Projet R&D

Prédiction de consommations futures

Spécialité Bâtiments – 5ème année

Réalisé par BENMDES Eya, FONTAINE Maëva et MORENA Ugo

Encadré par Pr. FRANQUET Erwin

Septembre-Décembre 2022

Table des matières

Introduction

La consommation mondiale d'énergie a augmenté de 100 % au cours des 40 dernières années[1], bien que la mise en place de Systèmes de Management de l'Energie (SME) se développe afin de réduire les impacts environnementaux et de diminuer les coûts énergétiques. De plus, selon une étude de l'Observatoire national de la rénovation énergétique (ONRE) publiée le 22 juillet 2022, en France, plus de 5 millions de logements sur 30 millions sont considérés comme des passoires thermiques[2]. L'émergence des entreprises, labels et certifications de rénovation énergétique met en lumière l'enjeu de demain dans le domaine de la construction : comprendre nos consommations pour mieux les optimiser et fournir les rénovations nécessaires en vue de diminuer nos consommations d'énergie. C'est dans cette optique que certaines entreprises et organisations cherchent à appliquer le Machine Learning au profit de l'industrie de l'énergie.

Le but de notre projet est de tester des méthodes de Machine Learning afin de prédire au mieux la consommation future en énergie d'un bâtiment, quel que soit le type, à partir de ses consommations passées. Nous allons tout d'abord, lors d'un état de l'art sur le Machine Learning, étudier différentes méthodes de Machine Learning ainsi que leur domaine d'application. Ensuite, nous allons choisir deux méthodes grâce aux recherches bibliographiques effectuées et les détailler. Enfin, nous allons tester ces deux méthodes en les implémentant sur Python, et observer les résultats obtenus en fonction des diverses variables d'entrée.

[1] International Energy Agency

[2] https://www.effy.fr/pro/actualite/en-france-5-millions-de-residences-principales-sont-des-passoires-thermiques

1. Qu'est-ce que le Machine Learning ?

Le Machine Learning ou Apprentissage Automatique, est une science permettant de découvrir des répétitions dans un ou plusieurs flux de données et d'en tirer des prédictions en se basant sur des statistiques[3].

Réaliser des analyses prédictives consiste à exploiter les données issues du Big Data et les traiter à l'aide d'algorithmes statistiques et de techniques de Machine Learning afin de prédire des probabilités en se basant sur les données passées. Pour citer Tom Mitchell, *« a machine learning algorithm is an algorithm that improves performance on a task as it is exposed to data »*. Plus il y a de données dans un dataset, c'est-à-dire dans un jeu de données, et plus il est aisé d'obtenir de bons résultats en termes de prédictions. Dans notre cas les données d'entrée sont les consommations énergétiques des bâtiments.

2. Quels sont les avantages de la prévision des consommations d'énergie ?

Il existe plusieurs avantages à la prédiction des consommations d'énergie :

- Economique : en estimant les futures quantités d'énergies, et donc les futures factures énergétiques grâce aux consommations passées, les usagers peuvent prendre des décisions à partir de ces données afin d'effectuer des rénovations dans l'optique d'économiser de l'argent.

- Environnemental : la construction d'un comportement de référence permet de repérer les anomalies dans les consommations réelles et de les corriger. Cela évite des surconsommations d'énergie.

- Temporel : prédire permet aussi d'anticiper si des rénovations sont nécessaires. Certaines situations peuvent se dégrader et engendrer de nouveaux sinistres en cascade, d'ordre structurel par exemple.

3. Comment le Machine Learning prédit les consommations d'énergie ?

Généralement, les habitudes en termes de consommation d'énergie au sein d'un foyer se conservent tant qu'un évènement inattendu ne se produit pas. Ainsi, en collectant des données sur la consommation d'énergie passée pour un bâtiment, il est possible de révéler des tendances grâce à des modèles probabilistes et ainsi prédire les futurs modèles de consommation d'énergie.

[3] https://ia-data-analytics.fr/machine-learning/

4. Quelles sont les différentes méthodes de Machine Learning ?

En effectuant des recherches bibliographiques sur le sujet des prédictions de consommations grâce au Machine Learning, nous avons sélectionné une dizaine d'articles scientifiques qui analysaient le fonctionnement et la pertinence de différentes méthodes de Machine Learning. Ces méthodes peuvent tout d'abord être classées par mode d'apprentissage : supervisé, renforcé et profond.

4.1 Apprentissage supervisé

Tout d'abord, l'apprentissage supervisé est l'un des modes d'entrainement le plus utilisé. C'est une méthode qui s'appuie sur des données ou des exemples étiquetées pour entrainer des modèles d'intelligence artificielle prédictifs. En effet, au fur et à mesure de l'enrichissement du modèle, il est remarqué que le résultat gagne en pertinence et permet ainsi de réduire la marge d'erreur. La durée d'apprentissage est rapide, de l'ordre de la seconde à l'heure, et possède une très bonne transparence avec des résultats facilement interprétables. Cependant, il faut bénéficier d'un gros volume de données labellisées pour que le modèle soit efficace[4].

On peut classer les différentes méthodes d'apprentissage supervisé suivant le type de problème à traiter : les algorithmes de classification et de régression.

Pour les algorithmes de régression, le but est de trouver la fonction pour faire correspondre la variable d'entrée "x" avec la variable de sortie continue "y". Les algorithmes de classification, eux, ont pour but de trouver la fonction qui fait correspondre l'entrée "x" à la sortie discrète "y", en fonction de variables indépendantes (Figure 1).

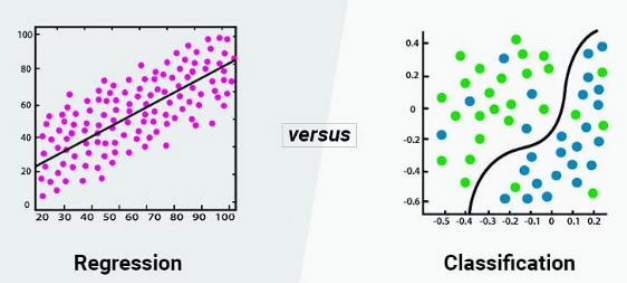


Figure 1 : Graphiques représentant les algorithmes de régression et de classification

[4] https://www.researchgate.net/profile/Batta-Mahesh/publication/344717762_Machine_Learning_Algorithms_-A_Review/links/5f8b2365299bf1b53e2d243a/Machine-Learning-Algorithms-A-Review.pdf?eid=5082902844932096

4.1.1 Les algorithmes de classification

La classification (ou clustering) est une tâche typique d'apprentissage supervisé qui est utilisée dans le cas où il faut prédire un type catégoriel, c'est-à-dire si un exemple appartient à une catégorie ou non, la valeur de sortie est discrète. Différents modèles de classification existent[5] :

- Le modèle de **régression logistique** est utilisé pour modéliser des variables dépendantes binaires, elle explique la survenance ou non d'un événement par le niveau de variables explicatives. Par exemple, on peut chercher à évaluer à partir de quelle dose de médicament un patient sera guéri ;

- Le modèle des **K-voisins les plus proches** prédit à quelle classe appartient un nouveau point de données en identifiant la classe majoritaire de ses k voisins les plus proches (Figure 2).

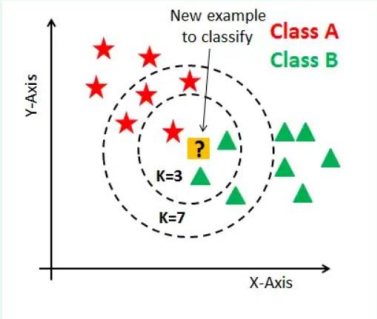


Figure 2 : Représentation graphique du modèle des K-voisins les plus proches[6]

L'article nommé *« Comparison of integrated clustering methods for accurate and stable prediction of building energy consumption data »*[7] paru dans l'ouvrage *« Applied Energy »* en 2015 compare et analyse les différentes méthodes de classification en étudiant la précision et la stabilité de prédiction de consommation d'énergie dans des immeubles résidentiels à New-York. La méthode de régression logistique a donné des prédictions extrêmement précises mais des clusters quelque peu instables, tandis que les K-voisins les plus proches donnent des clusters plus stables, mais des prédictions très faibles dans certains groupes. On dit qu'un cluster est stable s'il est insensible à des petits changements dans les données de base. Il semble donc y avoir un compromis entre la précision de prédiction de la méthode de régression et la stabilité de la méthode K-voisins, il est donc important de définir ses objectifs pour choisir une des deux méthodes.

[5] https://arxiv.org/ftp/arxiv/papers/1205/1205.1117.pdf

[6] https://www.researchgate.net/publication/348232336_Prediction_du_Churn_Rate_Par_le_Machine_Learning_dans_le_secteur_des_MA_Application_au_sein_de_KPMG

[7] https://www.sciencedirect.com/science/article/pii/S0306261915010624

4.1.2 Les algorithmes de régression

La régression, quant à elle, est une tâche d'apprentissage supervisé où la valeur de sortie est continue, l'objectif étant de prédire au mieux une valeur aussi proche de la valeur de sortie réelle que possible. Cette technique établit donc une relation entre l'entrée et la sortie. Les algorithmes de régression trouvent des corrélations entre les variables dépendantes et indépendantes, ils ont pour but de prédire des variables continues, par exemple le prix de l'immobilier au cours du temps.

Plusieurs algorithmes de régression se distinguent[8] :

- Les **Random Forest Regression**, ou forêt d'arbres de régression, sont constitués d'une multitude d'arbres de régression combinés qui sont entrainés par le principe de bagging et la sélection de variables aléatoires.
- La **régression linéaire simple**, qui établit une relation entre la variable d'entrée indépendante, « x », et la variable de sortie continue, « y » (Figure 3).

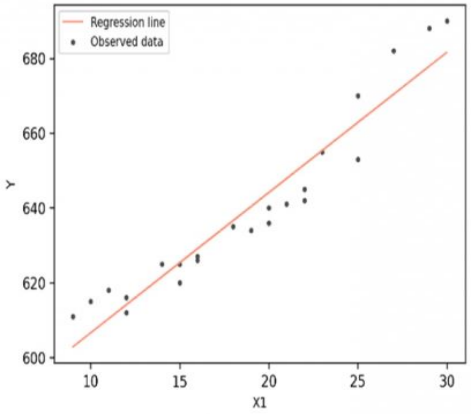


Figure 3 : Représentation graphique de la régression linéaire simple

- La **régression linéaire multiple**, qui utilise un grand nombre de variables indépendantes pour déterminer la valeur de sortie, « y ». Cela peut permettre d'améliorer la précision du modèle, si les variables ajoutées sont pertinentes.
- La **régression linéaire polynomiale**, qui est un cas particulier de la régression linéaire multiple, se différencie par le fait que la relation entre la variable d'entrée, « x », et la variable de prédiction, « y », est modélisée comme le N-ième degré polynomial de « x » (Figure 4).

[8] https://www.researchgate.net/profile/Batta-Mahesh/publication/344717762_Machine_Learning_Algorithms_-A_Review/links/5f8b2365299bf1b53e2d243a/Machine-Learning-Algorithms-A-Review.pdf?eid=5082902844932096

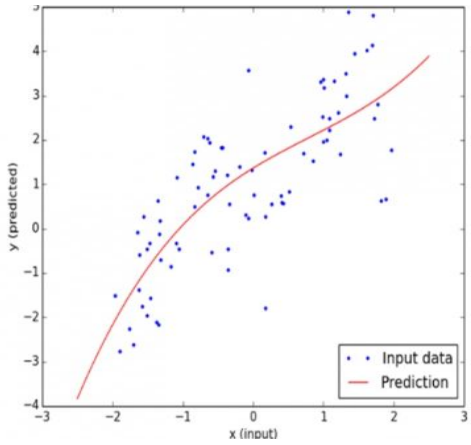


Figure 4 : Représentation graphique de la régression linéaire polynomiale

L'article *« Random Forest based hourly building energy prediction »*[9] publié dans l'ouvrage "*Energy and Buildings*" en 2018, étudie les prédictions d'énergie heure par heure dans des bâtiments scolaires en Floride. La méthode de Random Forest Regression a été comparée avec les méthodes de Regression Tree et Linear Regression. Il a été étudié que les performances de prédiction du Random Forest Regression mesurées grâce à l'indice de performance (PI) étaient de 14 à 25% supérieures à celles de l'arbre de régression et de 5 à 5.5% supérieures à celles de la régression linéaire. L'indice de performance est la proportion de la variance d'une variable dépendante qui s'explique par une ou plusieurs variables indépendantes dans le modèle de régression. Ainsi, il a été prouvé que la méthode Random Forest Regression demeure être la meilleure.

[9] https://www.sciencedirect.com/science/article/pii/S0378778818311290

4.2 Apprentissage par renforcement

Ensuite, l'apprentissage par renforcement est continu et automatique, c'est-à-dire qu'il apprend de ses erreurs. Il diffère des apprentissages supervisés par le fait que l'algorithme essaie plusieurs solutions, observe la réaction de l'environnement et adapte son comportement pour obtenir les meilleurs résultats. Ainsi, il n'est pas nécessaire de fournir des données d'entrainement à l'algorithme, il acquiert des données directement au contact de son environnement et les mémorise, c'est un apprentissage itératif. La durée de l'apprentissage est alors en continu. L'algorithme d'apprentissage par renforcement le plus utilisé est le **Q-Learning**. C'est une politique d'apprentissage par renforcement qui trouvera la meilleure action suivante, étant donné l'état actuel. Elle choisit cette action au hasard et vise à maximiser la récompense.

D'après l'article « *User comfort and energy efficiency in HVAC systems by Q-learning* »[10] publié par l'IEEE (Institute of Electrical and Electronics Engineers) en 2018, la méthode Q-Learning s'est montrée avantageuse pour maintenir la qualité de l'air intérieure (la concentration de CO2) au niveau souhaité tout en faisant fonctionner correctement le système CVC.

4.3 Apprentissage profond

Enfin, l'apprentissage profond est un apprentissage automatique inspiré de l'anatomie du cerveau humain, il est associé à une structure algorithmique appelée réseau de neurones. Ce type d'apprentissage présente plusieurs avantages : leur structure permet une énorme capacité de données et agissent comme des algorithmes de classification et comme des extracteurs de caractéristiques. En effet, ce sont les données elles-mêmes qui sont fournies en entrée à un réseau de neurones. Cependant, le temps et la quantité de ressources informatiques nécessaire à l'entrainement de l'algorithme augmentent de manière exponentielle avec le volume du dataset et la complexité du réseau de neurones, cela peut créer un « sur-apprentissage »[11]. L'apprentissage profond repose sur le modèle de « **réseaux de neurones** ». Un réseau de neurones est un modèle se composant de couches de neurones artificiels qui permet de garder en mémoire une valeur qui permettra de prédire la suivante (Figure 5).

[10] https://ieeexplore.ieee.org/document/8404287

[11] https://culturesciencesphysique.ens-lyon.fr/ressource/IA-apprentissage-Rousseau.xml

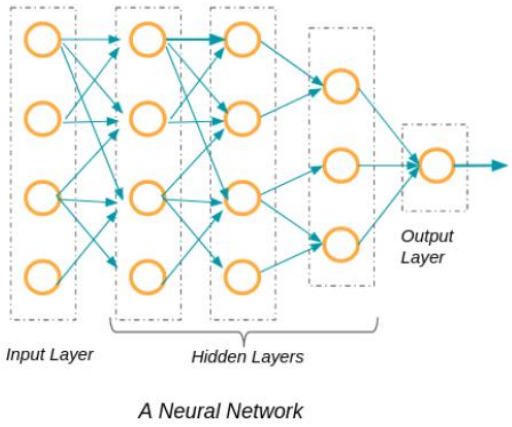


Figure 5 : Schéma d'un réseau de neurones

La première couche d'un réseau de neurone est la couche d'entrée, appelée « input layer », elle contient les données d'entrée. Ensuite, il y a un grand nombre de couches cachées, appelées « hidden layer » qui représentent autant de données qu'il y a de couches. Plus il y a de couches cachées dans le réseau, plus celui-ci est profond. La couche finale, « output layer », contient le résultat.

Il existe 3 types d'architectures différents de réseaux de neurones :

- Les **réseaux de neurones convolutifs** (CNN). L'article « *Predicting residential energy consumption using CNN-LSTM neural networks* »[12] publié dans l'ouvrage "*Energy*" en 2019, présente un réseau de neurones CNN-LSTM qui extrait des données spatiales et temporelles pour prédire la consommation d'énergie d'un logement. La couche CNN est utilisée pour extraire les caractéristiques entre plusieurs variables qui affectent la consommation d'énergie et la couche LSTM modélise les données temporelles des courbes irrégulières des séries temporelles. Il a été montré que la méthode CNN-LSTM prédit la demande d'électricité d'un bâtiment résidentiel de manière efficace et stable, et affiche les meilleures performances par rapport aux autres méthodes. Cependant, ces performances sont atteintes à condition de collecter un grand nombre de données de consommations d'énergie.

- Les réseaux de neurones récurrents (RNN) sont des réseaux de neurones artificiels qui présentent des connexions récurrentes constitués d'unités interconnectées[13].

- Les auto-encodeurs, eux, apprennent une représentation d'un ensemble de données dans le but de réduire la dimension de cet ensemble.

[12] https://www.sciencedirect.com/science/article/pii/S0360544219311223

[13] https://icml.cc/2011/papers/524_icmlpaper.pdf

5. Les méthodes de ML choisies

5.1 Random Forest Regression

Le « Random Forest Regression » ou la forêt d'arbres décisionnels régressifs est un algorithme de prédiction qui se base sur l'assemblage de plusieurs arbres de décision (Figure 6)[14].

Le Random Forest Regression a besoin de 3 paramètres principaux qui doivent être définis. Il s'agit de la taille des arbres, du nombre d'arbres à utiliser et le nombre de caractéristiques (nombre de variables aléatoires). A partir de là, le modèle peut être utilisé pour résoudre des problèmes de régression ou de classification. Pour notre cas, il sert à résoudre des problèmes de régression.

Etape 1 : appliquer le principe de bagging, c'est-à-dire créer des sous-échantillons aléatoires de notre ensemble de données avec possibilité de sélectionner la même valeur plusieurs fois. Le **bagging** est une technique utilisée pour améliorer la classification.

Etape 2 : des arbres de décision individuels sont ensuite construit pour chaque échantillon. Chaque arbre est entrainé sur une portion aléatoire afin de recréer une prédiction. Chaque arbre de décision fonctionne individuellement et indépendamment des autres.

Etape 3 : chaque arbre va prédire un résultat. La moyenne des votes de tous les arbres devient le résultat final du modèle.

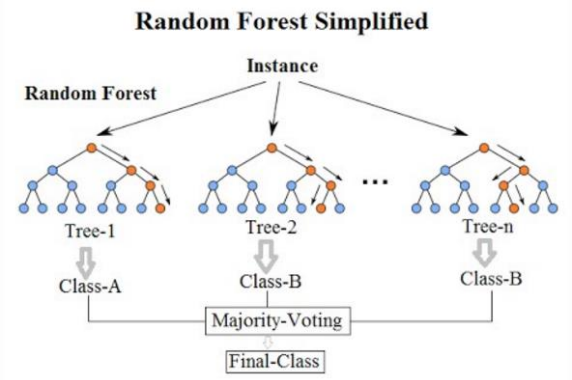


Figure 6 : Modèle de forêt d'arbres décisionnels régressifs

Les avantages du modèle Random Forest Regression :

Cet algorithme est rapide à entrainer parce qu'il combine les résultats de plusieurs sous-modèles. C'est un algorithme à utiliser avec des données tabulaires qui est notre cas. C'est un outil qui permet de faire des prédictions précises à la prise de décisions stratégiques. Random Forest présente l'avantage d'utiliser plus intelligemment l'ensemble de ces données initiales, afin de limiter ses erreurs.

[14] https://datascientest.com/random-forest-definition

5.2 LSTM

Le LSTM (Long Short-Term Memory) est un algorithme de prédiction qui aide le système à transporter les données pendant une longue période. Le LSTM se souvient du problème pendant plus longtemps et possède une structure en chaîne permettant de répéter le module. Ils interagissent selon une méthode spéciale et contiennent quatre couches de réseau neuronal.

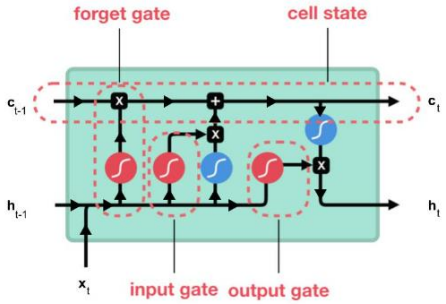
Le mécanisme de fonctionnement de LSTM se base sur la propagation de l'information. L'opération décide quelle information parmi les informations passées doit être traitée avant et de quelle information elle doit se défaire. Une fonction qui permet de trier les informations qu'elle veut stocker sur une longue séquence. L'opération principale consiste en des cellules et des portes. Les cellules sont considérées comme une mémoire.

Lorsque la cellule porte l'information, les portes aident le nouveau flux d'information. Les portes indiquent les données qui sont utiles de conserver et celles qui ne le sont pas, ce qui permet de les jeter. Ainsi, seules les données pertinentes circulent à travers chaîne de séquence pour faciliter la prédiction.

Il existe 3 types de portes pour faciliter le fonctionnement de l'opération (Figure 7) :

- Porte d'oubli : La fonction de cette porte est de décider de garder ou d'oublier l'information en y appliquant la fonction sigmoïde pour une matrice de poids qui contient les inputs de la cellule à l'instant t et l'état de la cellule à l'instant t-1 afin de normaliser les valeurs entre 0 et 1. Si la sortie de la sigmoïde est proche de 0, cela signifie que l'on doit **oublier l'information** et si on est proche de 1 alors il faut **la mémoriser** pour la suite.
- Porte d'entrée : La porte d'entrée aide à mettre à jour l'état de la cellule et rajouter une information dans la mémoire. D'abord on multiplie un par un la sortie de l'oubli avec l'ancien état de la cellule. Cela permet d'**oublier certaines informations** qui ne servent pas pour la nouvelle prédiction à faire. Ensuite, on additionne le tout avec la sortie de la porte d'entrée
- Etat de la cellule : L'état de la cellule se calcule à partir de la porte d'oubli et de la porte d'entrée. On met à jour l'état de la cellule précédente de la même manière que pour la porte d'entrée. Ensuite on additionne le tout avec la sortie de la porte d'entrée, ce qui permet d'enregistrer dans l'état de la cellule ce que le LSTM (parmi les entrées et l'état caché précédent) a **jugé pertinent**.
- Porte de sortie : Cette porte doit décider le prochain état caché, qui contient des informations sur les entrées précédentes du réseau et sert aux prédictions. L'état caché précédent passe, pour sa part, dans une fonction sigmoïde dont le but est de décider des **informations à conserver.** Cette porte aussi permet de définir l'état de la cellule à l'instant t.

- Forget gate (porte d'oubli)
- Input gate (porte d'entrée)
- Output gate (porte de sortie)
- Hidden state (état caché)
- Cell state (état de la cellule)



Cellule LSTM (crédit : image modifiée de Michaël Nguyen)

Figure 7 : Différents ports de l'algorithme LSTM[15]

Les avantages du modèle LSTM

Le LSTM permet d'oublier ou de mémoriser sélectivement les informations de la séquence temporelle précédente dans une mémoire dynamique. Cette méthode permet aussi de traiter les données isolées. Non seulement ça mais aussi elle permet d'éviter les problèmes liés à une dépendance à long terme. Ce qui permet d'avoir des résultats d'une haute précision.

6. Codage des méthodes de Machine Learning

L'objectif de notre code est d'implémenter deux méthodes de prédictions de séries temporelles afin d'obtenir des résultats anticipés des consommations de bâtiment. On pourra ensuite comparer les deux méthodes et déterminer laquelle est la plus précise en précisant ce qui les différencie.

Librairies pré requises : matplotlib, scipy, numpy, tensorflow, pandas, sklearn

Nous avons décidé d'utiliser l'interface web JupyterLab, permettant une meilleure organisation du code et des données. Aussi, la représentation graphique sera un outil précieux en vue de comparer nos modèles prédictifs, que ce soit entre eux ou avec la réalité.

6.1 Chargement des données et lissage

La première phase de la programmation est de choisir quel type de données nous allons choisir comme variable d'entrée du code. Les méthodes de Machine Learning utilisées se basent sur des données enfin d'établir des prévisions. L'importance des variables d'entrée n'est pas à négliger. Dans le cas de notre projet, nous avons établi que nos données d'entrée seront les consommations antérieures d'un bâtiment (qu'il soit individuel ou collectif). Néanmoins, en vue d'obtenir la meilleure corrélation possible, il serait nécessaire de prendre en compte d'autres variables.

Prenons comme exemple l'évolution de la consommation d'un bâtiment entre les périodes avant/après COVID-19. En règle générale, le confinement a entraîné l'augmentation du nombre d'usagers quotidien du bâtiment, et donc les consommations du bâtiment. Ainsi, en utilisant une

[15] https://penseeartificielle.fr/comprendre-lstm-gru-fonctionnement-schema/

méthode de prédiction avec les consommations avant COVID comme données d'entrée, les consommations prédictives ne devraient pas corréler avec les consommations réelles, puisqu'un évènement non prédictible est apparu. Le même problème fait face pour prédire les consommations après COVID. Pour pallier cette éventualité, avoir une deuxième variable d'entrée, comme le taux de CO_2 dans le bâtiment, peut être une solution.

Nos données d'entrées sont toutes au format *.csv* qui est l'un des plus performants quand il s'agit de traitement de données. L'utilisation d'autres formats (comme *.xls*) reste possible.

La lecture et l'assignation des données du fichier à un *dataset* se fait grâce à la librairie pandas, et plus particulièrement la fonction *pandas.csv_read* issu de ce module. La fonction prend en arguments différents paramètres comme le nom du fichier et les colonnes servants d'index dans notre cas.

Nous avons choisi trois différentes données d'entrée :

1. Les consommations d'énergie (en W) d'un bâtiment ;
2. Les besoins en chauffage (en W) d'une maquette Pleiades réalisée en 4ème année du cycle ingénieur bâtiment ;
3. Consommation (en W) d'une zone de la ville de Tétouan (Maroc), pas de temps de 10 min, issu d'un des articles étudiés[16].

Utiliser différentes données d'entrée permet d'évaluer les deux méthodes de Machine Learning choisie avec différente taille et type de données.

[16] https://doi.org/10.1109/irsec.2018.8703007

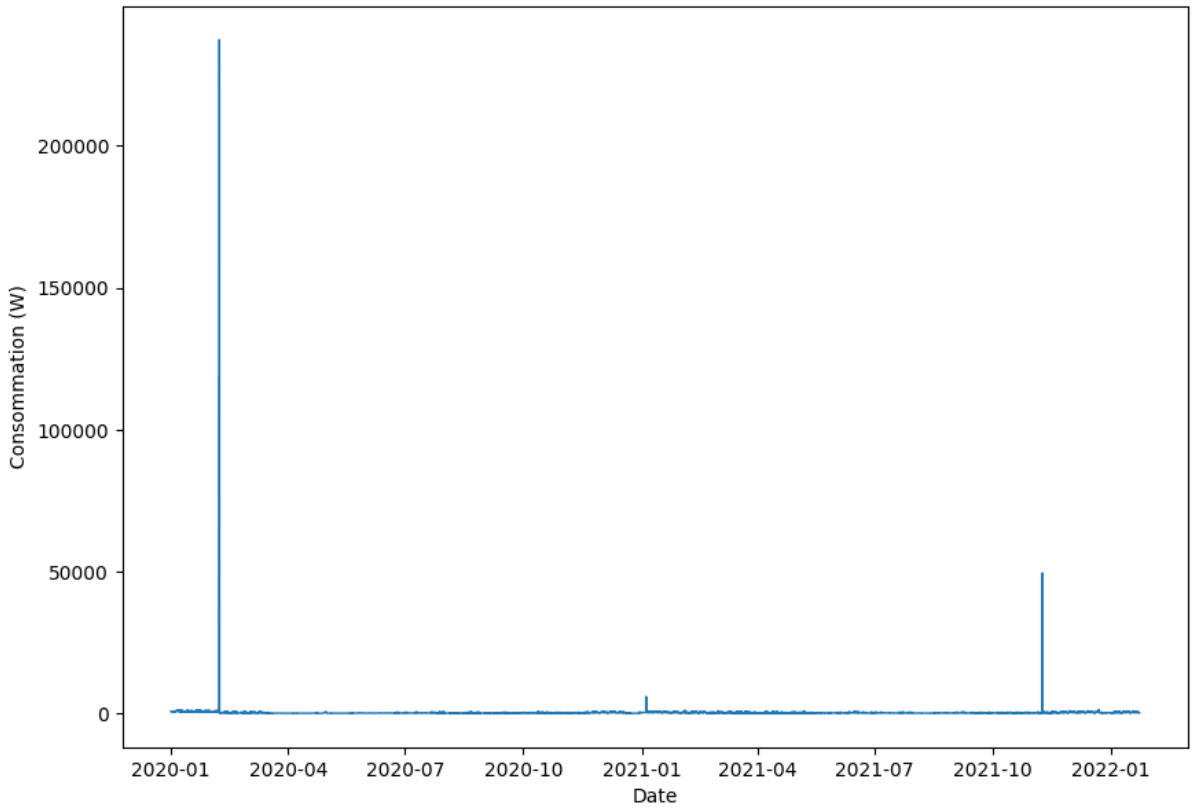


Figure 8 : échantillon 1 après importation – Consommation d'électricité (en W) en fonction du temps

Après l'importation des données, il est important de les lisser afin d'éliminer toute valeur incohérente ou en dehors de la plage d'étude, pouvant fausser les schémas prédictifs. En Python, nous utiliserons la fonction *stats.zscore*[17] de la librairie *scipy*, permettant de calculer le *z_score* de chaque valeur du *dataset* par rapport à la moyenne et à l'écart-type de l'échantillon. Toute valeur supérieure à une valeur seuil définie par nos soins sera supprimée de la plage de donnée d'étude.

[17] https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.zscore.html

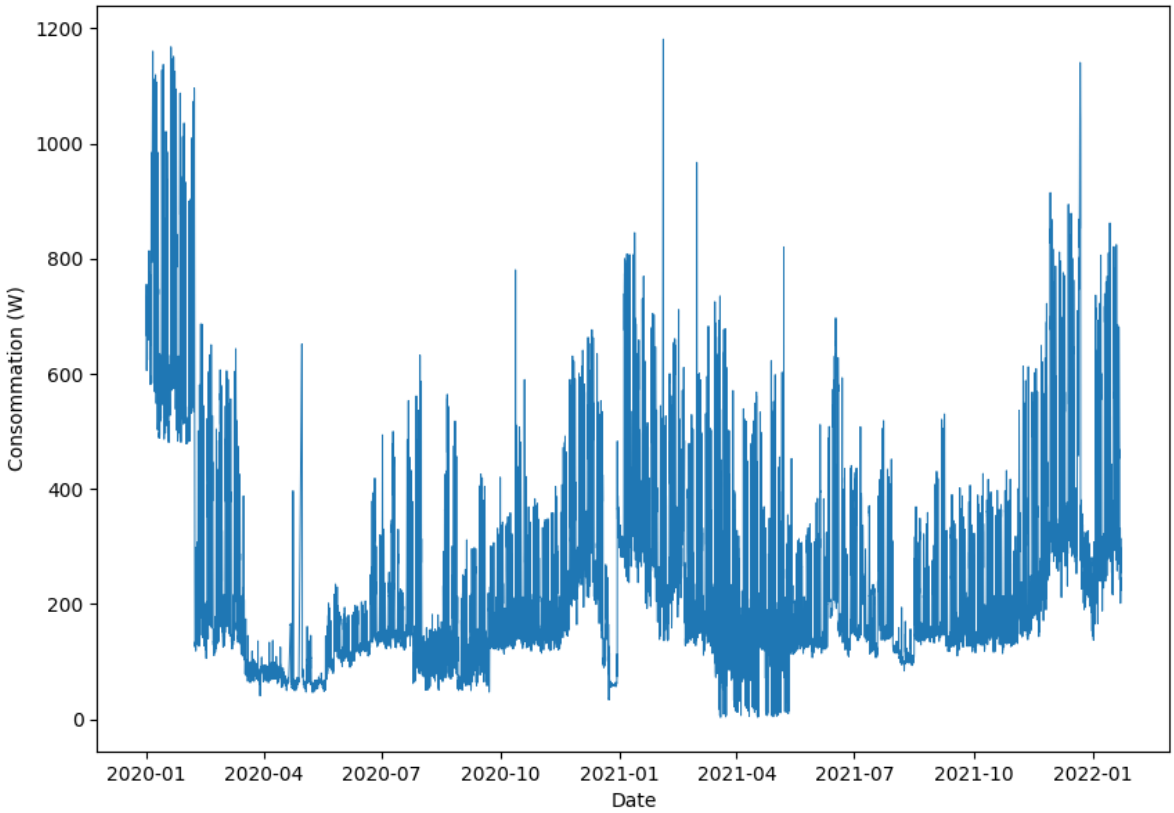


Figure 9 : échantillon 1 après lissage des données

Nous procédons enfin à une mise à l'échelle permettant de limiter les variations de grandeurs et utiliser une échelle unique peu importe la donnée d'entrée, dans le but de faciliter la tâche des algorithmes de prédiction. La normalisation des données permet donc d'éviter les explosions du gradient.

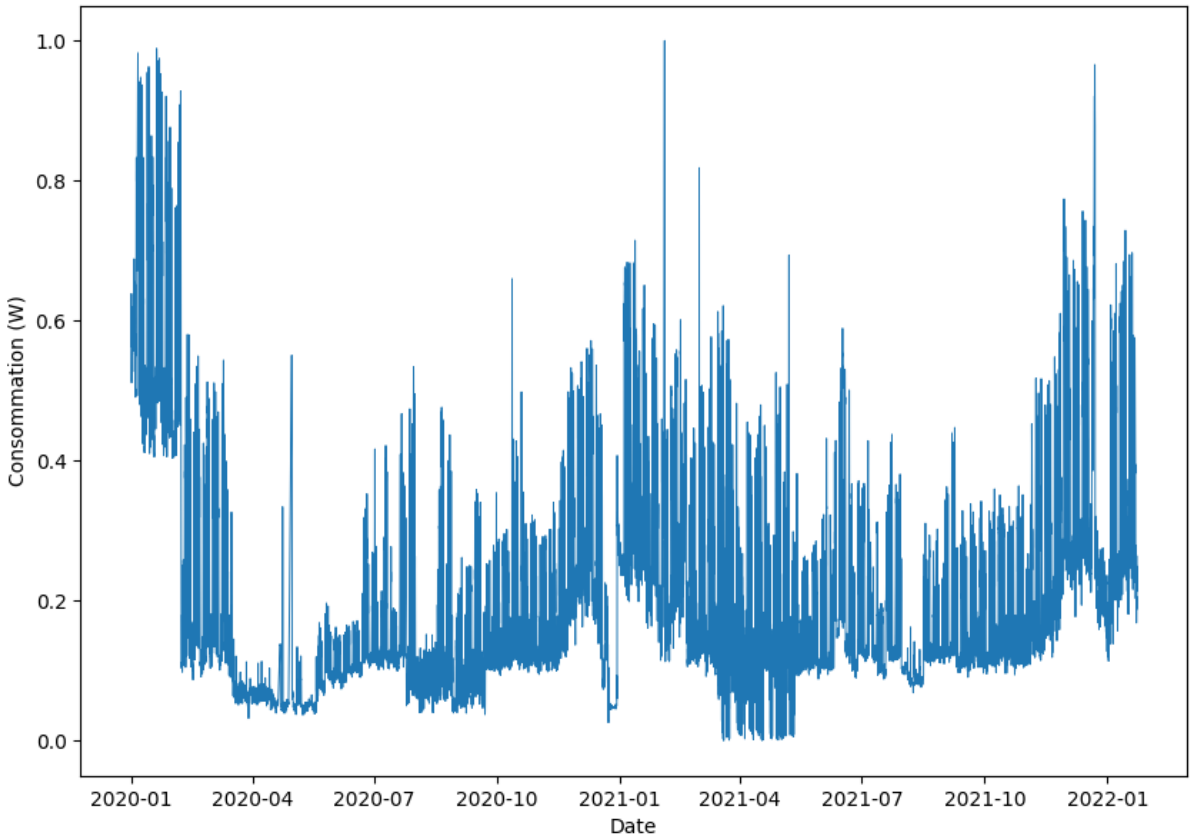


Figure 10 : échantillon 1 après la mise à l'échelle

a. Méthode de prédiction n°1 : LSTM

Dans cette première méthode, on définit dès le début deux types de séries : *trainGenerator* et *testGenerator*. L'une représente la série d'entrée qui va être utilisée comme modèle afin de déterminer une prédiction, l'autre correspond aux valeurs réelles de la série qui va être prédite. La taille de la série modèle représente 95% de la liste de données entière, celle de la série test est de 5%. Il est commun de prendre cette répartition dans le cas de série temporelles, mais dans d'autres applications, la répartition sera différente.
L'utilisation de la fonction Sequential() issu de la librairie tenserflow permet d'initialiser le modèle prédictif, auquel on vient ajouter des paramètres comme la taille de la séquence étudiée et l'échelle d'unité. Dans notre cas, nous avons choisi d'utiliser l'algorithme d'optimisation RMSprop, permettant d'accélérer l'apprentissage du réseau de neurones en accélérant la descente de gradient.
Avant de lancer la prédiction, il est important de définir les données d'entrée, qui vont être utilisées afin d'entrainer l'algorithme à l'aide de la méthode « *fit* ».
On utilise dans la fonction le paramètre epochs correspondant au nombre de fois où l'on va parcourir l'échantillon d'entraînement. Ce paramètre peut être assimilé au nombre de fois où l'on va relire son cours pour un examen par exemple. Logiquement, plus l'on révise, plus le sujet va être maitrisé. Néanmoins, dans la théorie comme dans notre analyse avec le Machine Learning, on va constater que si le modèle révise trop, il va perdre en précision à cause du sur entraînement sur le jeu de données.

L'entraînement de notre modèle prédictif se base sur 10 epochs nous étudierons le comportement de notre modèle en changeant ces paramètres.
A la fin de la prédiction, la fonction *mean_squarred_error* (RMSE) permet d'avoir un indice de précision sur notre méthode de machine learning. Cette fonction est la moyenne des sommes des carrés des différences entre les valeurs réelles et les valeurs prédites telle que : $\frac{(x_1-y_1)^2+(x_2-y_2)^2+\cdots(x_n-y_n)^2}{n}$

b. Méthode de prédiction n°2 : Random Forest Regression Model

Afin d'implémenter le modèle de forêt d'arbres décisionnels, nous allons utiliser le module python sklearn et plus particulièrement la fonction *RandomForestRegressor*. De nombreux paramètres détaillés dans la documentation de la fonction[18] peuvent être utilisés. Dans notre cas, nous nous focaliserons uniquement sur les deux paramètres suivants :

- n_estimators : correspond au nombre d'arbres de décisions utilisés dans le modèle ;
- max_depth : la profondeur maximale pour chaque arbre de décision. Ce paramètre influence considérablement la précision du modèle.

Pour le moment, on initialise notre méthode de forêts aléatoires avec 5 arbres de profondeur maximale 4. Nous verrons par la suite les impacts du changement de ces paramètres.

Après avoir initialisé le modèle, on utilise une nouvelle fois la méthode *fit()*. Cette fonction prend en compte deux paramètres cette fois-ci : les données caractéristiques *dataX*, et les données cibles *dataY* (target value), toutes deux issues des données d'entraînement.

Une fois la prédiction terminée, notre indice d'erreur va être déterminé grâce à la fonction *mean_squarred_error*, comme expliquée dans la partie 6.2. A partir du RMSE, on en déduira une « précision », en réalité issue de l'erreur moyenne de la méthode. Il faut bien distinguer un indice d'erreur et une précision, mais cette analogie permettra d'avoir une compréhension plus aisée de l'efficacité des méthodes.

c. Comparaison des méthodes

Grâce aux différents tests réalisés, nous pouvons à présent analyser les résultats obtenus et définir des paramètres influençant la précision des deux méthodes.

Les données d'entrée jouent un premier rôle sur la précision de chaque méthode. La taille de chaque échantillon étudié a un impact important, puisqu'elle détermine le nombre de données utilisées par l'algorithme. Même sans résultat, on aurait tendance à dire que le schéma prédictif sera plus précis si le nombre de données en entrée est conséquent. Mais qu'en est-il en réalité ?

Dans la première simulation, la taille de l'échantillon était de 18073 données de consommation heure par heure d'un bâtiment. Voici les résultats obtenus des deux modèles :

[18] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html

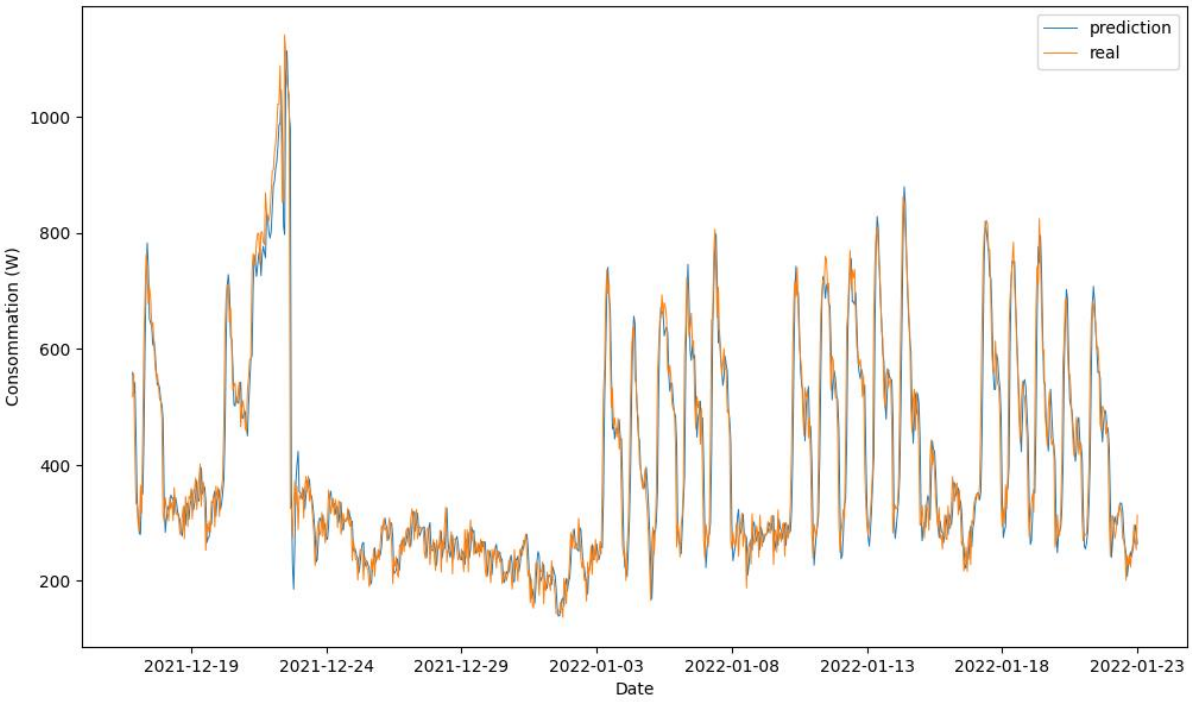


Figure 11 : LSTM - échantillon 1 (epoch=10)

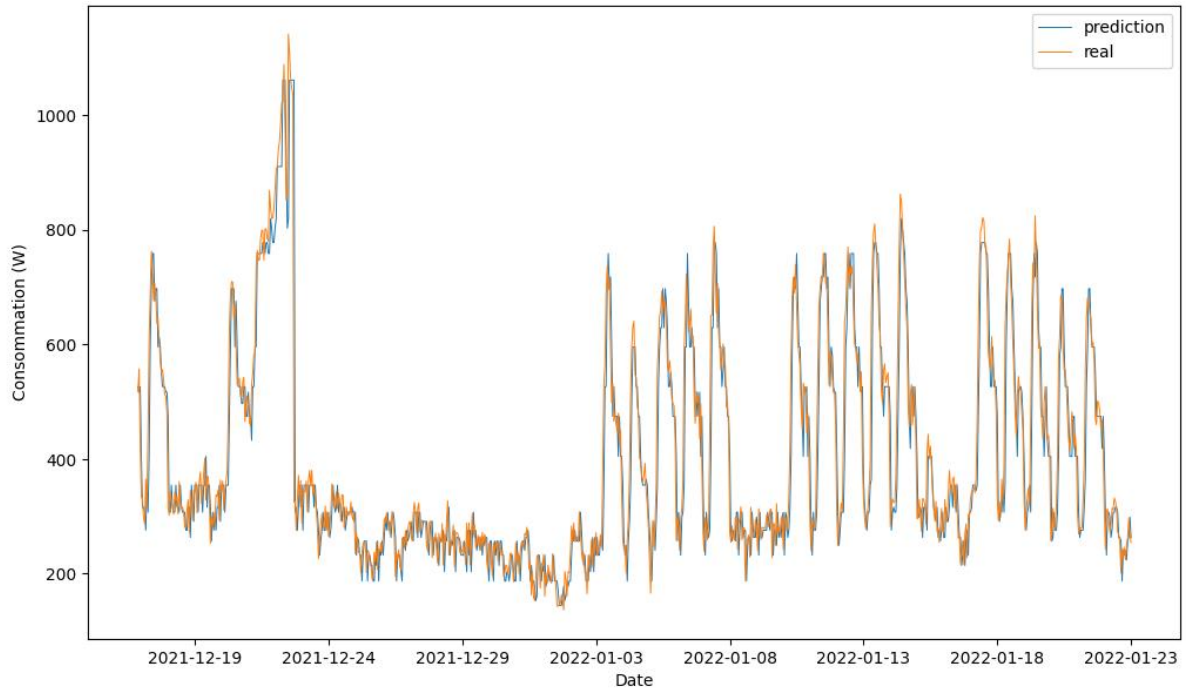


Figure 12 : Random Forest Regressor - échantillon 1 (max_depth=4, n_estimators=5)

Tout d'abord, il est important de souligner que chaque méthode a été testée un faible nombre de fois (4 fois), et qu'il faudrait idéalement les tester un grand nombre de fois afin d'obtenir une valeur moyenne plus précise. Voici les résultats de précision obtenus à partir du RMSE:

- LSTM : 94,94%

- Random Forest Regressor : 94,86%

Pour cet échantillon, on constate une précision légèrement plus faible pour le modèle de forêts aléatoires. En revanche, si on compare les temps de calcul, une simulation dure environ 20 fois plus longtemps pour le modèle LSTM (1,6 secondes contre 32 secondes). Sur un grand nombre de simulations et en fonction de l'usage, un modèle de forêt aléatoire semble donc plus intéressant pour une précision quasi-équivalente.

Prenons comme exemple le deuxième échantillon, le besoin en chauffage (en W) d'un bâtiment sur un an, obtenu via une modélisation sur le logiciel de modélisation thermique Pleiades. Cette fois-ci, le jeu de données est composé de 8760 données.

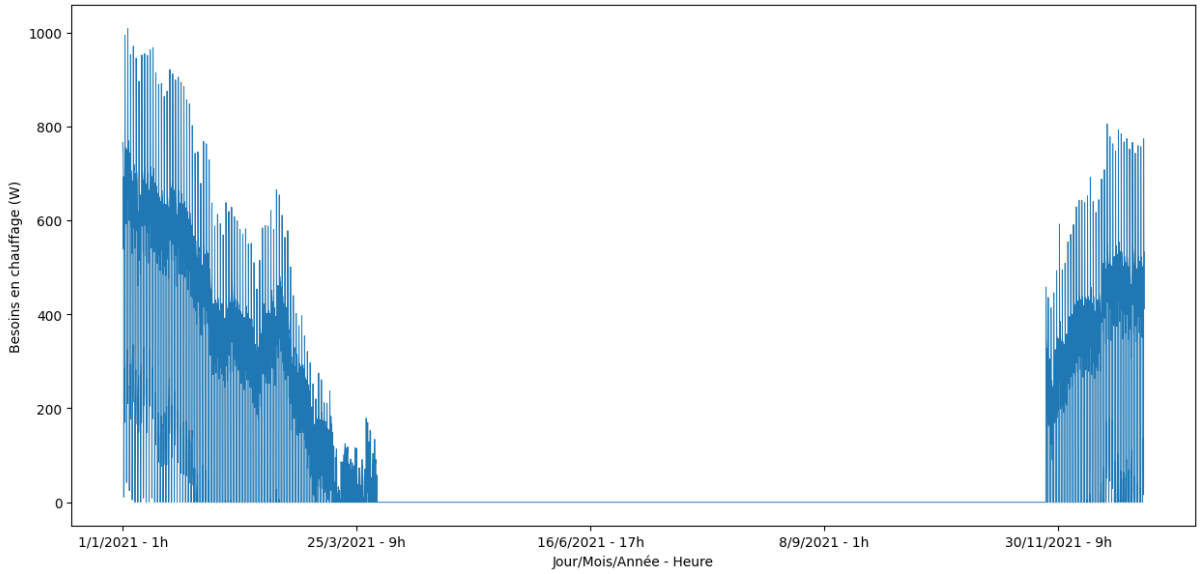


Figure 13 : échantillon 2 - Besoins en chauffage (en W) d'une maquette Pleiades réalisée en 4ème année du cycle ingénieur bâtiment

Voici les résultats obtenus pour les deux méthodes, LSTM et Random Forest Regression :

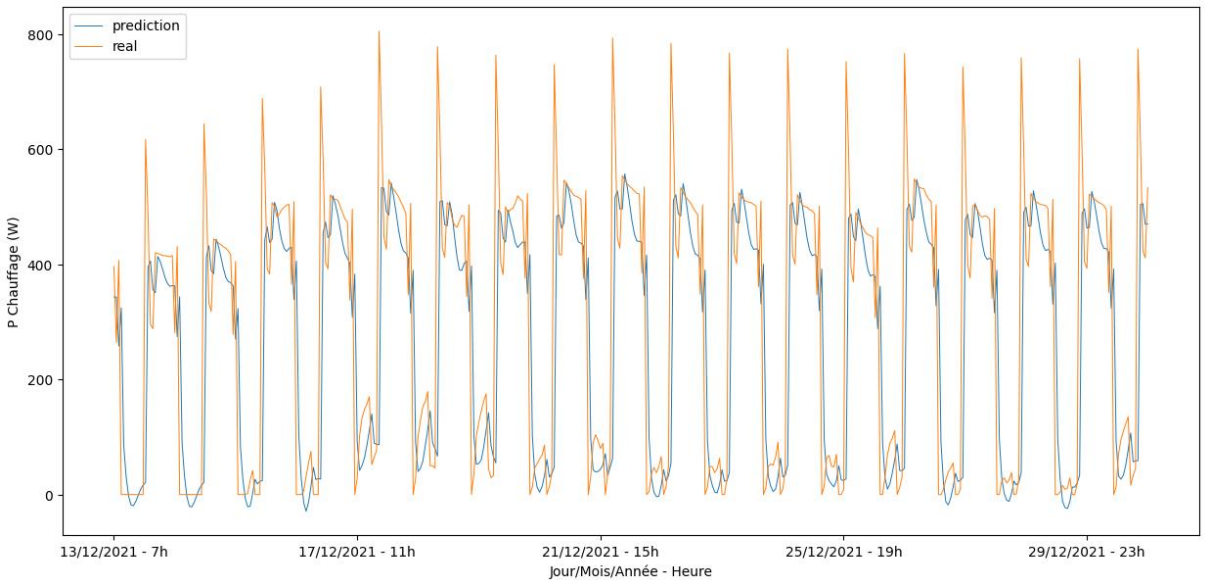


Figure 14 : LSTM - échantillon 2 (epoch=10)

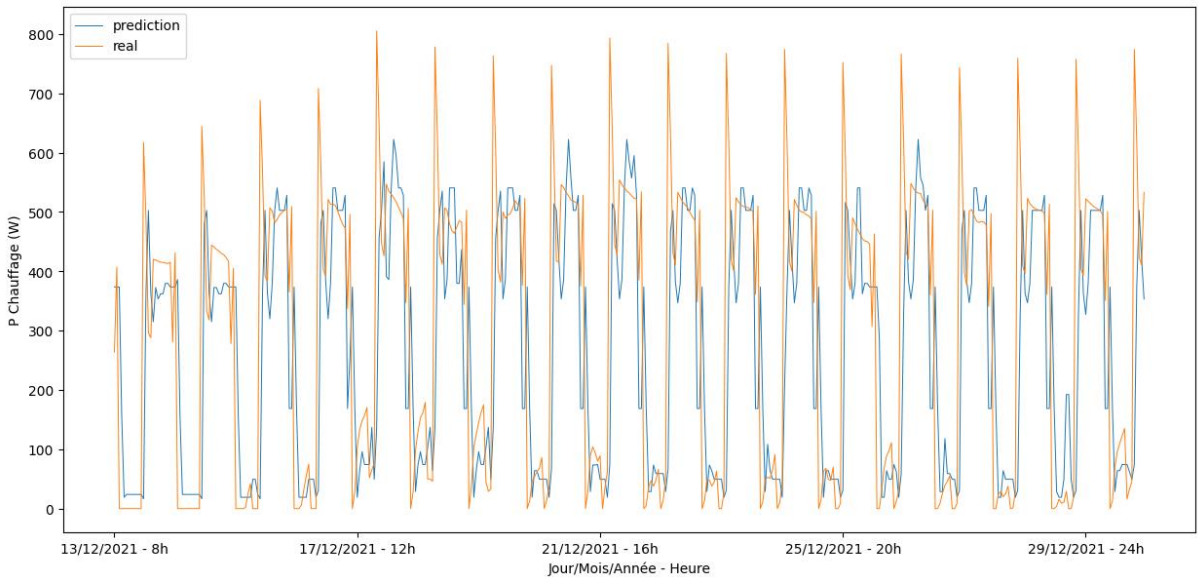


Figure 15 : Random Forest Regressor - échantillon 2 (max_depth=4, n_estimators=5)

D'après nos premiers résultats, on constate une précision nettement différente en comparaison avec ceux obtenus lors des premières simulations. La précision est de 81,90% pour le réseau de neurones (LSTM) et de 90,04% pour l'arbre de forêts (Random Forest Regression).

La différence de précision sur cet échantillon est bien plus importante que précédemment et la tendance s'est inversée. Etudions maintenant les paramètres définis lors de la définition des méthodes afin de tenter d'améliorer leur précision.

Pour la méthode LSTM, on cherche à augmenter le nombre d'entraînement de notre modèle à travers les échantillons, en modifiant la valeur du paramètre *epoch (=10)*. On procède par tâtonnement afin de trouver le paramètre optimal (en fonction de la précision et du temps), même si cette méthode prend du temps.

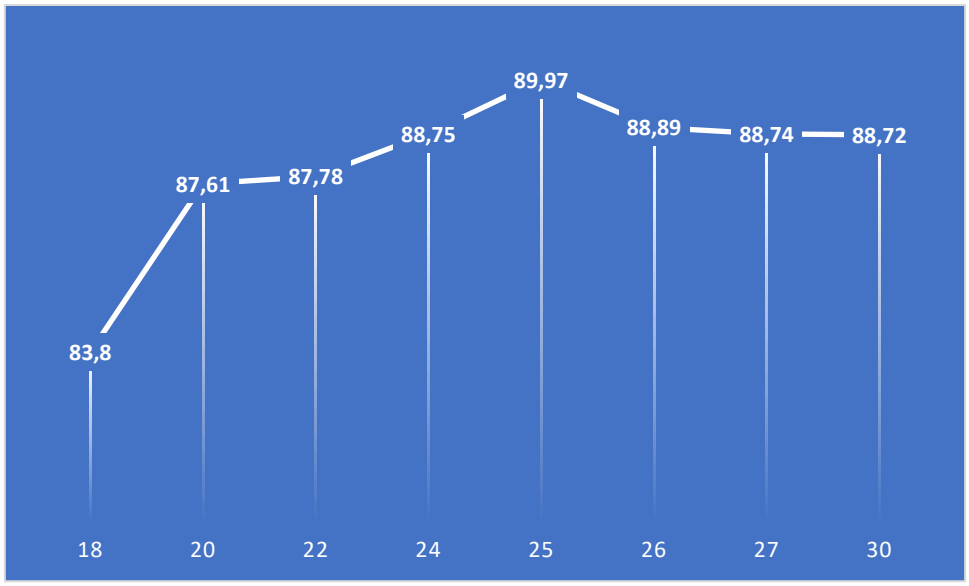


Figure 16 : Précision de la méthode LSTM en fonction du paramètre "epoch"

Après quelques simulations, on constate comme attendu que si la valeur d'*epoch diminue, (ex : epoch=5)*, la précision diminue aussi. La précision augmente uniquement lorsqu'on augmente considérablement la valeur, soit à partir du double. On peut observer sur le graphique ci-dessus la précision en fonction du nombre d'entraînement effectué dans chaque lot. On en déduit qu'il est préférable de prendre une valeur de 25 afin d'augmenter la précision de 8% et atteindre une valeur de 89,97%. Globalement, on constate qu'à partir de ce seuil (*epoch=25*), la précision diminue. On peut en déduire qu'à partir d'un certain nombre d'entraînement sur un échantillon, le modèle perd en précision et se fatigue.

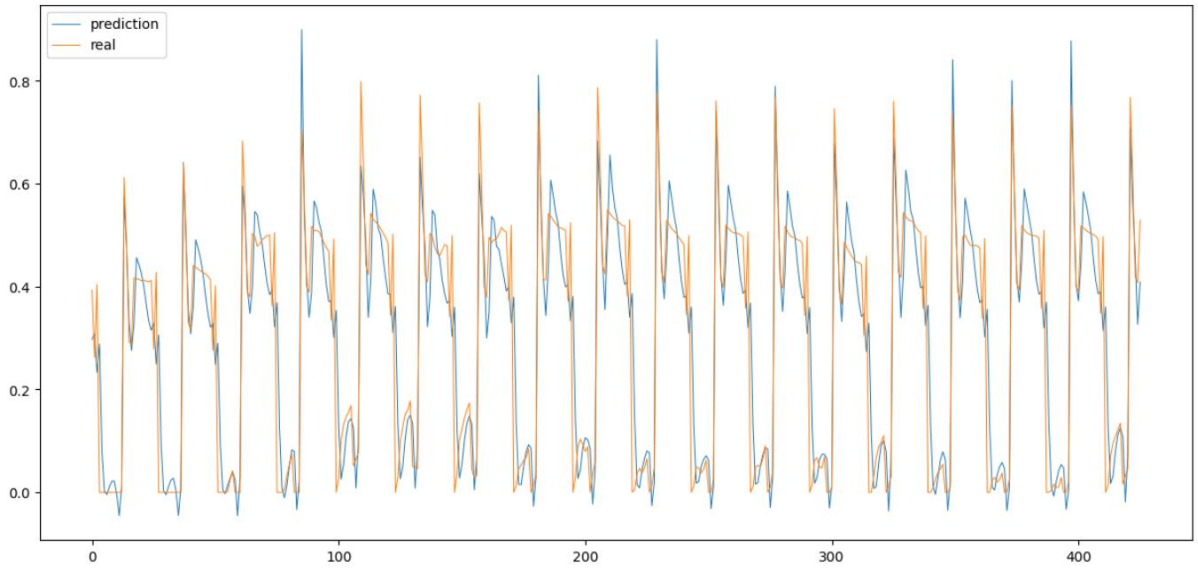


Figure 17 : LSTM – échantillon 2 (epoch=25)

Pour la méthode Random Forest Regression, nous allons étudier les paramètres *n_estimators* et *max_depth*, respectivement le nombre d'arbres et la profondeur des arbres. Afin d'étudier chaque

paramètre plus efficacement, il est nécessaire de tester différentes combinaisons de ces paramètres. En faisant varier les paramètres de manière unitaire, on constate que lors de l'augmentation de la profondeur de l'arbre, la précision s'améliore mais la dispersion des valeurs de précision reste assez élevée (d'environ 3% pour toutes les valeurs).

On réalise désormais des essais en augmentant parallèlement le nombre d'arbres utilisés (*n_estimators*) dans le but d'améliorer la précision et aussi la variance de notre méthode. On peut générer la heatmap suivante en testant le modèle avec différents paramètres. Chaque combinaison a été testée 20 fois afin d'avoir la précision moyenne la plus juste.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	80.823487	80.842896	84.859030	86.438054	88.474473	90.818817	88.971322	89.762158	89.948458	91.076048	90.393477	89.939721	90.238651	89.458178
2	80.984031	81.277338	85.080490	87.870547	89.943256	91.426263	92.507251	92.262218	92.447623	93.254110	92.640320	93.607093	91.123452	91.740893
3	81.117817	81.661753	85.403417	88.724831	91.294814	91.338207	92.866121	93.388980	92.852241	93.560510	92.634863	92.388508	92.762288	92.737300
4	81.060072	81.654985	85.649930	89.216427	91.460810	92.730906	92.811337	92.994782	93.439021	92.898759	93.014893	93.809191	93.506772	92.848605
5	81.089059	81.752176	85.278878	89.021738	92.222235	93.194898	92.479883	92.791149	92.897859	93.364703	93.978248	93.887414	93.590632	93.453161
6	81.104006	81.696171	85.535330	89.803554	92.585883	92.859792	93.272936	94.234881	93.450650	93.711000	93.452457	93.717658	93.604100	92.964418
7	81.099654	81.758883	85.722164	88.998923	92.163780	92.744837	93.395137	93.596981	94.149954	93.571984	94.050793	93.739263	93.939959	93.208024
8	81.128848	81.787952	86.102202	89.920270	91.700417	92.757002	93.492437	93.878290	94.153707	94.166305	93.924234	94.014353	94.029446	94.118045
9	81.114102	81.828160	86.079863	89.934036	92.187365	93.386837	93.672219	93.906550	94.538952	93.995056	93.672986	93.871563	93.623997	94.052146
10	81.112932	81.891929	85.924013	89.669052	92.539377	93.210958	93.796525	93.657386	94.216550	93.413090	93.795353	94.041718	93.896785	94.077489
11	81.108541	81.890797	85.370818	89.846553	92.194528	92.914253	94.132087	93.677751	94.117273	93.719288	93.922145	94.280226	94.002309	94.163167
12	81.112229	81.764920	85.704421	90.066545	92.575309	93.262732	93.742306	93.522115	93.730984	93.836464	94.238336	94.721138	94.001180	94.055555
13	81.118687	81.810604	85.582394	89.868925	92.240370	93.252323	94.180695	93.909430	94.116089	94.314886	93.563816	94.201641	94.071072	94.223640
14	81.124051	81.820046	85.422577	90.011245	92.425064	93.614070	93.862172	93.972857	93.473878	94.308632	94.015764	94.650102	94.253836	94.242536

Figure 18 : Précision du modèle Random Forest Regression en fonction des paramètres max_depth (lignes) et n_estimators (colonnes) pour l'échantillon 2

En lançant différentes simulations, on constate que pour des paramètres différents on obtient des précisions équivalentes à partir d'un certain seuil. On prend donc la combinaison *n_estimators=12*, *max_depth=12* pour obtenir une précision de 94,72%, disponible à partir de la figure suivante.

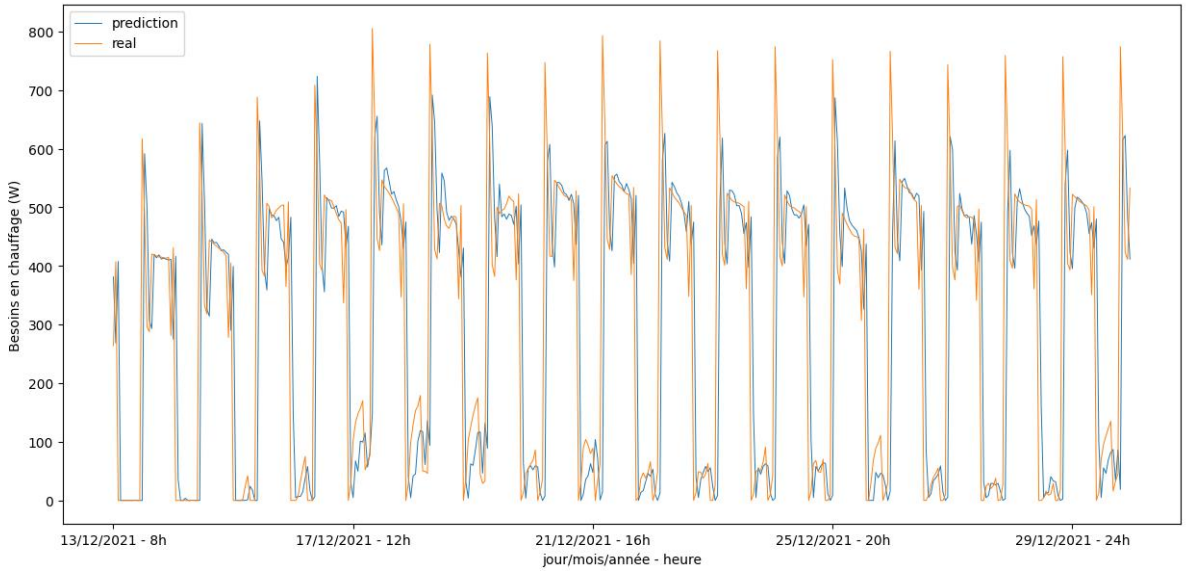


Figure 19 : Random Forest Regressor - échantillon 2 (max_depth=12, n_estimators=12)

D’après notre étude sur cet échantillon, on peut conclure qu’il est important d’adapter l’initialisation des méthodes en fonction des paramètres étudiés. On en revient aux observations émises sur l’échantillon 1 concernant le temps de calcul. Afin d’effectuer une étude de sensibilité des différents paramètres, la méthode LSTM requiert un long temps de calcul (1h voir plus). En faisant une étude plus poussée (20 simulations pour chaque combinaison) sur les paramètres de la méthode Random Forest, le temps de calcul reste inférieur à 10 minutes.

Enfin, étudions le dernier échantillon, issu d’un article scientifique[19], à travers lequel on exploite les données de consommation (en W) d'une zone de la ville de Tétouan (Maroc), pas de temps de 10 min. Ce jeu de données comporte un total de 52416 données.

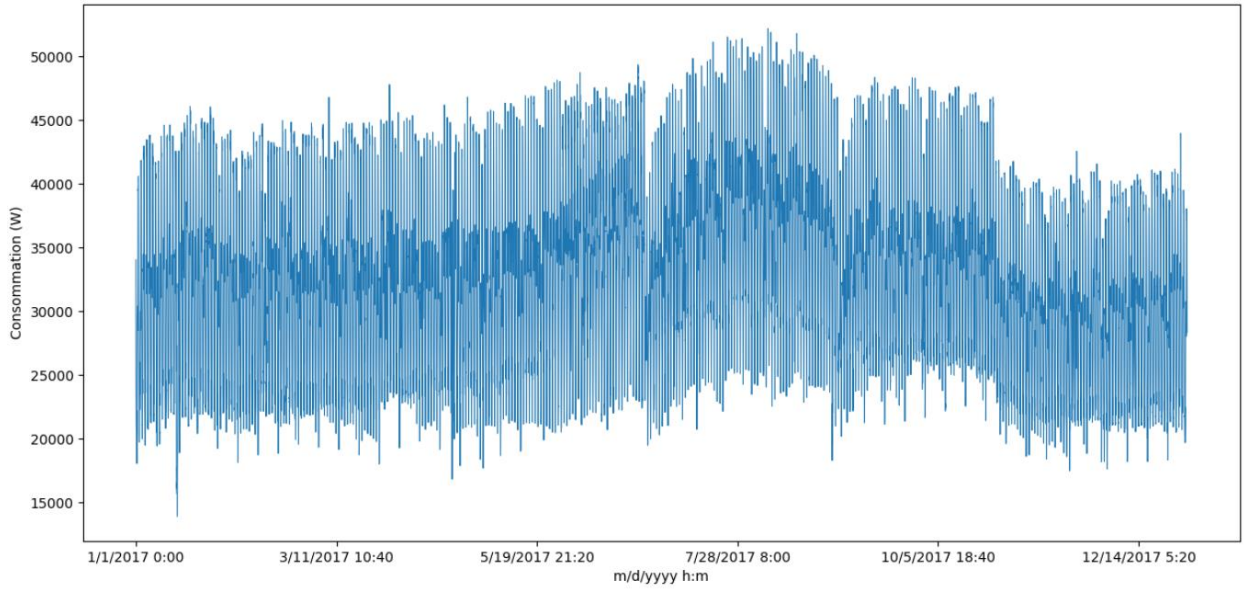


Figure 20 : échantillon 3 - avant lissage et mise à l'échelle

Pour l’initialisation des méthodes, on étudie ici une valeur d’epoch égale à 10 pour la méthode LSTM, et un nombre d’arbres égal à 12 avec une profondeur de 12 pour la méthode Random Forest.

[19] https://doi.org/10.1109/irsec.2018.8703007

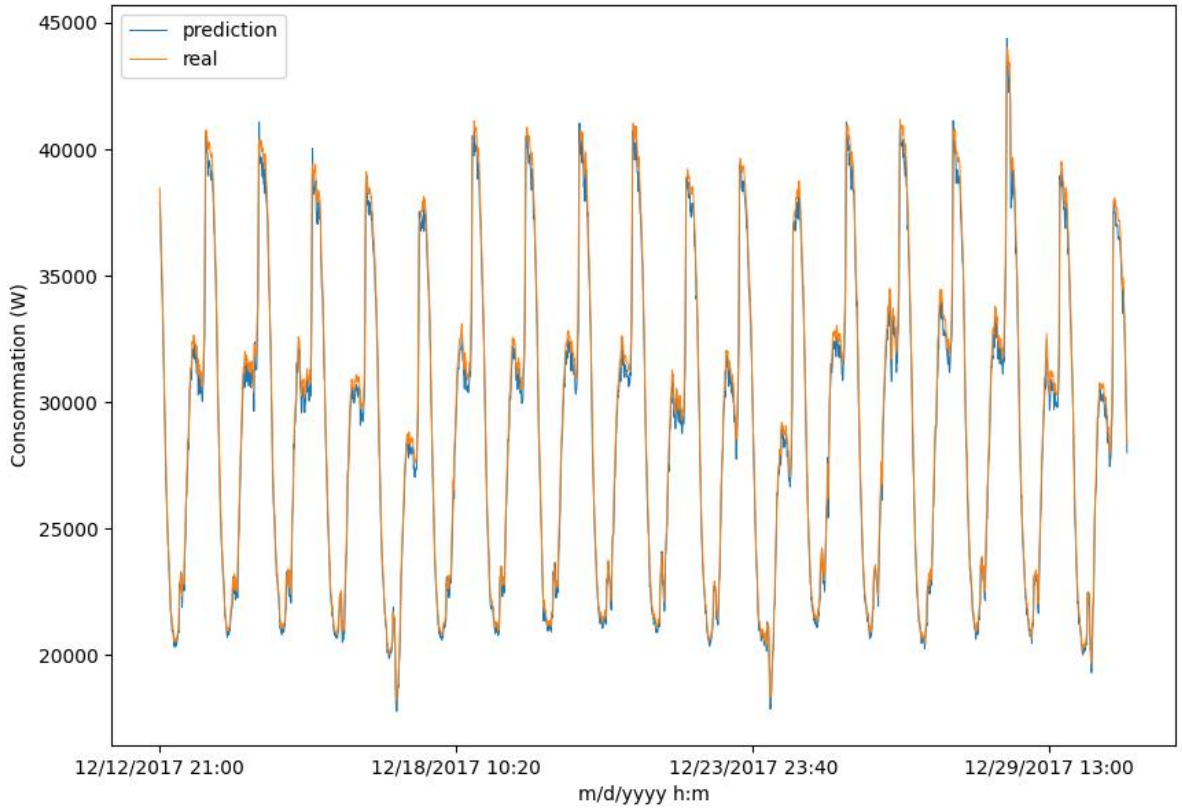


Figure 21 : LSTM – échantillon 3 (epoch=10)

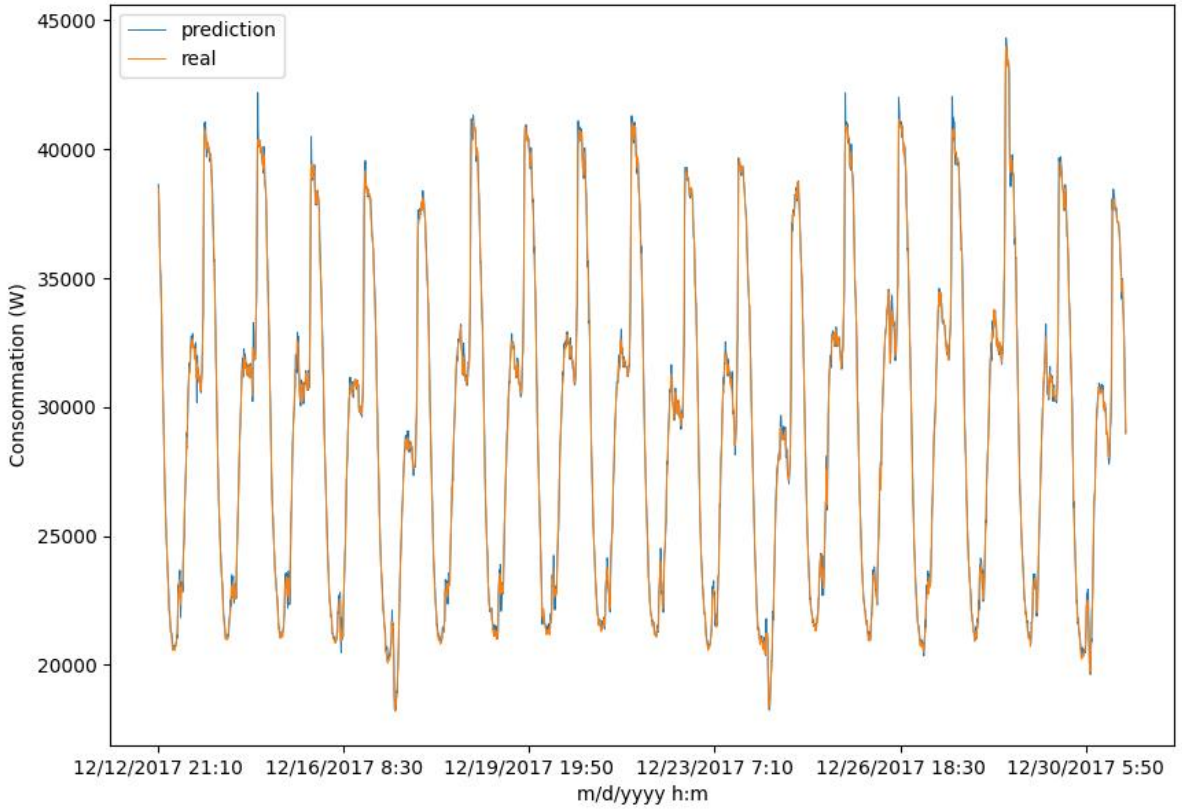


Figure 22 : Random Forest Regressor - échantillon 3 (max_depth=12, n_estimators=12)

Pour les deux méthodes, on obtient des précisions proches, avec 98,56% de précision pour la LSTM et 99,16% pour la Random Forest Regression.

Nos résultats sont cohérents avec ceux observés dans l'article, à travers lequel il était conclu que le modèle Random Forest était celui qui affichait les plus petites erreurs de prédiction. Aussi, du fait de la grande taille du jeu de données, notre précision est impactée positivement.

Conclusion

Tout d'abord, nous nous sommes rendu compte, à travers nos recherches bibliographiques et notre travail de programmation des différents modèles, que le Machine Learning est un domaine complexe qui est doté de nombreuses spécificités. En effet, un grand nombre d'algorithmes suivant des modèles d'apprentissages différents existent, il est alors primordial de fixer des objectifs afin de déterminer la méthode la plus adaptée.

Ensuite, le Machine Learning est un outil qui est pratique et qui peut s'appliquer dans le domaine de l'énergie car il permet d'optimiser et d'anticiper des dysfonctionnements de différents systèmes (PV, ou éoliennes par exemple).

À la suite des nombreuses simulations de prédictions de consommations d'énergie grâce aux deux méthodes choisies, Random Forest Regression et LSTM, il a été montré qu'elles ont chacune une efficacité différente, selon le type de données d'entrée (taille de l'échantillon, continuité des données) et l'initialisation des paramètres (nombre de fois ou l'échantillon d'entrainement est parcouru, profondeur maximale des arbres de décision et nombre d'arbres de décisions) qui leur sont propres.

Enfin, la méthode de forêt d'arbres décisionnels régressifs s'est montrée plus efficace que les réseaux de neurones LSTM, en termes de précision et de durée d'exécution de l'algorithme. Néanmoins, il est nécessaire d'adapter l'initialisation des méthodes en fonction de nos jeux de données.

Bibliographie

An overview on clustering methods. (2012). IOSR Journal of Engineering. https://arxiv.org/ftp/arxiv/papers/1205/1205.1117.pdf

Hsu, D. (2015). Comparison of integrated clustering methods for accurate and stable prediction of building energy consumption data. Applied Energy, 160, 153-163. https://doi.org/10.1016/j.apenergy.2015.08.126

Kim, T. Y. & Cho, S. B. (2019). Predicting residential energy consumption using CNN-LSTM neural networks. Energy, 182, 72-81. https://doi.org/10.1016/j.energy.2019.05.230

Machine Learning Algorithms -A Review. (2018). International Journal of Science and Research. https://www.researchgate.net/publication/344717762_Machine_Learning_Algorithms_-A_Review

Prédiction du Churn Rate par le Machine Learning dans le secteur des M&A. Application au sein de KPMG. (2020). Thesis. https://www.researchgate.net/publication/348232336_Prediction_du_Churn_Rate_Par_le_Machine_Learning_dans_le_secteur_des_MA_Application_au_sein_de_KPMG

Salam, A. & Hibaoui, A. E. (2018). Comparison of Machine Learning Algorithms for the Power Consumption Prediction: - Case Study of Tetouan city –. 2018 6th International Renewable and Sustainable Energy Conference (IRSEC). https://doi.org/10.1109/irsec.2018.8703007

S. Baghaee and I. Ulusoy, "User comfort and energy efficiency in HVAC systems by Q-learning," 2018 26th Signal Processing and Communications Applications Conference (SIU), 2018, pp. 1-4. https://doi.org/10.1109/SIU.2018.8404287

Sutskever, I., Martens, J. & Hinton, G. E. (2011). Generating Text with Recurrent Neural Networks. International Conference on Machine Learning, 1017-1024. https://icml.cc/2011/papers/524_icmlpaper.pdf

Wang, Z., Wang, Y., Zeng, R., Srinivasan, R. S. & Ahrentzen, S. (2018b). Random Forest based hourly building energy prediction. Energy and Buildings, 171, 11-25. https://doi.org/10.1016/j.enbuild.2018.04.008