# Обнаружение аномалий

import numpy as np  
import pandas as pd  
  
import matplotlib.pyplot as plt  
from datetime import datetime  
import seaborn as sns  
sns.set(style="whitegrid")  
  
import warnings  
warnings.filterwarnings('ignore')  
  
RANDOM\_SEED = np.random.seed(0)

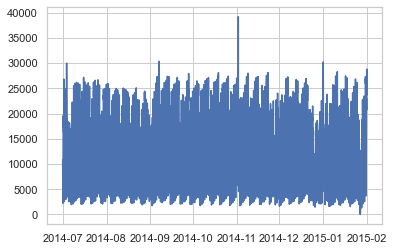
Данные: набор востребованности такси в городе Нью Йорк с шагом в 1 час.

import pandas as pd  
data = pd.read\_csv('https://raw.githubusercontent.com/numenta/NAB/master/data/realKnownCause/nyc\_taxi.csv')  
data['timestamp'] = pd.to\_datetime(data['timestamp'])  
data = data.set\_index(['timestamp']).asfreq('h')  
data.head()

value  
timestamp   
2014-07-01 00:00:00 10844  
2014-07-01 01:00:00 6210  
2014-07-01 02:00:00 3820  
2014-07-01 03:00:00 2369  
2014-07-01 04:00:00 2221

y = data.value.values.reshape(-1, 1)

plt.plot(data.index, y)

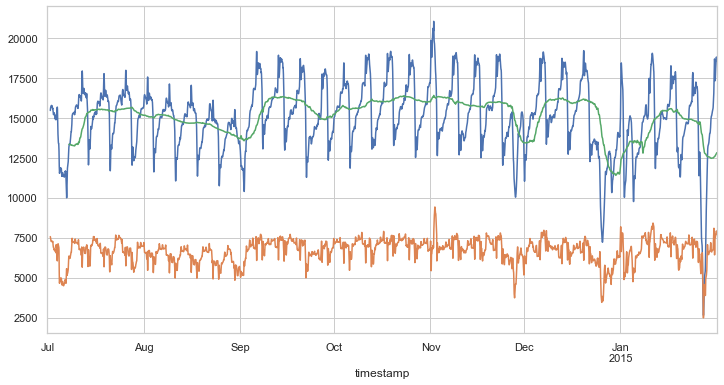


Данные во временных рядах можно рассматривать как в чистом виде, так и преобразовать.

Попробуем скользящее средние и скользящие СКО

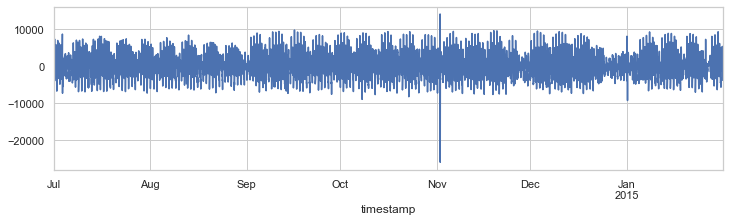
plt.rc('figure',figsize=(12,6))  
plt.rc('font',size=15)  
# create moving-averages  
data['value'].rolling(24).mean().plot()  
data['value'].rolling(24).std().plot()  
data['value'].rolling(24\*7).mean().plot()

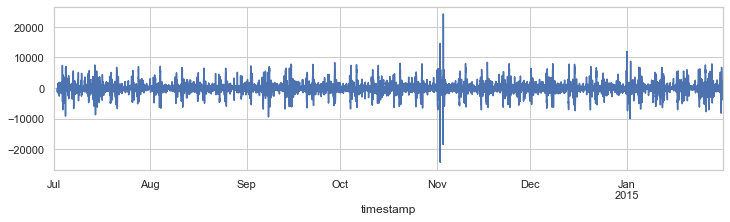
<AxesSubplot:xlabel='timestamp'>



Дифференцирование

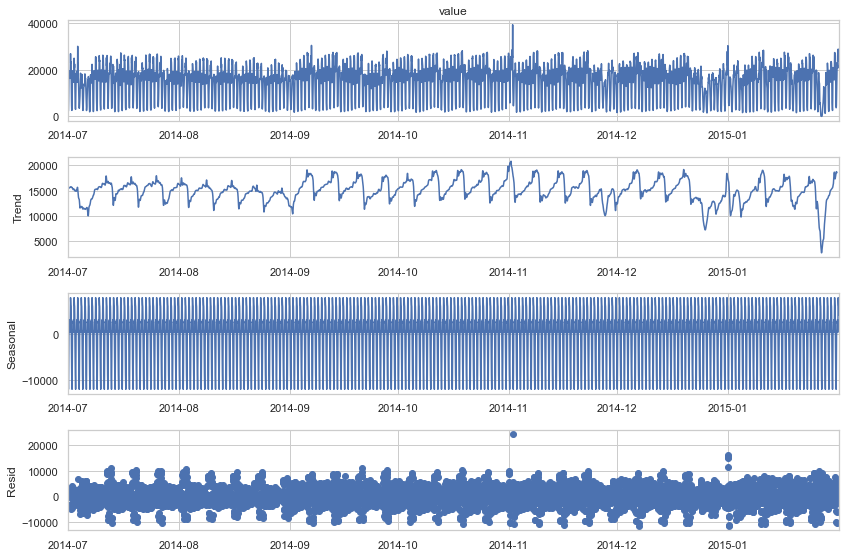
plt.rc('figure',figsize=(12,3))  
plt.rc('font',size=15)  
  
data.value.diff(1).plot(); plt.show()  
data.value.diff(24).diff(1).plot(); plt.show()





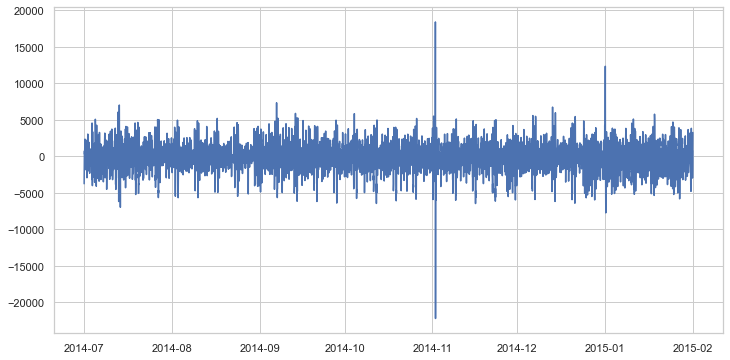
Простое разложение

from statsmodels.tsa.seasonal import seasonal\_decompose  
plt.rc('figure',figsize=(12,8))  
plt.rc('font',size=15)  
  
result = seasonal\_decompose(data.value,model='additive')  
fig = result.plot()



сравнение (остаток, метрика) от предсказания простой моделью

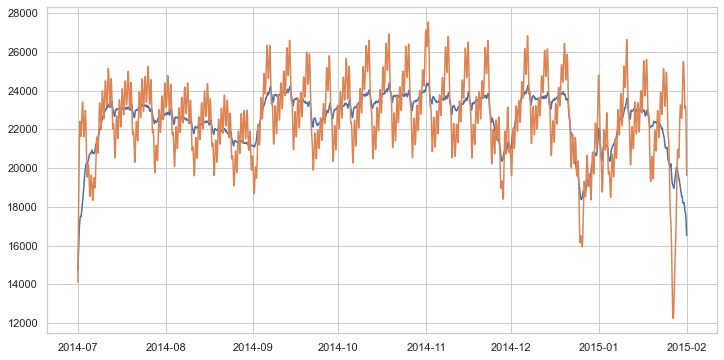
from statsmodels.tsa.api import ExponentialSmoothing, SimpleExpSmoothing, Holt  
forecaster = ExponentialSmoothing(y,seasonal\_periods=48,trend="add",seasonal="add").fit()  
predicts = forecaster.predict(start=0, end=len(y)-1)  
  
plt.rc('figure',figsize=(12,6))  
plt.plot(data.index, y.reshape(-1) - predicts)



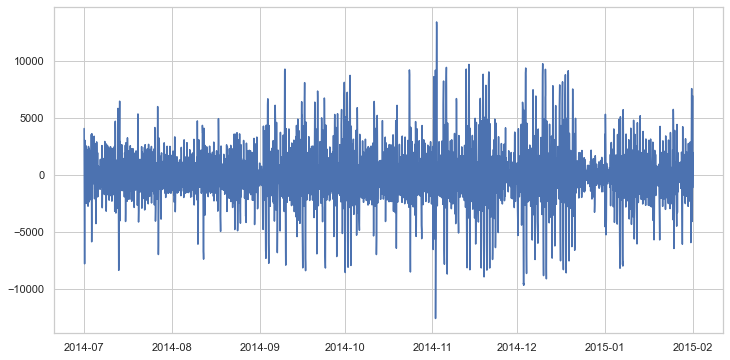
Более тонкие разложения, например по т.н. внутренним модам

import dsatools   
y\_dec = dsatools.decomposition.vmd(y.reshape(-1)[:], order =4)  
plt.plot(data.index[1:-1],y\_dec[0,1:-1])  
  
y\_dec = dsatools.decomposition.vmd(y.reshape(-1)[:], order =2)  
plt.plot(data.index[1:-1],y\_dec[0,1:-1])

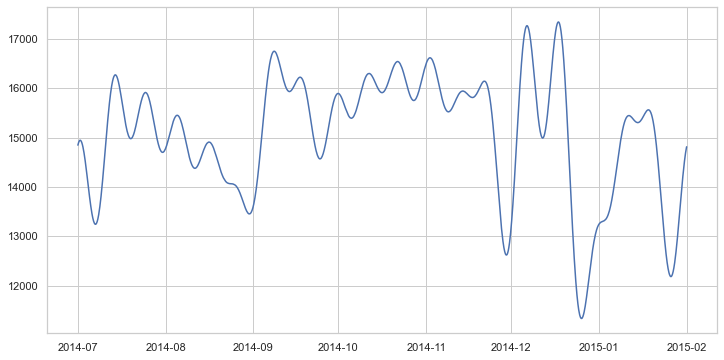
[<matplotlib.lines.Line2D at 0x240926c8da0>]



y\_dec = dsatools.decomposition.emd(y.reshape(-1)[:], order =10)  
plt.plot(data.index[1:-1],y\_dec[0,1:-1]); plt.show()

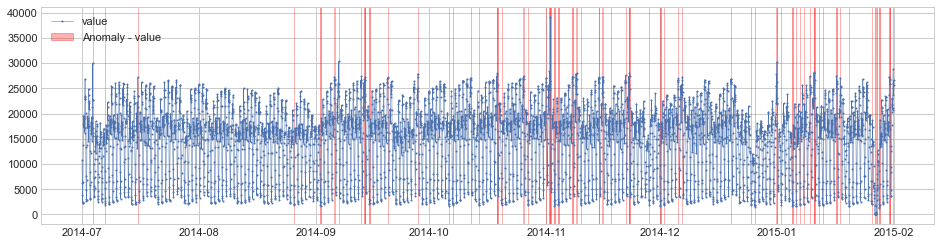


y\_dec = dsatools.decomposition.hvd(y.reshape(-1)[:], order =2)  
plt.plot(data.index[1:-1],y\_dec[0,1:-1]); plt.show()

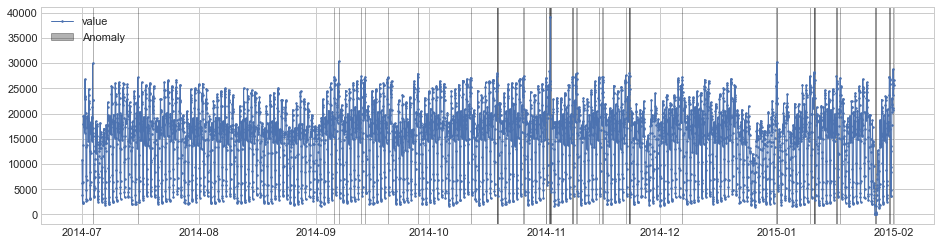


Ряд может быть обработан на основе четких правил

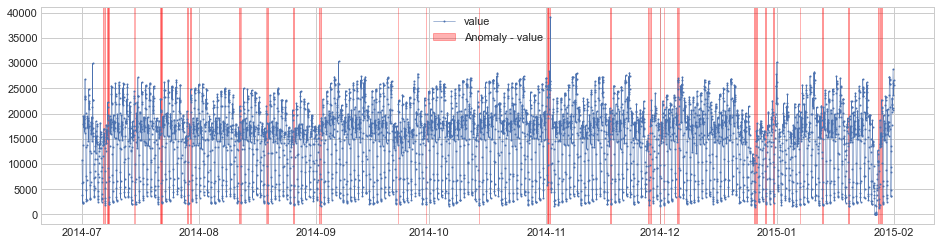
from adtk.detector import QuantileAD  
from adtk.visualization import plot  
quantile\_ad = QuantileAD(high=0.99, low=0.01)  
anomalies = quantile\_ad.fit\_detect(data)  
plot(data, anomaly=anomalies, anomaly\_color='red');



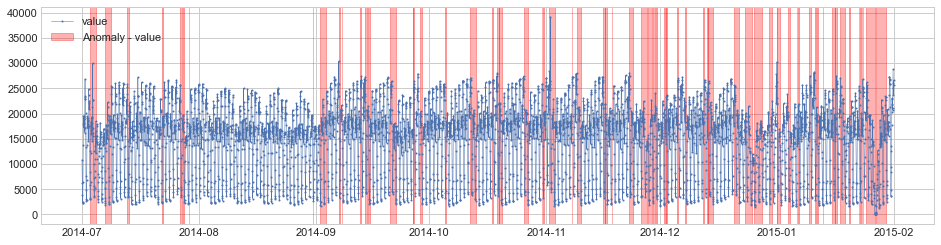
from adtk.detector import ThresholdAD  
threshold\_val = ThresholdAD(high=27000, low=1000)  
anomalies\_thresh = threshold\_val.detect(data.value)  
anomalies\_thresh.value\_counts()  
plot(data, anomaly=anomalies\_thresh, ts\_linewidth=1, ts\_markersize=3, anomaly\_markersize=5, anomaly\_color='black');



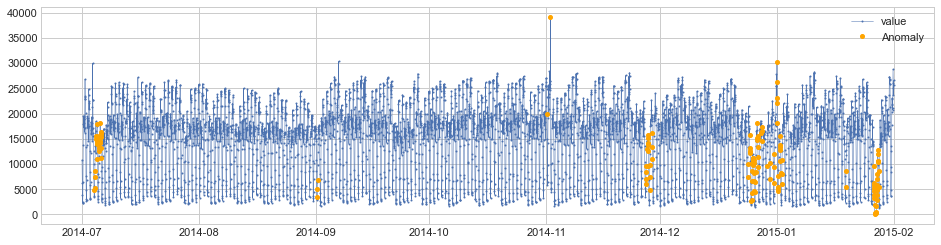
from adtk.detector import VolatilityShiftAD  
volatility\_shift\_ad = VolatilityShiftAD(c=1.0, side='positive', window=30)  
anomalies = volatility\_shift\_ad.fit\_detect(data)  
plot(data, anomaly=anomalies, anomaly\_color='red');



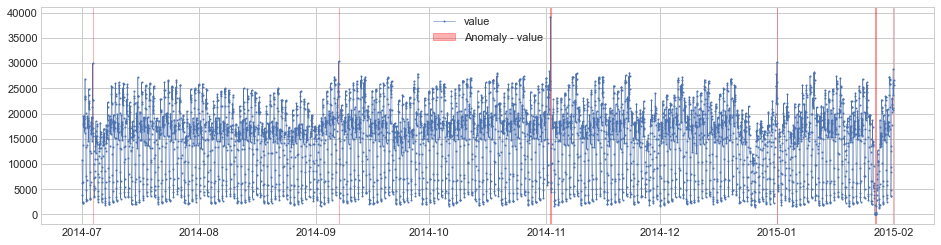
from adtk.detector import LevelShiftAD  
level\_shift\_ad = LevelShiftAD(c=0.2, side='both', window=50)  
anomalies = level\_shift\_ad.fit\_detect(data)  
plot(data, anomaly=anomalies, anomaly\_color='red');



from adtk.detector import SeasonalAD  
seasonal\_vol = SeasonalAD()  
anomalies = seasonal\_vol.fit\_detect(data.value)  
anomalies.value\_counts()  
  
plot(data, anomaly=anomalies, anomaly\_color="orange", anomaly\_tag="marker");

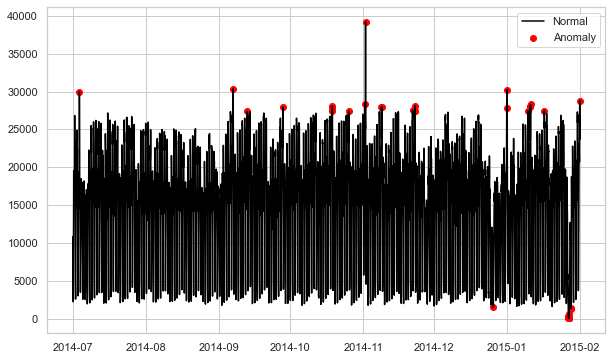


from adtk.detector import InterQuartileRangeAD  
iqr\_ad = InterQuartileRangeAD(c=0.9)  
anomalies = iqr\_ad.fit\_detect(data)  
plot(data, anomaly=anomalies, anomaly\_color='red');

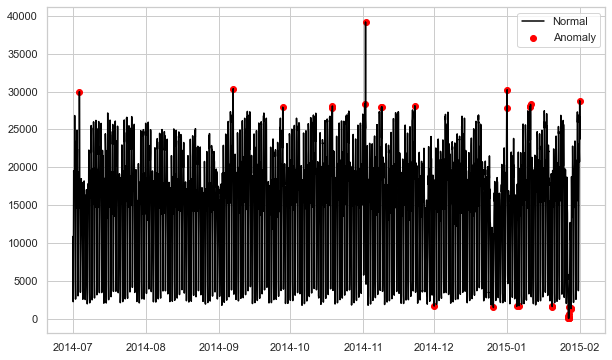


Также могут быть использованы специализированные алгоритмы

from sklearn.ensemble import IsolationForest  
  
outliers\_fraction = 0.007  
  
model = IsolationForest(contamination=outliers\_fraction)  
model.fit(data.value.values.reshape(-1, 1))  
  
data['anomaly\_IF'] = model.predict(y)  
  
# visualization  
fig, ax = plt.subplots(figsize=(10,6))  
  
a = data.loc[data['anomaly\_IF'] == -1, ['value']] #anomaly  
  
ax.plot(data.index, data['value'], color='black', label = 'Normal')  
ax.scatter(a.index,a['value'], color='red', label = 'Anomaly')  
plt.legend()  
plt.show();



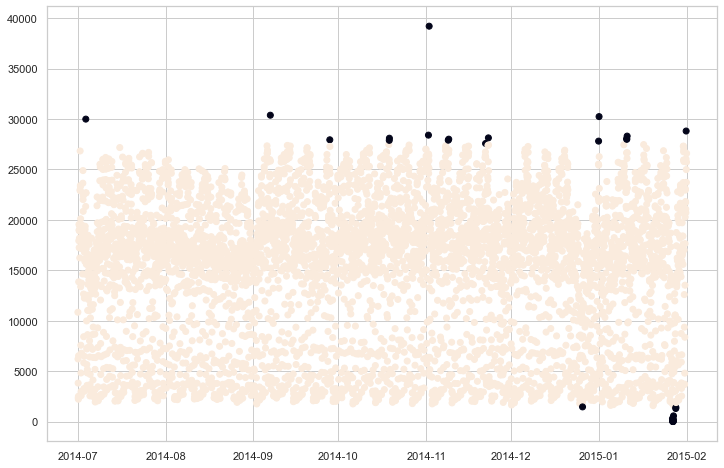
from sklearn.neighbors import LocalOutlierFactor  
  
lof = LocalOutlierFactor(novelty=True)  
  
lof.fit(y)  
  
data['anomaly\_LOF'] = lof.predict()  
  
  
fig, ax = plt.subplots(figsize=(10,6))  
  
a = data.loc[data['anomaly\_LOF'] == -1, ['value']] #anomaly  
  
ax.plot(data.index, data['value'], color='black', label = 'Normal')  
ax.scatter(a.index,a['value'], color='red', label = 'Anomaly')  
plt.legend()  
plt.show();



from sklearn.cluster import DBSCAN, KMeans  
model=DBSCAN(eps = 100.)  
# model = KMeans(n\_clusters=10)  
  
model.fit(y)  
colors = model.labels\_  
cls,counts = np.unique(colors, return\_counts=True)  
sort\_idx = np.argsort(counts)#[::-1]  
print(counts[sort\_idx], cls[sort\_idx])  
plt.scatter(data.index, y, c = colors == cls[sort\_idx][-1])

[ 7 9 14 5130] [ 2 1 -1 0]

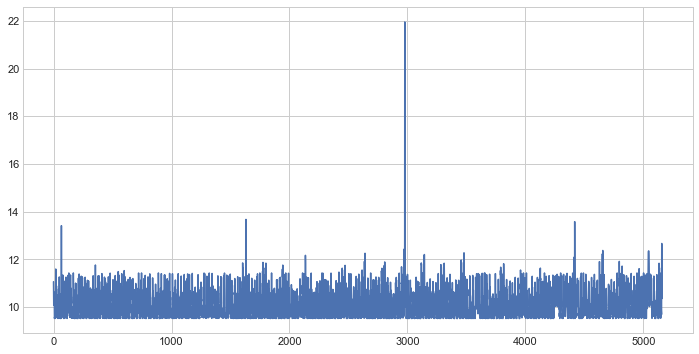
<matplotlib.collections.PathCollection at 0x29567b884e0>



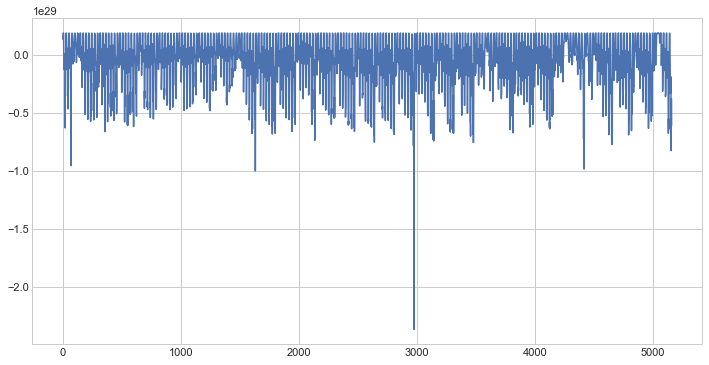
По мимо sklearn есть и другие библиотеки работы с аномалиями, например <https://pyod.readthedocs.io/en/latest/>

import pyod.models  
from pyod.models.gmm import GMM  
  
clf = GMM( n\_components=2, )  
clf.fit(y)  
y\_scores = clf.decision\_scores\_  
plt.plot(y\_scores)

[<matplotlib.lines.Line2D at 0x240adad3978>]

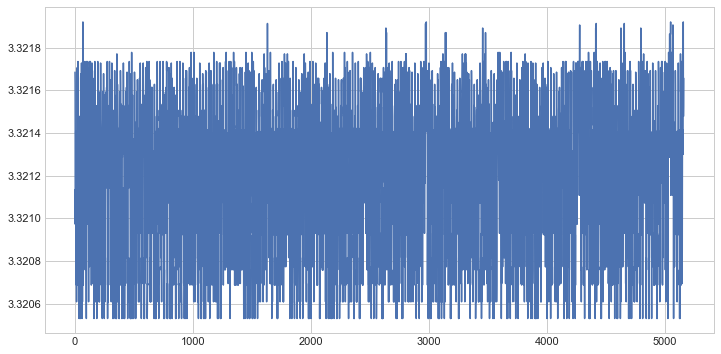


from pyod.models.ocsvm import OCSVM  
pyod.models.anogan.AnoGAN  
clf = OCSVM(kernel='poly',degree=3)  
clf.fit(y)  
y\_scores = clf.decision\_scores\_  
plt.plot(y\_scores)



from pyod.models.hbos import HBOS  
  
clf = HBOS(n\_bins=100, alpha=0.1, tol=0.5, contamination=0.1)  
clf.fit(y)  
y\_scores = clf.decision\_scores\_  
plt.plot(y\_scores)

[<matplotlib.lines.Line2D at 0x240acc81a20>]

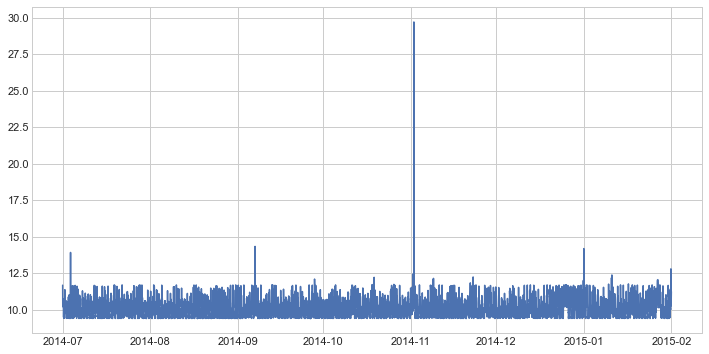


в данной библиотеки pyod можно в т.ч. реализовать попробовать на основе восстановления автоэнокдером

import numpy as np  
import pandas as pd  
from pyod.models.auto\_encoder import AutoEncoder  
  
clf = AutoEncoder(hidden\_neurons =[1,125, 125, 1])  
clf.fit(y)  
y\_scores = clf.decision\_scores\_

Model: "sequential\_1"  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
 Layer (type) Output Shape Param #   
=================================================================  
 dense\_21 (Dense) (None, 1) 2   
   
 dropout\_18 (Dropout) (None, 1) 0   
   
 dense\_22 (Dense) (None, 1) 2   
   
 dropout\_19 (Dropout) (None, 1) 0   
   
 dense\_23 (Dense) (None, 1) 2   
   
 dropout\_20 (Dropout) (None, 1) 0   
   
 dense\_24 (Dense) (None, 125) 250   
   
 dropout\_21 (Dropout) (None, 125) 0   
   
 dense\_25 (Dense) (None, 125) 15750   
   
 dropout\_22 (Dropout) (None, 125) 0   
   
 dense\_26 (Dense) (None, 1) 126   
   
 dropout\_23 (Dropout) (None, 1) 0   
   
 dense\_27 (Dense) (None, 1) 2   
   
=================================================================  
Total params: 16,134  
Trainable params: 16,134  
Non-trainable params: 0  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

plt.plot(data.index,y\_scores); plt.show()



<https://www.kaggle.com/code/joshuaswords/time-series-anomaly-detection>

<https://github.com/rob-med/awesome-TS-anomaly-detection>

<https://github.com/yzhao062/anomaly-detection-resources>

Вопросы:

1 какие методы представления мы рассмотрели:

* Дифференцирование
* Преобразование бокса кокса
* Разложение в ряд Фурье
* Разложение в ряд Уошла-Адомара

Ответ: Дифференцирование

2 какой метод позволил выделить аномалии на основе четких правил

* QuantileAD
* isolationForest
* OСSVM
* AutoEncoder

Ответ: QuantileAD

3 Какой метод выделил аномалии типа гетеросекдастичности

* SeasonalAD
* ThresholdAD
* InterQuartileRangeAD
* VolatilityShiftAD

Ответ: VolatilityShiftAD

4 какой метод из пакета SKLearn мы рассмотрели

* LocalOutlierFactor
* GMM
* AutoEncoder
* QuantileAD

Ответ LocalOutlierFactor