INTRODUCTION

I. Introduction aux séries temporelles et ARIMA

(A) Introduction générale aux séries temporelles

Les séries temporelles représentent un aspect interessant de l'analyse des données, offrant un aperçu des phénomènes qui évoluent avec le temps. Avant de plonger dans les détails du modèle ARIMA, il est important de comprendre les fondamentaux des séries temporelles.

Les séries temporelles sont des ensembles de données indexées par le temps, reflétant ainsi l'évolution d'un phénomène au fil du temps. Ces données jouent un rôle important dans de nombreux domaines, de la finance à la météorologie, en passant par l'économie et la santé publique. Leur analyse et leur prédiction ont une importance pour la prise de décisions.

(B) Décomposition des séries temporelle

Les séries temporelles sont décomposées en trois composantes principales :
- **Tendance (Tt)** : Représente la variation à long terme de la série, pouvant être croissante, décroissante ou stable.
- **Saisonnalité (St)** : Indique les variations périodiques qui se répètent à des intervalles réguliers, comme les saisons ou les cycles économiques.
- **Résidu ou erreur (εt)** : Comprend les fluctuations aléatoires non expliquées par la tendance ou la saisonnalité.

Cette décomposition permet une compréhension approfondie de la structure des données temporelles, facilitant ainsi la modélisation et la prédiction.

(C) Introduction à ARIMA

L'approche ARIMA (*AutoRegressive Integrated Moving Average*) est un modèle statistique puissant largement utilisé pour analyser et prévoir les séries temporelles. ARIMA tire son efficacité de la combinaison de trois concepts principaux : l'autorégression (AR), l'intégration (I) et la moyenne mobile (MA). Comprendre ces composantes est essentiel pour maîtriser ARIMA et exploiter ses capacités dans l'analyse des séries temporelles.

- **Autorégression (AR)** : Ce composant met l'accent sur la dépendance des observations actuelles avec les observations précédentes dans la série temporelle. Un modèle utilise les 'p' observations précédentes pour prédire la prochaine observation.

- **Intégration (I)** : L'intégration est une étape cruciale pour rendre la série temporelle stationnaire. Une série temporelle stationnaire est caractérisée par des propriétés constantes au fil du temps telles que la moyenne et la variance. La différenciation, qui est une forme d'intégration, est appliquée à la série temporelle brute pour éliminer les tendances et les structures de dépendance temporelle. Le degré de différenciation, noté par 'd', indique le nombre de différences successives nécessaires pour rendre la série temporelle stationnaire.

- **Moyenne mobile (MA)** : Ce composant modélise la relation entre une observation et une moyenne mobile de ses erreurs résiduelles précédentes. L'ordre de la moyenne mobile, noté

par 'q', spécifie la taille de la fenêtre de la moyenne mobile utilisée dans le modèle. Un modèle utilise les 'q' erreurs résiduelles précédentes pour prédire la prochaine observation.

En définissant les paramètres 'p', 'd' et 'q', le modèle ARIMA peut s'adapter aux structures temporelles spécifiques des données, capturant à la fois les tendances à long terme et les variations aléatoires. Cette flexibilité permet à ARIMA de modéliser une grande variété de séries temporelles avec précision. En comprenant ces concepts fondamentaux, nous sommes prêts à explorer plus en profondeur le fonctionnement et les applications pratiques du modèle ARIMA dans l'analyse et la prévision des séries temporelles.

II. Configuration et évaluation des modèles ARIMA

Le processus de configuration et d'évaluation des modèles ARIMA est essentiel pour obtenir des prévisions précises et fiables des séries temporelles. Cette partie aborde les étapes clés de ce processus avec le paramètrage initial du modèle et l'évaluation de sa performance.

(A) Paramètrage du modele ARIMA

La configuration d'un modèle ARIMA implique de choisir les valeurs appropriées pour les paramètres p, d et q, qui déterminent l'ordre de l'autorégression, le degré de différenciation et l'ordre de la moyenne mobile, respectivement. Cependant, estimer ces paramètres peut être un défi, nécessitant souvent des méthodes d'essais et d'erreurs itératifs.

(B) Evaluation des modele ARIMA

En raison de la complexité de l'estimation des paramètres ARIMA, des méthodes d'essais et d'erreurs itératifs sont souvent utilisées pour trouver les valeurs optimales. Cela implique d'ajuster différents modèles ARIMA avec des combinaisons de paramètres et de sélectionner celui qui minimise les erreurs de performance, telles que l'erreur quadratique moyenne (RMSE) ou l'erreur absolue moyenne (MAE).

(C) Recherche en grille des hyperparamètres ARIMA

Pour simplifier et accélérer le processus de configuration des modèles ARIMA, l'automatisation à l'aide de la recherche en grille est souvent utilisée. Cette approche implique de spécifier une grille de valeurs pour les paramètres p, d et q, puis d'évaluer la performance de chaque combinaison de paramètres à l'aide d'une métrique de performance définie.

Une fois que les modèles ARIMA ont été configurés, il est essentiel d'évaluer leur performance avant de les utiliser pour faire des prévisions. Cela implique de diviser les données en ensembles de formation et de test, d'ajuster les modèles sur l'ensemble de formation, de faire des prévisions sur l'ensemble de test et d'évaluer la précision des prévisions à l'aide de mesures d'évaluation appropriées.

L'évaluation des modèles ARIMA peut être réalisée en utilisant des bibliothèques Python telles que ***statsmodels*** et ***pandas***. Ces bibliothèques offrent des fonctionnalités pour ajuster les modèles ARIMA, faire des prévisions et évaluer la performance des modèles à l'aide de différentes métriques.

La recherche en grille des hyperparamètres ARIMA est une approche systématique pour trouver les valeurs optimales des paramètres p, d et q. En résumé cette procédure implique de spécifier une grille de valeurs pour chaque paramètre, d'évaluer les performances de chaque combinaison de paramètres et de sélectionner celle qui donne les meilleures performances prévisionnelles.

(PARLER PLUS DU CODE)

III. Application et prédictions avec ARIMA

(A) Mise en place du modele ARIMA

Une fois les paramètres du modèle ARIMA déterminés, le modèle peut être utilisées sur des données d'entraînement. L'utilisation du modèle ARIMA pour faire des prédictions implique plusieurs étapes.

- **Importation des bibliothèques** : La première étape consiste à importer les bibliothèques nécessaires. Dans ce cas, la bibliothèque ***sklearn.metrics*** est utilisée pour calculer le score R2 et l'erreur quadratique moyenne (RMSE), qui sont des mesures couramment utilisées pour évaluer la précision des prédictions.

- **Préparation des données :** Les données sont divisées en un ensemble d'entraînement et un ensemble de test. L'ensemble d'entraînement contient généralement environ 80 % des données, tandis que l'ensemble de test contient les 20 % restants. Cette division permet d'évaluer la performance du modèle sur des données qu'il n'a pas vues pendant l'entraînement.

- **Initialisation des listes history et predictions** : Deux listes sont initialisées : history, qui contient les valeurs de la série temporelle jusqu'à présent, et predictions, qui contiendra les prédictions du modèle. Ces listes sont utilisées pour stocker les données réelles et prédites au fur et à mesure que le modèle est évalué sur l'ensemble de test.

(B) Prédictions avec ARIMA

Après que le modèle ARIMA soit ajusté aux données d'entraînement, il peut être utilisé pour faire des prédictions sur de nouvelles données. Pour ce faire, on peut utiliser la fonction ***ARIMA.predict()*** pour générer des prévisions pour les points de données futurs.

- **Prédictions :** Pour chaque donnée dans l'ensemble de test, le modèle ARIMA génère une prévision pour le prochain point de données, et cette prévision est ajoutée à la liste des prédictions. Ensuite, la valeur réelle correspondante est ajoutée à la liste de l'historique. Ce processus se poursuit jusqu'à ce que toutes les données de l'ensemble de test soient prédites.

- **Calcul de l'erreur :** Une fois que toutes les prédictions ont été générées, l'erreur quadratique moyenne (RMSE) et le score R2 sont calculés entre les prédictions et les valeurs réelles de l'ensemble de test. Ces mesures fournissent une indication de la précision du modèle par rapport aux données observées.

- **Affichage des résultats :** Enfin, les résultats sont affichés à l'aide d'un graphique qui compare les valeurs réelles de l'ensemble de test aux prédictions du modèle. Cela permet de visualiser la performance du modèle et de déterminer s'il détermine correctement les tendances et les modèles dans les données.

(C) Prédictions hors échantillon

Les prédictions hors échantillon sont des prévisions faites pour des périodes futures qui ne font pas partie de l'ensemble de données d'entraînement initial. Ces prédictions sont importantes pour évaluer la capacité du modèle ARIMA à généraliser de nouvelles données et à maintenir sa performance prédictive dans des conditions réelles.

Pour faire des prédictions hors échantillon, on peut utiliser la fonction ***ARIMA.forecast()*** dans les bibliothèques ***Python*** comme ***statsmodels***. Cette fonction permet de générer des prédictions pour un certain nombre de pas de temps dans le futur, en fonction du modèle ARIMA ajusté aux données d'entraînement.

Après avoir généré les prédictions hors échantillon, il est crucial de les évaluer en les comparant aux valeurs réelles pour les périodes correspondantes. Cela permet de déterminer si le modèle maintient sa précision prédictive sur des données inconnues. Nous avons utilisées des mesures d'évaluation telles que le RMSE et le R^2 à cette fin.

En outre, voici un détail des étapes utilisées pour la génération des prédictions hors échantillon :

- ***difference(dataset)*** **:** Cette fonction calcule la différence entre chaque point de données et le point de données précédent, nous l'avons utilisée pour rendre une série temporelle stationnaire.
- ***inverse_difference(history, prediction)*** **:** Cette fonction inverse la différenciation appliquée à la série temporelle, transformant ainsi les prédictions faites sur la série différenciée en prédictions sur la série originale.
- **differenced = difference(X) :** La série temporelle est différenciée pour la rendre stationnaire.
- **model = ARIMA(differenced, order=(10,2,1)) :** Un modèle ARIMA est créé avec l'ordre déterminé lors du processus de paramétrage du modèle.
- **model_fit = model.fit() :** Le modèle ARIMA est ajusté aux données différenciées.
- **forecast = model_fit.forecast(steps=7) :** Des prévisions sont générées pour un nombre spécifié de pas de temps dans le futur.
- **history = [x for x in X] :** L'historique est initialisé avec les valeurs de la série temporelle originale.
- **for prediction in forecast: :** Pour chaque prévision, la prévision est inversée pour correspondre à la série originale, puis ajoutée à l'historique, et une nouvelle année est crée.
- **pyplot.plot(history, color='blue') :** L'historique, qui comprend maintenant les prévisions, est tracé en bleu.
- **pyplot.plot(X, color='red') :** La serie temporelle est tracé en rouge.
- **pyplot.legend(['Prediction', 'Reel']) :** Une légende est ajoutée au graphique pour identifier les prédictions et les valeurs réelles.
- **pyplot.show() :** Le graphique est affiché pour visualiser les prédictions et les valeurs réelles.

IV. Conclusion et perspectives

(A) Récapitulatif des points clés

(B) Perspectives d'avenir

- Rajouter une parties ou je parle des variables que j'utilise
- Interprétation des résultats pour les pays et variables que j'utilise