Machine learning and predictive analytics
Cyber Attack: Network Overload

This report presents the assessment for the machine learning module, focusing on designing a model to detect cyberattack events that saturate website traffic.

Problem Identification

A Denial of Service (DoS) attack is a form of cyberattack that has experienced a significant increase in both frequency and scale. In 2025, reported incidents rose sharply, with traffic volume associated with DoS activity increasing by 358% (Khalil, 2025).

The most frequently targeted industries are concentrated within the financial sector, however, attack patterns can vary significantly, with reported attacks spanning a broad range of sectors from marketing firms to government services.

The objective of this report is to develop a predictive model capable of identifying and flagging potential DoS attacks based on observed normal and abnormal traffic behavior, using a provided wireless network traffic dataset to detect deviations from baseline conditions.

To achieve this, and by analysing individual network connections provided in a given dataset, anomaly detections could be identified through comparison against normal network conditions, versus unusual flows.

A key constraint of this analysis is that network traffic, particularly during cyberattacks, is neither a static flow or sequentially linear.

Therefore, the aim is not to deliver a complete solution, but to develop a model that detects and clusters anomalous traffic based on deviations from normal patterns, while minimizing false positives and maintaining a high detection rate.

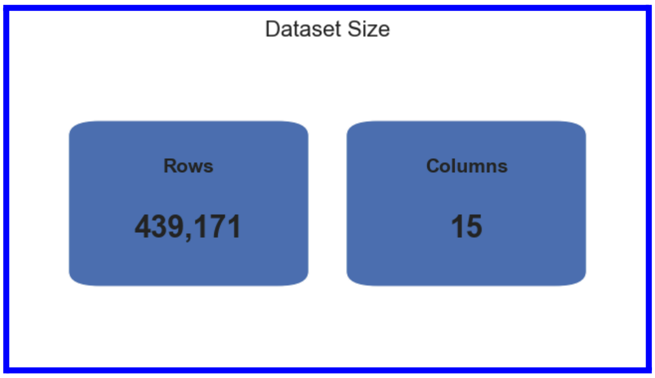
As mitigation strategies within cybersecurity continue to evolve, this report focuses on approaching the problem by developing a model based on a dataset containing four attack types: ARP spoofing, ARP storm, SYN flood, and PING flood (Rakić, 2024).

Dataset description and analysis

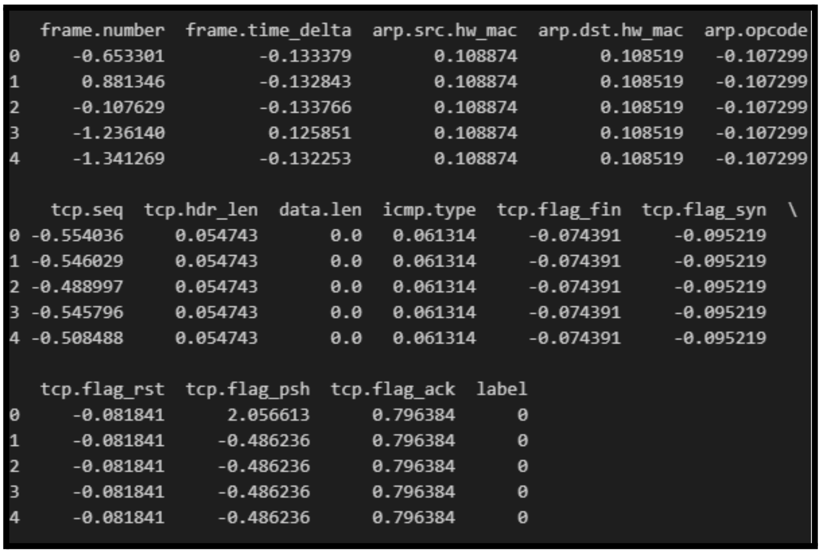
Data collection and preparation account for approximately 60–80% of the total project time in machine learning (University of the West of England). From data acquisition through preparation, data preprocessing represents one of the most critical stages of the machine learning workflow.

Before model selection and algorithm choice, an exploratory analysis is required to characterize the dataset and identify necessary preprocessing steps, such as normalization, feature transformation, or structural adjustments. This section focuses on dataset size, feature attributes, and preparations needed for seamless integration into the algorithmic pipeline.

This initial analysis proves that this dataset is well structured from the outset, containing no missing values and comprising a medium-to-large number of instances. The feature-to-sample ratio (439,171 / 15 ≈ **29,278:1**) is sufficiently high, thereby reducing the risk of variance instability.



As shown, the dataset comprises **439,171** rows and **15** columns resulting in a total of **6,587,565** observations, giving a sufficient sample size for model training, and supplying the input feature matrix (X) in the form of an already normalized numerical vector.



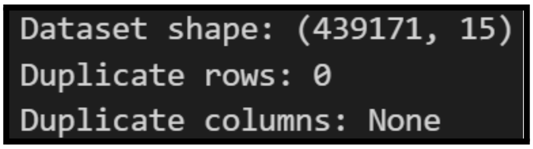
```
   frame.number  frame.time_delta  arp.src.hw_mac  arp.dst.hw_mac  arp.opcode
0     -0.653301         -0.133379        0.108874        0.108519   -0.107299
1      0.881346         -0.132843        0.108874        0.108519   -0.107299
2     -0.107629         -0.133766        0.108874        0.108519   -0.107299
3     -1.236140          0.125851        0.108874        0.108519   -0.107299
4     -1.341269         -0.132253        0.108874        0.108519   -0.107299

    tcp.seq  tcp.hdr_len  data.len  icmp.type  tcp.flag_fin  tcp.flag_syn  \
0 -0.554036     0.054743       0.0   0.061314     -0.074391     -0.095219
1 -0.546029     0.054743       0.0   0.061314     -0.074391     -0.095219
2 -0.488997     0.054743       0.0   0.061314     -0.074391     -0.095219
3 -0.545796     0.054743       0.0   0.061314     -0.074391     -0.095219
4 -0.508488     0.054743       0.0   0.061314     -0.074391     -0.095219

   tcp.flag_rst  tcp.flag_psh  tcp.flag_ack  label
0     -0.081841      2.056613      0.796384      0
1     -0.081841     -0.486236      0.796384      0
2     -0.081841     -0.486236      0.796384      0
3     -0.081841     -0.486236      0.796384      0
4     -0.081841     -0.486236      0.796384      0
```

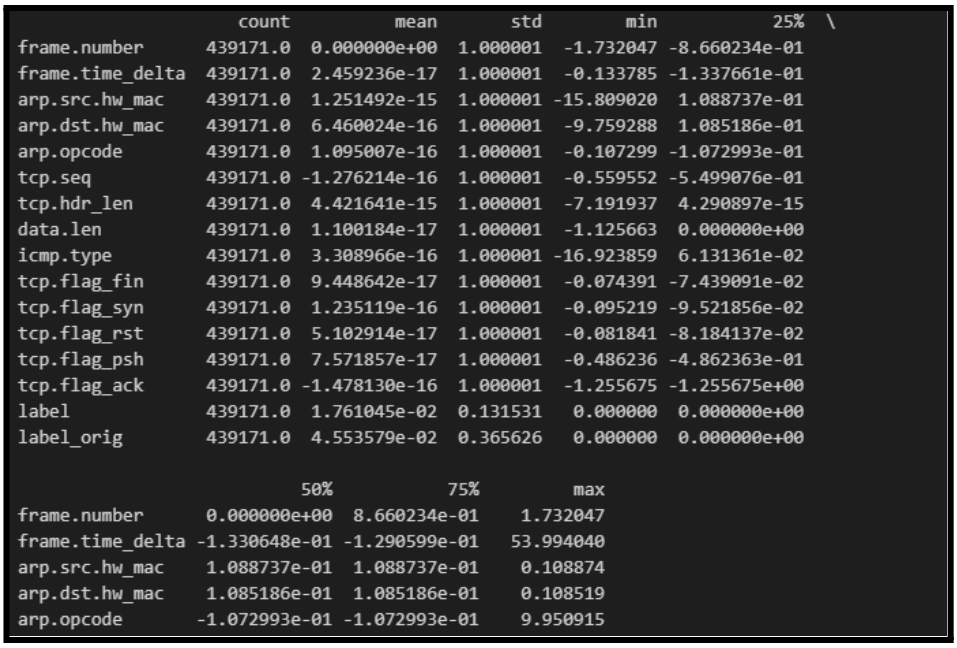
Normalization is essential for preventing scale dominance, particularly with respect to classification as it reduces noise while improving predictive performance and overall accuracy (GeeksforGeeks, 2025).

With a shape of (439,171, 15), a low-dimensional feature space, and no duplicates or missing values, the dataset is well suited for stable statistical estimation without inflation from irrelevant information. This supports reliable interpretability, robust decision boundary estimation, and improved predictive performance.



```
Dataset shape: (439171, 15)
Duplicate rows: 0
Duplicate columns: None
```

Descriptive statistics indicate that the dataset has already been substantially preprocessed, with raw data transformed to meet the numerical and statistical assumptions required for machine learning algorithms.



```
                     count          mean       std        min           25%  \
frame.number      439171.0  0.000000e+00  1.000001  -1.732047 -8.660234e-01
frame.time_delta  439171.0  2.459236e-17  1.000001  -0.133785 -1.337661e-01
arp.src.hw_mac    439171.0  1.251492e-15  1.000001 -15.809020  1.088737e-01
arp.dst.hw_mac    439171.0  6.460024e-16  1.000001  -9.759288  1.085186e-01
arp.opcode        439171.0  1.095007e-16  1.000001  -0.107299 -1.072993e-01
tcp.seq           439171.0 -1.276214e-16  1.000001  -0.559552 -5.499076e-01
tcp.hdr_len       439171.0  4.421641e-15  1.000001  -7.191937  4.290897e-15
data.len          439171.0  1.100184e-17  1.000001  -1.125663  0.000000e+00
icmp.type         439171.0  3.308966e-16  1.000001 -16.923859  6.131361e-02
tcp.flag_fin      439171.0  9.448642e-17  1.000001  -0.074391 -7.439091e-02
tcp.flag_syn      439171.0  1.235119e-16  1.000001  -0.095219 -9.521856e-02
tcp.flag_rst      439171.0  5.102914e-17  1.000001  -0.081841 -8.184137e-02
tcp.flag_psh      439171.0  7.571857e-17  1.000001  -0.486236 -4.862363e-01
tcp.flag_ack      439171.0 -1.478130e-16  1.000001  -1.255675 -1.255675e+00
label             439171.0  1.761045e-02  0.131531   0.000000  0.000000e+00
label_orig        439171.0  4.553579e-02  0.365626   0.000000  0.000000e+00

                           50%           75%        max
frame.number      0.000000e+00  8.660234e-01   1.732047
frame.time_delta -1.330648e-01 -1.290599e-01  53.994040
arp.src.hw_mac    1.088737e-01  1.088737e-01   0.108874
arp.dst.hw_mac    1.085186e-01  1.085186e-01   0.108519
arp.opcode       -1.072993e-01 -1.072993e-01   9.950915
```

Based on these considerations, several key attributes were selected as essential for training the model. These are listed below:

- frame.time_delta (primary)
- data.len (primary)
- tcp.flag_syn (primary)
- tcp.flag_ack (primary)
- tcp.flag_rst (primary)
- tcp.flag_fin (secondary)
- tcp.seq (secondary)

All directly characterize traffic intensity and flow behavior and are therefore central to the analysis, supporting validation of the following key characteristics:

Temporal Pressure	frame.time_delta
Load magnitude	data.len
Transport-state stress	TCP flags
Flow integrity context	tcp.seq

As confirmed above, only limited additional preprocessing is required to ensure data completeness. Observations are assumed to be largely independent, with no missing values, and potential outliers are treated as indicators of suspicious activity rather than measurement errors.

As a result, a small subset of features, such as TCP flags, should be converted into binary variables for classification. Identifier variables, including *arp.src.hw_mac* and *arp.dst.hw_mac*, are excluded, as they are categorical identifiers numerically encoded and would distort the model if included.

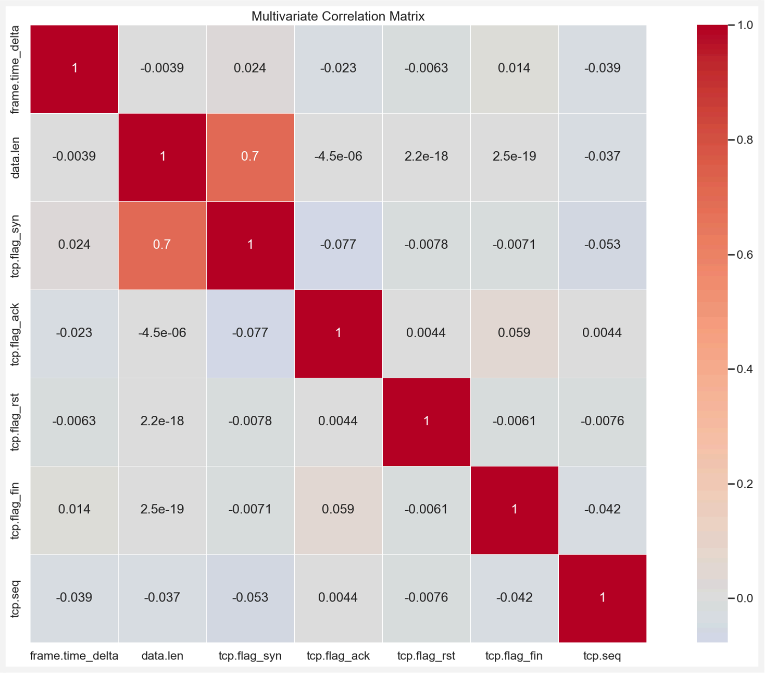
Given that the dataset is already suitably preprocessed and the selected features are informative, the analysis can proceed to model selection. Additional metrics for further feature engineering can be introduced later through targeted aggregations if needed.

Analysis approach

Identifying relationships between variables and assessing multicollinearity requires bivariate and multivariate analysis. Bivariate analysis examines pairs of variables, while multivariate analysis considers multiple variables simultaneously to uncover more complex patterns and interactions (Statistics How To, 2026; Mengual-Macénlle et al., 2015).

Selecting an appropriate analytical approach is essential for detecting multicollinearity and understanding relationships, supporting effective feature selection. Accordingly, a multivariate correlation matrix is the most suitable method for examining overall feature relationships at this stage.

The correlation matrix presented below forms the basis for identifying linear dependencies among attributes.



	frame.time_delta	data.len	tcp.flag_syn	tcp.flag_ack	tcp.flag_rst	tcp.flag_fin	tcp.seq
frame.time_delta	1	-0.0039	0.024	-0.023	-0.0063	0.014	-0.039
data.len	-0.0039	1	0.7	-4.5e-06	2.2e-18	2.5e-19	-0.037
tcp.flag_syn	0.024	0.7	1	-0.077	-0.0078	-0.0071	-0.053
tcp.flag_ack	-0.023	-4.5e-06	-0.077	1	0.0044	0.059	0.0044
tcp.flag_rst	-0.0063	2.2e-18	-0.0078	0.0044	1	-0.0061	-0.0076
tcp.flag_fin	0.014	2.5e-19	-0.0071	0.059	-0.0061	1	-0.042
tcp.seq	-0.039	-0.037	-0.053	0.0044	-0.0076	-0.042	1

When variables are highly correlated, a predictive model may struggle to isolate their individual effects, leading to unstable decision boundaries and an increased risk of false positives and false negatives.

Through this analysis, a strong correlation is identified between **data.len** and **tcp.flag_syn**, highlighting the importance of carefully adjusting features to prevent highly related variables from distorting the model.

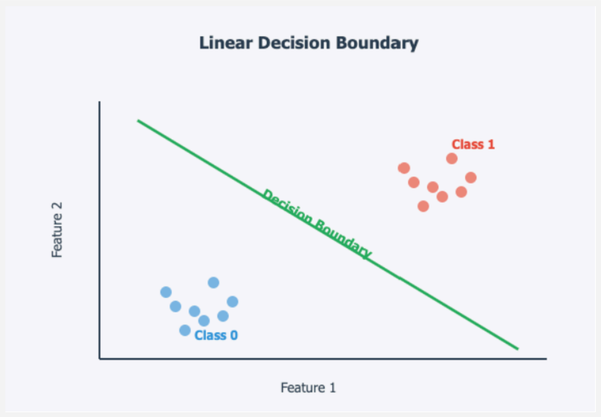
A key finding at this stage is the potential need for regularization techniques, such as Ridge or Lasso, to reduce feature weights or eliminate less informative variables. Subsequent bivariate analysis can then be used diagnostically to examine specific variable pairs, validate findings, and support further refinement.

Algorithm selection

Given the nature of the dataset, logistic regression is selected as the primary modelling algorithm. It is a well-established classification method for predicting categorical outcomes (LinkedIn Learning, 2026) and is well suited to detecting abnormal behaviour in this attack scenario.

Within this context, the availability of labeled input and output data enables a robust supervised learning approach that leverages historical observations for predictive classification. Combined with a clean, preprocessed dataset, supervised methods can systematically learn decision boundaries that effectively separate normal and abnormal behaviour.

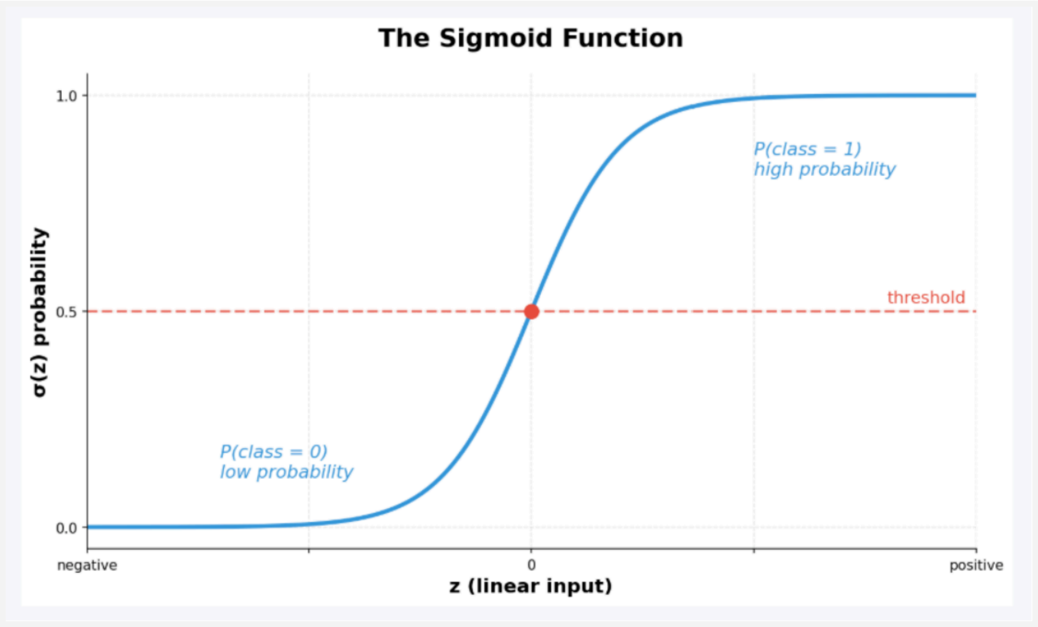
In this case, class separation is achieved using a linear decision boundary, as shown:



This is a key consideration in algorithm selection, as logistic regression is well suited to problems with linear separability.

Linear separation may take the form of a line, plane, or hyperplane dividing classes (University of the West of England), and by using the sigmoid function, the model performs binary classification by estimating class membership probabilities and assigning observations to class 0 or class 1 based on a decision threshold.

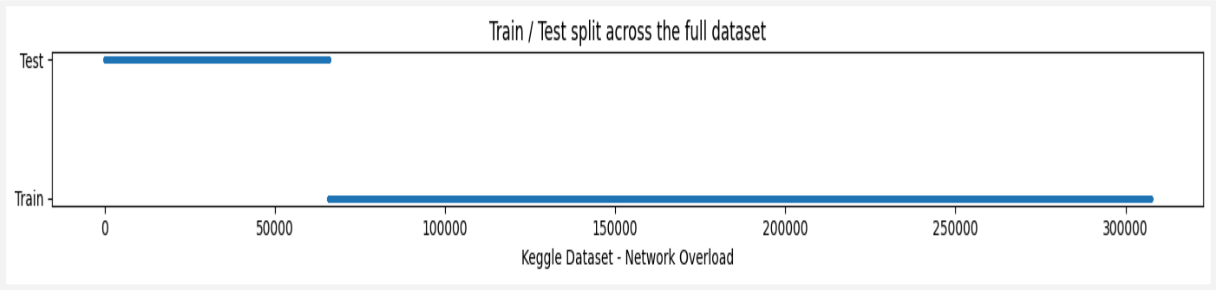
Adjusting this threshold allows further refinement, improving detection sensitivity while controlling false positive rates.



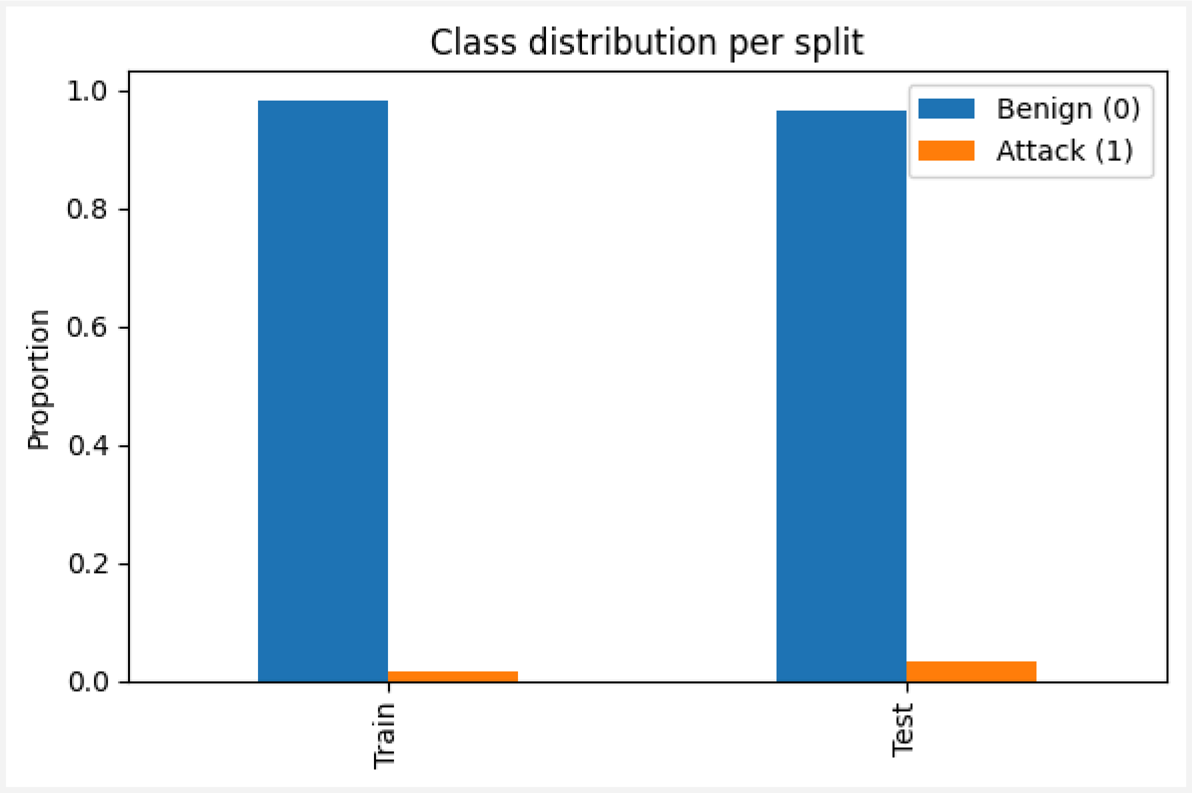
By default, the decision threshold is set to 0.5, providing standard class separation; however, it can be adjusted to address class imbalance or specific operational requirements.

Before applying the model, the dataset must be split into training and testing subsets. This is a fundamental step to ensure the model is trained and evaluated on separate data, allowing performance to be assessed on unseen observations, supporting generalisation and preventing overfitting.

The resulting data split forms the basis for subsequent model training and testing.



↓

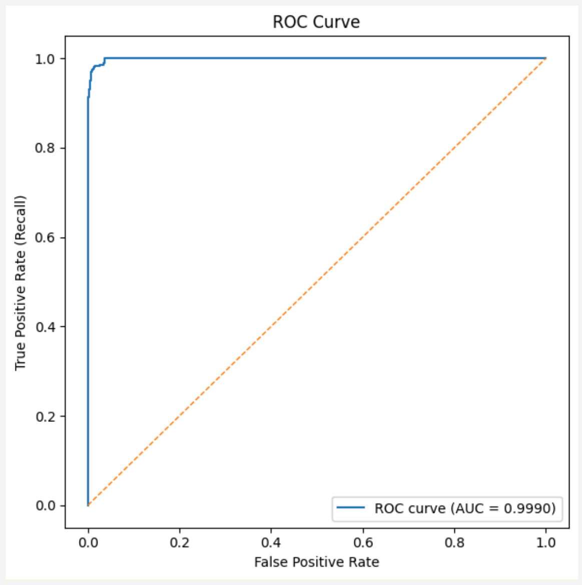
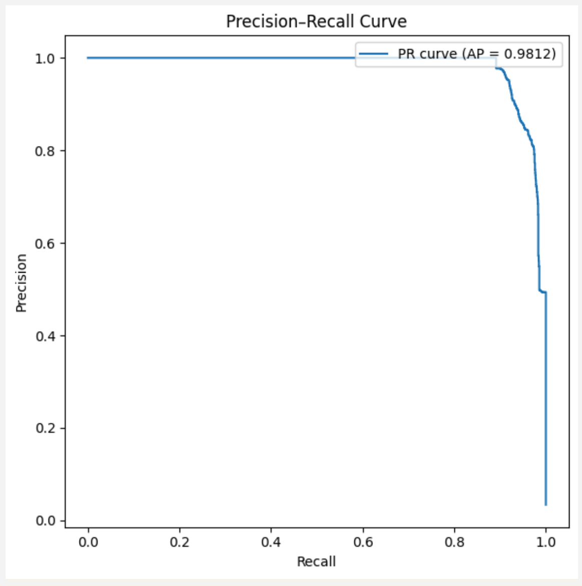


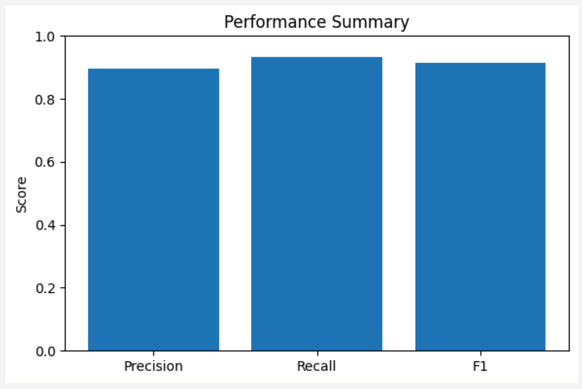
While confirming that the model functions correctly within the pipeline is necessary, it is insufficient to justify its use. A performance analysis is required to quantitatively assess model effectiveness.

Evaluation

To justify my choice and analyse the quality of the predicted findings primary evaluation metrics will be:

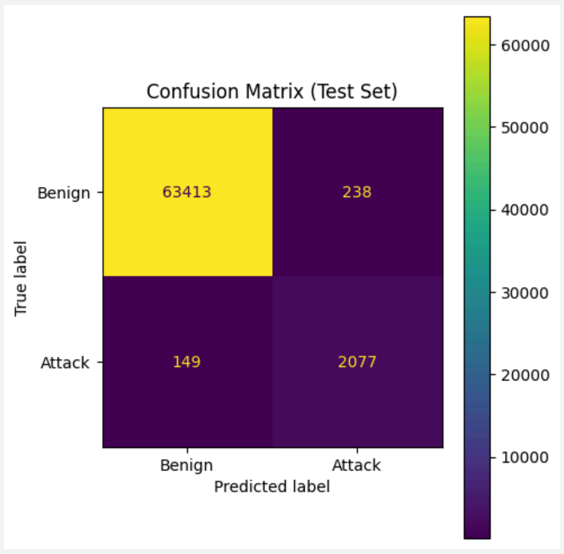
Recall	
Recall evaluates the model's ability to correctly detect abnormal instances present in the dataset.	

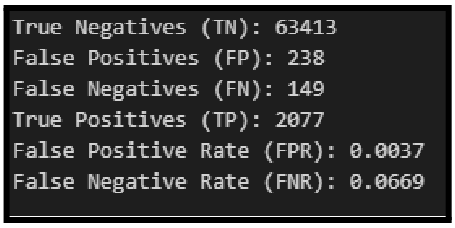
The output shows high recall, indicating the model identifies most abnormal cases, which is critical given the higher cost of missed events.	
Precision–Recall AUC	
Precision measures the proportion of correctly identified abnormal instances among all predicted abnormal instances.	
The output indicates a good generalization and reliable precision–recall trade-offs.	

The summary indicates balanced performance: high precision limits false alarms, high recall captures most anomalies, and the F1-score confirms this balance, indicating reliable anomaly detection.	

Precision	Control False Alarm	**> 0.89**
Recall	Detection Rate	**> 0.93**
F1	Performance summary	**> 0.91**
Precision Recall AUC	Model Performance	**> 0.98**

Overall, the results demonstrate strong model performance; however, accuracy alone can be insufficient (University of the West of England). A confusion matrix is therefore used to explicitly quantify false alarms and missed detections, clarifying the distribution of correct predictions and error types.





```
True Negatives (TN): 63413
False Positives (FP): 238
False Negatives (FN): 149
True Positives (TP): 2077
False Positive Rate (FPR): 0.0037
False Negative Rate (FNR): 0.0669
```

To reduce false positives two primary strategies could be implemented:

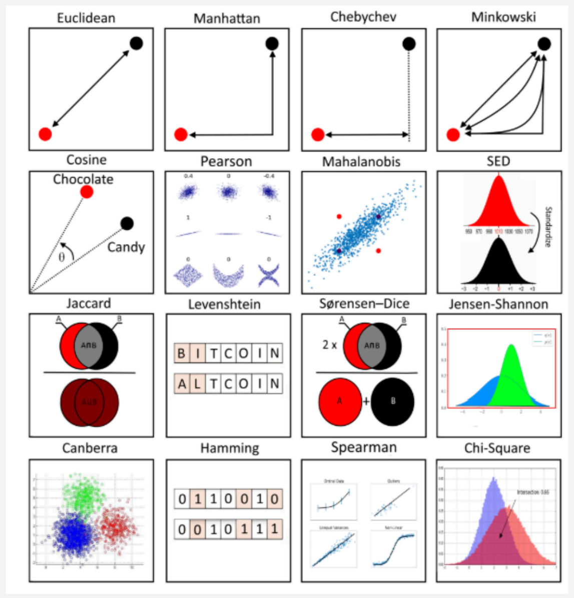
Threshold Tuning	Adjusting the decision threshold offers high impact with minimal implementation effort.

	Slightly increasing the threshold for classifying an instance as abnormal can reduce unnecessary alerts while preserving most of the model’s detection capability.
Ridge Regularization	Applying Ridge (L2) regularization can further improve performance by stabilizing coefficient estimates. By penalizing large weights, it reduces sensitivity to noise and feature correlation, improving generalization and yielding more robust classification outcomes.

Finally, overfitting is unlikely to be a major concern given the large, normalized dataset, which exhibits robust behaviour even with highly correlated features. Under these conditions, the model achieves strong classification performance, indicating good generalization capability.

Comparative analysis

An alternative approach considered but ultimately discarded was k-nearest neighbors (k-NN). There are at least seventeen types of similarity and dissimilarity measures used in data science (Aggarwal, 2020), and some of the most common have been extracted from the article and presented below:



Similarity checking has the objective of measuring distances between observations, assuming that the distance between two points increases their difference, while proximity to a distance of zero implies they are identical.

1. Distance: Measures how **far** apart two points are. Larger values mean more differences

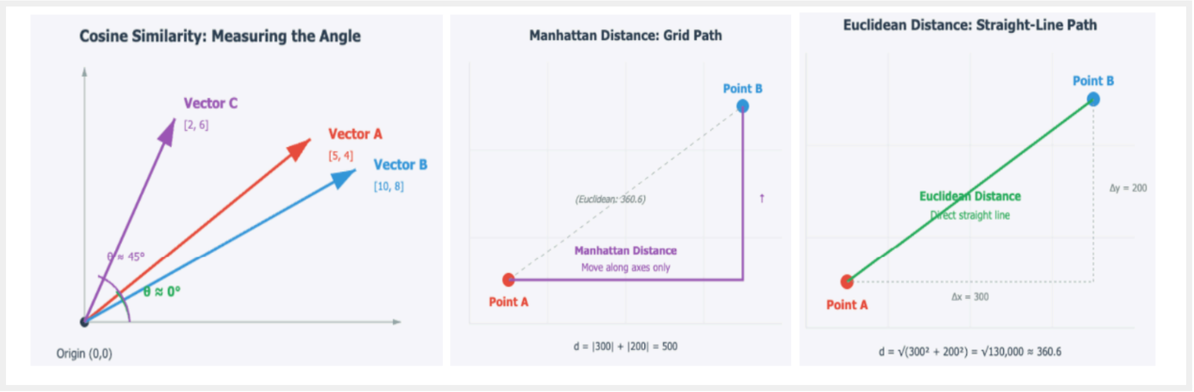
Distance = 0	Identical

2. Similarity: Measures how **alike** two points are. larger values indicate greater similarity.

Similarity = 1	Identical (in a normalized range 0,1)

The most fundamental distance measures are:

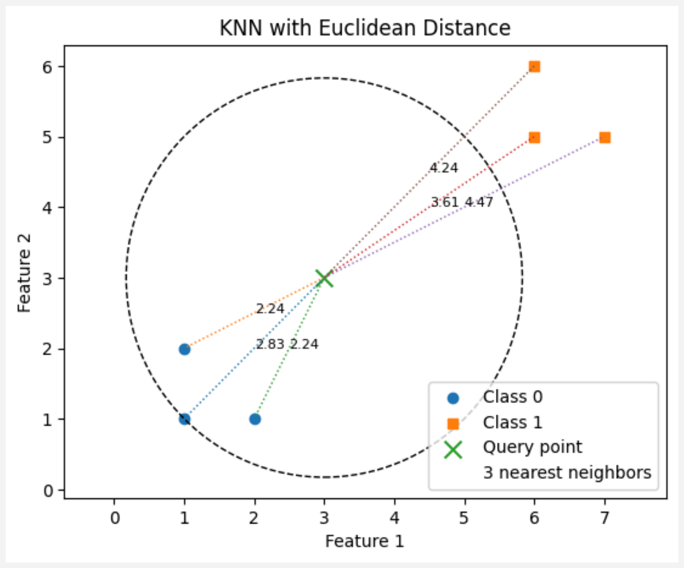
- Euclidean: straight-line path between point A and point B
- Manhattan: grid-path distance
- Cosine: measure of the angle between vectors



(University of the West of England)

When choosing **k**, the algorithm considers the nearest neighbours.

For example, with **k = 3** and a query point at the centre, the three nearest instances are selected as neighbours. Distances are computed to measure similarity, with smaller distances indicating greater similarity. A majority voting scheme is then applied to assign a class label to future observations, as shown below.



This method is effective for small to medium-sized problems with complex relationships, such as medical diagnosis, but is computationally expensive at scale. Applied to the network overload dataset with over 9 million observations, it also shows reduced interpretability and less stable behaviour compared to logistic regression.

The learning capability of KNN is therefore intrinsically limited because it does not estimate model parameters (weights) and is highly sensitive to irrelevant features, a limitation that becomes particularly pronounced in large-scale measurement.

A further advantage of logistic regression is its **probabilistic** output, which enables effective **hyperparameter adjustments**, such as decision threshold tuning, to manage false positive and false negative rates.

When applied to this dataset, k-nearest neighbors offer limited tuning flexibility.

It does not natively produce calibrated probabilistic outputs and instead relies on majority voting, restricting fine-grained control over the decision boundary and reducing the interpretability of ROC–AUC–based performance evaluation.

Moreover, as a lazy learning algorithm, k-nearest neighbors must store and repeatedly access the entire training set at inference time, resulting in higher computational cost and lower portability. In contrast, once logistic regression has learned its parameters, it can perform repeated inference using only the trained model, without requiring access to the training data.

In conclusion, given the dataset characteristics, k-nearest neighbors would likely yield highly flexible and irregular decision boundaries, reducing global interpretability relative to logistic regression.

REFERENCE LIST

Aggarwal, C.C. (2020) *17 types of similarity and dissimilarity measures used in data science*. *Towards Data Science*, 30 March. Available at: https://towardsdatascience.com/17-types-of-similarity-and-dissimilarity-measures-used-in-data-science-3eb914d2681/

GeeksforGeeks (2025) Feature engineering: scaling, normalization and standardization. GeeksforGeeks, 17 September. Available at: ttps://www.geeksforgeeks.org/machine-learning/feature-engineering-scaling-normalization-and-standardization/

Khalil, M. (2025) DDoS attack statistics: How attacks are escalating worldwide. Deepstrike.io, 24 June. Available at: https://deepstrike.io/blog/ddos-attack-statistics

LinkedIn Learning (2026) Machine Learning with Python: Logistic Regression – Classifying data with logistic regression. LinkedIn Learning. Available at: https://www.linkedin.com/learning/machine-learning-with-python-logistic-regression/classifying-data-with-logistic-regression

Mengual-Macénlle, N., Marcos, P.J., Golpe, R. and González-Rivas, D. (2015) 'Multivariate analysis in thoracic research', Journal of Thoracic Disease, 7(3), pp. E2–E6. https://doi.org/10.3978/j.issn.2072-1439.2015.01.43
. Available at: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4387392/

National Cyber Security Centre (no date) Denial of Service (DoS) guidance collection. Available at: https://www.ncsc.gov.uk/collection/denial-service-dos-guidance-collection

Rakić, A. (2024) Cyber Attack Dataset: ARP, SYN, PING Flood. Kaggle. Available at: https://www.kaggle.com/datasets/aleksandarraki/cyber-attack-dataset-arp-syn-ping-flood

Statistics How To (2026) Bivariate analysis definition & example. Available at: https://www.statisticshowto.com/probability-and-statistics/statistics-definitions/bivariate-analysis/

University of the West of England (UWE) (2025) Machine learning and predictive analytics. Internal module material, Faculty of Data Science, University of the West of England.