Visualization Of Network Intrusions in Simulated Environments: UNSW-NB 15 Dataset

Divija Kalluri (MS in Computer Science, University of Houston, dkalluri@courgarnet.uh.edu)
Charan Gajjala Chenchu (MS in Computer Science, University of Houston, cgajjala@cougarnet.uh.edu)

***Abstract*— This paper aims to show the implementation of the Dashboard to Detect Network Intrusions in Simulated Environments of the UNSW-NB15 Dataset, it visualizes the issues inherent in the original Dataset that can be further addressed by researchers before working on this UNSW-NB15 dataset for any Machine learning classifier model development. The main steps include Data acquisition and cleaning, data preprocessing, label encoding for conversion of nominal features, Visualizations include bar charts, heat maps, Elastic net Algorithm and Random Forest, PCA, t-SNE, LDA, Elastic net Algorithm and Random Forest. Analysis through visualizations showcased the class imbalances and class overlaps present in the UNSW-NB15 Dataset, these are the important problems to be addressed before feeding into any model.**

Keywords— Intrusion Detection, Visualization, Dashboard, Feature Selection, Dimensionality Reduction

I. INTRODUCTION

The UNSW-NB15 computer network security dataset was released in 2015 (Moustafa & Slay, 2015). This dataset is comprised of 2,540,044 realistic modern normal and abnormal (also known as attack) network activities, The dataset is available on the UNSW web page, it consists of 49 feature columns, out of which there are 2 target feature columns namely label, attack class, the other feature columns are non-target features. Flow-based features are crafted by analyzing the sequential movement of packets as they traverse the network from a source to a destination. Key characteristics such as direction, inter-packet length, and inter-arrival times play a pivotal role in shaping these features. Notable examples of flow-based features include total duration (dur) and destination-to-source-time-to-live (dttl). These features are systematically divided into three sets: basic features (6 to 18), content features (19 to 26), and time features (27 to 35). Additionally, features 36 to 40 and 41 to 47 are designated as general-purpose features.

A. Problem Statement

The target feature attack_cat column consists of 10 different attacks such as normal , generic, Analysis, Backdoor, DoS, Exploits, Fuzzers, Generic, Reconnaissance, Shellcode and Worms, the count of them individually are Normal,Analysis, Backdoor, DoS, Exploits, Fuzzers, Generic, Reconnaissance, Shellcode and Worms.from the counts in the dataset we can see a significant lack of balance in the datasets 87% of the dataset constitutes of normal attacks . Designing a Dashboard to visualize the intrusion detection in the dataset is the main task so as to detect the class imbalances and class overlap, the bar chart and the Mahanoblis distance heatmaps help us see the class imbalances, for rectifying the redundant features problem elastic net algorithm and Random forest are applied where as further more on class overlap can be visualized using the dimensionality reduction techniques such as PCA (Principal Component Analysis), LDA (Linear discriminant Analysis), t-sne (t-Distributed Stochastic Neighbor Embedding) , k-means (inter-distance overlap) clustering.

B. Motivation

A dashboard can provide a comprehensive visual representation of attack classes, allowing for a clearer understanding of the distribution, patterns, and characteristics of different attack types present in the dataset. Visualizations such as histograms, pie charts, or geographical maps can offer intuitive insights into the frequency, prevalence, and geographical spread of attacks. By presenting information in a visually digestible format, a dashboard facilitates rapid decision-making. Security personnel or analysts can quickly spot irregularities or clusters of specific attacks, allowing them to promptly investigate and respond to potential security threats.

II. RELATED WORK

This research introduces interactive methods, termed "probing," for understanding dimensionality-reduced data visualizations. By enabling error observation, questioning of element positions, data point comparison, and visualization of dimension influences, these tools aid in interpreting complex multidimensional datasets. The study evaluates these techniques through a web-based system, gathering insights from data analysts using the prototype. [1]

To improve network security, this study explores intrusion detection datasets, namely the UNSW-NB15 dataset, using machine learning. It focuses on feature analysis, investigating feature relevance and high dimensionality to address data shortage concerns. Using machine learning, it finds important attributes and attempts to simplify processing by eliminating those that are unnecessary, suggesting a subset for enhanced intrusion detection. This study is enhanced by a comparative comparison with earlier approaches used on the KDD'99 dataset. [2]

Through the optimization of feature selection from the UNSW-NB15 dataset, this work seeks to improve intrusion detection systems (NIDS). Using a combination of Random Forest and Decision Tree classifiers with Python's Anaconda3, it extracts four salient attributes out of 45, improving the accuracy of discerning network assaults from normalcy through Deep Learning. [3]

This work presents "t-SNE," a method for multi-scale structure portrayal and enhanced clarity when displaying high-dimensional data. By using an altered version of Stochastic Neighbor Embedding, t-SNE produces better visuals by overcoming crowding problems and outperforming other visualization methods such as Sammon mapping and Iso-map on a variety of datasets. [4]

To uncover underlying problems such class imbalance and overlap, this study visually examines the UNSW-NB15 dataset for network security. PCA, t-SNE, and K-means are used in this process. The results of the investigation highlight the need to resolve these issues before using the dataset to create classifier models since they may influence performance and accuracy. [5]

III. METHODOLOGY

A. Workflow

The Major steps involved in the process of finding class overlap and class imbalances are shown in Figure 1.1

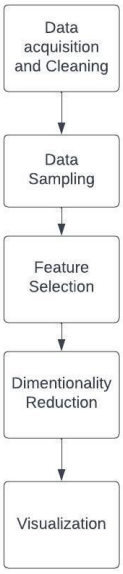

Figure 1.1 workflow

B. Data Aquisition and Cleaning

The dataset available in the UNSW web page consists of the original dataset distributed into 4 csv files, which is further combined into a single csv file, which is read in python using the pandas dataframe , all the Nan values are removed.

B. Data Preprocessing

Label encoding is performed on the whole dataset to convert all the nominal values into numerical features for the input feature columns , the nominal columns such as (proto, service) except for the attack_cat column. The feature columns in the data set are wide spread some columns have higher range of values and some other columns have lower range values , we will be performing scaling so as to convert the values in the feature columns to fall in the range 0 to 1, there are a total of 6 different scaling techniques applied for the original dataset they are(min-max scaler, robust scaler, standard scaler, quantile transformer, and power transformer),We employ the nearest shrunken centroid method to assess each attack and normal class type. Subsequently, we calculate the Mahalanobis distance between class centroids, both before and after the application of scalers.

D. Feature Selection

1) There are many popular algorithms for feature selection, the most commonly used and well known algorithms are Elastic Net Algorithm and Random Forest, since the min-max scaling was working fine out of all the scaling techniques we used the min max scaled dataset to be fed to the elastic net algorithm and Random forest, the Elastic Net algorithm significantly reduces the input feature and helps us find better visualizations and also reduces the complexity of computation, the algorithm is implemented in python using the ElasticNetCV with the hyperparameters (folds=5,L1 ratio = 0.5), we get an optimal alpha to lie in between 0.002 to 0.004 and upon applying grid search cv we will get the optimal alpha which can be then applied for important feature selection. Below is the pseudo code for it.

```
FUNCTION Elastic_Net_Algorithm(data)
   // Data preprocessing
   - Drop highly correlated features
   - Create a correlation matrix heatmap

   // Data splitting and scaling
   - Split data into train and test sets
   - Scale the data

   // Model training and evaluation
   - Find best alpha using ElasticNetCV
   - Fit model with the best alpha
   - Evaluate model performance for different alphas

   // Feature importance
   - Identify important features
END FUNCTION
```

2) The Random Forest Algorithm is the other feature selection algorithm commonly used we implement the Random Forest with hyperparameters (n_estimators = 100), below is the pseudo code for it

```
FUNCTION
generate_feature_importance_path(data)
   // Dropping highly correlated features
   numerical_features=
data.select_dtypes(include=[np.number])
   correlation_matrix =
numerical_features.corr().abs()
   upper_tri=
correlation_matrix.where(np.triu(np.ones(correlatio
n_matrix.shape), k=1).astype(np.bool_))
   correlated_features = [column for column in
upper_tri.columns if any(upper_tri[column] >
0.95)]
   data_filtered = data.drop(correlated_features,
axis=1)

   // Preparing data for modeling
```

```
    X = data_filtered.drop(['attack_cat', 'label'],
axis=1)
    y = data['label']
    y = LabelEncoder().fit_transform(y)
    X_train, X_test, y_train, y_test =
train_test_split(X, y, test_size=0.2,
random_state=42)

    // Training a Random Forest model
    rf = RandomForestClassifier(n_estimators=100,
random_state=42)
    rf.fit(X_train, y_train)

    // Extracting important features
    important_features = rf.feature_importances_
END FUNCTION
```

E. Dimensionality Reduction

1) Algorithms such as PCA , LDA, t-sne and K-means overlap are applied for dimensionality reduction and class overlap detection,From the extracted feature columns we run our dimensionality reduction techniques such as PCA(principal Component Analysis) both in 2D and 3D, we create a 2D plot from PCA,to essentially represent the data in a two-dimensional space where the axes correspond to the principal components, similarly for the PCA in 3D as well, below is the listed pseudocode for PCA implementation

```
FUNCTION apply_pca(data, n_components)
    pca = PCA(n_components=n_components) //
      Initialize PCA with specified components
    features_to_drop = ['attack_cat', 'label'] //
      Columns to drop from the data
    data_without_labels =
      data.drop(features_to_drop, axis=1) // Remove
      specified columns
    pca_result =
      pca.fit_transform(data_without_labels) //
      Apply PCA on the modified data
    RETURN pca_result // Return the transformed
      data
END FUNCTION
```

2) LDA is also one of the dimensionality reduction technique useful for visualizing the attacks of attack_class int both 2D and 3D plots, the implementation is done to visualize the attack_class properly in both 2D and 3D, below lists the implementation of LDA

```
FUNCTION apply_lda(data, n_components)
    lda =
LinearDiscriminantAnalysis(n_components=n_compon
ents) // Initialize LDA with specified components
    X = data.drop(['attack_cat', 'label'], axis=1) // Features
(independent variables)
    y = data['attack_cat'] // Target variable
    lda_result = lda.fit_transform(X, y) // Apply LDA on
the features and target variable
    RETURN lda_result // Return the transformed data
END FUNCTION
```

F. t-sne (t-Distributed Stochastic Neighbor Embedding)

t-sne is applied to visualize the data spread of different attacks in the attack_class of our UNSW-NB15 dataset, the t-SNE aims to preserve the pairwise similarities between data points.

G. K-means Inter cluster distance map

K-means clustering helps us identify the class overlap in our UNSW-NB15 Dataset, the above extracted shrunken centroid distance using Mahalanobis is used again in the plotting of k-means Inter cluster distance map, below is the pseudo code for it.

```
FUNCTION update_kmeans_plot(pathname)
    // Apply PCA on a copy of the training data
    X = apply_pca(train_df.copy(), 2)

    // Extract 'attack_cat' labels from the training
data
    labels = train_df['attack_cat']

    // Compute centroids using NearestCentroid
    shrunken_centroids =
NearestCentroid(shrink_threshold=None)
    shrunken_centroids.fit(X, labels)
    class_centroids = shrunken_centroids.centroids_

    // Initialize KMeans with the centroids obtained
from NearestCentroid
    kmeans =
KMeans(n_clusters=len(class_centroids),
init=class_centroids, n_init=1, max_iter=1)
    kmeans.fit(X)

    // Retrieve cluster centers and sizes
    cluster_centers = kmeans.cluster_centers_
    cluster_sizes = np.bincount(kmeans.labels_)

    // Create a new empty figure for the plot
    fig = go.Figure()

    // Iterate through cluster centers to plot text
markers for each center
    FOR i FROM 0 TO length(cluster_centers) - 1
DO
        center = go.Scatter(
            x=[cluster_centers[i][0]],
            y=[cluster_centers[i][1]],
            mode='text',
            text=[str(i)],
            marker=dict(size=5,
color='rgba(255,0,0,1)'),
            showlegend=False,
        )
        fig.add_trace(center)

        // Plot circles around cluster centers with sizes
based on cluster sizes
        circle = go.Scatter(
            x=[cluster_centers[i][0]],
            y=[cluster_centers[i][1]],
            mode='markers',
            marker=dict(size=np.sqrt(cluster_sizes[i]),
color='rgba(155,155,223,0.56)'),
            name=f'Cluster {i}',
        )
        fig.add_trace(circle)
```

END FOR

// Return the constructed figure
RETURN fig
END FUNCTION

H. Contribution

Divija Kalluri: Data Acquisition and cleaning of the UNSW-NB Dataset, applying stratified sampling, development of dimensionality reduction techniques such as PCA, t-sne and feature selection algorithm such as Random Forest.

Charan Gajjala Chenchu: Development of the Dashboard using plotly, worked with the feature selection algorithm (Elastic net algorithm), Dimensionality reduction technique (LDA), and k mean Intercluster distance mapping.

IV. RESULTS AND SOLUTION

A.Dashboard

The Dashboard consists of the three main sections 1. Basic Visualization, which consists of Bar charts and mahalanobis Distance Heatmaps, 2. Feature Selection consists of Algorithms such as Elastic Net Algorithm and Random Forest Algorithm, 3. Class Imbalance Visualization consists of the PCA, LDA, t-sne, K-means intercluster distance mapping. The homepage of the Dashboard can be seen below in Figure 1.2

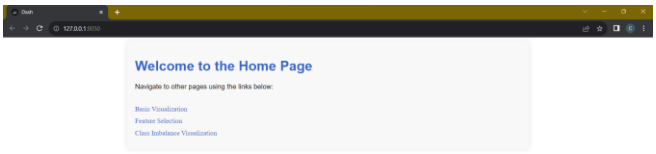

Figure 1.2- Homepage

A.Basic Visualization

1.Bar Charts

The Bar Charts for attack_class has been plotted to see the distribution of the different attack classes across the dataset, The results in Figure 2.1 depict that the Normal attack class is dominant of all attacks constituting to around 87% of the Dataset and the least one being Backdoors, the class imbalances problem is partially shown through the bar charts.

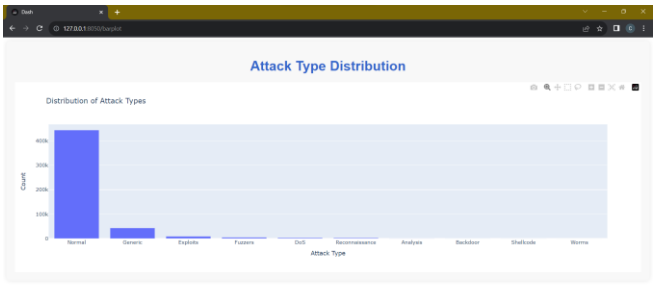

Figure 2.1- Bar Chart for attack_class

2. Scaling

The findings are visually represented through seven heatmaps (Figure 2.2). Below displays the original dataset heatmap and six additional heatmaps illustrating the distances between class centroids after scaler application. The color spectrum, ranging from red to green, is used to depict centroid distances; a red cell signifies closer centroids for corresponding classes, while a green cell indicates greater separation between class centroids (Figure 2.2) and (Figure2.3). The min-max scaler in (Figure 2.2) relocates values to a range where class centroids are positioned at greater distances from each other, causing data points belonging to a specific attack category to cluster together. This characteristic of the min-max scaler contributes to reduced overlap and enhanced clarity in data representation, potentially leading to improved performance in the subsequent phases of machine learning model classification. The heat maps visualization partially shows the class imbalance and overlap problem.

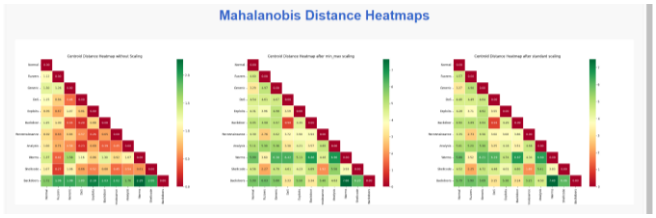

Figure 2.2 original, min-max scaler, robust scaler

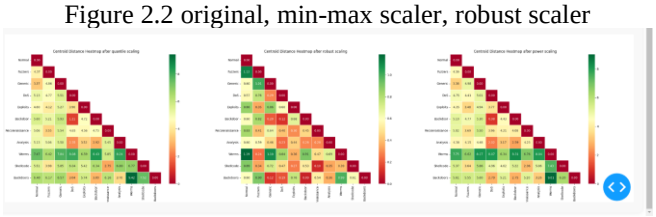

Figure2.3 - standard scaler, quantile transformer, and power transformer

B. Feature Selection

1. Elastic Net Algorithm

For the elastic Net Algorithm the optimal alpha has been sorted out to be 0.003 from the range 0.001 to 0.004 as there is high overlap between train MSE and test MSE in that range, the plot for finding the optimum alpha against training and testing is shown in Figure 3.1, Upon running this algorithm we were able to extract 25 features as important with the help of the regularization graph (Figure3.2), all the coefficients equal to 0 are considered to be not important, the

bar chart is also extracted to see which features are given more importance the features with 0 coefficient value has no bar (Figure 3.3), for the evaluation of the elastic net algorithm we plot the R-square vs alpha is plotted against train and test and we can see that there is less error in the plot (Figure 3.4)

Figure 3.1 Mean squared error graph.

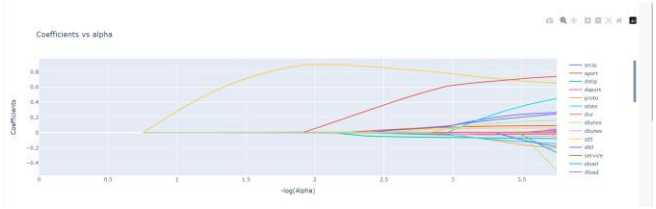

Figure 3.2 Regularization plot

Figure 3.3 Important Features

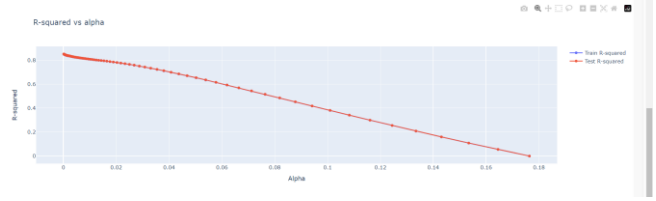

Figure 3.4 R-squared Vs Alpha

2. Random Forest Algorithm

The Random Forest Algorithm was able to extract 35 feature columns as important but the drawback with this algorithm is it was picking some of the unnecessary feature columns such as dur, fp_login etc. This algorithm is more vulnerable, we can see in Figure (3.5) below the columns with no bars are non-important feature columns and the others are considered to be important feature columns.

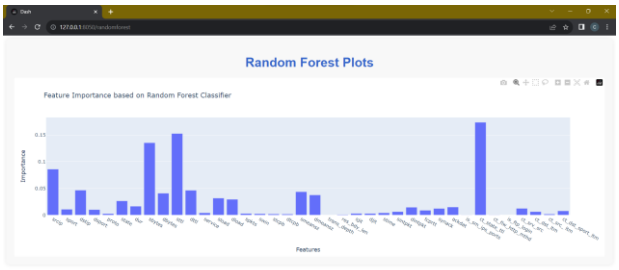

Figure 3.5 Feature Selection for Random Forest

From both the Algorithms we can confirm that the Elastic Net Algorithm with an optimal alpha of 0.003 is ought to be performing good when compared to Random Forest with 100 n-estimators, so we employ elastic net algorithm and extract the important features to further visualize the dataset using PCA, t-sne, LDA and k-means inter-cluster distance mapping.

C. Class Imbalance Visualization

1. t-sne (t-Distributed Stochastic Neighbor Embedding)

In the 2D plot we can see clusters of points representing similar instances. Points that are close together in the plot are more like each other in the high-dimensional space. The Figure4.1 shows different attack classes mimicking the behavior of the Normal attack class, It is effective at revealing the local structure and capturing the finer details in the data.

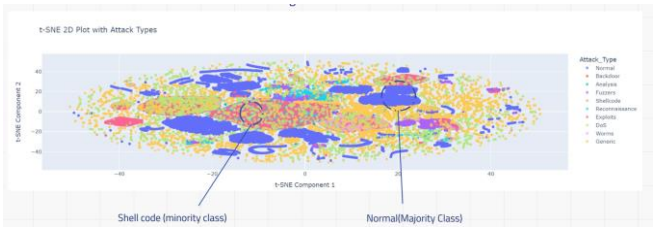

Figure 4.1 t-sne plot in 2D

2. PCA(Principal Component Analysis)

Here in 2D we can visualize the distribution of different attacks in the form of clusters. The below figure 4.2 shows us the plot for PCA in 2D for all the attack classes, we can Identify the significant overlap of the clusters of all the attacks in attack class.

Figure 4.2 PCA for all attacks in 2D

The PCA in 3D is well suited to visualize plots for individual attack classes, The Dashboard is designed in such a way that one can toggle on different attack labels to individually see about the different attacks cluster, one can easily identify the information about outliers through the plots, the below Figure 4.3 shows the PCA of Exploit attack class in 3D. We can observe that the exploits attack class is divided into multiple subgroups which is not really desired by any classifier model.

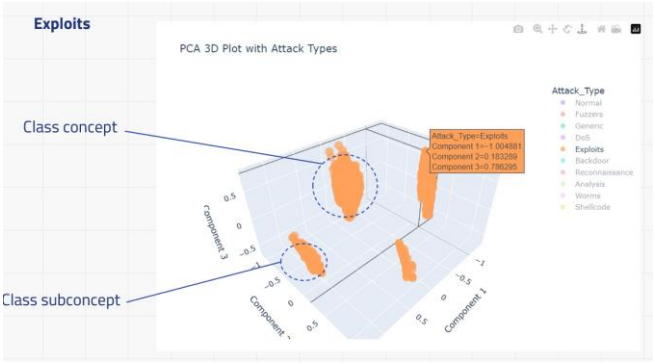

Figure 4.3 PCA for Exploit class in 3D

The overlap of all the attack classes is seen in Figure 4.2, a 3D plot visualization of it gives us more fine details when compared to the 2D plot.

Figure 4.2 PCA in 3D

3.LDA(Linear Discriminant Analysis)

The below Figure4.3 shows the visualization in 2D, similar to PCA we can see there is a significant overlap of attacks.

Figure 4.3 LDA in 2D

In Figure 4.4, the 3D plot of all the attack classes is shown, we can observe a clutter of all those attack classes, whereas PCA gave us a proper visualization for overlap in 3D, we can observe the plot for individual attack classes as well.

Figure 4.4 LDA in 3D

4.K-means Intercluster Distance map

The different attacks (10) in the attack class column are grouped into their respective clusters (from cluster 0 to cluster9) and the overlap of those clusters can be clearly see in the below figure, we can now come to a conclusion through k-means that the attacks are not well separated. And some classes such as Fuzzers completely overlap with the Normal attack class, it is clearly visible in the below Figure 4.5

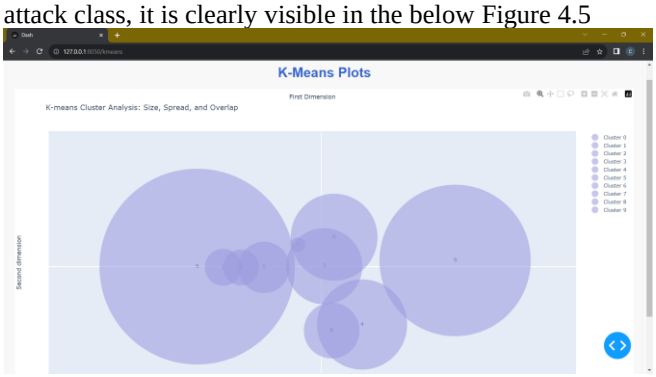

Figure 4.5 K-means Intercluster Distance Overlap

V. FUTURE WORK AND CONCLUSION

1) Upon examination of this dataset's Exploits attack class, several noteworthy findings emerge. To begin with, the Exploits class has a heterogeneous structure made up of several bigger clusters of different sizes, some of which are further divided into smaller groupings that form separate but related entities. Compactness in these clusters suggests rather tight intra-cluster data points. But there are obvious gaps among the bigger categories, suggesting divisions or differences within these subgroups. Furthermore, the Exploits class exhibits a class overlap situation in which several occurrences closely mirror the actions noted in the Normal records. This overlap problem makes it more difficult to discern between actions that are aggressive and those that are not. Moreover, the dataset exhibits class imbalances, which indicate uneven proportions across various assault types and may affect the model's capacity to correctly identify these occurrences. In conclusion, the visualizations show significant overlaps between behaviors that are attacks and those that are not, complex cluster structures within the Exploits class, and imbalances among various attack kinds, all of which make accurate categorization and analysis difficult.

2) In further research, more complex clustering methods like density-based or hierarchical clustering may be

investigated to better represent the complex subgroupings and gaps found inside the bigger clusters. Furthermore, the classification accuracy may be improved by using anomaly detection techniques, particularly if one is skilled at identifying minute differences and overlaps between attack and non-attack behaviors. As new attack behaviors appear or change over time, real-time or continuous monitoring and model improvement utilizing streaming data can assist adjust and enhance the model's performance. By using an adaptive strategy, the model would be guaranteed to continue its usefulness in recognizing and categorizing new threats within the attack class.

VI. LIMITATIONS

1.Algorithmic Restrictions: The ability of clustering algorithms to capture complex patterns and overlaps in attack_class assaults may be constrained. Certain algorithms could find it difficult to distinguish minute differences between attack and non-attack behaviors or to handle intricate cluster structures.

2. Data Quality: The quality and representativeness of the data set could pose limitations.

3. Unbalanced Data: Unbalanced classes, particularly within attack classes, cause models to be skewed in favor of majority classes, which affects how accurately minority attack types are classified.

4. Static Analysis: Changes in assault patterns over time or temporal variations may not be taken into consideration. The model may lose effectiveness over time if it is not updated or modified in response to evolving attack strategies.

5. Interpretability: Complex clustering structures and overlaps could make it challenging to interpret and explain the model's decisions, impacting its usability and trustworthiness, especially in sensitive domains.

6. Resource Intensiveness: Some advanced algorithms or techniques proposed for handling overlaps and complex structures might require significant computational resources or time, limiting their practicality in real-time or resource-constrained environments.

REFERENCES

[1] J. Stahnke, M. Dork, B. Muller, and A. Thom, “Probing Projections: Interaction Techniques for Interpreting Arrangements and Errors of Dimensionality Reductions,” IEEE Transactions on Visualization and Computer Graphics, vol. 22, no. 1, pp. 629–638, Jan. 2016, doi: https://doi.org/10.1109/tvcg.2015.2467717.

[2] T. Janarthanan and S. Zargari, “Feature selection in UNSW-NB15 and KDDCUP’99 datasets,” IEEE Xplore, Jun. 01, 2017. https://ieeexplore.ieee.org/abstract/document/8001537.

[3] Kanimozhi, V. & Jacob, Prem. (2019). UNSW-NB15 dataset feature selection and network intrusion detection using deep learning. International Journal of Recent Technology and Engineering. 7. 443-446.

[4] van der Maaten, Laurens & Hinton, Geoffrey. (2008). Viualizing data using t-SNE. Journal of Machine Learning Research. 9. 2579-2605.

[5] Zoghi, Zeinab & Serpen, Gursel. (2021). UNSW-NB15 Computer Security Dataset: Analysis through Visualization.