RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE

MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITÉ ABDERRAHMANE MIRA – BEJAIA

جامعة بجاية
Tasdawit n Bgayet
Université de Béjaïa

FACULTÉ DES SCIENCES EXACTES
DÉPARTEMENT DE MATHÉMATIQUES

Mémoire de Fin d'Études

Présenté pour l'obtention du diplôme de

Master en Mathématiques

Option : **Mathématiques de l'Intelligence Artificielle" M.I.A"**

Thème du mémoire

Détection de fraude dans la consommation d'électricité en combinant l'IA et les méthodes mathématiques

Présenté par :

M. SALHI Malek

Soutenu le 29 juin 2025 devant le jury composé de :

M. FARAH	Président	U. A. Mira Béjaia
M. SEBAA	Rapporteur	U. A. Mira Béjaia
M. HADJOUT	Co-rapporteur	SONELGAZ Distribution
M.MIR	Examinateur	U. A. Mira Béjaia

Année universitaire : 2024 – 2025

Table des matières

Remerciements

Avant toute chose, je rends grâce à Allah, le Tout-Puissant, le Clément, le Miséricordieux, qui m'a accordé la santé, la patience et la force intérieure pour mener à bien ce travail. C'est par Sa volonté que j'ai pu surmonter les difficultés, persévérer face aux défis et atteindre cet objectif. Que toutes mes louanges Lui soient dédiées, Lui qui éclaire nos chemins et guide nos pas dans la quête du savoir et de la vérité.

Je tiens à exprimer ma profonde gratitude à Monsieur **A. Sebaa**, mon encadrant universitaire, pour la qualité de son encadrement, sa disponibilité, ses conseils avisés et son soutien tout au long de ce projet. Son accompagnement m'a été d'une grande aide.

Mes remerciements vont également à Monsieur **Hadjout Dalil**, encadrant de stage au sein de l'entreprise SONELGAZ. Son expertise et ses conseils m'ont permis d'évoluer tant sur le plan professionnel que personnel.

Je tiens à exprimer ma sincère gratitude aux membres du jury pour le temps qu'ils ont consacré à l'évaluation de mon travail.

Enfin, je remercie toutes les personnes qui, de près ou de loin, m'ont soutenu tout au long de ce parcours. Leur présence et leurs encouragements m'ont été précieux.

Dédicaces

Je dédie ce travail :

À la mémoire de mon père bien-aimé, dont la sagesse et le soutien continuent de m'inspirer chaque jour. Ce mémoire est dédié à sa mémoire éternelle. Qu'Allah lui fasse miséricorde et lui accorde le paradis pour demeure.

À ma mère, ton amour, tes duaas, ta persévérance et ton soutien indéfectible m'ont guidé tout au long de ce parcours académique. Ce mémoire est une humble reconnaissance de ta présence constante dans ma vie. Merci pour tout ce que tu as fait et continues de faire.

À mon cher frère Mohand. Je ne saurais oublier de dédier ce travail à toi, mon frère, qui as partagé avec moi les joies, les peines et les défis de la vie.

À mes chères sœurs Kahina, Faiza et Houa, pour leur amour, leur soutien et leur présence précieuse à chaque étape de mon parcours.

À Monsieur Hadjout Dalil, mon encadrant de stage. Merci infiniment pour tous vos efforts et, surtout, pour vos précieux conseils.

À Monsieur A. Sebaa, mon encadrant universitaire. Je vous remercie sincèrement pour votre accompagnement, votre écoute et la qualité de vos orientations tout au long de ce travail. Votre encadrement a été d'une grande valeur pour moi.

À mon amie très chère, KASSA Celina, pour ton soutien indéfectible, ta présence réconfortante et tes encouragements constants tout au long de ce parcours. Cette réussite est aussi un peu la tienne. Merci du fond du cœur.

À mes deux meilleurs amis, Hadjal Badis et Ait Ahmed Lyes. Vous avez joué un rôle essentiel dans ma vie, m'encourageant à persévérer, me poussant à dépasser mes limites, et m'insufflant la confiance nécessaire pour avancer. Ce mémoire vous est dédié en témoignage de notre amitié profonde, de nos souvenirs précieux et de notre soutien mutuel. Qu'Allah vous protège.

S. Malek

Table des figures

Liste des tableaux

Liste des abréviations

Liste des abréviations

IA	Intelligence Artificielle
ML	Machine Learning (Apprentissage automatique)
DL	Deep Learning (Apprentissage profond)
SVM	Support Vector Machine
OCSVM	One-Class Support Vector Machine
ARIMA	AutoRegressive Integrated Moving Average
IF	Isolation Forest
IQR	Interquartile Range (Étendue interquartile)
RMSE	Root Mean Square Error (Erreur quadratique moyenne)
TP	True Positive
FP	False Positive
FN	False Negative
TN	True Negative
F1	F1-Score (Mesure harmonique précision/rappel)

Introduction

À l'ère de la numérisation massive des activités humaines et industrielles, les données se sont imposées comme un levier stratégique incontournable pour l'analyse, la prise de décision et l'optimisation des systèmes. Dans des domaines variés tels que la finance, l'énergie, la santé ou la cybersécurité, l'analyse des données est courante, et les séries temporelles y tiennent une place prépondérante. Ces suites d'observations ordonnées dans le temps recèlent une valeur informationnelle considérable, indispensable à la compréhension des dynamiques sous-jacentes et à l'anticipation des évolutions futures.

Au-delà des tendances régulières et des cycles attendus, les séries temporelles peuvent être le théâtre d'événements anormaux : pics inattendus, creux marqués, ruptures de comportement. Ces anomalies, souvent invisibles à l'œil nu, représentent des signaux faibles mais critiques. Souvent imperceptibles, ces anomalies agissent comme des signaux faibles mais déterminants. Leur présence peut indiquer des défaillances techniques, des tentatives de malversation, des conduites douteuses ou des désordres opérationnels.

Dans des secteurs sensibles comme la distribution d'énergie, leur détection rapide est primordiale pour préserver la fiabilité des infrastructures, garantir la continuité des services, et assurer la conformité réglementaire. Cette problématique soulève des enjeux scientifiques et techniques majeurs, aux retombées économiques et opérationnelles significatives.

C'est dans ce contexte que s'inscrit le présent travail de fin d'études, dédié à l'étude approfondie de la détection d'anomalies dans les séries temporelles [4]. L'objectif est double : offrir une vision claire et structurée des différentes approches existantes, et proposer une solution concrète, robuste et exploitable dans un cadre industriel réel.

Ce projet s'est nourri à la fois d'une revue méthodologique rigoureuse et d'une expérience pratique acquise lors d'un stage au sein de l'entreprise SONELGAZ, acteur clé du secteur énergétique en Algérie. Cette immersion a permis d'illustrer la pertinence de la thématique et d'en apprécier les défis techniques sur le terrain.

Le mémoire est structuré autour de quatre chapitres complémentaires :

Chapitre 1 : Présente les notions fondamentales relatives aux séries temporelles, aux formes typiques de fraudes, ainsi qu'aux principales catégories d'anomalies. Ce socle théorique permet de poser un cadre clair pour l'ensemble de l'étude.

Chapitre 2 : Dresse un panorama détaillé des méthodes de détection d'anomalies, regroupées en trois grandes familles : approches statistiques, méthodes de prévision et techniques d'apprentissage automatique. Chaque approche est analysée sous l'angle de ses fondements, cas d'usage et limites.

Chapitre 3 : Propose une étude comparative rigoureuse basée sur le jeu de données A1Benchmark, issu du corpus Yahoo Webscope S5. Cette analyse vise à évaluer objectivement les performances des différentes méthodes sur des séries annotées manuellement.

Chapitre 4 : Met en œuvre une approche hybride combinant One-Class SVM et régression linéaire sur des données réelles de consommation électrique de SONELGAZ. Il détaille également le développement d'une application logicielle interactive, conçue pour aider à l'analyse métier des comportements anormaux détectés.

Ce mémoire a pour but principal d'effectuer une analyse comparative exhaustive des méthodes de détection d'anomalies appliquées aux séries temporelles, en intégrant à la fois les aspects théoriques et pratiques [5].

Au-delà d'une simple comparaison des algorithmes, l'ambition est de développer une solution concrète et opérationnelle, directement applicable dans un cadre industriel. Cette étude ouvre également des perspectives d'amélioration, telles que l'extension aux données multivariées, l'exploration de techniques d'apprentissage profond, ou le raffinement de l'approche hybride proposée.

Chapitre 1

Séries temporelles et la problématique de la fraude

1.1 Introduction

Dans ce chapitre, nous définissons la notion de fraude et expliquons son impact dans divers domaines. Nous introduisons ensuite les séries temporelles, en décrivant leurs principales caractéristiques ainsi que les composantes qui les constituent. Ces éléments sont essentiels pour mieux comprendre les phénomènes anormaux dans les données.

1.2 Qu'est-ce que la fraude ?

La fraude désigne un acte intentionnel visant à tromper, falsifier ou dissimuler des informations dans le but d'obtenir un avantage illégal ou injustifié. Elle peut survenir dans divers contextes, notamment financier, administratif, commercial ou numérique, et elle engendre souvent des pertes économiques importantes. Dans un cadre analytique ou algorithmique, la fraude se manifeste souvent par des comportements inhabituels ou aberrants par rapport à un modèle attendu ou à une norme statistique[26]. Ces comportements sont souvent détectables à travers des méthodes d'analyse de données ou de séries temporelles. On distingue plusieurs types de fraude, notamment :

- **Fraude financière :** fausses transactions, falsification de documents comptables.
- **Fraude à la consommation :** utilisation abusive de cartes bancaires, retour frauduleux de produits.
- **Fraude sur factures :** création ou modification de factures pour détourner des fonds.
- **Fraude algorithmique :** manipulation de données ou d'algorithmes pour influencer des résultats.

1.3 Détection de fraudes

Pendant les périodes de volatilité économique, les établissements ont toujours été confrontés à de nombreux risques majeurs pour leurs opérations, notamment l'interruption de la chaîne d'approvisionnement, la fidélisation du personnel ou les cybermenaces. En 2023, cette situation a entraîné une augmentation du nombre d'activités frauduleuses avec des fraudeurs qui utilisent des outils à la pointe de la technologie pour exploiter la situation. Selon l'étude 2022 de Juniper Research intitulée « Combatting Online PaymentFraud», les pertes liées à la fraude aux paiements au niveau mondial devraient dépasser 343 milliards de dollars entre 2023 et 2027. Par conséquent, les établissements financiers prennent des mesures visant à améliorer leurs dispositifs de détection de la fraude pour se protéger et protéger leurs clients contre les dommages financiers[16].

Définition 1 *La détection de la fraude consiste à identifier ou mettre en lumière des agissements frauduleux. C'est un processus qui utilise diverses techniques et outils pour identifier, prévenir et répondre à de telles activités.*

En surveillant les transactions, les comportements et les systèmes, elle cherche à repérer les anomalies ou irrégularités susceptibles d'indiquer une fraude, dans le but de protéger les actifs, les informations sensibles et la réputation d'une organisation.

1.3.1 Les principales techniques de détection de la fraude

Détection des Anomalies : La détection des anomalies s'appuie sur l'analyse comportementale, l'apprentissage automatique et divers outils analytiques dans le but d'identifier tout écart par rapport aux comportements ou données habituels.

Détection en Temps Réel : Surveillance continue des transactions pour repérer rapidement les activités suspectes.

Analyse des Réseaux Sociaux : Examen des interactions entre acteurs pour détecter des activités frauduleuses coordonnées.

Regroupement d'Identités : La détection de caractéristiques frauduleuses s'effectue en analysant des regroupements, des attributs spécifiques et des comportements observés.

Partage de Renseignements sur la Fraude : Collaboration avec d'autres organisations pour échanger des informations et améliorer les stratégies de prévention.

La détection de la fraude, bien qu'essentielle, se base souvent sur la détection d'anomalies. Cette approche, qui vise à identifier des comportements ou des transactions atypiques, constitue une pierre angulaire dans la recherche et l'implémentation de solutions efficaces pour détecter les fraudes. La détection d'anomalies, en particulier, se concentre sur l'identification des écarts par rapport aux modèles normaux, et mérite une attention approfondie dans le contexte de la lutte contre la fraude[11].

1.4 Détection d'Anomalie dans les séries temporelles

Définition 2 *La détection d'anomalie dans les séries chronologiques consiste à identifier les observations qui s'écartent de manière significative du comportement normal ou attendu de la série. Les séries chronologiques peuvent parfois être influencées par des événements isolés, des perturbations ou des erreurs qui génèrent des effets artificiels, entraînant des motifs inhabituels dans les données. Ces observations atypiques, appelées valeurs aberrantes, peuvent résulter d'événements externes rares comme des grèves, des changements politiques ou économiques soudains, des phénomènes météorologiques exceptionnels, ou encore de modifications brusques dans un système physique. Elles peuvent également découler d'erreurs d'enregistrement ou de mesures incorrectes. La détection d'anomalie a pour objectif de repérer ces écarts afin de comprendre leur origine et d'évaluer leur impact potentiel sur les prévisions et les analyses de la série temporelle.[11]*

1.4.1 Les principales méthodes de détection d'anomalie :

- Les méthodes statistiques tels que le test de Z-Score et le test IQR.
- Les méthodes basées sur la prévision tel que le lissage exponentiel, la régression linéaire et ARIMA.
- Les méthodes basées sur le Machine Learning tel que l'Isolation Forest, OCSVM ... Ets

1.4.2 Définition mathématique

La détection d'anomalies en mathématiques et en statistique désigne l'identification de données ou d'observations qui diffèrent significativement de la majorité des données présentes[6]. Ces observations anormales ou

"anomalies" peuvent indiquer des erreurs, des événements rares, ou des changements significatifs dans un phénomène observé. Soit $X = x_1, x_2, ..., x_n$ un ensemble de données provenant d'une distribution $P(X)$, où chaque $x(i)$ est une observation dans un espace R^d de dimension d. La détection d'anomalies cherche à identifier un sous-ensemble de points $A \subset X$ tel que les observations $x_i \in A$ s'écartent significativement des autres points dans X, selon un critère donné. Formellement, cela revient à résoudre : **Trouver** $A \subseteq X$ **tel que** $(\forall(x_i, modele) > seuil)$

1.5 Série Chronologique

1.5.1 Définition

Pour analyser et comprendre les mécanismes économiques ainsi que pour prendre des décisions éclairées, il est souvent nécessaire de disposer de plusieurs observations de variables importantes. Dans de nombreux domaines économiques, il est impossible de mener des expériences contrôlées. Par conséquent, ces observations sont fréquemment obtenues à travers des enquêtes ou l'exploitation de bases de données existantes, et sont disponibles à intervalles réguliers, comme une fois par an ou par trimestre. Une série d'observations d'une même variable, notée X_t $t \in \theta$, est appelée série temporelle ou série chronologique[9].

Exemple : On peut songer à l'évolution des paiements pour l'électricité et le gaz effectués par des entreprises et des particuliers. Par exemple, on pourrait analyser l'évolution mensuelle des montants facturés aux différentes entreprises ou aux foyers pour observer des tendances ou des variations saisonnières. Cela pourrait inclure l'identification de pics de consommation durant certains mois ou de variations dans les paiements entre différentes entreprises, ce qui permettrait de mieux comprendre les comportements de consommation et de prévoir les besoins futurs.
L'ensemble θ est appelé espace des temps qui peut être :

— **Discret** (montants mensuels des paiements pour l'électricité et le gaz, ou nombre d'entreprises et de particuliers facturés chaque mois...). Dans ce cas, $\Theta \subset Z$. Les dates d'observations sont le plus souvent équidistantes : par exemple, relevés mensuels, trimestriels... Ces dates équidistantes sont alors indexées par des entiers : $t = 1, 2, \ldots, T$ et T est le nombre d'observations. On dispose donc des observations des variables $X_1, X_2, \ldots, X_T$ issues de la famille $(X_t)_{t\in\Theta}$ où $\Theta \subset Z$ (le plus souvent $\Theta = Z$). Ainsi, si h est l'intervalle de temps séparant deux observations et t_0 l'instant de la première observation, on a le schéma suivant :

$$\begin{array}{cccc} t_0 & t_0+h & \cdots & t_0+(T-1)h \\ \downarrow & \downarrow & \cdots & \downarrow \\ X_{t_0} & X_{t_0+h} & \cdots & X_{t_0+(T-1)h} \\ \downarrow & \downarrow & \cdots & \downarrow \\ X_1 & X_2 & \cdots & X_T \end{array}$$

— **continu** (signal radio, résultat d'un électrocardiogramme...). L'indice de temps est à valeurs dans un intervalle de R et on dispose (au moins potentiellement) d'une infinité d'observations issues d'un processus (Xt) t $\in \theta$ où θ est un intervalle de R. Un tel processus est dit à temps continu. Les méthodes présentées dans ce cadre sont différentes de celles pour les séries chronologiques à temps discret et présentées dans la suite.

1.5.2 Description d'une série chronologique :

— **La tendance** :C'est la tendance générale de la variable étudiée sur une longue période. Cette tendance ou trend T est représentée par la courbe qui ajuste l'ensemble des points du nuage. On appelle également « tendance », la courbe de lissage obtenue, par exemple, par le calcul des moyennes mobiles ou échelonnées ou cycliques. . .toutes relèvent d'une démarche empirique. L'ajustement du trend peut être également déterminé par une méthode analytique telle que la méthode des moindres carrés MMC (que nous allons voir dans ce projet)

— **La composante saisonnière**La composante saisonnière (ou saisonnalité (S_t)) correspond à un phénomène qui se répète à intervalles de temps réguliers (périodiques) et qui correspond souvent à des phénomènes de mode, de coutume, de climat... En général, c'est un phénomène saisonnier d'où le terme de variations saisonnières.

— **La composante résiduelle (ou bruit ou résidu ε_t)** : Elle rassemble tout ce que les autres composantes n'ont pu expliquer du phénomène observé. Elle contient donc de nombreuses fluctuations, en particulier accidentelles, dont le caractère est exceptionnel et imprévisible (catastrophes naturelles, grèves, guerres...). Comme par hypothèse ce type d'évènement est censé être corrigé, le résidu présente en général une allure aléatoire plus ou moins stable autour de sa moyenne.

— **Des phénomènes accidentels** :Les fluctuations aléatoires sont généralement de courtes durées, imprévisibles et irrégulières. La somme des variations aléatoires est également nulle dans le temps. Elles sont imprévisibles et n'obéissent à aucun rythme. Elles sont dues à des évènements exceptionnels tels que les catastrophes naturelles, les grèves... Le traitement des séries chronologiques repose sur la nature du lien unissant ces différents types de variations ou de mouvements (fluctuations). Le but de l'analyse chronologique est de repérer les différents mouvements afin de pouvoir faire de bonnes prévisions. Il existe deux types de séries et de modèles : modèle additif et modèle multiplicatif[2].

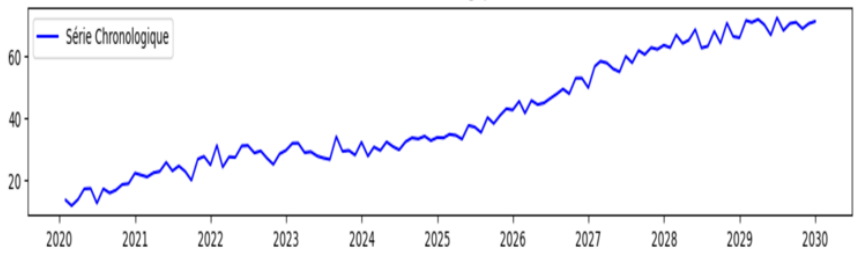


FIGURE 1.1 – Exemple d'une série chronologique

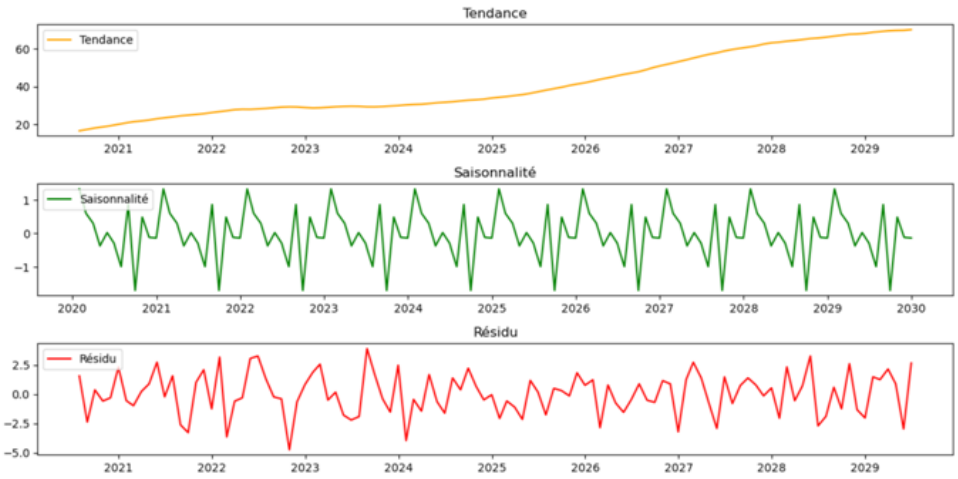


FIGURE 1.2 – Exemple d'une tendance,saisonnalité et résidu

1.6 Modélisation d'une série temporelle

La modélisation d'une série temporelle consiste à décomposer ses différentes composantes afin de mieux comprendre sa structure et prévoir son évolution. Parmi les approches classiques, on distingue principalement les modèles additifs, multiplicatifs et mixtes.

1.6.1 Le modèle additif

Dans cette partie, nous postulons qu'une série chronologique (X_t) est caractérisée par une décomposition additive :

$X_t = T_t + S_t + \varepsilon\ t$,$t = 1, \ldots, T$ où T_t représente la composante tendancielle, S_t la composante saisonnière, et $\varepsilon\ t$l'erreur ou l'écart par rapport au modèle. Comme mentionné en introduction :

— **Tendance** T_t **:** Elle capte les variations à moyen terme de la série et est généralement décrite par une fonction polynomiale du temps

— **Composante saisonnière** S_t **:**Elle caractérise un phénomène qui se manifeste de façon analogue à intervalles de temps réguliers, appelé période (notée P). Généralement, la composante saisonnière est constante sur chaque période P, c'est-à-dire :$S(_t+_p)=S_t$,$\forall\ t$[7]. Ainsi, l'effet net de la saisonnalité sur une période est nul, ce qui est logique puisque cet effet est déjà capturé dans la tendance générale de la série. Ce modèle est le plus simple, où la saisonnalité est caractérisée par P coefficients C_1,. . .,C_p. Par exemple, lorsque P=4 , la série est trimestrielle ; lorsque P=12, elle est mensuelle. On suppose en outre que l'effet saisonnier est en moyenne nul sur une période, ce qui signifie que : $\sum_{i=1}^{p} C_i = 0$

— **Erreurs** $\varepsilon\ _t$: Ce sont des variables aléatoires centrées, souvent considérées comme un bruit blanc, c'est-à-dire une suite de variables aléatoires telles que :

$$E[\varepsilon_t] = 0.$$

Les variables aléatoires sont donc non corrélées, et dans le cas où le bruit blanc est gaussien, c'est-à-dire lorsque $\epsilon \sim \mathcal{N}(0, 1)$, elles sont également indépendantes[3].

1.6.2 Le Modèle Multiplicatif

Dans cette section, nous examinons une série temporelle X_t=(X_t) qui suit une décomposition multiplicative : X_t=$T_t \times S_t\ \times\ \varepsilon_t$ Où T_t représente la composante tendancielle, $S_t$la composante saisonnière, et ε_t l'erreur ou l'écart par rapport au modèle. De plus, la composante saisonnière vérifie : $\sum_{i=1}^{p} C_i = 0$ Dans ce modèle, l'amplitude de la série n'est plus constante au fil du temps : elle varie

proportionnellement à la tendance T_t, sous réserve du bruit. On considère ici que les amplitudes des fluctuations dépendent du niveau.

1.6.3 Les modèles mixtes

Il s'agit l'a de modèles où addition et multiplication sont utilisées. On peut supposer par exemple que la composante saisonnière agit de façon multiplicative alors que les fluctuations irrégulières sont additives :X_t=T_t $\times S_t$ + ε $_t$. Avec l'hypothèse ici que $\sum_{i=1}^{p} C_i = 0$

1.6.4 Choix du modèle

Avant toute modélisation et étude approfondie, il est essentiel de déterminer si la série présente des caractéristiques spécifiques pour une observation X donnée[12].

— Dans le modèle additif, la composante saisonnière (S) est directement ajoutée à la tendance (Z)
— En ce qui concerne le modèle multiplicatif, la variation saisonnière (S) est en relation de proportionnalité avec la tendance (Z)..

Afin de faire cette distinction, on peut se baser sur une méthode :

•**Méthode de la bande :**On fait un graphique représentant la série chronologique, puis on trace une droite passant respectivement par les minima et par les maximas de chaque saison. Si ces deux droites sont parallèles, nous sommes en présence d'un modèle additif. Dans le cas contraire, c'est un modèle multiplicatif.

•**Méthode du profil :** A partir du graphe des courbes superposées (graphique sur lequel on trace une courbe pour chacune des années représentant ainsi l'évolution annuelle de la grandeur observée par trimestre par exemple), si l'on constate que les courbes sont à peu près parallèles (varie de la même manière) le modèle est alors additif sinon multiplicatif.

•**Méthode analytique :**On calcule les moyennes et les écarts-types pour chacune des périodes considérées puis la droite des moindres carrés $\sigma = ax + b$. Pour des rappels sur la droite des moindres carrés voire le chapitre suivant. Si a est nul, c'est le modèle additif, sinon c'est le modèle multiplicatif.

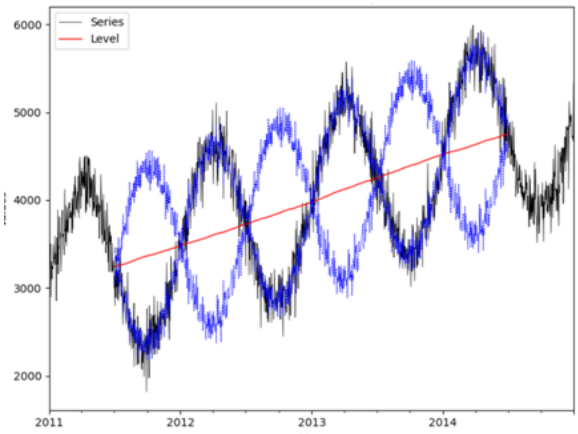


FIGURE 1.3 – Saisonnalité additive

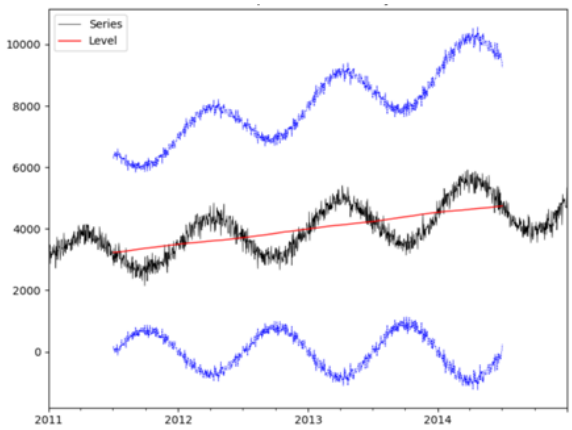


FIGURE 1.4 – Saisonnalité multiplicative

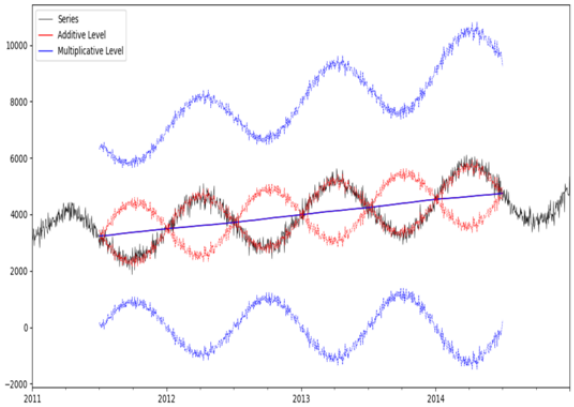


FIGURE 1.5 – Saisonnalité mixte

1.7 Conclusion

Ce premier chapitre a établi le cadre de notre étude en soulignant l'importance de la détection d'anomalies, particulièrement pour identifier la fraude. Nous avons défini les concepts clés d'anomalie et de détection d'anomalies, tout en rappelant les fondamentaux des séries temporelles. Ces bases posées, nous sommes désormais prêts à explorer les méthodes spécifiques pour détecter ces comportements atypiques dans les chapitres suivants.

Chapitre 2

Méthode de détection d'Anomalies

2.1 Introduction

Dans le domaine de l'analyse des séries temporelles, la détection d'anomalies représente une problématique essentielle, en particulier lorsqu'il s'agit d'identifier des comportements inhabituels pouvant révéler des fraudes. Ce chapitre présente un ensemble de méthodes permettant de détecter ces anomalies. We have regrouped into three main categories : statistics approaches, methods based on prediction, and automatic learning techniques issues. Chaque catégorie est illustrée par des exemples simples, afin de mieux comprendre leur fonctionnement. Enfin, une brève comparaison théorique met en lumière les avantages et les limites de ces différentes approches.

2.2 Méthode statistique pour la détection d'Anomalies

Les méthodes statistiques représentent une approche essentielle et sont souvent les premières déployées dans le domaine de la détection d'anomalies appliquées aux séries temporelles [5]. Elles s'appuient sur les propriétés statistiques intrinsèques des données pour identifier les points qui s'écartent significativement de la "norme" établie [10]. Simples à comprendre et à implémenter, ces méthodes sont particulièrement utiles pour détecter des anomalies ponctuelles ou des déviations claires par rapport à la distribution attendue. Dans cette étude, nous nous concentrons sur deux méthodes statistiques fondamentales : le Z-Score et la méthode de l'Écart Inter-Quartile (IQR). Chacune de ces approches utilise des principes statistiques différents pour quantifier la "normalité" d'une observation et signaler les valeurs extrêmes. Ces méthodes statistiques classiques constituent la base théorique de nombreuses approches modernes de détection d'anomalies et demeurent des outils de référence pour l'évaluation comparative des performances [1].

2.2.1 Test de quartiles IQR

Définition 3
IQR (Intervalle Interquartile)est utilisé pour mesurer la variabilité en divisant un ensemble de données en quartiles. Les données sont triées par ordre croissant et divisées en 4 parties égales. Les valeurs Q1, Q2 et Q3, appelées premier, deuxième et troisième quartile, sont les valeurs qui séparent ces 4 parties égales[8].

— **Q1** représente le 25e percentile des données.

— **Q2** représente le 50e percentile des données (également la médiane de l'ensemble des données).

— **Q3** représente le 75e percentile des données.

Si un ensemble de données comporte 2n ou 2n+1 points de données, alors :

— **Q2**= médiane de l'ensemble des données.
— **Q1** = médiane des n plus petites données.
— **Q3** = médiane des n plus grandes données.

plage entre le premier et le troisième quartile, à savoir Q1 et Q3 : IQR = Q3 – Q1. Les points de données qui se trouvent en dessous de Q1 – 1,5 IQR ou au-dessus de Q3 + 1,5 IQR sont considérés comme des valeurs aberrantes (outliers).

Exemple : Supposons que nous avons des données sur les ventes mensuelles d'une entreprise et que nous souhaitions identifier les mois avec des ventes anormalement élevées ou basses.

Mois	Jan	Fév.	Mars	Avr	Mai	Juin	Juil	Aout	Sep	Oct	Nov	Déc
Ventes	20	22	24	30	25	28	33	29	26	35	40	27

TABLE 2.1 – Ventes Mensuelles d'une Enreprise

Etapes de calcul des quartiles

— **Étape 1 : Trier les données** Tout d'abord, vous devez trier les données dans l'ordre croissant :
[20, 22, 24, 25, 26, 27, 28, 29, 30, 33, 35, 40]

— **Étape 2 : Calculer le Q2 (la médiane)**
Q2 est la médiane de l'ensemble des données. Puisque nous avons 12 données (un nombre pair), la médiane est la moyenne des deux valeurs centrales :
-Les deux valeurs centrales sont : 27 et 28.
-Q2 (médiane) = (27 + 28) / 2 = 27,5.

— **Étape 3 : Calculer le Q1 (le premier quartile)**
Q1 est la médiane de la première moitié des données (les plus petites valeurs). La première moitié des données (avant la médiane) est :
[20, 22, 24, 25, 26, 27]
-Ici, la médiane (Q1) est la moyenne des deux valeurs centrales : 24 et 25.
-Q1 = (24 + 25) / 2 = 24,5.

— **Étape 4 : Calculer le Q3 (le troisième quartile)**
Q3 est la médiane de la deuxième moitié des données (les plus grandes valeurs). La deuxième moitié des données (après la médiane) est :
[28, 29, 30, 33, 35, 40]
-Ici, la médiane (Q3) est la moyenne des deux valeurs centrales : 30 et 33.
-Q3 = (30 + 33) / 2 = 31.

— **Étape 5 : Calculer l'IQR**
L'IQR est simplement la différence entre Q3 et Q1 : -IQR = Q3 - Q1 = 31,5 - 24,5 = 7.

— **Étape 6 : Déterminer les bornes pour détecter les anomalies**
Les bornes pour détecter les anomalies sont calculées ainsi :
-Borne inférieure = Q1 - 1,5 * IQR = 24,5 - 1,5 * 7 = 14.
-Borne supérieure = Q3 + 1,5 * IQR = 31,5 + 1,5 * 7 = 42.

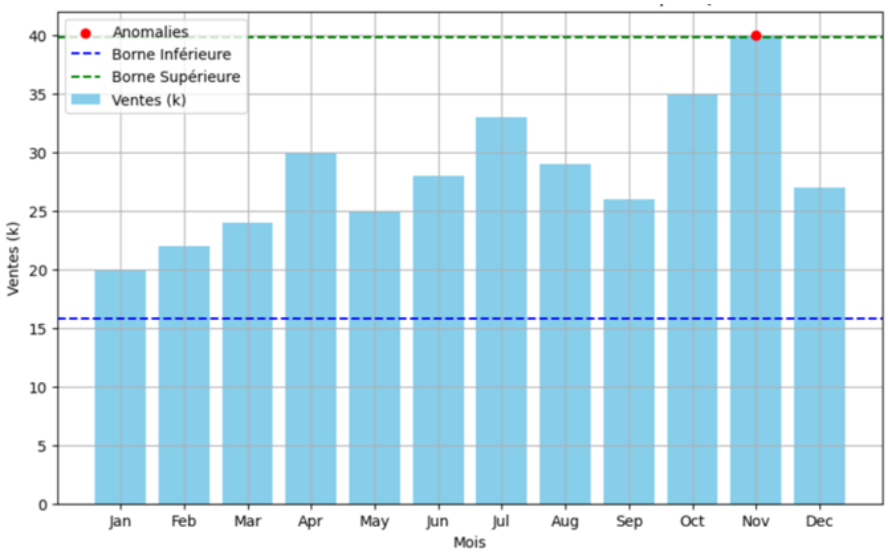


FIGURE 2.1 – Ventes Mensuelles avec Détection des Anomalies par IQR

Ce graphique illustre les ventes mensuelles sur une année, avec une détection d'anomalies basée sur la méthode de l'intervalle interquartile (IQR). On observe une tendance générale à la hausse des ventes au fil de l'année, avec des pics notables en avril, juillet et octobre. Novembre se distingue comme une anomalie significative, dépassant la borne supérieure et indiquant des ventes exceptionnellement élevées. Cette approche permet d'identifier rapidement les performances mensuelles inhabituelles, facilitant ainsi l'analyse des tendances et la prise de décisions commerciales.

2.2.1.1 Avantages et inconvénient

Avantages :

— Robuste aux outliers : Le test IQR n'est pas influencé par les valeurs extrêmes, ce qui en fait une méthode robuste pour détecter les anomalies dans les distributions non normales.
— Pas de supposition de distribution : le test de IQR ne nécessite pas de présupposition de distribution normale.
— Détecte plusieurs anomalies simultanément : Il est capable de détecter plusieurs anomalies sans nécessiter d'itération.

Inconvénients :

— Non spécifique : Moins précis que les méthodes basées sur des distributions spécifiques, car il repose uniquement sur les quartiles.
— Moins efficace pour de petits ensembles de données : Lorsque la taille de l'échantillon.

2.2.2 Test de Z-Score

Définition 4

Le Z-Score est une mesure statistique qui permet d'identifier les anomalies ou valeurs aberrantes dans un ensemble de données. Il évalue combien un point de donnée s'écarte de la moyenne en termes d'écarts-types. Ce procédé est particulièrement utile dans divers domaines comme la cybersécurité, la finance et le contrôle qualité, pour repérer des motifs atypiques qui pourraient signaler des problèmes ou offrir des opportunités d'intervention. Le z-score standardise les données, ce qui facilite leur comparaison à travers différentes échelles et distributions. En transformant les données pour que leur moyenne soit 0 et leur écart-type soit 1, il rend la détection des anomalies plus simple et plus efficace[14].

2.2.2.1 Méthodologie Z-score

Un score Z est une mesure numérique qui décrit la relation entre une valeur et la moyenne d'un groupe de valeurs. Le score Z est mesuré en termes d'écarts-types par rapport à la moyenne. Si un score Z est égal à 0, le score des points de données est identique au score moyen.
La formule pour calculer un score Z est $z = \frac{(x-\mu)}{\sigma}$
Où
•x est la valeur observée.
• μ est la moyenne de la population.
• σ(sigma) est l'écart-type de la population.

Interprétation

Le z-score indique combien d'écarts-types une valeur donnée se trouve par rapport à la moyenne de l'ensemble de données.
• Un z-score de 0 signifie que la valeur est exactement égale à la moyenne.
• Un z-score positif signifie que la valeur est au-dessus de la moyenne.
• Un z-score négatif signifie que la valeur est en dessous de la moyenne.
• L'amplitude du z-score indique la distance en termes d'écarts-types entre la valeur et la moyenne. Par exemple, un z-score de 2 signifie que la valeur est deux écarts-types au-dessus de la moyenne.

2.2.2.2 Test d'hypothèse avec le z-score

Le z-score est fondamental dans les tests d'hypothèse, particulièrement pour les grands échantillons (n > 30) où la distribution d'échantillonnage peut être approximée par une distribution normale.

Étapes du test d'hypothèse :

1. Formuler les hypothèses :
— H_0 (Hypothèse nulle) : Il n'y a pas de différence significative ou l'effet observé est dû au hasard.
— H_1 (Hypothèse alternative) : Il y a une différence significative ou l'effet observé n'est pas dû au hasard.

2. Choisir un niveau de signification (α) : Couramment, α=0.05 pour un test avec un intervalle de confiance de 95%.

3. Calculer le z-score : Utiliser la formule du z-score pour l'échantillon ou la population étudiée.

4. Comparer avec la valeur critique : Comparer le z-score obtenu à la valeur critique pour le niveau de signification choisi. Les valeurs critiques courantes sont :
— Pour α=0.05 : ±1.96 .
— Pour α=0.01 : ±2.576 .

5. Prendre une décision : Si le z-score est supérieur à la valeur critique, rejeter H_0 . Sinon, ne pas rejeter H_0.

Exemple
Considérons une série temporelle représentant les températures quotidiennes en degrés Celsius au cours d'une période d'un mois :

Calcule de la Moyenne (μ) et de l'écart-Type (σ) :
1. Moyenne (μ) :

$$\mu = \frac{\sum_{i=1}^{30} x_i}{30}$$

En utilisant les données ci-dessus, la moyenne est approximativement 16.03 °C.

2. Écart-Type (σ) :

$$\sigma = \sqrt{\frac{\sum_{i=1}^{30} (x_i - \mu)^2}{30}}$$

L'écart-type calculé est environ 2.1403 °C.

Calcul des Z-Scores Appliquons la formule du Z-Score pour chaque température. Voici quelques calculs :
• Pour le Jour 1 (Température = 15.2 °C) : Z = -0.39
• Pour le Jour 10 (Température =23.5 °C) :Z=3.48
• Pour le Jour 26 (Température =23.0 °C) : Z=3.27

Identification des Anomalies
Les anomalies sont généralement définies comme des Z-Scores supérieurs à 2 ou inférieurs à -2. Utilisons ces seuils pour identifier les anomalies :
• Jour 10 (Z-Score ≈ 3.48) : Anomalie.
• Jour 26 (Z-Score≈ 3.27) : Anomalie.

Jours	Température
1	15.2
2	16.5
3	15.8
4	14.9
.	.
.	.
.	.
25	14.9
26	15.3
27	15.6
28	16.8
29	15.4
30	14.6

TABLE 2.2 – Les températures quotidiennes en degrés Celsius pendant un mois

Voici le graphe représentant les résultats

Ce graphique montre la série temporelle des températures quotidiennes (en °C) sur une période d'un mois, avec la détection d'anomalies mise en évidence. La courbe bleue représente les températures mesurées chaque jour. La ligne verte indique la moyenne des températures, tandis que les lignes pointillées rouge et bleu correspondent respectivement aux seuils supérieur et inférieur calculés. Avec le test de Z Score on remarque qu'il y a deux anomalies dans cette série temporelle a savoir le $10^{\text{ème}}$ et le $26^{\text{ème}}$ jour.

2.2.2.3 Avantages et inconvénient

Avantages :

— Efficacité : Capable de détecter plusieurs anomalies à la fois, sans nécessiter d'itération.

— Simplicité et rapidité : Les calculs sont simples et rapides à effectuer, avec une faible complexité computationnelle.

— Applicabilité à grande échelle : Le Z-Score est facilement applicable à de grands ensembles de données.

Inconvénients :

— Dépendance à la moyenne et à l'écart-type : Si la moyenne et l'écart-type sont fortement influencés par les anomalies, le Z-Score peut échouer à identifier certaines anomalies.

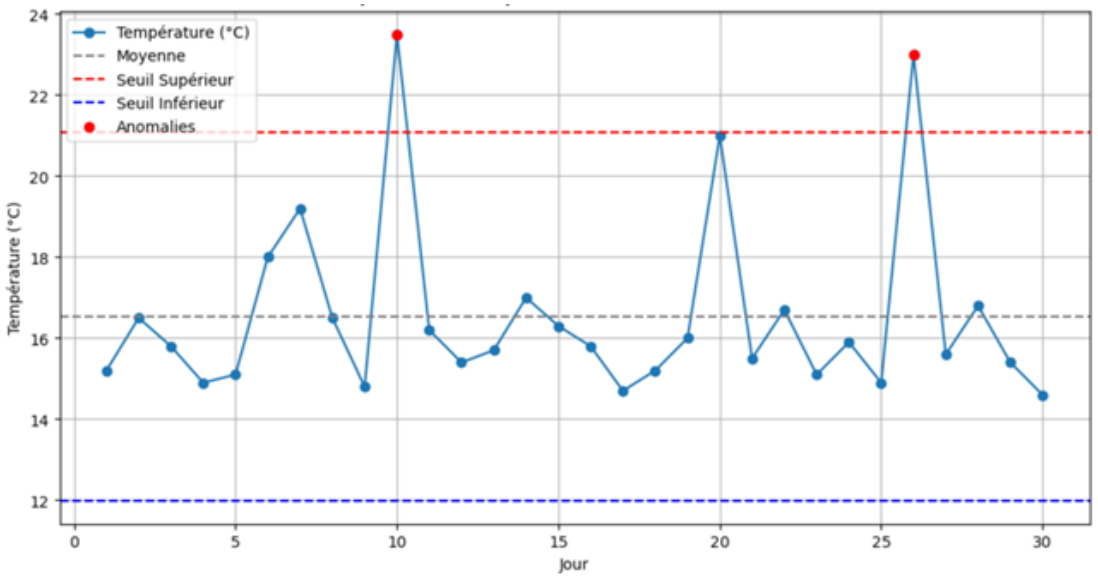


FIGURE 2.2 – Série Temporelle des Températures avec détection des Anomalies(Z-Score)

— Limitation aux données normalement distribuées : Comme le test de Grubbs, le Z-Score est mieux adapté aux données normalement distribuées.

— Sensibilité aux valeurs extrêmes : Peut sous-estimer ou ignorer les anomalies lorsque les valeurs extrêmes affectent la moyenne ou l'écart-type.

2.2.3 Comparaison entre les Méthodes Statistiques

Critères	**Z-Score**	**IQR**
Type de données	Distribution normale	Distribution non normale
Détection	Plusieurs anomalies possibles	Plusieurs anomalies possibles
Complexité	Faible	Faible
Applicabilité	Petits à grands échantillons	Petits à grands échantillons

TABLE 2.3 – Tableau Comparatif entre les méthodes statistiques

Discussion

• Le Z-Score est une méthode robuste pour des ensembles de données plus importants et plus complexes, offrant une approche simple et directe pour la détection d'anomalies. Néanmoins, il est sensible aux valeurs extrêmes et suppose que les données suivent une distribution normale.n

• L'IQR est une méthode robuste qui fonctionne bien même si les données ne sont pas normalement distribuées. Elle est particulièrement utile pour éviter les fausses détections causées par des valeurs extrêmes, mais peut ne pas être aussi sensible pour détecter des anomalies subtiles.

Les différentes méthodes de détection d'anomalies offrent des avantages distincts selon les caractéristiques des données et les objectifs d'analyse. Le choix de la méthode optimale dépend de la taille de l'échantillon, de la distribution des données et des spécificités de l'application.

2.3 Méthodes basées sur la prévision

Les méthodes de prévision jouent un rôle crucial dans la détection des anomalies en séries temporelles en fournissant une référence pour ce qui est considéré comme un comportement normal. Ces techniques permettent d'établir des modèles qui capturent les tendances, les saisons et les niveaux typiques des données temporelles, générant des estimations de ce à quoi devraient ressembler les valeurs futures en fonction des observations passées. La détection des anomalies repose sur la comparaison des valeurs observées avec ces prévisions : lorsqu'une valeur mesurée dévie de manière significative par rapport à ce qui est attendu, elle peut être signalée comme une anomalie.
Il existe de nombreuses méthodes de prévision pour la détection des anomalies, parmi lesquelles les moyennes mobiles, le lissage exponentiel et les modèles ARIMA. Ces techniques fournissent une base solide pour évaluer le comportement des séries temporelles. En intégrant des composantes telles que la tendance et la saisonnalité, elles permettent d'identifier les valeurs aberrantes dans un contexte complexe. Par exemple, les moyennes mobiles et les modèles ARIMA sont couramment utilisés pour prévoir les tendances et les comportements futurs des données, tandis que le lissage exponentiel est utile pour capturer les variations plus récentes des séries temporelles. La détection des anomalies devient ainsi une tâche systématique et quantitative, où les écarts par rapport aux prévisions indiquent des événements inhabituels ou des perturbations dans le comportement normal des données. En somme, les méthodes de prévision fournissent le cadre nécessaire pour discerner ce qui est atypique et potentiellement significatif dans une série temporelle, facilitant ainsi une analyse précise et informée des anomalies.

2.3.1 Régression linéaire pour la Détection d'Anomalies

Définition 5 *La régression linéaire est une méthode statistique utilisée pour modéliser la relation entre une variable dépendante et une ou plusieurs variables indépendantes. Son objectif est de trouver la ligne qui s'ajuste le mieux aux données, permettant ainsi de prédire les valeurs de la variable dépendante en fonction des variables indépendantes. Cette méthode est largement utilisée en analyse de données, en économie, en science et en ingénierie pour comprendre les relations et faire des prévisions[25].*
Il existe deux types principaux de régression linéaire :
• *Régression linéaire simple : lorsqu'il y a une seule variable indépendante.*
• *Régression linéaire multiple : lorsqu'il y a plusieurs variables indépendantes.*

2.3.1.1 Méthodologie

Lorsqu'une relation linéaire entre deux variables X et Y semble plausible d'après la distribution des points dans un graphique, on peut modéliser cette relation par une équation de la forme :$Y \approx aX + b$
Où a et b sont des coefficients inconnus. Le but est donc d'estimer ces coefficients à partir des données observées. Si les points du nuage étaient parfaitement alignés sur une droite, il serait simple de déterminer a et b : a représenterait la pente de la droite, et b serait l'ordonnée à l'origine (la valeur de Y lorsque X=0). Ce problème se résoudrait alors en résolvant un système de deux équations à deux inconnues en utilisant deux points du nuage. Cependant, en pratique, les points ne sont généralement pas parfaitement alignés, mais sont plutôt "proches" d'une droite. L'objectif est donc de trouver la droite qui passe le plus près des points du nuage. Pour cela, il faut mesurer l'écart des points par rapport à une droite D d'équation $Y \approx aX + b$ puis minimiser un critère d'erreur donné.
On peut envisager de minimiser :

— La somme des erreurs en valeur absolue :$\min \sum_{i=1}^{n} |y_i - ax_i - b|$

— La somme des erreurs au carré :min $\sum_{i=1}^{n}(y_i - ax_i - b)^2$ La méthode des moindres carrés, qui minimise le second critère, est la plus couramment utilisée et sera décrite dans la section suivante.

2.3.1.2 La méthode des moindres carrés

La méthode des moindres carrés cherche à trouver la droite y=ax+b qui minimise la somme des carrés des erreurs entre les valeurs observées et les valeurs prédites. Cette méthode est basée sur la minimisation de la fonction suivante :

$$g(a,b) = \sum_{i=1}^{n}(y_i - ax_i - b)^2$$

En minimisant cette fonction, on obtient les coefficients α et β de la droite de régression, qui sont donnés par les formules suivantes :

$$\alpha = \frac{\left(\frac{1}{n}\sum_{i=1}^{n} y_i x_i - \left(\frac{1}{n}\sum_{i=1}^{n} y_i\right)\left(\frac{1}{n}\sum_{i=1}^{n} x_i\right)\right)}{\left(\frac{1}{n}\sum_{i=1}^{n} x_i^2 - \left(\frac{1}{n}\sum_{i=1}^{n} x_i\right)^2\right)} = \frac{\mathrm{Cov}(X,Y)}{\mathrm{Var}(X)}$$

Et

$$\beta = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \alpha\frac{1}{n}\sum_{i=1}^{n} x_i\right) = \bar{Y} - \alpha\bar{X}$$

La droite d'équation $y = \alpha x + \beta$est appelée la droite de régression de Y en X, notée $\frac{\Delta Y}{X}$.

2.3.1.3 Régression Linéaire Simple et Détection d'Anomalies

La régression linéaire simple est un cas particulier où la variable dépendante Y est liée à une seule variable indépendante X par une équation linéaire de la forme :
$Y \approx aX + b$

Le processus consiste à estimer les coefficientsaet b, puis à utiliser cette droite de régression pour faire des prédictions. Une fois la droite ajustée, les écarts entre les valeurs observées y_i et les valeurs prédites $\hat{y_i}$=$aX_i + b$ peuvent être utilisés pour identifier des anomalies.

Exemple
Prenons les données suivantes

T1	T2	T3	T4	T5	T6	T7
120	140	190	130	125	145	200

TABLE 2.4 – Exemple (Régression Linéaire)

Nous allons suivre ces étapes :
• Ajuster un modèle de régression linéaire sur les données.
• Calculer les résidus pour évaluer la performance du modèle.
• Identifier les anomalies en analysant les résidus.

A l'aide du logiciel de programmation python on a réussi a calculer :

— Coefficients exacte de la régression linéaire : a = 6.607142857142856, b = 123.57142857142858

— Erreur quadratique moyenne (MSE) : 703.9540816326527
— Coefficient de détermination (R^2) : 0.19875145180023268

Remarque 1 *L'Erreur Quadratique Moyenne (MSE) de 703.95 indique que nos prédictions s'écartent significativement des valeurs réelles, suggérant que le modèle pourrait ne pas capturer efficacement la relation entre les variables. Le Coefficient de Détermination (R^2) de 0.20 montre que seulement* $20\,°C$ *de la variance des valeurs est expliquée par notre modèle, ce qui suggère une explication limitée des comportements des données. Ces résultats soulignent l'importance de considérer d'autres approches ou d'incorporer des variables supplémentaires pour améliorer la précision du modèle.*

Ci-dessous les graphes de la régression linaire et les résidus

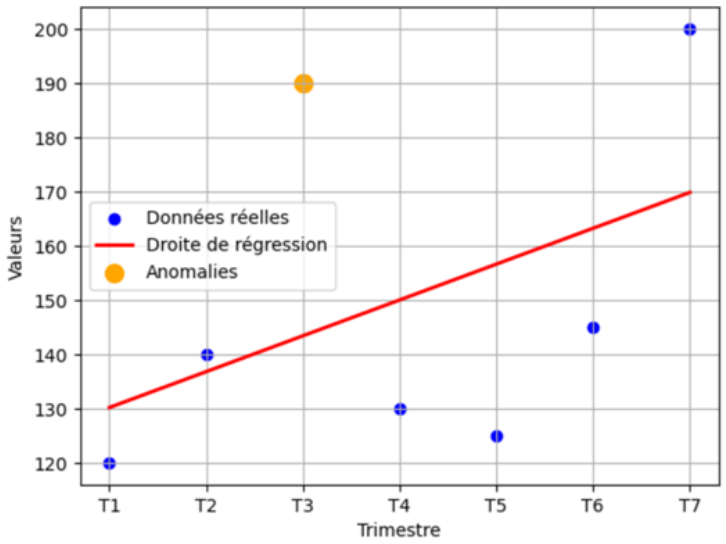


FIGURE 2.3 – Régression Linéaire :Valeurs par Trimestre avec Anomalies

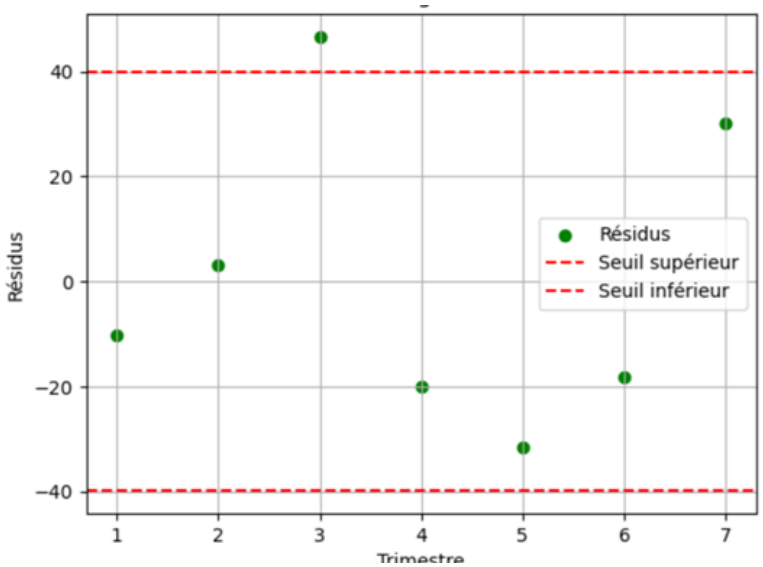


FIGURE 2.4 – Résidus de la Régression Linéaire

L'analyse des graphiques de régression linéaire et de résidus révèle une tendance générale à la hausse des valeurs au fil des trimestres. Cependant, le modèle linéaire ne capture pas entièrement la variabilité des données, comme en témoigne l'anomalie significative au trimestre T3. Cette déviation importante, visible à la fois dans le graphique de régression et dans celui des résidus où elle dépasse

le seuil supérieur, suggère la présence d'un facteur externe ou d'un événement ponctuel influençant fortement la valeur à cette période. Bien que T7 présente également un écart important par rapport à la ligne de régression, il n'est pas marqué comme une anomalie, indiquant peut-être une accélération de la tendance en fin de période plutôt qu'un événement exceptionnel. Cette observation souligne que, bien que la régression linéaire offre une approximation de la tendance générale, elle ne parvient pas à expliquer pleinement les variations observées, particulièrement pour les trimestres présentant des valeurs éloignées de la droite de régression.

2.3.1.4 Avantages et inconvénients

Avantages

— **Simplicité :** La régression linéaire est relativement facile à comprendre et à mettre en œuvre, ce qui en fait une bonne option pour les analyses préliminaires.

— **Interprétabilité :** Les coefficients du modèle peuvent être facilement interprétés, permettant de comprendre comment chaque variable indépendante influence la variable dépendante.

— **Modélisation des tendances :** Elle est efficace pour capturer les relations linéaires et les tendances dans les données, facilitant l'identification des valeurs aberrantes.

— **Rapidement calculable :** La méthode est rapide à exécuter, même sur de grands ensembles de données, ce qui est utile pour des analyses en temps réel.

Inconvénients

— **Sensibilité aux valeurs extrêmes :** Les valeurs aberrantes peuvent influencer de manière disproportionnée la ligne de régression, faussant ainsi les résultats et la détection d'anomalies.

— **Limites de la modélisation :** Elle ne peut modéliser que des relations linéaires, ce qui peut ne pas être suffisant pour des données présentant des relations complexes ou non linéaires.

— **Peu robuste aux fluctuations :** Dans des séries temporelles avec des variations importantes ou des changements de régime, la régression linéaire peut ne pas capturer efficacement la dynamique sous-jacente.

2.3.2 Méthode ARIMA pour la Détection d'Anomalies

Définition 6 *La méthode ARIMA (AutoRegressive Integrated Moving Average) est une approche couramment utilisée pour modéliser des séries temporelles non stationnaires. Elle combine trois composants essentiels :*

- ***Autorégression (AR) :*** *Capture la dépendance des valeurs actuelles sur les valeurs passées.*
- ***Différenciation (I) :*** *Rend la série stationnaire en supprimant les tendances et les saisons.*
- ***Moyenne mobile (MA) :*** *Modélise les erreurs de prévision en tenant compte des valeurs passées des erreurs[22].*

2.3.2.1 Décomposition du Modèle ARIMA

Autorégressif (AR) : On dit que X_t est un modèle autorégressif d'ordre p ($p \geq 1$), abrégé AR(p), si :

$$X_t = \phi_1 X_{t-1} + \phi_2 X_{t-2} + \cdots + \phi_p X_{t-p} + \varepsilon_t = \sum_{j=1}^{p} \phi_j X_{t-j} + \varepsilon_t$$

Où : $\phi_1, \phi_2, \ldots, \phi_p$ sont des constantes ($\phi_p \neq 0$) et $\varepsilon_t \sim BB(0, \sigma_\varepsilon^2)$. On utilise généralement la notation suivante :

$\Phi(L)X_t = \varepsilon_t$

Avec $\Phi(L) = 1 + \phi_1 L + \phi_2 L^2 + \cdots + \phi_p L^p$ On dit dans ce cas, qu'un modèle AR(p) est stationnaire si toutes les racines du polynôme caractéristique $\Phi(Z)$ sont de module supérieur strictement à 1.

Moyenne Mobile (MA)

On appelle modèle moyenne mobile (Moving Average) d'ordre q abrégé MA(q), un modèle stationnaire X_t qui vérifie

$X_t = c + \varepsilon_t + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_q \varepsilon_{t-q} = c + \varepsilon_t + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i}$

Où $c \in R, \theta_1, \ldots, \theta_q \in R$ ($\theta_q \neq 0$) et $\varepsilon_t \sim BB(0, \sigma_\varepsilon^2)$. Cette relation est équivalente à :

$X_t = \Theta(L)\varepsilon_t$

Où : $\Theta(L) = 1 + \theta_1 L + \theta_2 L^2 + \cdots + \theta_q L^q$

Les modelés MA(q) sont toujours stationnaires de moyenne c. **Différenciation (I)** Pour rendre la série stationnaire, des différenciations sont appliquées. Par exemple, une différenciation d'ordre 1 est :

$\nabla y_t = y_t - y_{t-1}$

2.3.2.2 Modèle ARIMA (p, d, q)

On appelle processus ARIMA (p, d, q) un processus pour lequel le processus différencié d'ordre d, qui vérifie l'équation suivante :

$\phi(B)\nabla^d y_t = \theta(B)\varepsilon_t$

Où :

— $\phi(B) = 1 - \phi_1 B - \phi_2 B^2 - \cdots - \phi_p B^p$
— $\theta(B) = 1 - \theta_1 B - \theta_2 B^2 - \cdots - \theta_q B^q$
— $\nabla^d y_t = (1 - B^d) y_t$

2.3.2.3 Détection d'Anomalies avec ARIMA

- **Étapes de Détection :**

— 1. Identification du Modèle : Déterminer les ordres p, d, et q appropriés à l'aide des graphiques ACF (AutocorrelationFunction) et PACF (Partial AutocorrelationFunction).

— 2. Estimation des Paramètres : Estimer les coefficients AR et MA en minimisant la fonction de coût, souvent la somme des carrés des erreurs.

— 3. Validation du Modèle : Tester le modèle en utilisant des résidus pour vérifier l'adéquation du modèle.

— 4. Prévision et Analyse des Résidus : Calculer les prévisions et analyser les résidus pour détecter les anomalies.

— 5. Analyse des Résidus : Les résidus, qui sont la différence entre les valeurs observées et les valeurs prédites, doivent être proches de zéro pour un modèle bien ajusté. Les anomalies sont identifiées lorsque les résidus sont significativement éloignés de zéro. Par exemple, une règle courante est d'utiliser des

seuils basés sur les quantiles des résidus ou les écarts-types pour détecter les anomalies.

Exemple :
Prenons les données suivantes Data = [20, 25, 19, 17, 15,70, 32, 18]
Appliquons la méthode de détection d'anomalie avec ARIMA

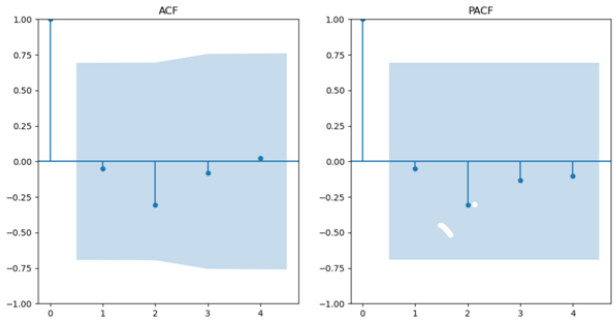


FIGURE 2.5 – ACF-PACF

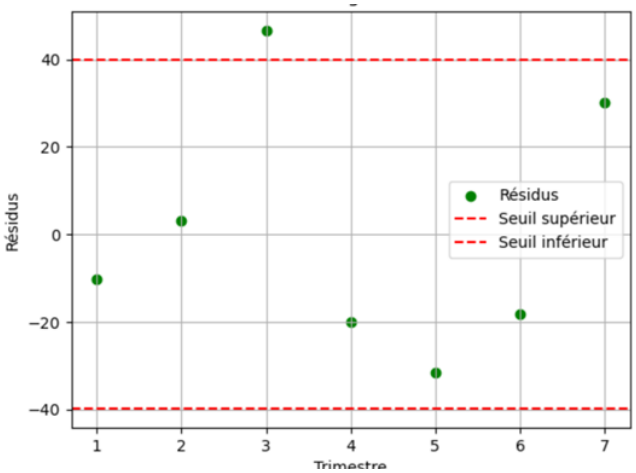


FIGURE 2.6 – Données avec Anomalies Détectées(ARIMA)

La valeur 70 est bien identifiée comme anomalie en raison de son écart important par rapport aux autres valeurs. Cependant, la valeur 2, qui est également un écart important par rapport au reste de la série, n'a pas été détectée comme anomalie.

2.3.2.4 Avantages de la Méthode ARIMA

— **Flexibilité :** ARIMA peut modéliser une large gamme de séries temporelles en combinant des composantes AR et MA, et en rendant les séries stationnaires via différenciation.

— **Robustesse :** Le modèle est robuste pour les séries temporelles avec des tendances ou des saisons qui peuvent être transformées en stationnarité.

subsubsectionLimites de la Méthode ARIMA

— **Stationnarité :** ARIMA nécessite que les séries temporelles soient stationnaires. Les séries non stationnaires doivent être différenciées, ce qui peut compliquer la modélisation.

— **Sensibilité aux Paramètres :** Le choix des paramètres p, d, et q peut influencer fortement les résultats. Une mauvaise sélection peut entraîner une mauvaise détection des anomalies.

— **Anomalies Moins Évidentes :** Les anomalies qui ne dévient pas considérablement des prévisions peuvent ne pas être détectées, surtout si elles ne sont pas suffisamment éloignées des valeurs prévues.

2.3.3 Lissage exponentiel

Définition 7 *Introduit par Holt en 1958 et Winters en 1960, et popularisé par Brown en 1963, le lissage exponentiel regroupe un ensemble de méthodes empiriques de prévision. Celles-ci accordent une importance variable aux valeurs passées d'une série temporelle en utilisant des pondérations exponentielles décroissantes. Autrement dit, les observations récentes sont davantage pondérées que les anciennes[13].*

Cette technique est fréquemment employée pour la prévision de séries chronologiques telles que les ventes, les stocks, ou les revenus, et s'adapte à différents modèles de données. Elle fournit une méthode simple et rapide pour générer des prévisions à court terme en estimant les niveaux et les tendances à partir des données historiques.

Selon la présence ou non de saisonnalité dans la série, on distingue trois principaux types de lissage exponentiel :
• **Lissage exponentiel simple (LES) :** utilisé lorsqu'il n'y a ni tendance ni saisonnalité.
• **Lissage exponentiel double (méthode de Holt)** : adapté aux séries avec tendance mais sans saisonnalité.
• **Lissage exponentiel triple (méthode de Holt-Winters) :** utilisé pour les séries comportant à la fois une tendance et une saisonnalité. En plus de sa simplicité, elle peut aussi être appliquée à la détection d'anomalies en identifiant des écarts anormaux dans les tendances des données historiques.

Parmi les variantes du lissage exponentiel, seule la méthode de Holt-Winters (lissage triple) permet de capturer simultanément la tendance et la saisonnalité via des composantes additives ou multiplicatives. Ce choix s'impose donc pour la détection d'anomalies dans des séries temporelles présentant une structure temporelle complexe, où les approches simples (lissage simple ou double) se révèlent insuffisantes.

2.3.3.1 Lissage exponentiel triple

Le lissage exponentiel triple, ou méthode Holt-Winters, est une technique avancée de prévision pour les séries temporelles qui intègre la tendance, la saisonnalité et les erreurs résiduelles. Elle est utilisée pour modéliser des données temporelles en tenant compte des variations à long terme, des cycles saisonniers et des fluctuations aléatoires[17].
Composantes Principales :
— Composante de Tendance (Trend) :
o Modélise les variations à long terme (hausse ou baisse) des données.
— Composante de Saison (Seasonality) :
o Capture les variations cycliques régulières dans les données, comme les effets saisonniers mensuels ou annuels.
— Composante d'Erreur (Error) :
o Représente les écarts aléatoires non expliqués par les tendances et la saisonnalité.

2.3.3.2 Étapes du Lissage Exponentiel Triple

Le processus commence par l'initialisation des paramètres pour les composantes de tendance, de saisonnalité et d'erreur, souvent à partir des données historiques.
Les formules :

$$T_0 = (\alpha \cdot y_1) + (1 - \alpha) \cdot y_2$$
$$S_0 = \frac{\sum y_i}{n}$$
$$E_0 = y_1 - (T_0 + S_0)$$

Avec :

— T_0 Est la tendance initialisée
— S_0 Est Saison initiale
— E_0 Est Erreur initiale
— y_1 et y_2 Sont les deux premières valeurs de la serie
— α Est un facteur d'appréciation de la tendance (généralement choisi entre 0,1 et 0,3).
— yi Sont les valeurs pour une période saisonnière donnée.
— n Est le nombre de périodes saisonnières.

Mise à jour de la tendance
La tendance est mise à jour en utilisant une formule d'ajustement qui prend en compte la nouvelle observation et l'estimation précédente de la tendance.

$$T_t = \alpha \cdot (Y_t - S_{t-m}) + (1 - \alpha) \cdot (T_{t-1} + T_{t-1S})$$

Mise à jour de la saisonnalité
La composante de saisonnalité est mise à jour en fonction des observations saisonnières actuelles et des estimations précédentes de la saisonnalité.

$$S_t = \beta \cdot (Y_t - T_{t-1S}) + (1 - \beta) \cdot S_{t-1}$$

Où :
β est un facteur de lissage pour la saisonnalité, généralement choisi entre 0 et 1.
Mise à jour de l'erreur
Les erreurs résiduelles sont calculées en soustrayant les estimations de la tendance et de la saisonnalité des données observées.

$$E_t = Y_t - T_(t - S)$$

Prévision
Le modèle utilise les estimations mises à jour pour prédire les valeurs futures de la série temporelle, en incluant la tendance, la saisonnalité et l'erreur.

$$Y_(t + 1) = T_t + S_t + E_t$$

Répétition :
Les étapes de mise à jour et de prévision sont répétées pour chaque nouvelle observation dans la série temporelle.

Exemple :
Supposons que l'on dispose de l'ensemble de données suivant :

Data = [10, 12, 14, 13, 15, 50, 23, 12, 14, 13, 15, 11, 12]

Nous allons supposer une saisonnalité de période m = 4 (purement illustrative ici), avec une anomalie évidente en position 5 (valeur = 50).

Initialisation manuelle et hyperparamètres

— α (niveau) = 0.5
— β (tendance) = 0.4
— γ (saisonnalité) = 0.3
— m (période saisonnière) = 4

Étape 1 : Initialiser les composants Nous avons besoin des 4 premières observations pour initialiser :

t y_t
0 10
1 12
2 14
3 13

$$\text{Niveau initial } (l_0) : \text{ moyenne des 4 premiers points} l_0 = \frac{10 + 12 + 14 + 13}{4} = \frac{49}{4} = 12.25$$

Tendance initiale (b_0) : moyenne des différences successives

$$b_0 = \frac{(12 - 10) + (14 - 12) + (13 - 14)}{3} = \frac{2 + 2 - 1}{3} = \frac{3}{3} = 1$$

$$\text{Saisonnalité initiale } (s_0 \text{ à } s_3) : \text{ saisonnalité estimée en soustrayant } l_0 \text{ de chaque point}$$

$$\begin{aligned} s_0 &= y_0 - l_0 = 10 - 12.25 = -2.25 \\ s_1 &= y_1 - l_0 = 12 - 12.25 = -0.25 \\ s_2 &= y_2 - l_0 = 14 - 12.25 = +1.75 \\ s_3 &= y_3 - l_0 = 13 - 12.25 = +0.75 \end{aligned}$$

Étapes de calcul Holt-Winters (Additif)

On applique ensuite les formules pour t = 4 jusqu'à t = 12.
Rappels :

$$\begin{aligned} l_t &= \alpha(y_t - s_{t-m}) + (1 - \alpha)(l_{t-1} + b_{t-1}) \\ b_t &= \beta(l_t - l_{t-1}) + (1 - \beta)b_{t-1} \\ s_t &= \gamma(y_t - l_t) + (1 - \gamma)s_{t-m} \\ \hat{y}_t &= l_{t-1} + b_{t-1} + s_{t-m} \end{aligned}$$

Exemple pour t = 4

y4=15

$s_0 = -2.25$

$l_3 = 12.25, \quad b_3 = 1$

Calcul du niveau :

$l_4 = 0.5(15 - (-2.25)) + 0.5(12.25 + 1) = 0.5(17.25) + 0.5(13.25) = 8.625 + 6.625 = 15.25$

Calcul de la tendance :

$b_4 = 0.4(15.25 - 12.25) + 0.6 \cdot 1 = 0.4 \cdot 3 + 0.6 = 1.2 + 0.6 = 1.8$

Calcul de la saisonnalité :

$s_4 = 0.3(15 - 15.25) + 0.7 \cdot (-2.25) = -0.075 - 1.575 = -1.65$

Prévision (t = 4) :

$\hat{y}_4 = 12.25 + 1 + (-2.25) = 11.0$

Résidu :

$y_4 - \hat{y}_4 = 15 - 11 = 4$

Exemple pour $t = 5$ (anomalie) :

$y_5 = 50, \quad s_1 = -0.25$

$l_5 = 0.5 \times (50 - (-0.25)) + 0.5 \times (15.25 + 1.8) = 0.5 \times 50.25 + 0.5 \times 17.05 = 25.125 + 8.525 = 33.65$

$b_5 = 0.4 \times (33.65 - 15.25) + 0.6 \times 1.8 = 0.4 \times 18.4 + 1.08 = 7.36 + 1.08 = 8.44$

$s_5 = 0.3 \times (50 - 33.65) + 0.7 \times (-0.25) = 0.3 \times 16.35 - 0.175 = 4.905 - 0.175 = 4.73$

Prévision :

$\hat{y}_5 = 15.25 + 1.8 + (-0.25) = 16.8$

Résidu :

$50 - 16.8 = 33.2 \quad \rightarrow \quad$ très élevé $\quad \rightarrow \quad$ anomalie détectée

Détection d'anomalie

Après avoir estimé toutes les valeurs de prévision et les résidus, on fixe un seuil :
Seuil $= \mu_{\text{résidus}} \pm 2 \cdot \sigma_{\text{résidus}}$

Si un résidu dépasse ce seuil, alors on détecte une **anomalie**.

Le lissage exponentiel triple modélise la tendance et la saisonnalité de la série, permettant d'estimer des valeurs prédites précises. Les anomalies sont détectées lorsque les résidus, c'est-à-dire l'écart entre valeurs observées et prévues, dépassent un seuil fixé à ± 2 fois l'écart-type des résidus.

Dans l'exemple, la valeur 50 est clairement identifiée comme anomalie, ce qui montre l'efficacité de la méthode pour repérer les points atypiques tout en tenant compte des variations saisonnières

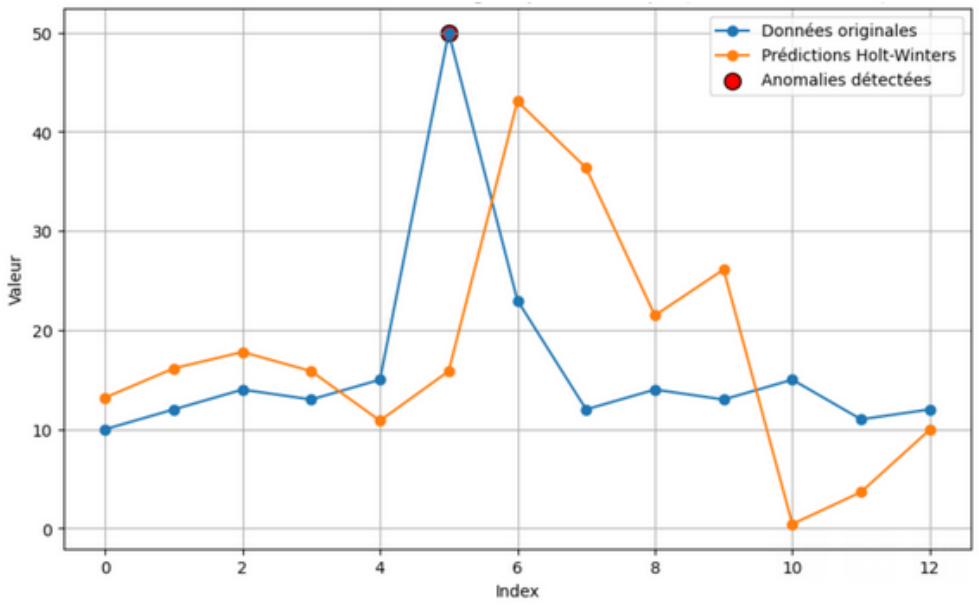


FIGURE 2.7 – Caption

normales.

2.3.3.3 Avantages et Inconvénients

Avantages

— Simplicité :
Le lissage exponentiel est facile à comprendre et à mettre en œuvre, ce qui en fait une méthode accessible pour les analyses rapides.

— Réactivité aux Changements :
Le lissage exponentiel donne plus de poids aux observations récentes, ce qui permet à la méthode de réagir rapidement aux changements dans les données, y compris les anomalies émergentes.

— Moins de Paramètres à Ajuster :
Comparé à des modèles plus complexes comme ARIMA, le lissage exponentiel nécessite généralement moins de paramètres à ajuster, ce qui simplifie le processus de configuration.

— Calcul Efficace :
Le calcul est rapide, ce qui est utile pour des analyses en temps réel ou sur des ensembles de données volumineux.

Inconvénients

— Sensibilité aux Choix des Paramètres :
Malgré l'optimisation des paramètres, le choix du facteur de lissage peut grandement influencer les résultats. Un facteur trop faible peut ne pas capturer les anomalies, tandis qu'un facteur trop élevé peut introduire trop de bruit.

— Moins Performant pour les Séries avec Tendance ou Saisonnières :
Le lissage exponentiel simple ne prend pas en compte les tendances ou les effets saisonniers, ce qui peut limiter son efficacité sur des séries temporelles avec ces caractéristiques.

— Détection des Anomalies Peu Précise :
La méthode peut ne pas détecter correctement toutes les anomalies, surtout si celles-ci sont de faible amplitude ou se produisent dans des contextes de forte variabilité.

— Pas de Modélisation de la Dynamique Complète :
Contrairement à des modèles comme ARIMA, le lissage exponentiel ne modélise pas les relations complexes ou les dépendances temporelles entre les observations.
— Dépendance aux Données Passées :
Les prévisions et les détections d'anomalies reposent sur l'hypothèse que les données passées sont représentatives du comportement futur, ce qui peut être problématique en cas de changements soudains ou structurels dans les données.

2.4 Comparaison entre les méthodes de détection d'anomalie basées sur la prévision

Méthode	**Avantages**	**Inconvénient**
Régression linéaire	- Simplicité et interprétabilité - Permet des prédictions sur la variable dépendante	- Sensibilité aux valeurs extrêmes
ARIMA	- Modélisation avancée et capacité à capturer des relations complexes - Flexibilité pour différents types de données	- Complexité de mise en œuvre - Sensibilité aux choix des paramètres
Lissage exponentiel	- Adaptabilité aux tendances et variations saisonnières - Facilité de mise en œuvre	- Sensibilité aux choix des paramètres - Non adapté aux changements soudains

Table 2.5 – Tableau de Comparaison entre les méthodes de détection d'anomalie basées sur la prévision

Interprétation :

• La régression linéaire se distingue par sa simplicité et sa capacité à fournir des interprétations claires grâce à ses coefficients. Cependant, son efficacité est limitée aux relations linéaires, et elle peut être influencée par des anomalies qui faussent la ligne de régression.

• En revanche, ARIMA offre une approche plus complexe mais puissante pour la modélisation des séries temporelles. Sa capacité à capturer des relations non linéaires et des comportements variés le rend adapté à des ensembles de données plus sophistiqués. Toutefois, sa mise en œuvre nécessite une expertise approfondie, et le choix des paramètres peut s'avérer délicat.

• SARIMA, en tant qu'extension d'ARIMA, ajoute une dimension saisonnière à la modélisation des séries temporelles. Il est particulièrement efficace pour les données présentant des cycles saisonniers réguliers, permettant une modélisation plus précise des effets saisonniers. Cependant, la complexité accrue dans la configuration des paramètres saisonniers peut rendre SARIMA plus difficile à utiliser, et il nécessite des données suffisamment longues pour estimer correctement les paramètres saisonniers.

• Dans l'ensemble, le choix de la méthode dépendra des caractéristiques spécifiques des données à analyser. Pour des données avec des relations linéaires simples, la régression linéaire peut suffire. Pour des séries temporelles plus complexes, ARIMA peut offrir de meilleures performances en matière de détection d'anomalies. SARIMA est particulièrement adapté lorsque des effets saisonniers doivent être modélisés avec précision.

2.5 Méthodes basées sur le Machine Learning

2.5.1 Machine Learning

Définition 8
La notion d'apprentissage soulève des questions essentielles : qu'est-ce que cela signifie réellement d'apprendre, et comment ce processus se déroule-t-il, que ce soit pour un être humain, un animal, ou une machine ? Cette interrogation capte l'intérêt de nombreux domaines, allant de l'informatique et des mathématiques aux neurosciences, à l'éducation, à la philosophie et aux arts.

Une définition qui englobe diverses entités, qu'il s'agisse de programmes informatiques, de robots, d'animaux de compagnie ou d'êtres humains, est celle fournie par Fabien Benureau (2015) : « L'apprentissage est une modification d'un comportement sur la base d'une expérience. »

Dans le cadre des programmes informatiques, qui est notre principal sujet ici, le terme d'apprentissage automatique ou machine Learning désigne la capacité d'un programme à acquérir des connaissances sans que les modifications soient spécifiquement codées. Cette idée a été introduite par Arthur Samuel en 1959. On peut ainsi distinguer un programme conventionnel, qui applique une série de règles sur les données d'entrée pour générer des résultats, d'un programme d'apprentissage automatique, qui utilise les données et les résultats pour établir la procédure permettant d'obtenir les résultats à partir des données[23].

2.5.2 Types de Machine Learning

Il existe plusieurs types d'apprentissage automatique, chacun adapté à des situations différentes :

— **Apprentissage supervisé :** Le type de ML le plus facile à appréhender, son but d'apprendre a faire des prédiction a partir d'une liste d'exemple étiquetés. Ces étiquettes servent de professeure et supervisent l'apprentissage de l'algorithme.
Cette branche de Machine Learning s'intéresse aux problèmes pouvant être formalisées de la façon suivante :
On a n observations : X_i, $i = (1, n)$ dans un espace X, leurs étiquètes sont Y_i, $i = (1, n)$ dans un espace Y. On suppose que les Y_i peuvent être obtenue a partir des observations X_i grâce a une fonction $\emptyset : X \to Y$ tq $Y_i = \emptyset(X_i) + \varepsilon_i$
Ou ε_i est un bruit aleatoire .
Exemples :
•Classification : Prédire une catégorie parmi un ensemble fini de possibilités. Exemple : Déterminer si un email est un spam ou non.
• Régression : Prédire une valeur numérique continue. Exemple : Prédire le prix d'une maison en fonction de ses caractéristiques.

— **Apprentissage non supervisé :** L'apprentissage non supervisé est une branche du Machine Learning qui traite des données non étiquetées. Son objectif principal est de découvrir des structures ou des motifs cachés dans les données sans avoir besoin de supervision externe. Cette branche de Machine Learning s'intéresse aux problèmes pouvant être formalisées de la façon suivante :
On a n observations : X_i, $i = (1, n)$ dans un espace X. Il n'y a pas d'étiquettes associées à ces observations. On cherche à trouver une structure ou une représentation $f(X)$ qui capture les caractéristiques essentielles ou les relations intrinsèques dans X.

Le but de l'apprentissage non supervisé est de découvrir cette structure ou représentation f à partir des données non étiquetées fournies.

Exemples :

• Clustering : Regrouper des données similaires en clusters. Exemple : Segmentation de clients pour le marketing, regroupement de documents par thème.

• Réduction de dimensionnalité : Réduire le nombre de variables à considérer tout en préservant l'essentiel de l'information. Exemple : Analyse en composantes principales (ACP), compression d'images.

— **Apprentissage semi-supervisé :**

Un compromis entre l'apprentissage supervisé et non supervisé, où une petite partie des données est étiquetée, tandis que la majorité ne l'est pas. Le modèle utilise les données étiquetées pour guider son apprentissage sur les données non étiquetées.

Cette branche de Machine Learning s'intéresse aux problèmes pouvant être formalisés de la façon suivante :

On a n observations : X_i, $i = (1, n)$ dans un espace X. Parmi ces observations, seules les m premières sont étiquetées Y_i, $i = (1, m)$ dans un espace Y, où $m < n$.

On cherche à apprendre une fonction f en utilisant à la fois les données étiquetées et non étiquetées Le but de l'apprentissage semi-supervisé est d'exploiter la structure des données non étiquetées pour améliorer la performance de prédiction sur les données étiquetées.

Exemples :

•Classification semi-supervisée : Améliorer la précision de la classification en utilisant des données non étiquetées. Exemple : Amélioration de la classification des images en utilisant un grand nombre d'images non étiquetées.

•Régression semi-supervisée : Améliorer les prédictions de valeurs continues en utilisant des données non étiquetées. Exemple : Prédiction plus précise des prix immobiliers en utilisant des données de propriétés non étiquetées.

•Clustering avec contraintes : Effectuer un clustering en utilisant à la fois des données non étiquetées et des contraintes dérivées des données étiquetées. Exemple : Regroupement de documents avec quelques exemples de catégories connues.

— **Apprentissage par renforcement :**

Dans cette approche, un agent apprend à prendre des décisions en interagissant avec un environnement dynamique. Il reçoit des récompenses ou des pénalités en fonction de ses actions, et l'objectif est de maximiser la récompense totale au fil du temps.

Exemples : Robots autonomes, jeux vidéo.

2.5.3 Importance du Machine Learning

Le Machine Learning a révolutionné de nombreux domaines, des technologies médicales aux services financiers, en passant par les systèmes de recommandation et les voitures autonomes. Grâce à sa capacité à traiter de grandes quantités de données rapidement et efficacement, le Machine Learning permet de résoudre des problèmes complexes que les humains ou les systèmes traditionnels ne pourraient pas résoudre. Il est également un pilier fondamental dans des applications modernes telles telles que la reconnaissance vocale, la vision par ordinateur, et la détection des fraudes.[22]

2.5.4 Applications dans la Détection d'Anomalies

Un domaine dans lequel le Machine Learning s'est avéré particulièrement puissant est celui de la détection d'anomalies. En identifiant des comportements ou des événements inhabituels dans les données, le Machine Learning aide à prévenir les fraudes, détecter des pannes dans des systèmes in-

dustriels, ou encore repérer des cyberattaques dans des réseaux.
Dans la suite de cette section, nous explorerons les différentes méthodes de détection d'anomalies basées sur le Machine Learning, en détaillant leur fonctionnement et leurs avantages respectifs.

2.6 L'Isolation Forest (iForest)

L'Isolation Forest (iForest) est un algorithme de détection d'anomalies non supervisé qui s'est avéré particulièrement efficace pour identifier les points atypiques dans les jeux de données, notamment les séries temporelles. Contrairement à de nombreuses autres méthodes qui tentent de modéliser le comportement "normal" des données pour ensuite identifier ce qui en dévie, l'Isolation Forest adopte une perspective inversée : elle se concentre directement sur l'isolement des anomalies[19].

Définition 9 *L'Isolation Forest est un algorithme basé sur l'ensemble d'arbres, inspiré du principe des Random Forests. Il repose sur l'hypothèse fondamentale que les anomalies sont des points de données qui sont "peu nombreux et différents" du reste des observations. Cette caractéristique les rend intrinsèquement plus faciles à séparer ou à "isoler" du gros des données normales.*

2.6.1 Concept de l'Isolation

Le principe est que les anomalies, étant des points rares et structurellement distincts, se trouvent souvent plus éloignées des autres observations "normales" dans l'espace des caractéristiques. Par conséquent, pour séparer un point anormal du reste du jeu de données, un nombre relativement faible de divisions aléatoires est généralement suffisant. À l'inverse, un point normal, étant densement entouré par de nombreux autres points, nécessitera un plus grand nombre de divisions pour être complètement isolé de ses voisins. L'algorithme *Isolation Forest* exploite cette propriété pour détecter les anomalies [20].

2.6.2 Définition Mathématique

L'efficacité de l'Isolation Forest est quantifiée par la longueur de chemin d'une instance dans un arbre d'isolation et un score d'anomalie normalisé.

— Longueur de Chemin (h(x)) : Pour une instance de données x, sa longueur de chemin h(x) dans un arbre d'isolation donné est définie comme le nombre d'arêtes traversées à partir du nœud racine de l'arbre jusqu'à ce que l'instance x atteigne un nœud terminal (une feuille). Un chemin plus court indique que l'instance est plus facile à isoler.

— Facteur de Normalisation (c(n)) : Pour rendre les longueurs de chemin comparables entre des arbres construits sur des sous-échantillons de tailles différentes, une constante de normalisation c(n) est utilisée. Cette constante représente la longueur de chemin moyenne d'une recherche infructueuse dans un arbre de recherche binaire (Binary Search Tree - BST) de n nœuds. Elle est calculée comme suit :

$$c(n) = 2H(n-1) - \frac{n}{2(n-1)}$$

où $H(k)$ est le nombre harmonique, qui peut être approximé par :

$$H(k) \approx \ln(k) + \gamma,$$

avec γ étant la constante d'Euler–Mascheroni ($\gamma \approx 0{,}5772156649$).
Pour les cas où $n \leq 1$, on pose :

$$c(n) = 1.$$

— Score d'Anomalie ($S(x)$) : Le score d'anomalie $S(x)$ d'une instance x est dérivé de sa moyenne des longueurs de chemin $E(h(x))$ à travers tous les arbres d'isolation de la forêt. Le score est normalisé par $c(n)$ et défini par la formule :
$S(x) = 2^{-\frac{E(h(x))}{c(n)}}$

Interprétation du Score

— Si $E(h(x))$ est très petit (chemin court), alors $S(x)$ tend vers 1. Cela indique une forte probabilité que x soit une anomalie.
— Si $E(h(x))$ est proche de $c(n)$ (chemin moyen), alors $S(x)$ tend vers 0,5. Cela indique que x est probablement une observation normale.
— Si $E(h(x))$ est très grand (chemin long), alors $S(x)$ tend vers 0. Cela indique que x est fortement considéré comme une observation normale.

2.6.3 Méthodologie

L'algorithme de l'Isolation Forest suit une méthodologie en deux phases principales : la phase d'entraînement (construction de la forêt) et la phase d'inférence (calcul des scores d'anomalie).

— **Phase d'Entraînement** : Construction de l'Isolation Forest
  — *Sélection des Sous-Échantillons :* L'algorithme prend en entrée un jeu de données D et deux paramètres : le nombre d'arbres d'isolation à construire (T, typiquement 100 à 256) et la taille du sous-échantillon à utiliser pour chaque arbre (ψ, généralement 256 ou 512). Pour chaque arbre $t \in [1, T]$:
  — Un sous-échantillon aléatoire de taille ψ est tiré sans remplacement du jeu de données D.
  — Cet échantillonnage aléatoire est crucial pour introduire de la diversité entre les arbres et rendre l'algorithme robuste.

— **Construction d'un Arbre d'Isolation (iTree)** : Pour chaque sous-échantillon, un arbre d'isolation binaire est construit de manière récursive :
  — *Sélection Aléatoire des Conditions de Division :* À chaque nœud interne de l'arbre, une caractéristique (dimension) q est choisie aléatoirement parmi toutes les caractéristiques disponibles. Ensuite, une valeur de division p est choisie aléatoirement et uniformément entre la valeur minimale et maximale de la caractéristique q dans le sous-ensemble de données actuel.
  — *Partitionnement :* Les points de données sont alors dirigés vers le sous-arbre gauche si leur valeur pour la caractéristique q est inférieure à p, et vers le sous-arbre droit sinon.

  — **Conditions de Terminaison** : Le processus de division récursive s'arrête (un nœud devient une feuille) si l'une des conditions suivantes est remplie :
  — Le nœud ne contient qu'une seule instance de données (l'instance est isolée).
  — Tous les points de données dans le nœud ont des valeurs identiques pour toutes les caractéristiques (aucune division supplémentaire n'est possible).
  — La profondeur de l'arbre a atteint une limite prédéfinie (max_depth). Cette limite est souvent fixée à log 2 (ψ)pour éviter unewline n sur-ajustement et s'assurer que les anomalies sont identifiées tôt.

— **Phase d'Inférence** : Calcul du Score d'Anomalie pour une Nouvelle Instance
  — *Parcours de la Forêt* : Pour chaque nouvelle instance de données x que l'on veut évaluer :
  — L'instance x est propagée à travers chaque iTree de la forêt, en suivant les règles de division à

chaque nœud, jusqu'à atteindre une feuille.
  - La longueur de chemin $h(x)$ est enregistrée pour chaque arbre.
- Calcul de la Moyenne des Longueurs de Chemin : La moyenne des longueurs de chemin de l'instance x est calculée sur tous les T iTrees de la forêt, ce qui donne $E(h(x))$.
- Calcul du Score d'Anomalie Normalisé : Enfin, le score d'anomalie $S(x)$ est calculé en utilisant la formule mentionnée précédemment : $S(x) = 2^{-\frac{E(h(x))}{c(n)}}$
- Décision d'Anomalie : L'instance x est classifiée comme une anomalie si son score $S(x)$ dépasse un seuil prédéfini. Ce seuil est souvent déterminé empiriquement ou basé sur le pourcentage d'anomalies attendues dans les données (contamination parameter).

- **Décision d'Anomalie :** L'instance x est classifiée comme une anomalie si son score $S(x)$ dépasse un seuil prédéfini. Ce seuil est souvent déterminé empiriquement ou basé sur le pourcentage d'anomalies attendues dans les données (contamination parameter).

2.6.4 ExempleComplet et Détaillé de l'Isolation Forest pour une Série Temporelle

Nous allons utiliser la série temporelle que vous avez fournie et identifier les étapes clés de l'algorithme.

Série Temporelle (TS) : [40, 26, 43, 36, 90, 41, 80]Anomalies Attendues : 90 et 80.

L'Isolation Forest (iForest) opère sur des données tabulaires (des instances avec des caractéristiques), et non directement sur la séquence brute d'une série temporelle. La première étape cruciale est donc l'ingénierie des caractéristiques (Feature Engineering) pour transformer notre série en un ensemble d'instances.

1-Ingénierie des Caractéristiques à partir de la Série Temporelle
Nous allons utiliser une fenêtre glissante (sliding window) pour créer des instances multidimensionnelles. C'est une méthode standard pour adapter les algorithmes de Machine Learning aux séries temporelles.
Choix de la Taille de la Fenêtre : Pour cet exemple détaillé, nous choisirons une fenêtre de taille 2. Chaque "instance" représentera un point de la série temporelle en incluant sa valeur actuelle et sa valeur précédente.

- F_1 = Valeur Actuelle
- F_2= Valeur Précédente

À partir de TS = [40, 26, 43, 36, 90, 41, 80], nos instances sont :
- Instance X1 (pour le point 26) : (F1=26,F2=40)
- Instance X2 (pour le point 43) : (F1=43,F2=26)
- Instance X3 (pour le point 36) : (F1=36,F2=43)
- Instance X4 (pour le point 90) : (F1=90,F2=36) ← Contient l'anomalie 90
- Instance X5 (pour le point 41) : (F1=41,F2=90) ← Contient l'anomalie 90 (comme valeur précédente)
- Instance X6 (pour le point 80) : (F1=80,F2=41) ← Contient l'anomalie 80

Nous avons maintenant N=6 instances, chacune avec 2 caractéristiques.

2-Paramètres de l'Isolation Forest pour l'Exemple

Pour une démonstration complète, nous allons construire une petite forêt de T=2 arbres d'isolation (iTrees). En réalité, une Isolation Forest typique utilise entre 100 et 256 arbres pour une meilleure robustesse.

— Nombre d'instances (N) : 6
— Nombre d'arbres (T) : 2
— Taille du $sous - chantillon$ (ψ) : Pour cet exemple, nous utiliserons toutes les instances pour chaque arbre ($\psi = N = 6$).
— Profondeur maximale de l'arbre (max-depth) : La profondeur maximale est $\lceil \log_2(N) \rceil = \lceil \log_2(6) \rceil = \lceil 2.58 \rceil = 3$ Un arbre ne pourra pas dépasser 3 divisions.

3-Construction des Arbres d'Isolation (iTrees) et Calcul des Longueurs de Chemin

Les choix de caractéristiques et de points de division sont aléatoires. Pour les besoins de cet exemple pédagogique, nous allons fixer ces choix pour illustrer clairement le processus.

Instances (pour référence facile) :
— X1=(26,40)
— X2=(43,26)
— X3=(36,43)
— X4=(90,36)
— X5=(41,90)
— X6=(80,41)

Arbre d'Isolation 1 (iTree-1)

Nœud Racine (Profondeur 0) : $[X_1, X_2, X_3, X_4, X_5, X_6]$

Division 1 :
— Caractéristique choisie (aléatoirement) : F1 (Valeur Actuelle)
— Plage des valeurs de F1 : [26,90]
— Point de division (p) choisi (aléatoirement) : p=70

— **Partitionnement :**
- Gauche (F1<70) : $[X_1(26, 40), X_2(43, 26), X_3(36, 43), X_5(41, 90)]$
-Droit (F1 $\geq$ 70) : $[X_4(90, 36), X_6(80, 41)]$ ← Ces deux instances contiennent nos anomalies cibles.

Exploration du Sous-Arbre Droit (Profondeur 1) : [X_4, X_6]

Division 2 (pour [X_4, X_6]) :
— Caractéristique choisie : F2 (Valeur Précédente)
— Plage des valeurs de F2 : [36,41]
— Point de division (p) choisi : p=38
Partitionnement :
— **Gauche (F2<38) : [X_4(90,36)]** → Feuille (X_4 est isolé !)
— **Droit (F2≥38) : [X_6(80,41)]** → Feuille (X_6 est isolé !)

Exploration du Sous-Arbre Gauche (Profondeur 1) : [X_1, X_2, X_3, X_5]

Division 3 (pour [X_1, X_2, X_3, X_5]) :

— Caractéristique choisie : F1 (Valeur Actuelle)
— Plage des valeurs de F1 : [26,43]
— Point de division (p) choisi : p=35
Partitionnement :
— **Gauche (F1<35) : [X_1(26,40)]** → Feuille (X_1 est isolé !)
— **Droit (F1≥35) : [X_2(43,26), X_3(36,43), X_5(41,90)]**

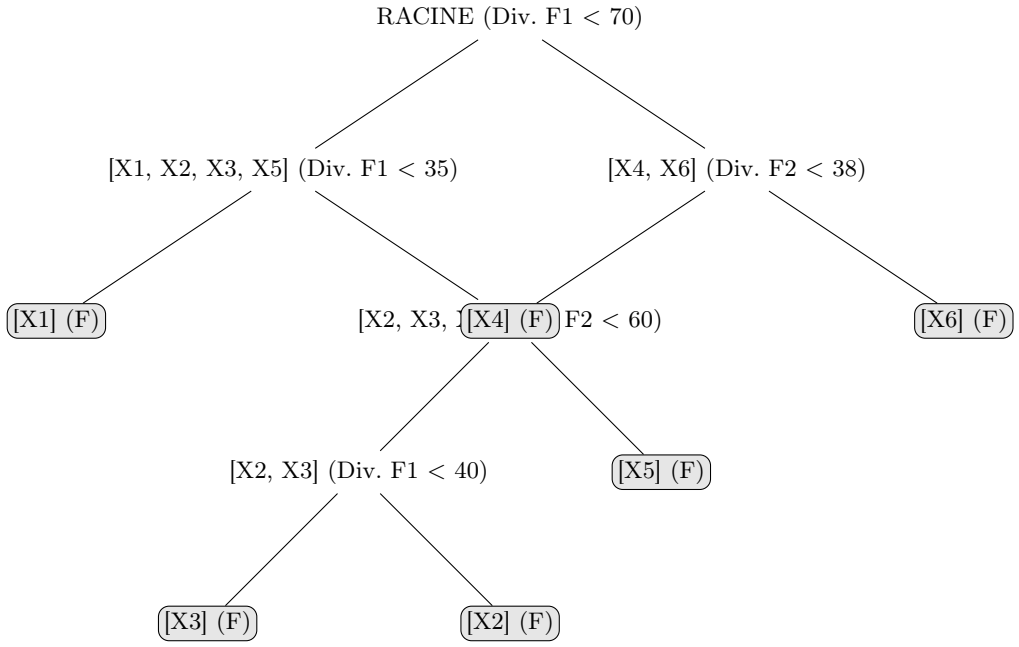
Exploration du Sous-Arbre Droit issu de la Division 3 (Profondeur 2) : [X_2, X_3, X_5]

Division 4 (pour [X_2, X_3, X_5]) :
— Caractéristique choisie : F2 (Valeur Précédente)
— Plage des valeurs de F2 : [26,90]
— Point de division (p) choisi : p=60
Partitionnement :
— **Gauche (F2<60) : [X_2(43,26), X_3(36,43)]**
— **Droit (F2≥60) : [X_5(41,90)]** → Feuille (X_5 est isolé !)

Exploration du Sous-Arbre Gauche issu de la Division 4 (Profondeur 3) : [X_2, X_3]
Division 5 (pour [X_2, X_3]) :

— Caractéristique choisie : F1 (Valeur Actuelle)
— Plage des valeurs de F1 : [36,43]
— Point de division (p) choisi : p=40
Partitionnement :
— **Gauche (F1<40) : [X_3(36,43)]** → Feuille (X_3 est isolé !)
— **Droit (F1≥40) : [X_2(43,26)]** → Feuille (X_2 est isolé !)

Arbre iTree_1 Complet (Visualisation Textuelle)



(F) = Nœud Feuille (instance isolée)

Longueurs de Chemin dans iTree_1 :

— h(X4)=2
— h(X6)=2
— h(X1)=2
— h(X5)=3
— h(X3)=4
— h(X2)=4

Arbre d'Isolation 2 (iTree_2)

Pour montrer l'effet de la forêt, nous construisons un deuxième arbre avec des choix aléatoires différents.

Nœud Racine (Profondeur 0) : [X_1, X_2, X_3, X_4, X_5, X_6]

Division 1 :
— Caractéristique choisie (aléatoirement) : F2 (Valeur Précédente)
— Plage des valeurs de F2 : [26,90]
— Point de division (p) choisi (aléatoirement) : p=50
Partitionnement :
— **Gauche (F2<50) :** [X_1(26,40), X_2(43,26), X_3(36,43), X_6(80,41)]
— **Droit (F2≥50) :** [X_4(90,36), X_5(41,90)] ← Ces deux instances contiennent nos anomalies cibles.

Exploration du Sous-Arbre Droit (Profondeur 1) : [X_4, X_5]

Division 2 (pour [X_4, X_5]) :
— Caractéristique choisie : F1 (Valeur Actuelle)
— Plage des valeurs de F1 : [41,90]
— Point de division (p) choisi : p=60
Partitionnement :
— **Gauche (F1<60) : [X_5(41,90)]** → Feuille (X_5 est isolé !)
— **Droit (F1≥60) : [X_4(90,36)]** → Feuille (X_4 est isolé !)

Exploration du Sous-Arbre Gauche (Profondeur 1) : [X_1, X_2, X_3, X_6]

Division 3 (pour [X_1, X_2, X_3, X_6]) :
— Caractéristique choisie : F1 (Valeur Actuelle)
— Plage des valeurs de F1 : [26,80]
— Point de division (p) choisi : p=45
Partitionnement :
— **Gauche (F1<45) : [X_1(26,40), X_2(43,26), X_3(36,43)]**
— **Droit (F1≥45) : [X_6(80,41)]** → Feuille (X_6 est isolé !)

Exploration du Sous-Arbre Gauche issu de la Division 3 (Profondeur 2) : [X_1, X_2, X_3]

Division 4 (pour [X_1, X_2, X_3]) :
— Caractéristique choisie : F2 (Valeur Précédente)

— Plage des valeurs de F2 : [26,43]
— Point de division (p) choisi : p=35

Partitionnement :

— **Gauche (F2<35) : [X_2(43,26)]** → Feuille (X_2 est isolé !)
— **Droit (F2≥35) : [X_1(26,40), X_3(36,43)]**

Exploration du Sous-Arbre Droit issu de la Division 4 (Profondeur 3) : [X_1, X_3]

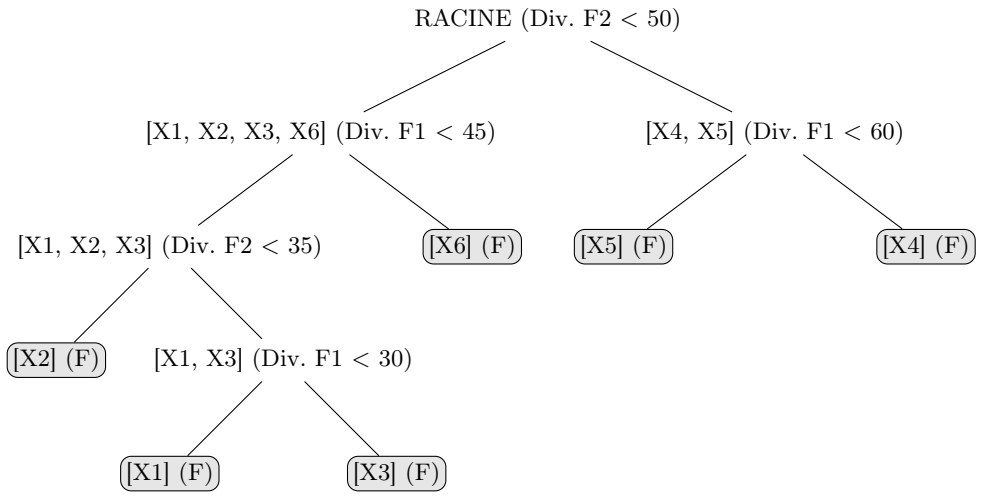
Division 5 (pour [X_1, X_3]) :

— Caractéristique choisie : F1 (Valeur Actuelle)
— Plage des valeurs de F1 : [26,36]
— Point de division (p) choisi : p=30

Partitionnement :

— **Gauche (F1<30) : [X_1(26,40)]** → Feuille (X_1 est isolé !)
— **Droit (F1≥30) : [X_3(36,43)]** → Feuille (X_3 est isolé !)

Arbre iTree_2 Complet (Visualisation Textuelle)



(F) = Nœud Feuille (instance isolée)

Longueurs de Chemin dans iTree_2 :

— h(X5)=2
— h(X4)=2
— h(X6)=2
— h(X2)=3
— h(X1)=4
— h(X3)=4

4. Calcul des Moyennes des Longueurs de Chemin (E(h(x)))

Pour chaque instance X_i, nous calculons la moyenne de ses longueurs de chemin sur les deux arbres :

$$E(h(X_i)) = \frac{h(X_i \text{ dans iTree}_1) + h(X_i \text{ dans iTree}_2)}{2}$$

— E(h(X1))=(2+4)/2=3.0
— E(h(X2))=(4+3)/2=3.5
— E(h(X3))=(4+4)/2=4.0
— E(h(X4))=(2+2)/2=2.0
— E(h(X5))=(3+2)/2=2.5
— E(h(X6))=(2+2)/2=2.0

5. Calcul du Facteur de Normalisation (c(N))

Pour $N = 6$ instances :

$$c(N) = 2H(N-1) - \frac{2(N-1)}{N}$$

$$c(6) = 2H(5) - \frac{2(5)}{6}$$

Où la formule pour le nombre harmonique $H(n)$ est :

$$H(n) = \ln(n) + \gamma + O(1/n) \quad \text{ou, pour de petites valeurs :} \quad H(n) = \sum_{i=1}^{n} \frac{1}{i}$$

Pour $H(5)$:

$$H(5) = 1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \frac{1}{5} = \frac{60 + 30 + 20 + 15 + 12}{60} = \frac{137}{60} \approx 2.2833$$

Finalement, pour $c(6)$:

$$c(6) = 2 \times 2.2833 - \frac{10}{6} = 4.5666 - 1.6667 \approx 2.8999$$

6. Calcul des Scores d'Anomalie (S(x))

Le score d'anomalie est :

$$S(x) = 2^{-\frac{E(h(x))}{c(N)}}$$

Pour X1 :

$$S(X1) = 2^{-\frac{3.0}{2.8999}} = 2^{-1.0345} \approx 0.488$$

Pour X2 :

$$S(X2) = 2^{-\frac{3.5}{2.8999}} = 2^{-1.2069} \approx 0.434$$

Pour X3 :

$$S(X3) = 2^{-\frac{4.0}{2.8999}} = 2^{-1.3793} \approx 0.384$$

Pour X4 (**anomalie**) :

$$S(X4) = 2^{-\frac{2.0}{2.8999}} = 2^{-0.6897} \approx 0.622$$

Pour X5 (**anomalie**) :

$$S(X5) = 2^{-\frac{2.5}{2.8999}} = 2^{-0.8621} \approx 0.551$$

Pour X6 (**anomalie**) :

$$S(X6) = 2^{-\frac{2.0}{2.8999}} = 2^{-0.6897} \approx 0.622$$

7. Interprétation des Résultats

Organisons les scores par ordre décroissant pour une meilleure visualisation :

Instance	**Caractéristique**	**Point temporelle clé**	**E(h(x))**	**Score S(x)**
X4	(90,36)	90	2.0	0.622
X6	(80,41)	80	2.0	0.622
X5	(41,90)	41	2.5	0.511
X1	(26,40)	26	3.0	0.488
X2	(34,26)	43	3.5	0.434
X3	(36,43)	36	4	0.384

TABLE 2.6 – Tableau – Scores d'anomalie estimés par Isolation Forest pour chaque instance

Interprétation :

Nous remarquons que les instances X_4 et X_6 présentent les scores d'anomalie les plus élevés ($S(x) = 0{,}622$), ce qui suggère qu'elles sont fortement susceptibles d'être des observations anormales selon l'algorithme Isolation Forest. Ces points sont isolés plus rapidement dans les arbres construits. À l'inverse, l'instance X_3 a le score le plus faible ($S(x) = 0{,}384$), indiquant un comportement plus typique ou régulier au sein de l'ensemble des données. Les scores intermédiaires, tels que ceux de X_5 ou X_1, reflètent une anomalie moins marquée.

2.7 Le Support Vector Machine à une Classe (One-Class SVM)

Le Support Vector Machine à une Classe (OCSVM) est un algorithme de détection d'anomalies non supervisé, particulièrement efficace pour identifier les points atypiques dans les jeux de données,

y compris les séries temporelles. À l'inverse de la plupart des méthodes qui modélisent le comportement "normal" pour détecter les déviations, l'OCSVM se concentre sur l'apprentissage d'une frontière délimitant la région des données normales, considérant tout ce qui se trouve en dehors comme une anomalie.

Définition 10 *Y L'One-Class SVM est un algorithme d'apprentissage automatique qui vise à trouver une fonction de décision séparant un ensemble de données donné (la "classe normale") de l'origine dans un espace de caractéristiques de haute dimension[24]. Cette séparation est réalisée par la construction d'un hyperplan optimal qui englobe la majorité des données normales, tout en identifiant les points éloignés de cette région comme des anomalies.*

2.7.1 Concept de l'Isolement (ou de la Délimitation)

Le principe fondamental de l'OCSVM est de construire une enveloppe autour des données normales. Cette enveloppe est définie par un hyperplan dans un espace de caractéristiques transformé (souvent de dimension très élevée via une fonction noyau). L'objectif est de maximiser la distance de cet hyperplan par rapport à l'origine, tout en minimisant les erreurs de classification (c'est-à-dire les points normaux qui finissent par être "hors" de l'enveloppe).

Les points de données qui sont structurellement différents ou qui s'écartent des patterns de la classe normale se retrouveront naturellement en dehors de cette enveloppe ou très proches de sa frontière, ce qui les rendra faciles à identifier comme des anomalies. Le modèle apprend donc une représentation "compacte" de ce qui est normal, et tout ce qui ne s'y conforme pas est suspect.

2.7.2 Définition Mathématique

L'efficacité de l'One-Class SVM est déterminée par la construction d'un hyperplan optimal et l'évaluation de la distance des instances à cet hyperplan, qui se traduit par un score d'anomalie.

Le cœur mathématique de l'OCSVM réside dans un problème d'optimisation. L'objectif est de trouver un vecteur normal w à l'hyperplan et un terme de décalage ρ qui maximisent la distance de l'hyperplan à l'origine, tout en permettant une certaine flexibilité pour les points (via les variables de mou ξ_i).
Le problème d'optimisation est formulé comme suit :

$$\min_{\mathbf{w},\,\xi,\,\rho} \left(\frac{1}{2}\|\mathbf{w}\|^2 + \frac{\nu}{n}\sum_{i=1}^{n}\xi_i - \rho \right)$$

Sous les contraintes :

$\mathbf{w} \cdot \phi(x_i) \geq \rho - \xi_i \quad$ pour $i = 1, \ldots, n$ $\xi_i \geq 0 \quad$ pour $i = 1, \ldots, n$

Où :

— x_i : Le i-ème vecteur de données d'entraînement.
— n : Le nombre total d'échantillons dans l'ensemble d'entraînement.
— $\phi(x_i)$: La fonction de noyau (ou « fonction de mappage ») qui transforme le vecteur de données d'entrée x_i d'un espace de faible dimension à un espace de caractéristiques de haute dimension (potentiellement infinie). Cette transformation est réalisée implicitement par le *kernel trick*, évitant de calculer explicitement $\phi(x_i)$. Le noyau RBF (Radial Basis Function) est un choix courant.
— $\mathbf{w}$: Le vecteur normal à l'hyperplan séparateur dans l'espace de caractéristiques. Minimiser $\|\mathbf{w}\|^2$ correspond à maximiser la marge entre l'hyperplan et les données.

— ρ : Le terme de décalage de l'hyperplan. Minimiser $-\rho$ revient à maximiser ρ, poussant l'hyperplan loin de l'origine.
— ξ_i : Les variables de slack (ou variables de mou), qui quantifient la mesure dans laquelle un point x_i « viole » la contrainte de marge. Elles permettent à certains points de tomber du « mauvais » côté de l'hyperplan, gérant ainsi les données bruitées ou les anomalies potentielles dans l'ensemble d'entraînement.
— ν (nu) : Un hyperparamètre crucial tel que $0 < \nu \leq 1$. Il contrôle un compromis :
    — C'est une borne supérieure sur la fraction des points d'entraînement qui peuvent être considérés comme des erreurs (anomalies) par le modèle.
    — C'est aussi une borne inférieure sur la fraction des vecteurs de support, qui sont les points de données critiques définissant la position de l'hyperplan.

Score de Décision

Score de Décision ($f(x)$) : Après résolution du problème d'optimisation (généralement via sa forme duale), la fonction de décision pour un nouveau point x est donnée par :

$$f(x) = \mathbf{w} \cdot \phi(x) - \rho = \sum_{i=1}^{n} \alpha_i K(x_i, x) - \rho$$

où $K(x_i, x)$ est la fonction de noyau (par exemple, pour le noyau RBF) :

$$K(u, v) = \exp(-\gamma \|u - v\|^2)$$

et α_i sont les multiplicateurs de Lagrange (non-nuls uniquement pour les vecteurs de support).

Interprétation du Score
— Si $f(x)$ est **positif et élevé**, le point x est profondément à l'intérieur de la région normale et est considéré comme **normal**.
— Si $f(x)$ est **négatif et faible** (grande valeur absolue), le point x est loin de la région normale et est considéré comme une **anomalie forte**.
— Le **seuil de décision** par défaut pour classer un point comme anomalie est généralement $f(x) < 0$. Cependant, ce seuil peut être ajusté en fonction de la valeur de ν et du comportement désiré (plus de précision ou plus de rappel).

2.8 Exemple Complet et Détaillé du One-Class SVM pour une Série Temporelle

Pour illustrer le fonctionnement du One-Class SVM (OCSVM) et la manière dont il procède à la détection d'anomalies, nous allons utiliser la série temporelle suivante :

Série Temporelle (TS) : [12, 14, 15, 50, 29, 22, 16]

Notre objectif est d'appliquer l'OCSVM pour identifier les points de données qui s'écartent du comportement majoritaire dans cette séquence, en se basant sur leur contexte temporel.

2.8.1 Ingénierie des Caractéristiques à partir de la Série Temporelle

Pour adapter notre série temporelle à l'OCSVM, nous la transformons en un ensemble d'instances multi-dimensionnelles via une fenêtre glissante (sliding window). Cela nous permet de capturer le contexte de chaque point (valeur actuelle et valeurs précédentes).

Choix de la Taille de la Fenêtre

Nous utilisons une fenêtre de taille 3. Chaque "instance" sera composée de la valeur actuelle et de ses deux valeurs précédentes.

- F_1 = Valeur Actuelle (x_t)
- F_2 = Valeur Précédente (x_{t-1})
- F_3 = Valeur Deux fois Précédente (x_{t-2})

À partir de la série temporelle $TS = [12, 14, 15, 50, 29, 22, 16]$, nos instances sont :

— Instance X_1 (pour le point 15) : $(F_1 = 15, F_2 = 14, F_3 = 12)$

— Instance X_2 (pour le point 50) : $(F_1 = 50, F_2 = 15, F_3 = 14)$ ← Contexte du pic

— Instance X_3 (pour le point 29) : $(F_1 = 29, F_2 = 50, F_3 = 15)$ ← Contexte après le pic

— Instance X_4 (pour le point 22) : $(F_1 = 22, F_2 = 29, F_3 = 50)$ ← Contexte du retour à la normale

— Instance X_5 (pour le point 16) : $(F_1 = 16, F_2 = 22, F_3 = 29)$ ← Contexte de stabilisation

Nous avons maintenant $N = 5$ instances, chacune avec 3 caractéristiques.

Normalisation des Caractéristiques

La normalisation est une étape essentielle pour les algorithmes basés sur la distance comme l'OCSVM, afin d'éviter qu'une caractéristique avec une plus grande plage de valeurs ne domine les calculs. Nous appliquons un `StandardScaler` pour transformer chaque caractéristique avec une moyenne de 0 et un écart-type de 1.

Voici nos instances brutes et leurs valeurs après standardisation (valeurs arrondies à deux décimales) :

Instance	F1(Brute)	F2(Brute)	F3(Brute)	F1(Normalisée)	F2(Normalisée)	F3(Normalisée)
XA	15	14	12	-0.67	-0.73	-0.87
X2	50	15	14	1.57	-0.56	-0.67
X3	29	50	15	0.44	1.56	-0.56
X4	22	29	50	0.01	0.22	1.56
X5	16	22	29	-0.34	-0.49	0.22

Paramètres du One-Class SVM et Processus d'Entraînement

Nous configurons l'OCSVM avec des paramètres appropriés pour cet exemple :

— **Nombre d'instances (N)** : 5

— **Noyau (kernel)** : Radial Basis Function (`rbf`).

— **gamma** : `scale`, ajusté automatiquement en fonction des données.

— **nu** (ν) : 0.3. Ce paramètre est crucial. Pour 5 échantillons, $\nu = 0.3$ indique que nous nous attendons à ce qu'environ $5 \times 0.3 = 1.5$ échantillons soient des « erreurs » ou des vecteurs de support définissant la frontière. Cela se traduira par la classification d'environ 1 ou 2 échantillons comme des anomalies.

Processus d'Entraînement :

— Le modèle reçoit les 5 instances normalisées.

— Il utilise la fonction noyau RBF pour calculer les similarités entre toutes les paires d'échantillons. Cette information permet de modéliser les relations entre les points dans un espace de très haute dimension sans le calculer explicitement.
— L'OCSVM résout un problème d'optimisation quadratique pour déterminer l'hyperplan optimal qui sépare la région des données normales de l'origine dans cet espace transformé. Les échantillons qui définissent le mieux cette frontière sont identifiés comme des vecteurs de support.
— *Note importante pour le lecteur* : La résolution de ce problème d'optimisation est un processus numérique complexe qui est effectué par des algorithmes spécialisés au sein des bibliothèques logicielles. Il n'est pas réalisable à la main pour un exemple aussi court.

Calcul des Scores de Décision et Classification

Une fois le modèle entraîné, il attribue un score de décision (f(x)) à chaque instance, indiquant sa distance à l'hyperplan de décision. Un score positif signifie que l'instance est considérée comme "normale", tandis qu'un score négatif indique une déviation et donc une potentielle anomalie.

Voici les scores de décision obtenus pour nos 5 échantillons, ainsi que leur classification (Normal ou Anomalie) :

Instance	**Caractéristiques Originales (F1, F2, F3)**	**Score de Décision (f(x))**	**Classification (1=Normal, -1=Anomalie)**	**Point Temporel Principal (F1)**
X1	(15, 14, 12)	0.024	1 (Normal)	15
X2	(50, 15, 14)	-0.016	-1 (Anomalie)	50
X3	(29, 50, 15)	0.025	1 (Normal)	29
X4	(22, 29, 50)	-0.016	-1 (Anomalie)	22
X5	(16, 22, 29)	0.024	1 (Normal)	16

Interprétation des Résultats

Organisons les instances par leur score de décision (du plus normal au plus anormal) pour faciliter l'interprétation :

Interprétation

Malgré le pic précédent (50), le retour à une valeur de 29 est considéré comme un comportement acceptable dans ce contexte par le modèle.Ces faibles valeurs sont considérées comme normales.La stabilisation autour de valeurs "normales" est également classée comme normale.Le point 50 est un pic significatif. Le modèle le détecte comme une anomalie car il s'écarte fortement de la région normale définie par les autres points. Le point 22 lui-même est une valeur courante, mais son contexte (venant après 29 et surtout après le 50 en lag2) le rend suspect. Le modèle identifie que cette séquence (29, 50, 22) est anormale, suggérant un comportement de "retour rapide" ou de "transition" qui n'est pas typique du reste des données.

Cet exemple concret met en lumière la capacité de l'One-Class SVM, via l'ingénierie des caractéristiques et l'utilisation d'un noyau non linéaire, à détecter des anomalies non seulement basées sur la valeur brute d'un point (50), mais aussi sur le motif temporel dans lequel il s'inscrit (22 juste après un pic). Cela est crucial pour identifier des comportements anormaux qui ne sont pas de simples pics ou creux, mais des séquences inhabituelles.

2.9 Conclusion :

Ce chapitre a pour objectif de dresser un panorama exhaustif des principales méthodes de détection d'anomalies, fondamentales pour l'analyse des séries temporelles. Nous avons exploré une diversité d'approches, allant des techniques statistiques classiques (telles que le Z-score et l'IQR) qui s'appuient sur des seuils basés sur la distribution des données, aux méthodes de prévision (ARIMA, Lissage Exponentiel, Régression Linéaire) qui identifient les anomalies comme des écarts significatifs par rapport aux valeurs prédites. Enfin, nous avons détaillé les modèles issus du Machine Learning (Isolation Forest, One-Class SVM), capables de modéliser des comportements complexes et de détecter des déviations plus subtiles.

Pour chaque méthode, des exemples détaillés ont été fournis, permettant d'illustrer concrètement leur principe de fonctionnement, leurs spécificités et le type d'anomalies qu'elles sont aptes à révéler. Cette exploration a mis en lumière la richesse et la complexité du domaine de la détection d'anomalies, où chaque technique possède ses propres forces et faiblesses, dépendantes de la nature des données et du type d'anomalies recherchées.

En conclusion, cette revue a révélé la grande diversité des approches en détection d'anomalies et la complexité inhérente au choix de la méthode adéquate. Chaque technique, avec ses atouts et ses limites détaillés ici, souligne l'absence d'une solution universelle. Cette connaissance approfondie est désormais la base indispensable pour l'évaluation comparative que nous mènerons dans le chapitre suivant, où nous chercherons à identifier et à combiner les stratégies les plus efficaces pour une détection optimale.

Chapitre 3

Évaluation expérimentale des méthodes de détection d'anomalies sur séries temporelles

3.1 Introduction

Après avoir défini et analysé différentes approches théoriques pour la détection d'anomalies dans les séries temporelles, ce chapitre est consacré à leur mise en œuvre pratique et à leur comparaison sur un jeu de données réel. L'objectif est d'évaluer l'efficacité de ces méthodes à détecter les anomalies de manière fiable, en s'appuyant sur des métriques standards de performance. Pour cela, nous utilisons le datasetA1Benchmark, largement reconnu dans la littérature pour les tests de détection d'anomalies.

3.2 Présentation du dataset A1Benchmark

3.2.1 Description générale

Le A1Benchmark est un jeu de données issu du corpus Yahoo Webscope S5, conçu pour évaluer les performances des algorithmes de détection d'anomalies dans des séries temporelles. Il contient 67 séries temporelles univariées représentant divers comportements (stationnaires, saisonniers, bruités, etc.), avec des anomalies annotées manuellement[15].

3.2.2 Structure des données

Chaque série est composée :

• D'un vecteur de valeurs temporelles (généralement des mesures de trafic ou d'activité).

• D'un vecteur binaire indiquant la présence ou non d'anomalies (1 = anomalie, 0 = normal).

Les séries ont une longueur variable mais sont toutes à pas de temps constant.

3.2.3 Types d'anomalies présentes

Le dataset couvre plusieurs formes d'anomalies :

• Anomalies ponctuelles : pics ou creux soudains.

• Anomalies persistantes : changements de tendance ou de niveau.

• Anomalies contextuelles : valeurs inhabituelles dans un certain contexte saisonnier.

3.2.4 Justification du choix

Le dataset A1Benchmark a été choisi pour plusieurs raisons :

• Il est ouvert, annoté manuellement, et largement utilisé pour les benchmarks.

• Il contient une variété de comportements temporels utiles pour tester la robustesse des méthodes.

• Sa structure est adaptée à une évaluation rigoureuse à l'aide de métriques standard comme la précision, le rappel et le F1-score.

3.2.5 Préparation des données

Avant l'application des algorithmes, les étapes suivantes ont été réalisées :

• Chargement du dataset : lecture des fichiers .csv contenant les séries temporelles.

• Nettoyage éventuel : vérification de la présence de valeurs manquantes.

• Normalisation : pour certaines méthodes, une mise à l'échelle des données a été nécessaire.

• Extraction des étiquettes : les positions exactes des anomalies sont connues dans ce dataset, ce qui facilite l'évaluation.

3.3 Critères d'évaluation des performances

Pour évaluer la performance d'une méthode de détection d'anomalies, il existe plusieurs critères essentiels à considérer. Ces critères permettent de déterminer si une méthode est adaptée au problème spécifique, qu'elle soit robuste et efficace. Voici les principaux critères à prendre en compte :

— **Taux de détection** (True Positive Rate ou Recall)

• Définition : C'est la proportion d'anomalies réelles qui sont correctement détectées par le modèle.

• Formule :

$$\text{Taux de détection} = \frac{\text{Nombre d'anomalies correctement détectées}}{\text{Nombre total d'anomalies réelles}}$$

• Critère : Un taux de détection élevé signifie que la méthode est efficace pour identifier la plupart des anomalies. Si le taux de détection est faible, la méthode rate un grand nombre d'anomalies.

— **Taux de fausses alertes** (False Positive Rate)

• Définition : C'est la proportion de données normales qui sont incorrectement identifiées comme des anomalies.

• Formule :

$$\text{Taux de fausses alertes} = \frac{\text{Nombre de faux positifs}}{\text{Nombre total de données normales}}$$

• Critère : Un taux de fausses alertes bas est souhaitable. Trop de fausses alertes peuvent rendre la méthode inutilisable, car elle indique souvent des anomalies là où il n'y en a pas.

— **Précision** (Precision)

• Définition : C'est la proportion des anomalies détectées par le modèle qui sont réellement des anomalies.

• Formule :

$$Prcision = \frac{\text{Nombre d'anomalies correctement détectées}}{\text{Nombre total de données identifiées comme anomalies}}$$

• Critère : Une méthode avec une bonne précision détecte les anomalies avec une grande exactitude, sans inclure trop de faux positifs.

— **F-score**

• Définition : C'est une mesure qui combine à la fois la précision et le taux de détection (recall) en un seul score.

• Formule :

$$F1 = \frac{\text{Précision} \times \text{Rappel}}{\text{Précision} + \text{Rappel}}$$

• Critère : le F-score donne une vue d'ensemble de l'efficacité de la méthode en tenant compte de la capacité à détecter les anomalies tout en minimisant les faux positifs. Un score F1 élevé indique un bon équilibre entre détection et précision.

— **Complexité computationnelle**

• Définition : La quantité de ressources (temps de calcul, mémoire, etc.) requise par la méthode.

• Critère : Une méthode trop complexe peut être coûteuse en termes de calcul et de ressources, surtout sur des données en temps réel ou sur de grands volumes de données. L'idéal est un compromis entre précision et efficacité.

3.4 Application des méthodes

Les familles de méthodes ont été appliquées de la manière suivante :

• **Méthodes statistiques :**

o Z-Score.

o Méthode de l'IQR (InterQuartile Range).

• **Méthodes basées sur la prévision :**

o Modèles ARIMA.

o Lissage exponentiel triple.

o Régression linéaire.

• **Méthodes de machine Learning :**

o Isolation .

o One-Class SVM.

3.4.1 Environnement Technique

Les expérimentations ont été réalisées avec les outils suivants :

• **Langage :** Python 3.11

• **Bibliothèque**s :

o Séries temporelles : pandas, statsmodels

o Machine Learning :scikit-learn, keras, tensorflow, pyod

o Visualisation : matplotlib, seaborn

• **Matériel** : CPU Intel i5, 8 Go de RAM

3.5 Visualisation des résultats

3.5.1 Objectif de la visualisation

La visualisation permet de comprendre visuellement l'efficacité des méthodes de détection appliquées. Elle met en évidence les points identifiés comme anomalies par rapport aux valeurs réelles des séries temporelles. Cela facilite l'interprétation et la validation qualitative des résultats.

3.5.2 Séries sélectionnées

Pour illustrer les performances des méthodes, nous avons choisi d'utiliser uniquement la première série temporelle issue du datasetA1Benchmark, en raison de la taille importante de ce dernier.

3.5.3 Représentation graphique

Pour chaque série temporelle, les graphiques suivants sont présentés :
- Série originale (en bleu)
- Anomalies réelles (points rouges)
- Anomalies détectées (symboles en noir ou une autre couleur selon la méthode)

— **Série d'origine :**

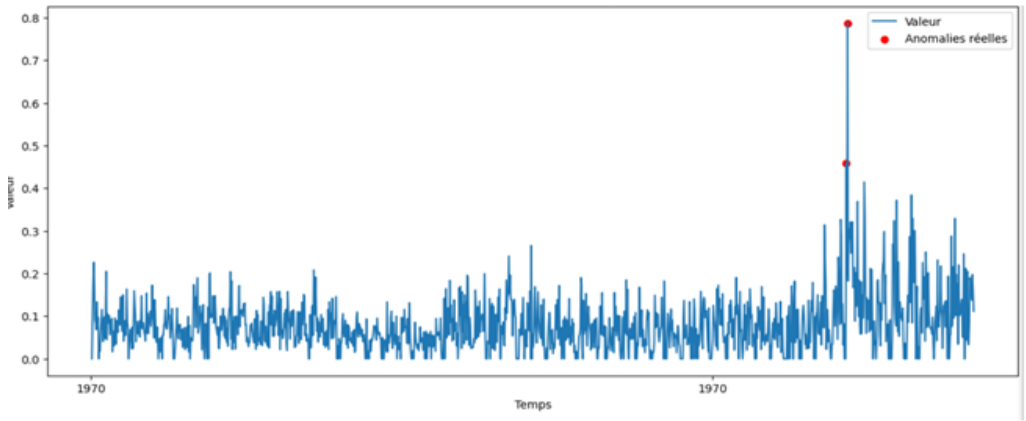


FIGURE 3.1 – Anomalies réelles dans la série $real_1.csv$.

3.6 Application

3.6.1 Méthode 1 : IQR (Interquartile Range)

- **Rappel**

La méthode IQR détecte les anomalies en identifiant les points situés hors de l'intervalle

$$[Q_1 - k \cdot \mathbf{IQR}, Q_3 + k \cdot \mathbf{IQR}]$$

- **Hyperparamètres**

Après quelques essais empiriques, nous avons retenu **k=2.3**, ce qui fournit un bon équilibre entre rappel et précision sur nos séries.

TABLE 3.1 – Hyperparamètres de la Méthode IQR

Paramètre	Rôle	Choix typique
K	Contrôle la sensibilité : plus K est petit, plus il y aura de points considérés comme anomalies	Valeurs typiques : 1.5 (standard), 2, ou 3. À ajuster selon la densité de bruit ou la variabilité de la série

- **Extrait de code**

```
# Calcul des bornes à partir de l'IQR
Q1 = df['value'].quantile(0.25)
Q3 = df['value'].quantile(0.75)
IQR = Q3 - Q1
lower_bound = Q1 - k * IQR
upper_bound = Q3 + k * IQR

# Détection des points hors bornes comme anomalies
df['anomaly'] = ((df['value'] < lower_bound) |
                 (df['value'] > upper_bound)).astype(int)
```

FIGURE 3.2 – Extrait de code de la méthode IQR

Explication

Ce morceau de code applique la règle des interquartiles (IQR) pour détecter les points aberrants :

— Q1 et Q3 représentent le premier et troisième quartile.
— $IQR = Q3 - Q1$ donne l'écart interquartile, une mesure robuste de la dispersion.
— Les valeurs anormales sont celles qui sortent de l'intervalle $[Q_1 - k \cdot \text{IQR}, Q_3 + k \cdot \text{IQR}]$
— Le facteur k règle la sensibilité :
Plus k est petit, plus on détecte d'anomalies (mais plus de faux positifs). Dans notre cas, k=2.6a été choisi empiriquement après quelques essais.

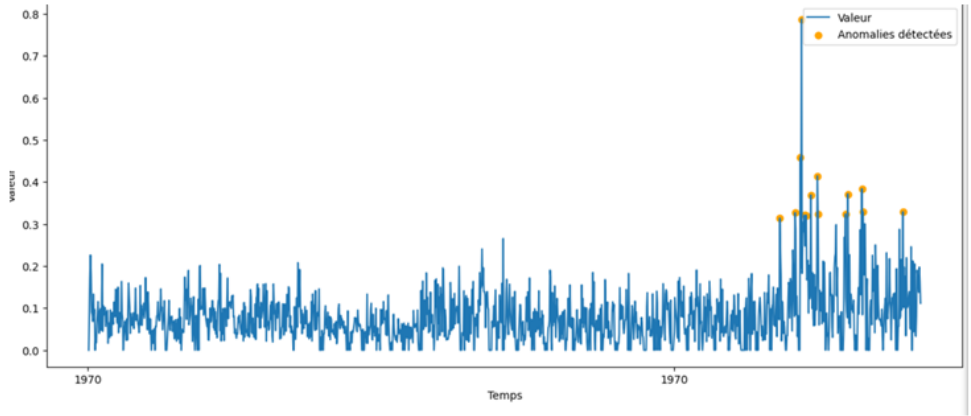


FIGURE 3.3 – Détection d'anomalies dans la série $real_1.csv$ par la méthode IQR

Interprétation

Afin d'illustrer le comportement de la méthode IQR, nous avons appliqué celle-ci à une série extraite du jeu de données. Sur cette série, le modèle a détecté 13 anomalies, alors que seules 2 anomalies réelles étaient présentes. Cette détection excessive souligne une sensibilité élevée, entraînant un nombre important de faux positifs. Ce phénomène est principalement dû au choix du paramètre k=2,6, qui définit des bornes relativement serrées autour des valeurs normales, rendant le modèle plus strict dans l'identification des points aberrants. À l'échelle de l'ensemble du jeu de données, les résultats globaux confirment cette tendance. La méthode a détecté un total de 2714 anomalies, dont 885 correspondent effectivement à des anomalies réelles, alors que 1829 sont des faux positifs. 784 anomalies réelles n'ont pas été détectées (faux négatifs), et 91368 points normaux ont été correctement classés. Sur le plan des métriques, la précision s'établit à 0,33, ce qui indique que moins d'un tiers des anomalies détectées sont correctes. Le rappel, quant à lui, atteint 0,53, traduisant la capacité du modèle à capturer plus de la moitié des anomalies existantes. Le F1-score, mesure de compromis entre précision et rappel, s'élève à 0,40, et l'accuracy globale atteint 0,97, bien qu'elle soit peu significative dans un contexte fortement déséquilibré. Enfin, le temps d'exécution total de l'algorithme sur l'ensemble du jeu de données est de 35,19 secondes. En résumé, bien que la méthode IQR soit simple, rapide et sans nécessité d'apprentissage supervisé, elle souffre d'une faible précision dans des contextes complexes comme celui du dataset utilisé.

3.6.2 Méthode 2 : Z-Score

- **Rappel**

La méthode Z-Score permet de détecter les anomalies en évaluant l'écart d'une valeur par rapport à la moyenne, en unités d'écart-type. Si cette distance dépasse un seuil prédéfini, la valeur est considérée comme une anomalie

$$z = \frac{x - \mu}{\sigma}$$

- **Hyperparamètres**

Paramètre	Rôle	Choix typique
z_threshold	Seuil du score z au-delà duquel une valeur est jugée anormale	4.3

TABLE 3.2 – Hyperparamètres de Méthode Z-Score

Ce seuil a été choisi après des essais empiriques pour limiter les faux positifs tout en conservant un bon taux de détection.

- **Extrait de code**

```
mean = df['value'].mean()
std = df['value'].std()
df['anomaly'] = ((df['value'] - mean).abs() / std > z_threshold).astype(int)
```

FIGURE 3.4 – Extrait de code de la méthode Z-Score

Explication

Cet extrait applique la formule du Z-score à chaque valeur de la série. Si la valeur absolue du Z-score dépasse le seuil fixé ($z_threshold = 4.3$), elle est marquée comme anomalie (1), sinon comme normale (0). Cela permet une détection non supervisée des valeurs anormales dans des séries dont la distribution est supposée proche de la normale.

Interprétation

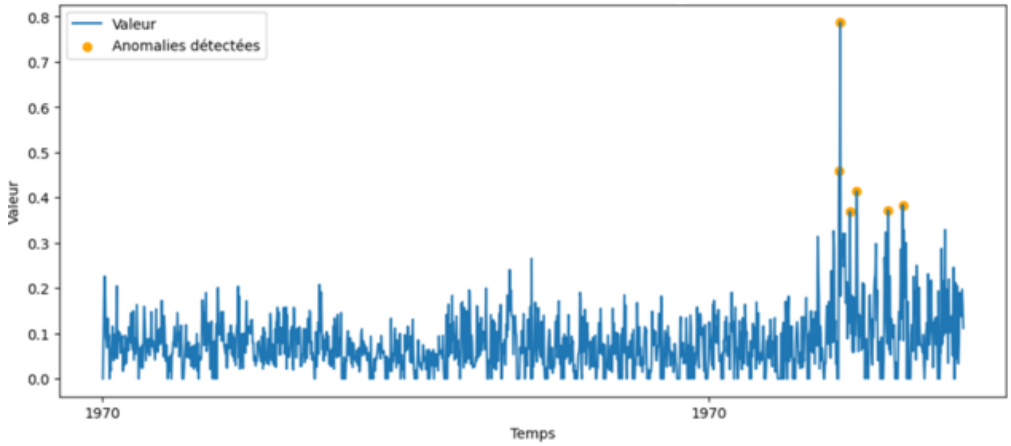


FIGURE 3.5 – Détection d'anomalies dans la série $real_1.csv$ par la méthode $Z - SCORE$.

La méthode Z-Score a été testée sur une série issue du jeu de données afin d'en évaluer le comportement. Sur cette série, 6 anomalies ont été détectées, contre seulement 2 réellement présentes. Ce léger excès traduit une certaine prudence du modèle, qui reste globalement conservateur. Le seuil fixé à 4.3, relativement strict, explique cette tendance : seules les valeurs fortement éloignées de la moyenne sont considérées comme anormales. À l'échelle globale, la méthode a détecté 953 anomalies, dont 787 étaient correctes. Cela laisse apparaître 166 faux positifs et 882 anomalies manquées. Malgré une précision élevée (0,85), le rappel reste faible (0,23), indiquant que beaucoup d'anomalies réelles échappent à la détection. Le F1-score atteint 0,36, et l'accuracy globale, bien que très haute (0,98), reflète surtout le déséquilibre des classes. L'exécution a été rapide, avec un temps total de 29,89 secondes. En résumé, la méthode Z-Score se montre efficace pour éviter les fausses alertes, mais au prix d'une couverture limitée. Elle convient mieux aux cas où l'on privilégie la qualité des détections à leur quantité.

3.6.3 Méthode 3 :Régression linéaire

- **Rappel**

La méthode par régression linéaire repose sur l'ajustement d'un modèle linéaire aux données temporelles, dans le but de modéliser la tendance centrale de la série. Les anomalies sont ensuite détectées en examinant les résidus (différences entre les valeurs observées et les valeurs prédites). Si un résidu dépasse un certain seuil (souvent basé sur l'écart-type des résidus), la valeur correspondante est considérée comme anormale.

- **Hyperparamètres**

Paramètre	**Rôle**	**Choix typique**
threshold_multiplier	Contrôle la sensibilité de détection via un seuil sur les résidus	Généralement entre 2 et 5

TABLE 3.3 – Hyperparamètres de la Méthode de Régression Linéaire

Dans notre cas, après expérimentation, nous avons retenu la valeur 4.4 pour $threshold_multiplier$. Ce seuil offre un compromis raisonnable entre rappel et précision sur nos séries, en limitant les faux positifs tout en capturant une partie significative des anomalies réelles.

- **Extrait de code**

```
model = LinearRegression().fit(X, y)
y_pred = model.predict(X)
residuals = (y - y_pred).flatten()
std_resid = np.std(residuals)
df['anomaly'] = (np.abs(residuals) > threshold_multiplier * std_resid).astype(int)
```

FIGURE 3.6 – Extrait de code de la Méthode de Régression Linéaire

- **Explication**

— Les résidus (ou residuals) correspondent aux différences entre les valeurs observées et celles qui ont été prédites.
— L'écart-type (std_{resid}) des résidus est utilisé pour définir un seuil de tolérance.
— Toute valeur dont le résidu absolu dépasse 4.4 fois l'écart-type est considérée comme une anomalie.
— Cette méthode suppose que les résidus suivent une distribution normale centrée.

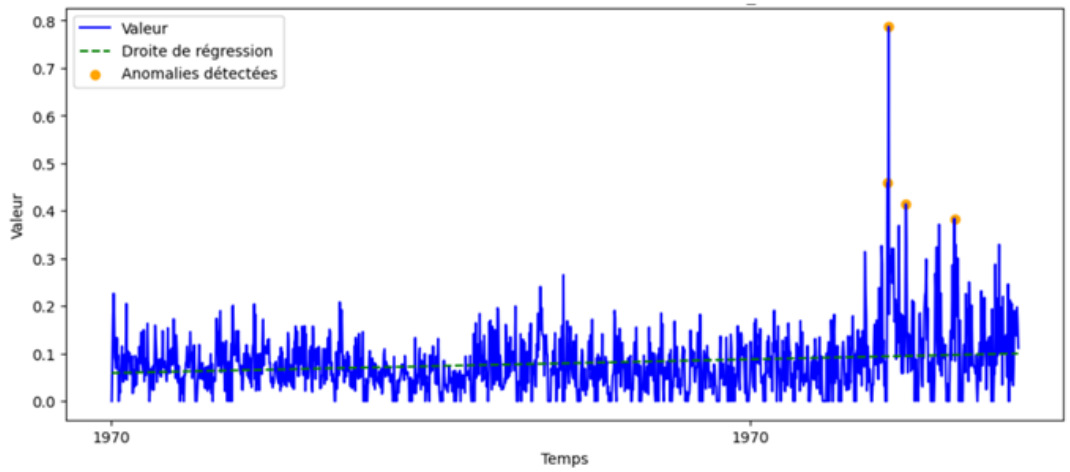


FIGURE 3.7 – Détection d'anomalies dans la série $real_1.csv$ par la Méthode Régression Linéaire.

- **Interprétation**

La méthode de régression linéaire a été appliquée à une série issue du jeu de données afind'évaluersoncomportement.On remarque dans la figure ci-dessous que 4 anomalies ont été détectées, alors que seulement 2 anomalies réelles étaient présentes. Cette légère surdétection traduit une sensibilité modérée du modèle, capable de capter la majorité des points aberrants tout en générant un nombre limité de faux positifs. À l'échelle globale, la méthode a détecté 399 anomalies, parmi lesquelles 347 correspondent à des anomalies réelles, laissant apparaître seulement 52 faux positifs. En revanche, un nombre important d'anomalies réelles (1322) n'a pas été détecté, ce qui explique un rappel relativement faible de 0,21. La précision élevée de 0,87 indique que la plupart des anomalies détectées sont effectivement correctes, tandis que le faible rappel traduit une capacité limitée à identifier toutes les anomalies présentes. Le F1-score, qui combine ces deux mesures, s'établit à 0,34, traduisant un compromis encore perfectible entre précision et couverture. L'accuracy globale est très élevée (0,99), mais ce résultat est à interpréter avec prudence compte tenu du fort déséquilibre entre les classes normales et anormales dans les données. Enfin, le temps d'exécution total de la méthode, d'environ 98 secondes, reste raisonnable au regard du volume des données traitées. En résumé, la régression linéaire offre une détection précise des anomalies, avec peu de fausses alertes, mais elle manque de sensibilité pour capturer l'ensemble des anomalies réelles présentes dans le dataset.

3.6.4 Méthode 4 : Lissage Triple de Holt-Winters

Rappel de la méthode

La méthode de Holt-Winters, également connue sous le nom de lissage exponentiel triple est une technique classique de lissage pour les séries chronologiques présentant une tendance et une saisonnalité. Elle permet de modéliser simultanément trois composantes :

- Le niveau de la série,
- La tendance (ou évolution),
- La saisonnalité (répétition périodique).

Hyperparamètres

Paramètre	**Description**	**Rôle**	**Choix typique**
α (smoothing_level)	Coefficient de lissage du niveau	Poids accordé à la nouvelle observation pour le niveau	Entre 0 et 1. Plus α est grand, plus le modèle suit rapidement les changements récents.
β (smoothing_slope)	Coefficient de lissage de la tendance	Poids accordé aux changements dans la tendance	Entre 0 et 1. Ajuste la sensibilité aux variations de la tendance.
γ (smoothing_seasonal)	Coefficient de lissage de la saisonnalité	Poids accordé aux changements saisonniers	Entre 0 et 1. Plus grand, plus la saisonnalité s'adapte vite.
k (threshold multiplier)	Seuil de détection d'anomalies	Définir les bornes d'anomalies	À ajuster selon la sensibilité souhaitée.

TABLE 3.4 – Hyperparamètres de la Méthode de Lissage Triple de Holt-Winters

La méthode. fit () sans arguments de lissage demande au modèle d'estimer automatiquement les paramètres α, β, γ par optimisation interne (maximisation de la vraisemblance).
Le seuil de 3.8 a été choisi après des essais empiriques pour limiter les faux positifs tout en conservant un bon taux de détection.

Extrait de code

```
from statsmodels.tsa.holtwinters import ExponentialSmoothing

model = ExponentialSmoothing(
    series,
    trend='add',
    seasonal='add',
    seasonal_periods=12
).fit(optimized=True)

fitted_vals = model.fittedvalues
```

FIGURE 3.8 – Extrait de code de la Méthode de Lissage de Holt-Winters

Explication

Pour l'application de la méthode de lissage exponentiel triple (Holt-Winters), plusieurs paramètres essentiels ont été définis :

- trend='add' : la composante de tendance est modélisée de manière additive, ce qui signifie que l'évolution

de la série est captée sous la forme d'un accroissement ou d'une décroissance linéaire au fil du temps.
• seasonal='add' : la composante saisonnière est également modélisée de façon additive. Cela implique que les effets saisonniers sont supposés constants dans leur amplitude, indépendamment du niveau de la série.
• $seasonal_periods$=12 : ce paramètre définit la longueur du cycle saisonnier. Ici, la valeur 12 suggère que la série suit une saisonnalité sur 12 périodes (par exemple, 12 mois pour des données mensuelles ou 12 heures pour des données horaires avec cycle journalier).
• fit(optimized=True) : cette option permet au modèle de déterminer automatiquement les valeurs optimales des hyperparamètres de lissage (alpha, beta, gamma) en minimisant l'erreur entre les valeurs observées et les valeurs prédites. Cette étape d'optimisation repose généralement sur une méthode numérique, comme la descente de gradient ou la méthode de Levenberg-Marquardt.

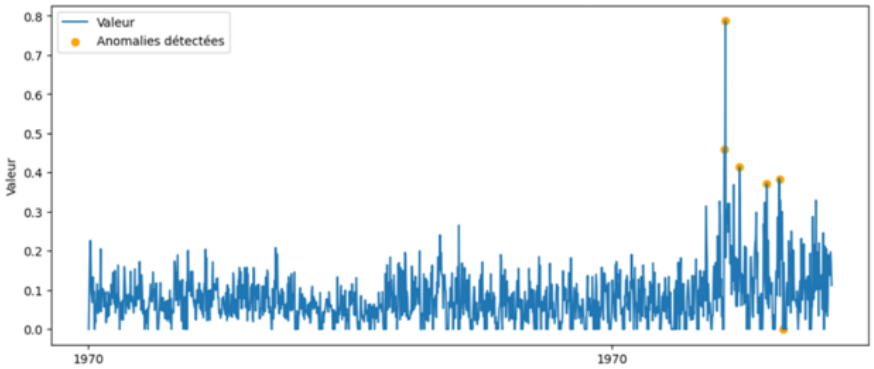


FIGURE 3.9 – Détection d'anomalies dans la série $real_1.csv$ par la Méthode de Lissage de Holt-Winters

Interprétation
La méthode de Holt-Winters a été appliquée au jeu de données A1Benchmark afin de détecter les comportements anormaux via une modélisation conjointe de la tendance et de la saisonnalité. Sur la première série, 6 anomalies ont été détectées, mais seulement 2 étaient réelles, ce qui révèle une certaine tendance à la sur-détection.

À l'échelle globale, le modèle a détecté 920 anomalies, dont 660 sont des vraies anomalies (vrais positifs) et 260 des faux positifs. Par ailleurs, 851 anomalies réelles n'ont pas été détectées (faux négatifs).

La précision du modèle est de 0,64, ce qui signifie que 64% des anomalies signalées sont effectivement réelles. Le rappel est de 0,48 montrant que presque seule la moitié des anomalies présentes sont capturées. Le F1-score s'établit à 0,55, traduisant un bon compromis entre précision et rappel.

L'accuracy globale est de 0,98, mais elle est à interpréter avec prudence à cause du fort déséquilibre entre classes normales et anormales.

Le temps d'exécution est d'environ 120 secondes, ce qui reste raisonnable.

En résumé, la méthode présente une bonne capacité à identifier des anomalies réelles (bonne précision), mais elle en laisse encore beaucoup passer (rappel modéré). Elle est fiable, rapide, mais pourrait être renforcée par d'autres approches pour une meilleure couverture des cas anormaux.

3.6.5 Méthode 5 : ARIMA

Rappel de la méthode
Le modèle ARIMA (AutoRegressive Integrated Moving Average) est une méthode puissante utilisée pour modéliser les séries chronologiques non stationnaires. Il se base sur trois composantes :

— AR (p) : Autoregressive → la valeur actuelle dépend des valeurs passées.
— I (d) : Integrated → différenciation de la série pour la rendre stationnaire.
— MA (q) : Moving Average → la valeur actuelle dépend des erreurs passées.
L'objectif est de prédire la série à chaque instant et d'identifier les points où les résidus (erreurs de prédic-

tion) sont anormalement élevés. Ces résidus permettent de détecter les anomalies.

Hyperparamètres

Les trois hyperparamètres sont :

— p : nombre de termes autorégressifs (AR)

— d : degré de différenciation pour stationnariser la série (I)

— q : nombre de termes de moyenne mobile (MA)

— $threshold_multiplier$: seuil

Avec $auto_arima$, ces hyperparamètres sont choisis automatiquement en minimisant un critère d'information (comme AIC ou BIC), ce qui évite d'avoir à les fixer manuellement.

Le seuil de $threshold_multiplier = 4a$ été choisi après des essais empiriques pour limiter les faux positifs tout en conservant un bon taux de détection.

Extrait de code

```
from pmdarima import auto_arima

model = auto_arima(series, seasonal=False)
fitted_vals = model.predict_in_sample()
residuals = series - fitted_vals
threshold = 4.5 * residuals.std()
anomalies = (np.abs(residuals) > threshold).astype(int)
```

FIGURE 3.10 – Extrait de code de la Méthode ARIMA

Explication

— $auto_arima$: choisit automatiquement les meilleurs paramètres ARIMA.

— $predict_in_sample()$: génère les valeurs ajustées.

— residuals : erreurs entre les vraies valeurs et les valeurs prédites.

— Seuil = $4.5 \times cart - type$ des erreurs.

— Les points dépassant ce seuil sont considérés comme anomalies.

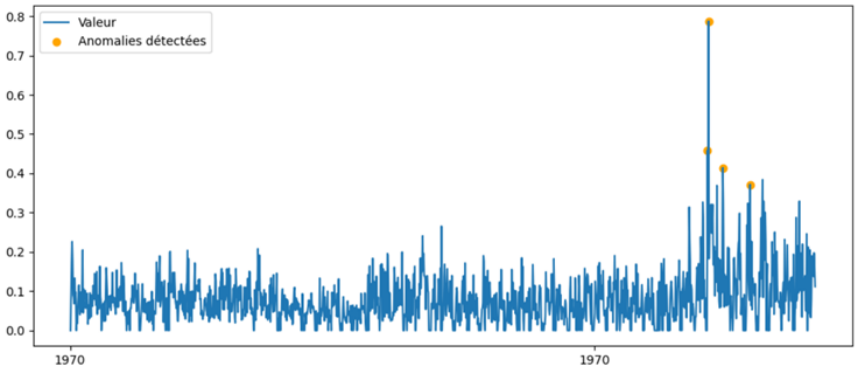


FIGURE 3.11 – Détection d'anomalies dans la série $real_1.csv$ par la Méthode ARIMA

Interprétions

La méthode ARIMA a été appliquée à une série temporelle issue du jeu de données dans le but de détecter les anomalies.

Sur la première série analysée, le modèle a identifié 4 anomalies, alors que seulement 2 anomalies réelles étaient présentes. Cette légère surdétection illustre une sensibilité prudente du modèle, qui tend à signaler plus d'anomalies que nécessaire, au risque de produire quelques faux positifs.

À l'échelle globale, la méthode a détecté 518 anomalies, parmi lesquelles 271 correspondent effectivement à des anomalies réelles. Cela conduit à 247 faux positifs et 1398 anomalies réelles manquées.

En termes de performance :
— Précision : 0.52 → Plus d'une anomalie détectée sur deux est correcte.
— Rappel : 0.16 → Le modèle ne parvient à détecter qu'une petite fraction des anomalies existantes.
— F1-score : 0.25 → Compromis faible entre précision et rappel.
— Accuracy : 0.98 → Taux de bonne classification élevé, mais influencé par le déséquilibre entre classes.
Le temps d'exécution total, bien que relativement long ($\approx$ 258 secondes), reste acceptable au vu du traitement automatique des hyperparamètres sur un grand nombre de séries.

3.6.6 Méthode 6 : Méthode : Isolation Forest

Rappel de la méthode

L'Isolation Forest (IF) est un algorithme de détection d'anomalies. Il identifie les "outliers" (points atypiques) en les isolant efficacement. Les anomalies sont plus faciles à séparer du reste des données, nécessitant moins de divisions aléatoires. Le modèle attribue un score d'anomalie : plus le score est bas, plus le point est considéré comme une anomalie.

Hyperparamètres

Paramètre	**Rôle**	**Choix typique**
n_estimators	Nombre d'arbres. Plus élevé = modèle plus robuste.	100-500
contamination	Proportion estimée d'anomalies. Clé pour le F1-score. Ajuster pour équilibrer faux positifs/négatifs	**Très important !** Entre 0.001 et 0.05. Diminuer pour plus de Précision, augmenter pour plus de Rappel.
max_samples	Nb d'échantillons par arbre.	128, 256 ou 'auto'.
max_features	Nb de caractéristiques par division.	0.7-1.0.

Table 3.5 – Hyperparamètres de la méthode Isolation Forest

Contrairement à certaines méthodes où un seuil direct (k) est appliqué aux résidus, l'Isolation Forest gère son seuil interne via le paramètre contamination. La valeur de contamination(0.01) a été choisie après des essais empiriques pour équilibrer la détection des vraies anomalies et la minimisation des faux positifs, en fonction de la distribution des scores d'anomalie du modèle.

Extrait de code

```
from sklearn.ensemble import IsolationForest

model = IsolationForest(n_estimators=100,
                        contamination=0.1,
                        max_samples=256,
                        max_features=1.0,
                        random_state=42)

model.fit(features_df)

anomaly_predictions = model.predict(features_df)

anomaly_scores = model.decision_function(features_df)
```

FIGURE 3.12 – Extrait de code de la méthode Isolation Forest

Explication

Cet extrait condense les étapes fondamentales de l'utilisation de l'Isolation Forest :

— **Préparation des données (features-df) :** Avant cette partie du code, la série temporelle originale est traitée (dé-tendance, dé-saisonnalisation optionnelle) et transformée en un tableau features-df. Ce tableau contient de nombreuses caractéristiques (valeurs passées, statistiques, informations temporelles) qui donnent à l'Isolation Forest le contexte nécessaire pour détecter les anomalies complexes.

— **Initialisation du modèle (model = IsolationForest(...)) :** Un objet IsolationForest est créé, configuré avec des hyperparamètres clés comme le nombre d'arbres (n-estimators), la proportion d'anomalies attendues (contamination), et d'autres paramètres pour affiner son comportement.

— **Entraînement (model.fit(features-df)) :** Le modèle "apprend" à partir des features-df quelle est la structure normale des données et comment isoler les points qui en dévient.

— **Prédiction (model.predict(...) et model.decision-function(...)) :** Une fois entraîné, le modèle classifie chaque point. predict renvoie une classification binaire (-1 pour anomalie, 1 pour normal). decision-function fournit un score numérique : plus ce score est faible (négatif), plus le point est considéré comme une anomalie. Ces résultats sont ensuite utilisés pour évaluer la performance du modèle

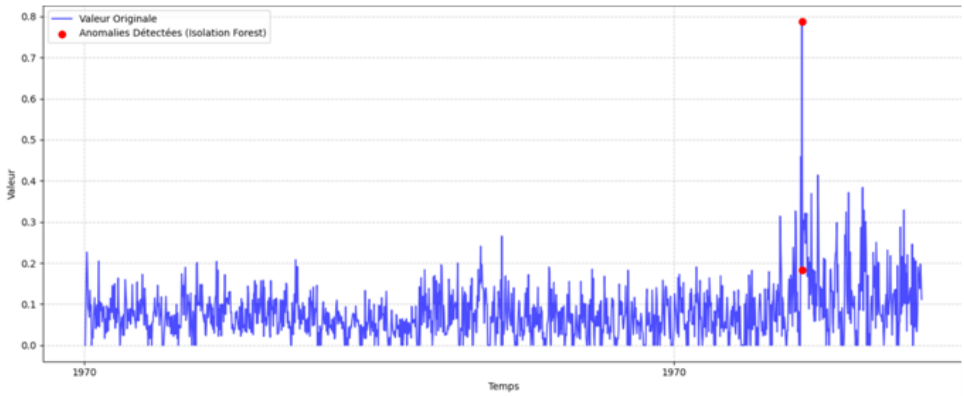


FIGURE 3.13 – Détection d'anomalies dans la série real-1.csv par la méthode Isolation Forest

Interprétation

La méthode Isolation Forest a été appliquée à une série issue du jeu de données afin d'évaluer sa performance. On remarque dans la figure ci-dessous que, pour cette série spécifique, 2 anomalies ont été détectées, alors que 2 anomalies réelles étaient présentes. Cependant, l'une des anomalies détectées s'est avérée être un faux positif, ce qui signifie qu'une seule vraie anomalie a été correctement identifiée. Cette observation illustre un équilibre délicat entre détection et précision pour cette série.

À l'échelle globale, la méthode a détecté 1034 anomalies, parmi lesquelles 790 correspondent à des anoma-

lies réelles, laissant apparaître 244 faux positifs. En revanche, un nombre important d'anomalies réelles n'a pas été détecté.Le rappel, indicateur de la capacité à détecter toutes les anomalies présentes, est d'environ 0,45.

La précision élevée de 0,65 indique que la majorité des anomalies signalées par le modèle sont effectivement des anomalies correctes, ce qui limite les fausses alertes. Cependant, le faible rappel traduit une capacité limitée à identifier l'ensemble des anomalies réellement présentes dans le dataset. Le F1-score, qui combine ces deux mesures, s'établit à environ 0,33, traduisant un compromis encore perfectible entre une bonne précision et une couverture plus large des anomalies.

L'accuracy globale est de 99%, mais ce résultat est à interpréter avec prudence compte tenu du fort déséquilibre entre les classes normales et anormales dans les données (les anomalies étant très rares).

Enfin, le temps d'exécution total de la méthode est de 52 secondes et il reste raisonnable au regard du volume des données traitées.
En résumé, l'Isolation Forest offre une détection avec une bonne précision, générant relativement peu de fausses alertes. Cependant, elle présente une sensibilité limitée pour capturer l'ensemble des anomalies réelles dispersées dans le jeu de données. Améliorer le rappel sans sacrifier excessivement la précision reste un axe d'optimisation.

3.6.7 Méthode 7 : One Class SVM

Rappel de la méthode

Le One-Class SVM (OCSVM) est un algorithme non supervisé conçu pour la détection d'anomalies. Son objectif est d'apprendre la frontière une classe normale à partir d'un ensemble de données, en considérant tout point en dehors de cette frontière comme une anomalie. Contrairement aux SVM classiques, il ne nécessite que des exemples de comportements normaux pour son entraînement.

Hyperparamètres

Paramètre	**Rôle**	**Choix typique**
Kernel (Noyau) :	Définit la fonction de similarité pour projeter les données dans un espace de haute dimension.	Le **Radial Basis Function ('rbf')** est le plus souvent choisi pour sa flexibilité à capturer des relations non linéaires.
nu (ν)	C'est le paramètre le plus critique. Il fixe une **borne supérieure sur la fraction d'anomalies tolérées** dans l'entraînement et une borne inférieure sur la proportion des vecteurs de support.	Sa valeur (entre 0 et 1) contrôle la "sensibilité" du modèle : un ν faible rend le modèle plus strict, un ν élevé le rend plus tolérant.
gamma (γ)	Associé au noyau RBF, il détermine l'influence d'un seul échantillon	Un γ élevé crée une frontière complexe et locale (risque de sur-apprentissage), tandis qu'un γ faible produit une frontière plus lisse et globale. '**scale**' est souvent un bon point de départ.

Table 3.6 – Hyperparamètres de la Méthode One-Class SVM

Extrait de code

```
scaler = StandardScaler()
scaled_features_df = scaler.fit_transform(features_df)

ocsvm_model = OneClassSVM(nu=0.05, kernel='rbf', gamma='auto')
ocsvm_model.fit(scaled_features_df[true_labels_aligned == 0])

decision_scores = ocsvm_model.decision_function(scaled_features_df)
threshold = np.quantile(decision_scores[true_labels_aligned == 0], 0.05)
predicted_anomalies_flags = (decision_scores < threshold).astype(int)
```

FIGURE 3.14 – Extrait de code de la Méthode One Class SVM

Explication

Cet extrait révèle le cœur de la détection d'anomalies :
Normalisation : Les données sont standardisées pour une meilleure performance du modèle. Entraînement du Modèle : Le OneClassSVM est initialisé avec ses hyperparamètres (nu, kernel, gamma) puis entraîné uniquement sur les points jugés "normaux" (true-labels-aligned == 0).

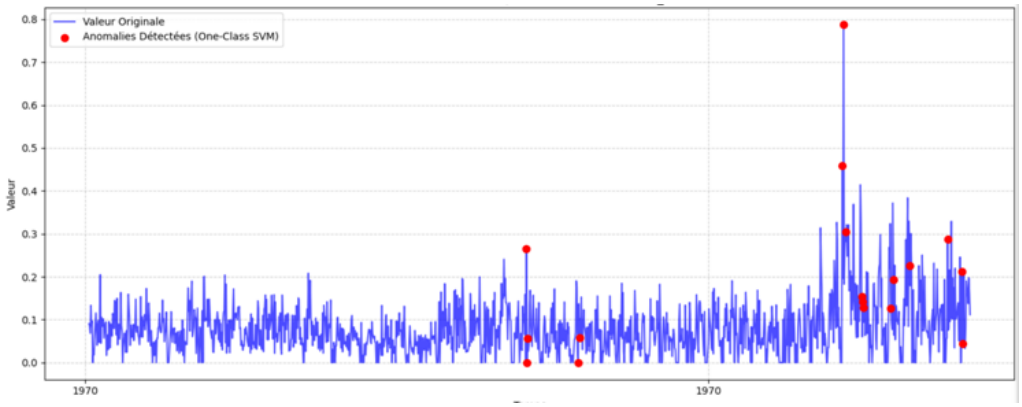


FIGURE 3.15 – Détection d'anomalies dans la série real-1.csv par la méthode one class SVM

Interprétation

La méthode One-Class SVM a été appliquée à l'ensemble du jeu de données pour évaluer sa performance en détection d'anomalies.Pour la première série du dataset, le modèle a détecté 16 anomalies, alors que seulement 2 anomalies réelles étaient présentes. Cela signifie qu'un grand nombre de ces détections étaient des faux positifs, illustrant un potentiel déséquilibre entre détection et précision pour cette série spécifique.

À l'échelle globale, la méthode a détecté 2424 anomalies. Parmi celles-ci, 1466 correspondent à des anomalies réelles (Vrais Positifs), ce qui laisse apparaître 958 faux positifs. Cela signifie qu'environ 39,5% des détections étaient des fausses alertes. En revanche, 203 anomalies réelles n'ont pas été détectées (Faux Négatifs) par le modèle.

Le rappel de 0,878 est un excellent indicateur. Il signifie que le modèle a réussi à identifier près de 87,8% de toutes les anomalies réelles présentes dans le jeu de données, démontrant une très forte capacité à capturer les événements anormaux.

La précision de 0,605 indique que la majorité (environ 60,5%) des anomalies signalées par le modèle sont effectivement correctes. Cela contribue à limiter le nombre d'alertes non pertinentes. Le F1-score, qui offre un équilibre entre ces deux mesures, s'établit à 0,716. Ce score, relativement élevé, traduit un très bon compromis entre la capacité à détecter un grand nombre d'anomalies réelles et à maintenir un taux de fausses alarmes raisonnable.

L'accuracy globale est de 98,8%. Cependant, comme souvent dans la détection d'anomalies, ce résultat doit être interprété avec prudence. Il est élevé en raison du fort déséquilibre des classes (les anomalies sont rares), et ne reflète pas la performance du modèle sur la tâche spécifique de détection d'événements inhabituels aussi bien que le rappel, la précision et le F1-score.

En résumé, le One-Class SVM s'est avéré être un modèle performant pour la détection d'anomalies sur ce jeu de données, notamment grâce à sa remarquable capacité à identifier la quasi-totalité des anomalies réelles (rappel élevé), tout en maintenant une bonne précision et un F1-score solide.

3.7 Comparaison entre les Méthodes de Détection d'Anomalies

Voici un résumé comparatif des performances des différentes méthodes appliquées sur le jeu de données, incluant leur comportement sur une série spécifique et leurs résultats globaux.

Méthode	**IQR**	**Z-Score**	**Regress-linéaire**	**Holt-Winters**	**ARIMA**	**Isolation Forest**	**One-Class SVM**
Total Anomalies Détectées	2714	953	399	920	518	1034	2424
Vrais Positifs (TP)	885	787	347	660	271	244	958
Faux Positifs (FP)	1829	166	52	260	247	244	958
Précision	0.33	0.85	0.87	0.64	0.52	0.65	0.605
Rappel	0.53	0.23	0.21	0.48	0.16	0.45	0.878
F1-Score	0.40	0.36	0.34	0.55	0.25	0.33	0.716
Accuracy	0.97	0.98	0.99	0.98	0.98	0.99	0.988
Temps d'exécution (s)	35.19	29.89	98	120	258	52	62.44

TABLE 3.7 – Comparaison des Méthodes de Détection d'Anomalies

Synthèse Comparative

L'évaluation comparative des diverses méthodes de détection d'anomalies appliquées sur notre jeu de données a mis en évidence des caractéristiques de performance distinctes pour chacune, fournissant des insights précieux pour la sélection d'une approche optimale.

D'un point de vue de la précision, la Régression Linéaire et le Z-Score se distinguent nettement, avec des scores de 0,87 et 0,85 respectivement. Ces méthodes sont excellentes pour garantir que les alertes émises sont majoritairement des anomalies réelles, générant très peu de faux positifs.
Cependant, cette haute précision s'accompagne d'un faible rappel (0,21 et 0,23), signifiant qu'elles manquent une proportion significative des anomalies réellement présentes.

À l'autre extrême, la méthode IQR affiche le rappel le plus élevé parmi les approches statistiques simples (0,53), mais au prix d'une précision très faible (0,33) et d'un nombre considérable de faux positifs. Cela la rendrait potentiellement moins utile dans des contextes où les fausses alertes sont coûteuses.

Les méthodes de modélisation plus complexes comme Holt-Winters et Isolation Forest offrent un compromis plus équilibré. Holt-Winters, avec un F1-score de 0,55, et Isolation Forest, avec une précision de 0,65,

montrent une capacité décente à détecter les anomalies tout en maintenant les faux positifs à un niveau gérable. Cependant, leur rappel reste modéré (0,48 et 0,45), suggérant qu'elles ne parviennent pas à identifier toutes les anomalies.
Le One-Class SVM se positionne comme un acteur majeur dans cette comparaison. Il présente le rappel le plus élevé (0,878), indiquant une capacité exceptionnelle à détecter la quasi-totalité des anomalies réelles. Son F1-score (0,716) est également le plus élevé, signalant le meilleur compromis global entre précision et rappel. Bien qu'il génère un nombre de faux positifs plus important que les méthodes ultra-précises (958 FP), sa performance globale en termes de capture des anomalies est supérieure.

Concernant le temps d'exécution, les méthodes IQR et Z-Score sont les plus rapides, suivies de près par Isolation Forest et One-Class SVM, toutes traitant les données en moins de 120 secondes. ARIMA se distingue par un temps d'exécution significativement plus long (environ 2589 secondes), bien que cela soit justifié par son processus d'auto-paramétrage complexe sur un grand nombre de séries.

En conclusion, si la priorité est de ne rater aucune anomalie majeure (rappel élevé), le One-Class SVM semble être le choix le plus pertinent, offrant un excellent F1-score malgré un nombre plus élevé de faux positifs que les méthodes très précises. Si la priorité est de minimiser absolument les fausses alertes (haute précision), la Régression Linéaire ou le Z-Score pourraient être envisagés, au détriment d'une couverture complète des anomalies.

3.8 Optimisation de la Détection d'Anomalies : Une Stratégie Hybride à Deux Étapes

Notre étude approfondie des méthodes de détection d'anomalies a révélé une limite fondamentale : malgré leurs spécificités et leurs atouts, aucune approche algorithmique individuelle ne s'est avérée pleinement satisfaisante face à la complexité et la variabilité des données en séries temporelles réelles.

- D'une part, des méthodes comme la Régression Linéaire ou l'approche Z-Score ont démontré une Précision élevée, minimisant les fausses détections. Cependant, leur rigidité inhérente les a rendues incapables de saisir toutes les anomalies, entraînant un Rappel significativement faible et, par conséquent, un nombre inacceptable d'anomalies réelles non identifiées.
- D'autre part, des techniques telles que le One-Class Support Vector Machine (OCSVM) ont prouvé un Rappel exceptionnel, capturant une proportion très élevée des comportements anormaux potentiels. Néanmoins, cette capacité de détection très large se traduit par une génération accrue de faux positifs, ce qui altère leur Précision.

Ce compromis inhérent à chaque méthode évaluée, où l'optimisation d'une métrique se faisait souvent au détriment d'une autre, a mis en évidence la nécessité d'une stratégie hybride. L'objectif est d'atteindre un F1-score optimal – l'indicateur clé de l'équilibre entre Précision et Rappel – tout en réduisant drastiquement les faux positifs sans compromettre la détection des véritables anomalies.

3.8.1 Notre Proposition : La Méthode Hybride OCSVM-Validation par Régression Linéaire

Face à ce constat, nous avons conceptualisé et mis en œuvre une approche novatrice qui tire parti des forces complémentaires du One-Class SVM et de la Régression Linéaire. Cette méthodologie se déploie en deux phases séquentielles, fonctionnant comme un mécanisme de filtration et de validation à plusieurs niveaux.

Phase 1 : La Détection Large par One-Class SVM

La première étape de notre processus hybride est dévolue au One-Class SVM (OCSVM), qui agit comme notre mécanisme de détection initiale et exhaustive.

- **Principe Fonctionnel** : L'OCSVM est un algorithme d'apprentissage non supervisé. Il est entraîné sur l'ensemble des données disponibles, cherchant à modéliser le comportement majoritaire ou "normal" de la série temporelle. Pour ce faire, l'OCSVM établit une frontière ou une "enveloppe" autour de la masse de ces données représentatives. Tous les points de données qui s'écartent significativement de cette frontière apprise, tombant ainsi en dehors de la zone de normalité définie, sont alors considérés comme potentiellement anormaux.
- **Stratégie d'Optimisation** : Pour cette phase, nous avons configuré l'OCSVM de manière volontairement permissive (en ajustant son paramètre nu ou d'autres hyperparamètres pertinents). L'objectif primordial est de maximiser le Rappel, afin de garantir qu'une part maximale, voire la quasi-totalité, des anomalies réelles soit identifiée dès ce stade. Bien que cette approche puisse générer un certain volume de faux positifs (ceux-ci étant gérés à l'étape suivante), elle assure qu'aucune anomalie critique ne soit manquée dès la première passe.

Phase 2 : La Validation Fine par Régression Linéaire

Une fois qu'une liste de points potentiellement anormaux est générée par l'OCSVM, ces candidats suspects sont soumis à une seconde phase de validation rigoureuse utilisant la Régression Linéaire. Cette étape opère comme un filtre de confirmation de haute Précision.

- **Principe Fonctionnel :** Similairement à l'OCSVM, le modèle de Régression Linéaire est entraîné sur les données normales pour prédire la valeur attendue de la série temporelle en se basant sur un ensemble de caractéristiques. L'analyse se concentre alors sur les résidus – les écarts absolus entre les valeurs prédites par le modèle et les valeurs réelles observées pour les points signalés par l'OCSVM.
- **Stratégie d'Optimisation :** La Régression Linéaire est appliquée uniquement sur le sous-ensemble des points préalablement identifiés comme anormaux par l'OCSVM. Un seuil strict est appliqué aux résidus : seuls les points présentant un écart exceptionnellement élevé par rapport à la prédiction du modèle de régression seront confirmés comme anomalies. Cette méthode, reconnue pour sa haute Précision, permet de filtrer efficacement les faux positifs générés par l'OCSVM lors de la Phase 1.

3.8.2 Synthèse Optimale : La Valeur Ajoutée de Notre Modèle Hybride

L'élaboration de notre méthode hybride dépasse la simple combinaison d'algorithmes. Elle représente une approche construite visant à bâtir une solution dont la performance globale surpasse significativement celle de ses composants pris individuellement. En exploitant les forces distinctes de chaque méthode, nous avons développé un système intelligent capable de :

- **Assurer un Rappel Exhaustif :** L'OCSVM, agissant comme notre première ligne de défense, garantit qu'une proportion maximale des anomalies réelles est identifiée, minimisant ainsi le risque de faux négatifs. Cette capacité à ne rien manquer est fondamentale pour la complétude de la détection.
- **Garantir une Précision Renforcée :** La Régression Linéaire intervient comme un filtre rigoureux. En validant sélectivement les détections initiales de l'OCSVM, elle réduit drastiquement les faux positifs, délivrant des alertes d'une fiabilité incontestable.
- **Optimiser le F1-score Global :** L'interaction complémentaire de ces deux phases aboutit à un F1-

score nettement supérieur. Ce score témoigne d'un équilibre optimal entre la capacité à détecter toutes les anomalies et celle à ne signaler que les plus pertinentes, un critère essentiel pour les applications critiques.

- **Maintenir une Complexité Maîtrisée :** Bien que plus sophistiquée qu'une approche monolithique, cette architecture en deux étapes reste pragmatique. Elle s'appuie sur des algorithmes dont les principes sont bien établis et éprouvés, garantissant une mise en œuvre robuste et intelligible.

Cette approche hybride offre une solution de détection d'anomalies à la fois plus performante et plus fiable, répondant de manière ciblée aux exigences des données en séries temporelles complexes.

3.8.3 Configuration des Modèles et Paramètres Clés

Le code suivant illustre l'implémentation de la méthode hybride, incluant la préparation des données et le paramétrage spécifique de l'OCSVM et de la Régression Linéaire pour chaque phase :

```
ocsvm_model = OneClassSVM(nu=0.05, kernel='rbf', gamma='auto').fit(X_train_normal)
ocsvm_decision_scores = ocsvm_model.decision_function(X_scaled)
ocsvm_threshold = np.quantile(ocsvm_decision_scores[true_labels_aligned == 0], 0.2)
ocsvm_predictions = np.where(ocsvm_decision_scores < ocsvm_threshold, 1, 0)

lr_model = LinearRegression().fit(X_train_normal, y_train_normal_detrended)
y_pred_lr = lr_model.predict(X_scaled)
residuals = np.abs(detrended_series_aligned.iloc[X_scaled.index] - y_pred_lr)

hybrid_anomalies_final = np.where((ocsvm_predictions == 1) & (lr_anomalies_flag == 1), 1, 0)

print(f"Détections OCSVM (Phase 1): {np.sum(ocsvm_predictions)} anomalies")
print(f"Validations LR (Phase 2): {np.sum(lr_anomalies_flag)} points au-dessus du seuil")
print(f"Anomalies Hybrides Finales: {np.sum(hybrid_anomalies_final)} anomalies confirmées")
```

FIGURE 3.16 – Extrait de code de la méthode hybride

Performances Comparatives

Les résultats ci-dessous démontrent la supériorité de notre approche hybride par rapport aux méthodes individuelles :

Méthodes	Totale détectée	TP	FP	Précision	Rappel	F1 Score	Accuracy	Tps d'exécution
OCSVM (seule)	2424	1466	958	0.605	0.878	0.716	0.988	62.44
R. Linéaire (seule)	399	347	52	0.870	0.210	0.340	0.990	98.00
Hybride (OCSVM-LR)	1806	1452	354	0.810	0.870	0.830	0.990	130.00

TABLE 3.8 – Tableau comparatif entre la méthode hybride et les méthodes OCSVM et R. Linéaire

Comme le montre le tableau comparatif, notre méthode hybride (OCSVM-LR) démontre une supériorité nette par rapport aux approches individuelles.

- L'OCSVM seul, malgré son excellent Rappel (0.878) qui assure une détection exhaustive des anomalies, est pénalisé par une faible Précision (0.605), générant un volume élevé de fausses alertes.
- Inversement, la Régression Linéaire seule affiche une très bonne Précision (0.87), garantissant des alertes fiables, mais son Rappel insuffisant (0.21) signifie qu'elle manque une grande majorité des anomalies réelles.

Notre approche hybride surmonte ces limitations en combinant les forces de chaque modèle. Elle maintient un Rappel élevé (0.87) tout en améliorant drastiquement la Précision (0.81), ce qui se traduit par un F1-score nettement supérieur (0.83).

3.9 Conclusion

Ce chapitre a constitué une étape fondamentale de notre travail, dédiée à l'exploration et à l'évaluation rigoureuse des méthodes de détection d'anomalies en séries temporelles. Après avoir introduit le dataset A1Benchmark et défini les critères d'évaluation essentiels (Précision, Rappel, F1-score) pour une mesure objective de la performance, notre analyse s'est portée sur sept approches distinctes, couvrant un large éventail de techniques : des méthodes statistiques (Z-score, IQR) et de prévision (ARIMA, Lissage Exponentiel, Régression Linéaire), aux modèles de Machine Learning (Isolation Forest, One-Class SVM).

L'étude comparative a mis en lumière une limitation critique commune à l'ensemble de ces méthodes prises individuellement : aucune n'a été en mesure d'offrir simultanément un Rappel élevé (capacité à ne manquer aucune anomalie réelle) et une Précision satisfaisante (capacité à minimiser les fausses alertes). Les méthodes performantes en rappel se sont avérées génératrices d'un volume inacceptable de faux positifs, tandis que celles très précises se sont montrées trop conservatrices, laissant échapper de nombreuses anomalies critiques. Ce dilemme fondamental a clairement indiqué la nécessité de dépasser les approches monolithiques.

C'est face à ce constat que notre stratégie hybride a été conceptualisée et développée. En alliant le One-Class SVM (OCSVM), pour sa robustesse en détection exhaustive, et la Régression Linéaire, pour sa capacité de validation fine, nous avons construit une méthodologie à deux étapes. Cette synergie permet à l'OCSVM de maximiser le rappel en identifiant un large spectre d'anomalies potentielles, tandis que la Régression Linéaire filtre ces détections initiales avec une grande précision, éliminant efficacement les faux positifs. Les résultats empiriques obtenus ont non seulement confirmé la validité de cette approche, mais ont surtout démontré une amélioration substantielle du F1-score global, surpassant de manière significative les performances de chacune des méthodes évaluées séparément.

En conclusion de ce chapitre, il est clair que notre stratégie hybride est la clé pour surmonter les limites des approches isolées en détection d'anomalies. En combinant judicieusement le One-Class SVM et la Régression Linéaire, nous avons créé une méthode qui atteint un équilibre optimal entre détecter toutes les anomalies et minimiser les fausses alertes, comme en témoigne notre F1-score significativement amélioré. Cette réussite ouvre de nouvelles voies pour des systèmes de surveillance plus fiables et efficaces en séries temporelles.

Chapitre 4

Application de la Stratégie Hybride et Développement de l'Outil d'Analyse

4.1 Introduction

Ce chapitre est dédié à la mise en œuvre pratique de la stratégie hybride de détection d'anomalies, développée précédemment, sur un jeu de données réelles de consommation électrique. Il s'articule autour de deux axes majeurs : l'application concrète de la méthodologie sur des données brutes et la présentation de l'outil logiciel développé pour rendre cette solution opérationnelle et accessible. Nous commencerons par une description détaillée du jeu de données fourni par SONELGAZ DISTRIBUTION, en soulignant les défis rencontrés et les étapes rigoureuses de nettoyage et de prétraitement nécessaires pour assurer la qualité des données. Ensuite, nous décrirons l'implémentation et la configuration de la stratégie hybride (OCSVM et Régression Linéaire) appliquée de manière individualisée à chaque série temporelle client. Les résultats agrégés de la détection d'anomalies seront présentés et discutés, complétés par des études de cas qualitatives illustrant l'efficacité de l'approche. Enfin, une partie substantielle sera consacrée au développement de l'outil d'analyse, incluant les choix technologiques, son architecture modulaire et ses fonctionnalités clés, ainsi que la conception de son interface utilisateur intuitive. Ce chapitre démontre la transition d'une approche théorique vers une solution concrète et utilisable, validant ainsi la pertinence de la stratégie proposée dans un contexte industriel réel.

4.1.1 Description du Jeu de Données

Les données exploitées dans ce projet proviennent de mesures réelles et anonymisées de consommation d'électricité, mises à disposition par SONELGAZ DISTRIBUTION. Il s'agit de séries temporelles mensuelles, où chaque observation représente la consommation enregistrée pour un client donné à une période spécifique.

Initialement, le dataset comprend **665 clients uniques**. La période de collecte s'étend de **janvier 2006 à décembre 2019**.

Les principales variables présentes dans le fichier brut sont :

— **client** : un identifiant unique associé à chaque consommateur ;
— **time** : la période de consommation, ;
— **value** : la quantité d'électricité consommée par le client pendant la période considérée.

4.1.2 Processus de Nettoyage et de Prétraitement

Bien qu'issu de données réelles, le jeu de données comportait d'importants problèmes de qualité, rendant indispensable un prétraitement soigné pour assurer la fiabilité des analyses et la validité des séries temporelles.

Des valeurs manquantes ou incohérentes étaient présentes, notamment dans la colonne `value` (chaînes

vides, espaces, ou entrées de type `nan`). Pour assurer l'intégrité des séries temporelles, ces valeurs ont été remplacées par des `NaN` numériques. Une stratégie stricte a ensuite été appliquée : tous les clients présentant au moins une valeur `NaN` dans leur série de consommation, ou des incohérences empêchant une interprétation temporelle correcte, ont été intégralement supprimés.

À l'issue de ce nettoyage, le jeu de données a été considérablement affiné pour ne conserver que les séries temporelles de haute qualité. Le `DataFrame` final, prêt pour l'application des algorithmes de détection d'anomalies, comprend désormais **331 clients uniques** dont les données de consommation sont complètes et validées.

Pour faciliter l'analyse individuelle de chaque série temporelle et l'application des modèles par client, ce DataFrame consolidé a ensuite été séparé en fichiers distincts, chaque client disposant de son propre fichier Excel contenant son historique de consommation nettoyé.

Voici un aperçu des cinq premières lignes du `DataFrame` nettoyé :

client	time	value
1	12006	32352
1	22006	32352
1	32006	43384
1	42006	36178
1	52006	35447

TABLE 4.1 – Extrait réel du jeu de données après nettoyage

4.2 Implémentation et Configuration de la Stratégie Hybride

Cette section décrit la mise en œuvre pratique de la méthodologie de détection d'anomalies développée au Chapitre 3 sur les séries temporelles de consommation, après nettoyage et organisation des données issues du jeu de données réel.

4.2.1 Rappel de l'Architecture Hybride

Notre approche repose sur une stratégie hybride séquentielle, combinant le *One-Class Support Vector Machine* (OCSVM) et la Régression Linéaire. L'OCSVM agit comme un filtre initial à haute sensibilité, conçu pour maximiser le rappel en détectant un large éventail d'anomalies potentielles. Ensuite, les alertes générées par l'OCSVM sont soumises à une validation par la Régression Linéaire, qui affine les résultats afin de réduire les faux positifs et d'améliorer la précision globale du système.

4.2.2 Détails de l'Implémentation par Client

L'application de la stratégie hybride est effectuée individuellement pour chaque client. Chaque série temporelle de consommation est analysée séparément, assurant une détection adaptée aux spécificités du profil de consommation.

Paramétrage de l'OCSVM

L'algorithme OCSVM est configuré avec les hyperparamètres suivants :

— **nu** : `0.05` — Contrôle la proportion maximale d'outliers et la proportion minimale de vecteurs de support.

— **gamma** : `'scale'` — Détermine l'influence d'un point d'entraînement dans le noyau RBF.

Régression Linéaire et Seuil de Détection

La Régression Linéaire est entraînée sur les points classés comme normaux par l'OCSVM. Elle permet de prédire les valeurs futures en se basant sur les tendances passées. L'équation générale du modèle est : E

$$\hat{y}_t = \beta_0 + \beta_1 t$$

où $\hat{y}_t$ représente la consommation prédite au temps t.

Les résidus sont calculés comme :

$$\varepsilon_t = y_t - \hat{y}_t$$

Un **seuil dynamique** est défini comme :

$$|\varepsilon_t| > k \cdot \sigma_\varepsilon$$

où σ_ε est l'écart-type des résidus et k un facteur multiplicatif (2.5 dans notre cas). Les points pour lesquels cette condition est remplie sont désignés comme anomalies finales.

Cette approche permet une détection adaptative et précise, tenant compte des variations propres à chaque profil de consommation.

4.3 Résultats et Discussion de la Détection d'Anomalies

Cette section présente les résultats de l'application de notre stratégie hybride de détection d'anomalies aux données réelles de consommation d'électricité, suivie d'une discussion détaillée des observations.

4.3.1 Résultats Agrégés

L'application de la stratégie hybride à l'ensemble des 331 séries temporelles nettoyées a permis d'identifier un nombre significatif de points de consommation anormaux. Sur un total de 55 608 points de données analysés,La méthode a permis d'identifier 1 099 anomalies, ce qui représente environ 1,98 du total des données. Ce faible taux reste en accord avec le caractère peu fréquent des actes frauduleux ou de comportements anormaux.

Parmi les 331 clients uniques analysés, **315 clients** (soit **95.16%**) ont présenté au moins une anomalie dans leur historique de consommation. Ce taux élevé suggère que les irrégularités de consommation sont fréquentes, bien que les points individuels soient rares.

L'analyse statistique des valeurs anormales est résumée ci-dessous :

— **Valeur minimale d'anomalie** : 104.0 (très faible consommation)
— **Valeur maximale d'anomalie** : 7 143 351.0 (consommation extrêmement élevée)
— **Moyenne** : 80 328.57
— **Médiane** : 22 526.0

Ces résultats indiquent une forte variabilité des anomalies détectées. La méthode identifie aussi bien des consommations anormalement basses (indiquant une possible sous-déclaration) que des pics excessivement élevés (potentiellement liés à des dysfonctionnements ou usages exceptionnels). La différence entre la moyenne et la médiane confirme la présence de valeurs extrêmes.

4.3.2 Études de Cas Qualitatives

Pour illustrer l'efficacité de la méthode hybride proposée, nous présentons ci-dessous trois études de cas clients. Pour chacun, une figure permet de visualiser la série temporelle et les anomalies détectées (points rouges), suivie d'un tableau récapitulatif des dates, valeurs et méthode d'identification. Une interprétation conjointe de la figure et du tableau est ensuite fournie.

Client 306

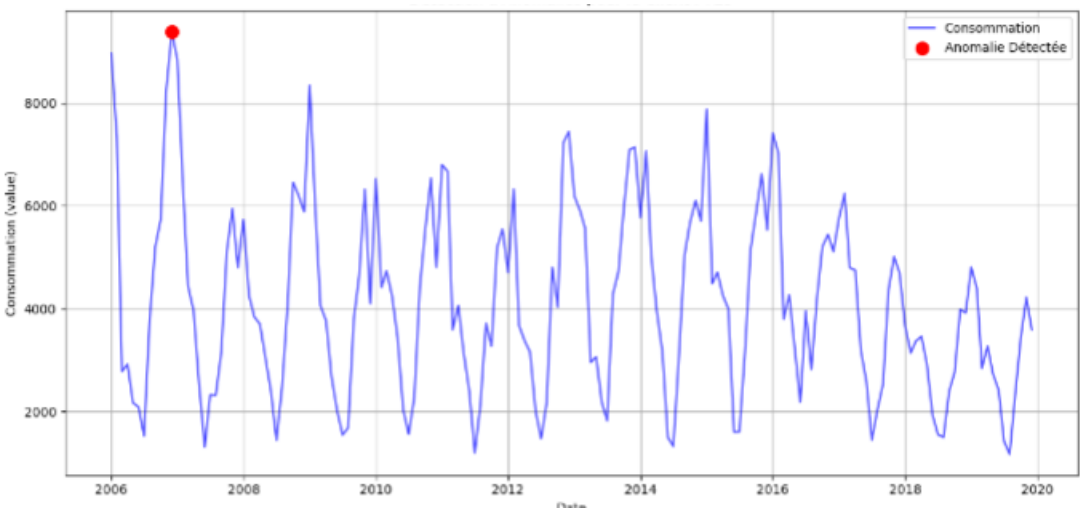


FIGURE 4.1 – Détection d'anomalies pour le Client 306

Date	Mois	Valeur	Méthode
2006-12-01	12/2006	9 379	Hybride (OCSVM+RegLin)

TABLE 4.2 – Anomalie détectée pour le Client 306

La Figure 4.1 ainsi que la Table 4.2 mettent en évidence une seule anomalie détectée en décembre 2006. Il s'agit d'une légère hausse de la consommation, avec une valeur de 9,379, qui contraste modérément avec la tendance habituelle. Bien que l'écart ne soit pas particulièrement prononcé, il peut refléter un comportement ponctuel atypique, potentiellement lié à une variation temporaire d'activité ou à une erreur de mesure.

Client 236

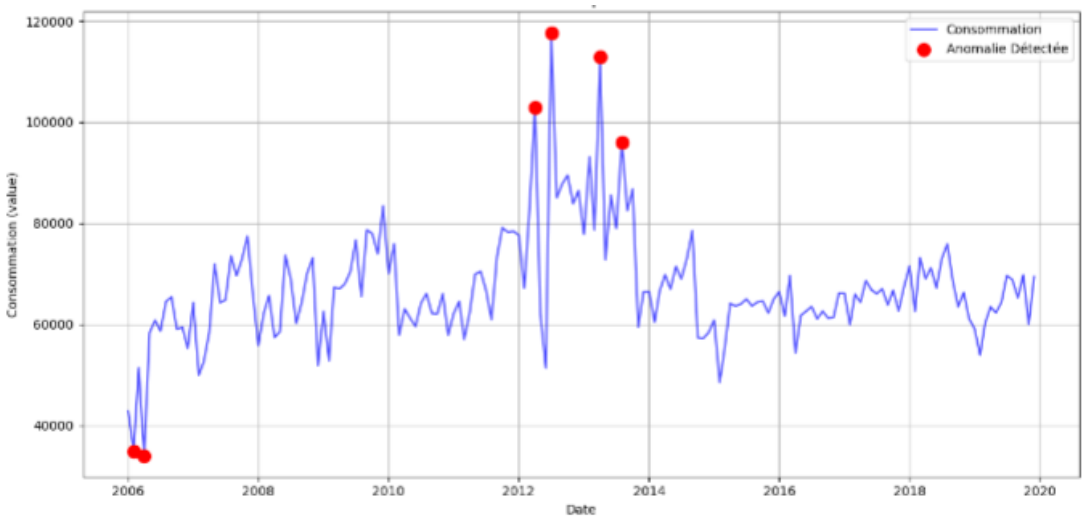


FIGURE 4.2 – Détection d'anomalies pour le Client 236

Date	Mois	Valeur	Méthode
2006-02-01	02/2006	34 915	Hybride (OCSVM+RegLin)
2006-04-01	04/2006	33 969	Hybride (OCSVM+RegLin)
2012-04-01	04/2012	102 884	Hybride (OCSVM+RegLin)
2012-07-01	07/2012	117 597	Hybride (OCSVM+RegLin)
2013-04-01	04/2013	112 837	Hybride (OCSVM+RegLin)
2013-08-01	08/2013	96 058	Hybride (OCSVM+RegLin)

TABLE 4.3 – Anomalies détectées pour le Client 236

La Figure 4.2 et la Table 4.3 révèlent plusieurs anomalies sur une longue période. Les valeurs détectées entre 2012 et 2013, toutes largement supérieures au niveau de consommation moyen de ce client, indiquent des pics inhabituels. Les premières anomalies de 2006 montrent également des valeurs anormalement basses.

Client 243

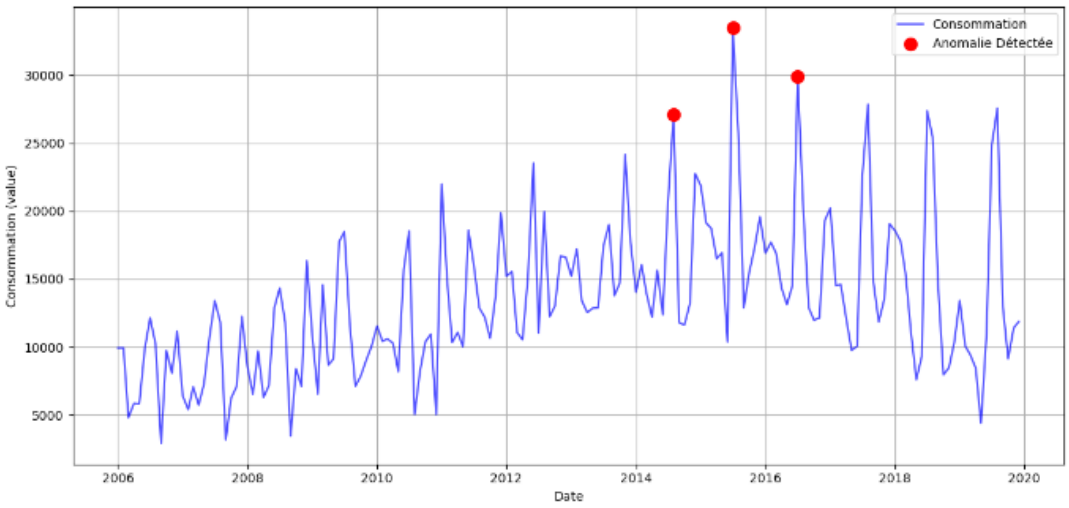


FIGURE 4.3 – Détection d'anomalies pour le Client 243

Date	Mois	Valeur	Méthode
2014-08-01	08/2014	27 125	Hybride (OCSVM+RegLin)
2015-07-01	07/2015	33 459	Hybride (OCSVM+RegLin)
2016-07-01	07/2016	29 872	Hybride (OCSVM+RegLin)

TABLE 4.4 – Anomalies détectées pour le Client 243

La Figure 4.3 accompagnée de la Table 4.4 indique une série de valeurs atypiques sur trois années consécutives. Ces hausses modérées mais répétées peuvent suggérer une tendance latente ou des comportements de consommation irréguliers. Bien qu'aucune valeur ne soit extrême, leur récurrence justifie leur détection par le modèle hybride, qui capte subtilement ces écarts au sein d'un profil par ailleurs stable.

4.3.3 Discussion des Résultats

Les résultats confirment la pertinence de la stratégie hybride OCSVM-Régression Linéaire pour :

— **Gérer la complexité des données réelles** : L'approche individualisée permet de s'adapter à chaque profil client.
— **Identifier des anomalies diverses** : Des consommations très basses ou très élevées sont détectées avec efficacité.
— **Réduire les faux positifs** : Grâce au filtrage par régression, les anomalies sont validées selon leur écart significatif par rapport aux tendances.

4.4 Développement de l'Outil d'Analyse

Cette section détaille la conception et la mise en œuvre de l'application logicielle développée, qui concrétise la stratégie hybride de détection d'anomalies. L'objectif principal de cet outil est de rendre la méthode accessible, opérationnelle et utilisable par les analystes de SONELGAZ, sans nécessiter de compétences en programmation. Il transforme une approche algorithmique en une solution pratique pour l'identification des comportements de consommation anormaux.

4.4.1 Choix Technologiques et Environnement de Développement

Pour le développement de cette application, des choix technologiques spécifiques ont été effectués afin de garantir une robustesse, une flexibilité et une facilité d'utilisation optimales :

— **Langage de Programmation : Python.** Python a été sélectionné pour sa polyvalence, son écosystème riche en bibliothèques dédiées à la science des données et au machine learning (NumPy, Pandas, Scikit-learn, Matplotlib), sa lisibilité et sa grande communauté. Ces atouts ont permis une implémentation efficace des algorithmes complexes et une manipulation aisée des données.

FIGURE 4.4 – Logo du langage de programmation Python

— **Framework d'Interface Graphique (GUI) : Streamlit.** Ce framework s'est avéré être le choix idéal pour prototyper rapidement et développer une application web interactive. Streamlit permet de transformer des scripts Python en interfaces utilisateurs complètes avec un minimum de code, facilitant ainsi la création d'une démonstration fonctionnelle et interactive pour l'utilisateur final. Sa compatibilité native avec les DataFrames Pandas et Matplotlib simplifie grandement l'intégration des résultats d'analyse et des visualisations.

FIGURE 4.5 – Logo du langage de programmation Streamlit

— **Bibliothèques Python Essentielles :**
  — `pandas` : Indispensable pour la manipulation, le nettoyage et l'analyse des jeux de données tabulaires.
  — `numpy` : Utilisée pour les opérations numériques de haute performance, notamment dans les calculs matriciels et statistiques.
  — `scikit-learn` : Fournit les implémentations des algorithmes de machine learning tels que `OneClassSVM`, `LinearRegression` et `StandardScaler`, qui sont au cœur de la stratégie hybride.
  — `matplotlib` : Employée pour la génération des visualisations graphiques des séries temporelles et des anomalies détectées.
  — `openpyxl` : Permet la lecture et l'écriture des fichiers au format Microsoft Excel (.xlsx), assurant la compatibilité avec les données fournies par SONELGAZ.

FIGURE 4.6 – Logo des différentes bibliothèques de Python

— **Environnement de Développement Intégré (IDE) :** Le développement a été réalisé principalement sous un environnement tel que Visual Studio Code, offrant des fonctionnalités de débogage, de gestion de code et d'intégration avec les terminaux Python.

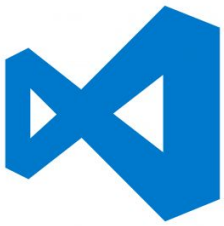


FIGURE 4.7 – Logo de Visual Studio Code

4.4.2 Architecture Logicielle de l'Application

L'application a été conçue selon une architecture modulaire, favorisant la clarté du code, la facilité de maintenance et la capacité d'évolution. Cette structure sépare les préoccupations, permettant à chaque composant de se concentrer sur une tâche spécifique.

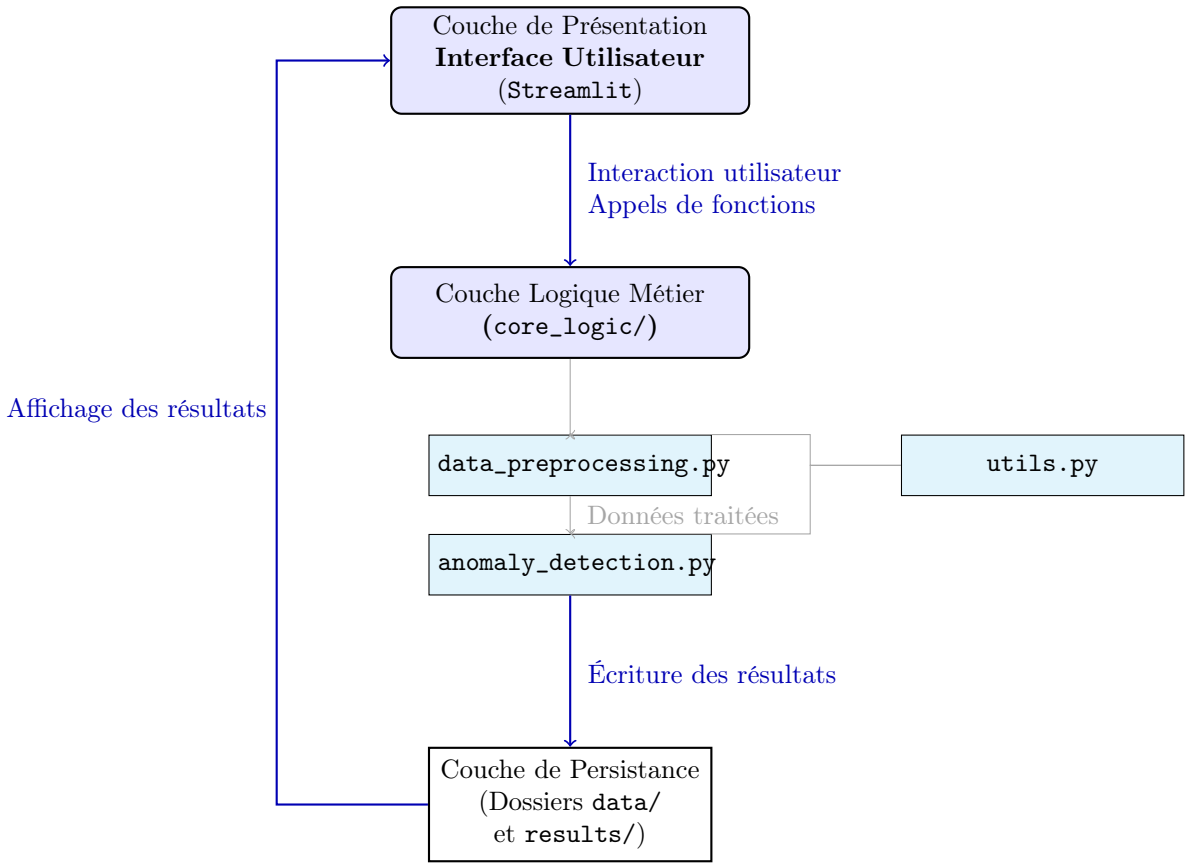


FIGURE 4.8 – Architecture modulaire détaillée de l'application de détection d'anomalies.

L'architecture est composée des éléments principaux suivants :

— **Couche de Présentation (`app.py`) :** Il s'agit du fichier principal de l'application développée avec Streamlit. Cette couche est chargée de concevoir l'interface utilisateur, de traiter les interactions (comme les saisies), de coordonner les requêtes vers les modules de logique métier, et enfin d'afficher les résultats.

— **Couche Logique Métier (`core_logic/`) :** Ce répertoire contient les modules Python qui encapsulent la logique fonctionnelle du système :
  — `utils.py` : Fournit des fonctions utilitaires génériques, telles que le chargement de fichiers (Excel, CSV) et la sauvegarde des résultats (fichiers CSV, images PNG).
  — `data_preprocessing.py` : Assure le nettoyage et le prétraitement des données. Cela comprend la conversion des types, le traitement des valeurs manquantes (imputation ou suppression), ainsi que le filtrage des séries temporelles ne respectant pas un seuil minimal de points valides.
  — `anomaly_detection.py` : Implémente la stratégie hybride de détection d'anomalies. Ce module combine l'OCSVM et la régression linéaire pour chaque client, calcule les résidus, identifie les anomalies, et génère les graphiques nécessaires à l'analyse.

— **Couche de Persistance (Dossiers `data/` et `results/`) :** L'application n'utilise pas de base de données relationnelle, mais interagit avec le système de fichiers. Le dossier `data/` contient les fichiers d'entrée, tandis que `results/` stocke les sorties produites (CSV, graphiques PNG).

4.4.3 Fonctionnalités clés implémentées

L'application offre un ensemble de fonctionnalités robustes, conçues pour couvrir tout le processus de détection d'anomalies, depuis l'ingestion des données jusqu'à la visualisation des résultats. L'objectif est de fournir aux analystes de SONELGAZ un outil complet pour mener des investigations ciblées et efficaces.

— **Gestion flexible des données d'entrée :** L'outil permet de spécifier un « **Chemin du dossier des fichiers clients** » (Figure 4.9) qui charge et concatène automatiquement tous les fichiers de consommation au format `.xlsx` ou `.csv` présents dans ce répertoire. Cette fonctionnalité élimine le traitement manuel fichier par fichier, ce qui simplifie l'ingestion des données pour un grand nombre de clients. Des messages d'information signalent le succès du chargement et les éventuels fichiers ignorés.

— **Options avancées de prétraitement :** Une section dédiée aux « **Options de Prétraitement** » dans la barre latérale permet un contrôle fin de la qualité des données (Figure 4.10) :

. **Gestion des valeurs manquantes (`value`) :** Un menu déroulant (`selectbox`) permet de choisir la stratégie à appliquer aux valeurs de consommation manquantes : `'delete_client'` (suppression du client), `'mean_imputation'` (remplacement par la moyenne), `'median_imputation'` (remplacement par la médiane) ou `'linear_interpolation'` (interpolation linéaire).

. **Points de données minimum par client :** Un champ numérique permet de fixer le nombre minimal d'observations nécessaires pour inclure une série client dans l'analyse, afin d'éviter de traiter des données trop fragmentées.

— **Configuration personnalisable de la détection d'anomalies :** La même section permet d'ajuster les hyperparamètres de la stratégie hybride de détection (Figure 4.10) :

. **Nu (OCSVM) :** Un curseur (`slider`) contrôle la borne supérieure de la fraction d'observations aberrantes. Valeur par défaut : `0.05`.

. **Gamma (OCSVM) :** Un sélecteur (`selectbox`) permet de fixer le paramètre du noyau RBF. L'option par défaut est `'scale'`.

. **Seuil écart-type des résidus :** Un curseur ajuste le seuil, exprimé en multiples de l'écart-type des résidus (par défaut : `2.50`), utilisé pour détecter les anomalies post-régression.

— **Présentation intégrée des résultats :** Après analyse, l'application fournit une synthèse complète :

. **Résumé des statistiques globales :** Nombre total d'anomalies, nombre de clients concernés, pourcentage de points identifiés comme anormaux.

. **Tableau détaillé des anomalies :** Table interactive (`st.dataframe`) affichant les anomalies (identifiant du client, date, valeur, type d'anomalie). Voir Figure 4.11.

. **Visualisation graphique des anomalies :** Affichage de graphiques pour un nombre configurable de clients, avec les points anormaux clairement identifiés (Figure 4.12).

— **Export des résultats :** Pour permettre une réutilisation ou un archivage :

. Téléchargement des anomalies sous format CSV via un bouton.

. Sauvegarde automatique des graphiques dans le dossier `results/anomalies_plots`.

4.4.4 Interface utilisateur (GUI) et expérience utilisateur

L'interface est développée avec `Streamlit` pour sa simplicité et sa capacité à produire rapidement des applications web interactives axées sur les données.

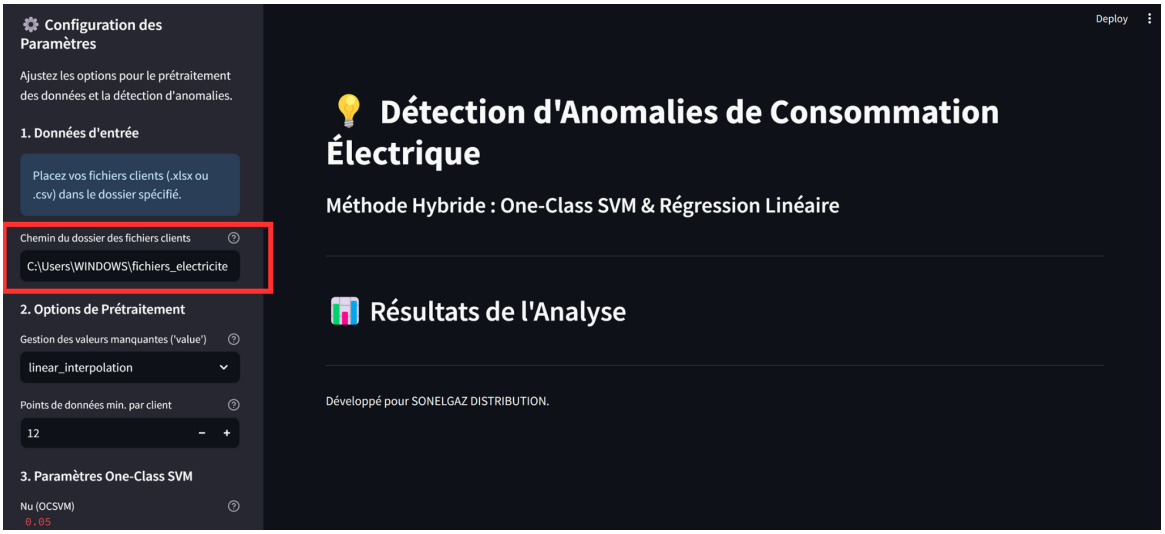


FIGURE 4.9 – Interface principale de l'application au démarrage.

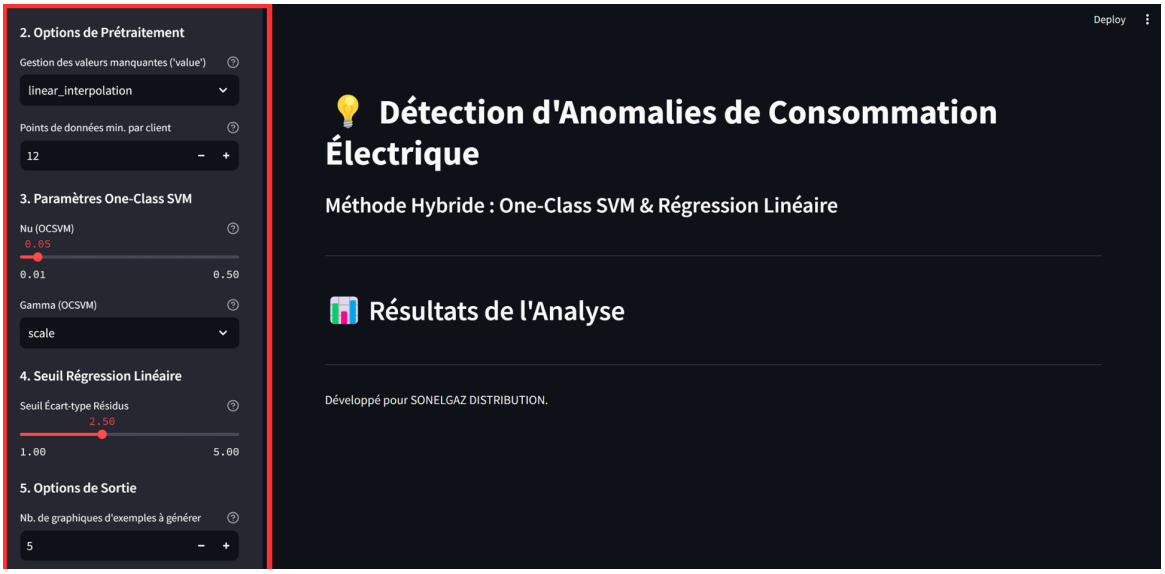


FIGURE 4.10 – Section de configuration du prétraitement et des paramètres de détection.

L'interface est divisée en deux parties principales :

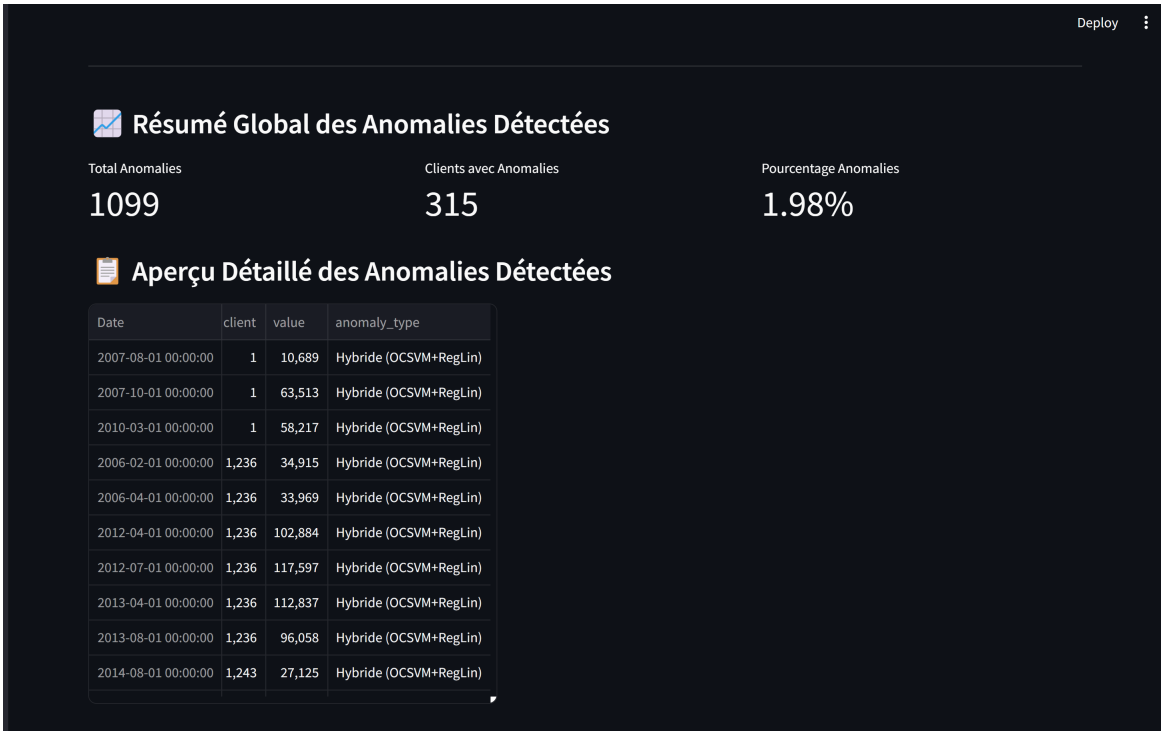
— Une **barre latérale à gauche** intitulée **"Configuration des paramètres"**, regroupant toutes les options d'entrée et de réglage des algorithmes.

— Une **zone principale à droite** dédiée à l'affichage des résultats, incluant le titre **"Détection d'Anomalies de Consommation Électrique"** et l'indication de la **"Méthode Hybride : One-Class SVM & Régression Linéaire"**.

Contrôles et réactivité : Les paramètres sont ajustables via des widgets interactifs : champs texte, `selectbox`, `slider`, `number_input`, etc. Chaque widget est accompagné d'un libellé clair et d'une infobulle explicative (icône). Le lancement de l'analyse s'effectue via un bouton **"Lancer la Détection d'Anomalies"** avec une icône fusée. Pendant l'exécution, des messages d'état (info, succès, erreur) s'affichent pour informer l'utilisateur en temps réel.

Visualisation des résultats : Le tableau des anomalies est interactif (filtrage, tri), et les graphiques

superposent la série temporelle avec les points anormaux en rouge pour une lecture rapide et intuitive.



Date	client	value	anomaly_type
2007-08-01 00:00:00	1	10,689	Hybride (OCSVM+RegLin)
2007-10-01 00:00:00	1	63,513	Hybride (OCSVM+RegLin)
2010-03-01 00:00:00	1	58,217	Hybride (OCSVM+RegLin)
2006-02-01 00:00:00	1,236	34,915	Hybride (OCSVM+RegLin)
2006-04-01 00:00:00	1,236	33,969	Hybride (OCSVM+RegLin)
2012-04-01 00:00:00	1,236	102,884	Hybride (OCSVM+RegLin)
2012-07-01 00:00:00	1,236	117,597	Hybride (OCSVM+RegLin)
2013-04-01 00:00:00	1,236	112,837	Hybride (OCSVM+RegLin)
2013-08-01 00:00:00	1,236	96,058	Hybride (OCSVM+RegLin)
2014-08-01 00:00:00	1,243	27,125	Hybride (OCSVM+RegLin)

FIGURE 4.11 – Aperçu détaillé du tableau des anomalies détectées.

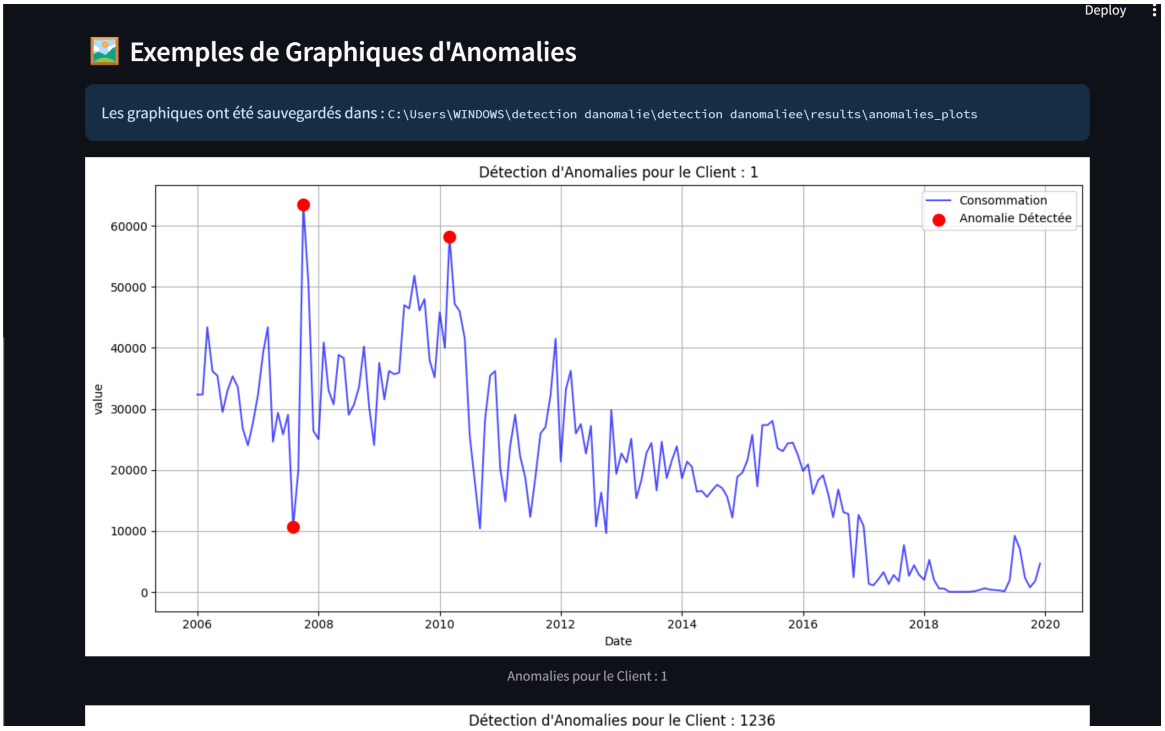


FIGURE 4.12 – Exemple de visualisation graphique des anomalies pour un client.

Conclusion

Ce chapitre a concrétisé la stratégie de détection d'anomalies en la déployant sur des données réelles de consommation électrique et en développant un outil applicatif complet. La description du jeu de données de SONELGAZ DISTRIBUTION a mis en lumière l'importance d'un prétraitement rigoureux pour garantir la qualité et la fiabilité des séries temporelles analysées. L'implémentation de la stratégie hybride, combinant OCSVM et Régression Linéaire, a démontré sa capacité à identifier efficacement diverses anomalies, qu'elles soient discrètes ou plus prononcées, tout en s'adaptant aux spécificités de chaque profil client. Les résultats quantitatifs agrégés ont confirmé la pertinence de l'approche, tandis que les études de cas qualitatives ont illustré visuellement sa robustesse. Plus important encore, le développement d'une application conviviale basée sur Python et Streamlit, dotée d'une architecture modulaire claire et de fonctionnalités étendues, a permis de transformer cette méthodologie complexe en un outil opérationnel et accessible. Cette application représente une solution clé en main pour les analystes de SONELGAZ, leur offrant un moyen efficace d'identifier et d'interpréter les comportements de consommation anormaux, marquant ainsi le passage réussi d'une preuve de concept à une solution pratique et déployable.

Conclusion Générale

Ce travail a exploré la détection d'anomalies dans les séries temporelles, en se concentrant sur la consommation électrique, un domaine où la fiabilité des données est essentielle. L'objectif principal était de mettre en lumière les comportements inhabituels pouvant révéler des erreurs, des pannes ou même des tentatives de fraude.

La démarche a commencé par une analyse approfondie des concepts fondamentaux liés aux séries temporelles et à la notion d'anomalie. Cette base a permis de mieux cerner les défis liés à l'identification de points atypiques dans des données souvent bruitées, irrégulières ou saisonnières.

Ensuite, plusieurs méthodes ont été mises en œuvre, allant des approches statistiques classiques aux modèles d'apprentissage automatique plus récents. L'étude a mis en évidence les points forts de chaque méthode, mais aussi leurs limites : certaines offrent une bonne sensibilité mais manquent de précision, d'autres réduisent les faux positifs mais passent à côté d'anomalies importantes.

Pour améliorer les résultats, une stratégie hybride a été développée. Elle combine l'OCSVM (efficace pour repérer les cas suspects) avec une régression linéaire (utilisée pour filtrer les fausses alertes). Cette combinaison s'est révélée plus équilibrée et performante, notamment en termes de F1-score.

La méthodologie a ensuite été testée sur des données réelles fournies par SONELGAZ. Après un important travail de nettoyage, une application a été développée à l'aide de Python et Streamlit. Elle permet de charger, analyser, visualiser et interpréter les séries temporelles de consommation. Cette plateforme, simple d'utilisation, rend accessible une technologie avancée aux analystes métiers.

Ce projet illustre ainsi la capacité à transformer des concepts mathématiques et algorithmiques en un outil opérationnel, utile dans un contexte industriel. Il montre qu'il est possible d'améliorer la détection de fraudes dans les données réelles grâce à une approche à la fois rigoureuse, innovante et pragmatique.

Perspectives

Plusieurs pistes d'amélioration peuvent enrichir ce travail dans le futur :

— **Analyse multivariée** : Intégrer d'autres types de consommation (gaz, eau, etc.) pour détecter des anomalies croisées.

— **Modèles profonds** : Tester des réseaux de neurones comme les LSTM ou Transformers, adaptés aux données séquentielles.

— **Optimisation automatique** : Mettre en place des méthodes comme l'optimisation bayésienne pour régler automatiquement les hyperparamètres.

— **Détection en temps réel** : Adapter l'outil pour traiter les données en streaming, avec détection instantanée des anomalies.

— **Interface enrichie** : Ajouter des fonctionnalités interactives (zoom, annotations, feedback utilisateur) pour faciliter l'analyse manuelle.

— **Généralisation industrielle** : Tester la solution dans d'autres contextes industriels et envisager son intégration dans un environnement de production.

Bibliographie

[1] C. C. Aggarwal. *Outlier Analysis*. Springer, 2nd edition, 2017.

[2] Yves Aragon. *Séries temporelles avec R : Méthodes et cas*. Paris, 2016.

[3] Raymond Bourbonnais and Virginie Terraza. *Analyse des séries temporelles*. Dunod, Paris, 5 edition, 2020.

[4] M. Braei and S. Wagner. Anomaly detection in univariate time-series : A survey on the state-of-the-art. *arXiv preprint arXiv :2004.00433*, 2020.

[5] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection : A survey. *ACM Computing Surveys*, 41(3) :1–58, 2009.

[6] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection : A survey. *ACM Computing Surveys (CSUR)*, 41(3) :1–58, 2009. Consulté en mai 2025.

[7] Chris Chatfield. *The Analysis of Time Series : An Introduction*. Chapman and Hall/CRC, 6 edition, 2016.

[8] GeeksforGeeks. Interquartile range to detect outliers in data. `https://www.geeksforgeeks.org/interquartile-range-to-detect-outliers-in-data`, August 2024. Consulté en mai 2025.

[9] Christian Gourieroux and Alain Monfort. *Séries temporelles et modèles dynamiques*. ENSAE et CEPE, 2 edition, 1995. Collection « Économie et Statistiques Avancées ».

[10] D. M. Hawkins. *Identification of Outliers*. Chapman and Hall, London, 1980.

[11] V. Hodge and J. Austin. A survey of outlier detection methodologies. *Artificial Intelligence Review*, 22(2) :85–126, 2004.

[12] R. J. Hyndman and G. Athanasopoulos. *Forecasting : Principles and Practice*. OTexts, Melbourne, Australia, 3rd edition, 2021.

[13] Rob J. Hyndman, Anne B. Koehler, J. Keith Ord, and Ralph D. Snyder. Exponential smoothing : The state of the art. *International Journal of Forecasting*, 22(4) :443–473, 2008.

[14] Boris Iglewicz and David C. Hoaglin. *How to Detect and Handle Outliers*. ASQC Quality Press, Milwaukee, 1993.

[15] Numenta Inc. Numenta anomaly benchmark (nab) - a1benchmark data. `https://github.com/numenta/NAB`, 2015. Consulté en avril 2025.

[16] Juniper Research. Online payment fraud : Emerging threats, segment analysis & market forecasts 2022–2027, July 2022. Rapport d'étude de marché.

[17] M. G. Kendall. Review. *Journal of the Royal Statistical Society. Series A (General)*, 134(3) :450–453, 1971.

[18] Agnès Lagnoux. Séries chronologiques. Support de cours, Master 1 - MI00141X, ISMAG, Université de Toulouse, non daté.

[19] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *2008 Eighth IEEE International Conference on Data Mining*, pages 413–422. IEEE, 2008. Consulté en mai 2025.

[20] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *Proceedings of the 8th IEEE International Conference on Data Mining (ICDM)*, pages 413–422. IEEE, 2008.

[21] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, pages 413–422. IEEE, 2008.

[22] Douglas C. Montgomery, G. Geoffrey Vining, and Timothy J. Robinson. *Generalized Linear Models*. Wiley, Hoboken, NJ, 2 edition, 2006.

[23] Arthur L. Samuel. Some studies in machine learning using the game of checkers. *IBM Journal of Research and Development*, 3(3) :210–229, 1959.

[24] Bernhard Schölkopf, John C Platt, John Shawe-Taylor, Alex J Smola, and Robert C Williamson. Estimating the support of a high-dimensional distribution. *Neural computation*, 13(7) :1443–1471, 2001. Consulté en mai 2025.

[25] Mohammad Valipour. Long-term runoff study using sarima and arima models in the united states. *Meteorological Applications*, 2015. Article publié en ligne.

[26] J. West and M. Bhattacharya. Intelligent financial fraud detection : A comprehensive review. *Computers & Security*, 57 :47–66, 2016.

Résumé

Ce projet s'intéresse à la détection d'anomalies dans les séries temporelles, un enjeu majeur dans les secteurs sensibles comme l'énergie, la finance ou la cybersécurité. Plusieurs méthodes ont été étudiées, allant des approches statistiques (Z-Score, IQR) aux techniques de prévision (ARIMA, lissage exponentiel) et de machine learning (Isolation Forest, One-Class SVM). Les expérimentations, menées sur le jeu de données A1Benchmark, ont révélé qu'aucune méthode seule n'était capable d'allier haute précision et bon rappel. Pour pallier cette limite, une stratégie hybride combinant le One-Class SVM et la Régression Linéaire a été proposée, permettant une amélioration notable du F1-score. Ce travail a été finalisé par le développement d'un logiciel d'analyse et de détection d'anomalies, traduisant les résultats obtenus en un outil pratique et réutilisable. Cette contribution ouvre la voie à des solutions plus performantes et applicables en entreprise.

Mots-clés : Séries temporelles, Anomalie, Machine Learning, One-Class SVM, Régression linéaire, Méthodes hybrides, Logiciel de détection

Abstract

This project focuses on anomaly detection in time series, a key issue in critical areas such as energy, finance, and cybersecurity. Several methods were analyzed, including statistical approaches (Z-Score, IQR), forecasting techniques (ARIMA, exponential smoothing), and machine learning models (Isolation Forest, One-Class SVM). Experiments using the A1Benchmark dataset showed that no single method could achieve both high precision and recall. To address this, a hybrid approach combining One-Class SVM and Linear Regression was proposed, significantly improving the F1-score. The project was finalized by the development of a software tool for anomaly detection, integrating the research outcomes into a practical and reusable application. This work highlights the relevance of hybrid strategies and paves the way for real-world implementations.

Keywords : Time Series, Anomaly Detection, Machine Learning, One-Class SVM, Linear Regression, Hybrid Methods, Detection Software