

Détection d'anomalies dans des séries temporelles

TX - RAPPORT D'ÉTUDE
Juin 2022

Table des matières

1	Introduction	3
2	Séries temporelles	4
2.1	Définition	4
2.2	Propriétés d'une série temporelle	4
2.3	Séries temporelles et <i>Machine Learning</i>	7
2.3.1	Stationnarité	7
2.3.2	Bruit blanc	8
3	Anomalies dans les séries temporelles	9
3.1	Anomalie ponctuelle	9
3.2	Anomalie contextuelle	9
3.3	Anomalie globale	10
3.4	Anomalie collective	11
4	Revue de l'état de l'art pour la détection d'anomalies dans les séries temporelles	12
4.1	Méthodes non supervisées	12
4.1.1	Clustering	12
4.1.2	Méthodes statistiques	13
4.1.3	Méthodes prédictives	13
4.1.4	Méthodes basées sur les arbres	13
4.2	Méthodes supervisées	14
4.2.1	K plus proches voisins	14
4.2.2	Régression logistique	14
4.2.3	Arbres de décision	15
4.2.4	Utilisation de réseaux de neurones	15
4.2.5	Pré-traitements utiles pour les méthodes supervisées	16
5	Implémentation d'une méthode de détection d'anomalies sur des séries temporelles en rapport avec la <i>smart city</i>	17
5.1	Choix des données	17
5.2	Analyse exploratoire des données	18
5.3	Choix de la méthode de détection	19
5.3.1	Justification du choix du modèle	20
5.4	Pré-traitement des données	21
5.5	Processus <i>SARIMA</i>	22
5.6	Entraînement des modèles prédictifs (modèles <i>SARIMA</i>)	23
5.6.1	Détermination des hyper-paramètres des modèles	23
5.6.2	Entraînement des modèles et visualisations	25
5.7	Entraînement d'un modèle de classification	27

5.7.1	Choix du modèle	27
5.7.2	Explications de la forêt aléatoire	28
5.7.3	Analyse des métriques d'évaluations des modèles	29
5.8	Comparaison avec des modèles sans la variable <i>difference</i>	30
5.9	Comparaison avec un modèle naïf	31
5.9.1	Estimation de noyau	32
5.9.2	Application	32
6	Conclusion et pistes d'améliorations	34

Chapitre 1

Introduction

L'objectif de cette TX est d'utiliser des techniques de *Machine Learning* pour détecter des problèmes de fuite dans le réseau d'alimentation en eau. En coopération avec la société ISTA du consortium Maille'Immo, nous devons implémenter une méthode permettant de détecter des anomalies dans le cadre d'une maintenance préventive. L'utilisation de techniques de *Deep Learning* ne rentre pas dans le cadre de cette TX.

Les données de cette étude devaient normalement être fournies par la société ISTA mais suite à un problème interne, nous n'avons pu obtenir ces données. Nous avons donc utilisé des données en *open data*.

Dans ce rapport, nous définirons dans un premier temps les termes et phénomènes liés à cette étude (séries temporelles, anomalies dans les séries temporelles). Nous ferons ensuite une revue de l'état de l'art des méthodes actuellement utilisées pour la détection d'anomalies dans les séries temporelles. Suite à cette revue, nous implémenterons une méthode, en lien avec les méthodes actuelles énoncées, permettant de détecter des anomalies à partir de données trouvées en *open data*. Enfin nous conclurons sur la méthode employée ainsi que ses performances, et nous proposerons des améliorations possibles à cette méthode.

Chapitre 2

Séries temporelles

2.1 Définition

Une série temporelle est une suite de valeurs qui représente l'évolution d'une quantité au cours du temps. Une série temporelle ne capture pas seulement la valeur d'une donnée à un instant t , elle permet également d'observer le changement des données dans le temps. Un exemple de série temporelle mettant en relation les températures relevées chaque jour par un capteur est montré figure 2.1

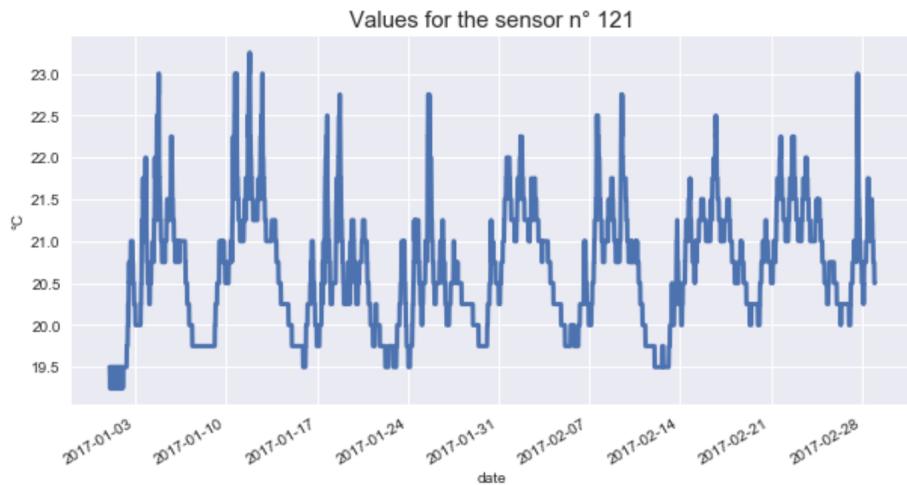


FIGURE 2.1 – Exemple d'une série temporelle

2.2 Propriétés d'une série temporelle

Une série temporelle peut être représentée par différentes statistiques. Les statistiques simples permettant de décrire une série temporelle sont les suivantes :

Moyenne arithmétique : valeur moyenne de la série, c'est à dire la somme de toutes ses valeurs divisée par le nombre de valeurs dans la série :

$$\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i$$

écart type : mesure la dispersion des valeurs autour de la valeur moyenne :

$$\sigma = \sqrt{\frac{1}{n} \sum_{i=1}^n (x_i - \bar{x})^2}$$

auto-corrélation : fait référence au fait que dans une série temporelle, la mesure d'un phénomène à un instant t peut être corrélée aux mesures $t - 1, t - 2, \dots, t - p$ ou aux mesures $t + 1, t + 2, \dots, t + p$. L'auto-corrélation permet donc de rendre compte de cette corrélation entre les différents décalages d'une même série. L'auto-corrélation est donnée par la formule suivante :

$$\hat{\rho}(t, t+h) = \frac{\hat{C}(t, t+h)}{\hat{C}(t, t)\hat{C}(t+h, t+h)}$$

avec

$$\hat{C}(t, t+h) = \frac{1}{n} \sum_{i=1}^{n-h} (x_i - \bar{x})(x_{i+h} - \bar{x})$$

la covariance empirique entre la valeur de la série à un instant t et la valeur de la série à un instant $t+h$. Nous remarquerons qu'il n'est pas judicieux de choisir h trop grand, car le nombre de réalisations serait trop faible pour pouvoir estimer correctement la covariance et ainsi l'auto-corrélation.

En plus de ces statistiques simples, une série temporelle peut être décrite par sa saisonnalité, sa tendance, sa périodicité et son bruit.

tendance : hausse ou baisse sur le long terme de la valeur mesurée de la série temporelle. Une tendance peut être une constante, affine ($m_t = at + b$), polynomiale, exponentielle ect. La figure 2.2 présente différentes tendances observables.

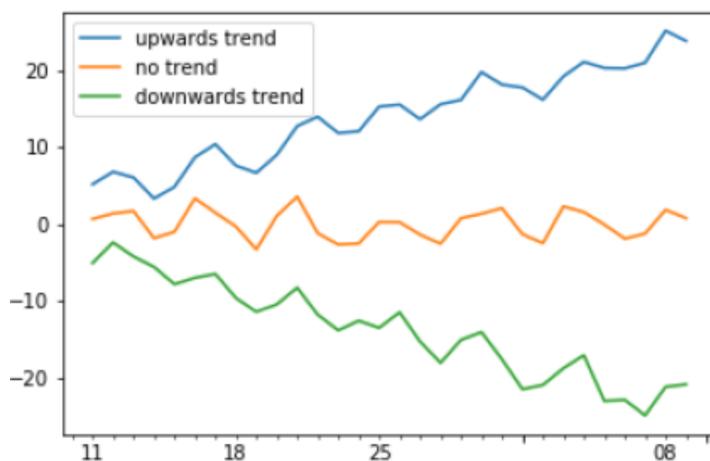
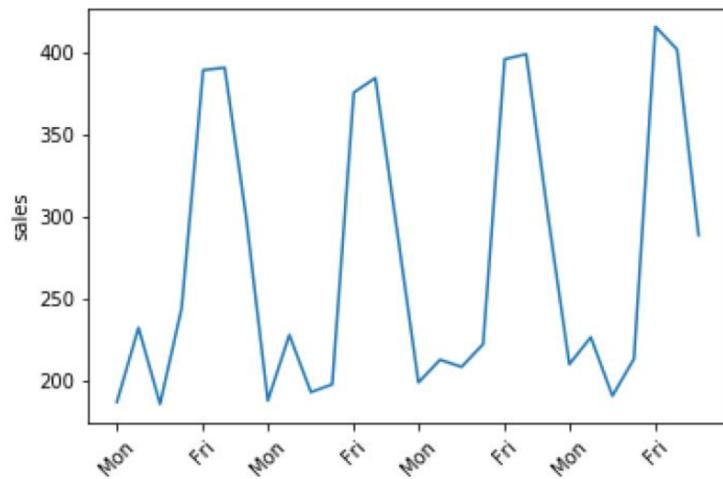


FIGURE 2.2 – Différentes tendances observables dans des séries temporelles

saisonalité : cycle de variation des valeurs d'une série temporelle. Par exemple, une série temporelle peut avoir une saisonnalité journalière, hebdomadaire ou mensuelle. Un exemple de série temporelle mettant en relation le nombre de ventes d'un produit chaque jour et ayant une saisonnalité hebdomadaire est montré figure 2.3 :



```
decomposition.plot()
plt.show()
```

Un exemple de décomposition est donné figure 2.5.

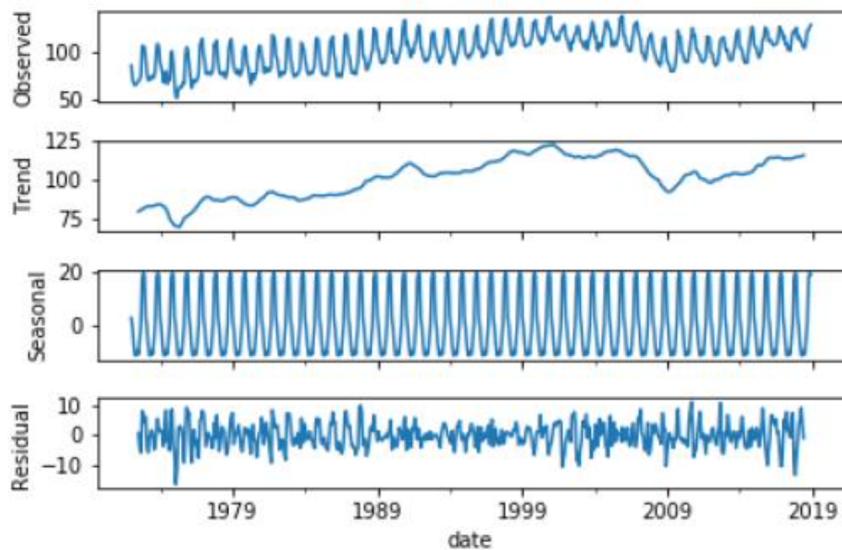


FIGURE 2.5 – Décomposition d'une série temporelle à l'aide de *python*

2.3 Séries temporelles et *Machine Learning*

Pour utiliser des techniques de *Machine Learning* avec des séries temporelles, il faut que celles-ci vérifient certaines propriétés.

2.3.1 Stationnarité

La première propriété qu'une série temporelle doit vérifier est la stationnarité. Une série est dite stationnaire lorsque ses propriétés statistiques ne changent pas au cours du temps (moyenne, écart type, ect). Cette propriété est indispensable pour modéliser une série temporelle et effectuer des prédictions à l'aide de techniques de *Machine Learning*. Un exemple de série stationnaire et non stationnaire est montré figure 2.6.

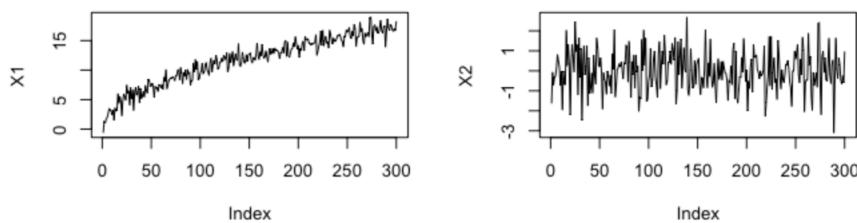


FIGURE 2.6 – Série non stationnaire (à gauche) et série stationnaire (à droite)

Pour tester la stationnarité d'une série temporelle, il est possible d'utiliser plusieurs tests statistiques, dont le plus connu est le test augmenté de Dickey-Fuller (*Augmented Dickey-Fuller test* en anglais).

Si une série temporelle n'est pas stationnaire, il est possible de la rendre stationnaire en appliquant certaines techniques, dont la plus commune est la différence : la nouvelle valeur à l'instant

t est égale à la différence entre la valeur à l'instant t et la valeur à l'instant $t - 1$, ce qui se traduit mathématiquement par $Y_t = y_t - y_{t-1}$.

2.3.2 Bruit blanc

Une série présentant un bruit blanc est une série où les futures valeurs sont imprédictibles car les valeurs semblent être générées de manière aléatoire : il n'existe pas de relation entre la valeur à l'instant t et les valeurs aux instants $t - 1, \dots, t - p$ (cf figure 2.7). Une série présentant un bruit blanc a les statistiques suivantes :

- moyenne empirique constante au cours du temps
- écart type constant au cours du temps
- quasiment aucune auto-corrélation à tous les décalages

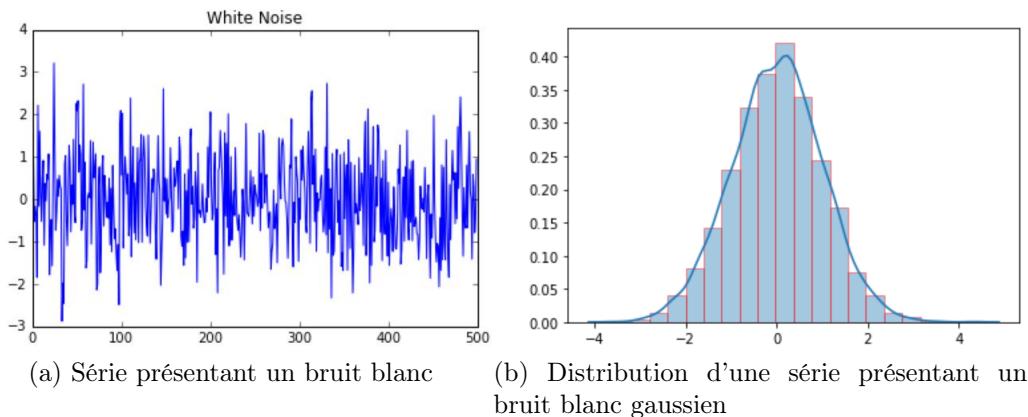


FIGURE 2.7 – Série présentant un bruit blanc (à gauche) ainsi qu'une distribution de série présentant un bruit blanc gaussien (à droite)

Ce type de série est très problématique pour l'utilisation de modèles de prédictions (comme les modèles *ARMA*, *ARIMA*, *SARIMA*) car le fait que les valeurs semblent être générées de manière aléatoire empêche d'établir un modèle permettant de prédire les valeurs futures en fonction des valeurs passées. Il est important de noter qu'une série présentant un bruit blanc est stationnaire. Pour utiliser des techniques de *Machine Learning*, il faudra s'assurer au préalable que la série temporelle ne présente pas de bruit blanc.

Chapitre 3

Anomalies dans les séries temporelles

Maintenant que nous avons défini ce qu'est une série temporelle, quelles sont ses propriétés et statistiques ainsi que les propriétés qu'elle doit vérifier pour pouvoir appliquer des modèles prédictifs, nous pouvons maintenant introduire la notion d'anomalies.

Par définition, une anomalie est "*ce qui s'écarte de la norme, de la régularité, de la règle*" (définition du *Larousse*). Dans une série temporelle, une anomalie se caractérise par une valeur abbérante, inattendue, compte tenu de la distribution globale des valeurs de la série. Nous pouvons distinguer 4 types majeurs d'anomalies dans les séries temporelles : les anomalies ponctuelles, contextuelles, globales et collectives.

3.1 Anomalie ponctuelle

Une valeur considérée comme anormale lorsque nous la comparons à l'ensemble des valeurs d'une série temporelle est considérée comme une anomalie ponctuelle. En voici un exemple

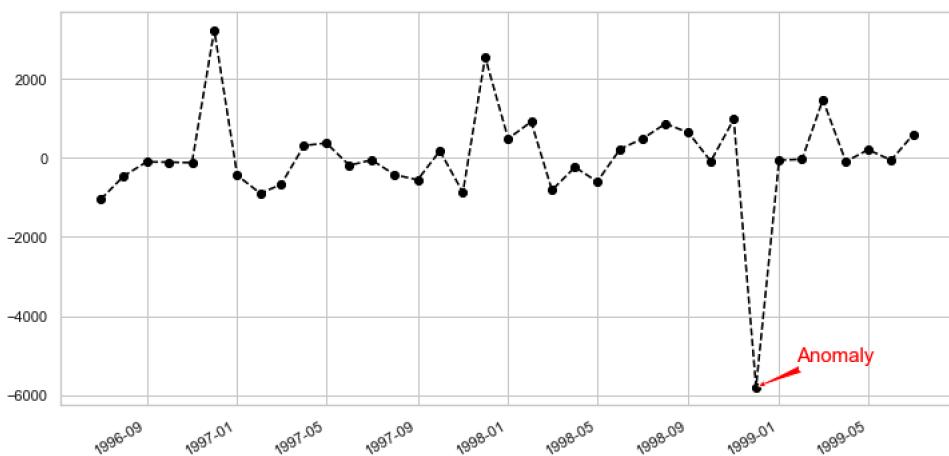


FIGURE 3.1 – Anomalie ponctuelle

3.2 Anomalie contextuelle

Une anomalie contextuelle est une valeur anormale au sein d'une série temporelle, que cette série soit *uni-variée* (c'est à dire ne présentant qu'une seule variable) ou *multi-variée* (c'est à dire présentant plusieurs variables). Il est important de noter que ce type d'anomalies dépend du contexte (d'où la notion d'anomalie contextuelle). Par exemple, nous pourrions considérer comme normale le fait que la consommation d'électricité due à l'allumage des lumières soit

relativement élevée le soir, mais nous considérerions anormale une consommation de ce type relativement élevée en plein milieu du jour. Un exemple d'anomalie contextuelle est donné figure 3.2.

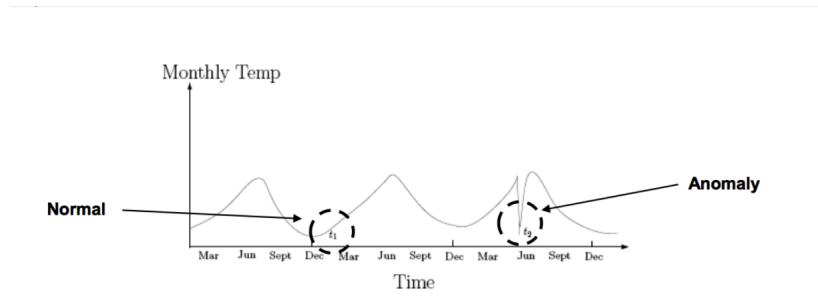


FIGURE 3.2 – Anomalie contextuelle dans le cadre des températures moyennes mesurées par mois

3.3 Anomalie globale

Ce type d'anomalie fait référence au fait qu'un série temporelle soit considérée entièrement comme anormale comparée à d'autres séries temporelles. Nous pourrions prendre comme exemple le nombre de vente d'un produit quelconque au cours de X années. Si de manière générale, les ventes au cours de chaque année sont relativement similaires mais les ventes au cours de l'année K sont beaucoup moins bonnes que celles des autres années, alors la série temporelle représentant les ventes de ce produit lors de l'année K peut être considérée comme étant une anomalie globale. Par exemple, la figure 3.3 montre, en vert, des séries dites "normales" et, en rouge, des séries dites "anormales".

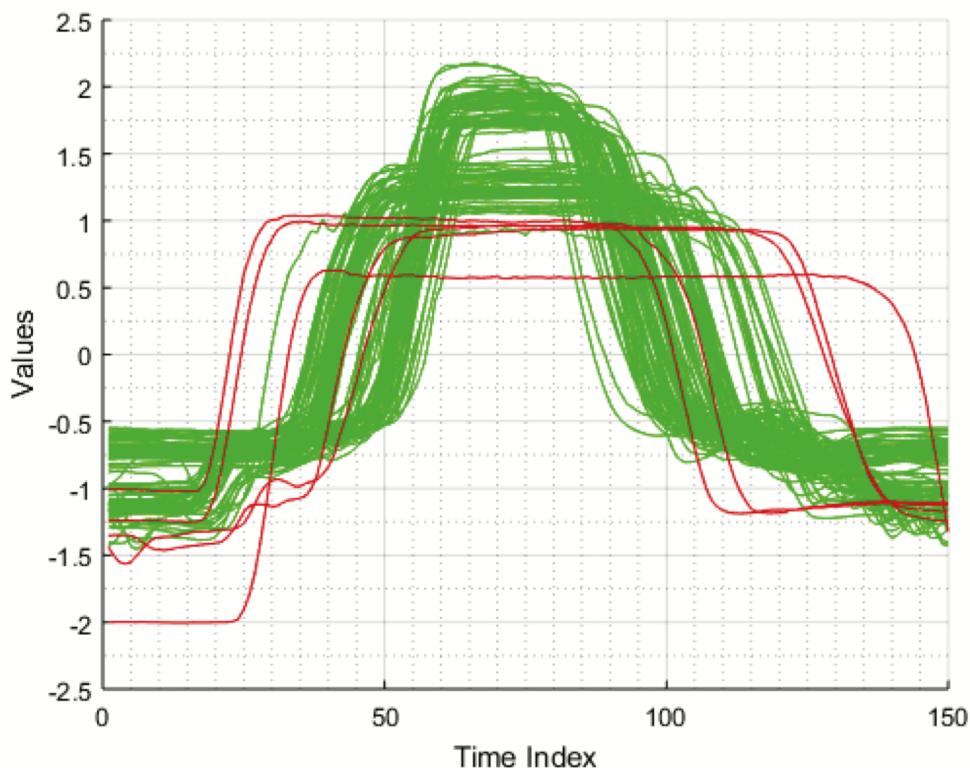


FIGURE 3.3 – Séries temporelles normales (en vert) et anormales (en rouge)

3.4 Anomalie collective

Une anomalie collective est un ensemble de valeurs se suivant dans le temps d'une même série temporelle déviant des valeurs normales de la série. Dit d'une autre manière, ce type d'anomalie est représenté par des sous-séries anormales au sein d'une même série temporelle. Par exemple, une fuite d'eau caractérisée par une hausse du flux d'eau dans un tuyau pendant 3 jours avant réparation peut représenter une anomalie collective. Voici un exemple d'anomalie collective au sein d'une même série temporelle (figure 3.4) :

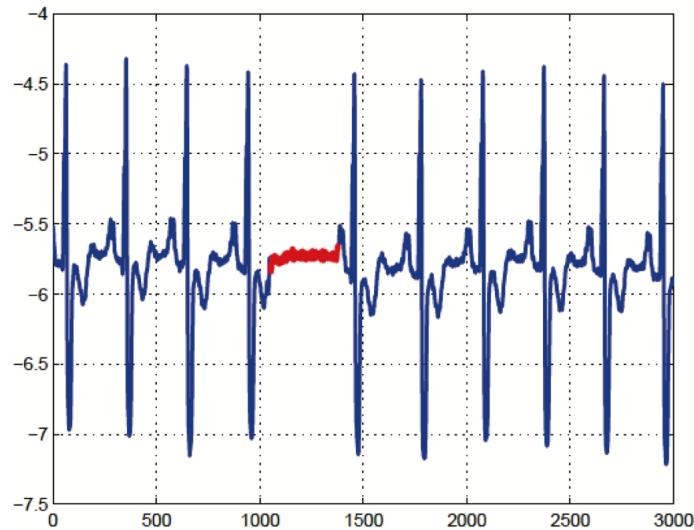


FIGURE 3.4 – Série temporelle présentant une anomalie collective (en rouge)

Ces 4 différents types d'anomalies peuvent être détectés grâce à différentes techniques et différentes approches. Dans le chapitre suivant, nous allons faire une revue de l'état de l'art des méthodes utilisées pour la détection d'anomalies dans les séries temporelles.

Chapitre 4

Revue de l'état de l'art pour la détection d'anomalies dans les séries temporelles

La détection d'anomalies dans les séries temporelles, comme d'autres problèmes de *Machine Learning*, peut être séparée en 2 grandes méthodes :

- les méthodes supervisées : les valeurs sont labellisées (anomalie ou normale) et nous apprenons un modèle permettant de classifier si une nouvelle valeur est une anomalie ou non.
- les méthodes non supervisées : aucune valeur n'est labellisée et nous essayons de regrouper les valeurs du jeu de données en groupes homogènes.

Le choix du type de méthode pour un problème de *Machine Learning* dépend essentiellement du type de données que nous avons. Nous allons présenter dans ce chapitre différentes méthodes non supervisées et supervisées de détection d'anomalies dans les séries temporelles.

4.1 Méthodes non supervisées

4.1.1 Clustering

Le *clustering* est une méthode visant à diviser l'ensemble de données hétérogène en différents groupes (ou classes) homogènes. L'algorithme des *K-Means* est le plus connu et le plus utilisé. Il s'applique à des données quantitatives et se base sur les distances entre les valeurs afin de les regrouper dans k groupes les plus homogènes possibles. L'objectif de l'algorithme est de minimiser l'inertie intra-classe (ce qui est équivalent à maximiser l'inertie inter-classe) en définissant k centres et en attribuant à un point du jeu de données l'indice du groupe d'où il est le plus proche du centre. A chaque itération de l'algorithme les centres sont re-calculés et les points ré-attribués. L'algorithme s'arrête lorsque les centres ne sont plus modifiés.

Le nombre de cluster k est un hyper-paramètre de l'algorithme et est défini de manière arbitraire (en général, nous utilisons la méthode du coude). L'idée d'utiliser les méthodes de *clustering* dans la détection d'anomalies est la suivante : les anomalies n'appartiennent à aucun groupe ou alors forment un petit groupe à elles seules. Un exemple de regroupement permettant d'identifier des anomalies est présenté figure 4.1 : le groupe se situant en haut à gauche sur le graphique ne contient qu'un seul point et peut être considéré comme une anomalie.

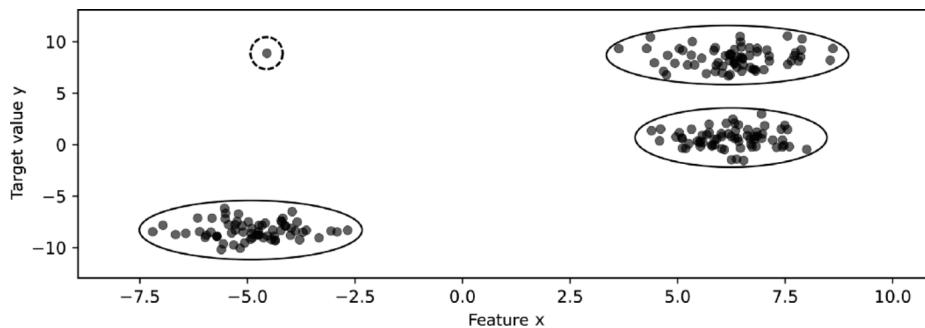


FIGURE 4.1 – Détection d'anomalies en utilisant une méthode de *clustering*

Ces méthodes peuvent être utiles pour détecter les anomalies ponctuelles. En revanche, elle pourrait ne pas être efficaces pour détecter des anomalies contextuelles (en reprenant l'exemple de la figure 3.2, le point anormal serait regroupé avec les points "normaux" de la période hivernale).

4.1.2 Méthodes statistiques

Une autre approche consiste à utiliser des méthodes statistiques ou probabilistes. La méthode du *Z-score* en est la plus directe. Le *Z-score* est calculé de la manière suivante :

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

avec μ la moyenne de l'échantillon et σ l'écart-type à la moyenne de l'échantillon. L'utilisation de cette méthode suppose une distribution normale des valeurs. Les valeurs éloignées de la moyenne (équivalent à un *Z-score* élevé en valeur absolue) sont susceptibles d'être une anomalie. Une approche très similaire consiste à considérer la valeur médiane de l'échantillon et à considérer un certain fractile comme délimitant les valeurs considérées comme une anomalie des autres valeurs.

4.1.3 Méthodes prédictives

Ce type de méthode consiste à prédire la valeur future de la série à l'aide d'un modèle mathématique entraîné, puis de comparer la valeur prédite à un intervalle de confiance. Si la valeur est hors de cet intervalle, elle est considérée comme étant une anomalie. Le modèle mathématique choisi peut être un modèle statistique, comme les modèles *ARMA*, *ARIMA* et *SARIMA* par exemple, ou alors des réseaux de neurones récurrents (*RNN*) comme les *Long Short Term Memory (LSTM)* par exemple.

4.1.4 Méthodes basées sur les arbres

Des arbres de décision peuvent être utilisés afin de déterminer si un point est une anomalie ou non. L'une des méthodes les plus utilisées est la méthode dite des *Isolation Forest*. Un *Isolation Forest* est une méthode d'apprentissage basée sur l'utilisation d'arbres de décision (*Decision Trees* en anglais). Elle ressemble fortement à la méthode des forêts aléatoires (*Random Forest* en anglais) à la différence que les séparations sont aléatoires : une variable explicative est choisie aléatoirement, ainsi que le seuil de séparation choisi sur cette variable. Ceci est répété jusqu'à ce que toutes les valeurs du jeu de données aient été isolées. Cette procédure est répétée k fois, k étant le nombre d'arbres de décision choisis.

L'idée derrière ce modèle est que si une valeur nécessite beaucoup de séparations pour être

isolée, alors elle n'est pas considérée comme anormale. A l'inverse, si cette valeur nécessite que peu de séparation pour être isolée, alors nous la considérons comme anormale. Un exemple est montré figure 4.2 : le point x_i nécessite un nombre important de séparations pour être isolé et n'est pas considéré comme anormal, tandis que le point x_0 nécessite peu de séparations pour être isolé et nous le considérons donc comme une anomalie.

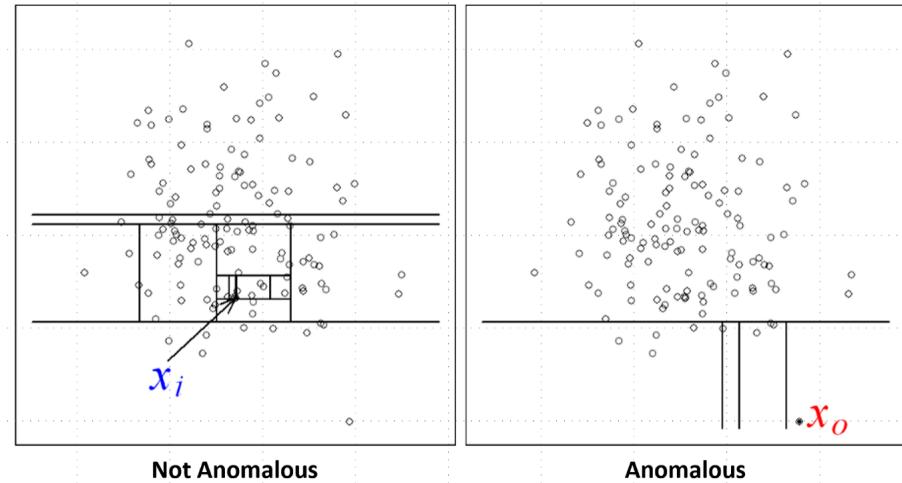


FIGURE 4.2 – Détection d'anomalies en utilisant une méthode d'*Isolation Forest*

4.2 Méthodes supervisées

Il est également possible d'utiliser des méthodes d'apprentissage supervisé sur les séries temporelles afin de déterminer si une valeur est anormale ou non. Pour ceci, nous pouvons appliquer des méthodes classiques de *Machine Learning*.

4.2.1 K plus proches voisins

Cette méthode consiste à estimer le fait qu'une valeur soit normale ou non en se basant sur ses k voisins les plus proches (au sens d'une distance). Le nombre k de voisins peut être choisi à l'aide d'une validation croisée sur un ensemble d'apprentissage, même si dans les faits, cette méthode ne requiert pas réellement d'apprentissage vu que c'est une méthode non paramétrique (il n'y a pas de paramètres à estimer, k étant un hyper-paramètre).

4.2.2 Régression logistique

La régression logistique est une méthode visant à attribuer à une valeur la classe (ici normale ou anomalie) de plus grande probabilité à posteriori ($P(Z = w_i | X = x)$). La régression logistique attribue donc à la valeur i la classe à laquelle elle a le plus de chances d'appartenir. Bien que cette méthode soit une méthode de classification (et donc d'attribution d'un label à une variable) et non une méthode de régression, elle est appelée régression logistique car l'objectif est d'estimer les probabilités à posteriori (c'est donc une valeur continue comprise entre 0 et 1). Sa formule mathématique est (si nous considérons seulement 2 classes) :

$$\mathbb{P}(Z = w_1 | X = x) = \frac{\exp(Y)}{1 + \exp(Y)}$$

$$\mathbb{P}(Z = w_2 | X = x) = \frac{1}{1 + \exp(Y)}$$

où

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_p X_p = \beta X_a^T$$

avec $\beta = (\beta_0, \dots, \beta_p)$ et $X_a = (1, X_1, \dots, X_p)$ le vecteur augmenté de X , p étant le nombre de variables explicatives considérées. L'objectif de l'apprentissage est d'estimer les paramètres β , à l'aide d'un algorithme d'optimisation d'un critère. Par exemple, nous pouvons optimiser la fonction de log-vraisemblance (que nous cherchons à maximiser) et utiliser l'algorithme de *Newton-Raphson* pour l'optimiser.

4.2.3 Arbres de décision

Tout comme dans les méthodes non supervisées, il est possible d'utiliser des arbres de décision afin de déterminer le caractère anormal d'une valeur ou non. Rappelons qu'un arbre de décision vise à séparer le jeu de données d'apprentissage en sous-ensembles homogènes selon un critère. Le plus souvent, nous utilisons le critère de *Gini* :

$$G(p) = \sum_{k=1}^g p_k(1 - p_k)$$

ou alors le critère d'entropie :

$$E(p) = - \sum_{k=1}^g p_k \ln(p_k)$$

Dans ces formules, g représente le nombre de classes considérées (dans notre cas $g = 2$) et p_k représente la proportion d'individus appartenant à la classe k par rapport à l'ensemble total. Pour la détection d'anomalies dans les séries temporelles, il est possible d'utiliser des arbres de décision simples, des forêts aléatoires ou des méthodes de *boosting* comme *Adaboost*.

Pour rappel, une forêt aléatoire consiste à créer m arbres. Les données d'entrée de ces m arbres sont des données tirées aléatoirement et avec remise du jeu de données d'entraînement. De plus, un sous-ensemble des variables explicatives est choisi de manière aléatoire pour chaque arbre afin de créer les frontières de décision. La décision finale prise par le modèle est la décision majoritaire prise par l'ensemble des arbres de décision formant la forêt.

Les méthodes de *boosting*, quant à elles, pondèrent chaque arbre de décision en attribuant un poids plus fort aux arbres ayant le plus de bonnes prédictions. De plus, un poids plus important peut être donné aux valeurs difficiles à prédire, c'est notamment le cas dans le modèle *Adaboost*.

4.2.4 Utilisation de réseaux de neurones

Ces dernières années, l'utilisation des réseaux de neurones est devenue de plus en plus populaire et de plus en plus fréquente. Le domaine de la détection d'anomalies ne fait pas exception. Nous avons par exemple énoncé dans la section 4.1.3 l'utilisation des *LSTM* pour prédire une valeur future de la série temporelle.

L'article 8 met en avant l'utilisation d'un réseau de neurones de convolution (*CNN* pour *Convolutional Neural Network*) pour classifier une sous-série temporelle comme anormale ou non.

Les réseaux de neurones de convolution sont généralement utilisés pour la reconnaissance d'images (dire si une image contient un chat ou non) et la détection d'objets dans des images ou vidéos. Pour utiliser ce type de réseaux de neurones dans le problème de la détection d'anomalies, l'article propose la transformation des séries en images sous niveaux de gris. Une fois cette

transformation effectuée, il est possible d'utiliser cette image en entrée du réseau de neurones afin de déterminer si cette série contient une anomalie ou non. La transformation effectuée dans l'article 8 est présentée figure 4.3.



FIGURE 4.3 – Transformation d'une série temporelle afin d'utiliser un *CNN*

Néanmoins, comme mentionné précédemment, nous n'allons pas utiliser de méthodes de *Deep Learning* basées sur l'utilisation des réseaux de neurones. Nous ne détaillerons donc pas le fonctionnement de ce type de réseaux de neurones.

4.2.5 Pré-traitements utiles pour les méthodes supervisées

Les méthodes présentées ci-dessus peuvent être utilisées directement sur les valeurs d'une série temporelle, elles nécessitent en revanche la présence d'un label indiquant si chaque valeur est anormale ou non.

Néanmoins, il est également possible de faire ce qui est appelé de la *feature extraction*. Cette méthode consiste à créer de nouvelles variables pouvant décrire la série temporelle ou une partie de la série temporelle. Ces variables descriptives peuvent être par exemple la valeur maximale ou minimale prise par la série sur la période considérée, la moyenne ou encore l'écart type. Certaines de ces *features* pouvant expliquer la série sont présentées dans l'article 9.

Chapitre 5

Implémentation d'une méthode de détection d'anomalies sur des séries temporelles en rapport avec la *smart city*

Dans ce chapitre nous allons maintenant mettre en place une méthode de détection d'anomalies en rapport avec les méthodes présentées dans le chapitre 4.

5.1 Choix des données

Comme mentionné dans l'introduction, le principe de cette TX était d'utiliser les données fournies par le consortium Maille'Immo de Lille. Particulièrement, les données devaient être fournies par l'entreprise ISTA. Néanmoins, l'entreprise n'a pas été en mesure de nous fournir les données nécessaires à temps. Nous avons donc dû chercher des données en *open data* afin de pouvoir appliquer une des méthodes décrites précédemment.

Après plusieurs recherches, nous avons décidé d'utiliser le jeu de données *leakDB* (article 10). Ce jeu de données a été créé en simulant les distributions en eau dans la ville d'Hanoï au Vitenam dont voici le réseau de distribution (figure 5.1) :

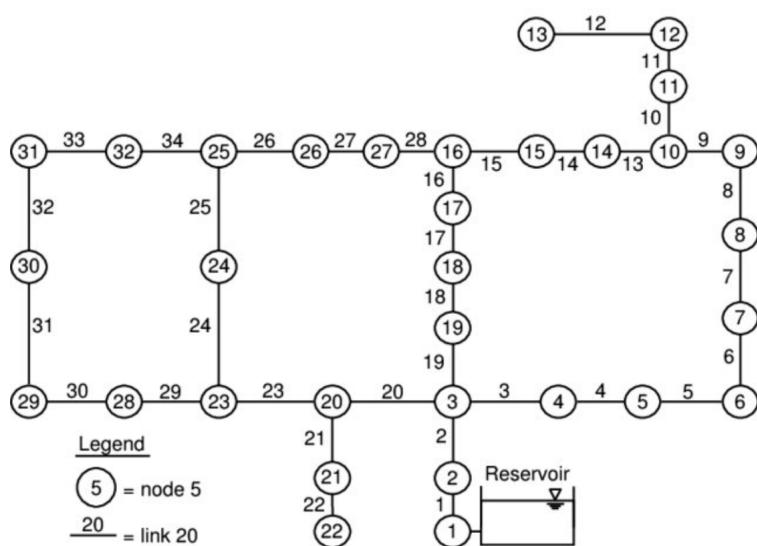


FIGURE 5.1 – Réseau de distribution de la ville d'Hanoï

Ce jeu de données contient 500 différents scénarios présentant pour certains des anomalies à différents noeuds du réseau.

Un scénario contient les pressions relevées par un capteur de pression au niveau du noeud, ainsi que les flux d'eau dans les liens entre les noeuds et les consommations d'eau à chaque noeud. Les valeurs sont relevées toutes les 30min pendant une année entière. Les données de *leakDB* sont labellisées.

Afin de simplifier l'étude, nous ne considérerons qu'un seul noeud et un seul lien : le noeud 21 et le lien 21. Nous avons donc cherché les scénarios présentant une fuite au noeud ou lien 21. Pour cette étude, nous allons donc utiliser le scénario 1 ne présentant aucune anomalie, ainsi que le scénario 5 et le scénario 78 présentant tous deux une anomalie au noeud ou lien 21. Pour les scénarios utilisés, nous n'allons considérer dans cette étude que les pressions et les flux d'eau.

5.2 Analyse exploratoire des données

Afin de déterminer la méthode que nous allons employer, nous effectuons une analyse exploratoire de nos données afin de mieux les appréhender.

Nous avons premièrement décidé de traiter les pressions et les flux d'eau séparément. Nous allons donc construire 2 modèles *uni-variés*.

Voici les distributions des pressions ainsi que des flux d'eau, en fonction du scénario considéré (figure 5.2) :

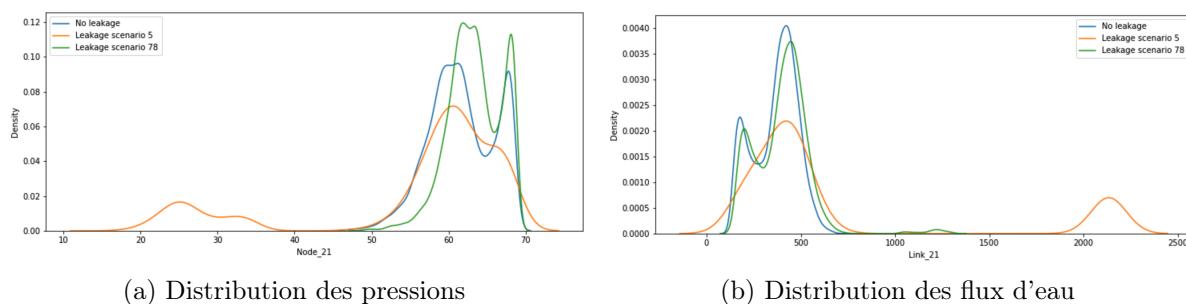


FIGURE 5.2 – Distribution des pressions et des flux d'eau en fonction du scénario considéré

Ces distributions nous montrent premièrement que les données ne sont pas distribuées selon une loi gaussienne. La méthode du *Z-score* ne peut donc être envisagée pour détecter les anomalies. Selon ces distributions, il semblerait plutôt simple de détecter les anomalies dans le scénario 5, en vu de la distribution hétérogène des données (il semble y avoir une séparation très nette entre deux groupes de valeurs). En revanche, détecter les anomalies dans le scénario 78 semble être un peu plus compliqué, car nous ne pouvons visualiser ces anomalies en regardant simplement la distributions des données : la distribution du scénario 78 semble être relativement similaire à la distribution du scénario 1 sans anomalies. Nous pouvons formuler l'hypothèse suivante : les anomalies du scénario 5 sont des anomalies ponctuelles et les anomalies du scénario 78 sont des anomalies contextuelles et/ou collectives.

Pour vérifier ces hypothèses, nous pouvons afficher les séries temporelles. Les figure 5.3 et 5.4 montrent les valeurs des pressions et des flux d'eau sur une période de 14 jours pour les trois scénarios (scénario 1 : du 1 janvier 2017 00 :00 au 13 janvier 2017 23 :30 ; scénario 5 : du 8 mars 2017 00 :00 au 21 mars 2017 23 :30 ; scénario 78 : du 14 octobre 2017 00 :00 au 27 octobre 2017 23 :30).

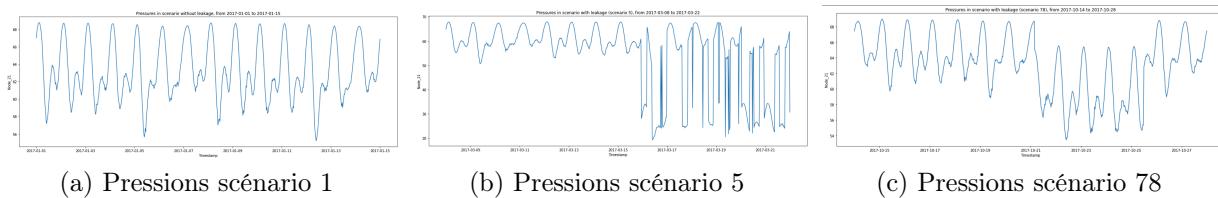


FIGURE 5.3 – Visualisation des pressions en fonction du scénario

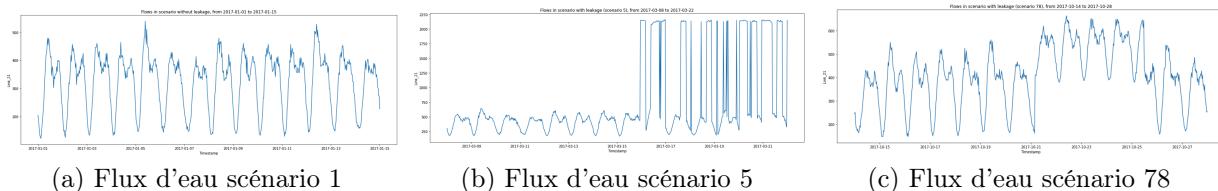


FIGURE 5.4 – Visualisation des flux d'eau en fonction du scénario

Ces différentes visualisations nous permettent premièrement de confirmer l'hypothèse exprimée ci-dessus, à savoir que les anomalies du scénario 5 semblent être des anomalies ponctuelles (visibles par ces grands "pics" de valeurs) tandis que les anomalies du scénario 78 semblent plutôt être des anomalies contextuelles (au vu de la distribution plus homogène des données et semblables à la distribution des données du scénario 1 sans anomalies) et également collectives : nous remarquons que le motif de la série temporelle semble être identique, mais avec une différence de $\pm \epsilon$.

Enfin, ces visualisations nous montrent que les séries temporelles présentent une saisonnalité qui semble être journalière. Cette information est importante pour la méthode de détection que nous allons tenter de mettre en place.

5.3 Choix de la méthode de détection

Nous avons fait le choix de considérer deux séries temporelles *uni-variées*. Comme montré section 5.2, nous devons considérer une méthode permettant de détecter à la fois les anomalies ponctuelles, contextuelles et collectives. Notre choix s'est donc tourné vers un modèle prédictif (plus particulièrement le modèle *SARIMA* car nos séries sont saisonnières). Pour rappel, un modèle prédictif vise à prédire la valeur future, puis la comparer à un intervalle de confiance. Si cette valeur prédite est hors de l'intervalle de confiance, alors nous considérons cette valeur comme anormale. Même si la prédiction d'une valeur à l'aide d'un modèle prédictif tel que *SARIMA* peut être plus ou moins compliquée, la détermination d'un intervalle de confiance permettant à la fois de bien détecter les anomalies et en même temps de ne pas considérer trop de points normaux comme des anomalies n'est pas une tâche aisée. Nous choisissons donc de modifier l'utilisation de notre modèle prédictif : celui-ci ne sera pas utiliser pour détecter des anomalies mais pour prédire la valeur qui serait normalement attendue. Nous combinons ensuite ce modèle prédictif à une méthode d'apprentissage supervisée en faisant de la *feature extraction* pour détecter les anomalies. Voici la modèle final que nous allons utiliser :

- Les valeurs des séries temporelles seront agrégées toutes les 3 heures et les variables suivantes seront créées :
 - mean* : moyenne empirique des valeurs au cours des 3 heures
 - std* : écart-type à la moyenne des valeurs au cours des 3 heures
 - median* : valeur médiane prise au cours des 3 heures

- *min* : valeur minimale prise au cours des 3 heures
 - *max* : valeur maximale prise au cours des 3 heures
2. Un modèle *SARIMA* entraîné sur le scénario 1 sans anomalies servira à prédire la valeur moyenne de la série prise au cours des 3 prochaines heures
 3. Une fois cette valeur prédite, elle est comparée avec la valeur moyenne réelle de la série et nous enregistrons la différence dans une nouvelle variable
 4. Nous obtenons donc un jeu de données composé des variables suivantes : *mean*, *std*, *median*, *min*, *max*, *difference*
 5. Ce jeu de données sera donné en entrée d'un modèle de classification entraîné qui permettra de déterminer si cette aggrégation de 3 heures est considérée comme une anomalie ou non

La méthode utilisée est résumée figure 5.5.

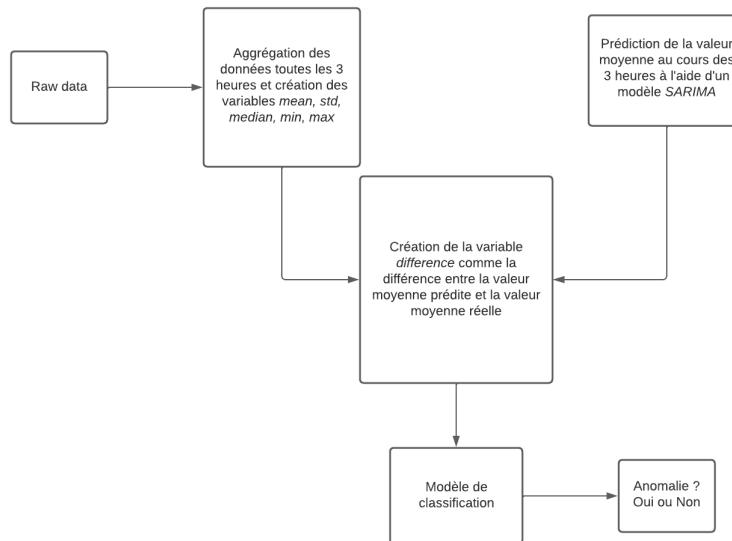


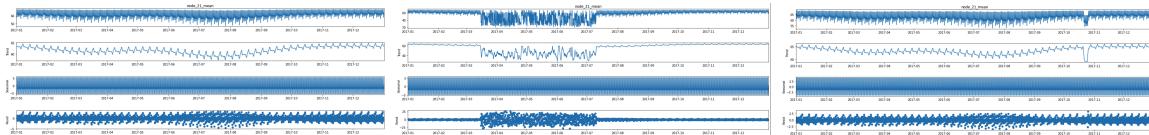
FIGURE 5.5 – Méthode de détection d'anomalies dans les séries temporelles

5.3.1 Justification du choix du modèle

En décomposant nos séries selon leur tendance, leur saisonnalité et leur bruit (voir figures 5.6 et 5.7) comme expliqué section 2.2 (décomposition additive), nous remarquons que l'utilisation d'un seuillage, ou bien d'une méthode statistique comme le *Z-score* sur les résidus, suffirait à détecter les valeurs anormales (en supposant que les résidus sont distribués selon une loi normale). Cette hypothèse peut être vérifiée en utilisant le test de *Shapiro-Wilk* si μ et σ^2 ne sont pas connus ou le test de *Kolmogorov-Smirnov* si on souhaite tester l'adéquation à une loi normale de moyenne et variance connues). Avec l'utilisation d'une méthode de seuillage (testée section 5.9) nous nous attendons à ce que certaines anomalies (notamment celles du scénario 78) ne soient pas détectées du fait que leurs valeurs sont anormales dans un contexte donné et non globalement au regard de la distribution de la série. Cependant nous cherchons ici à prédire la valeur normalement attendue à l'instant t , et non à détecter directement les anomalies à partir de la série temporelle brute. C'est pour cela que nous choisissons d'utiliser une méthode prédictive pour créer une variable indiquant la différence entre la valeur normalement espérée et la valeur réellement obtenue, afin de détecter des anomalies dont les valeurs sont anormales dans un contexte particulier.

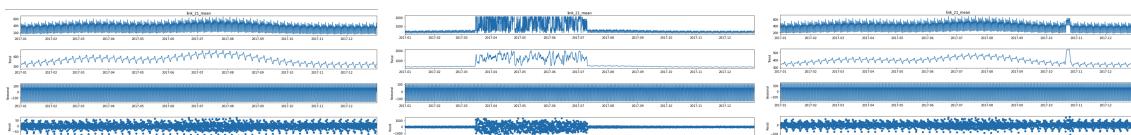
Pour pouvoir utiliser une méthode prédictive, nous devons nous assurer que nos séries ne présentent pas de bruit blanc. Les graphiques d'auto-corrélation (cf figure 5.11) montrent que nous

ne sommes pas en présence de bruit blanc (nous avons des valeurs significatives au seuil de 95% pour des décalages allant de $t - 1$ à $t - 7$).



(a) Décomposition additive des pressions scénario 1 (b) Décomposition additive des pressions scénario 5 (c) Décomposition additive des pressions scénario 78

FIGURE 5.6 – Décomposition additive des pressions en fonction du scénario

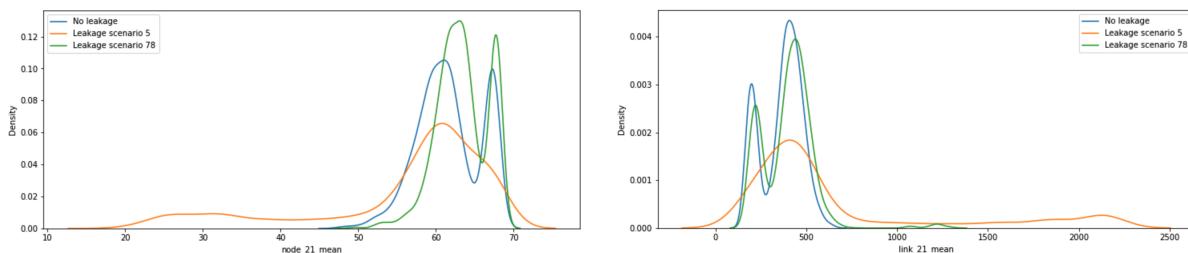


(a) Décomposition additive des flux scénario 1 (b) Décomposition additive des flux scénario 5 (c) Décomposition additive des flux scénario 78

FIGURE 5.7 – Décomposition additive des flux en fonction du scénario

5.4 Pré-traitement des données

Comme expliqué section 5.3, nous devons agrégger nos données toutes les 3 heures. Nous obtenons après aggrégation les distributions des valeurs moyennes suivantes (figure 5.8) :

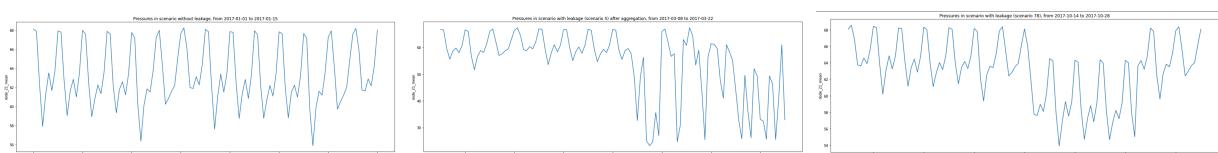


(a) Distribution des pressions agrégées

(b) Distribution des flux d'eau agrégés

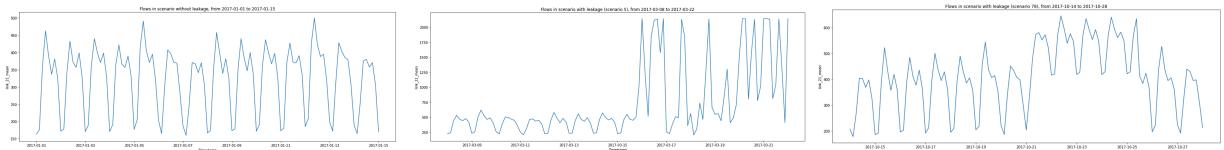
FIGURE 5.8 – Distribution des pressions et des flux d'eau agrégés en fonction du scénario considéré

Si nous cherchons à re-visualiser les pressions moyennes et flux d'eau moyens après agrégation, nous obtenons les visualisations suivantes (figures 5.9 et 5.10) :



(a) Pressions moyennes agrégées scénario 1 (b) Pressions moyennes agrégées scénario 5 (c) Pressions moyennes agrégées scénario 78

FIGURE 5.9 – Visualisation des pressions moyennes agrégées toutes en fonction du scénario



(a) Flux d'eau moyens agrégés scé- (b) Flux d'eau moyens agrégés scé- (c) Flux d'eau moyens agrégés scé-
nario 1 nario 5 nario 78

FIGURE 5.10 – Visualisation des flux d'eau moyens agrégés en fonction du scénario

De manière générale, nous remarquons que notre aggrégation a lissé nos données (elles semblent moins bruitées), n'a pas modifié drastiquement la distribution des données et n'a pas impacté la saisonnalité de nos séries temporelles.

5.5 Processus *SARIMA*

Notons $(1 - B)^d = \nabla^d$ et $(1 - B^s)^D = \nabla_s^D$ avec B l'opérateur de retard.
On dit que (X_t) est un processus $SARIMA(p, d, q)X(P, D, Q)_s$ si le processus

$$Y_t = \nabla_s^D \nabla^d X_t$$

est un processus $ARMA(p, q)$.

Autrement dit si :

$$\nabla_s^D \Gamma(B^s) \nabla^d \Phi(B) X_t = c + \Xi(B^s) \Theta(B) \epsilon_t$$

avec :

$$\Gamma(B) = 1 - \sum_{j=1}^P \gamma_j B^j, \quad \Phi(B) = 1 - \sum_{j=1}^p \phi_j B^j, \quad \Xi(B) = 1 - \sum_{j=1}^Q \chi_j B^j, \quad \Theta(B) = 1 - \sum_{j=1}^q \theta_j B^j$$

alors (X_t) est un processus $SARIMA(p, d, q)X(P, D, Q)_s$. Les polynômes $\Gamma(B)$, $\Phi(B)$, $\Xi(B)$, $\Theta(B)$ satisfont les mêmes hypothèses que les polynômes du processus $ARMA(p, q)$.

Pour rappel, un processus $ARMA(p, q)$ vérifie

$$X_t - \sum_{j=1}^p \phi_j X_{t-j} = c + \epsilon_t - \sum_{h=1}^q \theta_h \epsilon_{t-h}$$

où

- $(\epsilon_t)_{t \in \mathbf{Z}}$ est un bruit blanc de variance σ^2
- ϕ_p et θ_q sont non nuls
- les polynômes $\Phi(B) = 1 - \sum_{j=1}^p \phi_j B^j$ et $\Theta(B) = 1 - \sum_{h=1}^q \theta_h B^h$ ont toutes leurs racines de module différent de 1 (B étant l'opérateur de retard qui à une suite réelle $(x_t)_{t \in \mathbf{Z}}$ associe la suite $(x_{t-1})_{t \in \mathbf{Z}}$)

Nous avons donc pour le modèle $SARIMA(p, d, q)X(P, D, Q)_s$ les hypothèses suivantes :

- $(\epsilon_t)_{t \in \mathbf{Z}}$ est un bruit blanc de variance σ^2
- ϕ_j , θ_j , χ_j , γ_j non nuls $\forall j$
- les polynômes $\Gamma(B)$, $\Phi(B)$, $\Xi(B)$, $\Theta(B)$ n'ont pas de racines communes et ont toutes leurs racines de module différent de 1

Contrairement aux modèles de lissages exponentiels (simple, double, triple) qui visent à décrire la série temporelle par sa tendance et sa saisonnalité, un modèle $ARMA(p, q)$ décrit la série

temporelle par ses auto-corrélations.

Il est préférable d'utiliser une décomposition additive de la série temporelle pour utiliser un processus $ARMA(p, q)$. Certains opérateurs permettent de passer d'un modèle multiplicatif à un modèle additif comme le logarithme par exemple : $Y_t = \ln(X_t)$. Un processus $ARMA(p, q)$ nécessite que la série temporelle soit stationnaire.

Les opérateurs ∇^d et ∇_s^D permettent de différencier la série en tendance et en saisonnalité respectivement. Ils permettent donc de rendre une série saisonnière et tendancielle stationnaire en absorbant respectivement la tendance m_t et la saisonnalité s .

De manière générale, un processus $SARIMA(p, d, q)X(P, D, Q)_s$ modélise bien des séries temporelles présentant une période de longueur D et une tendance polynomiale de degré d .

Pour estimer les 7 paramètres d'un processus $SARIMA(p, d, q)X(P, D, Q)_s$ nous devons procéder de la manière suivante :

1. Identifier la saisonnalité s (auto-corrélogramme, visualisation de la série)
2. Différencier la série en saisonnalité (estimer D) puis en tendance (estimer d)
3. Déterminer des ordres p et q plausibles à l'aide de l'autocorrélogramme partiel et l'autocorrélogramme. Les ordres P et Q en regardant les ordre multiples de s de ces autocorrélogrammes.
4. Estimer le modèle avec les paramètres déterminés

5.6 Entraînement des modèles prédictifs (modèles *SARIMA*)

Une fois nos séries temporelles agrégées et nos nouvelles variables créées, nous pouvons entraîner notre modèle *SARIMA* sur le scénario 1 sans anomalies. Deux modèles seront entraînés : un modèle sur les pressions moyennes et un modèle sur les flux d'eau moyens.

Comme expliqué section 5.5, nous devons déterminer les valeurs s , D , d , p , q , P et Q .

5.6.1 Détermination des hyper-paramètres des modèles

détermination de s :

Nos visualisations nous ont montré que les séries temporelles semblent avoir une saisonnalité journalière. Un jour étant d'une durée de 24 heures et nos données agrégées étant toutes les 3 heures, nous pouvons en déduire que l'hyper-paramètre s est égal à 8 pour les deux modèles *SARIMA*. Cette hypothèse est vérifiée lorsque nous regardons les auto-corrélogrammes figure 5.11 (pour les pressions et pour les flux) : nous voyons des pics significatifs tous les 8 décalages (de plus, nos séries temporelles semblent présenter un cycle car nous avons un pic plus significatif tous les 56 décalages, soit $8 * 7$, soit un cycle de une semaine. Nous n'incluerons pas la prise en compte de ce cycle dans notre analyse) :

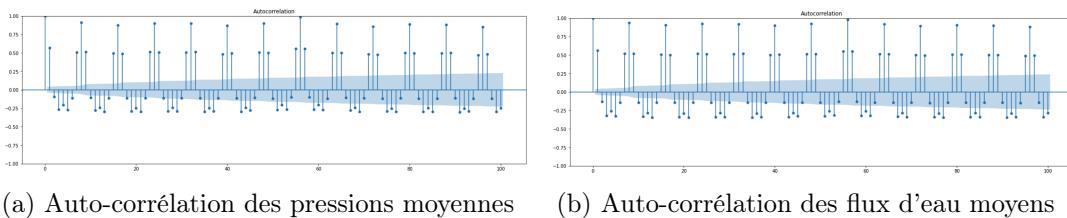


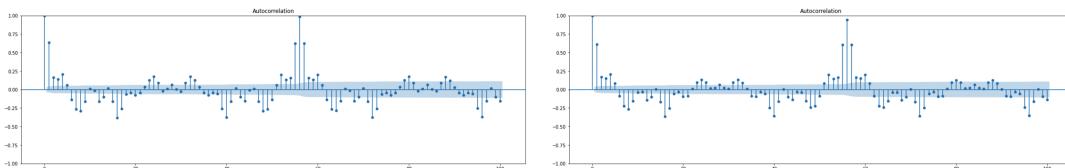
FIGURE 5.11 – Graphiques d'auto-corrélation pour déterminer l'hyper-paramètre s

Nous pouvons donc déterminer

$$s = 8$$

détermination de D :

Nous différencions donc nos séries avec l'opérateur ∇_8^1 et obtenons les auto-corrélogrammes suivants 5.12 :



(a) Auto-corrélation des pressions moyennes différencierées (b) Auto-corrélation des flux d'eau moyens différencierés

FIGURE 5.12 – Graphiques d'auto-corrélation pour déterminer l'hyper-paramètre D

Ces auto-corrélogrammes semblent être des auto-corrélogrammes simples (décroissance rapide vers 0) bien que présentant quelques oscillations (nous retrouvons d'ailleurs notre cycle énoncé plus tôt). Nous pouvons donc déterminer

$$D = 1$$

A ce stade nous avons le modèle $(1 - B^8)X_t$

détermination de d :

L'hyper-paramètre d peut être déterminer à l'aide du test augmenté de Dickey-Fuller. Nous appliquons ce test sur une série temporelle. Si le résultat du test montre que cette série n'est pas stationnaire, alors nous la différencions, puis nous ré-appliquons le test augmenté de Dickey-Fuller. Nous appliquons cette procédure jusqu'à ce que le résultat du test statistique affirme que la série différenciée d fois est stationnaire (d étant le nombre de fois où nous avons différencié la série).

Nous appliquons cette procédure sur nos 2 séries temporelles (à savoir les pressions moyennes agrégées toutes les trois 3 du scénario 1 ainsi que les flux d'eau moyens du même scénario) et obtenons les résultats suivants (figure 5.13) :

```
Pressure node 21 difference
1. ADF : -12.949001592217696
2. P-Value : 3.4076131135327646e-24
3. Num Of Lags : 28
4. Num Of Observations Used For ADF Regression and Critical Values Calculation : 2883
5. Critical Values :
 1% : -3.432620250421088
 5% : -2.8625430431596545
 10% : -2.567303948779845
the serie is stationary
```

(a) Test ADF sur les pressions moyennes

```
Flow link 21 difference
1. ADF : -12.909983024778054
2. P-Value : 4.076265688543343e-24
3. Num Of Lags : 28
4. Num Of Observations Used For ADF Regression and Critical Values Calculation : 2883
5. Critical Values :
 1% : -3.432620250421088
 5% : -2.8625430431586545
 10% : -2.567303948779845
the serie is stationary
```

(b) Test ADF sur les flux d'eau moyens

FIGURE 5.13 – Résultats du test ADF sur les deux séries temporelles non différencierées

Selon les résultats du test augmenté de Dickey-Fuller sur nos deux séries temporelles, elles sont stationnaires (ce qui peut être surprenant étant donné la saisonnalité de nos séries). Nous avons donc

$$d = 0$$

détermination de p et P :

Pour déterminer les hyper-paramètres p et P , nous devons utiliser les graphiques d'auto-corrélation partielle. Ces graphiques (présenté figure 5.14) nous montrent que pour la partie auto-régressive, le décalage avec le plus de significativité est le décalage $h = 1$ pour les deux

séries. Pour la partie saisonnière, le décalage (multiple de s) ayant le plus de significativité est le décalage 46 ($8 * 6$) pour les deux séries.

$$p = 1, \quad P = 6$$

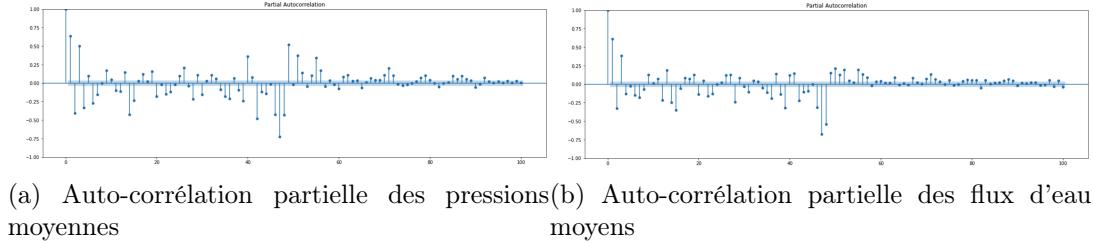


FIGURE 5.14 – Graphiques d’auto-corrélation partielle pour déterminer les hyper-paramètres p et P

détermination de q et Q :

Enfin, nous devons également déterminer les hyper-paramètres q et Q pour avoir nos modèles *SARIMA*. La procédure est très similaire à la détermination des hyper-paramètres p et P à la différence que nous utilisons les graphiques d’auto-corrélation et non les graphiques d’auto-corrélation partielle. Selon ces graphiques (figure 5.12), le décalage ayant le plus de significativité pour la partie de moyenne mobile est $h = 1$ pour les deux séries. La partie saisonnière présentant le plus de significativité est de 7 (décalage $h = 56 = 8 * 7$), ce qui semble logique en rapport avec le cycle présumé de nos données. Nous avons donc

$$q = 1, \quad Q = 7$$

Finalement, nous avons les deux modèles *SARIMA* suivants :

- $SARIMA_{pressions}(1, 0, 1)X(6, 1, 7, 8) = \nabla_8^1 \Gamma(B^8)\Phi(B)X_t = c + \Xi(B^8)\Theta(B)\epsilon_t$
- $SARIMA_{flux}(1, 0, 1)X(6, 1, 7, 8) = \nabla_8^1 \Gamma(B^8)\Phi(B)X_t = c + \Xi(B^8)\Theta(B)\epsilon_t$

5.6.2 Entrainement des modèles et visualisations

Nous pouvons désormais entraîner nos 2 modèles sur les 2 séries temporelles que nous avons choisies. Après entraînement, nous obtenons ces résultats sur les pressions moyennes et les flux d’eau moyens du scénario 1 (figure 5.15) :

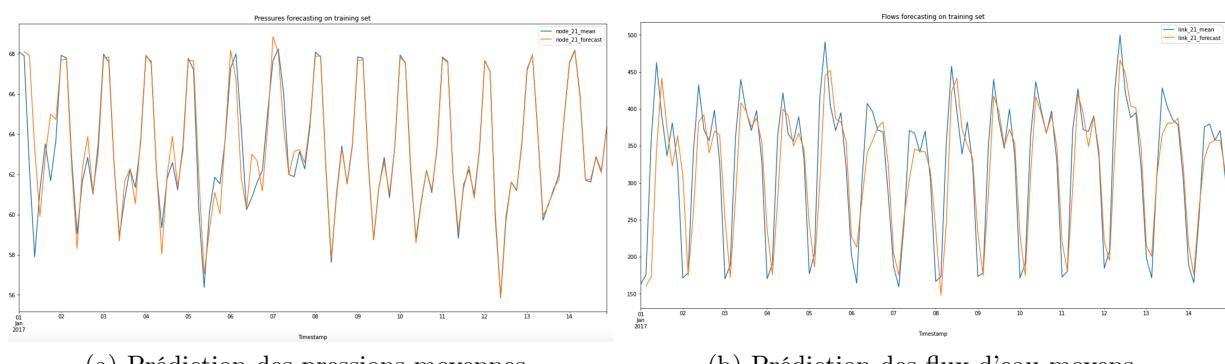


FIGURE 5.15 – Résultats de la prédiction des modèles *SARIMA* sur le scénario sans anomalies

La prédiction semble être plutôt bonne. Pour avoir une information plus précise sur les performances de nos modèles, nous allons utiliser la métrique de pourcentage moyen d'erreur absolue (*mean absolute percentage error (MAPE)* en anglais) :

$$MAPE = 100 * \frac{1}{n} \sum_{i=1}^n \frac{|x_i - \hat{x}_i|}{x_i}$$

Nous avons $MAPE = 0.21$ pour les pressions et $MAPE = 4,5$ pour les flux. Ceci nous montre que les performances de nos modèles sont bonnes (ou du moins acceptables pour notre problème), et que nous semblons prédire avec plus de précision les pressions que les flux. Essayons maintenant de voir ce que donne cette prédiction sur les scénarios avec des anomalies (figures 5.16 et 5.17) :

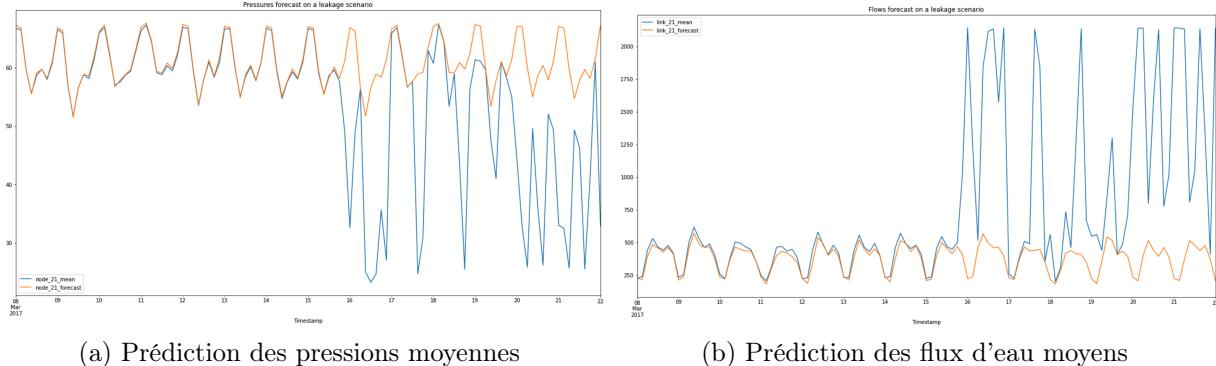


FIGURE 5.16 – Résultats de la prédiction des modèles *SARIMA* sur le scénario 5 avec anomalies

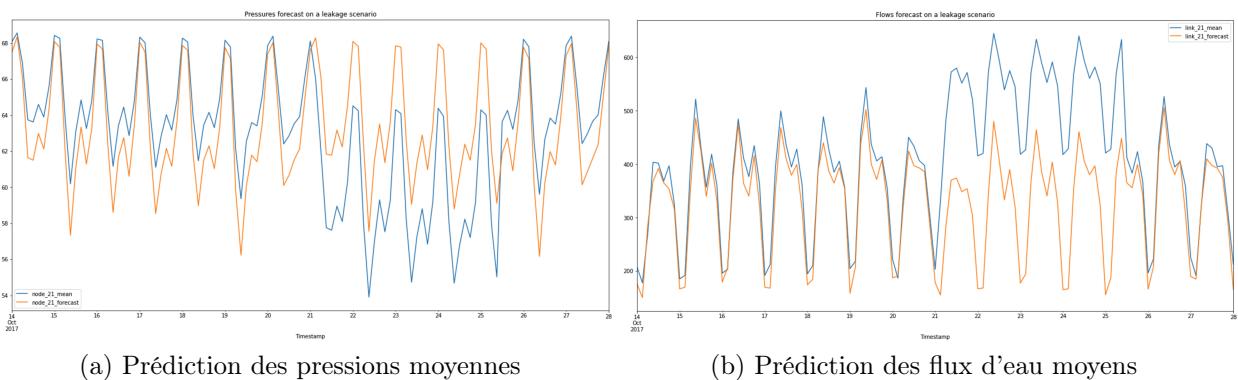


FIGURE 5.17 – Résultats de la prédiction des modèles *SARIMA* sur le scénario 78 avec anomalies

Nous remarquons que la différence entre la valeur prédictive par le modèle prédictif et la valeur réelle augmente plus ou moins fortement lorsqu'une anomalie apparaît. Cette remarque se confirme lorsque nous nous intéressons au *MAPE* de la prédiction sur les scénarios avec anomalies de nos modèles *SARIMA*(1, 0, 1)X(6, 1, 7)₈ : pour le scénario 5, les *MAPE* sont respectivement de 20% pour les pressions et de 23,4% pour les flux. Pour le scénario 78, les *MAPE* sont de 3% pour les pressions et de 8% pour les flux.

Il semble, selon les représentations graphiques, que les modèles prédictifs soient un peu moins performants sur le scénario 78 (l'erreur de prédiction semble être légèrement plus importante que sur les deux autres scénarios de manière générale), mais lorsque l'on regarde les différentes

MAPE, nous remarquons que les erreurs globales des modèles prédictifs sur les valeurs du scénario 78 n'ont pas fortement augmentées, ce qui confirme l'hypothèse d'anomalies collectives et contextuelles formulée précédemment.

Une fois nos modèles prédictifs entraînés, nous pouvons créer la variable *difference* (comme la différence entre la valeur prédite par le modèle et la valeur réelle) puis entraîner un modèle de classification permettant de déterminer si une aggrégation de 3 heures est anormale ou non.

5.7 Entraînement d'un modèle de classification

Nos données d'entrée du modèle de classification sont composées des variables suivantes : *valeur réelle moyenne, écart type, mediane, min, max, difference*. Afin de créer un seul jeu de données pour les pressions et un seul jeu de données pour les flux d'eau, nous concatérons nos 3 scénarios et nous modifions les années d'enregistrement des données. Nous avons donc un jeu de données des pressions de 2017 à 2020, ainsi qu'un jeu de données des flux d'eau de 2017 à 2020. Cette concaténation sera utile pour comparer les performances de notre modèle avec des modèles naïfs (seuillage et *Z-score* sur le bruit de nos séries temporelles) sur l'ensemble des valeurs des 3 scénarios que nous avons.

A partir de cette concaténation, nous allons créer deux ensemble de données :

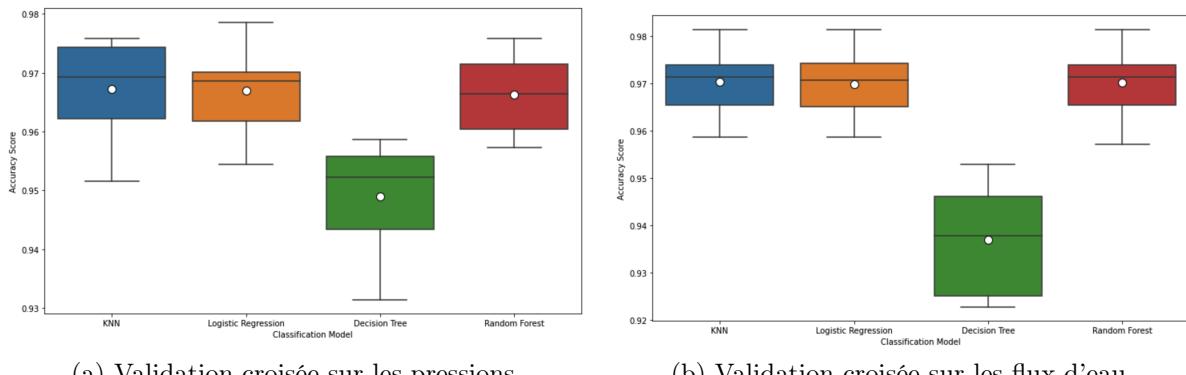
- Ensemble d'entraînement : servira à entraîner nos différents modèles (80% du jeu de données initial)
- Ensemble de validation : servira à tester les performances de nos différents modèles (20% du jeu de données initial)

5.7.1 Choix du modèle

Pour choisir le modèle, nous séparons nos jeux de données en deux sous-ensembles différents : un ensemble d'entraînement et un ensemble de validation. Nous effectuons une validation croisée de 10 plis sur les ensembles d'entraînement (des pressions et des flux d'eau) permettant de comparer les performances moyennes de différents modèles de classification. Nous choisissons de comparer les modèles suivants :

- K plus proches voisins ($k = 5$)
- Régression logistique (*optimiseur = logit*)
- Arbre de décision (*criterion = Gini*)
- Forêt aléatoire ($m = 100$, *criterion = Gini*)

Les résultats des validations croisées sont montrés figure 5.18



(a) Validation croisée sur les pressions

(b) Validation croisée sur les flux d'eau

FIGURE 5.18 – Résultats des validations croisées (les points blancs représentent les scores moyens de chaque validation)

Ces deux graphiques ne nous permettent pas d'affirmer qu'un modèle est plus performant que tous les autres (mise à part l'arbre de décision classique qui semble moins performant que les autres modèles). Nous pourrions donc choisir un modèle de manière arbitraire (sachant que chaque modèle est adapté aux données quantitatives que nous avons). Pour vérifier cette hypothèse, nous pouvons appliquer un test *ANOVA* pour déterminer si les performances moyennes de nos modèles sont significativement différentes. Pour rappel l'hypothèse H_0 du test *ANOVA* est : les moyennes ne sont pas différentes. Nous appliquons ce test à nos données et obtenons la p_{value} de $1 * 10^{-5}$. Nous pouvons donc rejeter l'hypothèse H_0 et affirmer que les performances des modèles ne sont pas égales. Néanmoins, le test *ANOVA* ne permet pas de déterminer quel modèle a des performances significativement différentes. Nous appliquons dans cet objectif le test de *Tuckey (HSD)* qui est utilisé pour déterminer les différences significatives entre les moyennes de groupes dans une analyse de variance. Ce test est effectué sur chaque paire de modèles et l'hypothèse H_0 est que les moyennes ne sont pas significativement différentes. Nous obtenons les résultats suivants :

TABLE 5.1 – Résultats du test de *Tuckey (HSD)* sur les performances des différents modèles

Couple de modèles	p_value
(<i>KNN</i> , <i>Logistic regression</i>)	0.9
(<i>KNN</i> , <i>Decision tree</i>)	$1 * 10^{-3}$
(<i>KNN</i> , <i>Random forest</i>)	0.9
(<i>Logistic regression</i> , <i>Decision tree</i>)	$1 * 10^{-3}$
(<i>Logistic regression</i> , <i>Random forest</i>)	0.9
(<i>Random forest</i> , <i>Decision tree</i>)	$1 * 10^{-3}$

Nous remarquons qu'à part les couples de modèles dont l'un des membres est l'arbre de décision, les p_{value} montrent que nous ne pouvons pas rejeter l'hypothèse H_0 et donc que les performances moyennes ne sont pas différentes. Nous choisissons donc d'utiliser la forêt aléatoire comme modèle de classification (car il semble être le modèle qui présente le moins de variance entre les 10 plis de la validation croisée) pour déterminer si une aggrégation de valeur est anormale ou non.

5.7.2 Explications de la forêt aléatoire

La forêt aléatoire (*Random forest* en anglais) est une méthode ensembliste s'appuyant sur l'utilisation d'arbres de décisions (*Decision tree*). Un arbre de décision est un modèle d'apprentissage supervisé se basant sur des séparations successives de l'ensemble d'apprentissage pour créer une partition de cet ensemble en régions *homogènes* au sens d'un critère d'impureté. Le plus souvent, le critère de *Gini* est utilisé (notamment dans l'algorithme *CART*)

$$G(P) = \sum_{k=1}^g p_k(1 - p_k)$$

g étant le nombre de classes que nous avons et P le vecteur comprenant (p_1, \dots, p_g) les proportions de chaque classe dans l'ensemble considéré. D'autres algorithmes utilisent un autre critère d'impureté : *l'entropie E*

$$E(P) = - \sum_{k=1}^g p_k \ln(p_k)$$

La forêt aléatoire utilise un nombre m d'arbres de décision (dans notre cas nous avons $m = 100$). Les données d'entrée des m arbres sont créées avec la méthode du *bagging* : n observations sont

sélectionnées par un tirage avec remise dans l'ensemble d'apprentissage (n étant le nombre d'observations total de notre ensemble d'apprentissage) afin de créer m jeu de données d'entrée différents. Les m arbres sont ensuite créés à partir de ces m jeu de données. La décision finale que la forêt aléatoire prend se fait au "vote majoritaire" : la classe ayant la plus grande proportion dans les décisions prises par les m arbres.

La forêt aléatoire vise finalement à diminuer la variance dans les décisions prises par m arbres : les arbres sont très sensibles aux variations dans les données d'origine. Construire m jeux de données différents à partir du jeu de données d'apprentissage augmente les variations, le choix final de la forêt aléatoire étant la classe dont la proportion est la plus grande dans l'ensemble des décisions prises permet donc de diminuer l'influence des variations dans les données, et donc la variance, sous l'hypothèse que les m arbres de décision sont décorrélés.

De plus, la forêt aléatoire permet de diminuer l'influence d'observations proches de la frontière de décision (celles-ci étant minoritaires dans le jeu de données, la probabilité que ces observations soient majoritaires dans les m jeux de données créer grâce au *bagging* est faible, leur influence est donc diminuée), dans l'objectif de diminuer l'*overfitting*.

5.7.3 Analyse des métriques d'évaluations des modèles

Pour connaître les performances de nos modèles, nous allons nous intéresser aux métriques suivantes : *precision*, *recall* et *F1 score*. Pourquoi nous intéresser à ces métriques et non pas seulement à l'*accuracy* ?

L'*accuracy*, dont voici la formule :

$$\text{accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

autrement dit, le ratio entre les bonnes prédictions et le total des prédictions, nous donne une information globale de notre modèle à bien prédire. Cependant, la grande majorité des valeurs des jeux de données sont considérées comme normales. Si nous considérons que 95% de nos valeurs sont normales et que notre modèle prédit tout le temps qu'une valeur est normale, nous avons alors une *accuracy* de 95%, ce qui est très bon, mais notre modèle n'est pas bon dans la détection des points anormaux (ce qui est l'objectif de notre étude). Rappelons les formules de la *precision*, du *recall* et du *F1 score* :

$$\text{precision} = \frac{TP}{TP + FP} \quad \text{recall} = \frac{TP}{TP + FN} \quad F1 \text{ score} = 2 * \frac{\text{precision} * \text{recall}}{\text{precision} + \text{recall}}$$

La *precision* nous donne une indication sur notre modèle à bien prédire les anomalies, c'est à dire que si nous avons une *precision* de 0.9, alors si notre modèle prédit une anomalie, il y a 90% de chances que ce soit réellement une anomalie.

Le *recall* nous donne quant à lui une indication sur la capacité de notre modèle à détecter les anomalies, c'est à dire que si notre modèle a un *recall* de 0.6, alors 60% des anomalies ont été détectées.

Enfin, le *F1 score* (moyenne harmonique des deux métriques précédentes) nous donne finalement une indication sur notre modèle à détecter toutes les anomalies tout en minimisant les fausses anomalies détectées.

Nous pouvons calculer ces métriques grâce aux matrices de confusion des modèles sur les données de validation (figure 5.19) :

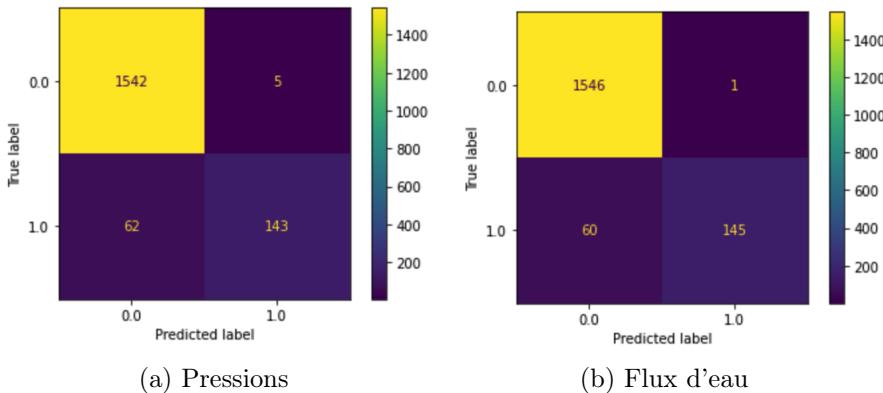


FIGURE 5.19 – Matrices de confusion obtenues sur les données de validation après entraînement des modèles

Pour la série temporelle sur les pressions, nous avons une *precision* de 0.97, un *recall* de 0.70 et un *F1 score* de 0.81.

Pour la série temporelle sur les flux d'eau, nous avons une *precision* de 0.99, un *recall* de 0.71 et un *F1 score* de 0.83.

De manière générale, nos modèles semblent plutôt performant, mais environ 30% des anomalies présentes dans les données de validation n'ont pas été détectées. Ce pourcentage d'anomalies non détectées peut venir du fait que dans le scénario 5, des valeurs labellisées comme anormales sont très proches des valeurs normales (voir figure 5.20). Ce type de valeurs, que nous pouvons considérer comme mal labellisées, peut altérer les performances de nos modèles.

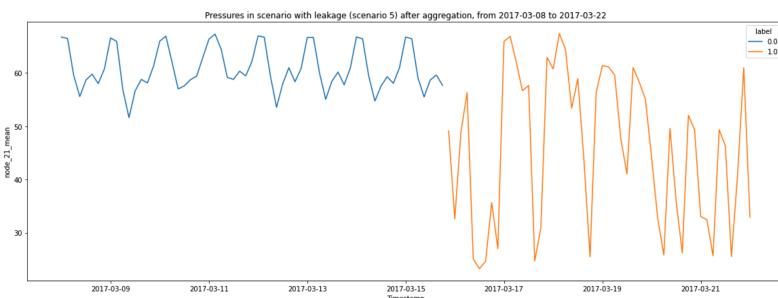


FIGURE 5.20 – Pressions du scénario 5 (en bleu les valeurs normales et en orange les valeurs anormales)

5.8 Comparaison avec des modèles sans la variable *difference*

La classe *RandomForestClassifier* de la librairie *sklearn* nous permet d'obtenir des informations sur l'importance des variables d'entrée de la forêt aléatoire. Ces indices d'importance sont déterminées à partir de la diminution moyenne du critère d'impureté (le critère de *Gini*), ce qui équivaut à calculer la contribution de chaque variable à la diminution du critère de *Gini* lors de l'entraînement. La figure 5.21 montre l'importance de chaque variable pour les données de pressions et de flux d'eau :

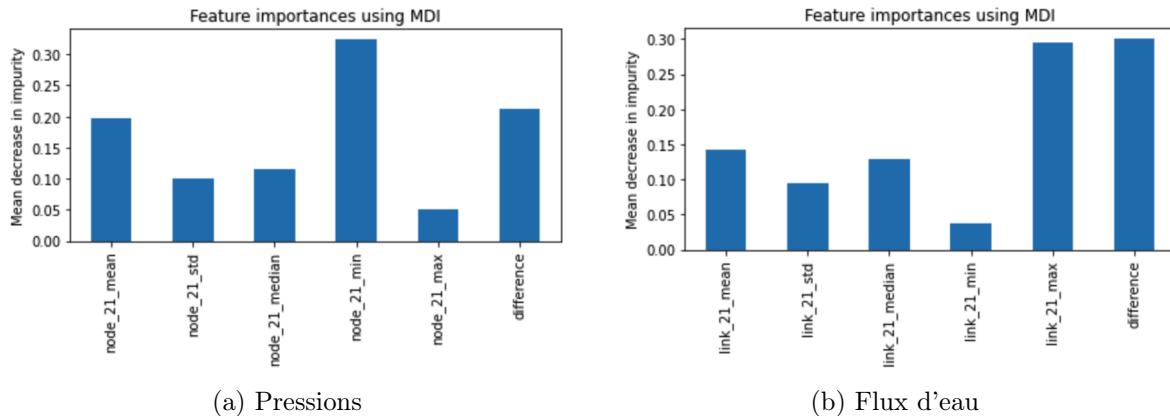


FIGURE 5.21 – Graphique montrant l’importance de chaque variable pour la détection d’anomalies

Nous remarquons que la variable *difference*, dans les deux modèles, semble avoir une importance significative. Nous pouvons donc nous attendre à ce que les 2 modèles aient de moins bonnes performances que les modèles présentés plus haut.

Nous entraînons donc ces deux nouveaux modèles sans la variable *difference* et obtenons les matrices de confusion suivantes (figure 5.22) :

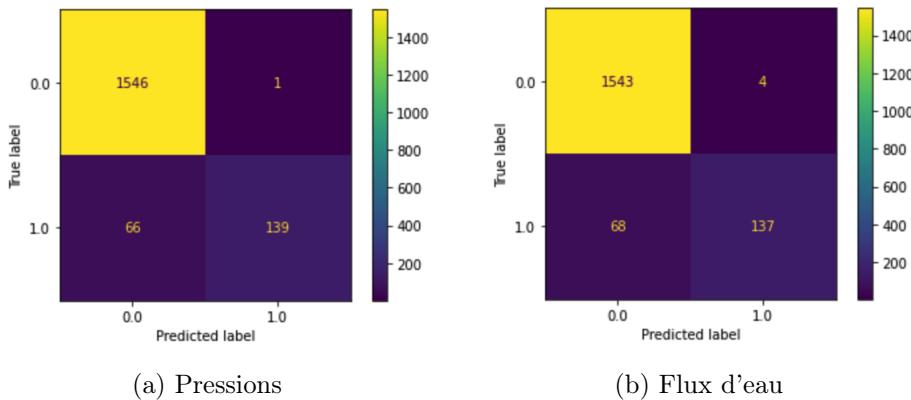


FIGURE 5.22 – Matrices de confusion obtenues sur les données de validation après entraînement des modèles sans la variable *difference*

Si nous nous intéressons aux métriques précédemment utilisées, nous remarquons que la *precision* du modèle basé sur les flux d'eau sans la variable *difference* a diminué (0.97 contre 0.99 auparavant), ainsi que le *recall* (0.68 contre 0.71). Le recall du modèle basé sur les pressions sans cette variable est de 0.68 (0.70 auparavant), mais la précision a augmentée (0.99 contre 0.97). L'utilisation de la variable *difference* nous permet donc de détecter plus d'anomalies si l'on considère les pressions à chaque noeud, au prix d'une très légère baisse de précision, et de détecter plus d'anomalies tout en augmentant la précision si l'on considère les flux d'eau.

5.9 Comparaison avec un modèle naïf

Dans cette section, nous allons comparer les performances de nos modèles avec les performances d'un modèle naïf : un seuillage (en utilisant la méthode d'estimation de noyau) sur les valeurs moyennes des agrégations.

5.9.1 Estimation de noyau

L'estimation de noyau (*kernel density estimation (KDE)* en anglais) est une méthode non-paramétrique dont l'objectif est d'estimer la densité de probabilité d'une variable aléatoire. Elle se base sur un échantillon d'une population et permet d'estimer la densité. Si x_1, \dots, x_N est un échantillon i.i.d d'une variable aléatoire X (dans notre cas la valeur moyenne des pressions et des flux sur une période de 3 heures), alors l'estimateur non-paramétrique par la méthode du noyau de la densité est :

$$\hat{f}_h(x) = \frac{1}{Nh} \sum_{i=1}^N K\left(\frac{x - x_i}{h}\right)$$

où K est un noyau et h est un paramètre qui régit le lissage (on l'appelle généralement fenêtre). Nous choisissons souvent K comme la densité d'une fonction gaussienne d'espérance μ nulle et de variance σ^2 unitaire :

$$K(x) = \frac{1}{2\sqrt{\pi}} e^{-\frac{1}{2}x^2}$$

5.9.2 Application

Nous allons considérer pour notre modèle un noyau gaussien. La distribution de notre jeu de données d'apprentissage est la suivante (figure 5.23) :

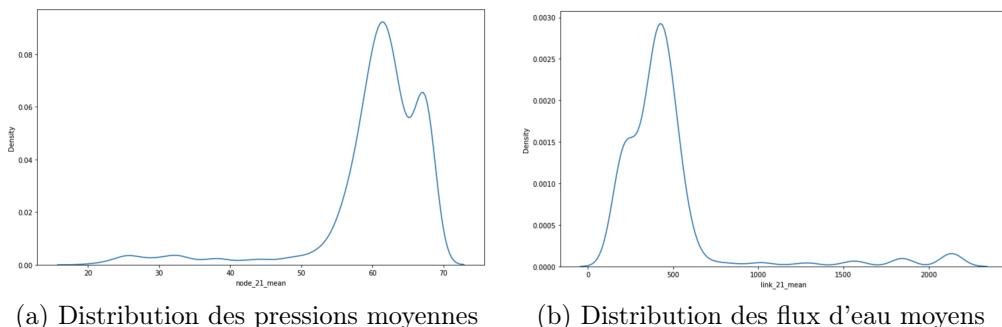


FIGURE 5.23 – Distributions des valeurs moyennes du jeu de données d'apprentissage

Comme énoncé dans l'explication du modèle section 5.9.1, nous faisons une grosse hypothèse (ce qui en fait un modèle "naïf") sur nos données : elles sont i.i.d.

Nous savons que cette hypothèse n'est pas forcément vérifiée : même si les observations de notre jeu de données d'apprentissage ont été tirées aléatoirement dans l'ensemble du jeu de données, il n'est pas impossible que des relations (mises en avant par exemple) existent entre nos observations utilisées pour l'apprentissage. Nos observations ne seraient donc pas indépendantes et par conséquent l'hypothèse d'indépendance n'est pas vérifiée.

Nous estimons la densité de nos valeurs moyennes avec la classe *KernelDensity* de *sklearn* puis nous estimons le seuil (le quantile) pour lequel notre modèle a la plus grande exactitude (comme nous avons fait pour sélectionner les modèles de la section 5.7.1).

Pour rappel, le quantile d'ordre α d'une variable aléatoire X

$$q(\alpha) = F^{-1}(\alpha)$$

avec F la fonction de répartition de la variable aléatoire X

$$F(x) = \mathbb{P}(X \leq x)$$

Les ordres des quantiles obtenus suite à l'entraînement des modèles sont $\alpha_{pressions} = 0,076$ et $\alpha_{flux} = 0,06$ et les quantiles correspondants sont

$$q_{pressions}(0,076) = 51,46$$

$$q_{flux}(0,06) = 195,85$$

Nous obtenons, à partir des données d'apprentissage, les *accuracy scores* suivants

$$accuracy_{pression} = 0,95$$

$$accuracy_{flux} = 0,92$$

La visualisation des résultats (figure 5.24) nous montre que la détection d'anomalies en utilisant un seuillage (à partir d'une estimation de la densité) modélise l'idée naïve qu'une valeur anormale est hors de la distribution moyenne des valeurs de la série (ce qui ne se vérifie pas forcément si nous avons des anomalies contextuelles comme nous l'avons vu).

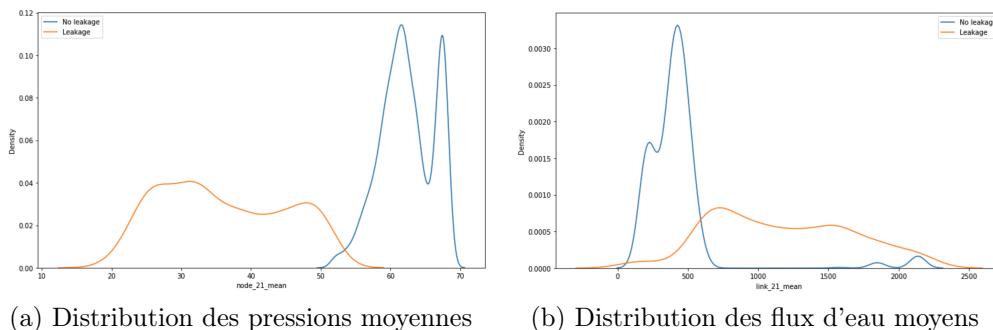


FIGURE 5.24 – Distributions des valeurs moyennes du jeu de données d'apprentissage avec la séparation entre valeurs normales et anormales selon le seuillage appliqué

Après application de ce seuillage sur les données de validation, nous obtenons les métriques suivantes (voir figure 5.25 pour les matrices de confusion) :

L'application du seuillage sur les pressions issues des données de validation nous donne une *precision* de 0,93, un *recall* de 0,56 et un *F1 score* de 0,70. Pour les flux, nous avons une *precision* de 0,75, un *recall* de 0,39 et un *F1 score* de 0,51.

Si nous décidons d'appliquer cette méthode, il semble plus judicieux de ne considérer que les pressions et non les flux. De plus, ces modèles naïfs semblent en tout point être moins performants que les modèles de la section 5.7.3 sur nos données.

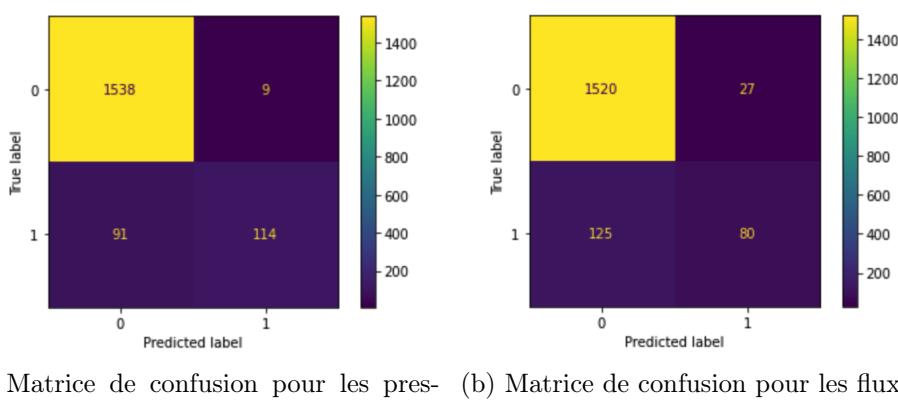


FIGURE 5.25 – Matrices de confusion pour l'application des seuillages sur les données de validation

Chapitre 6

Conclusion et pistes d'améliorations

Lors de cette TX nous avons pu construire une méthode de détection d'anomalies dans des séries temporelles. Cette méthode s'appuie sur des méthodes utilisées actuellement pour détecter différentes anomalies dans des séries temporelles et a été implémentée à partir d'un jeu de données (*leakDB*) en *open data*. Ce jeu de données est une simulation du réseau d'eau de la ville d'Hanoï au Vietnam.

Pour cette étude nous avons considéré les pressions et les flux d'eau à un unique noeud du réseau et nous avons considéré 3 scénarios : 1 sans fuite et 2 avec fuites.

Nous avons construit un modèle de détection s'appuyant sur l'utilisation de 2 modèles *SARIMA* (un pour les pressions et un autre pour les flux d'eau) permettant de déterminer la pression moyenne et le flux d'eau moyen sur 2 heures normalement espérés. Cette valeur est ensuite comparé avec la valeur moyenne réelle obtenue sur 3 heures, puis nous utilisons cette différence ainsi que la valeur moyenne réelle sur les 3 heures, l'écart type à cette valeur moyenne réelle, la médiane, la valeur minimale et la valeur maximale pour déterminer, à l'aide de forêts aléatoires (une pour les pressions et une pour les flux d'eau) si il y a une fuite (i.e si ces données sont considérées comme des anomalies). Nous obtenons des résultats satisfaisant à l'aide de cette méthode : 70% des anomalies sont détectées avec une précision de 99%. Notre méthode semble néanmoins peu performante dans la détection d'anomalies où les valeurs anormales très proches des valeurs normales.

La première amélioration que nous pouvons proposer serait de faire de l'*hyper-parameter tuning* pour les modèles mis en place. En effet, nous n'avons pas tenté d'optimiser nos modèles dans cette étude. Il pourrait être intéressant d'essayer d'optimiser les modèles pour voir si cette optimisation permettrait d'augmenter les performances de notre méthode.

Cette méthode requiert l'utilisation de 4 modèles pour un seul noeud du réseau (2 modèles *SARIMA* et 2 forêts aléatoires). Plus généralement, cette méthode nécessite 2 modèles par phénomène physique mesuré. Une des améliorations possibles serait de regrouper tous les phénomènes physiques dans un seul jeu de données et d'appliquer un modèle de prédiction *multi-varié* comme *VAR* par exemple (version *multi-variée* du modèle *ARIMA*).

De plus, nous pourrions inclure les consommations en eau au noeud considéré pour l'étude pour détecter les anomalies. Il serait également intéressant d'inclure plus de scénarios pour avoir une meilleure estimation sur les performances de notre modèle à bien détecter tout type d'anomalies.

Pour cette étude, nous n'avons considéré qu'un seul noeud. Une amélioration possible serait d'inclure plusieurs noeuds à l'étude afin de voir si notre méthode permet de dire si nous avons une anomalie à un instant t et également à quel noeud du réseau cette anomalie a lieu.

Enfin, dans le cadre de la maintenance préventive, notre méthode permet de dire toutes les 3 heures si nous avons une anomalie ou non. La prédiction avec le modèle *SARIMA* permet

de comparer la valeur moyenne actuelle d'un phénomène à la valeur normalement espérée (i.e prédictive). Il serait intéressant d'essayer de construire un modèle prédictif permettant de prédire au temps $t - 1$ la valeur au temps t la plus proche possible de la valeur réelle (et non de la valeur espérée). Cette prédition, couplée au modèle de classification, permettrait de prédire avec un temps d'avance si la valeur suivante est potentiellement anormale ou non.

Bibliographie

- [1] Jean-Yves Dauxois (2016/2017) *Introduction à l'Étude des Séries Temporelles*, INSA Toulouse. Accessible sur ce [lien](#)
- [2] (2015) *Séries temporelles 2A*, ENSAI. Accessible sur ce [lien](#)
- [3] M.-C. Viano, A. Philippe (1999 à 2004) *Cours de Séries Temporelles*, Université des Sciences et Technologies de Lille, U.F.R. de Mathématiques Pures et Appliquées. Accessible sur ce [lien](#)
- [4] Seif-Eddine Benkabou (2018) *Détection d'anomalies dans les séries temporelles : Application aux masses de données sur les pneumatiques*, Université Claude Bernard Lyon 1. Accessible sur ce [lien](#)
- [5] Gérard Govaert, Thierry Denoeux, Benjamin Quost et Sylvain Rousseau (2022) *Analyse de Données et Apprentissage Automatique*, Université de Technologie de Compiègne.
- [6] Fei Tony Liu, Kai Ming Ting, Zhi-Hua Zho *Isolation Forest*. Accessible sur ce [lien](#)
- [7] Halil Ertan (2020) *Unsupervised Anomaly Detection on Time Series*. Accessible sur ce [lien](#)
- [8] König C, Helmi AM (2020) *Sensitivity Analysis of Sensors in a Hydraulic Condition Monitoring System Using CNN Models*. Accessible sur ce [lien](#)
- [9] Elena Quatrini, Francesco Costantino, Cesare Poccia, Massimo Tronci (2020) *Predictive model for the degradation state of a hydraulic system with dimensionality reduction*. Accessible sur ce [lien](#)
- [10] Vrachimis SG, Kyriakou MS, Eliades (2018) *LeakDB : a Benchmark Dataset for Leakage Diagnosis in Water Distribution Networks*. Accessible sur ce [lien](#)
- [11] Eryk Lewinson (2019) *Explaining Feature Importance by example of a Random Forest*. Accessible sur ce [lien](#)
- [12] Bingblackbean (2021) *Use Clustering to Detect Time Series Anomaly*. Accessible sur ce [lien](#)
- [13] Oğuzhan Yediel (2014) *Time Series Anomaly Detection with PyFBAD*. Accessible sur ce [lien](#)
- [14] Bob Rupak Roy - II (2021) *OneClassSVM*. Accessible sur ce [lien](#)
- [15] Moez Ali (2021) *Introduction to Anomaly Detection in Python with PyCaret*. Accessible sur ce [lien](#)
- [16] Amol Mavuduru (2021) *How to perform anomaly detection with the Isolation Forest algorithm*. Accessible sur ce [lien](#)
- [17] Sanath Raj (2021) *Anomaly Detection using AutoEncoders*. Accessible sur ce [lien](#)
- [18] Insaf Ashrapov (2019) *Anomaly detection in time series with Prophet library*. Accessible sur ce [lien](#)
- [19] Tek Raj Awasthi (2020) *Anomaly Detection in Time Series Data Using Keras*. Accessible sur ce [lien](#)

- [20] Marco Cerliani (2020) *Anomaly Detection in Multivariate Time Series with VAR*. Accessible sur ce [lien](#)
- [21] Maarit Widmann (2021) *Anomaly Detection for Predictive Maintenance — Exploratory Data Analysis*. Accessible sur ce [lien](#)
- [22] Marco Cerliani (2020) *Real-Time Time Series Anomaly Detection*. Accessible sur ce [lien](#)
- [23] Joe El khoury (2018) *Introduction To Anomaly Detection Methods — Part I*. Accessible sur ce [lien](#)
- [24] Lleyton Ariton (2020) *Houston, we have a problem — Time Series Anomaly Detection*. Accessible sur ce [lien](#)
- [25] Christian Derquenne (2021) *Détection d'anomalies dans des séries temporelles régulières : une approche non paramétrique*. Accessible sur ce [lien](#)
- [26] Dominik Polzer (2021) *A Comprehensive Beginner's Guide to the Diverse Field of Anomaly Detection*. Accessible sur ce [lien](#)
- [27] Zhou (Joe) Xu (2020) *Time Series Pattern Recognition with Air Quality Sensor Data*. Accessible sur ce [lien](#)
- [28] Koen Peters (2020) *Random Forest for predictive maintenance of turbofan engines*. Accessible sur ce [lien](#)
- [29] Marco Cerliani (2019) *Predictive Maintenance : detect Faults from Sensors with CNN*. Accessible sur ce [lien](#)
- [30] Marco Cerliani (2019) *Predictive Maintenance with LSTM Siamese Network*. Accessible sur ce [lien](#)
- [31] Sunilkumar (2021) *Predictive Maintenance of Pumps*. Accessible sur ce [lien](#)
- [32] Alparslan Mesri (2022) *Predictive Maintenance Classification, Part II — Building Your Own Metric*. Accessible sur ce [lien](#)
- [33] Luis Conti (2021) *Machine learning models applied to turbofan engines predictive maintenance*. Accessible sur ce [lien](#)