

Hiver 2021

17-Mai-2021

**Projet Forage de Données**

**Option 02 :** **Prédiction dans les séries temporelles**

**Encadré par :**

Pr. ADDA Mehdi

**Réalisé par :**

Ayman Chafni

Achouch Mounia

**PLAN :**

Définition des séries temporelles :

Définition : Prévision des séries Temporelles(forecasting)

L'analyse des séries temporelles :

Les méthodes :

ARIMA

PROPHET

La préparation des données et le nettoyage

Le programme :

Les résultats Obtenus et leurs interprétations

Références :

**Définition des séries temporelles :**

Une série temporelle est une série de points de données ordonnés dans le temps qui ajoutent une dépendance d'ordre explicite entre les observations : une dimension temporelle. Dans un ensemble de données d'apprentissage automatique normal, l'ensemble de données est une collection d'observations qui sont traitées de la même manière lorsque l'avenir est prévu. Ainsi, dans les séries temporelles, l'ordre des observations fournit une source d'informations supplémentaires qui doivent être analysées et utilisées dans le processus de prédiction. Elles sont supposées être générées à des intervalles de temps régulièrement espacés (par exemple, température quotidienne), et sont donc appelées séries temporelles régulières. Mais les données n’ont pas besoin d'être à des intervalles de temps réguliers. Dans ce cas, on parle de séries temporelles irrégulières. Dans les séries chronologiques irrégulières, les données suivent une séquence temporelle, mais les mesures peuvent ne pas se produire à des intervalles de temps réguliers. Par exemple, les données peuvent être générées sous forme de rafale ou avec des intervalles de temps variables.

De plus, les séries temporelles peuvent avoir une ou plusieurs variables qui changent avec le temps.

S'il n'y a qu'une seule variable variant dans le temps, on l'appelle Série temporelle univariée. S'il y a plus d'une variable, on parle de série temporelle multivariée.

**Composants :**

* **Niveau :** la ligne de base d’une série temporelle si elle était une ligne droite
* **Tendance :** l’augmentation ou la diminution des séries temporelles en fonction du temps souvent linéaire
* **Saisonnalité :** les comportements répétitifs sur des cycles du temps des séries temporelles
* **Bruit :** la variabilité aléatoire des séries temporelles qui ne peut pas être expliquée par le modèle.

On peut modéliser n’importe quelle série temporelle en tant que :

|  |
| --- |
| **Y = niveau + tendance + saisonnalité + bruit** |

* L’objectif pour une série temporelle est d’atteindre la stationnarité c.à.d. : ses propriétés statistiques : moyenne, variance…etc. ne changent pas en fonction du temps.

**Définition : Prévision des séries Temporelles(forecasting)**

La prévision de séries temporelles utilise des informations concernant les valeurs historiques et les modèles associés pour prédire l'activité future. Cela concerne, souvent, l'analyse des tendances, l'analyse des fluctuations cycliques et les problèmes de saisonnalité.

Les techniques sont utilisées dans de nombreux domaines d'études. Ils prédisent les événements futurs en analysant les tendances du passé et en supposant que les tendances futures resteront similaires aux tendances historiques. Les données sont organisées autour d'horodatages relativement déterministes et, par conséquent, par rapport aux échantillons aléatoires, peuvent contenir des informations supplémentaires que l'on tente d'extraire.

* Les méthodes de séries temporelles conviennent mieux aux prévisions à court terme.
* La prévision des séries temporelles repose sur la disponibilité de suffisamment de données antérieures et sur le fait que les données sont de haute qualité et vraiment représentatives.
* Les méthodes de séries temporelles sont les mieux adaptées aux situations relativement stables. Lorsque des fluctuations importantes sont courantes et que les conditions sous-jacentes sont soumises à des changements extrêmes, les méthodes de séries chronologiques peuvent donner des résultats relativement médiocres.

**L'analyse des séries temporelles :**

L'analyse des séries temporelles consiste à utiliser des méthodes statistiques pour analyser les données et extraire des statistiques et des caractéristiques significatives sur les données. L'analyse des séries temporelles permet d'identifier les tendances, les cycles et les variations saisonnières pour faciliter la prévision d'un événement futur. Les facteurs pertinents pour l'analyse des séries temporelles comprennent la stationnarité, la saisonnalité et l'autocorrélation.

L'analyse des séries temporelles peut être utile pour voir comment une variable donnée évolue dans le temps (alors que le temps lui-même, dans les données des séries temporelles, est souvent la variable indépendante). L'analyse de séries temporelles peut également être utilisée pour examiner comment les changements associés au point de données choisi en se comparent aux changements d'autres variables au cours de la même période.

**Les Méthodes:**

**ARIMA :**

ARIMA ou « **A**uto **R**egressive **I**ntegrated **M**oving **A**verage » est un modèle statistique très simple, qui peut être représenté par la relation suivante :

|  |
| --- |
| **y[t+1] = (c + a0\*y[t] + a1\*y[t-1] +…+ ap\*y[t-p]) +**  **(e[t] + b1\*e[t-1] + b2\*e[t-2] +…+ bq\*e[t-q])** |

Et comme son nom l’indique, ARIMA se caractérise par trois caractéristiques représentées par trois composantes comme montrée dans la relation ci-dessus, c’est un modèle :

* Autorégressif : La régression s’effectue par rapport aux valeurs passées de y, cette caractéristique est représentée par la composante **p** ou **AR** : le nombre des coefficients pi .
* Moyenne mobile (Moving average) : **q** ou **MA** élimination du mouvement aléatoire des séries temporelles ou ce qu’on appelle le bruit
* Intégrée (Integrated) : c’est une propriété qui permet d’éliminer la tendance et la saisonnalité des séries temporelles, et est représentée par la composante **d** qui signifie le nombre de fois qu’il faut différencier y pour atteindre l’intégralité.

Cette dernière propriété nous permet d’avoir la stationnarité de notre modèle i.e. :

Un cas spécial de ARIMA est le modèle purement autorégressif i.e. : q = 0, d = 0, ce modèle signifie que les séries temporelles sont déjà stationnaires.

Un modèle avec q > 0 est plus compliquée avec et demande plus de temps pour se construire puisque des composants exponentiels s’introduisent a l’équation du modèle.

**PROPHET:**

Prophet ou bien Facebook Prophet est une procédure de prévision des données de séries temporelles basée sur un modèle additif où les tendances non linéaires sont adaptées à la saisonnalité annuelle, hebdomadaire et quotidienne, ainsi qu'aux effets des vacances.

Ils sont combinés dans l'équation suivante :

**Y(t)= g(t) + s(t) + h(t) + εt**

g (t): courbe de croissance linéaire ou logistique par morceaux pour modéliser les changements non périodiques dans les séries temporelles.

s (t): changements périodiques (par exemple, saisonnalité hebdomadaire / annuelle)

h (t): effets des vacances (fournies par l'utilisateur) avec des horaires irréguliers

εt: le terme d'erreur tient compte de tout changement inhabituel non pris en compte par le modèle.

En utilisant le temps comme régresser, Prophet essaie d'ajuster plusieurs fonctions linéaires et non linéaires du temps en tant que composantes. Prophet présente le problème de prévision comme un exercice d'ajustement de courbe plutôt que de regarder explicitement la dépendance temporelle de chaque observation dans une série temporelle.

**La préparation des données et le nettoyage:**

**La préparation des données:**

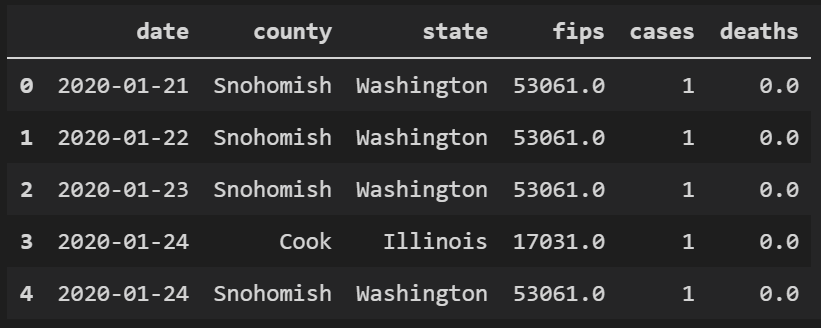
Nous avons utilisé la dataset publique suivante, trouvée sur :

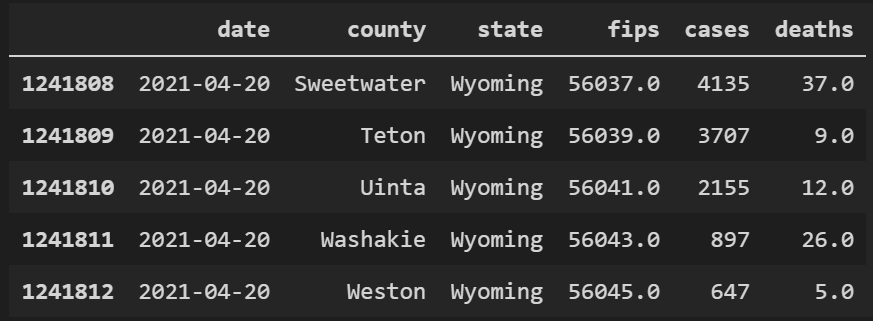
<https://www.kaggle.com/fireballbyedimyrnmom/us-counties-covid-19-dataset>

Cette dataset concerne le nombre de cas quotidiens des personnes atteintes par la Covid19 dans les Etats-Unis ainsi que le nombre quotidien des décès et ce regroupés par comté dans tout le pays.

Nous avons pensé alors que ça serait un bon exemple pour démontrer l’utilité de la prédiction des séries temporelles par des méthodes comme ARIMA et LSTM, et comment cela peut aider dans des situations de la vie réelle.

|  |
| --- |
| df = pd.read\_csv('us-counties.csv')  df.head()  df.tail() |



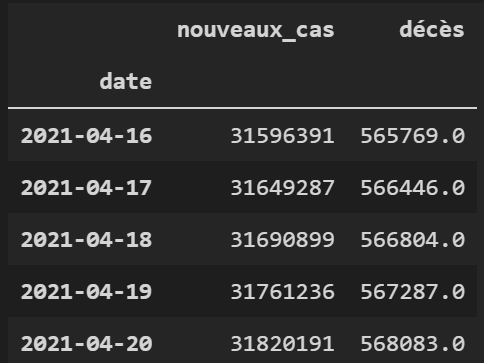


La dataset contient jusqu’à 1,241,812 entrées depuis le 21 janvier 2020 et jusqu’au 20 avril 2021.

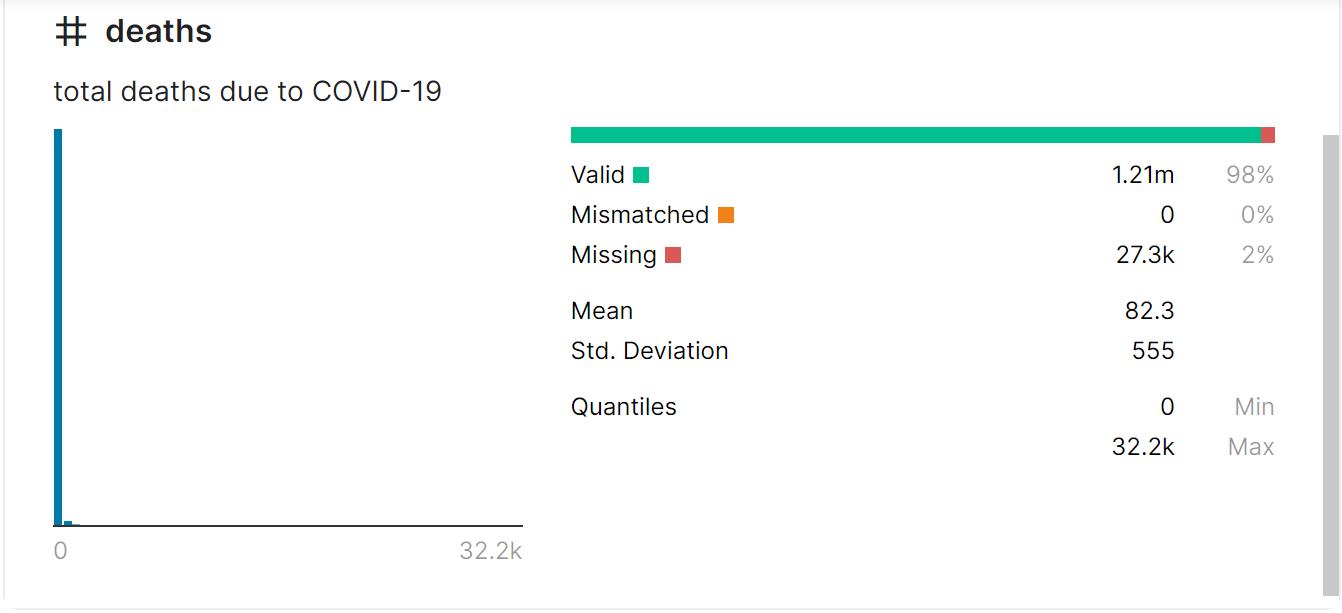
**Nettoyage des données :**

* Les données sont regroupées par chaque comté, ce qu’on n’en aura pas besoin, le plus important pour nous est d’avoir le nombre de cas de chaque jour dans tous les Etats-Unis. On a alors additionné le nombre de cas de chaque jour, en ne conservant que la colonne ‘date’, ‘nouveaux\_cas’ et ‘décès’.

|  |
| --- |
| #additionner les valeurs (cas et décès) de chaque date  df = df.groupby('date').sum()  # éliminer la colonne fips  df = df.drop("fips", axis= 1)  df.columns = ['nouveaux\_cas', 'décès']  df.head()  df.tail() |

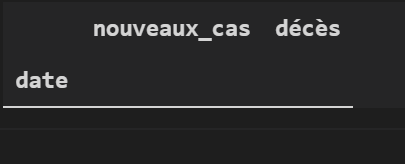
 

* Pour les cas de décès, il y a 27,3 valeurs manquantes, ce qu’on peut faire tout simplement est de les remplir avec des zéros.



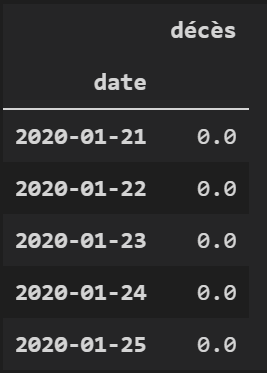
Mais apparemment après l’addition des valeurs de chaque date, on n’a plus de valeurs manquantes, comme affiché ci-dessous, ce qui implique que pandas a remplacé automatiquement ces valeurs par 0.

|  |
| --- |
| df.query("décès == ‘’ ") |



* Ensuite, on a divisé les données en deux, données pour les nouveaux cas, et données pour les cas de décès de chaque jour, pour prédire juste un à la fois.

|  |
| --- |
| df\_cas = pd.DataFrame(df['nouveaux\_cas'])  df\_deces = df\_cas = pd.DataFrame(df['décès'])  df\_cas.head()  df\_deces.head() |



**La finalité :**

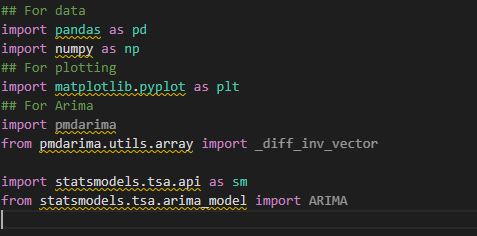
Notre objectif est de prédire les nouveaux cas enregistrés du Coronavirus en Etats-Unis ainsi que les nouveaux cas de décès.

**Le programme et les résultats obtenus :**

**ARIMA :**

Au lieu de prédire les décès et les nouveaux cas par un seul modèle, on a décidé de les séparer pour des raisons de simplification.

**Bibliothèques :**



Pandas : pour la lecture et présentation des données, et la représentation avec des graphes

Numpy : Pour la manipulation des listes et matrices

Matplotlib.pyplot : l’affichage des graphes

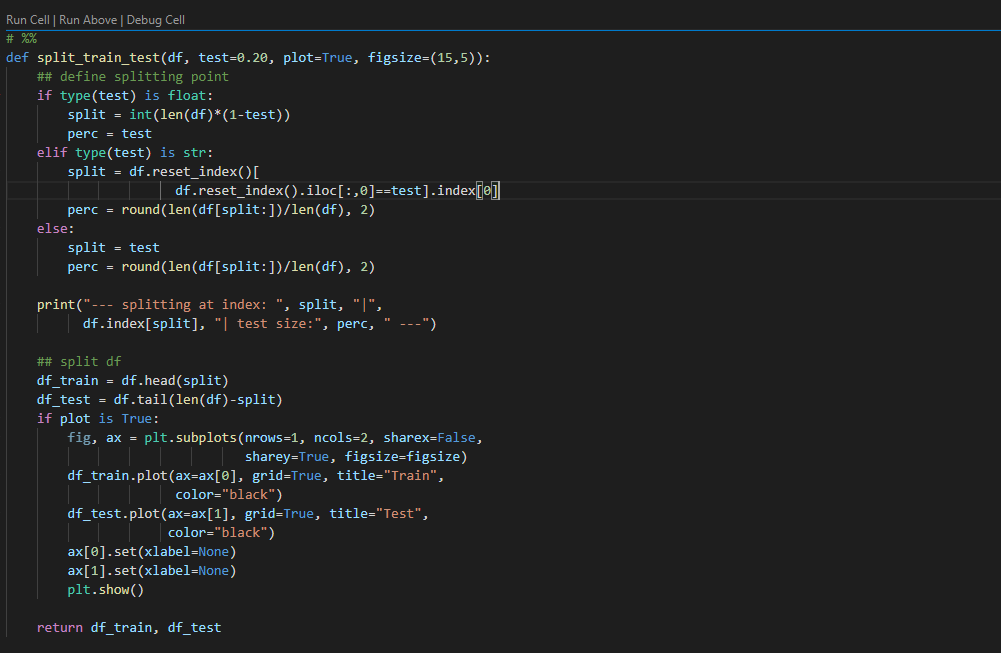
Pmdarima : nous permet de trouver les bons paramètres pour notre modèle

Statsmodels : nous permet de construire le modèle ARIMA

**Séparation des données entrainement/test :**

Pour cela on utilise une fonction qui peut faire la division de données de plusieurs manières, selon le type du paramètre ¨type¨ :

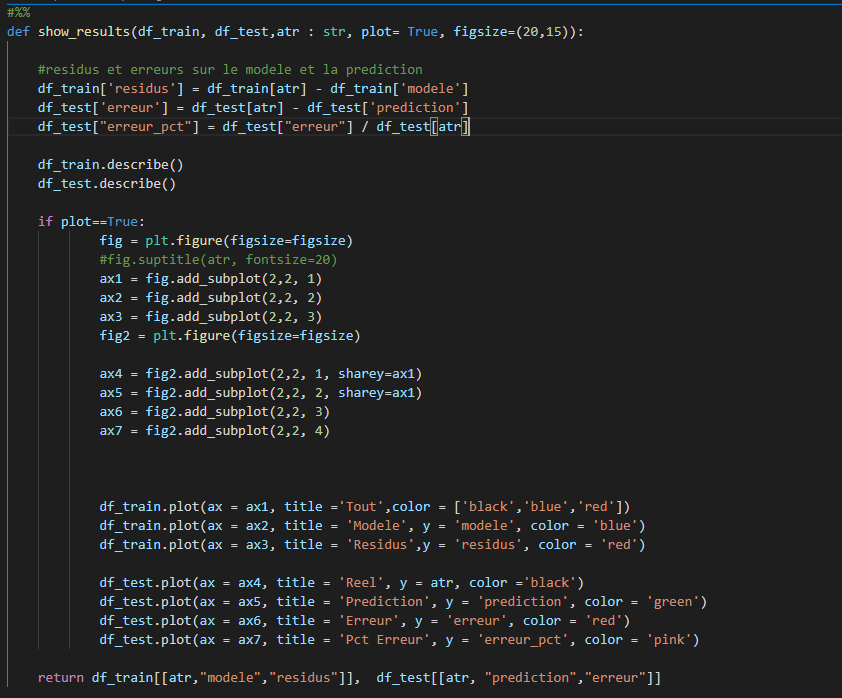
* Si c’est réel ou float alors ce réel va être la proportion des donnes du test par rapport a la totalité des données.
* Si c’est string ou chaine de caractères, ça serait considéré comme une date et on va faire la division à partir de cette date.
* Sinon, si c’est un entier, il serait considéré comme la position a partir duquel on va séparer les données du test des données d’entrainement.
* Dans tous les cas, on prend les données les plus recentes (les dernières)



**L’affichage des résultats**

Avant d’afficher les resultats sous forme d’un graphe, on fait une petite preparation, qui consiste a calculer les residus (l’erreur du modele par rapport aux donnees d’entrainement) et l’erreur de la prediction par rapport aux donnees du test.

La fonction describe du pandas permet d’afficher les statistiques sur la dataframe (table de donnees pandas) comme la moyenne, la variance…etc.



**Définition des paramètres du modèle**

pmdarima offre un outil pour tester si on doit differencier notre y (dans ce cas nombre de cas ou deces) pour atteindre la stationnarite. Cet outil est la classe ADFTest. On peut le faire comme suit :

|  |
| --- |
| **from** **pmdarima.arima.stationarity** **import** ADFTest  adf\_test = ADFTest(alpha=0.05)  p\_val, should\_diff = adf\_test.should\_diff(y) |

Cela nous retourne Vrai si la valeur de p est superieure a alpha. En fait p est la probabilite ou le degre par rapport auquel on doit refuser ou accepter une hypothese H0.

H0 : la serie temporelle est non-stationnaire (acceptee si p superieure a alpha)

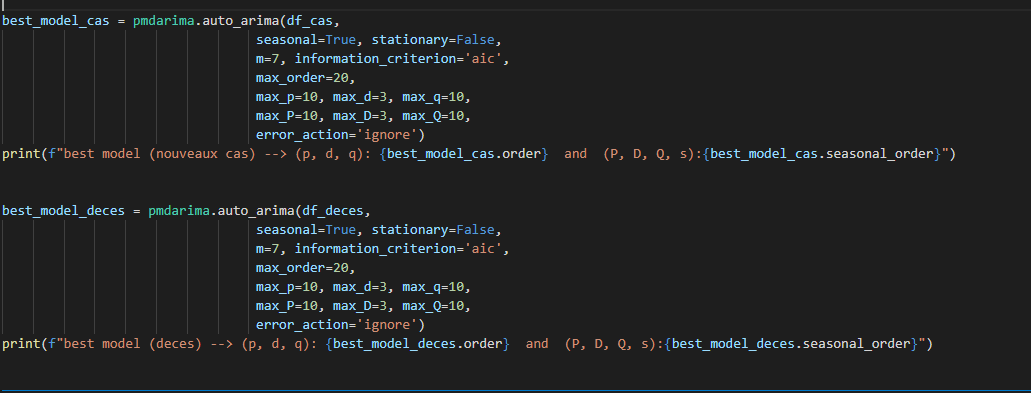
Si H0 est acceptee, le degre de differenciation d des series temporelle est superieur a 0.

On peut toutefois ne pas passer par la en utilisant directement auto\_arima.

La fonction auto\_arima de pmdarima nous permet de definir les parametres necessaires pour construire notre modele.

Ces parametres sont order = (p, d, q) qu’on a déjà abordee, et seasonal\_order = (P, D, Q, S) qui sont les parametres de saisonnalite qui definissent aussi p et d et q mais pour le composant de la saisonnalite.

S definit le cycle de la saisonnalite qui peut etre quotidien (7), hebdomadaire (52) ou mensuel (12).

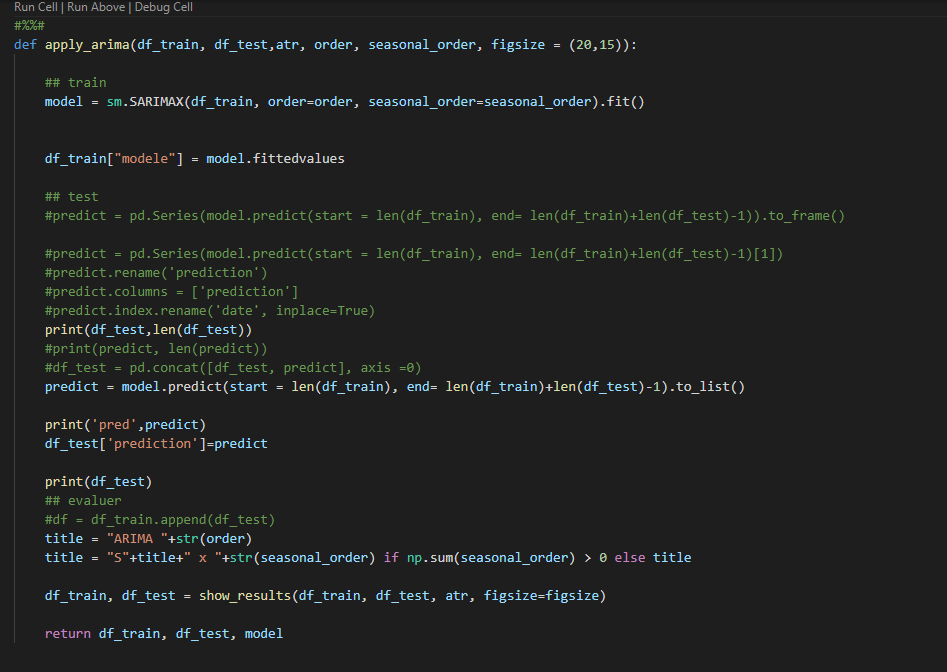


**Construction du modèle**

On utilise la classe SARIMAX de la biblitheque statsmodels pour construire notre modele basee sur les valeurs des parametres déjà trouvees avec pmdarima. Elle nous permet aussi de predire les valeurs des dates a venir pour pouvoir les comparer avec les donnees du test.

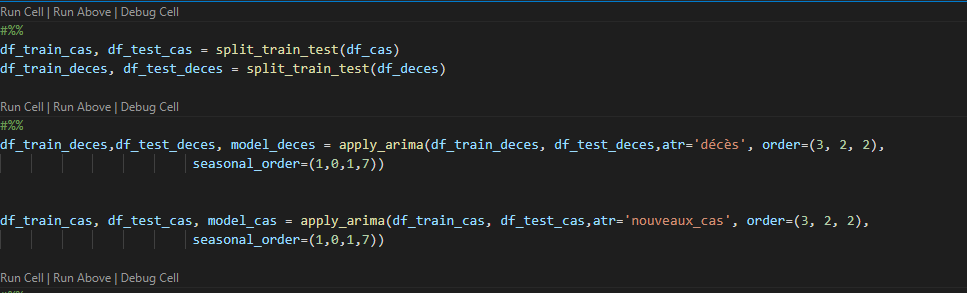
Le S dans SARIMAX est pour ¨Seasonal¨ et ce parce qu’il prend en charge la saisonnalite des donnees .

X est pour ¨eXogenous¨ parce que les parametres sont definits en dehors de la classe (a l’oppisition de ¨endogenous¨)



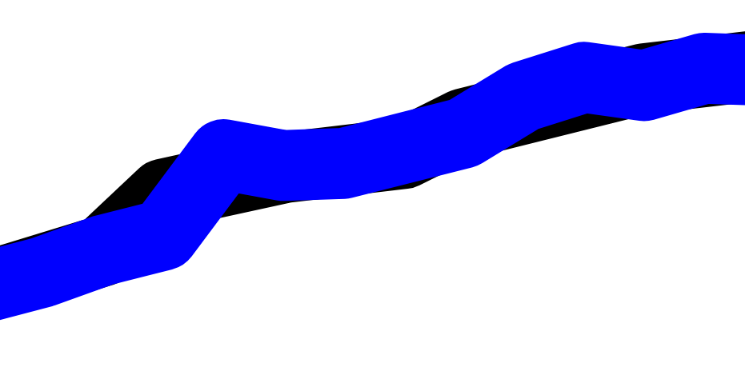
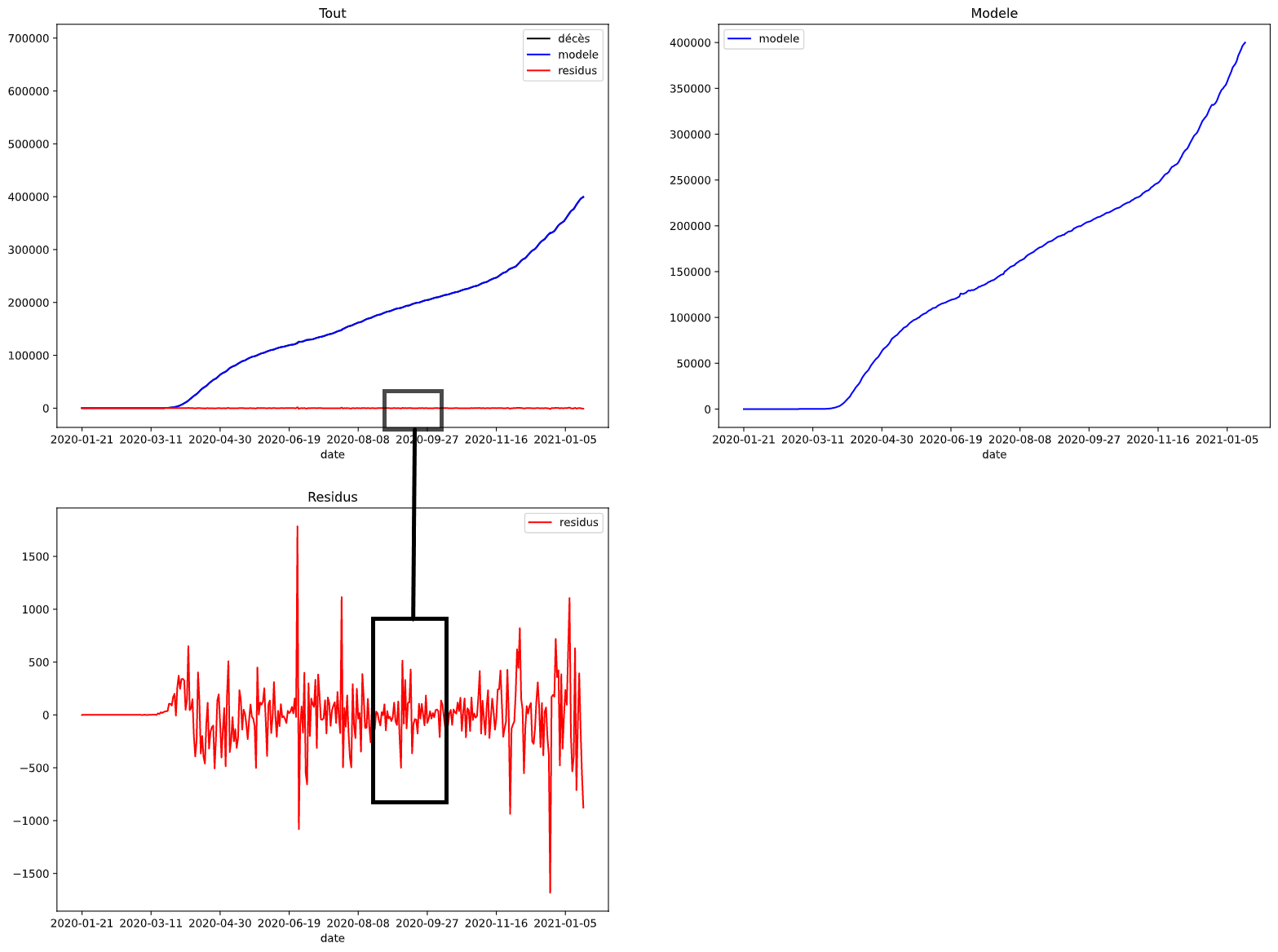
**Exécution et résultats**

On a execute avec order =(3, 2, 2) et seasonal\_order = (1, 0, 1, 7) car on a deja execute pmdarima.auto\_arima et on a obtenu ces valeurs et on a pas a l’executer a chaque fois.

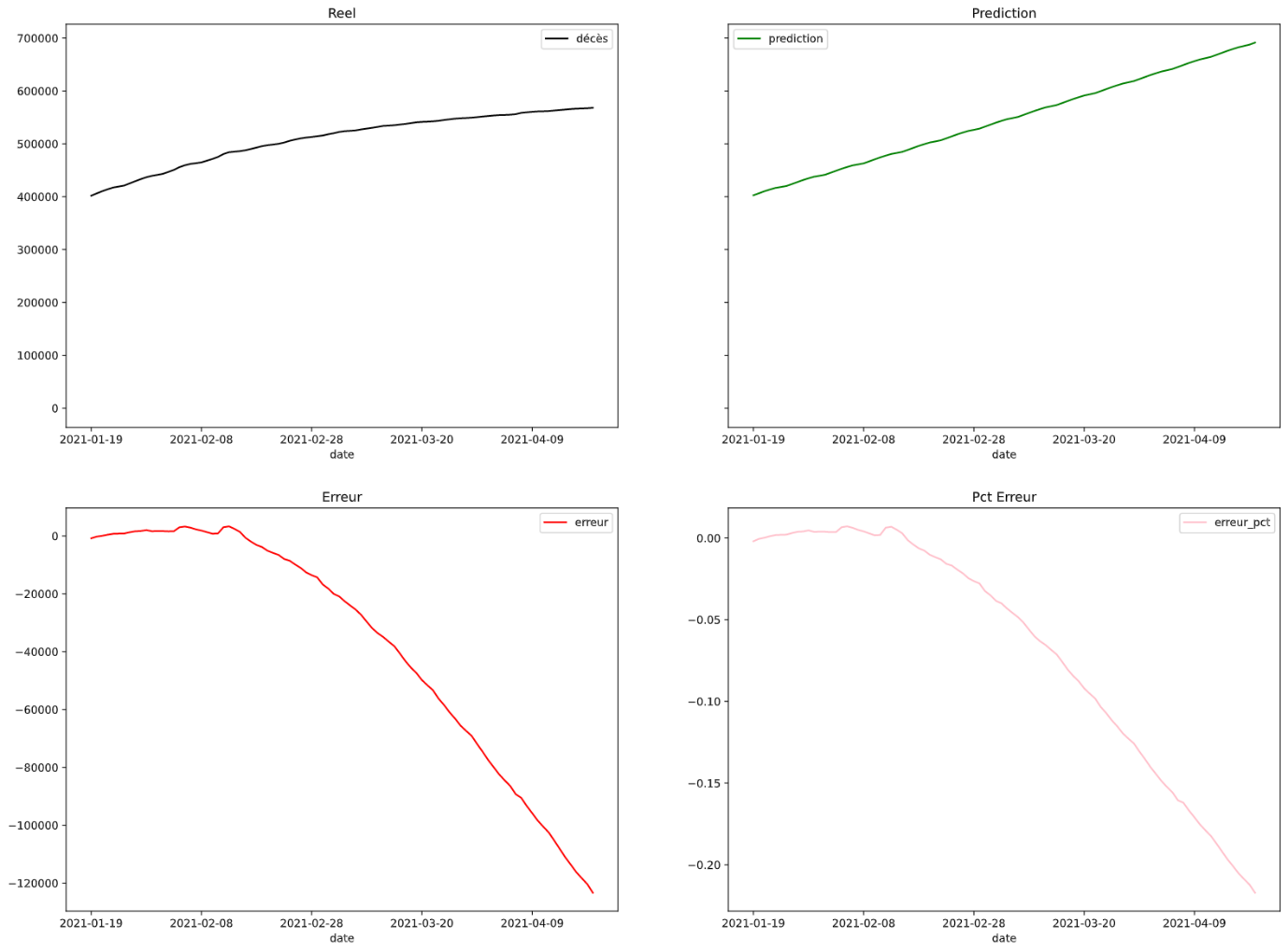


Le nombre de deces:

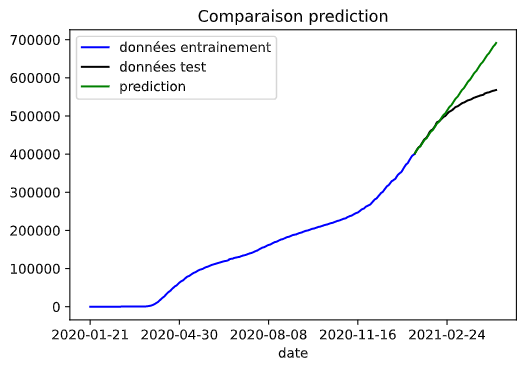
**Modele avec les residus :**



**La prediction avec l’erreur :**

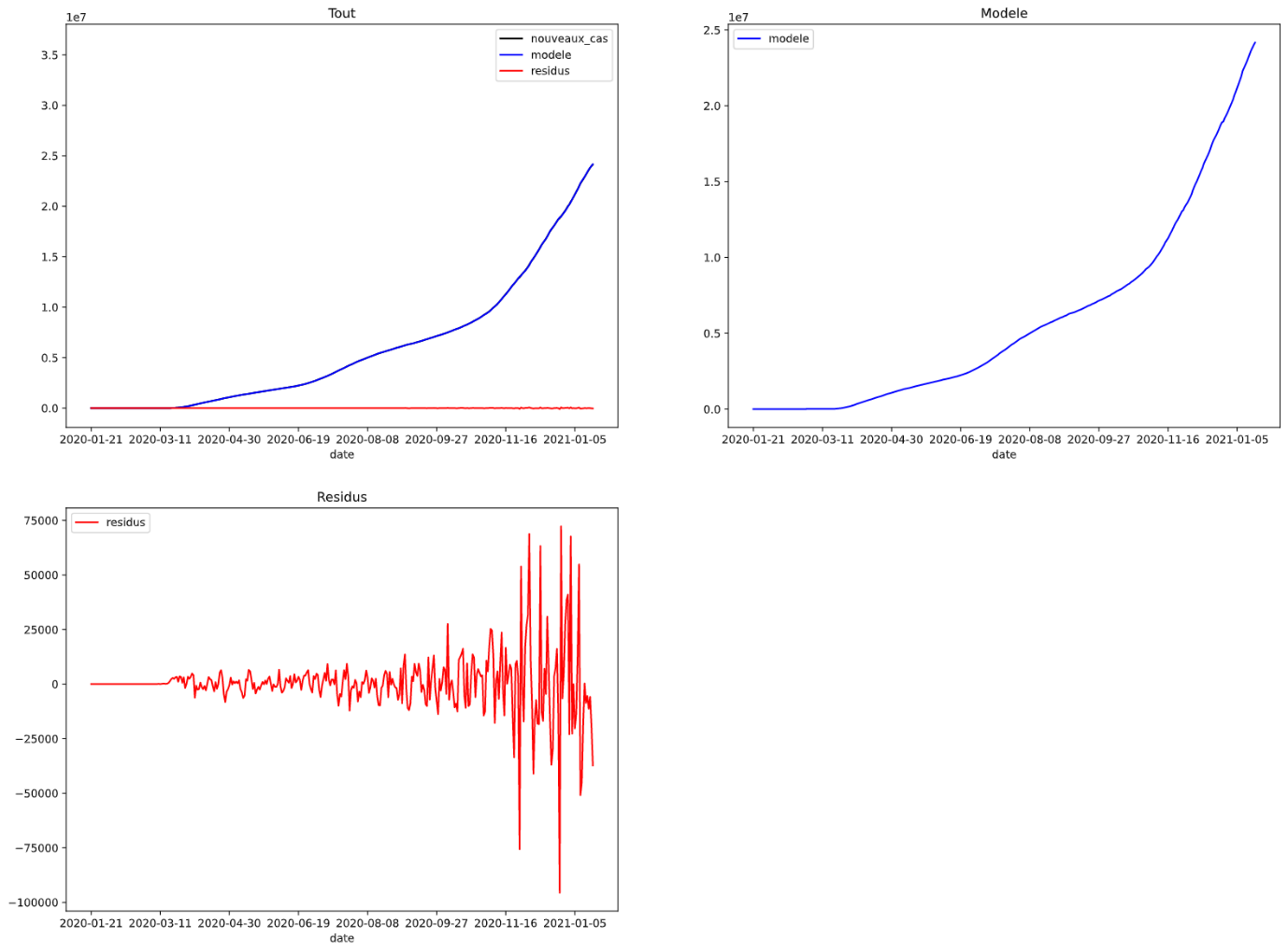
****

**Prediction des donnes de test**

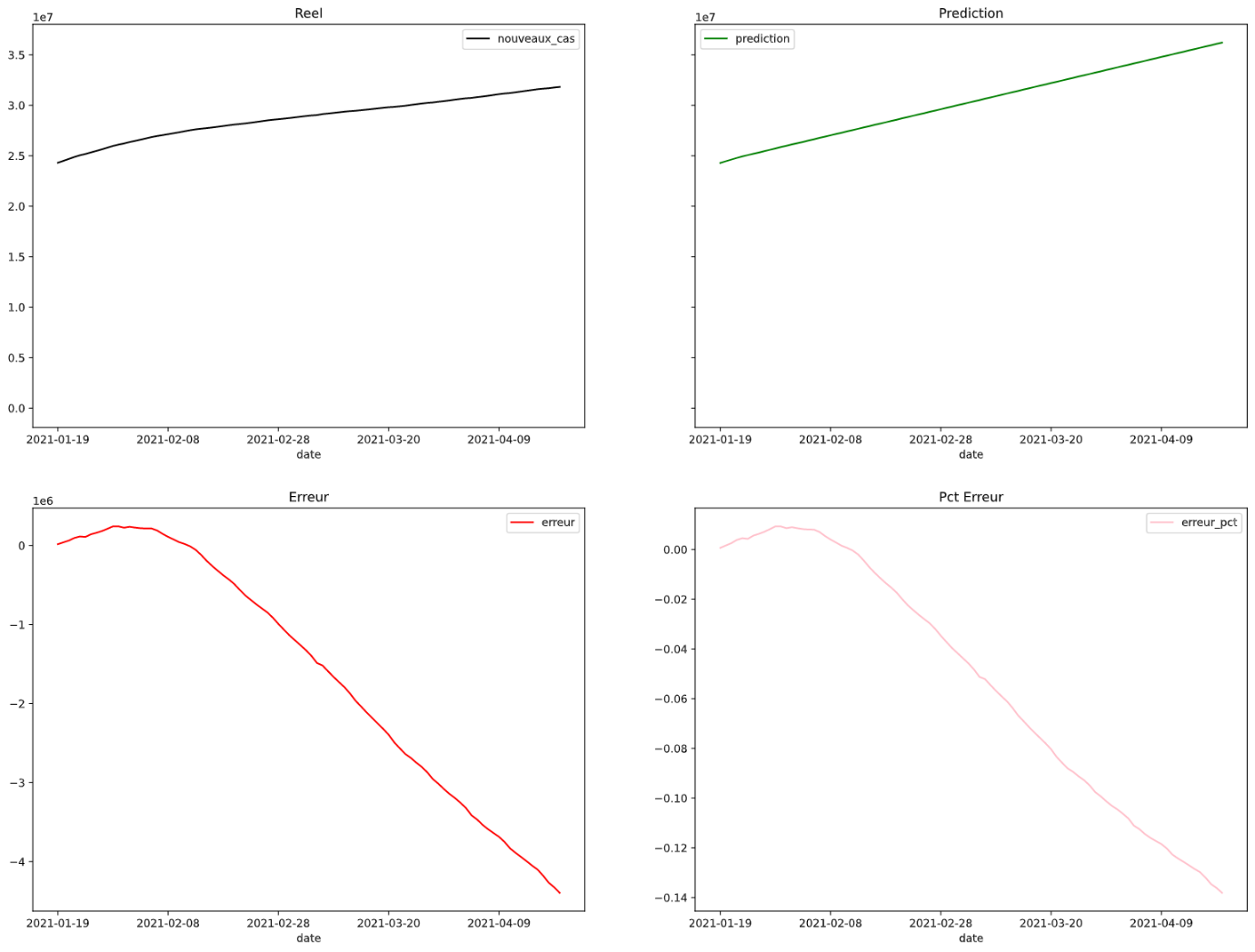


Nombre des nouveaux cas :

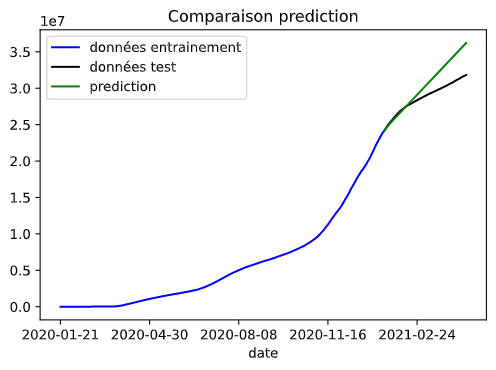
**Modele avec les residus :**



**La prediction avec l’erreur :**



**Prediction des donnes de test**

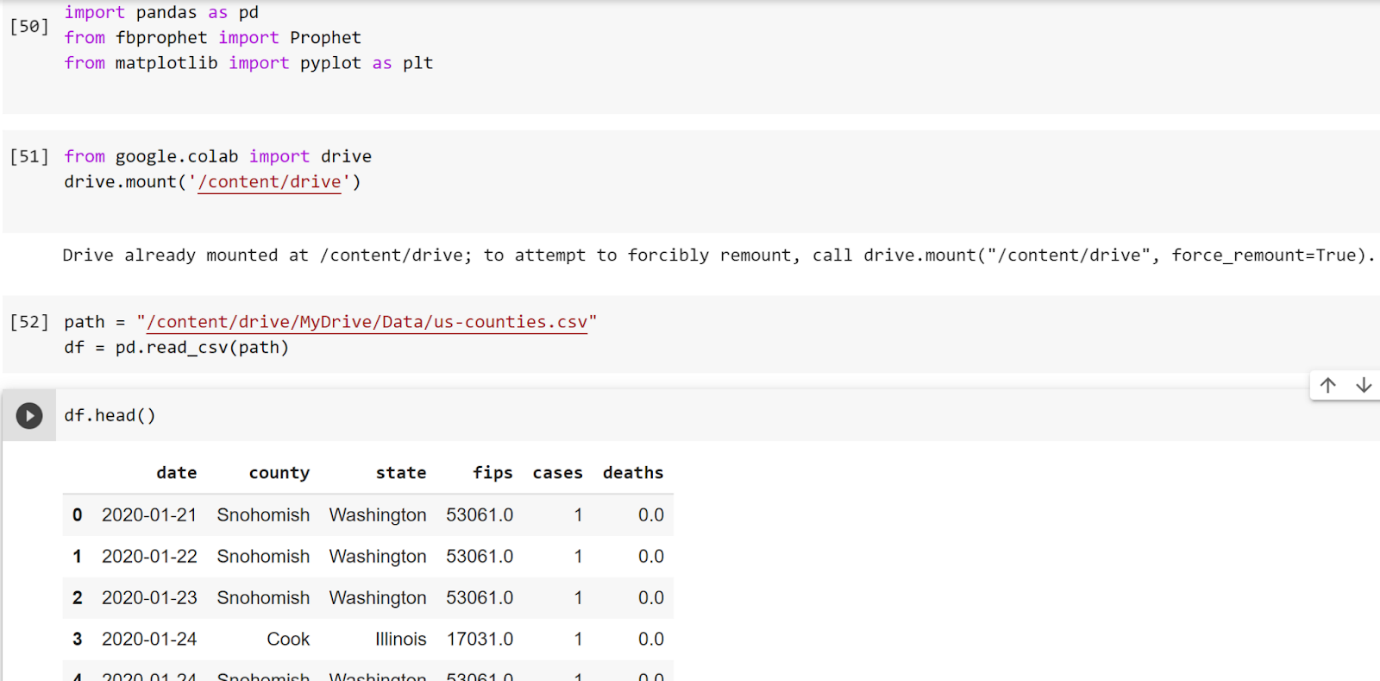


Commentaires :

On constate que l’erreur sur les données d’entrainement est trop petite, alors que l’erreur sur les données de test est grande

**Prophet :**

**Bibliothèques :**

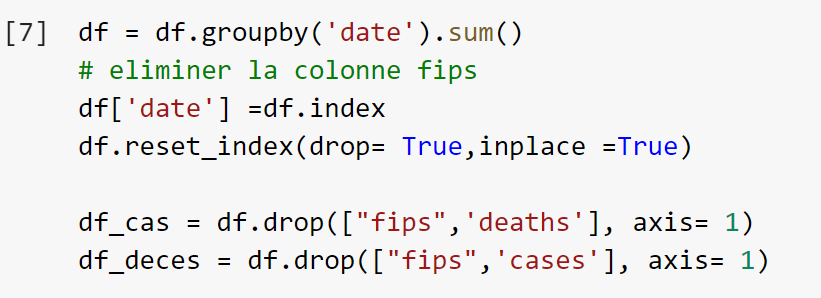


Pandas : pour la lecture et présentation des données, et la représentation avec des graphes

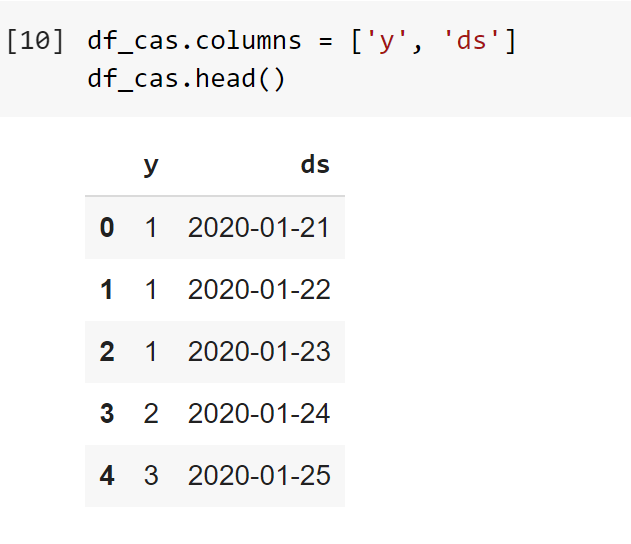
Numpy : Pour la manipulation des listes et matrices

Matplotlib.pyplot : l’affichage des graphes

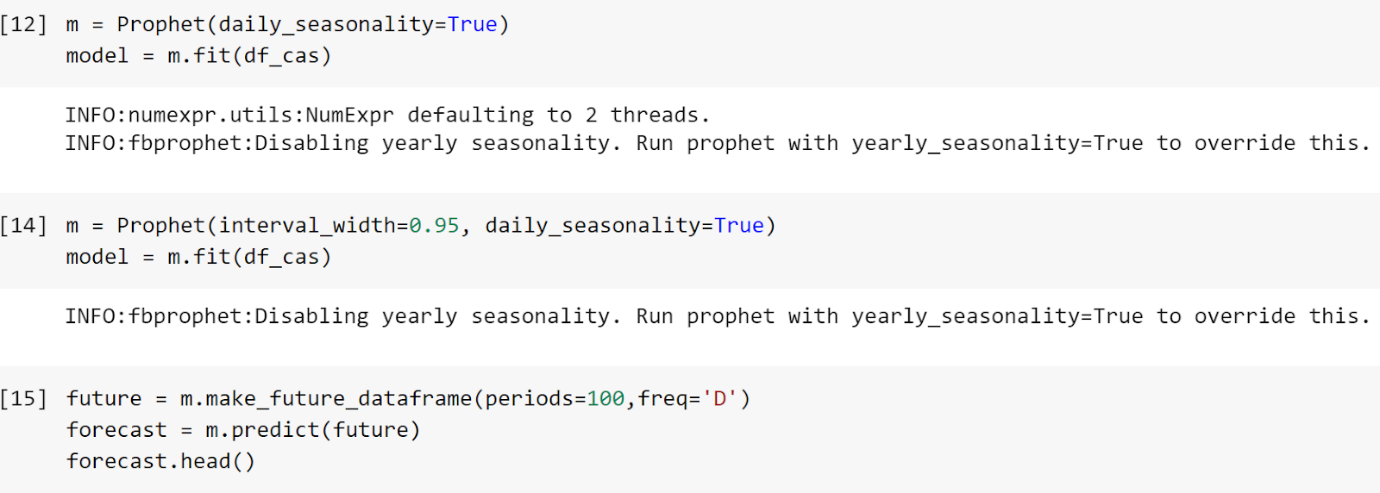
* + Après, on additionne les données du même jour et on supprime les autres colonnes sauf les cases et deaths. Ces deux derniers, on les a divisés pour faire la prédiction sur les deux a la fois.



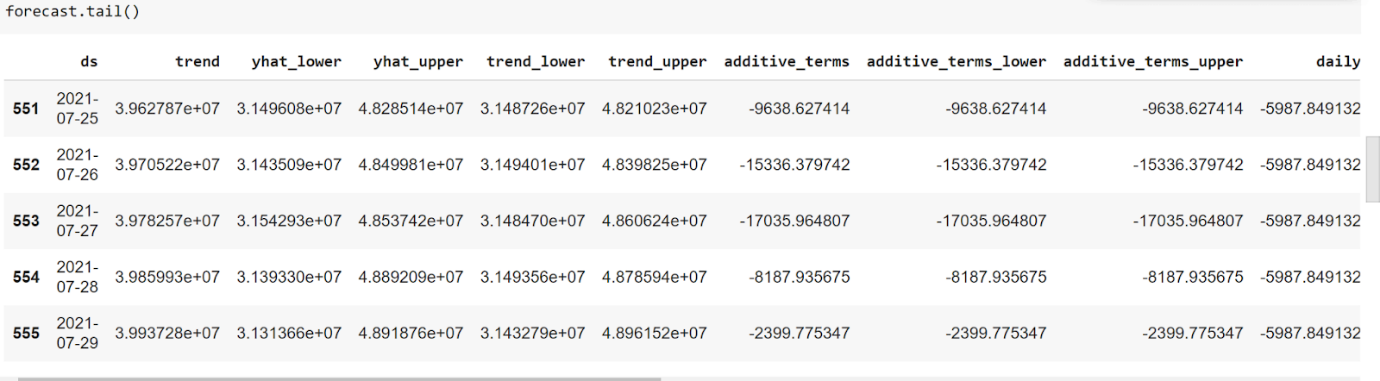
Ensuite, on nomme la colonne du date ‘ds’ et la colonne prédis ‘Y’



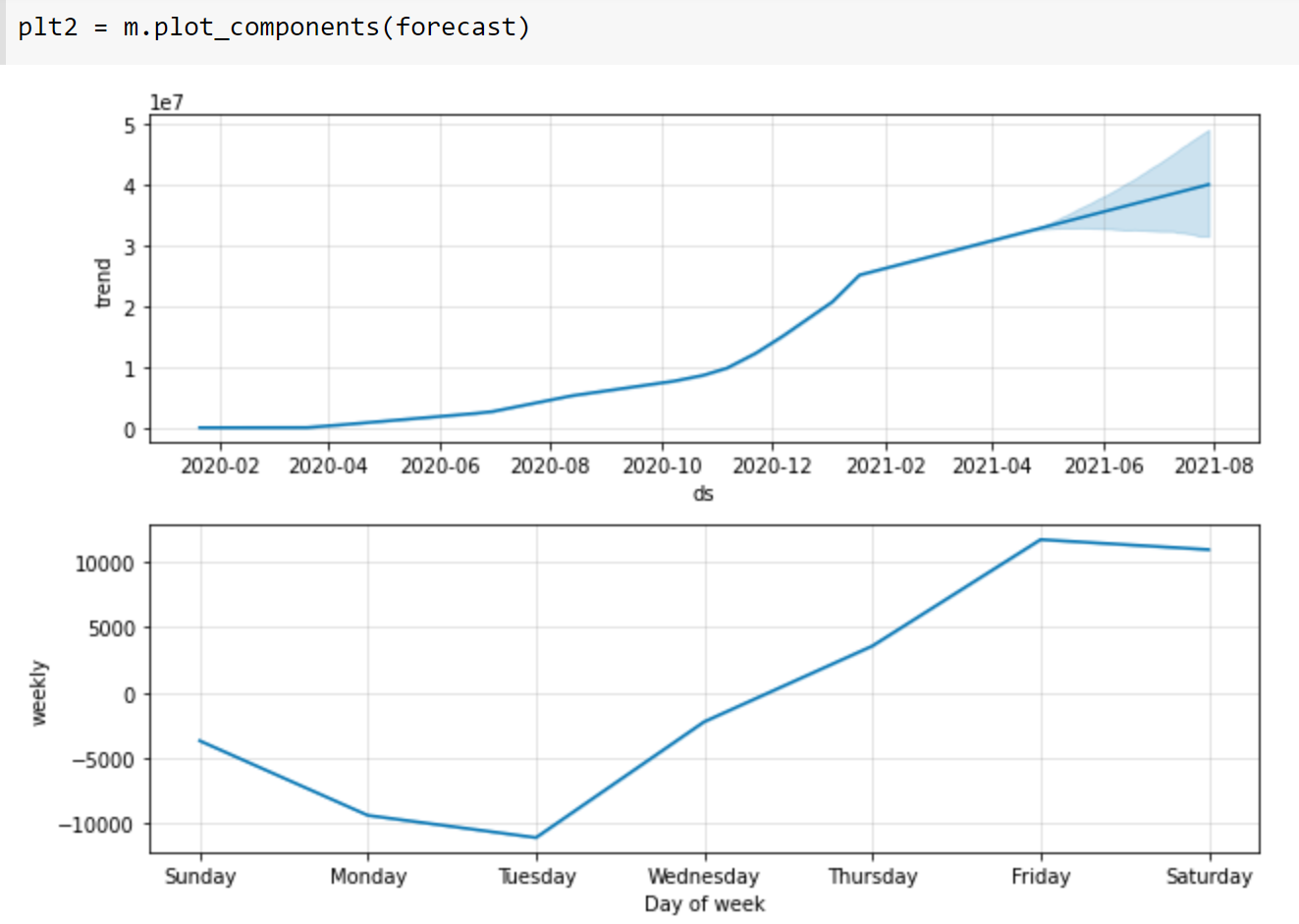
On applique le modèle du prophète, premièrement pour les nouveaux cas :



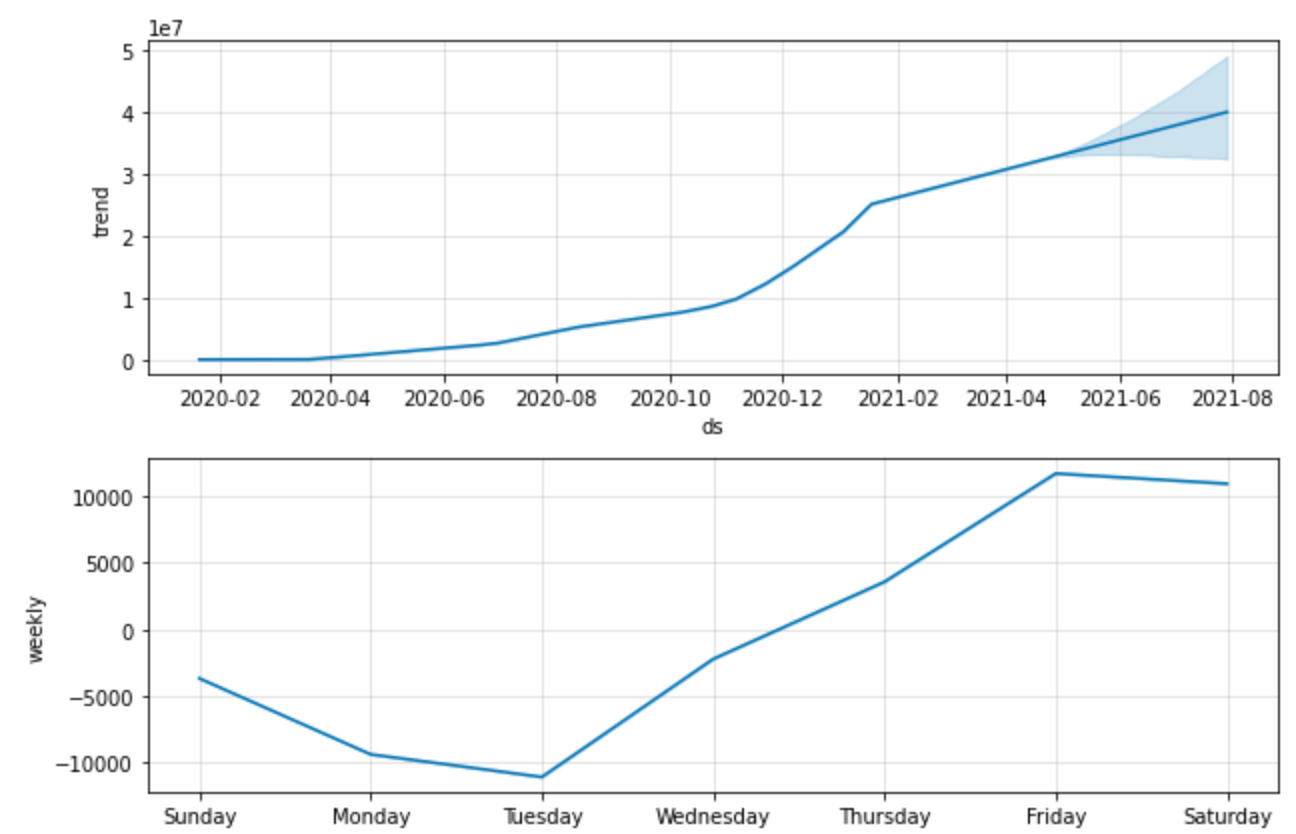
On aura, par la suite, le tableau qui contient les prédictions avec les valeurs élevée et faibles.



**Résultat de prédiction des cas :**



**Résultat de prédiction des décès**



**Références :**

<https://www.statsmodels.org/dev/examples/notebooks/generated/statespace_sarimax_stata.html>

[https://machinelearningmastery.com/time-series-forecasting](https://machinelearningmastery.com/time-series-forecasting/#:~:text=Making%20predictions%20about%20the%20future,them%20to%20predict%20future%20observations)

<https://medium.com/analytics-vidhya/time-series-forecasting-arima-vs-lstm-vs-prophet-62241c203a3b>

<https://people.duke.edu/~rnau/411arim.htm>

<https://www.analyticsvidhya.com/blog/2020/10/how-to-create-an-arima-model-for-time-series-forecasting-in-python/>

<https://towardsdatascience.com/time-series-forecasting-with-lstms-and-prophet-predict-your-email-workload-48bf9cdb1580>

<https://www.frontiersin.org/articles/10.3389/fdata.2020.00004/full>

<https://ichi.pro/fr/predire-les-cours-des-actions-a-l-aide-du-modele-prophet-de-facebook-195100708912806>