Relevant Alarms Detection

Relevant Alarm Detection Using Machine Learning for Alarm Reduction Systems

Dataset

The dataset used in this study has been collected from the ACE Fault Management platform. We are utilizing historical data to analyze and develop a machine learning model capable of predicting whether an alarm is relevant or irrelevant.

Irrelevant alarms are those considered to be transient alarms.

According to the **Fault Management Documentation**, transient (irrelevant) alarms can be defined as follows:

- **Maintenance Mode**: Alarms triggered when systems are in maintenance mode.
- **Low Priority Alarms**: Alarms classified with low priority levels.
- **Repetitive Alarms**: Alarms of the same type from the same Source or Site that occur more than a specified number of times within a defined period (e.g., more than three times in 24 hours) should be labeled as irrelevant.
- **Maintenance Period Alarms**: Alarms that occur during a scheduled maintenance window should be labeled as irrelevant.

Since our dataset does not come pre-labeled with **relevant** and **irrelevant** alarms, we need to employ a heuristic to label the dataset.

We will label an alarm as irrelevant if it is cleared within a short period of time, denoted as n. The value of n should ideally be chosen by a domain expert. For the purpose of this study, we will use $n = 7$ minutes.

The dataset contains **32,326 rows** representing alarm events and **28 columns** representing various features.

Features in the Dataset:

- Severity
- Status
- Alarm Name
- Technical ID
- Site Name
- Site ID
- Region
- Zone
- NE Name
- Source
- Cell Name
- NE Type
- Vendor
- Technology
- Location Information
- First Occurrence
- Last Occurrence
- Event Time
- CSN
- Clearance Time
- Clearance User
- Acknowledgement Time
- Acknowledgement User
- FM Receive Time
- Is Active
- Ticket ID
- Alarm Category
- Parent Node
- Target

After labeling the alarms as relevant or irrelevant according to the previously outlined rule, an additional column **Target** was added with binary values **relevant** or **irrelevant**.

Columns **Status**, **Cell Name**, and **Ticket ID** contain no data and will be removed from the study as they provide no information. Similarly, the **Vendor** column contains only one value, "Huawei," and will be excluded for lack of variability.

The columns **Region** and **Zone** contain only "R0" and "Z0," respectively. Since they do not offer useful information beyond these constant values, **Region** will be dropped from the study.

Data Exploration

Datetime Features Analysis

Analysis of the **First Occurrence** and **Last Occurrence** columns revealed alarms spanning an unrealistic range from **January 1, 1990**, to **February 4, 2037**. Further investigation uncovered three events with a **First Occurrence** in the year 2037, deemed outliers, and subsequently removed from the dataset.

CSN	First Occurence	Severity	Alarm Name	NE Type
326763515	2037-04-02 03:08:24	Critical	ETH_LOS	Optix RTN 950
326685958	2037-04-02 03:07:28	Cleared	SWDL_INPROCESS	Optix RTN 950
326763279	2037-04-02 03:05:00	Warning	SWDL_INPROCESS	Optix RTN 950

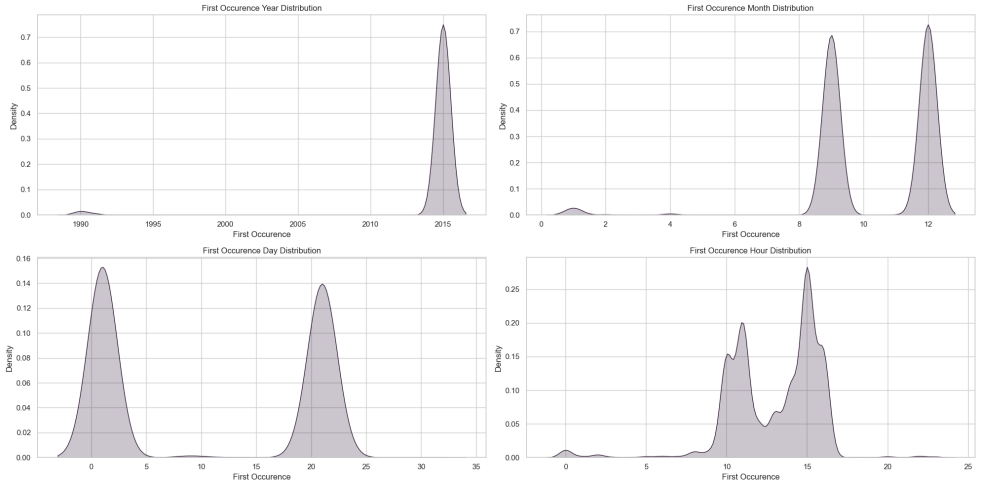
Further exploration revealed missing values in the following columns:

Columns	Missing values
Clearance Time	10456
Clearance User	10456
Acknoledgement Time	32299
Acknoledgement User	32299

Notably, the missing values in **Clearance Time** corresponded with identical missing values in **Clearance User**, and similarly for the **Acknowledgement** columns.

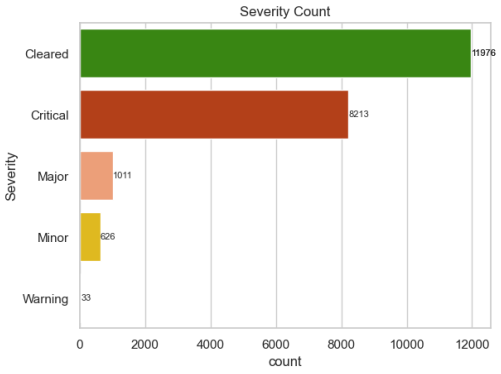
Additionally, there were 25 rows where both **Acknowledgement Time** and **Clearance Time** were null simultaneously. To maintain data integrity and facilitate labeling, rows with null values in **Clearance Time** were dropped.

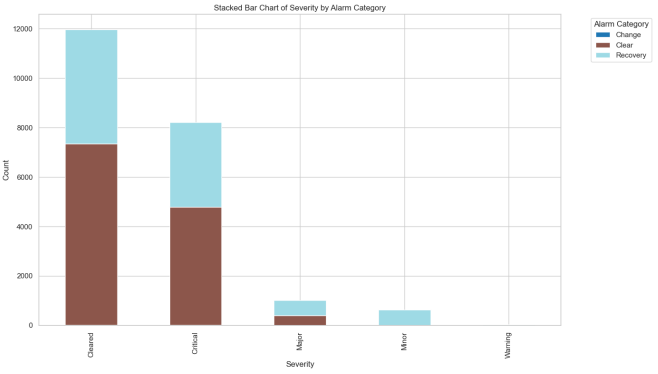
First Occurence Distributions



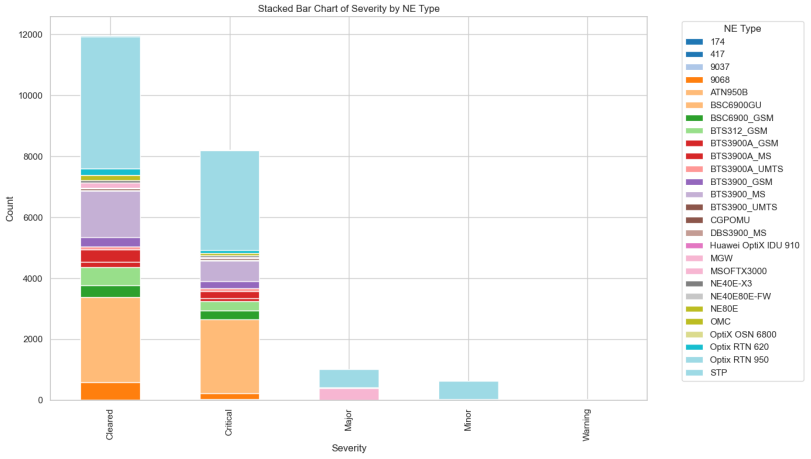
By examining this plot, we can conclude that most alarms first occurred in **2015**, with notable peaks in **September** and **December**. Specifically, the first occurrences are concentrated on the **1st**, **2nd**, **20th**, and **21st** days of these months, particularly between **10:00 AM** and **4:00 PM**.

Categorical Features analysis

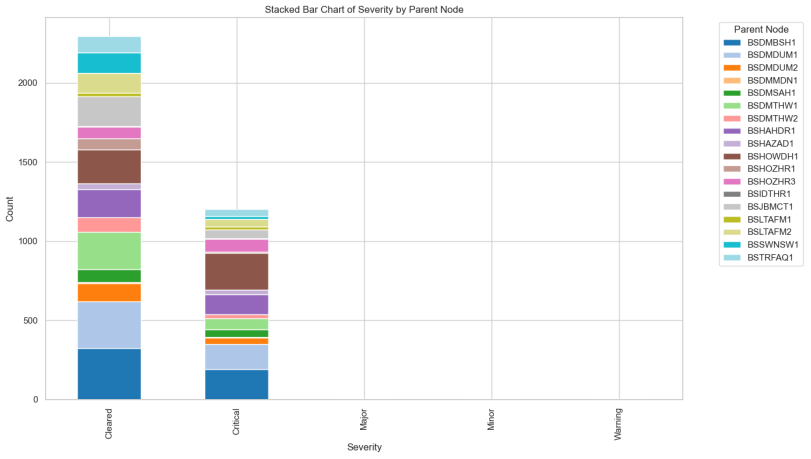




NE Type	Count
Optix RTN 950	8819
BSC6900GU	5212
BTS3900_MS	2176
BTS312_GSM	897
9068	781
BSC6900_GSM	675



Parent Node	Count
BSDMBSH1	513
BSDMDUM1	456
BSHOWDH1	447
BSDMTHW1	304
BSHAHDR1	303
BSJBMCT1	241



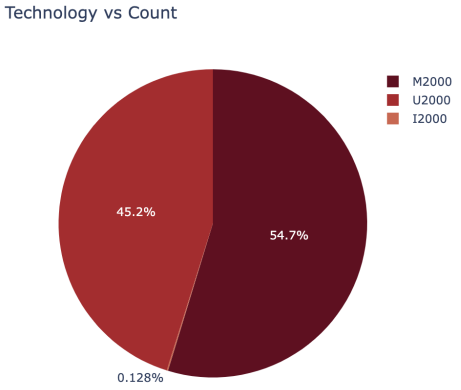


Figure - Technology distribution

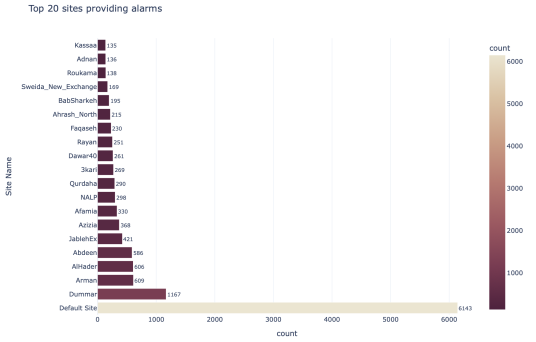


Figure - Top 20 sites providing alarms

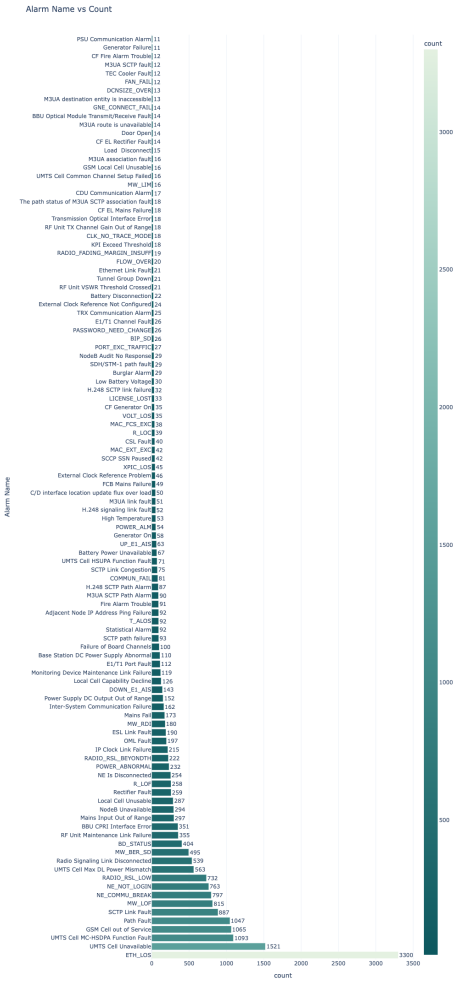


Figure - Alarm Names distribution

In this figure, it's evident that the site with the highest number of alarms is labeled as the "Default Site". However, it's essential to discuss the nature of the sites and whether they are fixed categories in this column.

Feature Engineering

In this section, we aim to derive additional insights from the original features (columns) and create new features accordingly.

An attempt was made to calculate the time taken to detect alarms by computing the difference between the Event Time and First Occurrence columns. However, this resulted in a value of 0 for each event, indicating that the system detects alarms in real-time. Consequently, the **Event Time** column will be dropped from the study as it duplicates information already present in the **First Occurrence** column.

Furthermore, we propose calculating the **Clearance Duration Time** as a feature to aid in labeling alarm events as **relevant** or **irrelevant**. As previously discussed, if the Clearance Duration Time for an event is less than $n = 7$ *minutes*, it will be labeled as **irrelevant**; otherwise, it will be labeled as **relevant**.

The **Clearance Duration Time** is calculated by determining the difference between the **Clearance Time** and **First Occurrence** columns.

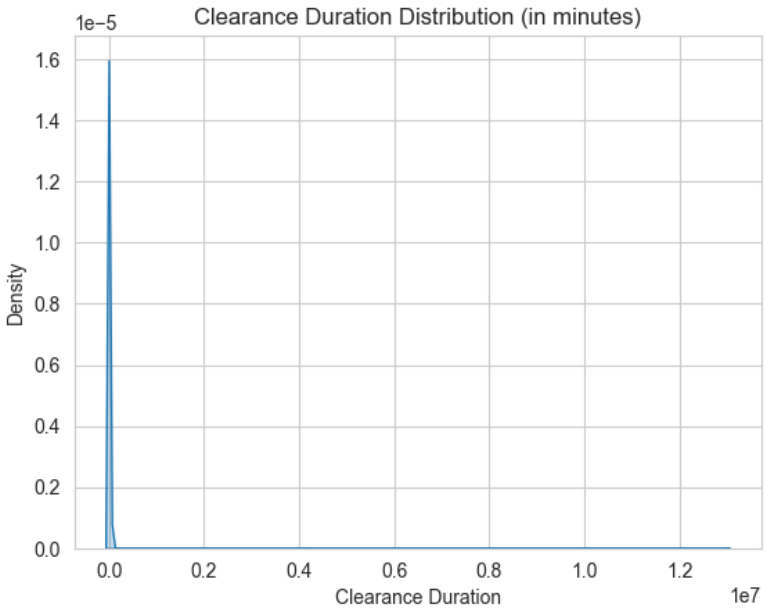
During this calculation, anomalies were identified, with six events exhibiting a negative **Clearance Duration Time**. This indicates that these alarms were cleared before they occurred, a scenario that is logically impossible.

CSN	Severity	First Occurence	Clearance Time	Alarm Name	NE Type	Alarm Category
326764282	Major	1990-01-01 04:02:18	1990-01-01 02:04:15	POWER_ABNORMAL	Optix RTN 950	Recovery
326764281	Major	1990-01-01 04:02:18	1990-01-01 02:04:15	POWER_ABNORMAL	Optix RTN 950	Recovery
215729941	Critical	1990-01-01 00:05:07	1990-01-01 00:03:58	ETH_LOS	Optix RTN 950	Recovery
215718959	Critical	1990-01-01 00:25:07	1990-01-01 00:04:13	ETH_LOS	Optix RTN 950	Recovery
215717587	Critical	1990-01-01 05:15:26	1990-01-01 00:04:59	BUS_ERR	Optix RTN 950	Recovery
215716711	Major	1990-01-01 05:05:23	1990-01-01 00:04:59	BD_STATUS	Optix RTN 950	Recovery

To ensure data integrity, these anomalous events will be investigated further and potentially removed from the dataset.

From the plot, it's evident that the Clearance Duration varies between 0 minutes and 1.3×10^7 minutes *(approximately 24.7 years)*

Clearance User	count
<SYSTEM>	21863
saurabh	1



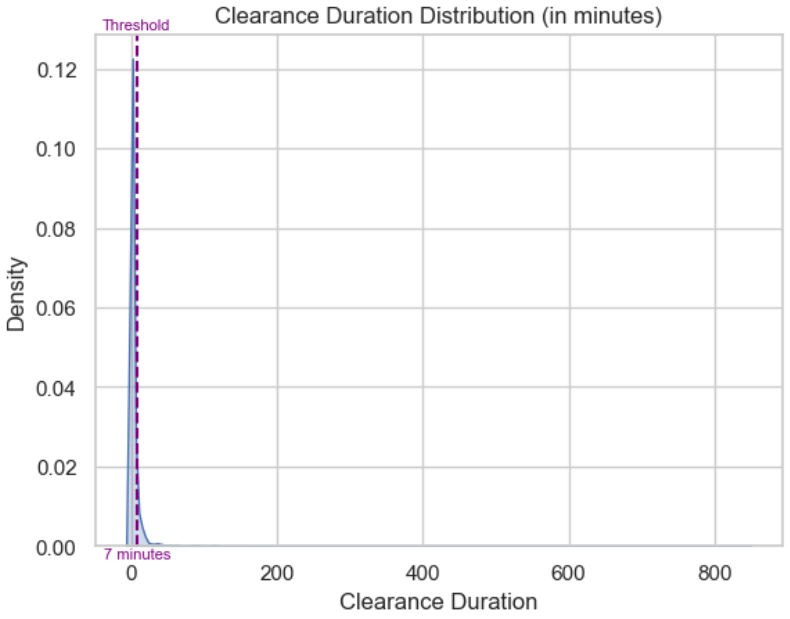
The plot visually confirms the presence of outliers in the dataset.

CSN	Severity	First Ocurrence	Clearance Duration (years)	NE Type	Alarm Category
136003519	Critical	2015-12-01 16:34:14	**7.994**	NE Is Disconnected	OMC
215718927	Major	1991-01-14 22:51:15	**24.683**	POWER_ABNORMAL	Optix RTN 950
215718921	Major	1991-01-14 22:51:06	**24.683**	PASSWORD_NEED_CHANGE	Optix RTN 950
215718920	Minor	1991-01-14 22:55:21	**24.683**	CLK_NO_TRACE_MODE	Optix RTN 950
215716520	Major	1991-01-14 22:42:35	**24.683**	POWER_ABNORMAL	Optix RTN 950

These outliers will be removed from the study to avoid biasing the data, which will lead to better results and more accurate predictions when using machine learning models.

Statistics

events	21859
mean	3.690100
std	17.995587
min	0
25%	0
50%	0
75%	0.816667
max	842.9333

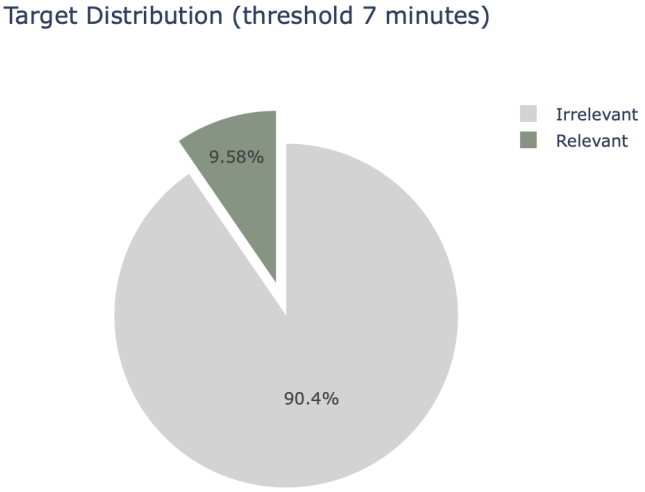


After removing the outliers, the Clearance Duration now ranges between 0 minutes and 800 minutes (13.3 hours), which is more logical.

Using a threshold of 7 minutes to label our dataset, we have the following distribution:

Relevant	2093
Irrelevant	19764

This dataset is clearly imbalanced, making it challenging for machine learning models to accurately detect irrelevant alarms.



To address this imbalance, we need to collect more "Relevant" alarms. If that is not feasible, we can employ advanced algorithms to generate synthetic "Relevant" alarms. *(This approach will be discussed in the following sections.)*

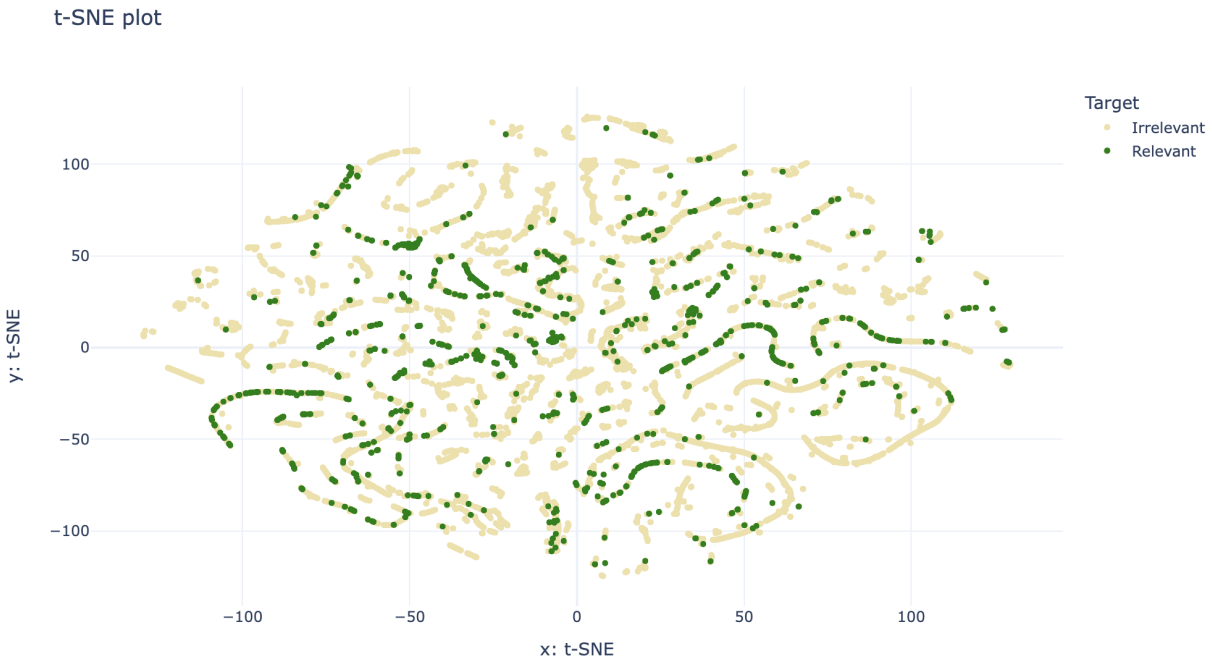
Modeling Approach

Data Processing

We encode categorical variables and transform datetime features into timestamps to improve the performance of machine learning models.

Dimensionality Reduction

We utilize the t-SNE algorithm to perform dimensionality reduction, allowing us to visualize the alarm events in a 2D space. The objective of this algorithm is to project the features (originally in higher dimensions) into a 2D space for better visualization.



In this plot, each point represents an alarm event. Ideally, closely clustered data points with the same color indicate that the machine learning model will find it easier to classify them. However, in this plot, it is evident that distinguishing relevant alarms (green points) from irrelevant ones is challenging. This difficulty may arise from a lack of distinguishing features in the dataset.

Machine Learning Modeling

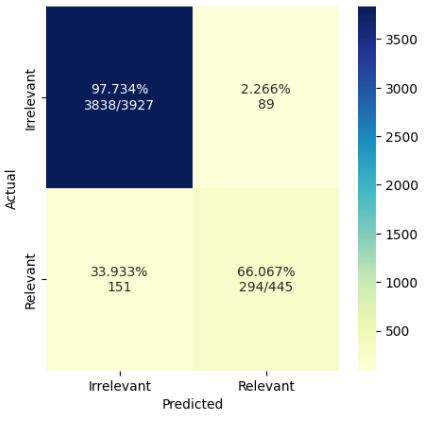
Baseline Model

For data splitting, we partition the dataset into 80% training data and 20% testing data. The testing set is used as unseen data to evaluate the model after training. We compare the model's predictions against actual values to derive performance scores.

We choose a Random Forest Classifier as the baseline model. Multiple score metrics are used to evaluate the model's performance.

	precision	recall	f1-score	support
Irrelevant	0.96	0.98	0.97	3927
Relevant	0.77	0.66	0.71	445
accuracy			0.95	4372
macro avg	0.86	0.82	0.84	4372
weighted avg	0.94	0.95	0.94	4372

Model performance



Confusion matrix

Explanation of Metrics:

- **Accuracy:** Accuracy shows how often the system is right overall, taking into account both relevant and irrelevant alarms.

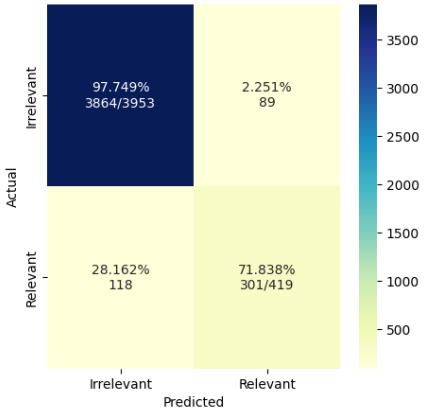
- **Precision:** It tells us how accurate the system is when it flags an alarm as relevant. It measures the percentage of alarms flagged as relevant that were actually relevant.
- **Recall (Sensivity):** It measures how good the system is at catching all the relevant alarms. It tells us the percentage of actual relevant alarms that the system successfully identifies.
  In the context of alarm detection,
  having a high recall means the system rarely misses an important alarm. This is crucial because missing a relevant alarm could potentially lead to unaddressed critical situations.
- **F1-score:** The F1-Score is a way to measure the system's accuracy by balancing recall and precision. It is particularly useful when you want a single metric that reflects both the system's ability to catch all relevant alarms and its accuracy in identifying them.

These metrics show that the classifier is highly effective at identifying **irrelevant alarms** (high precision and recall), but struggles somewhat with identifying **relevant alarms**, with a lower recall and F1-score, indicating missed relevant alarms and a balance between precision and recall.

For this project, we aim to maximize the **recall metric** because accurately identifying all irrelevant alarms is crucial for reducing unnecessary alerts in the network.

Handling Imbalanced Data

Given the imbalanced nature of the dataset, we use the Stratified Shuffle Split method for data splitting. This method preserves the original distribution of classes, ensuring that both training and testing sets reflect the class imbalance.



Confusion matrix

Model performance with Stratified Shuffle Split

	precision	recall	f1-score	support
Irrelevant	0.97	0.98	0.97	3953
Relevant	0.77	0.72	0.74	419
accuracy			0.95	4372
macro avg	0.87	0.85	0.86	4372
weighted avg	0.95	0.95	0.95	4372

Model performance

The confusion matrix indicates **improved scores** compared to our baseline model simply by using the stratified split method.

Fine-tuning

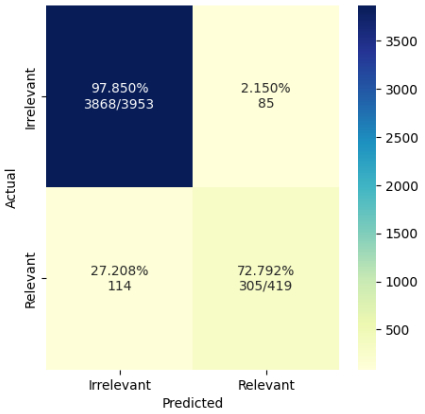
We will fine-tune the model using Bayesian optimization to obtain the best possible results from our baseline model. This method optimizes hyperparameters efficiently, potentially enhancing model performance further.

By fine-tuning the model, we achieved slightly better results with an overall.

Model performance after fine-tuning

	precision	recall	f1-score	support
Irrelevant	0.97	0.98	0.97	3953
Relevant	0.78	0.73	0.75	419
accuracy			0.95	4372
macro avg	0.88	0.85	0.86	4372
weighted avg	0.95	0.95	0.95	4372

Model performance



Confusion matrix

- *True Negatives (TN)*:

  3868/*3953* (Irrelevant correctly classified as Irrelevant)

- *False Positives (FP):* **85** (Irrelevant incorrectly classified as Relevant)
- *False Negatives (FN):* **114** (Relevant incorrectly classified as Irrelevant)
- *True Positives (TP):* **305**/*419* (Relevant correctly classified as Relevant)

Oversampling Model

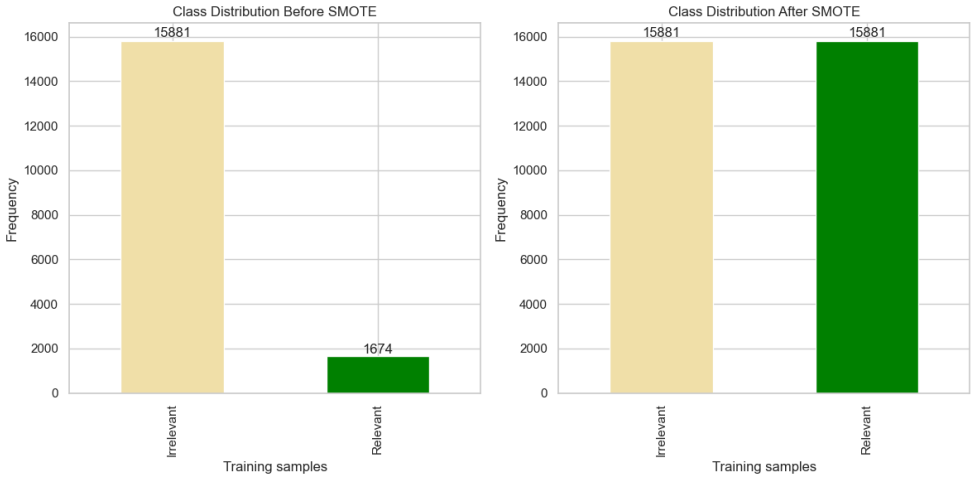
In this section, we address class imbalance. Despite fine-tuning the baseline model, we did not achieve significant improvements, indicating that the model struggles to differentiate between relevant and irrelevant alarms. This difficulty may stem from a lack of features or relevant samples.

Random Forest models typically perform well without feature scaling, but we will include scaling in this section for completeness.

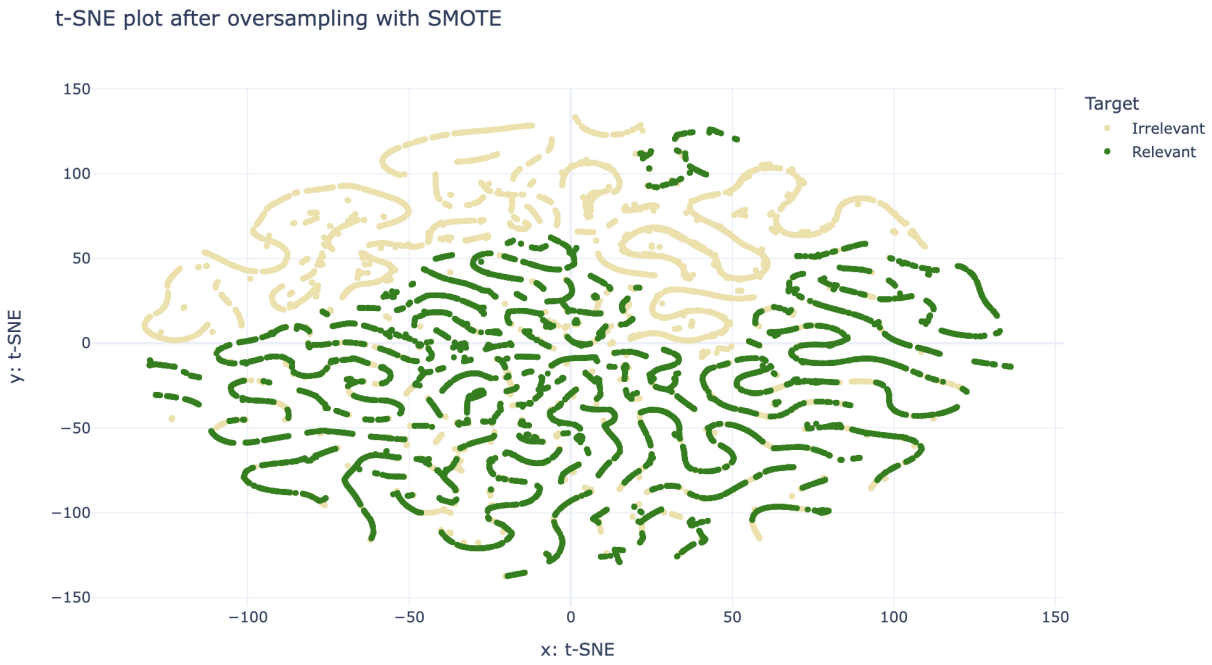
To improve our machine learning model, we need more relevant samples. Since our historical data is limited, we will generate these samples using an algorithm called SMOTE (Synthetic Minority Over-sampling Technique).

SMOTE generates synthetic samples from the minority class (Relevant) by interpolating between existing samples. This method effectively balances the training dataset by increasing the number of relevant alarms.

We apply SMOTE only to the training data to ensure the model is trained on a balanced dataset, while the test set remains untouched for unbiased evaluation.



Class distributions before and after oversampling

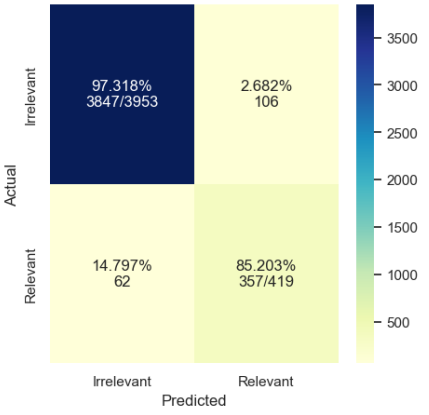


After oversampling and feature scaling, the t-SNE plot shows improved clustering, indicating better separability between classes.

Model performance after oversampling

	precision	recall	f1-score	support
Irrelevant	0.98	0.97	0.98	3953
Relevant	0.77	0.85	0.81	419
accuracy			0.96	4372
macro avg	0.88	0.91	0.89	4372
weighted avg	0.96	0.96	0.96	4372

Model performance



Confusion matrix

Overall, we achieved an F1-score, Recall, Precision, and Accuracy of 96%, which is a substantial improvement.

- **True Negatives (TN): 3847***/3953* (Irrelevant correctly classified as Irrelevant)
- **False Positives (FP): 62** (Irrelevant incorrectly classified as Relevant)
- **False Negatives (FN): 106** (Relevant incorrectly classified as Irrelevant)
- **True Positives (TP): 357***/419* (Relevant correctly classified as Relevant)

Feature Importance

Understanding the importance of each feature in our model is crucial for interpreting its decisions and improving its performance. The feature importance values extracted from our Random Forest classifier highlight which features contribute most significantly to predicting whether an alarm is relevant or irrelevant. Here are the feature importance scores:

Key Insights

Feature	Importance
Severity	0.381442
Technical ID	0.234363
FM Receive Time	0.170224
First Occurrence	0.156505
NE Type	0.049368
Technology	0.004361
Alarm Category	0.003737

1. **Severity (38.1%)**:
    - The most critical feature, indicating that the model heavily relies on the severity level to determine alarm relevance.
2. **Technical ID (23.4%)**:
    - The second most important feature, likely providing specific identifiers related to the equipment or system generating the alarm.
3. **FM Receive Time (17.0%)**:
    - This feature helps identify temporal patterns, contributing significantly to the model's decision-making.
4. **First Occurrence (15.7%)**:
    - The timestamp of the alarm's first occurrence is crucial for understanding temporal aspects, aiding in distinguishing relevant alarms.
5. **NE Type (4.9%)**:
    - Moderately important, indicating different types of network elements have varying probabilities of generating relevant alarms.
6. **Technology (0.4%)**:
    - Provides some context but is not a strong determinant of alarm relevance.
7. **Alarm Category (0.4%)**:
    - Has minimal impact, possibly due to overlapping categories or general nature of categorization.

Conclusion

In this study, we developed a machine learning model to detect relevant alarms for an alarm reduction system. We began by exploring and preprocessing the dataset, addressing issues such as missing values, irrelevant columns, and outliers. Through feature engineering, we created new features to better capture the characteristics of the alarms.

Initially, we faced challenges due to class imbalance, with a significantly higher number of irrelevant alarms compared to relevant ones. This imbalance hindered the model's performance, even after fine-tuning a Random Forest classifier.

To tackle this, we employed the SMOTE algorithm to oversample the minority class (relevant alarms). This approach helped balance the training data and significantly improved the model's ability to distinguish between relevant and irrelevant alarms. The t-SNE visualization post-oversampling showed better clustering, indicating improved separability of the classes.

Our final model, evaluated on an untouched test set, achieved an F1-score, Recall, Precision, and Accuracy of 96%. These results indicate that the model performs well in correctly identifying both relevant and irrelevant alarms, making it a reliable tool for alarm management.

Future Improvements

While the current model shows promising results, there is room for further enhancement. One key area is the quality of the labeled data. Collaborating with domain experts to obtain accurately labeled relevant alarms would enrich the dataset, providing more nuanced information for the model to learn from. This expert-labeled data would likely improve the model's performance even further, ensuring more precise alarm detection and ultimately reducing the operational burden of managing false alarms.

Overall, this study demonstrates the potential of machine learning in enhancing alarm management systems, offering a robust framework that can be built upon with richer data and further refinements.