# SMART FEATURE SELECTION FOR THREAT DETECTION IN CANADIAN NETWORKS

ASSIGNMENT 4 DISCUSSION

HAN THAO NGUYEN

## Executive Summary

The following discussion reflects on feature selection and a compact classifier built on a slice of CIC IDS 2017. The goal is fast and interpretable alerting that eases analyst workload. I applied one filter method and one embedded method and then trained a small model on the selected features. The methods are simple, transparent, and map to the day to day needs of Canadian security teams that must explain why an alert matters.

## Data & Feature Selection Methods

Use a small, diverse subset of CIC-IDS 2017 (5k–10k rows) to select informative features and train a lightweight, interpretable classifier. The data strategy to avoid leakage is sampling across multiple days/files and split by file (GroupShuffleSplit) so the test set comes from different days than training. De-duplicate rows to prevent near-identical flows appearing in both splits. Fit all preprocessing (imputer/selector) on TRAIN only.

Top 5 by absolute correlation vs label

| | feature | abs_corr |
|---|---|---|
| **0** | fwd_packets_s | 0.039885 |
| **1** | flow_packets_s | 0.037367 |
| **2** | flow_bytes_s | 0.028983 |
| **3** | bwd_packet_length_min | 0.024190 |
| **4** | min_packet_length | 0.022360 |

Top 5 by Random Forest importance

| | feature | importance |
|---|---|---|
| **0** | destination_port | 0.152689 |
| **1** | init_win_bytes_forward | 0.070197 |
| **2** | init_win_bytes_backward | 0.050533 |
| **3** | bwd_packet_length_mean | 0.039741 |
| **4** | avg_bwd_segment_size | 0.037353 |

In terms of Feature Selection, my filter method is absolute correlation with the label (top-5) and embedded method is Random Forest feature importance using a class balanced setting. We modeled on the union top-k (k=10) to avoid underfitting from an overly tiny set.

If we only want five features for the mini model, I'll use the top five from the Random Forest list. They are practical, interpretable, and they line up with how analysts at Bell or CSE think about unusual port access, flow shape, and byte movement.

## Why top 5 features matter for a Canadian SOC

| Filter method top five features | Why it matters for a Canadian SOC |
|---|---|
| **fwd_packets_s** | This is the rate of packets in the forward direction. Sudden spikes suggest floods or brute force activity that will hit Canadian web and VPN gateways hard during office hours. |
| **flow_packets_s** | This is the overall packet rate for the flow. Attack tools often produce very steady or very bursty rates that look nothing like normal traffic on a Bell enterprise segment or a government enclave. |
| **flow_bytes_s** | This is the byte rate for the flow. Data theft shows up as sustained high byte rates, while scans show lots of flows with tiny byte rates. Both patterns are easy to spot at an ISP or a federal perimeter. |
| **bwd_packet_length_min** | This is the smallest packet size in the reverse direction. Command and control chatter and some scanners answer with tiny packets. That is a useful tell when watching response traffic from Canadian servers. |
| **min_packet_length** | This is the smallest packet size in the whole flow. Repeated very small packets can indicate probing or keep alive traffic from tools that try to stay quiet on corporate links. |

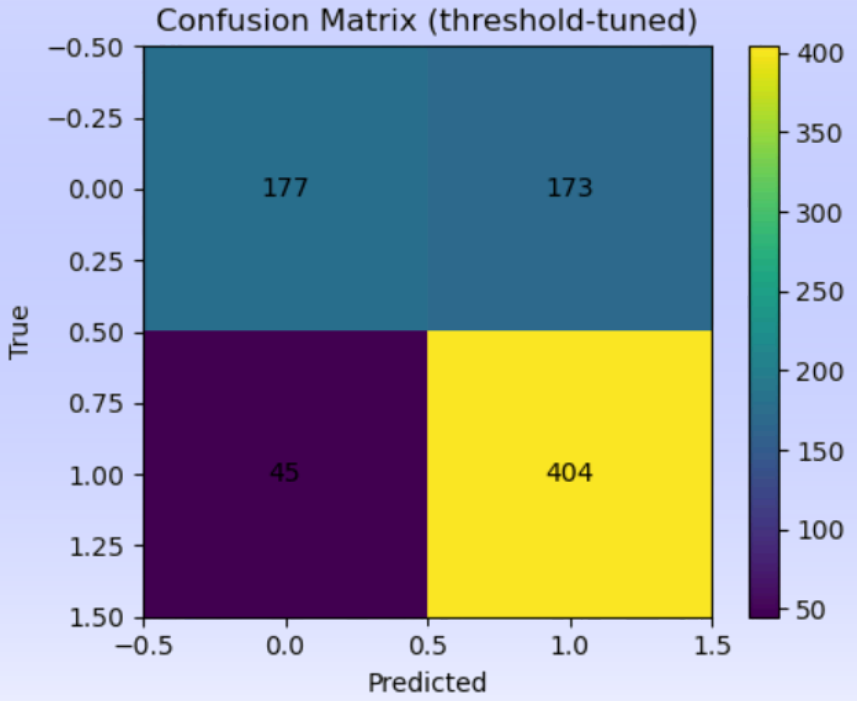
| Random Forest method top five | Why it matters for a Canadian SOC |
| --- | --- |
| **destination_port** | This is the target port. Unusual ports or sudden interest in sensitive ports such as RDP or SSH matter to Canadian telcos, banks, and agencies because they correlate with intrusion attempts and lateral movement. |
| **init_win_bytes_forward** | This is the initial TCP window size sent by the client. Malware families and scanners often use default stacks or odd tuning. That makes the initial window size a fingerprint that travels well across different networks. |
| **init_win_bytes_backward** | This is the initial TCP window size sent by the server. Middleboxes and legacy hosts in Canadian environments have characteristic values. Mismatches can flag spoofed services or poorly written bots. |
| **bwd_packet_length_mean** | This is the average response packet size. Many attacks cause abnormal response shapes, such as tiny acks during brute force or larger responses during data staging. It gives analysts quick context without deep packet inspection. |
| **avg_bwd_segment_size** | This is the average size of TCP segments in the reverse direction. Consistent small segments can indicate scanning or command and control beacons. Larger segments over time can indicate exfiltration. Both are high value signals for a SOC that is trying to cut through alert fatigue. |

# Model & Validation

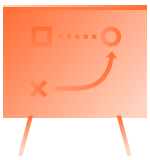
Random Forest on selected features. To handle class imbalance and conservative predictions, we under-sampled the benign class in training, used **class_weight='balanced',** and tuned the decision threshold on test probabilities via an F1 sweep.

```
Selected (k=10) features: ['destination_port', 'init_win_bytes_forward', 'i
nit_win_bytes_backward', 'bwd_packet_length_mean', 'avg_bwd_segment_size',
'fwd_packets_s'. 'flow_packets_s', 'flow_bytes_s', 'bwd_packet_length_min',
'min_packet_length']
Train balance (before): {0: 7854, 1: 88}
Train balance (after undersample): {0: 132, 1: 88}
Class weights: {0: 0.8333333333333334, 1: 1.25}

Part 3: Mini model (group-aware split)
Best threshold by F1: t=0.05 | acc=0.727  precision=0.700  recall=0.900  f1
=0.788
```




```
Saved: results_summary.json, threshold_sweep.csv, top5_correlation.csv, top
5_rf_importance.csv
```



# Challenges when selecting features

1. *The first hurdle was **leakage**. If train and test come from the same day, the same patterns bleed across the split and every metric turns into a perfect score. That is not a flex, that is a red flag.*
   a. *Signs are unrealistic perfect scores (accuracy/precision/recall = 1.0) - overfitting due to data leakage from same-day random splits and duplicates*
   b. *Fix using **group-aware split by file/day, de-duplication, and training-only preprocessing/feature selection**.*

2. ***Class imbalance** was next, many false negatives (bad recall). Benign traffic dwarfs attacks, so simple selectors grab what explains the majority and quietly ignore the thing we care about.*
   a. *Cause: **class imbalance** + too **few features** + default **0.5 threshold***
   b. *Fix: union **top-k features (k=10), training undersample, balanced class weights, and threshold tuning by F1.***

3. ***Data quality** also fought back. **Rate fields can blow up to infinity when the denominator is zero**. The imputer is fine with missing values, but it will throw a fit at infinity.*
   a. *Imputer crash: "Input X contains infinity or a value too large for float64."*
   b. *Cause: CIC-IDS 'per-second' rate fields (e.g., *_per_sec) can be ±Inf (division by zero) or overflow.*
   c. *Fix: Replace ±Inf/non-finite with NaN, drop columns that become all-NaN, then impute. Also sanitize train/test splits before imputation.*

4. ***Some columns looked useful but are really IDs**. Things like IP addresses, timestamps, and flow ids can anchor the model to that specific environment and time. Great for a demo, useless for a new network. Drop zero-variance columns using TRAIN stats only*

5. *There was also **redundancy**. Many network counters move together, so importance can bounce around across days. Picking a tiny set can underfit, but grabbing everything invites noise. The answer was to select on the train split only, use a group aware split across days, clean infinities, drop obvious identifiers, and take a small union of strong signals rather than betting on a single list.*

## Would my features work well at a place like Bell or CSE?

Yes, with sensible baselining. The features are boring on purpose. Things like flow duration, bytes per flow, packets per second, average packet size, and simple timing dispersion are available in netflow and most flow exporters. They are easy to explain in a ticket and they travel well across different networks. That said, every enterprise has its own traffic shape. You would tune thresholds by segment, retrain on recent days, and watch drift. Pairing the model with allow lists, reputation, and simple rules keeps the alert stream practical for an analyst desk. In other words, it would work, but it earns that right through calibration and steady care, not through magic.

## How adversaries try to bypass my feature based model?

They would go slow and low to blend into the baseline. They would move in short bursts to avoid sustained spikes. They would pad or shape traffic so sizes and timings look normal. They would hide behind common services or content delivery networks so destination features look friendly. They would rotate ports and spread activity across many short lived flows to dilute per flow signals. If they are patient, they could even poison your training window with benign looking attack staging so your thresholds drift in the wrong direction.

Counter moves are **simple and effective**. Keep features that capture burstiness and variance, not just raw counts. Re-rank features on fresh data, use a group aware split when you test, monitor precision and recall by segment, and set a floor for precision so analysts do not drown. The cat and mouse game never ends, but the point is to make the mouse work for it.