International Journal of Innovative Research and Scientific Studies, 8(2) 2025, pages: 4012-4026

IJIRSS
International Journal of Innovative Research & Scientific Studies

ISSN: 2617-6548

URL: www.ijirss.com

Analysis of the dynamics of cyberattacks and fraud methods using machine learning algorithms for IIoT: Information security of digital twins in Industry 4.0

Saltanat Adilzhanova[1], Murat Kunelbayev[1], Gulshat Amirkanova[1], Gulnur Tyulepberdinova[1], Sybanova Dana[1*]

[1]*Cybersecurity and cryptology department, Al-Farabi Kazakh National University, Almaty, Kazakhstan.*
[1]*Artificial Intelligent and Big Data department, Al-Farabi Kazakh National University, Almaty, Kazakhstan.*

Corresponding author: Sybanova Dana (*Email: dsybanovaa@gmail.com*)

Abstract

This paper analyzes the dynamics of cyberattacks and fraud techniques using machine learning algorithms for Industrial Internet of Things (IIoT) systems. Special attention is paid to information security issues of digital twins within the framework of the Industry 4.0 concept. Various machine learning techniques, such as logistic regression, random forest, and the nearest neighbor method, are considered for classifying attacks in IIoT systems, taking into account the problem of data imbalance inherent in rare attacks. The use of data balancing and cross-validation methods can improve the accuracy of the models and minimize the number of false positives. The results demonstrate that the random forest model outperforms other methods in terms of accuracy, confirming its effectiveness for IIoT security and digital twin protection. Additionally, feature selection and optimization techniques were explored to enhance model performance further. The study also highlights the importance of real-time threat detection, which is crucial for maintaining the integrity and resilience of IIoT environments. Future research is planned to investigate more sophisticated methods, including deep learning, to improve attack detection and classification in the context of Industry 4.0. The integration of federated learning and adaptive AI-driven security measures may also offer promising solutions to emerging cyber threats.

Keywords: Attack detection, S Dynamics of cyberattacks, IIoT (Industrial Internet of Things), machine learning algorithms, MOTE algorithm.

DOI: 10.53894/ijirss.v8i2.6201
Funding: This study received no specific financial support.
History: Received: 4 March 2025 / **Revised:** 7 April 2025 / **Accepted:** 9 April 2025 / **Published:** 14 April 2025

Competing Interests: The authors declare that they have no competing interests.
Authors' Contributions: All authors contributed equally to the conception and design of the study. All authors have read and agreed to the published version of the manuscript.
Transparency: The authors confirm that the manuscript is an honest, accurate, and transparent account of the study; that no vital features of the study have been omitted; and that any discrepancies from the study as planned have been explained. This study followed all ethical practices during writing.
Acknowledgements: This work was supported by the project BR24992975 "Development of a digital twin of a food processing enterprise using artificial intelligence and IIoT technologies".
Publisher: Innovative Research Publishing

1. Introduction

This study focuses on developing an efficient Intrusion Detection System (IDS) to secure the Industrial Internet of Things (IIoT) in edge-based environments. With the increasing number of IIoT devices and decentralized data interactions, significant security challenges arise. The research examines modern IDS detection techniques and system architectures, introducing a hybrid IDS framework augmented by machine learning to enhance detection performance across various benchmarks [1]. Another paper presents an AI-driven analysis of cyber threats within IIoT environments, specifically using deep learning techniques for detecting and mitigating cyber risks [2]. Furthermore, an anomaly detection algorithm is introduced for IIoT data platforms, addressing the limitations of traditional machine learning methods. This algorithm, based on Meta-Heuristic Optimized Deep Random Neural Networks (MH-DRNN), optimizes feature selection and classification, improving detection accuracy and achieving up to 99.2% performance [3]. The integration of Digital Twins and blockchain-assisted federated learning (HFL) is explored in the context of Industry 4.0, aiming to enhance modeling and secure cross-verification in cyber-physical systems. The proposed solution outperforms existing models in network overhead, optimization, and accuracy [4]. Another study combines rule-based detection with machine learning methods to mitigate Distributed Denial of Service (DDoS) attacks in Cyber-Physical Production Systems (CPPS), achieving high accuracy and real-time effectiveness when trained using real-time network traffic [5]. Additionally, a hybrid deep neural network (DNN) is proposed for detecting and classifying DDoS attacks in Software-Defined IIoT networks, leveraging XGBoost for feature selection and combining CNN and LSTM networks. This approach delivers high accuracy and low latency, crucial for IIoT environments [6]. For real-time anomaly detection in Industry 4.0 systems, a hybrid machine-learning ensemble pipeline is introduced, incorporating models like LOF, One-Class SVM, and Autoencoders to improve detection accuracy. The effectiveness of the system is validated through performance evaluations on industrial machines [7]. A Human-Cyber-Physical System (HCPS) for real-time anomaly detection is also proposed, combining human knowledge with cyber-physical systems to improve decision-making, system reliability, and job creation in Industry 5.0 [8]. Moreover, the study explores a collaborative mutation-based moving target defense (CM-MTD) for Digital Twin Mobile Networks (DTMN), using LSTM and deep reinforcement learning to address security challenges and enhance network security, outperforming baseline solutions [9]. The role of deep learning in credit card fraud detection is also examined, highlighting various architectures and their impact on improving fraud detection accuracy, with a focus on overcoming challenges like data imbalance and overfitting [10]. The integration of Digital Twins and Artificial Intelligence (AI) in Industry 4.0 is discussed, emphasizing how AI enhances automation and decision-making processes to optimize industrial operations [11]. A comprehensive review of Digital Twin models explores their functions, challenges, and industrial applications, offering insights into overcoming obstacles for wider implementation [12]. Additionally, a Digital Twin-based approach is proposed for simulating security attacks in smart irrigation systems, integrating real-time sensor data with simulations to improve intrusion detection and system resilience in IoT applications [13]. Machine Learning (ML) and Deep Learning (DL) play a critical role in optimizing anomaly detection, resource allocation, and predictive models in Digital Twin Networks (DTN), revolutionizing industries and enhancing system performance [14]. The increasing use of Internet of Things (IoT) technologies in various sectors, including smart grids, healthcare, and industrial management, raises the need for a comprehensive IoT security model to protect devices and ensure data security at all levels [15]. The paper examines cybersecurity risks in the Industrial Internet of Things (IIoT), with a focus on data integrity and confidentiality. Using blockchain technologies, the research aims to create a solution for managing data integrity and establishing consensus among nodes in the network [16]. Security threats in Agricultural Technologies 4.0 and 5.0 are also explored, with proposed mitigation strategies including AI, blockchain, and quantum computing to improve malware detection and prevent Denial-of-Service (DoS) attacks [17]. The detection of credit card fraud is analyzed using machine learning and deep learning algorithms, highlighting the effectiveness of deep learning in addressing issues like imbalanced data and high false alarm rates, achieving 99.9% accuracy and significant improvements in fraud detection performance [18]. An innovative fraud detection system is also presented, using anti-Benford subgraphs and machine learning algorithms to enhance security in financial networks, achieving 94.83% accuracy in fraud detection [19]. For cryptocurrency systems, a method based on the trimmed k-means approach is proposed for detecting fraud in Bitcoin networks, offering a robust solution for identifying fraudulent transactions [20]. Digital trust in the context of Industry 4.0 and 5.0 is explored, emphasizing the importance of fraud prevention and data protection to maintain user trust and ensure the resilience of digital technologies [21]. SQL injection attacks (SQLIAs) are addressed by a new framework called DIAVA, which uses network traffic analysis and regular expressions for accurate attack detection, significantly outperforming traditional web application firewalls [22]. Credit card fraud detection is further enhanced through a combined use of LSTM and GRU neural networks with multilayer perceptrons, achieving high accuracy and minimizing false alarms [23]. Lastly, the use of machine learning and cyber threat intelligence (CTI) for enhancing security in cyber supply chains is discussed, predicting threats and developing control measures to improve security [24]. The detection and defense against DDoS attacks in SDN are also improved through an ensemble online machine learning method, achieving better detection accuracy and providing reliable protection against low-level and zero-day attacks [25].

The aim of this work is to develop and evaluate the effectiveness of machine learning models for classifying cyberattacks in the Industrial Internet of Things (IIoT) infrastructure. In particular, the application of Logistic Regression, Random Forest, and K-Nearest Neighbors models is investigated to identify and classify various types of attacks, with a special focus on addressing the issue of data imbalance, which is critical when classifying rare attacks. The main goal of the work is to improve the accuracy and efficiency of IIoT protection systems, thereby enhancing their resilience to modern cyber threats.

2. Method

To solve the problem of data classification, a methodology has been developed that includes several key stages. At the preprocessing stage, the data was cleaned and prepared for use in model training. Then, three machine learning algorithms were trained: logistic regression, random forest, and the nearest neighbors method. Their performance was assessed based on classification accuracy and ability to process rare classes. The process of building, training, and evaluating models is shown in the flow diagram below (Figure 1).

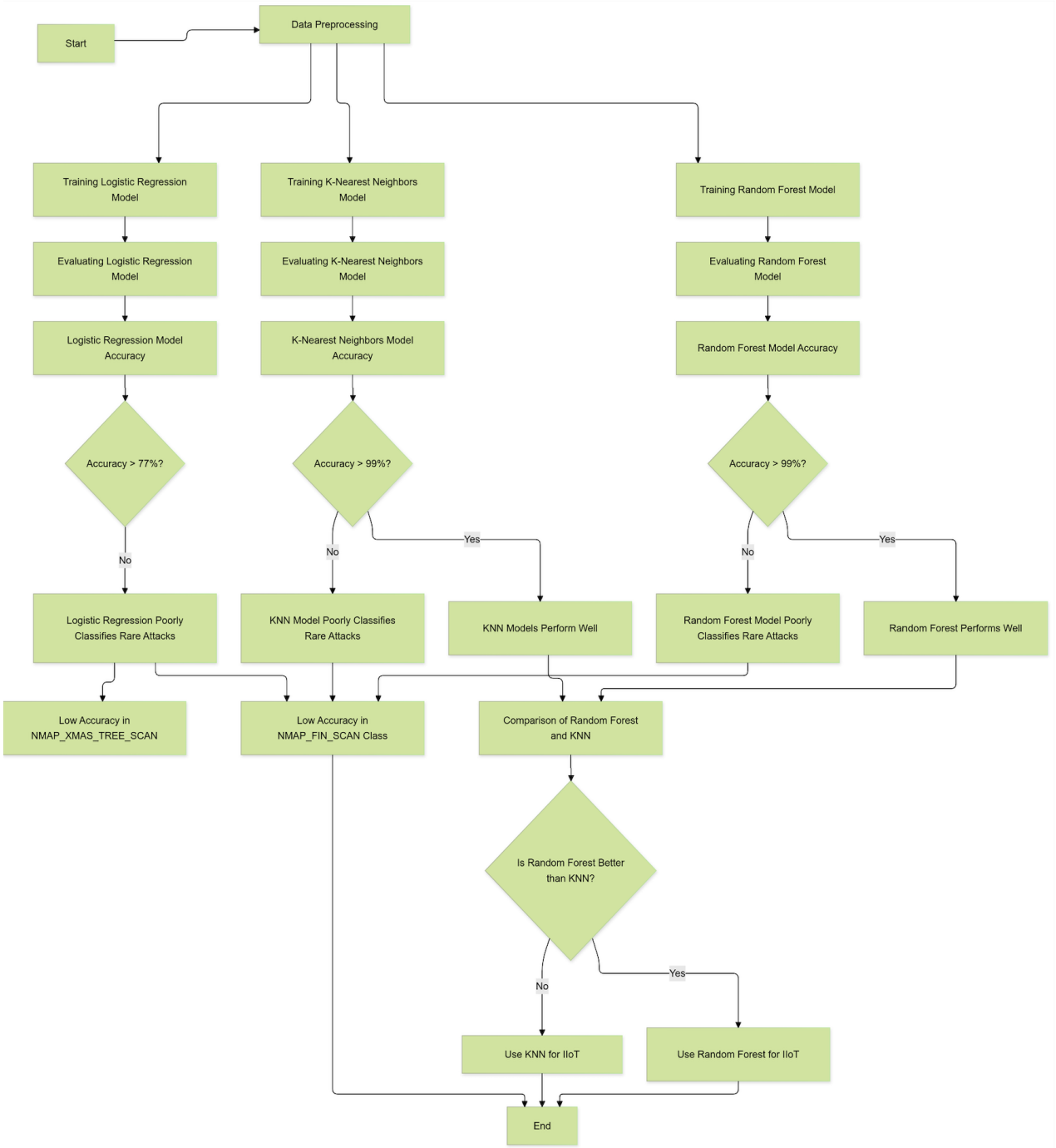

Figure 1.
Data Classification Process Flow Diagram.

The diagram shows the stages of building machine learning models, starting with data preprocessing, training three different algorithms, and evaluating their accuracy. The final model selection is based on specified performance criteria.

The hypothesis of this research is that the use of specially adapted machine learning models to work with imbalanced data in the context of classifying rare attacks in IIoT systems significantly increases the accuracy and effectiveness of threat detection, minimizing the number of false positives and improving the models' ability to generalize. This is achieved by applying data balancing methods, cross-validation, and selecting optimal models, which ultimately allows for a more effective response to complex and infrequent attacks, which is especially important for ensuring the security of industry and critical infrastructure.

The scientific novelty of this study lies in the definition of the model that most effectively copes with the classification of rare attacks. As part of the work, an architecture was developed for working with digital twin data, covering the process of constructing, training, and evaluating three machine learning models for data analysis in the context of threat detection in IIoT (Industrial Internet of Things) systems. The construction phase starts with the preparation of data for further analysis and training of models. At the Data Preprocessing stage, data is cleaned, converted, and prepared for submission to machine learning models. After training, each model is evaluated using test data. Based on the results obtained, threshold values are established. In the event that none of the models achieves the required accuracy, errors are analyzed to identify the cause: either the data was unbalanced or the models could not cope with the classification of rare attacks. The process ends with the selection of the optimal model to be used to classify attacks in IIoT systems.

We can also note the scientific significance of this study, which lies in the context of rapidly developing technologies and growing security threats in the field of IIoT, where this study plays an important role in ensuring the reliability and protection of such systems from complex and rare attacks. The study selected models specifically adapted to deal with unbalanced data, a characteristic problem in the analysis of rare attacks. This improves classification accuracy by minimizing the number of false positives. Each model was evaluated using a number of metrics such as accuracy, completeness, and F1 measures, which provided a comprehensive assessment of their effectiveness. Cross-validation techniques were used to improve the results, avoiding retraining and increasing the generalization capacity of the models. One of the main problems encountered during the work was the high degree of class imbalance in the data. A variety of balancing techniques have been applied to solve this problem, including random subsampling and the use of class weight algorithms. The results of this study can be directly used to improve the threat detection system in real IIoT environments, allowing more effective responses to rare and highly dangerous attacks, which is critical for industry and critical infrastructure.

3. Results and Discussion

This code, shown in Figure 2, creates a bar chart to compare three machine learning models (Logistic Regression, Random Forest, and K-Nearest Neighbors) on four metrics: Precision, Completeness, F1 score, and Accuracy. Each metric is displayed in a separate colored column, with precision values plotted above each column. The chart includes settings to enhance the visualization: indentation, legend, grid, and removal of unnecessary frames. The visualization is shown in Algorithm 1.

Algorithm 1. Code for building a bar chart comparing machine learning models by accuracy, completeness, F1-estimation, and precision labels.

```
models = ['Logistic Regression', 'Random Forest', 'K-Nearest Neighbors']
precision = [0.844262, 0.974953, 0.964729]
recall = [0.130545, 0.986692, 0.935995]
f1_score = [0.226125, 0.980787, 0.950145]
accuracy = [0.770670890188434, 0.9965074723846654, 0.9913092917478883]
x = np.arange(len(models))
# Plotting
plt.figure(figsize=(14, 8))
bar_width = 0.2
# Bar charts for different metrics
plt.bar(x - 1.5 * bar_width, precision, width=bar_width, label='Precision', color='steelblue', edgecolor='black', linewidth=1.2, alpha=0.6)
plt.bar(x - 0.5 * bar_width, recall, width=bar_width, label='Recall', color='firebrick', edgecolor='black', linewidth=1.2, alpha=0.6)
plt.bar(x + 0.5 * bar_width, f1_score, width=bar_width, label='F1-Score', color='seagreen', edgecolor='black', linewidth=1.2, alpha=0.6)
plt.bar(x + 1.5 * bar_width, accuracy, width=bar_width, label='Accuracy', color='darkorange', edgecolor='black', linewidth=1.2, alpha=0.6)
# Adding annotations for exact values
for i in range(len(models)):
    plt.text(x[i] - 1.5 * bar_width, precision[i] + 0.02, f'{precision[i]}', ha='center', va='bottom', fontsize=12, color='black')
    plt.text(x[i] - 0.5 * bar_width, recall[i] + 0.02, f'{recall[i]}', ha='center', va='bottom', fontsize=12, color='black')
    plt.text(x[i] + 0.5 * bar_width, f1_score[i] + 0.02, f'{f1_score[i]}', ha='center', va='bottom', fontsize=12, color='black')
    plt.text(x[i] + 1.5 * bar_width, accuracy[i] + 0.02, f'{accuracy[i]}', ha='center', va='bottom', fontsize=12, color='black')
# Customizing X axis labels
plt.xticks(x, models, fontsize=14, fontweight='bold')
plt.ylabel('Metrics', fontsize=14, fontweight='bold')
# Changing the position of the title
plt.title('Model Comparison by Metrics', fontsize=16, fontweight='bold', style='italic', pad=20)
# Moving metric names upwards
plt.legend(fontsize=12, title='Metrics', title_fontsize='13', loc='upper left', bbox_to_anchor=(1, 1))
# Adding grid lines for better readability
plt.grid(axis='y', linestyle='--', alpha=0.6)
# Removing the top and right borders
plt.gca().spines['top'].set_visible(False)
plt.gca().spines['right'].set_visible(False)
# Adjusting Y axis label position
plt.gca().yaxis.set_label_coords(-0.08, 0.5)
plt.tight_layout()  # For better graph display
plt.show()
```

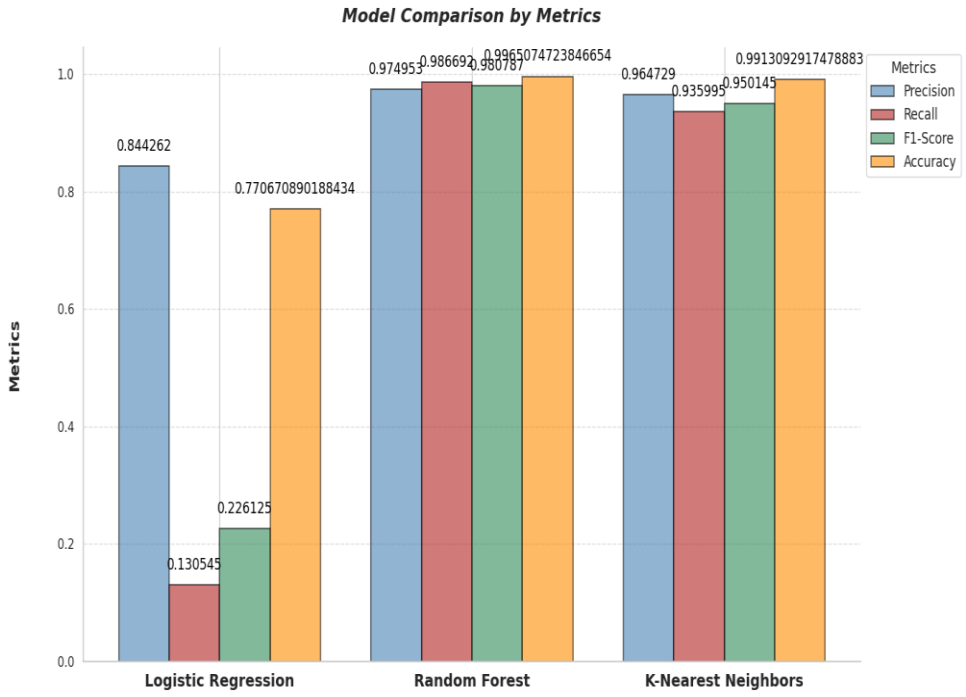

Figure 2.
Comparison of the duration and intensity of cyberattacks by type.

The code shown in Algorithm 2 plots the dynamics of cyberattacks by flow duration using the Seaborn and matplotlib libraries. For each type of attack, a line is displayed showing the relationship between flow duration and number of packets. A visualization of the results is shown in Figure 3.

Algorithm 2. Code to plot the dynamics of cyberattacks by flow duration

```
import matplotlib.pyplot as plt
import seaborn as sns
# Use Seaborn style for better aesthetics
sns.set(style="whitegrid", palette="muted")
# Unique attack types
attack_types = data['Attack_type'].unique()
# Create a figure with specified size
plt.figure(figsize=(12, 6))
# Loop through all attack types
for attack in attack_types:
    attack_data = data[data['Attack_type'] == attack]
    # Plot for each attack type, set color for each attack
    plt.plot(attack_data['flow_duration'], label=attack, lw=2)
# Set axis labels
plt.xlabel('Flow Duration (seconds)', fontsize=14, fontweight='bold')
plt.ylabel('Packet Count', fontsize=14, fontweight='bold')
# Title of the graph
plt.title('Cyberattack Dynamics by Flow Duration', fontsize=16, fontweight='bold')
# Set up the legend
plt.legend(title='Attack Type', fontsize=12, title_fontsize='13', loc='upper right')
# Add grid for better readability
plt.grid(True, linestyle='--', alpha=0.7)
# Set axis tick label font size
plt.xticks(fontsize=12)
plt.yticks(fontsize=12)
# Improve layout
plt.tight_layout()
# Show the plot
plt.show()
```

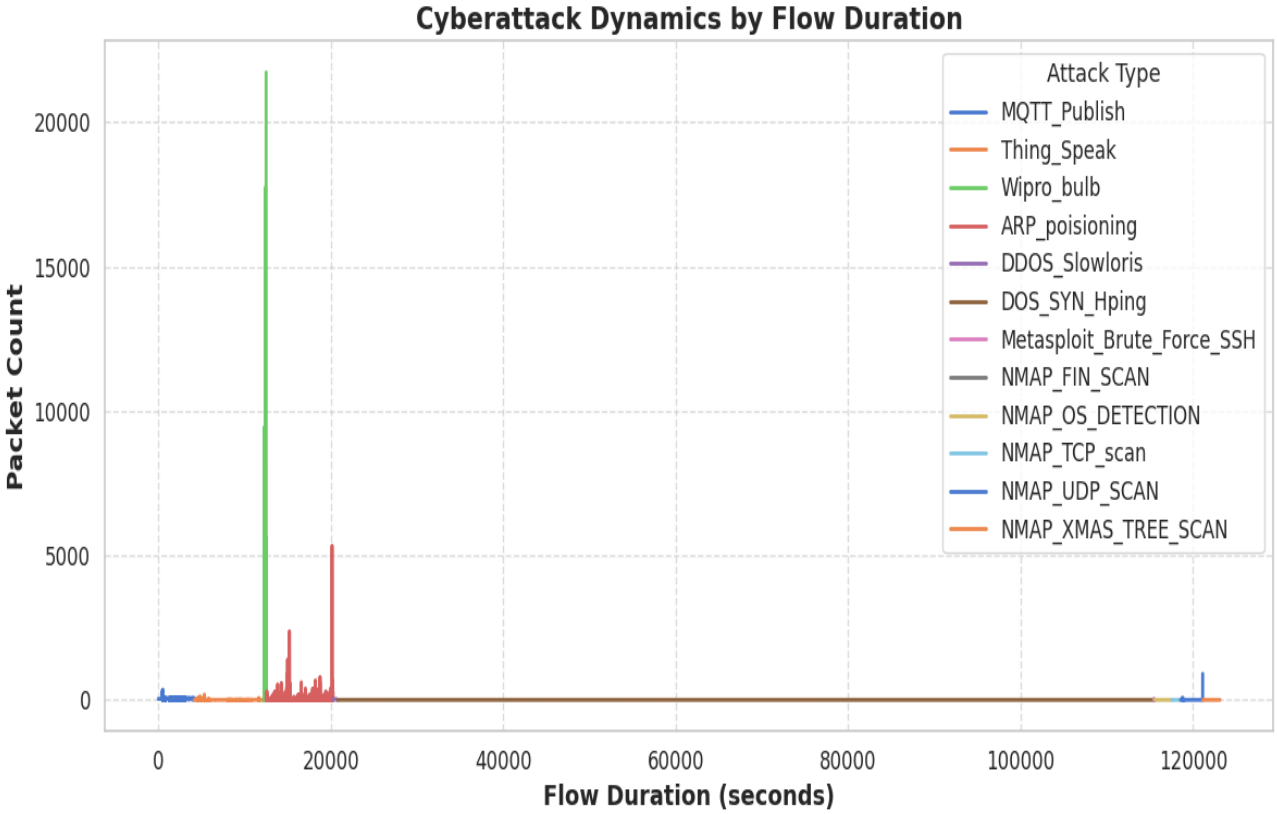

Figure 3.
Cyberattack Dynamics by Flow Duration.

The code in Algorithm 3 plots the change in flow duration over time using the matplotlib library. The graph displays the flow duration with the maximum value annotated, allowing key points to be visualized. A visualization of the results is presented in Figure 4.

Algorithm 3. Code for plotting flow duration over time with annotation of maximum value

```
import matplotlib.pyplot as plt

# Assuming the 'flow_duration' column exists in data
plt.figure(figsize=(10, 6))  # Increase the figure size

# Plotting the graph with improvements
plt.plot(data['flow_duration'], label='Flow Duration', color='b', linestyle='-', linewidth=2)

# Set axis labels
plt.xlabel('Index', fontsize=12)
plt.ylabel('Flow Duration (seconds)', fontsize=12)

# Title with increased font size
plt.title('Flow Duration over Time', fontsize=14)

# Add grid for better readability
plt.grid(True)

# Show legend
plt.legend(loc='upper right')

# Add annotation on the graph, if necessary
plt.annotate('Maximum Value', xy=(data['flow_duration'].idxmax(), data['flow_duration'].max()),
             xytext=(data['flow_duration'].idxmax()+10, data['flow_duration'].max()),
             arrowprops=dict(facecolor='red', arrowstyle='->'),
             fontsize=10)

# Display the plot
plt.tight_layout()
plt.show()
```

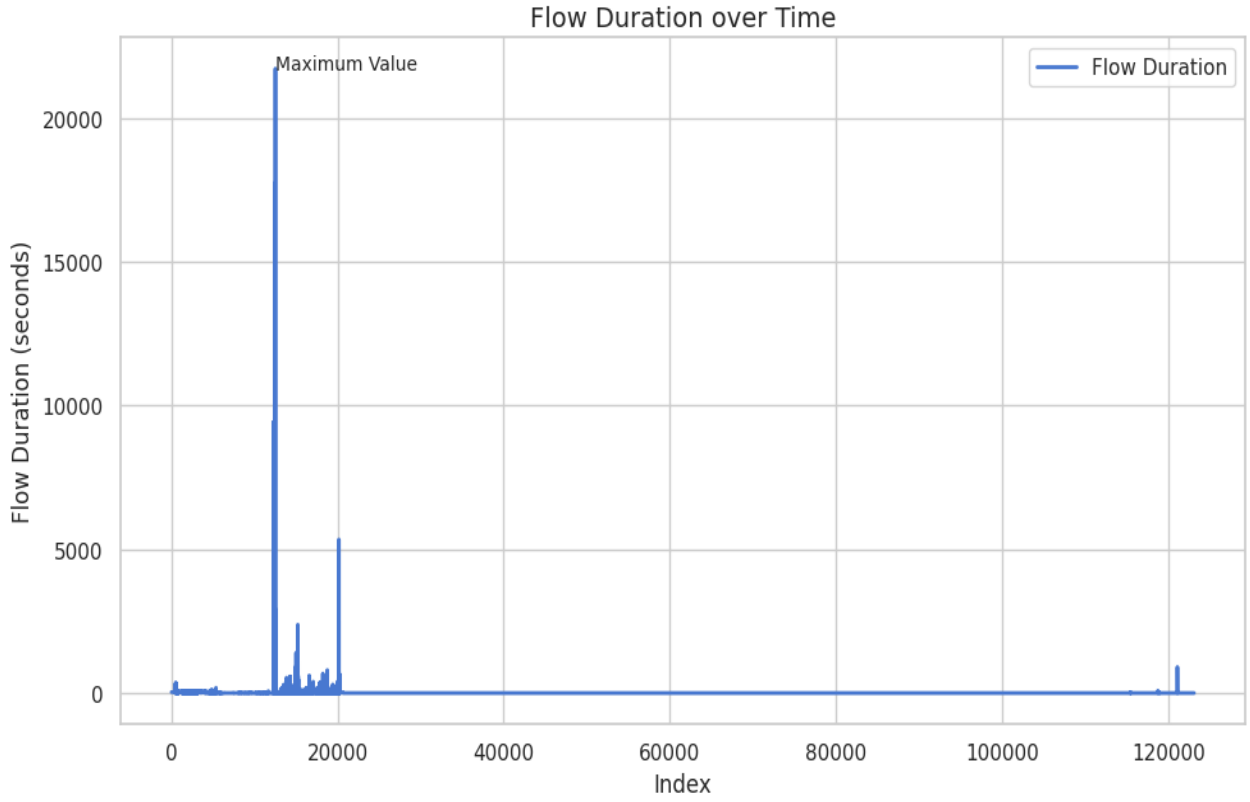

Figure 4.
Variation of flow duration over time.

The code in Algorithm 4 creates a bar chart to display the distribution of attack types in the data using the Seaborn library. It displays the number of each attack category and captions the axes. A visualization of the results is shown in Figure 5.

Algorithm 4. Code for building a bar chart of the distribution of attack type

```
import matplotlib.pyplot as plt
import seaborn as sns

attack_counts = data['Attack_type'].value_counts()

plt.figure(figsize=(10, 6))
sns.barplot(x=attack_counts.index, y=attack_counts.values)
plt.xticks(rotation=90)
plt.title('Distribution of Attack Types')
plt.xlabel('Attack Type')
plt.ylabel('Frequency')
plt.show()
```

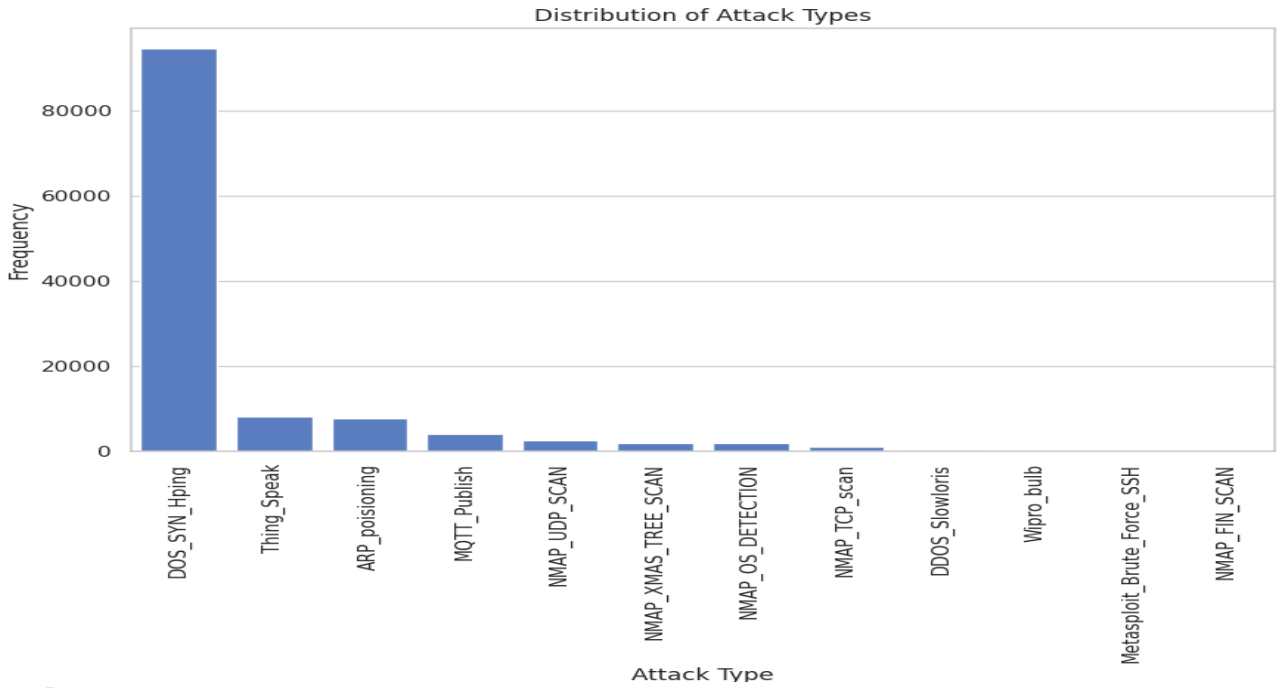

Figure 5.
Distribution of attack types.

The code in Algorithm 5 constructs a histogram with a density line for the distribution of DOS_SYN_Hping attack durations using the Seaborn library. The histogram shows the frequency of attack duration, and the density line (KDE) shows the distribution. A visualization of the results is shown in Figure 6.

Algorithm 5. Code to draw a histogram with a density line for the DOS_SYN_Hping attack duration distribution.

```
import seaborn as sns
import matplotlib.pyplot as plt

# Example for the DOS_SYN_Hping attack
plt.figure(figsize=(10, 6))

# Set plot style
sns.set(style="whitegrid")

# Plot histogram with KDE
sns.histplot(data[data['Attack_type'] == 'DOS_SYN_Hping']['flow_duration'], kde=True,
             color='mediumblue', bins=30)

# Add a KDE line with a different color
sns.kdeplot(data[data['Attack_type'] == 'DOS_SYN_Hping']['flow_duration'],
            color='darkorange', lw=2)

# Title
plt.title('Distribution of DOS_SYN_Hping Attack Duration', fontsize=16, fontweight='bold', style='italic')

# Axis labels
plt.xlabel('Attack Duration (Time)', fontsize=14, fontweight='bold')
plt.ylabel('Frequency', fontsize=14, fontweight='bold')

# Improve axis and grid appearance
plt.xticks(fontsize=12)
plt.yticks(fontsize=12)
plt.grid(True, linestyle='--', alpha=0.7)

# Adjust grid positioning for better visualization
plt.tight_layout()

# Show plot
plt.show()
```

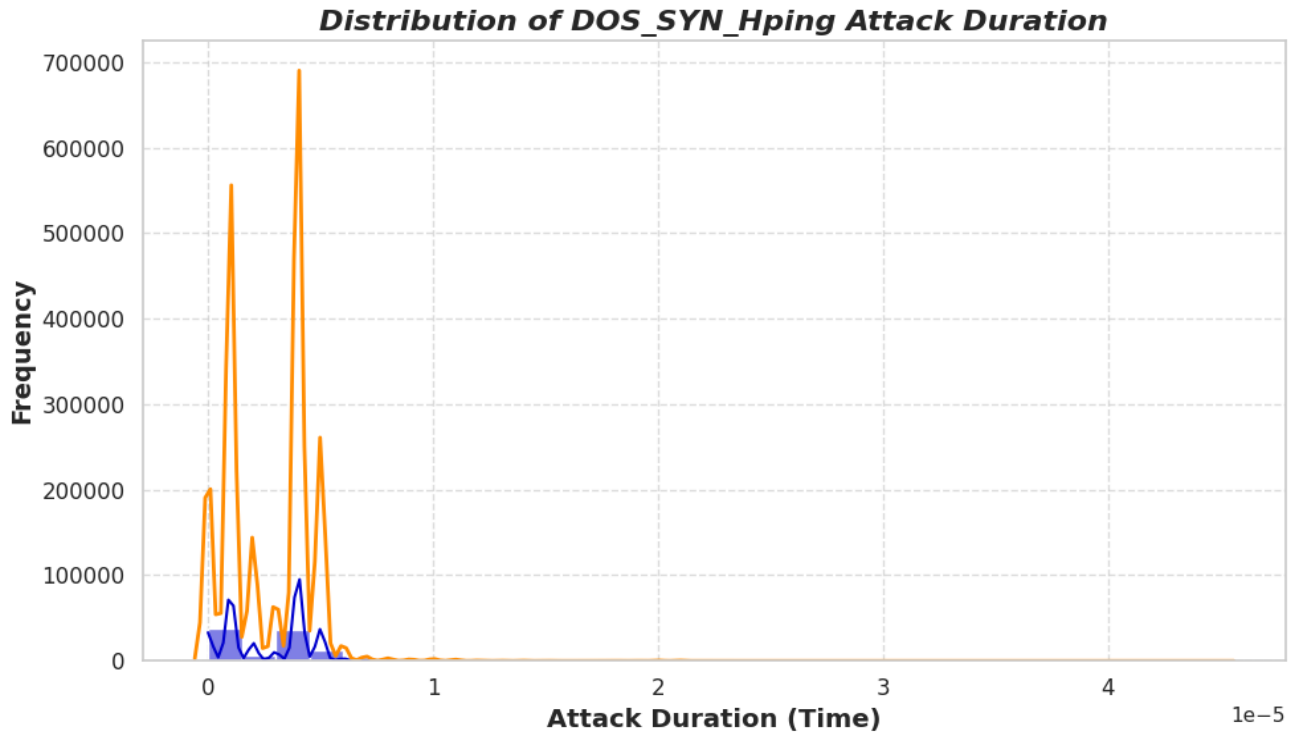

Figure 6.
DOS_SYN_Hping attack duration distribution.

The code shown in Algorithm 6 generates a classification report for different attack types using the sklearn library and builds a bar chart showing the prediction accuracy for each attack type. A visualization of the results is shown in Figure 7.

Algorithm 6. Code to generate a classification report and plot the prediction accuracy for different types of attacks

```
from sklearn.metrics import classification_report

# Using y_test and y_pred for reporting
y_true = y_test  # True labels
y_pred = rf.predict(X_test)  # Predicted labels

# Generating the classification report
report = classification_report(y_true, y_pred, output_dict=True)

# Extracting attack labels
attack_labels = list(report.keys())[:-3]  # Excluding labels for accuracy and macro/weighted avg

# Precision values for predictions
precision_values = [report[label]['precision'] for label in attack_labels]

# Plotting precision
plt.figure(figsize=(12, 6))
sns.barplot(x=attack_labels, y=precision_values)
plt.xticks(rotation=90)
plt.title('Prediction Precision for Different Attack Types')
plt.xlabel('Attack Type')
plt.ylabel('Precision')
plt.show()
```

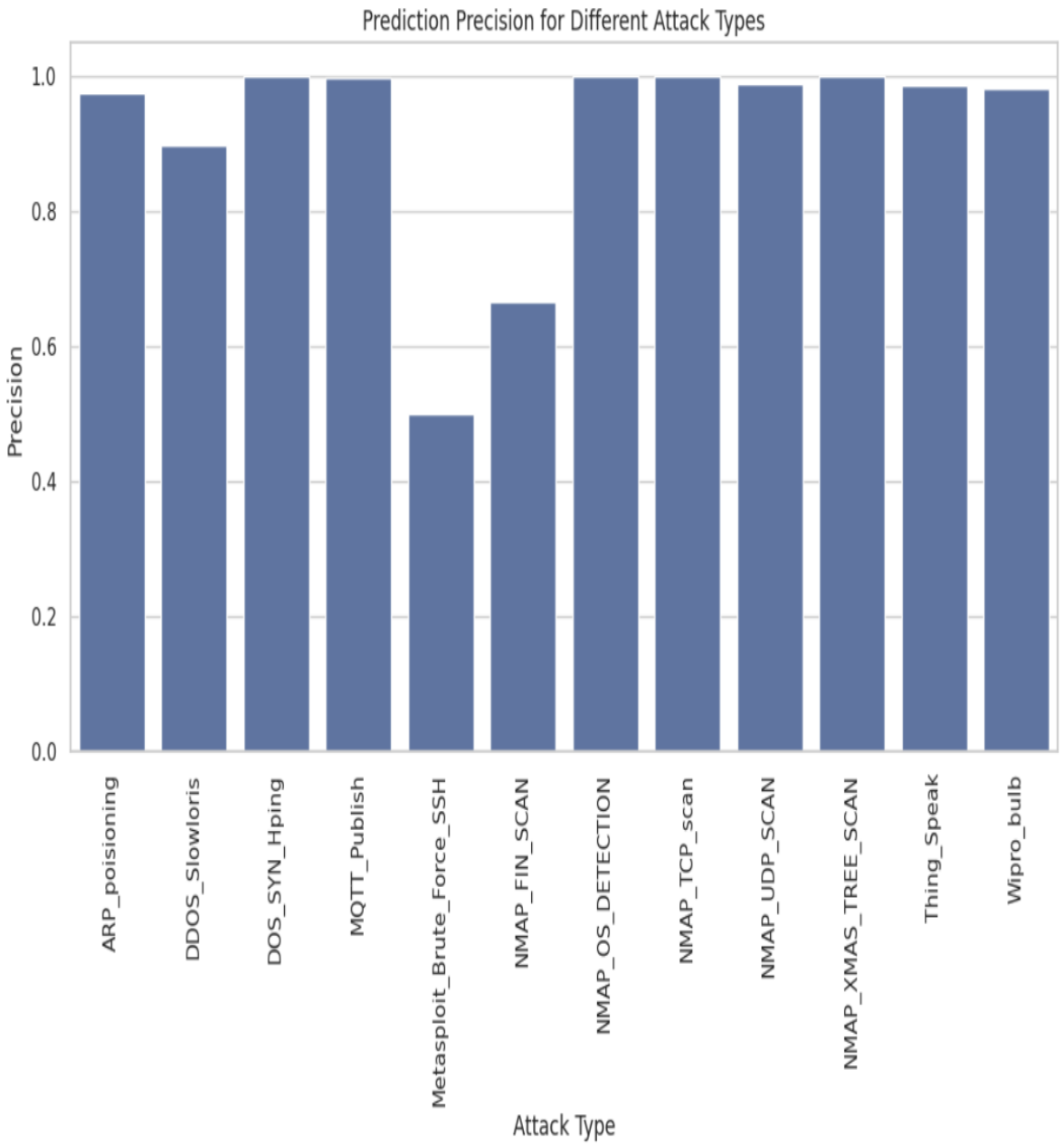

Figure 7.
Prediction accuracy for different types of attacks.

The cross-validation results show that Logistic Regression, Random Forest, and Nearest Neighbor Method (KNN) models showed high classification accuracy. The best average accuracy was achieved by the Random Forest model with 99.82%, followed by KNN with 99.59%, while Logistic Regression showed an accuracy of 99.10%. These data confirm the effectiveness of the models in classifying attacks, with Random Forest having the highest stability. The code in Algorithm 7 is responsible for performing cross-validation and calculating the average accuracy value. Figure 8 shows a plot of the average accuracy of the cross-validation-based models, the Logistic Regression model. Figure 9 shows the classification report of the Random Forest model, and Figure 10 shows the comparison report of K-Nearest Neighbors.

Algorithm 7. Performing cross-validation for performance evaluation.

```
from sklearn.model_selection import cross_val_score
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import classification_report

# Assuming you already have the X_train, X_test, y_train, and y_test variables

# Define the models
models = {
    'Logistic Regression': LogisticRegression(max_iter=10000, solver='lbfgs'),
    'Random Forest': RandomForestClassifier(n_estimators=50, n_jobs=-1),  # Reduced number of trees, enabled parallelism
    'K-Nearest Neighbors': KNeighborsClassifier(n_neighbors=5)  # Optimized neighbors for faster execution
}

# Use 3-fold cross-validation instead of 5 for faster results
cv_folds = 3

# Perform cross-validation and print the results for each model
for model_name, model in models.items():
    print(f"Model: {model_name}")

    # Cross-validation scores (with parallel processing for Random Forest)
    cv_scores = cross_val_score(model, X_train, y_train, cv=cv_folds, scoring='accuracy', n_jobs=-1)
    print(f"Cross-validation scores: {cv_scores}")
    print(f"Mean accuracy: {cv_scores.mean()}\n")

    # Fit the model and evaluate using classification report
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    print(f"Classification Report for {model_name}:\n")
    print(classification_report(y_test, y_pred))
    print('-' * 50)
```

```
Model: Logistic Regression
Cross-validation scores: [0.99129739 0.99084485 0.99091447]
Mean accuracy: 0.9910189020781844

Classification Report for Logistic Regression:

              precision    recall  f1-score   support

           0       0.94      0.94      0.94      2306
           1       0.97      0.76      0.85       154
           2       1.00      1.00      1.00     28409
           3       1.00      1.00      1.00      1273
           4       0.86      0.55      0.67        11
           5       1.00      0.86      0.92         7
           6       0.99      1.00      1.00       622
           7       0.99      0.99      0.99       319
           8       0.95      0.97      0.96       750
           9       1.00      0.99      1.00       582
          10       0.94      0.95      0.95      2424
          11       0.97      0.75      0.84        79

    accuracy                           0.99     36936
   macro avg       0.97      0.90      0.93     36936
weighted avg       0.99      0.99      0.99     36936
```

Figure 8.
Classification report of the Logistic Regression model.

```
Model: Random Forest
Cross-validation scores: [0.99818986 0.99812024 0.99829429]
Mean accuracy: 0.9982014597184995

Classification Report for Random Forest:

              precision    recall  f1-score   support

           0       0.99      1.00      0.99      2306
           1       0.99      0.99      0.99       154
           2       1.00      1.00      1.00     28409
           3       1.00      1.00      1.00      1273
           4       1.00      0.73      0.84        11
           5       1.00      0.86      0.92         7
           6       1.00      1.00      1.00       622
           7       1.00      1.00      1.00       319
           8       0.99      0.99      0.99       750
           9       1.00      0.99      1.00       582
          10       1.00      0.99      0.99      2424
          11       0.99      0.94      0.96        79

    accuracy                           1.00     36936
   macro avg       1.00      0.96      0.97     36936
weighted avg       1.00      1.00      1.00     36936
```

Figure 9.
Classification report of the Random Forest model.

```
Model: K-Nearest Neighbors
Cross-validation scores: [0.9963449  0.99543983 0.99592718]
Mean accuracy: 0.9959039695524536

Classification Report for K-Nearest Neighbors:

              precision    recall  f1-score   support

           0       0.97      0.97      0.97      2306
           1       0.96      0.97      0.96       154
           2       1.00      1.00      1.00     28409
           3       1.00      1.00      1.00      1273
           4       1.00      0.55      0.71        11
           5       0.86      0.86      0.86         7
           6       0.99      1.00      0.99       622
           7       1.00      0.99      1.00       319
           8       1.00      0.98      0.99       750
           9       1.00      0.99      1.00       582
          10       0.97      0.98      0.98      2424
          11       0.94      0.77      0.85        79

    accuracy                           1.00     36936
   macro avg       0.97      0.92      0.94     36936
weighted avg       1.00      1.00      1.00     36936
```

Figure 10.
Classification report of the K-Nearest Neighbors model.

4. Conclusion

As a result of the analysis of machine learning models for classifying attacks in Industrial Internet of Things (IIoT) infrastructure, several key conclusions can be drawn. The evaluation of models with and without cross-validation showed that the random forest is the most effective model, achieving a high classification accuracy of 99.82% with cross-validation and 99.65% without it. The nearest neighbors (KNN) method also performed well but was inferior to random forest, with accuracies of 99.59% and 99.13%, respectively, without cross-validation. On the other hand, logistic regression, despite some improvement with cross-validation (accuracy of 99.1%), remains the least efficient model, especially when dealing with unbalanced data, where its accuracy was only 77.07% without cross-validation. Thus, random forest showed the best results in terms of accuracy and stability, both with and without cross-validation, making it the preferred choice for classifying attacks in IIoT. The introduction of cross-validation significantly improves the results of all models, confirming the importance of this approach to enhance the generalizability and reliability of predictions in such tasks.

limitations when working with highly imbalanced data.

Based on the results obtained, my future research will focus on the following areas:

1. Utilizing more complex deep learning models, such as Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN), to improve accuracy and the ability to detect hidden and complex patterns in the data, which will enhance the detection of new attack types.

2. Exploring multi-task and multi-class classification approaches, which will allow for the analysis of not only the attack type but also its intensity, creating more flexible and precise protection systems.

Therefore, my further research will be directed towards developing more advanced and effective methods for protecting IIoT infrastructure using state-of-the-art machine learning and deep learning technologies. This will significantly improve the accuracy and adaptability of security systems, as well as enhance their resilience to new threats and attacks.

References

[1] H. Yao, P. Gao, P. Zhang, J. Wang, C. Jiang, and L. Lu, "Hybrid intrusion detection system for edge-based IIoT relying on machine-learning-aided detection," *IEEE Network*, vol. 33, no. 5, pp. 75-81, 2019. https://doi.org/10.1109/MNET.2019.1900172

[2] I. Bibi, A. Akhunzada, and N. Kumar, "Deep AI-powered cyber threat analysis in IIoT," *IEEE Internet of Things Journal*, vol. 10, no. 9, pp. 7749-7760, 2022. https://doi.org/10.1109/JIOT.2023.3205689

[3] X. Li, C. Xie, Z. Zhao, C. Wang, and H. Yu, "Anomaly detection algorithm of industrial Internet of Things data platform based on deep learning," *IEEE Transactions on Green Communications and Networking*, vol. 20, no. 4, pp. 2473-2400, 2024. https://doi.org/10.1109/TII.2024.1234567

[4] M. Aloqaily, I. Al Ridhawi, and S. Kanhere, "Reinforcing industry 4.0 with digital twins and blockchain-assisted federated learning," *IEEE Journal on Selected Areas in Communications*, vol. 41, no. 11, pp. 3504-3516, 2023. https://doi.org/10.1109/JSAC.2023.3205678

[5] A. Hussain, E. M. Tordera, X. Masip-Bruin, and H. C. Leligou, "Rule-based with Machine learning IDS for DDoS attack detection in cyber-physical production systems (CPPS)," *IEEE Access*, vol. 12, p. 3445261, 2024. https://doi.org/10.1109/ACCESS.2024.3445261

[6] A. Zainudin, L. A. C. Ahakonye, R. Akter, D.-S. Kim, and J.-M. Lee, "An efficient hybrid-dnn for ddos detection and classification in software-defined iiot networks," *IEEE Internet of Things Journal*, vol. 10, no. 10, pp. 8491-8504, 2022. https://doi.org/10.1109/JIOT.2023.3223456

[7] D. Velásquez *et al.*, "A hybrid machine-learning ensemble for anomaly detection in real-time industry 4.0 systems," *IEEE Access*, vol. 10, pp. 72024-72036, 2022. https://doi.org/10.1109/ACCESS.2022.3188102

[8] B. Bajic, A. Rikalovic, N. Suzic, and V. Piuri, "Toward a human-cyber-physical system for real-time anomaly detection," *IEEE Systems Journal*, vol. 18, no. 2, pp. 2204-2213. , 2024. https://doi.org/10.1109/JSYST.2024.3184236

[9] T. Zhang *et al.*, "When moving target defense meets attack prediction in digital twins: A convolutional and hierarchical reinforcement learning approach," *IEEE Journal on Selected Areas in Communications*, vol. 41, no. 10, pp. 3293-3305, 2023. https://doi.org/10.1109/JSAC.2023.3182751

[10] I. D. Mienye and N. Jere, "Deep learning for credit card fraud detection: A review of algorithms, challenges, and solutions," *IEEE Access*, vol. 12, pp. 3426955-3426975, 2024. https://doi.org/10.1109/ACCESS.2024.3426955

[11] M. Groshev, C. Guimaraes, J. Martín-Pérez, and A. de la Oliva, "Toward intelligent cyber-physical systems: Digital twin meets artificial intelligence," *IEEE Communications Magazine*, vol. 59, no. 8, pp. 14-20, 2021. https://doi.org/10.1109/MCOM.2021.XXXXXXX

[12] R. Rayhana, L. Bai, G. Xiao, M. Liao, and Z. Liu, "Digital twin models: Functions, challenges, and industry applications," *IEEE Journal of Radio Frequency Identification*, vol. 8, pp. 282-321, 2024.

[13] K. Ramesh, G. Sasirekha, M. Rao, J. Bapat, and D. Das, "Digital twin-based what-if simulation of security attacks in smart irrigation systems," in *2024 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT)*, 2024: IEEE, pp. 1-6.

[14] B. Qin *et al.*, "Machine and deep learning for digital twin networks: A survey," *IEEE Internet of Things Journal*, vol. 11, no. 21, pp. 1234-1245, 2024. https://doi.org/10.1109/JIOT.2024.XXXXXX

[15] S. El-Gendy and M. A. Azer, "Security framework for internet of things (IoT)," in *2020 15th International Conference on Computer Engineering and Systems (ICCES)*, 2020: IEEE, pp. 1-6.

[16] Y. I. L. Lucio, K. M. Villalba, and S. A. Donado, "Adaptive blockchain technology for a cybersecurity framework in IIoT," *IEEE Revista Iberoamericana de Tecnologias del Aprendizaje*, vol. 17, no. 2, pp. 178-184, 2022. https://doi.org/10.1109/RITA.2022.3071983

[17] C. Maraveas, M. Rajarajan, K. D. Arvanitis, and A. Vatsanidou, "Cybersecurity threats and mitigation measures in agriculture 4.0 and 5.0," *Smart Agricultural Technology*, vol. 9, p. 100616, 2024. https://doi.org/10.1016/j.atech.2024.100616

[18] F. K. Alarfaj, I. Malik, H. U. Khan, N. Almusallam, M. Ramzan, and M. Ahmed, "Credit card fraud detection using state-of-the-art machine learning and deep learning algorithms," *Ieee Access*, vol. 10, pp. 39700-39715, 2022. https://doi.org/10.1109/ACCESS.2022.3166891

[19] R. K. Somkunwar, A. Pimpalkar, K. M. Katakdound, A. S. Bhide, S. P. Chinchalkar, and Y. M. Patil, "A fraud detection system in financial networks using antibenford subgraphs and machine learning algorithms," in *2023 International Conference on Ambient Intelligence, Knowledge Informatics and Industrial Electronics (AIKIIE)*, 2023: IEEE, pp. 1-6.

[20] P. Monamo, V. Marivate, and B. Twala, "Unsupervised learning for robust Bitcoin fraud detection," in *2016 Information Security for South Africa (ISSA)*, 2016: IEEE, pp. 129-134.

[21] J. Chatterjee, M. Damle, and A. Aslekar, "Digital trust in industry 4.0 & 5.0: Impact of frauds," in *2023 7th International Conference on Intelligent Computing and Control Systems (ICICCS)*, 2023: IEEE, pp. 922-928.

[22] H. Gu *et al.*, "DIAVA: A traffic-based framework for detection of SQL injection attacks and vulnerability analysis of leaked data," *IEEE Transactions on Reliability*, vol. 69, no. 1, pp. 188-202, 2019. https://doi.org/10.1109/TR.2020.2977679

[23] I. D. Mienye and Y. Sun, "A deep learning ensemble with data resampling for credit card fraud detection," *Ieee Access*, vol. 11, pp. 30628-30638, 2023. https://doi.org/10.1109/ACCESS.2023.3262020

[24] A. Yeboah-Ofori *et al.*, "Cyber threat predictive analytics for improving cyber supply chain security," *IEEE Access*, vol. 9, pp. 94318-94337, 2021. https://doi.org/10.1109/ACCESS.2021.3087109

[25] A. A. Alashhab *et al.*, "Enhancing DDoS attack detection and mitigation in SDN using an ensemble online machine learning model," *IEEE Access*, vol. 12, pp. 51630-51649, 2024. https://doi.org/10.1109/ACCESS.2024.3384398