CHAPTER 4

INITIAL RESULTS

4.1 Introduction

This chapter explores the analysis and prediction of traffic anomaly detection in IoT devices. It begins with defining the dataset, processing it, and constructing machine learning models. The final results are derived from experiments conducted using these models. The machine learning techniques employed include Causal Forest + SHAP, Isolation Forest + Autoencoder, and Lightweight GNN + Autoencoder. The findings on traffic anomaly detection in IoT devices using these techniques will be presented in this chapter. Specific results and analyses will be detailed in the following sections.

4.2 Exploratory Data Analysis (EDA)

Exploratory data analysis (EDA) is a crucial step before in-depth analysis or modeling, aiding in data preprocessing and hypothesis formulation. It primarily uses visualization and statistical methods to understand the core characteristics of the dataset, identify patterns, and uncover potential relationships. In this study, EDA can better analyze the collected UNSW BoT-IoT dataset, facilitating subsequent preprocessing and other research tasks. It also provides an initial analysis of the data, yielding preliminary and concise information about the data.

4.2.1 Data Collection

(UNSW) BoT-IoT data set is collected by University of New South Wales through laboratory experiments. This data set is one of the widely used data sets, which contains more than 72 million traffic records. Focus on botnets in the smart home scenario, so this data set is selected as the experimental object.

4.2.2 exploratory Data Analysis

Through visualization and statistical methods, the dataset is initially observed and analyzed to better understand its characteristics, structure, and potential issues. The core objective is to lay a solid foundation for subsequent data modeling, analysis, or mining. EDA helps gain a deeper understanding of data features, providing a basis for subsequent preprocessing and modeling.

```
print("Shape of the data set: ", df.shape)
print("List information and data types: ")
print(df.dtypes)
print("Number of duplicate samples: ", df.duplicated().sum())
```

```
Shape of the data set:  (2934817, 19)
List information and data types:
pkSeqID                int64
proto                 object
saddr                 object
sport                 object
daddr                 object
dport                 object
seq                    int64
stddev               float64
N_IN_Conn_P_SrcIP      int64
min                  float64
```

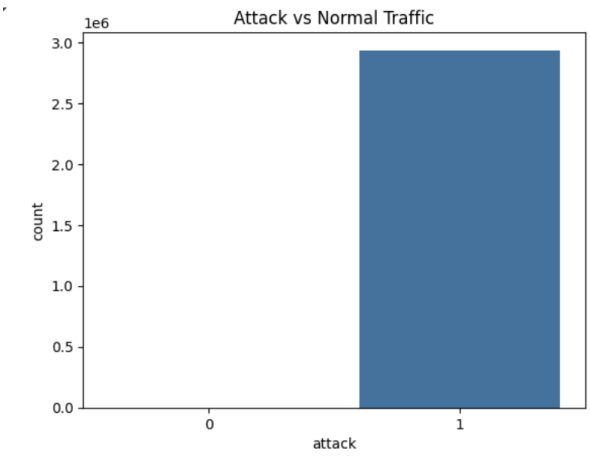
F4.1

According to the results (F4.1) of the code running, the data set contains 2,934,817 records and 19 features, which is a large scale; the data set contains a variety of data types: integer (int64), string (object), floating point (float64); therefore, it is necessary to convert the data types to improve the efficiency of model processing;

```
print("类别分布比例：")
print(df['attack'].value_counts(normalize=True))
```

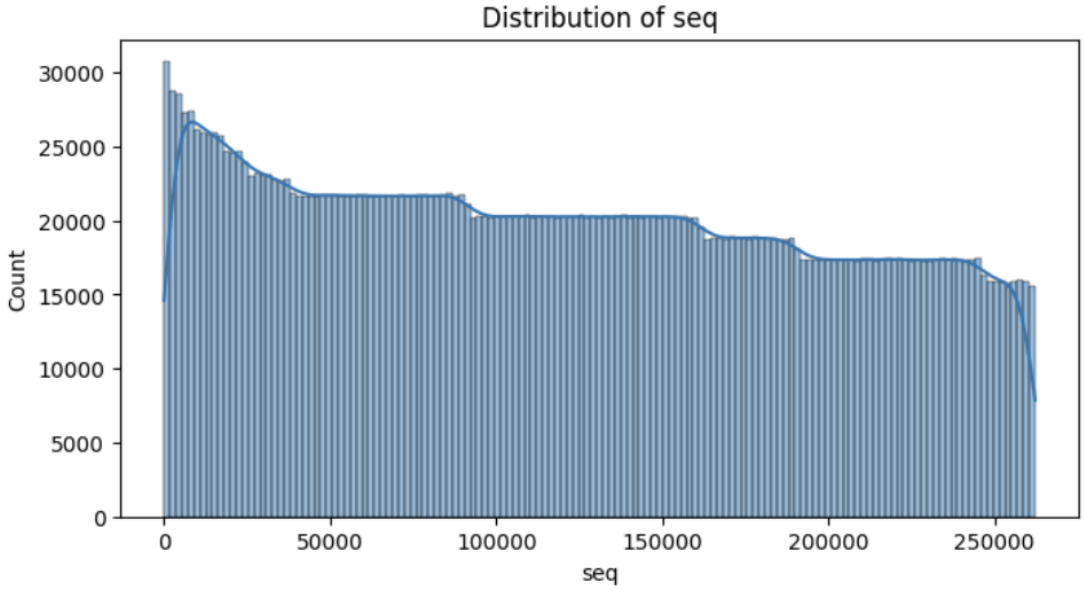
```
类别分布比例：
attack
1    0.999874
0    0.000126
Name: proportion, dtype: float64
```

F4.2



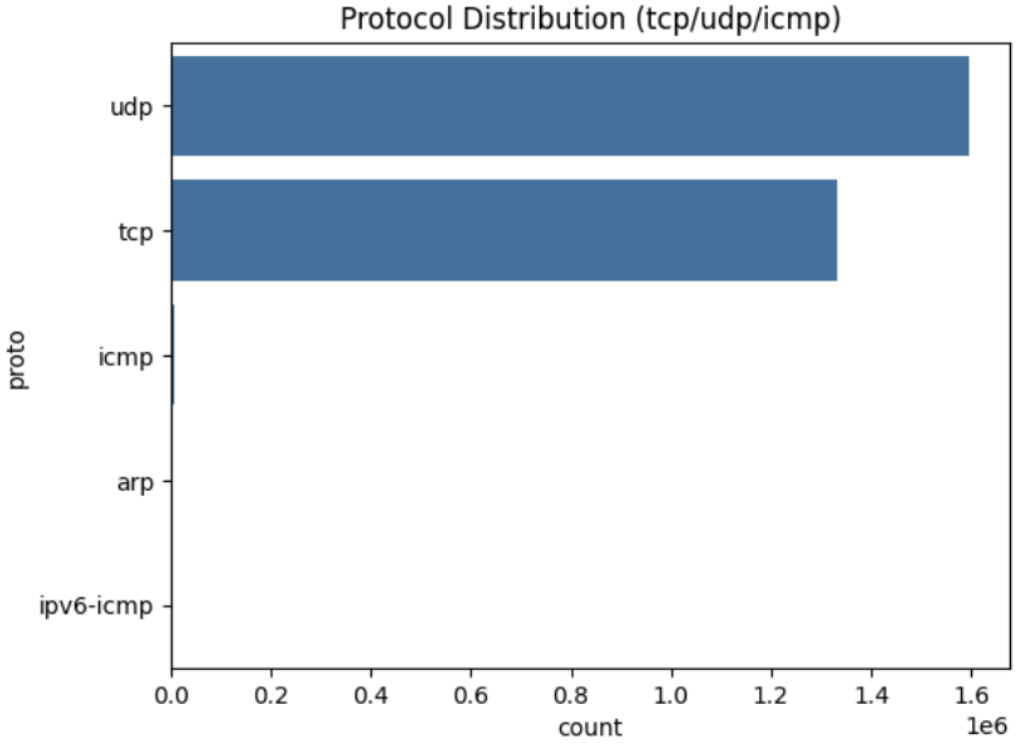
F4.3

As can be seen from the figure(F4.2;F4.3), the number of attacks in this data set is high, with 0.999874 attacks and 0.000126 non-attacks; such extremely unbalanced category distribution may lead to bias in model training, which needs to be processed in the future;



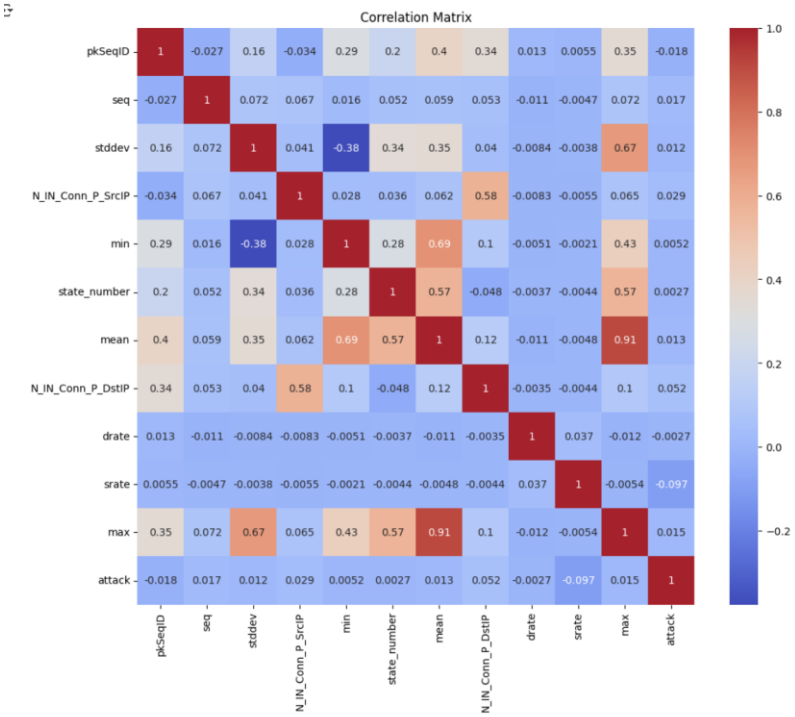
F4.4

Take a visualization table (F4.4) in the visualization of eigenvalues for analysis, which shows the distribution of the seq field. There are more samples with low order number and fewer samples with high order number, showing an overall downward trend. The distribution of samples with low order number and high order number is uneven, and further processing is needed to deal with abnormal traffic or data collection deviation;



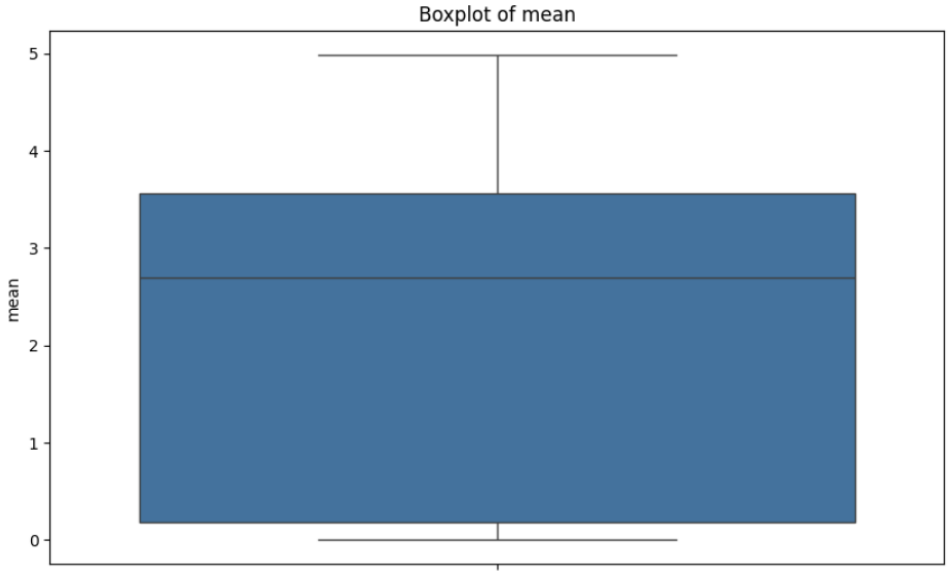
F4.5

The figure (F4.5) shows the distribution of different network protocols in the data set. The data set is mainly based on udp and tcp protocols, which occupy the vast majority of traffic. The data distribution is unbalanced. In the future, protocol-related modeling or analysis needs to be carried out, and attention should be paid to the imbalance of protocol distribution to avoid model bias towards mainstream protocols (such as UDP and TCP);



F4.6

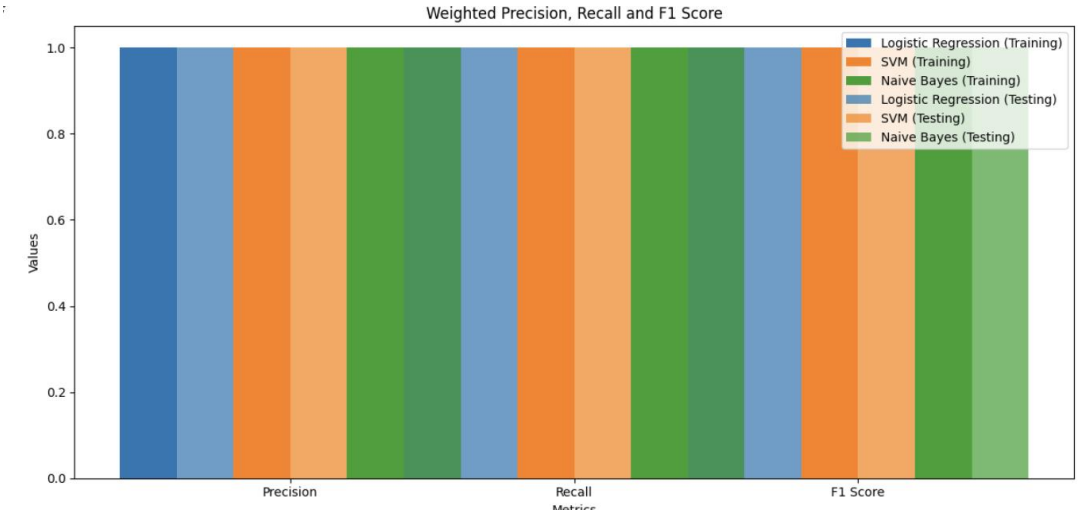
This chart (F4.6) illustrates the correlation matrix among various features within the dataset, with a particular focus on the relationship between the 'attack' label and other features. Strongly correlated features, such as mean, max, and N_IN_Conn_P_DstIP, are crucial for identifying attacks. Features with lower correlation to 'attack,' such as seq, stddev, and drate, contribute less to the model and can be considered for removal during the feature selection process to simplify the model and prevent overfitting.



F4.7

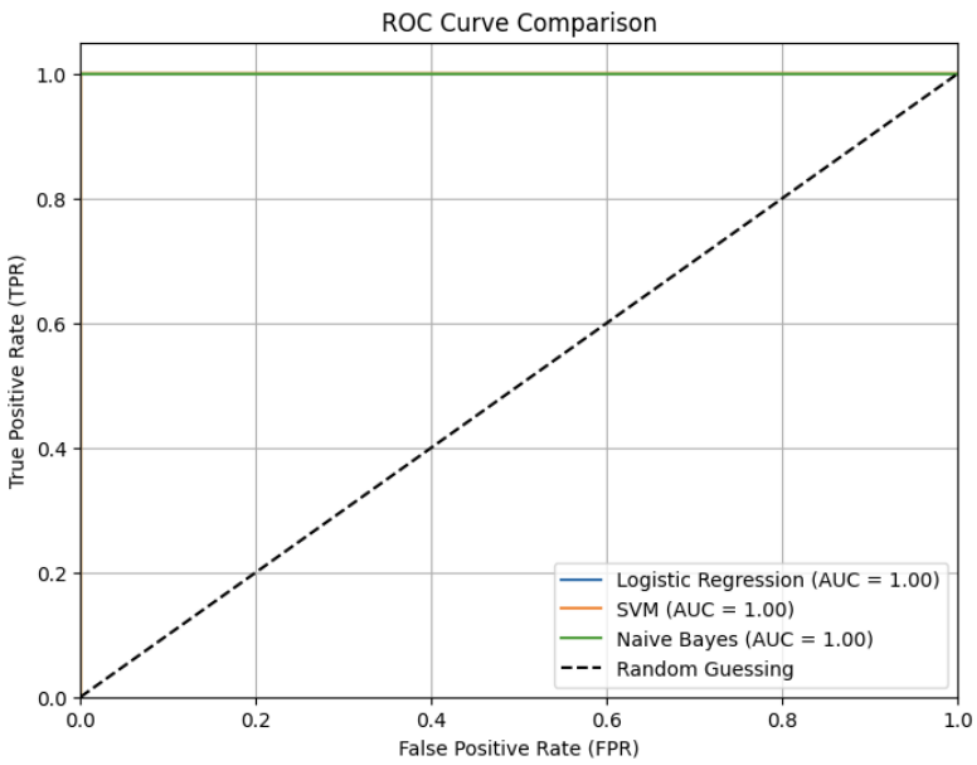
Take one of the box plots (F4.7) of eigenvalues for analysis. The chart is a boxplot (Boxplot), which is used to show the distribution of the characteristic mean. There are no obvious outliers in the figure, indicating that there are no extreme values in the data that deviate significantly from the overall distribution, so there is no need to process the value as an outlier.

4.3 Final Findings



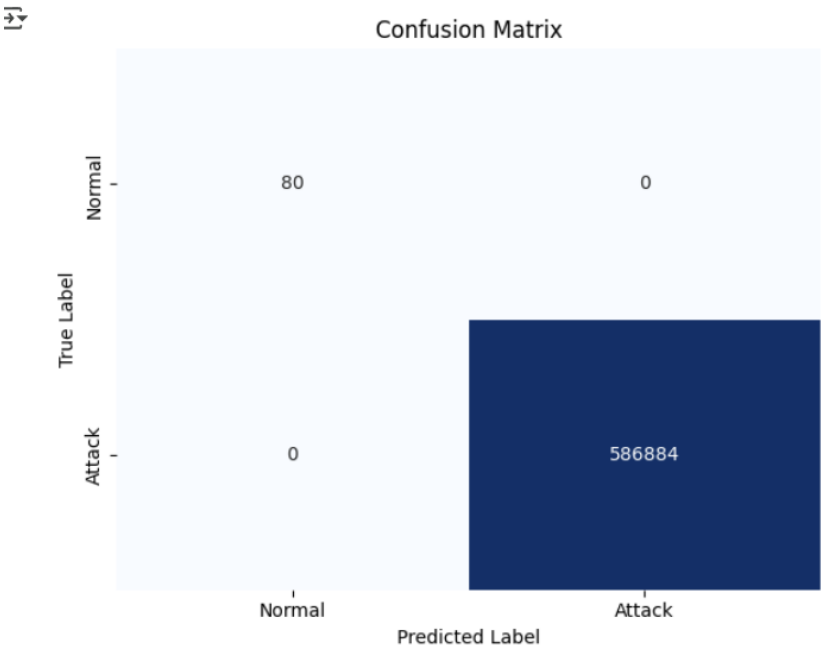
F4.8

The chart (F4.8) is a bar chart that compares the weighted precision, recall, and F1 score of three classification models: logistic regression, SVM, and Naive Bayes, across both the training set and the test set. These metrics are crucial for evaluating the performance of classification models, helping us gain a comprehensive understanding of their predictive capabilities.



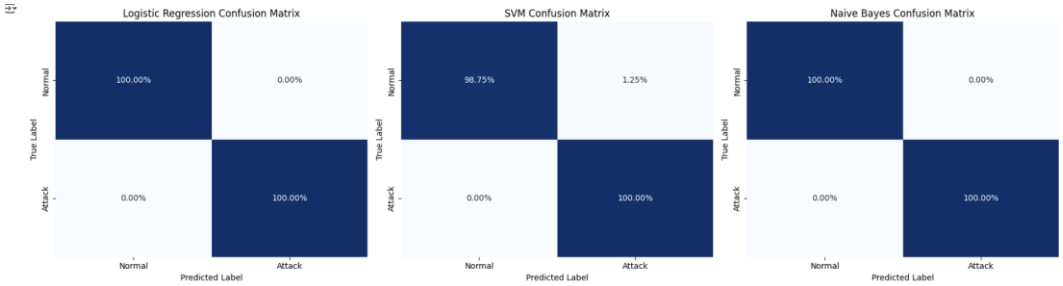
F4.9

The ROC curve (Receiver Operating Characteristic Curve) is illustrated in Figure F4.9. It is a commonly used tool for evaluating the performance of binary classification models by plotting the relationship between the True Positive Rate (TPR) and the False Positive Rate (FPR). All three models—logistic regression, SVM, and Naive Bayes—have ROC curves that are above the random guess line and completely overlap, indicating their perfect performance (AUC = 1.00). This is the basic concept of the ROC curve.



F4.10

This is a confusion matrix (F4.10). A confusion matrix is a common tool for evaluating the performance of classification models, illustrating the relationship between model predictions and actual labels. Through the confusion matrix, we can intuitively understand the model's classification accuracy and the types of errors. The chart shows that the model performs perfectly, correctly identifying all normal samples (TN = 80) and attack samples (TP = 586,884), with no false positives (FP = 0) or false negatives (FN = 0).



F4.11

The chart (F4.11) presents a confusion matrix, illustrating the performance of three classification models: logistic regression (Logistic Regression), support vector machine (SVM), and naive bayes (Naive Bayes). The confusion matrix evaluates how well the predictions of these models match the actual labels, helping us understand the model's accuracy and the types of errors. In this test, logistic regression and naive bayes performed exceptionally well, accurately distinguishing between' Normal 'and' Attack' samples without any misclassifications.

Initial Findings:ROC Curve Comparison（the relationship between the true example rate and the false positive example rate of the model is shown to help us understand the performance of the model at different thresholds）： Naive Bayes、 Logistic Regression
Confusion Matrix（the performance of the model in a binary classification task, that is, to evaluate the prediction accuracy of the model）： Naive Bayes