CyberMl

TRAORE Djibril
BRAIGHITH Sofian
DUCROCQ Tanguy
BOURGEOIS Damien

March 2025

Contents

1 Introduction

L'objectif de ce projet de MLSecu est de faire de la détection d'anomalies. Pour cela, nous avons travaillé sur le jeu de données SWaT (Secure Water Treatment), jeu de données conçu pour l'étude de systèmes industriels. Nous avons fait une première étape de prétraitement des données, puis dans un 2ème temps nous avons entraîné des modèles supervisés et non supervisés sur les données traités. Pour finir, nous avons fait des visualisation (anomalies détectées, matrices de confusion) pour évaluer les performances des différents modèles.

2 Présentation du dataset

Le jeu de données SWaT provient d'une station expérimentale de traitement de l'eau, conçue à des fins de recherche en cybersécurité des systèmes industriels. Cette installation reproduit à petite échelle une véritable station de traitement d'eau, avec une capacité de 5 gallons par minute, structurée en six étapes successives de purification. Chaque étape fait intervenir différents capteurs (niveaux, pressions, débits, analyses chimiques) et actionneurs (pompes, vannes motorisées, lampes UV), pilotés par des automates programmables industriels (API) reliés à un système de supervision (SCADA).

La collecte de données s'est déroulée en continu sur 11 jours, à raison d'une mesure par seconde. Durant les sept premiers jours, la station a fonctionné normalement, sans aucune attaque, afin de constituer une base de référence représentative du comportement attendu. Les quatre jours suivants, 36 attaques variées ont été déclenchées manuellement par les chercheurs, ciblant un ou plusieurs capteurs ou actionneurs, dans une ou plusieurs étapes du processus. Ces attaques, soigneusement documentées, ont pour objectif de simuler des scénarios réalistes de sabotage ou de dysfonctionnement volontaire.

Le jeu de données comprend deux volets principaux:

- les données physiques, issues de 51 capteurs et actionneurs, permettant de suivre l'évolution du système en temps réel.
- les données réseau, capturant les échanges entre les API et le système SCADA via le protocole Modbus.

Toutes les données ont été étiquetées manuellement selon qu'elles relèvent d'un fonctionnement normal ou d'un comportement anormal induit par une attaque. Cette caractéristique en fait une ressource particulièrement adaptée à la détection d'anomalies par apprentissage automatique, notamment en milieu industriel.

Dans le cadre de ce projet, nous avons uniquement utilisé les données physiques et non les données réseau.

2.1 Data cleaning

Avant de commencer quoi que ce soit, la première étape est de nettoyer le dataset. Cette étape consiste à rendre le dataset utilisable selon ce que l'on souhaite démontrer. Dans notre cas, cela se divise en quatre étapes.

2.1.1 Suppression des NaN

Les modèles de machine learning ne peuvent pas gérer directement les NaN, ce qui peut entraîner des erreurs ou une mauvaise performance du modèle. Les données manquantes peuvent fausser les résultats en perturbant l'apprentissage et en diminuant la capacité du modèle à détecter des anomalies importantes, comme des intrusions ou des comportements anormaux. Pour éviter cela, il est nécessaire de supprimer les lignes ou colonnes contenant des NaN ou d'imputer ces valeurs avec des techniques appropriées (moyenne, médiane, etc.), garantissant ainsi la cohérence et la précision des prédictions. Ici, nous avons décidé de les supprimer.

```
# Drop NaNs
data.dropna(inplace=True)
data.drop(columns=['GMT +0'], inplace=True)
```

2.1.2 Suppression des colonnes inutiles

Les colonnes contenant une seule valeur unique ne fournissent aucune information discriminante, car elles ne permettent pas de différencier les observations et ne contribuent donc pas à l'apprentissage du modèle. Laisser ces colonnes dans le dataset pourrait introduire du bruit inutile et nuire à la performance du modèle. En les supprimant, on simplifie le modèle et on évite l'overfitting, améliorant ainsi la capacité du modèle à détecter des anomalies pertinentes.

```
# Remove columns with unique values
useless_columns = [col for col in data.columns if data[col].nunique() == 1]
data.drop(columns=useless_columns, inplace=True)
print(f"Removed {len(useless_columns)} useless columns")
```

2.1.3 Suppression des valeurs non numériques

Certains modèles de machine learning ne peuvent pas traiter directement les valeurs non numériques. Ces valeurs peuvent perturber l'apprentissage en introduisant des incohérences et en réduisant la capacité du modèle à identifier des patterns ou des anomalies. En éliminant ces valeurs ou en les transformant en un format numérique approprié, on assure la compatibilité des données avec les algorithmes et on optimise les performances du modèle.

```
# drop non numerical columns
data.drop(data.select_dtypes(include=['object']).columns, axis=1, inplace=
    True)
```

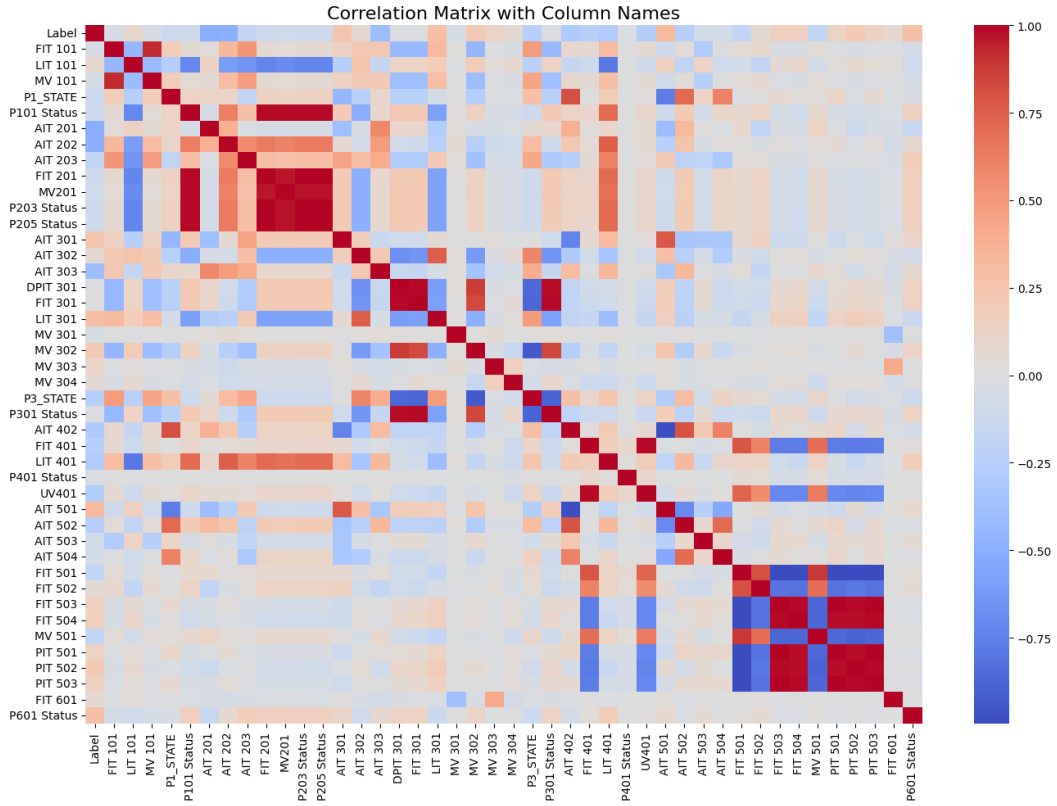
2.1.4 Normalisation

La normalisation de données consiste à mettre les différentes variables du dataset sur une échelle comparable, généralement en les ramenant à une plage standard, dans notre cas [0, 1]. Sans normalisation, les variables avec des plages de valeurs très larges peuvent dominer l'apprentissage du modèle, rendant difficile la détection d'anomalies subtiles. La normalisation permet d'éviter ce problème et d'améliorer la performance des algorithmes. Elle assure ainsi que chaque feature contribue de manière égale à l'analyse des anomalies.

```
# Normalize data
scaler = MinMaxScaler()
data[data.columns] = scaler.fit_transform(data[data.columns])
```

2.2 Data visualisation

Afin de mieux visualiser les données, nous avons décidé de faire une matrice de corrélation. Celle-ci permet de mieux comprendre les relations entre les variables et d'optimiser la sélection des caractéristiques pour améliorer la performance et l'efficacité de nos modèles de détection d'anomalies.



Après avoir nettoyé les données, voici à quoi ressemble la matrice de corrélation. On distingue facilement que la majorité des variables ne sont pas fortement reliées, ce qui nous permet de conclure que le dataset est prêt à être utilisé. Dans le cas contraire, il aurait fallu supprimer les valeurs qui sont trop liées entre elles afin de limiter la redondance.

3 Évaluation du modèle

L'évaluation du modèle permet de mesurer la capacité du modèle à distinguer correctement les comportements normaux des activités malveillantes. Pour ce faire, nous utiliserons des matrices de confusion, qui donnent une vue claire sur les performances du modèle en termes de prédictions correctes et d'erreurs. Cette visualisation permet d'analyser finement les faux positifs, les faux négatifs ainsi que les vrais cas détectés.

4 Non supervisé

Nous avons dans un premier temps entraîné trois modèles non supervisés à reconnaître les données aberrantes. En non-supervisé, les algorithmes divisent les données eux-même, selon des critères appris automatiquement, sans vérité terrain.

Nous avons choisi les trois modèles suivant:

- Isolation Forest
- Local Outlier Factor
- One class SVM

4.1 Isolation Forest

C'est une méthode d'ensemble qui consiste à diviser les données selon des seuils aléatoires sur leur caractéristiques, comptant sur le fait que les données aberrantes seront isolées plus rapidement que les données normales.

Cet algorithme a l'avantage d'être rapide, et fonctionne bien dans le cas où les données à identifier sont assez distinctes des données légitimes.

4.2 Local Outlier Factor

Cette méthode attribue un score (LOF) au points en fonction du nombre de points dans son entourage. L'idée reste la même que pour Isolation Forest, les points aberrants ayant des caractéristiques très differentes des points légitimes, ils sont plus probable d'avoir peu de voisins.

4.3 One Class SVM

Le One Class SVM apprend une frontière qui essaye de délimiter des données assez similaires, qui sont considérées comme légitimes, de celles qui sont bien distinctes et donc considérées comme des attaques.

4.4 Hyper paramètres

Nous avons gardé les hyperparamètres s par défaut pour les trois modèles. Pour chaque modèle, vu que leur entraînement est non supervisé, nous devons spécifier un pourcentage de "contamination". Ce pourcentage représente la proportions de données que le modèle doit identifier comme aberrantes.

En ce sens, les modèles non-supervisées ne sont donc pas totalement autonomes, car il est toujours nécessaire de spécifier un pourcentage de contamination attendu. Ce pourcentage peut être vu comme un quota, et le modèle sélectionne les points les plus suspect tant que le quota n'est pas atteint. Un faible pourcentage de contamination peut donc permettre d'arriver à un meilleure précision, tandis qu'un pourcentage plus élevé permet de détécter plus de points, mais augmente également le taux de faux positifs.

4.5 Analyse comparative des taux de contamination et approches combinées

Afin d'évaluer l'impact du paramètre de contamination sur les performances des différents modèles non supervisés, nous avons réalisé une étude comparative en faisant varier ce paramètre. Le taux de contamination, qui représente la proportion attendue de données aberrantes dans le jeu de données, est un hyperparamètre critique pour les modèles de détection d'anomalies non supervisés.

4.5.1 Méthodologie

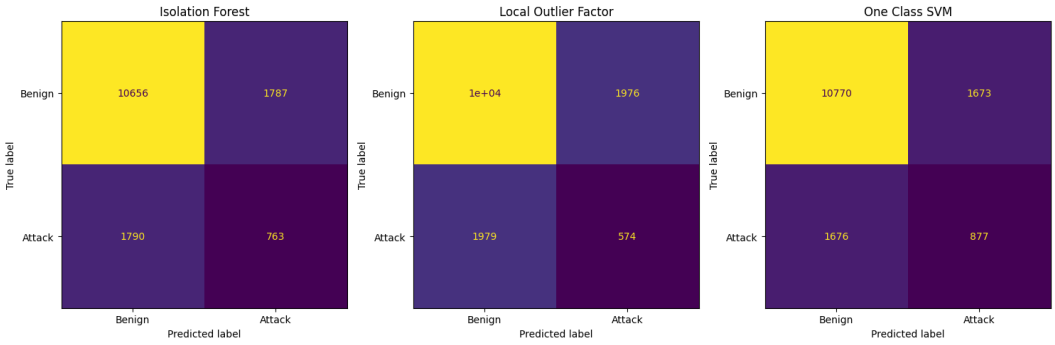
Notre approche a consisté à:

- Faire varier le taux de contamination de 1% à 50% avec 25 points d'échantillonnage
- Appliquer cette variation aux trois modèles précédemment décrits: Isolation Forest, Local Outlier Factor et One Class SVM
- Évaluer les performances individuelles de chaque modèle via le F1-score
- Expérimenter avec des approches d'ensemble combinant les prédictions des trois modèles selon différentes stratégies

Pour les approches d'ensemble, nous avons testé trois stratégies de fusion:

- **Union**: une donnée est considérée comme anomalie si au moins un des trois modèles la détecte comme telle
- **Intersection**: une donnée est considérée comme anomalie uniquement si les trois modèles s'accordent à la classifier comme telle
- **Vote majoritaire**: une donnée est considérée comme anomalie si au moins deux des trois modèles la détectent comme telle

4.5.2 Résultats et analyse



L'analyse des performances révèle plusieurs tendances intéressantes:

- Les performances des modèles individuels varient significativement en fonction du taux de contamination choisi
- Pour des taux de contamination faibles (autour de 10-15%), les modèles individuels atteignent leur meilleure performance, ce qui correspond approximativement à la proportion réelle d'anomalies dans le jeu de données
- L'approche par vote majoritaire (au moins 2 modèles sur 3) offre généralement un meilleur compromis que les méthodes individuelles, témoignant de la complémentarité des différentes approches
- L'union des trois modèles maximise le rappel mais au détriment de la précision, surtout à des taux de contamination élevés
- L'intersection des trois modèles privilégie la précision mais sacrifie le rappel, particulièrement à faible taux de contamination

On observe notamment que le One Class SVM offre les meilleures performances individuelles pour les taux de contamination optimaux, mais sa sensibilité au paramétrage est plus importante que celle des autres modèles.

4.5.3 Conclusion sur l'approche non supervisée

Cette analyse comparative nous permet de tirer plusieurs enseignements:

1. Le choix du taux de contamination est crucial pour les performances des modèles non supervisés
2. Les approches combinées, particulièrement le vote majoritaire, peuvent améliorer la robustesse de la détection
3. Un taux de contamination d'environ 10-15% semble optimal pour ce jeu de données, suggérant que cette proportion correspond effectivement à la distribution réelle des anomalies
4. La complémentarité des différentes méthodes de détection justifie l'utilisation d'approches d'ensemble pour améliorer les performances globales

Ces résultats soulignent l'importance d'une calibration appropriée des modèles non supervisés et montrent l'intérêt des approches combinées pour la détection d'anomalies dans les systèmes industriels.

5 Supervised

Contrairement aux méthodes non supervisées précédemment explorées, les algorithmes d'apprentissage supervisé nécessitent des données étiquetées. Dans le cas du dataset SWaT, nous bénéficions d'étiquettes qui distinguent les données normales des anomalies, ce qui nous permet d'utiliser ces méthodes pour créer des modèles de classification.

5.1 Préparation des données

La première étape consiste à diviser notre ensemble de données en ensembles d'entraînement et de test. Pour garantir une répartition équilibrée des classes dans chaque ensemble, nous utilisons l'option stratify qui maintient les proportions de classes identiques entre les ensembles.

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
    stratify=y, random_state=42)
```

Nous avons alloué 80% des données pour l'entraînement et 20% pour les tests. L'utilisation d'une graine aléatoire (random_state=42) assure la reproductibilité des résultats.

5.2 Modèles supervisés utilisés

Dans cette étude, nous avons implémenté trois algorithmes de classification supervisée différents.

5.2.1 Random Forest

Le Random Forest est un algorithme d'ensemble qui construit plusieurs arbres de décision et combine leurs prédictions. Cette méthode est particulièrement efficace pour gérer des données avec de nombreuses caractéristiques et peut capturer des relations non linéaires complexes. Pour notre implémentation, nous avons configuré le modèle avec 10 arbres de décision et utilisé l'option class_weight='balanced' pour compenser le déséquilibre de classes présent dans le dataset (beaucoup plus d'instances normales que d'anomalies).

5.2.2 Support Vector Machine (SVM)

SVM est un algorithme qui cherche à trouver un hyperplan optimal séparant les différentes classes. Nous avons utilisé un noyau linéaire qui permet une interprétation plus facile des résultats, tout en activant la sortie de probabilités pour une évaluation plus nuancée. L'option class_weight='balanced' ajuste automatiquement les poids des classes pour tenir compte de leur fréquence d'apparition, ce qui est particulièrement important pour notre problème de détection d'anomalies où les classes sont fortement déséquilibrées.

5.2.3 Régression Logistique

La régression logistique est un modèle de classification utilisé pour prédire la probabilité qu'un événement appartienne à une classe (souvent binaire, comme 0 ou 1). Elle utilise la fonction sigmoïde pour transformer une combinaison linéaire des variables d'entrée en une probabilité. Si cette probabilité dépasse un certain seuil (généralement 0,5), l'observation est classée dans une catégorie.

5.3 Entraînement et prédictions

Les trois modèles ont été entraînés sur le même ensemble de données d'entraînement:

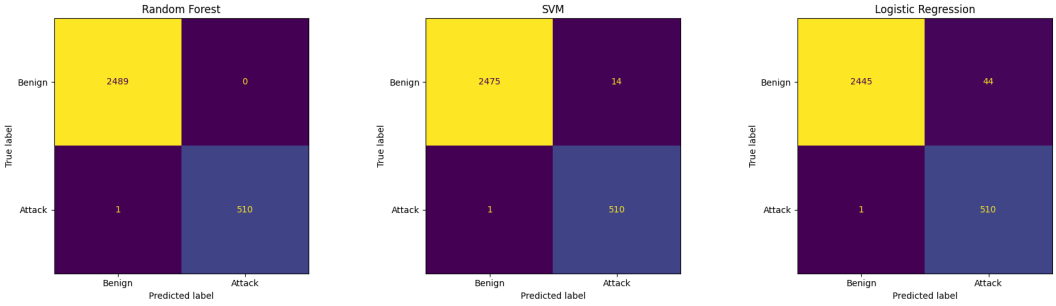
```
rf_model.fit(X_train, y_train)
svm_model.fit(X_train, y_train)
log_model.fit(X_train, y_train)
```

Puis nous avons généré des prédictions sur l'ensemble de test:

```
y_pred_rf = rf_model.predict(X_test)
y_pred_svm = svm_model.predict(X_test)
y_pred_log = log_model.predict(X_test)
```

5.4 Évaluation des performances

5.4.1 Matrices de confusion



Les matrices de confusion montrent la répartition des prédictions correctes et incorrectes pour chaque modèle. Elles permettent de visualiser les vrais positifs (TP), les faux positifs (FP), les vrais négatifs (TN) et les faux négatifs (FN), offrant ainsi une vue complète des performances de classification. D'après les matrices de confusion, nous pouvons observer que:

- le Random Forest présente le meilleur équilibre entre la détection des anomalies et la classification correcte des données normales.
- le SVM montre de bonnes performances globales, mais avec un taux légèrement plus élevé de faux positifs.
- La Régression Logistique, bien que plus simple, offre des résultats comparables aux deux autres modèles, ce qui suggère que certaines anomalies peuvent être détectées par des relations linéaires entre les variables.

5.4.2 Métriques de performance

Pour chaque modèle, nous avons calculé plusieurs métriques standard:

	Precision	Recall	f1-score	support
Random Forest	1.0	0.9980430528375733	0.9990205680705191	511.0
SVM	0.9732824427480916	0.9980430528375733	0.9855072463768116	511.0
Logistic Regression	0.9205776173285198	0.9980430528375733	0.9577464788732394	511.0

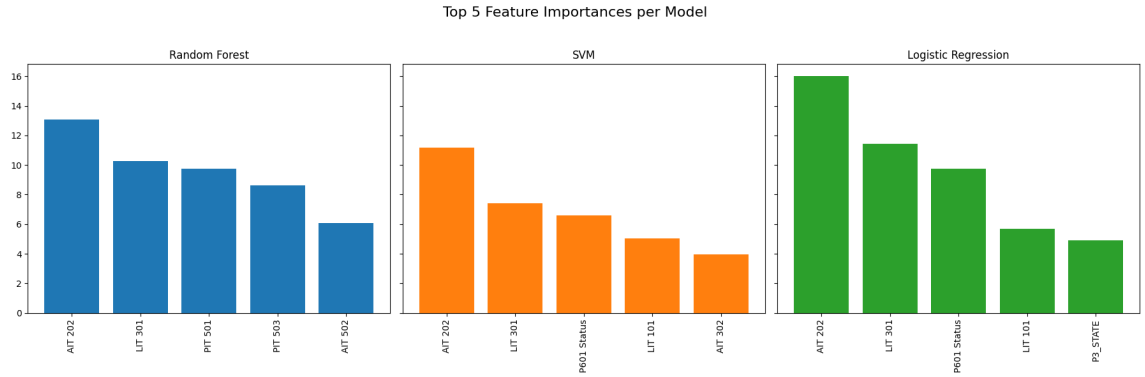
Les métriques clés que nous avons évaluées sont:

- Précision: proportion d'anomalies correctement identifiées parmi toutes les instances classées comme anomalies.
- Rappel: proportion d'anomalies réelles qui ont été correctement identifiées.
- F1-score: moyenne harmonique entre précision et rappel.
- Support: Le nombre d'anomalies à détecter au total.

Le Random Forest obtient les meilleures performances globales avec un F1-score supérieur, suivi de près par le SVM. La Régression Logistique présente des performances légèrement inférieures mais reste une alternative valable, surtout si la rapidité d'exécution ou l'interprétabilité sont des critères importants.

5.5 Analyse des caractéristiques importantes

Pour comprendre quelles variables ont le plus d'impact sur la détection d'anomalies, nous avons extrait et visualisé les 5 caractéristiques les plus importantes selon chaque modèle:



5.5.1 Random Forest

Pour le Random Forest, l'importance des caractéristiques est calculée en fonction de la réduction moyenne de l'impureté (Gini) qu'elles apportent. Les capteurs les plus influents d'après ce modèle sont principalement liés aux mesures de:

1. AIT 202
2. LIT 301
3. PIT 501
4. PIT 503
5. AIT 502

5.5.2 SVM

Pour le SVM linéaire, l'importance des caractéristiques est déterminée par la magnitude des coefficients du vecteur de poids. Les variables jugées les plus importantes par ce modèle sont:

1. AIT 202
2. LIT 301
3. P601 Status
4. LIT 101
5. p3_STATE

5.5.3 Régression Logistique

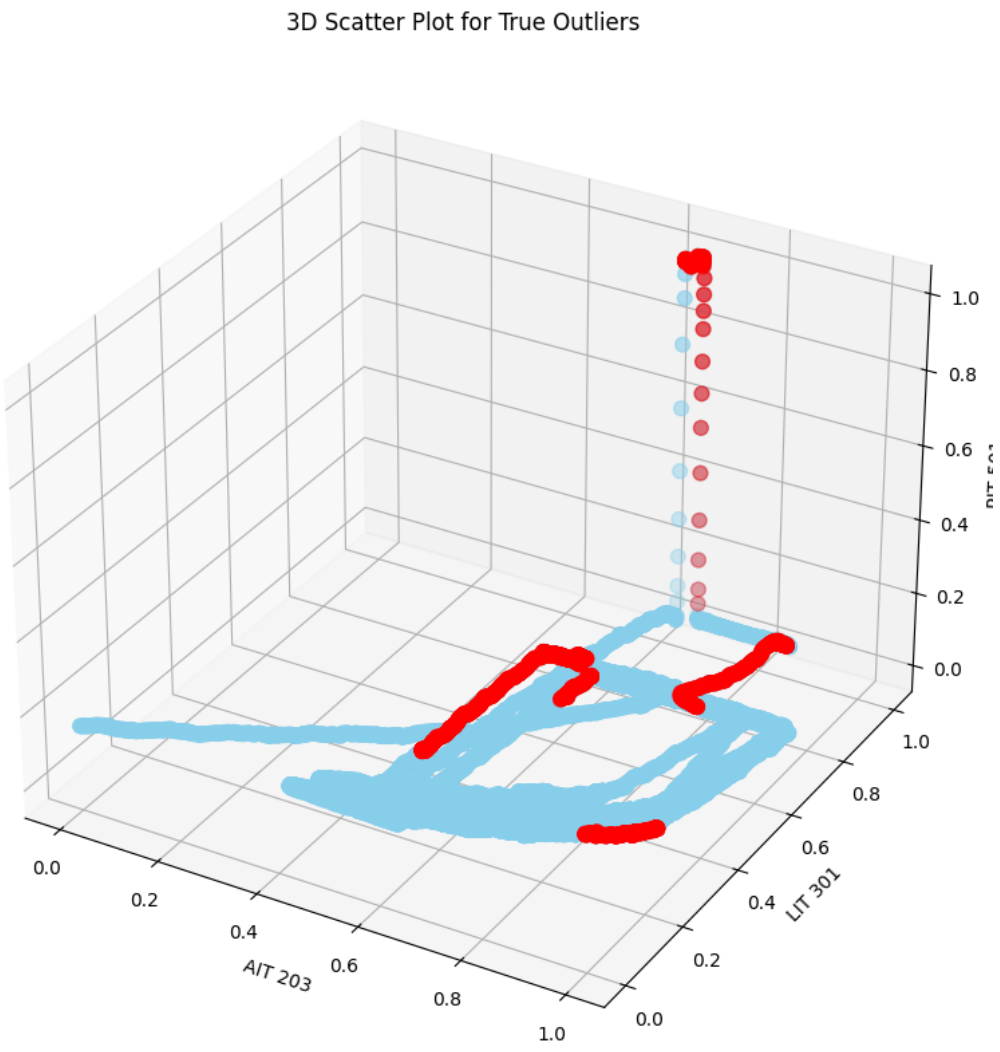
De façon similaire au SVM, l'importance des caractéristiques dans la régression logistique est calculée à partir des coefficients du modèle. Les variables clés identifiées sont:

1. AIT 202
2. LIT 301
3. P601 Status
4. LIT 101
5. AIT 302

5.6 Visualisation des données et des anomalies

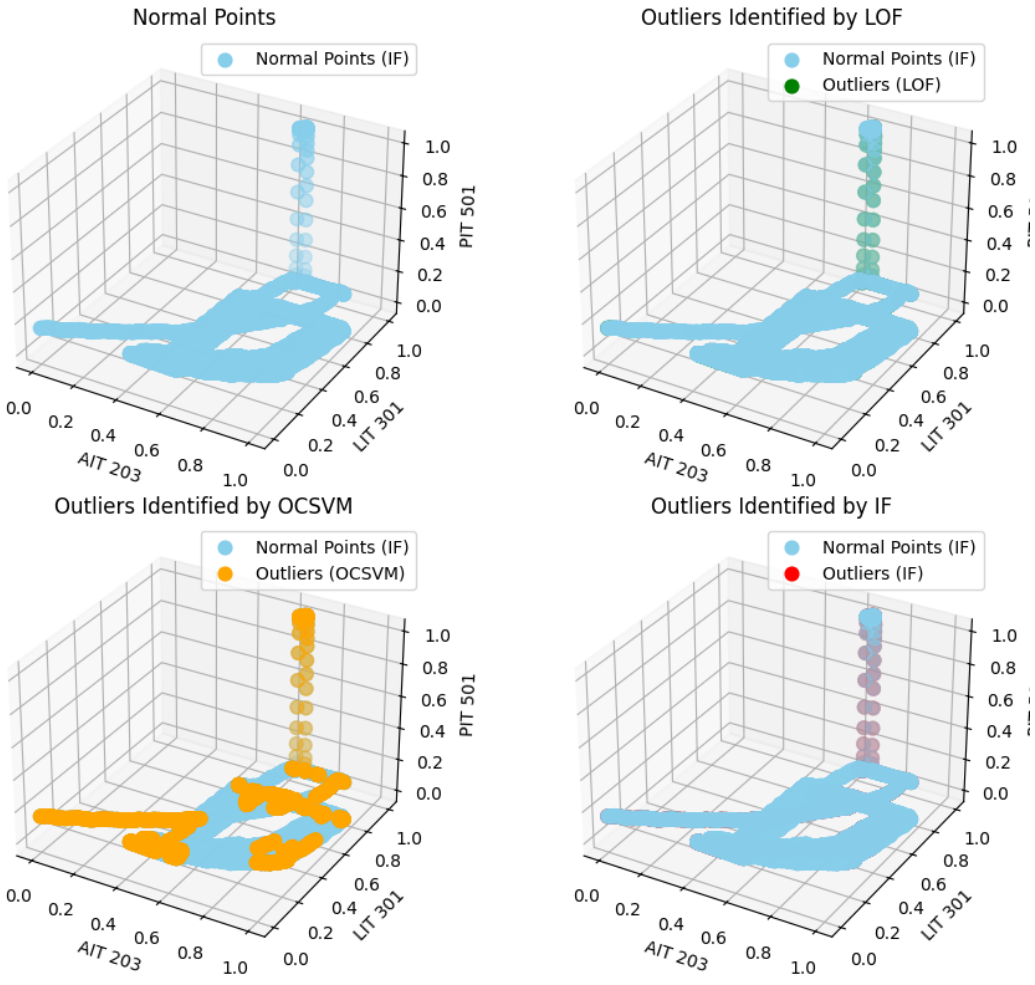
Pour mieux comprendre la distribution spatiale des données et la capacité des différents modèles à détecter les anomalies, nous avons réalisé plusieurs visualisations en 3D, en utilisant trois caractéristiques importantes: ***'AIT 203'***, ***'LIT 301'*** et ***'PIT 501'***.

5.6.1 Visualisation des anomalies réelles



Cette première visualisation montre la distribution des points de données normaux (en bleu ciel) et des anomalies réelles (en rouge). On peut observer que certaines anomalies forment des clusters distincts, tandis que d'autres sont mélangées avec les données normales, ce qui explique pourquoi certaines sont plus difficiles à détecter.

5.6.2 Comparaison des détections d'anomalies entre les modèles



Cette visualisation permet de comparer les points identifiés comme anomalies par les différents modèles non supervisés (Isolation Forest, LOF, One-Class SVM) dans l'espace tridimensionnel. On peut observer des différences notables dans les régions identifiées comme anormales par chaque algorithme:

- Local Outlier Factor (LOF) tend à identifier des points isolés dans l'espace des caractéristiques.
- One-Class SVM définit une frontière plus régulière autour des données normales.
- Isolation Forest semble capturer plus efficacement les clusters d'anomalies.

6 Comparaison des approches supervisées et non supervisées

En comparant les résultats obtenus avec les approches supervisées et non supervisées, plusieurs observations peuvent être faites.

6.1 Avantages des méthodes supervisées

- Performances supérieures: les modèles supervisés obtiennent généralement de meilleures performances en termes de précision et de rappel, car ils bénéficient des étiquettes pour l'apprentissage.
- Meilleure généralisation: Ces modèles peuvent mieux ce généraliser à de nouvelles anomalies similaires à celles vues pendant l'entraînement.
- Interprétabilité : l'importance des caractéristiques fournit des informations précieuses sur les variables les plus pertinentes pour la détection d'anomalies.

6.2 Avantages des méthodes non supervisées

- Indépendance des étiquettes: elles peuvent fonctionner même sans données étiquetées, ce qui est souvent le cas en début de mise en service d'un système.
- Détection de nouvelles anomalies: elles peuvent potentiellement identifier des types d'anomalies jamais observés auparavant.
- Coût réduit: l'étiquetage des données étant souvent coûteux et chronophage, les méthodes non supervisées offrent une alternative économique.

6.3 Limites comparées

Les méthodes supervisées nécessitent un ensemble de données d'entraînement représentatif des différents types d'anomalies, ce qui peut être difficile à obtenir dans des environnements dynamiques où de nouvelles formes d'attaques apparaissent régulièrement. Les méthodes non supervisées, quant à elles, peuvent générer un nombre important de faux positifs, surtout si les paramètres ne sont pas correctement ajustés, ce qui peut entraîner une fatigue d'alerte chez les opérateurs de sécurité.

6.4 Approche hybride

Une approche prometteuse consisterait à combiner les deux types de méthodes:

- Utiliser des modèles non supervisés pour la détection initiale d'anomalies potentielles.
- Faire valider ces détections par des experts pour créer un ensemble de données étiquetées.
- Entraîner des modèles supervisés sur ces données pour améliorer la précision.

Cette approche itérative permettrait de bénéficier des avantages des deux types de méthodes tout en atténuant leurs limites respectives.