informatics

Article

# Web Traffic Anomaly Detection Using Isolation Forest

Wilson Chua [1], Arsenn Lorette Diamond Pajas [2], Crizelle Shane Castro [2], Sean Patrick Panganiban [2], April Joy Pasuquin [2], Merwin Jan Purganan [2], Rica Malupeng [2], Divine Jessa Pingad [2], John Paul Orolfo [2], Haron Hakeen Lua [3,*] and Lemuel Clark Velasco [3,4,5]

[1] Analytics Division, Future Gen International Pte. Ltd., Singapore 534260, Singapore; wilson.chua@gmail.com
[2] Department of Information Technology, College of Information Technology and Engineering, International School of Asia and the Pacific, Peñablanca, Cagayan 3502, Philippines; diamondmoncespajas@gmail.com (A.L.D.P.); shanecastro620@gmail.com (C.S.C.); seanpatrickpanganiban08@gmail.com (S.P.P.); pasuquinapriljoy@gmail.com (A.J.P.); purgananmerwinjan@gmail.com (M.J.P.); ricamalupeng2424@gmail.com (R.M.); divinepingad4@gmail.com (D.J.P.); paulj7521@gmail.com (J.P.O.)
[3] Department of Information Technology, College of Computer Studies, Mindanao State University-Iligan Institute of Technology, Iligan City 9200, Philippines; lemuelclark.velasco@g.msuiit.edu.ph
[4] Department of Industrial and Information Management, College of Management, National Cheng Kung University, Tainan City 701, Taiwan
[5] Center for Computational Analytics and Modelling, Premier Research Institute of Science and Mathematics, Mindanao State University-Iligan Institute of Technology, Iligan City 9200, Philippines
[*] Correspondence: haronhakeen.lua@g.msuiit.edu.ph

**Abstract:** As companies increasingly undergo digital transformation, the value of their data assets also rises, making them even more attractive targets for hackers. The large volume of weblogs warrants the use of advanced classification methodologies in order for cybersecurity specialists to identify web traffic anomalies. This study aims to implement Isolation Forest, an unsupervised machine learning methodology in the identification of anomalous and non-anomalous web traffic. The publicly available weblogs dataset from an e-commerce website underwent data preparation through a systematic pipeline of processes involving data ingestion, data type conversion, data cleaning, and normalization. This led to the addition of derived columns in the training set and manually labeled testing set that was then used to compare the anomaly detection performance of the Isolation Forest model with that of cybersecurity experts. The developed Isolation Forest model was implemented using the Python Scikit-learn library, and exhibited a superior Accuracy of 93%, Precision of 95%, Recall of 90% and F1-Score of 92%. By appropriate data preparation, model development, model implementation, and model evaluation, this study shows that Isolation Forest can be a viable solution for close to accurate web traffic anomaly detection.

**Keywords:** anomaly detection; web traffic; web traffic anomaly detection; machine learning; isolation forest

check for updates

**Citation:** Chua, W.; Pajas, A.L.D.; Castro, C.S.; Panganiban, S.P.; Pasuquin, A.J.; Purganan, M.J.; Malupeng, R.; Pingad, D.J.; Orolfo, J.P.; Lua, H.H.; et al. Web Traffic Anomaly Detection Using Isolation Forest. *Informatics* **2024**, *11*, 83. https://doi.org/10.3390/informatics11040083

Academic Editors: Antony Bryant and Gian Antonio Susto

Received: 5 July 2024
Revised: 29 October 2024
Accepted: 30 October 2024
Published: 5 November 2024

## 1. Introduction

With stricter data privacy laws now in place, security breaches have become costlier for website owners due to penalties for data breaches, reputation loss and lost revenues from outages caused by such attacks. Thus, there is a pressing need for website owners and security practitioners to quickly identify any anomalous web traffic that might lead to data breaches early [1–5]. Researchers analyzed web traffic data from weblogs, recognizing that both normal and malicious activities leave digital breadcrumbs [2,3]. However, analyzing such weblogs for anomalies poses several challenges as traditional defenses like network and web server firewalls are becoming increasingly ineffective as hackers focus more on web-based attacks [1,3,6,7]. Additionally, hackers frequently obfuscate their methods to avoid detection. The sheer volume of weblogs from even moderately trafficked websites also makes manual analysis impractical [3,5]. Machine learning offers the potential for productivity gains by accelerating the identification of anomalous traffic [3–6]. However, web traffic data typically exhibits significant class imbalance, with anomalies being a small fraction of the total traffic. These deviations, or anomalies, can be categorized as either benign or malicious [2,8]. Benign anomalies represent harmless fluctuations in traffic patterns. Seasonal variations in user activity, for instance, can lead to the emergence of benign anomalies. Conversely, malicious anomalies signal potentially harmful activities; these could be cyberattacks seeking to exploit vulnerabilities in website security, fraudulent activities aiming to deceive or manipulate users, or even deliberate attempts to disrupt website functionality altogether. Malicious actors often attempt to mask their activities within the flow of regular web traffic, making anomaly detection a crucial tool for safeguarding online security [1,4,8]. By delving into the intricate patterns woven within this web traffic data, web traffic analysis not only facilitates the optimization of website performance but also plays a vital role in ensuring online security and anomaly detection.

Prior research has explored web traffic through a variety of lenses. For instance, the use of stack architecture was examined to combine classifier ensembles for detecting anomalies in a web application, demonstrating enhanced anomaly detection in web traffic [4]. Further research provided a comprehensive survey of web traffic anomaly detection techniques, encompassing statistical methods, machine learning approaches, and hybrid methodologies [1–3,5,8]. In the rapidly growing realm of web traffic, it is essential to implement advanced methods for detecting anomalies. While traditional rule-based approaches are effective in identifying known threats, they encounter challenges in keeping up with the constantly evolving tactics employed by malicious actors. Machine learning offers a promising approach to address this challenge. By continuously learning from vast amounts of web traffic data, machine learning models can identify patterns and anomalies that may indicate malicious activity, even if they employ novel tactics [4,6]. This continuous learning capability empowers machine learning to adapt to the dynamic threat landscape, making it a powerful tool for web traffic anomaly detection. A study investigated the use of unsupervised learning techniques, specifically Extended Isolation Forests, for anomaly detection in network traffic data, and findings suggest that unsupervised methods offer a compelling approach, particularly in scenarios where labeled attack data might be limited [7]. Despite the significant strides made in machine learning for web traffic anomaly detection, several challenges and research gaps continue to hinder its full potential; a critical hurdle lies in the inherent imbalance of web traffic data.

Among the various machine learning techniques, Isolation Forest (IF) has emerged as a compelling approach for web traffic anomaly detection due to its unique strengths. Unlike supervised learning methods, which require labeled data for training, Isolation Forest leverages an unsupervised learning paradigm, making it particularly well-suited for scenarios where labeled anomaly data might be scarce [7,9,10]. With this, Isolation Forest has great potential to help network and security administrators in bridging the security gap as it has proven useful in analyzing large datasets due to its speed and ability to generate models based on subsampling of the dataset [10–14]. Despite the usage of Isolation Forest in network traffic anomaly detection, Isolation Forest has not yet been widely implemented to detect web traffic anomalies. Thus, this research aims to enhance current techniques, trends, and methods in anomaly detection, particularly in the context of web traffic, through the implementation of Isolation Forest. The findings of this study hope to aid cybersecurity practitioners and data scientists with valuable insights into the accurate and precise detection of web traffic anomalies using machine learning, particularly Isolation Forest. The detection of these anomalies will also help network and website administrators identify hidden vulnerabilities within the network infrastructure as well as develop countermeasures to prevent such anomalies from occurring. This study seeks to answer three research questions:

- What are the necessary data and parameters for Isolation Forest, and how are they prepared for web traffic anomaly detection?

- How is Isolation Forest implemented in web traffic anomaly detection?
- What is the performance of Isolation Forest in detecting anomalies in web traffic?

## 2. Methodology

This study aims to identify anomalous and non-anomalous web traffic by implementing Isolation Forest and evaluate its performance using metrics such as Accuracy, Precision, Recall, and F1-score by utilizing a publicly available network traffic dataset containing both normal and anomalous traffic patterns [15]. As outlined in Figure 1, there are three phases of this study, namely: Data Preparation, Model Generation, and Model Evaluation.

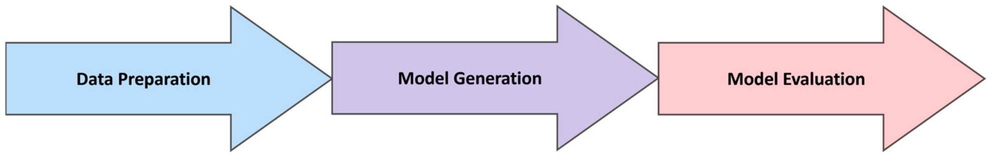


**Figure 1.** Overview of the research design.

### 2.1. Web Traffic Data Preparation

Web traffic data used for anomaly detection needs appropriate data preparation to ensure the integrity of inputs for the Isolation Forest model. The dataset that was used for this study was the 2019 Online Shopping Store—Web Server Logs, Harvard Dataverse, V1 by Farzin Zaker, which the researchers acquired from the Kaggle website [9,16]. As shown in Table 1, the 3.3 GB dataset containing approximately 10 million rows in the native log format acquired from Kaggle are Nginx server access logs from the Iranian e-commerce website zanbil.ir.

**Table 1.** Preview of the web traffic dataset used in this study [16].

```
40.77.167.129—[22/Jan/2019:03:56:18 +0330] “GET /image/57710/productModel/100x100
HTTP/1.1” 200 1695 “-” “Mozilla/5.0 (compatible; bingbot/2.0;
+http://www.bing.com/bingbot.htm)” “-”
```

Normal web traffic represents the typical patterns and behaviors observed during interactions between users and web servers. Normal traffic usually shows consistent behaviors typical of regular web applications. Although outliers deviating from regular web traffic should not be automatically classified as anomalous, they have been a harbinger of later successful attacks [1,4,5,17]. These also have a higher likelihood of potential malicious activity and warrant further investigation. Anomaly detection using Isolation Forest aims to identify data points in a dataset that significantly differ from expected behavior [9,10,12]. These anomalies require further investigation, as they could signal impending issues, fraud, or other important findings. Table 2 shows that the web server logs used in this study detailed information for each site visit or interaction, including visitor details, request methods, protocol, user agents, referrer, site Uniform Resource Identifier (URI), bytes, and response codes.

**Table 2.** Schema of the raw dataset used in this study.

| Column | Data Type | Column | Data Type | Column | Data Type |
|---|---|---|---|---|---|
| Srcip | object | URIs | category | Bytes | int32 |
| Timestamp | object | Protocol | category | Referrer | category |
| Method | category | Status | category | User-Agent | category |

The researchers devised a systematic process as a pipeline that covers the various stages of data ingestion, data type conversion, and data cleansing as part of the data preparation and cleansing stage. After ingesting the dataset, Figure 2 shows that the researchers extracted two sets of data (1): one set for training the model, and another to be used as a testing dataset [18]. For data preparation, the researchers sampled 25% of the weblogs to act as the training dataset. As part of the data cleaning, the researchers then deleted by dropping any rows with missing data (2). For the training set, the researchers set the parameter max_sampling = 25%, which enabled the project to work on a 25% subset of the full dataset. The researchers anticipated that swamping and masking could possibly occur in this instance [12,19,20]. Swamping is the phenomenon where normal data points are incorrectly labeled as anomalies or false positives, while masking occurs when actual anomalies are incorrectly labeled as normal data points or false negatives [12,20]. With this, a small sample size yields better iTrees because the swamping and masking effects are decreased [13,21,22]. The cleaning process initially started with the researchers manually evaluating and deleting the rows with missing data, then followed the web traffic dataset cleaning protocols. The data cleaning procedure for the dataset was implemented using Python code augmented in Scikit in order to improve the quality of the data for machine learning by handling different data types appropriately. The researchers noted that the weblog data was in Apache log format and not CSV. Thus, the process of converting the log format to the CSV table format was anticipated to leave some artifacts that researchers need to correct. This cleansing and preparation action was resolved in order to prevent Isolation Forest from throwing an exception when it encounters missing data or incorrect data type.

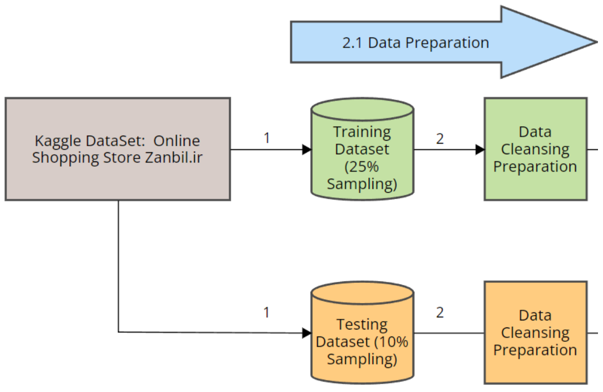

2.1 Data Preparation
Kaggle DataSet: Online Shopping Store Zanbil.ir
1
Training Dataset (25% Sampling)
2
Data Cleansing Preparation
1
Testing Dataset (10% Sampling)
2
Data Cleansing Preparation

**Figure 2.** Pipeline of the data preparation processes used in the study.

## *2.2. Isolation Forest Model Implementation*

After ensuring the integrity of the dataset through appropriate data preparation procedures conducted for the web traffic dataset, the implementation of the Isolation Forest as an unsupervised machine learning model is the next step in the pipeline established by the researchers. Figure 3 shows that feature generation (3) resulted in the addition of generated features—namely, URI_occurences, IOC_occurences, User-Agent_occurences, URI_length, and UserAgentLength that were derived from the URI, User-Agent, and Referrer columns. These generated features were then added to the input along with the Source IP (Srcip) Address, Timestamp, Method, URIs, Protocol, Status, Bytes, Referrer, and User-Agent. The User-Agent string and the URI string were initially analyzed using a generated feature set designed to detect anomalous character sets and occurrences of indicators of compromise (IoCs). Detection of an IoC in either the User-Agent string or the URI can trigger an anomaly, as malicious indicators are typically absent from legitimate User-Agent strings or present in the URI. A URI is a string of characters that identifies a name or a resource on the Internet, enabling interaction with these resources over a network [3,23]. Furthermore, a Uniform Resource Locator (URL) is a subset of the URI that specifies the location of an identified resource and the mechanism for retrieving it, such as through Hypertext Transfer Protocol (HTTP), File Transfer Protocol (FTP), among other protocols. URLs are then more specific than URIs and point to a final destination such as a specific web page or file. By generating features out of these fields along computed IoC values, Isolation Forest is better able to detect potential threats, focusing particularly on the user-agent strings as well as URI strings associated with network traffic or login attempts.

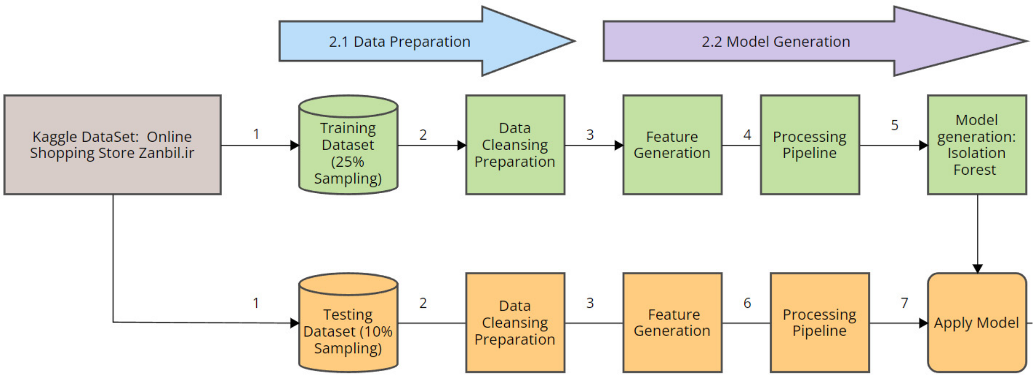

2.1 Data Preparation
2.2 Model Generation
Kaggle DataSet: Online Shopping Store Zanbil.ir
Training Dataset (25% Sampling)
Data Cleansing Preparation
Feature Generation
Processing Pipeline
Model generation: Isolation Forest
Testing Dataset (10% Sampling)
Apply Model
1
2
3
4
5
6
7

**Figure 3.** Pipeline of the data preparation and model generation processes used in the study.

The data processing pipeline (4) was implemented in Python using Scikit-learn package and then converted the non-numeric object data type variables into numerical values using OneHotEncoder(). The researchers initially opted for OneHotEncoding over LabelEncoding in the 'bag of words' features due to the reason that each word creates a unique "one-hot" dimension where HTTP sessions containing the word are marked with 1, while others are marked with 0 [13,21]. Unfortunately, most of the attacks use nonsensible words that are not in the 'bag of words'. The numeric variables had StandardScaler() processes applied by the researchers to normalize the numeric data and avoid any large values from creating any implicit bias. This pipeline holds both the StandardScaler and the OneHotEncoder with StandardScaler, which is used to center and scale all the numeric features while the OneHotEncoder for categorical features was used as a standardizing feature to focus on relative differences. The ColumnTransformer class is then used for different transformations, with the initial transformation adjusting numerical data to a mean of 0 and standard deviation of 1.

After the post-processing pipeline data, the researchers then generated the Isolation Forest model (5). After the generated Isolation Forest model was saved, the researchers then applied the same pipeline processing to the testing dataset (6). Then, the generated Isolation Forest model was used to apply against the post-pipeline testing dataset in (7). In the Apply the Model (7), two additional fields were then created, namely, Predict and Anomaly_Score. The Predict field represents the model's predictions or classifications for the data instances in the testing set with values of either −1 for the outlier or +1 for the inlier. These predictions represent the model's best estimate of the result or label for each data point. Conversely, the Anomaly_Score column shows the level of abnormality or divergence of each data point from the expected patterns based on the model's training. The definition of isolation in this study means separating an instance from the rest of the instances. In other words, isolation-based techniques measure each instance's specific sensitivity to isolation, with anomalies showing the highest susceptibility [10,11,20]. Thus, this study's execution of isolation was conducted with the weblog data using Isolation Forest. Specifically, the implementation of Isolation Forest is based on the established principles of random partitioning implemented by past researchers, where data is recursively split using randomly chosen attributes and values to create Isolation Trees [9–13,20,21]. The path length in these trees, indicating the number of partitions needed to isolate anomalies, serves as a measure of anomaly detection [11,15,20,24]. The principle of Isolation Forest is that outliers are abnormal instances that deviate from the pattern of the majority of dataset instances. These outliers are located closer to the root of the tree due to random partitioning, hence, it yields trees with shorter paths. If an instance is located along a shorter branch of the tree, then it is an outlier [18]. For this study, the anomalies that were identified by shorter path lengths are assigned higher anomaly scores due to their greater ease of isolation from normal data points within the weblog dataset.

The researchers then applied several parameters to the Isolation Forest algorithm. The value of 0.03 was set as the 'contamination' parameter. Though contamination is usually set at 0.10, for this study, which is dependent on the nature of the weblog dataset, initial results found out that contamination is at 0.0169 or less than 1.6%, leading the researchers to a contamination of 0.03 or 3%. This parameter is sensitive, and the results vary according to how close we can approximate the actual outlier rate [11,25–27]. In order to ensure reproducibility, the 'random_state' parameter, which enables the same outcomes to be obtained across numerous executions, was set to 42. Furthermore, to achieve multiprocessing that will increase the analysis' speed and scalability, 'n_jobs = 16′ was implemented to make full use of the computer's processing power of 16 cores. The max_samples was then set to 0.25, which should result in a sampling of about 1/4 of the 10 million records or about 2.5 million rows. The reduction in size resulted in faster processing without unduly affecting the performance.

## 2.3. Isolation Forest Model Evaluation

After implementing the Isolation Forest model with the prepared weblog dataset, model evaluation was then conducted to measure the classification power of the model in detecting web anomalies. Figure 4 shows that in the model evaluation and visualization process (8), the testing dataset now has a total of 17 columns after the Isolation Forest is applied. The additional columns include Predict and Anomaly_Score along with two more fields for manual human scoring, namely, Human Rating and Attack Type. These two fields were then used by the researchers in the model evaluation phase as the basis for computing model performance where Human Rating contained the ratings of human scorers as to whether the row represented an anomaly, represented by −1, or was normal, represented by +1. Additionally, Attack Type contains specific attack types like SQL injection, cross-site scripting, probe, and connect tunnel.

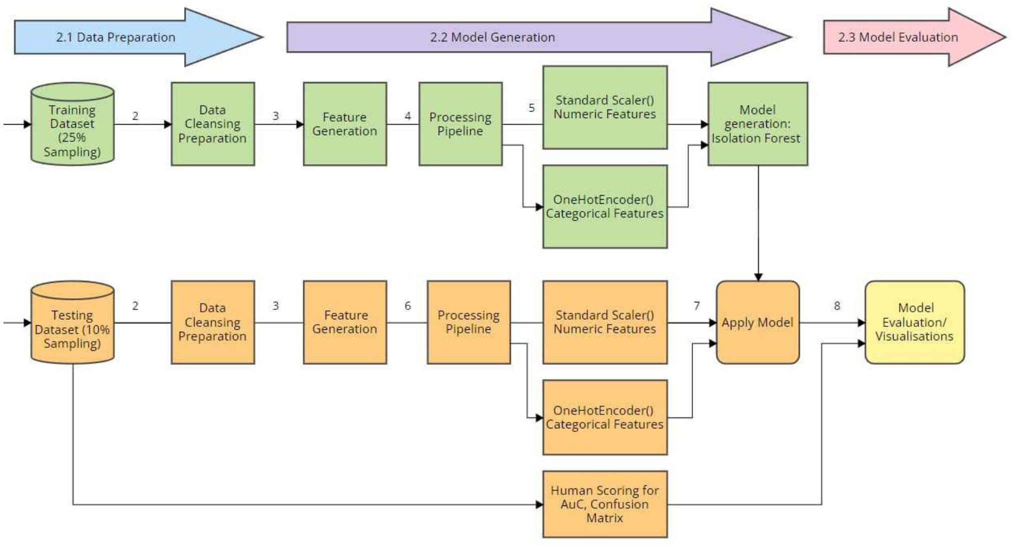
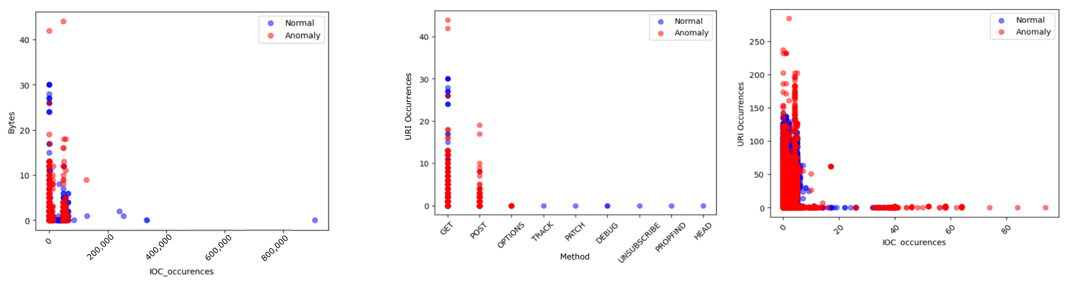

2.1 Data Preparation
2.2 Model Generation
2.3 Model Evaluation
Training Dataset (25% Sampling)
2
Data Cleansing Preparation
3
Feature Generation
4
Processing Pipeline
5
Standard Scaler() Numeric Features
OneHotEncoder() Categorical Features
Model generation: Isolation Forest
Testing Dataset (10% Sampling)
2
Data Cleansing Preparation
3
Feature Generation
6
Processing Pipeline
Standard Scaler() Numeric Features
OneHotEncoder() Categorical Features
7
Apply Model
8
Model Evaluation/ Visualisations
Human Scoring for AuC, Confusion Matrix

**Figure 4.** Pipeline of the data preparation, model generation, and model evaluation processes used in the study.

Recognizing that humans process visual data more easily, the researchers also plotted the predicted outcomes (8) based on several features to enhance comprehension and insight into the Isolation Forest model's performance and behavior. Figure 5 shows that the researchers plotted the predicted outcomes based on several features to enhance comprehension and insight into the Isolation Forest model's performance and behavior. Along with this, the researchers also used scatter diagrams (7) and a confusion matrix, noting that a high anomaly score indicates that the data points are more likely to be outliers or uncommon when compared to the norm.



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

**Figure 5.** (**a**) Anomalies in IOC occurrences in bytes. (**b**) Anomalies in URI occurrences in method. (**c**) Anomalies in IOC occurrences in User-Agent.

In the model evaluation phase, the researchers determined how accurately the Isolation Forest model performs in identifying web traffic anomalies by minimizing false positives and false negatives. This analysis is important as it shows how effective Isolation Forest is as an algorithm for web traffic classification by accurately classifying anomalous and non-anomalous traffic. Though Accuracy is a crucial metric, as it represents the model's ability to guess abnormal or normal traffic, it may overestimate performance in imbalanced datasets. For instance, if a model tends to predict the majority of classes, the Accuracy metric can yield a relatively high value even if the minority of categories are completely ignored, thus making such a metric insensitive to the categorization performance of the minority categories, leading to bias in evaluation. Moreover, the Accuracy metric may fail to recognize imbalances, even if a model's classification performance for a minority of categories is poor, when the accuracy for the majority of categories is high [28]. Thus, for this study, other metrics like Precision, Recall, and F1-score are used along with Accuracy as they show varying facets of classification performance that can be used to evaluate the Isolation Forest implementation [15,21,24,27,29].

Accuracy measures a model's ability to predict outcomes [14,21,30,31]. Equation (1) shows how accuracy is calculated, where True Positive (TP) counts instances where the model correctly identifies web traffic anomalies, True Negative (TN) counts instances where the model correctly identifies normal instances, False Positive (FP) counts instances where the model incorrectly identifies anomalies, and False Negative (FN) counts instances where the model fails to identify actual anomalies.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{1}$$

While Accuracy measures how correct the model's predictions are overall, Precision calculates the proportion of true positive predictions of web traffic anomalies out of all the positive predictions made by the Isolation Forest model [14,21,32,33]. Equation (2) shows that high precision indicates when the model predicts the positive class.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{2}$$

Recall measures how well the Isolation Forest model can correctly identify all the actual positive instances of the web traffic anomaly out of the total number of positive instances in the weblog dataset [14,21,30,32,33]. Equation (3) shows the formula for Recall, calculated as the ratio of true positive predictions to the total number of actual anomalies.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3}$$

The F1-score combines Precision and Recall into a single value to offer a balanced assessment of a system's performance, which is especially beneficial in scenarios with imbalanced class distributions [14,21,31,32]. Equation (4) shows that the F1-score harmonizes the trade-offs between Precision and Recall, hence it is calculated using the harmonic mean of the two quantities. It provides a consolidated metric to evaluate the system's overall effectiveness in uneven handling.

$$\text{F1 Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4}$$

## 3. Results and Discussions

### *3.1. Web Traffic Data Preparation Results*

To properly prepare the training dataset for model implementation, the researchers needed to ensure that Isolation Forest could work with the weblog dataset. With this, Isolation Forest requires the data to be non-null and an integer data type; thus, all rows with null values were deleted from the training dataset. All non-numerical fields were converted into numeric values, which are mostly held in the object data type in the Data Type (D-type) column. This signifies that it can hold various data types, unlike columns with numeric types *int64I*, which are restricted to numbers. Table 3 shows that for the training dataset, the standard given by the weblog dataset was in columns 1 to 8, and only column 7 for Bytes was numeric for Isolation Forest's usual implementation [9]. With this, the researchers used feature engineering to generate derived features along with their generated computations in columns 10 to 13 (highlighted in light gold), namely: URI_occurrences, IOC_occurences, User-Agent_occurences, and URI_length. The researchers included the *int64* to StandardScaler from the numeric column, and for the rest of the D-type, they were included in OneHotEncoding.

**Table 3.** Schema of the prepared training dataset used in this study.

| # | Column | Data Type | # | Column | Data Type | # | Column | Data Type |
|---|---|---|---|---|---|---|---|---|
| 1 | Srcip | Object | 5 | Protocol | category | 9 | User-Agent | category |
| 2 | Timestamp | Object | 6 | Status | category | 10 | URI_occuren | int64 |
| 3 | Method | Category | 7 | Bytes | int32 | 11 | IOC_occuren | int64 |
| 4 | URIs | Category | 8 | Referrer | category | 12 | User-Agent_occure | int64 |
|  |  |  |  |  |  | 13 | URI_length | int64 |

Table 4 shows that there were additional columns, namely, Attack Type, Human Rating, Predict, and Anomaly_Score (highlighted in green). The researchers then applied the testing dataset to the same cleaning and feature generation procedures that it used for the training dataset. Furthermore, the researchers applied the Isolation Forest model to the testing dataset and saved the results in columns named Predict and Anomaly_Score, respectively. The resulting dataset output was then saved as a CSV file.

**Table 4.** Schema of the prepared testing dataset used in this study.

| # | Column | Data Type | # | Column | Data Type | # | Column | Data Type |
|---|---|---|---|---|---|---|---|---|
| 1 | Srcip | object | 6 | Status | Category | 12 | User-Agent_occure | int64 |
| 2 | Timestamp | object | 7 | Bytes | int32 | 13 | URI_length | int64 |
| 3 | Method | category | 8 | Referrer | Category | 14 | AttackType | object |
| 4 | URIs | category | 9 | User-Agent | Category | 15 | Human Rating | object |
| 5 | Protocol | category | 10 | URI_occuren | int64 | 16 | Predict | int32 |
|  |  |  | 11 | IOC_occuren | int64 | 17 | Anomaly_Sc | int32 |

For the training dataset and testing dataset, the researchers used statistical sampling of 25% and 10%, respectively. Since Isolation Forest is unsupervised, the researchers did not formally partition the dataset, but instead used random sampling tests [11,15,24]. In this study, the weblog dataset was partitioned according to the non-null count and D-type of training and testing. As shown in Table 5, the weblog dataset that was used for the Isolation Forest model implementation made up 25% or 2,591,287 records for the training dataset and 10% or 1,035,020 for the testing dataset. This means that more than 1 million records in the testing dataset were scored by human raters to compare the actual and the machine-learning-classified web traffic anomalies. The datasets were then normalized and scaled in order to prepare for model generation. Normalization was conducted through the use of the OneHotEncoding and StandardScaler, where data was scaled into a range index of 0 to 10,365,151 for the testing dataset and a range index of 0 to 10,365,151 for the testing dataset.

**Table 5.** The datasets that were used for the Isolation Forest model implementation after data preparation.

| | **Percentage** | **Number of Records** |
|---|---|---|
| Training Set | 25% | 2,591,287 |
| Testing Set | 10% | 1,035,020 |

This study shows nearly similar data preparation results from the network anomaly classification using Isolation Forest [12–15,21]. Although the use of OneHotEncoding and StandardScaling is common for data preparation before Isolation Forest implementation, this study made use of human label encoding of the testing dataset that can be used to compare how actual cybersecurity experts classify web traffic anomalies from that of machine learning classification. Another difference that this study has explored is the derivation of additional columns that were added to both the training and testing datasets. With the researchers' expertise in actual web traffic anomaly detection, these derived columns were seen as helpful for Isolation Forest implementation.

### *3.2. Isolation Forest Model Implementation and Evaluation Results*

Classification metric scores were used to understand a balanced analysis of the Isolation Forest model implementation in web traffic anomaly detection. Table 6 presents a comparison of Accuracy, Precision, Recall, and F1-score that was calculated from the testing dataset. The Isolation Forest model achieves an Accuracy of 93% and a Precision of 95%, indicating how well the model predicts positive cases. Recall, which focuses on the true positive rate, is 90%, which measures the proportion of actual positive cases correctly identified. The F1-score is 92%, which balances Precision and Recall.

**Table 6.** Classification metric scores of the Isolation Forest mode.

| **Accuracy** | **Precision** | **Recall** | **F1-Score** |
|---|---|---|---|
| 0.93 | 0.95 | 0.90 | 0.92 |

Figure 6 shows the confusion matrix that provides additional insight into the performance of the Isolation Forest model in terms of its generated true positives, true negatives, false positives, and false negatives. True positive results show that 90.2% of the predictions were correctly classified while false negative results show that in 9.8% of instances where the actual class was positive, the model predicted it as negative. Additionally, false positive results show that in 4.5% of instances where the actual class was negative, the model predicted it as positive, while true negative results show that 95.5% of the predictions were correctly classified as negative by the Isolation Forest model. This model performance may seem average for data scientists and machine learning experts, but as a detector of web traffic anomalies, the confusion matrix already shows above-average and excellent performance for cybersecurity and network security applications [2,5,6,14,17,18,27,29,34,35]. This is due to the reason that this machine learning performance is already enough for the cybersecurity expert to detect possible web traffic attackers. For the exhibited Accuracy, Precision, Recall, and F1-score of the Isolation Forest model, cybersecurity experts can already match the Source IP Address (Srcip) of the attackers and uncover what the model failed to pick up. Additionally, finding all the traffic of these attackers will further surface the rest.

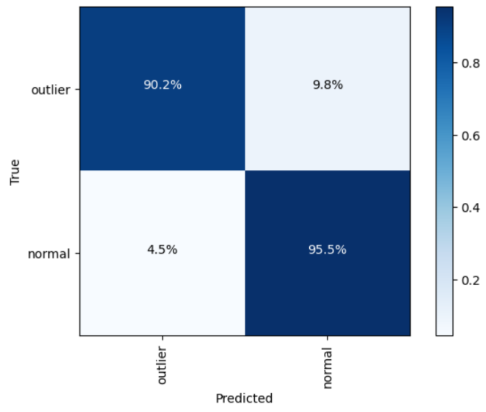


**Figure 6.** Confusion matrix of the Isolation Forest model implementation.

Figure 7 further shows the 3D presentation of the normal or inlier and anomalous or outlier data. Each dot corresponds to specific data instances of this study, which can be a web request or user session, with red points indicating predicted anomaly instances while blue points represent predicted normal instances. The isolated nature shown is due to abnormalities with some of the blue dots mixed with the isolated red dots.

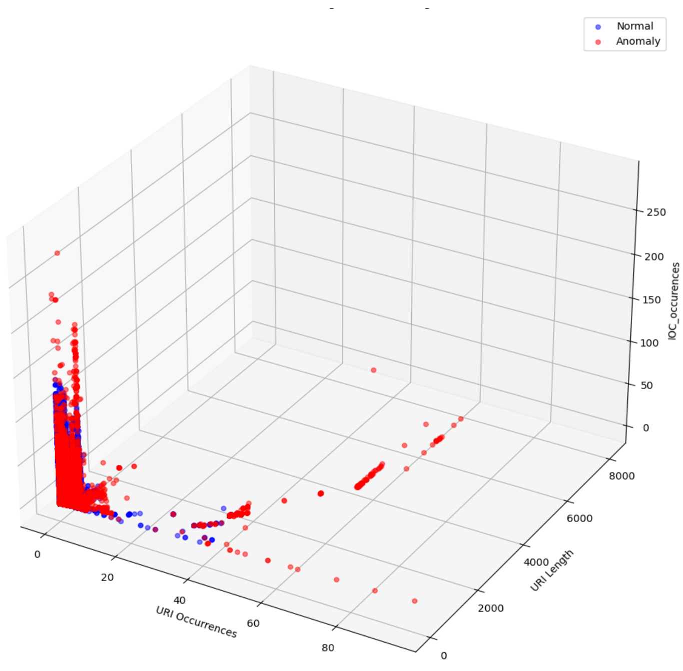


**Figure 7.** A 3D plot of URI_Occurrences, URI_Length, and User-Agent_Occurrences.

Evaluating the performance of Isolation Forest model implementation for web traffic anomaly detection can also be assessed based on how well it performs on applications for almost similar use cases. As shown in Figure 8, in comparison to other papers using Isolation Forest, the implemented methodology of this study shows comparable performance and faster processing time [21,22]. A comparable study that also normalized the datasets containing numerical and nominal values then implemented Isolation Forest exhibited an Accuracy of 95.4%, Precision of 94.81%, Recall of 97.25%, and F1-score of 96.01% [21]. A similar study solely implemented Isolation Forest and compared Isolation Forest with kernel principal component analysis (KPCA) [22]. The study that processed the dataset comprising 307,510 rows and 122 columns exhibited an Accuracy of 87.3%, Precision of 93.7%, Recall of 92.5%, and F1-score of 93.1% for Isolation Forest with KPCA implementation. For sole Isolation Forest model implementation, it recorded an Accuracy of 80.9%, Precision of 86.0%, Recall of 85.5%, and F1-score of 85.2%. This comparison of performance shows that the study has optimally implemented the Isolation Forest model for web traffic anomaly detection. Also, since Isolation Forest works well with highly imbalanced data, it offers a promising alternative to traditional machine learning techniques and ensemble methods [18].

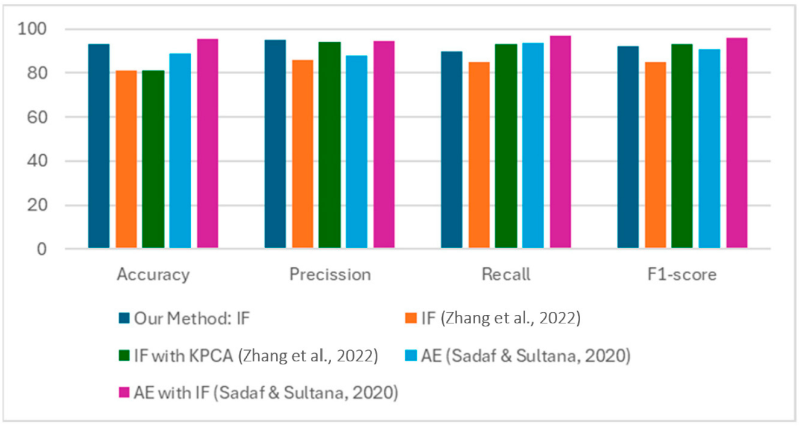


**Figure 8.** This study's comparison of performance with other Isolation Forest model implementations [21,22].

Furthermore, the total time to apply the model represents the duration taken by the model to apply its algorithms and yield results. For this study, the time recorded is 23.620082 s, which is a relatively quick response that implies that model efficiency that is crucial for swift predictions can lead to more timely decisions in web traffic anomaly detection [6,32,34,36]. However, it is equally important to remember that the speed of the model application should not compromise the quality of results. Balancing both speed and accuracy remains a core challenge, not just for Isolation Forest implementation but also in the general field of data analysis and machine learning.

Using the Isolation Forest model for detecting anomalies in web traffic involves data preparation, model implementation, and performance evaluation. The Kaggle dataset was split into samples of 25% for training and 10% for testing. Non-numeric fields like Method, Status, and Protocol were encoded using OneHotEncoding, while additional features were derived for fields such as URI, Referrer, and User-Agent (e.g., User-agent string length). Despite being unsupervised, the testing dataset, comprising approximately 1 million records, was manually labeled for validation among approximately 2.5 million rows. The model achieved an Accuracy of 0.93, Precision of 0.95, Recall of 0.90, and F1-Score of 0.92, with an execution time of 23.620082 s.

This study solely implements the Isolation Forest model without the use of other anomaly detection methods such as Principal Component Analysis (PCA), Autoencoder, and Ensemble, among others. Moreover, this study only deals with anomaly detection from web traffic, which is composed of HTTP data, and does not capture data below the application layer, such as transport layer segments or network layer packets.

## 4. Conclusions and Recommendations

This study explores the development, implementation, and evaluation of Isolation Forest in the detection of anomalies in web traffic. Data preparation was conducted on a publicly available dataset through a systematic pipeline involving data ingestion, data type conversion, and data cleansing stages to ensure optimal model implementation. This resulted in various derived columns that were added to the training dataset to process the 2,591,287 records and additional derived and labeled columns in the testing set to process the 1,035,020 records of weblogs. The implemented unsupervised Isolation Forest model exhibited an outstanding Accuracy of 93%, Precision of 95%, Recall of 90%, and F1-Score of 92%. The results show that Isolation Forest can greatly aid network administrators/defenders and security researchers in quickly identifying anomalies in web traffic. These anomalies, in turn, can point defenders to likely attacks that are launched against their websites. Considering the faster speed of execution, which is just 23.620082 s, the Isolation Forest model implemented in this study also enables quicker reaction and response times for network defenders. This shorter reaction time, in turn, helps to reduce the likelihood of a successful breach as most attack vectors can be neutralized once they are identified.

Future work on Isolation Forest model implementation could improve its predictive abilities further by mapping the weblog traffic to recently published Common Vulnerabilities and Exposures (CVEs). Understanding the unique fingerprint of the new CVEs and incorporating them into feature generation could increase the optimal machine learning model performance of Isolation Forest. Future studies could also explore Isolation Forest's efficacy in detecting zero-day attacks, as its ability to highlight rare occurrences can potentially facilitate their detection amid normal traffic patterns. Incorporating CVEs for detecting new threats allows for the integration of the Isolation Forest algorithm in network and website management tools and intrusion prevention systems because it allows these systems to proactively recognize traffic associated with recent exploits, patch gaps, and specific attack vectors. This will enhance the monitoring, evaluation, and improvement of cybersecurity measures for website owners. Integrating other machine learning techniques with Isolation Forest could also be explored to further optimize classification performance. Also, to set a comparative performance for Isolation Forest in web traffic anomaly detection, methods such as synthetic minority oversampling technique (SMOTE), Local Outlier Factor, Support Vector Machine (SVM), and other ensemble methods can be explored. With this, future research on the use of machine learning in anomaly detection should focus on expanding and refining Isolation Forest methodologies on web traffic applications in order to provide enhanced insights and significant tools for both cybersecurity practitioners and data scientists.

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

**Funding:** The APC was funded by the Mindanao State University-Iligan Institute of Technology under the Office of Research Management of the Office of the Vice Chancellor for Research and Enterprise.

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

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** This study used the publicly available dataset of the 2019 Online Shopping Store—Web Server Logs, Harvard Dataverse, V1 by Farzin Zaker available at https://doi.org/10.7910/DVN/3QBYB5, accessed on 28 May 2024.

**Acknowledgments:** The researchers would like to thank Farzin Zaker for providing the Online Shopping Store—Web Server Logs dataset that the researchers used in the conduct of this study.

**Conflicts of Interest:** Author Wilson Chua is currently working with the Analytics Division of Future Gen International Pte. Ltd. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

## References

1. Trivedi, J.; Shah, M. A Systematic and Comprehensive Study on Machine Learning and Deep Learning Models in Web Traffic Prediction. *Arch. Comput. Methods Eng.* **2024**, *31*, 3171–3195. [CrossRef]
2. Lu, T.; Wang, L.; Zhao, X. Review of Anomaly Detection Algorithms for Data Streams. *Appl. Sci.* **2023**, *13*, 6353. [CrossRef]
3. Ji, I.H.; Lee, J.H.; Kang, M.J.; Park, W.J.; Jeon, S.H.; Seo, J.T. Artificial Intelligence-Based Anomaly Detection Technology over Encrypted Traffic: A Systematic Literature Review. *Sensors* **2024**, *24*, 898. [CrossRef] [PubMed]
4. Tama, B.A.; Nkenyereye, L.; Islam, S.M.R.; Kwak, K.-S. An Enhanced Anomaly Detection in Web Traffic Using a Stack of Classifier Ensemble. *IEEE Access* **2020**, *8*, 24120–24134. [CrossRef]
5. Kim, T.-Y.; Cho, S.-B. Web traffic anomaly detection using C-LSTM neural networks. *Expert Syst. Appl.* **2018**, *106*, 66–76. [CrossRef]
6. Nassif, A.B.; Talib, M.A.; Nasir, Q.; Dakalbab, F.M. Machine Learning for Anomaly Detection: A Systematic Review. *IEEE Access* **2021**, *9*, 78658–78700. [CrossRef]
7. Carrera, F.; Dentamaro, V.; Galantucci, S.; Iannacone, A.; Impedovo, D.; Pirlo, G. Combining Unsupervised Approaches for Near Real-Time Network Traffic Anomaly Detection. *Appl. Sci.* **2022**, *12*, 1759. [CrossRef]
8. Inuwa, M.M.; Das, R. A comparative analysis of various machine learning methods for anomaly detection in cyber attacks on IoT networks. *Internet Things* **2024**, *26*, 101162. [CrossRef]
9. Li, X.; Liang, D.; Li, X.; Huang, J.; Wu, J.; Gou, H. Quality monitoring of real-time PPP service using isolation forest-based residual anomaly detection. *GPS Solut.* **2024**, *28*, 118. [CrossRef]
10. Gałka, Ł.; Karczmarek, P.; Tokovarov, M. Isolation Forest Based on Minimal Spanning Tree. *IEEE Access* **2022**, *10*, 74175–74186. [CrossRef]
11. Liu, F.T.; Ting, K.M.; Zhou, Z.-H. Isolation Forest. In Proceedings of the 2008 Eighth IEEE International Conference on Data Mining, Pisa, Italy, 15–19 December 2008; pp. 413–422. [CrossRef]
12. Ding, Z.; Fei, M. An Anomaly Detection Approach Based on Isolation Forest Algorithm for Streaming Data using Sliding Window. *IFAC Proc. Vol.* **2013**, *46*, 12–17. [CrossRef]
13. Karev, D.; McCubbin, C.; Vaulin, R. Cyber Threat Hunting Through the Use of an Isolation Forest. In *CompSysTech '17: Proceedings of the 18th International Conference on Computer Systems and Technologies*; Association for Computing Machinery: New York, NY, USA, 2017; pp. 163–170. [CrossRef]
14. Ripan, R.C.; Sarker, I.H.; Anwar, M.M.; Furhad, M.H.; Rahat, F.; Hoque, M.M.; Sarfraz, M. An Isolation Forest Learning Based Outlier Detection Approach for Effectively Classifying Cyber Anomalies. In *Hybrid Intelligent Systems*; Abraham, A., Hanne, T., Castillo, O., Gandhi, N., Rios, T.N., Hong, T.-P., Eds.; Springer International Publishing: Cham, Switzerland, 2021; pp. 270–279.
15. John, H.; Naaz, S. Credit Card Fraud Detection using Local Outlier Factor and Isolation Forest. *Int. J. Comput. Sci. Eng.* **2019**, *7*, 1060–1064. [CrossRef]
16. Zaker, F. Online Shopping Store-Web Server Logs. *Harvard Dataverse.* 2019. [CrossRef]
17. Gabryel, M.; Lada, D.; Filutowicz, Z.; Patora-Wysocka, Z.; Kisiel-Dorohinicki, M.; Chen, G.Y. Detecting Anomalies in Advertising Web Traffic with the Use of the Variational Autoencoder. *J. Artif. Intell. Soft Comput. Res.* **2022**, *12*, 255–256. [CrossRef]
18. Al-Shehari, T.; Al-Razgan, M.; Alfakih, T.; Alsowail, R.A.; Pandiaraj, S. Insider Threat Detection Model using Anomaly-Based Isolation Forest Algorithm. *IEEE Access* **2023**, *11*, 118170–118185. [CrossRef]
19. Franklin, R.J.; Mohana; Dabbagol, V. Anomaly Detection in Videos for Video Surveillance Applications using Neural Networks. In Proceedings of the 2020 Fourth International Conference on Inventive Systems and Control (ICISC), Coimbatore, India, 8–10 January 2020; pp. 632–637. [CrossRef]
20. Liu, F.T.; Ting, K.M.; Zhou, Z.-H. Isolation-Based Anomaly Detection. *ACM Trans. Knowl. Discov. Data* **2012**, *6*, 1–39. [CrossRef]
21. Sadaf, K.; Sultana, J. Intrusion Detection Based on Autoencoder and Isolation Forest in Fog Computing. *IEEE Access* **2020**, *8*, 167059–167068. [CrossRef]
22. Zhang, Y.F.; Lu, H.L.; Lin, H.F.; Qiao, X.C.; Zheng, H. The Optimized Anomaly Detection Models Based on an Approach of Dealing with Imbalanced Dataset for Credit Card Fraud Detection. *Mob. Inf. Syst.* **2022**, *2022*, 8027903. [CrossRef]
23. Hamon, V. Malicious URI resolving in PDF documents. *J. Comput. Virol. Hacking Tech.* **2013**, *9*, 65–76. [CrossRef]
24. Chabchoub, Y.; Togbe, M.U.; Boly, A.; Chiky, R. An In-Depth Study and Improvement of Isolation Forest. *IEEE Access* **2022**, *10*, 10219–10237. [CrossRef]
25. Aldrich, C.; Liu, X. Monitoring of Mineral Processing Operations with Isolation Forests. *Minerals* **2024**, *14*, 76. [CrossRef]

26. Zhang, Q.; Liang, Z.; Liu, W.; Peng, W.; Huang, H.; Zhang, S.; Chen, L.; Jiang, K.; Liu, L. Landslide Susceptibility Prediction: Improving the Quality of Landslide Samples by Isolation Forests. *Sustainability* **2022**, *14*, 16692. [CrossRef]
27. Priyanto, C.Y.; Hendry; Purnomo, H.D. Combination of Isolation Forest and LSTM Autoencoder for Anomaly Detection. In Proceedings of the 2021 2nd International Conference on Innovative and Creative Information Technology (ICITech), Salatiga, Indonesia, 23–25 September 2021; pp. 35–38. [CrossRef]
28. Chen, W.; Yang, K.; Yu, Z.; Shi, Y.; Chen, P. A survey on imbalanced learning: Latest research, applications and future directions. *Artif. Intell. Rev.* **2024**, *57*, 137. [CrossRef]
29. Madhukar Rao, G.; Ramesh, D. A Hybrid and Improved Isolation Forest Algorithm for Anomaly Detection. In *Proceedings of the International Conference on Recent Trends in Machine Learning, IoT, Smart Cities and Applications*; Gunjan, V.K., Zurada, J.M., Eds.; Springer: Singapore, 2021; pp. 589–598.
30. Foody, G.M. Challenges in the real world use of classification accuracy metrics: From recall and precision to the Matthews correlation coefficient. *PLoS ONE* **2023**, *18*, e0291908. [CrossRef] [PubMed]
31. Sokolova, M.; Japkowicz, N.; Szpakowicz, S. Beyond Accuracy, F-Score and ROC: A Family of Discriminant Measures for Performance Evaluation. In *AI 2006: Advances in Artificial Intelligence*; Sattar, A., Kang, B., Eds.; Springer: Berlin/Heidelberg, Germany, 2006; pp. 1015–1021.
32. Yacouby, R.; Axman, D. Probabilistic Extension of Precision, Recall, and F1 Score for More Thorough Evaluation of Classification Models. In *EVAL4NLP*; Association for Computational Linguistics: Stroudsburg, PA, USA, 2020; [Online]; Available online: https://api.semanticscholar.org/CorpusID:226283839 (accessed on 27 May 2024).
33. Powers, D.M.W. Evaluation: From precision, recall and F-measure to ROC, informedness, markedness and correlation. *J. Mach. Learn. Technol.* **2011**, *2*, 37–63, [Online]. Available online: https://api.semanticscholar.org/CorpusID:3770261 (accessed on 8 June 2024).
34. Naseer, S.; Saleem, Y.; Khalid, S.; Bashir, M.K.; Han, J.; Iqbal, M.M.; Han, K. Enhanced Network Anomaly Detection Based on Deep Neural Networks. *IEEE Access* **2018**, *6*, 48231–48246. [CrossRef]
35. Benova, L.; Hudec, L. Comprehensive Analysis and Evaluation of Anomalous User Activity in Web Server Logs. *Sensors* **2024**, *24*, 746. [CrossRef]
36. Sharova, E. Unsupervised Anomaly Detection with Isolation Forest. In Proceedings of the PyData, London, UK, 26 April 2018.