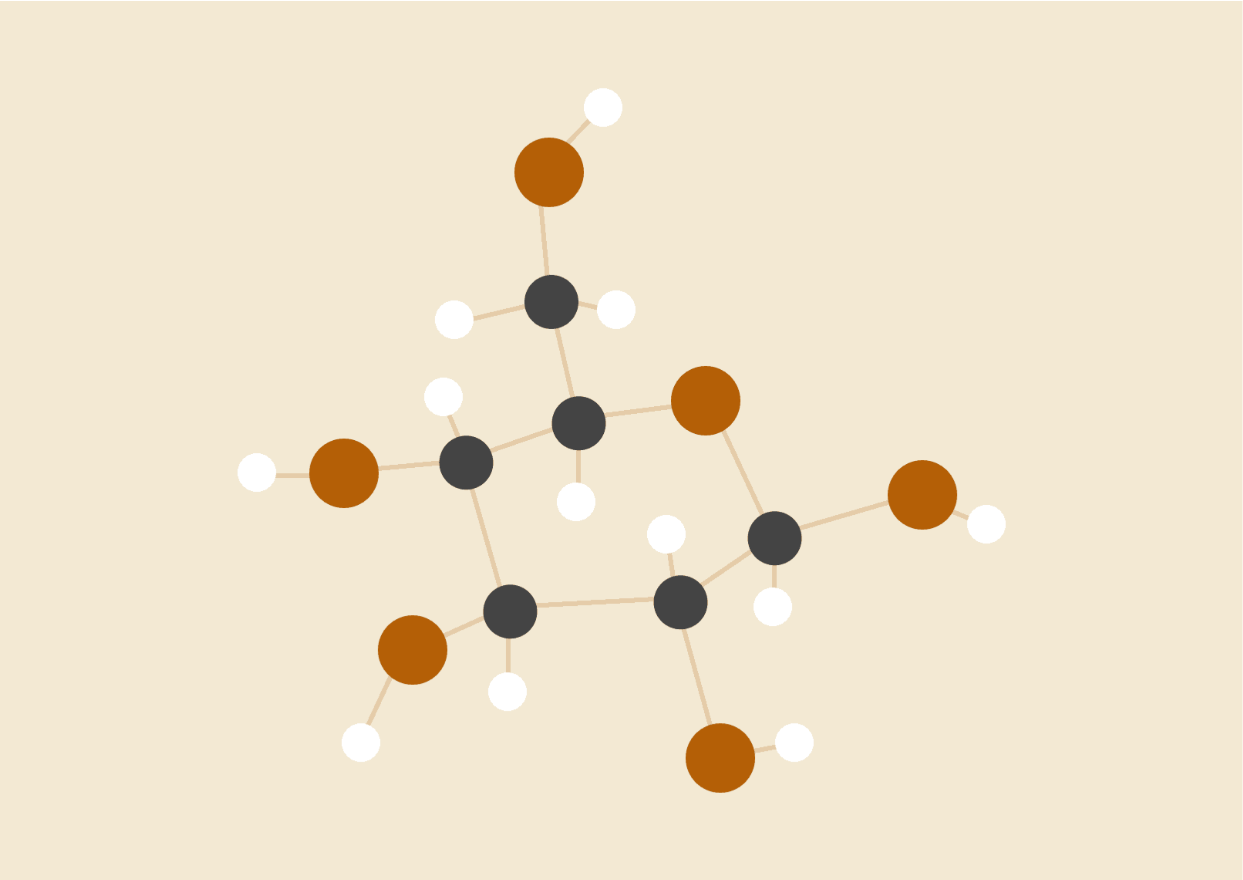
MACHINE LEARNING ASSIGNMENT

BY

Aryan Srivastava	(2020B4A41179G)
Raaghav Rajesh	(2020B4A81357G)
Tejas Khadke	(2020B4AA0758G)



Abstract

Intrusion Detection Systems (IDS) are security tools used to monitor and analyze network traffic or system activity for potential malicious activities or policy violations. IDS play a crucial role in identifying and responding to security incidents in real-time.

The constant growth of computer networks has raised serious concerns about vulnerability and security, necessitating the adoption of effective intrusion detection systems (IDS). However, commercial IDS in the market often struggle to identify novel attacks and generate false alarms for legitimate user activities. To address these issues and enhance accuracy, we propose a novel approach that integrates correlation-based feature selection (CFS) with a neural network for anomaly detection in IDS.

Our experimental analysis focuses on benchmark datasets of intrusion detection, namely NSL-KDD and UNSW-NB, which encompass current attack scenarios. The results demonstrate that our proposed model outperforms several state-of-the-art techniques in terms of accuracy, sensitivity, and specificity. By leveraging neural networks and correlation-based attribute selection, we achieve improved accuracy in identifying anomalies and distinguishing them from legitimate user activities.

Furthermore, our research highlights the potential of deploying such IDS models for securing Internet of Things (IoT) servers in the future. This would contribute to the establishment of robust security measures for wireless payment systems and enable secure integrated network management, leading to error-free operations and improved performance.

In conclusion, our study showcases the efficacy of integrating correlation-based feature selection with neural networks for enhancing anomaly detection in intrusion detection systems. The proposed model exhibits superior performance compared to existing techniques, offering promising prospects for securing computer networks and IoT infrastructures in the face of evolving threats.

Contents

Contribution

- The project combines correlation-based feature selection (CFS) with an artificial neural network (ANN) for improved performance in intrusion detection systems (IDS).
- The integration of CFS and ANN aims to enhance the efficiency and accuracy of IDS, particularly in detecting malicious software.
- Rule-based expert systems were integrated with the neural network to facilitate learning of normal system behavior and detect statistical variations indicating potential intrusions.
- The project utilized recent benchmark attack datasets, such as the UNSW-NB15 dataset, for rigorous testing and evaluation of the proposed approach.
- Correlation-based feature selection technique was applied to extract informative attributes and reduce the dimensionality of the dataset, resulting in improved IDS performance.
- Hyperparameter tuning was performed to optimize key parameters of the neural network, including learning rate, hidden layers, number of epochs, and batch size as ANN is a time consuming algorithm.
- The optimization process led to significant enhancements in the computational efficiency and overall performance of the IDS.
- The project's contribution lies in successfully integrating CFS and ANN, showcasing superior performance compared to traditional techniques in intrusion detection.
- Comprehensive experimental analysis and evaluation were conducted to validate the effectiveness and potential of the proposed approach.
- The project has implications for advancing the field of intrusion detection and offers a promising solution for enhancing the security of computer networks.

Data Collection and Preprocessing

Adding old and New Dataset:

New dataset : (125973, 43)
Old data : (494021, 42)
Dropping the difficulty column in the new dataset to maintain uniformity in the dataset.

The following are in the format attack type : attack category.

- 'normal': 'normal' - This represents normal network traffic without any malicious intent.
- 'back': 'dos' - 'back' attacks are classified as Denial of Service (DoS) attacks.
- 'buffer_overflow': 'u2r' - 'buffer_overflow' attacks are classified as User-to-Root (U2R) attacks.
- 'ftp_write': 'r2l' - 'ftp_write' attacks are classified as Remote-to-Local (R2L) attacks.
- 'guess_passwd': 'r2l' - 'guess_passwd' attacks are classified as R2L attacks.
- 'imap': 'r2l' - 'imap' attacks are classified as R2L attacks.
- 'ipsweep': 'probe' - 'ipsweep' attacks are classified as probe attacks.
- 'land': 'dos' - 'land' attacks are classified as DoS attacks.
- 'loadmodule': 'u2r' - 'loadmodule' attacks are classified as U2R attacks.
- 'multihop': 'r2l' - 'multihop' attacks are classified as R2L attacks.
- 'neptune': 'dos' - 'neptune' attacks are classified as DoS attacks.
- 'nmap': 'probe' - 'nmap' attacks are classified as probe attacks.
- 'perl': 'u2r' - 'perl' attacks are classified as U2R attacks.
- 'phf': 'r2l' - 'phf' attacks are classified as R2L attacks.
- 'pod': 'dos' - 'pod' attacks are classified as DoS attacks.
- 'portsweep': 'probe' - 'portsweep' attacks are classified as probe attacks.
- 'rootkit': 'u2r' - 'rootkit' attacks are classified as U2R attacks.
- 'satan': 'probe' - 'satan' attacks are classified as probe attacks.
- 'smurf': 'dos' - 'smurf' attacks are classified as DoS attacks.
- 'spy': 'r2l' - 'spy' attacks are classified as R2L attacks.
- 'teardrop': 'dos' - 'teardrop' attacks are classified as DoS attacks.
- 'warezclient': 'r2l' - 'warezclient' attacks are classified as R2L attacks.

- 'warezmaster': 'r2l' - 'warezmaster' attacks are classified as R2L attacks.

- 'mscan': 'probe' - 'mscan' attacks are classified as probe attacks.

- 'apache2': 'normal' - 'apache2' attacks are classified as normal traffic.

- 'processtable': 'dos' - 'processtable' attacks are classified as DoS attacks.

- 'snmpguess': 'probe' - 'snmpguess' attacks are classified as probe attacks.

- 'saint': 'probe' - 'saint' attacks are classified as probe attacks.

- 'mailbomb': 'dos' - 'mailbomb' attacks are classified as DoS attacks.

- 'snmpgetattack': 'probe' - 'snmpgetattack' attacks are classified as probe attacks.

- 'httptunnel': 'probe' - 'htt

Cleaning:

Cleaning the datasets' target attribute as it had "." after each value in the old dataset. Replace all values with required attribute values.

Feature selection:

We have employed the Correlation-Based Feature Selection (CFS) technique to perform feature selection. CFS is a widely adopted approach that assesses the individual predictive ability of each feature and considers the degree of redundancy among them.

By applying CFS, we aim to identify the most relevant features from the dataset. This technique quantifies the correlation between the class variable and subsets of features, taking into account both the average correlation between the class variable and the subset attributes and the average inter-correlation among the subset attributes.

Formula:

$$r = \frac{\sum\left(x_i - \bar{x}\right)\left(y_i - \bar{y}\right)}{\sqrt{\sum\left(x_i - \bar{x}\right)^2 \sum\left(y_i - \bar{y}\right)^2}}$$

- r = correlation coefficient
- x_i = values of the x-variable in a sample
- $\bar{x}$ = mean of the values of the x-variable
- y_i = values of the y-variable in a sample
- $\bar{y}$ = mean of the values of the y-variable

The threshold that we choose in our project is 0.9 . With this threshold nearly 9 features have been dropped.

Attributes removed: *Num_compromised, serror_rate, num_outbound_cmds, srv_serror_rate, rerror_rate, srv_rerror_rate, dst_host_serror_rate*

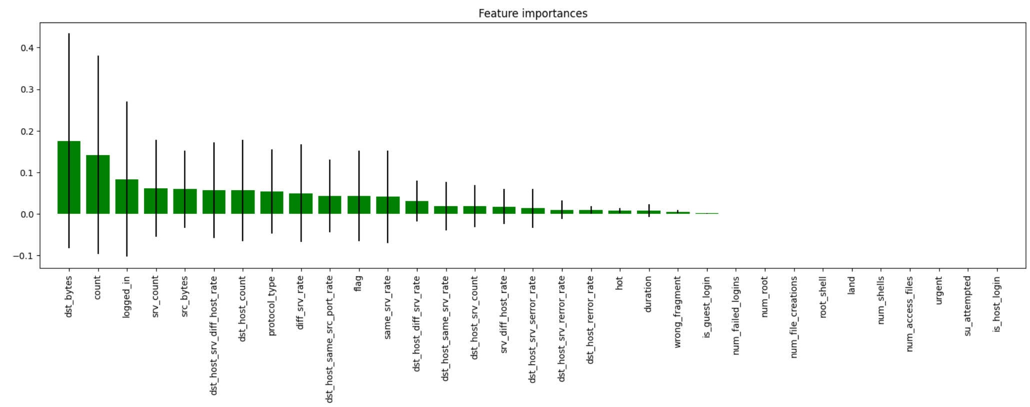
We have selected the features in the first column and the correlated features have been dropped.

Feature Importance:

It is a technique used to determine the relative importance or contribution of each feature (input variable) in a model.

- Feature importance helps in understanding which features have the most influence on the model's predictions.
- It allows us to identify the key factors that drive the target variable and gain insights into the underlying problem.
- Feature importance helps in feature selection by identifying the most relevant features and eliminating less important ones.
- It can be used for dimensionality reduction, improving model performance, and reducing computational resources.
- Feature importance aids in feature engineering by highlighting the most informative features for creating new variables.
- It provides interpretability by explaining why the model made certain predictions or decisions.
- Feature importance helps in identifying potential data quality issues or biases associated with specific features.
- It guides domain experts in focusing on the most important features when making decisions or taking actions based on the model's predictions.
- Feature importance can assist in comparing the importance of different features across multiple models and selecting the best model for deployment.
- It helps in communicating the relevance and impact of features to stakeholders and decision-makers.

Random Forest resulted in better performance and accuracy for device identification and up to 97 % accuracy for abnormal traffic identification. As random forest gives a good prediction we use it for feature importance analysis.

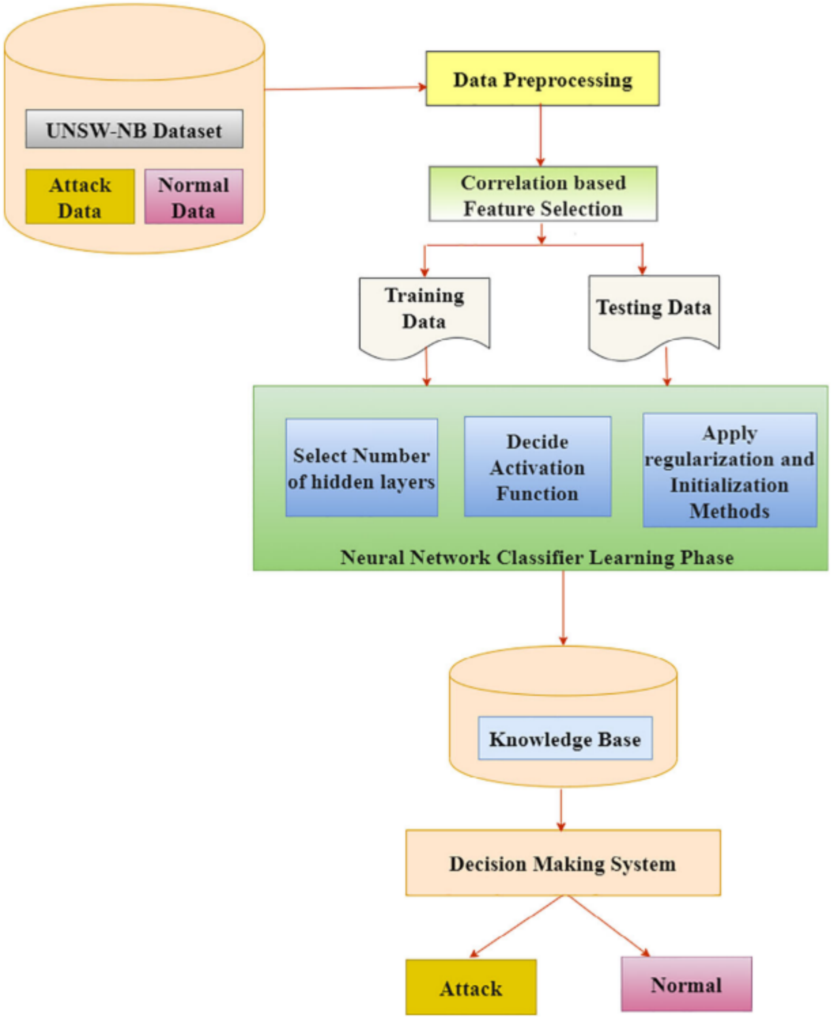


Methodology

Classifier learning:

The ANN classifier in our project utilizes user-specified initialization of activation functions, regularization techniques, and layer sizes. The output layer is determined based on the classes present, leading to accurate predictions.

Proposed intrusion detection


UNSW-NB Dataset
Attack Data
Normal Data
Data Preprocessing
Correlation based Feature Selection
Training Data
Testing Data
Select Number of hidden layers
Decide Activation Function
Apply regularization and Initialization Methods
Neural Network Classifier Learning Phase
Knowledge Base
Decision Making System
Attack
Normal

Gaussian Naive Bayes:

Simplicity and Speed: Gaussian Naive Bayes is a simple and computationally efficient algorithm. It can handle large datasets quickly, making it suitable for processing the NSL-KDD dataset efficiently.

Independence Assumption: The algorithm assumes that features are conditionally independent given the class label. This assumption may hold reasonably well for some features in the NSL-KDD dataset, allowing Gaussian Naive Bayes to make accurate predictions using fewer parameters.

Gaussian Distribution: It assumes that features follow a Gaussian distribution. This assumption may reasonably apply to certain features in the dataset. If the features align with this distribution, the algorithm can effectively model the probability density functions for each class, leading to accurate predictions.

Adequate Class Separation: The NSL-KDD dataset may exhibit well-separated class distributions. This clear separation between different classes makes it easier for Gaussian Naive Bayes to discriminate between them and achieve a good accuracy.

Decision Tree:

Nonlinear Relationships: Decision trees can capture nonlinear relationships between features and the target variable. The NSL-KDD dataset may contain complex patterns and interactions that decision trees are able to model effectively, leading to accurate predictions.

Feature Selection: Decision trees perform automatic feature selection by evaluating the importance of different features in the tree construction process. If the dataset contains informative features that are relevant for distinguishing between different classes, decision trees can identify and utilize them to achieve high accuracy.

Handling of Irrelevant Features: Decision trees are robust to irrelevant features, as they can ignore such features when constructing the tree. The NSL-KDD dataset may contain irrelevant features that do not contribute much to the classification task. Decision trees can effectively filter out these irrelevant features and focus on the informative ones.

Handling of Categorical and Numerical Features: Decision trees can handle both categorical and numerical features naturally. The NSL-KDD dataset contains a mix of categorical and numerical attributes, and decision trees can handle them without requiring extensive preprocessing or encoding.

Interpretability: Decision trees provide human-readable rules that can be easily interpreted and understood. This interpretability allows analysts to gain insights into the decision-making process and potential intrusion patterns, aiding in further analysis and system improvement.

Support Vector Machine:

It work by finding an optimal hyperplane that separates different classes of data with the largest possible margin. SVMs can handle both linear and nonlinear classification problems using different kernel functions.

Effective Separation: SVM finds a clear boundary between different classes, allowing them to accurately separate normal instances from various attacks in the NSL-KDD dataset.

Margin Maximization: SVM aims to create the largest possible gap between the decision boundary and the nearest data points of each class, leading to better accuracy and generalization.

Nonlinear Transformations: SVMs can handle complex patterns by transforming the data into a higher-dimensional space, enabling them to find intricate decision boundaries.

Handling of Outliers: SVMs are less affected by outliers as they focus on important data points near the decision boundary, resulting in improved accuracy.

Tuning Parameters: SVMs have adjustable settings that can be optimized to improve their performance, such as choosing the right kernel function and regularization parameters.

Logistic Regression:

Logistic regression is a classification algorithm that estimates the probability of an instance belonging to a specific class using the logistic function. It finds a linear decision boundary that separates different classes in the data. The model is trained by adjusting parameters to maximize the likelihood of observed class labels

Linear Separation: Logistic regression finds a straight line (or hyperplane) that separates normal instances from attacks in the NSL-KDD dataset.

Probability Estimation: It estimates the probability of an instance belonging to a specific class using the logistic function (sigmoid function).

Optimization Objective: Logistic regression adjusts its parameters during training to improve the likelihood of correctly predicting class labels.

Handling Nonlinear Relationships: Logistic regression can handle some nonlinear relationships but may struggle with complex nonlinear patterns.

Interpretability: Logistic regression provides interpretable results by estimating the impact of each feature on the class probabilities

Random Forest:

Ensemble of Trees: Random Forest is an algorithm that combines multiple decision trees to make predictions.

Voting/Averaging: Each tree in the Random Forest independently makes a prediction, and the final prediction is determined by majority voting (classification) or averaging (regression) the predictions of all the trees.

Robustness: Random Forest is robust to noise and outliers in the data due to the averaging or voting process, which reduces their impact on the final prediction.

Handling Nonlinear Patterns: Random Forest can capture complex relationships and nonlinear patterns in the data by constructing multiple trees and combining their predictions.

Improved Accuracy: By leveraging the collective knowledge of multiple trees, Random Forest often achieves higher accuracy compared to individual decision trees

ANN:

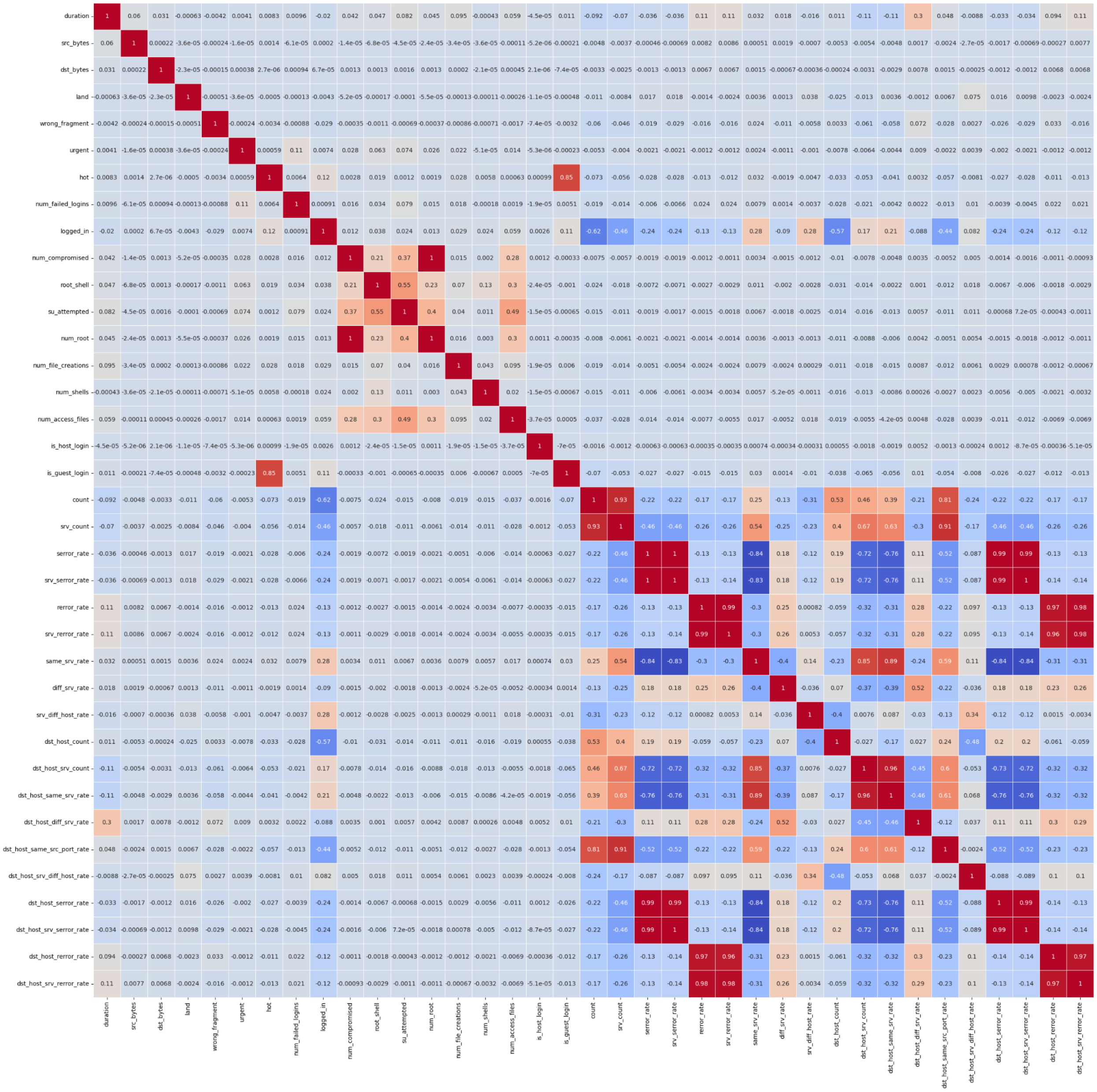
1. After the dataset is preprocessed to handle missing values, normalize features, and encode categorical variables
2. using ANN to classify network traffic as normal or malicious and subsequently classifying the malicious attack into which type of attack it is
3. The ANN architecture is designed, comprising multiple layers with various activation functions and regularization techniques.
4. We designed our ANN to have 3 hidden layers , the activation methods in the hidden layers were ‘**relu**’ , ‘**sigmoid**’ and ‘**sofmax**’.
5. We compiled our ANN based on categorical_crossentropy which measures the discrepancy between the predicted output and the true output labels. Categorical crossentropy is commonly used for multi-class classification problems, where the target variable is categorical and mutually exclusive.
6. Finally we compared the various metrics such as accuracy , recall , f1_score and precision.

Meta_Model Creation:

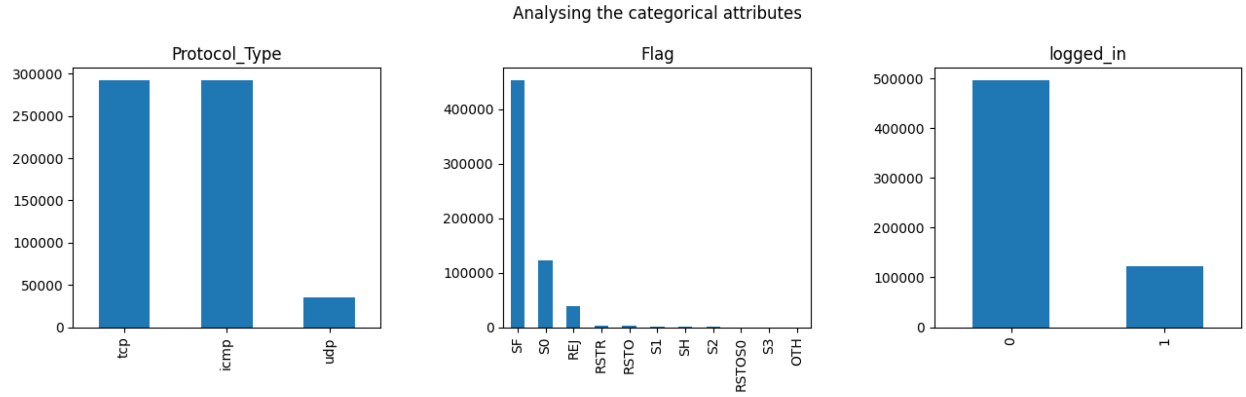
We combine multiple models and use that to create 3 meta models which give better accuracy compared to individual models.

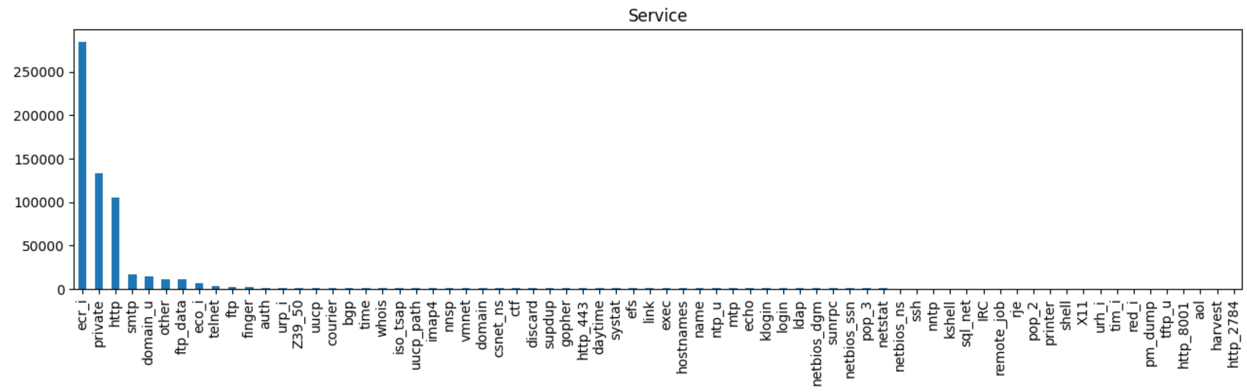
Results

Correlation Matrix:

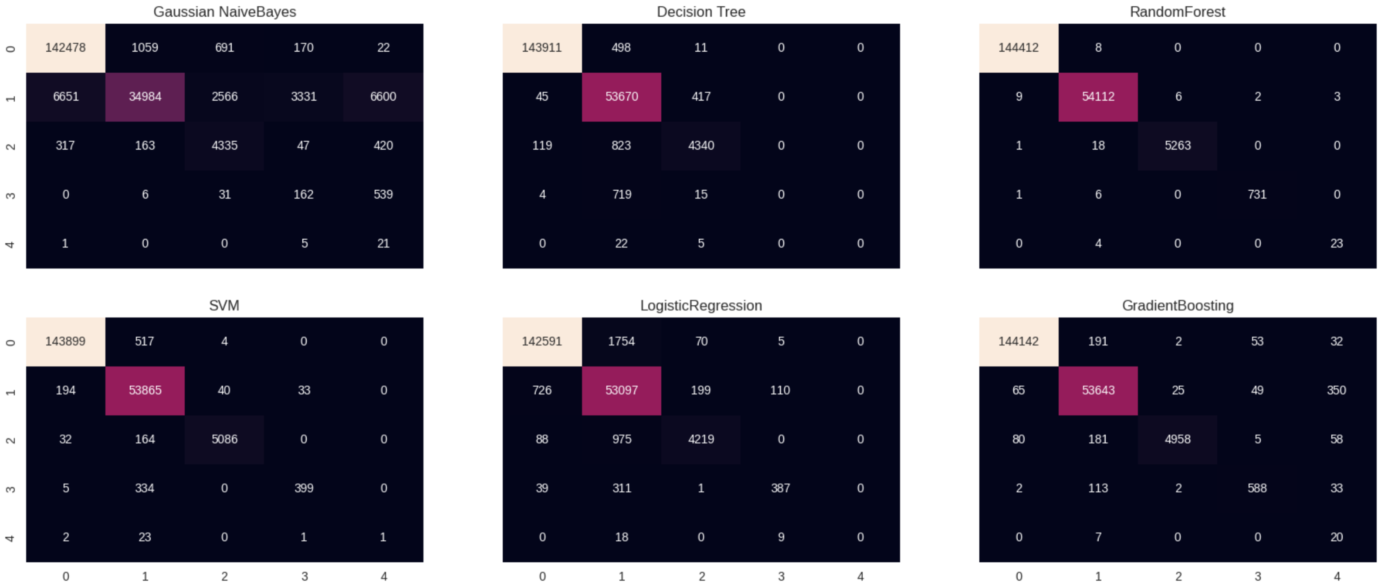


Category Analysis:





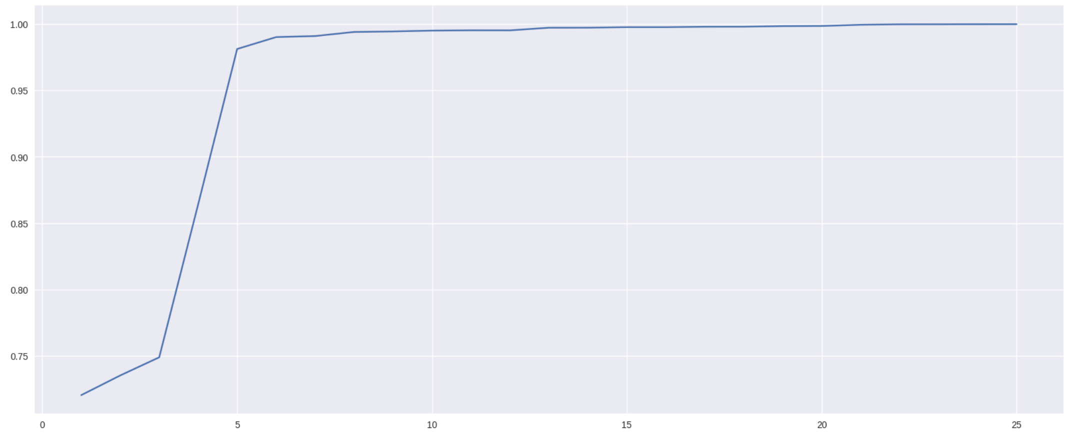
Confusion matrix of applied models:



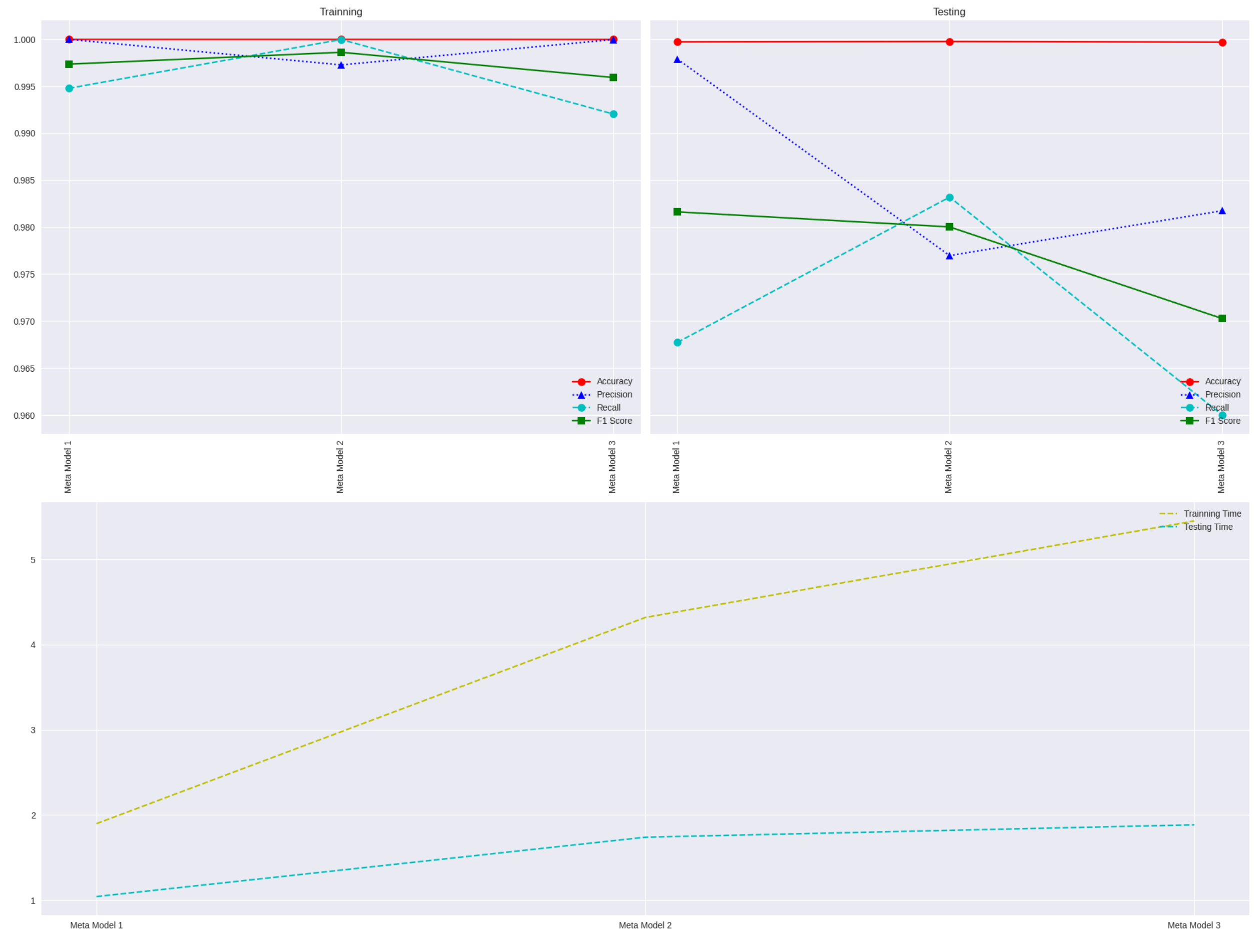
Accuracy vs Features graph:

X-axis: Number of features

Y-axis: Accuracy



Meta model graphs:



Accuracies of Applied Model:

Gaussian NaiveBayes : 88.94
Decision Tree : 98.69
RandomForest : 99.97
SVM : 99.34
LogisticRegression : 97.89
GradientBoosting : 99.39