Time Series Project

EDDAOU M'hamed Issam

Index

Ce que cette présentation va couvrir

- Introduction
- Data Analysis (EDA)
- Modélisation Arima

Introduction

Objet de modelisation

Time Series
ARIMA

DataSet

les ventes mensuelles de champagne français

Outils

Jupyter notebook python
statsmodel

Introduction

L'ensemble de données n'est pas à jour. Cela signifie que nous ne pouvons pas facilement collecter des données mises à jour pour valider le modèle.
Par conséquent, nous prétendrons que nous sommes en septembre 1971 et retiendrons la dernière année de données issues de l'analyse et de la sélection du modèle.
Cette dernière année de données servira à valider le modèle définitif.
Le code ci-dessous chargera l'ensemble de données en tant que série Pandas et le divisera en deux, un pour le développement du modèle (dataset.csv) et l'autre pour la validation (validation.csv).

```
# separate out a validation dataset

series = pd.read_csv(
    'https://raw.githubusercontent.com/jbrownlee/Datasets/master/monthly_champagne_sales.csv',
    header=0,
    index_col=0,
    parse_dates=True)
split_point = len(series) - 12
dataset, validation = series[0:split_point], series[split_point:]
print('Dataset %d, Validation %d' % (len(dataset), len(validation)))
dataset.to_csv('dataset.csv')
validation.to_csv('validation.csv')
```

```
Dataset 93, Validation 12
```

Data Analysis

Persistance Model

```python
# evaluate persistence model on time series
from sklearn.metrics import mean_squared_error
from math import sqrt
# load data
series = pd.read_csv('dataset.csv', header=0, index_col=0, parse_dates=True)
# prepare data
X = series.values
X = X.astype('float32')
train_size = int(len(X) * 0.50)
train, test = X[0:train_size], X[train_size:]
# walk-forward validation
history = [x for x in train]
predictions = list()
for i in range(len(test)):
    # predict

    yhat = history[-1]
    predictions.append(yhat)
    # observation
    obs = test[i]
    history.append(obs)
    print('>Predicted=%.3f, Expected=%3.f' % (yhat, obs))
# report performance
rmse = sqrt(mean_squared_error(test, predictions))
print('RMSE: %.3f' % rmse)
```

La première étape avant de s'enliser dans le Data Analysis et la modélisation des données consiste à établir une base de référence de performances.

Cela fournira à la fois un modèle pour évaluer les modèles à l'aide des Test Harness et une mesure de performance par laquelle tous les modèles prédictifs plus élaborés peuvent être par rapport.

La prévision de base pour la prévision de séries chronologiques est appelée prévision naïve, ou persistance.

-> RMSE : 3186.501

Data Analysis

01

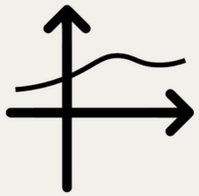
Résumé Statistique

02

Line plot

03

Seasonal Line plot

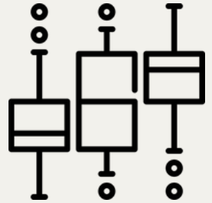


04

Histogram and Density Plot

05

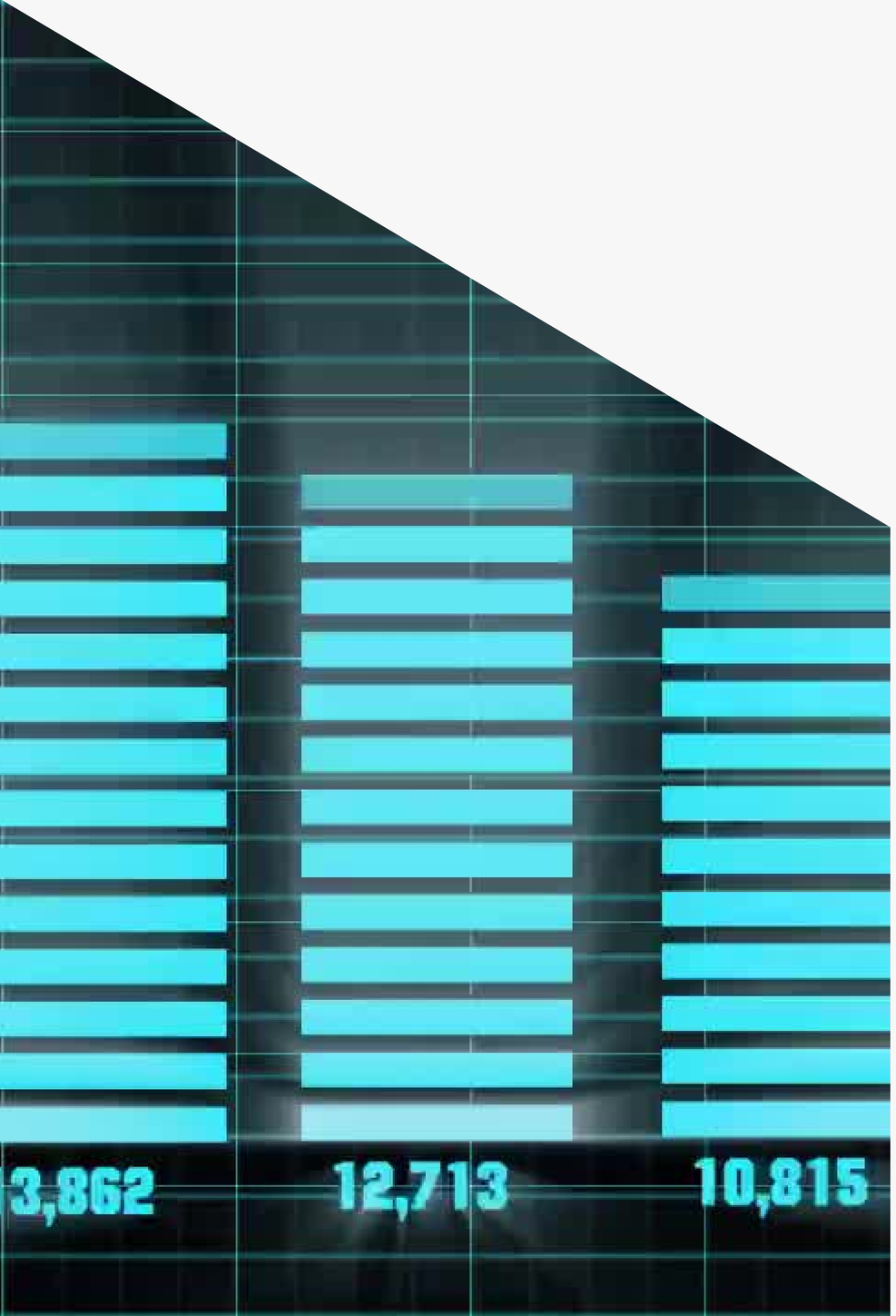
Box and whisker plot



06

Augmented dicky Fuller test

$$ADF$$

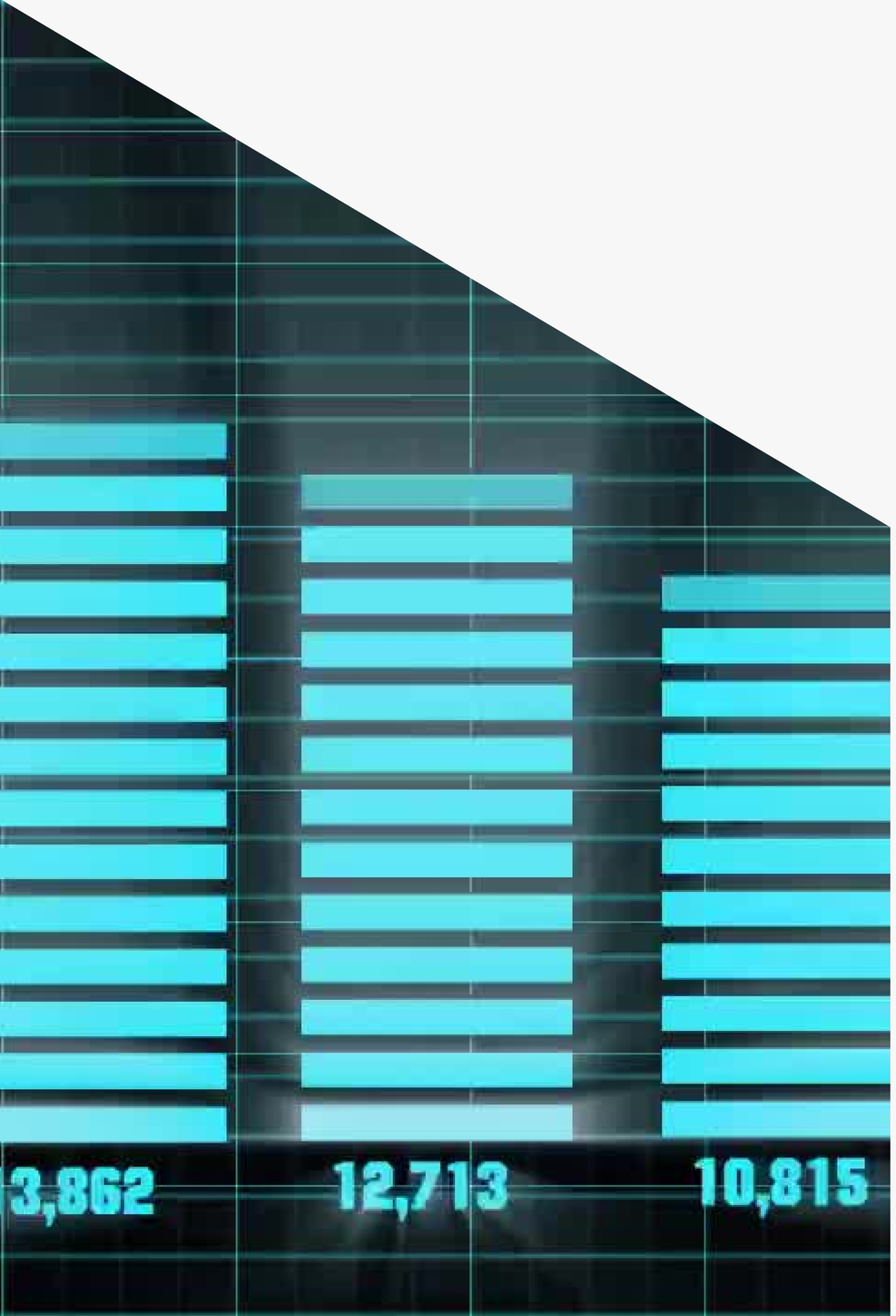


Data Analysis

1. Résumé Statistique

```
print("Résumé statistique du 'dataset.csv'")
print(series.describe())
```

```
Résumé statistique du 'dataset.csv'
               Sales
count      93.000000
mean     4641.118280
std      2486.403841
min      1573.000000
25%      3036.000000
50%      4016.000000
75%      5048.000000
max     13916.000000
```

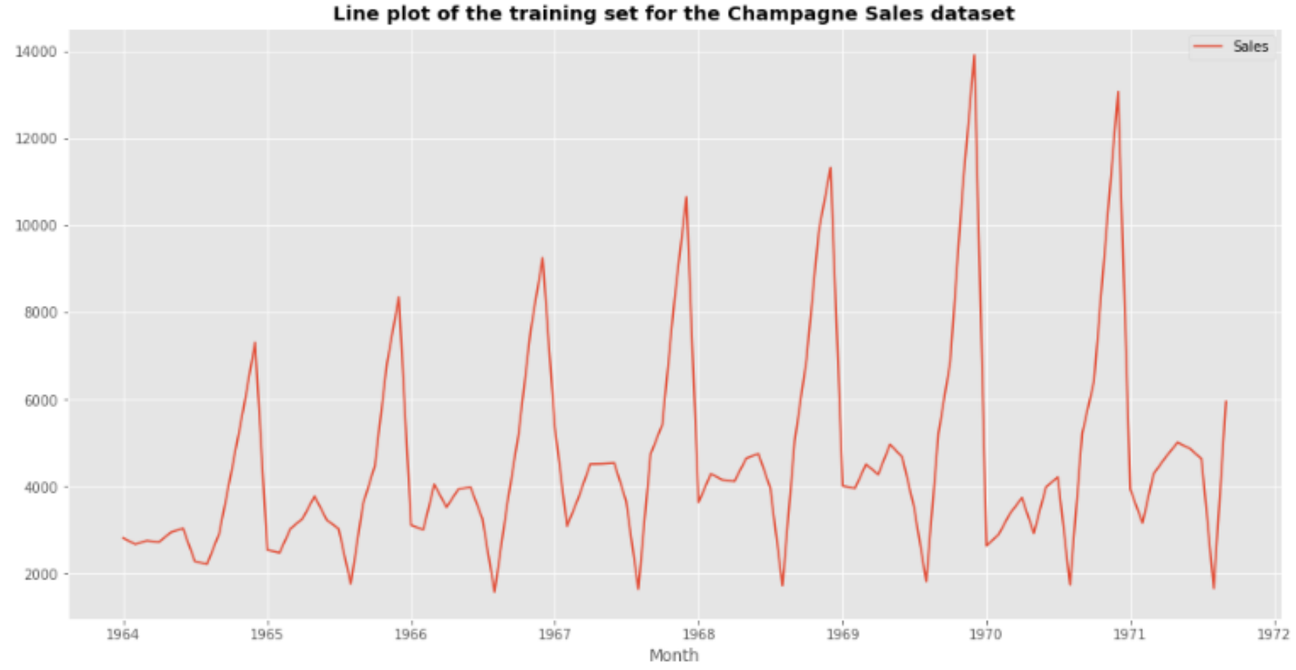


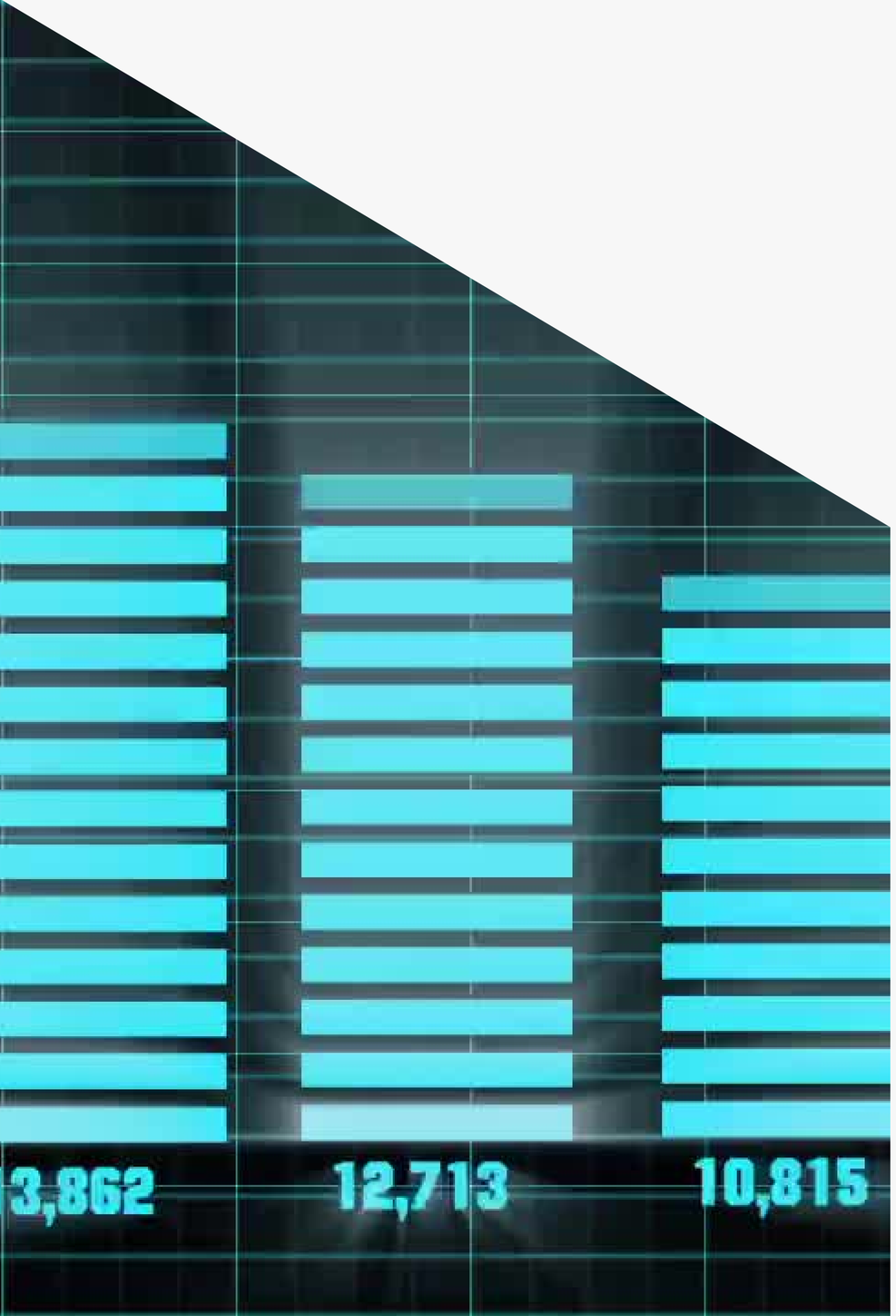
Data Analysis

2. Line Plot

```
plt.figure(figsize=(16, 8))
plt.title("Line plot of the training set for the Champagne Sales dataset",
          fontweight='bold')
sns.lineplot(data=series)
```

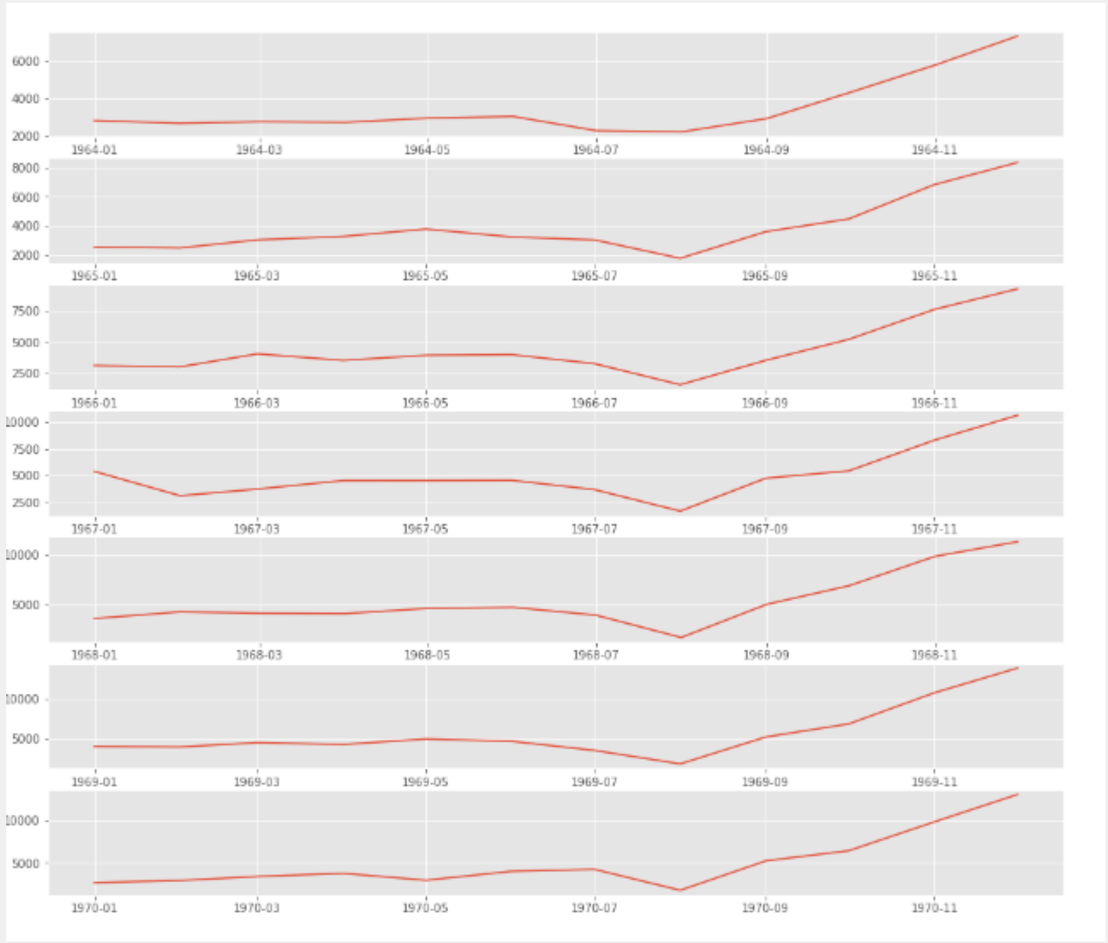
```
<AxesSubplot:title={'center':'Line plot of the training set for the Champagne Sales dataset'}, xlabel='Month'>
```

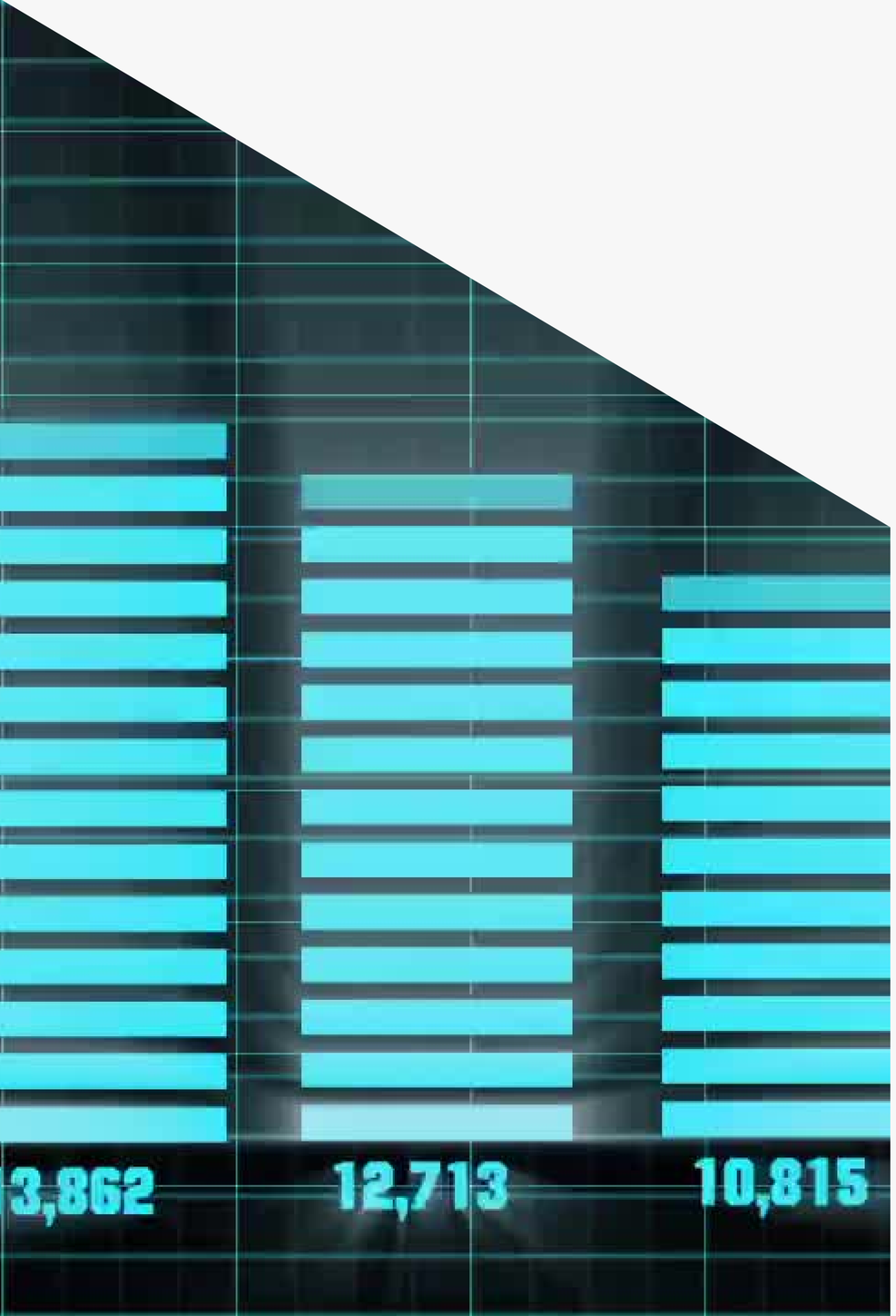




Data Analysis

3. Seasonal Line plot



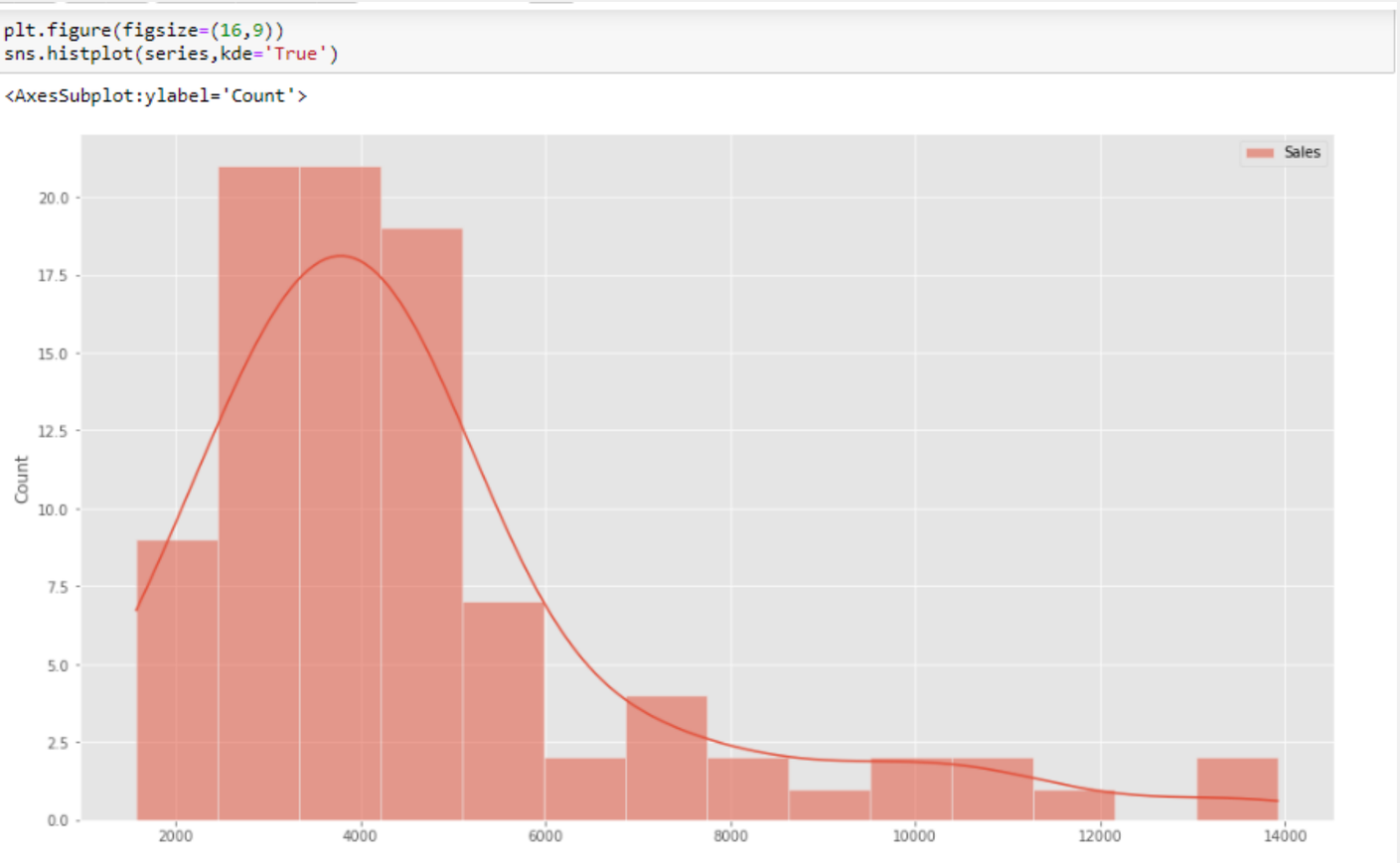


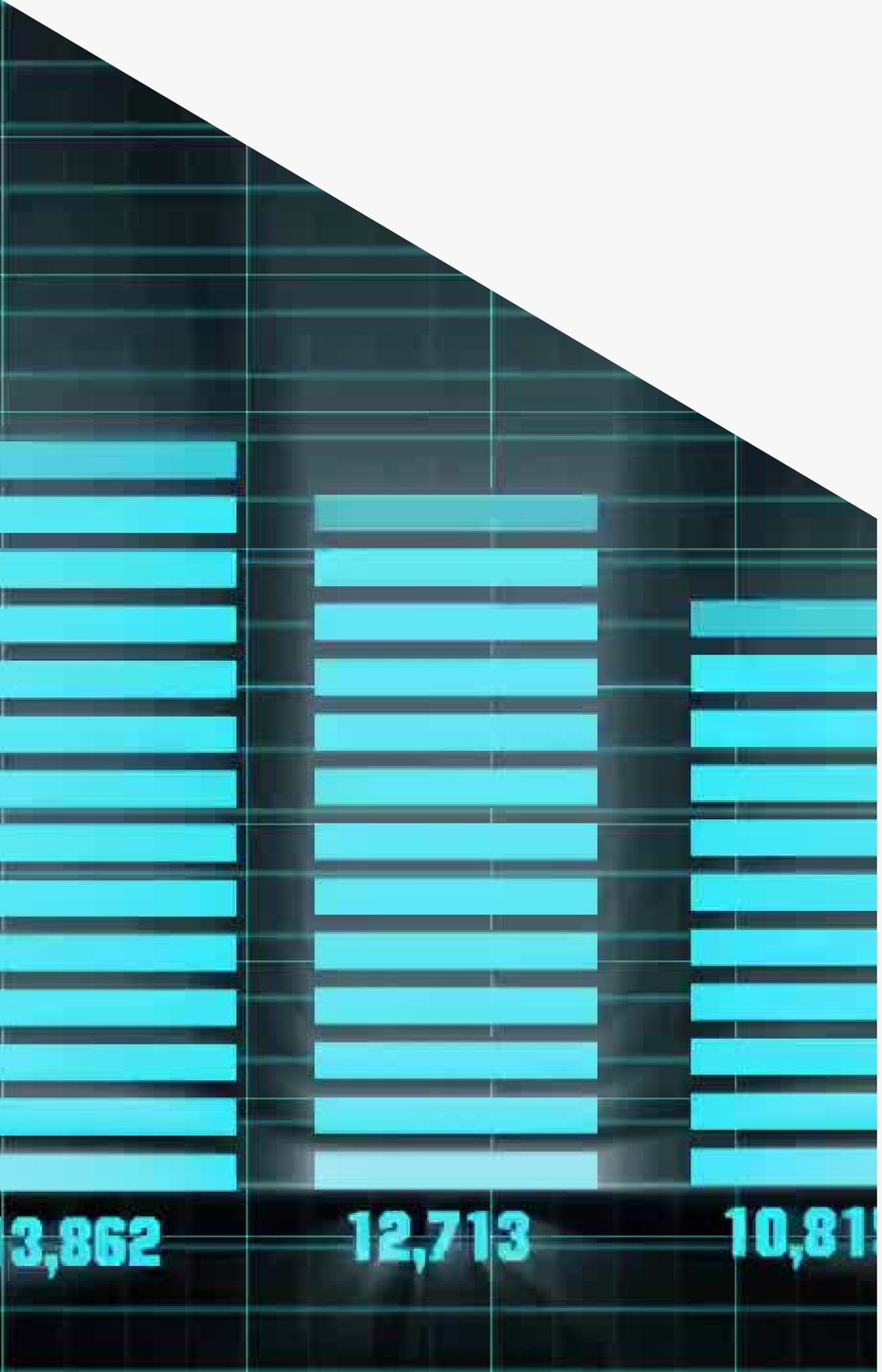
Data Analysis

4. Histogram and Density Plot

```
plt.figure(figsize=(16,9))
sns.histplot(series,kde='True')
```

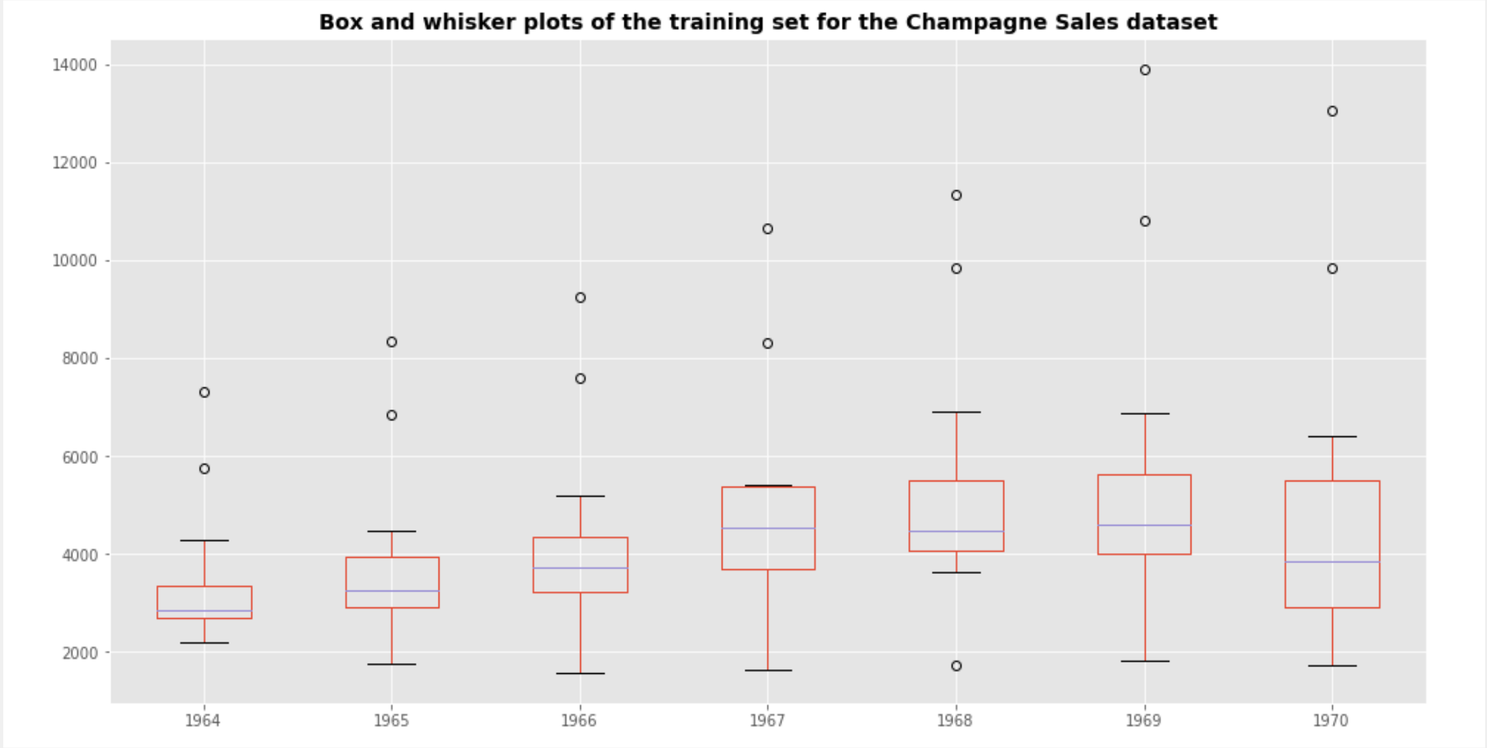
```
<AxesSubplot:ylabel='Count'>
```

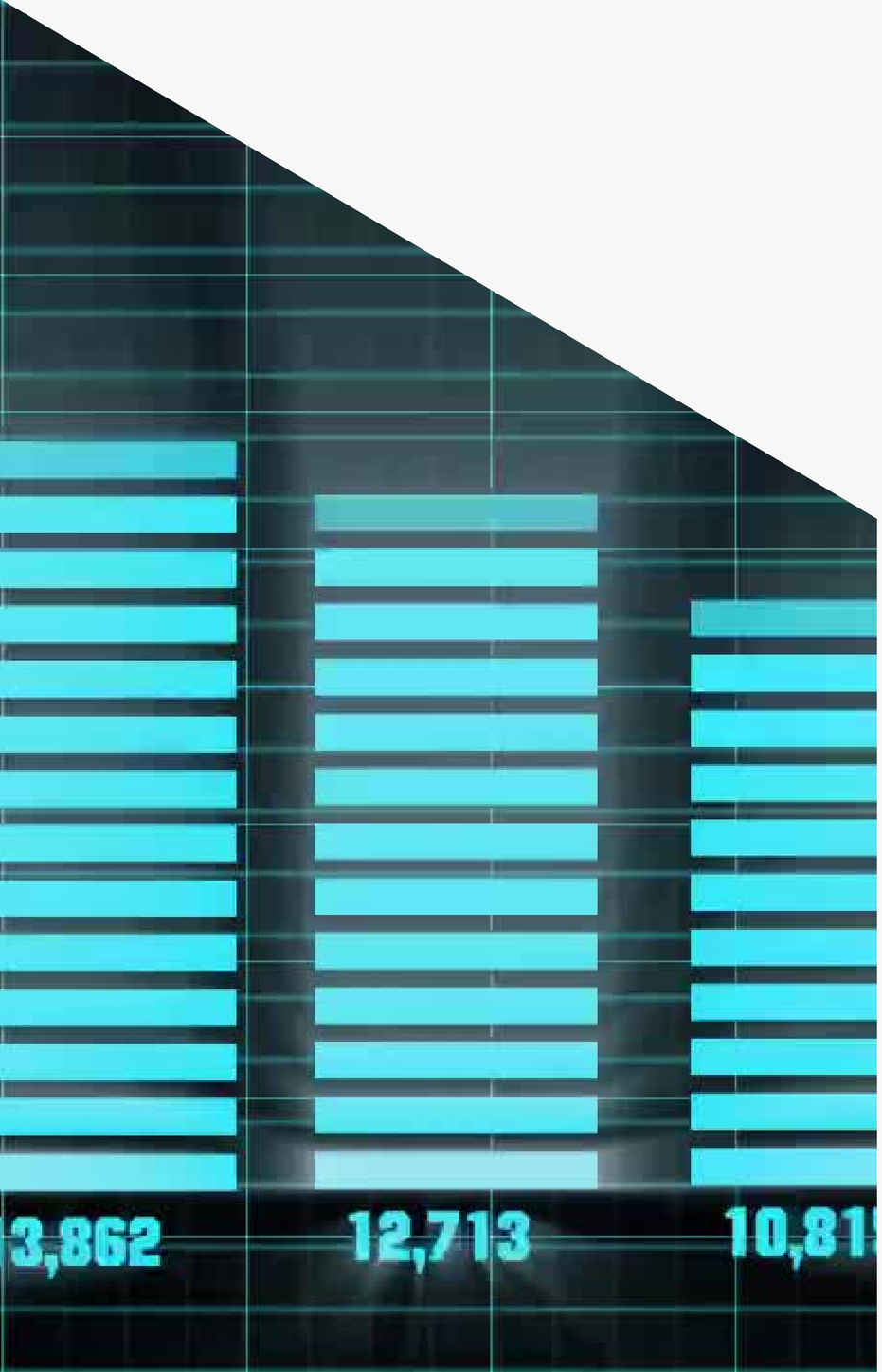




Data Analysis

5. Box and Whisker Plots





Data Analysis

6. Augmented Dicky Fuller Test (ADF)

```python
from statsmodels.tsa.stattools import adfuller


# Perform the ADF test
result = adfuller(series.values)

# Print the test statistic and p-value
print('ADF Statistic: %f' % result[0])
print('p-value: %f' % result[1])
```

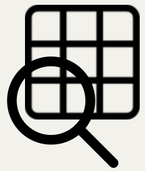
```
ADF Statistic: -1.445970
p-value: 0.560050
```

- Le test d'adfuller confirme la non-stationnarité des données , car $pvalue$=0.56>0.01=1%
- l'hypothese $H0$ stipule que nos données ont une racine unitaire , cad qu'ils sont non stationnaires , et l'hypothese alternative $H1$ stipule que les données sont stationnaires .
- Donc on accepte l'hypothèse nulle ici .

Modelisation ARIMA

01

Grid Search des Hyperparamètres



02

Vérification des résidus

03

Bias-Correction

04

ACF and PACF pour les résidus

05

Validation du modèle

ARIMA

1. Grid Search des Hyperparamètres

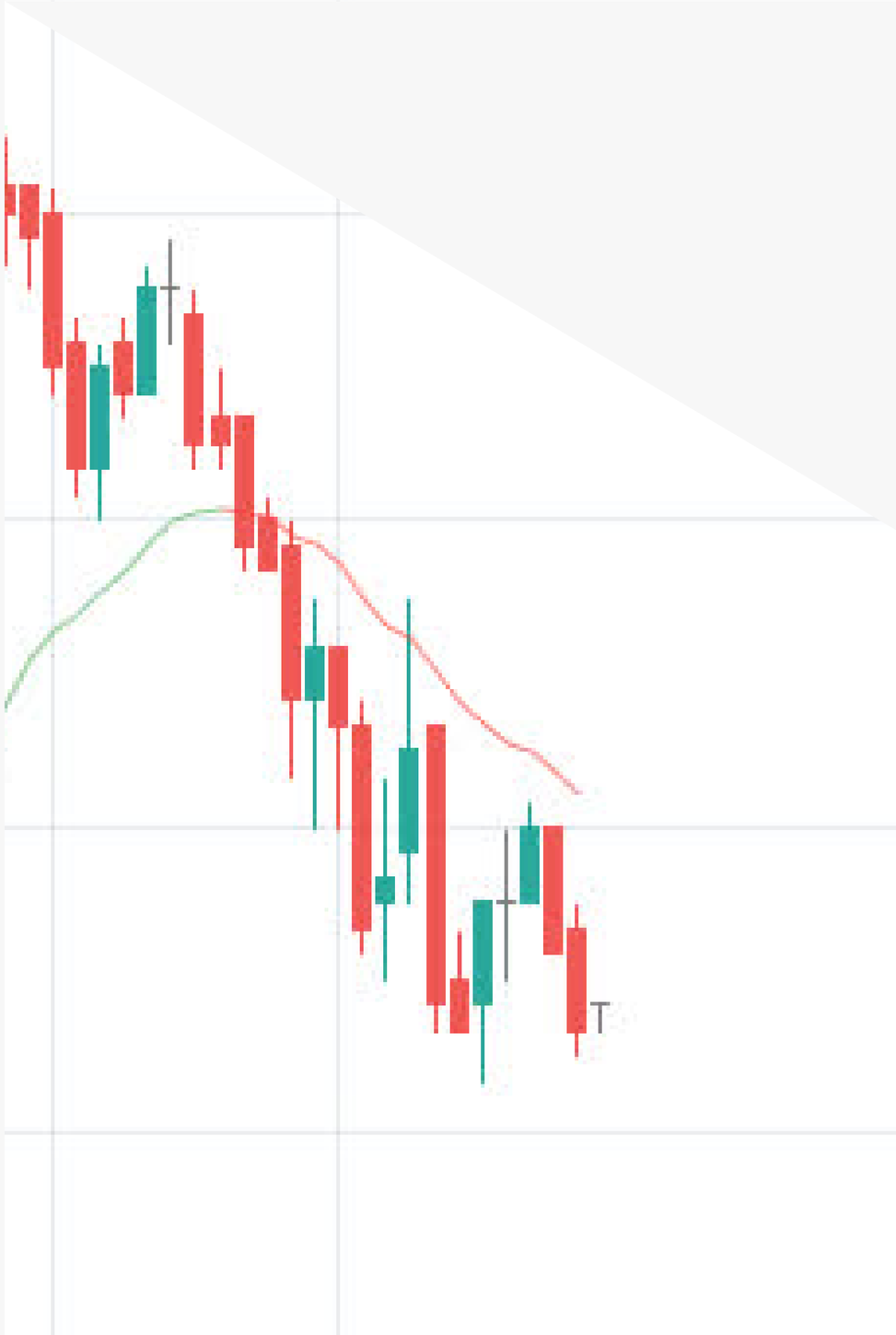
Nous pouvons rechercher sur une grille(Grid) une suite d'hyperparamètres ARIMA et vérifier qu'aucun modèle n'entraîne de meilleures performances RMSE hors échantillon.

Dans cette section, nous rechercherons les valeurs de p, d et q pour les combinaisons (en sautant celles qui ne convergent pas), et trouver la combinaison qui donne les meilleures performances sur l'ensemble de test. Nous utiliserons un Grid Search pour explorer toutes les combinaisons dans un sous-ensemble de valeurs entières. Plus précisément, nous allons rechercher tous combinaisons des paramètres suivants :

p : range(0,2) d : range(0,2) q : range(0,2)

Résultats :

```
ARIMA(0, 0, 0) RMSE=947.677
ARIMA(1, 0, 0) RMSE=945.107
Best ARIMA(1, 0, 0) RMSE=945.107
```



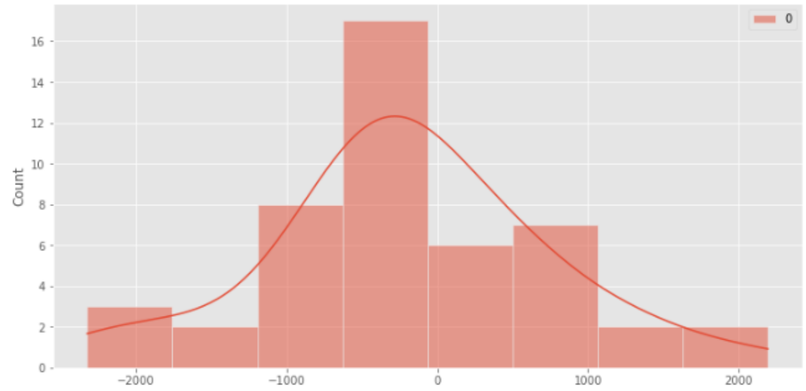
ARIMA

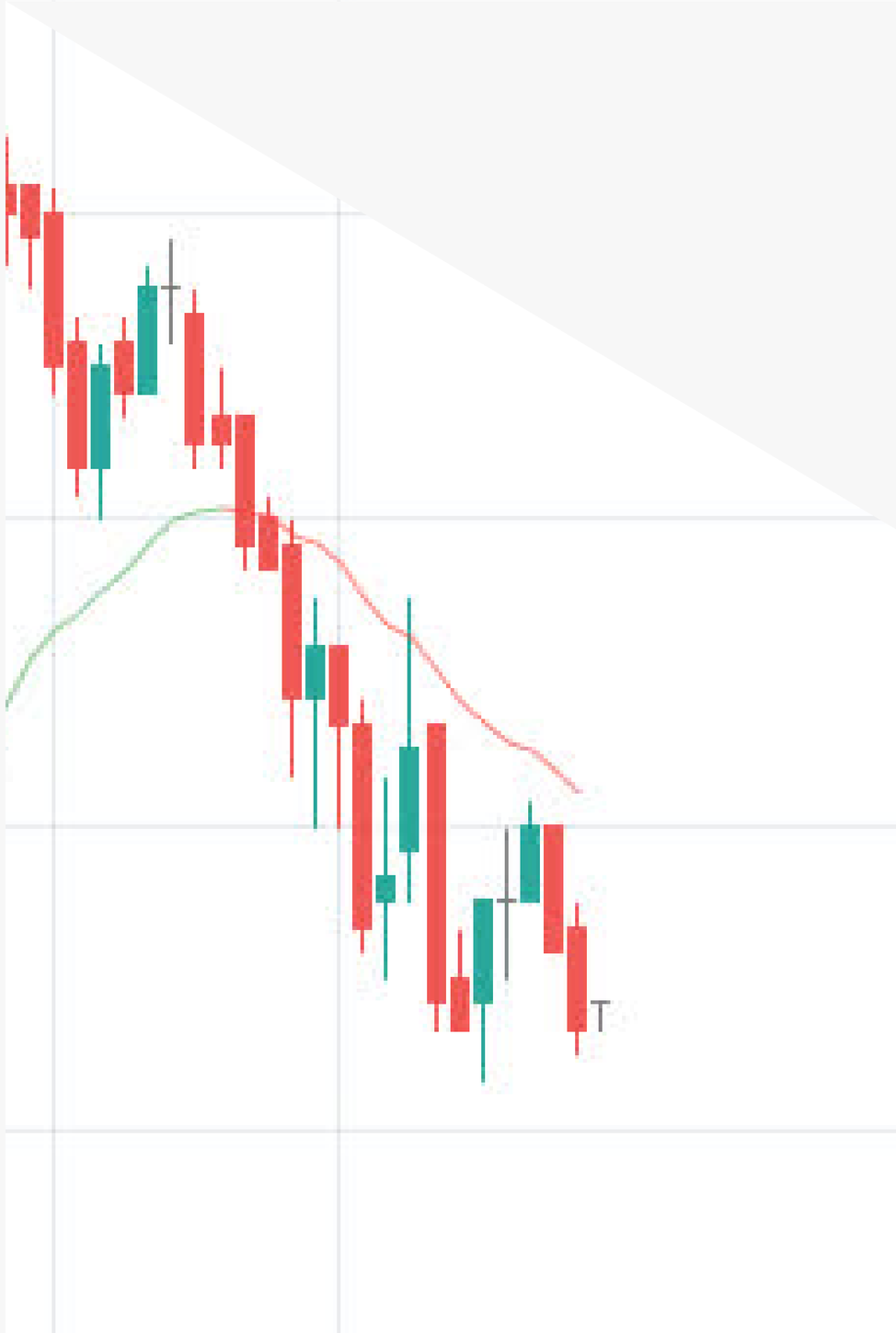
2. Vérification des résidus

Une bonne vérification finale d'un modèle consiste à examiner les erreurs de prévision résiduelles. Idéalement, la distribution de les erreurs résiduelles doivent être une gaussienne avec une moyenne nulle. Nous pouvons vérifier cela en utilisant le résumé statistiques et graphiques pour étudier les erreurs résiduelles du modèle ARIMA(1,0,0).

Résultats :

	0
count	47.000000
mean	-196.036836
std	934.547653
min	-2321.638606
25%	-685.704035
50%	-240.907923
75%	362.380192
max	2190.719757





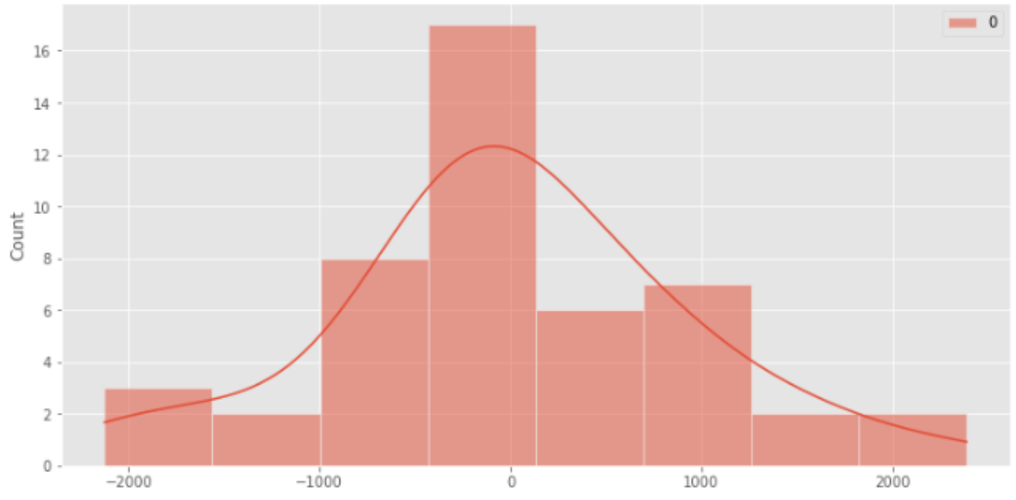
ARIMA

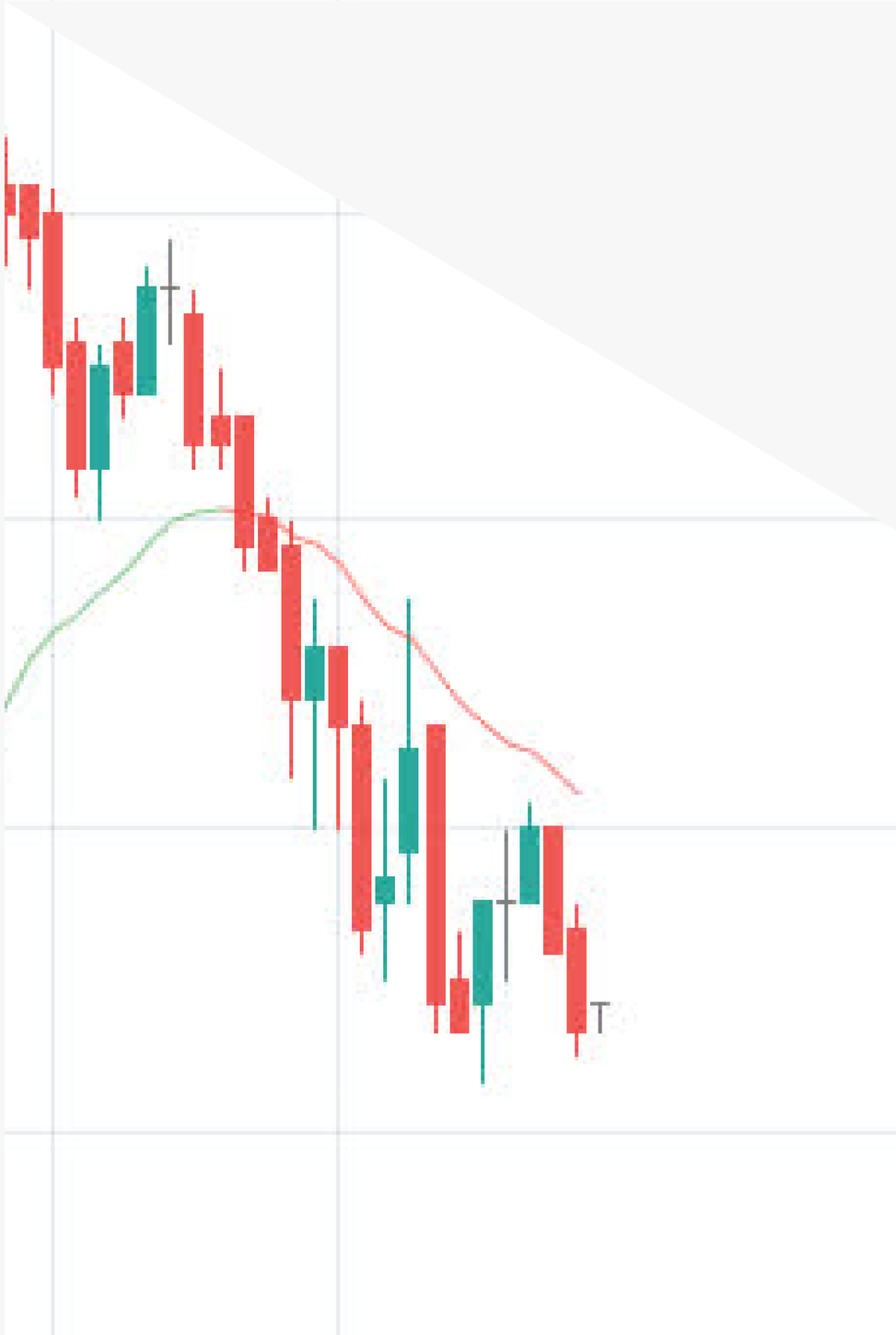
3. Bias-Correction

Nous pourrions utiliser ces informations pour corriger les biais des prédictions en ajoutant l'erreur résiduelle moyenne de -196.036836 à chaque prévision faite.

Résultats :

```
RMSE: 924.552
                0
count  4.700000e+01
mean  -1.006982e-07
std    9.345477e+02
min   -2.125602e+03
25%   -4.896672e+02
50%   -4.487109e+01
75%    5.584170e+02
max    2.386757e+03
```



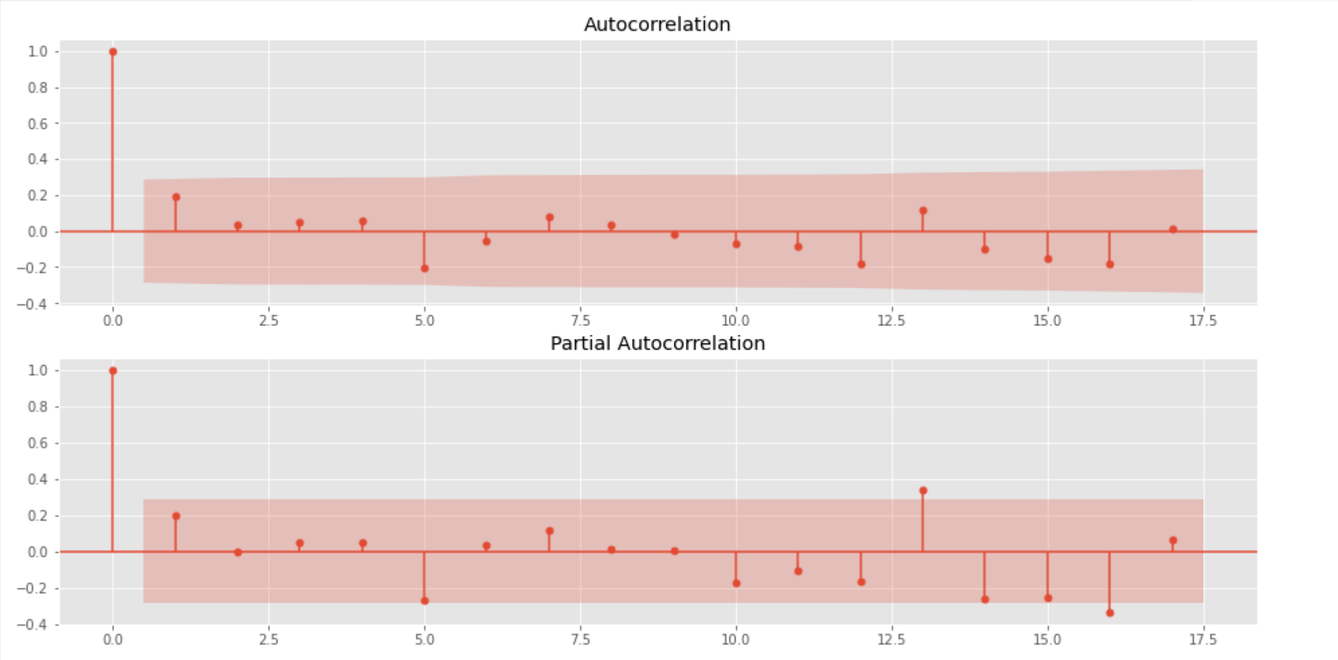


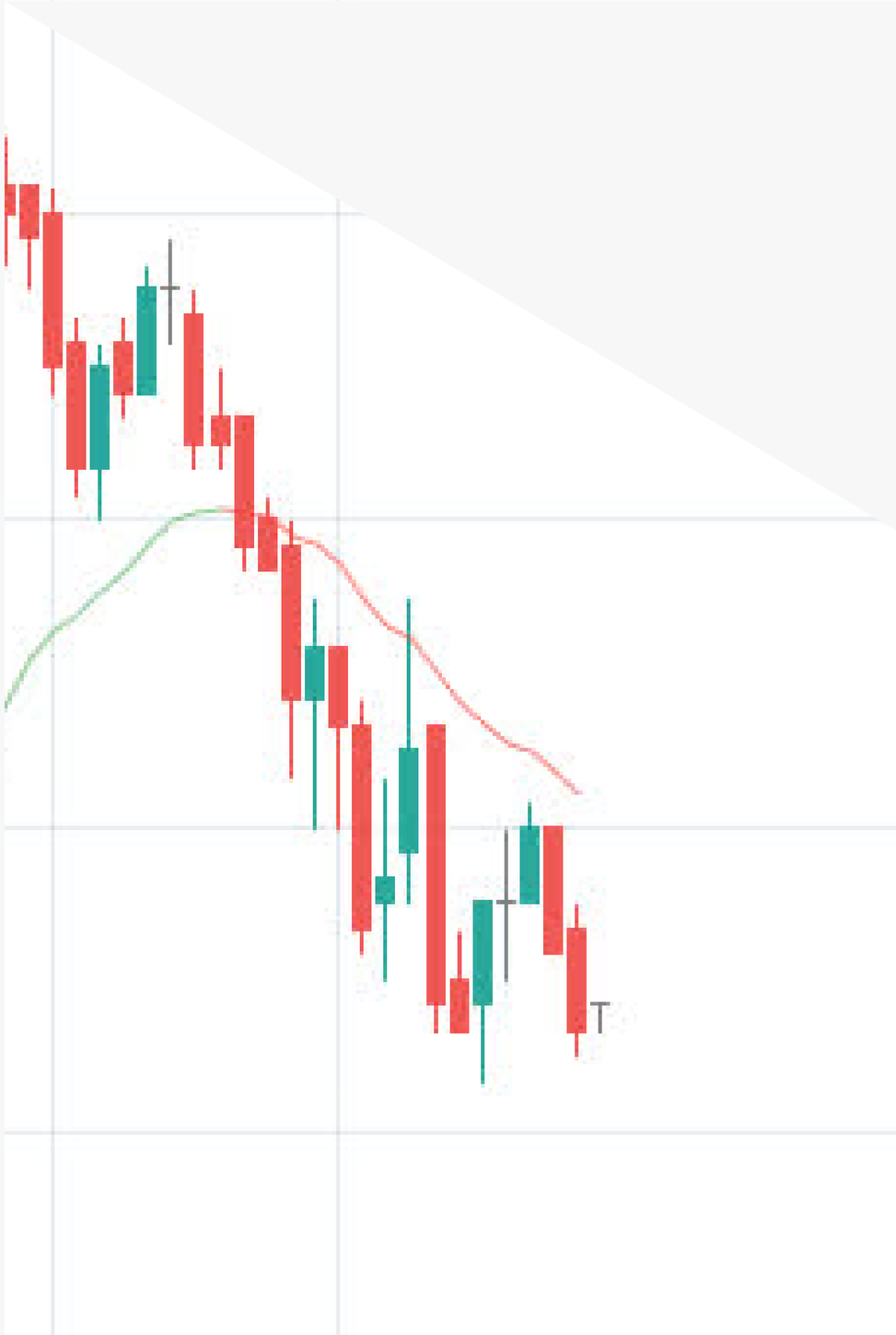
ARIMA

4. ACF and PACF plots

Vérifier la time series des erreurs résiduelles pour tout type d'autocorrélation.

Résultats :



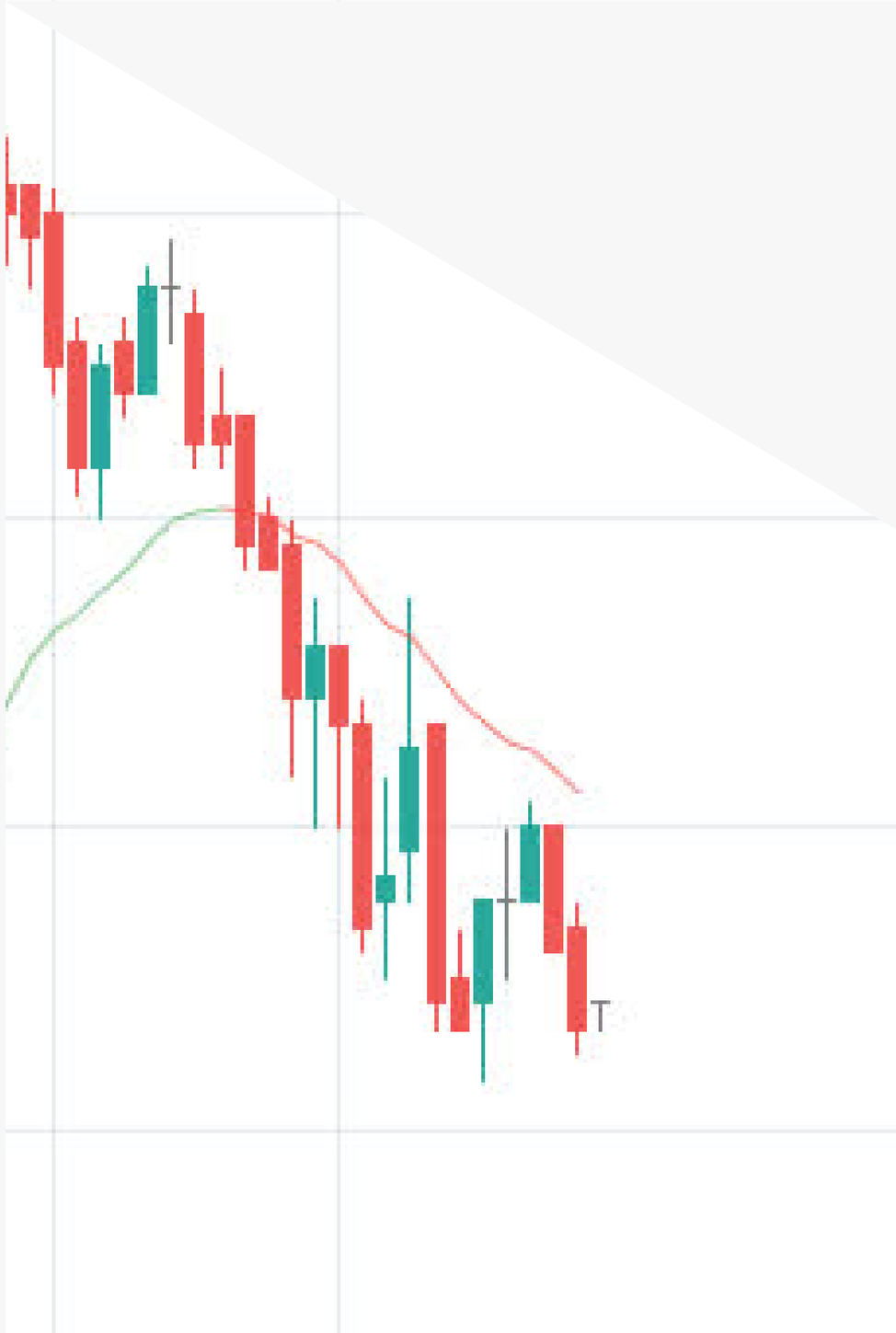


ARIMA

Validation du Modèle

Une fois que les modèles ont été développés et qu'un modèle final a été sélectionné, il doit être validé et finalisé. La validation est une partie facultative du processus, mais qui fournit une dernière vérification pour s'assurer que nous ne nous sommes pas trompés ou induits en erreur. Cette section comprend les étapes suivantes :

- Finaliser le modèle : entraînez et enregistrez le modèle final.
- Faire une prédiction : chargez le modèle finalisé et faites une prédiction.
- Valider le modèle : chargez et validez le modèle final.



ARIMA

Validation du Modèle

Résultats :

```
>Predicted=13132.592, Expected=12670.000
>Predicted=3893.654, Expected=4348.000
>Predicted=3298.008, Expected=3564.000
>Predicted=4420.336, Expected=4577.000
>Predicted=4786.736, Expected=4788.000
>Predicted=5081.173, Expected=4618.000
>Predicted=4830.852, Expected=5312.000
>Predicted=4767.798, Expected=4298.000
>Predicted=1627.129, Expected=1413.000
>Predicted=5930.366, Expected=5877.000
RMSE: 339.095
```