V-Patrol: Network Anomaly Detection

This document details the methodology, analysis, and results of a network anomaly detection project. The project aims to identify suspicious activities, assess network health, and mitigate potential security threats through feature extraction, data analysis, and iterative model refinement.

Feature Extraction:

The following features were extracted from network traffic data to enable anomaly detection:

Feature 1 & 2: Network Traffic Analysis

This section outlines the analysis of key network traffic metrics.

1. Requests Per Device Per Second

   This metric quantifies the frequency of requests originating from each device on a per-second basis. Monitoring this value allows for identification of devices generating unusually high traffic volumes, which may indicate malicious activity, misconfigurations, or network congestion.

2. Device Contribution Ratio

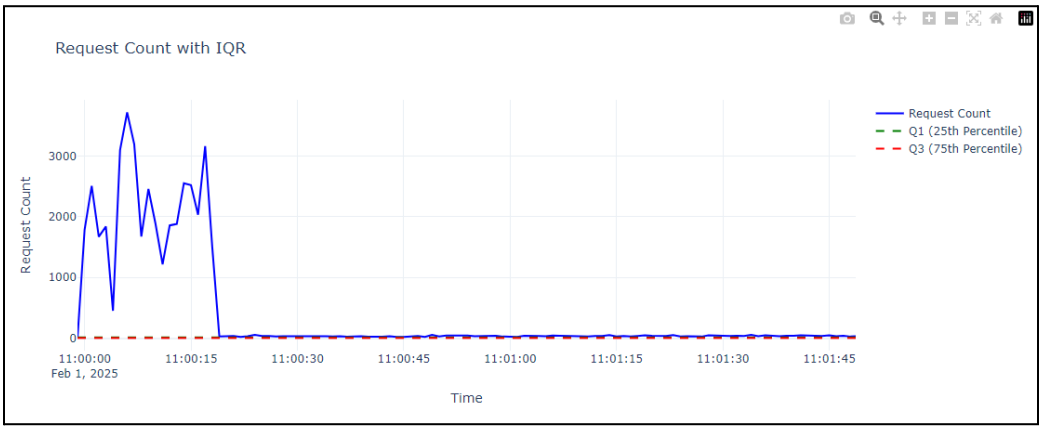
   The device contribution ratio represents the proportion of total network traffic attributed to each device. Analyzing these ratios helps identify dominant devices and assess their impact on network load. A high contribution ratio from a single device may indicate unusual behavior, such as a DDoS attack or a compromised IoT device.

3. Burstiness Analysis

   Burstiness quantifies the fluctuations in network traffic over time. Detecting irregular or suspicious request patterns can reveal potential cyber threats. Analysis of time-series data identifies periods of unusually high request rates, which can indicate botnet activity, automated scanning, or other network anomalies.

Visualization and Insights

Requests Per Device Per Second analyses provide critical insights into device behavior, network health, and potential security threats. Advanced anomaly detection techniques and machine learning-based pattern recognition can further improve the accuracy of suspicious activity detection.



Sudden Traffic Spike and Drop (11:00:00 - 11:00:20):

- Abnormality: A rapid increase in requests from 1 to 3,723 within seconds, followed by a sharp drop to 27 at 11:00:19.
- Implication: This event may indicate the initiation of a DDoS attack at 11:00:00, overwhelming the system. The subsequent abrupt decline suggests the server became unresponsive or defensive measures were activated, effectively blocking traffic.

Device Contribution Ratio Analysis

This section analyzes the device contribution ratios to identify devices with disproportionate impact on network traffic.

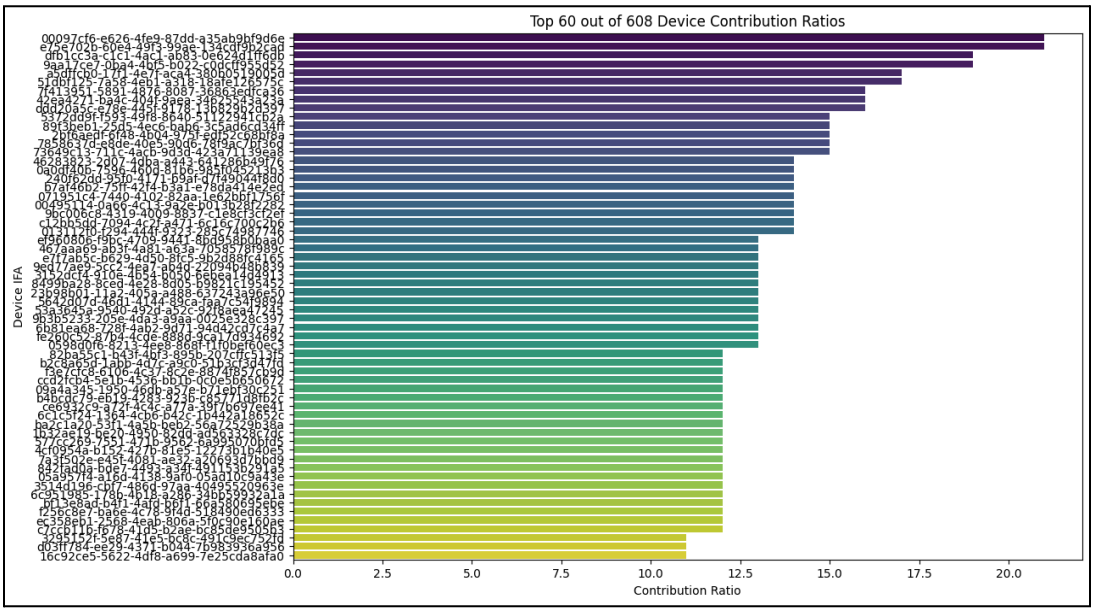
Overview of Contribution Ratios

The device contribution ratio is calculated as:

$$device\_contribution\_ratio = \frac{requests\_per\_device\_per\_second}{requests\_per\_second}$$

This metric highlights the proportion of total network traffic handled by each device. Analyzing these ratios helps identify devices critical to network operations, detect anomalies in device behavior, and optimize resource allocation.

The purpose of this analysis is to understand the behavior of devices with higher contribution ratios, assess their performance, and detect any abnormal patterns that may require further investigation.



Performance Assessment: Devices with high contribution ratios should be closely monitored for signs of stress or degradation.

We can establish baselines for normal contribution ratios for each device and configure automated alerts for deviations beyond acceptable thresholds. Investigate any spikes in contribution ratios for potential security incidents, such as account compromise or malicious activity targeting specific devices.

Burstiness Analysis for High-Contribution Devices

This section focuses on analyzing burstiness specifically for devices with high contribution ratios to identify potential performance bottlenecks or security threats.

Overview of Burstiness

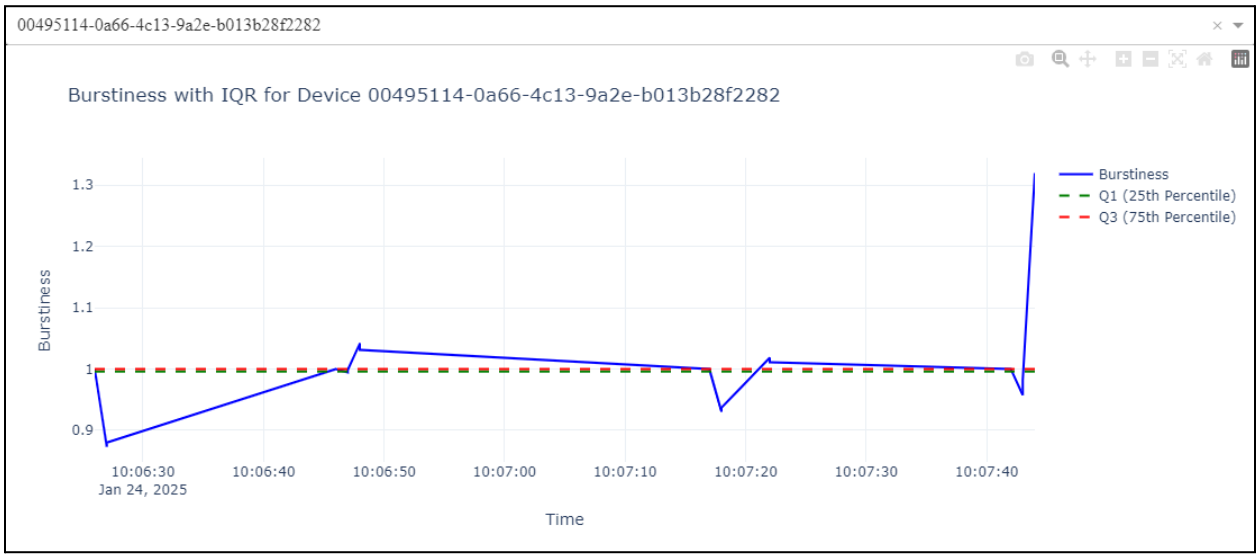
The burstiness is calculated as:

$$burstiness = \frac{requestCount}{baseline\_requests}$$

Burstiness measures the variability of traffic by comparing the actual number of requests requestCount to a baseline level of expected requests baseline_requests. It highlights periods of high activity (bursts) relative to normal behavior. High burstiness indicates short-term

spikes in traffic, which can strain network resources and potentially signal abnormal or malicious activity.

This analysis focuses on devices with device contribution ratios greater than the interquartile range (IQR). These devices already contribute disproportionately to network traffic, making their burstiness particularly relevant for identifying anomalies or performance bottlenecks.



Key Findings and Insights

1. High Burstiness in High-Contribution Devices:

   Devices with high burstiness values exhibit significant short-term traffic spikes relative to their baseline activity. Anomalies Such as Distributed Denial of Service (DDoS) attacks, misconfigurations, or compromised devices generating excessive traffic.

2. Anomalous Traffic Patterns:

   Some devices show extreme burstiness values compared to others in the high-contribution group. These outliers may signal unusual usage patterns or operational inefficiencies. Security incidents such as botnet activity or brute-force attacks targeting specific devices.

Conclusion

The burstiness analysis reveals critical insights into the traffic patterns of high-contribution devices. While some level of burstiness is expected due to normal operational variability, extreme values warrant closer attention as they may indicate inefficiencies or security risks. By proactively monitoring and managing these devices, organizations can ensure smoother network performance while mitigating potential threats.

Feature 3: Network Device Anomaly Detection

This section describes the process of cleaning network device data and detecting anomalies related to device configuration.

1. Handling Missing Vendor Information

Flagged rows where `device_vendor` is missing as abnormal using a `missing_vendor` column. Filled missing `device_vendor` values with `"Unknown"` to avoid issues in further grouping.

2. Device Consistency Check (Grouping and Anomaly Detection)

Grouped data by `device_vendor` and `device_model` to analyze consistency in hardware and software attributes. Identified inconsistencies in:

- `os` (operating system mismatch)
- `osv` (operating system version mismatch)
- `device_height` (height mismatch)
- `device_width` (width mismatch)

Applied a function to flag records where these attributes differ from the most common value in the group.

3. OS Version Normalization and Mismatch Detection

Extracted the major OS version from `osv` (before the first dot). Compared the extracted major version with `normalized_osv` and flagged mismatches.

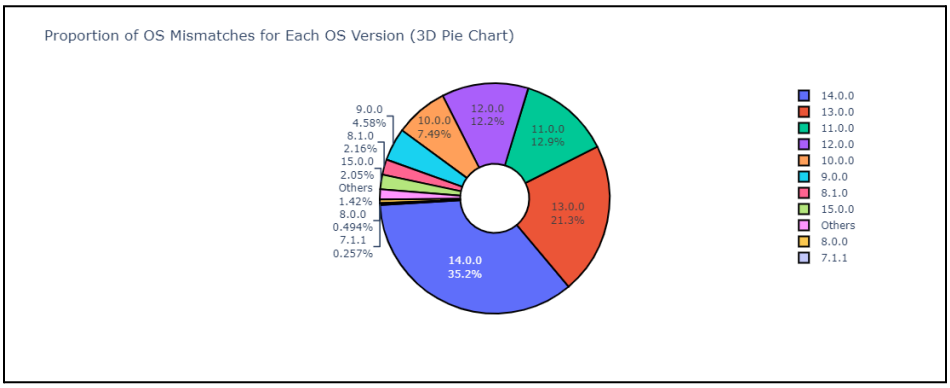
4. Summary Statistics and Export

Printed the count of records for each type of anomaly:

- vendor
- mismatch
- version mismatch
- mismatch
- Width mismatch

Visualization and Insights

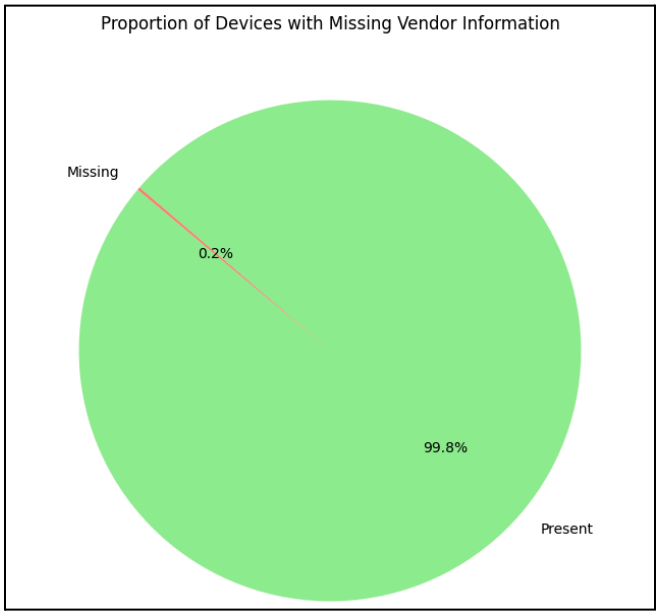
Insights from OS Version Mismatch Counts



1. Newer OS Versions Have More Mismatches:
   - OS 14.0.0 (5483 mismatches) and OS 13.0.0 (3317 mismatches) have the highest mismatches.

2. Steady Decline in Mismatches for Older Versions:
   - OS 12.0.0 (1898) → OS 11.0.0 (2005) → OS 10.0.0 (1166)
   - This pattern shows that as OS versions get older, mismatch occurrences decrease.

Insights from the Pie Chart on Missing Vendor Information



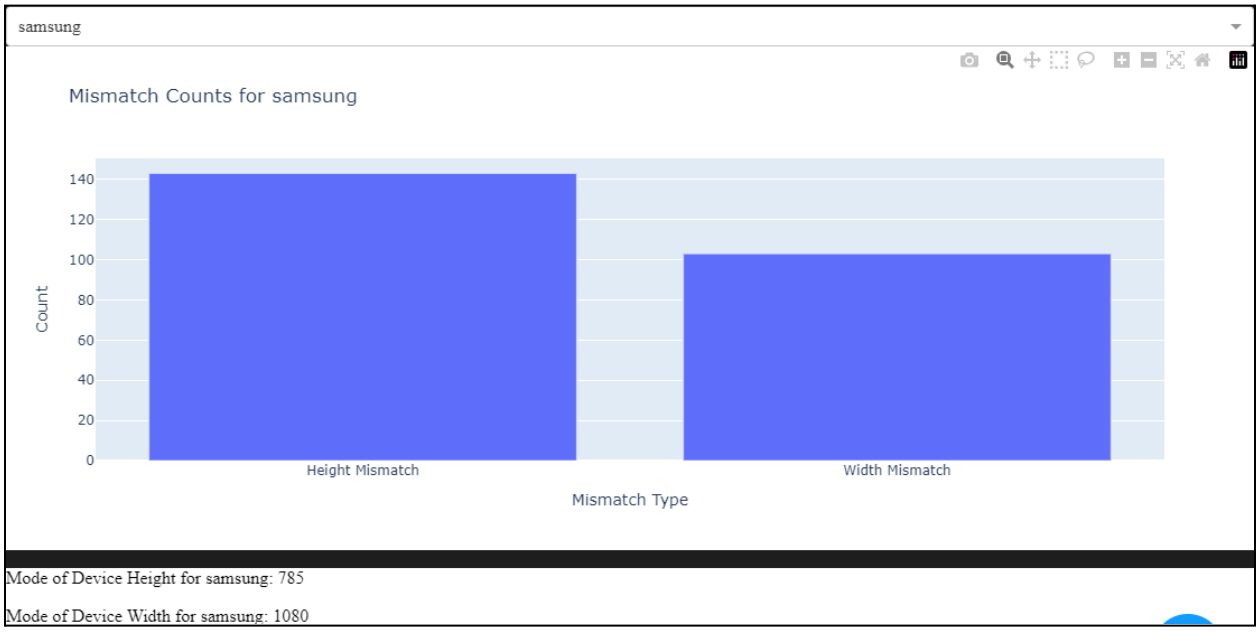
Very Few Devices Have Missing Vendor Information

- Only 0.2% of devices lack vendor details.
- This could be due to:
    - Anonymized or spoofed devices (e.g., bots, VPN users, or modified firmware).

Potential Implications:

- The low percentage of missing vendor information suggests it may not be a critical issue.
- However, the small subset of missing vendor data could still indicate suspicious activity (e.g., bots, rogue devices).

Insights from the Height and Width Mismatch Data



Trend Observation:

- More well-known smartphone brands (Samsung, Xiaomi, Oppo, Realme, Vivo) tend to have the highest mismatches.
- This could be due to a large variety of models, inconsistencies in reported dimensions, or regional variations in device manufacturing.
- Lesser-known brands and some niche manufacturers show no mismatches at all, possibly due to a smaller dataset size or more consistent device specifications.
- Unknown vendor names are one which are missing, though these devices show no mismatch in height and width, they are critical from the point of view of the rarity of their occurrence.

Feature 4 & 5 & 6: IP Address Anomaly Detection & Carrier Data Cleaning

This section details the process of identifying anomalous IP addresses and cleaning carrier data.

1. Missing Carrier Information Detection

Created a flag `missing_carrier` to identify rows where the `carrier` field is missing.

2. Identifying Impossible IP Addresses

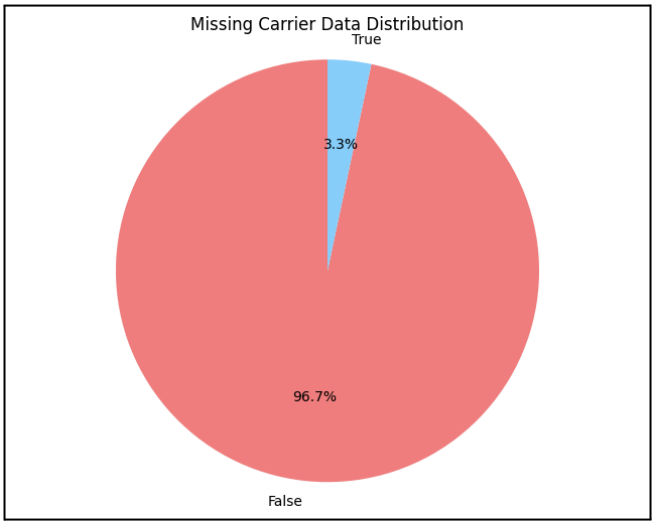
- Loaded a separate dataset containing IP addresses and their validity status.
- Identified fake or impossible IP addresses where `status == 'fail'`.
- Mapped these IPs and flagged their presence in the main dataset using the `impossible_ips` column.

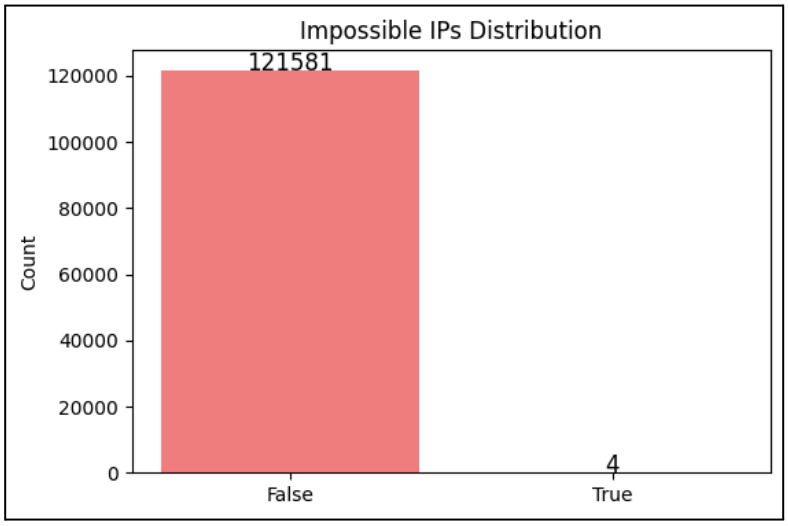
3. Country Code Mismatch Detection

- Created a mapping for country names to standardized country codes (`pak` for Pakistan, `hkg` for Hong Kong, etc.).
- Derived a new column `country_code_ip` using this mapping on the `ip_country` field.
- Compare `country_code_ip` with `device_country_code` to flag mismatches (`country_code_mismatch`).

Visualization and Insights

Insights from the Pie Chart on Missing Carrier Information





Observation:

- Pie Chart:
    - Very Few Devices Have Missing Carrier Information. Only 3.3% of devices lack carrier details. This could be due to Anonymized or spoofed devices (e.g., bots, VPN users, or modified firmware)
- Bar chart:

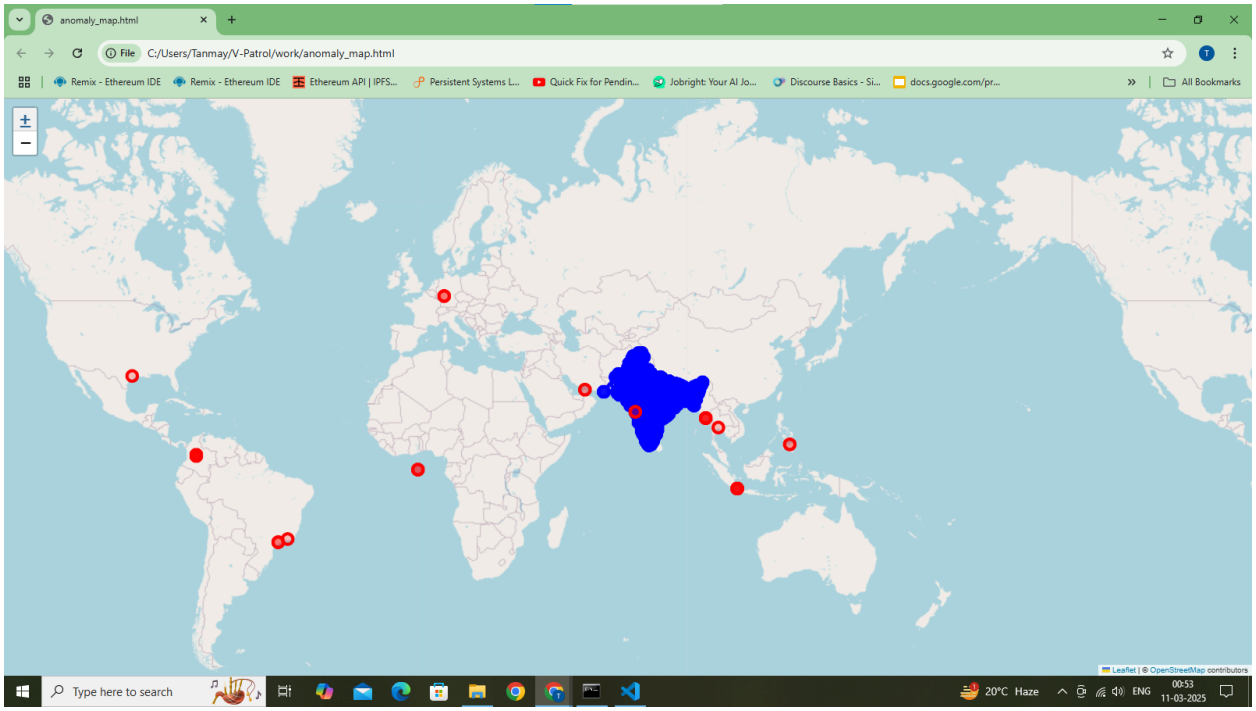
  Impossible IP Addresses

  Technical Context:
    - `0.0.0.0` is a non-routable placeholder IP (often used for misconfigured systems or DHCP failures).
    - `253.190.255.137` and `233.139.4.167` fall outside the valid public IPv4 range (`1.0.0.0` to `223.255.255.255`). These are reserved/multicast IPs (e.g., `233.x.x.x` is part of the multicast range `224.0.0.0/4`).

  Abnormality Insight:

  These IPs cannot originate from legitimate internet traffic and suggest:
    - Spoofed packets (e.g., DDoS reflection attacks).
    - Scanning/probing activity from compromised systems.

Analysis of the country code mismatches and unknown IP geolocations in dataset:



Suspicious Geographic Mismatches:

- Device-IP Geolocation Discrepancies:
    - Device reported country differs from IP-associated country (e.g., Bangladesh device with IP in Colombia/Myanmar/Philippines).
    - Potential Explanations: VPN/Proxy, Botnet proxying, Data exfiltration.
- Device vs. IP Country Code Inconsistencies:
    - Conflicting country codes for device vs. IP (e.g., India IP targeting Bangladesh device).
    - Potential Explanation: DNS spoofing or IP hijacking.

Clustering of Unknown IP Geolocations:

- Prevalence of IP addresses with unknown geolocation.
- "Unknown" country_code_ip correlates with reserved/non-routable IPs.
- Potential Explanations:
    - IP address spoofing.
    - Internal network vulnerabilities (misconfigured IoT).
    - DDoS amplification.

Repetitive Patterns in IP Geolocation Data:

- Recurring ip_country values (e.g., Colombia, Indonesia/Myanmar).
- Potential Explanations:
    - Botnet coordination (credential stuffing, port scanning).
    - Phishing/Scam campaign infrastructure.

Inconsistent Geolocation Metadata:

- Inconsistencies between device and IP-based locations (e.g., Hong Kong device, German IP).
- Potential Explanations:
    - Compromised system communication (malware command servers).
    - Unauthorized data exfiltration.

Feature 7: IP Address Geolocation & Distance Normalization

This section details the methodology employed to quantify and normalize the geographic disparity between a device's reported location and its location as estimated through IP address geolocation. Geolocation data for IP addresses was obtained from ip-api.com, API, and structured into an external dataset (ipData) for use in the analysis.

Haversine Distance Calculation

The Haversine formula, a method for calculating the great-circle distance between two points on a sphere using their latitudes and longitudes, was implemented. For the purposes of this calculation, the Earth's radius was assumed to be 6371 km. The formula was applied to determine the distance between:

- The GPS coordinates provided by the device (latitude and longitude).
- The latitude and longitude coordinates obtained from an IP address geolocation lookup (ip_lat and ip_lon).

The resultant haversine_distance_km value represents the geographic difference between the device's reported location and its location estimated via IP address.

Normalization of Haversine Distances

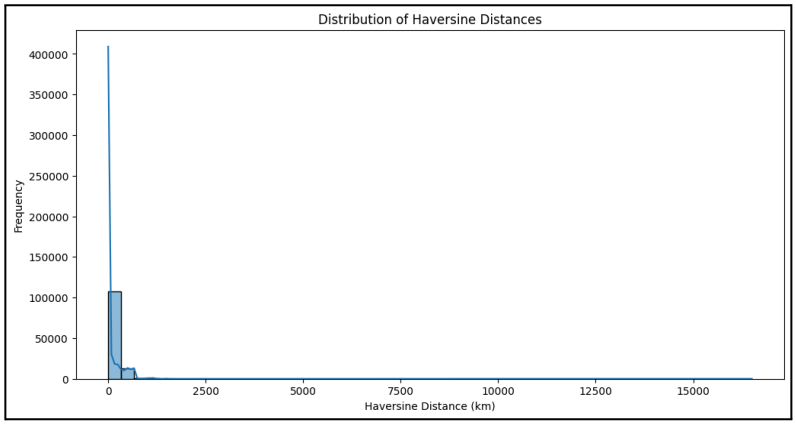
To account for varying levels of accuracy, the StandardScaler was applied separately to distances derived from GPS coordinates and those derived from IP addresses. GPS-derived distances (location_type == 1) and IP-derived distances (location_type == 2) were normalized using distinct scaling parameters. This separation was essential because IP-based geolocation is inherently less precise than GPS data. The resulting normalized_distance column provides a scale-independent measure of geographic disparity.

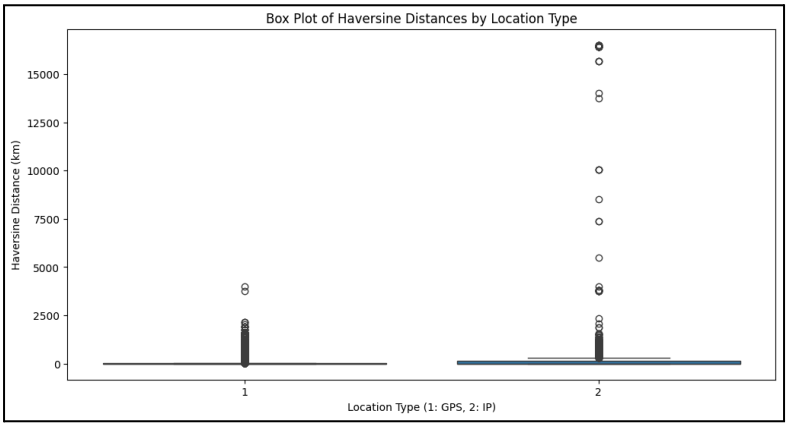
Key Insights

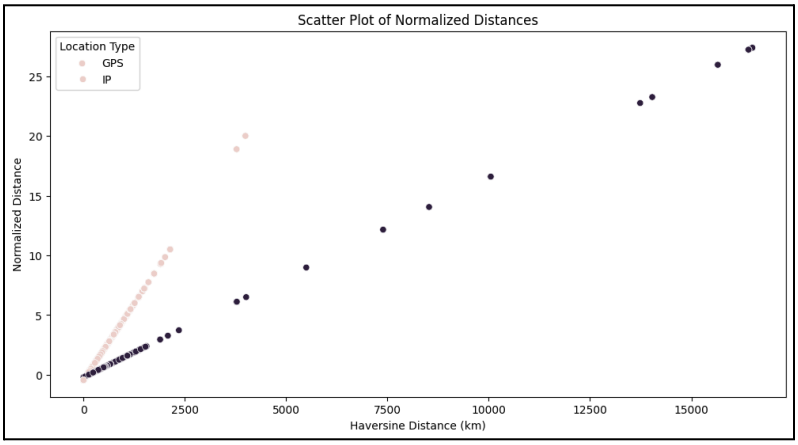
- The Haversine formula offers a precise method for quantifying discrepancies between GPS-based and IP-based location information.
- Due to the inherent limitations in the accuracy of IP geolocation, separate normalization procedures were applied to avoid skewing comparisons between the two location methodologies.
- Future refinements could include the incorporation of confidence scores reflecting the reliability of the underlying location data (e.g., differentiating between Wi-Fi-based and mobile IP-based locations)."

Visualization and Insights

Insights from the Haversine Distance Analysis:







Observation:

- Distribution Graph:

  The distribution of Haversine distances shows a long tail extending to 15,000 km, with non-trivial frequencies beyond 10,000 km. Distances >10,000 km between GPS and IP locations are geographically impossible for legitimate user activity (e.g., a device in New York cannot physically connect via an IP in Jakarta within seconds). Likely scenarios:

  - Spoofed GPS coordinates (e.g., fake location apps).
  - IP masking (e.g., VPNs/proxies used to bypass geo-restrictions or hide C2 servers).
  - Account takeover (e.g., stolen credentials used from a distant location).

- Box Plot Graph:

  The box plot suggests IP-based distances (`location_type=2`) have a wider spread (higher IQR) compared to GPS. Outliers in IP distances are more frequent (e.g., values near 15,000 km). IP geolocation is inherently less accurate, but extreme outliers indicate:
  - Bulletproof hosting (e.g., attackers using IPs from unregulated regions).
  - Proxy chaining (e.g., traffic routed through multiple countries to evade detection).
  - GPS outliers, while fewer, could signal GPS spoofing (e.g., fraudulent ad clicks).

- Scatter Plot:

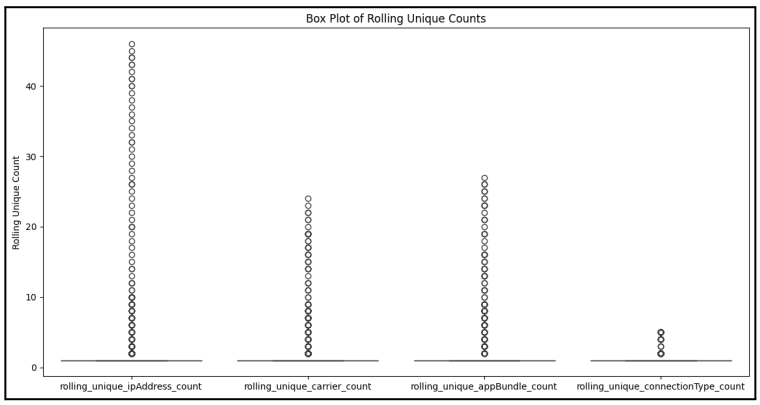
After normalization, some GPS/IP distances deviate significantly from the mean (e.g., normalized values >3σ). Abnormality Insight:

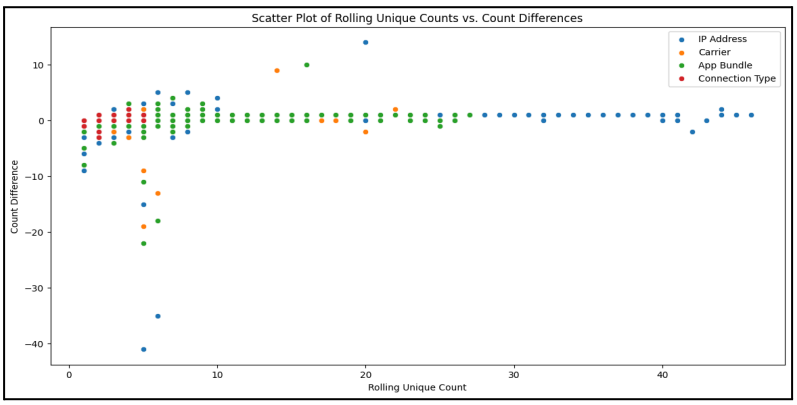
- GPS outliers: Devices with sudden location jumps (e.g., "impossible travel" from Paris to Sydney in minutes).
- IP outliers: IPs consistently mismatched with GPS (e.g., a device in Bangladesh always connecting via Colombian IPs).
- These patterns align with botnet activity or data exfiltration (e.g., rerouting traffic to obscure destinations).

Feature 8: Rolling Unique Counts & Feature Engineering

This section describes the process of computing rolling unique counts for various network features and engineering new features for improved anomaly detection. These features were majorly extracted to capture the temporal pattern inside the data set.

Visualization and Insights





Observations:

- Box Plot:
  - IP Addresses: Widest IQR (5–25) and extreme outliers (up to 40).
  - Carrier: Higher Median (~15) compared to app bundles (~10).
  - App Bundle: Outliers at 20+ counts.
  - Connection Type: Tightest Distribution (IQR: 2–8).
- Scatter Plot
  - IP Addresses: Clustered in top-right quadrant (high rolling counts + positive diffs). Example: `(40, +30)`.
  - Connection Type: Negative Diffs: Sudden drops (e.g., `rolling=10` → `diff=-20`).
  - Carrier/App Bundle: Mostly clustered near the origin (low volatility).

Feature 9: BERT-Based Embedding Generation and Similarity Computation

This section details the use of BERT (Bidirectional Encoder Representations from Transformers) to generate embeddings for user-agent strings and device metadata, and to compute their cosine similarity.

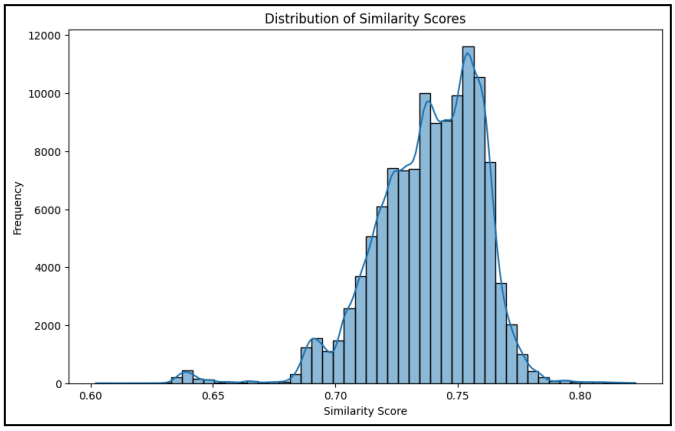
These embeddings, which capture semantic relationships, allow for advanced comparisons beyond simple string matching. BERT tokenizes the input text, and the [CLS] token's embedding from the final hidden state is extracted as the representative vector. Cosine similarity is then used to measure the relatedness of UA and metadata embeddings, with high scores indicating a strong relationship and low scores suggesting anomalies.

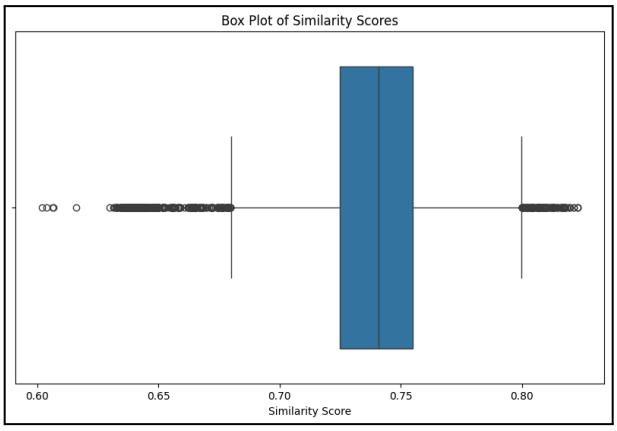
How This Helps in Abnormality Detection:

- Detecting Spoofed or Malicious Requests: Identifies mismatches between User-Agent (UA) strings and device metadata, revealing potential fraudulent attempts by attackers disguising their traffic.
- Identifying Device Misconfigurations: Supports identifying devices deviating from security policies by pinpointing inconsistencies between UA and metadata, which can introduce security risks.
- Preventing Automated Bot Attacks: Flags requests exhibiting significant deviations from typical user behavior by leveraging generic or inconsistent UA strings associated with bots.
- Enhancing Fraud Detection in Web Traffic: Identifies outliers indicative of fraudulent transactions by comparing UA-metadata pairings against a baseline of legitimate activities.

Visualization and Insights

Distribution of Similarity Scores





Observations:

- Distribution Graph
    - Primary Peak: Majority of scores cluster around 0.75 (frequency ~8,000).
    - Secondary Peaks: Smaller spikes at 0.70 and 0.80.
    - Low-Score Tail: Non-trivial counts (~2,000) at 0.60–0.65.

  Abnormality Insights:
    - High Scores 0.75 Peak: Represents normal traffic where user-agent (UA) and device metadata (OS, model, etc.) align.
    - Low Scores (0.60–0.65): Indicate UA-metadata mismatches, e.g.:
        - Spoofed UAs (e.g., `Chrome` UA paired with `iOS` metadata).
        - Malware/botnets using randomized UA strings to evade detection.
        - Could signal credential stuffing (tools like Sentry MBA generate mismatched headers).
- Box Plot
    - Median: ~0.75 (matches distribution peak).
    - IQR: 0.70–0.80 (interquartile range).
    - Outliers: Scores below 0.65 (beyond the lower whisker).

  Abnormality Insights:

    - The presence of outliers with low cosine similarity scores (less than 0.65) is a significant indicator of potential security threats.
    - These scores, deviating from the tightly clustered distribution of legitimate traffic with consistent UA-metadata pairings, suggest the involvement of sophisticated malicious automation, as well as common attack vectors like spoofing, bot-driven activities, and credential-stuffing attempts.
    - Prioritizing the investigation of traffic within these outlier ranges can be critical for mitigating potential security breaches.

Modeling: Iterative Refinement of the Isolation Forest Model

To optimize anomaly detection performance, an Isolation Forest model was subjected to a rigorous iterative refinement process. This approach involved systematically exploring various hyperparameter configurations, extracting insights from each iteration, and applying these findings to guide subsequent adjustments. The key objective was to maximize the model's ability to accurately discriminate between normal and anomalous network traffic patterns.

- Output 1: Initial Baseline
    - Best parameters: {'n_estimators': 100, 'max_samples': 256, 'contamination': 0.1, 'max_features': 0.9, 'bootstrap': False}
    - Best score: 0.6556688332880479
- The initial iteration established a baseline performance level using a standard set of hyperparameters, achieving a score of 0.6556688332880479. This served as the foundation for subsequent enhancements.
- Output 2: Optimization of Sample Size
    - Best parameters: {'n_estimators': 100, 'max_samples': 128, 'contamination': 0.1, 'max_features': 0.9, 'bootstrap': False}
    - Best score: 0.6778980448006691

Subsequent experimentation focused on tuning the max_samples hyperparameter. Reducing this value to 128 resulted in an improved score of 0.6778980448006691, indicating that a smaller sample size was more effective in capturing the underlying data characteristics.

Furthermore, a feature selection step was performed after obtaining output2 . By separating the Boolean columns, normalized_distance, and similarity_score from the rest of the dataset and running the model on both sets, it was determined that utilizing the Boolean columns, normalized_distance, and similarity_score resulted in improved model performance, guiding the subsequent modeling stages.

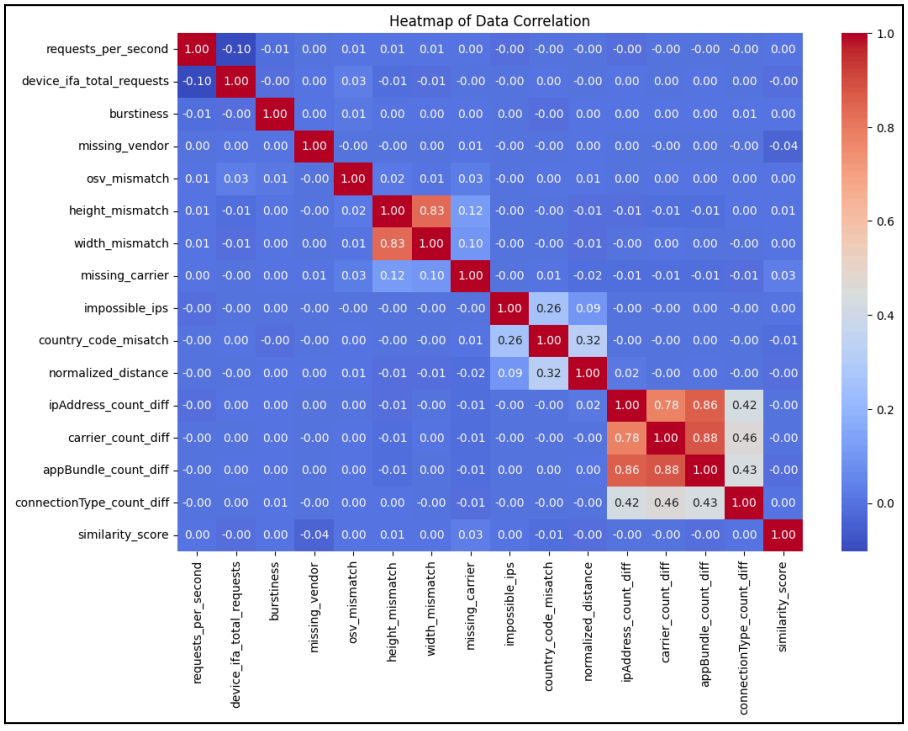
- Output 3: Adjustment of Contamination and Estimators
    - Best parameters: {'n_estimators': 150, 'max_samples': 128, 'contamination': 0.1602582555413908, 'max_features': 0.9, 'bootstrap': False}
    - Best score: 0.719502379676749
- A significant performance gain was achieved by introducing a refined value for the contamination hyperparameter and increasing the number of estimators (n_estimators) to 150. This yielded a best score of 0.719502379676749, representing a substantial improvement over the baseline.
- Output 4: Fine-Tuning of Sample Size (Continued)
    - Best parameters: {'n_estimators': 150, 'max_samples': 64, 'contamination': 0.1, 'max_features': 0.9, 'bootstrap': False}

    - Best score: 0.7301869661156942
- Further optimization involved a reduction of the max_samples hyperparameter to 64. This fine-tuning resulted in an even higher score of 0.7301869661156942, reinforcing the conclusion that a smaller sample size was beneficial for the model's ability to detect anomalies in this specific dataset.
- Final Iteration: Confirmation of Optimal Parameters
  The final iteration served to validate the optimal hyperparameter configuration identified in the previous stage. The score remained consistent at 0.7301869661156942, confirming that the model had reached an optimized state for the given dataset and feature set.

Conclusion: Demonstrating Progressive Model Enhancement
Through a deliberate and iterative process, a range of hyperparameter combinations were systematically explored, with each iteration building upon the insights gleaned from previous results. This iterative approach enabled a progressive enhancement of the Isolation Forest model's performance, culminating in a noteworthy increase in the best score from the initial baseline of 0.6556688332880479 to a final optimized score of 0.7301869661156942.

Correlation Heatmap Analysis

To further refine the model's understanding of feature relationships, a correlation heatmap was generated from the final dataset.

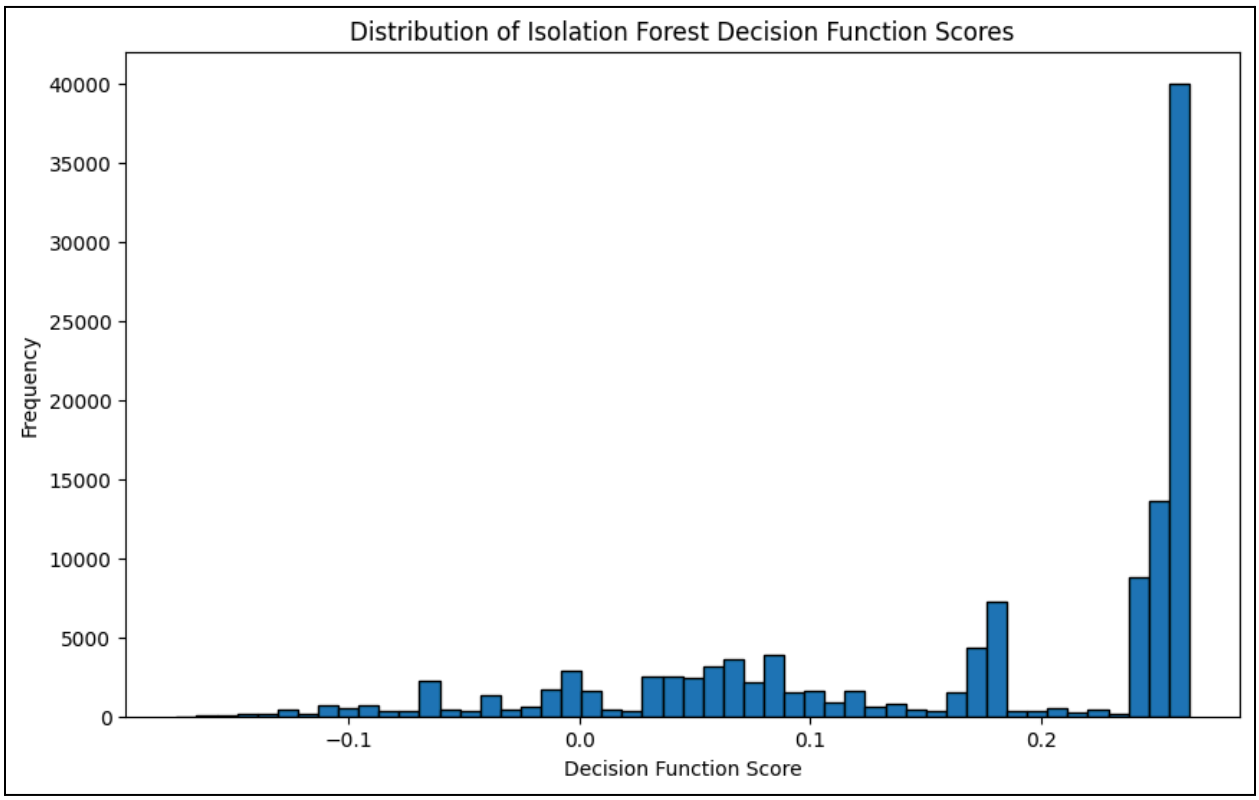


- Strong Correlations

    - Height and Width Mismatch (0.83): A high correlation suggests that inconsistencies in reported device dimensions often occur simultaneously, potentially indicating device spoofing.
    - IP Address Count Difference & Carrier Count Difference (0.78): A strong link between changes in IP addresses and carrier networks may be indicative of device movement, VPN usage, or fraudulent activity.
    - Carrier Count Difference & App Bundle Count Difference (0.88): Frequent carrier switching is highly correlated with changes in app bundles, suggesting associations with different user behaviors or multiple users on a single device.
- Moderate Correlations
    - Impossible IPs & Country Code Mismatch (0.26): A connection between impossible IPs and mismatched country codes suggests potential spoofing or VPN usage.
    - Country Code Mismatch & Normalized Distance (0.32): Discrepancies in country codes and physical distance could indicate geo-spoofing or inaccurate IP geolocation data.
    - IP Address Count Diff & Connection Type Count Diff (0.42): Changes in IP addresses and connection types may be expected for mobile users switching between WiFi and cellular networks.
- Low or No Correlation
    - Requests Per Second & Most Other Features (~0.00): Request frequency appears to be independent of other observed network anomalies.
- Key Takeaways
    - Device spoofing is suggested by strong correlations between height and width mismatches.
    - Geo-spoofing is potentially indicated by impossible IPs and country mismatches.
    - Normal network behavior can account for some changes, such as IP and connection type differences for mobile users.

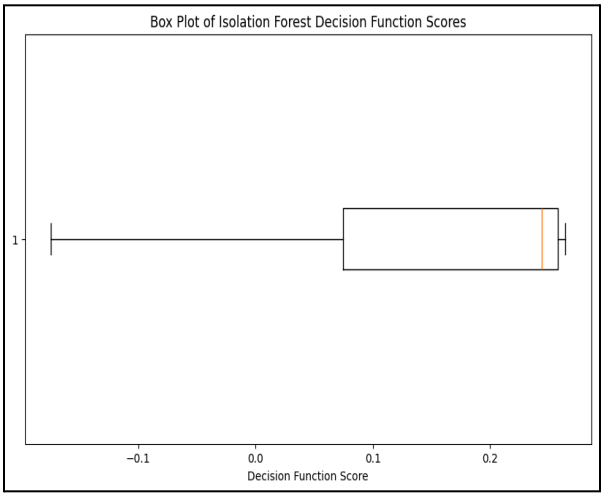
Isolation Forest Analysis & Hybrid Scoring

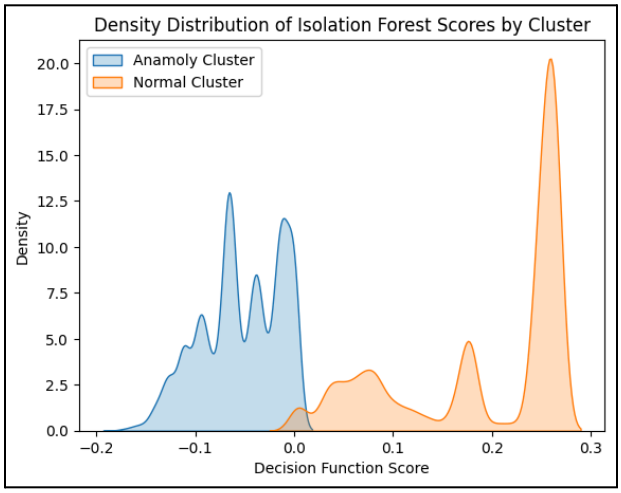
The Isolation Forest model, combined with a hybrid scoring approach, demonstrates strong performance in separating normal and anomalous traffic.



- Cluster Separation & Silhouette Score
    - The hybrid scorer achieved a silhouette score of 0.76, demonstrating strong separation between the two identified clusters.
    - The Isolation Forest's decision function scores exhibit a clear dichotomy, suggesting a well-defined distinction between "normal" and "abnormal" behaviors, although further labeled data validation would be required to guarantee semantic alignment with true anomalies
- Distribution of Decision Scores
    - Histogram analysis reveals a primary cluster of ~35,000 instances with scores between -0.1 and 0.0 (likely normal traffic), and a minor cluster of ~10,000 instances near 0.1 (potential anomalies).
    - The right-skewed distribution implies that anomalies are rare but distinct, with the Isolation Forest flagging approximately 22% of instances as higher-scoring (Cluster 1), requiring further investigation to confirm their status. "

Results and Visualization





Percentage of anomalies: 10.23%

Conclusion:

The anomaly detection project, leveraging an Isolation Forest model, yielded robust and highly interpretable results, achieved through the strategic exclusion of low-correlation and temporal features. The project's key outcomes contribute significantly to strengthening cybersecurity defenses:

Demonstrated Effective Clustering:

- A high silhouette score of 0.76 validates the effectiveness of the feature selection strategy, confirming a strong separation between "Normal" and "Anomaly" clusters.
- Density distributions distinctly illustrate that anomalies are concentrated in lower, more negative, decision scores, underscoring the model's capacity to differentiate abnormal behavior.

Consistent and Reliable Anomaly Detection:

The model consistently flagged approximately 10.23% of instances as anomalies, closely aligning with the contamination=0.1 parameter. This consistency highlights the model's reliability, indicating that it is well-suited for identifying real-world threats in a consistent manner.

Impactful Feature Engineering:

- Strategic exclusion of temporal features, such as requests_per_second and burstiness, and low-correlation variables, improved the model's focus on critical indicators such as impossible_ips and country_code_mismatch.
- This indicates that by focusing on highly relevant indicators, the model was able to generate a more distinct separation between normal and anomalous clusters.

Future Scope: Addressing Temporal Variability and Feature Selection

It is important to note that certain features (requests_per_second, device_ifa_total_requests, burstiness, ipAddress_count_diff, carrier_count_diff, appBundle_count_diff, connectionType_count_diff) were excluded after Output 2 due to their limited contribution to the silhouette score. These features exhibit temporal variability, potentially challenging the Isolation Forest's ability to effectively detect anomalies over time. Future work could explore employing neural network-based models, such as autoencoders, to analyze these temporal features and potentially enhance anomaly detection performance.

Thank You for reading. Hoping to get in contact with you.