Kod R

21.04.2022

```
train<-read.csv(”C:\\Studia\\MAGISTERKA\\2.␣semestr\\Warsztaty\\dane\\train.csv”)
head(train)

train<-train[,-1]
str(train)
train$date<-as.Date(train$date)
train2<-train[train$date<as.Date(’2016-06-01’),]
test2<-train[train$date>=as.Date(’2016-06-01’),]

train2_mean<-aggregate(train2$sales, list(train2$date), FUN=mean)
head(train2_mean)

plot(train2_mean$x~train2_mean$Group.1,type=”l”)


ts<-ts(train2_mean$x)
head(ts)
ts.plot(ts)

acf(ts) #widac sezonosc i trend
ts_diff<-diff(ts)

ts.plot(ts_diff)
acf(ts_diff) #pozbylismy si  trendu, ale zostala sezonowosc
pacf(ts_diff)

#wyznaczmy okres
spec<-spectrum(ts_diff)
#widac ze nie jest to bialy szum
#wyznaczmy okres
cpgram(ts_diff)

(spec$freq[order(-spec$spec)[1:2]])
# 0.2864583 0.1435185

#czyli okresy
1/(spec$freq[order(-spec$spec)[1:2]])
#4(takie 3.5) i 7

1/spec$freq[which.max(spec$spec)] #oko o 7 przy freq=1
#0.0190897 przy freq=365
#czyli okres to oko o tydzie

okres<-ceiling(1/spec$freq[which.max(spec$spec)])
okres

#roznicujemy o okres
ts_diff2<-diff(ts_diff,lag=7)
```

```
acf(ts_diff2)
pacf(ts_diff2, lag.max=100)

#jedno roznicowanie z lagiem -> D=1
#jedno zwykle roznicowanie => d=1
#z acf(ts_diff2) bysmy moze wzieli q=6 alob q=2, Q=1
#z pacf  => p=6, P=5
ar(ts_diff2)
30/7

sarima<-arima(ts,order=c(7,1,2),seasonal=list(order=c(8,1,6),period=7))
?arima
#AIC(arima(ts,order=c(6,1,6),seasonal=list(order=c(6,1,1),period=7)))
Box.test(sarima$residuals,lag=round(sqrt(length(sarima$residuals))),
type="Ljung-Box",fitdf=22)
#p-value = 8.818e-05 => nie jest to bialy szum

acf(sarima$residuals)
pacf(sarima$residuals,lag.max=100)
```

Modelling

April 21, 2022

[1]:

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import statsmodels
import statsmodels.api as sm
import statsmodels.formula.api as smf
import datetime as dt
import scipy.signal as ss
```

[2]:

```python
from statsmodels.tsa.seasonal import seasonal_decompose
```

[3]:

```python
holidays_events = pd.read_csv("https://www.dropbox.com/s/bxyamlpevkiwwoq/
 ↪holidays_events.csv?dl=1")
oil = pd.read_csv("https://www.dropbox.com/s/l6ln0ztl4m0pw3a/oil.csv?
 ↪dl=1",parse_dates=['date'],index_col='date')
sample_submission = pd.read_csv("https://www.dropbox.com/s/68jjl61x6u3klos/
 ↪sample_submission.csv?dl=1")
stores = pd.read_csv("https://www.dropbox.com/s/lcxn6r9bs2exguq/stores.csv?
 ↪dl=1")
test = pd.read_csv("https://www.dropbox.com/s/cvdo1gn7r5lu2uz/test.csv?
 ↪dl=1",index_col='id')
train = pd.read_csv("https://www.dropbox.com/s/s8p2b5awnuqfk0d/train.csv?
 ↪dl=1",index_col='id')
transactions = pd.read_csv("https://www.dropbox.com/s/92fij9bcwt0e0cj/
 ↪transactions.csv?dl=1")
```

[4]:

```python
train['date'] = pd.to_datetime(train['date'])
```

Dzielimy próbkę na treningową i testową.

[5]:

```python
train2 = train.loc[(train['date']<'2016-06-01')]
```

[6]:

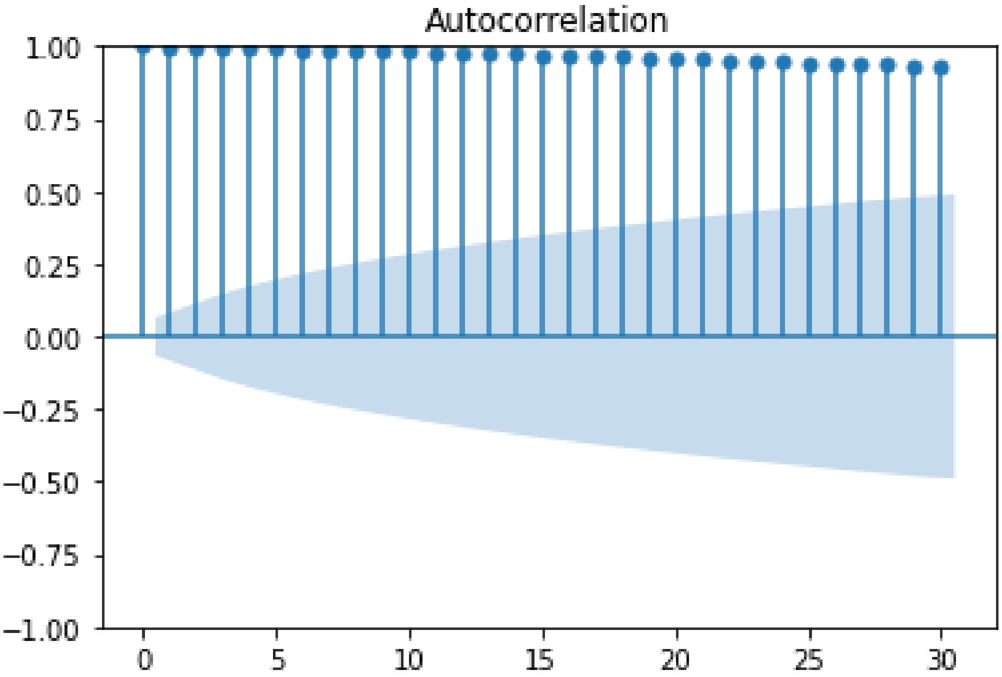
```python
test2 = train.loc[(train['date']>='2016-06-01')]
```

[7]:

```python
oil_train2 = oil.loc[(oil.index<'2016-06-01')].fillna(method ="bfill")
oil_test2 = oil.loc[(oil.index>='2016-06-01')].fillna(method ="bfill")
```

[8]:

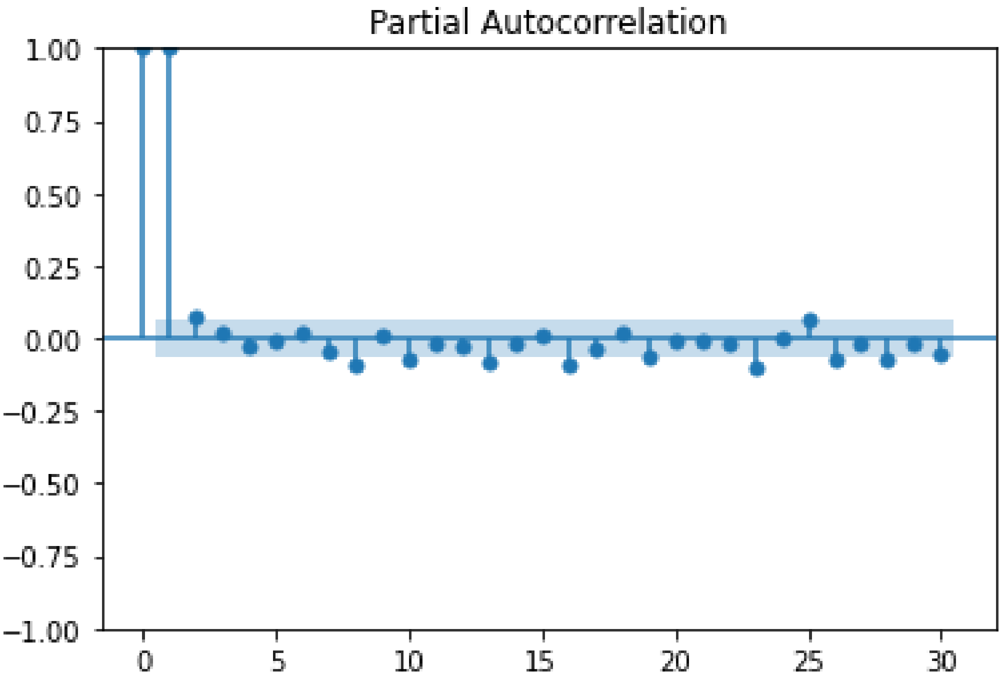
```
sm.graphics.tsa.plot_acf(oil_train2,lags=np.round(np.sqrt(len(oil_train2))))
plt.show()
```



[9]:

```
sm.graphics.tsa.plot_pacf(oil_train2,lags=np.round(np.sqrt(len(oil_train2))))
plt.show()
```

```
C:\Users\Lenovo\AppData\Local\Programs\Python\Python39\lib\site-
packages\statsmodels\graphics\tsaplots.py:348: FutureWarning: The default method
'yw' can produce PACF values outside of the [-1,1] interval. After 0.13, the
default will change tounadjusted Yule-Walker ('ywm'). You can use this method
now by setting method='ywm'.
  warnings.warn(
```



[10]:

```
oil_diff = oil_train2.shift().diff().dropna()
```

[11]:

```
sm.stats.acorr_ljungbox(oil_diff, lags=[np.round(np.sqrt(len(oil_diff)))],␣
 ↪return_df=True)
```

[11]:

```
     lb_stat  lb_pvalue
30  33.33492   0.308193
```

Zatem jest to biały szum.

[13]:

```
mod_oil = sm.tsa.arima.ARIMA(oil_train2,order=(0,1,0)).fit()
```

```
C:\Users\ndzad\anaconda3\lib\site-
packages\statsmodels\tsa\base\tsa_model.py:524: ValueWarning: No frequency
information was provided, so inferred frequency B will be used.
  warnings.warn('No frequency information was'
C:\Users\ndzad\anaconda3\lib\site-
packages\statsmodels\tsa\base\tsa_model.py:524: ValueWarning: No frequency
information was provided, so inferred frequency B will be used.
  warnings.warn('No frequency information was'
C:\Users\ndzad\anaconda3\lib\site-
packages\statsmodels\tsa\base\tsa_model.py:524: ValueWarning: No frequency
information was provided, so inferred frequency B will be used.
  warnings.warn('No frequency information was'
```

[14]:

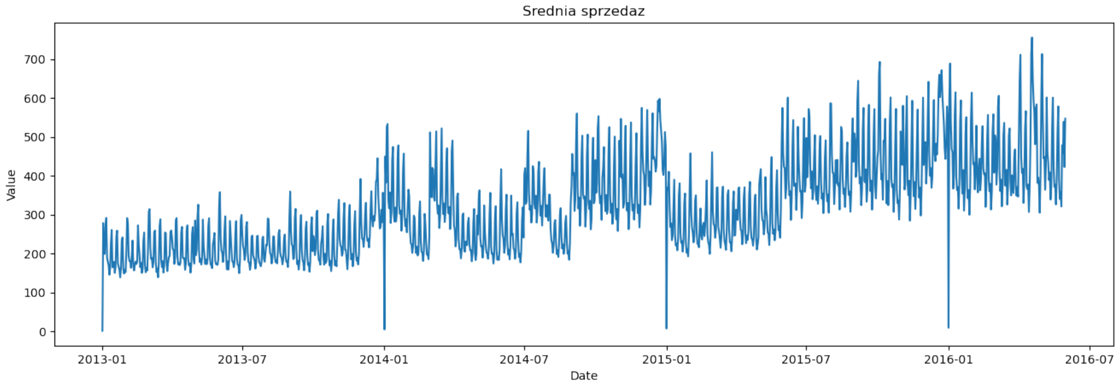
```
train2_avg= train2.groupby('date')['sales'].mean().to_frame()
```

[15]:

```
#Funkcja pomocniczna do rysowania wykresów
def plot_df(df, x, y, title="", xlabel='Date', ylabel='Value', dpi=100,␣
 ↪axiscolor ='black'):
    plt.figure(figsize=(16,5), dpi=dpi)
    plt.plot(x, y, color='tab:blue')
    plt.gca().set(title=title, xlabel=xlabel, ylabel=ylabel)
    plt.gca().title.set_color(axiscolor)
    plt.gca().xaxis.label.set_color(axiscolor)
    plt.gca().yaxis.label.set_color(axiscolor)
    plt.tick_params(colors=axiscolor, which='both')
    plt.show()
```

[16]:

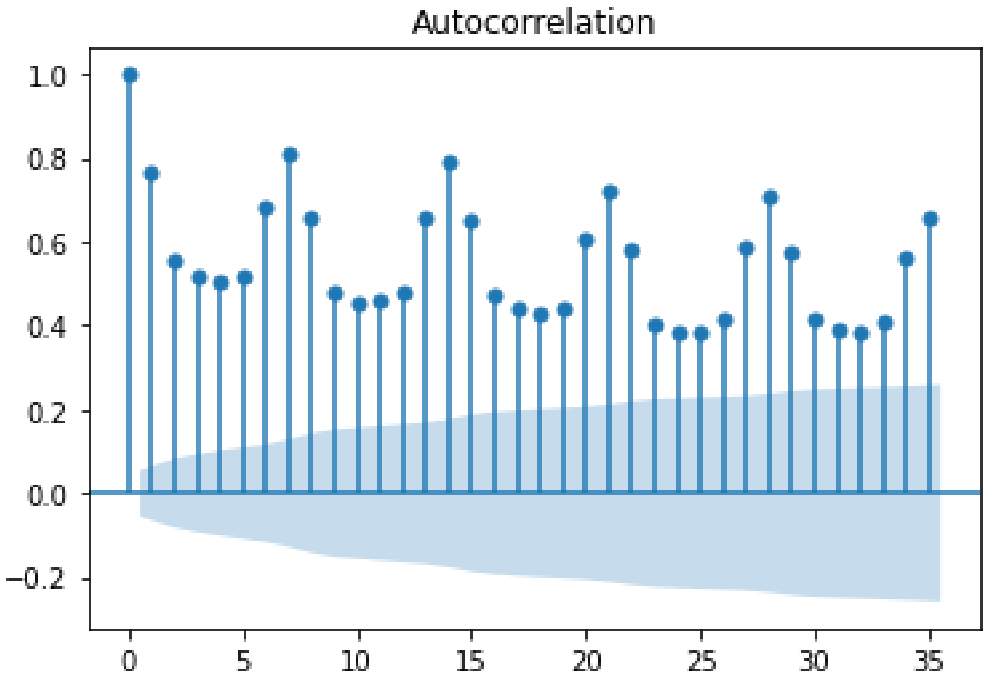
```
plot_df(train2_avg,x=train2_avg.index, y=train2_avg.sales, title='Srednia␣
 ↪sprzedaz', axiscolor='black')
```



Wykres autokorelacji (ACF)

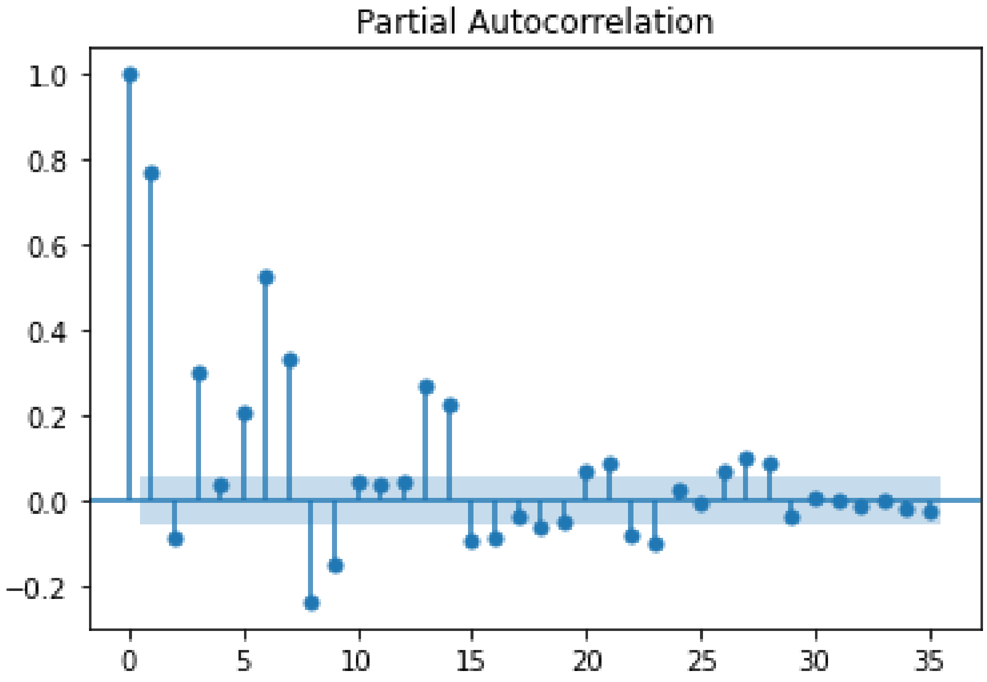
[17]:

```
sm.graphics.tsa.plot_acf(train2_avg,lags=np.round(np.sqrt(len(train2_avg))))
plt.show()
```



Wykres cześciowych korelacji (PACF)

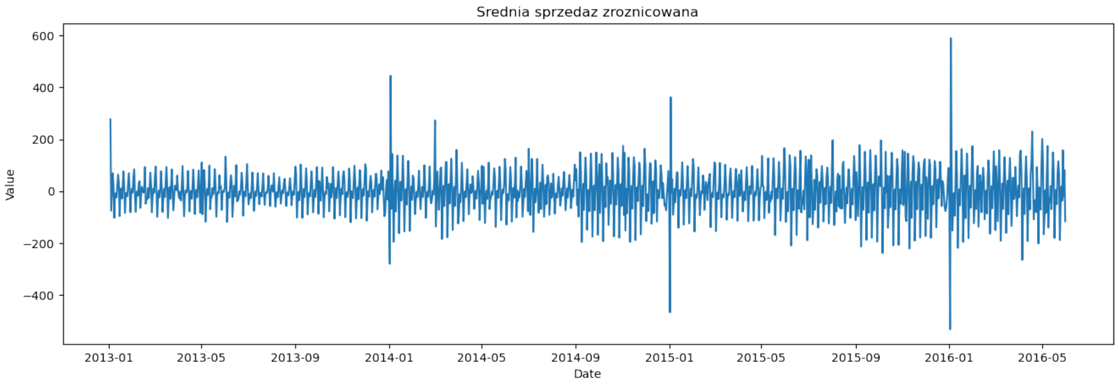
```
[18]: sm.graphics.tsa.plot_pacf(train2_avg,lags=np.round(np.sqrt(len(train2_avg))))
plt.show()
```



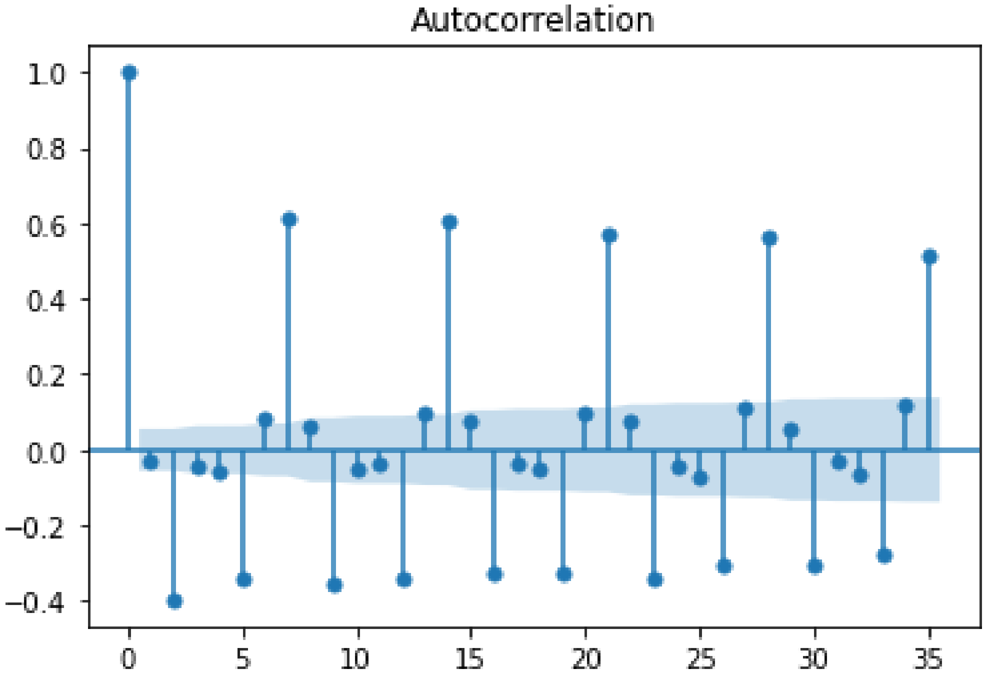
Różnicujemy szereg

[19]:
```
train2_avg_diff = train2_avg.shift().diff().dropna()
```

[32]:
```
plot_df(train2_avg_diff,x=train2_avg_diff.index, y=train2_avg_diff.sales,␣
 ↪title='Srednia sprzedaz zroznicowana', axiscolor='black')
```

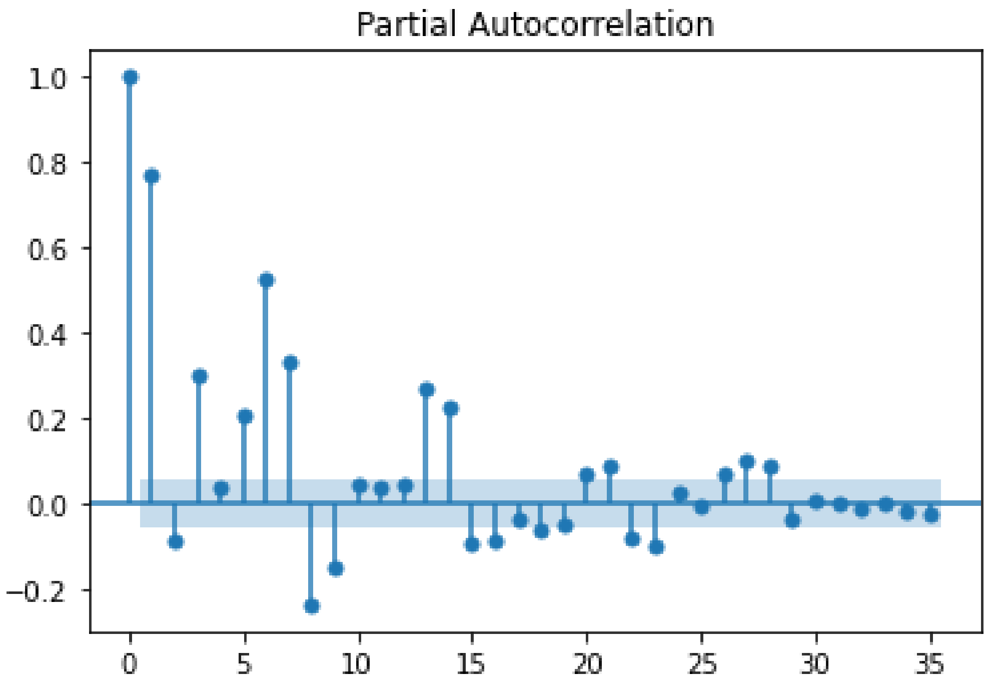


[20]:
```
sm.graphics.tsa.plot_acf(train2_avg_diff,lags=np.round(np.
 ↪sqrt(len(train2_avg_diff))))
plt.show()
```



[21]:

```
sm.graphics.tsa.plot_pacf(train2_avg,lags=np.round(np.sqrt(len(train2_avg))))
plt.show()
```



[88]:

```
f, Pxx=ss.periodogram(train2_avg_diff,365)
```

[90]:

```
Pxx
```

[90]:

```
array([[0.],
       [0.],
       [0.],
       …,
       [0.],
       [0.],
       [0.]])
```

[59]:

```
analysis = train2_avg_diff[['sales']].copy()
```

[38]:

```
analysis.head(365)
```

[38]:

```
                  sales
date
2013-01-01     1.409438
```

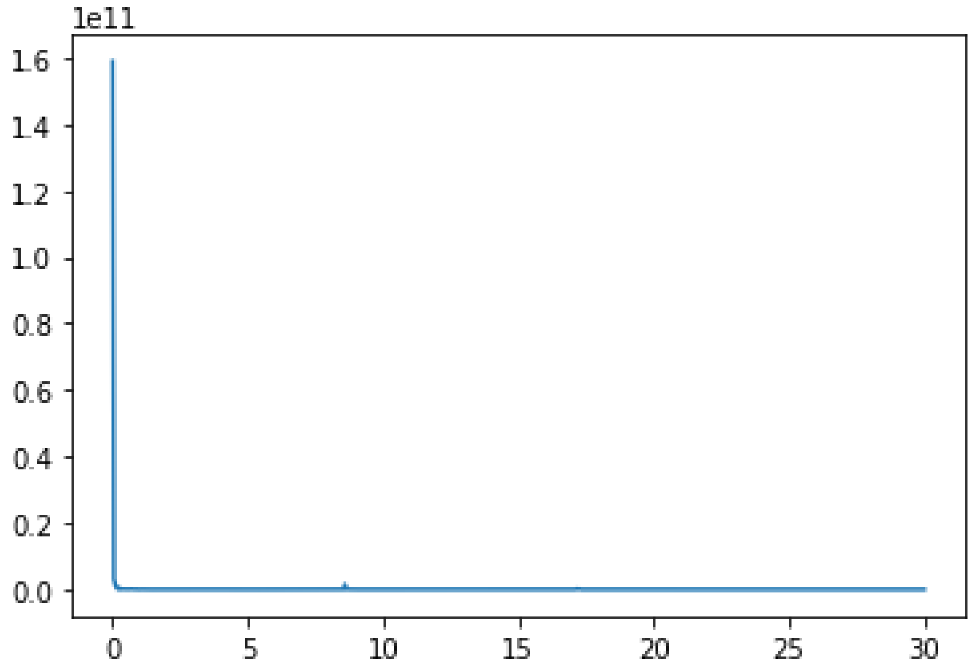
```
2013-01-02  278.390807
2013-01-03  202.840197
2013-01-04  198.911154
2013-01-05  267.873244
…                  …
2013-12-28  312.543382
2013-12-29  280.426209
2013-12-30  356.416799
2013-12-31  284.660305
2014-01-01    4.827197

[365 rows x 1 columns]
```

```
[60]: decompose_result_add = seasonal_decompose(analysis,model="additive",freq=365)
```

```
C:\Users\ndzad\AppData\Local\Temp/ipykernel_14372/3939235887.py:1:
FutureWarning: the 'freq'' keyword is deprecated, use 'period' instead
  decompose_result_add = seasonal_decompose(analysis,model="additive",freq=365)
```

```
[98]: fourier_transform = np.fft.rfft(train2_avg['sales'])

      abs_fourier_transform = np.abs(fourier_transform)

      power_spectrum = np.square(abs_fourier_transform)

      frequency = np.linspace(0, 30, len(power_spectrum))

      plt.plot(frequency, power_spectrum)
```

```
[98]: [<matplotlib.lines.Line2D at 0x16a66f68070>]
```



```
[97]: power_spectrum
```

```
[97]: array([1.59001702e+11, 3.02906674e+09, 1.49353573e+09, 5.18029454e+08,
             1.00869311e+08, 5.89838015e+08, 2.21398735e+07, 8.11218716e+07,
             1.13933789e+08, 8.74962096e+07, 9.66002254e+07, 3.37104385e+07,
             5.83902176e+07, 1.08142605e+08, 1.22123753e+08, 7.54546592e+07,
             1.30685279e+07, 6.27719461e+07, 2.17099013e+07, 1.05621241e+08,
             3.46946163e+06, 4.15632494e+07, 3.87117969e+07, 2.70529026e+07,
             2.15227676e+07, 3.08356684e+06, 1.79380148e+07, 9.30038951e+06,
             6.56590360e+06, 2.95658957e+06, 1.68079513e+06, 3.61347386e+07,
             5.64471074e+06, 7.34638383e+06, 2.42738931e+07, 2.99809416e+04,
             3.74773287e+06, 6.93868227e+06, 3.27015925e+07, 7.65861101e+06,
             1.49680456e+06, 7.76926064e+07, 7.72051266e+06, 5.29216095e+06,
             6.80140231e+06, 5.51919223e+05, 6.78136834e+06, 2.68732768e+05,
             7.11964080e+06, 6.42707396e+06, 6.56755773e+04, 1.31116314e+07,
             2.90434330e+06, 1.36922912e+06, 2.60621140e+05, 1.27779557e+07,
             1.88454342e+06, 3.15555548e+06, 7.61812380e+06, 7.13714438e+06,
             3.97206781e+06, 1.78535909e+07, 3.72979405e+06, 1.88046436e+07,
             3.86725222e+06, 7.69212365e+06, 4.17715565e+06, 3.17131224e+06,
             1.66839938e+06, 1.11652681e+07, 7.09918826e+06, 1.58362391e+06,
             6.80855002e+06, 6.88635439e+06, 2.85031142e+06, 1.26719471e+07,
             3.70858617e+06, 1.28709430e+06, 1.37126504e+07, 1.72970198e+07,
             3.71960246e+06, 3.60701674e+06, 1.47920555e+08, 4.41618952e+06,
             4.79256076e+06, 1.09035539e+07, 4.07634546e+06, 5.03140203e+06,
             2.16764746e+06, 7.38655451e+05, 1.77745025e+06, 4.72308340e+06,
             1.01586120e+05, 7.06212693e+06, 5.01728891e+06, 7.47160999e+06,
             1.58448847e+07, 8.67071186e+06, 1.46704517e+06, 7.54438141e+06,
             2.72506323e+05, 4.58186342e+06, 3.48305561e+06, 5.55014493e+06,
             5.44362771e+06, 4.45017338e+05, 1.11039521e+07, 1.22962303e+05,
             3.55248500e+04, 2.92946071e+06, 1.65136045e+06, 7.81459759e+05,
             6.97745515e+04, 5.53351226e+05, 4.26287139e+05, 1.83042754e+06,
             3.84715423e+05, 7.53164232e+05, 4.81686888e+05, 4.39204352e+06,
             2.39622852e+06, 1.34681999e+06, 1.05659908e+05, 1.89546952e+07,
             3.73127120e+06, 4.29043925e+06, 5.80300912e+06, 7.39542821e+06,
             2.96936582e+06, 5.31218697e+06, 2.15392051e+06, 2.07773096e+06,
             2.13722232e+06, 1.14506462e+05, 5.56905848e+06, 1.51942829e+06,
             4.53810387e+05, 2.62619655e+05, 1.08198296e+06, 4.47888478e+05,
             7.05792564e+05, 6.98171015e+05, 6.38248778e+05, 2.63948674e+06,
             5.23966182e+05, 2.82315714e+06, 3.36330001e+06, 4.51501824e+05,
             2.89295292e+06, 3.12977014e+06, 5.74582936e+06, 1.28619489e+07,
             3.11201301e+05, 1.92085516e+06, 1.35974596e+06, 4.23670819e+06,
             1.43110252e+06, 1.47560478e+06, 1.85266336e+07, 2.94148589e+04,
             1.68263357e+06, 5.92988330e+06, 2.09000950e+06, 6.41881607e+05,
             1.72757512e+07, 3.08886720e+06, 1.52409654e+06, 3.81051568e+06,
             1.33175317e+07, 1.62294835e+06, 2.11420307e+06, 2.01632745e+07,
             2.53057509e+07, 8.81008138e+06, 2.77324378e+07, 5.04957300e+07,
             1.25630492e+07, 3.06026777e+07, 1.51154356e+09, 1.01484848e+08,
             2.51731140e+07, 3.39157976e+07, 2.07259936e+07, 3.09218335e+06,
             2.57899475e+06, 1.79212995e+06, 3.30974799e+05, 7.02463241e+05,
             6.61592104e+06, 4.76851271e+06, 4.99809688e+06, 6.20326859e+06,
             7.10656478e+06, 6.28608410e+05, 1.34279202e+06, 5.32561346e+05,
             3.84360549e+05, 1.17671971e+06, 2.56872513e+06, 1.11606162e+07,
             1.53714642e+06, 1.12939647e+07, 1.25309487e+07, 1.63376316e+06,
             1.32259738e+05, 9.07831063e+05, 2.37863177e+06, 2.86108446e+06,
             1.70884144e+06, 5.51513416e+05, 1.23885900e+06, 6.09458012e+04,
             3.21524506e+06, 9.24063274e+04, 2.01716619e+06, 2.50171516e+06,
             8.25720944e+05, 4.66878758e+04, 5.15549174e+05, 2.37064368e+06,
             1.09603573e+06, 4.39618475e+04, 1.06898266e+07, 1.02003543e+06,
             7.69940991e+05, 5.51637225e+06, 4.41934873e+06, 3.78019488e+06,
             1.53068834e+06, 6.15817052e+06, 2.08137953e+06, 6.08288015e+06,
             4.13242813e+06, 1.26641376e+07, 2.56794274e+06, 1.14968888e+06,
             7.02347215e+05, 1.46744804e+06, 1.38449500e+06, 2.36118355e+06,
             1.96917988e+06, 3.26449894e+05, 6.23368814e+06, 3.33281242e+06,
             4.61610277e+05, 6.32288541e+05, 1.37423602e+06, 4.34990367e+05,
             6.21005546e+04, 2.09023681e+06, 4.78047547e+06, 3.22874256e+05,
             8.06086499e+04, 2.72404830e+06, 1.31453829e+06, 1.31711439e+06,
             4.93557640e+05, 1.63904172e+06, 3.45015791e+06, 2.57382334e+06,
             1.65054099e+06, 3.28865120e+06, 5.44064217e+05, 1.31045994e+07,
             3.03306379e+05, 2.21174729e+06, 1.75907472e+06, 3.22195335e+06,
             1.11656240e+06, 4.56141000e+05, 4.03983629e+06, 2.12104495e+05,
             1.05545264e+06, 5.31358408e+06, 1.79761554e+06, 7.98438396e+05,
             6.17464784e+05, 6.75945865e+06, 5.32381434e+05, 1.36936707e+05,
             4.16789563e+05, 6.00347387e+05, 2.14419942e+04, 4.54338620e+06,
             1.87769675e+06, 2.81107464e+06, 3.82193174e+05, 3.18776867e+05,
             9.64063232e+05, 5.15902685e+05, 4.53955316e+05, 3.96395414e+06,
             3.04105354e+06, 3.68019903e+06, 7.42125600e+06, 1.98211715e+06,
             1.48136724e+06, 1.33066739e+06, 3.89706395e+06, 1.18879378e+06,
             2.17159779e+05, 7.55898838e+04, 2.04791497e+06, 4.21540603e+05,
             5.05842851e+06, 3.06585116e+05, 3.49131296e+05, 5.38082008e+06,
             4.91059700e+06, 1.16654155e+06, 1.42852620e+05, 4.73743140e+06,
             3.61397888e+05, 1.45731748e+02, 4.84147789e+05, 2.45030843e+06,
             1.27011201e+06, 2.25757811e+06, 4.24391081e+06, 4.37409220e+05,
             6.61622614e+05, 3.20000282e+05, 1.29949849e+05, 7.77370181e+04,
             1.93654950e+05, 7.54236831e+06, 8.34184629e+05, 7.97399585e+04,
             2.31656414e+06, 2.60809202e+06, 2.13850198e+05, 3.30339403e+05,
             1.72149685e+06, 5.98975575e+05, 9.18807545e+05, 7.65530957e+06,
             1.58330128e+07, 1.30000386e+06, 4.23380506e+06, 8.74499297e+06,
             2.72197929e+06, 8.41979055e+05, 4.52954602e+04, 1.43588022e+05,
             5.80121232e+04, 4.50405482e+06, 7.60205108e+06, 2.23685556e+06,
             2.95451420e+06, 1.18640258e+07, 2.32946235e+07, 5.22864027e+06,
             1.16438644e+07, 7.54309616e+07, 1.37218072e+07, 2.22082280e+07,
             2.51241608e+08, 1.45295020e+08, 2.99360749e+07, 1.40676904e+07,
             4.26654927e+07, 2.60994693e+06, 4.42035182e+06, 6.27733874e+06,
             2.90326492e+06, 1.11695560e+06, 2.45401075e+06, 4.78121059e+06,
             3.45475219e+06, 5.87958904e+06, 8.77867399e+06, 1.06992521e+05,
             2.07669182e+05, 1.75320260e+06, 2.60261990e+06, 1.73725434e+06,
             8.23603662e+06, 3.20629918e+06, 1.72477221e+06, 4.40710477e+06,
             1.50256777e+05, 1.09551827e+06, 1.18432301e+06, 2.11267028e+06,
             2.46791250e+04, 9.08826527e+05, 2.93714583e+06, 6.15834257e+05,
             2.78652948e+05, 1.53218419e+06, 2.57850916e+06, 1.48002318e+06,
             4.57074103e+05, 3.58287897e+06, 2.12878997e+05, 3.28289910e+05,
             2.05466926e+06, 1.18044150e+06, 3.42442320e+06, 1.15777794e+06,
             6.86475497e+06, 1.22043647e+04, 5.38230935e+05, 5.71595599e+05,
             9.41652219e+05, 1.31492303e+05, 1.19976345e+06, 8.20642899e+06,
             1.50010302e+06, 1.47797979e+06, 3.23917345e+06, 4.64350321e+06,
             3.53474533e+05, 1.45624646e+06, 1.17888805e+06, 5.38815223e+05,
             8.62151311e+04, 4.85939141e+06, 9.70667312e+05, 6.45828396e+04,
             2.64193024e+06, 2.57750510e+06, 1.52321534e+05, 3.30042876e+05,
             1.53371652e+06, 8.50451344e+05, 1.08336671e+05, 2.18694842e+06,
             2.83620562e+06, 4.48609131e+05, 8.77427137e+05, 2.73111832e+06,
             9.83290336e+05, 3.15141862e+05, 2.19906094e+06, 6.05394966e+05,
             2.92750859e+05, 6.18170483e+06, 1.21395055e+05, 8.77233474e+05,
             1.23802809e+05, 5.28085287e+06, 5.02504448e+05, 1.50443663e+05,
             3.76566382e+06, 8.19433514e+05, 5.09639568e+04, 1.10431514e+06,
             3.26773792e+06, 6.92304567e+05, 1.54258516e+06, 1.66370312e+06,
             2.25505098e+06, 1.85970889e+05, 5.64403687e+05, 1.93043369e+04,
             3.78806940e+06, 6.49824964e+05, 6.82135812e+06, 2.72147166e+05,
             6.26237588e+04, 6.25750975e+05, 1.59227097e+06, 2.17604212e+06,
             3.65808843e+05, 4.66496002e+06, 1.51347477e+06, 4.43625860e+05,
             2.75809746e+06, 2.71125554e+06, 1.47578643e+06, 1.87718142e+06,
             1.81977287e+06, 2.40843442e+06, 2.33603406e+05, 3.87546593e+06,
             1.61120008e+06, 1.35558334e+06, 1.31496239e+06, 4.68638862e+05,
             7.97565540e+04, 3.81208257e+05, 1.79616922e+06, 7.85362117e+04,
             3.60047768e+05, 9.16002195e+05, 6.29838029e+05, 4.31064476e+05,
             2.45586465e+06, 2.33188300e+06, 1.55477914e+05, 1.07255421e+05,
             2.09026696e+06, 1.25907110e+05, 8.93212329e+04, 4.16804802e+05,
             5.42400938e+05, 1.70315899e+06, 2.51382673e+04, 3.41780187e+06,
             7.67005699e+05, 8.82985302e+05, 1.54107889e+06, 7.28702894e+05,
             3.18280218e+06, 5.38391399e+05, 3.39047300e+06, 7.29722392e+05,
             1.39247847e+06, 2.24200842e+06, 7.78327734e+05, 1.44580444e+04,
             1.16917343e+06, 2.81960520e+06, 2.34791999e+06, 3.44259596e+05,
             2.38748612e+06, 1.21180400e+06, 9.77704780e+05, 1.33473253e+06,
             9.32421986e+05, 1.78174327e+05, 1.40159966e+05, 1.55101304e+06,
             1.00175759e+06, 1.45002791e+06, 4.48506766e+06, 1.18063934e+05,
             2.48000216e+06, 1.64121872e+06, 8.12106831e+05, 9.37750836e+06,
             9.24297707e+05, 1.97682428e+06, 2.95261322e+06, 4.32843999e+06,
             1.34654033e+06, 4.66564743e+06, 4.73284259e+06, 1.82197375e+05,
             2.55347573e+06, 2.35509627e+05, 2.09880122e+05, 2.24863382e+06,
             9.63992528e+05, 2.78029800e+05, 3.74356805e+04, 2.49425824e+06,
             1.89079544e+05, 8.67642141e+05, 2.72656546e+06, 3.85280931e+05,
             5.44821023e+05, 4.92561852e+05, 2.63745621e+06, 5.68414184e+05,
             6.43709139e+05, 1.89804474e+06, 1.63548423e+06, 1.40435330e+06,
             2.67315014e+06, 1.28755811e+06, 3.12334192e+06, 9.80350014e+05,
             6.20572349e+06, 5.84437130e+05, 2.40590301e+05, 5.02289867e+06,
             1.62515586e+06, 7.65140571e+05, 9.16267955e+05, 4.80694409e+06,
             6.70391713e+05, 6.78243355e+05, 3.03530215e+06, 1.36365537e+06,
             1.69964312e+06, 3.50219345e+05, 8.82622367e+05, 1.65201656e+06,
             1.60200993e+05, 5.02857573e+06, 2.42053632e+05, 1.16472851e+05,
             1.97208867e+06, 1.70401093e+05, 3.08545145e+05, 7.03839470e+05,
             3.87398842e+06, 1.54963747e+06, 5.66769691e+05, 1.05344195e+06,
             4.23638864e+04, 2.72389260e+05, 7.46946395e+05, 2.56728552e+06,
             4.53662176e+05, 6.09584207e+05, 2.05995797e+06, 5.46923309e+05,
             1.21137242e+05, 1.48262728e+05, 5.34029049e+05, 7.56146567e+05,
             3.88916156e+05, 4.23916825e+06, 1.70263162e+06, 5.88294907e+05,
             2.21103373e+06, 6.14201988e+05, 1.05505484e+06, 3.38056265e+05,
             3.77348215e+06, 1.39127386e+06, 1.50576788e+06, 1.77485838e+06,
             1.40819166e+06, 1.77289250e+06, 2.24920804e+05, 1.41027291e+06,
             1.00628854e+06, 1.17508837e+06, 2.88886079e+06])
```

modelling_v2

April 21, 2022

```
[40]: import numpy as np
      import pandas as pd
      import matplotlib.pyplot as plt
      import statsmodels
      import statsmodels.api as sm
      import statsmodels.formula.api as smf
      import datetime as dt
      import scipy.signal as ss
```

```
[41]: from sklearn.metrics import mean_squared_error
```

```
[42]: from sklearn.preprocessing import OneHotEncoder
```

```
[43]: holidays_events = pd.read_csv("https://www.dropbox.com/s/bxyamlpevkiwwoq/
       ↪holidays_events.csv?dl=1")
      oil = pd.read_csv("https://www.dropbox.com/s/l6ln0ztl4m0pw3a/oil.csv?
       ↪dl=1",parse_dates=['date'],index_col='date')
      oil2 = pd.read_csv("https://www.dropbox.com/s/l6ln0ztl4m0pw3a/oil.csv?dl=1")
      sample_submission = pd.read_csv("https://www.dropbox.com/s/68jjl61x6u3klos/
       ↪sample_submission.csv?dl=1")
      stores = pd.read_csv("https://www.dropbox.com/s/lcxn6r9bs2exguq/stores.csv?
       ↪dl=1")
      test = pd.read_csv("https://www.dropbox.com/s/cvdo1gn7r5lu2uz/test.csv?
       ↪dl=1",index_col='id')
      train = pd.read_csv("https://www.dropbox.com/s/s8p2b5awnuqfk0d/train.csv?
       ↪dl=1",index_col='id')
      transactions = pd.read_csv("https://www.dropbox.com/s/92fij9bcwt0e0cj/
       ↪transactions.csv?dl=1")
```

Wybieramy obserwacje dla family=AUTOMOTIVE

```
[44]: train_automotive = train.loc[(train['family']=='AUTOMOTIVE')]
```

```
[45]: train_automotive.tail()
```

```
[45]:                  date  store_nbr      family  sales  onpromotion
      id
```

```
3000723  2017-08-15          54  AUTOMOTIVE    8.0            0
3000756  2017-08-15           6  AUTOMOTIVE    7.0            0
3000789  2017-08-15           7  AUTOMOTIVE    5.0            0
3000822  2017-08-15           8  AUTOMOTIVE    4.0            0
3000855  2017-08-15           9  AUTOMOTIVE   15.0            0
```

Wyliczamy średnią sprzedaż na daną datę

[46]:
```
train_automotive2= train_automotive.groupby(['date'])['sales'].mean().to_frame()
```

[47]:
```
train_automotive2.head()
```

[47]:
```
               sales
date
2013-01-01  0.000000
2013-01-02  4.722222
2013-01-03  2.981481
2013-01-04  3.129630
2013-01-05  6.333333
```

Dołączamy informacje o zmiennych i robimy one-hot encoding zmiennej locale

[48]:
```
train_automotive_merged = train_automotive2.
 ↪merge(holidays_events,how="left",left_on=['date'],right_on=['date'])

encoder = OneHotEncoder(handle_unknown='ignore')

encoder_df = pd.DataFrame(encoder.
 ↪fit_transform(train_automotive_merged[['locale']]).toarray())

final_train_automotive = train_automotive_merged.join(encoder_df)

final_train_automotive.drop('locale', axis=1, inplace=True)

final_train_automotive.columns = ['date',␣
 ↪'sales','type','local_name','description','transferred','isLocal','isNational','isRegional'
```

[49]:
```
final_train_automotive.head()
```

[49]:
```
         date     sales      type local_name              description  \
0  2013-01-01  0.000000   Holiday    Ecuador       Primer dia del ano
1  2013-01-02  4.722222       NaN        NaN                      NaN
2  2013-01-03  2.981481       NaN        NaN                      NaN
3  2013-01-04  3.129630       NaN        NaN                      NaN
4  2013-01-05  6.333333  Work Day    Ecuador  Recupero puente Navidad

  transferred  isLocal  isNational  isRegional  isNormalDay
```

```
0       False      0.0        1.0        0.0          0.0
1         NaN      0.0        0.0        0.0          1.0
2         NaN      0.0        0.0        0.0          1.0
3         NaN      0.0        0.0        0.0          1.0
4       False      0.0        1.0        0.0          0.0
```

Tworzymy zmienną dayofweek

```
[50]: final_train_automotive['dayofweek'] = pd.
       ↪DatetimeIndex(final_train_automotive['date']).dayofweek + 1
```

```
[51]: final_train_automotive.head()
```

```
[51]:          date     sales      type local_name               description  \
      0  2013-01-01  0.000000   Holiday    Ecuador       Primer dia del ano
      1  2013-01-02  4.722222       NaN        NaN                      NaN
      2  2013-01-03  2.981481       NaN        NaN                      NaN
      3  2013-01-04  3.129630       NaN        NaN                      NaN
      4  2013-01-05  6.333333  Work Day    Ecuador  Recupero puente Navidad

        transferred  isLocal  isNational  isRegional  isNormalDay  dayofweek
      0       False      0.0         1.0         0.0          0.0          2
      1         NaN      0.0         0.0         0.0          1.0          3
      2         NaN      0.0         0.0         0.0          1.0          4
      3         NaN      0.0         0.0         0.0          1.0          5
      4       False      0.0         1.0         0.0          0.0          6
```

Dodajemy oil jako zmienną objaśniającą.

```
[52]: train_automotive_oil = final_train_automotive.
       ↪merge(oil2,how="left",left_on=['date'],right_on=['date'])
```

```
[53]: train_automotive_oil.head()
```

```
[53]:          date     sales      type local_name               description  \
      0  2013-01-01  0.000000   Holiday    Ecuador       Primer dia del ano
      1  2013-01-02  4.722222       NaN        NaN                      NaN
      2  2013-01-03  2.981481       NaN        NaN                      NaN
      3  2013-01-04  3.129630       NaN        NaN                      NaN
      4  2013-01-05  6.333333  Work Day    Ecuador  Recupero puente Navidad

        transferred  isLocal  isNational  isRegional  isNormalDay  dayofweek  \
      0       False      0.0         1.0         0.0          0.0          2
      1         NaN      0.0         0.0         0.0          1.0          3
      2         NaN      0.0         0.0         0.0          1.0          4
      3         NaN      0.0         0.0         0.0          1.0          5
      4       False      0.0         1.0         0.0          0.0          6
```

```
   dcoilwtico
0         NaN
1       93.14
2       92.97
3       93.12
4         NaN
```

Interpolacja

```
[54]: train_automotive_oil.interpolate(method ='linear', limit_direction ='backward',␣
       ↪inplace=True)
```

```
[55]: train_automotive_oil.tail(10)
```

```
[55]:            date      sales      type local_name  \
      1704  2017-08-06  10.796296       NaN        NaN
      1705  2017-08-07   6.574074       NaN        NaN
      1706  2017-08-08   6.055556       NaN        NaN
      1707  2017-08-09   5.814815       NaN        NaN
      1708  2017-08-10   5.796296   Holiday    Ecuador
      1709  2017-08-11   8.166667  Transfer    Ecuador
      1710  2017-08-12   7.462963       NaN        NaN
      1711  2017-08-13   8.907407       NaN        NaN
      1712  2017-08-14   5.407407       NaN        NaN
      1713  2017-08-15   6.240741   Holiday   Riobamba

                                  description transferred  isLocal  isNational  \
      1704                                NaN         NaN      0.0         0.0
      1705                                NaN         NaN      0.0         0.0
      1706                                NaN         NaN      0.0         0.0
      1707                                NaN         NaN      0.0         0.0
      1708          Primer Grito de Independencia        True      0.0         1.0
      1709 Traslado Primer Grito de Independencia       False      0.0         1.0
      1710                                NaN         NaN      0.0         0.0
      1711                                NaN         NaN      0.0         0.0
      1712                                NaN         NaN      0.0         0.0
      1713                 Fundacion de Riobamba       False      1.0         0.0

            isRegional  isNormalDay  dayofweek  dcoilwtico
      1704         0.0          1.0          7   49.436667
      1705         0.0          1.0          1   49.370000
      1706         0.0          1.0          2   49.070000
      1707         0.0          1.0          3   49.590000
      1708         0.0          0.0          4   48.540000
      1709         0.0          0.0          5   48.810000
      1710         0.0          1.0          6   48.403333
      1711         0.0          1.0          7   47.996667
```

```
1712          0.0          1.0          1   47.590000
1713          0.0          0.0          2   47.570000
```

Dzielimy próbkę train na treningową i testową.

```
[56]: train2 = train_automotive_oil.loc[(train_automotive_oil['date']<'2016-06-01')]
      test2 = train_automotive_oil.loc[(train_automotive_oil['date']>='2016-06-01')].
       ↪reset_index(drop=True)
```

```
[57]: test2.head()
```

```
[57]:         date      sales type local_name description transferred  isLocal  \
      0  2016-06-01   6.425926  NaN        NaN         NaN         NaN      0.0
      1  2016-06-02   5.740741  NaN        NaN         NaN         NaN      0.0
      2  2016-06-03   5.888889  NaN        NaN         NaN         NaN      0.0
      3  2016-06-04   9.000000  NaN        NaN         NaN         NaN      0.0
      4  2016-06-05  11.185185  NaN        NaN         NaN         NaN      0.0

         isNational  isRegional  isNormalDay  dayofweek  dcoilwtico
      0         0.0         0.0          1.0          3       49.07
      1         0.0         0.0          1.0          4       49.14
      2         0.0         0.0          1.0          5       48.69
      3         0.0         0.0          1.0          6       49.03
      4         0.0         0.0          1.0          7       49.37
```

```
[58]: train2_date = train2.copy()
      train2.
       ↪drop(["description","date","type","local_name","transferred"],axis=1,inplace=True)
      test2.
       ↪drop(["description","date","type","local_name","transferred"],axis=1,inplace=True)
```

```
C:\Users\Lenovo\AppData\Local\Temp\ipykernel_15988\2658594707.py:2:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  train2.drop(["description","date","type","local_name","transferred"],axis=1,in
place=True)
```

Tworzymy model liniowy

```
[59]: X = train2.drop(['sales'],axis=1)
      Y = train2['sales']
```

```
[60]: model = sm.OLS(Y,X).fit()
```

```
[61]: print(model.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  sales   R-squared:                       0.487
Model:                            OLS   Adj. R-squared:                  0.485
Method:                 Least Squares   F-statistic:                     239.3
Date:                Thu, 21 Apr 2022   Prob (F-statistic):          1.08e-179
Time:                        21:26:18   Log-Likelihood:                -2206.9
No. Observations:                1264   AIC:                             4426.
Df Residuals:                    1258   BIC:                             4457.
Df Model:                           5
Covariance Type:            nonrobust
===============================================================================
                  coef    std err          t      P>|t|      [0.025      0.975]
-------------------------------------------------------------------------------
isLocal         6.2705      0.206     30.386      0.000       5.866       6.675
isNational      6.7287      0.181     37.276      0.000       6.375       7.083
isRegional      6.2289      0.407     15.288      0.000       5.430       7.028
isNormalDay     6.1840      0.143     43.316      0.000       5.904       6.464
dayofweek       0.5017      0.020     25.632      0.000       0.463       0.540
dcoilwtico     -0.0329      0.001    -22.276      0.000      -0.036      -0.030
==============================================================================
Omnibus:                       65.606   Durbin-Watson:                   1.098
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              209.515
Skew:                           0.148   Prob(JB):                     3.19e-46
Kurtosis:                       4.972   Cond. No.                         849.
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

```
[62]: test2.head()
```

```
[62]:        sales  isLocal  isNational  isRegional  isNormalDay  dayofweek  \
      0   6.425926      0.0         0.0         0.0          1.0          3
      1   5.740741      0.0         0.0         0.0          1.0          4
      2   5.888889      0.0         0.0         0.0          1.0          5
      3   9.000000      0.0         0.0         0.0          1.0          6
      4  11.185185      0.0         0.0         0.0          1.0          7

         dcoilwtico
      0       49.07
      1       49.14
      2       48.69
      3       49.03
      4       49.37
```

```
[63]: train2.head()
```

```
[63]:        sales  isLocal  isNational  isRegional  isNormalDay  dayofweek  \
      0  0.000000      0.0         1.0         0.0          0.0          2
      1  4.722222      0.0         0.0         0.0          1.0          3
      2  2.981481      0.0         0.0         0.0          1.0          4
      3  3.129630      0.0         0.0         0.0          1.0          5
      4  6.333333      0.0         1.0         0.0          0.0          6

         dcoilwtico
      0   93.140000
      1   93.140000
      2   92.970000
      3   93.120000
      4   93.146667
```

```
[64]: test2_drop = test2.drop(['sales'],axis=1)
      Y_test = test2['sales']
```

```
[65]: test2.head()
```

```
[65]:         sales  isLocal  isNational  isRegional  isNormalDay  dayofweek  \
      0   6.425926      0.0         0.0         0.0          1.0          3
      1   5.740741      0.0         0.0         0.0          1.0          4
      2   5.888889      0.0         0.0         0.0          1.0          5
      3   9.000000      0.0         0.0         0.0          1.0          6
      4  11.185185      0.0         0.0         0.0          1.0          7

         dcoilwtico
      0       49.07
      1       49.14
      2       48.69
      3       49.03
      4       49.37
```

```
[66]: Y_pred = model.predict(test2_drop)
```

Policzmy MSE.

```
[67]: mean_squared_error(Y_test,Y_pred)
```

```
[67]: 2.775696197477054
```