applied sciences

MDPI

*Article*

# Performance Evaluation of Machine Learning Methods for Anomaly Detection in CubeSat Solar Panels

**Adolfo Javier Jara Cespedes *, Bramandika Holy Bagas Pangestu, Akitoshi Hanazawa and Mengu Cho**

Laboratory of Lean Satellite Enterprises and In-Orbit Experiments (LaSEINE), Kyushu Institute of Technology, Kitakyushu 8048550, Japan

* Correspondence: javier.jara-cespedes758@mail.kyutech.jp

**Abstract:** CubeSat requirements in terms of size, weight, and power restrict the possibility of having redundant systems. Consequently, telemetry data are the primary way to verify the status of the satellites in operation. The monitoring and interpretation of telemetry parameters relies on the operator's experience. Therefore, telemetry data analysis is less reliable, considering the data's complexity. This paper presents a Machine Learning (ML) approach to detecting anomalies in solar panel systems. The main challenge inherited from CubeSat is its capability to perform onboard inference of the ML model. Nowadays, several simple yet powerful ML algorithms for performing anomaly detection are available. This study investigates five ML algorithm candidates, considering classification score, execution time, model size, and power consumption in a constrained computational environment. The pre-processing stage introduces the windowed averaging technique besides standardization and principal component analysis. Furthermore, the paper features the background, bus system, and initial operational data of BIRDS-4, a constellation made of three 1U CubeSats released from the International Space Station in March 2021, with a ML model proposal for future satellite missions.

**Keywords:** anomaly detection; BIRDS project; CubeSat; machine learning; electrical power system



**Citation:** Cespedes, A.J.J.; Pangestu, B.H.B.; Hanazawa, A.; Cho, M. Performance Evaluation of Machine Learning Methods for Anomaly Detection in CubeSat Solar Panels. *Appl. Sci.* **2022**, *12*, 8634. https://doi.org/10.3390/app12178634

Academic Editors: Simone Battistini, Filippo Graziani and Mauro Pontani

Received: 19 July 2022
Accepted: 25 August 2022
Published: 29 August 2022



## 1. Introduction

The development of small satellites by non-space-faring nations is significantly driven by the availability of low-cost launch and commercial off-the-shelf (COTS) components. A clear example is the Joint Global Multi-National Birds or BIRDS program, a multinational small satellite research and educational program led by the Kyushu Institute of Technology (Kyutech) in Japan [1]. The BIRDS program allows non-space-faring nations to design, integrate, build, test, launch, and operate their respective first satellites.

BIRDS-4 is the fourth iteration of the BIRDS program (previously BIRDS-1, 2, and 3). It is a constellation of three CubeSats: GuaraniSat-1, Maya-2, and Tsuru, deployed into orbit on 14 March 2021 [2]. The satellites use the BIRDS standardized bus, an open-source initiative by Kyutech designed for educational CubeSat projects inherited from the previous generation [3].

The BIRDS program follows the approach of a Lean Satellite concept [3]. The concept relies on utilizing commercially available yet non-space-proven components to obtain effective and efficient development [4]. On the other hand, the mass and size limitations usually indicate only a few or no redundant systems available [5]. The intermittent and short-term communication window limits the data transmission capability, potentially affecting the housekeeping-data-monitoring analysis. Furthermore, the CubeSat system's limitations generally include power generation, telemetry bandwidth, computational power, and memory [6].

The BIRDS team uses conventional threshold values to perform telemetry analysis for the satellite's health monitoring. However, considering the available bandwidth and large volumes of data collected onboard the satellite, it is impossible to download all the data

during the operational phase. In addition, data monitoring requires a team with sufficient knowledge and experience due to the complex variations in telemetry patterns.

According to the BIRDS-3 telemetry data, two satellites revealed power generation loss in one of their panels. The team could not immediately detect the symptoms within 29 (on Raavana) and 72 (on Uguisu) days after being deployed into orbit. Recently, the BIRDS-4 team discovered a critical issue related to the electrical power subsystem of GuaraniSat-1 [2]. The satellite stopped transmitting after three days of operation. Subsequent analysis of the continuous wave (CW) beacon data showed a lower charging capacity than its sister satellite, Tsuru, confirming no power generation on two solar panels. Therefore, there is an urgent need to develop solutions to mitigate the issue from any possible technical approach. One of the possible approaches would be utilizing Machine Learning (ML), considering the substantial amount of data collected by the BIRDS team.

The analysis for related research comes from these three main aspects: pre-processing techniques, proximity-based algorithms, and linear models. As a representative case, the work presented by Wu J. et al. [7] used Long Short-Term Memory (LSTM) and Ensembled One-Class Support Vector Machines to perform anomaly detection. Multiple One-Class Support Vector Machines were used to obtain high-precision and high-recall outputs. The method was evaluated using the telemetry data of the SMAP (Soil Moisture Active Passive) satellite and MSL (Mars Science Laboratory). Jin W. et al. [8] proposed a cluster-based anomaly threshold determination method in another paper. Experiments were conducted on satellite telemetry data, showing that the proposed method outperformed the autoencoders.

Probabilistic clustering approaches, such as the work presented by Yairi T. et al. [9], are also intensely interesting. In this work, the authors proposed a data-driven health monitoring and anomaly detection method for artificial satellites based on probabilistic dimensionality reduction and clustering. The proposed method was experimentally applied to Japan Aerospace Exploration Agency (JAXA) housekeeping data of Small Demonstration Satellite 4 (SDS-4) and validated over two years.

Finally, different ML-based approaches have also received considerable attention for anomaly detection. For example, techniques based on Principal Component Analysis (PCA), such as the technique presented by Zamry N. et al. [10], have been used to improve efficiency by reducing the computational complexity and improving memory utilization overhead while maintaining high accuracy. The works presented by Pan D. et al. [11], Peng Y. et al. [12], and Li J. et al. [13] also used PCA for feature extraction and fault detection in telemetry data. The k-nearest neighbor (kNN), Recurrent Neural Networks (RNN), autoencoder, and One-Class Support Vector Machine (OC-SVM) have also been studied with spacecraft (Suzaku) electrical power system data [14].

The term "anomaly" refers to any point or value within the dataset that produces patterns of oddity, novelty, fault, deviation, or exceptions. Hence, the activity of identifying anomalies from normal data is known as anomaly detection.

This paper presents a comparative study of different ML techniques to detect anomalies in CubeSat telemetry data, considering system limitations such as computational power, memory footprint, and communication window. The approach considers both pre-processing methods and ML models. We applied the process to a solar panel's dataset generated by BIRDS-3 (NepaliSat, Raavana, Uguisu) and BIRDS-4 (Tsuru) satellites. It is important to note that this is the first utilization of such data. In summary, the contributions of this work are as follows:

- Analysis of the novel solar panel dataset collected from four CubeSats.
- Analysis of five different ML models based on their classification scores, execution times, model sizes, and power consumption.
- The proposal of ML model candidates for solar panel anomaly detection on CubeSat systems.

The contributions of this work to the research community are the findings showing that specific ML models, i.e., linear models, are most suitable for solar panel anomaly de-

tection onboard CubeSats, given the lack of attitude control and constrained computational capability of future satellite projects. The rest of the paper is organized in the following way: Section 2 introduces the satellite system and the dataset. Section 3 describes the methods utilized. Section 4 provides the results obtained. Section 5 presents analysis and discussion regarding the results of the experiments on ML modelling and future works. Finally, Section 6 summarizes the outcome of the research and proposes the best model to detect the anomalies within the dataset.

## 2. Materials

### 2.1. BIRDS Satellite System

The satellites of the BIRDS-3 and BIRDS-4 constellations are based on the 1U CubeSat standard. The external dimensions are (114.5 × 107.8 × 104) mm, and they have an average mass of 1.3 kg. The bus system consists of the front access board (FAB), the electrical power system (EPS), the onboard computer (OBC), the communication board (COM), and the rear access board (RAB). All printed circuit boards (PCBs) are connected to a backplane board via 50-pin connectors. Figure 1 shows the bus system configuration.

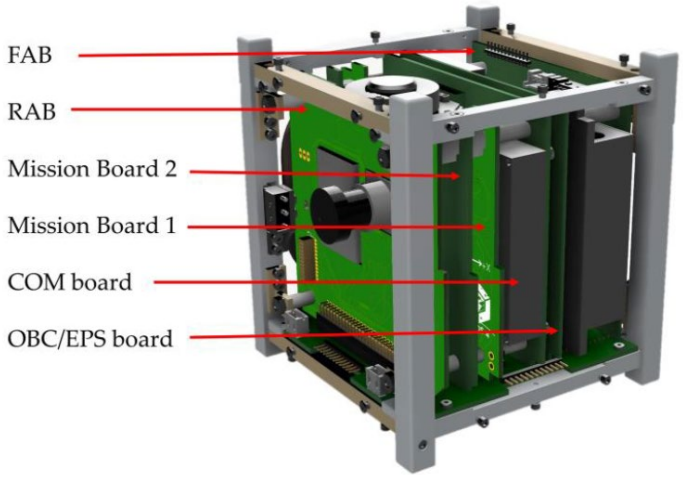



**Figure 1.** Bus system configuration.

The RAB is used for programming and monitoring each mission's microcontrollers. The FAB is used for programming and monitoring the Main PIC, FAB PIC, RESET PIC, and COM PIC microcontrollers. The details of the satellite system can be found in the BIRDS Open-Source Repository [15].

In BIRDS-3, two available mission boards are utilized to space-qualify a low-powered long range (LoRa) modulation module, which has massive potential for future S&F missions. This mission is called a LoRa Demonstration Mission (LDM) [16].

On the other hand, the BIRDS-4 constellation uses the mission boards for the nine satellite missions. The satellite structure was renewed by taking the BIRDS structure as a base to satisfy the requirements of the HNT mission [2,17]. The downlink frequency is 437.375 MHz, whereas APRS-DP and store-and-forward missions use the VHF band.

Through the uplink signal sent by the ground station, the satellite receives, processes, and downlinks telemetry and mission data back to the ground station. The satellites use dipole antennas for communication. The antennas are deployed 30 min after satellite deployment. The uplink is performed through the BIRDS Ground Station Network, which has 13 participating countries worldwide, except in Europe and North America.

Both constellations share the same EPS design and components. Solar panels placed on five sides of the CubeSat (+X, +Y, +Z, −X, −Z) are responsible for power generation. The solar panel consisting of two cells connected in series is shown in Figure 2.

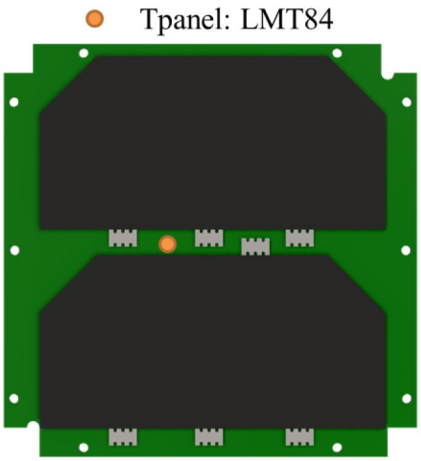



**Figure 2.** Solar panel with two cells connected in series.

The specific solar cell (AZURESPACE 3G30A) has an ideal maximum power generation of 1.2 W. The BIRDS BUS uses Linear Technology Corporation's LTC3119 as the Battery Charge Regulator (BCR), due to its ability to control the maximum power point of solar cells. The energy generated is stored in six rechargeable Eneloop NiMH batteries with a capacity of 1900 mAh per battery arranged in a 3-series 2-parallel configuration. The total power generation per orbit is 2600 mWh, and the total power available to the satellite load is 1600 mWh. The power system has two 3.3 V lines, one 5 V line, and two unregulated lines. All of the EPS's vital components are illustrated in Figure 3.

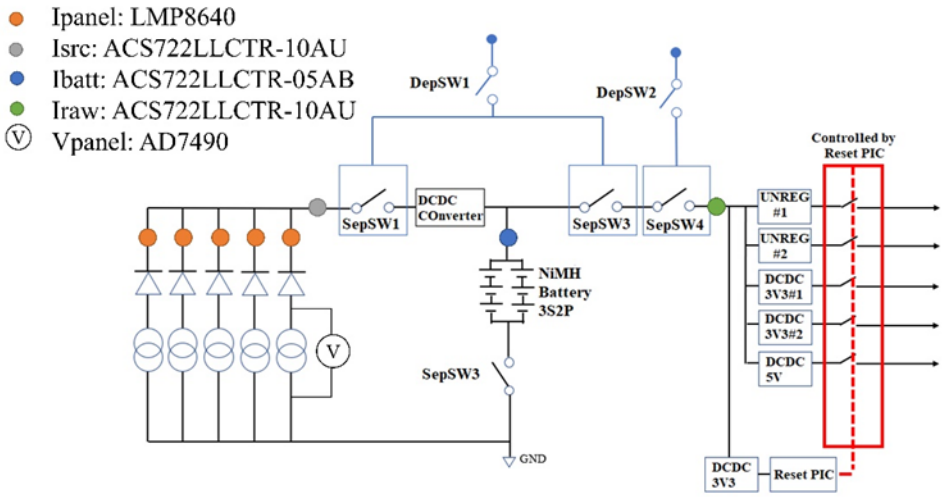



**Figure 3.** EPS block diagram with power distribution.

### *2.2. Dataset Overview*

The dataset contains solar panel data of Tsuru satellite (from March 2021 to November 2021). It includes 16,704 temperature, current, and voltage data points from the satellite panels. Additionally, the dataset consists of 8439 samples from Nepalisat, Raavana, and Uguisu Satellites (from June 2019 to October 2020). Please refer to TSURU.xlsx, NEPALISAT.xlsx, RAAVANA.xlsx and UGUISU.xlsx in Supplementary Materials section. The satellites use the sensors described in Table 1.

**Table 1.** Descriptions of solar panel sensors.

| Sensor | Variable | Unit | Data Size |
|---|---|---|---|
| LMP8640 | Current | mA | 8 bits |
| AD7490 | Voltage | mV | 12 bits |
| LMT84 | Temperature | °C | 12 bits |

The BIRDS team operators periodically execute the High Sampling Sensors Data Collection Mission (HSSC) to obtain the data during one orbit. Subsequently, the team downloads housekeeping data to the ground station. The sampling rate is 10 s for the Tsuru satellite and 5 s for Nepalisat, Raavana, and Uguisu Satellites.

### *2.3. Data Exploration and Pre-Processing*

Before detecting outliers in the data, it is essential to understand how the satellite's orbit and attitude affect the panel's voltage, current, and temperature. The BIRDS-4 constellation was deployed into orbit from the ISS (altitude 400 km, inclination: 51.6°, duration: 92.6 min). Since no attitude control is applied, the satellites are in free rotation at approximately 3 °/s on each axis. Therefore, the panel condition is affected by two factors: the beta angle that determines the eclipse and sunlight periods and the attitude perturbation of the satellite. In Figure 4a, at a low beta angle we can observe significant temperature changes (~48 °C) due to the transition of the satellite from eclipse to sunlight, and a slight temperature variance (~4 °C) due to the satellite's rotation on its axes. In Figure 4b, at a high beta angle, we observe temperature variation (~30 °C) only due to the satellite's rotation.

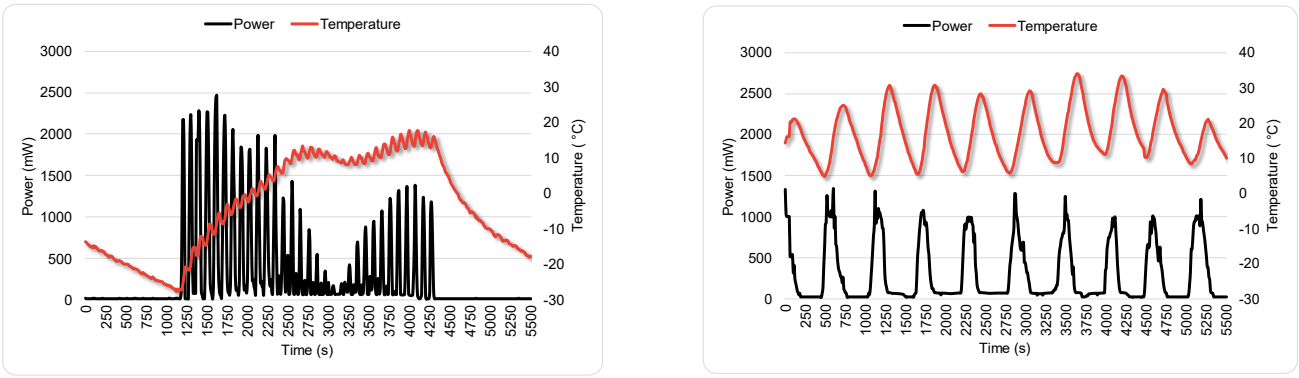


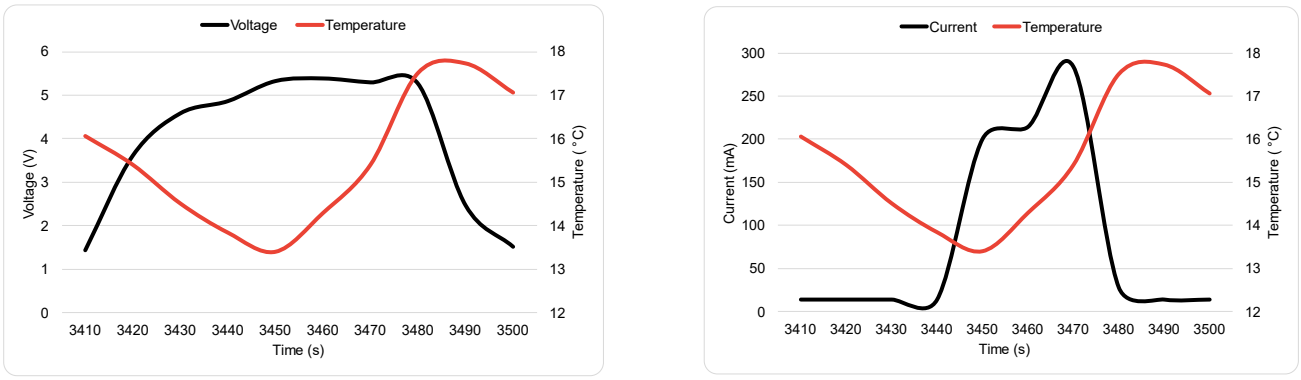

**Figure 4.** One orbit +Y panel power generation and temperature: (**a**) beta angle 15°; (**b**) beta angle 70°.

In addition, at a low beta angle, the power requirement from the satellite is high right after the eclipse period, so the power generation is high. However, at a high beta angle, since the satellite does not experience an eclipse throughout the orbit, the power requirement is less and so the generation is also less. The generated power of each panel shows large fluctuations due to the satellite's attitude (i.e., rotation), which causes different sides of the satellite to face the sun at different instants.

### *2.4. Dataset Correlation*

We can find a correlation among the solar panel's parameters by analyzing the telemetry data. Figure 5a shows the correlation between the voltage and the temperature of the +Y panel when it faces the sun for approximately 90 s at varying angles while rotating. Figure 5b shows the correlation between the current and the temperature of the same panel for the same time window. A slight delay can be observed in the current graph compared to the voltage graph. Significant current only flows for approximately 40 s because of a

specific potential difference required in the circuit between the panel output and the input of the DC/DC converter for the current to start flowing.




(**a**) (**b**)

**Figure 5.** +Y panel parameters correlation: (**a**) voltage vs. temperature, (**b**) current vs. temperature.

Due to the photovoltaic characteristic of the solar cell, the voltage and current readings are more sensitive to the change in light intensity. However, for the temperature reading, due to the heat transfer from the solar cell to the temperature sensor located behind the panel, a delay is observed for the change in temperature as compared to instant change in voltage and current during the transition from eclipse to sunlight. Hence, the satellite's temperature readings were not utilized in this study due to the low correlations of the trend with the current and voltage samples. From here on, the dataset comprised ten dimensions of current and voltage telemetry data from each panel.

## 3. Methods

### *3.1. Anomaly Definition*

This section presents is a critical step in the dataset's preparation to define anomalies. The process determines the scope of anomalies investigated in the study. The definition of anomalous and normal data is based on the analysis results of the operational satellite team. The analysis generally depends on the satellite engineering model, on-orbit condition, and several previous tests before launch. Moreover, the definition is a reference for the subsequent process of dataset labeling. Labeling is a manual process of classifying the data points as normal (label: 0) or anomalous (label: 1). According to the telemetry data analysis, two types of anomalies are happening in the solar panel system of BIRDS satellites: solar panel failure (more than one cell) and solar cell failure, which are described in detail in the following subsections.

The anomalous points are marked in red, and the regular points are in black. The data point is marked as an anomaly if it relates to the conditions of a solar panel or solar cell failure.

#### 3.1.1. Type 1: Solar Panel Failure

Solar panel failure (or type 1) specifies an anomalous condition wherein the voltage sensor reads a value below an expected threshold of 1200 mV on one or more panels at any given observation time. The expected lowest values of the sensor readings for a non-illuminated solar panel must be within 1200 to 1500 mV. The numbers refer to the residual voltage generated by the panel system's Analog to Digital Converter (ADC) circuit. Any data below the threshold indicate a bad voltage condition for a panel of two solar cells connected in series. The team specified those expected values from the operational telemetry data of the BIRDS-3 satellite (Raavana). Thus, the dataset is labeled with the following condition:

$$V_i(t) < 1200 \text{ mV}, \quad (1)$$

with $V_i(t)$ being the values of voltage sensor readings at samples of $t$ and for $i$ panel. Correspondingly, Figure 6 presents the voltage and current sensor readings in both normal (black) and anomalous (red) sections for type 1 anomalies by applying (1). It covers approximately 2.5 orbital periods. Type 1 anomalies in the dataset contaminate 8.59% of the total sample.

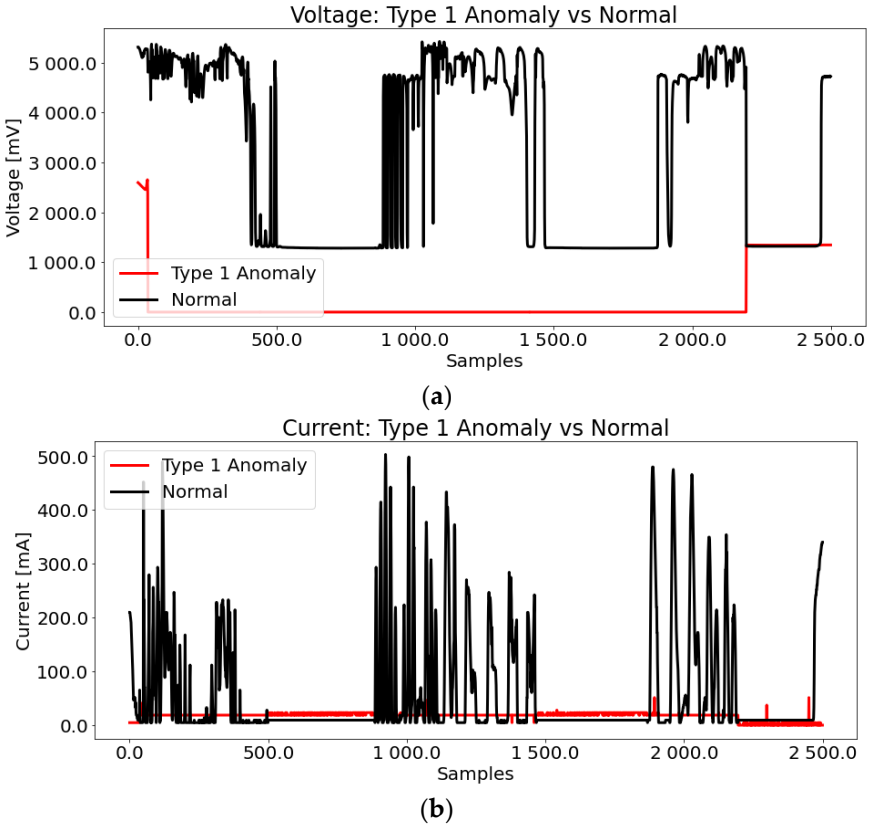



**Figure 6.** Type 1 anomalies in (**a**) voltage and (**b**) current data.

### 3.1.2. Type 2: Solar Cell Failure

Solar cell failure (or type 2) characterizes the anomaly of a compromised cell in a panel (of any panels). This specific failure is only noticeable during the illumination period. Based on the actual telemetry data of the BIRDS-3 satellite (Uguisu), the voltage sensor reading shows consecutive values (at least three samples) within 2000 to 3000 mV with no expected electrical current reading (<50 mA). Analytically, one solar cell in the Uguisu satellite is subject to an anomaly condition. However, this particular condition can happen instantly in the transition period of entering and exiting the eclipse. Therefore, the labeling method considers the previous and the following data points in a sequence. We define such anomaly as

$$2000\ \text{mV} \ \leq \ V_i(t-1,\ t,\ t+1) \leq 3000\ \text{mV} \ \wedge I_i(t) < 50\ \text{mA}\ , \tag{2}$$

with $V_i(t)$ and $I_i(t)$ represent the value of voltage and current sensor readings in a sample of $t$ and for panel $i$, respectively. Accordingly, Figure 7 depicts the voltage and current sensor readings in both normal (black) and anomalous (red) sections for type 2 anomaly by applying (2). It covers approximately 2.5 orbital periods. Type 2 anomalies in the dataset contaminate 7.22% of the samples within the dataset.

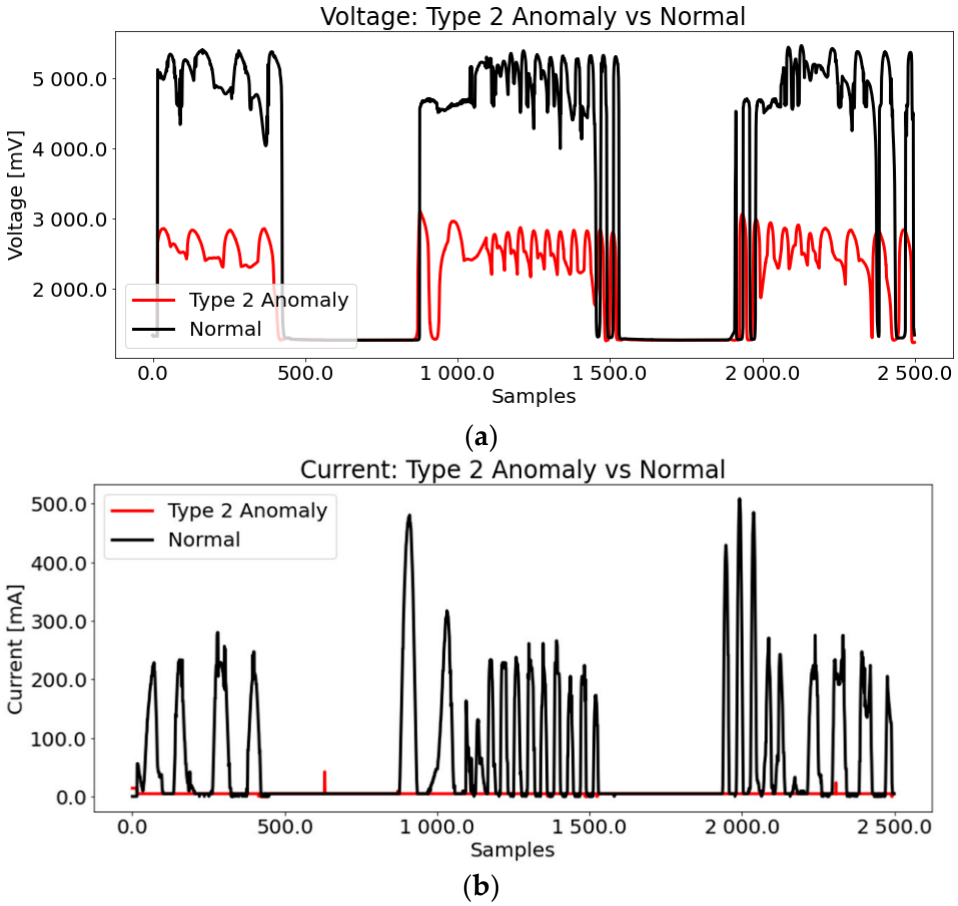



**Figure 7.** Type 2 anomalies in (**a**) voltage and (**b**) current data.

### *3.2. Pre-Processing Techniques*

Pre-processing techniques attempt to extract and enhance specific features (e.g., patterns, structure, trends) within the dataset sample. In the non-sequential analysis, the model expects this stage to increase the variance between classes in the dataset. Hence, the output forms distinctive features among classes. Implicitly, it allows the ML models to distinguish the samples. The process begins with data pre-processing using windowed averaging, standardization, and Principal Component Analysis. All three techniques were applied to the input samples that were the output of the dataset preparation phase. The analysis used two telemetry data, namely, current and voltage. Figure 8 shows the initial data distribution patterns for the voltage parameter before applying the pre-processing techniques. The figure indicates the different sample distributions of the anomalous and normal data classes. Through visual observation, we can clearly distinguish the two categories. However, it is difficult to identify the anomaly class if the analysis is based on simple thresholding. It can be seen that the anomalous patterns are located in the center of the vertical axis. However, the problem with simple thresholding is that the process may incorrectly categorize a sample in the transition state as an anomaly, especially a type 2 anomaly. Thus, a data pre-processing step is crucial to extract the anomalous patterns so that the ML model has distinguishable input samples.

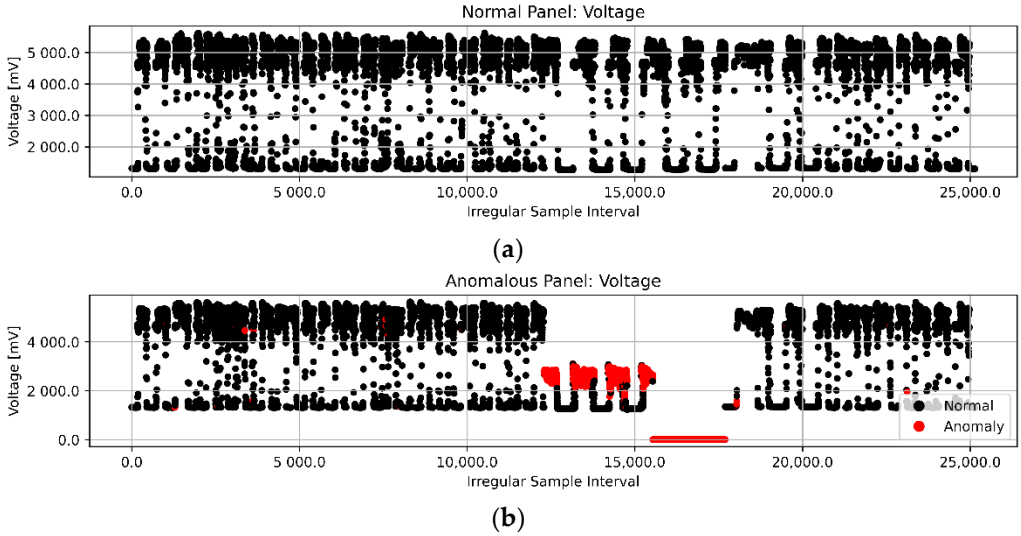



**Figure 8.** Initial voltage data plots: (**a**) normal and (**b**) anomalous.

### 3.2.1. Windowed Averaging

We propose a unique statistical method to enhance local or temporary (specific interval) trends within the dataset's parameters. Despite its utilization for non-sequential analysis, the technique introduces sequential information from the local trend intervals. It calculates an average value (average window, $\overline{X_{a,k}}$) for a particular interval (window size, $a$) in data points. Then, it replaces all values within the specific estimated window. Applying those processes for the rest of the data points creates isolated window size groups for the entire dataset. We define the required iteration ($b$) for any given number ($n$) of data points as

$$b = \left\lceil \frac{n}{a} \right\rceil , \tag{3}$$

and formulate the average window calculation as

$$\overline{X_{a,k}} = \begin{cases} \frac{\sum_i^{ak} X_i}{a}, & k < b \\ \frac{\sum_i^{ak} X_i}{n - a(b-1)}, & k = b \end{cases} , \tag{4}$$

with $k$ representing the iteration index of a specific data point and $X$ a variable of interest within the dataset. For each iteration, we define the beginning of variable index $i$ as

$$i = a(k-1) + 1 . \tag{5}$$

The results of feature extraction by windowed averaging cause rapid fluctuations in data changes to be isolated for a specific interval. Figure 9 illustrates how the windowed averaging process results in a voltage dataset from an anomaly-contaminated panel by applying (3)–(5). We decided on the value of 50 as an optimal value of the window interval for the windowed averaging technique.

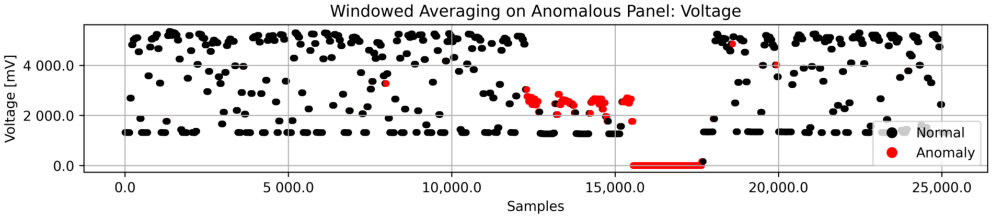



**Figure 9.** Initial voltage data plot after windowed averaging.

#### 3.2.2. Standardization

Z-score standardization or standardization is a method to standardize a set of data based on its global mean distance relative to all samples to balance the analysis. It begins with computing the global mean value for each parameter in the dataset. Then, it replaces all initial values with the differences between the global mean and the initial values. In this study, we simplified the dataset distribution according to Z-score to balance the analysis based on the global mean value of the dataset. It was applied directly after the windowed averaging technique. We used the standardization implemented by the sklearn library [18]. The method was implemented based on the z-score calculation [19].

#### 3.2.3. Principal Component Analysis

Principal Component Analysis, or PCA, is the most common dimensionality reduction technique [20]. PCA discovers the directions of the highest variance from higher-dimensional data and projects them onto a lower-dimensional subspace without losing much information. In practice, this method can reduce the number of parameters fed into the ML model without sacrificing necessary details. The lowest-dimensional projection of the PCA is known as the Principal Component (PC). Therefore, the first PC, or PC1, represents the axis consisting of the highest-variance projection. In contrast, the following PCs (PC2, PC3, and so forth) are the axis directions perpendicular to all previous PCs. We implemented PCA using the sklearn library, following the procedure in previous research [18,21].

We applied the PCA technique to the whole dataset after windowed averaging and standardization. It reduced the ten dimensions of the original dataset into two dimensions. Subsequently, the PCA-transformed data were fed into the ML modelling process.

### *3.3. Model Candidates*

We utilized ML algorithms or models capable of classifying values in an instance without considering trends or values from the previous measurement. The implemented code for all ML models is publicly available from sklearn and PyOD library [18,22]. In this work, the models originated from two categories: proximity-based algorithms and linear models. Typically, proximity-based algorithms classify data by computing the distances between samples to compare each sample's relative distance to its surroundings (e.g., local outlier factor, cluster-based local outlier factor, and k-nearest neighbor). On the other hand, the linear models learn to formulate a mathematical function to create class-based boundaries within the dataset (e.g., Linear Discriminant Analysis and One-Class Support Vector Machine). The section focuses on brief descriptions of the investigated models without going in-depth into the models' technical details. Moreover, we intended to discover the best of the two ML categories for future CubeSat system implementation.

#### 3.3.1. Proximity-Based Algorithms

Proximity-based algorithms use proximity or distance in hyperspace to classify each data point. It estimates the distance of one data point relative to its surroundings. Thus, the proximate sample number of surroundings is crucial for the result.

A. Local Outlier Factor

Local Outlier Factor (LOF) is one of the first famous local anomaly detection algorithms [23]. The LOF is essentially a ratio of local densities. It examines the local density of any sample in relation to its neighbor. Local means it depends on the object's isolation in its surroundings. It considers a significantly lower density than others as an outlier [24].

B. Cluster-Based Local Outlier Factor

Cluster-Based Local Outlier Factor (CBLOF) adopts a clustering algorithm (e.g., k-Means) to determine the area density in the dataset. Afterwards, it performs a density estimation for each cluster. The size of the cluster and the distance to the nearest large cluster define the anomaly score [25].

C. K-Nearest Neighbor

K-nearest neighbor (kNN), in the context of anomaly detection, is an algorithm that measures the distance of a data point to its kth nearest neighbor as the anomaly score [26]. We used this approach to measure the density of a dataset probability distribution. The value of the anomaly score depends on the dataset, the number of dimensions, and its normalization. As a result, it is challenging to select an appropriate value for the nearest neighbor.

#### 3.3.2. Linear Model

The ensuing algorithm learns a decision function from the dataset to create a boundary for each class. It generates a formula to create a function to predict unknown values. The linear model offers a relatively quick training phase and is straightforward to interpret [20].

A. Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) is a classifier with a linear decision function to set the boundary for each class [27]. It creates a function from the conditional densities of the labelled dataset using Bayes' rule. LDA assumes the dataset has a Gaussian distribution and each parameter has the same variance. The algorithm estimates the mean and variance of the dataset for each class.

B. One-Class Support Vector Machine

The One-Class Support Vector Machine (OC-SVM) is an ML algorithm intended to detect a novelty, i.e., a rare event. OC-SVM does not model any probability distribution from the dataset. It learns to find a function for the high- and low-density regions in the dataset based on max-margin methods. A lower density region implies a rare event, or a novelty, within the dataset [28].

### *3.4. Experimental Setup*

We performed experiments based on a computer simulation to investigate the performances of the ML algorithm candidates. The simulation followed the methodology illustrated in Figure 10.

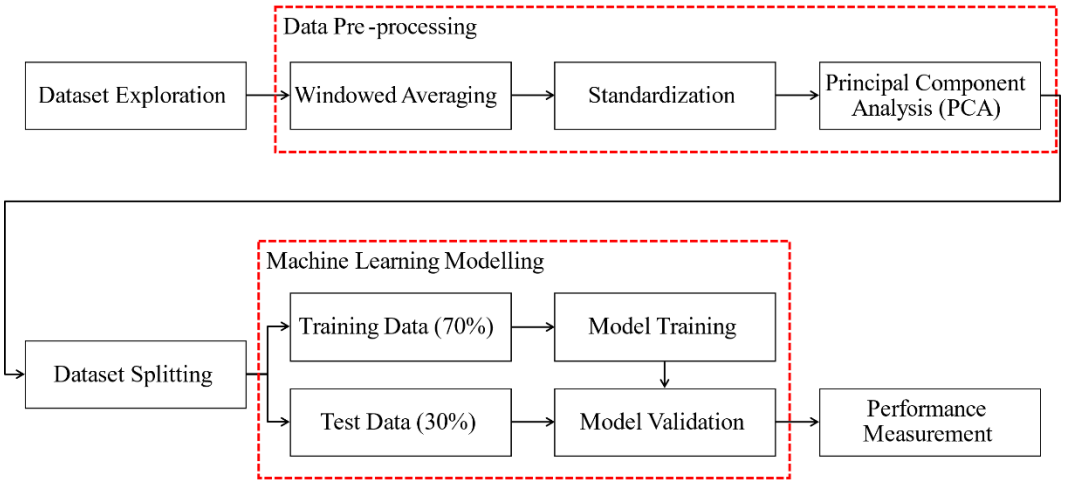



**Figure 10.** Experiment methodology.

First, we explored variables within the dataset for their correlations, characteristics, and anomaly types. Then, we manually categorized each observation point according to the anomaly definition for evaluation purposes. We applied some pre-processing techniques to the entire dataset using the following steps. By this point, we expected the data formed a distinct pattern or region density for each class. Afterwards, we randomly split the dataset

into a training set (70%) and a test set (30%) to prepare for the ML modelling stage. The ML model learned from the training set to generate a decision function. The training stage also introduced optimization, or tuning, for some variables within the ML model. Finally, we validated the optimum ML model with the test set and evaluated its performance.

#### 3.4.1. Environment and System

We performed simulations for its experimental investigation by utilizing a Jupyter Notebook (6.3.0) environment running Python (3.8.8) on Windows 11 [29,30]. A personal computer used to run the simulation was equipped with an Intel Core i7 8565U processor and 16 GB of memory. In addition, we also investigated the models' performances in the constrained computational environment, i.e., Raspberry Pi 2 Model B. The Raspberry Pi runs Raspbian OS with Python (3.9.2) on a Broadcom BCM2863 SoC (900-MHz quad-core ARM Cortex-A7 CPU 1 GB RAM). We selected this particular device as it has less computational capabilities than the planned future system implementing a CubeSat (Raspberry Pi CM3+) onboard. Raspberry Pi CM3+'s in-orbit performance has been proven in a recent operational satellite project (KITSUNE). It is an acceptable system representation to demonstrate the feasibility of model deployment. Finally, the simulation libraries were utilized as follows: pre-processing, ML modelling, evaluation, and plotting (sklearn, PyOD, pickle, NumPy, pandas, SciPy, matplotlib, scikitplot, and seaborn) [18,23,30–36].

#### 3.4.2. Simulation Parameters

According to the anomaly definition, the dataset has 8.59% type 1 and 7.22% type 2 anomalies. In total, the anomalous data across the dataset is 15.78%. Figure 11 depicts the labelled data according to the definition. Hence, it implies that the available dataset is imbalanced, with the majority consisting of normal class conditions.

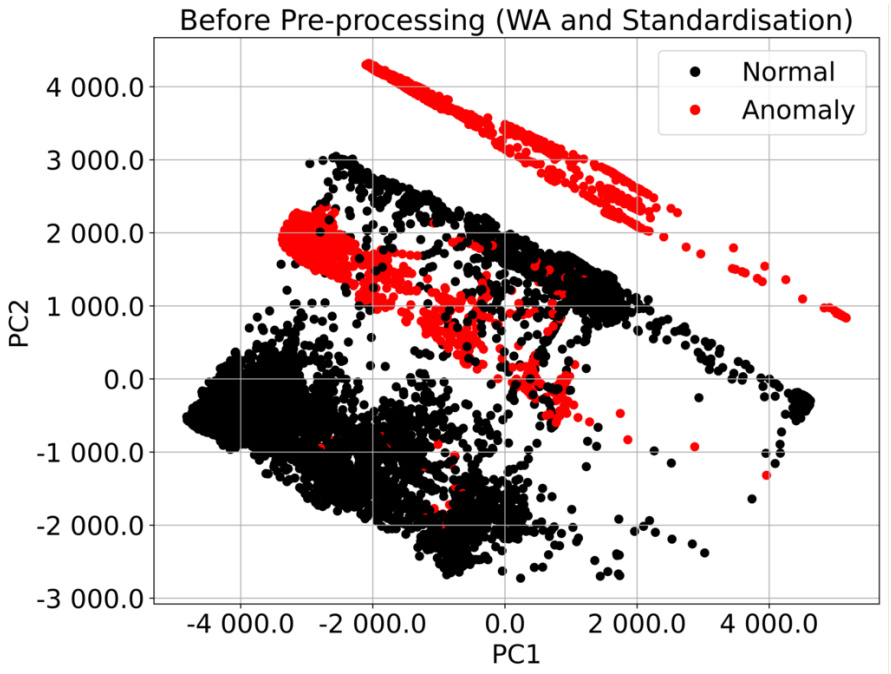



**Figure 11.** The first two principal component plots of the labelled dataset.

We implemented WA techniques in the pre-processing stage using the rolling and replace function in the pandas DataFrame library. The standardization was performed via the StandardScaler() function, available from the sklearn library. Next, the simulation implemented principal component analysis through a function provided by sklearn. Table 2 describes all necessary values for the simulation. Although we did not systematically tune the parameters, we determined the best values from an informal search for each algorithm.

**Table 2.** Simulation parameters.

| Algorithm | Optimized Parameters | Trials | Best Value |
|---|---|---|---|
| Windowed Averaging | window size | 10–300 | 50 |
| PyOD Models | outlier fraction | 0.05–0.2 | 0.16 |
| kNN | neighbors | 10–300 | 255 |
| LOF | neighbors | 10–350 | 280 |

## 4. Results

This section presents the evaluation and results regarding the experimental methodology. The pre-processing techniques were applied to emphasize any distinct pattern from each class. Figure 12 shows the labelled dataset after the pre-processing stage.

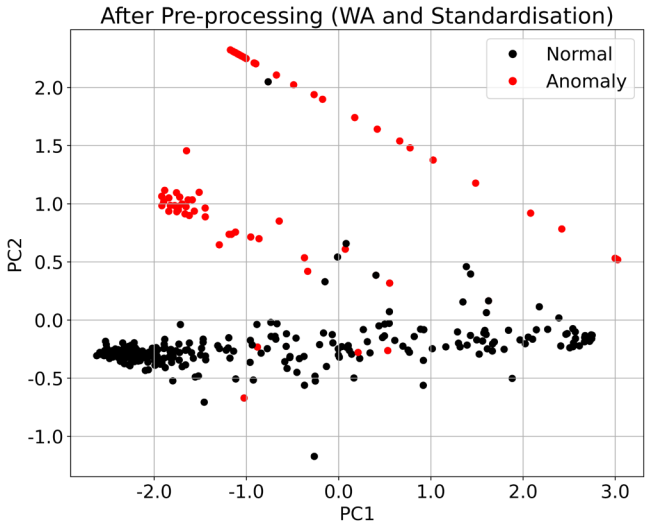



**Figure 12.** The first two principal component plots after pre-processing.

It provides a much more separable pattern between classes than the dataset plotted in Figure 11. The dataset forms a unique but linearly distinguishable pattern favoring the linear models over proximity-based models. Subsequently, the ML data were split and fed into the ML modelling phase. The rest of the section explains the evaluation of the ML models' performances.

### *4.1. Performance Evaluation*

The performance evaluation of the ML model candidates was based on classification score, inference time, model size, and power consumption. The measurement for each performance parameter was taken for an average of 10 iterations. This approach was performed to ensure that any external factors would not influence the consistency of the results.

This work evaluated a performance using F-score metrics from the classification report. F-score or F-measure represents the harmonic mean of the precision and recall scores from the predicted label to the actual label [37]. Precision is the number of true positives ($tp$) divided by the total number of positive classifications, including false positives ($fp$). Recall is the number of true positives divided by the total number of positives, including false negatives ($fn$). In binary classification, recall is usually referred to as sensitivity. The F-score ($F_\beta$) is formulated as follows:

$$F_\beta = \left(1 + \beta^2\right) \times \frac{Precision \times Recall}{(\beta^2 \times Precision) + Recall}\ , \tag{6}$$

or

$$F_\beta = \frac{(1+\beta^2)tp}{(1+\beta^2)tp + \beta^2 fn + fp}. \quad (7)$$

The F1-score ($\beta$ = 1) is selected as the minimum number of false positives, and false negatives are paramount for future onboard implementation. Further, the evaluation used F-score metrics considering the binary classification and imbalance class problem on the available dataset [38].

The inference time evaluation was measured directly by the standard package provided by the Jupyter Notebook. It measures the processing time of the CPU for specific instructions. The pickle library estimated the model size from a converted python object into a byte stream. Then, the power consumption was measured for the Raspberry Pi experiment, which considered the pre-processing stage and model inference.

### *4.2. Analysis*

Accordingly, we compare the five selected ML algorithms based on their classification performances described in Table 3.

**Table 3.** F-score for each algorithm regarding the anomaly class.

| Models | Precision * | Sensitivity * | F1-Score * |
|---|---|---|---|
| LOF | 0.26 ± 0.02 | 0.26 ± 0.03 | 0.26 ± 0.02 |
| CB-LOF | 0.64 ± 0.02 | 0.65 ± 0.02 | 0.64 ± 0.01 |
| kNN | 0.34 ± 0.01 | 0.34 ± 0.01 | 0.34 ± 0.01 |
| LDA | **0.97 ± 0.01** | 0.75 ± 0.01 | **0.85 ± 0.01** |
| OC-SVM | 0.83 ± 0.01 | **0.83 ± 0.02** | 0.83 ± 0.01 |

* Mean ± standard deviation. Maximum scores are bolded. The metrics range from 0 to 1.

The results suggest that the LDA algorithm performed best for performance evaluation. It achieved the highest capability for anomaly detection concerning its F1-score. It is also important to acknowledge that this algorithm is the simplest to implement without advanced libraries. Although it showed the overall highest F1-score, there is a significant disparity between the precision and the sensitivity. A higher precision score means the model has low false-positive rates, whereas a higher sensitivity score means the model has a low false-negative rates. Table 3 suggests that the precision score of LDA is proportionally better than its sensitivity score, which means that the model might introduce more false negatives compared to false positives. The OC-SVM reached second in anomaly detection. It was slightly behind the LDA in the overall score but offered a better balance of the precision and sensitivity scores.

Furthermore, this study investigated both linear models in a Raspberry Pi. It was essential to confirm the model's technical viability in this study for future implementation. Table 4 shows the experimental result on the Raspberry Pi.

**Table 4.** The experimental results on Raspberry Pi.

| Models | Execution Time [s] * | Model Size [kB] | Power Consumption [mWh] |
|---|---|---|---|
| LDA | **3.00 ± 0.17** | 40.17 | **4.288** |
| OC-SVM | 9.45 ± 0.39 | **15.25** | 9.342 |

* Mean ± standard deviation. Maximum scores are bolded.

The LDA performed faster than OC-SVM in overall execution time, caused by it being a relatively more straightforward algorithm. This would influence the total power consumption in general. Both models result in a nominal size for the Raspberry Pi in terms of memory footprint. The OC-SVM offers a smaller size because it learns only from the original data (unsupervised) instead of finding correlations from the actual labels (supervised).

## 5. Discussion

In this work, both linear models indicated better performance than proximity-based algorithms, as expected. As for the proximity-based algorithms, the CB-LOF attained the best results. It had a relatively good F1-score compared to the rest of the proximity group. The performance differences can be explained by the characteristics of linear models that create a boundary function between classes. In contrast, the proximity-based algorithms use the mean value as a center point for a clustered class. In the dataset, the features within each class prefer a specific boundary function rather than clustered classes. Therefore, our dataset favors linear models.

The F1-score represents the ability of a specific algorithm to classify data points. The score is influenced by how each model approaches the problem. On the other side of the coin, the inference time is directly associated with the computational effort required by the model. Algorithm improvement (e.g., simplified code) can accelerate the model inference regarding the target system constraint or limitation. This improvement also directly impacts power consumption, as the faster the inference, the less power consumed. Further, the model size correlates with the number of training data used in the ML modelling. It varies between the models, but the proximity-based algorithms require more memory to calculate a specific number of neighboring or surrounding data.

In the future, we plan to extend the results of this study in two aspects: First is the battery system analysis, another critical system included in the EPS. The scope of this preliminary work was limited to the solar panel system only. Second, this study examined the ML approach algorithm for semi-sequential types of data points. Therefore, it will be essential to investigate another approach: full-time series analysis. Finally, from the results of this study, we have favorable model candidates to be developed for analytical tools of ground station telemetry data and onboard system model development.

Accordingly, the future system implementation will be developed simultaneously with the CubeSat system. The model has to be adjusted with every satellite sensor calibration process to determine any hyperparameter within the ML models or pre-processing techniques which significantly influences the model's sensitivity to the anomalies. Therefore, the final embedded ML-based anomaly detection has the expected nominal operation in the ground-tested flight model. Ultimately, it could provide critical information immediately after the satellite's deployment and early operation. For example, by having an onboard autonomous anomaly detector, the CubeSat might be able to broadcast its initial condition by CW beacon without the need for the operational team to perform a long traditional analysis of the telemetry data, which potentially takes weeks after the deployment.

## 6. Conclusions

In this work, we analyzed a novel dataset of BIRDS CubeSats. We investigated anomalies in the CubeSat solar panel system through an experimental ML study to compare proximity-based algorithms and linear model algorithms using on-orbit fault data obtained from four different CubeSats. Further, we introduced the windowed averaging method, standardization, and PCA in the pre-processing stage. The evaluation results on F-score metrics indicate that LDA and OC-SVM are the best models for detecting these particular anomalies within the dataset. LDA and OC-SVM models were tested on Raspberry Pi to confirm their technical viability for future onboard implementation. The results confirm their feasibility for onboard applications and prove that LDA outperforms OC-SVM in terms of execution time and power consumption.

**Supplementary Materials:** The following supporting information can be downloaded at https://www.mdpi.com/article/10.3390/app12178634/s1. TSURU.xlsx: TSURU satellite dataset; NEPALISAT.xlsx: NEPALISAT satellite dataset; RAAVANA.xlsx: RAAVANA satellite dataset; UGUISU.xlsx: UGUISU satellite dataset.

**Author Contributions:** Conceptualization, A.J.J.C. and B.H.B.P.; methodology, A.J.J.C.; software, A.J.J.C. and B.H.B.P.; validation, A.J.J.C., B.H.B.P., A.H. and M.C.; formal analysis, A.J.J.C. and B.H.B.P.; investigation, A.J.J.C. and B.H.B.P.; resources, M.C.; data curation, A.J.J.C.; writing—original draft preparation, A.J.J.C.; writing—review and editing, B.H.B.P., A.H. and M.C.; visualization, A.J.J.C. and B.H.B.P.; supervision, M.C.; project administration, M.C.; funding acquisition, M.C. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was partially funded by JSPS Core-to-Core Program B: Asia–Africa Science Platforms (JPJSCCB20200005).

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Not applicable.

**Acknowledgments:** The authors would like to express their appreciation for the valuable comments of the associate editors and anonymous reviewers. This work was supported by BIRDS-3 and BIRDS-4 engineering team members. In addition, the authors would like to acknowledge the support provided by George Maeda, Sankyun Kim, Hirokazu Masui, and Takashi Yamauchi.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. BIRDS Program Digital Textbook CubeSTD-2019-001G. Available online: https://www.birds-project.com (accessed on 10 August 2021).
2. Jara, A.; Bautista, I.; Maeda, G.; Kim, S.; Masui, H.; Yamauchi, T.; Cho, M. An Overview of the BIRDS-4 Satellite Project and the First Satellite of Paraguay. In Proceedings of the 35th Conference on Small Satellites, Logan, UT, USA, 7–12 August 2021.
3. Cho, M.; Graziani, F. *Definition and Requirements of Small Satellites Seeking Low-Cost and Fast-Delivery*; IAA SG-2007-4.18; IAA: Paris, France, 2017.
4. Wertz, J.R.; Everett, D.F.; Puschell, J.J. *Space Mission Engineering: The New SMAD*, 1st ed.; Microcosm Press: Hawthorne, CA, USA, 2011; pp. 45–59.
5. Langer, M.; Bouwmeester, J. Reliability of CubeSats—Statistical Data, Developers' Beliefs and the Way Forward. In Proceedings of the 30th Conference on Small Satellites, Logan, UT, USA, 6–11 August 2016.
6. Maskey, A.; Cho, M. CubeSatNet: Ultralight Convolutional Neural Network designed for on-orbit binary image classification on a 1U CubeSat. *Eng. Appl. Artif. Intell.* **2020**, *96*, 103952. [CrossRef]
7. Wu, J.; Yao, L.; Liu, B.; Ding, Z.; Zhang, L. Combining OC-SVMs With LSTM for Detecting Anomalies in Telemetry Data with Irregular Intervals. *IEEE Access* **2020**, *8*, 106648–106659. [CrossRef]
8. Jin, W.; Sun, B.; Li, Z.; Zhang, S.; Chen, Z. Detecting Anomalies of Satellite Power Subsystem via Stage-Training Denoising Autoencoders. *Sensors* **2019**, *19*, 3216. [CrossRef]
9. Yairi, T.; Takeishi, N.; Oda, T.; Nakajima, Y.; Nishimura, N.; Takata, N. A Data-Driven Health Monitoring Method for Satellite Housekeeping Data Based on Probabilistic Clustering and Dimensionality Reduction. *IEEE Trans. Aerosp. Electr. Syst.* **2017**, *53*, 1384–1401. [CrossRef]
10. Zamry, N.M.; Zainal, A.; Rassam, M.A.; Alkhammash, E.H.; Ghaleb, F.A.; Saeed, F. Lightweight Anomaly Detection Scheme Using Incremental Principal Component Analysis and Support Vector Machine. *Sensors* **2021**, *21*, 8017. [CrossRef] [PubMed]
11. Pan, D.; Liu, D.; Zhou, J.; Zhang, G. Anomaly detection for satellite power subsystem with associated rules based on Kernel Principal Component Analysis. *Microelect. Reliab.* **2015**, *55*, 2082–2086. [CrossRef]
12. Peng, Y.; Jia, S.; Feng, X.; Su, F. Telemetry fault detection for meteorological satellite based on PCA. In Proceedings of the 16th International Symposium on Communications and Information Technologies, Qingdao, China, 8–10 December 2016.
13. Li, J.; Yang, W.; Liu, D.; Liu, J. Kernel Self-Adaptive Learning-Based Satellite Telemetry Data Classification. In Proceedings of the 3rd International Conference on Computing Measurement Control and Sensor Network, Matsue, Japan, 20–22 May 2016.
14. Umezu, R.; Sugie, T.; Nagase, M.; Kokai, R.; Takeshima, T.; Ebisawa, K.; Mitsuda, K.; Yamamoto, Y. Detection of Failure Sign of Spacecraft using Machine Learning. *Space Sci. Inf. Japan* **2019**, *8*, 11–20.
15. BIRDS Open-Source Repository. Available online: https://github.com/BIRDSOpenSource (accessed on 13 March 2022).
16. Chamika, D.; Cho, M.; Maeda, G.; Kim, S.; Masui, H.; Yamauchi, T.; Panawennage, S.; Shrestha, B. BIRDS-3 Satellite Project Including the First Satellites of Sri Lanka and Nepal. In Proceedings of the 70th International Astronautical Congress, Washington, DC, USA, 21 October 2019.
17. Nakayama, D.; Yamauchi, T.; Masui, H.; Kim, S.; Toyoda, K.; Malmadayalage, T.L.D.; Cho, M.; the BIRDS-4 Project Team. On-Orbit Experimental Result of a Non-Deployable 430-MHz-Band Antenna Using a 1U CubeSat Structure. *Electronics* **2022**, *11*, 1163. [CrossRef]

18. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B. Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.
19. Kreyszig, E. *Advanced Engineering Mathematics*, 10th ed.; Wiley: Hoboken, NJ, USA, 2011; p. 880.
20. Murphy, K. *Machine Learning: A Probabilistic Perspective*, 1st ed.; MIT Press: Cambridge, UK, 2012.
21. Minka, T.P. Automatic choice of dimensionality for PCA. In Proceedings of the 2000 Neural Information Processing Systems (NIPS) Conference, Denver, CO, USA, 27 November–2 December 2000; Volume 13, pp. 598–604.
22. Zhao, Y.; Nasrullah, Z.; Li, Z. PyOD: A Python Toolbox for Scalable Outlier Detection. *J. Mach. Learn. Res.* **2019**, *20*, 1–7.
23. Goldstein, M.; Uchida, S. A Comparative Evaluation of Unsupervised Anomaly Detection Algorithms for Multivariate Data. *PLoS ONE* **2016**, *11*, e0152173. [CrossRef] [PubMed]
24. Breunig, M.M.; Kriegel, H.P.; Raymond, T.; Sander, J. LOF: Identifying density-based local outliers. *ACM Sig. Rec.* **2000**, *29*, 93–104. [CrossRef]
25. He, Z.; Xu, X.; Deng, S. Discovering cluster-based local outliers. *Patt. Recog. Lett.* **2003**, *24*, 1641–1650. [CrossRef]
26. Ramaswamy, S.; Rastogi, R.; Shim, K. Efficient algorithms for mining outliers from large data sets. *ACM Sig. Rec.* **2000**, *29*, 427–438. [CrossRef]
27. Duda, R.; Hart, P.; Stork, D. *Pattern Classification*, 2nd ed.; Wiley: Hoboken, NJ, USA, 2000; pp. 215–264.
28. Scholkopf, B.; Platt, J.C.; Shawe-Taylor, J.; Smola, A.J.; Williamson, R.C. Estimating the Support of a High-dimensional Distribution. *Neural Comp.* **2001**, *13*, 1443–1471. [CrossRef] [PubMed]
29. Kluyver, T.; Ragan-Kelley, B.; Pérez, F.; Granger, B.E.; Bussonnier, M.; Frederic, J.; Kelley, K.; Hamrick, J.B.; Grout, J.; Corlay, S.; et al. Jupyter Notebooks—A publishing format for reproducible computational workflows. In *Positioning and Power in Academic, Players, Agents and Agendas*; IOS Press: Amsterdam, The Netherlands, 2016; pp. 87–90.
30. The Python Library Reference, Release 3.8.8, Python Software Foundation. Available online: https://www.python.org/downloads/release/python-388/ (accessed on 13 March 2021).
31. Harris, C.R.; Millman, K.J.; van der Walt, S.J. Array programming with NumPy. *Nature* **2020**, *585*, 357–362. [CrossRef] [PubMed]
32. McKinney, W. Data Structures for Statistical Computing in Python. In Proceedings of the 9th Python in Science Conference, Austin, TX, USA, 28 June 2010.
33. Virtanen, P.; Gommers, R.; Oliphant, T.E. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nat. Methods* **2020**, *17*, 261–272. [CrossRef]
34. Hunter, J.D. Matplotlib: A 2D Graphics Environment. *IEEE Comput. Sci. Eng.* **2007**, *9*, 90–95. [CrossRef]
35. Reiinakano/Scikit-Plot: V0.3.5. Available online: https://zenodo.org/record/1245853#.Yogp4ujMJPY (accessed on 13 March 2021).
36. Waskom, M.L. Seaborn: Statistical Data Visualization. *J. Open Source Softw.* **2021**, *6*, 3021. [CrossRef]
37. Manning, C.D.; Raghavan, P.; Schütze, H. *Introduction to Information Retrieval*, 1st ed.; Cambridge University Press: Cambridge, UK, 2008; pp. 151–175.
38. Sokolova, M.; Lapalme, G. A Systematic Analysis of Performance Measures for Classification Tasks. *Inf. Process. Manag.* **2009**, *45*, 427–437. [CrossRef]