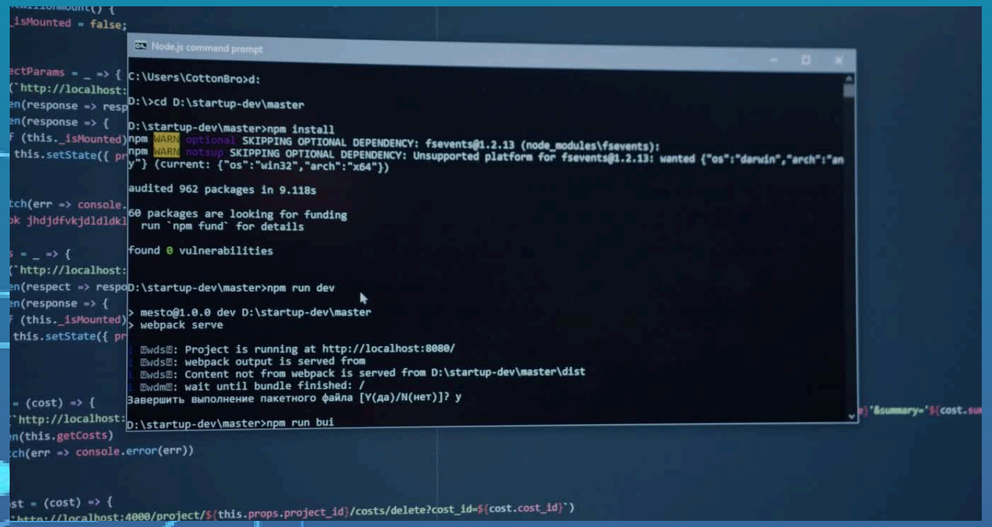
NETWORK INTRUSION DETECTION

Piotr Bonar, Maddox Hurlbert, Sonia Serra
Universitat Autònoma de Barcelona
(group 20)

MOTIVATION

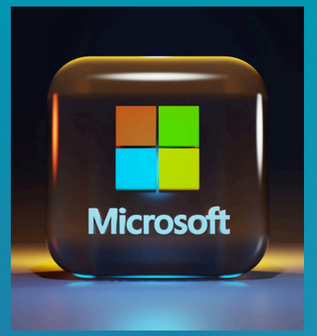
When networks fail… Why detection matters?



October 21th, 2016

Dyn DDoS Attack

Twitter, Netflix, and Reddit



July 19th, 2024

Microsoft's fall

IT Outage for 8.5 mill devices

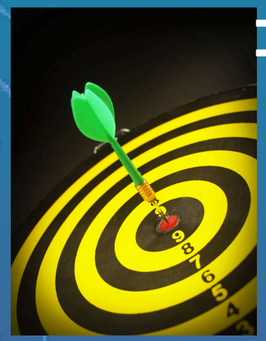
(Including Sonia)

December 9th, 2020

European Medicines Agency

File Edit View Help

INTRODUCTION



- Network intrusions pose **significant threats**
- Automated detections using ML offers **scalable solutions**

File Edit View Help

Traditionally, detection methods are **rule based and reactive**

Our Goal: Prepare and Develop a network intrusion detection system using the UNSW-NB15 dataset, so as to prepare for the attacks

DATASET FEATURES

UNSW THE UNIVERSITY OF NEW SOUTH WALES

ACSC Australian Cyber Security Centre

UNSW-NB15 DATASET

DEVELOPED BY THE ACSC

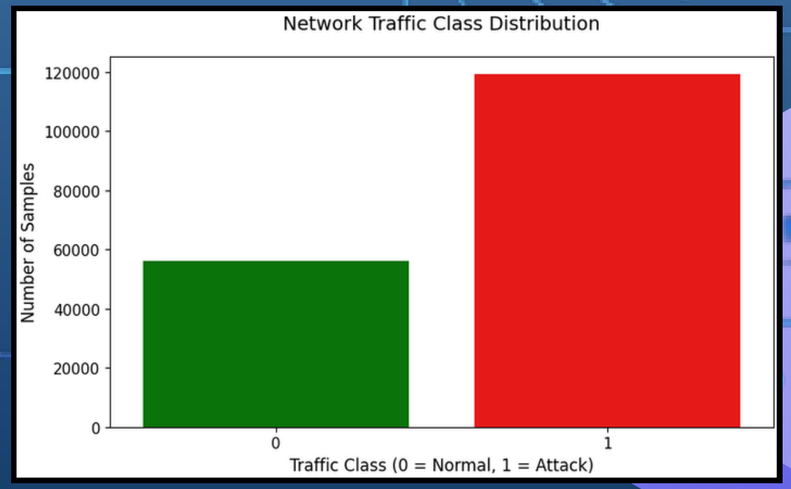
LABELED NORMAL AND MALICIOUS TRAFFIC PATTERNS

AVAILABLE IN KAGGLE

- ~250,000 network traffic records (~180,000 in our training set)
- Consisting of normal traffic and 9 different types of attacks
- 45 features include basic, content-based, time-based, and traffic-based features

9 attack categories

Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode, and Worms



RE Search QUESTIONS

- Can **PCA** improve model performance?
- How well do **K-Means** and **KNN** distinguish malicious traffic?
- Can **collaborative filtering** predict threats based on historical patterns?
- What are the **trade-offs** of each analyzed technique?

PREPROCESSING STEP

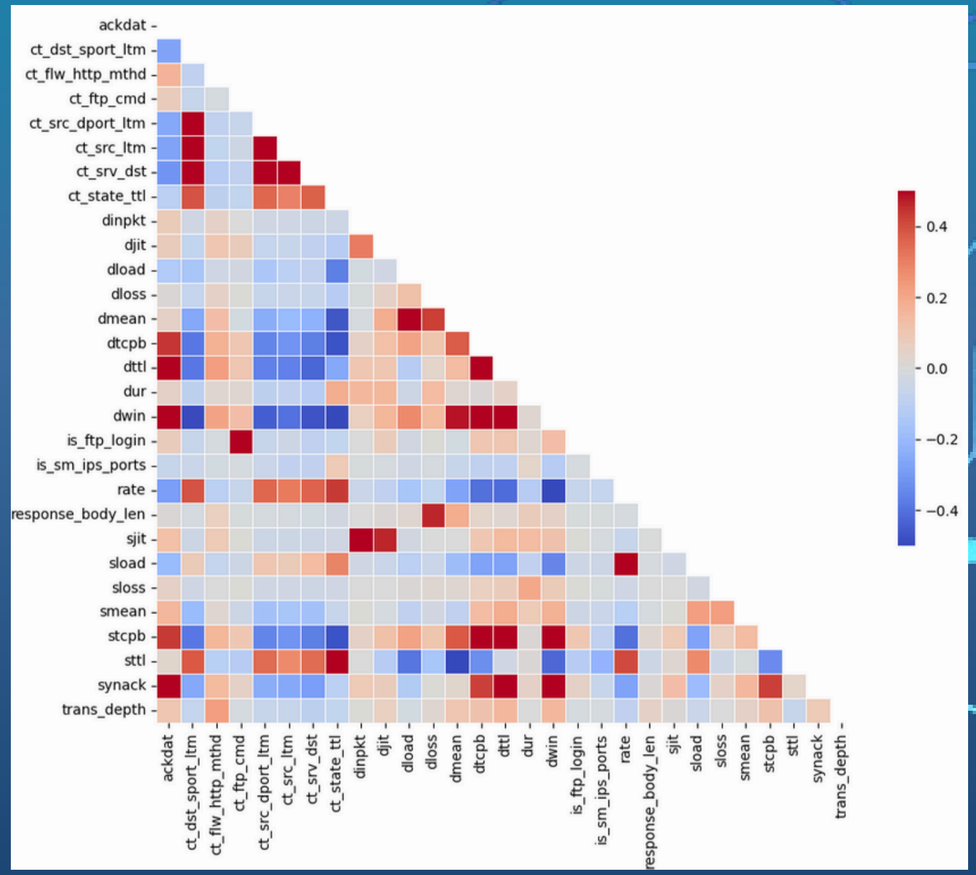
CLEANING

- Remove irrelevant features like attack category, label number, and id number (no insight to training)
- Drop variables that have high correlation with each other
- 29 features remain to be trained and tested

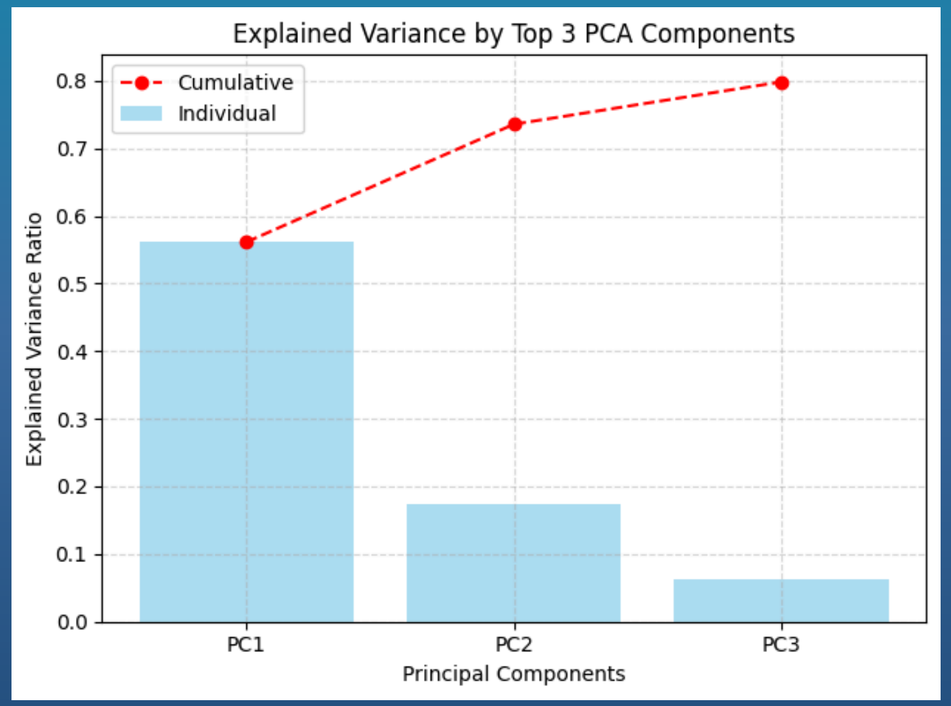
SCALING

Normalize the data to perform PCA

- StandardScaler. Centers data (mean=0, std=1).
- **MinMaxScaler**. Scales to [0, 1] range



PRINCIPAL COMPONENT ANALYSIS

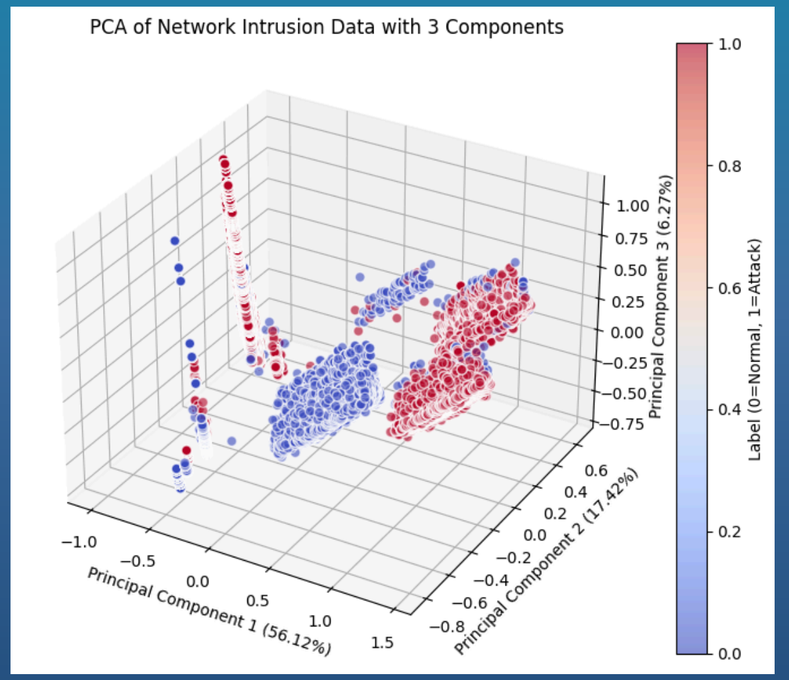


USED:

min-max scaled matrix
due to skewness in the data

CHOICE:

3 feature components
over initial 2

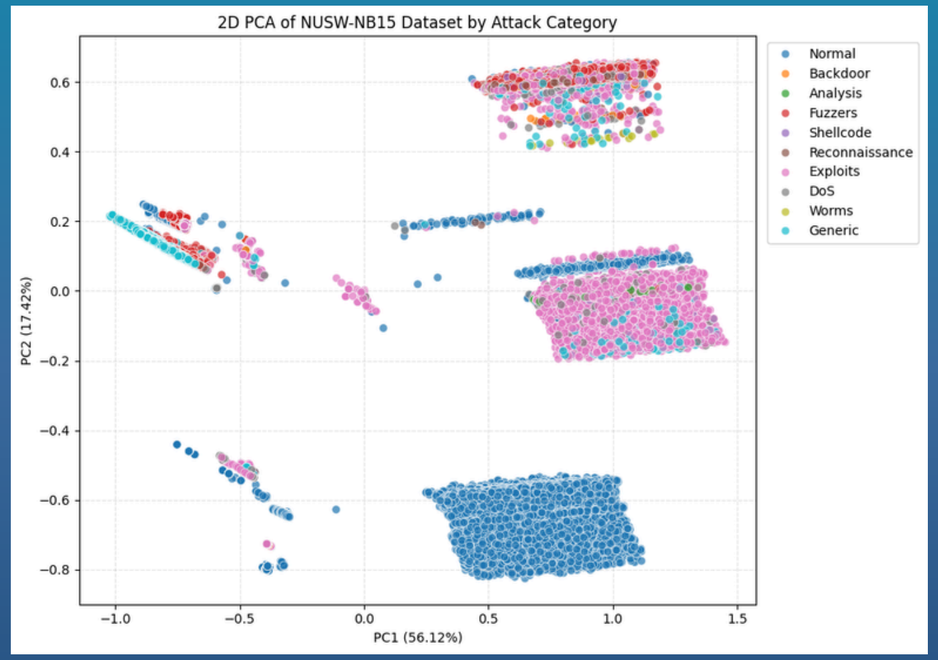


Numerical columns reduced: from 39 to 29

PCAs consist of most deterministic variables

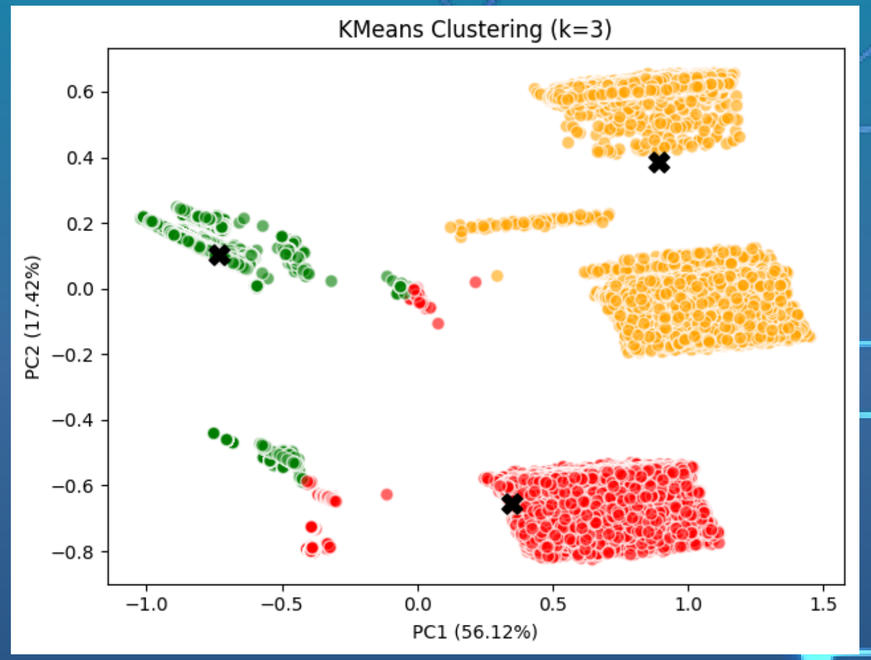
Categorical variables (strings): *proto, service* and *state*

SUPERVISED LEARNING ALGORITHM



Data in PCA shows clear groups between attakcs and not

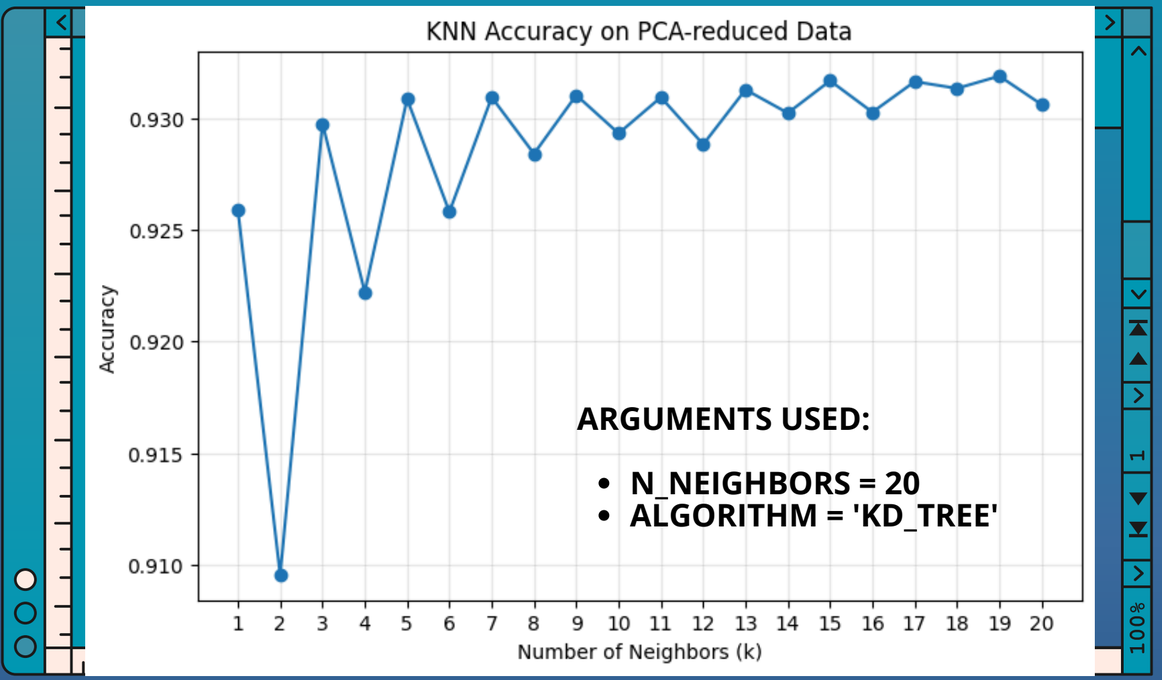
K-MEANS CLUSTERING

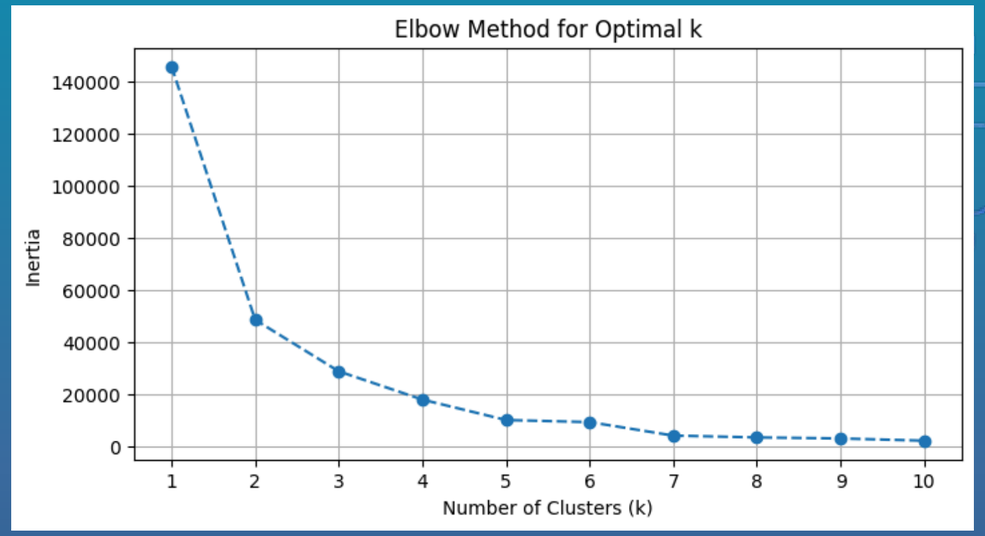


K-means does not improve on PCA separation, but rather just separates normal traffic from attacks

CLASSIFYING

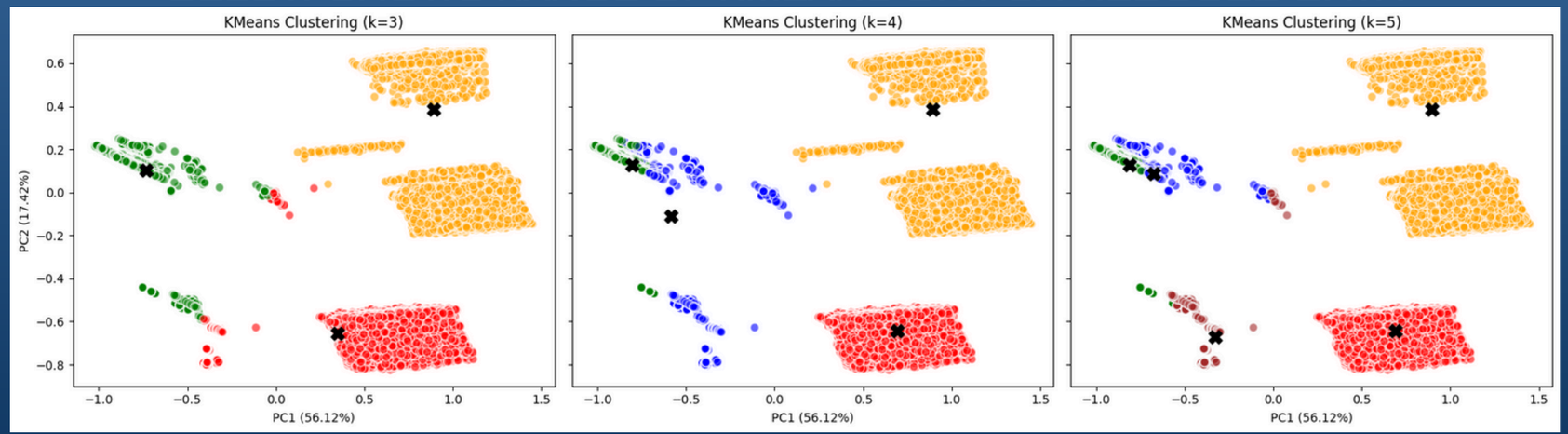
K-NEAREST NEIGHBORS





ELBOW METHOD INDICATES K ~ [3,5], LIKELY OPTIMAL CLUSTER RANGE

BELOW SHOWS RANGE OF KMEANS ON PCA-REDUCED DATA

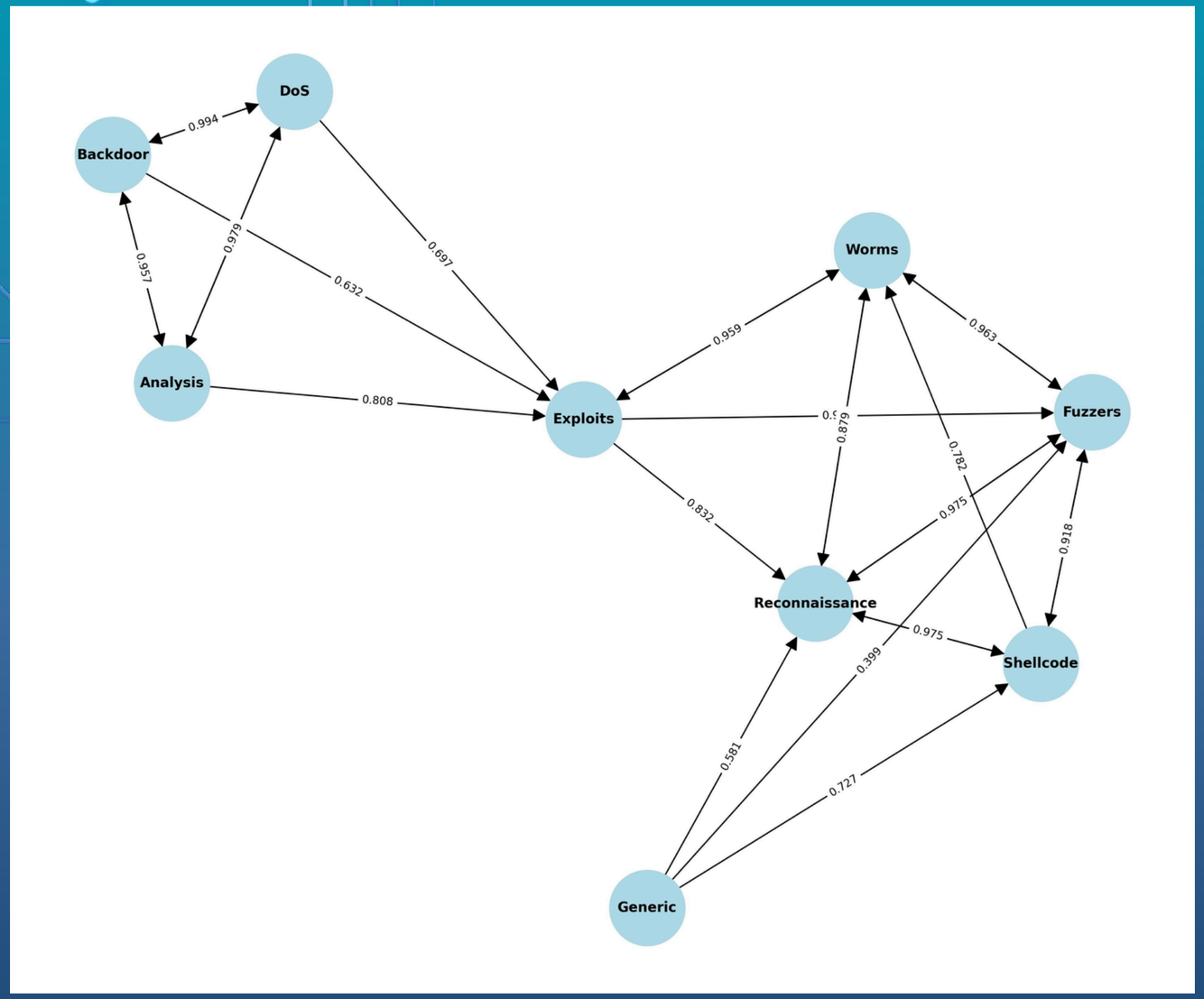


CREATING A RECOMMENDATION SYSTEM

Use of **collaborative filtering** to predict which category of attack a given user is likely to experience

USER-BASED *v.s. ITEM-BASED*

take **ID** column + **attack cat** column



attack similarity

category distribution

attack_cat	count	percentage
Normal	56000	31.94
Generic	40000	22.81
Exploits	33393	19.04
Fuzzers	18184	10.37
DoS	12264	6.99
Reconnaissance	10491	5.98
Analysis	2000	1.14
Backdoor	1746	1.00
Shellcode	1133	0.65
Worms	130	0.07

9 ATTACK TYPES

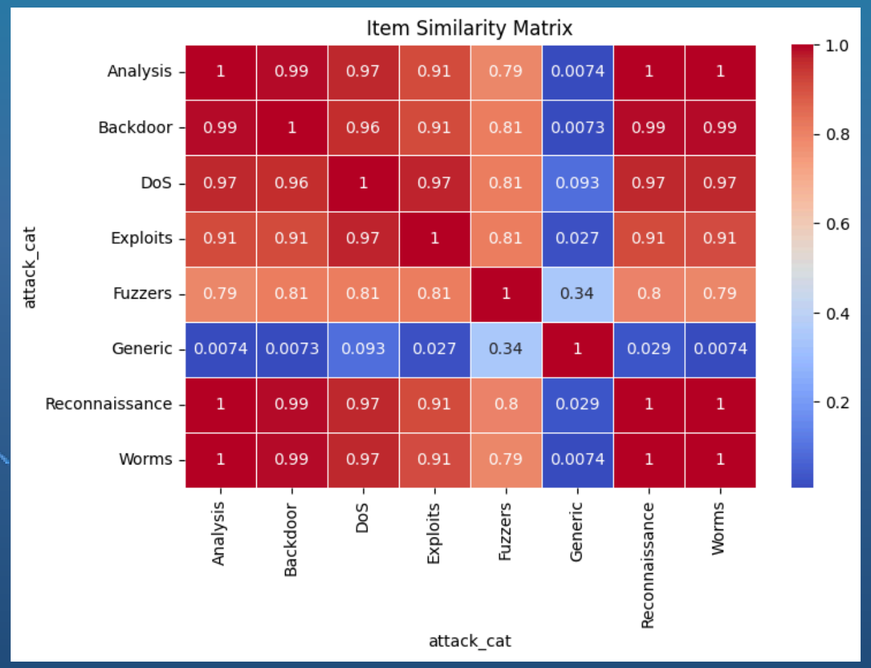
TOTAL: 175341

label column

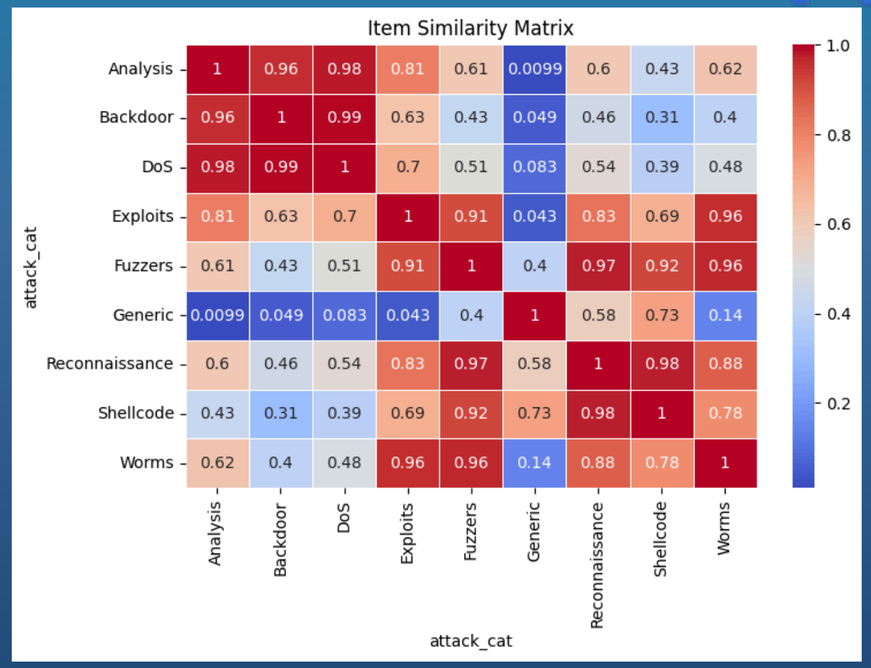
NOT ATTACKED (0) OR ATTACKED (1)

‘Similar users experience similar attack patterns’

Cosine similarity makes difference in the prediction accuracy in **multi-class classification** (attack category).



based on service usage patterns



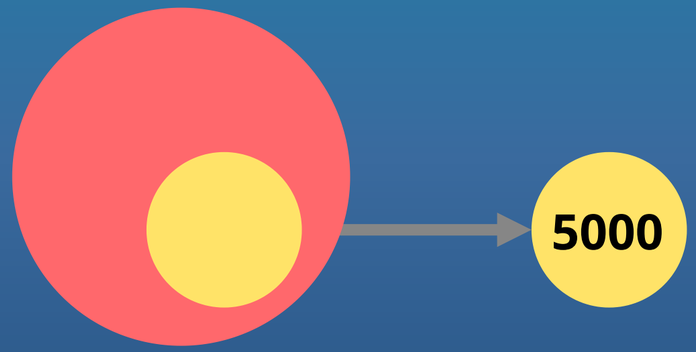
based on transaction protocols

STRATIFIED SAMPLING

LABELS: NORMAL (=0) AND MALICIOUS TRAFFIC (=1)

TWO STRATAS BASED ON SHARED CHARACTERISTICS

175341 X 175341
(matrix size)



RANDOM SAMPLE
OF **5000** USERS

NORMAL TRAFFIC

Consider the **worst** scenario

'UNKNOWN' USERS

Label 0

MALICIOUS TRAFFIC

ATTACKED KNOWN USERS

Label 1

Users with no attack (0=Normal) are the **targets** for the later prediction

SAMPLE SIMILARITY MATRIX

Sampled user similarity matrix shape: (5000, 5000)

id	15483	133350	80486	29973	18340	170501	165831	55216	37757	22161	...	93512	154465	146874	73134	97379	57479	145823	9277	1091	6986
id																					
15483	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
133350	0.0	1.0	0.0	0.0	0.0	1.0	1.0	0.0	0.0	0.0	...	0.0	1.0	1.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0
80486	0.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	1.0	1.0	0.0	0.0	0.0	0.0	0.0
29973	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
18340	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
57479	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0
145823	0.0	1.0	0.0	0.0	0.0	1.0	1.0	0.0	0.0	0.0	...	0.0	1.0	1.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0
9277	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
1091	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
6986	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

5000 rows × 5000 columns

NEAREST NEIGHBORS FOR PREDICTING CYBER THREADS

1. Loop through target users and find their nearest neighbors
2. Compare them to known users to infer likely attack types
3. Recommend the most common attack type from those neighbors

Random selected target ID

target_id = random.choice(similarity_df.index)

closest neighbors RESULTS

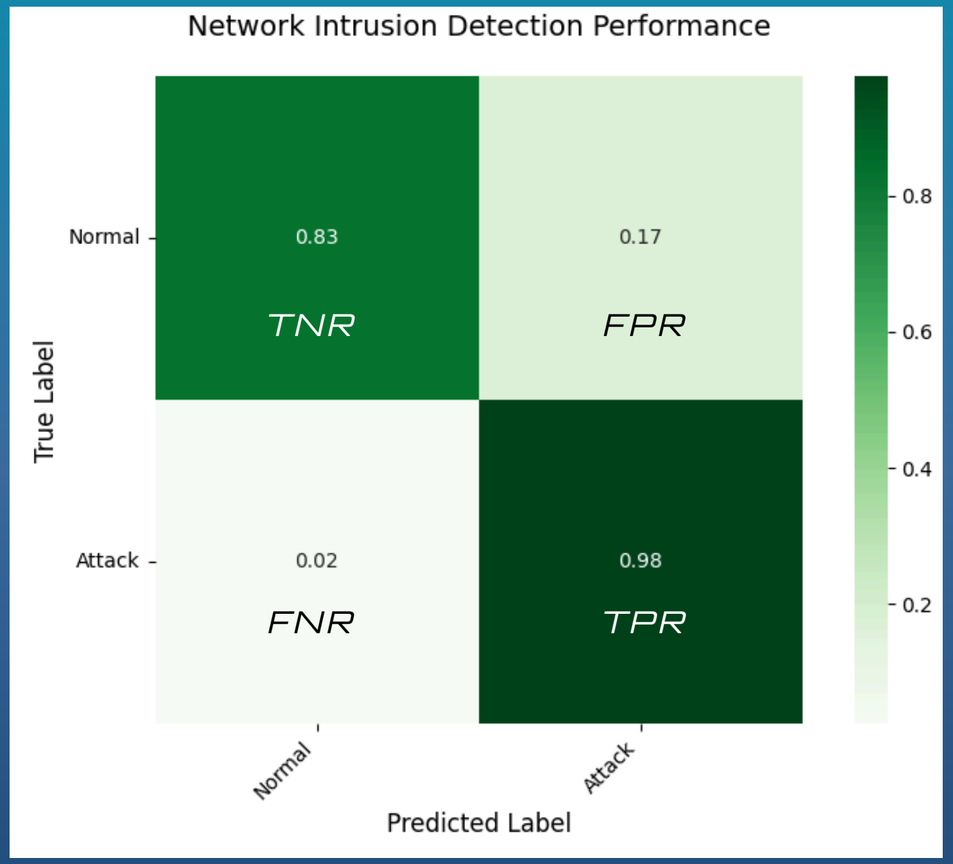
Attack category counts:

attack_cat	0
Analysis	0
Backdoor	0
DoS	0
Exploits	0
Fuzzers	0
Generic	0
Reconnaissance	0
Shellcode	5
Worms	0

	Target ID	Top Neighbors	Attack Category Counts	Predicted Attack Category
0	118838	[83977, 52190, 91316, 82963, 145861]	[0, 0, 0, 0, 0, 0, 0, 5, 0]	Shellcode

Prediction for that ID

RESULTS. PERFORMANCE METRICS.



CONFUSION MATRIX

NORMAL AS - CLASS, ATTACK AS + CLASS

FP (FALSE ALARMS) > FN (MISSED ATTACKS)

CLASSIFICATION REPORT AFTER PREDICTIONS

	Normal	Attack	accuracy	macro avg	weighted avg
precision	0.943791	0.925461	0.930612	0.934626	0.931305
recall	0.831922	0.976808	0.930612	0.904365	0.930612
f1-score	0.884333	0.950441	0.930612	0.917387	0.929363
support	16772.000000	35831.000000	0.930612	52603.000000	52603.000000

 # actual occurrences

TNR: *~83% of the 16,772 normal users were correctly identified*

High TPR: *~98% of actual attacks were correctly detected*

harmonic mean *The model balances precision/recall better for attacks than normal traffic (F1 score = 2pr / (p+r))*

CONCLUSION. PROS AND CONS.

pca>

- Reduces noise, speeds up training (~80% variance in 2-3 PCs)
- Loses interpretability

<k-means

- Groups similar traffic (unsupervised)
- Needs manual labeling, sensitive to noise

knn>

- 93% accuracy, easy to implement
- Slow on large data, imbalance-sensitive

<CF

- Predicts co-occurring attacks (e.g., Exploits after DoS)
- Needs rich history, as it struggles with new threats

Best for UNSW-NB15: PCA + KNN (balance of speed & accuracy)

THANK YOU FOR YOUR ATTENTION