Hackathon

December 14, 2021

Realizado por: - Judit Lozano Gondolbeu - Francisco Diz Senra

0.0.1 Se debe predecir las matriculaciones de turismo de noviembre y diciembre de 2021.

[1]:
```
import pandas as pd
import seaborn as sns

df = pd.read_excel('Matriculaciones (1).xlsx')
df['SERIES'] = pd.to_datetime(df['SERIES'], format='%Y%m')
df
```

[1]:

	SERIES	PRODUCCIÓN DE TURISMOS	EXPORTACIONES DE TURISMOS	MATRICULACIONES DE TURISMOS.	TOTAL
0	1960-01-01	-	-		3360
1	1960-02-01	-	-		4385
2	1960-03-01	-	-		4055
3	1960-04-01	-	-		5380
4	1960-05-01	-	-		6309
..	…	…	…		…
737	2021-06-01	146311	127058		103467
738	2021-07-01	105838	89616		92171
739	2021-08-01	52399	50770		54561
740	2021-09-01	141930	135302		67792
741	2021-10-01	129672	111829		67899

```
[742 rows x 4 columns]
```

[2]:
```
import pandas as pd
import seaborn as sns

df = pd.read_excel('Matriculaciones (1).xlsx')

df

df['SERIES'] = pd.to_datetime(df['SERIES'], format='%Y%m')
df
```

[2]:
```
        SERIES PRODUCCIÓN DE TURISMOS EXPORTACIONES DE TURISMOS  \
0   1960-01-01                      -                         -
1   1960-02-01                      -                         -
2   1960-03-01                      -                         -
3   1960-04-01                      -                         -
4   1960-05-01                      -                         -
..         …                      …                         …
737 2021-06-01                 146311                    127058
738 2021-07-01                 105838                     89616
739 2021-08-01                  52399                     50770
740 2021-09-01                 141930                    135302
741 2021-10-01                 129672                    111829

     MATRICULACIONES DE TURISMOS.  TOTAL
0                                    3360
1                                    4385
2                                    4055
3                                    5380
4                                    6309
..                                    …
737                                103467
738                                 92171
739                                 54561
740                                 67792
741                                 67899

[742 rows x 4 columns]
```

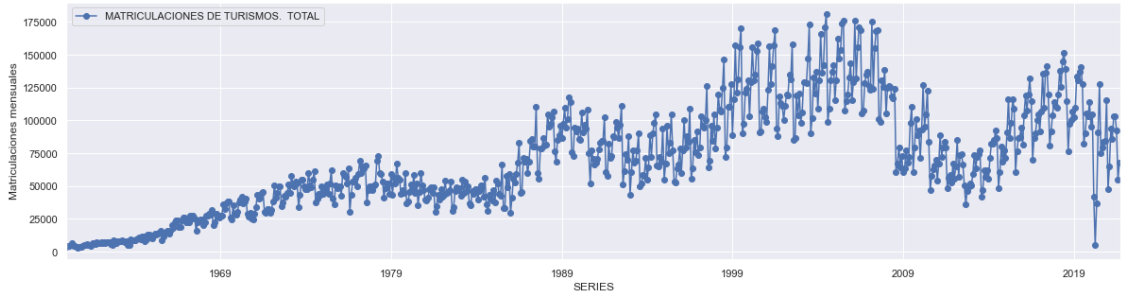
[3]:
```
df = df.set_index('SERIES')
df
```

[3]:
```
           PRODUCCIÓN DE TURISMOS EXPORTACIONES DE TURISMOS  \
SERIES
1960-01-01                      -                         -
1960-02-01                      -                         -
```

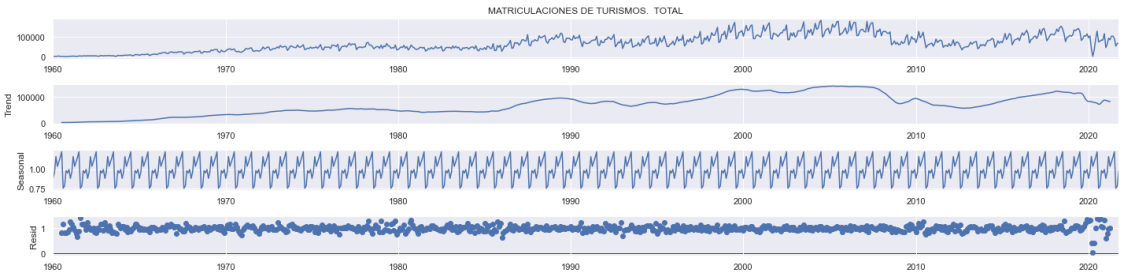
```
1960-03-01                        -                                 -
1960-04-01                        -                                 -
1960-05-01                        -                                 -
…                                 …                                 …
2021-06-01                   146311                            127058
2021-07-01                   105838                             89616
2021-08-01                    52399                             50770
2021-09-01                   141930                            135302
2021-10-01                   129672                            111829

            MATRICULACIONES DE TURISMOS.  TOTAL
SERIES
1960-01-01                                 3360
1960-02-01                                 4385
1960-03-01                                 4055
1960-04-01                                 5380
1960-05-01                                 6309
…                                           …
2021-06-01                               103467
2021-07-01                                92171
2021-08-01                                54561
2021-09-01                                67792
2021-10-01                                67899

[742 rows x 3 columns]
```

[4]:
```
sns.set(rc={'figure.figsize':(20,5)})
ax = df.plot(marker='o', linestyle='-')
ax.set_ylabel('Matriculaciones mensuales')

df.columns
```

[4]:
```
Index(['PRODUCCIÓN DE TURISMOS', 'EXPORTACIONES DE TURISMOS',
       'MATRICULACIONES DE TURISMOS.  TOTAL'],
      dtype='object')
```



[5]:
```
from statsmodels.tsa.seasonal import seasonal_decompose
decomposition = seasonal_decompose(df['MATRICULACIONES DE TURISMOS.  TOTAL'],
                                   period=12, model="multiplicative")
decomposition.plot();
```



0.0.2 Cogemos muestra mas pequeña

[6]:
```
df_reduced = df.loc['2000-01-01':'2021-10-01']
df_reduced.columns
```

[6]:
```
Index(['PRODUCCIÓN DE TURISMOS', 'EXPORTACIONES DE TURISMOS',
       'MATRICULACIONES DE TURISMOS.  TOTAL'],
      dtype='object')
```

[7]:
```
df_reduced
```

[7]:

SERIES	PRODUCCIÓN DE TURISMOS	EXPORTACIONES DE TURISMOS	\
2000-01-01	178851	144351	
2000-02-01	217794	171375	
2000-03-01	240481	189238	
2000-04-01	188108	152279	
2000-05-01	244495	192912	
…	…	…	
2021-06-01	146311	127058	
2021-07-01	105838	89616	
2021-08-01	52399	50770	
2021-09-01	141930	135302	
2021-10-01	129672	111829	

SERIES	MATRICULACIONES DE TURISMOS. TOTAL
2000-01-01	102859
2000-02-01	129416
2000-03-01	155433
2000-04-01	130722

```
2000-05-01                                134991
…                                          …
2021-06-01                                103467
2021-07-01                                 92171
2021-08-01                                 54561
2021-09-01                                 67792
2021-10-01                                 67899

[262 rows x 3 columns]
```

[8]:
```
from sktime.forecasting.base import ForecastingHorizon
from sktime.utils.plotting import plot_series
from sktime.forecasting.model_selection import temporal_train_test_split
```

[69]:
```
ts_df = pd.DataFrame(df_reduced).reset_index()
ts_df
```

[69]:
```
         SERIES PRODUCCIÓN DE TURISMOS EXPORTACIONES DE TURISMOS  \
0   2000-01-01                 178851                    144351
1   2000-02-01                 217794                    171375
2   2000-03-01                 240481                    189238
3   2000-04-01                 188108                    152279
4   2000-05-01                 244495                    192912
..         …                    …                        …
257 2021-06-01                 146311                    127058
258 2021-07-01                 105838                     89616
259 2021-08-01                  52399                     50770
260 2021-09-01                 141930                    135302
261 2021-10-01                 129672                    111829

     MATRICULACIONES DE TURISMOS.  TOTAL
0                                 102859
1                                 129416
2                                 155433
3                                 130722
4                                 134991
..                                   …
257                               103467
258                                92171
259                                54561
260                                67792
261                                67899

[262 rows x 4 columns]
```

[70]:
```
# import packages
from kats.consts import TimeSeriesData, TimeSeriesIterator
```

```
from kats.detectors.cusum_detection import CUSUMDetector
```

[71]:
```
ts_df.columns
```

[71]:
```
Index(['SERIES', 'PRODUCCIÓN DE TURISMOS', 'EXPORTACIONES DE TURISMOS',
       'MATRICULACIONES DE TURISMOS.  TOTAL'],
      dtype='object')
```

[72]:
```
df_matri = ts_df[['MATRICULACIONES DE TURISMOS.  TOTAL', 'SERIES']].
↪reset_index(drop=True)
mask = df_matri['SERIES']>'2015-01-01'
df_matri = df_matri[mask].reset_index(drop=True)
df_matri
```

[72]:

	MATRICULACIONES DE TURISMOS. TOTAL	SERIES
0	90517	2015-02-01
1	116336	2015-03-01
2	86779	2015-04-01
3	98265	2015-05-01
4	116042	2015-06-01
..	…	…
76	103467	2021-06-01
77	92171	2021-07-01
78	54561	2021-08-01
79	67792	2021-09-01
80	67899	2021-10-01

[81 rows x 2 columns]

[77]:
```
df_matri
```

[77]:

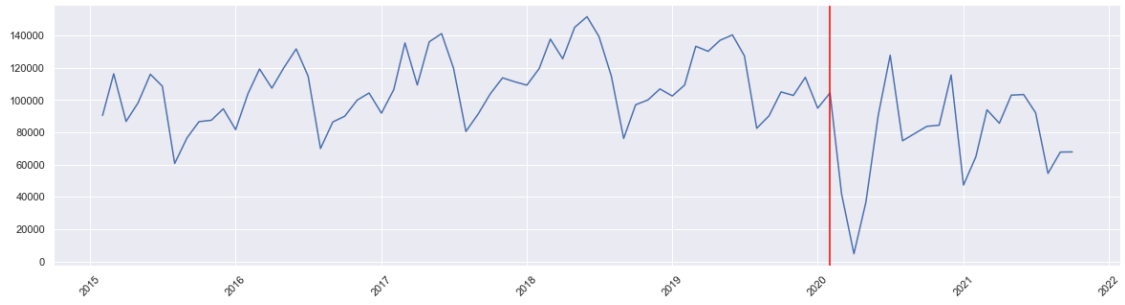
	MATRICULACIONES DE TURISMOS. TOTAL	SERIES
0	90517	2015-02-01
1	116336	2015-03-01
2	86779	2015-04-01
3	98265	2015-05-01
4	116042	2015-06-01
..	…	…
76	103467	2021-06-01
77	92171	2021-07-01
78	54561	2021-08-01
79	67792	2021-09-01
80	67899	2021-10-01

[81 rows x 2 columns]

[78]:
```
# detect increase
timeseries = TimeSeriesData(df=df_matri, time_col_name="SERIES")

detector = CUSUMDetector(timeseries)
```

[79]:
```
import matplotlib.pyplot as plt
```

[80]:
```
# run detector
change_point = detector.detector(change_directions=['decrease'])
# plot the results
plt.xticks(rotation=45)
detector.plot(change_point)
plt.show()
```



[81]:
```
change_point
```

[81]:
```
[(TimeSeriesChangePoint(start_time: 2020-02-01 00:00:00, end_time: 2020-02-01
00:00:00, confidence: 0.9999977034360332),
  <kats.detectors.cusum_detection.CUSUMMetadata at 0x7f8e8fd20b20>)]
```

[84]:
```
df_matri.head()
```

[84]:

	MATRICULACIONES DE TURISMOS. TOTAL	SERIES
0	90517	2015-02-01
1	116336	2015-03-01
2	86779	2015-04-01
3	98265	2015-05-01
4	116042	2015-06-01

[85]:
```
ts_df = df_matri.set_index('SERIES')
```

[86]:
```
ts_df.columns
```

[86]:
```
Index(['MATRICULACIONES DE TURISMOS.  TOTAL'], dtype='object')
```

[87]:
```
#Time period for pre-intervention and post-intervention
pre_period = ['2015-02-01','2020-02-01']
post_period = ['2020-03-01','2021-10-01']
```

[88]:
```
from causalimpact import CausalImpact
ci = CausalImpact(ts_df['MATRICULACIONES DE TURISMOS.  TOTAL'], pre_period,␣
↪post_period, nseasons=[{'period': 12}])
```

```
/Users/JuditLozano/opt/anaconda3/lib/python3.8/site-
packages/statsmodels/tsa/base/tsa_model.py:159: ValueWarning:

No frequency information was provided, so inferred frequency MS will be used.
```

[89]:
```
ci.plot()
```

Note: The first 13 observations were removed due to approximate diffuse initialization.

[90]:
```
print(ci.summary())
```

```
Posterior Inference {Causal Impact}
```

```
                          Average            Cumulative
Actual                    76024.85           1520497.0
Prediction (s.d.)         110572.1 (3234.14) 2211441.92 (64682.73)
95% CI                    [104404.79, 117082.37][2088095.81, 2341647.44]

Absolute effect (s.d.)    -34547.25 (3234.14)-690944.92 (64682.73)
95% CI                    [-41057.52, -28379.94][-821150.44, -567598.81]

Relative effect (s.d.)    -31.24% (2.92%)    -31.24% (2.92%)
95% CI                    [-37.13%, -25.67%] [-37.13%, -25.67%]

Posterior tail-area probability p: 0.0
Posterior prob. of a causal effect: 100.0%

For more details run the command: print(impact.summary('report'))
```

[91]:
```
print(ci.summary('report'))
```

```
Analysis report {CausalImpact}


During the post-intervention period, the response variable had
an average value of approx. 76024.85. By contrast, in the absence of an
intervention, we would have expected an average response of 110572.1.
The 95% interval of this counterfactual prediction is [104404.79, 117082.37].
Subtracting this prediction from the observed response yields
an estimate of the causal effect the intervention had on the
response variable. This effect is -34547.25 with a 95% interval of
[-41057.52, -28379.94]. For a discussion of the significance of this effect,
see below.


Summing up the individual data points during the post-intervention
period (which can only sometimes be meaningfully interpreted), the
response variable had an overall value of 1520497.0.
By contrast, had the intervention not taken place, we would have expected
a sum of 2211441.92. The 95% interval of this prediction is [2088095.81,
2341647.44].


The above results are given in terms of absolute numbers. In relative
terms, the response variable showed a decrease of -31.24%. The 95%
interval of this percentage is [-37.13%, -25.67%].


This means that the negative effect observed during the intervention
period is statistically significant.
If the experimenter had expected a positive effect, it is recommended
```

to double-check whether anomalies in the control variables may have
caused an overly optimistic expectation of what should have happened
in the response variable in the absence of the intervention.

The probability of obtaining this effect by chance is very small
(Bayesian one-sided tail-area probability p = 0.0).
This means the causal effect can be considered statistically
significant.

[92]:
```
algo = ci.inferences
algo
```

[92]:

SERIES	post_cum_y	preds	post_preds	post_preds_lower	post_preds_upper	preds_lower	preds_upper	post_cum_pred	post_cum_pred_lower	post_cum_pred_upper	point_effects
2015-02-01	NaN	108089.147541	NaN	NaN	NaN	-1.044907e+08	1.047069e+08	NaN	NaN	NaN	-17572.147541
2015-03-01	NaN	108089.147541	NaN	NaN	NaN	-1.044907e+08	1.047069e+08	NaN	NaN	NaN	8246.852459
2015-04-01	NaN	105578.840873	NaN	NaN	NaN	-1.034204e+08	1.036316e+08	NaN	NaN	NaN	-18799.840873
2015-05-01	NaN	109267.269262	NaN	NaN	NaN	-1.034167e+08	1.036353e+08	NaN	NaN	NaN	-11002.269262
2015-06-01	NaN	103228.860863	NaN	NaN	NaN	-1.026108e+08	1.028172e+08	NaN	NaN	NaN	-15228.860863
…	…	…	…	…	…	…	…	…	…	…	…
2021-06-01	1238074.0	137586.069671	137586.069671	112045.109950	163127.029392	1.120451e+05	1.631270e+05	1.822028e+06	1.707660e+06	1.936396e+06	1100487.930329
2021-07-01	1330245.0	123320.849203	123320.849203	97779.374512	148862.323893	9.777937e+04	1.488623e+05	1.945348e+06	1.823031e+06	2.067665e+06	1206924.150797
2021-08-01	1384806.0	83005.229134	83005.229134	57463.255831	108547.202438	5.746326e+04	1.085472e+05	2.028354e+06	1.911890e+06	2.144818e+06	1301800.770866
2021-09-01	1452598.0	85405.608304	85405.608304	59863.152746	110948.063862	5.986315e+04	1.109481e+05	2.113759e+06	1.997127e+06	2.230391e+06	1367192.391696
2021-10-01	1520497.0	97682.587658	97682.587658	72139.666201	123225.509115	7.213967e+04	1.232255e+05	2.211442e+06	2.094844e+06	2.328040e+06	1422814.412342

```
2015-06-01                  NaN                  NaN   12813.139137
…                           …                    …          …
2021-06-01         1.716219e+06         1.934003e+06  -34119.069671
2021-07-01         1.831106e+06         2.061462e+06  -31149.849203
2021-08-01         1.911615e+06         2.150408e+06  -28444.229134
2021-09-01         1.993302e+06         2.243546e+06  -17613.608304
2021-10-01         2.088096e+06         2.341647e+06  -29783.587658

            point_effects_lower  point_effects_upper  post_cum_effects  \
SERIES
2015-02-01        -1.046164e+08         1.045813e+08               NaN
2015-03-01        -1.045906e+08         1.046071e+08               NaN
2015-04-01        -1.035448e+08         1.035072e+08               NaN
2015-05-01        -1.035370e+08         1.035150e+08               NaN
2015-06-01        -1.027012e+08         1.027268e+08               NaN
…                           …                    …             …
2021-06-01        -5.966003e+04        -8.578110e+03    -583953.646240
2021-07-01        -5.669132e+04        -5.608375e+03    -615103.495442
2021-08-01        -5.398620e+04        -2.902256e+03    -643547.724576
2021-09-01        -4.315606e+04         7.928847e+03    -661161.332880
2021-10-01        -5.532651e+04        -4.240666e+03    -690944.920538

            post_cum_effects_lower  post_cum_effects_upper
SERIES
2015-02-01                     NaN                     NaN
2015-03-01                     NaN                     NaN
2015-04-01                     NaN                     NaN
2015-05-01                     NaN                     NaN
2015-06-01                     NaN                     NaN
…                              …                       …
2021-06-01          -695929.169918          -478145.210491
2021-07-01          -731216.748498          -500860.604901
2021-08-01          -765601.873491          -526809.125634
2021-09-01          -790948.465388          -540704.037868
2021-10-01          -821150.444908          -567598.813637

[81 rows x 16 columns]
```

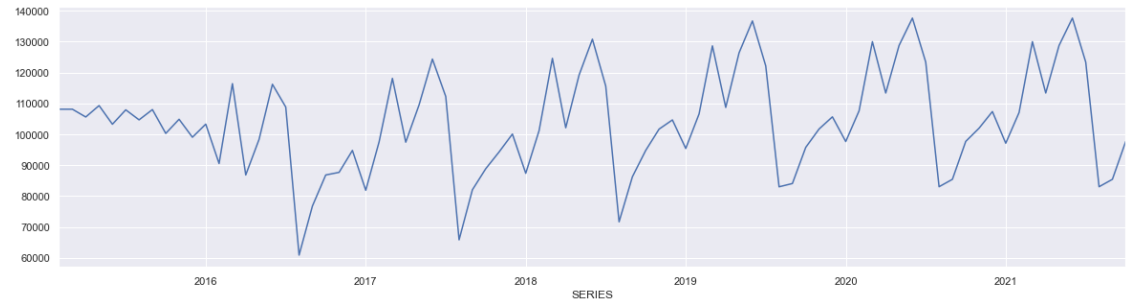
[93]:
```
df_preds = algo.preds
df_preds.tail()
```

[93]:
```
SERIES
2021-06-01    137586.069671
2021-07-01    123320.849203
2021-08-01     83005.229134
2021-09-01     85405.608304
2021-10-01     97682.587658
```

```
Freq: MS, Name: preds, dtype: float64
```

[94]:
```
df_preds.plot()
```

[94]:
```
<AxesSubplot:xlabel='SERIES'>
```



[56]:
```
differencia_gg = df_preds[-1]
differencia_gg
```

[56]:
```
97682.58765784859
```

[99]:
```
differencia_re = ts_df
differencia_re
```

[99]:

SERIES	MATRICULACIONES DE TURISMOS. TOTAL
2015-02-01	90517
2015-03-01	116336
2015-04-01	86779
2015-05-01	98265
2015-06-01	116042
…	…
2021-06-01	103467
2021-07-01	92171
2021-08-01	54561
2021-09-01	67792
2021-10-01	67899

[81 rows x 1 columns]

[100]:
```
differencia_re = 67899
```

[102]:
```
gap = differencia_gg - differencia_re
gap
```

[102]:
```
29783.58765784859
```

[111]:
```
df_preds = df_preds.drop('index',axis= 1)
```

[112]:
```
df_preds
```

[112]:

	SERIES	preds
0	2015-02-01	108089.147541
1	2015-03-01	108089.147541
2	2015-04-01	105578.840873
3	2015-05-01	109267.269262
4	2015-06-01	103228.860863
..	…	…
76	2021-06-01	137586.069671
77	2021-07-01	123320.849203
78	2021-08-01	83005.229134
79	2021-09-01	85405.608304
80	2021-10-01	97682.587658

[81 rows x 2 columns]

[103]:
```
from kats.models.prophet import ProphetModel, ProphetParams
```

[121]:
```
df_corrected = pd.DataFrame(df_preds).reset_index(drop=True)
df_corrected.tail()
```

[121]:

	SERIES	preds
76	2021-06-01	137586.069671
77	2021-07-01	123320.849203
78	2021-08-01	83005.229134
79	2021-09-01	85405.608304
80	2021-10-01	97682.587658

[114]:
```
df_corrected.columns
```

[114]:
```
Index(['SERIES', 'preds'], dtype='object')
```

[115]:
```
from kats.consts import TimeSeriesData

df_corrected.columns = ["time", "value"]
corrected_ts = TimeSeriesData(df_corrected)
type(corrected_ts)
```

[115]:
```
kats.consts.TimeSeriesData
```

[116]:
```
# create a model param instance
params = ProphetParams(seasonality_mode='multiplicative') # additive mode
# create a prophet model instance
m = ProphetModel(corrected_ts, params)
# fit model simply by calling m.fit()
```

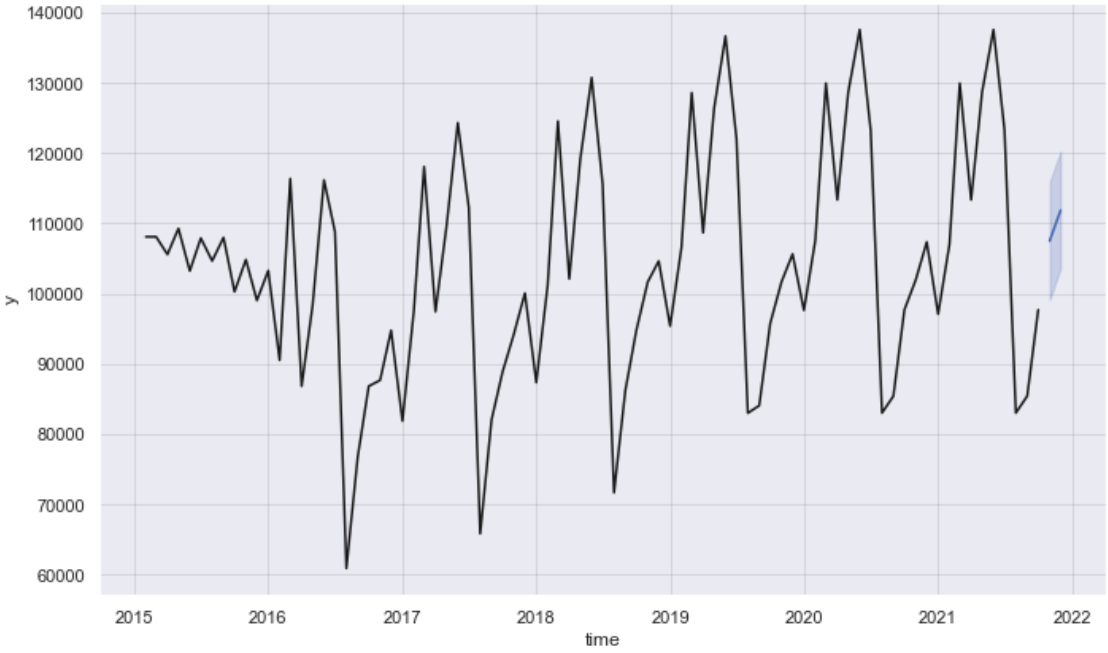
```
m.fit()
# make prediction for next 2 months
fcst = m.predict(steps=2, freq="MS")
# plot to visualize
m.plot()
```

```
INFO:fbprophet:Disabling weekly seasonality. Run prophet with
weekly_seasonality=True to override this.
INFO:fbprophet:Disabling daily seasonality. Run prophet with
daily_seasonality=True to override this.


Initial log joint probability = -89.2435
Iteration  1. Log joint probability =    65.0342. Improved by 154.278.
Iteration  2. Log joint probability =    117.804. Improved by 52.7694.
Iteration  3. Log joint probability =    162.377. Improved by 44.5737.
Iteration  4. Log joint probability =    172.519. Improved by 10.1421.
Iteration  5. Log joint probability =    178.637. Improved by 6.11762.
Iteration  6. Log joint probability =    178.699. Improved by 0.0623579.
Iteration  7. Log joint probability =    178.885. Improved by 0.185631.
Iteration  8. Log joint probability =    178.887. Improved by 0.00235605.
Iteration  9. Log joint probability =    179.006. Improved by 0.118966.
Iteration 10. Log joint probability =    179.066. Improved by 0.0598344.
Iteration 11. Log joint probability =    179.145. Improved by 0.0791828.
Iteration 12. Log joint probability =    179.184. Improved by 0.0388274.
Iteration 13. Log joint probability =     179.27. Improved by 0.0859349.
Iteration 14. Log joint probability =    179.604. Improved by 0.333651.
Iteration 15. Log joint probability =    179.666. Improved by 0.0622773.
Iteration 16. Log joint probability =    179.726. Improved by 0.0601544.
Iteration 17. Log joint probability =    179.897. Improved by 0.170429.
Iteration 18. Log joint probability =    180.019. Improved by 0.122272.
Iteration 19. Log joint probability =     180.06. Improved by 0.0414112.
Iteration 20. Log joint probability =    180.182. Improved by 0.121586.
Iteration 21. Log joint probability =    180.399. Improved by 0.217523.
Iteration 22. Log joint probability =    180.666. Improved by 0.266785.
Iteration 23. Log joint probability =    180.769. Improved by 0.102376.
Iteration 24. Log joint probability =    180.871. Improved by 0.102721.
Iteration 25. Log joint probability =    181.047. Improved by 0.175874.
Iteration 26. Log joint probability =    181.083. Improved by 0.0362584.
Iteration 27. Log joint probability =    181.234. Improved by 0.150803.
Iteration 28. Log joint probability =    181.268. Improved by 0.0339625.
Iteration 29. Log joint probability =    181.299. Improved by 0.0307546.
Iteration 30. Log joint probability =    181.336. Improved by 0.0373143.
Iteration 31. Log joint probability =     181.36. Improved by 0.0235476.
Iteration 32. Log joint probability =    181.405. Improved by 0.0450447.
Iteration 33. Log joint probability =    181.417. Improved by 0.0124431.
Iteration 34. Log joint probability =    181.425. Improved by 0.00763727.
Iteration 35. Log joint probability =    181.536. Improved by 0.110716.
```

```
Iteration 36. Log joint probability =    181.562. Improved by 0.0268483.
Iteration 37. Log joint probability =    181.642. Improved by 0.0792369.
Iteration 38. Log joint probability =    181.701. Improved by 0.0596426.
Iteration 39. Log joint probability =    181.738. Improved by 0.0366243.
Iteration 40. Log joint probability =    181.841. Improved by 0.10324.
Iteration 41. Log joint probability =    181.908. Improved by 0.0665447.
Iteration 42. Log joint probability =    182.603. Improved by 0.695292.
Iteration 43. Log joint probability =    182.777. Improved by 0.174055.
Iteration 44. Log joint probability =    183.488. Improved by 0.711297.
Iteration 45. Log joint probability =    184.922. Improved by 1.43366.
Iteration 46. Log joint probability =    187.145. Improved by 2.22295.
Iteration 47. Log joint probability =    187.728. Improved by 0.582941.
Iteration 48. Log joint probability =    187.841. Improved by 0.112916.
Iteration 49. Log joint probability =    201.368. Improved by 13.5267.
Iteration 50. Log joint probability =    201.515. Improved by 0.147662.
Iteration 51. Log joint probability =    201.549. Improved by 0.0333694.
Iteration 52. Log joint probability =    201.557. Improved by 0.00883493.
Iteration 53. Log joint probability =    201.614. Improved by 0.0567168.
Iteration 54. Log joint probability =    201.832. Improved by 0.217389.
Iteration 55. Log joint probability =    201.868. Improved by 0.0368579.
Iteration 56. Log joint probability =    201.912. Improved by 0.043493.
Iteration 57. Log joint probability =    201.934. Improved by 0.0225369.
Iteration 58. Log joint probability =    201.984. Improved by 0.0499333.
Iteration 59. Log joint probability =    201.998. Improved by 0.0135078.
Iteration 60. Log joint probability =    202.073. Improved by 0.0752854.
Iteration 61. Log joint probability =    202.097. Improved by 0.02391.
Iteration 62. Log joint probability =      202.2. Improved by 0.103201.
Iteration 63. Log joint probability =    202.339. Improved by 0.138807.
Iteration 64. Log joint probability =    202.351. Improved by 0.0121637.
Iteration 65. Log joint probability =    202.495. Improved by 0.143931.
Iteration 66. Log joint probability =    202.656. Improved by 0.160826.
Iteration 67. Log joint probability =    202.674. Improved by 0.0182338.
Iteration 68. Log joint probability =    203.185. Improved by 0.510811.
Iteration 69. Log joint probability =    203.273. Improved by 0.0879023.
Iteration 70. Log joint probability =    203.295. Improved by 0.0218597.
Iteration 71. Log joint probability =    203.306. Improved by 0.011569.
Iteration 72. Log joint probability =    203.314. Improved by 0.00811429.
Iteration 73. Log joint probability =    203.324. Improved by 0.00962869.
Iteration 74. Log joint probability =    203.328. Improved by 0.0041257.
Iteration 75. Log joint probability =    203.331. Improved by 0.00271797.
Iteration 76. Log joint probability =    203.335. Improved by 0.00384529.
Iteration 77. Log joint probability =    203.337. Improved by 0.00251158.
Iteration 78. Log joint probability =    203.341. Improved by 0.00328796.
Iteration 79. Log joint probability =    203.341. Improved by 0.00068706.
Iteration 80. Log joint probability =    203.342. Improved by 0.000688589.
Iteration 81. Log joint probability =    203.344. Improved by 0.00198283.
Iteration 82. Log joint probability =    203.346. Improved by 0.00189546.
Iteration 83. Log joint probability =    203.347. Improved by 0.000725695.
```

```
Iteration 84. Log joint probability =    203.347. Improved by 0.000620393.
Iteration 85. Log joint probability =    203.348. Improved by 0.00130275.
Iteration 86. Log joint probability =    203.349. Improved by 5.97529e-05.
Iteration 87. Log joint probability =    203.349. Improved by 0.00060245.
Iteration 88. Log joint probability =     203.35. Improved by 0.000406434.
Iteration 89. Log joint probability =     203.35. Improved by 0.000643031.
Iteration 90. Log joint probability =     203.35. Improved by 0.000302102.
Iteration 91. Log joint probability =    203.351. Improved by 0.000224397.
Iteration 92. Log joint probability =    203.351. Improved by 4.78013e-05.
Iteration 93. Log joint probability =    203.351. Improved by 4.82978e-05.
Iteration 94. Log joint probability =    203.351. Improved by 0.000106462.
Iteration 95. Log joint probability =    203.351. Improved by 0.000173399.
Iteration 96. Log joint probability =    203.351. Improved by 3.72636e-05.
Iteration 97. Log joint probability =    203.351. Improved by 5.5623e-05.
Iteration 98. Log joint probability =    203.351. Improved by 7.75916e-05.
Iteration 99. Log joint probability =    203.351. Improved by 9.62983e-06.
Iteration 100. Log joint probability =    203.351. Improved by 3.84307e-05.
Iteration 101. Log joint probability =    203.351. Improved by 2.42201e-05.
Iteration 102. Log joint probability =    203.351. Improved by 2.1239e-06.
Iteration 103. Log joint probability =    203.351. Improved by 7.1994e-06.
Iteration 104. Log joint probability =    203.351. Improved by 1.18995e-05.
Iteration 105. Log joint probability =    203.351. Improved by 2.03425e-05.
Iteration 106. Log joint probability =    203.351. Improved by 3.72415e-05.
Iteration 107. Log joint probability =    203.351. Improved by 1.47467e-06.
Iteration 108. Log joint probability =    203.351. Improved by 3.26629e-06.
Iteration 109. Log joint probability =    203.351. Improved by 1.24234e-05.
Iteration 110. Log joint probability =    203.351. Improved by 4.38178e-06.
Iteration 111. Log joint probability =    203.351. Improved by 2.77752e-05.
Iteration 112. Log joint probability =    203.351. Improved by 1.14223e-06.
Iteration 113. Log joint probability =    203.351. Improved by 1.50214e-06.
Iteration 114. Log joint probability =    203.351. Improved by 8.29288e-06.
Iteration 115. Log joint probability =    203.351. Improved by 1.53891e-06.
Iteration 116. Log joint probability =    203.351. Improved by 1.09276e-05.
Iteration 117. Log joint probability =    203.351. Improved by 7.02253e-06.
Iteration 118. Log joint probability =    203.352. Improved by 3.24168e-06.
Iteration 119. Log joint probability =    203.352. Improved by 1.08806e-06.
Iteration 120. Log joint probability =    203.352. Improved by 6.93898e-07.
Iteration 121. Log joint probability =    203.352. Improved by 1.05565e-06.
Iteration 122. Log joint probability =    203.352. Improved by 9.72e-07.
Iteration 123. Log joint probability =    203.352. Improved by 2.38021e-06.
Iteration 124. Log joint probability =    203.352. Improved by 1.08971e-06.
Iteration 125. Log joint probability =    203.352. Improved by 2.20965e-07.
Iteration 126. Log joint probability =    203.352. Improved by 4.72217e-08.
Iteration 127. Log joint probability =    203.352. Improved by 9.9767e-07.
Iteration 128. Log joint probability =    203.352. Improved by 7.18394e-07.
Iteration 129. Log joint probability =    203.352. Improved by 7.13988e-07.
Iteration 130. Log joint probability =    203.352. Improved by 2.69456e-07.
Iteration 131. Log joint probability =    203.352. Improved by 6.76646e-07.
```

```
Iteration 132. Log joint probability =    203.352. Improved by 1.73205e-07.
Iteration 133. Log joint probability =    203.352. Improved by 2.813e-07.
Iteration 134. Log joint probability =    203.352. Improved by 5.00236e-07.
Iteration 135. Log joint probability =    203.352. Improved by 3.6357e-07.
Iteration 136. Log joint probability =    203.352. Improved by 1.04654e-07.
Iteration 137. Log joint probability =    203.352. Improved by 3.1204e-07.
Iteration 138. Log joint probability =    203.352. Improved by 1.12459e-07.
Iteration 139. Log joint probability =    203.352. Improved by 8.66828e-08.
Iteration 140. Log joint probability =    203.352. Improved by 1.24744e-07.
Iteration 141. Log joint probability =    203.352. Improved by 1.0591e-07.
Iteration 142. Log joint probability =    203.352. Improved by 4.9268e-08.
Iteration 143. Log joint probability =    203.352. Improved by 5.73068e-08.
Iteration 144. Log joint probability =    203.352. Improved by 5.40003e-08.
Iteration 145. Log joint probability =    203.352. Improved by 3.44139e-08.
Iteration 146. Log joint probability =    203.352. Improved by 3.51918e-08.
Iteration 147. Log joint probability =    203.352. Improved by 9.42879e-09.
```



[117]:
```
#dos meses prediccion
fcst
```

[117]:

	time	fcst	fcst_lower	fcst_upper
0	2021-11-01	107482.352820	99127.996567	115886.736669
1	2021-12-01	111854.016906	103615.056779	120254.639907

0.0.3 Resultado final tras restar el gap

[118]:
```
df_predicciones = fcst
df_predicciones['nueva']= df_predicciones['fcst']-gap
```

[135]:
```
df_predicciones
```

[135]:

	time	fcst	fcst_lower	fcst_upper	nueva
0	2021-11-01	107482.352820	99127.996567	115886.736669	77698.765163
1	2021-12-01	111854.016906	103615.056779	120254.639907	82070.429248

[145]:
```
print('Predicción para Noviembre 2021: ', df_predicciones.nueva[0])
print('Predicción para Diciembre 2021: ', df_predicciones.nueva[1])
```

```
Predicción para Noviembre 2021:  77698.76516251125
Predicción para Diciembre 2021:  82070.42924791807
```

[ ]: