Harnessing Decimal Datasets for Enhanced Intrusion Detection System

Project Summary: In this project, I developed an Intrusion Detection System (IDS) using machine learning techniques and the **CICOV2024** dataset. The system is designed to classify network traffic into benign or malicious categories, such as Denial of Service (DoS) and Spoofing attacks. Additionally, it can perform more granular classifications within Spoofing attacks, identifying specific types such as RPM, Speed, and Steering Wheel manipulation.

I employed multiple machine learning algorithms, including Logistic Regression, Random Forest, Deep Neural Networks (DNN), and Decision Trees, to classify the network traffic. While Logistic Regression demonstrated average accuracy, the other models—Random Forest, DNN, and Decision Trees—achieved near-perfect results in classifying both normal and attack traffic. The system processes network packet data to predict whether the traffic is benign or malicious, and, in the case of Spoofing attacks, it can further identify the type of spoofing. This system enhances real-time threat detection and response, making it a valuable tool for improving cybersecurity.

Data set: The dataset used in this project is the **CICOV2024 dataset**, available on Kaggle, which contains network traffic data, including benign and attack scenarios like DoS and Spoofing. It provides features related to packet flows, such as packet length, protocol types, and flow duration, which are important for detecting malicious activity. The dataset was generated using a 2019 Ford vehicle, with attacks carried out safely.The decimal datasets include files representing various network behaviors, both normal and malicious:

Data Content and Types:

- **Benign Traffic Data**: Represents normal IoV operations with no malicious activity, serving as a baseline for typical traffic.
- **DoS Attack Data**: Simulates Denial of Service (DoS) attacks, where excessive traffic disrupts service.
- **Spoofing Attack Data**: Contains datasets for different spoofing attacks targeting specific vehicle controls:

  GAS Spoofing: Manipulates gas control systems.
  RPM Spoofing: Affects RPM controls.
  SPEED Spoofing: Targets vehicle speed controls.
  STEERING_WHEEL Spoofing: Mimics attacks on the steering system.

```
## info about the dataset
combined_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1408219 entries, 0 to 1408218
Data columns (total 12 columns):
 #   Column          Non-Null Count    Dtype
---  ------          --------------    -----
 0   ID              1408219 non-null  int64
 1   DATA_0          1408219 non-null  int64
 2   DATA_1          1408219 non-null  int64
 3   DATA_2          1408219 non-null  int64
 4   DATA_3          1408219 non-null  int64
 5   DATA_4          1408219 non-null  int64
 6   DATA_5          1408219 non-null  int64
 7   DATA_6          1408219 non-null  int64
 8   DATA_7          1408219 non-null  int64
 9   label           1408219 non-null  object
 10  category        1408219 non-null  object
 11  specific_class  1408219 non-null  object
dtypes: int64(9), object(3)
```

Data Description:

```
## Get summary statistics
combined_df.describe()
```

	ID	DATA_0	DATA_1	DATA_2	DATA_3	DATA_4	DATA_5	DATA_6	DATA_7
count	1.408219e+06	1.408219e+06	1.408219e+06	1.408219e+06	1.408219e+06	1.408219e+06	1.408219e+06	1.408219e+06	1.408219e+06
mean	5.372079e+02	7.108660e+01	6.998925e+01	5.501127e+01	5.745364e+01	4.528517e+01	5.388261e+01	7.174914e+01	6.027477e+01
std	3.224800e+02	8.897717e+01	9.558374e+01	7.276584e+01	9.032077e+01	6.445835e+01	9.433612e+01	1.016872e+02	9.996547e+01
min	6.500000e+01	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00
25%	3.570000e+02	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00
50%	5.160000e+02	1.600000e+01	1.200000e+01	1.300000e+01	0.000000e+00	6.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00
75%	5.780000e+02	1.270000e+02	1.280000e+02	1.250000e+02	9.200000e+01	8.600000e+01	6.300000e+01	1.380000e+02	8.000000e+01
max	1.438000e+03	2.550000e+02	2.550000e+02	2.550000e+02	2.550000e+02	2.550000e+02	2.550000e+02	2.550000e+02	2.550000e+02

Number of instances for each class present in the CICIoV2024 dataset

```
---------------Label-----------------
label
BENIGN     1223737
ATTACK      184482
Name: count, dtype: int64
---------------Category-----------------
category
BENIGN       1223737
SPOOFING      109819
DoS            74663
Name: count, dtype: int64
---------------Specific_Class-----------------
specific_class
BENIGN              1223737
DoS                   74663
RPM                   54900
SPEED                 24951
STEERING_WHEEL        19977
GAS                    9991
Name: count, dtype: int64
```

- **Independent Variables**: These features include various network characteristics such as packet length, protocol type, and flow duration, which are used to classify traffic.

- **Dependent Variable**: The target variable is the classification of network traffic as *Benign*, *DoS*, or *Spoofing*. For *Spoofing*, an inner classification is performed to distinguish between RPM, Speed, and Steering Wheel manipulations.

The dataset was split into 80% for training and 20% for testing to ensure proper evaluation of the model's performance using unseen data.

Feature engineering: In this project, feature engineering focused on preparing the dataset for machine learning by enhancing data quality and refining the input features.

- The initial step was checking for missing values, and since there were none in the CICOV2024 dataset, the data was complete.

```
## checking for missing values
print(combined_df.isnull().sum())
```

```
ID                0
DATA_0            0
DATA_1            0
DATA_2            0
DATA_3            0
DATA_4            0
DATA_5            0
DATA_6            0
DATA_7            0
label             0
category          0
specific_class    0
dtype: int64
```

- Label encoding was applied to transform categorical features into numerical values, making them suitable for machine learning models.
- Additionally, the data was standardized, ensuring that all features were on a comparable scale, which improved the performance and efficiency of the algorithms used

ML algorithm: In this project, I utilized four machine learning algorithms to build an Intrusion Detection System (IDS) aimed at classifying network traffic.

1. **Logistic Regression**: A straightforward algorithm used for binary classification, distinguishing benign traffic from attack traffic. Although it provided decent results, its performance was not as high as the other models.
2. **Random Forest**: An ensemble method that constructs multiple decision trees and combines their predictions. It achieved near-perfect accuracy and was effective in identifying attack traffic.
3. **Deep Neural Networks (DNN):** A deep learning model capable of recognizing intricate patterns in the data, delivering exceptional accuracy for classifying both benign and attack traffic types.

4. **Decision Tree**: A model that splits data based on feature values, which helps in classification. It provided good accuracy and interpretability, especially for distinguishing between different attack types.

These models were used to classify network traffic and detect malicious activities, contributing to improved cybersecurity measures.

Results and discussion: The models' performance was assessed based on metrics such as precision, recall, F1-score, and accuracy.

Binary Classification (Benign vs. Malicious) For binary classification, the models' performance metrics are as follows:

Model	Accuracy	Recall	Precision	F1-Score
Logistic Regression(LR)	0.72	0.79	0.64	0.63
Random Forest(RF)	1.00	1.00	1.00	1.00
Deep Neural Network(DNN)	1.00	1.00	1.00	1.00
Decision Tree(DT)	1.00	1.00	1.00	1.00

Analysis :

- **Logistic Regression (LR)** showed moderate performance, with an **accuracy** of 0.72 and decent **recall** (0.79) but lower **precision** and **F1-score** (0.64, 0.63).
- **Random Forest (RF)**, **DNN**, and **DT** performed excellently across all metrics, achieving perfect scores of **1.00** in **accuracy**, **recall**, **precision**, and **F1-score**. These models demonstrated strong generalization and were able to classify the binary labels (Benign vs. Malicious) with high precision.

Category Classification:

```
categories = ['BENIGN', 'DoS', 'SPOOFING']
metrics = ['precision', 'recall', 'f1-score']

# Logistic Regression (LR) scores
lr_scores = {
    'precision': [0.98, 0.30, 0.33],
    'recall': [0.70, 1.00, 0.86],
    'f1-score': [0.82, 0.47, 0.48]
}

# Random Forest (RF) scores
rf_scores = {
    'precision': [1.00, 1.00, 1.00],
    'recall': [1.00, 1.00, 1.00],
    'f1-score': [1.00, 1.00, 1.00]
}

# Deep Neural Network (DNN) scores
dnn_scores = {
    'precision': [1.00, 1.00, 1.00],
    'recall': [1.00, 1.00, 1.00],
    'f1-score': [1.00, 1.00, 1.00]
}

# Decision Tree (DT) scores based on your provided classification report
dt_scores = {
    'precision': [1.00, 1.00, 1.00],
    'recall': [1.00, 1.00, 1.00],
    'f1-score': [1.00, 1.00, 1.00]
}
```

In the category classification task, Random Forest (RF), Deep Neural Network (DNN), and Decision Tree (DT) demonstrated outstanding performance, achieving perfect scores (1.00) across all evaluation metrics (precision, recall, and F1-score) for each category. These models effectively identified both benign and malicious activities. On the other hand, Logistic Regression (LR) faced challenges, particularly with precision in the DoS and SPOOFING categories (0.30 and 0.33), but performed well in terms of recall for DoS. Overall, RF, DNN, and DT were the top performers for multi-class classification, while LR showed limitations in accurately classifying some attack types.

Multi-Class Classification (Benign, DoS, Gas, RPM, Speed, Steering Wheel)

```
# Logistic Regression (LR) scores
lr_scores = {
    'precision': [0.99, 0.38, 1.00, 0.12, 0.62, 0.94],
    'recall': [0.66, 1.00, 1.00, 0.73, 1.00, 1.00],
    'f1-score': [0.79, 0.55, 1.00, 0.21, 0.77, 0.97]
}

# Random Forest (RF) scores
rf_scores = {
    'precision': [1.00, 1.00, 1.00, 1.00, 0.83, 1.00],
    'recall': [1.00, 1.00, 1.00, 0.91, 1.00, 1.00],
    'f1-score': [1.00, 1.00, 1.00, 0.95, 0.91, 1.00]
}

# Deep Neural Network (DNN) scores
dnn_scores = {
    'precision': [1.00, 1.00, 1.00, 0.92, 1.00, 1.00],
    'recall': [1.00, 1.00, 1.00, 1.00, 0.80, 1.00],
    'f1-score': [1.00, 1.00, 1.00, 0.96, 0.89, 1.00]
}

# Decision Tree (DT) scores based on your provided classification report
dt_scores = {
    'precision': [1.00, 1.00, 1.00, 1.00, 0.83, 1.00],
    'recall': [1.00, 1.00, 1.00, 0.91, 1.00, 1.00],
    'f1-score': [1.00, 1.00, 1.00, 0.95, 0.91, 1.00]
}
```

Analysis

- Logistic Regression (LR) struggled with precision in the DoS and RPM categories (0.38, 0.12), but it demonstrated strong recall, particularly for the Steering Wheel category (1.00). However, its F1-score was lower in some categories, especially RPM (0.21), indicating difficulty in differentiating certain attack types.
- Random Forest (RF), Deep Neural Network (DNN), and Decision Tree (DT) performed exceptionally well across all categories, achieving perfect precision, recall, and F1-scores for most classes. These models excelled in handling the fine-grained classification of multiple attack types, such as Gas, RPM, and Steering Wheel spoofing.
- DNN experienced a slight decrease in recall for Speed spoofing (0.80), but overall, it exhibited strong classification performance, comparable to RF and DT, making these models more reliable in multi-class scenarios.

Conclusion: The models exhibited different levels of performance for binary and multi-class classification tasks. In binary classification, Random Forest, Deep Neural Network, and Decision Tree achieved perfect scores across all metrics, while Logistic Regression showed moderate results. For multi-class classification, Random Forest, DNN, and Decision Tree excelled in classifying multiple attack types, achieving high accuracy and consistency. Logistic Regression struggled, particularly with distinguishing certain attack categories. Overall, Random Forest, DNN, and Decision Tree emerged as the most reliable models for both tasks.