In [2]:

```
!conda install -c conda-forge fbprophet -y
```

```
Collecting package metadata (current_repodata.json): ...working... done
Solving environment: ...working... done

# All requested packages already installed.
```

In [3]:

```
!pip install --upgrade plotly
```

```
Requirement already up-to-date: plotly in c:\users\bviei\anaconda3\lib\site-packages (4.
14.3)
Requirement already satisfied, skipping upgrade: retrying>=1.3.3 in c:\users\bviei\anaco
nda3\lib\site-packages (from plotly) (1.3.3)
Requirement already satisfied, skipping upgrade: six in c:\users\bviei\anaconda3\lib\sit
e-packages (from plotly) (1.15.0)
```

In [4]:

```
!pip install fbprophet
```

```
Requirement already satisfied: fbprophet in c:\users\bviei\anaconda3\lib\site-packages
(0.7.1)
Requirement already satisfied: LunarCalendar>=0.0.9 in c:\users\bviei\anaconda3\lib\site
-packages (from fbprophet) (0.0.9)
Requirement already satisfied: matplotlib>=2.0.0 in c:\users\bviei\anaconda3\lib\site-pa
ckages (from fbprophet) (3.3.2)
Requirement already satisfied: convertdate>=2.1.2 in c:\users\bviei\anaconda3\lib\site-p
ackages (from fbprophet) (2.1.3)
Requirement already satisfied: setuptools-git>=1.2 in c:\users\bviei\anaconda3\lib\site-
packages (from fbprophet) (1.2)
Requirement already satisfied: holidays>=0.10.2 in c:\users\bviei\anaconda3\lib\site-pac
kages (from fbprophet) (0.10.4)
Requirement already satisfied: numpy>=1.15.4 in c:\users\bviei\anaconda3\lib\site-packag
es (from fbprophet) (1.19.2)
Requirement already satisfied: python-dateutil>=2.8.0 in c:\users\bviei\anaconda3\lib\si
te-packages (from fbprophet) (2.8.1)
Requirement already satisfied: tqdm>=4.36.1 in c:\users\bviei\anaconda3\lib\site-package
s (from fbprophet) (4.50.2)
Requirement already satisfied: Cython>=0.22 in c:\users\bviei\anaconda3\lib\site-package
s (from fbprophet) (0.29.17)
Requirement already satisfied: pystan>=2.14 in c:\users\bviei\anaconda3\lib\site-package
s (from fbprophet) (2.19.1.1)
Requirement already satisfied: pandas>=1.0.4 in c:\users\bviei\anaconda3\lib\site-packag
es (from fbprophet) (1.1.3)
Requirement already satisfied: cmdstanpy==0.9.5 in c:\users\bviei\anaconda3\lib\site-pac
kages (from fbprophet) (0.9.5)
Requirement already satisfied: ephem>=3.7.5.3 in c:\users\bviei\anaconda3\lib\site-packa
ges (from LunarCalendar>=0.0.9->fbprophet) (3.7.7.1)
Requirement already satisfied: pytz in c:\users\bviei\anaconda3\lib\site-packages (from
LunarCalendar>=0.0.9->fbprophet) (2019.3)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\bviei\anaconda3\lib\site-pa
ckages (from matplotlib>=2.0.0->fbprophet) (1.3.0)
Requirement already satisfied: certifi>=2020.06.20 in c:\users\bviei\anaconda3\lib\site-
packages (from matplotlib>=2.0.0->fbprophet) (2020.12.5)
Requirement already satisfied: pillow>=6.2.0 in c:\users\bviei\anaconda3\lib\site-packag
es (from matplotlib>=2.0.0->fbprophet) (8.0.1)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.3 in c:\users\bvie
i\anaconda3\lib\site-packages (from matplotlib>=2.0.0->fbprophet) (2.4.7)
Requirement already satisfied: cycler>=0.10 in c:\users\bviei\anaconda3\lib\site-package
s (from matplotlib>=2.0.0->fbprophet) (0.10.0)
Requirement already satisfied: korean-lunar-calendar in c:\users\bviei\anaconda3\lib\sit
e-packages (from holidays>=0.10.2->fbprophet) (0.2.1)
Requirement already satisfied: six in c:\users\bviei\anaconda3\lib\site-packages (from h
olidays>=0.10.2->fbprophet) (1.15.0)
```

In [5]:

```
import pandas as pd
import numpy as np
from numpy import sqrt
import matplotlib.pyplot as plt
import plotly.offline as py
import plotly.graph_objs as go
py.init_notebook_mode(connected=True)

from statsmodels.tsa.arima_model import ARIMA
from statsmodels.tsa.stattools import adfuller
from statsmodels.graphics.tsaplots import plot_acf,plot_pacf
from statsmodels.tsa.seasonal import seasonal_decompose
from pandas.plotting import lag_plot
!pip install pmdarima

import sklearn.metrics
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import mean_squared_error
from sklearn.metrics import r2_score
import statsmodels.api as sm
from scipy import stats

import warnings
warnings.filterwarnings("ignore")

from fbprophet import Prophet
from fbprophet.diagnostics import cross_validation, performance_metrics
from fbprophet.plot import plot_cross_validation_metric
```

```
Requirement already satisfied: pmdarima in c:\users\bviei\anaconda3\lib\site-packages
(1.8.0)
Requirement already satisfied: joblib>=0.11 in c:\users\bviei\anaconda3\lib\site-package
s (from pmdarima) (0.17.0)
Requirement already satisfied: numpy>=1.17.3 in c:\users\bviei\anaconda3\lib\site-packag
es (from pmdarima) (1.19.2)
Requirement already satisfied: pandas>=0.19 in c:\users\bviei\anaconda3\lib\site-package
s (from pmdarima) (1.1.3)
Requirement already satisfied: scikit-learn>=0.22 in c:\users\bviei\anaconda3\lib\site-p
ackages (from pmdarima) (0.23.2)
Requirement already satisfied: urllib3 in c:\users\bviei\anaconda3\lib\site-packages (fr
om pmdarima) (1.25.11)
Requirement already satisfied: Cython<0.29.18,>=0.29 in c:\users\bviei\anaconda3\lib\sit
e-packages (from pmdarima) (0.29.17)
Requirement already satisfied: statsmodels!=0.12.0,>=0.11 in c:\users\bviei\anaconda3\li
b\site-packages (from pmdarima) (0.12.1)
Requirement already satisfied: setuptools!=50.0.0,>=38.6.0 in c:\users\bviei\anaconda3\l
ib\site-packages (from pmdarima) (50.3.1.post20201107)
Requirement already satisfied: scipy>=1.3.2 in c:\users\bviei\anaconda3\lib\site-package
s (from pmdarima) (1.5.2)
Requirement already satisfied: python-dateutil>=2.7.3 in c:\users\bviei\anaconda3\lib\si
te-packages (from pandas>=0.19->pmdarima) (2.8.1)
Requirement already satisfied: pytz>=2017.2 in c:\users\bviei\anaconda3\lib\site-package
s (from pandas>=0.19->pmdarima) (2019.3)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\users\bviei\anaconda3\lib\site
-packages (from scikit-learn>=0.22->pmdarima) (2.1.0)
Requirement already satisfied: patsy>=0.5 in c:\users\bviei\anaconda3\lib\site-packages
(from statsmodels!=0.12.0,>=0.11->pmdarima) (0.5.1)
Requirement already satisfied: six>=1.5 in c:\users\bviei\anaconda3\lib\site-packages (f
rom python-dateutil>=2.7.3->pandas>=0.19->pmdarima) (1.15.0)
```

In [6]:

```
brent = pd.read_csv("brent.csv", usecols=[0,1,2,3,4,5,6], decimal=",")
wti = pd.read_csv('wti.csv', usecols=[0,1,2,3,4,5,6], decimal=",")
```

In [7]:

```
brent
```

Out[7]:

	Data	Último	Abertura	Máxima	Mínima	Vol.	Var%
0	31.12.2019	66.00	66.65	66.93	65.63	171,01K	-3,57%
1	30.12.2019	68.44	68.20	68.99	68.16	29,42K	0,41%
2	27.12.2019	68.16	67.91	68.33	67.57	112,22K	0,35%
3	26.12.2019	67.92	67.27	67.99	67.22	69,82K	1,07%
4	24.12.2019	67.20	66.44	67.26	66.36	104,94K	1,22%
...	...	...	...	...	...	...	...
2576	08.01.2010	81.37	81.51	82.05	80.59	157,49K	-0,17%
2577	07.01.2010	81.51	82.00	82.05	81.05	131,28K	-0,46%
2578	06.01.2010	81.89	80.38	82.21	79.77	157,87K	1,61%
2579	05.01.2010	80.59	80.29	80.84	79.75	131,75K	0,59%
2580	04.01.2010	80.12	78.49	80.48	78.34	122,64K	2,81%

2581 rows × 7 columns

In [8]:

```
wti
```

Out[8]:

	Data	Último	Abertura	Máxima	Mínima	Vol.	Var%
0	31.12.2019	61.06	61.68	61.88	60.63	494,54K	-1,01%
1	30.12.2019	61.68	61.71	62.34	61.09	427,15K	-0,06%
2	27.12.2019	61.72	61.73	61.97	61.24	351,90K	0,06%
3	26.12.2019	61.68	61.20	61.83	61.06	265,09K	0,80%
4	25.12.2019	61.19	61.45	61.52	61.17	-	0,13%
...	...	...	...	...	...	...	...
2586	08.01.2010	82.75	82.65	83.47	81.80	310,38K	0,11%
2587	07.01.2010	82.66	83.20	83.36	82.26	246,63K	-0,63%
2588	06.01.2010	83.18	81.43	83.52	80.85	370,06K	1,72%
2589	05.01.2010	81.77	81.63	82.00	80.95	258,89K	0,32%
2590	04.01.2010	81.51	79.63	81.79	79.63	263,54K	2,71%

2591 rows × 7 columns

In [9]:

```
#DATAS - adequa as datas e também as define como index (o que facilitará a indentificaç

brent['Data'] = pd.to_datetime(brent['Data'], format='%d.%m.%Y')
wti['Data'] = pd.to_datetime(wti['Data'], format='%d.%m.%Y')
```

```
brent.index = pd.to_datetime(brent.Data, format='%d.%m.%Y')
wti.index = pd.to_datetime(wti.Data, format='%d.%m.%Y')

brent.index.to_period('D')
wti.index.to_period('D')
```

Out[9]:

```
PeriodIndex(['2019-12-31', '2019-12-30', '2019-12-27', '2019-12-26',
             '2019-12-25', '2019-12-24', '2019-12-23', '2019-12-20',
             '2019-12-19', '2019-12-18',
             ...
             '2010-01-15', '2010-01-14', '2010-01-13', '2010-01-12',
             '2010-01-11', '2010-01-08', '2010-01-07', '2010-01-06',
             '2010-01-05', '2010-01-04'],
            dtype='period[D]', name='Data', length=2591, freq='D')
```

In [10]:

```
#CAMPOS COM VALORES - ajuste nos números e também no campo volume, que cotinha letras i
brent["Abertura"] = brent["Abertura"].astype(str)
brent["Último"] = brent["Último"].astype(str)
brent["Máxima"] = brent["Máxima"].astype(str)
brent["Mínima"] = brent["Mínima"].astype(str)
brent["Var%"] = brent["Var%"].astype(str)
brent["Vol."] = brent["Vol."].astype(str)

brent["Abertura"]=pd.Series(brent["Abertura"]).str.replace(',', '.', regex=True)
brent["Último"]=pd.Series(brent["Último"]).str.replace(',', '.', regex=True)
brent["Máxima"]=pd.Series(brent["Máxima"]).str.replace(',', '.', regex=True)
brent["Mínima"]=pd.Series(brent["Mínima"]).str.replace(',', '.', regex=True)
brent["Var%"]=pd.Series(brent["Var%"]).str.replace(',', '.', regex=True)
brent["Var%"]=pd.Series(brent["Var%"]).str.replace('%', '', regex=True)
brent["Vol."]=pd.Series(brent["Vol."]).str.replace('-', '', regex=True)
brent["Vol."]=pd.Series(brent["Vol."]).str.replace(',', '', regex=True)
brent["Vol."]=pd.Series(brent["Vol."]).str.replace('M',"000000", regex=True)
brent["Vol."]=pd.Series(brent["Vol."]).str.replace('K',"000", regex=True)
brent["Vol."]=pd.Series(brent["Vol."]).str.replace('B',"000000000", regex=True)

brent["Abertura"] = pd.to_numeric(brent["Abertura"])
brent["Último"] = pd.to_numeric(brent["Último"])
brent["Máxima"] = pd.to_numeric(brent["Máxima"])
brent["Mínima"] = pd.to_numeric(brent["Mínima"])
brent["Var%"] = pd.to_numeric(brent["Var%"])
brent["Vol."] = pd.to_numeric(brent["Vol."])
```

In [11]:

```
#CAMPOS COM VALORES - ajuste nos números e também no campo volume, que cotinha letras i
wti["Abertura"] = wti["Abertura"].astype(str)
wti["Último"] = wti["Último"].astype(str)
wti["Máxima"] = wti["Máxima"].astype(str)
wti["Mínima"] = wti["Mínima"].astype(str)
wti["Var%"] = wti["Var%"].astype(str)
wti["Vol."] = wti["Vol."].astype(str)

wti["Abertura"]=pd.Series(wti["Abertura"]).str.replace(',', '.', regex=True)
wti["Último"]=pd.Series(wti["Último"]).str.replace(',', '.', regex=True)
wti["Máxima"]=pd.Series(wti["Máxima"]).str.replace(',', '.', regex=True)
wti["Mínima"]=pd.Series(wti["Mínima"]).str.replace(',', '.', regex=True)
wti["Var%"]=pd.Series(wti["Var%"]).str.replace(',', '.', regex=True)
wti["Var%"]=pd.Series(wti["Var%"]).str.replace('%', '', regex=True)
wti["Vol."]=pd.Series(wti["Vol."]).str.replace('-', '', regex=True)
wti["Vol."]=pd.Series(wti["Vol."]).str.replace(',', '', regex=True)
wti["Vol."]=pd.Series(wti["Vol."]).str.replace('M',"000000", regex=True)
wti["Vol."]=pd.Series(wti["Vol."]).str.replace('K',"000", regex=True)
```

```
wti["Vol."]=pd.Series(wti["Vol."]).str.replace('B',"000000000", regex=True)

wti["Abertura"] = pd.to_numeric(wti["Abertura"])
wti["Último"] = pd.to_numeric(wti["Último"])
wti["Máxima"] = pd.to_numeric(wti["Máxima"])
wti["Mínima"] = pd.to_numeric(wti["Mínima"])
wti["Var%"] = pd.to_numeric(wti["Var%"])
wti["Vol."] = pd.to_numeric(wti["Vol."])
```

In [12]:

```
brent
```

Out[12]:

	Data	Último	Abertura	Máxima	Mínima	Vol.	Var%
Data							
2019-12-31	2019-12-31	66.00	66.65	66.93	65.63	17101000.0	-3.57
2019-12-30	2019-12-30	68.44	68.20	68.99	68.16	2942000.0	0.41
2019-12-27	2019-12-27	68.16	67.91	68.33	67.57	11222000.0	0.35
2019-12-26	2019-12-26	67.92	67.27	67.99	67.22	6982000.0	1.07
2019-12-24	2019-12-24	67.20	66.44	67.26	66.36	10494000.0	1.22
...	...	...	...	...	...	...	...
2010-01-08	2010-01-08	81.37	81.51	82.05	80.59	15749000.0	-0.17
2010-01-07	2010-01-07	81.51	82.00	82.05	81.05	13128000.0	-0.46
2010-01-06	2010-01-06	81.89	80.38	82.21	79.77	15787000.0	1.61
2010-01-05	2010-01-05	80.59	80.29	80.84	79.75	13175000.0	0.59
2010-01-04	2010-01-04	80.12	78.49	80.48	78.34	12264000.0	2.81

2581 rows × 7 columns

In [13]:

```
wti
```

Out[13]:

	Data	Último	Abertura	Máxima	Mínima	Vol.	Var%
Data							
2019-12-31	2019-12-31	61.06	61.68	61.88	60.63	49454000.0	-1.01
2019-12-30	2019-12-30	61.68	61.71	62.34	61.09	42715000.0	-0.06
2019-12-27	2019-12-27	61.72	61.73	61.97	61.24	35190000.0	0.06
2019-12-26	2019-12-26	61.68	61.20	61.83	61.06	26509000.0	0.80
2019-12-25	2019-12-25	61.19	61.45	61.52	61.17	NaN	0.13
...	...	...	...	...	...	...	...
2010-01-08	2010-01-08	82.75	82.65	83.47	81.80	31038000.0	0.11
2010-01-07	2010-01-07	82.66	83.20	83.36	82.26	24663000.0	-0.63
2010-01-06	2010-01-06	83.18	81.43	83.52	80.85	37006000.0	1.72
2010-01-05	2010-01-05	81.77	81.63	82.00	80.95	25889000.0	0.32

	Data	Último	Abertura	Máxima	Mínima	Vol.	Var%
Data							
2010-01-04	2010-01-04	81.51	79.63	81.79	79.63	26354000.0	2.71

2591 rows × 7 columns

```
In [14]: brent.isnull().sum()
```

```
Out[14]: Data        0
         Último      0
         Abertura    0
         Máxima      0
         Mínima      0
         Vol.        1
         Var%        0
         dtype: int64
```

```
In [15]: wti.isnull().sum()
```

```
Out[15]: Data         0
         Último       0
         Abertura     0
         Máxima       0
         Mínima       0
         Vol.        74
         Var%         0
         dtype: int64
```

```
In [16]: brent_medianaMA = brent["Máxima"].rolling(5).mean().shift(-5).round(0)
         brent_medianaMI = brent["Mínima"].rolling(5).mean().shift(-5).round(0)
         brent_medianaAB = brent["Abertura"].rolling(5).mean().shift(-5).round(0)
         brent_medianaUL = brent["Último"].rolling(5).mean().shift(-5).round(0)
         brent_medianaVOL = brent["Vol."].rolling(5).mean().shift(-5).round(0)
         brent["Máxima"].fillna(brent_medianaMA, inplace=True)
         brent["Mínima"].fillna(brent_medianaMI, inplace=True)
         brent["Abertura"].fillna(brent_medianaAB, inplace=True)
         brent["Último"].fillna(brent_medianaUL, inplace=True)
         brent["Vol."].fillna(brent_medianaVOL, inplace=True)
```

```
In [17]: wti_medianaMA = wti["Máxima"].rolling(5).mean().shift(-5).round(0)
         wti_medianaMI = wti["Mínima"].rolling(5).mean().shift(-5).round(0)
         wti_medianaAB = wti["Abertura"].rolling(5).mean().shift(-5).round(0)
         wti_medianaUL = wti["Último"].rolling(5).mean().shift(-5).round(0)
         wti_medianaVOL = wti["Vol."].rolling(5).mean().shift(-5).round(0)
         wti["Máxima"].fillna(wti_medianaMA, inplace=True)
         wti["Mínima"].fillna(wti_medianaMI, inplace=True)
         wti["Abertura"].fillna(wti_medianaAB, inplace=True)
         wti["Último"].fillna(wti_medianaUL, inplace=True)
         wti["Vol."].fillna(wti_medianaVOL, inplace=True)
```

```
In [18]: brent.isnull().sum()
```

```
Out[18]: Data        0
         Último      0
         Abertura    0
         Máxima      0
         Mínima      0
         Vol.        0
```

```
Var%        0
dtype: int64
```

In [19]:
```
wti.isnull().sum()
```

Out[19]:
```
Data         0
Último       0
Abertura     0
Máxima       0
Mínima       0
Vol.        27
Var%         0
dtype: int64
```

In [20]:
```
wti_medianaVOL = wti["Vol."].rolling(5).mean().shift(-5).round(0)
wti["Vol."].fillna(wti_medianaVOL, inplace=True)
```

In [21]:
```
wti.isnull().sum()
```

Out[21]:
```
Data        0
Último      0
Abertura    0
Máxima      0
Mínima      0
Vol.        1
Var%        0
dtype: int64
```

In [22]:
```
wti_medianaVOL = wti["Vol."].rolling(5).mean().shift(-5).round(0)
wti["Vol."].fillna(wti_medianaVOL, inplace=True)
```

In [23]:
```
wti.isnull().sum()
```

Out[23]:
```
Data        0
Último      0
Abertura    0
Máxima      0
Mínima      0
Vol.        0
Var%        0
dtype: int64
```

In [24]:
```
#JUNÇÃO DAS TABELAS, cria dataset bw

brent.rename(columns= {'Data': 'data'}, inplace=True)
wti.rename(columns= {'Data': 'data'}, inplace=True)

bw = pd.merge(brent,wti,how='inner', on=['data'],suffixes=('_B', '_W'))
```

In [25]:
```
bw.isnull().sum()
```

Out[25]:
```
data          0
Último_B      0
Abertura_B    0
Máxima_B      0
Mínima_B      0
Vol._B        0
Var%_B        0
Último_W      0
Abertura_W    0
Máxima_W      0
Mínima_W      0
```

```
Vol._W        0
Var%_W        0
dtype: int64
```

In [26]:

```
bw
```

Out[26]:

	data	Último_B	Abertura_B	Máxima_B	Mínima_B	Vol._B	Var%_B	Último_W	Abertura_W
0	2019-12-31	66.00	66.65	66.93	65.63	17101000.0	-3.57	61.06	61.68
1	2019-12-30	68.44	68.20	68.99	68.16	2942000.0	0.41	61.68	61.71
2	2019-12-27	68.16	67.91	68.33	67.57	11222000.0	0.35	61.72	61.73
3	2019-12-26	67.92	67.27	67.99	67.22	6982000.0	1.07	61.68	61.20
4	2019-12-24	67.20	66.44	67.26	66.36	10494000.0	1.22	61.11	60.63
...	...	...	...	...	...	...	...	...	...
2555	2010-01-08	81.37	81.51	82.05	80.59	15749000.0	-0.17	82.75	82.65
2556	2010-01-07	81.51	82.00	82.05	81.05	13128000.0	-0.46	82.66	83.20
2557	2010-01-06	81.89	80.38	82.21	79.77	15787000.0	1.61	83.18	81.43
2558	2010-01-05	80.59	80.29	80.84	79.75	13175000.0	0.59	81.77	81.63
2559	2010-01-04	80.12	78.49	80.48	78.34	12264000.0	2.81	81.51	79.63

2560 rows × 13 columns

In [27]:

```
bw.index = pd.to_datetime(bw.data, format='%d.%m.%Y')
```

In [28]:

```
bw
```

Out[28]:

	data	Último_B	Abertura_B	Máxima_B	Mínima_B	Vol._B	Var%_B	Último_W	Abertura_W
data									
2019-12-31	2019-12-31	66.00	66.65	66.93	65.63	17101000.0	-3.57	61.06	61.68
2019-12-30	2019-12-30	68.44	68.20	68.99	68.16	2942000.0	0.41	61.68	61.71
2019-12-27	2019-12-27	68.16	67.91	68.33	67.57	11222000.0	0.35	61.72	61.73
2019-12-26	2019-12-26	67.92	67.27	67.99	67.22	6982000.0	1.07	61.68	61.20

	data	Último_B	Abertura_B	Máxima_B	Mínima_B	Vol._B	Var%_B	Último_W	Abertura_W
data									
2019-12-24	2019-12-24	67.20	66.44	67.26	66.36	10494000.0	1.22	61.11	60.63
...	...	...	...	...	...	...	...	...	..
2010-01-08	2010-01-08	81.37	81.51	82.05	80.59	15749000.0	-0.17	82.75	82.65
2010-01-07	2010-01-07	81.51	82.00	82.05	81.05	13128000.0	-0.46	82.66	83.20
2010-01-06	2010-01-06	81.89	80.38	82.21	79.77	15787000.0	1.61	83.18	81.43
2010-01-05	2010-01-05	80.59	80.29	80.84	79.75	13175000.0	0.59	81.77	81.63
2010-01-04	2010-01-04	80.12	78.49	80.48	78.34	12264000.0	2.81	81.51	79.63

2560 rows × 13 columns

In [29]:

```
brent1 = pd.DataFrame(columns={"data","Abertura_B", "Máxima_B", "Mínima_B","Último_B",
brent1["data"]= bw["data"]
brent1["Abertura_B"]= bw["Abertura_B"]
brent1["Máxima_B"]= bw["Máxima_B"]
brent1["Mínima_B"]= bw["Mínima_B"]
brent1["Último_B"]= bw["Último_B"]
brent1["Vol._B"]= bw["Vol._B"]
```

In [30]:

```
brent1
```

Out[30]:

	Abertura_B	Último_B	Máxima_B	data	Vol._B	Mínima_B
data						
2019-12-31	66.65	66.00	66.93	2019-12-31	17101000.0	65.63
2019-12-30	68.20	68.44	68.99	2019-12-30	2942000.0	68.16
2019-12-27	67.91	68.16	68.33	2019-12-27	11222000.0	67.57
2019-12-26	67.27	67.92	67.99	2019-12-26	6982000.0	67.22
2019-12-24	66.44	67.20	67.26	2019-12-24	10494000.0	66.36
...	...	...	...	...	...	...
2010-01-08	81.51	81.37	82.05	2010-01-08	15749000.0	80.59
2010-01-07	82.00	81.51	82.05	2010-01-07	13128000.0	81.05
2010-01-06	80.38	81.89	82.21	2010-01-06	15787000.0	79.77
2010-01-05	80.29	80.59	80.84	2010-01-05	13175000.0	79.75
2010-01-04	78.49	80.12	80.48	2010-01-04	12264000.0	78.34

2560 rows × 6 columns

In [31]:

```
wti1 = pd.DataFrame(columns={"data","Abertura_W", "Máxima_W", "Mínima_W","Último_W", "V
wti1["data"]= bw["data"]
wti1["Abertura_W"]= bw["Abertura_W"]
wti1["Máxima_W"]= bw["Máxima_W"]
wti1["Mínima_W"]= bw["Mínima_W"]
wti1["Último_W"]= bw["Último_W"]
wti1["Vol._W"]= bw["Vol._W"]
```

In [32]:

```
wti1
```

Out[32]:

	Mínima_W	data	Abertura_W	Vol._W	Máxima_W	Último_W
data						
2019-12-31	60.63	2019-12-31	61.68	49454000.0	61.88	61.06
2019-12-30	61.09	2019-12-30	61.71	42715000.0	62.34	61.68
2019-12-27	61.24	2019-12-27	61.73	35190000.0	61.97	61.72
2019-12-26	61.06	2019-12-26	61.20	26509000.0	61.83	61.68
2019-12-24	60.47	2019-12-24	60.63	20454000.0	61.16	61.11
...	...	...	...	...	...	...
2010-01-08	81.80	2010-01-08	82.65	31038000.0	83.47	82.75
2010-01-07	82.26	2010-01-07	83.20	24663000.0	83.36	82.66
2010-01-06	80.85	2010-01-06	81.43	37006000.0	83.52	83.18
2010-01-05	80.95	2010-01-05	81.63	25889000.0	82.00	81.77
2010-01-04	79.63	2010-01-04	79.63	26354000.0	81.79	81.51

2560 rows × 6 columns

In [33]:

```
brent1.rename(columns= {'data': 'Data'}, inplace=True)
wti1.rename(columns= {'data': 'Data'}, inplace=True)
```

In [34]:

```
brent1
```

Out[34]:

	Abertura_B	Último_B	Máxima_B	Data	Vol._B	Mínima_B
data						
2019-12-31	66.65	66.00	66.93	2019-12-31	17101000.0	65.63
2019-12-30	68.20	68.44	68.99	2019-12-30	2942000.0	68.16
2019-12-27	67.91	68.16	68.33	2019-12-27	11222000.0	67.57
2019-12-26	67.27	67.92	67.99	2019-12-26	6982000.0	67.22
2019-12-24	66.44	67.20	67.26	2019-12-24	10494000.0	66.36
...	...	...	...	...	...	...
2010-01-08	81.51	81.37	82.05	2010-01-08	15749000.0	80.59

	Abertura_B	Último_B	Máxima_B	Data	Vol._B	Mínima_B
data						
2010-01-07	82.00	81.51	82.05	2010-01-07	13128000.0	81.05
2010-01-06	80.38	81.89	82.21	2010-01-06	15787000.0	79.77
2010-01-05	80.29	80.59	80.84	2010-01-05	13175000.0	79.75
2010-01-04	78.49	80.12	80.48	2010-01-04	12264000.0	78.34

2560 rows × 6 columns

In [35]:

```
#Serparar treino e teste

filtroB  = brent1['Data']<= "2016-12-31"
train_B = brent1[filtroB]

filtroW  = wti1['Data']<= "2016-12-31"
train_W = wti1[filtroW]

filtroBteste  = brent1['Data']> "2016-12-31"
teste_B = brent1[filtroBteste]

filtroWteste  = wti1['Data']> "2016-12-31"
teste_W = wti1[filtroWteste]
```

In [36]:

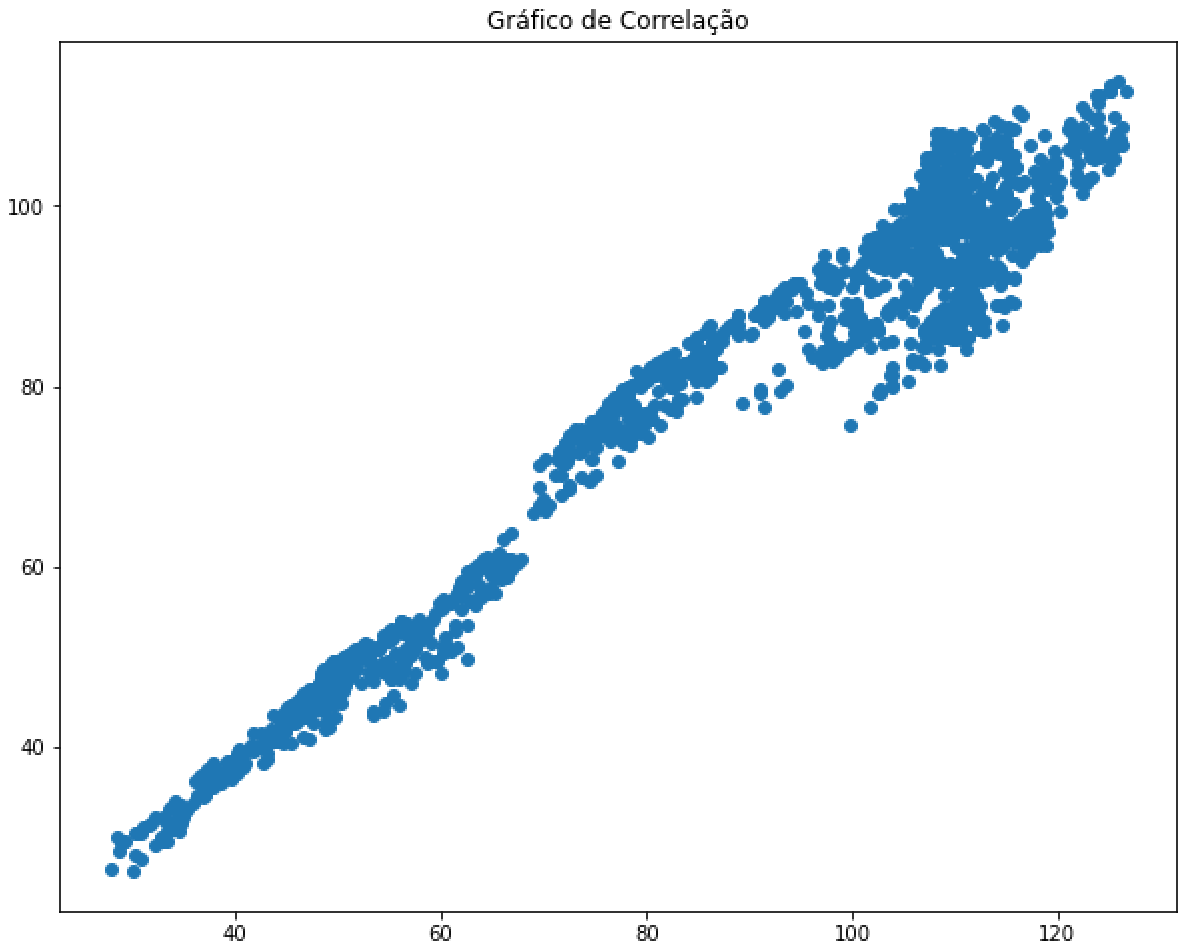
```
#correlação

train_B["Último_B"].corr(train_W["Último_W"])
```

Out[36]:

```
0.9745391323161525
```

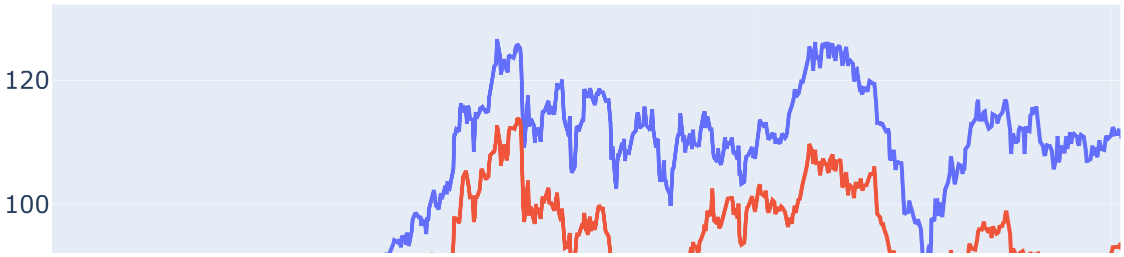
In [37]:

```
data1=train_B["Último_B"]
data2=train_W["Último_W"]

plt.scatter(data1, data2)
plt.title('Gráfico de Correlação')
plt.gcf().set_size_inches(10, 8)
plt.show()

#diretamente proporcionais com alta correlação
```



In [38]:

```
eixo_x = train_B['Data']
linha_brent1_ultimo = train_B["Último_B"]
linha_wti1_ultimo = train_W["Último_W"]
trace1 = go.Scatter(x = eixo_x,y = linha_brent1_ultimo,mode = 'lines', name = 'BRENT')
trace2 = go.Scatter(x = eixo_x,y = linha_wti1_ultimo,mode = 'lines',name = 'WTI')
data = [trace1, trace2]
py.iplot(data)
```



In [39]:

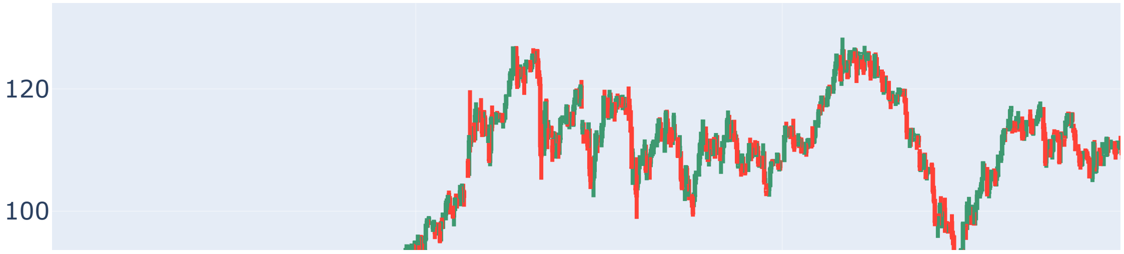
```
train_B.describe()
```

Out[39]:

	Abertura_B	Último_B	Máxima_B	Vol._B	Mínima_B
count	1787.000000	1787.000000	1787.000000	1.787000e+03	1787.000000
mean	86.967857	86.978215	87.963212	1.943510e+07	85.900235
std	27.278378	27.329470	27.340258	7.398364e+06	27.210115
min	27.990000	27.880000	28.750000	1.152000e+06	27.100000
25%	59.580000	59.230000	60.560000	1.535850e+07	58.375000
50%	99.540000	99.650000	100.620000	1.950300e+07	97.920000
75%	109.760000	109.885000	110.800000	2.356550e+07	108.900000
max	126.580000	126.650000	128.400000	4.638100e+07	125.000000

In [40]:

```
import plotly.graph_objects as go



fig = go.Figure(data=[go.Candlestick(x=train_B['Data'],
                open=train_B['Abertura_B'], high=train_B['Máxima_B'],
                low=train_B['Mínima_B'], close=train_B["Último_B"])
                     ])

fig.update_layout(xaxis_rangeslider_visible=False)
fig.show()
```

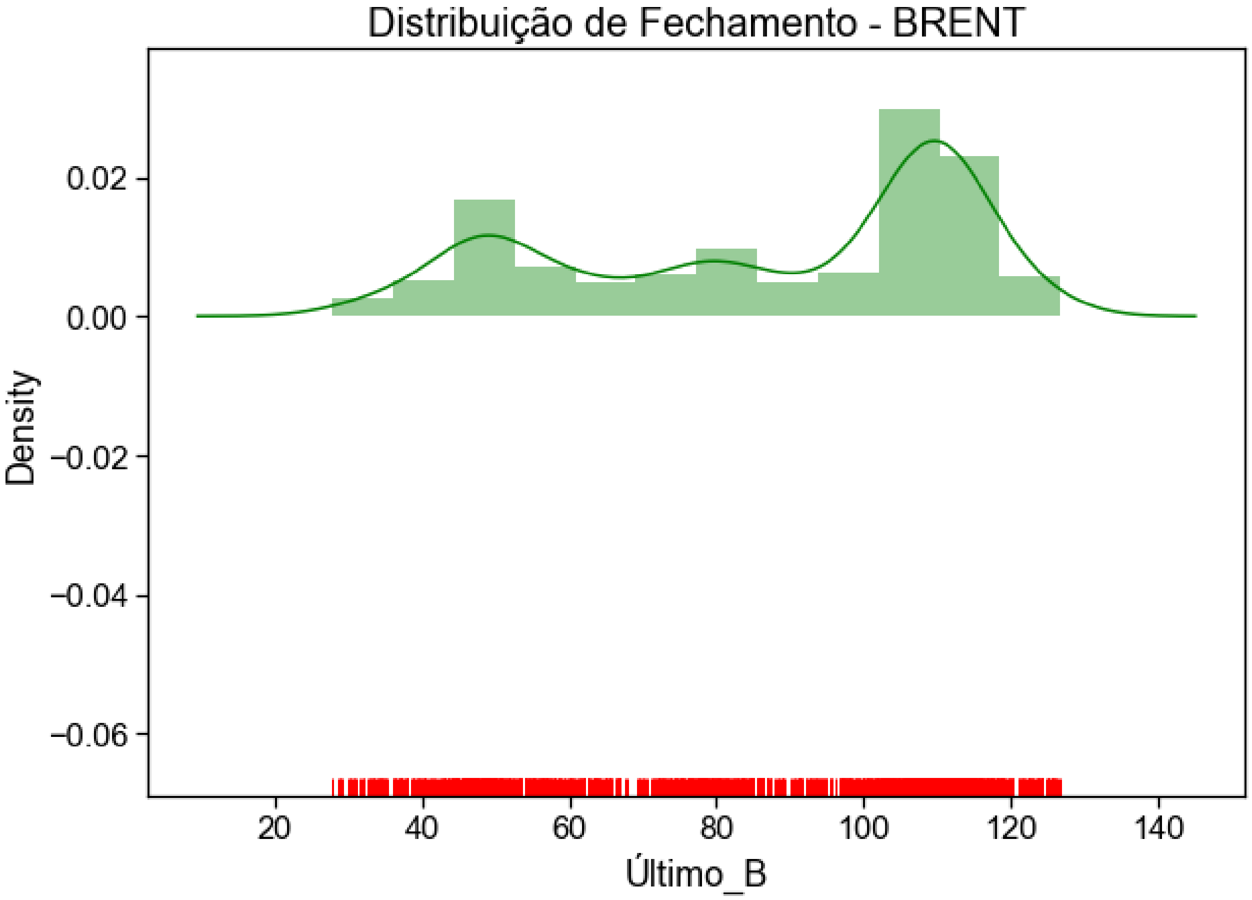


In [41]:

```
import seaborn as sns
plt.figure(figsize=(10,7))
sns.set_context("notebook", font_scale=1.5, rc={'font.size':20, 'axes.titlesize':20, 'a
sns.rugplot(train_B["Último_B"], color ='red')
sns.distplot(train_B["Último_B"], color ='green')
sns.set_style("darkgrid")
plt.title("Distribuição de Fechamento - BRENT")
```

Out[41]:

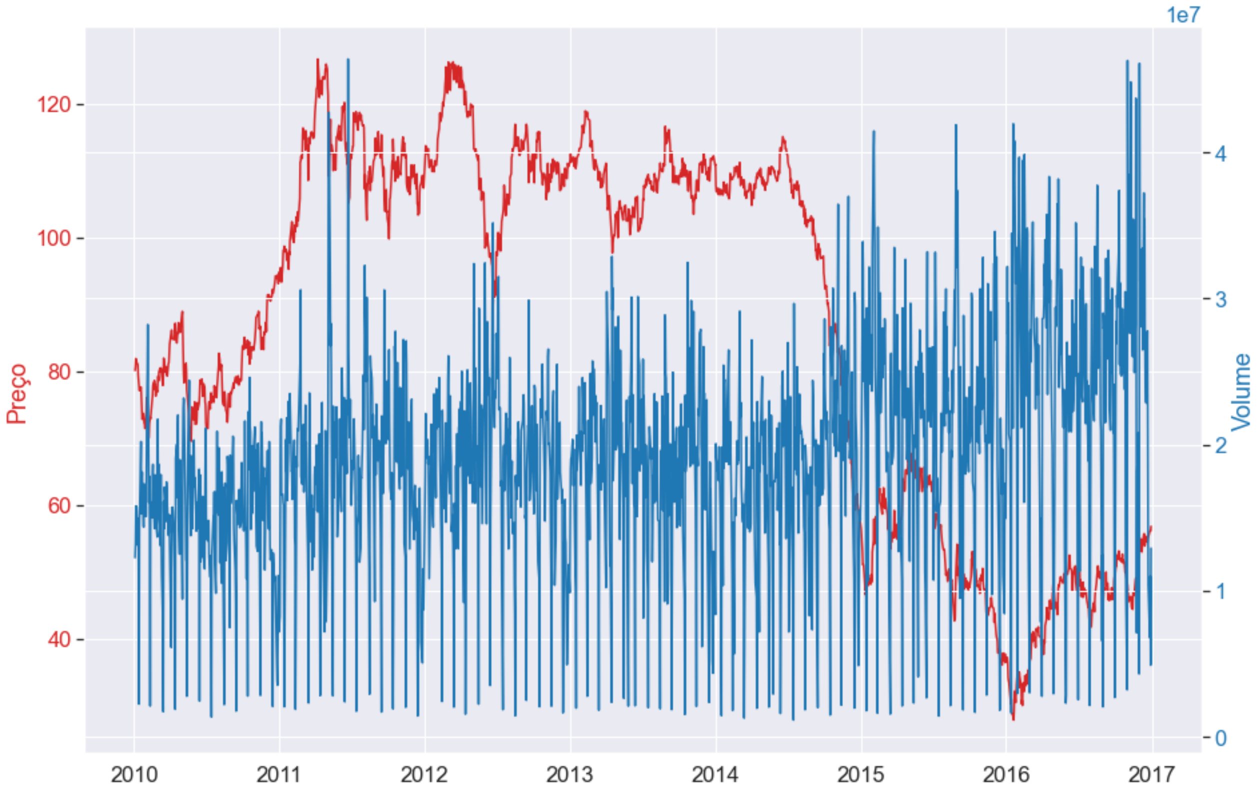
```
Text(0.5, 1.0, 'Distribuição de Fechamento - BRENT')
```



In [42]:

```
#gráfico com dois eixos y

b = train_B["Data"]
data1 = train_B["Último_B"]
data2 = train_B["Vol._B"]
```

```
fig, ax1 = plt.subplots()

color = 'tab:red'
ax1.set_xlabel('')
ax1.set_ylabel('Preço', color=color)
ax1.plot(b, data1, color=color)
ax1.tick_params(axis='y', labelcolor=color)

ax2 = ax1.twinx()

color = 'tab:blue'
ax2.set_ylabel('Volume', color=color)
ax2.plot(b, data2, color=color)
ax2.tick_params(axis='y', labelcolor=color)
plt.gcf().set_size_inches(15, 10)
plt.show()
```



In [43]:

```
train_W.describe()
```

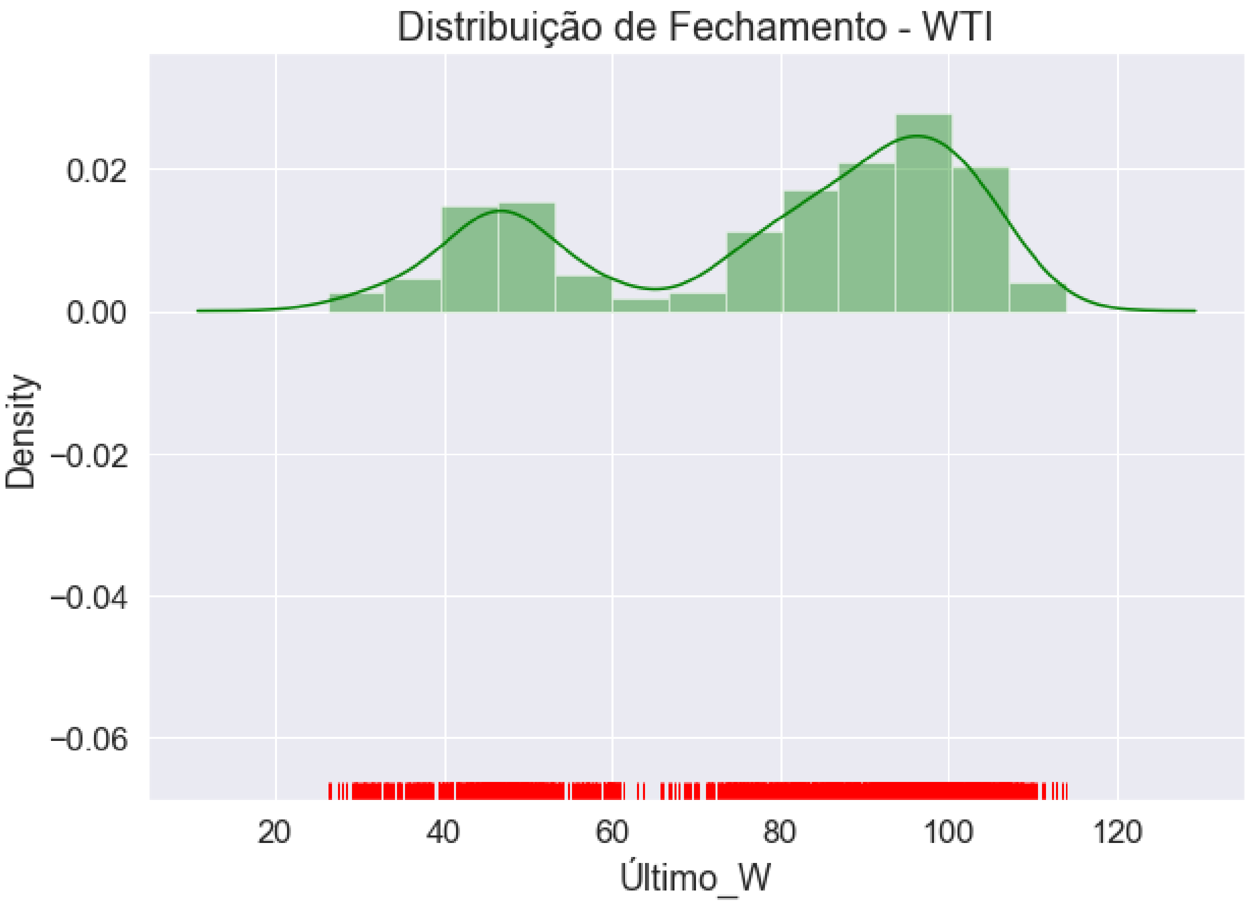
Out[43]:

```
       Mínima_W     Abertura_W   Vol._W         Máxima_W     Último_W
count  1787.000000  1787.000000  1.787000e+03  1787.000000  1787.000000
mean     77.664057    78.779239  3.200921e+07    79.767185    78.753643
std      22.891260    22.954251  1.581953e+07    22.972323    22.978843
min      26.050000    27.300000  2.210000e+06    27.480000    26.210000
25%      52.055000    53.045000  2.244200e+07    54.005000    53.000000
50%      85.550000    86.570000  2.941880e+07    87.770000    86.670000
75%      96.260000    97.310000  3.937150e+07    98.065000    97.330000
```

	Mínima_W	Abertura_W	Vol._W	Máxima_W	Último_W
max	112.250000	113.890000	1.310000e+08	114.830000	113.930000

In [44]:
```
plt.figure(figsize=(10,7))
sns.set_context("notebook", font_scale=1.5, rc={'font.size':20, 'axes.titlesize':20, 'a
sns.rugplot(train_W["Último_W"], color ='red')
sns.distplot(train_W["Último_W"], color ='green')
sns.set_style("darkgrid")
plt.title("Distribuição de Fechamento - WTI")
```

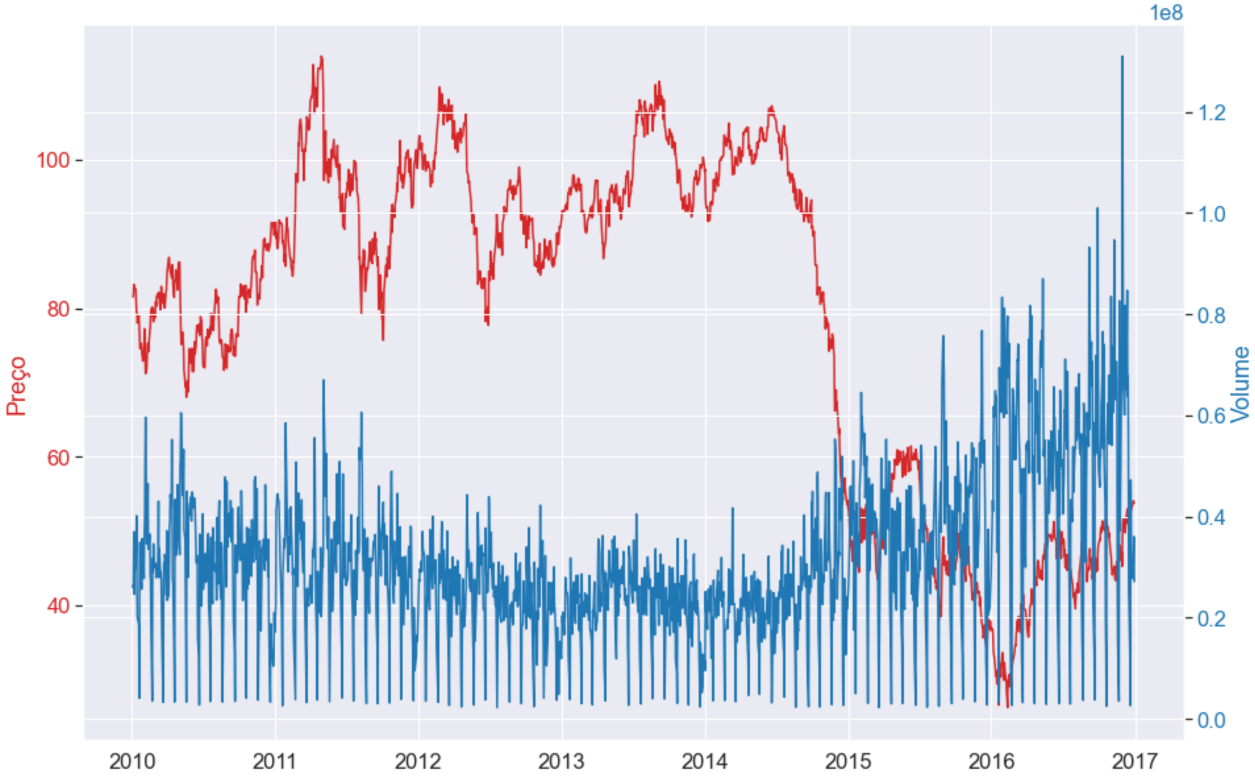
Out[44]:
```
Text(0.5, 1.0, 'Distribuição de Fechamento - WTI')
```



In [45]:
```
import plotly.graph_objects as go



fig = go.Figure(data=[go.Candlestick(x=train_W['Data'],
                open=train_W['Abertura_W'], high=train_W['Máxima_W'],
                low=train_W['Mínima_W'], close=train_W["Último_W"])
                     ])

fig.update_layout(xaxis_rangeslider_visible=False)
fig.show()
```

In [46]:

```
w = train_W["Data"]
data1 = train_W["Último_W"]
data2 = train_W["Vol._W"]

fig, ax1 = plt.subplots()

color = 'tab:red'
ax1.set_xlabel('')
ax1.set_ylabel('Preço', color=color)
ax1.plot(w, data1, color=color)
ax1.tick_params(axis='y', labelcolor=color)

ax2 = ax1.twinx()

color = 'tab:blue'
ax2.set_ylabel('Volume', color=color)
ax2.plot(w, data2, color=color)
ax2.tick_params(axis='y', labelcolor=color)
plt.gcf().set_size_inches(15, 10)
plt.show()
```



FACEBOOK PROPHET

In [47]:

```
train_start_date = '2010-01-01'
train_end_date = '2016-12-31'

test_start_date = '2017-01-01'
test_end_date = '2019-12-31'
```

In [48]:

```
brent2 = brent1
```

In [49]:

```
brent2
```

Out[49]:

	Abertura_B	Último_B	Máxima_B	Data	Vol._B	Mínima_B
data						
2019-12-31	66.65	66.00	66.93	2019-12-31	17101000.0	65.63
2019-12-30	68.20	68.44	68.99	2019-12-30	2942000.0	68.16
2019-12-27	67.91	68.16	68.33	2019-12-27	11222000.0	67.57
2019-12-26	67.27	67.92	67.99	2019-12-26	6982000.0	67.22
2019-12-24	66.44	67.20	67.26	2019-12-24	10494000.0	66.36
...	...	...	...	...	...	...
2010-01-08	81.51	81.37	82.05	2010-01-08	15749000.0	80.59
2010-01-07	82.00	81.51	82.05	2010-01-07	13128000.0	81.05
2010-01-06	80.38	81.89	82.21	2010-01-06	15787000.0	79.77
2010-01-05	80.29	80.59	80.84	2010-01-05	13175000.0	79.75

	Abertura_B	Último_B	Máxima_B	Data	Vol._B	Mínima_B
data						
2010-01-04	78.49	80.12	80.48	2010-01-04	12264000.0	78.34

2560 rows × 6 columns

In [50]:

```
filtroB2  = brent2['Data']<= train_end_date
train_B2 = brent2[filtroB2]

filtroB2teste  = brent2['Data']> train_end_date
teste_B2 = brent2[filtroB2teste]
```

In [51]:

```
train_B2.drop(columns=["Máxima_B", "Abertura_B", "Mínima_B", "Vol._B"])
```

Out[51]:

	Último_B	Data
data		
2016-12-30	56.82	2016-12-30
2016-12-29	56.14	2016-12-29
2016-12-28	56.22	2016-12-28
2016-12-27	56.09	2016-12-27
2016-12-23	55.16	2016-12-23
...	...	...
2010-01-08	81.37	2010-01-08
2010-01-07	81.51	2010-01-07
2010-01-06	81.89	2010-01-06
2010-01-05	80.59	2010-01-05
2010-01-04	80.12	2010-01-04

1787 rows × 2 columns

In [52]:

```
teste_B2.drop(columns=["Máxima_B", "Abertura_B", "Mínima_B", "Vol._B"])
```

Out[52]:

	Último_B	Data
data		
2019-12-31	66.00	2019-12-31
2019-12-30	68.44	2019-12-30
2019-12-27	68.16	2019-12-27
2019-12-26	67.92	2019-12-26
2019-12-24	67.20	2019-12-24
...	...	...

	Último_B	Data
data		
2017-01-09	54.94	2017-01-09
2017-01-06	57.10	2017-01-06
2017-01-05	56.89	2017-01-05
2017-01-04	56.46	2017-01-04
2017-01-03	55.47	2017-01-03

773 rows × 2 columns

In [53]:

```
train_B2_FP = pd.DataFrame({"ds":train_B2['Data'],"y":train_B2['Último_B']})
train_B2_FP.reset_index(drop=True, inplace=True)
train_B2_FP
```

Out[53]:

	ds	y
0	2016-12-30	56.82
1	2016-12-29	56.14
2	2016-12-28	56.22
3	2016-12-27	56.09
4	2016-12-23	55.16
...	...	...
1782	2010-01-08	81.37
1783	2010-01-07	81.51
1784	2010-01-06	81.89
1785	2010-01-05	80.59
1786	2010-01-04	80.12

1787 rows × 2 columns

In [54]:

```
teste_B2_FP = pd.DataFrame({"ds_teste":teste_B2['Data'],"y_teste":teste_B2['Último_B']}
teste_B2_FP.reset_index(drop=True, inplace=True)
teste_B2_FP
```

Out[54]:

	ds_teste	y_teste
0	2019-12-31	66.00
1	2019-12-30	68.44
2	2019-12-27	68.16
3	2019-12-26	67.92
4	2019-12-24	67.20
...	...	...

	ds_teste	y_teste
768	2017-01-09	54.94
769	2017-01-06	57.10
770	2017-01-05	56.89
771	2017-01-04	56.46
772	2017-01-03	55.47

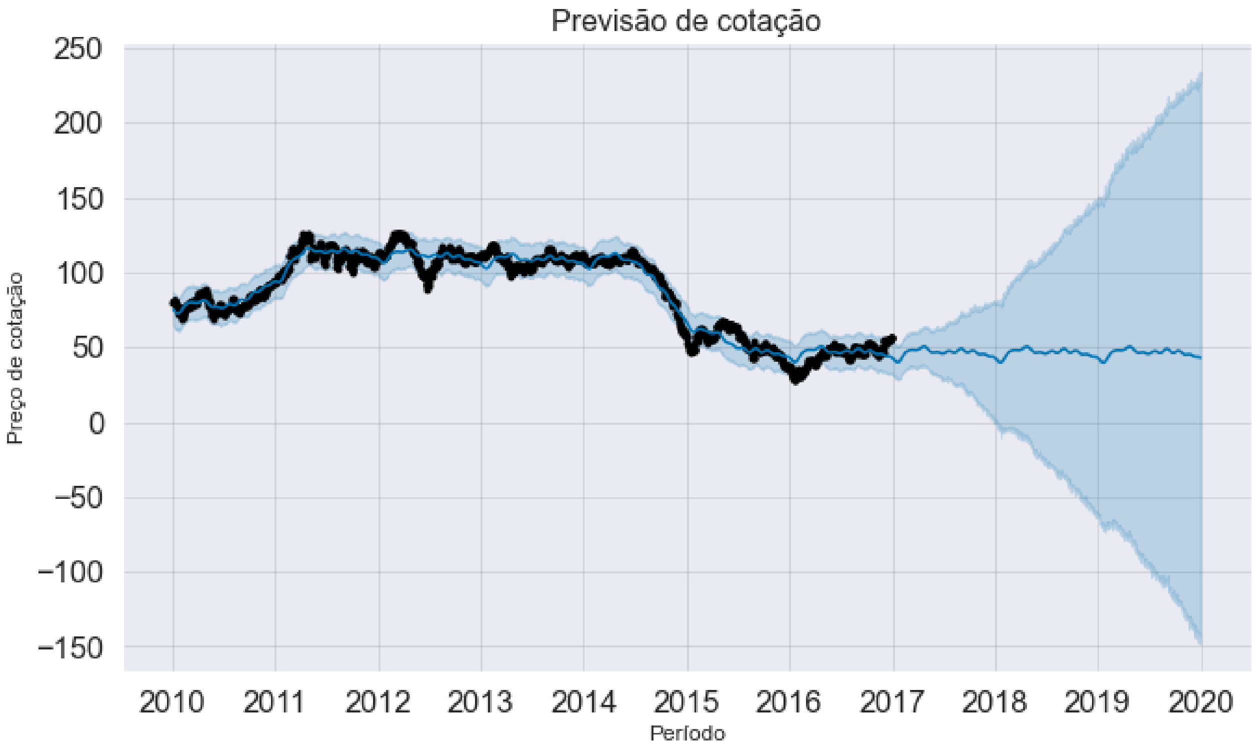
773 rows × 2 columns

```
In [55]: prophet_model = Prophet(changepoint_prior_scale=0.05, interval_width=0.95, daily_season
         prophet_model.fit(train_B2_FP)
```

```
Out[55]: <fbprophet.forecaster.Prophet at 0x26f7ad79cd0>
```

```
In [56]: prophet_forecast = prophet_model.make_future_dataframe(periods=1096, freq='D')
         prophet_forecast = prophet_model.predict(prophet_forecast)

         fig=prophet_model.plot(prophet_forecast)
         ax1=fig.gca()
         ax1.set_title('Previsão de cotação', fontsize=16)
         ax1.set_xlabel('Período', fontsize=12)
         ax1.set_ylabel('Preço de cotação', fontsize=12)
```

```
Out[56]: Text(39.5, 0.5, 'Preço de cotação')
```



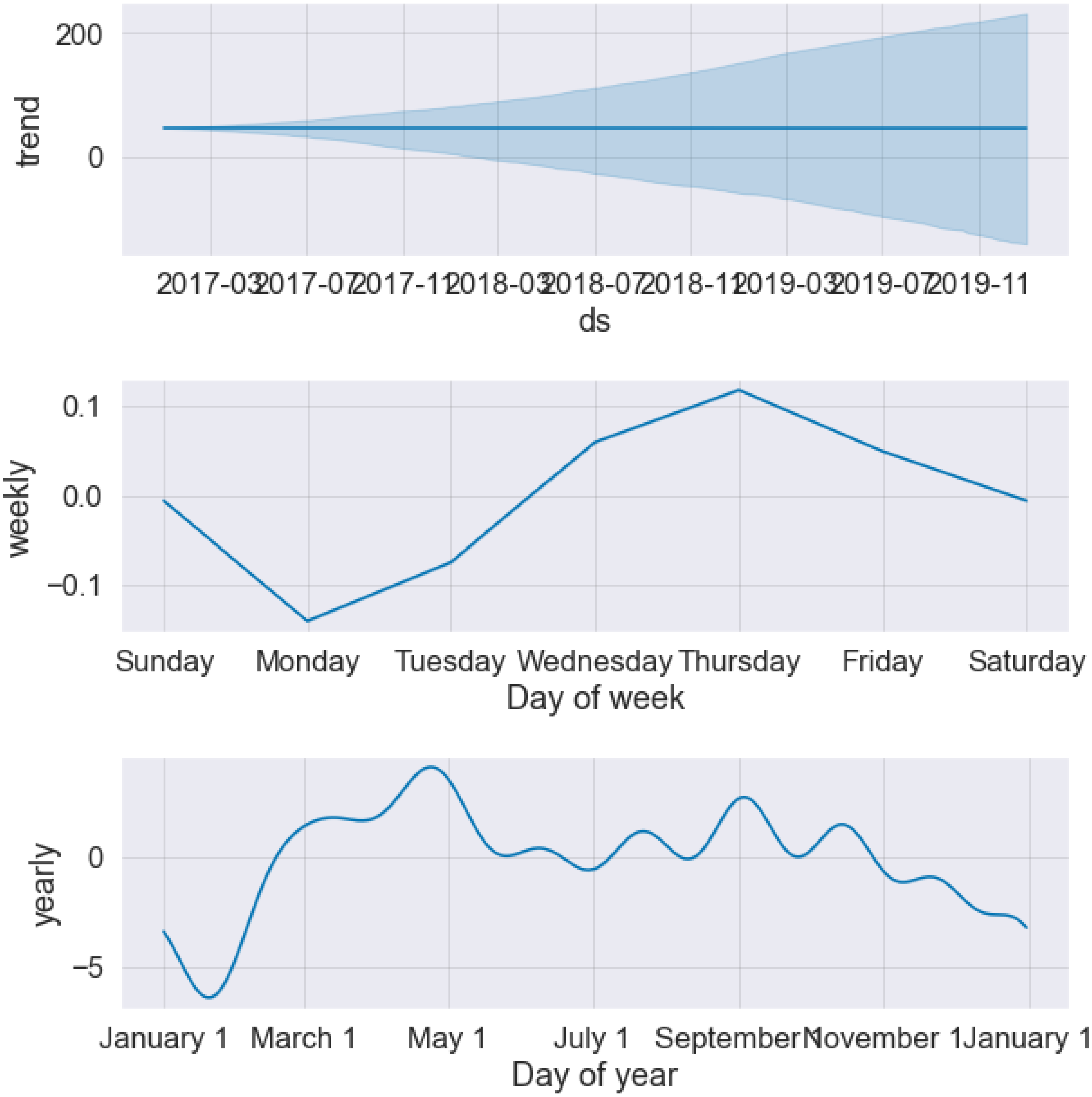
```
In [57]: prophet_forecast = prophet_forecast[prophet_forecast['ds'] > train_end_date]
         prophet_forecast.head()
```

Out[57]:

	ds	trend	yhat_lower	yhat_upper	trend_lower	trend_upper	additive_terms	additive_tern

	ds	trend	yhat_lower	yhat_upper	trend_lower	trend_upper	additive_terms	additive_terr
1788	2017-01-01	46.522636	32.271853	54.417461	46.522636	46.522636	-3.394820	-
1789	2017-01-02	46.522401	31.804644	53.528329	46.522401	46.522401	-3.689596	-
1790	2017-01-03	46.522166	31.753264	53.856961	46.522166	46.522166	-3.798483	-
1791	2017-01-04	46.521930	31.267200	53.303155	46.521930	46.523158	-3.851718	-
1792	2017-01-05	46.521695	31.252054	53.775329	46.518990	46.529545	-3.991363	-

```
In [58]: fig=prophet_model.plot_components(prophet_forecast)
```

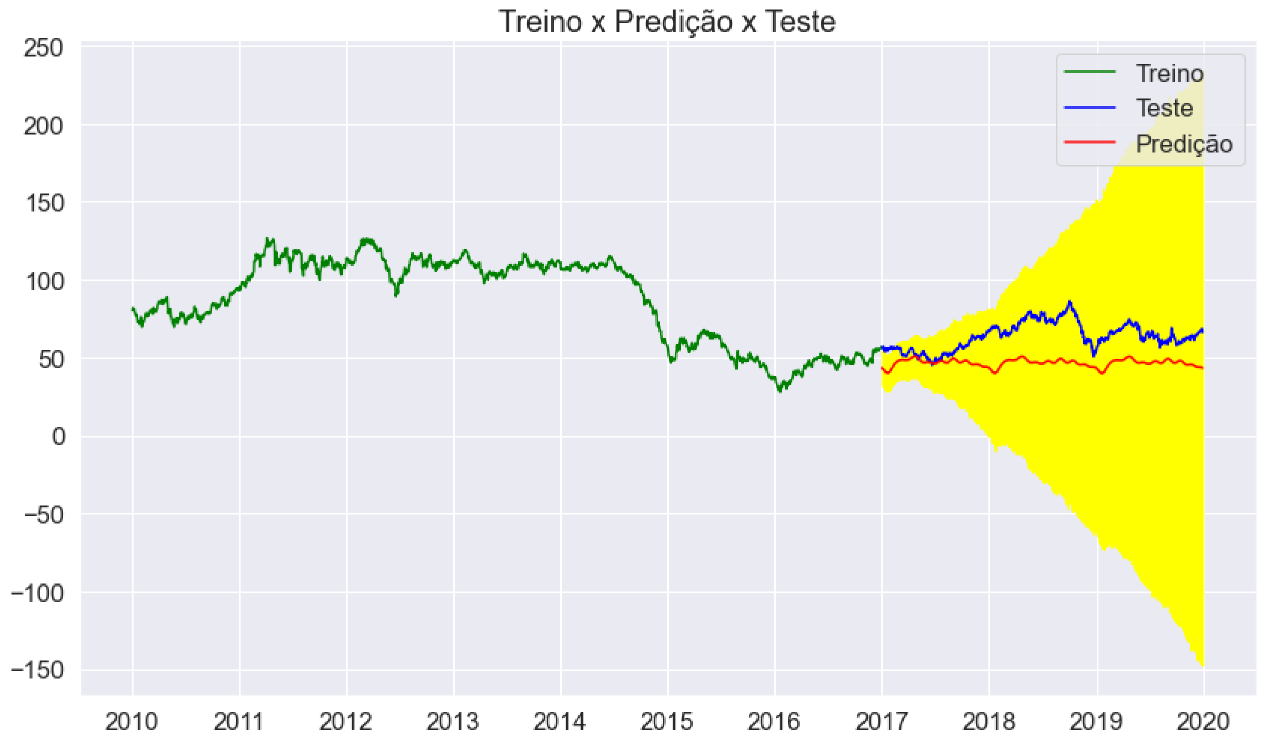


```
In [59]: prophet_forecast.set_index(prophet_forecast['ds'], inplace=True)
```

```
teste_B2_FP.set_index(teste_B2_FP['ds_teste'], inplace=True)
train_B2_FP.set_index(train_B2_FP['ds'], inplace=True)
```

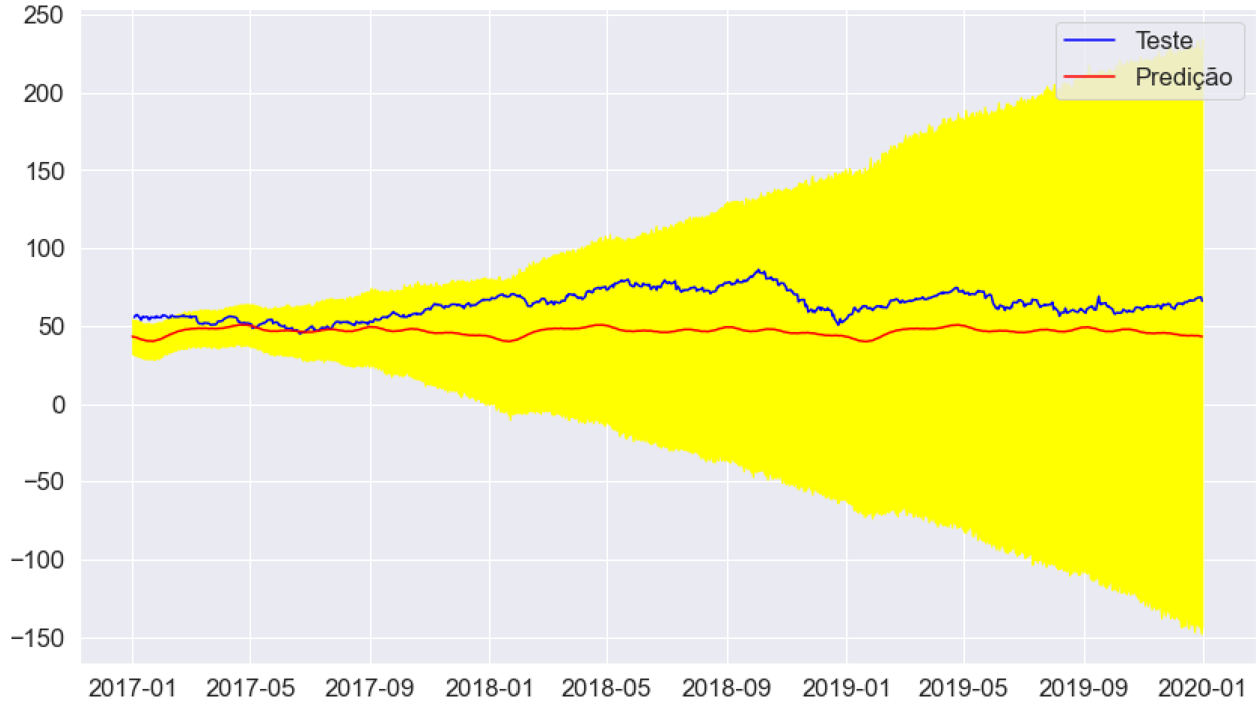
In [60]:

```
n = train_B2_FP.shape[0]
plt.figure(figsize=(14,8))
plt.title("Treino x Predição x Teste")
plt.plot(train_B2_FP['y'], 'green', label='Treino')
plt.plot(teste_B2_FP['y_teste'], 'blue', label='Teste')
plt.plot(prophet_forecast['yhat'], 'red', label='Predição')
plt.fill_between(prophet_forecast.index, prophet_forecast['yhat_lower'], prophet_foreca
plt.legend()
plt.grid(True)
```



In [61]:

```
plt.figure(figsize=(14,8))
plt.plot(teste_B2_FP['y_teste'], 'blue', label='Teste')
plt.plot(prophet_forecast['yhat'], 'red', label='Predição')
plt.fill_between(prophet_forecast.index, prophet_forecast['yhat_lower'], prophet_foreca
plt.legend()
plt.grid(True)
```



In [63]:

```
pf = prophet_forecast.copy()
tb2 = teste_B2_FP.copy()

pf.rename(columns= {'ds': 'data'}, inplace=True)
tb2.rename(columns= {'ds_teste': 'data'}, inplace=True)

pftb2 = pd.merge(pf,tb2,how='inner', on=['data'],suffixes=('_P', '_T'))
pftb2
```

Out[63]:

	data	trend	yhat_lower	yhat_upper	trend_lower	trend_upper	additive_terms	additive_term
0	2017-01-03	46.522166	31.753264	53.856961	46.522166	46.522166	-3.798483	-3
1	2017-01-04	46.521930	31.267200	53.303155	46.521930	46.523158	-3.851718	-3
2	2017-01-05	46.521695	31.252054	53.775329	46.518990	46.529545	-3.991363	-3
3	2017-01-06	46.521460	30.806468	52.818402	46.500924	46.541872	-4.266456	-4
4	2017-01-09	46.520755	30.199963	53.182456	46.422887	46.607116	-5.097600	-5
...	...	...	...	...	...	...	...	
768	2019-12-24	46.267069	-140.691328	229.004072	-139.839156	230.808042	-2.731992	-2
769	2019-12-26	46.266599	-140.823405	233.369853	-140.121356	231.242182	-2.607993	-2
770	2019-12-27	46.266364	-143.319539	227.894682	-140.260581	231.459252	-2.729845	-2
771	2019-12-30	46.265659	-148.020037	227.181423	-140.970103	232.110462	-3.169816	-3

	data	trend	yhat_lower	yhat_upper	trend_lower	trend_upper	additive_terms	additive_term
772	2019-12-31	46.265424	-142.323374	234.443056	-141.216303	232.327532	-3.220225	-3

773 rows × 20 columns

In [64]:

```
#Cálculo do erro
print('MAE: ', mean_absolute_error(pftb2['y_teste'],pftb2['yhat']))
print('MSE: ', mean_squared_error(pftb2['y_teste'],pftb2['yhat']))
print('RMSE: ', np.sqrt(mean_squared_error(pftb2['y_teste'],pftb2['yhat'])))
```

```
MAE:  17.145523353934433
MSE:  373.3109475977907
RMSE:  19.321256366959958
```

SKTIME

In [65]:

```
pip install sktime
```

```
Requirement already satisfied: sktime in c:\users\bviei\anaconda3\lib\site-packages (0.
5.2)
Requirement already satisfied: numba>=0.50 in c:\users\bviei\anaconda3\lib\site-packages
(from sktime) (0.51.2)
Requirement already satisfied: wheel in c:\users\bviei\anaconda3\lib\site-packages (from
sktime) (0.35.1)
Requirement already satisfied: pandas>=1.1.0 in c:\users\bviei\anaconda3\lib\site-packag
es (from sktime) (1.1.3)
Requirement already satisfied: statsmodels>=0.12.1 in c:\users\bviei\anaconda3\lib\site-
packages (from sktime) (0.12.1)
Requirement already satisfied: scikit-learn>=0.23.0 in c:\users\bviei\anaconda3\lib\site
-packages (from sktime) (0.23.2)
Requirement already satisfied: numpy>=1.19.0 in c:\users\bviei\anaconda3\lib\site-packag
es (from sktime) (1.19.2)
Requirement already satisfied: llvmlite<0.35,>=0.34.0.dev0 in c:\users\bviei\anaconda3\l
ib\site-packages (from numba>=0.50->sktime) (0.34.0)
Requirement already satisfied: setuptools in c:\users\bviei\anaconda3\lib\site-packages
(from numba>=0.50->sktime) (50.3.1.post20201107)
Requirement already satisfied: python-dateutil>=2.7.3 in c:\users\bviei\anaconda3\lib\si
te-packages (from pandas>=1.1.0->sktime) (2.8.1)
Requirement already satisfied: pytz>=2017.2 in c:\users\bviei\anaconda3\lib\site-package
s (from pandas>=1.1.0->sktime) (2019.3)
Requirement already satisfied: scipy>=1.1 in c:\users\bviei\anaconda3\lib\site-packages
(from statsmodels>=0.12.1->sktime) (1.5.2)
Requirement already satisfied: patsy>=0.5 in c:\users\bviei\anaconda3\lib\site-packages
(from statsmodels>=0.12.1->sktime) (0.5.1)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\users\bviei\anaconda3\lib\site
-packages (from scikit-learn>=0.23.0->sktime) (2.1.0)
Requirement already satisfied: joblib>=0.11 in c:\users\bviei\anaconda3\lib\site-package
s (from scikit-learn>=0.23.0->sktime) (0.17.0)
Requirement already satisfied: six>=1.5 in c:\users\bviei\anaconda3\lib\site-packages (f
rom python-dateutil>=2.7.3->pandas>=1.1.0->sktime) (1.15.0)
Note: you may need to restart the kernel to use updated packages.
```

In [66]:

```
from sktime.forecasting.arima import ARIMA, AutoARIMA
from sktime.forecasting.base import ForecastingHorizon
from sktime.forecasting.compose import (
    EnsembleForecaster,
    ReducedRegressionForecaster,
    TransformedTargetForecaster,
)
from sktime.forecasting.exp_smoothing import ExponentialSmoothing
from sktime.forecasting.model_selection import (
    ForecastingGridSearchCV,
    SlidingWindowSplitter,
    temporal_train_test_split,
)
from sktime.forecasting.naive import NaiveForecaster
from sktime.forecasting.theta import ThetaForecaster
from sktime.forecasting.trend import PolynomialTrendForecaster
from sktime.performance_metrics.forecasting import sMAPE, smape_loss
from sktime.transformations.series.detrend import Deseasonalizer, Detrender
from sktime.utils.plotting import plot_series

%matplotlib inline
```

In [135…

```
train_B3=train_B2
teste_B3=teste_B2

y_trainSK=train_B3
y_testSK=teste_B3
```

In [136…

```
y_testSK
```

Out[136…

	Abertura_B	Último_B	Máxima_B	Data	Vol._B	Mínima_B
data						
2019-12-31	66.65	66.00	66.93	2019-12-31	17101000.0	65.63
2019-12-30	68.20	68.44	68.99	2019-12-30	2942000.0	68.16
2019-12-27	67.91	68.16	68.33	2019-12-27	11222000.0	67.57
2019-12-26	67.27	67.92	67.99	2019-12-26	6982000.0	67.22
2019-12-24	66.44	67.20	67.26	2019-12-24	10494000.0	66.36
...	...	...	...	...	...	...
2017-01-09	56.81	54.94	57.00	2017-01-09	26639000.0	54.74
2017-01-06	56.88	57.10	57.47	2017-01-06	23487000.0	56.28
2017-01-05	56.35	56.89	57.35	2017-01-05	26961000.0	56.01
2017-01-04	55.73	56.46	56.55	2017-01-04	28255000.0	55.33
2017-01-03	57.05	55.47	58.37	2017-01-03	34082000.0	55.30

773 rows × 6 columns

In [137…

```
y_trainSK=y_trainSK.drop(columns=["Máxima_B", "Abertura_B", "Mínima_B", "Vol._B"])
y_testSK=y_testSK.drop(columns=["Máxima_B", "Abertura_B", "Mínima_B", "Vol._B"])
```

In [138…

```
y_trainSK
```

Out[138…

	Último_B	Data
data		

	Último_B	Data
data		
2016-12-30	56.82	2016-12-30
2016-12-29	56.14	2016-12-29
2016-12-28	56.22	2016-12-28
2016-12-27	56.09	2016-12-27
2016-12-23	55.16	2016-12-23
...	...	...
2010-01-08	81.37	2010-01-08
2010-01-07	81.51	2010-01-07
2010-01-06	81.89	2010-01-06
2010-01-05	80.59	2010-01-05
2010-01-04	80.12	2010-01-04

1787 rows × 2 columns

In [139…

```
y_trainI = y_trainSK.sort_values(by='Data')
y_trainI
```

Out[139…

	Último_B	Data
data		
2010-01-04	80.12	2010-01-04
2010-01-05	80.59	2010-01-05
2010-01-06	81.89	2010-01-06
2010-01-07	81.51	2010-01-07
2010-01-08	81.37	2010-01-08
...	...	...
2016-12-23	55.16	2016-12-23
2016-12-27	56.09	2016-12-27
2016-12-28	56.22	2016-12-28
2016-12-29	56.14	2016-12-29
2016-12-30	56.82	2016-12-30

1787 rows × 2 columns

In [140…

```
y_testI = y_testSK.sort_values(by='Data')
y_testI
```

Out[140…

	Último_B	Data

data	Último_B	Data
data		
2017-01-03	55.47	2017-01-03
2017-01-04	56.46	2017-01-04
2017-01-05	56.89	2017-01-05
2017-01-06	57.10	2017-01-06
2017-01-09	54.94	2017-01-09
...	...	...
2019-12-24	67.20	2019-12-24
2019-12-26	67.92	2019-12-26
2019-12-27	68.16	2019-12-27
2019-12-30	68.44	2019-12-30
2019-12-31	66.00	2019-12-31

773 rows × 2 columns

In [141…

```
brent3 = brent2
brent3=brent3.drop(columns=["Máxima_B", "Abertura_B", "Mínima_B", "Vol._B"])
brent3I = brent3.sort_values(by='Data')
brent3I=brent3I.drop(columns=["Data"])
y = brent3I['Último_B']
y
```

Out[141…

```
data
2010-01-04    80.12
2010-01-05    80.59
2010-01-06    81.89
2010-01-07    81.51
2010-01-08    81.37
              ...
2019-12-24    67.20
2019-12-26    67.92
2019-12-27    68.16
2019-12-30    68.44
2019-12-31    66.00
Name: Último_B, Length: 2560, dtype: float64
```

In [142…

```
y.index
```

Out[142…

```
DatetimeIndex(['2010-01-04', '2010-01-05', '2010-01-06', '2010-01-07',
               '2010-01-08', '2010-01-11', '2010-01-12', '2010-01-13',
               '2010-01-14', '2010-01-15',
               ...
               '2019-12-17', '2019-12-18', '2019-12-19', '2019-12-20',
               '2019-12-23', '2019-12-24', '2019-12-26', '2019-12-27',
               '2019-12-30', '2019-12-31'],
              dtype='datetime64[ns]', name='data', length=2560, freq=None)
```

In [143…

```
y
```

```
data
```

Out[143…

```
2010-01-04    80.12
2010-01-05    80.59
2010-01-06    81.89
2010-01-07    81.51
2010-01-08    81.37
              ...
2019-12-24    67.20
2019-12-26    67.92
2019-12-27    68.16
2019-12-30    68.44
2019-12-31    66.00
Name: Último_B, Length: 2560, dtype: float64
```

In [144…

```
y=y.resample('d').mean()
```

In [145…

```
y
```

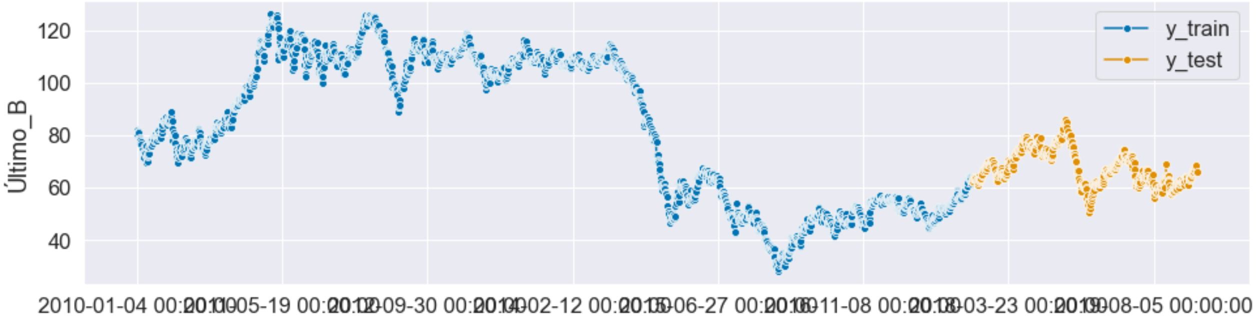
Out[145…

```
data
2010-01-04    80.12
2010-01-05    80.59
2010-01-06    81.89
2010-01-07    81.51
2010-01-08    81.37
              ...
2019-12-27    68.16
2019-12-28      NaN
2019-12-29      NaN
2019-12-30    68.44
2019-12-31    66.00
Freq: D, Name: Último_B, Length: 3649, dtype: float64
```

In [146…

```
y_trainSK, y_testSK = temporal_train_test_split(y, test_size=773)
plot_series(y_trainSK, y_testSK, labels=["y_train", "y_test"])
print(y_trainSK.shape[0], y_testSK.shape[0])
```

```
2876 773
```



In [147…

```
fh = np.arange(len(y_testSK)) + 1
fh
```

Out[147…

```
array([  1,   2,   3,   4,   5,   6,   7,   8,   9,  10,  11,  12,  13,
        14,  15,  16,  17,  18,  19,  20,  21,  22,  23,  24,  25,  26,
        27,  28,  29,  30,  31,  32,  33,  34,  35,  36,  37,  38,  39,
        40,  41,  42,  43,  44,  45,  46,  47,  48,  49,  50,  51,  52,
        53,  54,  55,  56,  57,  58,  59,  60,  61,  62,  63,  64,  65,
        66,  67,  68,  69,  70,  71,  72,  73,  74,  75,  76,  77,  78,
        79,  80,  81,  82,  83,  84,  85,  86,  87,  88,  89,  90,  91,
        92,  93,  94,  95,  96,  97,  98,  99, 100, 101, 102, 103, 104,
       105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117,
       118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129, 130,
       131, 132, 133, 134, 135, 136, 137, 138, 139, 140, 141, 142, 143,
       144, 145, 146, 147, 148, 149, 150, 151, 152, 153, 154, 155, 156,
       157, 158, 159, 160, 161, 162, 163, 164, 165, 166, 167, 168, 169,
       170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 180, 181, 182,
       183, 184, 185, 186, 187, 188, 189, 190, 191, 192, 193, 194, 195,
       196, 197, 198, 199, 200, 201, 202, 203, 204, 205, 206, 207, 208,
       209, 210, 211, 212, 213, 214, 215, 216, 217, 218, 219, 220, 221,
       222, 223, 224, 225, 226, 227, 228, 229, 230, 231, 232, 233, 234,
       235, 236, 237, 238, 239, 240, 241, 242, 243, 244, 245, 246, 247,
       248, 249, 250, 251, 252, 253, 254, 255, 256, 257, 258, 259, 260,
       261, 262, 263, 264, 265, 266, 267, 268, 269, 270, 271, 272, 273,
       274, 275, 276, 277, 278, 279, 280, 281, 282, 283, 284, 285, 286,
       287, 288, 289, 290, 291, 292, 293, 294, 295, 296, 297, 298, 299,
       300, 301, 302, 303, 304, 305, 306, 307, 308, 309, 310, 311, 312,
       313, 314, 315, 316, 317, 318, 319, 320, 321, 322, 323, 324, 325,
       326, 327, 328, 329, 330, 331, 332, 333, 334, 335, 336, 337, 338,
       339, 340, 341, 342, 343, 344, 345, 346, 347, 348, 349, 350, 351,
       352, 353, 354, 355, 356, 357, 358, 359, 360, 361, 362, 363, 364,
       365, 366, 367, 368, 369, 370, 371, 372, 373, 374, 375, 376, 377,
       378, 379, 380, 381, 382, 383, 384, 385, 386, 387, 388, 389, 390,
       391, 392, 393, 394, 395, 396, 397, 398, 399, 400, 401, 402, 403,
       404, 405, 406, 407, 408, 409, 410, 411, 412, 413, 414, 415, 416,
       417, 418, 419, 420, 421, 422, 423, 424, 425, 426, 427, 428, 429,
       430, 431, 432, 433, 434, 435, 436, 437, 438, 439, 440, 441, 442,
       443, 444, 445, 446, 447, 448, 449, 450, 451, 452, 453, 454, 455,
       456, 457, 458, 459, 460, 461, 462, 463, 464, 465, 466, 467, 468,
       469, 470, 471, 472, 473, 474, 475, 476, 477, 478, 479, 480, 481,
       482, 483, 484, 485, 486, 487, 488, 489, 490, 491, 492, 493, 494,
       495, 496, 497, 498, 499, 500, 501, 502, 503, 504, 505, 506, 507,
       508, 509, 510, 511, 512, 513, 514, 515, 516, 517, 518, 519, 520,
       521, 522, 523, 524, 525, 526, 527, 528, 529, 530, 531, 532, 533,
       534, 535, 536, 537, 538, 539, 540, 541, 542, 543, 544, 545, 546,
       547, 548, 549, 550, 551, 552, 553, 554, 555, 556, 557, 558, 559,
       560, 561, 562, 563, 564, 565, 566, 567, 568, 569, 570, 571, 572,
       573, 574, 575, 576, 577, 578, 579, 580, 581, 582, 583, 584, 585,
       586, 587, 588, 589, 590, 591, 592, 593, 594, 595, 596, 597, 598,
       599, 600, 601, 602, 603, 604, 605, 606, 607, 608, 609, 610, 611,
       612, 613, 614, 615, 616, 617, 618, 619, 620, 621, 622, 623, 624,
       625, 626, 627, 628, 629, 630, 631, 632, 633, 634, 635, 636, 637,
       638, 639, 640, 641, 642, 643, 644, 645, 646, 647, 648, 649, 650,
       651, 652, 653, 654, 655, 656, 657, 658, 659, 660, 661, 662, 663,
       664, 665, 666, 667, 668, 669, 670, 671, 672, 673, 674, 675, 676,
       677, 678, 679, 680, 681, 682, 683, 684, 685, 686, 687, 688, 689,
       690, 691, 692, 693, 694, 695, 696, 697, 698, 699, 700, 701, 702,
       703, 704, 705, 706, 707, 708, 709, 710, 711, 712, 713, 714, 715,
       716, 717, 718, 719, 720, 721, 722, 723, 724, 725, 726, 727, 728,
       729, 730, 731, 732, 733, 734, 735, 736, 737, 738, 739, 740, 741,
       742, 743, 744, 745, 746, 747, 748, 749, 750, 751, 752, 753, 754,
       755, 756, 757, 758, 759, 760, 761, 762, 763, 764, 765, 766, 767,
       768, 769, 770, 771, 772, 773])
```

In [148…

```
fh = ForecastingHorizon(y_testSK.index, is_relative=False)
fh
```

Out[148…

```
ForecastingHorizon(['2017-11-19', '2017-11-20', '2017-11-21', '2017-11-22',
                    '2017-11-23', '2017-11-24', '2017-11-25', '2017-11-26',
                    '2017-11-27', '2017-11-28',
                    ...
                    '2019-12-22', '2019-12-23', '2019-12-24', '2019-12-25',
                    '2019-12-26', '2019-12-27', '2019-12-28', '2019-12-29',
                    '2019-12-30', '2019-12-31'],
                   dtype='datetime64[ns]', name='data', length=773, freq='D', is_relative=Fal
se)
```

In [149…

```
y.isnull().sum()
```

```
Out[149… 1089
```

```
In [170… y_mediana = y.rolling(5).mean().shift(-5).round(0)
          y.fillna(y_mediana, inplace=True)
```

```
In [171… y.isnull().sum()
```

```
Out[171… 0
```

```
In [175… y_train_mediana = y.rolling(5).mean().shift(-5).round(0)
          y_trainSK.fillna(y_train_mediana, inplace=True)

          y_test_mediana = y.rolling(5).mean().shift(-5).round(0)
          y_testSK.fillna(y_test_mediana, inplace=True)
```

```
In [176… y_trainSK.isnull().sum()
```

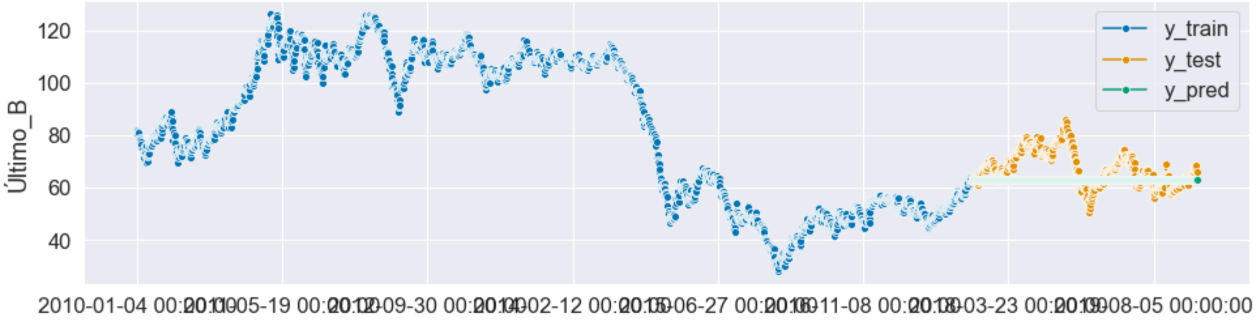
```
Out[176… 0
```

```
In [177… y_testSK.isnull().sum()
```

```
Out[177… 0
```

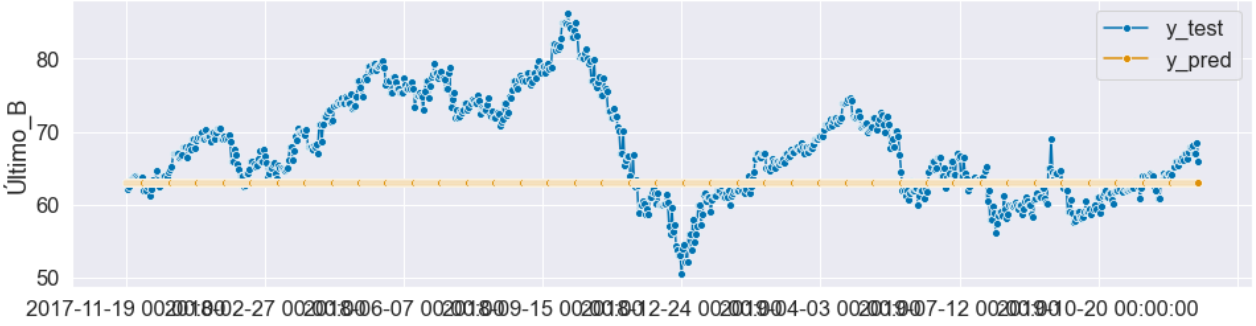
```
In [178… forecaster = NaiveForecaster(strategy="last")
          forecaster.fit(y_trainSK)
          y_pred = forecaster.predict(fh)
          plot_series(y_trainSK, y_testSK, y_pred, labels=["y_train", "y_test", "y_pred"])
          smape_loss(y_pred, y_testSK)
```

```
Out[178… 0.09236973081303312
```



```
In [179… plot_series(y_testSK, y_pred, labels=["y_test", "y_pred"])
```

```
Out[179… (<Figure size 1152x288 with 1 Axes>, <AxesSubplot:ylabel='Último_B'>)
```



In [181…

```
print('MAE: ', mean_absolute_error(y_testSK,y_pred))
print('MSE: ', mean_squared_error(y_testSK,y_pred))
print('RMSE: ', np.sqrt(mean_squared_error(y_testSK,y_pred)))
```

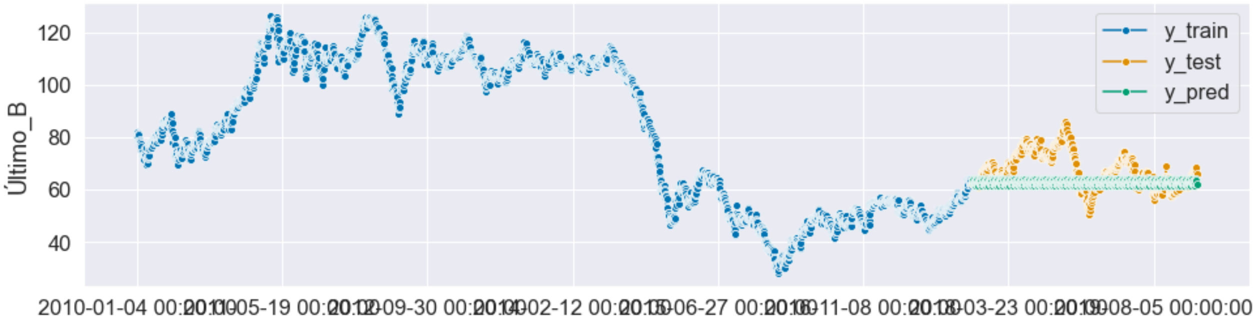
```
MAE:  6.2411254851228986
MSE:  66.676482923674
RMSE:  8.165566907672364
```

In [182…

```
forecaster = NaiveForecaster(strategy="last", sp=12)
forecaster.fit(y_trainSK)
y_pred = forecaster.predict(fh)
plot_series(y_trainSK, y_testSK, y_pred, labels=["y_train", "y_test", "y_pred"])
smape_loss(y_pred, y_testSK)
```

Out[182…

```
0.0951743349663283
```

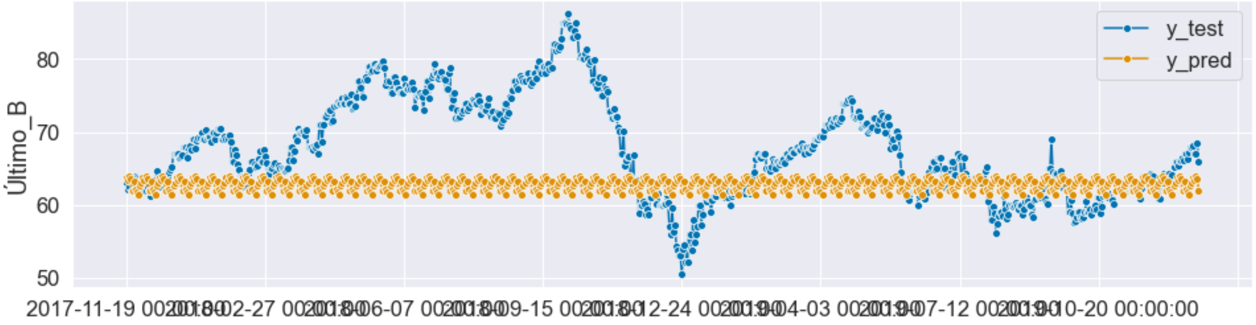


In [183…

```
plot_series(y_testSK, y_pred, labels=[ "y_test", "y_pred"])
```

Out[183…

```
(<Figure size 1152x288 with 1 Axes>, <AxesSubplot:ylabel='Último_B'>)
```

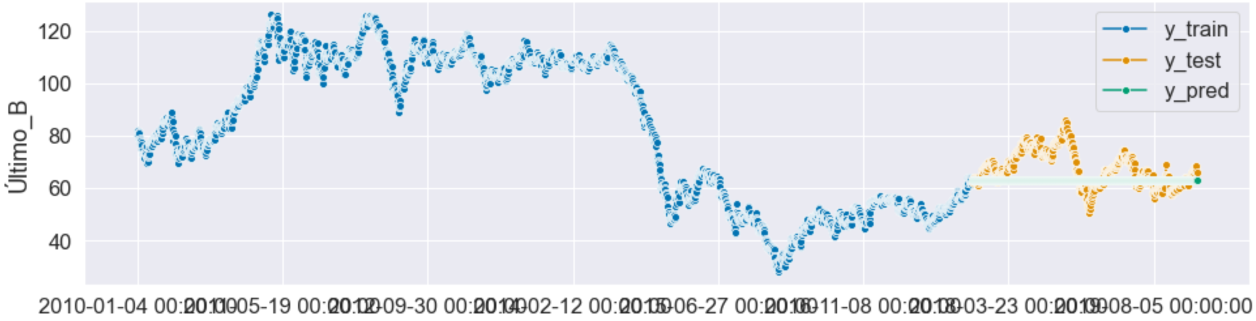


In [184…

```
from sktime.forecasting.ets import AutoETS

forecaster = AutoETS(auto=True, sp=12, n_jobs=-1)
forecaster.fit(y_trainSK)
y_pred = forecaster.predict(fh)
plot_series(y_trainSK, y_testSK, y_pred, labels=["y_train", "y_test", "y_pred"])
smape_loss(y_testSK, y_pred)
```

Out[184…

```
0.09332332551947344
```

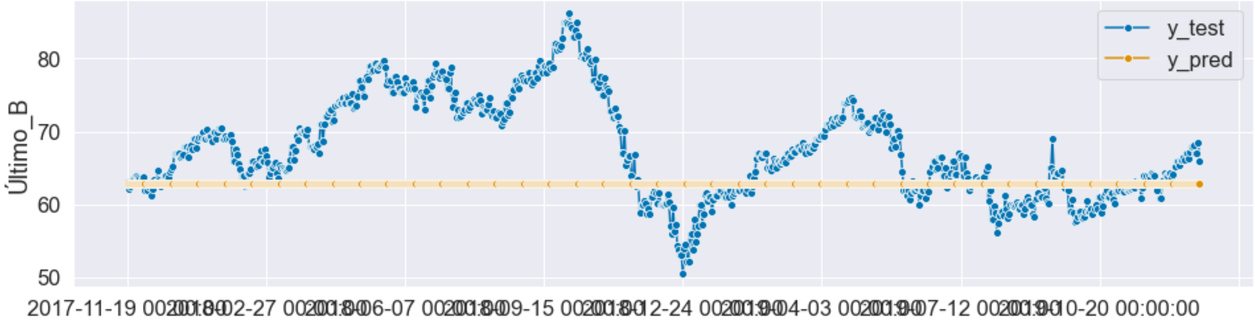


In [185…

```
plot_series(y_testSK, y_pred, labels=[ "y_test", "y_pred"])
```

Out[185…

```
(<Figure size 1152x288 with 1 Axes>, <AxesSubplot:ylabel='Último_B'>)
```



In [186…

```
print('MAE: ', mean_absolute_error(y_testSK,y_pred))
print('MSE: ', mean_squared_error(y_testSK,y_pred))
print('RMSE: ', np.sqrt(mean_squared_error(y_testSK,y_pred)))
```

```
MAE:  6.3015513938898815
MSE:  67.97582616714709
RMSE:  8.244745367029056
```

In [187…

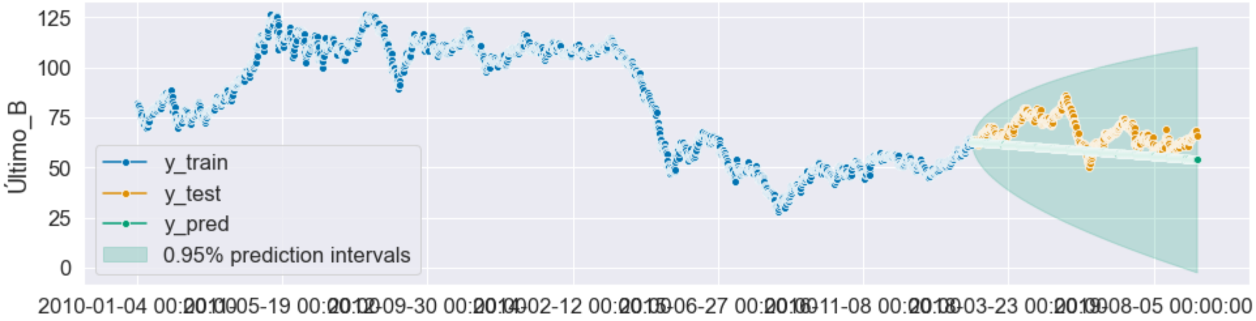
```
forecaster = ThetaForecaster(sp=12)
forecaster.fit(y_trainSK)
alpha = 0.05  # 95% prediction intervals
y_pred, pred_ints = forecaster.predict(fh, return_pred_int=True, alpha=alpha)
smape_loss(y_testSK, y_pred)
```

Out[187…

```
0.14674560330930167
```

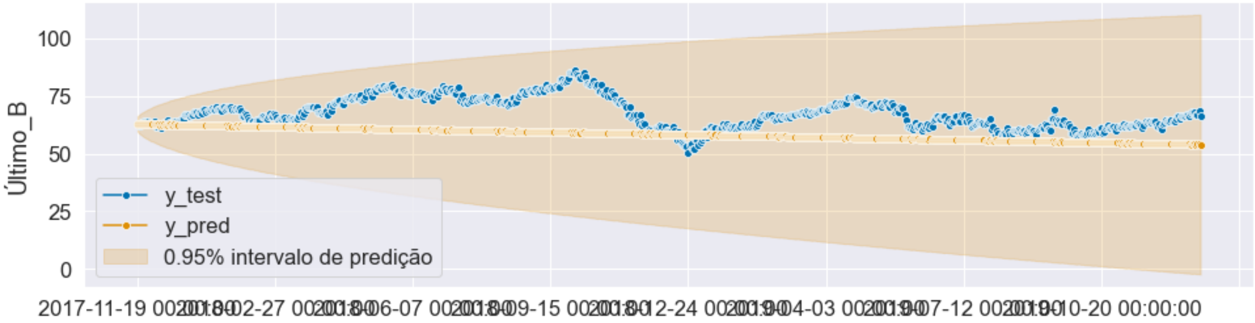
In [188…

```
fig, ax = plot_series(y_trainSK, y_testSK, y_pred, labels=["y_train", "y_test", "y_pred
ax.fill_between(
    ax.get_lines()[-1].get_xdata(),
    pred_ints["lower"],
    pred_ints["upper"],
    alpha=0.2,
    color=ax.get_lines()[-1].get_c(),
    label=f"{1 - alpha}% prediction intervals",
)
ax.legend();
```



In [189…

```
fig, ax = plot_series(y_testSK, y_pred, labels=["y_test", "y_pred"])
ax.fill_between(
    ax.get_lines()[-1].get_xdata(),
    pred_ints["lower"],
    pred_ints["upper"],
    alpha=0.2,
    color=ax.get_lines()[-1].get_c(),
    label=f"{1 - alpha}% intervalo de predição",
)
ax.legend();
```



In [190…

```
print('MAE: ', mean_absolute_error(y_testSK,y_pred))
print('MSE: ', mean_squared_error(y_testSK,y_pred))
print('RMSE: ', np.sqrt(mean_squared_error(y_testSK,y_pred)))
```

```
MAE:  9.488035578112644
MSE:  123.48841447144301
RMSE:  11.112534115648105
```

ARIMA

In [192…

```
%matplotlib inline
from matplotlib.pylab import rcParams
rcParams['figure.figsize']=15,6
```

In [193…

```
from pmdarima.arima import auto_arima
```

In [194…

```
z= y_trainI.copy()
z=z.drop(columns=["Data"])
z
```

Out[194…

	Último_B
data	
2010-01-04	80.12

	Último_B
data	
2010-01-05	80.59
2010-01-06	81.89
2010-01-07	81.51
2010-01-08	81.37
...	...
2016-12-23	55.16
2016-12-27	56.09
2016-12-28	56.22
2016-12-29	56.14
2016-12-30	56.82

1787 rows × 1 columns

In [195…

```
stepwise_model=auto_arima(z, start_p=1, start_q=1, max_p=6, max_q=6, m=12, start_P=0, s
```

```
 ARIMA(0,1,0)(0,0,0)[0] intercept   : AIC=6378.379, Time=1.36 sec
 ARIMA(0,1,1)(0,0,0)[0] intercept   : AIC=6378.085, Time=0.47 sec
 ARIMA(0,1,2)(0,0,0)[0] intercept   : AIC=6378.694, Time=0.56 sec
 ARIMA(0,1,3)(0,0,0)[0] intercept   : AIC=6380.694, Time=0.73 sec
 ARIMA(0,1,4)(0,0,0)[0] intercept   : AIC=6381.979, Time=0.75 sec
 ARIMA(0,1,5)(0,0,0)[0] intercept   : AIC=6383.962, Time=1.05 sec
 ARIMA(1,1,0)(0,0,0)[0] intercept   : AIC=6378.218, Time=0.30 sec
 ARIMA(1,1,1)(0,0,0)[0] intercept   : AIC=6379.239, Time=1.52 sec
 ARIMA(1,1,2)(0,0,0)[0] intercept   : AIC=6380.695, Time=0.56 sec
 ARIMA(1,1,3)(0,0,0)[0] intercept   : AIC=6381.425, Time=3.27 sec
 ARIMA(1,1,4)(0,0,0)[0] intercept   : AIC=6383.988, Time=1.10 sec
 ARIMA(2,1,0)(0,0,0)[0] intercept   : AIC=6378.652, Time=0.61 sec
 ARIMA(2,1,1)(0,0,0)[0] intercept   : AIC=6380.642, Time=1.85 sec
 ARIMA(2,1,2)(0,0,0)[0] intercept   : AIC=6381.539, Time=3.60 sec
 ARIMA(2,1,3)(0,0,0)[0] intercept   : AIC=inf, Time=5.92 sec
 ARIMA(3,1,0)(0,0,0)[0] intercept   : AIC=6380.629, Time=0.56 sec
 ARIMA(3,1,1)(0,0,0)[0] intercept   : AIC=6382.633, Time=0.85 sec
 ARIMA(3,1,2)(0,0,0)[0] intercept   : AIC=6381.090, Time=4.46 sec
 ARIMA(4,1,0)(0,0,0)[0] intercept   : AIC=6382.056, Time=0.67 sec
 ARIMA(4,1,1)(0,0,0)[0] intercept   : AIC=6384.054, Time=0.78 sec
 ARIMA(5,1,0)(0,0,0)[0] intercept   : AIC=6384.055, Time=0.89 sec

Best model:  ARIMA(0,1,1)(0,0,0)[0] intercept
Total fit time: 31.973 seconds
```

In [196…

```
print(stepwise_model.aic())
```

```
6378.084892238259
```

In [197…

```
trainarima=z.loc['2010-01-01':'2016-12-31']
testarima=y.loc['2017-01-01':]
```

In [198…

```
stepwise_model.fit(trainarima)
```

Out[198…

```
ARIMA(order=(0, 1, 1), scoring_args={}, suppress_warnings=True)
```

In [199…

```
future_forecastarima=stepwise_model.predict(n_periods=1095)
```

In [200…

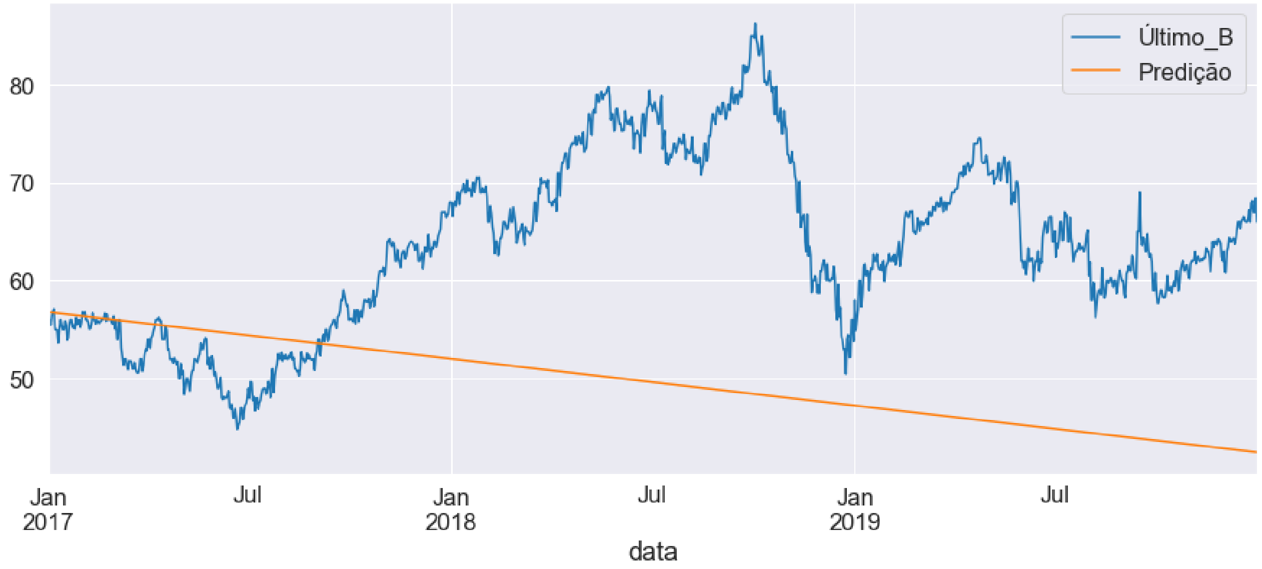
```
future_forecastarima=pd.DataFrame(future_forecastarima,index=testarima.index, columns=[
```

In [201…

```
pd.concat([testarima, future_forecastarima], axis=1).plot()
```

Out[201…

```
<AxesSubplot:xlabel='data'>
```

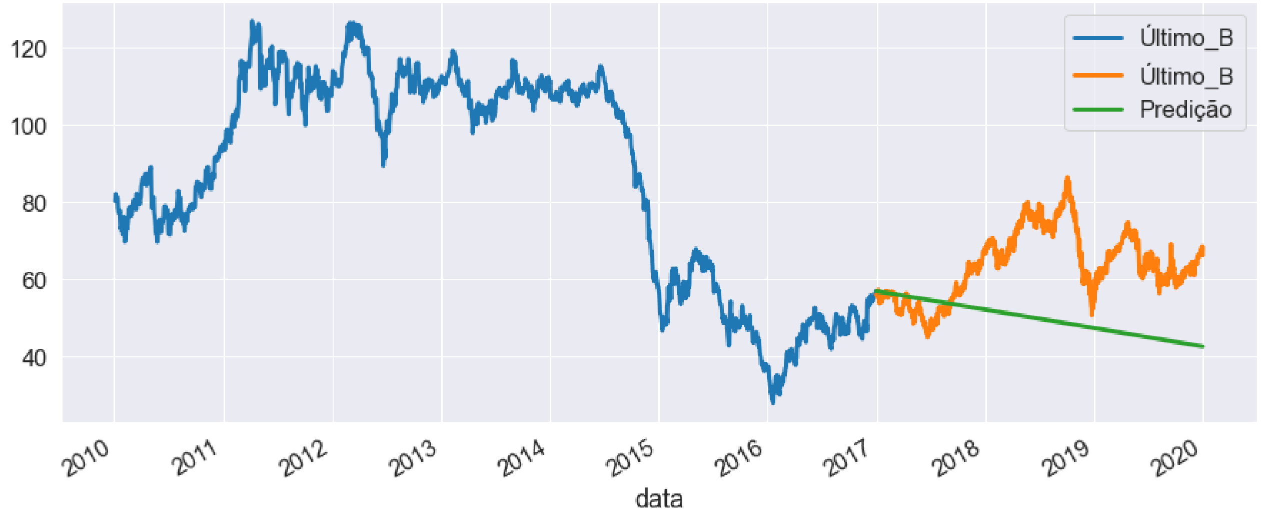


In [202…

```
pd.concat([z,testarima, future_forecastarima], axis=1).plot(linewidth=3)
```

Out[202…

```
<AxesSubplot:xlabel='data'>
```



In [203…

```
print('MAE: ', mean_absolute_error(testarima, future_forecastarima))
print('MSE: ', mean_squared_error(testarima, future_forecastarima))
print('RMSE: ', np.sqrt(mean_squared_error(testarima, future_forecastarima)))
```

```
MAE:  15.283343938135479
MSE:  319.1676593546785
RMSE:  17.86526404379959
```

In [ ]: