In [1]:

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from scipy.stats.stats import pearsonr
from scipy.stats import linregress
%matplotlib inline

df_tweets = pd.read_excel(r'C:\Users\timod\Desktop\SPX_tweets_all.xlsx')
df_price = pd.read_csv(r'C:\Users\timod\Desktop\SPX_historical data_all.csv')
```

In [2]:

```
#Gathering the following for each of the first seven days of february:
#number of positive tweets, number of negative tweets, number of neutral tweets, total volume, sen
timent polarity

'''
pos0331 = df_tweets.loc[df_tweets.date=='2020-03-31'][df_tweets.polarity > 0].shape[0]
neg0331 = df_tweets.loc[df_tweets.date=='2020-03-31'][df_tweets.polarity < 0].shape[0]
neutral0331 = df_tweets.loc[df_tweets.date=='2020-03-31'][df_tweets.polarity == 0].shape[0]
nonZero0331 = pos0331 + neg0331
px0331 = round(((pos0331-neg0331)/nonZero0331),3)
volume0331 = nonZero0331 + neutral0331

pos0330 = df_tweets.loc[df_tweets.date=='2020-03-30'][df_tweets.polarity > 0].shape[0]
neg0330 = df_tweets.loc[df_tweets.date=='2020-03-30'][df_tweets.polarity < 0].shape[0]
neutral0330 = df_tweets.loc[df_tweets.date=='2020-03-30'][df_tweets.polarity == 0].shape[0]
nonZero0330 = pos0330 + neg0330
px0330 = round(((pos0330-neg0330)/nonZero0330),3)
volume0330 = nonZero0330 + neutral0330

pos0329 = df_tweets.loc[df_tweets.date=='2020-03-29'][df_tweets.polarity > 0].shape[0]
neg0329 = df_tweets.loc[df_tweets.date=='2020-03-29'][df_tweets.polarity < 0].shape[0]
neutral0329 = df_tweets.loc[df_tweets.date=='2020-03-29'][df_tweets.polarity == 0].shape[0]
nonZero0329 = pos0329 + neg0329
px0329 = round(((pos0329-neg0329)/nonZero0329),3)
volume0329 = nonZero0329 + neutral0329

pos0328 = df_tweets.loc[df_tweets.date=='2020-03-28'][df_tweets.polarity > 0].shape[0]
neg0328 = df_tweets.loc[df_tweets.date=='2020-03-28'][df_tweets.polarity < 0].shape[0]
neutral0328 = df_tweets.loc[df_tweets.date=='2020-03-28'][df_tweets.polarity == 0].shape[0]
nonZero0328 = pos0328 + neg0328
px0328 = round(((pos0328-neg0328)/nonZero0328),3)
volume0328 = nonZero0328 + neutral0328
'''

pos0327 = df_tweets.loc[df_tweets.date=='2020-03-27'][df_tweets.polarity > 0].shape[0]
neg0327 = df_tweets.loc[df_tweets.date=='2020-03-27'][df_tweets.polarity < 0].shape[0]
neutral0327 = df_tweets.loc[df_tweets.date=='2020-03-27'][df_tweets.polarity == 0].shape[0]
nonZero0327 = pos0327 + neg0327
px0327 = round(((pos0327-neg0327)/nonZero0327),3)
volume0327 = nonZero0327 + neutral0327

pos0326 = df_tweets.loc[df_tweets.date=='2020-03-26'][df_tweets.polarity > 0].shape[0]
neg0326 = df_tweets.loc[df_tweets.date=='2020-03-26'][df_tweets.polarity < 0].shape[0]
neutral0326 = df_tweets.loc[df_tweets.date=='2020-03-26'][df_tweets.polarity == 0].shape[0]
nonZero0326 = pos0326 + neg0326
px0326 = round(((pos0326-neg0326)/nonZero0326),3)
volume0326 = nonZero0326 + neutral0326

pos0325 = df_tweets.loc[df_tweets.date=='2020-03-25'][df_tweets.polarity > 0].shape[0]
neg0325 = df_tweets.loc[df_tweets.date=='2020-03-25'][df_tweets.polarity < 0].shape[0]
neutral0325 = df_tweets.loc[df_tweets.date=='2020-03-25'][df_tweets.polarity == 0].shape[0]
nonZero0325 = pos0325 + neg0325
px0325 = round(((pos0325-neg0325)/nonZero0325),3)
volume0325 = nonZero0325 + neutral0325

pos0324 = df_tweets.loc[df_tweets.date=='2020-03-24'][df_tweets.polarity > 0].shape[0]
neg0324 = df_tweets.loc[df_tweets.date=='2020-03-24'][df_tweets.polarity < 0].shape[0]
neutral0324 = df_tweets.loc[df_tweets.date=='2020-03-24'][df_tweets.polarity == 0].shape[0]
```

```
nonZero0324 = pos0324 + neg0324
px0324 = round(((pos0324-neg0324)/nonZero0324),3)
volume0324 = nonZero0324 + neutral0324

pos0323 = df_tweets.loc[df_tweets.date=='2020-03-23'][df_tweets.polarity > 0].shape[0]
neg0323 = df_tweets.loc[df_tweets.date=='2020-03-23'][df_tweets.polarity < 0].shape[0]
neutral0323 = df_tweets.loc[df_tweets.date=='2020-03-23'][df_tweets.polarity == 0].shape[0]
nonZero0323 = pos0323 + neg0323
px0323 = round(((pos0323-neg0323)/nonZero0323),3)
volume0323 = nonZero0323 + neutral0323

pos0322 = df_tweets.loc[df_tweets.date=='2020-03-22'][df_tweets.polarity > 0].shape[0]
neg0322 = df_tweets.loc[df_tweets.date=='2020-03-22'][df_tweets.polarity < 0].shape[0]
neutral0322 = df_tweets.loc[df_tweets.date=='2020-03-22'][df_tweets.polarity == 0].shape[0]
nonZero0322 = pos0322 + neg0322
px0322 = round(((pos0322-neg0322)/nonZero0322),3)
volume0322 = nonZero0322 + neutral0322

pos0321 = df_tweets.loc[df_tweets.date=='2020-03-21'][df_tweets.polarity > 0].shape[0]
neg0321 = df_tweets.loc[df_tweets.date=='2020-03-21'][df_tweets.polarity < 0].shape[0]
neutral0321 = df_tweets.loc[df_tweets.date=='2020-03-21'][df_tweets.polarity == 0].shape[0]
nonZero0321 = pos0321 + neg0321
px0321 = round(((pos0321-neg0321)/nonZero0321),3)
volume0321 = nonZero0321 + neutral0321

pos0320 = df_tweets.loc[df_tweets.date=='2020-03-20'][df_tweets.polarity > 0].shape[0]
neg0320 = df_tweets.loc[df_tweets.date=='2020-03-20'][df_tweets.polarity < 0].shape[0]
neutral0320 = df_tweets.loc[df_tweets.date=='2020-03-20'][df_tweets.polarity == 0].shape[0]
nonZero0320 = pos0320 + neg0320
px0320 = round(((pos0320-neg0320)/nonZero0320),3)
volume0320 = nonZero0320 + neutral0320

pos0319 = df_tweets.loc[df_tweets.date=='2020-03-19'][df_tweets.polarity > 0].shape[0]
neg0319 = df_tweets.loc[df_tweets.date=='2020-03-19'][df_tweets.polarity < 0].shape[0]
neutral0319 = df_tweets.loc[df_tweets.date=='2020-03-19'][df_tweets.polarity == 0].shape[0]
nonZero0319 = pos0319 + neg0319
px0319 = round(((pos0319-neg0319)/nonZero0319),3)
volume0319 = nonZero0319 + neutral0319

pos0318 = df_tweets.loc[df_tweets.date=='2020-03-18'][df_tweets.polarity > 0].shape[0]
neg0318 = df_tweets.loc[df_tweets.date=='2020-03-18'][df_tweets.polarity < 0].shape[0]
neutral0318 = df_tweets.loc[df_tweets.date=='2020-03-18'][df_tweets.polarity == 0].shape[0]
nonZero0318 = pos0318 + neg0318
px0318 = round(((pos0318-neg0318)/nonZero0318),3)
volume0318 = nonZero0318 + neutral0318

pos0317 = df_tweets.loc[df_tweets.date=='2020-03-17'][df_tweets.polarity > 0].shape[0]
neg0317 = df_tweets.loc[df_tweets.date=='2020-03-17'][df_tweets.polarity < 0].shape[0]
neutral0317 = df_tweets.loc[df_tweets.date=='2020-03-17'][df_tweets.polarity == 0].shape[0]
nonZero0317 = pos0317 + neg0317
px0317 = round(((pos0317-neg0317)/nonZero0317),3)
volume0317 = nonZero0317 + neutral0317

pos0316 = df_tweets.loc[df_tweets.date=='2020-03-16'][df_tweets.polarity > 0].shape[0]
neg0316 = df_tweets.loc[df_tweets.date=='2020-03-16'][df_tweets.polarity < 0].shape[0]
neutral0316 = df_tweets.loc[df_tweets.date=='2020-03-16'][df_tweets.polarity == 0].shape[0]
nonZero0316 = pos0316 + neg0316
px0316 = round(((pos0316-neg0316)/nonZero0316),3)
volume0316 = nonZero0316 + neutral0316

pos0315 = df_tweets.loc[df_tweets.date=='2020-03-15'][df_tweets.polarity > 0].shape[0]
neg0315 = df_tweets.loc[df_tweets.date=='2020-03-15'][df_tweets.polarity < 0].shape[0]
neutral0315 = df_tweets.loc[df_tweets.date=='2020-03-15'][df_tweets.polarity == 0].shape[0]
nonZero0315 = pos0315 + neg0315
px0315 = round(((pos0315-neg0315)/nonZero0315),3)
volume0315 = nonZero0315 + neutral0315

pos0314 = df_tweets.loc[df_tweets.date=='2020-03-14'][df_tweets.polarity > 0].shape[0]
neg0314 = df_tweets.loc[df_tweets.date=='2020-03-14'][df_tweets.polarity < 0].shape[0]
neutral0314 = df_tweets.loc[df_tweets.date=='2020-03-14'][df_tweets.polarity == 0].shape[0]
nonZero0314 = pos0314 + neg0314
px0314 = round(((pos0314-neg0314)/nonZero0314),3)
volume0314 = nonZero0314 + neutral0314

pos0313 = df_tweets.loc[df_tweets.date=='2020-03-13'][df_tweets.polarity > 0].shape[0]
neg0313 = df_tweets.loc[df_tweets.date=='2020-03-13'][df_tweets.polarity < 0].shape[0]
neutral0313 = df_tweets.loc[df_tweets.date=='2020-03-13'][df_tweets.polarity == 0].shape[0]
```

```
nonZero0313 = pos0313 + neg0313
px0313 = round(((pos0313-neg0313)/nonZero0313),3)
volume0313 = nonZero0313 + neutral0313

pos0312 = df_tweets.loc[df_tweets.date=='2020-03-12'][df_tweets.polarity > 0].shape[0]
neg0312 = df_tweets.loc[df_tweets.date=='2020-03-12'][df_tweets.polarity < 0].shape[0]
neutral0312 = df_tweets.loc[df_tweets.date=='2020-03-12'][df_tweets.polarity == 0].shape[0]
nonZero0312 = pos0312 + neg0312
px0312 = round(((pos0312-neg0312)/nonZero0312),3)
volume0312 = nonZero0312 + neutral0312

pos0311 = df_tweets.loc[df_tweets.date=='2020-03-11'][df_tweets.polarity > 0].shape[0]
neg0311 = df_tweets.loc[df_tweets.date=='2020-03-11'][df_tweets.polarity < 0].shape[0]
neutral0311 = df_tweets.loc[df_tweets.date=='2020-03-11'][df_tweets.polarity == 0].shape[0]
nonZero0311 = pos0311 + neg0311
px0311 = round(((pos0311-neg0311)/nonZero0311),3)
volume0311 = nonZero0311 + neutral0311

pos0310 = df_tweets.loc[df_tweets.date=='2020-03-10'][df_tweets.polarity > 0].shape[0]
neg0310 = df_tweets.loc[df_tweets.date=='2020-03-10'][df_tweets.polarity < 0].shape[0]
neutral0310 = df_tweets.loc[df_tweets.date=='2020-03-10'][df_tweets.polarity == 0].shape[0]
nonZero0310 = pos0310 + neg0310
px0310 = round(((pos0310-neg0310)/nonZero0310),3)
volume0310 = nonZero0310 + neutral0310

pos0309 = df_tweets.loc[df_tweets.date=='2020-03-09'][df_tweets.polarity > 0].shape[0]
neg0309 = df_tweets.loc[df_tweets.date=='2020-03-09'][df_tweets.polarity < 0].shape[0]
neutral0309 = df_tweets.loc[df_tweets.date=='2020-03-09'][df_tweets.polarity == 0].shape[0]
nonZero0309 = pos0309 + neg0309
px0309 = round(((pos0309-neg0309)/nonZero0309),3)
volume0309 = nonZero0309 + neutral0309

pos0308 = df_tweets.loc[df_tweets.date=='2020-03-08'][df_tweets.polarity > 0].shape[0]
neg0308 = df_tweets.loc[df_tweets.date=='2020-03-08'][df_tweets.polarity < 0].shape[0]
neutral0308 = df_tweets.loc[df_tweets.date=='2020-03-08'][df_tweets.polarity == 0].shape[0]
nonZero0308 = pos0308 + neg0308
px0308 = round(((pos0308-neg0308)/nonZero0308),3)
volume0308 = nonZero0308 + neutral0308

pos0307 = df_tweets.loc[df_tweets.date=='2020-03-07'][df_tweets.polarity > 0].shape[0]
neg0307 = df_tweets.loc[df_tweets.date=='2020-03-07'][df_tweets.polarity < 0].shape[0]
neutral0307 = df_tweets.loc[df_tweets.date=='2020-03-07'][df_tweets.polarity == 0].shape[0]
nonZero0307 = pos0307 + neg0307
px0307 = round(((pos0307-neg0307)/nonZero0307),3)
volume0307 = nonZero0307 + neutral0307

pos0306 = df_tweets.loc[df_tweets.date=='2020-03-06'][df_tweets.polarity > 0].shape[0]
neg0306 = df_tweets.loc[df_tweets.date=='2020-03-06'][df_tweets.polarity < 0].shape[0]
neutral0306 = df_tweets.loc[df_tweets.date=='2020-03-06'][df_tweets.polarity == 0].shape[0]
nonZero0306 = pos0306 + neg0306
px0306 = round(((pos0306-neg0306)/nonZero0306),3)
volume0306 = nonZero0306 + neutral0306

pos0305 = df_tweets.loc[df_tweets.date=='2020-03-05'][df_tweets.polarity > 0].shape[0]
neg0305 = df_tweets.loc[df_tweets.date=='2020-03-05'][df_tweets.polarity < 0].shape[0]
neutral0305 = df_tweets.loc[df_tweets.date=='2020-03-05'][df_tweets.polarity == 0].shape[0]
nonZero0305 = pos0305 + neg0305
px0305 = round(((pos0305-neg0305)/nonZero0305),3)
volume0305 = nonZero0305 + neutral0305

pos0304 = df_tweets.loc[df_tweets.date=='2020-03-04'][df_tweets.polarity > 0].shape[0]
neg0304 = df_tweets.loc[df_tweets.date=='2020-03-04'][df_tweets.polarity < 0].shape[0]
neutral0304 = df_tweets.loc[df_tweets.date=='2020-03-04'][df_tweets.polarity == 0].shape[0]
nonZero0304 = pos0304 + neg0304
px0304 = round(((pos0304-neg0304)/nonZero0304),3)
volume0304 = nonZero0304 + neutral0304

pos0303 = df_tweets.loc[df_tweets.date=='2020-03-03'][df_tweets.polarity > 0].shape[0]
neg0303 = df_tweets.loc[df_tweets.date=='2020-03-03'][df_tweets.polarity < 0].shape[0]
neutral0303 = df_tweets.loc[df_tweets.date=='2020-03-03'][df_tweets.polarity == 0].shape[0]
nonZero0303 = pos0303 + neg0303
px0303 = round(((pos0303-neg0303)/nonZero0303),3)
volume0303 = nonZero0303 + neutral0303

pos0302 = df_tweets.loc[df_tweets.date=='2020-03-02'][df_tweets.polarity > 0].shape[0]
neg0302 = df_tweets.loc[df_tweets.date=='2020-03-02'][df_tweets.polarity < 0].shape[0]
neutral0302 = df_tweets.loc[df_tweets.date=='2020-03-02'][df_tweets.polarity == 0].shape[0]
```

```
nonZero0302 = pos0302 + neg0302
px0302 = round(((pos0302-neg0302)/nonZero0302),3)
volume0302 = nonZero0302 + neutral0302

pos0301 = df_tweets.loc[df_tweets.date=='2020-03-01'][df_tweets.polarity > 0].shape[0]
neg0301 = df_tweets.loc[df_tweets.date=='2020-03-01'][df_tweets.polarity < 0].shape[0]
neutral0301 = df_tweets.loc[df_tweets.date=='2020-03-01'][df_tweets.polarity == 0].shape[0]
nonZero0301 = pos0301 + neg0301
px0301 = round(((pos0301-neg0301)/nonZero0301),3)
volume0301 = nonZero0301 + neutral0301

pos0229 = df_tweets.loc[df_tweets.date=='2020-02-29'][df_tweets.polarity > 0].shape[0]
neg0229 = df_tweets.loc[df_tweets.date=='2020-02-29'][df_tweets.polarity < 0].shape[0]
neutral0229 = df_tweets.loc[df_tweets.date=='2020-02-29'][df_tweets.polarity == 0].shape[0]
nonZero0229 = pos0229 + neg0229
px0229 = round(((pos0229-neg0229)/nonZero0229),3)
volume0229 = nonZero0229 + neutral0229

pos0228 = df_tweets.loc[df_tweets.date=='2020-02-28'][df_tweets.polarity > 0].shape[0]
neg0228 = df_tweets.loc[df_tweets.date=='2020-02-28'][df_tweets.polarity < 0].shape[0]
neutral0228 = df_tweets.loc[df_tweets.date=='2020-02-28'][df_tweets.polarity == 0].shape[0]
nonZero0228 = pos0228 + neg0228
px0228 = round(((pos0228-neg0228)/nonZero0228),3)
volume0228 = nonZero0228 + neutral0228

pos0227 = df_tweets.loc[df_tweets.date=='2020-02-27'][df_tweets.polarity > 0].shape[0]
neg0227 = df_tweets.loc[df_tweets.date=='2020-02-27'][df_tweets.polarity < 0].shape[0]
neutral0227 = df_tweets.loc[df_tweets.date=='2020-02-27'][df_tweets.polarity == 0].shape[0]
nonZero0227 = pos0227 + neg0227
px0227 = round(((pos0227-neg0227)/nonZero0227),3)
volume0227 = nonZero0227 + neutral0227

pos0226 = df_tweets.loc[df_tweets.date=='2020-02-26'][df_tweets.polarity > 0].shape[0]
neg0226 = df_tweets.loc[df_tweets.date=='2020-02-26'][df_tweets.polarity < 0].shape[0]
neutral0226 = df_tweets.loc[df_tweets.date=='2020-02-26'][df_tweets.polarity == 0].shape[0]
nonZero0226 = pos0226 + neg0226
px0226 = round(((pos0226-neg0226)/nonZero0226),3)
volume0226 = nonZero0226 + neutral0226

pos0225 = df_tweets.loc[df_tweets.date=='2020-02-25'][df_tweets.polarity > 0].shape[0]
neg0225 = df_tweets.loc[df_tweets.date=='2020-02-25'][df_tweets.polarity < 0].shape[0]
neutral0225 = df_tweets.loc[df_tweets.date=='2020-02-25'][df_tweets.polarity == 0].shape[0]
nonZero0225 = pos0225 + neg0225
px0225 = round(((pos0225-neg0225)/nonZero0225),3)
volume0225 = nonZero0225 + neutral0225

pos0224 = df_tweets.loc[df_tweets.date=='2020-02-24'][df_tweets.polarity > 0].shape[0]
neg0224 = df_tweets.loc[df_tweets.date=='2020-02-24'][df_tweets.polarity < 0].shape[0]
neutral0224 = df_tweets.loc[df_tweets.date=='2020-02-24'][df_tweets.polarity == 0].shape[0]
nonZero0224 = pos0224 + neg0224
px0224 = round(((pos0224-neg0224)/nonZero0224),3)
volume0224 = nonZero0224 + neutral0224


pos0223 = df_tweets.loc[df_tweets.date=='2020-02-23'][df_tweets.polarity > 0].shape[0]
neg0223 = df_tweets.loc[df_tweets.date=='2020-02-23'][df_tweets.polarity < 0].shape[0]
neutral0223 = df_tweets.loc[df_tweets.date=='2020-02-23'][df_tweets.polarity == 0].shape[0]
nonZero0223 = pos0223 + neg0223
px0223 = round(((pos0223-neg0223)/nonZero0223),3)
volume0223 = nonZero0223 + neutral0223

pos0222 = df_tweets.loc[df_tweets.date=='2020-02-22'][df_tweets.polarity > 0].shape[0]
neg0222 = df_tweets.loc[df_tweets.date=='2020-02-22'][df_tweets.polarity < 0].shape[0]
neutral0222 = df_tweets.loc[df_tweets.date=='2020-02-22'][df_tweets.polarity == 0].shape[0]
nonZero0222 = pos0222 + neg0222
px0222 = round(((pos0222-neg0222)/nonZero0222),3)
volume0222 = nonZero0222 + neutral0222

pos0221 = df_tweets.loc[df_tweets.date=='2020-02-21'][df_tweets.polarity > 0].shape[0]
neg0221 = df_tweets.loc[df_tweets.date=='2020-02-21'][df_tweets.polarity < 0].shape[0]
neutral0221 = df_tweets.loc[df_tweets.date=='2020-02-21'][df_tweets.polarity == 0].shape[0]
nonZero0221 = pos0221 + neg0221
px0221 = round(((pos0221-neg0221)/nonZero0221),3)
volume0221 = nonZero0221 + neutral0221

pos0220 = df_tweets.loc[df_tweets.date=='2020-02-20'][df_tweets.polarity > 0].shape[0]
neg0220 = df_tweets.loc[df_tweets.date=='2020-02-20'][df_tweets.polarity < 0].shape[0]
```

```
neutral0220 = df_tweets.loc[df_tweets.date=='2020-02-20'][df_tweets.polarity == 0].shape[0]
nonZero0220 = pos0220 + neg0220
px0220 = round(((pos0220-neg0220)/nonZero0220),3)
volume0220 = nonZero0220 + neutral0220

pos0219 = df_tweets.loc[df_tweets.date=='2020-02-19'][df_tweets.polarity > 0].shape[0]
neg0219 = df_tweets.loc[df_tweets.date=='2020-02-19'][df_tweets.polarity < 0].shape[0]
neutral0219 = df_tweets.loc[df_tweets.date=='2020-02-19'][df_tweets.polarity == 0].shape[0]
nonZero0219 = pos0219 + neg0219
px0219 = round(((pos0219-neg0219)/nonZero0219),3)
volume0219 = nonZero0219 + neutral0219

pos0218 = df_tweets.loc[df_tweets.date=='2020-02-18'][df_tweets.polarity > 0].shape[0]
neg0218 = df_tweets.loc[df_tweets.date=='2020-02-18'][df_tweets.polarity < 0].shape[0]
neutral0218 = df_tweets.loc[df_tweets.date=='2020-02-18'][df_tweets.polarity == 0].shape[0]
nonZero0218 = pos0218 + neg0218
px0218 = round(((pos0218-neg0218)/nonZero0218),3)
volume0218 = nonZero0218 + neutral0218

pos0217 = df_tweets.loc[df_tweets.date=='2020-02-17'][df_tweets.polarity > 0].shape[0]
neg0217 = df_tweets.loc[df_tweets.date=='2020-02-17'][df_tweets.polarity < 0].shape[0]
neutral0217 = df_tweets.loc[df_tweets.date=='2020-02-17'][df_tweets.polarity == 0].shape[0]
nonZero0217 = pos0217 + neg0217
px0217 = round(((pos0217-neg0217)/nonZero0217),3)
volume0217 = nonZero0217 + neutral0217

pos0216 = df_tweets.loc[df_tweets.date=='2020-02-16'][df_tweets.polarity > 0].shape[0]
neg0216 = df_tweets.loc[df_tweets.date=='2020-02-16'][df_tweets.polarity < 0].shape[0]
neutral0216 = df_tweets.loc[df_tweets.date=='2020-02-16'][df_tweets.polarity == 0].shape[0]
nonZero0216 = pos0216 + neg0216
px0216 = round(((pos0216-neg0216)/nonZero0216),3)
volume0216 = nonZero0216 + neutral0216

pos0215 = df_tweets.loc[df_tweets.date=='2020-02-15'][df_tweets.polarity > 0].shape[0]
neg0215 = df_tweets.loc[df_tweets.date=='2020-02-15'][df_tweets.polarity < 0].shape[0]
neutral0215 = df_tweets.loc[df_tweets.date=='2020-02-15'][df_tweets.polarity == 0].shape[0]
nonZero0215 = pos0215 + neg0215
px0215 = round(((pos0215-neg0215)/nonZero0215),3)
volume0215 = nonZero0215 + neutral0215

pos0214 = df_tweets.loc[df_tweets.date=='2020-02-14'][df_tweets.polarity > 0].shape[0]
neg0214 = df_tweets.loc[df_tweets.date=='2020-02-14'][df_tweets.polarity < 0].shape[0]
neutral0214 = df_tweets.loc[df_tweets.date=='2020-02-14'][df_tweets.polarity == 0].shape[0]
nonZero0214 = pos0214 + neg0214
px0214 = round(((pos0214-neg0214)/nonZero0214),3)
volume0214 = nonZero0214 + neutral0214

pos0213 = df_tweets.loc[df_tweets.date=='2020-02-13'][df_tweets.polarity > 0].shape[0]
neg0213 = df_tweets.loc[df_tweets.date=='2020-02-13'][df_tweets.polarity < 0].shape[0]
neutral0213 = df_tweets.loc[df_tweets.date=='2020-02-13'][df_tweets.polarity == 0].shape[0]
nonZero0213 = pos0213 + neg0213
px0213 = round(((pos0213-neg0213)/nonZero0213),3)
volume0213 = nonZero0213 + neutral0213

pos0212 = df_tweets.loc[df_tweets.date=='2020-02-12'][df_tweets.polarity > 0].shape[0]
neg0212 = df_tweets.loc[df_tweets.date=='2020-02-12'][df_tweets.polarity < 0].shape[0]
neutral0212 = df_tweets.loc[df_tweets.date=='2020-02-12'][df_tweets.polarity == 0].shape[0]
nonZero0212 = pos0212 + neg0212
px0212 = round(((pos0212-neg0212)/nonZero0212),3)
volume0212 = nonZero0212 + neutral0212

pos0211 = df_tweets.loc[df_tweets.date=='2020-02-11'][df_tweets.polarity > 0].shape[0]
neg0211 = df_tweets.loc[df_tweets.date=='2020-02-11'][df_tweets.polarity < 0].shape[0]
neutral0211 = df_tweets.loc[df_tweets.date=='2020-02-11'][df_tweets.polarity == 0].shape[0]
nonZero0211 = pos0211 + neg0211
px0211 = round(((pos0211-neg0211)/nonZero0211),3)
volume0211 = nonZero0211 + neutral0211

pos0210 = df_tweets.loc[df_tweets.date=='2020-02-10'][df_tweets.polarity > 0].shape[0]
neg0210 = df_tweets.loc[df_tweets.date=='2020-02-10'][df_tweets.polarity < 0].shape[0]
neutral0210 = df_tweets.loc[df_tweets.date=='2020-02-10'][df_tweets.polarity == 0].shape[0]
nonZero0210 = pos0210 + neg0210
px0210 = round(((pos0210-neg0210)/nonZero0210),3)
volume0210 = nonZero0210 + neutral0210

pos0209 = df_tweets.loc[df_tweets.date=='2020-02-09'][df_tweets.polarity > 0].shape[0]
neg0209 = df_tweets.loc[df_tweets.date=='2020-02-09'][df_tweets.polarity < 0].shape[0]
```

```
neutral0209 = df_tweets.loc[df_tweets.date=='2020-02-09'][df_tweets.polarity == 0].shape[0]
nonZero0209 = pos0209 + neg0209
px0209 = round(((pos0209-neg0209)/nonZero0209),3)
volume0209 = nonZero0209 + neutral0209

pos0208 = df_tweets.loc[df_tweets.date=='2020-02-08'][df_tweets.polarity > 0].shape[0]
neg0208 = df_tweets.loc[df_tweets.date=='2020-02-08'][df_tweets.polarity < 0].shape[0]
neutral0208 = df_tweets.loc[df_tweets.date=='2020-02-08'][df_tweets.polarity == 0].shape[0]
nonZero0208 = pos0208 + neg0208
px0208 = round(((pos0208-neg0208)/nonZero0208),3)
volume0208 = nonZero0208 + neutral0208

pos0207 = df_tweets.loc[df_tweets.date=='2020-02-07'][df_tweets.polarity > 0].shape[0]
neg0207 = df_tweets.loc[df_tweets.date=='2020-02-07'][df_tweets.polarity < 0].shape[0]
neutral0207 = df_tweets.loc[df_tweets.date=='2020-02-07'][df_tweets.polarity == 0].shape[0]
nonZero0207 = pos0207 + neg0207
px0207 = round(((pos0207-neg0207)/nonZero0207),3)
volume0207 = nonZero0207 + neutral0207

pos0206 = df_tweets.loc[df_tweets.date=='2020-02-06'][df_tweets.polarity > 0].shape[0]
neg0206 = df_tweets.loc[df_tweets.date=='2020-02-06'][df_tweets.polarity < 0].shape[0]
neutral0206 = df_tweets.loc[df_tweets.date=='2020-02-06'][df_tweets.polarity == 0].shape[0]
nonZero0206 = pos0206 + neg0206
px0206 = round(((pos0206-neg0206)/nonZero0206),3)
volume0206 = nonZero0206 + neutral0206

pos0205 = df_tweets.loc[df_tweets.date=='2020-02-05'][df_tweets.polarity > 0].shape[0]
neg0205 = df_tweets.loc[df_tweets.date=='2020-02-05'][df_tweets.polarity < 0].shape[0]
neutral0205 = df_tweets.loc[df_tweets.date=='2020-02-05'][df_tweets.polarity == 0].shape[0]
nonZero0205 = pos0205 + neg0205
px0205 = round(((pos0205-neg0205)/nonZero0205),3)
volume0205 = nonZero0205 + neutral0205

pos0204 = df_tweets.loc[df_tweets.date=='2020-02-04'][df_tweets.polarity > 0].shape[0]
neg0204 = df_tweets.loc[df_tweets.date=='2020-02-04'][df_tweets.polarity < 0].shape[0]
neutral0204 = df_tweets.loc[df_tweets.date=='2020-02-04'][df_tweets.polarity == 0].shape[0]
nonZero0204 = pos0204 + neg0204
px0204 = round(((pos0204-neg0204)/nonZero0204),3)
volume0204 = nonZero0204 + neutral0204

pos0203 = df_tweets.loc[df_tweets.date=='2020-02-03'][df_tweets.polarity > 0].shape[0]
neg0203 = df_tweets.loc[df_tweets.date=='2020-02-03'][df_tweets.polarity < 0].shape[0]
neutral0203 = df_tweets.loc[df_tweets.date=='2020-02-03'][df_tweets.polarity == 0].shape[0]
nonZero0203 = pos0203 + neg0203
px0203 = round(((pos0203-neg0203)/nonZero0203),3)
volume0203 = nonZero0203 + neutral0203

pos0202 = df_tweets.loc[df_tweets.date=='2020-02-02'][df_tweets.polarity > 0].shape[0]
neg0202 = df_tweets.loc[df_tweets.date=='2020-02-02'][df_tweets.polarity < 0].shape[0]
neutral0202 = df_tweets.loc[df_tweets.date=='2020-02-02'][df_tweets.polarity == 0].shape[0]
nonZero0202 = pos0202 + neg0202
px0202 = round(((pos0202-neg0202)/nonZero0202),3)
volume0202 = nonZero0202 + neutral0202

pos0201 = df_tweets.loc[df_tweets.date=='2020-02-01'][df_tweets.polarity > 0].shape[0]
neg0201 = df_tweets.loc[df_tweets.date=='2020-02-01'][df_tweets.polarity < 0].shape[0]
neutral0201 = df_tweets.loc[df_tweets.date=='2020-02-01'][df_tweets.polarity == 0].shape[0]
nonZero0201 = pos0201 + neg0201
px0201 = round(((pos0201-neg0201)/nonZero0201),3)
volume0201 = nonZero0201 + neutral0201
```

```
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:34: UserWarning: Boolean Series k
ey will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:35: UserWarning: Boolean Series k
ey will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:36: UserWarning: Boolean Series k
ey will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:41: UserWarning: Boolean Series k
ey will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:42: UserWarning: Boolean Series k
ey will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:43: UserWarning: Boolean Series k
ey will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:48: UserWarning: Boolean Series k
ey will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:49: UserWarning: Boolean Series k
```

```
ey will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:50: UserWarning: Boolean Series k
ey will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:55: UserWarning: Boolean Series k
ey will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:56: UserWarning: Boolean Series k
ey will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:57: UserWarning: Boolean Series k
ey will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:62: UserWarning: Boolean Series k
ey will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:63: UserWarning: Boolean Series k
ey will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:64: UserWarning: Boolean Series k
ey will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:69: UserWarning: Boolean Series k
ey will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:70: UserWarning: Boolean Series k
ey will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:71: UserWarning: Boolean Series k
ey will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:76: UserWarning: Boolean Series k
ey will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:77: UserWarning: Boolean Series k
ey will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:78: UserWarning: Boolean Series k
ey will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:83: UserWarning: Boolean Series k
ey will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:84: UserWarning: Boolean Series k
ey will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:85: UserWarning: Boolean Series k
ey will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:90: UserWarning: Boolean Series k
ey will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:91: UserWarning: Boolean Series k
ey will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:92: UserWarning: Boolean Series k
ey will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:97: UserWarning: Boolean Series k
ey will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:98: UserWarning: Boolean Series k
ey will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:99: UserWarning: Boolean Series k
ey will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:104: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:105: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:106: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:111: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:112: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:113: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:118: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:119: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:120: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:125: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:126: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:127: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:132: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:133: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:134: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:139: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
```

```
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:140: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:141: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:146: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:147: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:148: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:153: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:154: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:155: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:160: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:161: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:162: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:167: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:168: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:169: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:174: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:175: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:176: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:181: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:182: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:183: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:188: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:189: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:190: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:195: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:196: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:197: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:202: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:203: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:204: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:209: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:210: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:211: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:216: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:217: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:218: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:223: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:224: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:225: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:230: UserWarning: Boolean Series
```

```
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:231: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:232: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:237: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:238: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:239: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:244: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:245: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:246: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:251: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:252: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:253: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:258: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:259: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:260: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:266: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:267: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:268: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:273: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:274: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:275: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:280: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:281: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:282: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:287: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:288: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:289: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:294: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:295: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:296: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:301: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:302: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:303: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:308: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:309: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:310: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:315: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:316: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:317: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
```

```
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:322: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:323: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:324: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:329: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:330: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:331: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:336: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:337: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:338: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:343: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:344: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:345: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:350: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:351: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:352: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:357: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:358: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:359: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:364: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:365: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:366: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:371: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:372: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:373: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:378: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:379: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:380: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:385: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:386: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:387: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:392: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:393: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:394: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:399: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:400: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:401: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:406: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:407: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:408: UserWarning: Boolean Series
```

```
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:413: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:414: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:415: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:420: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:421: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
C:\Users\timod\anaconda3\lib\site-packages\ipykernel_launcher.py:422: UserWarning: Boolean Series
key will be reindexed to match DataFrame index.
```

In [212]:

```
sentimentSeries = [px0201, px0202, px0203, px0204, px0205, px0206, px0207,
                   px0208, px0209, px0210, px0211, px0212, px0213, px0214,
                   px0215, px0216, px0217, px0218, px0219, px0220, px0221,
                   px0222, px0223, px0224, px0225, px0226, px0227, px0228,
                   px0229, px0301, px0302, px0303, px0304, px0305, px0306,
                   px0307, px0308, px0309, px0310, px0311, px0312, px0313,
                   px0314, px0315, px0316, px0317, px0318, px0319, px0320,
                   px0321, px0322, px0323, px0324, px0325, px0326, px0327]

sentiment = pd.DataFrame()
sentiment['Sentiment Polarity'] = sentimentSeries
#sentiment = sentiment.iloc[::-1].reset_index(drop=True)
#index 0 equals February 1, 2020
sentiment
```

Out[212]:

	Sentiment Polarity
0	0.474
1	0.388
2	0.476
3	0.496
4	0.531
5	0.470
6	0.489
7	0.532
8	0.438
9	0.457
10	0.489
11	0.524
12	0.414
13	0.501
14	0.668
15	0.545
16	0.463
17	0.393
18	0.534
19	0.418
20	0.468
21	0.582
22	0.455
23	0.311
24	0.318
25	0.337

	Sentiment Polarity
26	0.295
27	0.280
28	0.381
29	0.332
30	0.469
31	0.371
32	0.416
33	0.318
34	0.320
35	0.501
36	0.308
37	0.318
38	0.397
39	0.254
40	0.277
41	0.283
42	0.457
43	0.326
44	0.284
45	0.381
46	0.316
47	0.355
48	0.282
49	0.540
50	0.326
51	0.292
52	0.406
53	0.399
54	0.423
55	0.357

In [14]:

```
gt = pd.read_excel(r'C:\Users\timod\Desktop\SPX Chapter 6.xlsx')
gt.head()
```

Out[14]:

	Date	Relevance
0	2020-02-01	6
1	2020-02-02	5
2	2020-02-03	16
3	2020-02-04	17
4	2020-02-05	16

In [213]:

```
sGT = pd.DataFrame()
sGT['Date'], sGT['Sentiment Polarity'], sGT['Relevance'] = gt['Date'], sentiment['Sentiment Polarit
y'], gt['Relevance']
sGT
```

Out[213]:

	Date	Sentiment Polarity	Relevance

	Date	Sentiment Polarity	Relevance
0	2020-02-01	0.474	6
1	2020-02-02	0.388	5
2	2020-02-03	0.476	16
3	2020-02-04	0.496	17
4	2020-02-05	0.531	16
5	2020-02-06	0.470	16
6	2020-02-07	0.489	14
7	2020-02-08	0.532	5
8	2020-02-09	0.438	4
9	2020-02-10	0.457	12
10	2020-02-11	0.489	15
11	2020-02-12	0.524	15
12	2020-02-13	0.414	14
13	2020-02-14	0.501	13
14	2020-02-15	0.668	5
15	2020-02-16	0.545	5
16	2020-02-17	0.463	8
17	2020-02-18	0.393	13
18	2020-02-19	0.534	14
19	2020-02-20	0.418	14
20	2020-02-21	0.468	14
21	2020-02-22	0.582	5
22	2020-02-23	0.455	4
23	2020-02-24	0.311	28
24	2020-02-25	0.318	32
25	2020-02-26	0.337	32
26	2020-02-27	0.295	42
27	2020-02-28	0.280	59
28	2020-02-29	0.381	17
29	2020-03-01	0.332	11
30	2020-03-02	0.469	42
31	2020-03-03	0.371	40
32	2020-03-04	0.416	34
33	2020-03-05	0.318	34
34	2020-03-06	0.320	37
35	2020-03-07	0.501	12
36	2020-03-08	0.308	8
37	2020-03-09	0.318	90
38	2020-03-10	0.397	62
39	2020-03-11	0.254	56
40	2020-03-12	0.277	100
41	2020-03-13	0.283	80
42	2020-03-14	0.457	22
43	2020-03-15	0.326	21
44	2020-03-16	0.284	92
45	2020-03-17	0.381	74
46	2020-03-18	0.316	78
47	2020-03-19	0.355	63
48	2020-03-20	0.282	58
49	2020-03-21	0.540	21

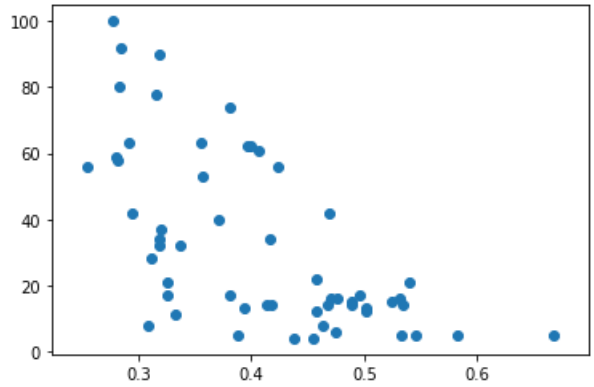
	Date	Sentiment Polarity	Relevance
50	2020-03-22	0.326	17
51	2020-03-23	0.292	63
52	2020-03-24	0.406	61
53	2020-03-25	0.399	62
54	2020-03-26	0.423	56
55	2020-03-27	0.357	53

In [18]:

```
plt.scatter(x='Sentiment Polarity',y='Relevance',data=sGT)
```

Out[18]:

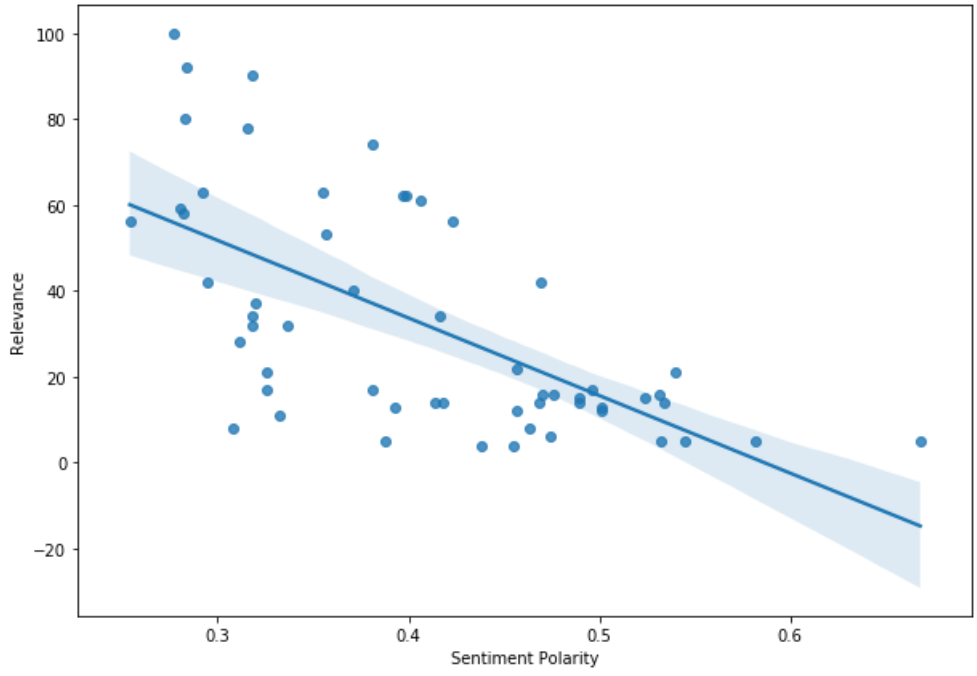
```
<matplotlib.collections.PathCollection at 0x27d9a1c3188>
```



In [21]:

```
plt.figure(figsize=(10,7))
sns.regplot(x='Sentiment Polarity',y='Relevance',data=sGT)
print(linregress(sGT['Sentiment Polarity'],sGT['Relevance']))
```

```
LinregressResult(slope=-180.54197181591144, intercept=105.88707011457745, rvalue=-
0.6411637212662549, pvalue=1.0175812945019365e-07, stderr=29.406036148399416)
```

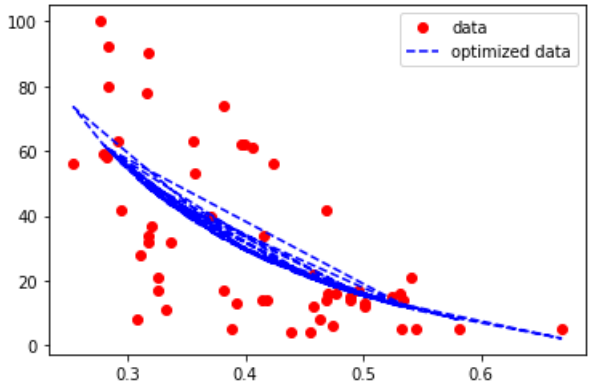


In [44]:

```
import numpy as np
```

```
from scipy.optimize import curve_fit

x = sGT['Sentiment Polarity']
y = sGT['Relevance']

def test(x, a, b, c):
    return a*(x**(-b))+c

c, cov = curve_fit(test, x,y)

print(c)
print(cov)


ans = c[0]*(x**(-c[1]))+c[2]


plt.plot(x, y, 'o', color ='red', label ="data")
plt.plot(x, ans, '--', color ='blue', label ="optimized data")
plt.legend()
plt.show()
```

```
[ 17.43288922   1.27789453 -26.94818202]
[[ 2.05518308e+03 -6.64655015e+01 -3.08563890e+03]
 [-6.64655015e+01  2.15723231e+00  9.93389227e+01]
 [-3.08563890e+03  9.93389227e+01  4.66656520e+03]]
```



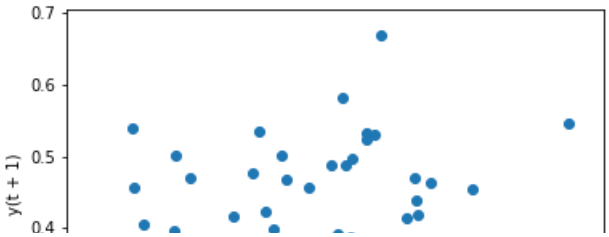
Twitter Sentiment

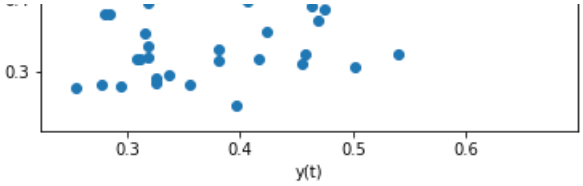
In [ ]:

```
```

In [69]:

```
#autoregression Chapter 6
#twitter sentiment
from pandas.plotting import lag_plot
from pandas import read_csv
from pandas import DataFrame
from pandas import concat

sentiment_series = sGT['Sentiment Polarity']

lag_plot(sentiment_series)
plt.show()
```





Autocorrelation plots

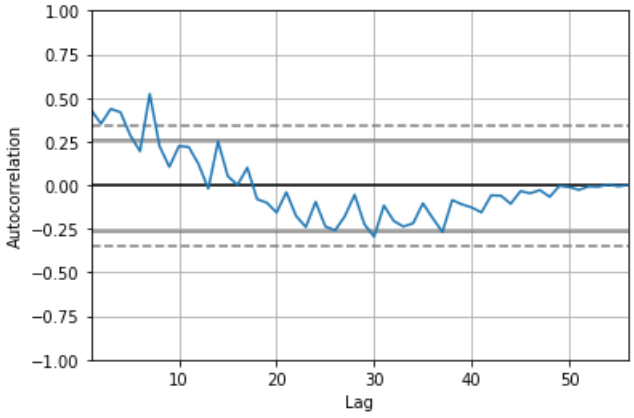
In [49]:

```
#checking for degree of correlation
values = DataFrame(sentiment_series.values)
dataframe = concat([values.shift(1), values], axis=1)
dataframe.columns = ['t-1', 't+1']
result = dataframe.corr()
print(result)
```

```
          t-1       t+1
t-1  1.000000  0.434031
t+1  0.434031  1.000000
```

In [50]:

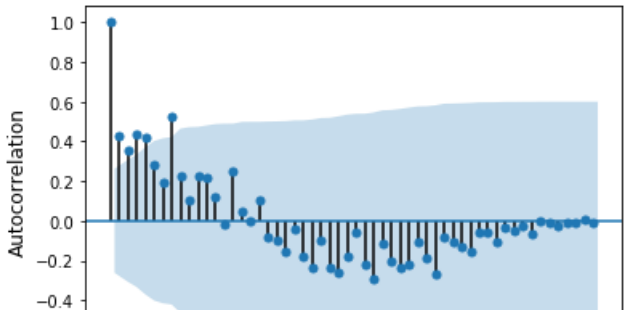
```
from pandas.plotting import autocorrelation_plot

autocorrelation_plot(sentiment_series)
plt.show()
```

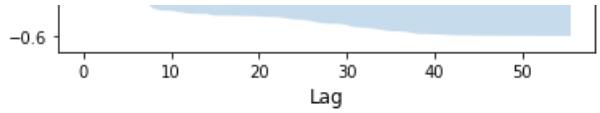


In [122]:

```
from statsmodels.graphics.tsaplots import plot_acf
plt.figure(figsize=(7,5))
plot_acf(sentiment_series, lags=55)
plt.xlabel('Lag',fontsize=12)
plt.ylabel('Autocorrelation',fontsize=12)
plt.title('')
plt.show()
```

```
<Figure size 504x360 with 0 Axes>
```



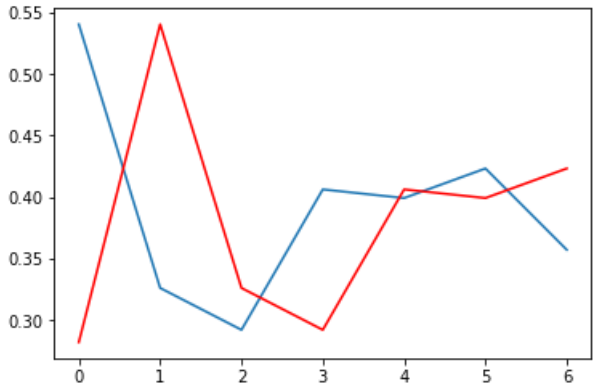


Persistence model (data not stationary)

In [52]:

```
from sklearn.metrics import mean_squared_error

def sqrt(x):
    return x**0.5
# split into train and test sets
X = dataframe.values
train, test = X[0:len(X)-7], X[len(X)-7:]
train_X, train_y = train[:,0], train[:,1]
test_X, test_y = test[:,0], test[:,1]

# persistence model
def model_persistence(x):
    return x

# walk-forward validation
predictions = list()
for x in test_X:
    yhat = model_persistence(x)
    predictions.append(yhat)
test_score = mean_squared_error(test_y, predictions)
print('Test MSE: %.3f' % test_score)
print('Test RMSE: %.3f' % sqrt(test_score))
# plot predictions vs expected
print("Red: Predicted")
print("Blue: Actual")
plt.plot(test_y)
plt.plot(predictions, color='red')
plt.show()
```

```
Test MSE: 0.019
Test RMSE: 0.137
Red: Predicted
Blue: Actual
```



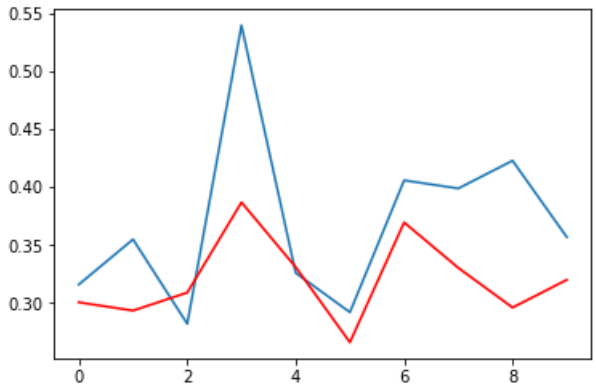
In [53]:

```
from statsmodels.tsa.ar_model import AR
from sklearn.metrics import mean_squared_error
# split dataset
X = series.values
train, test = X[0:len(X)-10], X[len(X)-10:]
# train autoregression
model = AR(train)
model_fit = model.fit()
print('Lag: %s' % model_fit.k_ar)
print('Coefficients: %s' % model_fit.params)
# make predictions
predictions = model_fit.predict(start=len(train), end=len(train)+len(test)-1, dynamic=False)
```

```
predictions = model_fit.predict(start=len(train), end=len(train)+len(test)-1, dynamic=False)
for i in range(len(predictions)):
 print('predicted=%f, expected=%f' % (predictions[i], test[i]))
error = mean_squared_error(test, predictions)
print('Test MSE: %.3f' % error)
print('Test RMSE: %.3f' % sqrt(error))
# plot results
plt.plot(test)
plt.plot(predictions, color='red')
plt.show()
```

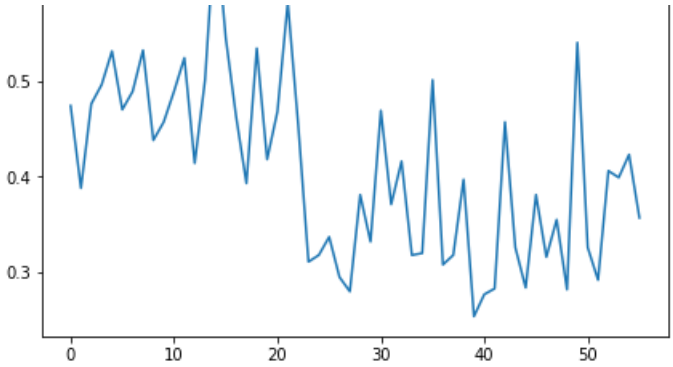
```
C:\Users\timod\anaconda3\lib\site-packages\statsmodels\tsa\ar_model.py:691: FutureWarning:
statsmodels.tsa.AR has been deprecated in favor of statsmodels.tsa.AutoReg and
statsmodels.tsa.SARIMAX.

AutoReg adds the ability to specify exogenous variables, include time trends,
and add seasonal dummies. The AutoReg API differs from AR since the model is
treated as immutable, and so the entire specification including the lag
length must be specified when creating the model. This change is too
substantial to incorporate into the existing AR api. The function
ar_select_order performs lag length selection for AutoReg models.

AutoReg only estimates parameters using conditional MLE (OLS). Use SARIMAX to
estimate ARX and related models using full MLE via the Kalman Filter.

To silence this warning and continue using AR until it is removed, use:

import warnings
warnings.filterwarnings('ignore', 'statsmodels.tsa.ar_model.AR', FutureWarning)

  warnings.warn(AR_DEPRECATION_WARN, FutureWarning)
```

```
Lag: 10
Coefficients: [ 0.04756819  0.42612376  0.06584497  0.11616215  0.00740938  0.02376644
 -0.14010855  0.53781123 -0.16928791 -0.19044988  0.17521746]
predicted=0.300593, expected=0.316000
predicted=0.293446, expected=0.355000
predicted=0.308980, expected=0.282000
predicted=0.386991, expected=0.540000
predicted=0.330997, expected=0.326000
predicted=0.266026, expected=0.292000
predicted=0.369640, expected=0.406000
predicted=0.330339, expected=0.399000
predicted=0.296066, expected=0.423000
predicted=0.320042, expected=0.357000
Test MSE: 0.005
Test RMSE: 0.072
```



In [124]:

```
plt.figure(figsize=(7,5))
sentiment_series.plot()
plt.show()
```



In [ ]:

```
sGT['diff_1'] = sGT['Sentiment Polarity'] - sGT['Sentiment Polarity'].shift(1)
sGT['diff_1'] = sGT['diff_1'].fillna(0)
```

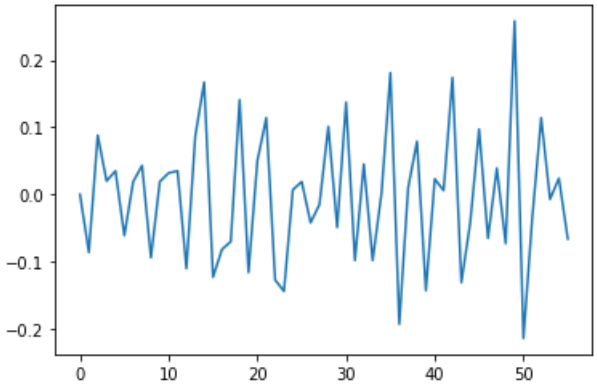
In [127]:

```
print(adf_test(sGT['diff_1']))
sGT['diff_1'].plot()
```

```
Results of Dickey-Fuller Test:
Test Statistic                  -3.256276
p-value                          0.016950
#Lags Used                       6.000000
Number of Observations Used     49.000000
Critical Value (1%)             -3.571472
Critical Value (5%)             -2.922629
Critical Value (10%)            -2.599336
dtype: float64
None
```

Out[127]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x27d9b7f2c08>
```

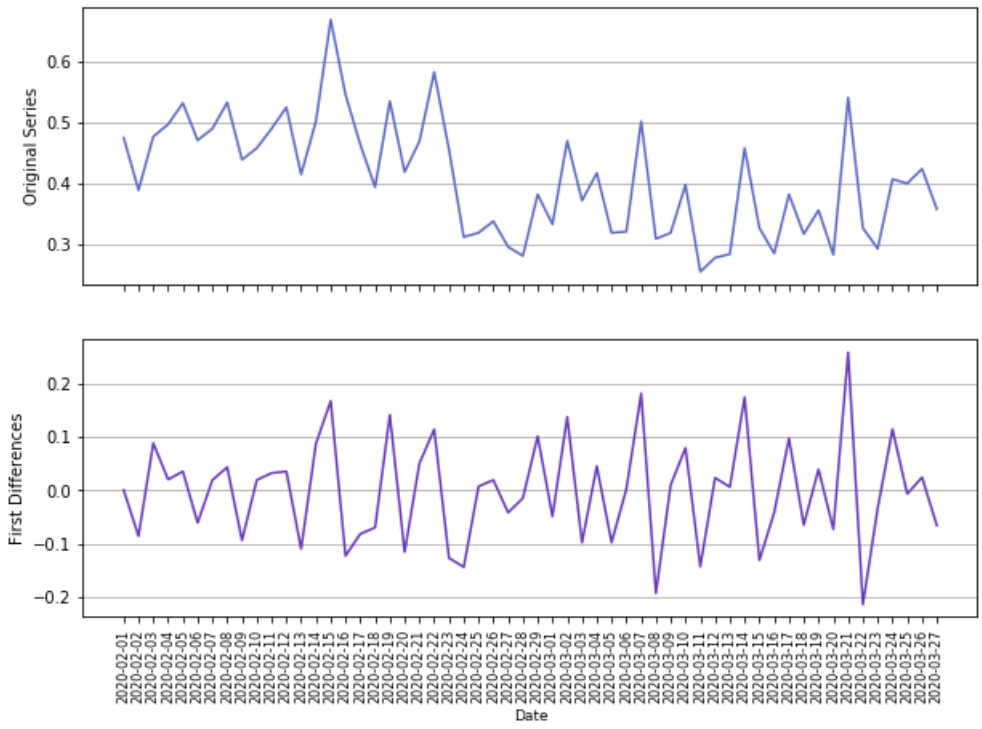


first differences plot

In [126]:

```
fig, axs = plt.subplots(2, sharex=True, sharey=False, figsize=(10,7))

series = sGT['Sentiment Polarity']
series_diff1 = sGT['diff_1']

x = sGT['Date']
plt.xticks(rotation=90, fontsize=8)

axs[0].yaxis.grid()
axs[1].yaxis.grid()

axs[0].plot(x, series,color='#5C6BC0')
```

```
axs[1].plot(x, series_diff1,color='#673AB7')

axs[0].set_ylabel("Original Series",fontsize=10)
axs[1].set_ylabel("First Differences",fontsize=10)

axs[1].set_xlabel("Date",fontsize=9)
plt.show()
```



In [200]:

```
from statsmodels.tsa.stattools import kpss
#define KPSS
def kpss_test(timeseries):
    print ('Results of KPSS Test:')
    kpsstest = kpss(timeseries, regression='c')
    kpss_output = pd.Series(kpsstest[0:3], index=['Test Statistic','p-value','Lags Used'])
    for key,value in kpsstest[3].items():
        kpss_output['Critical Value (%s)'%key] = value
    print (kpss_output)

kpss_test(sGT['diff_1'])
```

```
Results of KPSS Test:
Test Statistic            0.10414
p-value                   0.10000
Lags Used                11.00000
Critical Value (10%)      0.34700
Critical Value (5%)       0.46300
Critical Value (2.5%)     0.57400
Critical Value (1%)       0.73900
dtype: float64
```

In [93]:

In [ ]:

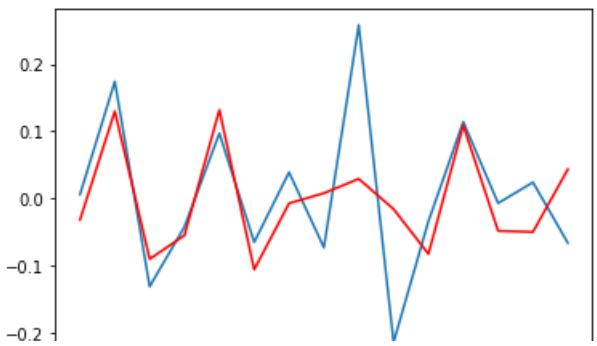
In [105]:

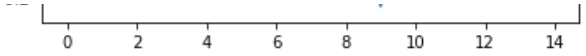
```
from statsmodels.tsa.ar_model import AR
from sklearn.metrics import mean_squared_error
```

```
from sklearn.metrics import mean_squared_error
# split dataset
X = sGT['diff_1'].values
train, test = X[0:len(X)-15], X[len(X)-15:]
# train autoregression
model = AR(train)
model_fit = model.fit()
print('Lag: %s' % model_fit.k_ar)
print('Coefficients: %s' % model_fit.params)
# make predictions
predictions = model_fit.predict(start=len(train), end=len(train)+len(test)-1, dynamic=False)
for i in range(len(predictions)):
 print('predicted=%f, expected=%f' % (predictions[i], test[i]))
error = mean_squared_error(test, predictions)
print('Test MSE: %.3f' % error)
print('Test RMSE: %.3f' % sqrt(error))
# plot results
plt.plot(test)
plt.plot(predictions, color='red')
plt.show()
```

```
C:\Users\timod\anaconda3\lib\site-packages\statsmodels\tsa\ar_model.py:691: FutureWarning:
statsmodels.tsa.AR has been deprecated in favor of statsmodels.tsa.AutoReg and
statsmodels.tsa.SARIMAX.

AutoReg adds the ability to specify exogenous variables, include time trends,
and add seasonal dummies. The AutoReg API differs from AR since the model is
treated as immutable, and so the entire specification including the lag
length must be specified when creating the model. This change is too
substantial to incorporate into the existing AR api. The function
ar_select_order performs lag length selection for AutoReg models.

AutoReg only estimates parameters using conditional MLE (OLS). Use SARIMAX to
estimate ARX and related models using full MLE via the Kalman Filter.

To silence this warning and continue using AR until it is removed, use:

import warnings
warnings.filterwarnings('ignore', 'statsmodels.tsa.ar_model.AR', FutureWarning)

  warnings.warn(AR_DEPRECATION_WARN, FutureWarning)
```

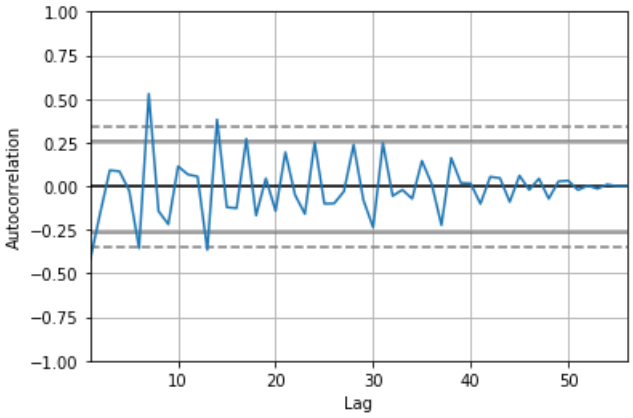
```
Lag: 10
Coefficients: [-0.01675757 -0.5448986  -0.45910225 -0.3102352  -0.2932247  -0.24260998
 -0.40131962  0.12662468  0.01047048 -0.18251469  0.06080979]
predicted=-0.031840, expected=0.006000
predicted=0.129823, expected=0.174000
predicted=-0.090132, expected=-0.131000
predicted=-0.054792, expected=-0.042000
predicted=0.131688, expected=0.097000
predicted=-0.105811, expected=-0.065000
predicted=-0.007246, expected=0.039000
predicted=0.007860, expected=-0.073000
predicted=0.029175, expected=0.258000
predicted=-0.015792, expected=-0.214000
predicted=-0.082551, expected=-0.034000
predicted=0.108787, expected=0.114000
predicted=-0.048290, expected=-0.007000
predicted=-0.049772, expected=0.024000
predicted=0.043352, expected=-0.066000
Test MSE: 0.009
Test RMSE: 0.093
```





In [108]:

```
autocorrelation_plot(sGT['diff_1'])
plt.show()
```



building an ARIMA Model

Building an ARIMA model. Using gridsearch for p,d,q paramters.

In [143]:

```
import warnings
warnings.filterwarnings("ignore")
```

In [145]:

```
def evaluate_arima_model(X, arima_order):
    # prepare training dataset
    train_size = int(len(X) * 0.66)
    train, test = X[0:train_size], X[train_size:]
    history = [x for x in train]
    # make predictions
    predictions = list()
    for t in range(len(test)):
        model = ARIMA(history, order=arima_order)
        model_fit = model.fit(disp=0)
        yhat = model_fit.forecast()[0]
        predictions.append(yhat)
        history.append(test[t])
    # calculate out of sample error
    error = mean_squared_error(test, predictions)
    return error
```

In [146]:

```
#evaluate combinations of p, d and q values for an ARIMA model
def evaluate_models(dataset, p_values, d_values, q_values):
    dataset = dataset.astype('float32')
    best_score, best_cfg = float("inf"), None
    for p in p_values:
        for d in d_values:
            for q in q_values:
                order = (p,d,q)
                try:
                    mse = evaluate_arima_model(dataset, order)
                    if mse < best_score:
                        best_score, best_cfg = mse, order
                    print('ARIMA%s MSE=%.3f' % (order,mse))
                except:
                    continue
    print('Best ARIMA%s MSE=%.3f' % (best_cfg, best_score))
```

```
    print('Best ARIMA%s MSE=%.3f' % (best_cfg, best_score))
```

In [147]:

```
p_range = [0, 1, 2, 4, 6, 8, 10]
d_range = range(0,3)
q_range = range(0,3)
data = sGT['diff_1'].values
evaluate_models(data, p_range, d_range, q_range )
```

```
ARIMA(0, 0, 0) MSE=0.014
ARIMA(0, 0, 1) MSE=0.007
ARIMA(0, 0, 2) MSE=0.007
ARIMA(0, 1, 0) MSE=0.045
ARIMA(0, 1, 1) MSE=0.014
ARIMA(0, 2, 0) MSE=0.150
ARIMA(0, 2, 1) MSE=0.047
ARIMA(0, 2, 2) MSE=0.014
ARIMA(1, 0, 0) MSE=0.010
ARIMA(1, 0, 1) MSE=0.007
ARIMA(1, 1, 0) MSE=0.027
ARIMA(1, 2, 0) MSE=0.078
ARIMA(2, 0, 0) MSE=0.008
ARIMA(2, 0, 1) MSE=0.007
ARIMA(2, 0, 2) MSE=0.007
ARIMA(2, 1, 0) MSE=0.018
ARIMA(2, 1, 1) MSE=0.008
ARIMA(2, 2, 0) MSE=0.045
ARIMA(4, 0, 0) MSE=0.007
ARIMA(4, 0, 1) MSE=0.007
ARIMA(4, 1, 0) MSE=0.010
ARIMA(4, 1, 1) MSE=0.009
ARIMA(4, 1, 2) MSE=0.009
ARIMA(4, 2, 1) MSE=0.017
ARIMA(6, 0, 0) MSE=0.005
ARIMA(6, 0, 1) MSE=0.005
ARIMA(8, 0, 0) MSE=0.005
ARIMA(10, 0, 0) MSE=0.006
Best ARIMA(6, 0, 0) MSE=0.005
```

In [148]:

```
p_range = [0, 1, 2, 4, 6, 8, 10]
d_range = range(0,3)
q_range = range(0,3)
data = sGT['Sentiment Polarity'].values
evaluate_models(data, p_range, d_range, q_range )
```

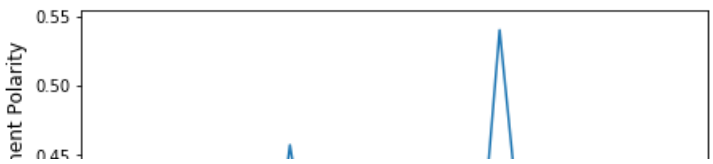
```
ARIMA(0, 0, 0) MSE=0.011
ARIMA(0, 0, 1) MSE=0.010
ARIMA(0, 0, 2) MSE=0.010
ARIMA(0, 1, 0) MSE=0.014
ARIMA(0, 1, 1) MSE=0.007
ARIMA(0, 1, 2) MSE=0.008
ARIMA(0, 2, 0) MSE=0.045
ARIMA(0, 2, 1) MSE=0.014
ARIMA(0, 2, 2) MSE=0.007
ARIMA(1, 0, 0) MSE=0.010
ARIMA(1, 0, 1) MSE=0.007
ARIMA(1, 1, 0) MSE=0.010
ARIMA(1, 1, 1) MSE=0.007
ARIMA(1, 1, 2) MSE=0.007
ARIMA(1, 2, 0) MSE=0.027
ARIMA(2, 0, 0) MSE=0.009
ARIMA(2, 0, 1) MSE=0.008
ARIMA(2, 1, 0) MSE=0.008
ARIMA(2, 1, 1) MSE=0.008
ARIMA(2, 1, 2) MSE=0.007
ARIMA(2, 2, 0) MSE=0.018
ARIMA(2, 2, 1) MSE=0.008
ARIMA(4, 0, 0) MSE=0.008
ARIMA(4, 1, 0) MSE=0.007
ARIMA(4, 1, 1) MSE=0.007
ARIMA(4, 2, 0) MSE=0.010
```

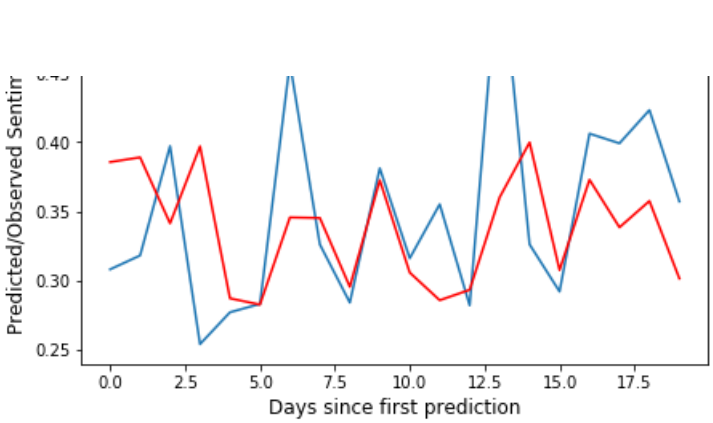
```
ARIMA(4, 2, 1) MSE=0.008
ARIMA(4, 2, 2) MSE=0.009
ARIMA(6, 0, 0) MSE=0.008
ARIMA(6, 0, 1) MSE=0.007
ARIMA(6, 1, 0) MSE=0.005
ARIMA(6, 1, 1) MSE=0.005
ARIMA(6, 1, 2) MSE=0.005
ARIMA(6, 2, 0) MSE=0.008
ARIMA(6, 2, 1) MSE=0.006
ARIMA(8, 0, 0) MSE=0.006
ARIMA(8, 1, 0) MSE=0.005
ARIMA(10, 0, 0) MSE=0.006
ARIMA(10, 1, 0) MSE=0.006
Best ARIMA(6, 1, 0) MSE=0.005
```

In [218]:

```
sentiment_data = sGT['Sentiment Polarity'].values
size = int(len(sentiment_data) * 0.66)
train, test = sentiment_data[0:size], sentiment_data[size:len(sentiment_data)]
history = [x for x in train]
predictions = list()
for t in range(len(test)):
    model = ARIMA(history, order=(6,1,0))
    model_fit = model.fit(disp=0)
    output = model_fit.forecast()
    yhat = output[0]
    predictions.append(yhat)
    obs = test[t]
    history.append(obs)
    print('predicted=%f, expected=%f' % (yhat, obs))
error = mean_squared_error(test, predictions)
print('Test MSE: %.3f' % error)
print('Test RMSE: %.3f' % sqrt(error))
print('Red: Predictions')
print('Blue: Observed')
# plot
plt.figure(figsize=(7,5))
plt.plot(test)
plt.plot(predictions, color='red')
plt.xlabel('Days since first prediction',fontsize=12)
plt.ylabel('Predicted/Observed Sentiment Polarity', fontsize=12)
plt.show()
```

```
predicted=0.385481, expected=0.308000
predicted=0.388843, expected=0.318000
predicted=0.341218, expected=0.397000
predicted=0.396806, expected=0.254000
predicted=0.287059, expected=0.277000
predicted=0.282545, expected=0.283000
predicted=0.345482, expected=0.457000
predicted=0.345094, expected=0.326000
predicted=0.295417, expected=0.284000
predicted=0.372376, expected=0.381000
predicted=0.305669, expected=0.316000
predicted=0.285697, expected=0.355000
predicted=0.293215, expected=0.282000
predicted=0.359991, expected=0.540000
predicted=0.399671, expected=0.326000
predicted=0.307529, expected=0.292000
predicted=0.372719, expected=0.406000
predicted=0.338323, expected=0.399000
predicted=0.357346, expected=0.423000
predicted=0.301374, expected=0.357000
Test MSE: 0.005
Test RMSE: 0.072
Red: Predictions
Blue: Observed
```





In [ ]:

```
```

In [210]:

```
len(test)
```

Out[210]:

20

In [211]:

```
len(train)
```

Out[211]:

36

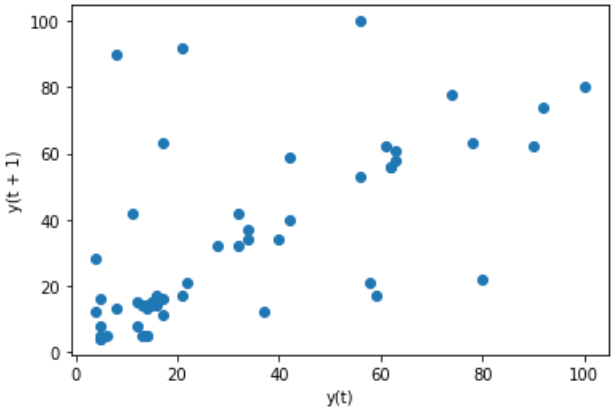
Google Trends

In [163]:

```
relevance_series = sGT['Relevance']
```

In [164]:

```
lag_plot(relevance_series)
plt.show()
```



In [165]:

```
#checking for degree of correlation
values = DataFrame(relevance_series.values)
dataframe = concat([values.shift(1), values], axis=1)
dataframe.columns = ['t-1', 't+1']
result = dataframe.corr()
```

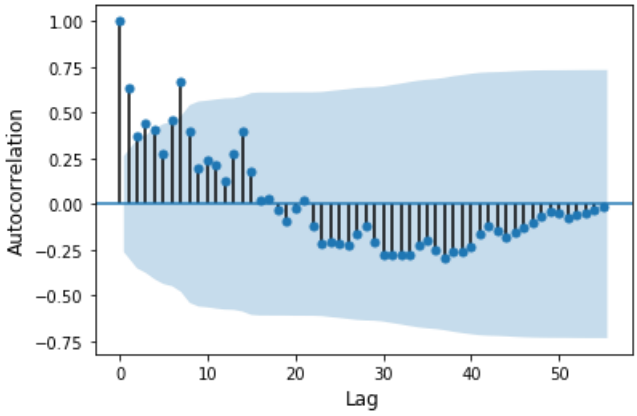
```
print(result)
```

```
          t-1       t+1
t-1  1.000000  0.641548
t+1  0.641548  1.000000
```

In [166]:

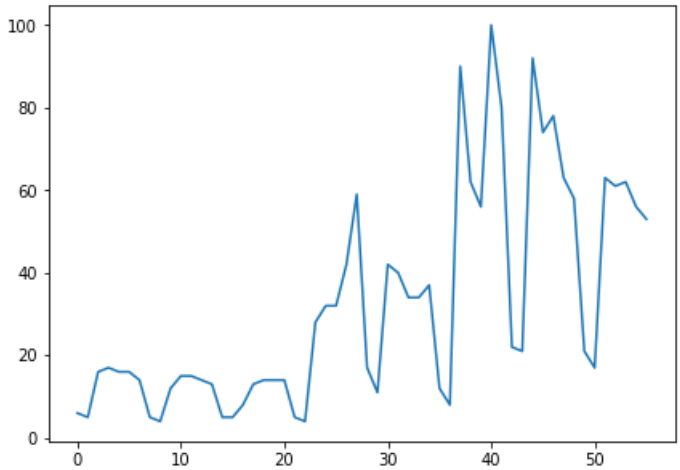
```
from statsmodels.graphics.tsaplots import plot_acf
plt.figure(figsize=(7,5))
plot_acf(relevance_series, lags=55)
plt.xlabel('Lag',fontsize=12)
plt.ylabel('Autocorrelation',fontsize=12)
plt.title('')
plt.show()
```

```
<Figure size 504x360 with 0 Axes>
```



In [170]:

```
plt.figure(figsize=(7,5))
relevance_series.plot()
plt.show()
```



In [169]:

```
print(adf_test(relevance_series))
```

```
Results of Dickey-Fuller Test:
Test Statistic                  -0.991032
p-value                          0.756521
#Lags Used                       7.000000
Number of Observations Used     48.000000
Critical Value (1%)             -3.574589
Critical Value (5%)             -2.923954
```

```
Critical Value (10%)           -2.600039
dtype: float64
None
```

In [175]:

```
sGT['G_diff_1'] = sGT['Relevance'] - sGT['Relevance'].shift(1)
sGT['G_diff_1'] = sGT['G_diff_1'].fillna(0)

sGT['G_diff_2'] = sGT['G_diff_1'] - sGT['G_diff_1'].shift(1)
sGT['G_diff_2'] = sGT['G_diff_2'].fillna(0)
```

In [172]:

```
print(adf_test(sGT['G_diff_1']))
```

```
Results of Dickey-Fuller Test:
Test Statistic                -3.316947
p-value                        0.014138
#Lags Used                     6.000000
Number of Observations Used   49.000000
Critical Value (1%)           -3.571472
Critical Value (5%)           -2.922629
Critical Value (10%)          -2.599336
dtype: float64
None
```

In [201]:

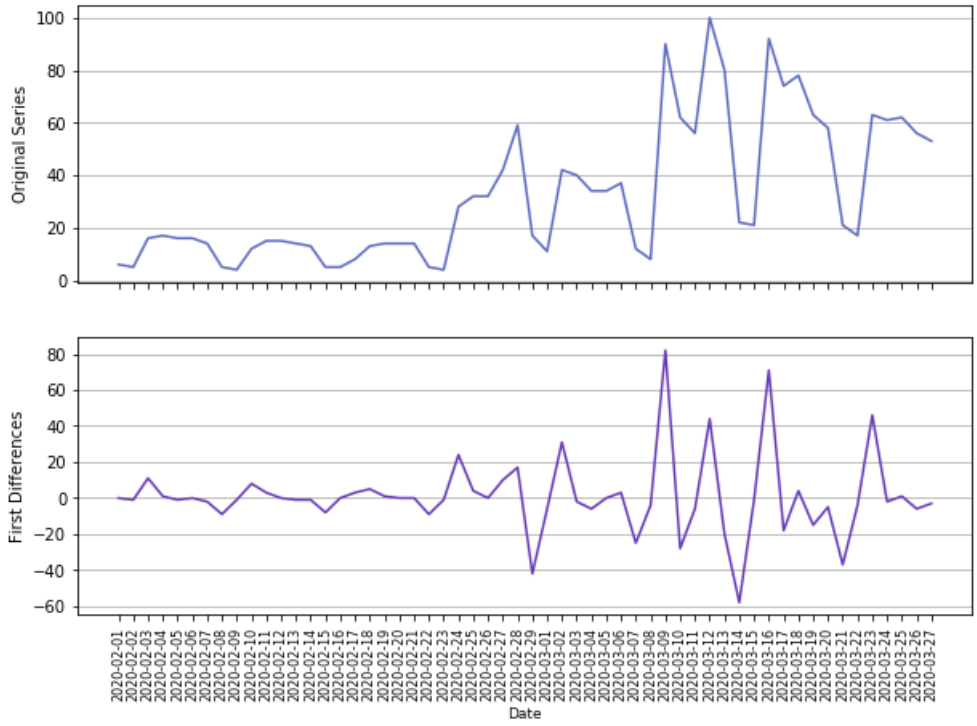
```
def kpss_test(timeseries):
    print ('Results of KPSS Test:')
    kpsstest = kpss(timeseries, regression='c')
    kpss_output = pd.Series(kpsstest[0:3], index=['Test Statistic','p-value','Lags Used'])
    for key,value in kpsstest[3].items():
        kpss_output['Critical Value (%s)'%key] = value
    print (kpss_output)

kpss_test(sGT['G_diff_1'])
```

```
Results of KPSS Test:
Test Statistic            0.092774
p-value                   0.100000
Lags Used                11.000000
Critical Value (10%)      0.347000
Critical Value (5%)       0.463000
Critical Value (2.5%)     0.574000
Critical Value (1%)       0.739000
dtype: float64
```

In [199]:

```
fig, axs = plt.subplots(2, sharex=True, sharey=False, figsize=(10,7))

relevance_series = sGT['Relevance']
series_diff1 = sGT['G_diff_1']

x = sGT['Date']
plt.xticks(rotation=90, fontsize=8)

axs[0].yaxis.grid()
axs[1].yaxis.grid()

axs[0].plot(x, relevance_series,color='#5C6BC0')
axs[1].plot(x, series_diff1,color='#673AB7')


axs[0].set_ylabel("Original Series",fontsize=10)
axs[1].set_ylabel("First Differences",fontsize=10)



axs[1].set_xlabel("Date",fontsize=9)
```

```
plt.show()
```



In [225]:

```
p_range = range(0,8)
d_range = range(0,3)
q_range = range(0,5)
G_data = sGT['Relevance'].values
evaluate_models(G_data, p_range, d_range, q_range )
```

```
ARIMA(0, 0, 0) MSE=1647.808
ARIMA(0, 0, 1) MSE=1060.967
ARIMA(0, 0, 2) MSE=1113.447
ARIMA(0, 0, 3) MSE=1255.010
ARIMA(0, 0, 4) MSE=1355.922
ARIMA(0, 1, 0) MSE=1147.795
ARIMA(0, 1, 1) MSE=1096.316
ARIMA(0, 1, 2) MSE=770.732
ARIMA(0, 1, 3) MSE=862.917
ARIMA(0, 2, 0) MSE=2704.617
ARIMA(0, 2, 1) MSE=1227.397
ARIMA(0, 2, 2) MSE=1540.636
ARIMA(0, 2, 4) MSE=1006.523
ARIMA(1, 0, 0) MSE=996.979
ARIMA(1, 0, 1) MSE=149165.705
ARIMA(1, 1, 0) MSE=1142.683
ARIMA(1, 1, 1) MSE=945.144
ARIMA(1, 1, 2) MSE=821.945
ARIMA(1, 2, 0) MSE=2699.936
ARIMA(2, 0, 0) MSE=1035.868
ARIMA(2, 0, 1) MSE=1068.809
ARIMA(2, 1, 0) MSE=979.530
ARIMA(2, 1, 1) MSE=1134.293
ARIMA(2, 2, 0) MSE=1677.520
ARIMA(3, 0, 0) MSE=1003.637
ARIMA(3, 1, 0) MSE=1034.199
ARIMA(3, 1, 1) MSE=1109.773
ARIMA(3, 2, 0) MSE=1597.995
ARIMA(4, 0, 0) MSE=1091.123
ARIMA(4, 0, 1) MSE=1223.485
ARIMA(4, 1, 0) MSE=1082.511
ARIMA(4, 1, 1) MSE=982.159
ARIMA(4, 2, 0) MSE=1711.570
ARIMA(5, 0, 0) MSE=1159.025
ARIMA(5, 0, 1) MSE=1118.815
ARIMA(5, 1, 0) MSE=946.066
ARIMA(6, 1, 0) MSE=805.198
```

```
Best ARIMA(0, 1, 2) MSE=770.732
```

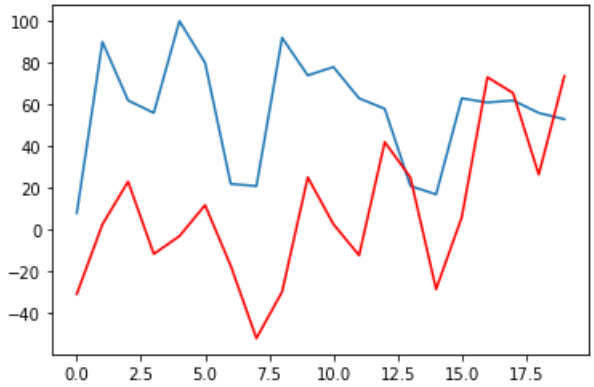
In [ ]:

```
```

In [226]:

```
from statsmodels.tsa.ar_model import AR
from sklearn.metrics import mean_squared_error
# split dataset
X = sGT['Relevance'].values
train, test = X[0:len(X)-20], X[len(X)-20:]
# train autoregression
model = AR(train)
model_fit = model.fit()
print('Lag: %s' % model_fit.k_ar)
print('Coefficients: %s' % model_fit.params)
# make predictions
predictions = model_fit.predict(start=len(train), end=len(train)+len(test)-1, dynamic=False)
for i in range(len(predictions)):
 print('predicted=%f, expected=%f' % (predictions[i], test[i]))
error = mean_squared_error(test, predictions)
print('Test MSE: %.3f' % error)
print('Test RMSE: %.3f' % sqrt(error))
# plot results
plt.plot(test)
plt.plot(predictions, color='red')
plt.show()
```

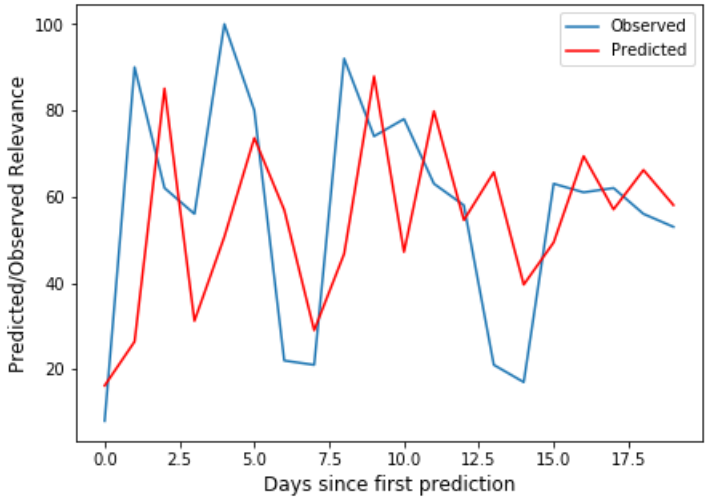
```
Lag: 9
Coefficients: [ 9.25628458e+00  6.35757116e-01 -7.76577609e-02  4.72435213e-01
 -3.42738708e-01 -2.53552825e-02 -3.50856215e-04  5.64762586e-01
  1.34720728e-01 -9.59538623e-01]
predicted=-30.717371, expected=8.000000
predicted=2.636277, expected=90.000000
predicted=23.125490, expected=62.000000
predicted=-11.531020, expected=56.000000
predicted=-3.013372, expected=100.000000
predicted=11.884771, expected=80.000000
predicted=-17.246098, expected=22.000000
predicted=-51.924066, expected=21.000000
predicted=-29.647534, expected=92.000000
predicted=25.189453, expected=74.000000
predicted=2.726467, expected=78.000000
predicted=-12.188651, expected=63.000000
predicted=42.050444, expected=58.000000
predicted=25.113936, expected=21.000000
predicted=-28.416066, expected=17.000000
predicted=6.015172, expected=63.000000
predicted=73.102754, expected=61.000000
predicted=65.551866, expected=62.000000
predicted=26.497258, expected=56.000000
predicted=73.688166, expected=53.000000
Test MSE: 3642.574
Test RMSE: 60.354
```



In [229]:

```
relevance_data = sGT['Relevance'].values
size = int(len(relevance_data) * 0.66)
train, test = relevance_data[0:size], relevance_data[size:len(relevance_data)]
history = [x for x in train]
predictions = list()
for t in range(len(test)):
    model = ARIMA(history, order=(0,1,2))
    model_fit = model.fit(disp=0)
    output = model_fit.forecast()
    yhat = output[0]
    predictions.append(yhat)
    obs = test[t]
    history.append(obs)
    print('predicted=%f, expected=%f' % (yhat, obs))
error = mean_squared_error(test, predictions)
print('Test MSE: %.3f' % error)
print('Test RMSE: %.3f' % sqrt(error))
print('Red: Predictions')
print('Blue: Observed')
# plot
plt.figure(figsize=(7,5))
plt.plot(test)
plt.plot(predictions, color='red')
plt.ylabel('Predicted/Observed Relevance',fontsize=12)
plt.xlabel('Days since first prediction',fontsize=12)
plt.legend(loc='best',labels=('Observed', 'Predicted'))
plt.show()
```

```
predicted=16.187090, expected=8.000000
predicted=26.399488, expected=90.000000
predicted=85.067194, expected=62.000000
predicted=31.203483, expected=56.000000
predicted=50.710634, expected=100.000000
predicted=73.591617, expected=80.000000
predicted=56.923418, expected=22.000000
predicted=29.023269, expected=21.000000
predicted=46.738694, expected=92.000000
predicted=87.881451, expected=74.000000
predicted=47.194720, expected=78.000000
predicted=79.816474, expected=63.000000
predicted=54.568511, expected=58.000000
predicted=65.674963, expected=21.000000
predicted=39.610559, expected=17.000000
predicted=49.395909, expected=63.000000
predicted=69.397584, expected=61.000000
predicted=57.044006, expected=62.000000
predicted=66.177726, expected=56.000000
predicted=58.045230, expected=53.000000
Test MSE: 770.732
Test RMSE: 27.762
Red: Predictions
Blue: Observed
```



In [ ]:

```

```

In [ ]:

```

```