

Modélisation des séries temporelles



Sommaire

| | |
|---|-----------|
| Introduction | 3 |
| I. ARIMA, SARIMA, VARIMAX | 4 |
| A. ARMA | 4 |
| B. ARIMA | 5 |
| C. SARIMA | 5 |
| D. VARIMAX | 6 |
| II. Prophet | 6 |
| III. XGboost | 7 |
| IV. Comparaison et choix du modèle | 8 |
| Bibliographie | 10 |

Introduction

Une time serie, ou série temporelle (ou chronologique), est une séquence dans laquelle une métrique est enregistrée à intervalle de temps réguliers [1]. D'un point de vue mathématique, on dit qu'une série temporelle est une suite d'observations x_1, x_2, \dots, x_n indexée par le temps. On supposera qu'il s'agit d'une réalisation d'un processus X , c'est-à-dire d'une suite $\{X_i\}$ de variables aléatoires [2]. En machine learning, les séries temporelles sont les seuls problèmes dont la composante de temps joue un rôle crucial dans les prédictions du modèle [3].

Une série temporelle est composée de plusieurs éléments [2][4] :

- tendance : c'est la tendance générale d'une série à augmenter, diminuer ou stagner à long terme
- saisonnalité : fluctuations dans la série temporelle qui se répètent de manière régulière à différents intervalles de temps (saisonnalité annuelle, journalière...) ;
- résidu (ou bruit) : fluctuations aléatoires qui ne suivent aucun pattern.

Mathématiquement, une série temporelle est décrite par [5] :

$$Y_t = T_t + S_t + \epsilon_t$$

Où Y_t représente la série temporelle, T_t représente la tendance, S_t la saisonnalité et ϵ_t le terme d'erreur.

Après avoir analysé une série temporelle, nous cherchons à la modéliser dans le but de pouvoir prédire son évolution [1]. Cette tâche de prédiction, également appelée forecasting, est une tâche de data science qui peut être centrale dans beaucoup d'organisations et la qualité de ces prédictions est un enjeu central dans l'anticipation de gestion des ressources d'une entreprise. La compétence d'analyse et de modélisation d'une série temporelle est donc fortement recherchée sur le marché du travail, notamment dans les domaines commerciaux.

Pour mener à bien cette tâche de forecasting, il est important de choisir un modèle adapté à la situation et aux données. L'objet de ce rapport est de faire un état de l'art des modèles les plus répandus et de donner des critères qui permettent de les différencier et de choisir le plus adapté selon le contexte. Les modèles étudiés seront ARIMA, SARIMA, VARIMAX puis Prophet et XGboost.

I. ARIMA, SARIMA, VARIMAX

A. ARMA

ARMA, pour AutoRégressive Moving Average, est un algorithme de régression basé sur l'idée que les informations contenues dans les valeurs passées de la série temporelle peuvent être utilisées pour prédire les valeurs futures [1]. Les régresseurs utilisés dans le modèle sont donc les états passés de la variable à prédire.

ARMA est une combinaison de deux modèles [5] :

- AR : autoregressive, signifie que les prédictions sont basées sur les valeurs passées.
- MA : moving average, signifie que les prédictions sont basées sur les erreurs de prédiction passées.

Pour utiliser ARMA, on suppose que la série temporelle est stationnaire. En effet, ARMA étant un algorithme de régression, il est nécessaire de vérifier que les régresseurs sont indépendants les uns des autres. Dans notre cas, les régresseurs sont les états passés de notre variable à prédire. S'il existe une colinéarité entre ces états passés, la série n'est pas stationnaire et il n'est pas possible d'appliquer l'algorithme ARMA. On considère qu'une série est stationnaire lorsque ses propriétés statistiques telles que la moyenne et la variance ne dépendent pas du temps [5]. Cela ne signifie pas que la série temporelle ne varie pas, mais qu'elle ne varie pas en fonction du temps [6].

Les paramètres du modèle ARMA sont :

- p : terme de AR, qui détermine le nombre d'états passés de la série temporelle nécessaires pour réaliser des prédictions ;
- q : terme de MA, qui détermine le nombre d'erreurs passées à prendre en compte pour réaliser les prédictions.

Pour déterminer p , on se base sur le graphique d'auto-corrélation partielle [3] : on retient $p = k$ le nombre d'états passés qui ont des auto-corrélations partielles significatives, sans compter la première autocorrélation partielle qui correspond à la corrélation entre l'état actuel et lui-même et qui est toujours égal à 1.

Pour déterminer q , on se base sur le graphique d'auto-corrélation [3] : on retient $q =$ le nombre d'états passés qui ont des auto-corrélations significatives, toujours sans tenir compte de la première autocorrélation.

Si on constate une tendance dans la série temporelle et donc qu'elle n'est pas stationnaire, comme c'est souvent le cas, il est nécessaire de faire évoluer le modèle pour faire disparaître cette tendance.

B. ARIMA

ARIMA est un modèle ARMA auquel est ajouté le terme I pour Integrated. Il permet de généraliser le modèle ARMA en incluant le cas des séries non stationnaires en ajoutant une étape avant le forecast dans le but d'enlever la tendance de la série [5].

Pour faire disparaître la tendance, il existe différentes techniques dont la plus répandue est la différenciation. La différenciation consiste à calculer la différence entre une observation et sa valeur antérieure [1].

Le paramètre pour I est d et détermine le nombre de différenciations nécessaire pour obtenir une série dépourvue de tendance. Dans le cadre de l'implémentation informatique, ARIMA nécessite toujours les trois paramètres p, q et d. Si on souhaite implémenter un modèle ARMA, $d = 0$.

ARIMA prend en compte les séries temporelles présentant une tendance mais pas celles qui présentent une saisonnalité. Pour cela, il faut à nouveau faire évoluer le modèle.

C. SARIMA

SARIMA généralise le modèle ARIMA en ajoutant le terme S pour Seasonal ARIMA et ajoute une étape supplémentaire pour supprimer la saisonnalité, rendant ainsi la série stationnaire, avant d'appliquer le modèle ARMA et de réaliser le forecast [3].

Pour supprimer la saisonnalité, SARIMA utilise la différenciation [3] en effectuant la différence entre la valeur actuelle Y_t et la valeur Y_{t-s} où s représente le nombre de périodes appartenant à la saisonnalité identifiée. Par exemple, si notre série est indexée en mois et qu'on a détecté une saisonnalité annuelle, $s = 12$. Si Y_t est la valeur de janvier 2020, on la soustrait avec janvier 2019.

Une autre technique consisterait à utiliser la moyenne mobile centrée. Il s'agit d'une technique qui consiste à calculer, pour chaque valeur, la moyenne des valeurs environnantes en donnant un poids égal aux données précédentes et suivantes. On détermine le nombre de valeurs à prendre en compte dans la moyenne mobile en définissant une fenêtre. Lorsque la fenêtre est égal à s, on peut lisser la série et faire disparaître la saisonnalité.

Pour implémenter SARIMA informatiquement, il faut configurer les paramètres p, d, q comme pour un modèle ARIMA et ajouter les termes propres à SARIMA :

- P : ordre saisonnier de l'auto régression (nombre de lags saisonniers à prendre en compte)
- D : ordre saisonnier de différenciation (nombre de différenciation saisonnières pour stationnariser la série)
- Q : ordre saisonnier de la variable mobile (nombre de termes de termes d'erreur saisonniers à prendre en compte)

- m : périodicité saisonnière, nombre d'observations dans une saison complète (si on a une saisonnalité annuelle avec des observations au mois, on a 12 observations dans une saison donc $m = 12$).

Dans le cas des modèles ARMA, ARIMA et SARIMA, il est impossible de prendre en compte d'autres variables, on parle de modèle univarié. Toutefois, il est possible de faire évoluer ces modèles afin de prendre en compte des variables extérieures à la série appelées variables exogènes et de considérer plusieurs séries temporelles à la fois.

D. VARIMAX

VARIMAX mixe les modèles ARIMA, VAR et des variables exogènes [11]. Le terme V ajouté signifie Vector et introduit l'idée que, contrairement aux modèles précédents, il s'agit d'un modèle multivarié [7]. Les modèles VARMA incluent des moyennes mobiles pour chaque variable [11]. Le terme X représente les variables exogènes. Ainsi, VARIMAX prend en compte les états passés de plusieurs séries temporelles et des variables exogènes pour effectuer ses prédictions. Ces variables sont indépendantes de la série temporelle principale mais peuvent influencer son comportement.

Les approches statistiques des modèles présentés ci-dessous s'adressent à un public expérimenté et formé en statistiques ou en data sciences qui maîtrise les notions mathématiques sous-jacentes. Le fine tuning des modèles peut s'avérer fastidieux. Ces modèles sont plus efficaces sur des données dont la tendance et la saisonnalité sont bien définies et seront moins pertinents pour des séries temporelles qui présentent des patterns plus complexes. Leur approche est adaptée aux datasets de petite taille.

II. Prophet

Prophet est un algorithme open-source développé par Facebook afin de modéliser les séries temporelles, notamment dans le domaine commercial [6][8]. Pour prophet, la série temporelle se décompose de la manière suivante :

$$y_t = g(t) + s(t) + h(t) + \varepsilon_t$$

Où $g(t)$ représente la tendance, $s(t)$ représente la saisonnalité, $h(t)$ représente les vacances et jours fériés, et représente le terme d'erreur ou les fluctuations aléatoires non expliquées par le modèle prophet. On suppose que l'erreur suit une distribution normale centrée sur 0.

L'objectif premier de Prophet est de fournir un outil de forecasting flexible, facile à utiliser et à ajuster aux données. Ainsi, Prophet ne nécessite pas de preprocessing ou de transformation des données et est facile à implémenter informatiquement, accessible via des librairies python et R. Il est donc accessible à un public moins expérimenté en statistiques

ou en datascience, par exemple des commerciaux, qui souhaitent modéliser de façon simple et rapide les séries temporelles de leur entreprise. Prophet ne permet pas de réaliser de modélisation multivariée. Par ailleurs, il est particulièrement efficace sur les données commerciales mais peut s'avérer moins performant dans d'autres secteurs d'activité. Il s'adapte bien aux datasets moyens à grands et gère efficacement les valeurs manquantes en les remplaçant automatiquement.

III. XGboost

XGboost (Extreme Gradient Boosting) est une implémentation du gradient boosting pour les problèmes de classification et de régression [9]. C'est un outil puissant qui utilise des algorithmes d'arbres de décision dont les nouveaux arbres corrigent les erreurs des précédents. XGboost implémente un modèle de régression basé sur une ou plusieurs variables numériques pour prédire des variables numériques. Etant donné que dans notre cas les variables qui permettent la prédiction sont la date, il faut transformer cette date en valeurs numériques.

XGboost peut utiliser des approches multivariées et ainsi prendre en compte plusieurs séries temporelles dans ses prédictions. Ce modèle nous permet également d'étudier la feature importance du modèle.



Ce modèle particulièrement complexe demande donc de réaliser une préparation des données et un bon feature engineering qui nécessite une bonne connaissance des données et des compétences avancées en machine learning. Il est donc plutôt adapté à un public expérimenté et formé en data sciences. Le temps de calcul peut être long et l'optimisation des hyperparamètres peut être fastidieuse. En revanche, il est très efficace sur des datasets importants et des séries temporelles complexes [10].

IV. Comparaison et choix du modèle



Notre contexte professionnel est le suivant :

“Vous travaillez pour un fournisseur d’électricité. Votre manager vous demande pour la région des Hauts de France d’être capable de prédire à court terme (sur la semaine qui arrive) la consommation électrique. Il vous indique qu’il pourrait être intéressant d’utiliser d’autres variables exogènes comme la température.”

Le tableau suivant permet une comparaison des modèles en fonction de différents critères :

| | VARIMAX |  |  |
|--|---|--|---|
| Public | Expérimenté | Peu expérimenté | Expérimenté |
| Approche | Statistique | Commerciale | Datascience |
| Prise en compte des variables exogènes | Oui | Oui | Oui |
| Temps de calcul | + | - | ++ |
| Finetuning | long | court | long |
| Secteur d'application | tous domaines | commercial | tous domaines |
| Taille des datasets | petits | moyens à grands | grands |
| Gestion des données manquantes | Difficile | Remplissage automatique des valeurs manquantes | Peut gérer tous types de données |
| Données | tendance et saisonnalité bien définies | tendance et saisonnalité bien définies | séries temporelles complexes avec structures linéaires |
| Type de modèle | Multivarié | Univarié | Multivarié |
| Interprétabilité | Complexe, nécessité de connaître les concepts mathématiques | Facilement interprétable, mais boîte noire | Complexe, nécessite une expertise en data science |

Le tableau suivant permet de situer chaque modèle au regard des exigences du contexte :

| | VARIMAX |  |  |
|---|---------|--|---|
| Intégrer la température | ✓ | ✓ | ✓ |
| Grand dataset | ✗ | ✓ | ✓ |
| Saisonnalité bien définie | ✓ | ✓ | ✓ |
| Temps de fine tuning et feature engineering court | ✗ | ✓ | ✗ |
| Court terme | ✗ | ✓ | ✓ |
| Approche data science | ✗ | ✗ | ✓ |
| Interprétabilité rapide | ✗ | ✓ | ✗ |

Comme indiqué dans le tableau précédent, la taille importante de notre dataset est compatible avec l'utilisation d'XGboost ou de Prophet de façon optimale. De plus, nous avons besoin d'une prévision à court terme et ces modèles sont particulièrement efficaces dans ces situations, alors que VARIMAX peut s'avérer plus performant pour les prévisions à long terme. Toutefois, XGboost et VARIMAX demandent un travail important de feature engineering et de fine tuning que ne nous permet pas forcément la deadline du projet. Par ailleurs, nos données montrent une saisonnalité bien définie ce qui permet à un modèle VARIMAX ou Prophet d'être performant et peut nous éviter de rechercher des patterns plus complexes avec XGboost. Ainsi, chaque modèle possède des avantages et des inconvénients au regard de notre contexte professionnel. Il serait donc intéressant de

pouvoir les explorer, particulièrement XGboost et Prophet, afin de voir lequel nous offre la meilleure prédiction.

Bibliographie

1. Prabhakaran, Selva. "ARIMA Model - Complete Guide to Time Series Forecasting in Python." *Machine Learning Plus*. Consulté le 18 septembre 2023.
<https://www.machinelearningplus.com/time-series/arima-model-time-series-forecasting-python/>
- Selva Prabhakaran est un auteur reconnu dans le domaine de l'apprentissage automatique. Machine Learning Plus est un site Web bien établi comptant plus de 4 millions de lectures.
2. Monbet, Valérie. "Modélisation de séries temporelles." Université de Rennes 1, 2011.
- Valérie Monbet est une enseignante-chercheuse spécialisée en statistiques à l'Université de Rennes 1.
3. Gong, Destin. "Time Series Analysis - ARMA, ARIMA, SARIMA." *Visual Design*. Consulté le 18 septembre 2023.
<https://www.visual-design.net/post/time-series-analysis-arma-arima-sarima>
- Destin Gong, data scientist néo-zélandais, publie un article sur site visual-design, papier non scientifique mais à visée informative pour un public général.
4. Adhikari, Ratnadip, et Agrawal, R. K. "An Introductory Study on Time Series: Modeling and Forecasting." 2013.
- Ratnadip Adhikari est un docteur en mathématiques spécialisé en intelligence artificielle et machine learning.
5. Goude, Yannig. "Time Series Forecasting." Université Paris-Sud. Consulté le 18 septembre 2023.
https://www.imo.universite-paris-saclay.fr/~yannig.goude/Materials/ProjetMLF/time_series.html
- Yannig Goude est professeur associé en mathématiques et consultant pour EDF.
6. Kutzkov, Konstantin. "ARIMA vs Prophet vs LSTM for Time Series Prediction." *Neptune.ai*, 2023.
<https://neptune.ai/blog/arima-vs-prophet-vs-lstm>
- Konstantin Kutzkov est docteur en sciences informatiques et auteur de plusieurs publications en exploration de données et apprentissage automatique.
7. Verma, Yugesh. "A Guide to VARMA with Auto ARIMA in Time Series Modelling." *Analytics India Magazine*, 28 septembre 2021.
<https://analyticsindiamag.com/a-guide-to-varma-with-auto-arima-in-time-series-modelling/>
- Yugesh Verma est un data scientist senior.

8. Menculini, Lorenzo; Marini, Andrea; Proietti, Massimiliano; Garinei, Alberto; Bozza, Alessio; Moretti, Cecilia; Marconi, Marcello. "Comparing Prophet and Deep Learning to ARIMA in Forecasting Wholesale Food Prices." 2021

- Papier scientifique rédigé par le département des sciences de l'ingénierie de l'Université de Rome.

9. Brownlee, Jason. "How to Use XGBoost for Time Series Forecasting." *Machine Learning Mastery*, 5 août 2020.

(<https://machinelearningmastery.com/xgboost-for-time-series-forecasting/>)

- Jason Brownlee, diplômé d'un doctorat en informatique et intelligence artificielle, est un ingénieur logiciel et chercheur scientifique en intelligence artificielle, en modélisation prédictive, en optimisation stochastique et en calcul haute performance.

10. Filho, Mario. "Multiple Time Series Forecasting With XGBoost In Python." *ForecastEgy*, 28 février 2023.

(<https://forecastegy.com/posts/multiple-time-series-forecasting-with-xgboost-in-python/>)

- Mario Filho dispose de 10 ans d'expérience en data science et machine learning et a créé son site forecastegy qui met à disposition des articles sur différents sujets du machine learning.

11. Pruethsan Sutthichaimethee. "VARIMAX MODEL TO FORECAST THE EMISSION OF CARBON DIOXIDE FROM ENERGY CONSUMPTION IN RUBBER AND PETROLEUM INDUSTRIES SECTORS IN THAILAND." *Journal of Ecological Engineering*, 2017.

- Article publié dans une revue scientifique et rédigé par un membre de la faculté d'économie de Chulalongkorn.