Лабораторная работа 6

Анализ и прогнозирование временного ряда.

Цель лабораторной работы: изучение основных методов анализа и прогнозирование временных рядов.

Задание: Выберите набор данных (датасет) для решения задачи прогнозирования временного ряда. Визуализируйте временной ряд и его основные характеристики. Разделите временной ряд на обучающую и тестовую выборку. Произведите прогнозирование временного ряда с использованием как минимум двух методов. Визуализируйте тестовую выборку и каждый из прогнозов. Оцените качество прогноза в каждом случае с помощью метрик.

In [1]:
```
import numpy as np
import pandas as pd

# Plots
# ==============================================================================
import matplotlib.pyplot as plt
plt.style.use('fivethirtyeight')
plt.rcParams['lines.linewidth'] = 1.5
%matplotlib inline

# Modeling and Forecasting
# ==============================================================================
from sklearn.linear_model import LinearRegression
from sklearn.linear_model import Lasso
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import MinMaxScaler, StandardScaler

from skforecast.ForecasterAutoreg import ForecasterAutoreg
from skforecast.ForecasterAutoregCustom import ForecasterAutoregCustom
from skforecast.ForecasterAutoregMultiOutput import ForecasterAutoregMul
from skforecast.model_selection import grid_search_forecaster
from skforecast.model_selection import backtesting_forecaster

from joblib import dump, load

# Warnings configuration
# ==============================================================================
import warnings
warnings.filterwarnings('ignore')
```

In [2]:
```
data = pd.read_csv('https://raw.githubusercontent.com/JoaquinAmatRodrigo
data
```

Out[2]:

	fecha	y	exog_1	exog_2
0	1992-04-01	0.379808	0.958792	1.166029

1	1992-05-01	0.361801	0.951993	1.117859
2	1992-06-01	0.410534	0.952955	1.067942
3	1992-07-01	0.483389	0.958078	1.097376
4	1992-08-01	0.475463	0.956370	1.122199
...	...	...	...	...
190	2008-02-01	0.761822	1.515840	1.786373
191	2008-03-01	0.649435	1.506258	1.694264
192	2008-04-01	0.827887	1.505253	1.627135
193	2008-05-01	0.816255	1.491464	1.555068
194	2008-06-01	0.762137	1.459856	1.463507

195 rows × 4 columns

Предварительная обработка

Удаляем столбцы с пустыми значениями:

In [3]:
```
data = data.dropna(axis=1, how='any')
data
```

Out[3]:

	fecha	y	exog_1	exog_2
0	1992-04-01	0.379808	0.958792	1.166029
1	1992-05-01	0.361801	0.951993	1.117859
2	1992-06-01	0.410534	0.952955	1.067942
3	1992-07-01	0.483389	0.958078	1.097376
4	1992-08-01	0.475463	0.956370	1.122199
...	...	...	...	...
190	2008-02-01	0.761822	1.515840	1.786373
191	2008-03-01	0.649435	1.506258	1.694264
192	2008-04-01	0.827887	1.505253	1.627135
193	2008-05-01	0.816255	1.491464	1.555068
194	2008-06-01	0.762137	1.459856	1.463507

195 rows × 4 columns

In [4]:
```
data = data.rename(columns={'fecha': 'date'})
data['date'] = pd.to_datetime(data['date'], format='%Y/%m/%d')
data = data.set_index('date')
data = data.asfreq(freq ='MS')
data = data.sort_index()
data
```

Out[4]:

	y	exog_1	exog_2
date			
1992-04-01	0.379808	0.958792	1.166029

1992-05-01	0.361801	0.951993	1.117859
1992-06-01	0.410534	0.952955	1.067942
1992-07-01	0.483389	0.958078	1.097376
1992-08-01	0.475463	0.956370	1.122199
...	...	...	...
2008-02-01	0.761822	1.515840	1.786373
2008-03-01	0.649435	1.506258	1.694264
2008-04-01	0.827887	1.505253	1.627135
2008-05-01	0.816255	1.491464	1.555068
2008-06-01	0.762137	1.459856	1.463507

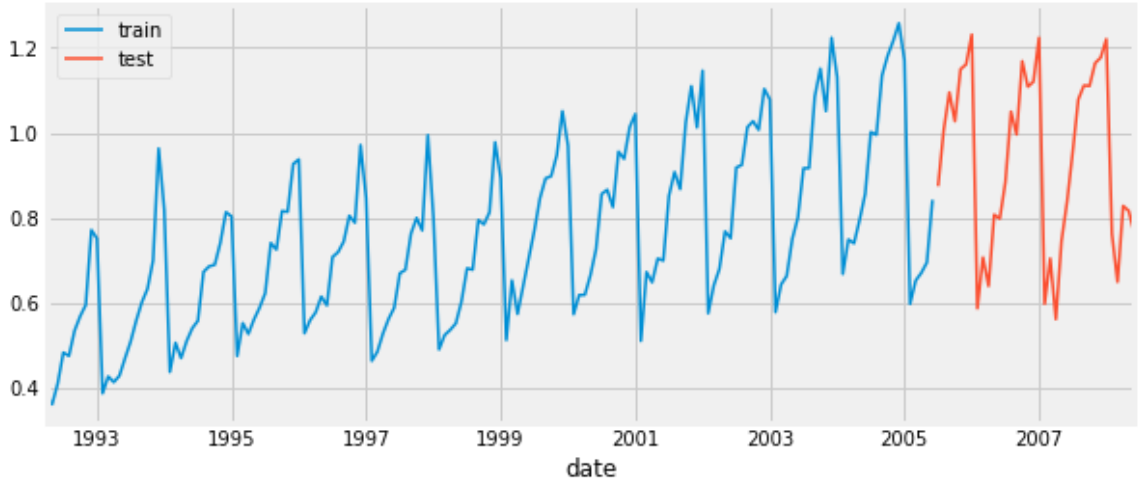
195 rows × 3 columns

Разделение выборки на обучающую и тестовую

In [5]:

```
import matplotlib.pyplot as plt
steps = 36
# scaler = MinMaxScaler().fit(data_train[['open']])

# data['open'] = scaler.transform(data[['open']])
data_train = data[:-steps]
data_test  = data[-steps:]
print(data)


print(f"Train dates : {data_train.index.min()} --- {data_train.index.max
print(f"Test dates  : {data_test.index.min()} --- {data_test.index.max()

fig, ax=plt.subplots(figsize=(9, 4))
data_train['y'].plot(ax=ax, label='train')
data_test['y'].plot(ax=ax, label='test')
ax.legend();
```

```
                 y    exog_1    exog_2
date
1992-04-01  0.379808  0.958792  1.166029
1992-05-01  0.361801  0.951993  1.117859
1992-06-01  0.410534  0.952955  1.067942
1992-07-01  0.483389  0.958078  1.097376
1992-08-01  0.475463  0.956370  1.122199
...              ...       ...       ...
2008-02-01  0.761822  1.515840  1.786373
2008-03-01  0.649435  1.506258  1.694264
2008-04-01  0.827887  1.505253  1.627135
2008-05-01  0.816255  1.491464  1.555068
2008-06-01  0.762137  1.459856  1.463507

[195 rows x 3 columns]
Train dates : 1992-04-01 00:00:00 --- 2005-06-01 00:00:00  (n=159)
Test dates  : 2005-07-01 00:00:00 --- 2008-06-01 00:00:00  (n=36)
```



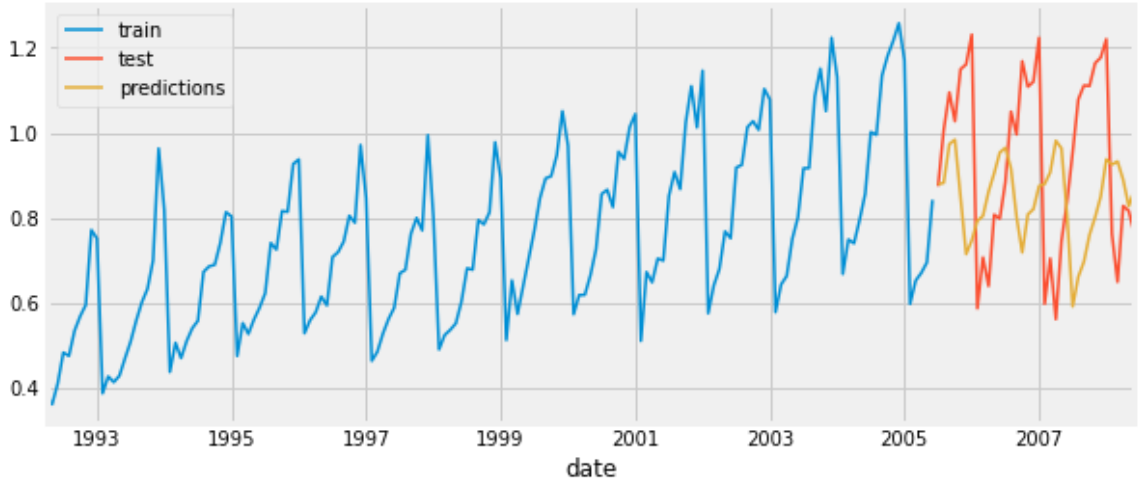
In [6]:
```
from sklearn.neighbors import KNeighborsRegressor
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import median_absolute_error, r2_score

def test_model(predictions):
    print('mean_absolute_error: {}'.format(round(mean_absolute_error(dat
    print('median_absolute_error: {}'.format(round(median_absolute_error
    print('r2_score: {}'.format(round(r2_score(data_test['y'], predictio
```

Обучение моделей

Skforecast-ForecasterAutoreg

In [7]:
```
forecaster = ForecasterAutoreg(
                regressor = RandomForestRegressor(random_state=123),
                lags = 6
                )

forecaster.fit(y=data_train['y'])
forecaster
steps = 36
predictions = forecaster.predict(steps=steps)
```

In [8]:
```
fig, ax2 = plt.subplots(figsize=(9, 4))
data_train['y'].plot(ax=ax2, label='train')
data_test['y'].plot(ax=ax2, label='test')
predictions.plot(ax=ax2, label='predictions')
ax2.legend();
```



In [9]:

```
lags_grid = [10, 16]

# Regressor's hyperparameters
param_grid = {'n_estimators': [100, 500],
              'max_depth': [3, 5, 10]}

results_grid = grid_search_forecaster(
                        forecaster         = forecaster,
                        y                  = data_train['y'],
                        param_grid         = param_grid,
                        lags_grid          = lags_grid,
                        steps              = steps,
                        refit              = True,
                        metric             = 'mean_squared_error',
                        initial_train_size = int(len(data_train)*0.5),
                        fixed_train_size   = False,
                        return_best        = True,
                        verbose            = False
               )
```

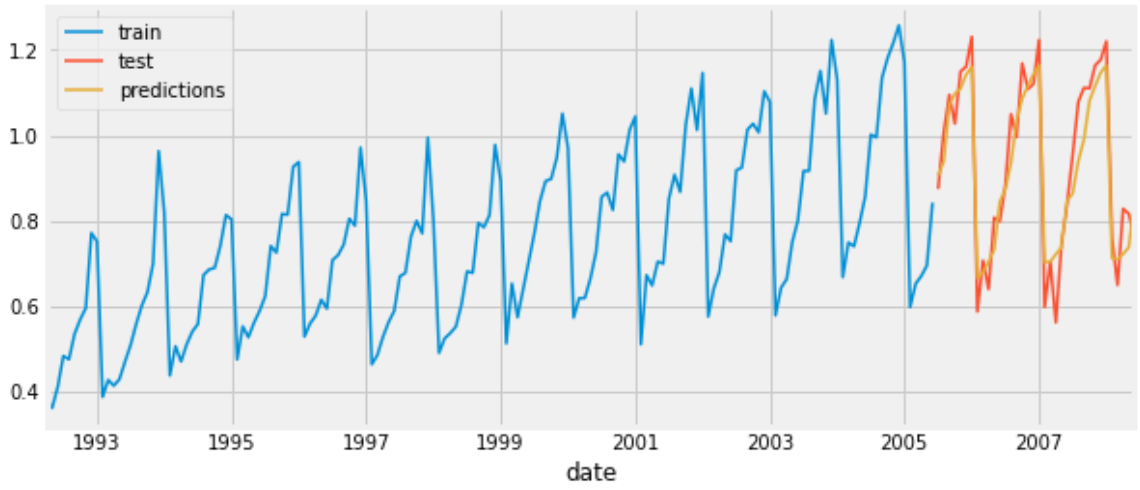
```
Number of models compared: 12
loop lags_grid:   0%|                                                  | 0/2
[00:00<?, ?it/s]
loop param_grid:   0%|                                                 | 0/6
[00:00<?, ?it/s]
loop param_grid:  17%|██████                                  | 1/6 [00:02
<00:13,  2.74s/it]
loop param_grid:  33%|████████████                            | 2/6 [00:18
<00:42, 10.69s/it]
loop param_grid:  50%|██████████████████                      | 3/6 [00:22<
00:22,  7.35s/it]
loop param_grid:  67%|████████████████████████                | 4/6 [00:36
<00:20, 10.16s/it]
loop param_grid:  83%|██████████████████████████████          | 5/6 [00:39
<00:07,  7.43s/it]
loop param_grid: 100%|████████████████████████████████████| 6/6 [00:50<
00:00,  8.54s/it]
loop lags_grid:  50%|██████████████████                      | 1/2 [00:50<
00:50, 50.13s/it]
loop param_grid:   0%|                                                 | 0/6
[00:00<?, ?it/s]
loop param_grid:  17%|██████                                  | 1/6 [00:01
<00:08,  1.72s/it]
loop param_grid:  33%|████████████                            | 2/6 [00:11
<00:25,  6.38s/it]
```

```
loop param_grid:  50%|████████████████                | 3/6 [00:14<
00:14,  4.68s/it]
loop param_grid:  67%|█████████████████████           | 4/6 [00:23
<00:13,  6.63s/it]
loop param_grid:  83%|██████████████████████████      | 5/6 [00:26
<00:05,  5.31s/it]
loop param_grid: 100%|████████████████████████████████| 6/6 [00:36<
00:00,  6.88s/it]
loop lags_grid: 100%|████████████████████████████████| 2/2 [01:26<
00:00, 43.34s/it]
`Forecaster` refitted using the best-found lags and parameters, and the w
hole data set:
  Lags: [ 1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16]
  Parameters: {'max_depth': 3, 'n_estimators': 100}
  Backtesting metric: 0.012757579470218128
```

In [10]:

```
regressor = RandomForestRegressor(max_depth=3, n_estimators=100, random_
forecaster = ForecasterAutoreg(
                regressor = regressor,
                lags      = 16
             )

forecaster.fit(y=data_train['y'])
predictions = forecaster.predict(steps=steps)
fig, ax = plt.subplots(figsize=(9, 4))
data_train['y'].plot(ax=ax, label='train')
data_test['y'].plot(ax=ax, label='test')
predictions.plot(ax=ax, label='predictions')
ax.legend();
test_model(predictions)
```

```
mean_absolute_error: 0.06
median_absolute_error: 0.06
r2_score: 0.88
```



SARIMAX

In [11]:

```
SARIMAXmodel = SARIMAX(data_train['y'], order = (2, 1, 2), seasonal_orde
SARIMAXmodel = SARIMAXmodel.fit()

y_pred = SARIMAXmodel.get_forecast(len(data_test.index))
y_pred_df = y_pred.conf_int(alpha = 0.05)
y_pred_df["Predictions"] = SARIMAXmodel.predict(start = y_pred_df.index[
y_pred_df.index = data_test.index
```

```
y_pred_out = y_pred_df["Predictions"]
fig, ax2 = plt.subplots(figsize=(9, 4))
data_train['y'].plot(ax=ax2, label='train')
data_test['y'].plot(ax=ax2, label='test')
y_pred_out.plot(ax=ax2, label='predictions')
ax2.legend()
test_model(y_pred_out)
```

```
---------------------------------------------------------------------------
--
NameError                                 Traceback (most recent call las
t)
C:\Users\3C8A~1\AppData\Local\Temp/ipykernel_9832/1934187388.py in <modul
e>
----> 1 SARIMAXmodel = SARIMAX(data_train['y'], order = (2, 1, 2), season
al_order=(2,1,2,12))
      2 SARIMAXmodel = SARIMAXmodel.fit()
      3
      4 y_pred = SARIMAXmodel.get_forecast(len(data_test.index))
      5 y_pred_df = y_pred.conf_int(alpha = 0.05)

NameError: name 'SARIMAX' is not defined
```