

DATS6301 - Data Mining - Final Project

Will Arliss

Network Intrusion Detection

- Network intrusion: “Any unauthorized activity on a digital network” meant enable malicious activity
- Flow: “Series of communications between two router endpoints bounded by the opening and closing of a session”
- We are looking at the telemetry of flows as they pass from one endpoint to another to try to identify intrusions or attacks on the network



Exploration

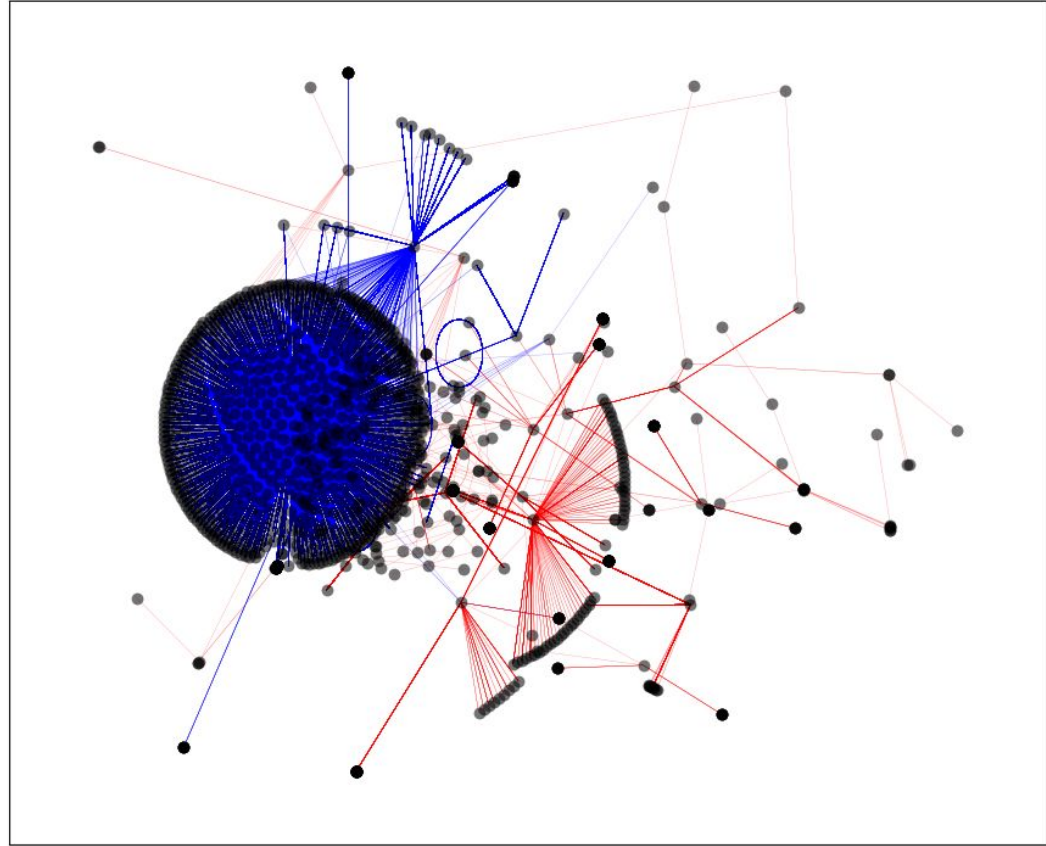
LUFlow Network Intrusion Detection Data Set

- Network flow telemetry measured at Lancaster University
- Spans June 2020 to February 2021
- Sample ~1.5M from 2020 for training and ~300K from 2021 for testing/holdout
- ~31% “malicious” and ~69% “benign” (after dropping “outlier” labels)
- Includes 16 features, such source, destination, protocol, bytes in/out, packets in/out, entropy, time, duration, rate
 - Used 12 features, some engineered, some raw
- No missing values present

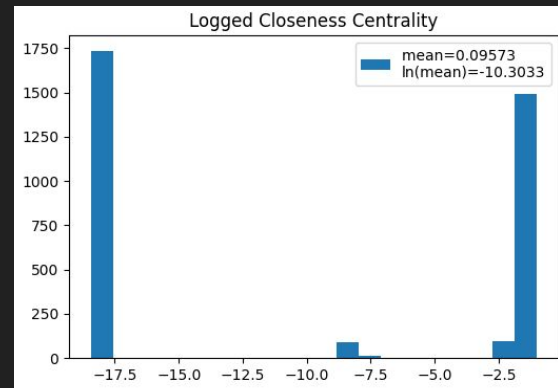
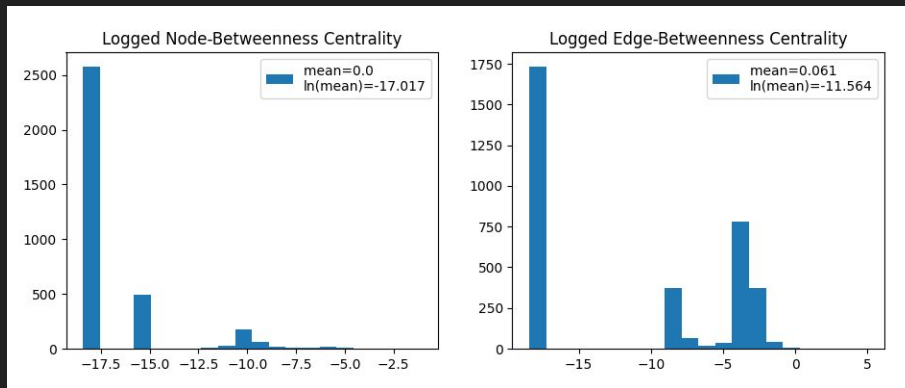
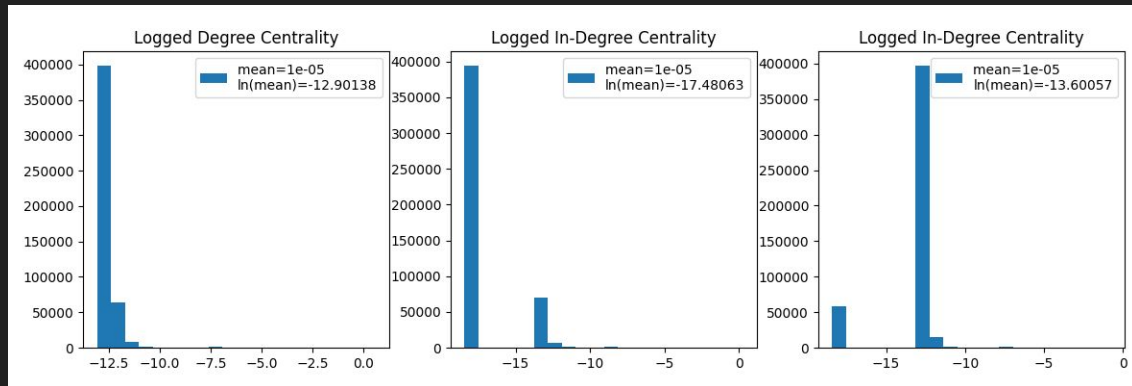
Exploration

- nodes: 475,033
- edges: 1,441,018
- connections: 479,121
- density: $\sim 6.4E-6$
- avg degree centrality: $1E-5$
- avg closeness: 0.096
- avg betweenness: $1E-5$

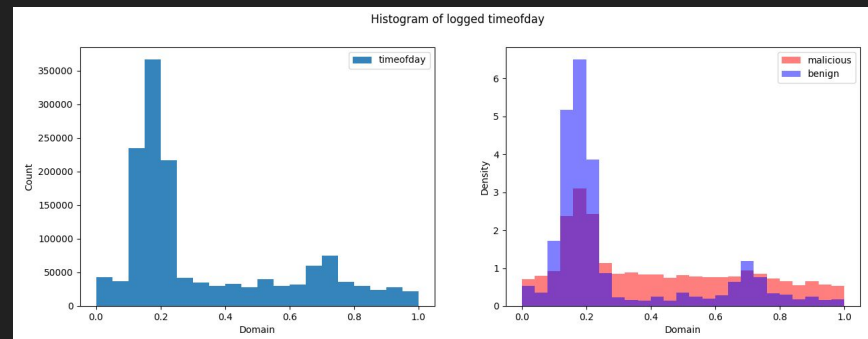
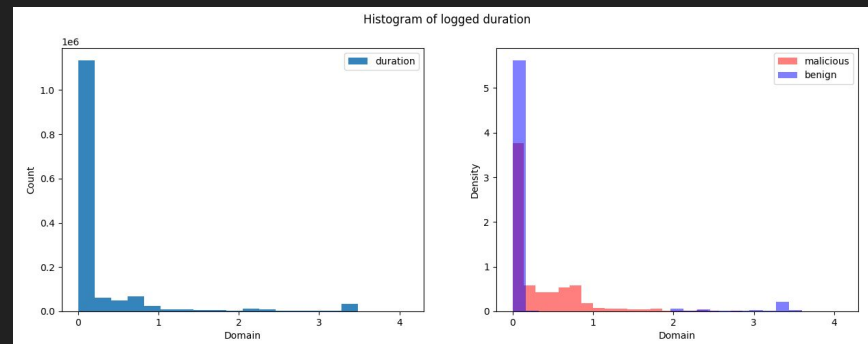
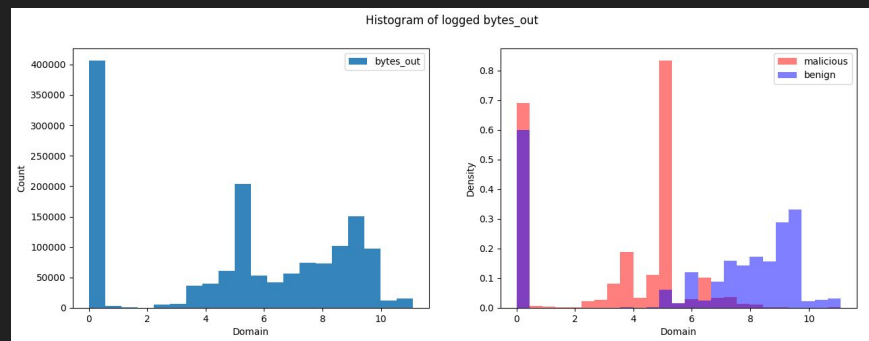
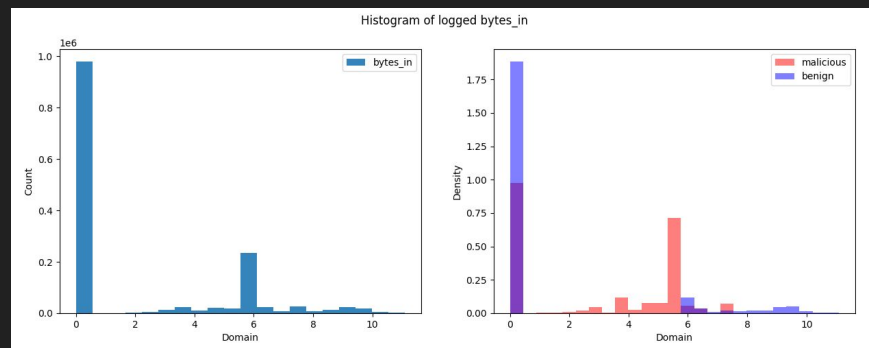
Malicious (red) and Benign (blue) Network Flows



Exploration



Exploration



Preprocessing steps

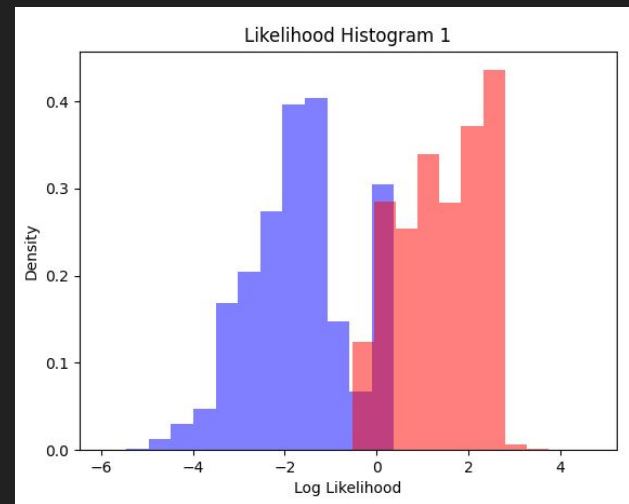
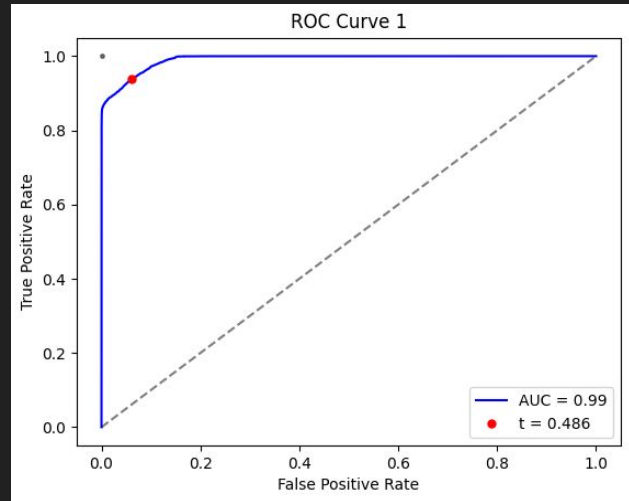
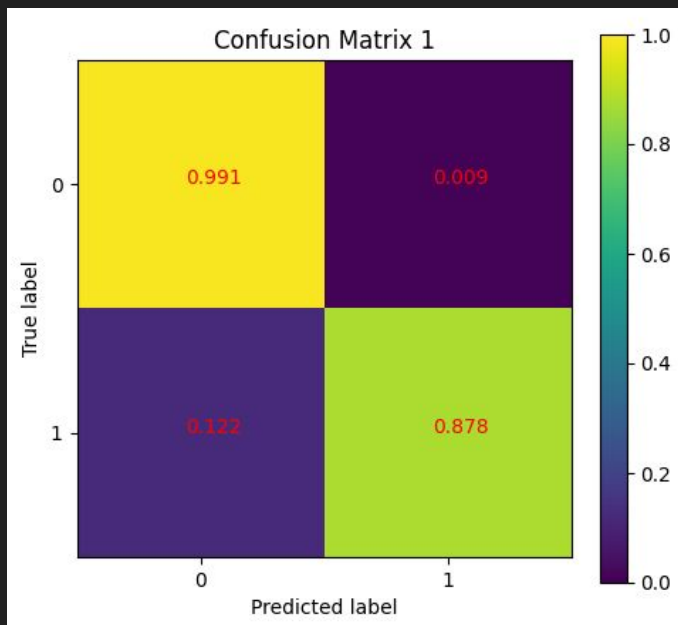
- Create new features from the “start” timestamp to represent “day-of-week” and “time-of-day” of the flow
- Create new “src” and “dst” feature as a concatenation of IP and port
 - Used for building graph, not initial modeling
- One-hot-encode the “protocol” and “day-of-week” categorical features
- Create new “duration” computed from “start” and “end” timestamps
- After some exploration, select subset of original variables for modeling
- Remove observations where label is “outlier”

Initial modeling

- Three kinds of decision tree ensembles:
 - Random Forest
 - AdaBoost
 - Gradient Boost (with scaling)
- Tune hyper-parameters using grid-search and cross validation, optimizing for ROC-AUC
- Refit model/hyper-parameters with highest score on all the data
- Test on holdout

Modeling

AdaBoost Decision Trees



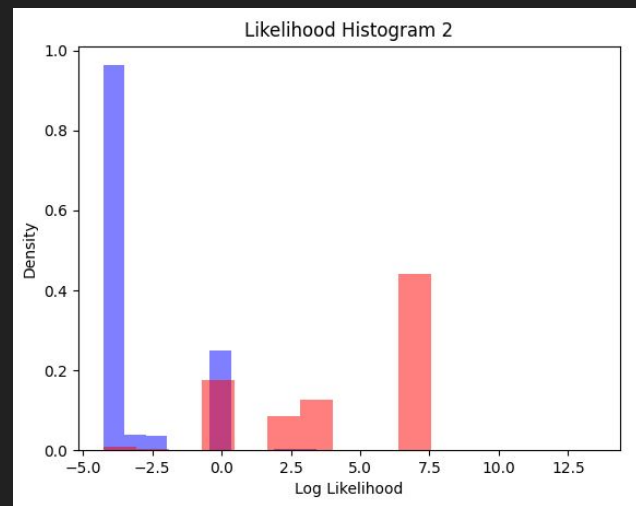
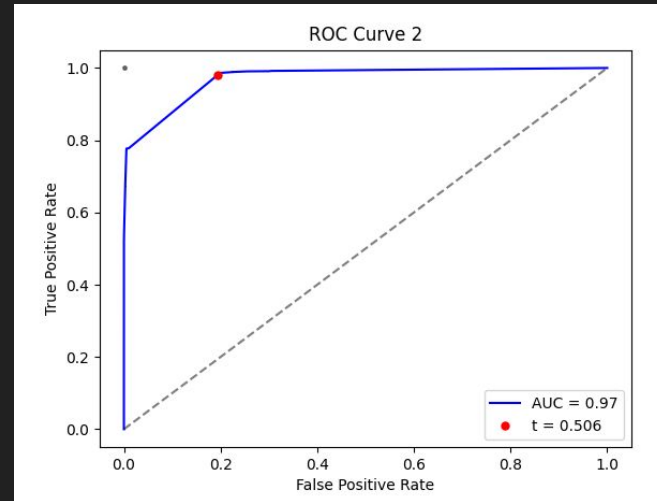
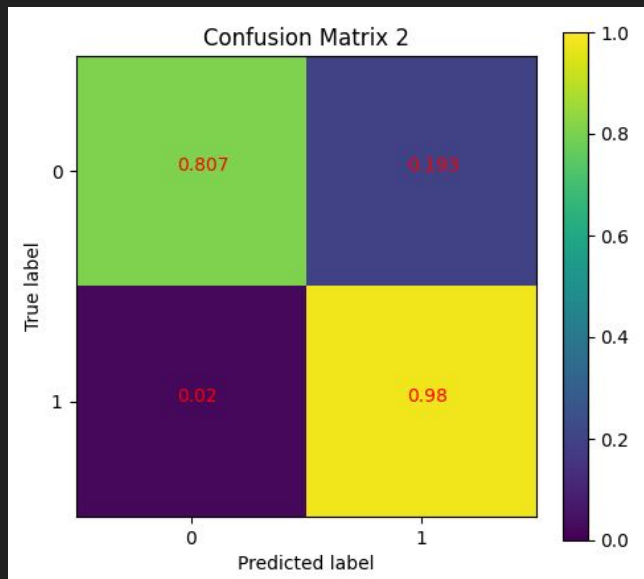
Modeling contd.

- Why was TNR so much higher than TPR
 - Class imbalance (31% +)?
 - Wrong metric?
- Tune hyper-parameters again using grid-search and cross validation, this time optimizing for recall
- Refit model/hyper-parameters with highest score on all the data
- Test on holdout again

Modeling

AdaBoost Decision Trees

- shallower trees
- balanced instance weights
- lower learning rate



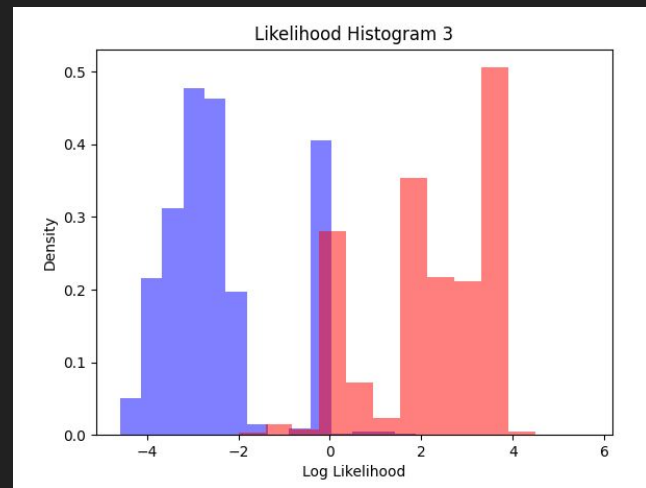
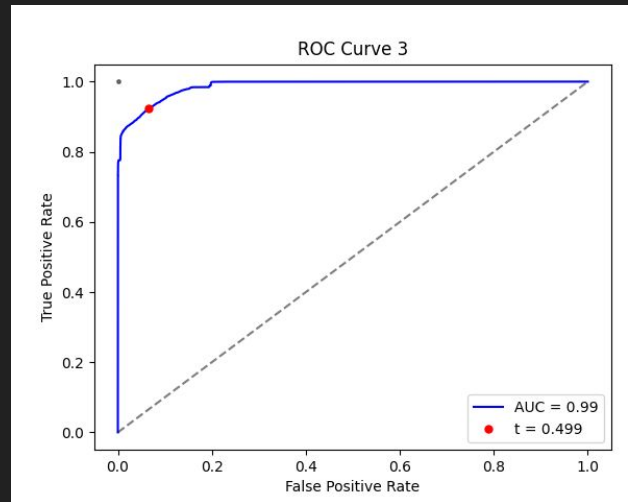
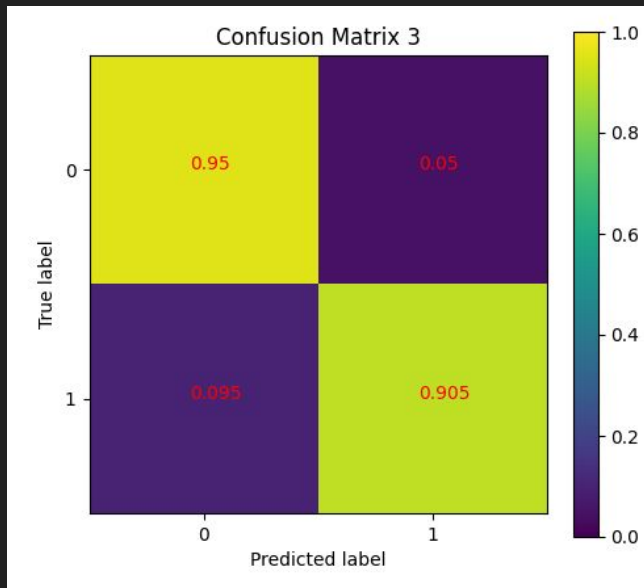
Modeling contd.

- Why was TPR so much higher than TNR this time
 - Class imbalance over-correction?
- Combining them might even out TPR and TNR
- No training needed, just more evaluation

Modeling

Soft Voting Ensemble

- combo of both



Limitations

- Graph structure of data not exploited
- Sample used is only 1% of data available
- Limited domain knowledge

Next Steps

- Node/edge embedding of graph for more model inputs
- Link prediction algorithms
- Graph neural networks
- Scaling