UNSW-NB15 computer security dataset: Analysis through visualization

Zeinab Zoghi | Gursel Serpen

Abstract

Class imbalance refers to a major issue in data mining where data with unequal class distribution can deteriorate classification performance. Although it alone affects the performance of the classifiers, the joint-effect of class imbalance and overlap is more damaging. Data overlap happens when multiple classes are assigned to a single data point causing the classifiers to misidentify the class boundaries. This study offers a deep insight into the intricacies of the UNSW-NB15 dataset and two issues that may lead the data-driven models to demonstrate poor performance. The most commonly used visualization methods such as bar chart, 3D and 2D scatter plots, intercluster distance map, and parallel coordinate diagram were employed to depict the data imbalanced and overlap. However, their limitations in capturing the overlapping issue led us to propose an accurate, easy-to-interpret, and scalable overlapping visualization method. The method clearly detects the data overlap and illustrates the effect of several data scalers in dealing with the data overlap. To verify the accuracy of the proposed method, a number of classifiers were implemented along with the scalers and the calculated AUC scores were compared to those calculated from the classifiers that were implemented on the original dataset.

KEYWORDS
class imbalance, class overlap, data preprocessing, data visualization, intrusion detection system

1 | INTRODUCTION

Class imbalance and overlap are two major data mining issues. Although they individually affect the data analysis negatively, joining one to another causes even worst effects. Assuming an unequal representation of classes, if the class overlap occurs, the regions of the data space are populated by an unequal number of data points from each class containing the majority and minority classes. Thus, class overlap is a main source of data complexity particularly for the pattern recognition algorithms. Several research studies have been done throughout the years to characterize the joint-effect of class imbalance and overlap. The variety of data domains in cybersecurity, medicine, and other fields of study with different imbalance ratios and overlap severity were generated and the classification methodologies were evaluated to quantify the impact of data imbalance and overlap.
Data visualization is an approach to depict data formation and detect the issues such as data imbalance and overlap. It illustrates the complex data, clearly communicates important information, and offers valuable clues for building an optimized data-analysis model if the appropriate method is used. A principal objective of information visualization is to translate large and complex datasets and illustrate them in a visual format. This process facilitates data interpretation in order to identify patterns, trends, clusters, correlations, and relationships among the data points. There are many different techniques and tools to visualize data. Of those, Stahneke et al. employed Multidimensional Scaling (MDS) to project high-dimensional data into a low-dimensional space and visualize the data using scatterplot and heatmap. The purpose of using MDS was to find the distance dissimilarities for any pair of data points. They proposed the concept of “probing” that represents an integrated method containing dynamic selections, class selection, and clustering, providing valuable insight into data and correcting the projection errors. They demonstrated the application of their method on the dataset of OECD countries.

The Wine Recognition dataset was utilized to evaluate the performance of the c-PCA visualization method proposed by Fujiwara et al. The authors used t-distributed Stochastic Neighbor Embedding (t-SNE) to visualize high-dimensional data. Three clusters were generated by applying Density-Based Spatial Clustering of Applications with Noise (DBSCAN) to the output of t-SNE. In the final step, c-PCA was implemented on the clusters and the engagement rate of features in each cluster was calculated. The engagement rate identified the dissimilarities between one cluster and the rest.

In Reference 7, the authors proposed a deep learning model using a recurrent neural network (RNN) called RetainEX. The model was implemented on electronic medical records containing the events such as histories of patients’ diagnoses and medications. This method is comprised of visual analytics carried out by the RetainVis visualization tool. To form a 2D shape of patients’ medical information, t-SNE was implemented on contribution scores calculated from all codes that each patient had for every visit. The final 2D projection view of Electronic Medical Records (EMRs) data was summarized by representative points using clustering approaches. Similarly, Reference 8 implemented dimensionality reduction-based techniques on medical datasets. They utilized Hierarchical Stochastic Neighbor Embedding (HSNE) in order to eliminate the scalability limit of t-SNE by considering rare cell populations that were removed in the last undersampling procedure and reproducing former observations.

Wilkinson presented a new algorithm called HDoutliers to visualize the observations that are remarkably far from the closest point to the majority number of the data points in high-dimensional space. The algorithm brings all the variables to the same range to prevent a disproportionate impact on Euclidean distances due to data skewness. The authors employed the Leader algorithm to characterize the high density of data points using a large number of small balls rather than a small number of clusters. The distance of the members in each cluster was calculated using the nearest neighbor algorithm. The values were deployed for outlier identification or anomaly detection procedure.

Heatmap and line graph were utilized in Reference 11 in terms of data visualization. The literature presented a case study using Eventpad to study sequential patterns in network traffic by visualizing them. They depicted the patterns in network traffic by capturing the recurrent access patterns. The patterns were shown by the line graph and the behaviors of the network traffic, containing open, close, and find, were illustrated using a heatmap.

Ruan et al. deployed a hash algorithm, weight table, and sampling method to address volume, variety, and velocity issues arising from big data. They formed a weight table and assigned large weights to the classes with a smaller population to guarantee the less probable data records to be selected by the sampling method. The data records were selected based on the weight table and the hash code retrieved during the sampling approach which minimizes class imbalance issues and any redundancies, respectively. MDS and PCA were used for big data visualization on the resampling data. Karami proposed a modified Self-Organizing Map (SOM) to both detect the anomalies and provide the information summary from 2D data visualization using UNSW-NB15.

Of the above-mentioned visualization methods, scatter plot, heatmap, and t-SNE were among the most commonly used visualization methods and PCA was the most popular dimensionality reduction technique. However, each method suffers from several weaknesses. In Eventpad, tracing the pattern of non-recurrent network behavior is challenging. Some state-of-the-art solutions are linear (PCA), and some are non-linear (t-SNE) but limited by the number of variables and high-dimensional space causing clutter and misleading interpretation. In scatter plots, it is difficult to identify patterns particularly when there is no clear correlation between two variables. It is challenging to interpret the values in heatmaps, when there are several variables or when the color scale is not well-defined. They mainly focus on analyzing the network
### TABLE 1  Comparing the state-of-the-art solutions.

| Name | Methods | Advantages |
|------|---------|------------|
| Probing³ | • Distance correlation  
• Dendogram  
• Scatter Plot  
• Multi-level comparison  
• Grouping sample  
• Density plot  
• Heatmap | • Visualizes the low-dimensional data  
• Calculates and solving the dimensionality reduction errors |
| ccPCA⁴ | • Contrastive Learning  
• Heatmap  
• t-SNE  
• Scatter plot  
• PCA + LDA | • Compares two different datasets/clusters  
• Finds variation within a cluster  
• Finds difference between two clusters |
| Retain Vis⁷ RetainEX | • t-SNE  
• Scatter plot  
• Bar chart  
• Area chart  
• Pie chart  
• RNN | • Tackles the problem of interpretability and interactivity of RNN data visualization for clinical records |
| HSNE⁸ | • Hierarchical stochastic neighbor embedding  
• Weighted k-nearest neighbor (kNN) graph  
• Administering the area of influence (AoI) of the landmarks | • Avoids down-sampling  
• Preserves the non-linear high-dimensional relationships between cells  
• Builds hierarchical representation of the data and unfolds the non-linearity in the high-dimensional data |
| Hdoutliers⁹ | • The Distance from the Center Rule  
• The Box Plot Rule  
• The Gaps Rule  
• Mahalanobis Distance  
• Multivariate Gap Tests  
• Clustering  
• Correspondence analysis  
• Random projection  
• Histogram  
• Probability plot  
• Dot plot | • Deals with the combination of nominal and numerical variables  
• Deals with the curse of dimensionality  
• Deals with the outliers  
• Deals with masking issue  
• Works with both multi and single dimensional data |
| EventPad¹¹ | • Data aggregation  
• Heatmap  
• Line graph | • Increases readability of data  
• Reduce screen clutter |
| PCA + MDS + Colormap¹²,¹³ | • Hash algorithm  
• Weight table  
• Sampling method  
• Pie chart  
• MDS  
• PCA  
• Scatter plot | • Capture the major information and details.  
• High precision and less information loss |
TABLE 1 (Continued)

| Name                  | Methods                                      | Advantages                        |
|-----------------------|----------------------------------------------|-----------------------------------|
| SOM\textsuperscript{14} | • Outlier identification                     | • Lower computational costs       |
|                       | • Roulette Wheel (RW) selection method       | • High learnability and satisfaction |
|                       | • Lattice graph                              | • Easy to use and analyze the data |
|                       | • Neuron visualization                       |                                   |
| Voxel-based approach\textsuperscript{15-17} | • PCA                                        | • Facilitates understanding data   |
|                       | • One-hot transformer                         | • Visualizes ML decision spaces    |
|                       | • 3D scatter plot                             |                                   |
| Proposed method       | • Bar chart                                   | • Easy identification of pattern and relationships |
|                       | • Scatter plot + PCA                          | • Non-linear dimensionality reduction |
|                       | • t-SNE                                       | • Clustering visualization         |
|                       | • K-Mean distance map                         | • Interactive exploration         |
|                       | • Parallel coordinate                        | • Highlights key information      |
|                       | • Heatmap                                     | • Scalable                        |
|                       | • Mahalanobis Distance                        | • Easy to interpret               |
|                       | • Nearest Shrunken Centroid                  | • Accurate                        |

traffic data by splitting it into small subsets and narrowing it down to particular aspects of the issue such as ports or system-calls monitoring. Bar charts are though easy to use, they are best suited for discrete data and when the values are big enough/far from each other.

In this study, we implemented the commonly used data visualization techniques on the UNSW-NB15 to depict the data imbalance and overlap. However, their limitations make it challenging to clearly detect the data overlap. To address the limitations, we propose a scalable, interpretable, and efficient method for depicting the data overlap. In this method, a class centroid was measured using the nearest shrunken centroid, the centroid distance was calculated using Mahalanobis distance, and the data pattern was summarized in a heatmap. A variety of data scalers were applied to the data and the distance between class centroids was measured and presented using the proposed method to estimate how far the data points of each class are from each other after implementing the data scalers. Comparing the heatmaps revealed that the robust scaler, min-max scaler, and power transformer could keep the classes at a reasonable distance in comparison to the other scalers. The accuracy of the proposed method was verified by comparing the AUC scores of several classifiers implemented on the original and scaled data. The higher AUC scores belonging to robust scaler, min-max scaler, and power transformer confirmed that the proposed method is strong enough to calculate and depict the class overlap and that the scalers can minimize the effect of data overlap on data classification by increasing the distance between the class centroids. Several preprocessing techniques are implemented to prepare the data for visual analysis. The redundancy in UNSW-NB15 is reduced manually, the nominal input features are converted to numerical, and the relevant input features are selected using Elastic Net. PCA is also utilized in order to project the preprocessed data into low-dimensional space.

2 | UNSW-NB15 DATASET

The UNSW-NB15 is one of the popular\textsuperscript{14-17} and comprehensive cybersecurity datasets released in 2015.\textsuperscript{18,19} This dataset is comprised of 2 540 044 realistic modern normal and abnormal (also known as an attack) network activities. These records were gathered by the IXIA traffic generator using three virtual servers. Two servers were configured to distribute the normal network traffic and the third one was configured to generate the abnormal network traffic.

A total of 49 features including packet-based and flow-based features were extracted from the raw network packets by Argus and Bro-IDS tools. Packet-based features are extracted from the packet header and its payload (also called packet data). In contrast, flow-based features are generated using the sequencing of packets, from a source to a destination, traveling in the network. The direction, inter-packet length, and inter-arrival times are the most important properties in the flow-based feature formulation: total duration (dur) and destination-to-source-time-to-live (dttl) are two
examples of flow-based features. The features are categorized into three sets, namely basic (6–18), content (19–26), and time (27–35). Features 36–40 and 41–47 are labeled as general-purpose features and connection features, respectively. The general-purpose features category includes those features intended to explain the purpose of an individual record while connection features depict the character of the connection among a hundred sequentially ordered records. The last two features include attack categories and labels.

Attacks are categorized as Analysis, Backdoor, DoS, Exploits, Fuzzers, Generic, Reconnaissance, Shellcode, and Worms. In the original dataset, the Normal class contains 2,218,761 records while Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode, and Worms include 24,246, 2677, 2329, 16,535, 44,525, 215,481, 13,987, 1511, and 174 records, respectively. However, in this study, a subsample of the dataset that is generally used in different studies will be analyzed. The dataset and the subsample are available on the UNSW web page. The structure of this dataset is more complex in comparison with the other benchmark datasets such as DARPA98, KDDCUP 99, and NSL-KDD.

3 | PREPROCESSING

The training and testing subsets of UNSW-NB15 as provided by Moustafa are utilized for this study. The authors selected 175,341 records to form the training subset and 82,332 records for the testing subset among the original 2,218,761 records. This dataset contains 49 features. However, all 49 features are not necessarily relevant to the class labels. Two input features, namely record_start_time, and record_last_time, are redundant due to the presence of total duration (dur) that is obtained from the difference of values for these two features. Since some features are specific to the computing infrastructure such as source IP address, source port number, destination IP address, and destination port number, they do not possess relevant information for generic intrusion detection purposes. Accordingly, record_start_time, record_last_time, source IP address, source port number, and destination IP address are eliminated and the explanatory input features are kept. Of the remaining 43 input features, two are output/target features, namely attack_cat, and label. The label is binary and the attack_cat feature is of type nominal and contains the names of attack categories. The latter is utilized to deeply analyze the data using the name of the overlapping and imbalanced classes. Three of the 41 non-target features are nominal. They are converted to numerical as most of the machine learning models and scalers can readily work with the numerical values. To convert the nominal to numerical, LabelEncoder is implemented from the sci-kit-learn library in Python.

The Elastic Net algorithm is implemented on the dataset in order to redundancy elimination, dimensionality reduction, and subsequently decrease the computational cost associated with the process of visualization. This algorithm (ElasticNetCV in Python) is applied five times using 5-fold cross-validation with different values for alpha determined randomly and an L1 ratio with the value 0.5. The mean squared error (MSE) and regularization path are calculated and shown in Figures 1 and 2, respectively. MSE is calculated as \( \frac{1}{n} \sum_{i=1}^{n} (Y_i - \hat{Y}_i)^2 \) where \( n \) is the number of samples; \( Y \) is the vector of real class or target values; and \( \hat{Y} \) is the vector of predicted class or target values. The cyclic coordinate descent is applied across all the alpha values from 0 to 0.001 to find the minimum alpha and solve the Elastic Net penalized regression model. It is shown in Figure 1. As the alpha value gets larger, the mean squared error in each fold also increases. The minimum amount of error is reported at the point where the optimum alpha value is roughly 0.0003.

In order to implement the feature selection based on the optimum alpha and subsequently improve the data interpretation in the visualization phase, the regularization path is generated. The irrelevant features are eliminated where the alpha is 0.0003 using the Elastic Net algorithm. This algorithm zeroes out the coefficients of irrelevant features. Figure 2 presents the regularization paths. The parameter lambda represents the number of alphas used for the L1 ratio or the list of L1 ratios along the regularization path. In this case, a value of 100 is chosen for this variable. The greater the value of this variable is, the smoother the curves that are shown in the graph are. The log base 10 of lambda I negated to reverse the graph and interpret it from right to left. There are 41 curves in this figure, some are overlapping with each other. Each curve represents an input feature. From right to left in this graph, by reaching the point 0.0003 where alpha was minimum, 17 curves touch the coefficient boundary which is zero. These features depict high correlation and removing them decreases the computational costs in the visualization phase. Therefore, these features are eliminated and the 24 most informative and the least correlated features among the 41 input features are selected as presented in Table 2.
The entire process of applying the Elastic Net algorithm on the training subset for selecting a value for alpha takes approximately 4.70 seconds where the algorithm converges at the 409th iteration (on a platform with Windows 7 Enterprise 64-bit operating system, Intel® Core™ i5-4690 CPU @ 3.50 GHz processor, and 16.0 GB RAM).

4 RESULT AND DISCUSSION

4.1 Data visualization

The data visualization analysis is conducted on Windows 7 Enterprise 64-bit operating system, Intel® Core™ i5-4690 CPU @ 3.50 GHz processor, and 16.0 GB RAM. To avoid the clutter issue in the visualization phase, which may arise due to a large number of data points, a proposed subsample of the data points is utilized to present the data using t-SNE and K-means clustering where the data points are illustrated in 2D projection view. Stratified sampling is deployed to randomly extract 20% of the records from each class type. These records are randomly selected from the training subset and form a new subset containing a smaller number of records to present UNSW-NB15 with more clarity.

Two prominent problems are identified through this analysis. The issues are known as class imbalance and class overlap. Between-class and within-class imbalance are two types of class imbalance. Between-class imbalance corresponds to the case where one class or multiple classes in a dataset are underrepresented in comparison with other classes. In other words, a dataset shows significantly unequal distribution among its classes. To illustrate this problem, the distribution of entire attack classes for the UNSW-NB15 dataset is plotted in Figure 3. As seen in this figure and Table 3, normal records constitute 36.09% of all records while the combined record count for all nine attack classes is 63.91%. Of 63.91%, 22.85% of attack data points belong to the Generic attack class while only 0.07% belong to the Worms attack class. Although the
| Feature no. | Input feature name | Description |
|------------|-------------------|-------------|
| 1          | dur               | Record total duration |
| 2          | proto             | Transaction protocol |
| 3          | service           | Contains the network services |
| 4          | state             | Contains the state and its dependent protocol |
| 5          | spkts             | Source to destination packet count |
| 6          | dpkts             | Destination to source packet count |
| 7          | sbytes            | Source to destination transaction bytes |
| 8          | dbytes            | Destination to source transaction bytes |
| 9          | rate              | Ethernet data rates transmitted and received |
| 10         | sttl              | Source to destination time to live value |
| 11         | dttl              | Destination to source time to live value |
| 12         | sload             | Source bits per second |
| 13         | dload             | Destination bits per second |
| 14         | sloss             | Source packets retransmitted or dropped |
| 15         | dloss             | Destination packets retransmitted or dropped |
| 16         | sinpkt            | Source interpacket arrival time (ms) |
| 17         | dinpkt            | Destination interpacket arrival time (ms) |
| 18         | sjit              | Source jitter (ms) |
| 19         | djit              | Destination jitter (ms) |
| 20         | swin              | Source TCP window advertisement value |
| 21         | stcpb             | Destination TCP window advertisement value |
| 22         | dtcpb             | Destination TCP base sequence number |
| 23         | dwin              | Destination TCP window advertisement value |
| 24         | tcprtt            | TCP connection setup round-trip time |

The list of features selected by Elastic Net. The dataset seems imbalanced in the binary format, it is possible to balance it by adding Normal data point from the original data. However, the attack classes are imbalanced in both binary and multi-class formats.

Within-class imbalance represents the class that is comprised of several different subclasses with different distributions. To discover whether the classes are made up of imbalanced sub-clusters, two visualization techniques are used from scikit-learn machine learning library in Python. The PCA plus 3D scatter plot and t-SNE with the time complexity of respectively $O \left[ \min(n^3, p^3) \right]$ and $O(n^2)$ are implemented where $n$ is the number of data records and $p$ is the number of input features. The PCA is deployed to project the data to 2 and 3 dimensions. It took PCA, t-SNE, and the scatter plot to process and generate Figures 4–6 in 1.8 and 2178.33 s, respectively. Figures 4 and 5 show the within-class imbalance in the UNSW-NB15 dataset. The class types appear in different colors tagged with the name of the class types. As Figure 4 shows, the normal (shown in navy blue) and the attack classes are composed of several sub-clusters with different sizes representing subconcept (small sub-clusters) and concept (large sub-clusters). Except for Shellcode and Worms which are only formed with concept and subconcept sub-clusters, respectively, the remaining classes are made up of both types of sub-clusters.

Figure 6 illustrates all the concepts using a 2-dimensional scatter plot for the data points in the dataset using the t-SNE algorithm. The class types have multiple clusters of different sizes and are spread across the two-dimensional analysis space. The classes are composed of a few relatively large clusters and many small clusters. Additionally, the boundaries separating classes are not clear-cut: there is a noticeable overlap between or among multiple clusters belonging to different classes. For this method, Kullback–Leibler (KL) divergence score is calculated from 250 iterations at the early stages of optimization to 1000 iterations. The divergence score varied from 96.52 to 2.93 which represents the difference of a class
probability distribution from another. During the 250 iterations, 91 nearest neighbors of 175,341 samples are computed in 179.02 s, the samples are indexed in 39.35 s, and conditional probability assessments and KL divergence occurred in 1959.96 s.

Class overlap, on the other hand, is the case where one or more attack class records mimic the behavior of the Normal records or records belonging to other attack classes. While the emphasis of intrusion detection systems is detecting and/or identifying the malicious network traffic, a satisfactory outcome will be achieved if the overlapping issue is addressed particularly where the attack classes overlap the normal activities. In order to expose the scale of this problem in the dataset, the data points from 10 existing classes are sketched in parallel by a 3-dimensional scatter plot as shown in Figure 7. The figure shows that the attack class instances overlap the normal and attack instances as indicated by the dashed circles.

Further investigations are conducted for those attack class records that reside in the subspace where primarily Normal records are present. Figures 8 and 9 illustrate the overlap issue in detail particularly where the boundaries of attack classes coincide with the boundary of the Normal class. Figure 8 represents the overlapping issue between the normal and attack classes. The more the number of data points in an attack class, the higher degree of overlapping the attack class has against
the normal class. Figure 9 is a close-up illustration from the Exploits attack class that has an overlapping problem with the Normal class. In Figure 9, there is a noticeable overlap between the sets of data points belonging to Exploits (in dark salmon) and those belonging to the Normal class (in navy blue).

Figure 10 shows overlaps for the entire dataset using K-means inter-cluster distance map. We used the K-means clustering method to compute the distance between the class centroids that was already calculated using the nearest shrunken centroid algorithm. Inter-cluster distances are also utilized to sketch the map. Inter-cluster distance maps illustrate an embedding of the class centers in 2D view with the distance against other centers. The closer the circles drawn on the map, the closer the data points are to the original feature space. As the figure shows, the overlap degree among classes 8, 1, and 4 as well as between 0 and 6, and between 9 and 2 are substantial.

We employ the K-means clustering algorithm that has a time complexity of $O(n^3)$ along with the Inter-cluster Distance visualizer in Python to present the distance between the clusters. It takes 0.001 s for these algorithms to generate Figure 10.

The clusters are sized according to the number of members in a single cluster. In other words, the number of samples whose distance to the centroid is much shorter than the other data points belonging to the corresponding center that forms a cluster. The map gives a sense of the number of data points in a cluster, how they spread all across the cluster, and the probability of class overlap.
The parallel coordinate diagram shows more details about the relationship between input features in each class. Figure 11 depicts the overlapping issue by considering the input features along with the output feature (0 = Analysis, 1 = Backdoor, 2 = DoS, 3 = Exploits, 4 = Fuzzers, 5 = Generic, 6 = Normal, 7 = Reconnaissance, 8 = Shellcode, 9 = Worms). By selecting all the outputs shown in the last column of the diagram, the figure outlines the value of the features.

A significant part of the diagram is filled with the lines in salmon diverging to teal rose color, representing Exploits, Fuzzers, Generic, Normal, and Reconnaissance. The classes are distributed across the features dur (column #1), proto (column #2), spkts (column #5), sbytes (column #7), stcpb (column #21), dtcpb (column #22), smeans (column #27), and dmeans (column #28), while the other classes are mostly distributed across the features spkts, dpkts, sbytes, and sloss. The diagram is split into 10 sections in Figure 12, to analyze each class separately. The full parallel coordinate diagrams for
**FIGURE 8** 3D Visualization of the Normal against attack classes.

**FIGURE 9** 3D close-up from the Normal and Exploits data points.

**FIGURE 10** Illustration of class overlap using inter-cluster distances and K-means.
each class are shown in Appendices B–K. In addition, the input features are analyzed in Appendix A in terms of mean, standard deviation, mode, minimum, maximum, and quartiles. The data analysis is implemented using SPSS and the diagrams are sketched using Plotly, the Python library.

Except for the normal activities, the attack classes contain only zeroes for is_sm_ips_ports. Shellcode and Worms distinctly overlap with each other in swin, stcpb, dtcpb, and dwin and proto. However, 255 appears most often in swin and dwin in Worms while 0 appears frequently in the same features in Shellcode. In addition, the max values in stcpb and dtcpb of the aforementioned classes are slightly different. Although they have some values in common in is_sm_ips_ports, Shellcode values are 2 and 3 for state and 0 for service, which makes it distinguishable from Worms. In addition, the mean value of dload in Worms is almost 56 times more than that in Shellcode, and dloss in Shellcode contains zeroes and ones while it contains values between 0 and 347 in Worms.

Except for Shellcode, Worms, and Normal classes, the values of proto in the rest of the attack categories are distributed equally in the range of 0–132. The values of dur spread equally between 0 and ~60 for Analysis, Backdoor, DoS, Exploits, Fuzzers, Generic, Normal, and Reconnaissance while its value is in the range of 0 to ~15 for Shellcode and Worm. Comparing Generic and Exploit, the two majority attack classes, although they seem to overlap in many features such as dur, proto, service, state, rate, sttl, dttl, swin, stcpb, dtcpb, dwin, and so forth, they differ in mean and mode in many cases. In addition, they can be distinguished by the values of spkts, sbytes, sloss, and sjit. The values of those features in Generic do not exceed ~3000, ~3.5M, ~1400, and ~0.25M, respectively.
Comparing the Normal and the attack categories, normal activities cover the lines for all the features except for dload. The values of the attack classes for dload do not exceed ∼2.5M even for the majority of attack categories. The mean value in response_body_len of Normal, DoS, and Exploit are almost the same, while this value in Shellcode is 0 and in Worms is almost ten times bigger.

4.2 Proposed approach

In the last section, several widely used visualization methods were used for depicting the potential imbalanced and overlapping issues. Both the between-class and within-class imbalance were illustrated clearly using a bar chart and the 3D scatter plot. Although t-SNE is a non-linear dimensionality reduction method, robust to outliers, and useful for clustering analysis which makes it powerful to analyze the class imbalance and overlap, with different implementations of the algorithm, it produced different results representing that it is non-deterministic. Since it is also sensitive to hyperparameters, the accuracy of the result is questioned. Thus neither class imbalance nor overlap can be identified 100% correctly using t-SNE. On the other hand, the 3D scatter plot was rotatable which was easy to understand the relationships between input features, however, the use of perspective could make certain points appear closer or farther away than they actually were which may lead to misinterpretation of the data. In the K-Mean inter-cluster distance map, overlapping in 2D space does not necessarily imply that the data suffers from overlapping issue in the original space. In addition, the accuracy of the map closely depends on the accuracy of K-Mean clustering. A parallel coordinate diagram could effectively visualize high-dimensional data with the power of customizing the diagram by selecting one or multiple classes. Yet, it is not appropriate to depict too many data points and/or variables that may cause overplotting and clutter.

To address the above-mentioned limitations, we propose a scalable, accurate, and interpretable method to visualize the data overlap. In this method, the nearest shrunk centroid is measured for each attack class as well as normal activities, and the distance between class centroids is calculated using the Mahalanobis distance. The nearest shrunk centroid was deployed on the data. Compared to the K-Mean inter-cluster distance map, the centroids were calculated from the data points that actually belong to the classes rather than using unsupervised machine learning algorithms to calculate the centroid from probably wrongly predicted clusters. Thus, the calculations are more accurate. Given the centroids, the data is summarized into smaller dimensions without losing informative data for detecting the data overlap, which is well-suited for heatmap visualization. In addition, the heatmaps are easy to interpret because they illustrate the distance between the class centroids.

Six different data transformation algorithms called data normalizer, min-max scaler, robust scaler, standard scaler, quantile transformer, and power transformer are implemented and evaluated in addressing the overlapping issue. The distance between the class centroids is outlined using a heatmap before and after the data scaling and the results are compared in Figures 13 and 14.

The results are illustrated through seven heatmaps. The heatmap made from the original dataset in Figure 13 is compared with the six other heatmaps in Figure 14 representing the distances between class centroids when the scalers

![Figure 13](image_url) The Mahalanobis distance of the centroids when the dataset is not normalized.
FIGURE 14 The Mahalanobis distance of the centroids after normalizing the data (upper-left), scaling the dataset with min-max, maxabs (upper-right), robust (left-center), standard (right-center), quantile transformation (lower-left), and power transformation (lower-right).
are applied. The distances of the centroids are depicted with the range of colors between red and green. A red single cell in a heatmap indicates that the centroids of the corresponding classes are close to each other and a green cell represents that the centroids of the corresponding classes are far from each other. All the algorithms increase the distance between the normal class and the rest. However, in quantile transformation, there is an overlap issue between the normal and Fuzzers classes. In addition, the Generic category moves farther in Min-Max and Standard scalers as well as Quantile and Power transformation algorithms. Although the Robust scaler indicates unsatisfactory performance, it raises the distance of the centroids of the Exploit category against all the rest of the categories except for Dos. This attack class is the one with a high rate of overlapping issue with Shellcode, Reconnaissance, Analysis, Backdoor, and Dos categories.

Table 4 reveals how each algorithm improves the performance of classifiers in binary and multi-class classification. It presents the distance between the Normal class and the attack classes. The right-side column shows the average distance between the centroids of the Normal class against the rest. These values are significantly higher where the Robust scaler, Min-Max scaler, and Power transformation are applied. However, Figure 14 shows that the Robust scaler generally decreased the distance of the data points among the attack classes. Therefore, it can be the most efficient algorithm in binary class classifications where the model is trained by the patterns belonging to two classes, namely of Normal and attack, and more distance among the data points of the two classes makes the data labeling more reliable. On the other hand, the Min-Max scaler and Power transformation are the best two algorithms with minor differences in the average values in comparison with the Robust scaler. In general, they performed efficiently. These algorithms shift the values such that the class centroids (for both the attack classes against attack classes and the normal class against attack classes) are placed farther from each other while the data points belonging to a specific attack category stay around in the same area. This property of the Min-Max scaler and Power transformation helps to present the data with less overlap and higher clarity, and thus potentially improve the performance of the classifiers in both the binary and multi-class classification.

### Table 4. The Mahalanobis distances between the Normal centroid and the rest.

|        | Analysis | Backdoor | Dos  | Exploit | Fuzzers | Generic | Recon | Shellcode | Worms | Average |
|--------|----------|----------|------|---------|---------|---------|-------|-----------|-------|---------|
| Original | 2.57     | 3.02     | 3.01 | 1.19    | 3.42    | 2.99    | 1.39  | 3.15      | 3.67  | 2.71    |
| Normal  | 3.06     | 2.68     | 2.79 | 2.83    | 3.07    | 3.93    | 3.16  | 3.40      | 4.08  | 3.22    |
| Min-Max | 2.71     | 3.78     | 3.43 | 2.39    | 3.64    | 4.09    | 3.31  | 3.32      | 3.81  | 3.39    |
| Robust  | 3.13     | 3.22     | 3.39 | 3.94    | 2.76    | 3.38    | 3.29  | 3.33      | 4.20  | 3.40    |
| Standard | 2.70     | 2.82     | 2.29 | 2.17    | 3.72    | 3.96    | 3.05  | 3.19      | 2.72  | 2.96    |
| Quantile | 2.84     | 3.01     | 3.04 | 2.14    | 0.71    | 3.37    | 2.74  | 2.88      | 3.68  | 2.71    |
| Power   | 3.29     | 3.47     | 3.36 | 2.79    | 3.09    | 4.03    | 3.29  | 3.20      | 4.01  | 3.39    |

### 4.3 Evaluation

In this section, the effect of the scalers on increasing the efficiency of the classifiers and the accuracy of the proposed method in visualizing the data overlap is evaluated. The performance of Logistic Regression (LR), Support Vector Machines (SVM), Multi-Layer Perceptron (MLP), Stochastic Gradient Descent (SGD), Naive Bayes (NB), K-Nearest Neighbors (KNN), K-Mean, DBSCAN, XGBoost, Bagging and Voting models from distinct types of ML algorithms are evaluated in terms of AUC score. The scores are calculated before and after implementing the data scalers and transformers and are compared to measure how each data scalers and transformers affect the predicting power. The scores are compared in Table 5.

In LR, the highest AUC scores belong to Normalization, Min-Max scaler, Robust scaler, and Power transformer. Among the Min-Max scaler, Robust scaler, and Power transformer that equally increased the distance, the Power transformer gives a higher score (AUC: 96%). However, the execution time in the Power transformer (29.48 s) is higher than the rest. On the other hand, the normalization method is executed in a short time (2.75 s) yet improves the performance of the model. SVM performed considerably well with the Power transformer both in terms of AUC score (96%) and execution time (3643.1 s). Min-max scaler improves the performance of the MLP model (AUC: 98%) comparatively more than the rest. It
5 | CONCLUSIONS

Data visualization is the graphical representation of data. It communicates complex data in a clear and concise manner and helps to identify data patterns and trends that may not be immediately detectable by looking at raw data. It is widely used in various fields of study. In cybersecurity, data visualization is used to depict the pattern of normal and attack traffic. In this study, we proposed a visualization method to detect the data overlap where the data also suffers from data imbalance. The combination of class imbalance and overlap specifically, if not addressed, is likely to hinder the attack detection and identification performance of intrusion detection systems. We performed exploratory data analysis on the UNSW-NB15 using the most commonly used visualization techniques and compared them with

| TABLE 5 | The comparison of the AUC scores in classification. |
|---------|-----------------------------------------------------|
| Classification | Clustering | Boosting |
| LR | SVM | MLP | SGD | NB | KNN | K-Mean | DBSCAN | XGBoost | Bagging | Voting |
| Original | 0.80 | 0.81 | 0.73 | 0.50 | 0.81 | 0.86 | 0.47 | 0.62 | 0.97 | 0.81 | 0.95 |
| Normal | 0.93 | 0.84 | 0.92 | 0.81 | 0.81 | 0.90 | 0.53 | 0.75 | 0.98 | 0.93 | 0.96 |
| Min-Max | 0.94 | 0.85 | 0.98 | 0.85 | 0.84 | 0.93 | 0.52 | 0.77 | 0.97 | 0.93 | 0.97 |
| Robust | 0.94 | 0.92 | 0.97 | 0.85 | 0.84 | 0.94 | 0.58 | 0.80 | 0.97 | 0.92 | 0.97 |
| Standard | 0.81 | 0.85 | 0.90 | 0.82 | 0.83 | 0.92 | 0.50 | 0.68 | 0.97 | 0.81 | 0.95 |
| Quantile | 0.82 | 0.83 | 0.85 | 0.78 | 0.80 | 0.88 | 0.50 | 0.70 | 0.96 | 0.80 | 0.90 |
| Power | 0.96 | 0.96 | 0.96 | 0.86 | 0.85 | 0.94 | 0.58 | 0.79 | 0.97 | 0.96 | 0.98 |

| TABLE 6 | ML models execution time. |
|---------|---------------------------|
| LR | SVM | MLP | SGD | NB | KNN | K-Mean | DBSCAN | XGBoost | Bagging | Voting |
| Original | 15.55 | 49.773 | 21.55 | 30.30 | 1.58 | 342.94 | 4.38 | 33.986.4 | 21.55 | 16.94 | 44.06 |
| Normal | 2.75 | 78.273 | 365.19 | 1.29 | 1.65 | 340.14 | 4.12 | 41.675.7 | 40.05 | 24.90 | 64.05 |
| Min-Max | 10.06 | 7641.9 | 140.98 | 1.76 | 2.19 | 347.22 | 9.77 | 48.193.2 | 21.55 | 24.70 | 48.81 |
| Robust | 16.16 | 12.560 | 86.29 | 2.99 | 2.96 | 369.36 | 5.38 | 30.642 | 21.78 | 29.43 | 47.77 |
| Standard | 15.93 | 4264.9 | 184.73 | 3.30 | 2.45 | 360.81 | 4.81 | 37.564 | 21.32 | 23.78 | 45.57 |
| Quantile | 18.48 | 4576.8 | 200.18 | 7.65 | 8.10 | 345.52 | 5.81 | 31.235.1 | 23.23 | 27.08 | 46.35 |
| Power | 29.48 | 3643.1 | 191.75 | 24.30 | 24.96 | 333.89 | 18.24 | 58.546.5 | 34.45 | 38.30 | 61.71 |

achieves third place in implementation in a short time (140.98 s) in comparison with the original dataset and the dataset scaled with the Robust scaler. In SGD and NB, the Power transformer returns the highest score (SGD-AUC: 86%, NB-AUC: 85%) in comparatively high execution time (SGD: 24.30 s, NB: 24.96 s). In K-NN and K-Mean, though they are from two different categories of ML algorithms, Robust scaler, and Power transformer equally performed efficiently. It took shorter for KNN and K-Mean to be employed with the Power transformer (333.89 s) and the Robust scaler (5.38 s), respectively. The distance between the data points is calculated using Euclidian in K-Mean and Mahalanobis in DBSCAN. The Robust scaler improves the performance of both models even with different distance metrics. The performance of DBSCAN is improved from 62% to 80% by scaling the data with the Robust scaler. In boosting method, though the normalization algorithm boosts up the AUC score by one percent (AUC: 98%), it doubles the execution time (original data: 21.55 s, scaled data using normalization: 40.05 s). However, in two other ensemble methods, the Power transformer improves the performance of the models up to 15 (from 81% to 96%) and 3 (from 95% to 98%) percent in bagging and voting, respectively. The calculations are done on a system with Windows 10 operation system, Intel core i5-7500 CPU, and 8 GB Ram. Table 6 represents the execution time of the ML models in seconds when they were implemented on the original data and the scaled data.
the proposed method in terms of data overlap visualization. The dataset was projected into 2D and 3D views using the PCA and visually analyzed with the scatter plot, t-SNE, K-means inter-cluster distance map, and parallel coordinate diagram. The proposed method was developed in order to address the visualization limitations identified in the above-mentioned techniques. It is a more accurate, scalable, and interpretable visualization method in comparison with the limited techniques. Using this method, the class centroids were calculated and centroids’ distances were measured. The distance between the class centroids in the original data was compared with those in the scaled data. Given the results, the robust scaler, min-max scaler, and power transformer could increase the distance between the class centroids. The accuracy of the proposed visualization method was verified and confirmed by running several classifiers. The highest AUC scores belonged to the classifiers that were implemented on the data scaled by robust scaler, min-max scaler, and power transformer; this also indicates that they could help split the boundary of the classes and improved the performance of the classifiers. Future works will focus on implementing a variety of outlier detectors to exclude the outliers prior to centroid calculation to increase the accuracy of the calculations. In addition, more centroid calculation methods can be deployed and compared with the current one in terms of visualization. A 2D visualization method can also be designed using the calculations in the proposed method to depict the boundary of the classes and reflect the severity of overlapping issue.

DATA AVAILABILITY STATEMENT
Data derived from public domain resources.

ORCID
Zeinab Zoghi https://orcid.org/0000-0002-1285-0438

REFERENCES
1. Lee HK, Kim SB. An overlap-sensitive margin classifier for imbalanced and overlapping data. Exp Syst Appl. 2018;98:72-83.
2. Das S, Datta S, Chaudhuri BB. Handling data irregularities in classification: foundations, trends, and future challenges. Pattern Recogn. 2018;81:674-693.
3. Stahnke J, Dörk M, Müller B, Thom A. Probing projections: interaction techniques for interpreting arrangements and errors of dimensionality reductions. IEEE Trans Vis Comput Graph. 2015;22:629-638.
4. Fujiwara T, Kwon O-H, Ma K-L. Supporting analysis of dimensionality reduction results with contrastive learning. IEEE Trans Vis Comput Graph. 2019;26:45-55.
5. Maaten LVD, Hinton G. Visualizing data using t-SNE. J Mach Learn Res. 2008;9:2579-2605.
6. Ester M, Kriegel H-P, Sander J, Xu X. A density-based algorithm for discovering clusters in large spatial databases with noise. In: Kdd; 1996:226-231.
7. Kwon BC, Choi M-J, Kim JT, et al. Retainvis: visual analytics with interpretable and interactive recurrent neural networks on electronic medical records. IEEE Trans Vis Comput Graph. 2018;25:299-309.
8. van Unen V, Höllt T, Pezzotti N, et al. Visual analysis of mass cytometry data by hierarchical stochastic neighbour embedding reveals rare cell types. Nat Commun. 2017;8:1-10.
9. Wilkinson L. Visualizing big data outliers through distributed aggregation. IEEE Trans Vis Comput Graph. 2017;24:256-266.
10. Hartigan JA. Clustering Algorithms. John Wiley & Sons, Inc; 1975.
11. Cappers BCM, Meessen PN, Etalle S, van Wijk JJ. Eventpad: rapid malware analysis and reverse engineering using visual analytics. 2018 IEEE Symposium on Visualization for Cyber Security (VisSec). IEEE; 2018.
12. Ruan Z, Mao Y, Pan L, Patterson N, Zhang J. Visualization of big data security: a case study on the KDD99 cup data set. Digit Commun Networks. 2017;3:250-259.
13. Ruan Z, Mao Y, Pan L, Xiang Y, Zhang J. Big network traffic data visualization. Multimed Tools Appl. 2018;77:11459-11487.
14. Karami A. An anomaly-based intrusion detection system in presence of benign outliers with visualization capabilities. Exp Syst Appl. 2018;108:36-60.
15. Zong W, Chow Y-W, Susilo W. Dimensionality reduction and visualization of network intrusion detection data. In: Australasian Conference on Information Security and Privacy; 2019:441-455.
16. Zong W, Chow Y-W, Susilo W. A 3D approach for the visualization of network intrusion detection data. In: 2018 International Conference on Cyberworlds (CW); 2018:308-315.
17. Zong W, Chow Y-W, Susilo W. Interactive three-dimensional visualization of network intrusion detection data for machine learning. Fut Gen Comput Syst. 2020;102:292-306.
18. Moustafa N, Slay J. UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In: 2015 Military Communications and Information Systems Conference (MilCIS); 2015:1-6.
19. Moustafa N, Slay J. The evaluation of Network Anomaly Detection Systems: statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set. Inform Secur J: Glob Perspect. 2016;25:18-31.
20. Kumar V, Sinha D, Das AK, Pandey SC, Goswami RT. An integrated rule based intrusion detection system: analysis on UNSW-NB15 data set and the real time online dataset. Clust Comput. 2020;23:1397-1418.

21. Kanimozhi V, Jacob P. UNSW-NB15 Dataset Feature Selection and Network Intrusion Detection using Deep Learning.

22. Moustafa N, Slay J. The UNSW-NB15 Dataset Description; 2018. https://www.unsw.adfa.edu.au/unsw-canberra-cyber/cybersecurity/ADFA-NB15-Datasets/

23. L. LABORATORY. 1998 DARPA Intrusion Detection Evaluation Dataset; 1998. https://www.ll.mit.edu/r-d/datasets/1998-darpa-intrusion-detection-evaluation-dataset

24. Janarthanan T, Zargari S. Feature selection in UNSW-NB15 and KDDCUP'99 datasets. In: 2017 IEEE 26th International Symposium on Industrial Electronics (ISIE); 2017:1881-1886.

25. Tavallaee M, Bagheri E, Lu W, Ghorbani AA. A detailed analysis of the KDD CUP 99 data set. In: 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications; 2009:1-6.

26. Moustafa N. Designing an Online and Reliable Statistical Anomaly Detection Framework for Dealing with Large High-speed Network Traffic. University of New South Wales; 2017.

27. Zou H, Hastie T. Regularization and variable selection via the elastic net. J R Stat Soc Series B Stat Methodology. 2005;67:301-320.

28. Cieslak DA, Chawla NV. Learning decision trees for unbalanced data. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases; 2008:241-256.

29. Cao P, Zhao D, Zaiane O. An optimized cost-sensitive SVM for imbalanced data learning. In: Pacific-Asia Conference on Knowledge Discovery and Data Mining; 2013:280-292.

30. Köknar-Tezel S, Latecki LJ. Improving SVM classification on imbalanced data sets in distance spaces. In: 2009 Ninth IEEE International Conference on Data Mining; 2009:259-267.

31. Imam T, Ting KM, Kamruzzaman J. z-SVM: an SVM for improved classification of imbalanced data. In: Australasian Joint Conference on Artificial Intelligence; 2006:264-273.

32. Kwon BC, Eysenbach B, Verma J, et al. Clustervision: visual supervision of unsupervised clustering. IEEE Trans Vis Comput Graph. 2017;24:142-151.

33. Taverniers S, Tartakovsky DM. Estimation of distributions via multilevel Monte Carlo with stratified sampling. J Comput Phys. 2020;419:109572.

34. Johnstone IM, Lu AY. Sparse principal components analysis. arXiv preprint arXiv:0901.4392; 2009.

35. Pezzotti N, Lelieveldt BP, van der Maaten L, Höllt T, Eisemann E, Vilanova A. Approximated and user steerable tSNE for progressive visualization of large data. IEEE Trans Vis Comput Graph. 2016;23:1739-1752.

36. Xiang L, Zhao G, Li Q, Hao W, Li F. TUMK-ELM: a fast unsupervised heterogeneous data learning approach. IEEE Access. 2018;6:35305-35315.

37. Tibshirani R, Hastie T, Narasimhan B, Chu G. Diagnosis of multiple cancer types by shrunk centroids of gene expression. Proc Natl Acad Sci. 2002;99:6567-6572.

38. Mahalanobis PC. On the Generalized Distance in Statistics; 1936.

39. ur Rehman N, Khan B, Naveed K. Data-driven multivariate signal denoising using Mahalanobis distance. IEEE Sign Process Lett. 2019;26:1408-1412.

How to cite this article: Zoghi Z, Serpen G. UNSW-NB15 computer security dataset: Analysis through visualization. Security and Privacy. 2024;7(1):e331. doi: 10.1002/spy2.331
## APPENDIX A. STATISTICS

| Input features | Analysis (*mean ± SD [min, max] (25th, 50th, 75th)) | Backdoor (*mean ± SD [min, max] (25th, 50th, 75th)) |
|----------------|--------------------------------------------------|--------------------------------------------------|
| dur            | 1.80 ± 8.22 [0, 59.93]                           | 2.66 ± 10.57 [0, 59.88]                          |
| proto          | 120 [0, 132] (86, 113, 120)                      | 120 [0, 132] (78, 113, 120)                      |
| service        | 0 [0, 9] (0, 0, 5)                               | 0 [0, 6] (0, 0, 0)                               |
| state          | 3 [0, 5] (2, 3, 3)                               | 3 [0, 5] (3, 3, 3)                               |
| spkts          | 2 [1, 250] (2, 2, 10)                            | 2 [1, 250] (2, 2, 4)                            |
| dpkts          | 0 [0, 24] (0, 0, 8)                              | 0 [0, 24] (0, 0, 8)                              |
| sbytes         | 1029.64 ± 200.00 [60, 248562]                    | 1834.58 ± 14960.50 [60, 248562]                 |
| dbytes         | 361.15 ± 655.57 [0, 14384]                       | 341.10 ± 2892.93 [0, 43634]                     |
| rate           | 1400 31.12 ± 180 814.75 [0, 1 000 000]           | 147 399.80 ± 176 744.32 [0, 1 000 000]          |
| sttl           | 254 [0, 254] (62, 254, 254)                      | 254 [0, 254] (254, 254, 254)                    |
| dttl           | 0 [0, 252] (0, 0, 252)                           | 0 [0, 252] (0, 0, 0)                            |
| sload          | 111 326 829.8 ± 143 402 398.1 [0, 800 000 000]   | 119 315 006.0 ± 154 453 911.9 [0, 2 560 000 000] |
| dload          | 2999.13 ± 6259.99 [0, 117 945.69]                | 2867.02 ± 17 958.32 [0, 376 243.56]             |
| sloss          | 0 [0, 6] (0, 0, 2)                               | 0 [0, 11] (0, 0, 0)                             |
| dloss          | 0 [0, 7] (0, 0, 2)                               | 0 [0, 21] (0, 0, 0)                             |
| sinpkt         | 183.44 ± 901.32 [0, 9217.83]                     | 102.81 ± 438.76 [0, 8527.32]                    |
| dinpkt         | 174.45 ± 1079.27 [0, 11 522.34]                  | 18.60 ± 85.83 [0, 2662.34]                      |
| sjit           | 13 686.02 ± 87 150.01 [0, 917 472.73]            | 1051.85 ± 5209.51 [0, 172 261.94]               |
| djit           | 458.84 ± 2929.19 [0, 30 356.49]                  | 55.30 ± 344.22 [0, 6483.55]                     |
| swin           | 0 [0, 255] (0, 0, 255)                           | 0 [0, 255] (0, 0, 0)                            |
| stcpb          | 611 173 005.90 ± 1 188 689 972 [0, 4 294 130 034]| 348 699 461.95 ± 956 864 802.02 [0, 4 293 292 977] |
| dtcpb          | 605 002 960.80 ± 1 154 202 269.1 [0, 4 280 019 001]| 328 400 124.29 ± 915 495 553.97 [0, 4 269 957 855]| |
| dwin           | 0 [0, 255] (0, 0, 255)                           | 0 [0, 255] (0, 0, 0)                            |
| tcprtt         | 0.045 ± 0.079 [0, 0.381]                         | 0.020 ± 0.054 [0, 0.433]                        |
| synack         | 0.024 ± 0.043 [0, 0.294]                         | 0.009 ± 0.028 [0, 0.226]                        |
| ackdat         | 0.021 ± 0.039 [0, 0.221]                         | 0.011 ± 0.028 [0, 0.245]                        |
| smean          | 100 [39, 994] (84, 100, 100)                     | 100 [38, 994] (100, 100, 100)                    |
| dmean          | 0 [0, 799] (0, 0, 118.75)                        | 0 [0, 944] (0, 0, 0)                            |
| trans_depth    | 0 [0, 1] (0, 0, 1)                               | 0 [0, 1] (0, 0, 0)                              |
| response_body_len | 52.23 ± 167.78 [0, 6581]                      | 15.46 ± 302.17 [0, 6314]                         |
| ct_srv_src     | 3 [1, 14] (3, 5, 6)                              | 1 [1, 14] (2, 4, 6)                             |
| ct_state_ttl   | 2 [0, 6] (1, 2, 2)                               | 2 [0, 6] (2, 2, 2)                              |
| ct_dst_ttl     | 2 [1, 19] (1, 2, 3)                              | 1 [1, 34] (1, 2, 3)                             |
| ct_src_dport_ttl | 1 [1, 10] (1, 2, 3)                           | 1 [1, 10] (1, 2, 3)                             |
| ct_dport_ttl   | 1 [1, 10] (1, 2, 3)                              | 1 [1, 10] (1, 2, 3)                             |
| ct_dst_ttl     | 3 [1, 17] (3, 5, 6.75)                           | 1 [1, 28] (1, 4, 6)                             |
### Input features

| Feature                  | Analysis (*mean ± SD [min, max] (25th, 50th, 75th)) | Backdoor (*mean ± SD [min, max] (25th, 50th, 75th)) |
|--------------------------|-----------------------------------------------------|-----------------------------------------------------|
| is_ftp_login             | 0 [0, 0] (0, 0, 0)                                  | 0 [0, 0] (0, 0, 0)                                  |
| ct_ftp_cmd               | 0 [0, 0] (0, 0, 0)                                  | 0 [0, 0] (0, 0, 0)                                  |
| ct_flw_http_mthd         | 0 [0, 30] (0, 0, 1)                                 | 0 [0, 4] (0, 0, 0)                                 |
| ct_src_ltm               | 2 [1, 42] (2, 3, 5)                                 | 1 [1, 37] (2, 3, 5)                                |
| ct_srv_dst               | 3 [1, 14] (3, 5, 6)                                 | 1 [1, 14] (1, 4, 6)                                |
| is_sm_ips_ports          | 0 [0, 0] (0, 0, 0)                                  | 0 [0, 0] (0, 0, 0)                                  |

### Input features

| Feature                  | DoS (*mean ± SD [min, max] (25th, 50th, 75th)) | Exploits (*mean ± SD/mode [min, max] (25th, 50th, 75th)) |
|--------------------------|------------------------------------------------|----------------------------------------------------------|
| dur                      | 2.57 ± 10.44 [0, 59.99]                          | 2.26 ± 7.84 [0, 59.99]                                   |
| proto                    | 120 [0, 132] (78, 113, 120)                      | 113 [0, 132] (113, 113, 113)                            |
| service                  | 0 [0, 12] (0, 0, 0)                              | 0 [0, 12] (0, 0, 5)                                    |
| state                    | 3 [0, 6] (3, 3, 3)                               | 2 [0, 6] (2, 2, 3)                                    |
| spkts                    | 2 [1, 8070] (2, 2, 4)                            | 2 [1, 9616] (2, 10, 20)                                |
| dpkts                    | 0 [0, 10974] (0, 0, 0)                           | 0 [0, 3698] (0, 8, 18)                                 |
| sbytes                   | 17 272.01 ± 296 395.64 [60, 10 678 012]          | 28 344.32 ± 354 103.20 [60, 12 965 233]                |
| dbytes                   | 18 969.22 ± 367 400.22 [0, 14 655 550]           | 16 702.78 ± 130 884.36 [0, 4 994 486]                  |
| rate                     | 151 574.78 ± 186 871.67 [0, 1 000 000]           | 73 646.76 ± 151 451.78 [0, 1 000 000]                  |
| sttl                     | 254 [0, 255] (254, 254, 254)                      | 254 [0, 254] (62, 254, 254)                            |
| dttl                     | 0 [0, 252] (0, 0, 0)                             | 252 [0, 252] (0, 252, 252)                             |
| sload                    | 126 281 382.6 ± 178 605 855.7 [0, 3 698 666 496] | 65 310 277.41 ± 159 777 894.09 [0, 4 908 000 256]     |
| dload                    | 15 009.76 ± 111 400.45 [0, 2 097 525.25]         | 53 272.99 ± 181 154.49 [0, 2 447 663.25]               |
| sloss                    | 0 [0, 4033] (0, 0, 0)                            | 0 [0, 4803] (0, 2, 3)                                  |
| dloss                    | 0 [0, 5484] (0, 0, 0)                            | 0 [0, 1846] (0, 1, 5)                                  |
| spkts                    | 87.55 ± 415.36 [0, 8527.32]                      | 95.80 ± 357.19 [0, 11 915.52]                          |
| dpkts                    | 18.23 ± 60.58 [0, 2122.85]                        | 58.76 ± 207.43 [0, 11 913.56]                          |
| sjit                     | 903.99 ± 3058.58 [0, 117 271.51]                 | 3625.93 ± 19 136.34 [0, 1 181 164.23]                 |
| djit                     | 111.47 ± 805.09 [0, 34 727.03]                    | 995.95 ± 3202.80 [0, 159 677.54]                      |
| swin                     | 0 [0, 255] (0, 0, 0)                             | 255 [0, 255] (0, 255, 255)                             |
| stcpb                    | 399 181 475.26 ± 994 324 538.35 [0, 4 293 671 787] | 1 263 513 569.79 ± 1 425 896 603.28 [0, 4 294 958 913] |
| dtcpb                    | 400 756 918.39 ± 996 406 460.55 [0, 4 294 535 454] | 1 271 355 303.01 ± 1 422 554 423.98 [0, 4 294 776 227]|
| dwin                     | 0 [0, 255] (0, 0, 0)                             | 255 [0, 255] (0, 255, 255)                             |
| tcprrt                   | 0.022 ± 0.053 [0, 0.434]                         | 0.073 ± 0.079 [0, 1.491]                               |
| synack                   | 0.010 ± 0.027 [0, 0.241]                         | 0.035 ± 0.043 [0, 0.513]                               |
| ackdat                   | 0.011 ± 0.027 [0, 0.302]                         | 0.038 ± 0.042 [0, 1.424]                               |
| Input features         | DoS (*mean ± SD [min, max] (25th, 50th, 75th)) | Exploits (*mean ± SD [mode [min, max] (25th, 50th, 75th)) |
|------------------------|-----------------------------------------------|----------------------------------------------------------|
| smean                  | 100 [38, 1457] (100, 100, 100)                | 100 [39, 1493] (78, 100, 107)                            |
| dmean                  | 0 [0, 1371] (0, 0, 0)                         | 0 [0, 1454] (0, 45, 134)                                 |
| trans_depth            | 0 [0, 5] (0, 0, 0)                            | 0 [0, 172] (0, 0, 0)                                     |
| response_body_len      | 3691.69 ± 120084.87 [0, 5242880]              | 2609.40 ± 70 017.73 [0, 6 558 056]                      |
| ct_srv_src             | 1 [1, 36] (2, 4, 6)                           | 1 [1, 45] (1, 1, 4)                                     |
| ct_state_ttl           | 2 [0, 6] (2, 2, 2)                            | 1 [0, 6] (1, 1, 2)                                      |
| ct_dst_ltm             | 1 [1, 45] (1, 2, 3)                           | 1 [1, 48] (1, 1, 2)                                     |
| ct_src_dport_ltm       | 1 [1, 33] (1, 2, 3)                           | 1 [1, 25] (1, 1, 2)                                     |
| ct_dst_sport_ltm       | 1 [1, 10] (1, 2, 3)                           | 1 [1, 10] (1, 1, 2)                                     |
| ct_dst_src_ltm         | 1 [1, 45] (1, 4, 6)                           | 1 [1, 47] (1, 2, 4)                                     |
| is_ftp_login           | 0 [0, 1] (0, 0, 0)                            | 0 [0, 4] (0, 0, 0)                                      |
| ct_ftp_cmd             | 0 [0, 1] (0, 0, 0)                            | 0 [0, 4] (0, 0, 0)                                      |
| ct_flw_http_mthd       | 0 [0, 6] (0, 0, 0)                            | 0 [0, 4] (0, 0, 0)                                      |
| ct_src_ltm             | 1 [1, 45] (1, 2, 4)                           | 1 [1, 60] (1, 2, 3)                                     |
| ct_srv_dst             | 1 [1, 20] (1, 4, 6)                           | 1 [1, 44] (1, 1, 4)                                     |
| is_sm_ips_ports        | 0 [0, 0] (0, 0, 0)                            | 0 [0, 0] (0, 0, 0)                                      |
| **Input features**     | **Fuzzers (*mean ± SD [mode [min, max] (25th, 50th, 75th))** | **Generic (*mean ± SD [mode [min, max] (25th, 50th, 75th))** |
| dur                    | 2.85 ± 9.92 [0, 59.99]                        | 0.061 ± 1.41 [0, 59.97]                                 |
| proto                  | 113 [0, 132] (113, 113, 119)                  | 119 [0, 132] (119, 119, 119)                            |
| service                | 0 [0, 5] (0, 0, 0)                            | 2 [0, 12] (2, 2, 2)                                     |
| state                  | 2 [0, 5] (2, 2, 3)                            | 3 [0, 6] (3, 3, 3)                                      |
| spkts                  | 2 [1, 1638] (2, 10, 12)                       | 2 [1, 2722] (2, 2, 2)                                   |
| dpkts                  | 0 [0, 436] (0, 6, 8)                          | 0 [0, 6494] (0, 0, 0)                                   |
| sbytes                 | 7286.69 ± 45511.59 [60, 2162963]              | 396.04 ± 25 671.98 [60, 3 592 470]                      |
| dbytes                 | 520.60 ± 5933.89 [0, 581063]                  | 1078.60 ± 56 924.36 [0, 8 645 430]                      |
| rate                   | 67 191.44 ± 145 567.33 [0, 1 000 000]         | 216 796.96 ± 190 140.81 [0, 1 000 000]                  |
| sttl                   | 254 [0, 255] (254, 254, 254)                  | 254 [0, 254] (254, 254, 254)                            |
| dttl                   | 252 [0, 252] (0, 252, 252)                    | 0 [0, 252] (0, 0, 0)                                    |
| sload                  | 134 145 333.44 ± 380 420 274.76 [0, 5 988 000 256] | 101 219 111.49 ± 104 600 264.24 [0, 5 600 000 000]      |
| dload                  | 3170.83 ± 15 282.36 [0, 1 732 368.37]         | 2212.96 ± 44 406.88 [0, 2 025 781.62]                   |
| sloss                  | 0 [0, 816] (0, 2, 3)                          | 0 [0, 1358] (0, 0, 0)                                   |
| dloss                  | 0 [0, 216] (0, 1, 2)                          | 0 [0, 3246] (0, 0, 0)                                   |
| sinpkt                 | 421.69 ± 2733.73 [0, 84 371.49]               | 2.41 ± 45.64 [0, 3773.82]                               |
| dinpkt                 | 290.26 ± 1388.06 [0, 19 658.66]               | 1.30 ± 29.66 [0, 3609.92]                               |
| sjit                   | 19 946.07 ± 99 203.20 [0, 1 174 993.80]       | 92.98 ± 2199.46 [0, 237 428.96]                         |
| djit                   | 770.71 ± 4121.47 [0, 188 290.91]             | 24.42 ± 755.96 [0, 75 556.74]                           |
| Input features | Fuzzers (*mean ± SD/mode [min, max] (25th, 50th, 75th)) | Generic (*mean ± SD/mode [min, max] (25th, 50th, 75th)) |
|---------------|-------------------------------------------------------|--------------------------------------------------------|
| swin          | 255 [0, 255] (0, 255, 255)                            | 0 [0, 255] (0, 0, 0)                                   |
| stcpb         | 1,378,147,300.36 ± 1,424,305,833.94 [0, 4,294,781,981] | 26,469,091.16 ± 274,343,059.15 [0, 4,291,804,097]    |
| dtcpb         | 1,388,281,645.76 ± 1,429,798,823.20 [0, 4,294,190,217] | 25,319,590.95 ± 264,881,744.37 [0, 4,288,821,039]    |
| dwin          | 255 [0, 255] (0, 255, 255)                            | 0 [0, 255] (0, 0, 0)                                   |
| tcprtt        | 0.101 ± 0.088 [0, 0.736]                              | 0.002 ± 0.016 [0, 0.380]                               |
| synack        | 0.053 ± 0.048 [0, 0.422]                              | 0.0008 ± 0.008 [0, 0.193]                              |
| ackdat        | 0.047 ± 0.044 [0, 0.369]                              | 0.0008 ± 0.008 [0, 0.244]                              |
| smeans        | 52 [38, 1504] (57, 89, 252)                           | 57 [39, 1372] (57, 57, 57)                            |
| dmeans        | 0 [0, 1333] (0, 44, 55)                               | 0 [0, 1392] (0, 0, 0)                                 |
| trans_depth   | 0 [0, 1] (0, 0, 0)                                    | 0 [0, 1] (0, 0, 0)                                    |
| response_body_len | 2.06 ± 94.50 [0, 10,244]                        | 589.01 ± 40,724.94 [0, 5,242,880]                     |
| ct_srv_src    | 3 [1, 52] (2, 4, 7)                                   | 33 [1, 52] (18, 26, 33)                               |
| ct_state_ttl  | 1 [0, 6] (1, 1, 2)                                    | 2 [0, 6] (2, 2, 2)                                    |
| ct_dst_ltm    | 1 [1, 47] (1, 2, 2)                                   | 16 [1, 51] (13, 17, 19)                               |
| ct_src_dport_ltm | 1 [1, 18] (1, 1, 2)                                | 16 [1, 51] (13, 16, 18)                               |
| ct_dst_sport_ltm | 1 [1, 10] (1, 1, 1)                                | 16 [1, 51] (11, 16, 17)                               |
| ct_dst_src_ltm | 2 [1, 46] (2, 3, 5)                                   | 33 [1, 52] (18, 26, 33)                               |
| is_ftp_login  | 0 [0, 2] (0, 0, 0)                                    | 0 [0, 0] (0, 0, 0)                                    |
| ct_ftp_cmd    | 0 [0, 2] (0, 0, 0)                                    | 0 [0, 0] (0, 0, 0)                                    |
| ct_flw_http_mthd | 0 [0, 6] (0, 0, 0)                                | 0 [0, 4] (0, 0, 0)                                    |
| ct_src_ltm    | 1 [1, 44] (1, 2, 3)                                   | 17 [1, 51] (14, 17, 19)                               |
| ct_srv_dst    | 2 [1, 44] (2, 3, 6)                                   | 33 [1, 52] (18, 26, 33)                               |
| is_sm_ips_ports | 0 [0, 0] (0, 0, 0)                                   | 0 [0, 0] (0, 0, 0)                                    |
| Input features | Normal (*mean ± SD/mode [min, max] (25th, 50th, 75th)) | Reconnaissance (*mean ± SD/mode [min, max] (25th, 50th, 75th)) |
| dur           | 1.017 ± 4.859 [0, 59.99]                              | 1.060 ± 5.198 [0, 59.99]                              |
| proto         | 113 [6, 119] (113, 113, 113)                          | 113 [0, 132] (113, 113, 113)                          |
| service       | 0 [0, 11] (0, 0, 2)                                   | 0 [0, 10] (0, 0, 0)                                   |
| state         | 2 [0, 8] (2, 2, 2)                                    | 3 [0, 6] (2, 3, 3)                                    |
| spkts         | 2 [1, 656] (4, 12, 40)                                | 2 [1, 818] (2, 10, 10)                                |
| dpkts         | 2 [0, 1,716] (2, 10, 40)                              | 0 [0, 5,254] (0, 0, 8)                                 |
| sbytes        | 4105.70 ± 11,348.05 [28, 338,718]                     | 772.14 ± 7473.66 [60, 248,562]                        |
| dbytes        | 31,049.46 ± 139,991.25 [0, 2,249,492]                 | 2291.04 ± 110,561.36 [0, 7085,342]                    |
| rate          | 13,799.31 ± 68,031.39 [0, 1,000,000]                  | 103,843.88 ± 174,492.04 [0, 1,000,000]                |
| sttl          | 31 [0, 255] (31, 31, 31)                              | 254 [0, 254] (254, 254, 254)                          |
| dttl          | 29 [0, 254] (29, 29, 29)                              | 0 [0, 252] (0, 0, 252)                                |
| sload         | 23,170,702.73 ± 151,427,436.45 [0, 5,344,000,000]     | 73,682,736.35 ± 126,687,364.06 [0, 2,560,000,000]     |
| Input features | Normal (*mean ± SD/mode [min, max] (25th, 50th, 75th)) | Reconnaissance (*mean ± SD/mode [min, max] (25th, 50th, 75th)) |
|----------------|--------------------------------------------------|--------------------------------------------------|
| dload          | 2062.949.24 ± 3 935 216.82 [0, 22 422 730]       | 2384.24 ± 25 884.75 [0, 1 523 372.37]            |
| sloss          | 0 [0, 144] (0, 3, 7)                              | 0 [0, 3] (0, 0, 2)                                |
| dloss          | 0 [0, 856] (0, 2, 8)                              | 0 [0, 2627] (0, 0, 1)                            |
| sinpkt         | 2847.88 ± 12 510.28 [0, 60 009.92]               | 72.62 ± 225.48 [0, 8527.32]                      |
| dnpkt          | 121.19 ± 1527.09 [0, 56 716.82]                  | 67.55 ± 105.55 [0, 2486.84]                      |
| sjit           | 5440.29 ± 50 340.80 [0, 1 460 480.01]            | 3508.90 ± 5643.01 [0, 145 278.35]                |
| djit           | 965.21 ± 6208.86 [0, 289 388.26]                 | 107.69 ± 216.28 [0, 7821.94]                     |
| swin           | 255 [0, 255] (0, 255, 255)                       | 0 [0, 255] (0, 0, 255)                           |
| stcpb          | 1 473 765 351.39 ± 1 431 070 038.32 [0, 4 294 813 520] | 1 038 646 894.00 ± 1 369 863 952.73 [0, 4 292 312 477] |
| dtcpb          | 1 463 697 455.08 ± 1 427 233 138.98 [0, 4 294 881 924] | 1 053 946 882.33 ± 1 393 449 806.83 [0, 4 294 228 344] |
| dwin           | 255 [0, 255] (0, 255, 255)                       | 0 [0, 255] (0, 0, 255)                           |
| tcprtt         | 0.031 ± 0.089 [0, 2.518]                         | 0.060 ± 0.075 [0, 0.634]                         |
| synack         | 0.017 ± 0.050 [0, 2.100]                         | 0.028 ± 0.040 [0, 0.486]                         |
| ackdat         | 0.014 ± 0.045 [0, 1.520]                         | 0.031 ± 0.039 [0, 0.456]                         |
| smean          | 73 [28, 1499] (59, 73, 130)                      | 84 [38, 994] (56, 84, 84)                        |
| dmean          | 0 [0, 1458] (53, 89, 482)                        | 0 [0, 1354] (0, 0, 44)                           |
| trans_depth    | 0 [0, 2] (0, 0, 0)                               | 0 [0, 1] (0, 0, 0)                               |
| response_body_len | 3834.69 ± 43 613.48 [0, 1 090 037]               | 35.50 ± 2027.45 [0, 193 683]                     |
| ct_srv_src     | 2 [1, 63] (2, 4, 7)                              | 1 [1, 40] (1, 1, 3)                              |
| ct_state_ttl   | 0 [0, 6] (0, 0, 1)                               | 2 [0, 6] (1, 2, 2)                               |
| ct_dst_ltm     | 2 [1, 46] (2, 3, 4)                              | 1 [1, 48] (1, 1, 1)                              |
| ct_src_dport_ltm | 1 [1, 46] (1, 1, 2)                          | 1 [1, 10] (1, 1, 1)                              |
| ct_dst_sport_ltm | 1 [1, 46] (1, 1, 1)                          | 1 [1, 10] (1, 1, 1)                              |
| ct_dst_src_ltm | 1 [1, 63] (1, 2, 4)                              | 1 [1, 65] (1, 1, 1)                              |
| trans_depth    | 0 [0, 2] (0, 0, 0)                               | 0 [0, 0] (0, 0, 0)                               |
| is_ftp_login    | 0 [0, 2] (0, 0, 0)                               | 0 [0, 0] (0, 0, 0)                               |
| ct_ftp_cmd      | 0 [0, 2] (0, 0, 0)                               | 0 [0, 0] (0, 0, 0)                               |
| ct_fiw_http_mthd | 0 [0, 12] (0, 0, 0)                          | 0 [0, 6] (0, 0, 0)                               |
| ct_src_ltm      | 2 [1, 47] (2, 3, 5)                              | 1 [1, 45] (1, 1, 2)                              |
| ct_srv_dst      | 2 [1, 62] (2, 4, 7)                              | 1 [1, 28] (1, 1, 2)                              |
| ct_ftp_login    | 0 [0, 2] (0, 0, 0)                               | 0 [0, 0] (0, 0, 0)                               |

| Input features | Shellcode (*mean ± SD/mode [min, max] (25th, 50th, 75th)) | Worms (*mean ± SD/mode [min, max] (25th, 50th, 75th)) |
|----------------|----------------------------------------------------------|------------------------------------------------------|
| dur            | 0.366 ± 0.813 [0, 12.34]                                 | 1.428 ± 2.316 [0, 15.89]                             |
| proto          | 119 [113, 119] (113, 119, 119)                           | 113 [113, 119] (113, 113, 113)                      |
| service        | 0 [0, 0] (0, 0, 0)                                       | 5 [0, 5] (5, 5, 5)                                  |
| state          | 3 [2, 3] (2, 3, 3)                                       | 2 [0, 3] (2, 2, 2)                                  |
| spkts          | 2 [2, 10] (2, 2, 10)                                     | 10 [2, 116] (10, 10, 10)                            |
| Input features | Shellcode (*mean ± SD/mode [min, max] (25th, 50th, 75th)) | Worms (*mean ± SD/mode [min, max] (25th, 50th, 75th)) |
|----------------|----------------------------------------------------------|---------------------------------------------------------|
| dpkts          | 0 [0, 8] (0, 0, 6)                                        | 6 [0, 698] (6, 6, 8)                                   |
| sbytes         | 529.74 ± 309.07 [102, 1862]                               | 2403.43 ± 6926.38 [92, 78 481]                        |
| dbytes         | 145.33 ± 150.52 [0, 354]                                  | 79 681.68 ± 207 906.61 [0, 915 302]                  |
| rate           | 100 977.77 ± 164 580.25 [1.37, 1 000 000]                | 20 857.49 ± 70 057.20 [1.07, 500 000]                 |
| sttl           | 254 [254, 254] (254, 254, 254)                            | 254 [62, 254] (254, 254, 254)                         |
| dttl           | 0 [0, 252] (0, 0, 252)                                   | 252 [0, 252] (252, 252, 252)                          |
| sload          | 132 999 703.31 ± 258 545 258.36 [293.01, 2 976 000 000]  | 73 755 358.09 ± 289 779 617.15 [585.62, 1 640 000 000] |
| dload          | 2305.32 ± 3014.74 [0, 10 789.26]                          | 129 753.08 ± 293 317.53 [0, 1 178 738.12]            |
| sloss          | 0 [0, 2] (0, 0, 2)                                        | 2 [0, 30] (2, 2, 2)                                   |
| dloss          | 0 [0, 1] (0, 0, 1)                                        | 1 [0, 347] (1, 1, 1)                                  |
| sinpkt         | 39.49 ± 89.89 [0, 1371.17]                                | 78.10 ± 160.48 [0.002, 1765.24]                       |
| dinpkt         | 55.16 ± 114.93 [0, 1739.00]                               | 92.19 ± 204.99 [0, 2256.68]                           |
| sjit           | 2364.19 ± 5851.86 [0, 88 534.59]                          | 4612.98 ± 10 817.45 [0, 119 205.59]                  |
| djit           | 89.41 ± 187.12 [0, 2708.36]                               | 415.56 ± 649.04 [0, 3604.42]                          |
| swin           | 0 [0, 255] (0, 0, 255)                                   | 255 [0, 255] (255, 255, 255)                          |
| stcpb          | 1 068 748 668.47 ± 1 399 201 223.78 [0, 4 288 814 251]   | 2 099 246 662.89 ± 1 398 455 334.24 [0, 4 261 080 249] |
| dtcpb          | 1 067 921 650.32 ± 1 399 379 479.12 [0, 4 282 661 650]   | 1 845 824 939.01 ± 1 369 175 366.54 [0, 4 187 928 499] |
| dwin           | 0 [0, 255] (0, 0, 255)                                   | 255 [0, 255] (255, 255, 255)                          |
| tcprtt         | 0.062 ± 0.078 [0, 0.457]                                 | 0.123 ± 0.073 [0, 0.402]                              |
| synack         | 0.029 ± 0.041 [0, 0.265]                                 | 0.059 ± 0.037 [0, 0.150]                              |
| ackdat         | 0.032 ± 0.041 [0, 0.232]                                 | 0.064 ± 0.042 [0, 0.296]                              |
| smean          | 72 [49, 737] (62, 87, 130)                               | 130 [45, 1189] (101, 130, 131)                        |
| dmean          | 0 [0, 45] (0, 0, 45)                                     | 45 [0, 1361] (44, 45, 45)                             |
| trans_depth    | 0 [0, 0] (0, 0, 0)                                       | 1 [0, 1] (1, 1, 1)                                   |
| response_body_len | 0.00 ± 0.00 [0, 0]                                | 36 358.20 ± 96 391.94 [0, 443 364]                    |
| ct_svr_src     | 1 [1, 36] (1, 1, 1)                                     | 1 [1, 6] (1, 1, 2)                                   |
| ct_state_ttl   | 2 [1, 2] (1, 2, 2)                                      | 1 [0, 2] (1, 1, 1)                                   |
| ct_dst_ltm     | 1 [1, 46] (1, 1, 1)                                     | 1 [1, 27] (1, 1, 1)                                  |
| ct_src_dport_ltm | 1 [1, 1] (1, 1, 1)                                    | 1 [1, 3] (1, 1, 1)                                   |
| ct_dst_sport_ltm | 1 [1, 1] (1, 1, 1)                                    | 1 [1, 2] (1, 1, 1)                                   |
| ct_dst_sport_ltm | 1 [1, 46] (1, 1, 1)                                   | 1 [1, 8] (1, 1, 1)                                   |
| is_ftp_login   | 0 [0, 0] (0, 0, 0)                                      | 0 [0, 0] (0, 0, 0)                                   |
| ct_fip_cmd     | 0 [0, 0] (0, 0, 0)                                      | 0 [0, 0] (0, 0, 0)                                   |
| ct_flw_http_mthd | 0 [0, 0] (0, 0, 0)                                     | 0 [0, 4] (1, 1, 1)                                   |
| ct_src_ltm     | 1 [1, 45] (1, 1, 2)                                     | 1 [1, 37] (1, 1, 2)                                  |
| ct_srv_dst     | 1 [1, 36] (1, 1, 1)                                     | 1 [1, 7] (1, 1, 1)                                   |
| is_sm_ips_ports | 0 [0, 0] (0, 0, 0)                                     | 0 [0, 0] (0, 0, 0)                                   |
APPENDIX B. PARALLEL COORDINATE DIAGRAM OF ANALYSIS CLASS

APPENDIX C. PARALLEL COORDINATE DIAGRAM OF BACKDOOR CLASS
APPENDIX D. PARALLEL COORDINATE DIAGRAM OF DOS CLASS

APPENDIX E. PARALLEL COORDINATE DIAGRAM OF EXPLOITS CLASS
APPENDIX F. PARALLEL COORDINATE DIAGRAM OF FUZZERS CLASS

APPENDIX G. PARALLEL COORDINATE DIAGRAM OF GENERIC CLASS
APPENDIX H. PARALLEL COORDINATE DIAGRAM OF NORMAL CLASS

APPENDIX I. PARALLEL COORDINATE DIAGRAM OF RECONNAISSANCE CLASS
APPENDIX J. PARALLEL COORDINATE DIAGRAM OF SHELLCODE CLASS

APPENDIX K. PARALLEL COORDINATE DIAGRAM OF WORMS CLASS