataset has a higher value of standard deviation against the uncleaned dataset, but only by 0.04 only. This concludes the training datasets after cleaning are better training datasets than uncleaned training datasets. As for the effects of a cleaned/uncleaned training dataset on the OSVM models, it is the aim of this research and will be experimented and discussed in the Results & Discussions section.
Illustrations

3.3 Training and Testing the Models, Analysis & Evaluation Metrics

The development of the supervised learning models happens after the data pre-processing phase. All the programming and coding are done in Python, and the models are developed on Google Collaboratory. The models come from the PYOD and SciKit Learn (sklearn) Library. The models used are the MCD and OSVM. All usage of the models (training the models and testing them out with test data) follows the instructions and guide from PYOD SciKit Learn (sklearn) Official Documentation. Below are the explanations of the baseline and proposed models from the PYOD SciKit Learn (sklearn) Official Documentation and their use cases in this analysis paper, alongside the pseudo-code for each main process in the program.

The outputs needed are the outlier scores and outlier labels of the testing dataset. After obtaining a cleaned datasets using MCD as the training datasets, fitting it to the OSVM model will allow it to determine and set the Normalcy Region. The trained OSVM model is then fitted with testing data to get the result needed in the form of outlier scores and outlier labels for the testing data. These results are then imported into Data Frame columns and exported together with the data points in the form of an Excel Workbook. It is important for the training data to be as clean as possible for the training purposes, and tweaking the various parameters, most importantly the kernel used and nu hyperparameter is required to obtain the best suited OSVM model for the research.

// Efficient MCD-OSVM Model//

```
INPUT: Training Data (Uncleaned)
OUTPUT: Testing Data, Labelled and Scored for Outliers
    BEGIN:
1.  Prepare the collected Datasets, CO2, QCAT & ROBOD.
2.  Data Pre-processing, each dataset is cleaned using the MCD model to remove any outliers, absurd, invalid, out-
    of-range and empty data points.
3.  Apply initial settings of OSVM: NU = 0.001 and GAMMA = "scale" (Default kernel)
4.  Train 3 different instances of OSVM models with the 3 training datasets.
5.  Trained OSVM models are used for outlier detection on testing datasets: //Testing datasets obtained from
    random mutation and modification of training datasets.
    a.  The OSVM Models score each data points with the outlier score based on how far the data points are from
        the Normalcy Region.
    b.  OSVM models label each data point with either -1 for outliers or 1 for inliers based on the outlier scores.
6.  Labelled testing datasets are outputted as result.
7.  Repeat the steps from 3 to 6 with different NU values. The recorded results have NU values settings at 0.001,
    0.00125 and 0.00075.
8.  END
```

The outputs needed are the outlier scores and outlier labels. After obtaining a cleaned datasets using MCD as the training datasets, fitting it to the OSVM model will allow it to determine and set the Normalcy Region. The trained OSVM model is then fitted with testing data to get the result needed in the form of outlier scores and outlier labels for the testing data. These results are then imported into Data Frame columns and exported together with the data points in the form of an Excel Workbook. It is important for the training data to be as clean as possible for the training purposes, and tweaking the various parameters, most importantly the kernel used and nu hyperparameter is required to obtain the best suited OSVM model for the research.

3.4 Performance Evaluation

After the model has been successfully trained and tested for the first time, additional settings are incorporated into the program to evaluate the model's performance. Two performance evaluation metrics are essential for this analysis. The first is accuracy, which measures the proportion of correctly identified data and can be calculated using a straightforward equation:

$$\frac{TN + TP}{TN + TP + FN + FP} \tag{5}$$

Where TN represents True Negatives, TP represents True Positives, FN represents False Negatives, and FP represents False Positives, this metric can be easily computed either manually or programmatically [26]. The second metric is execution time, which measures how long the model takes to run. This can be tracked using the Performance Counter (perf_counter).

4. Result and Discussion

Once the OSVM models are successfully trained, the testing dataset is used by the models to test their performances. Three different settings of the OSVM models are used, for Test 1, the hyperplane parameter NU is set to 0.001, Test 2, NU = 0.00125, and Test 3, NU = 0.00075. The results are as below:

Table 5 *Comparison of accuracy between base OSVM & efficient MCD-OSVM models on three datasets*

The efficient MCD-OSVM Model in overall has an improved accuracy than the normal OSVM model, which is

Experiment	Comparison of Accuracy between OSVM & Efficient MCD-OSVM Models on Three Datasets							
	CO2 Dataset (Outliers Detected)				QCAT Dataset (Outliers Detected)			
	OSVM	Accuracy	MCD-OSVM	Accuracy	OSVM	Accuracy	MCD-OSVM	Accuracy
EXP 1	484	57.21%	670	79.20%	607	87.09%	621	89.10%
EXP 2	479	58.06%	644	78.06%	559	82.09%	560	82.23%
EXP 3	498	57.94%	682	79.39%	589	83.31%	594	84.02%

expected as like most Supervised ML Outlier Detection models, which are ML models that identify patterns in the inputted data to classify whether a data point is outlier or not, by first learning the patterns from the training dataset. It is also stated in different documentations introducing OSVM that OSVM is a model sensitive to outliers, which is both a blessing and a curse in different situations [9-13]. Thus, it is crucial to obtain a training dataset that is as clear of outliers as possible to train the OSVM model. The Efficient MCD-OSVM Model can perform well enough in the above CO2 dataset and QCAT dataset, having 4 and 3 features or dimensions respectively, fine tuning with the optimal NU value is possible to improve the accuracy even more. However, the case is different with the third dataset, the ROBOD dataset which has 5 dimensions and largely varied values in each feature:

Table 6 *Comparison of accuracy between base OSVM & efficient MCD-OSVM models on three datasets (2)*

Experiment	Comparison of Accuracy between OSVM & Efficient MCD-OSVM Models on Three Datasets			
	ROBOD Dataset (Outliers Detected)			
	OSVM	Accuracy	MCD-OSVM	Accuracy
EXP 1	4898	20.87%	2246	45.50%
EXP 2	4918	21.39%	2202	47.77%
EXP 3	4945	20.69%	4600	22.24%

For the ROBOD dataset, while in a better accuracy is still obtained by the Efficient MCD-OSVM Model, it is far from a satisfactory accuracy for an outlier detection method, scoring only 45.5%, 47.8% and 22.2% accuracy for the 3 settings of NU = 0.001, NU=0.00125, and NU=0.00075 respectively. This is mainly due to the curse of dimensionality, where the generalization error of shallow structures in ML methods such as OSVM and SVM increases with the increasing number of irrelevant and redundant features. Thus, to able to use OSVM for a dataset with higher dimensions, it is significant to deploy some sort of dimensionality or feature reduction methods to allow the capturing large amounts of differences in the data pattern without the need for calculating all of them for outlier detection [27].

The main and most important limitation of this analysis is the experimentation and fine-tuning of the parameters of the OSVM. For OSVM, there are 6 critical parameters that can be set to control the OSVM model. First and foremost, the NU parameter, which is both the upper bound on the fraction of possible training errors and the lower bound of the fraction of support vectors, by default, 0.5 is set, which allows 50% of the data points in the training data to be identified as possible outliers and false positives; Secondly, Kernel, which is the kernel function used to identify the type of decision boundary the OSVM should use. For Python Scikit Learn OSVM and SVM, various types of kernels are supported, the default kernel used is the Gaussian Radial Basis Function or RBF in short, which can effectively encapsule complex non-linear relationships in the data, other choices are linear, polynomial and sigmoid kernel; The third parameter is a non-linear hyperplane exclusive, which is Gamma. It determines the influence of a single training sample is, where the higher the value, the closer the other samples must be to be affected; fourth and fifth parameters is degree and coefficient respectively, which are important for polynomial and sigmoid kernels for fine-tuning the OSVM for better performances; lastly is Tol, which indicates the tolerance of stopping criterion, which halts the algorithm when a duality gap smaller than the tolerance is achieved. Since this is a comparative analysis where the same setting is used across the different variations of the OSVM models, the effects of not fine-tuning the parameters are not affecting the results of the testing, however, when the models are used for outlier detection, fine-tuning is necessary for optimal performances.

As for how much extra time is recorded when the proposed model is in used, the average time used is recorded when the MCD model is in the data cleaning operation. The result is as below:

Table 7 *Average time used by the MCD model during data cleaning process*

Experiment	Datasets (Number of Features) (Seconds)			Average Time Used by Experiment
	CO2 Dataset (4 Features)	QCAT Dataset (3 Features)	ROBOD Dataset (5 Features)	
Experiment 1	6.2421	3.5943	13.6915	7.8427
Experiment 2	9.2585	3.4801	5.6820	6.1402
Experiment 3	19.2056	8.5914	15.5083	14.4351
Average Time Used by Dataset	11.56873333	5.221933333	11.62726667	9.4727

From the table above, with only an average extra 9.5 seconds spent, the proposed model can have an average 13.21 % increase in overall accuracy. This should be great as it is hard to determine the time used for data cleaning when applying manual cleaning and statistical methods for data cleaning in data preparation and preprocessing.

Another advantage that comes from a cleaner dataset resulted from the application of advance outlier detection model for data pre-processing is the indirect support in decreasing the work time of OSVM models. This is observed during the fine tuning of the NU parameter for OSVM model. The results are shown in the following table:

Table 8 *Average operation time of OSVM in different NU parameter value*

NU Value	Operation Time in each Experiment (s)			Average Operation Time (s)
	1	2	3	
0.001	0.0701	0.1451	0.0717	0.095633333
0.1	17.3921	9.4020	9.5853	12.12646667
0.5	42.6506	40.6293	38.3599	40.5466

The usage of lower NU value affects the work time of OSVM during the training and testing process, as shown from the table. Cleaning a dataset will help the process as it allows a lower NU value for the OSVM. MCD helps even more as it cleans the dataset even more effectively, and an even lower value of NU can be used during the training phase. Although, ultimately, fine-tuning the NU parameter requires testing according to different datasets and number of possible outliers existing in the training dataset.

5. Conclusion

The proposed MCD+OSVM can out-perform the baseline OSVM model as the usage of MCD in the data pre-processing phase for data cleaning can eliminate more extreme data with absurd values and out-of-range data, achieving a dataset with more tightly clustered data points (lower standard deviation), improving the accuracy of an OSVM model by 13.21%. Applying MCD model will increase the time taken for data cleaning process by an average of 9.5 seconds, but it will allow a lower NU parameter to be used during the training and testing of OSVM model which will help with decrease its operational time, resulting an overall shorter execution time and better efficiency. Since the final goal of the OSVM is to achieve a smaller and more precise normalcy region, the MCD cleaned dataset complements the OSVM's objective and thus able to achieve a more optimal performance in outlier detection.

Acknowledgements

This review is done under the support of the GPPS Grant of UTHM, at GPPS VOT Q596.

Conflict of Interest

The authors hereby declare that there are no financial, personal, or professional conflicts of interest that could inappropriately influence or bias the conduct, results, or interpretation of this research. The authors do not have any relationships or affiliations that could be perceived as potential conflicts of interest with respect to this work.

All sources of funding, if appropriate, should have been sincerely disclosed, and all collaborators and contributors have been properly acknowledged. The authors affirm that this research has been conducted with integrity and in accordance with the ethical guidelines relevant to the field.

Author Contribution

*The authors confirm contribution to the paper as follows: **study conception and design:** Parh, Nayef, Hairulnizam; **data collection:** Parh, Nayef; **analysis and interpretation of results:** Parh, Nayef, Waheed; **draft manuscript preparation:** Parh, Nayef, Abdul-Malik, Antar, Abdullah. All authors reviewed the results and approved the final version of the manuscript.*

References

[1] Saleem, M. U., Usman, M. R., & Shakir, M. (2021). Design, Implementation, and Deployment of an IoT-Based Smart Energy Management System. IEEE Access, 9, 59649–59664. doi:10.1109/access.2021.3070960

[2] Saleem, M., Shakir, M., Usman, M., Bajwa, M., Shabbir, N., Shams Ghahfarokhi, P., & Daniel, K. (2023). Integrating smart energy management system with internet of things and cloud computing for efficient demand side management in smart grids. Energies, 16(12), 4835. https://doi.org/10.3390/en16124835

[3] Wang, R. Y., & Strong, D. M. (1996). Beyond Accuracy: What Data Quality Means to Data Consumers. Journal of Management Information Systems, 12(4), 5–33. doi:10.1080/07421222.1996.11518099

[4] Geiger, R. S., Cope, D., Ip, J., Lotosh, M., Shah, A., Weng, J., & Tang, R. (2021). 'Garbage In, Garbage Out' Revisited: What Do Machine Learning Application Papers Report About Human-Labeled Training Data? doi:10.48550/ARXIV.2107.02278

[5] Yassine, S., & Stanulov, A. (2024). A comparative analysis of machine learning algorithms for the purpose of predicting norwegian air passenger traffic. International Journal of Mathematics, Statistics, and Computer Science, 2, 28–43.

[6] Hubert, M., & Debruyne, M. (2009). Minimum covariance determinant. WIREs Computational Statistics, 2(1), 36–43. doi:10.1002/wics.61

[7] Raymaekers, J., & Rousseeuw, P. J. (2023). The Cellwise Minimum Covariance Determinant Estimator. Journal of the American Statistical Association, 1–12. doi:10.1080/01621459.2023.2267777

[8] Leys, C., Klein, O., Dominicy, Y., & Ley, C. (2018). Detecting multivariate outliers: Use a robust variant of the Mahalanobis distance. Journal of Experimental Social Psychology, 74, 150–156. doi:10.1016/j.jesp.2017.09.011

[9] Al Shorman, A., Faris, H., & Aljarah, I. (2019). Unsupervised intelligent system based on one class support vector machine and Grey Wolf optimization for IoT botnet detection. Journal of Ambient Intelligence and Humanized Computing, 11(7), 2809–2825. doi:10.1007/s12652-019-01387-y

[10] Gomez-Verdejo, V., Arenas-Garcia, J., Lazaro-Gredilla, M., & Navia-Vazquez, Á. (2011). Adaptive One-Class Support Vector Machine. IEEE Transactions on Signal Processing, 59(6), 2975–2981. doi:10.1109/tsp.2011.2125961

[11] Ji, M., & Xing, H.-J. (2017, May). Adaptive-weighted one-class support vector machine for outlier detection. 2017 29th Chinese Control And Decision Conference (CCDC). doi:10.1109/ccdc.2017.7978802

[12] Shin, H. J., Eom, D.-H., & Kim, S.-S. (2005). One-class support vector machines—an application in machine fault detection and classification. Computers & Industrial Engineering, 48(2), 395–408. doi:10.1016/j.cie.2005.01.009

[13] Bounsiar, A., & Madden, M. G. (2014, May). One-Class Support Vector Machines Revisited. 2014 International Conference on Information Science & Applications (ICISA). doi:10.1109/icisa.2014.6847442

[14] Himeur, Y., Alsalemi, A., Bensaali, F., & Amira, A. (2021). Smart power consumption abnormality detection in buildings using micromoments and improved K-nearest neighbors. International Journal of Intelligent Systems, 36(6), 2865–2894. doi:10.1002/int.22404

[15] Park, C. H., & Kim, T. (2020). Energy Theft Detection in Advanced Metering Infrastructure Based on Anomaly Pattern Detection. Energies, 13(15), 3832. doi:10.3390/en13153832

[16] Wu, Y., Dai, H.-N., & Tang, H. (2022). Graph Neural Networks for Anomaly Detection in Industrial Internet of Things. IEEE Internet of Things Journal, 9(12), 9214–9231. doi:10.1109/jiot.2021.3094295

[17] Jaiswal, R., Chakravorty, A., & Rong, C. (2020, August). Distributed Fog Computing Architecture for Real-Time Anomaly Detection in Smart Meter Data. 2020 IEEE Sixth International Conference on Big Data Computing Service and Applications (BigDataService). doi:10.1109/bigdataservice49289.2020.00009

[18] Frikha, M. S., Gammar, S. M., & Lahmadi, A. (2021, November). Multi-Attribute Monitoring for Anomaly Detection: a Reinforcement Learning Approach based on Unsupervised Reward. 2021 10th IFIP International Conference on Performance Evaluation and Modeling in Wireless and Wired Networks (PEMWN). doi:10.23919/pemwn53042.2021.9664667

[19] Pourahmadi, V., Alameddine, H. A., Salahuddin, M. A., & Boutaba, R. (2023). Spotting Anomalies at the Edge: Outlier Exposure-Based Cross-Silo Federated Learning for DDoS Detection. IEEE Transactions on Dependable and Secure Computing, 20(5), 4002–4015. doi:10.1109/tdsc.2022.3224896

[20] Gulhare, A. K., Badholia, A., & Sharma, A. (2022, July). Mean-Shift and Local Outlier Factor-Based Ensemble Machine Learning Approach for Anomaly Detection in IoT Devices. 2022 International Conference on Inventive Computation Technologies (ICICT). doi:10.1109/icict54344.2022.9850880

[21] Bhatti, M. A., Riaz, R., Rizvi, S. S., Shokat, S., Riaz, F., & Kwon, S. J. (2020). Outlier detection in indoor localization and Internet of Things (IoT) using machine learning. Journal of Communications and Networks, 22(3), 236–243. doi:10.1109/jcn.2020.000018

[22] Wong, P. Y., Alduais, N. A., Omar, N. A., Mostafa, S. A., Saad, A. M. H., Abdul-Qawy, A. S. H., ... & Ghanem, W. A. H. (2024). Comparative Analysis of ML-Based Outlier Detection Techniques for IoT-Based Smart Energy Management Systems. Journal of Intelligent Systems & Internet of Things, 12(2). doi:10.54216/JISIoT.120204

[23] Wibisono, A. (2020). Data for: Short-term Prediction of CO2 Concentration based on a Wireless Sensor Network. doi:10.17632/6D798DKHPZ.1

[24] Kusy, B., Hovington, L., Hu, W., & Rana, R. (2012). QCAT Smart Office environment - Humidity. doi:10.4225/08/50629B0DE50C7

[25] Tekler, Z. D., Ono, E., Peng, Y., Zhan, S., Lasternas, B., & Chong, A. (2022). ROBOD, room-level occupancy and building operation dataset. Building Simulation, 15(12), 2127–2137. doi:10.1007/s12273-022-0925-9

[26] Varoquaux, G., & Colliot, O. (2023). Evaluating Machine Learning Models and Their Diagnostic Value. In Neuromethods (pp. 601–630). doi:10.1007/978-1-0716-3195-9_20

[27] Erfani, S. M., Rajasegarar, S., Karunasekera, S., & Leckie, C. (2016). High-dimensional and large-scale anomaly detection using a linear one-class SVM with Deep Learning. Pattern Recognition, 58, 121–134. https://doi.org/10.1016/j.patcog.2016.03.028