Network Denial of Service Detection

-Sanjoy Basu Denver, CO

What is Network Denial of Service?

Denial of Service attack is a malicious attempt to bring down network , web based applications or services by overwhelming these resources with too much data or impairing in some other way

Impacts of Denial of Service Attack on Business

Revenue Loss:
Average cost of downtime is $5600/minute

Productivity Loss:
Critical network system are shutdown
workforce productivity comes to halt

Reputation Damage:
Brand suffers or become casualties of data
Breach

Theft:
Attacks include stolen fund consumer data and
Intellectual property

-Source: Verisign

Top 3 Industries Targeted

1.  **57%** FINANCIAL SERVICES

2.  **26%** IT SERVICES/CLOUD/SAAS

3.  **17%** TELECOM

-Source Versign

Problem statement :

Detect denial of service connections

Approach to the solution

- Anomaly detection
- Supervised machine learning to classify malicious traffic
- Statistical model to identify malicious traffic

Training Data

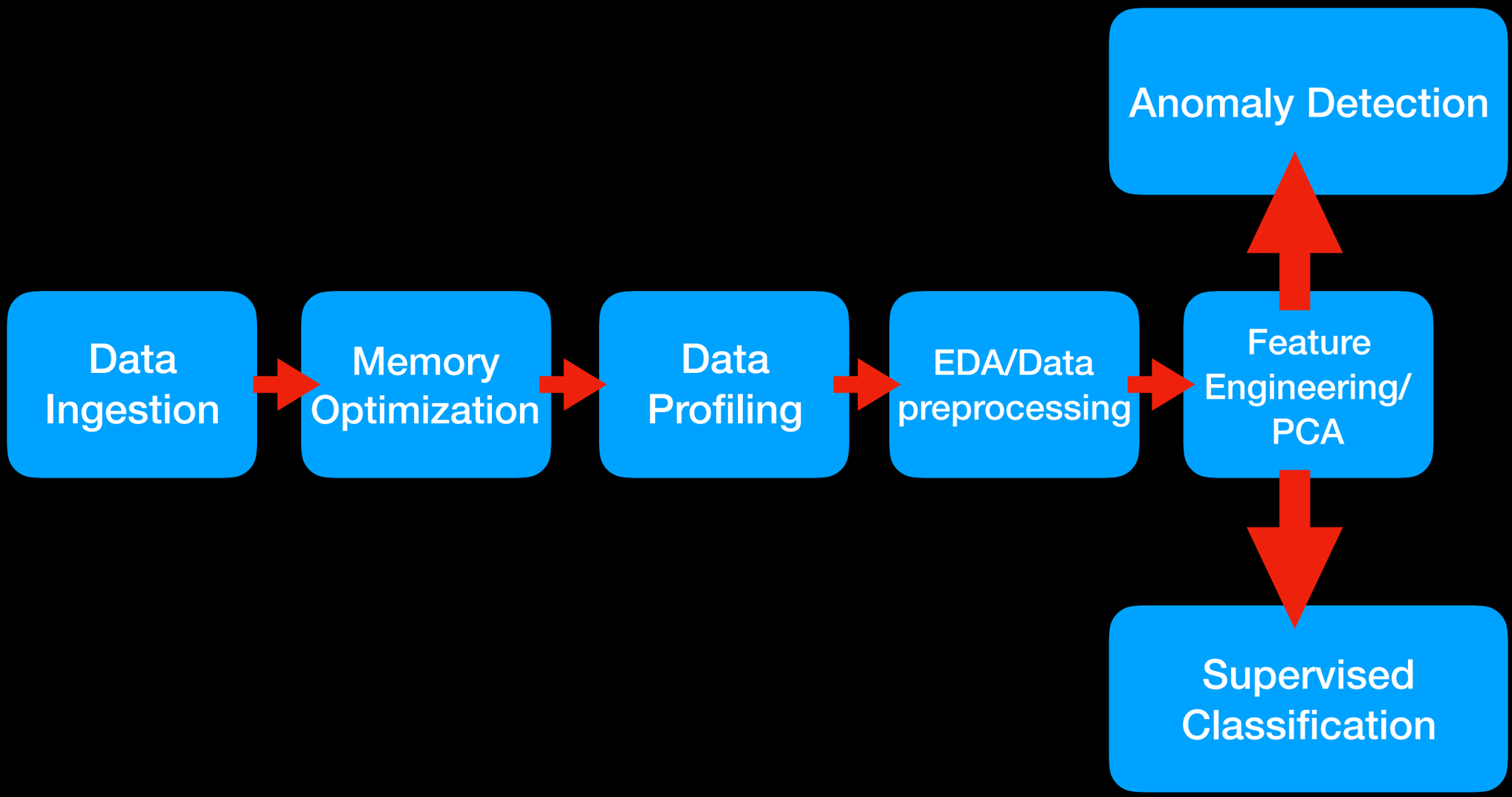
Source:

- Data was generated by MIT Lincoln Laboratory as part of cybersecurity research funded by Defense Advance Research Project Administration(DARPA)
- Data was collected on simulated US Air force Network
- Data was transformed by University of California, Irvine for analytics

Data Profile:

- Number of rows: >4.89 million number of columns:42
- Deep memory usage:>2.5 GB
- Missing data: 0
- Data were not labeled as DOS instead variation of DOS such as *teardrop, neptune, smurf attack* etc

Data Pipeline



Optimizing memory foot print

Challenges:
Deep memory utilization was over 2.5GB building multiple data frame for data analysis is causing memory exhaustion

Solution:

- **Pandas uses default data type consuming certain amount of memory for each columns even if the data can be accommodated in lower memory data type such as init8 or float16**
- **Built python function that estimates the maximum and minimum value the data can be accommodated and change data type accordingly**

Result :
Memory foot print reduced by 88%

New Label Generation

Label of original dataset

The original data set has no label indicating if connection is Denial of service or safe. Instead it has label indicating different variation of denial of service attack such as *smurf attack, Neptune attack , teardrop attack* etc

Approach:

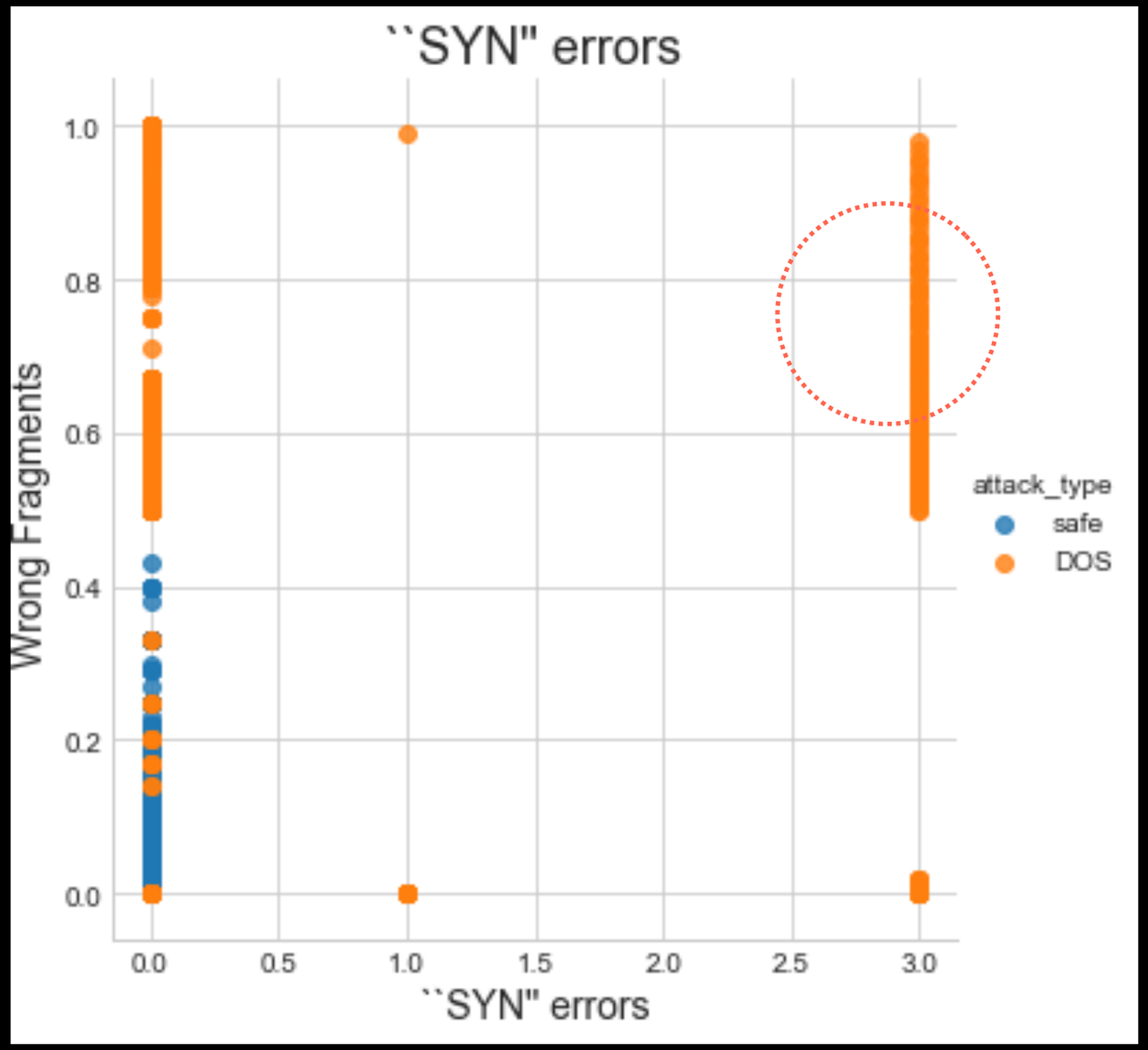
Built python function creating new label using existing label column categorizing traffic as ‘DOS’ or ‘safe’

Advantages of the approach

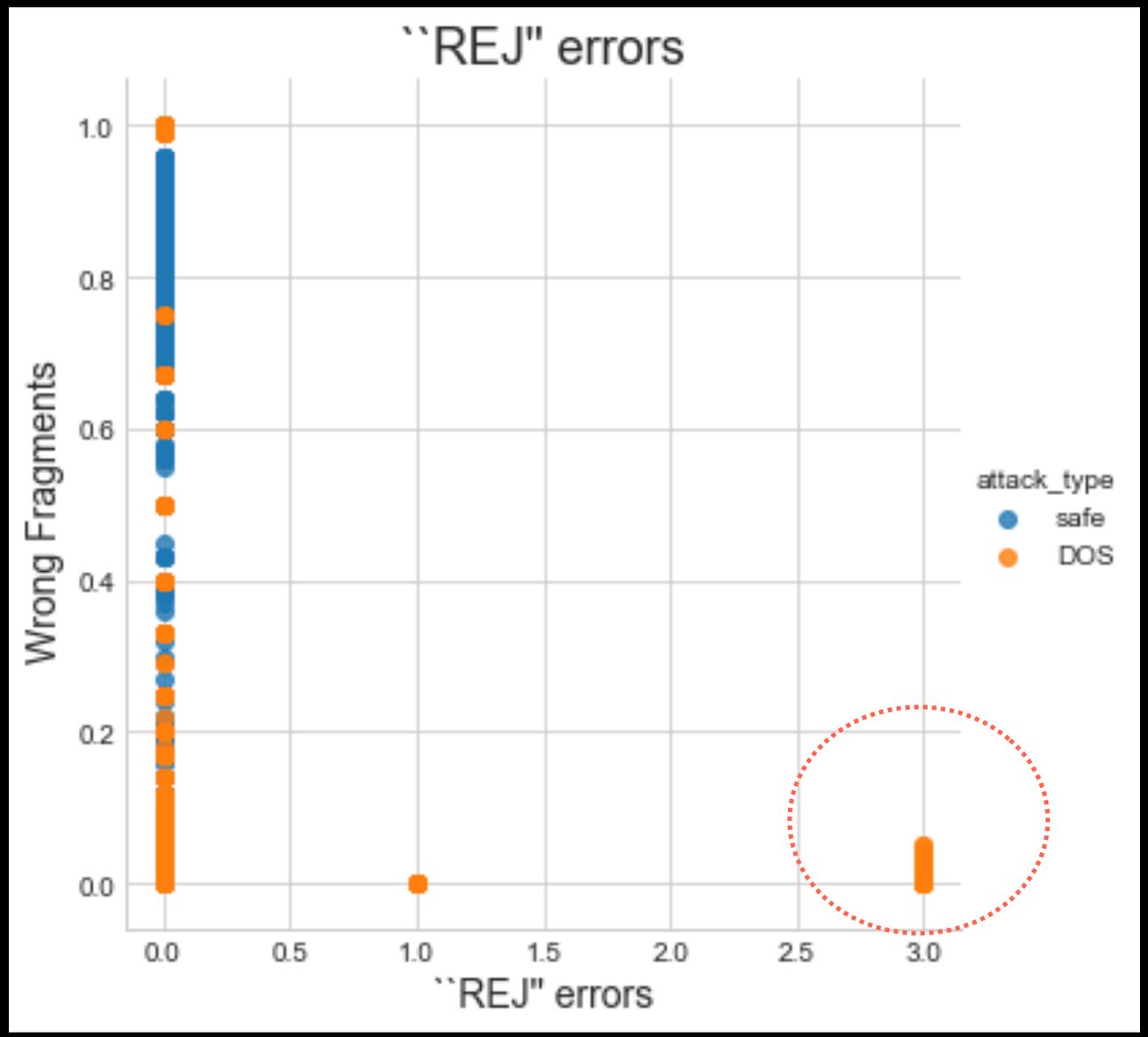
- **Computationally less resource intensive**
- **When a network is victim of denial of service attack it is important to detect Denial of Service attack instead of its variation**

Exploratory Data Analysis
&
Visualization

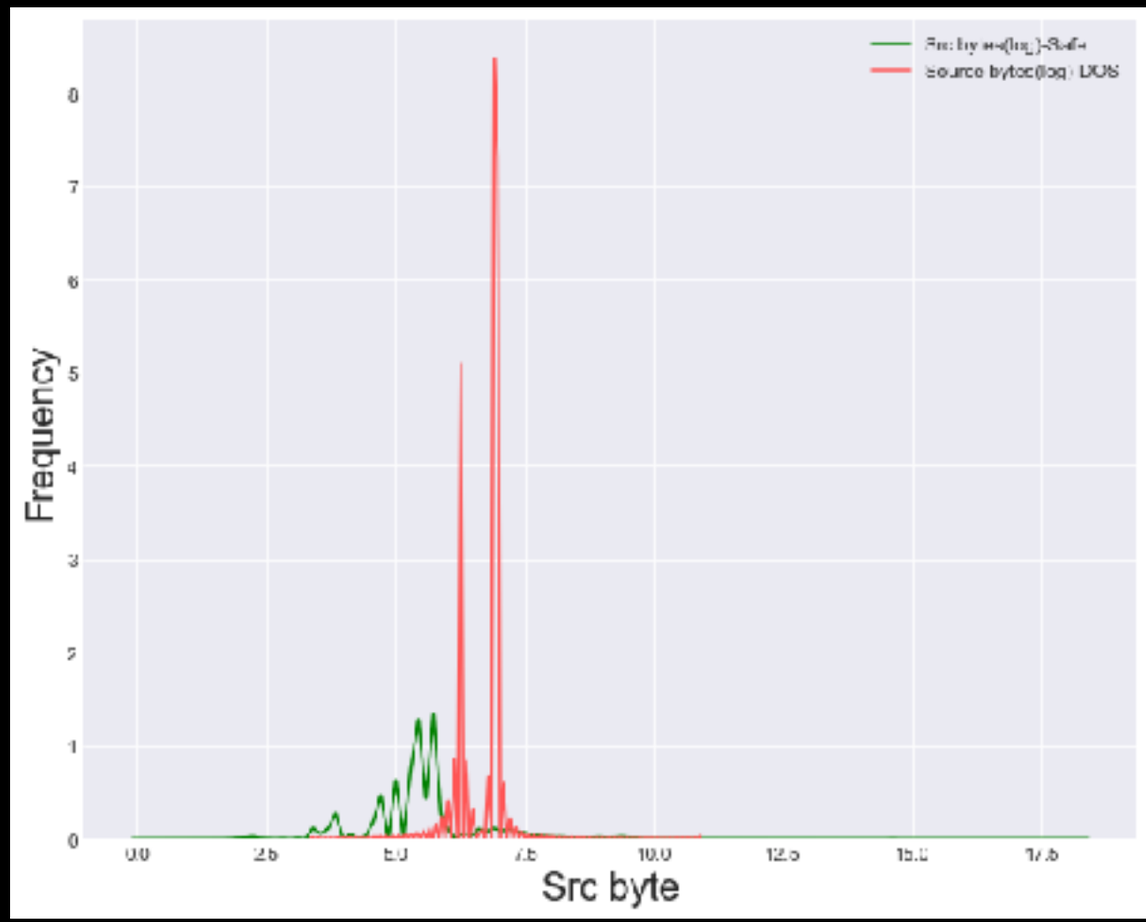
SYN Error vs Wrong Fragments

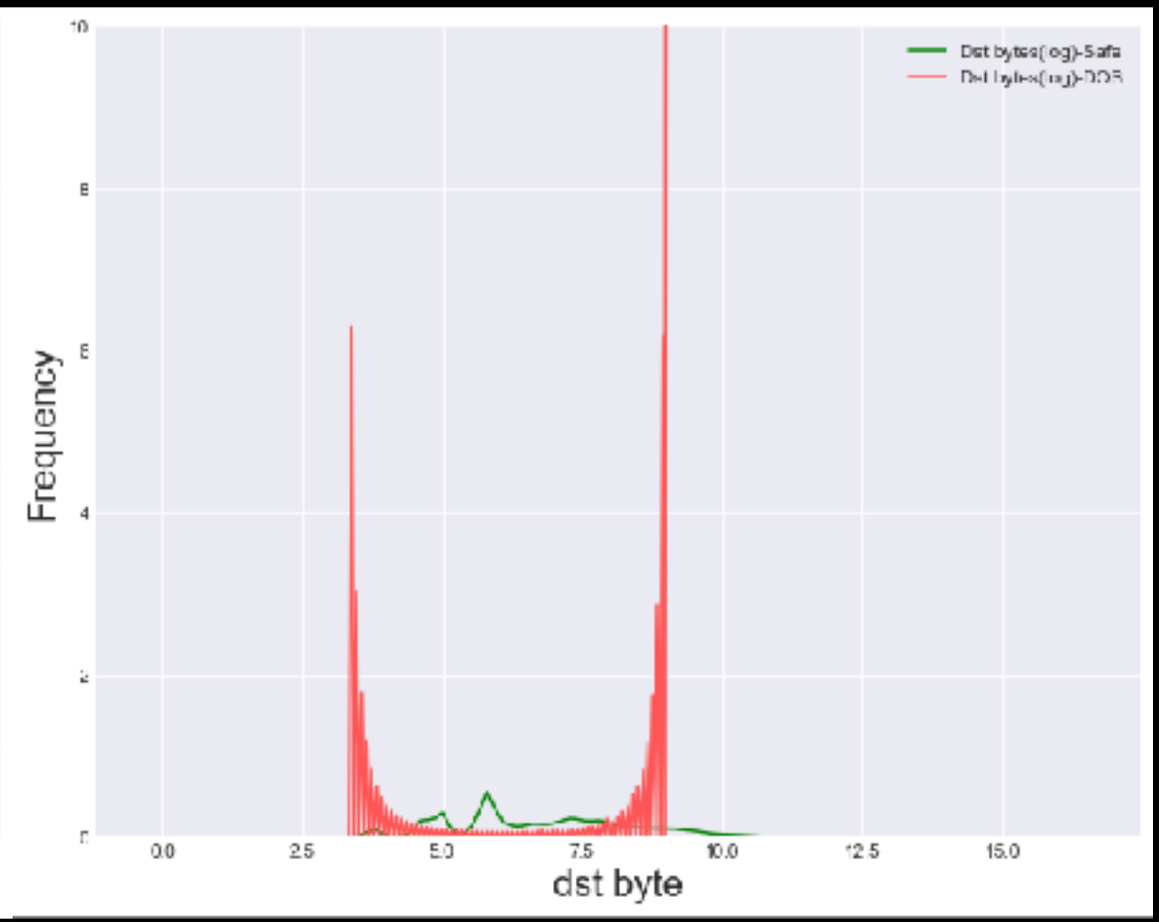


REJ Error vs Wrong Fragments

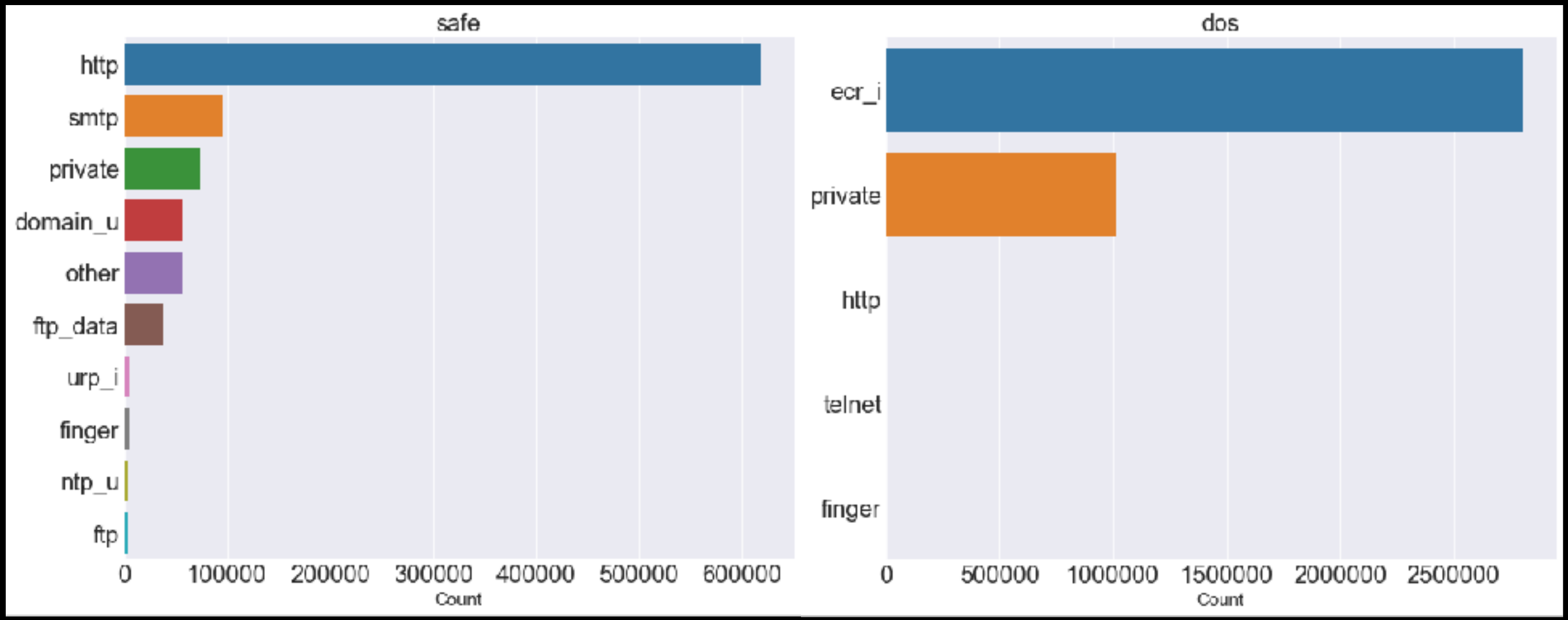


Distribution of Source and Destination bytes size

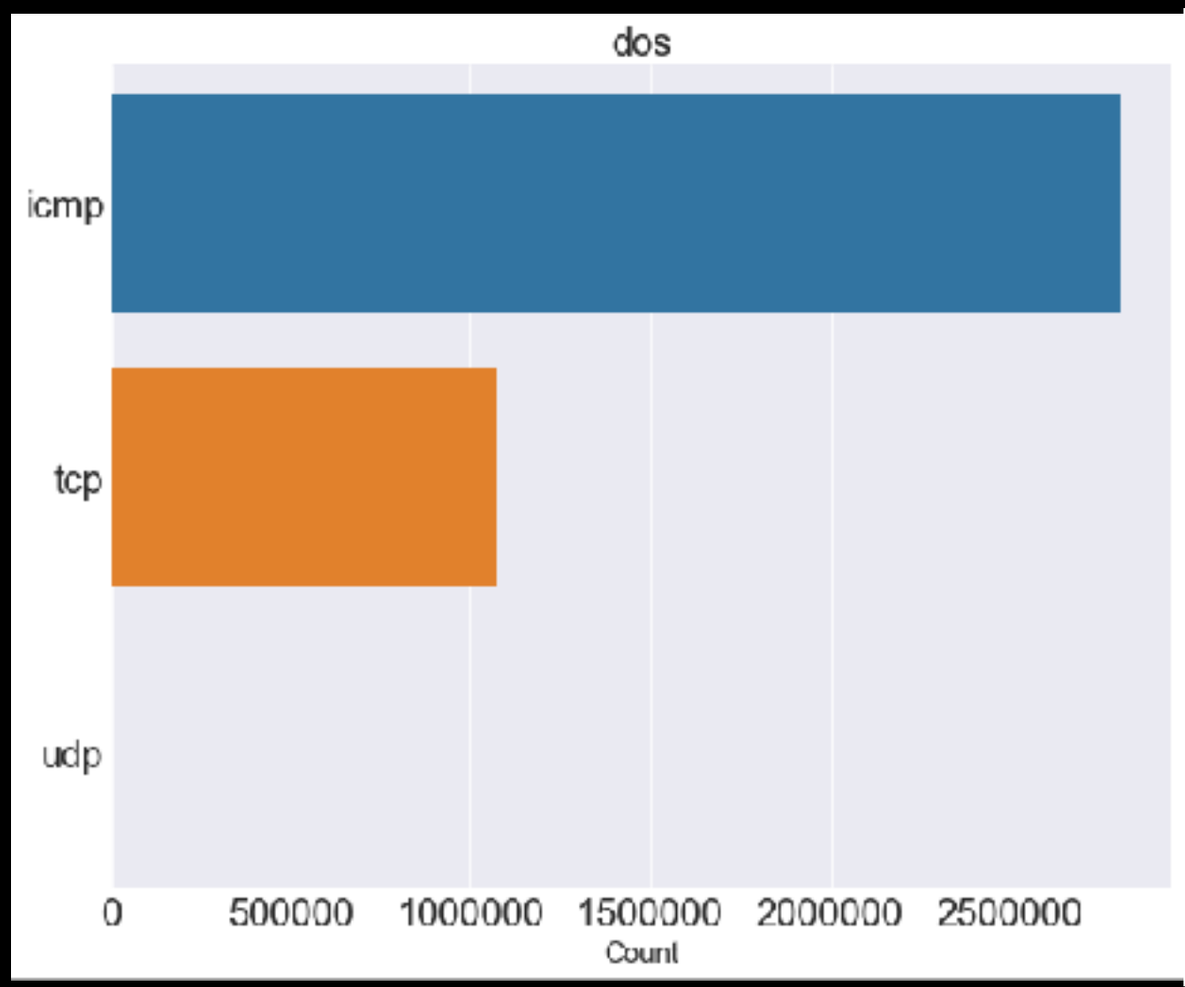


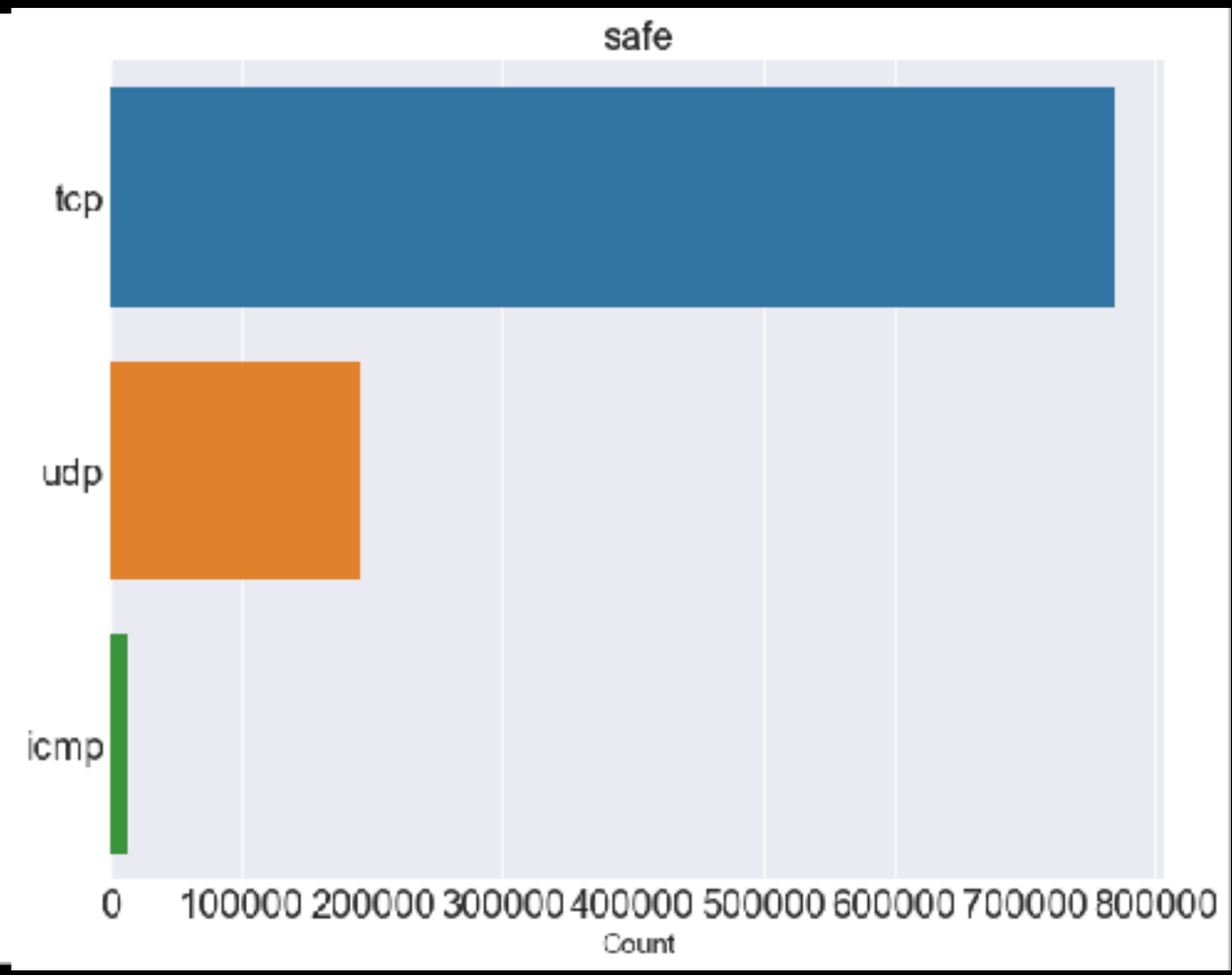


Service type malicious vs safe

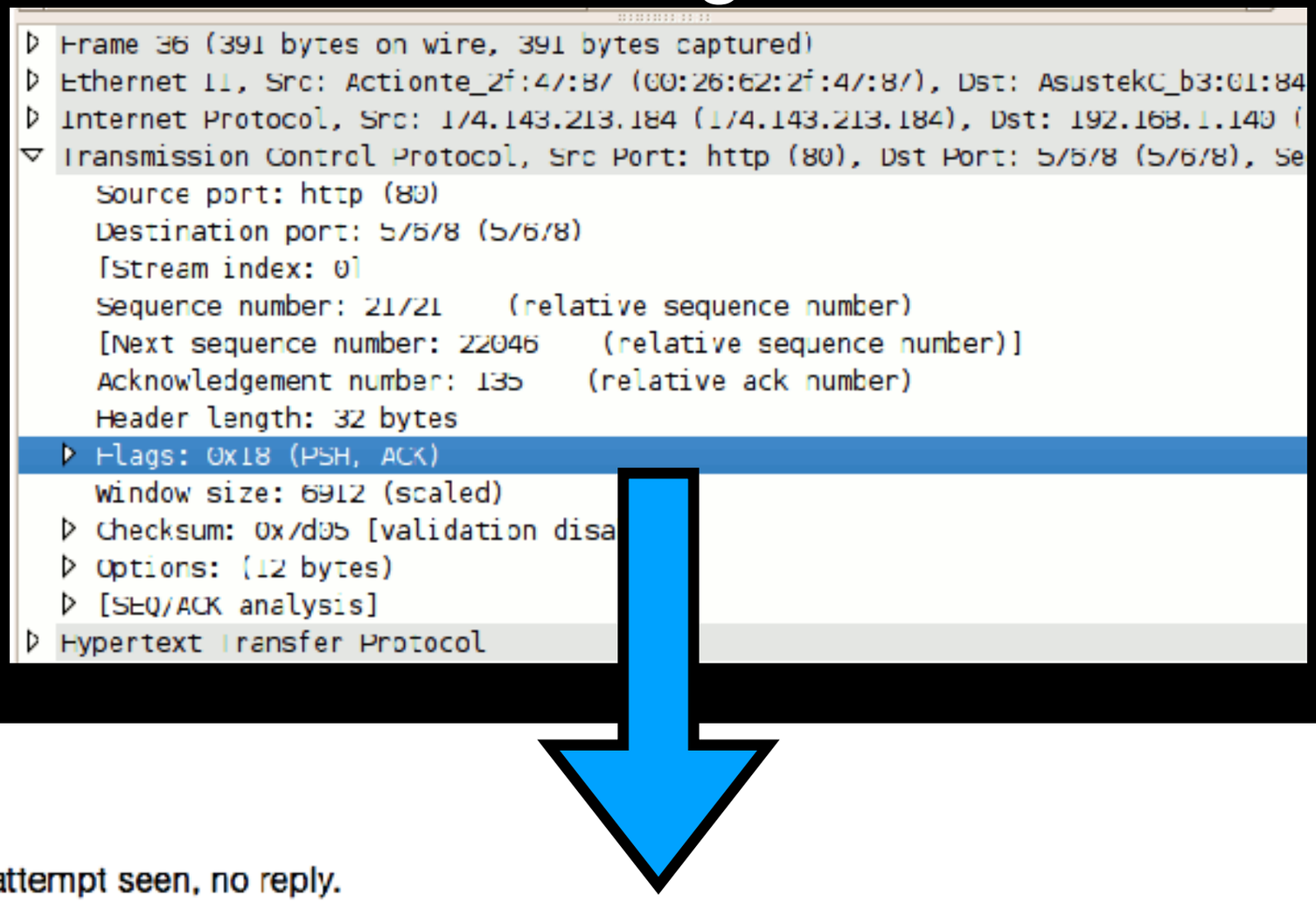


Protocol malicious vs safe





TCP Flags



TCP Flags

S0 Connection attempt seen, no reply.

S1 Connection established, not terminated.

SF Normal establishment and termination. Note that this is the same symbol as for state

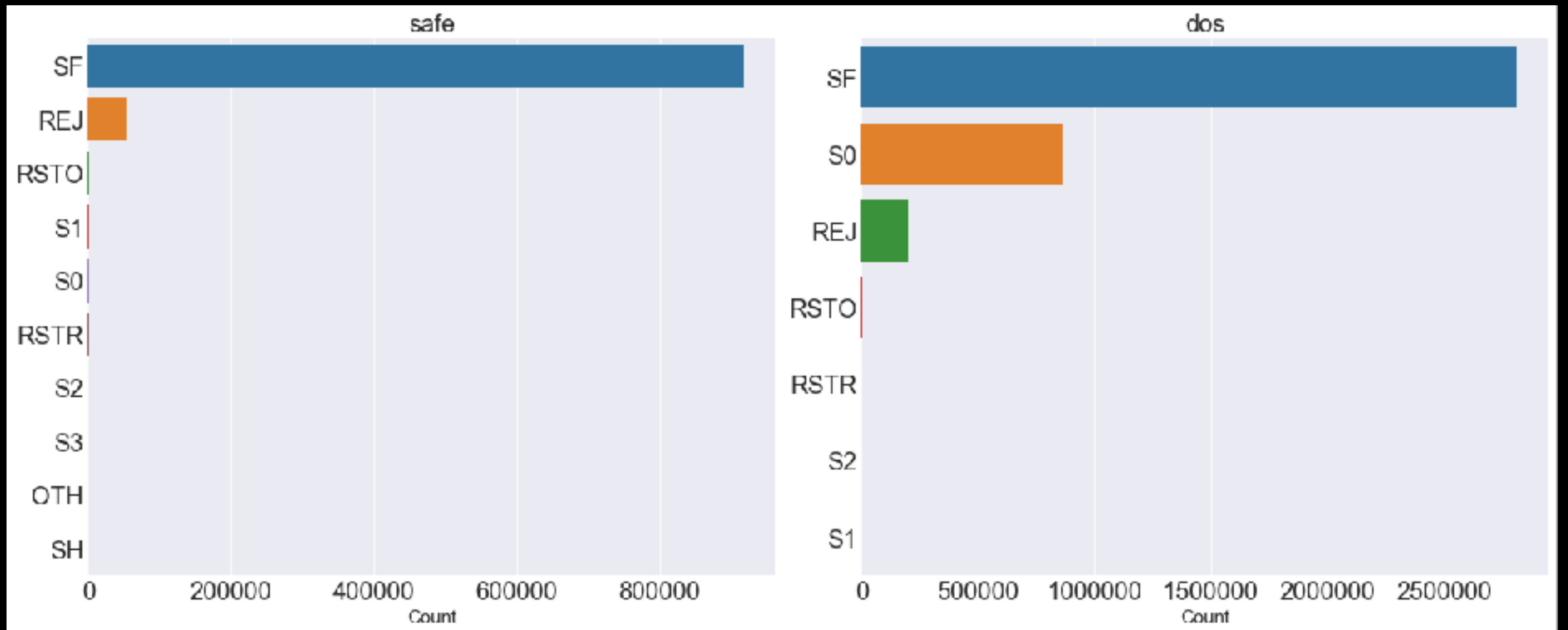
S1. You can tell the two apart because for S1 there will not be any byte counts in the summary, while for SF there will be.

REJ Connection attempt rejected.

S2 Connection established and close attempt by originator seen (but no reply from responder).

S3 Connection established and close attempt by responder seen (but no reply from originator).

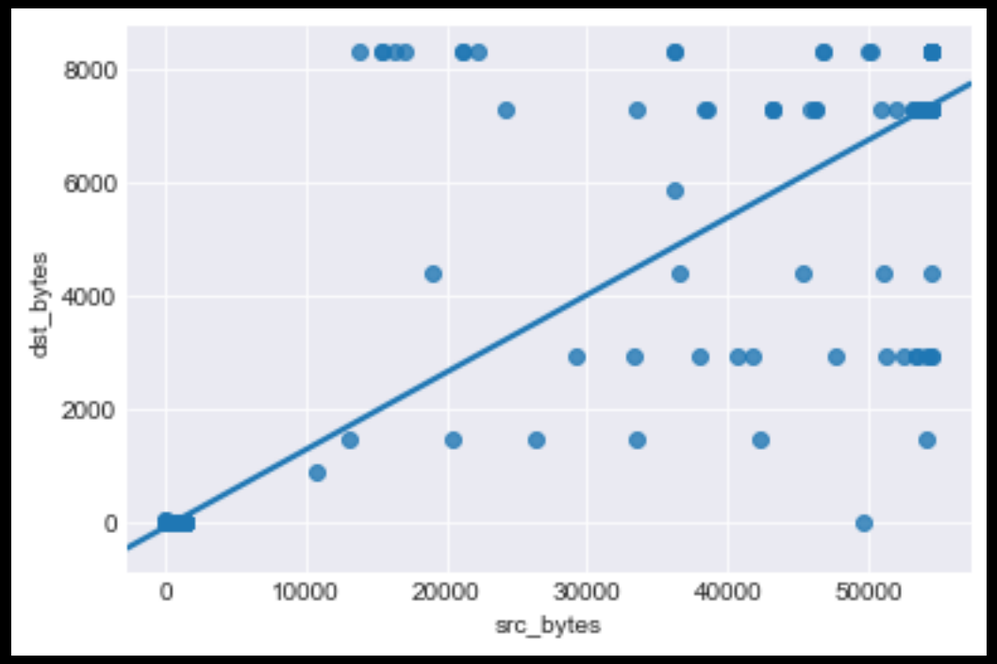
TCP Flags



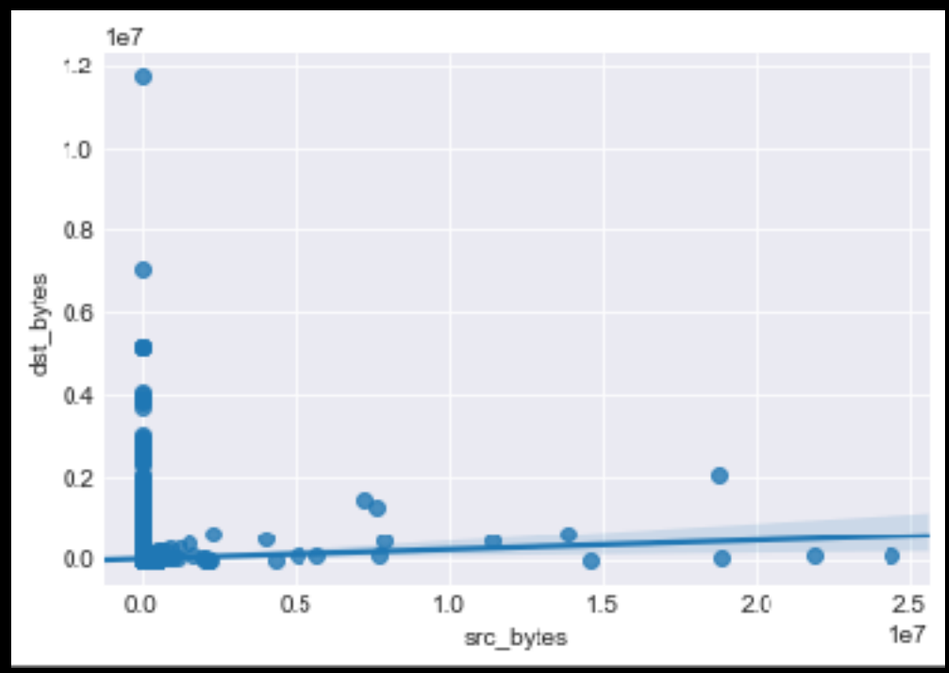
	Safe	DOS
SF	94.28%	72.39%
S0	0.04%	22.33%

Relational plot between dst_bytes and src_bytes

Malicious



Safe



Insight from exploratory data analysis

- Malicious connections are likely to have both wrong fragments and SYN error
- Malicious connections are likely to have both wrong fragments and REJ error
- ecr_i types of service are more likely to be malicious
- Malicious connections are almost likely to have flag S0 set indicating “connection attempt seen but no reply”
- Safe packets either has high destination packet size OR high source packet size but never both
- Malicious packets may have high destination packet size AND high source packet size and they may occur simultaneously
- 72.39% of malicious packets have TCP flags set to SF indicating normal establishment and termination of connection
- Packets with ZERO SYN or REJ error can be malicious

Feature Engineering
&
Dimensionality Reduction

Challenges

- All features do not have linear relation.
- Exploratory data analysis suggest that there are features which shows no linear relation but are important to help in distinguishing normal traffic from Denial of Service
- Principle component analysis assumes features has linear relation.

Strategies to overcome challenges

- Using Kernel based PCA. Disadvantage to the approach is that Kernel PCA demands high resource utilization resulting in Memory Error
- Using domain knowledge we identified set of features that may be related and see if there are linear relation between them.
- Connection features can be divided into 3 subsets

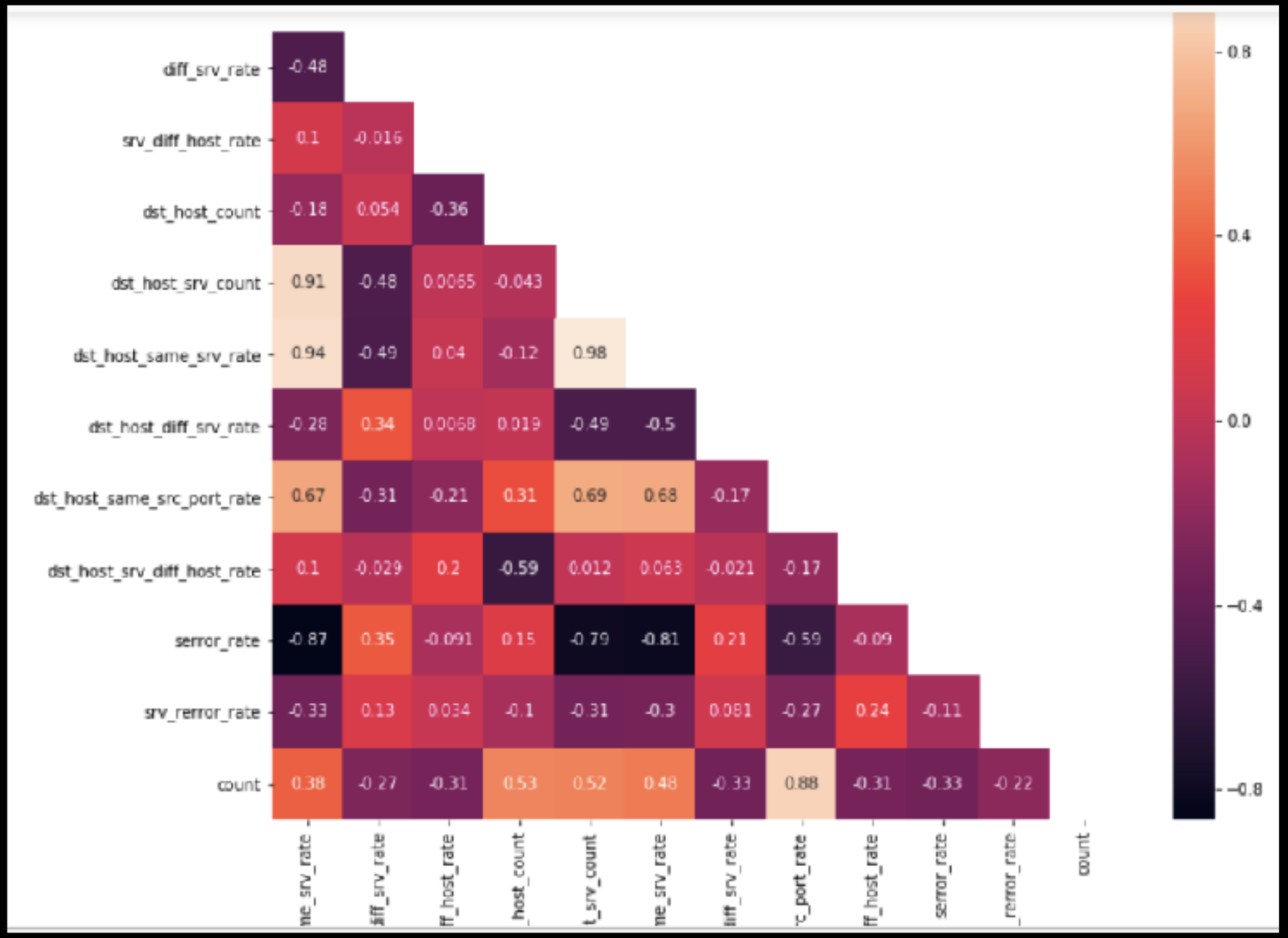
1) TCP packets headers

2) Errors

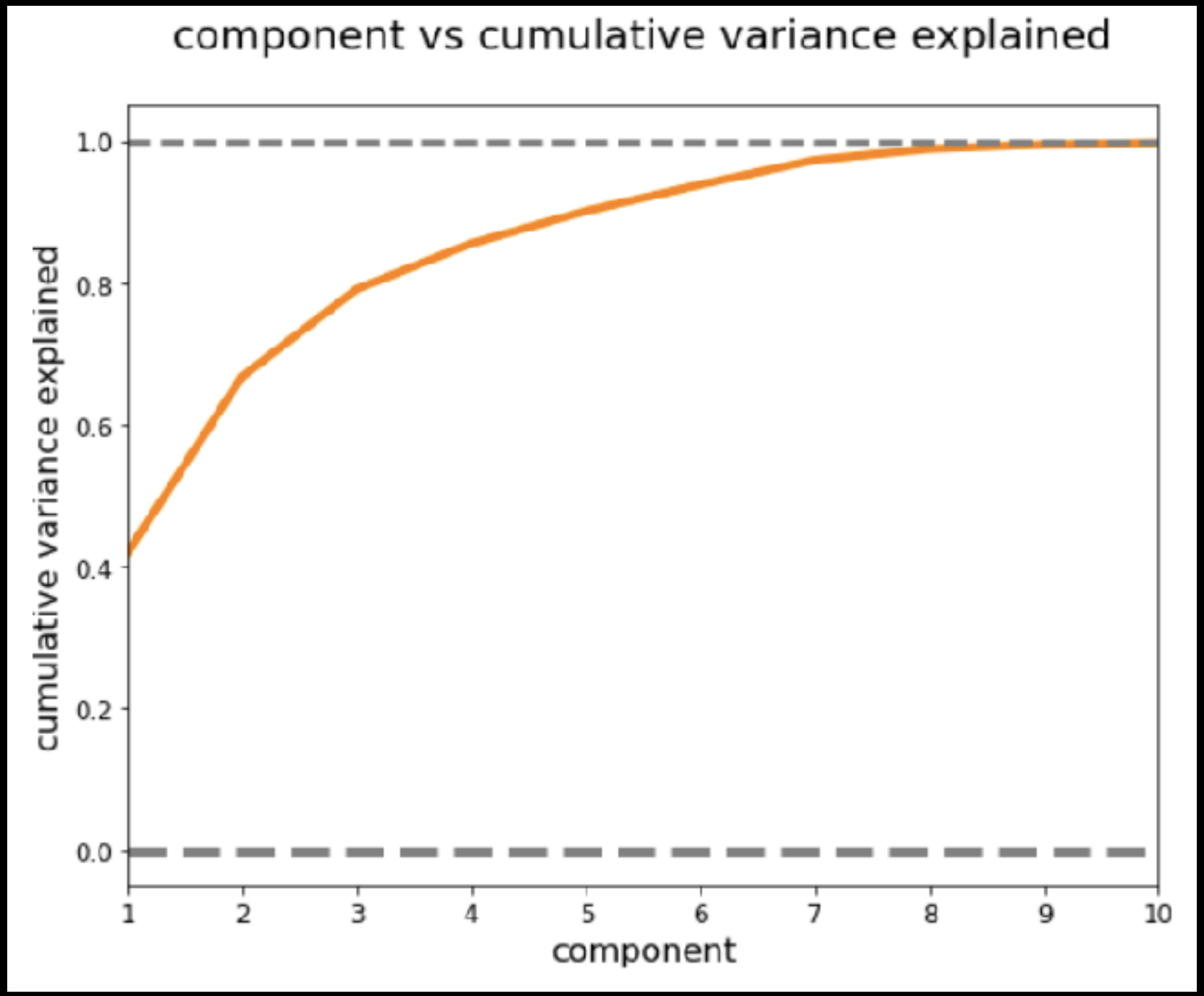
3) Packet size and connection duration

During Exploratory Data Analysis we discovered that Safe packets either has high destination packet size OR high source packet size but never both Malicious packets may have high destination packet size AND high source packet size and they may occur simultaneously. We will use absolute difference in source packets and destination packets as a a feature.
This will also help us to reduce or dimensions

Colinierity between errors

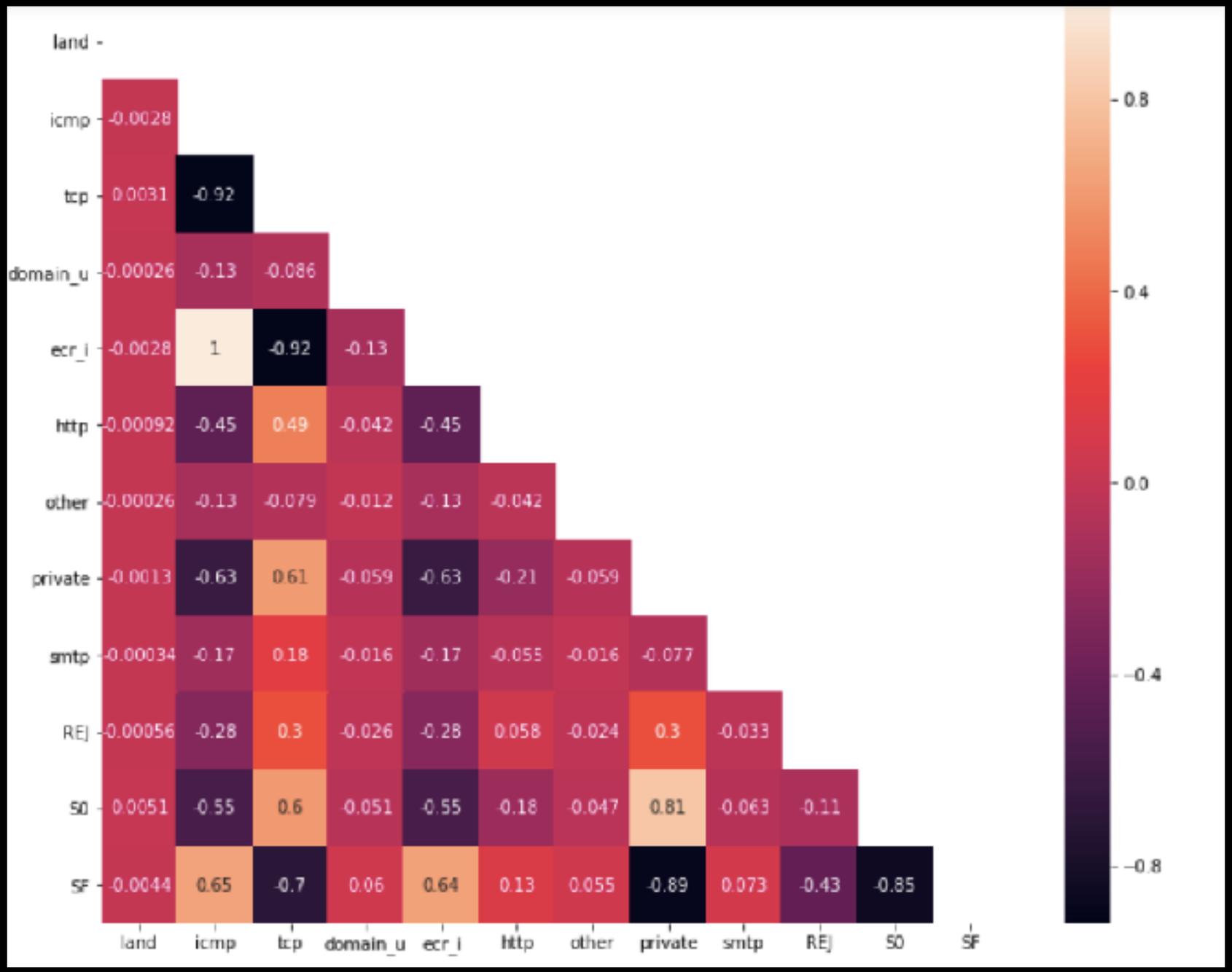


PCA of errors

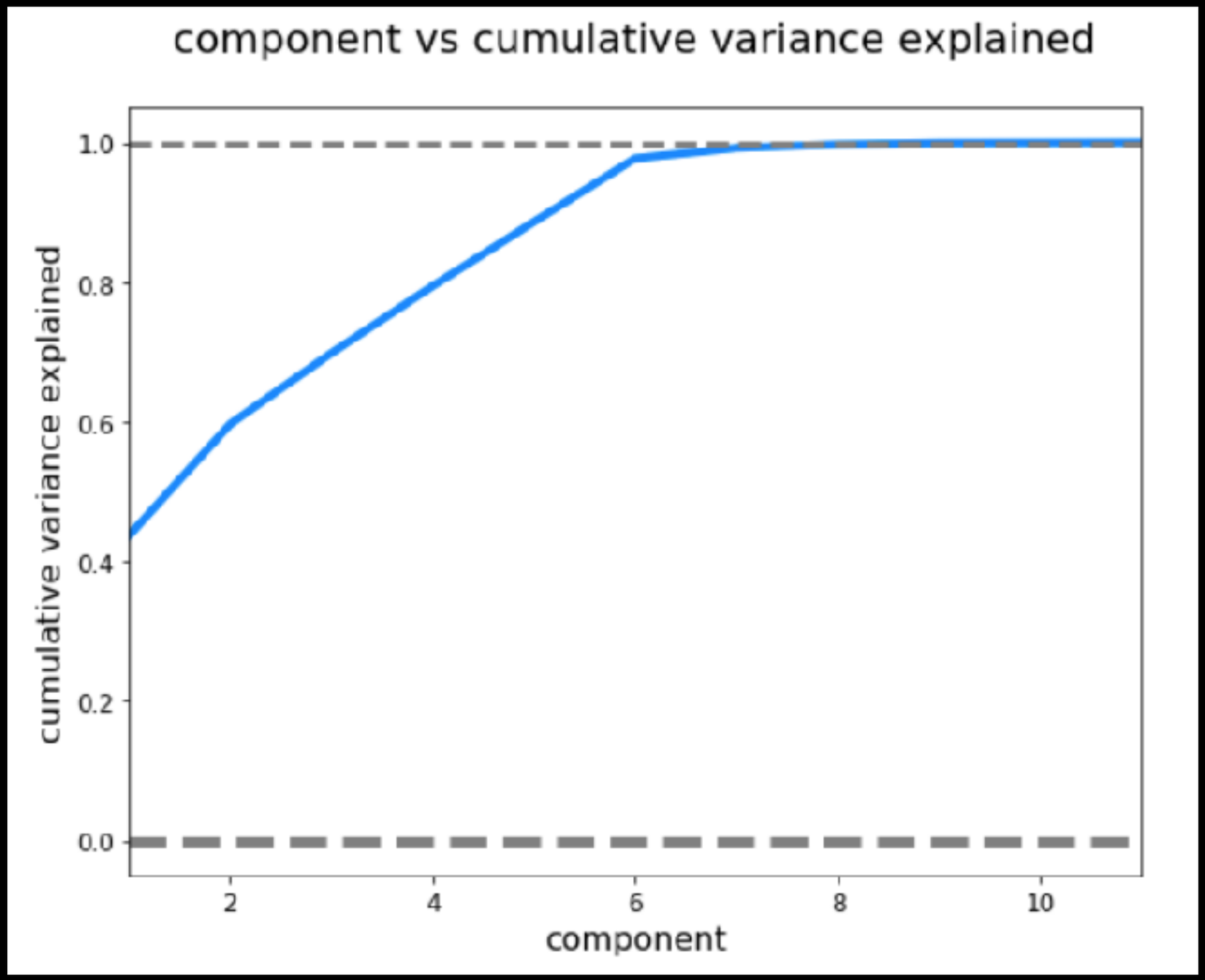


8 out 10 components capture the variance

Colinearity among TCP headers



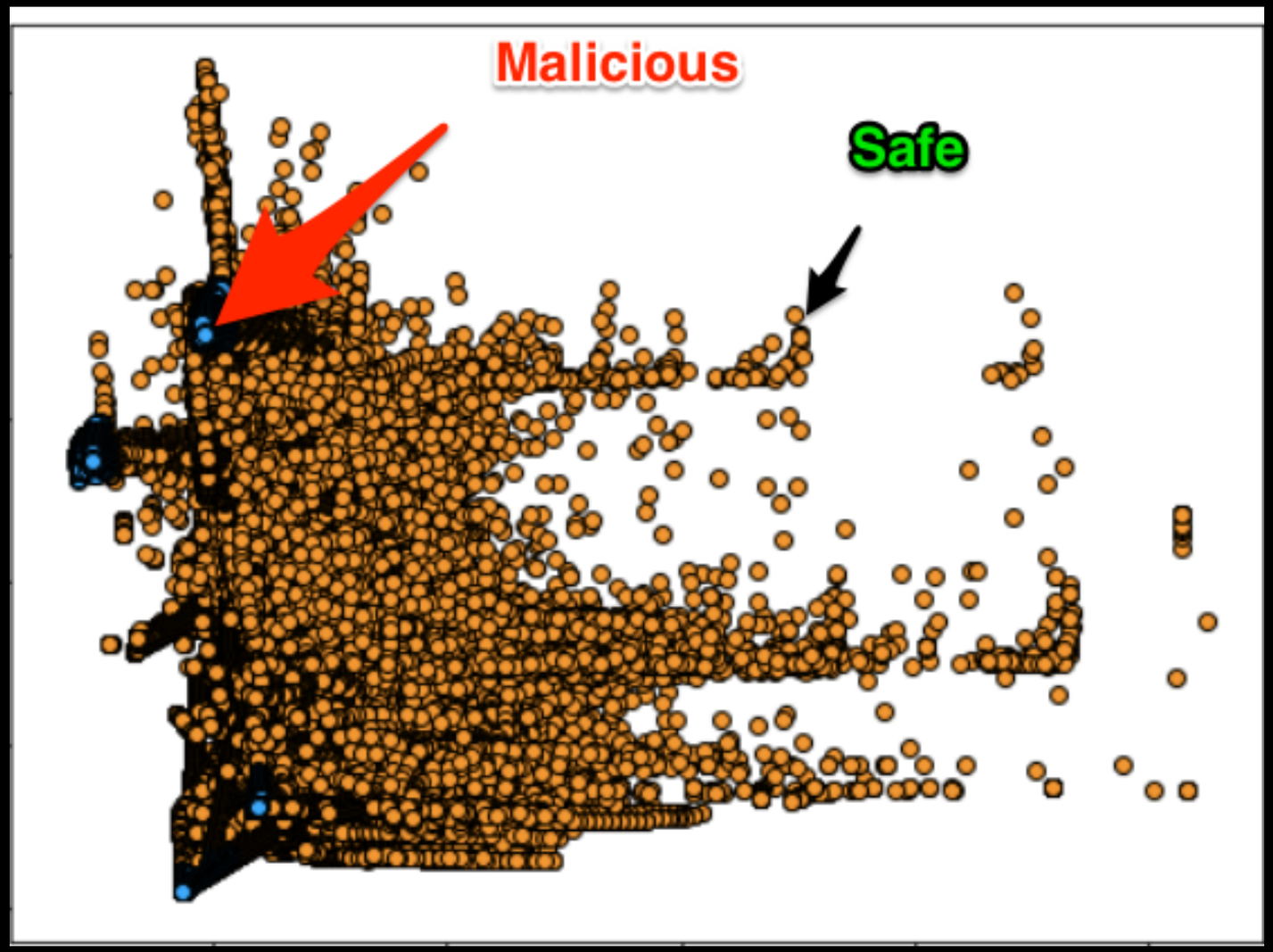
PCA of headers



8 out of 11 components capture all the variance

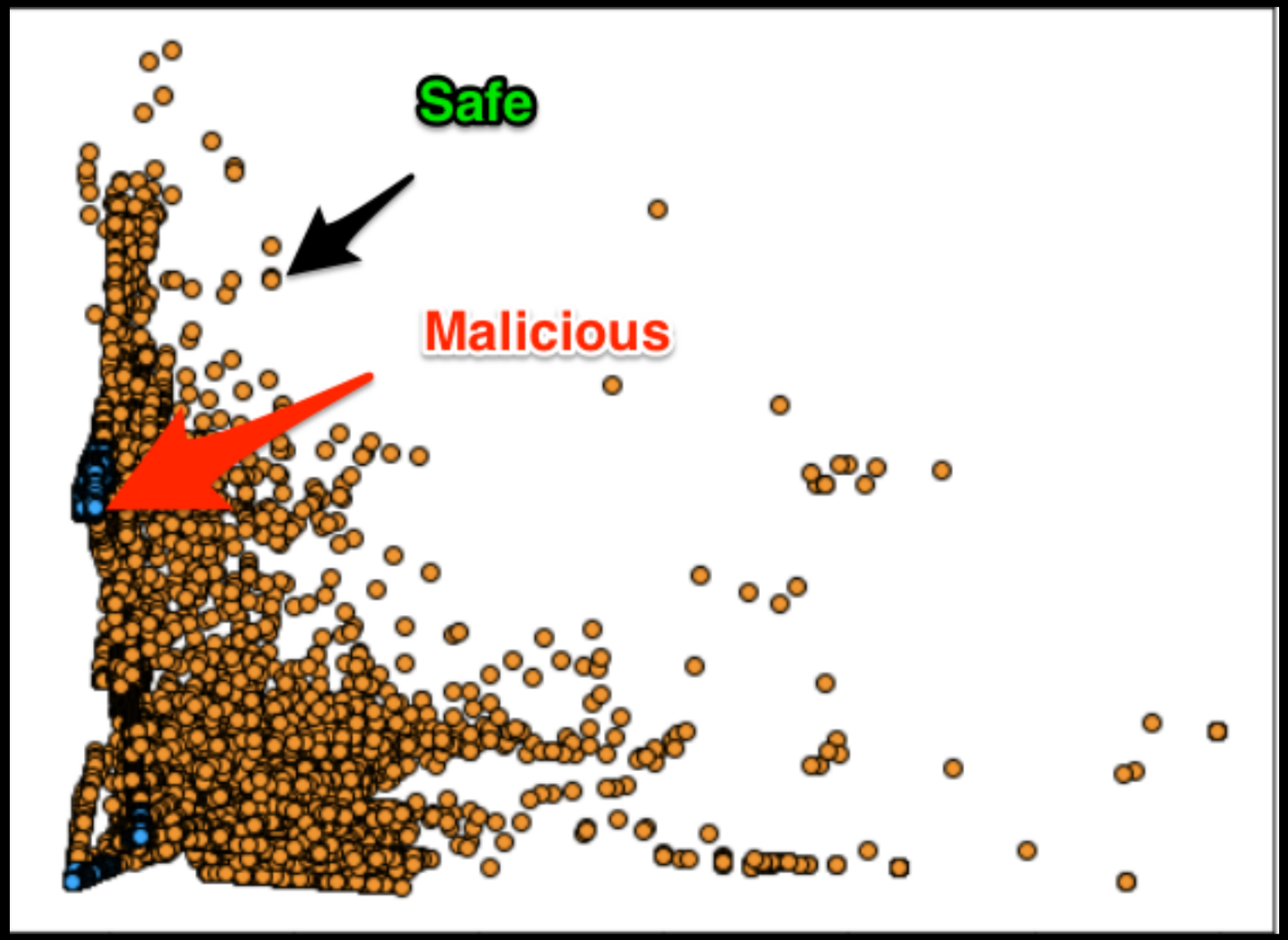
Anomaly Detection
Using
Gaussian Mixture Model

Anomaly Detection on training data



	Detection
Malicious	97.83%
Safe	81.48%

Anomaly Detection on validation data



	Detection
Malicious	95.9%
Safe	71.49%

Supervised Machine Learning

K-Fold Classification Models Accuracy Train/Test

	1	2	3
Logistic Regression L1	99.83%	100%	99.53%
Logistic Regression L2	99.82%	100%	99.54%
Random Forest	99.86%	100%	87.34%
XGBoost	99.86%	100%	94.13%

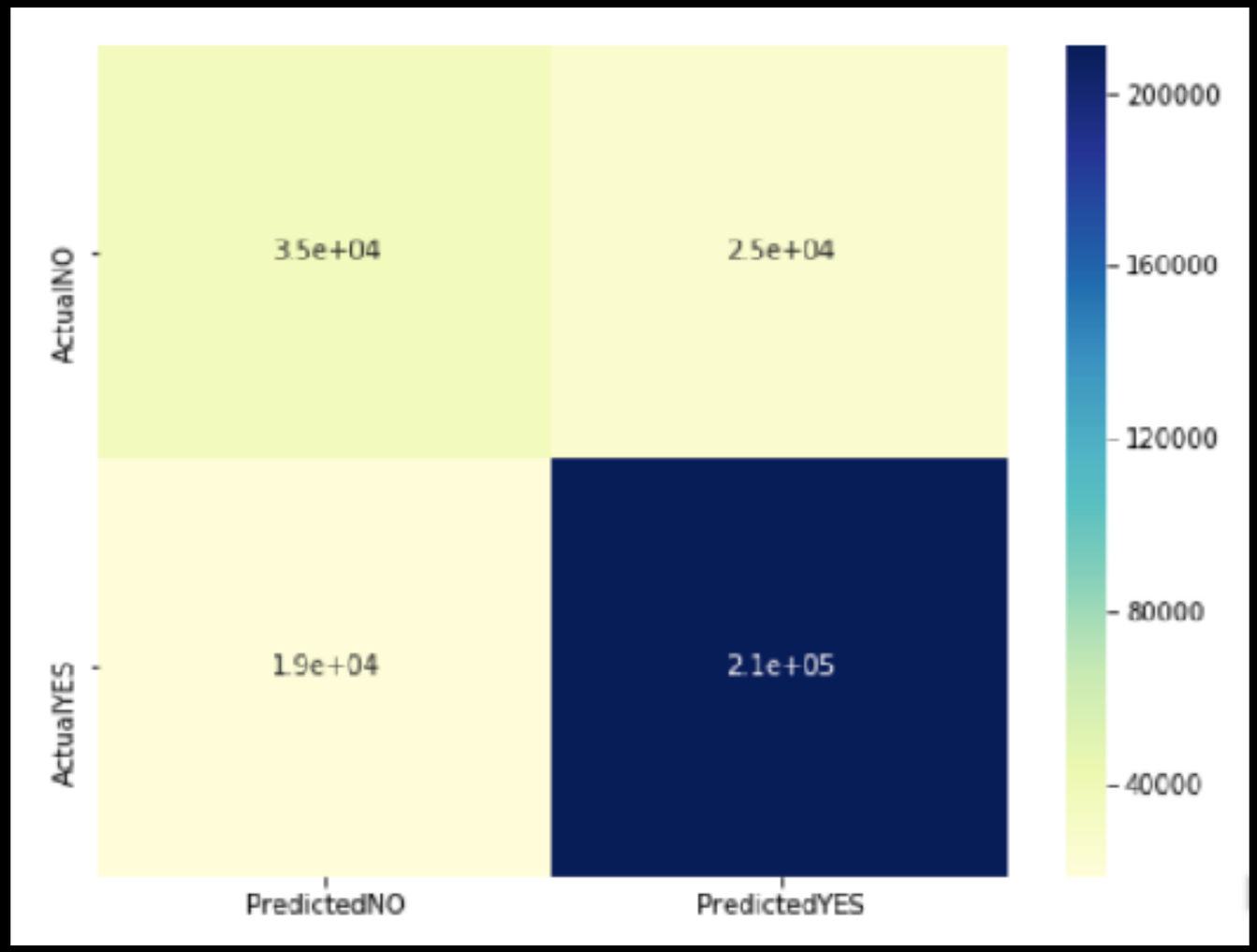
Model Building

- Logistic regression with L1 regularization has almost same score as L2 regularization. Will discard L1 regularization
- There could possibly some overfitting in all the models
- Random forest has lowest accuracy of 87.34% in 1 fold
- It is hard to choose any single model. Will test it against unseen validation data.
- All model except for L1 Logistic Regression will be persisted (pickle)

Validation data

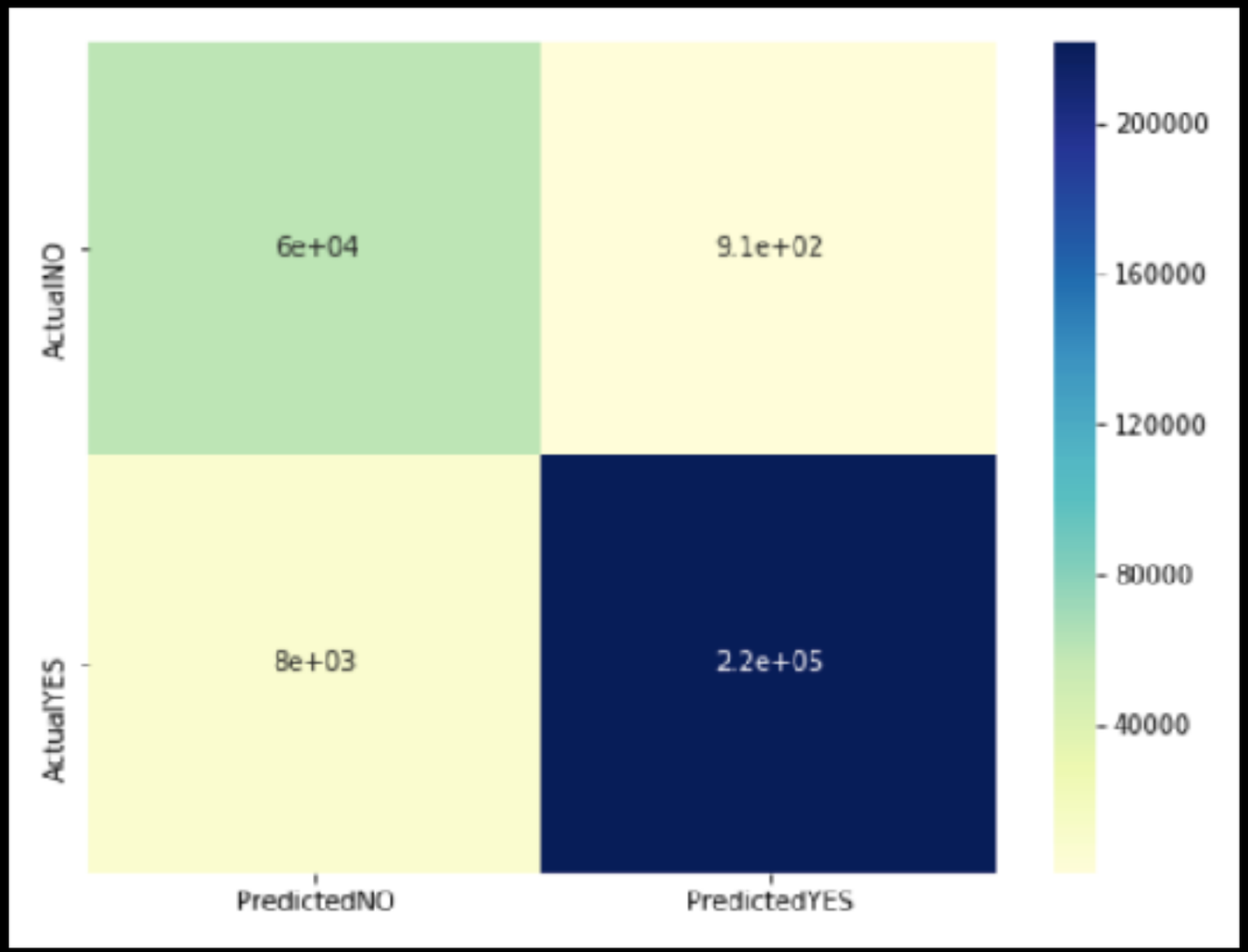
- Validation data is completely different data set
- Validation data has additional variation of Denial of Service connection not seen during training and testing
- The data set was never used either entirely or partially during model training
- Validation data set was ingested only after model was trained and persisted (pickled)

Confusion Matrix Logistic Regression



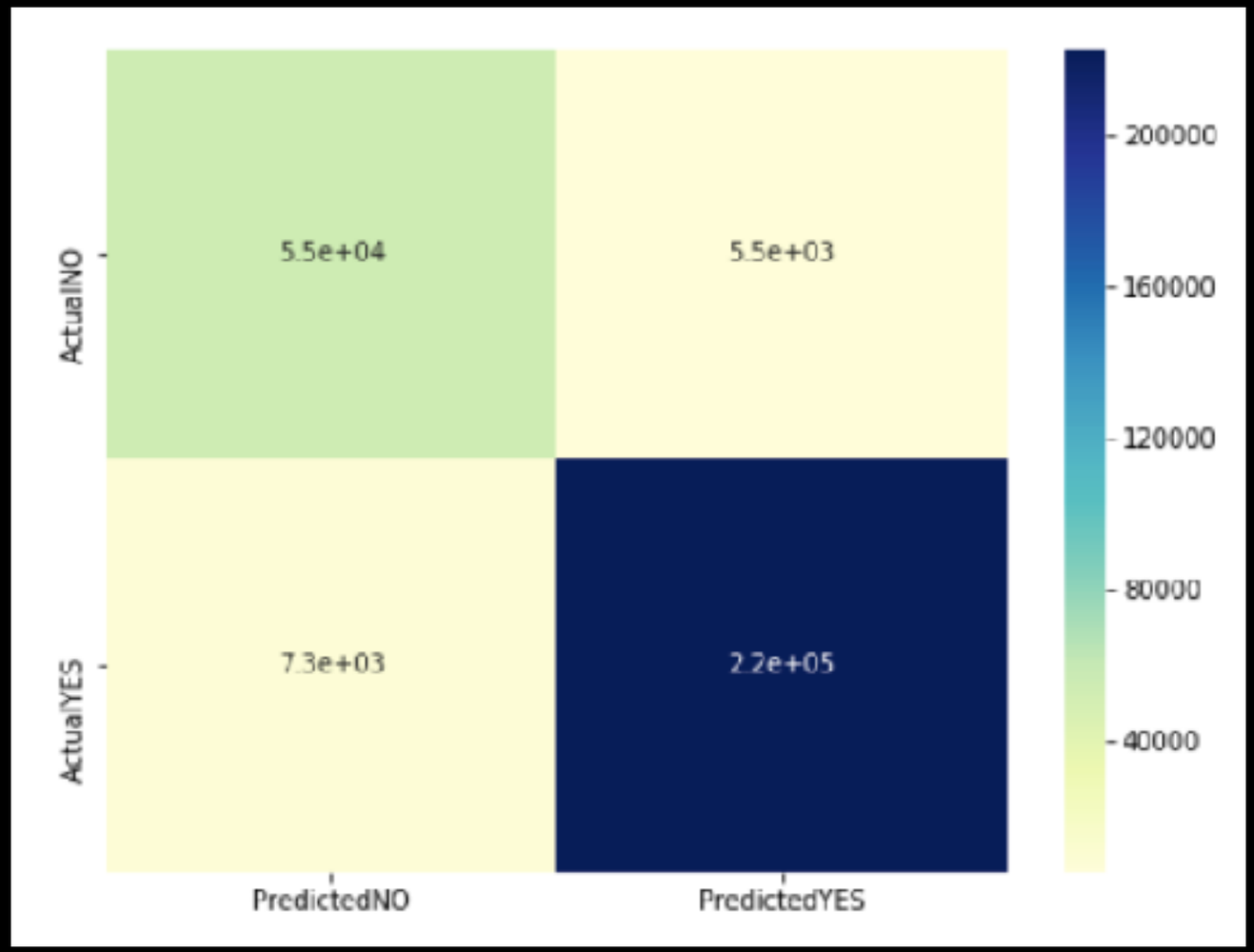
	PredictedNO	PredictedYES
ActualNO	35343	25250
ActualYES	18631	211224

Confusion Matrix Random Forest



	PredictedNO	PredictedYES
ActualNO	59686	907
ActualYES	7973	221882

Confusion Matrix XGBoost

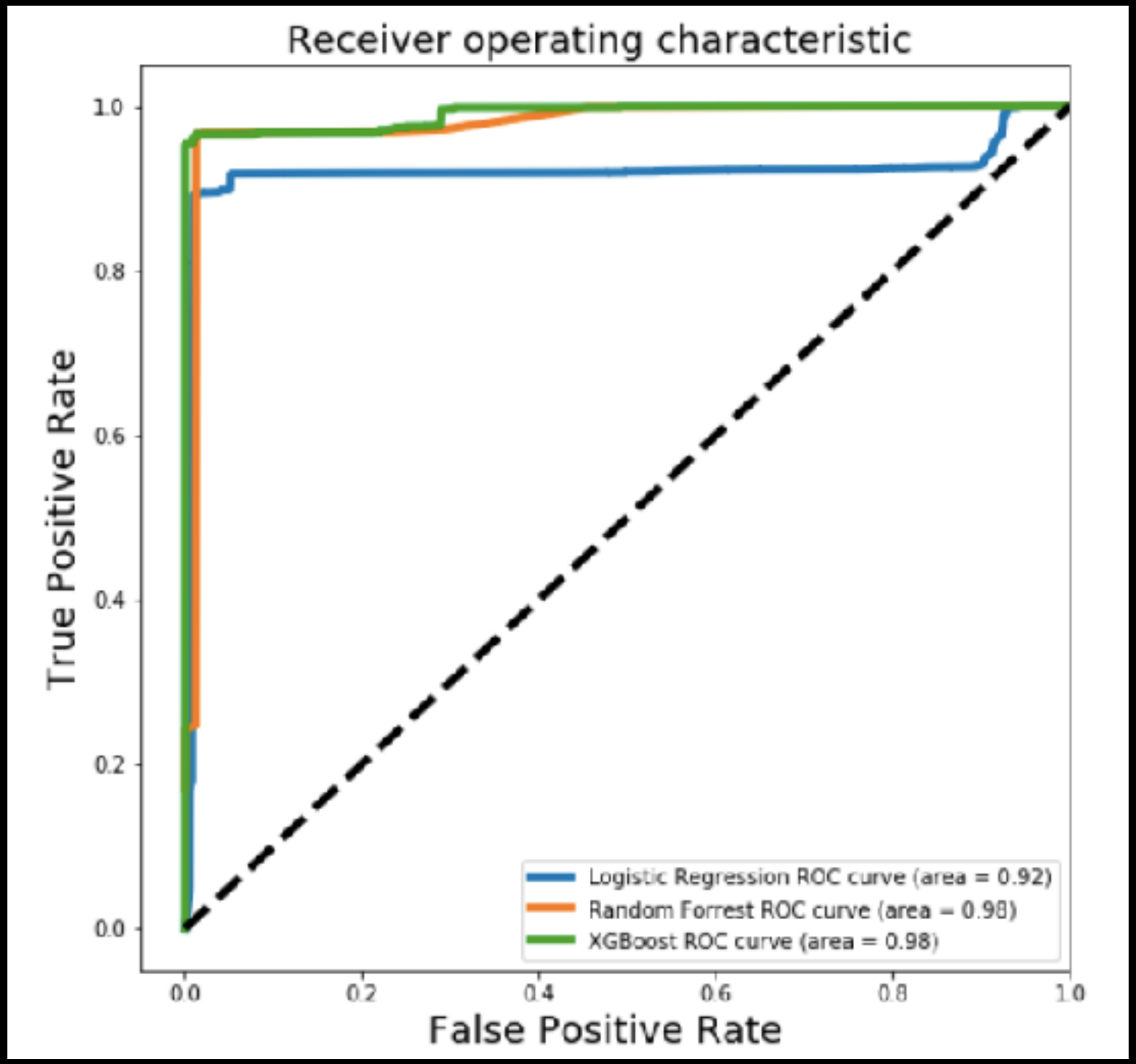


	PredictedNO	PredictedYES
ActualNO	55073	5520
ActualYES	7305	222550

Classification Report

	Precision	Recall	F1
Logistic Regression	84%	85%	85%
Random Forest	97%	97%	97%
XGBoost	96%	96%	96%

Receiver Operating Characteristics



Analysis of ROC

- With zero increase in false positive XGBoost true positive increased to 0.95 then with approximately 0.01 increase in false positive the true positive increased by 0.01. After gaining 0.3 false positive our XGBoost model was able to detect all malicious connections. We can say threshold for detecting all malicious connection is at least 0.7 probability for XGBoost prediction

- With zero increase in false positive Random Forest true positive increased to 0.25 then with slight increase in false positive the true positive increased to 0.96. After gaining 0.3 false positive our Random Forest model was able to gradually detect all malicious connections at around 0.5. We can say threshold for detecting all malicious connection is at least 0.5 probability

- With zero increase in false positive Logistic Regression true positive increased to 0.9 then with slight increase in false positive around 0.05 the true positive increased to around 0.91. Beyond this true positive remain constant. We can say threshold for detecting 0.92 true positive Logistic Regression has at least 0.9 probability.

Questions