AI-Powered Cybersecurity

By:

Fadya Hesham ELOraby 2205177

Malak Mohamed Osman 2205059

Abdelrahma Ayman Saad 2205033

AI-powered IDS project

01

Improved Detection Accuracy:

identify complex patterns and subtle anomalies in network traffic that traditional systems might miss.

02

Real-Time Response:

enabling quick detection and response to potential intrusions

03

Scalability and Efficiency:

AI models can handle larger datasets and complex networks more efficiently than traditional IDS, scaling better as the system grows.

Algorithm used : RANDOM FOREST

01 **Handles High-Dimensional Data:**

it divides features into subtrees and trains each tree individually, then averages the results

Algorithm used : RANDOM FOREST

02 **Feature Importance**

Random Forest can evaluate and rank feature importance, helping to identify the most relevant features for predicting the target variable.

Algorithm used : RANDOM FOREST

03 **Robust to Overfitting**

By averaging the predictions of many decision trees, it reduces the risk of overfitting, making it more reliable for generalization to unseen data.

QUESTION

WHY NOT A TRADITIONAL DECISION TREE?

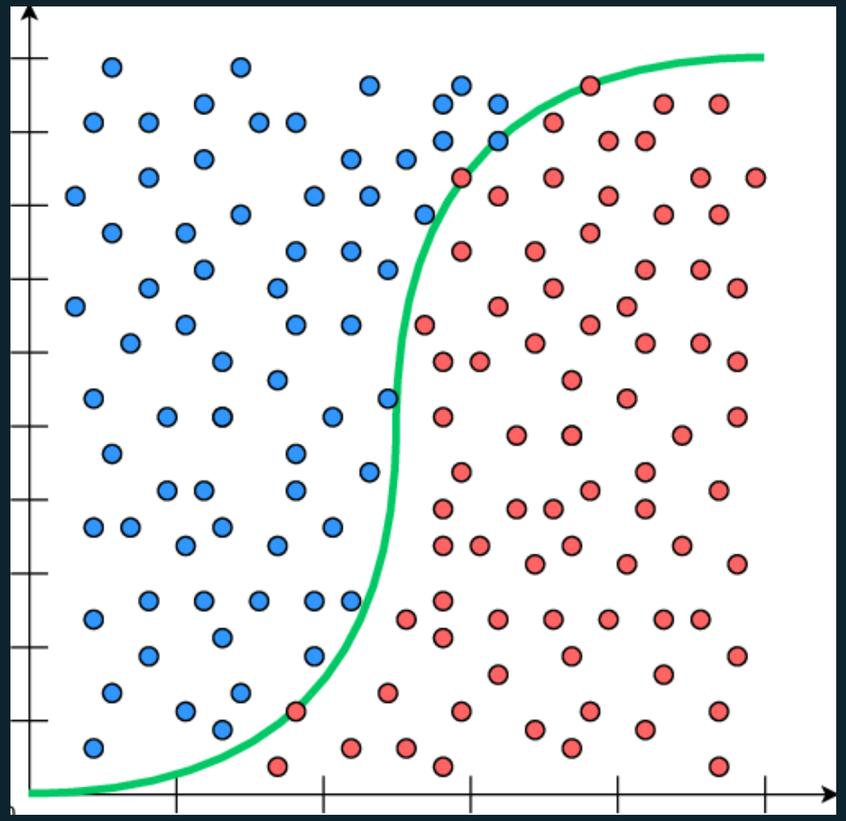
QUESTION

WHY NOT A TRADITIONAL DECISON TREE?

The main advantage of Random Forest over a traditional decision tree is that it works by reducing variance, which helps prevent overfitting and reduces model’s ability to generalize to new data.

QUESTION

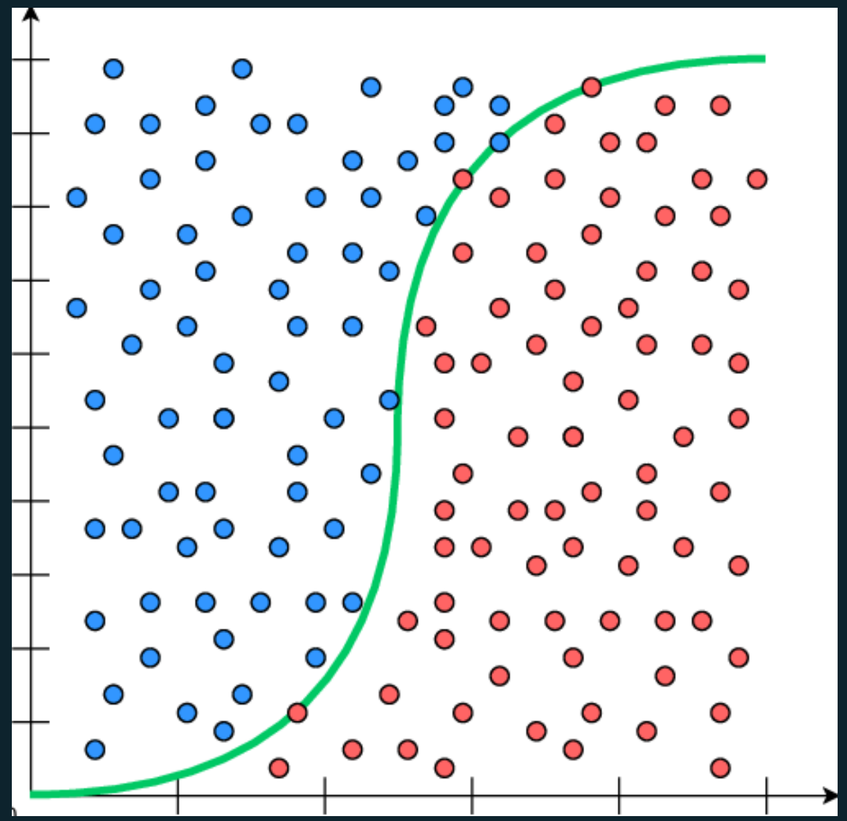
WHY NOT USE LOGISTIC REQRESSION INSTEAD?



QUESTION

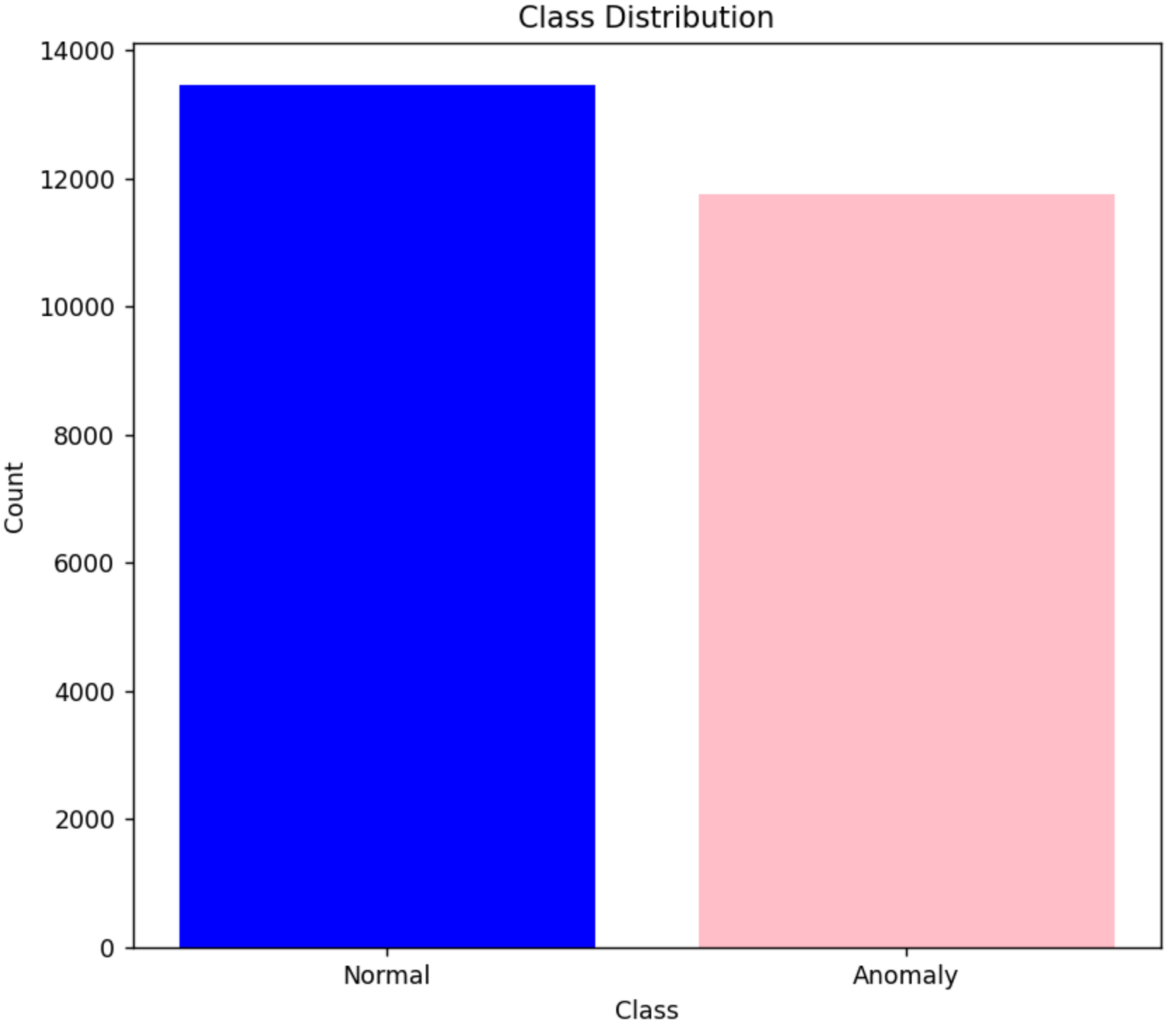
WHY NOT USE LOGISTIC REGRESSION INSTEAD?

Logistic regression assumes a linear relationship between features and the target, which may not hold in Intrusion Detection Systems (IDS), where relationships between features and intrusion types are often complex and non-linear.



1. Data Preprocessing

- **Data Loading**: The code loads the dataset from an ARFF file (KDDTrain+_20Percent.arff) and uses the metadata to assign column names.
- **Data Cleaning:** The data is **cleaned by decoding byte columns** into strings using
- **Class Distribution:** The distribution of 'normal' and 'anomaly' classes is printed. It shows that the dataset contains a total of 25,192 rows with **13,449 'normal' rows and 11,743 'anomaly' rows**
- **Class Encoding**: The 'class' column is **mapped** to numerical labels (0 for anomaly, 1 for normal).
- **Categorical Columns**: Features like 'protocol_type', 'service', 'flag', etc., are encoded into numeric values using LabelEncoder for further processing.


Class Distribution
14000
12000
10000
8000
6000
4000
2000
0
Count
Normal
Anomaly
Class

2. Data Normalization

- **Normalization: Feature scaling is applied using MinMaxScaler to scale the feature values between 0 and 1 .**

- **Feature and Label Separation: The features (input variables) and labels (target) are separated into x and y respectively .**

Here is a sample of the data after preprocessing:

A sample of data after:

	duration	protocol_type	service	flag	...	dst_host_srv_serror_rate	dst_host_rerror_rate	dst_host_srv_rerror_rate	class
0	0.0	0.5	0.292308	0.9	...	0.00	0.05	0.00	1
1	0.0	1.0	0.630769	0.9	...	0.00	0.00	0.00	1
2	0.0	0.5	0.707692	0.5	...	1.00	0.00	0.00	0
3	0.0	0.5	0.338462	0.9	...	0.01	0.00	0.01	1
4	0.0	0.5	0.338462	0.9	...	0.00	0.00	0.00	1
5	0.0	0.5	0.707692	0.1	...	0.00	1.00	1.00	0
6	0.0	0.5	0.707692	0.5	...	1.00	0.00	0.00	0
7	0.0	0.5	0.707692	0.5	...	1.00	0.00	0.00	0
8	0.0	0.5	0.738462	0.5	...	1.00	0.00	0.00	0
9	0.0	0.5	0.707692	0.5	...	1.00	0.00	0.00	0

[10 rows x 42 columns]

3. Model Training

- **Train-Test Split: The data is split into training and test sets (80% train, 20% test) using train_test_split.**
- **Random Forest Classifier: The model is trained using the RandomForestClassifier with different numbers of estimators (10, 50, 100, 200, 500) to find the optimal number of estimators for the best performance.**

4. Model Evaluation

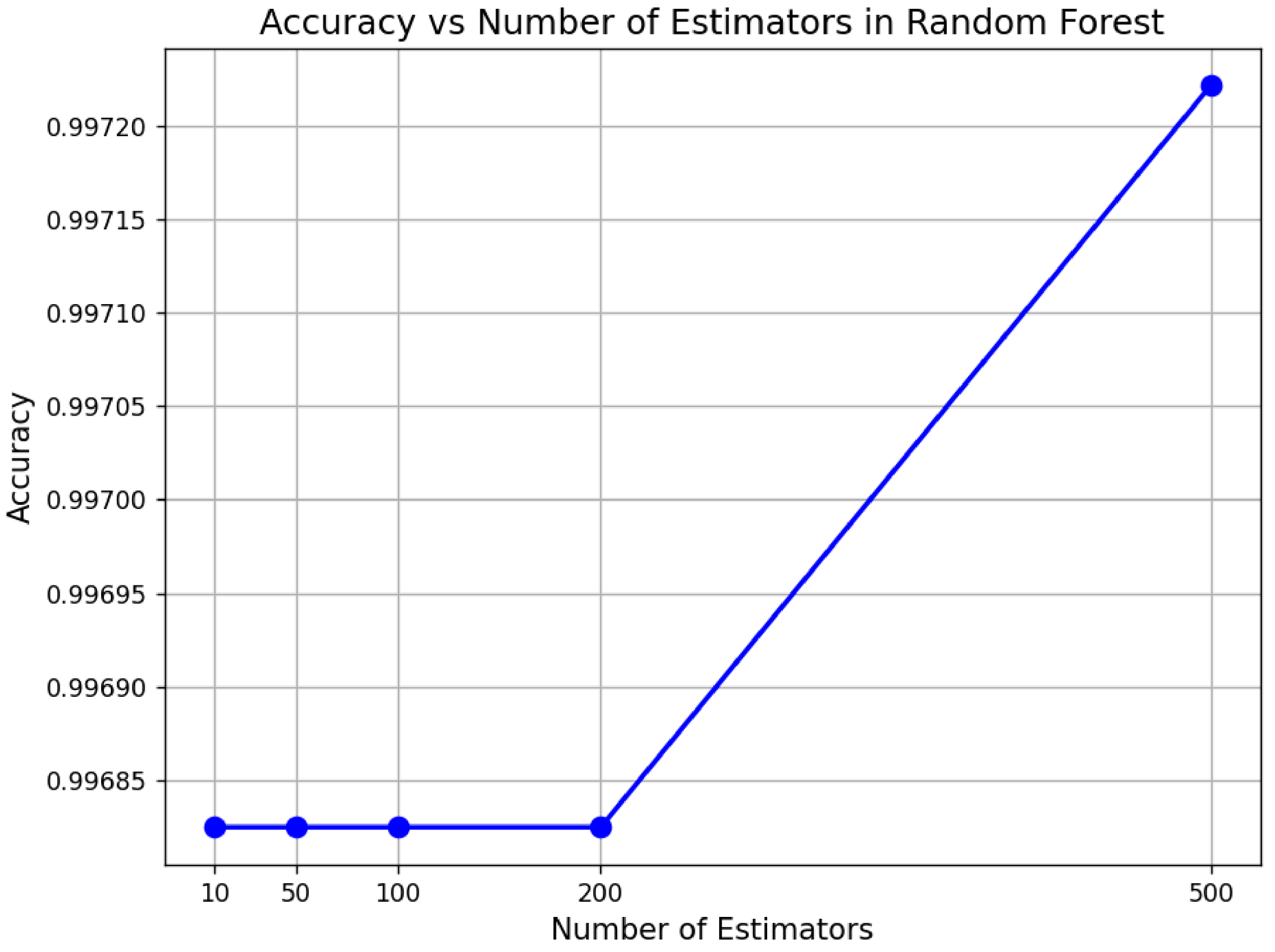
- **Accuracy: For each number of estimators, the accuracy is calculated and displayed.**
- **Classification Report: The classification report, which includes precision, recall, f1-score, and support for each class, is printed for every model.**
- **Best Model Selection: The model with the highest accuracy (500 estimators) is selected as the best .**

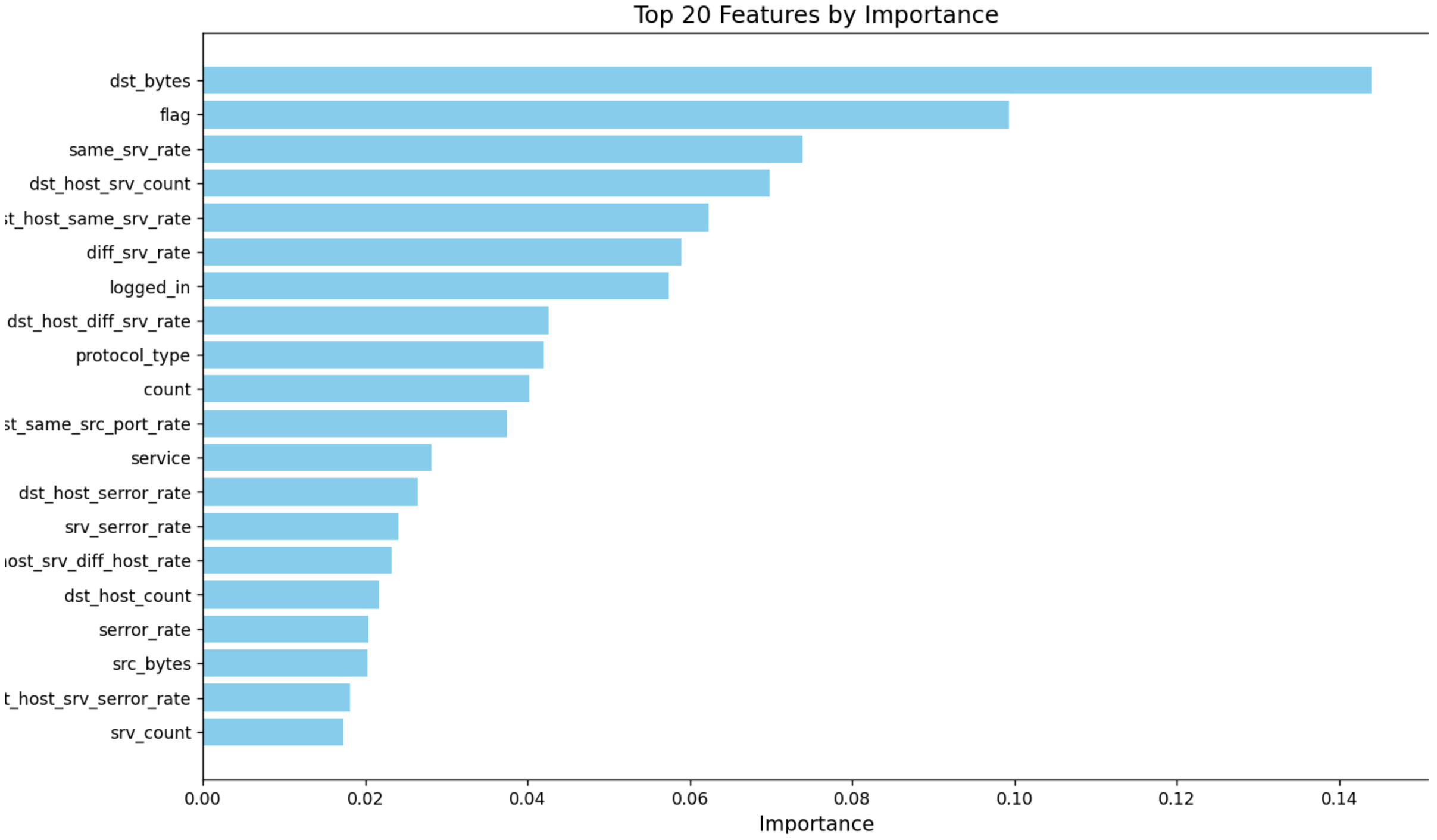
5. Results

- **Comparison: The accuracy results for different numbers of estimators are compared. It is observed that the accuracy reaches 99.68% with 10, 50, 100, and 200 estimators, and slightly improves to 99.72% with 500 estimators.**
- **Best Model: The model with 500 estimators achieves the best accuracy of 0.99722.**
- **Performance: The classification report shows the precision, recall, and f1-score values for both 'normal' and 'anomaly' classes.**

6. Visualization

- **Accuracy vs. Number of Estimators: A plot is generated to visualize how accuracy changes with the number of estimators in the Random Forest model.**
- **Feature Importance: A bar plot of the top 20 most important features is shown based on their feature importance values derived from the best model (according to their information gain) .**


Accuracy vs Number of Estimators in Random Forest
0.99720
0.99715
0.99710
0.99705
0.99700
0.99695
0.99690
0.99685
Accuracy
10
50
100
200
500
Number of Estimators


Top 20 Features by Importance
dst_bytes
flag
same_srv_rate
dst_host_srv_count
st_host_same_srv_rate
diff_srv_rate
logged_in
dst_host_diff_srv_rate
protocol_type
count
st_same_src_port_rate
service
dst_host_serror_rate
srv_serror_rate
ost_srv_diff_host_rate
dst_host_count
serror_rate
src_bytes
t_host_srv_serror_rate
srv_count
0.00
0.02
0.04
0.06
0.08
0.10
0.12
0.14
Importance