INF-tut2

June 18, 2021

1 Importing packages

[2]:
```
!pip install sklearn
```

```
Requirement already satisfied: sklearn in c:\users\rluck\anaconda3\lib\site-
packages (0.0)
Requirement already satisfied: scikit-learn in
c:\users\rluck\anaconda3\lib\site-packages (from sklearn) (0.24.1)
Requirement already satisfied: numpy>=1.13.3 in
c:\users\rluck\anaconda3\lib\site-packages (from scikit-learn->sklearn) (1.20.1)
Requirement already satisfied: scipy>=0.19.1 in
c:\users\rluck\anaconda3\lib\site-packages (from scikit-learn->sklearn) (1.6.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in
c:\users\rluck\anaconda3\lib\site-packages (from scikit-learn->sklearn) (2.1.0)
Requirement already satisfied: joblib>=0.11 in
c:\users\rluck\anaconda3\lib\site-packages (from scikit-learn->sklearn) (1.0.1)
```


[3]:
```
#importing packages
import pandas as pd
import numpy as np
import statsmodels.api as sm
import statsmodels.formula.api as smf
import matplotlib.pyplot as plt
from sklearn import linear_model
```

2 Reading Excel file saved in hard drive

[4]:
```
#reading the file
df = pd.read_excel("C:\\Users\\rluck\\OneDrive\\fisher.xlsx", usecols␣
↪=["P","R"])
df.head()
```

[4]:

	P	R
0	17.0	5.90
1	17.1	5.65
2	17.3	6.42
3	17.5	8.80

```
4  17.6  6.85
```

3 Calculating annual inflation from quarterly CPI

[5]:
```
#computing the inflation rate
df['Inf'] = 400*np.log(df['P']/df['P'].shift(1)).dropna()
df.head()
```

[5]:
```
      P     R       Inf
0  17.0  5.90       NaN
1  17.1  5.65  2.346048
2  17.3  6.42  4.651215
3  17.5  8.80  4.597752
4  17.6  6.85  2.279208
```

[6]:
```
df.tail()
```

[6]:
```
          P     R       Inf
103  116.2  7.73  5.197128
104  117.6  7.54  4.790476
105  118.5  7.44  3.049570
106  119.0  7.51  1.684213
107  119.8  7.56  2.680077
```


[7]:
```
#dropping the N/A values
df1 = df.dropna(subset=["Inf"])
```

[8]:
```
df1.head()
```

[8]:
```
      P     R       Inf
1  17.1  5.65  2.346048
2  17.3  6.42  4.651215
3  17.5  8.80  4.597752
4  17.6  6.85  2.279208
5  17.9  6.37  6.760724
```

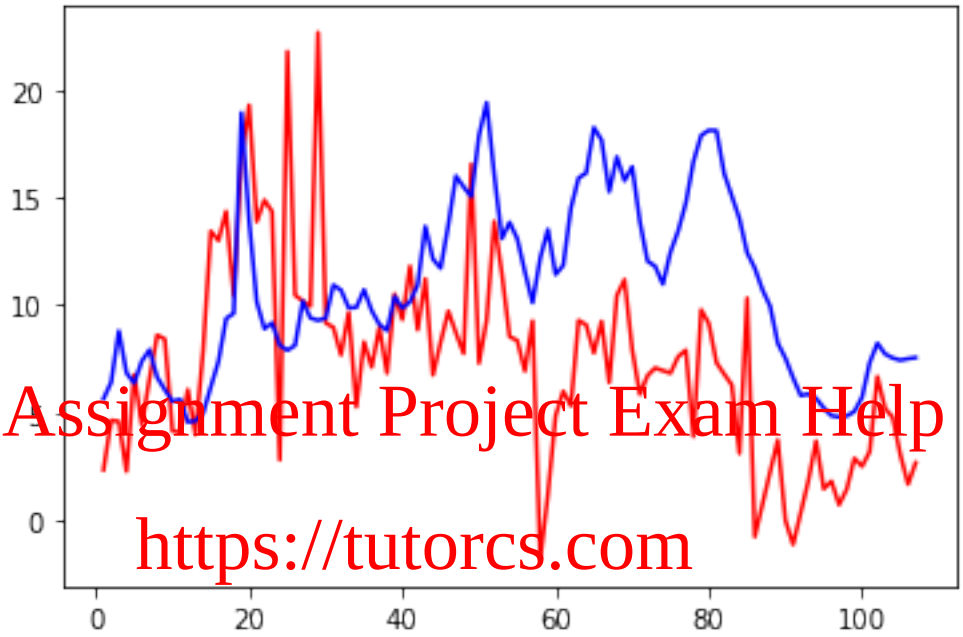
[9]:
```
df1.tail()
```

[9]:
```
          P     R       Inf
103  116.2  7.73  5.197128
104  117.6  7.54  4.790476
105  118.5  7.44  3.049570
106  119.0  7.51  1.684213
107  119.8  7.56  2.680077
```

4 Plotting the time series: Inflation

[10]:
```
#plotting the series
plt.plot(df1["Inf"], color='red', label='INF')
plt.plot(df1['R'], color='blue', label ='R')
```

[10]:
```
[<matplotlib.lines.Line2D at 0x22fe33d3a00>]
```



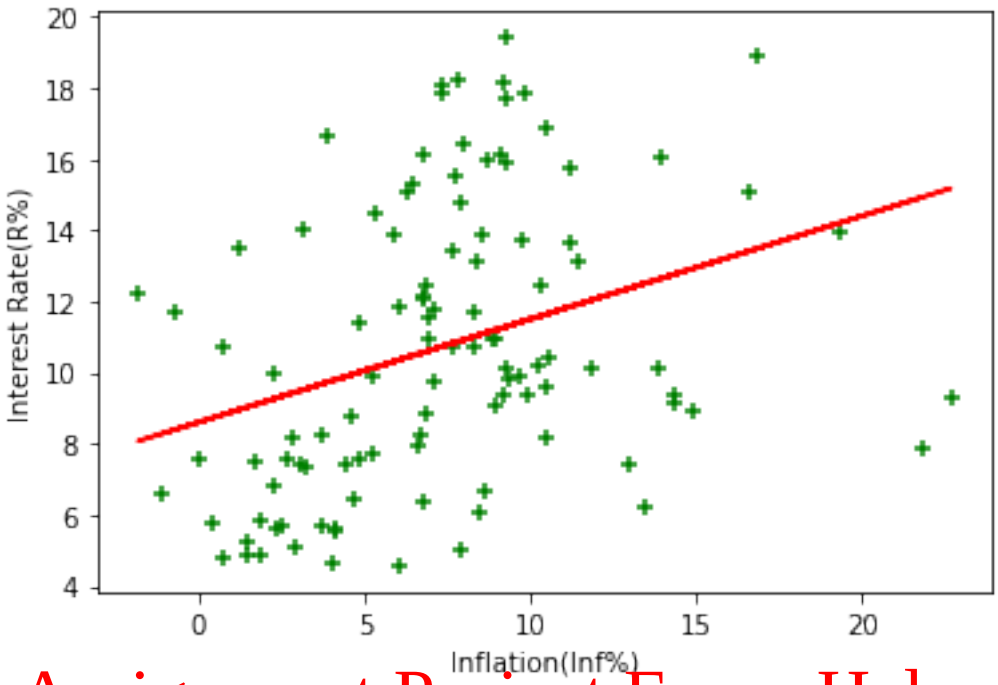
5 Linear Regression

[11]:
```
reg =linear_model.LinearRegression()
x =df1[['Inf']]
y =df1['R']
reg.fit(x,y)
```

[11]:
```
LinearRegression()
```

[12]:
```
plt.xlabel('Inflation(Inf%)')
plt.ylabel('Interest Rate(R%)')
plt.scatter(df1.Inf, df1.R, color='green', marker= '+')
plt.plot(df1.Inf, reg.predict(x), color='red')
```

[12]:
```
[<matplotlib.lines.Line2D at 0x22fe3b4cb20>]
```



[13]:
```
#X & y Variables defined
X = df1[['Inf']]
X = sm.add_constant(X)
y= df1['R']
#OLS model
model = sm.OLS(y,X).fit()
predictions =model.predict(X)
Q = model.summary()
print(Q)
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      R   R-squared:                       0.110
Model:                            OLS   Adj. R-squared:                  0.102
Method:                 Least Squares   F-statistic:                     13.01
Date:                Fri, 18 Jun 2021   Prob (F-statistic):           0.000475
Time:                        22:31:48   Log-Likelihood:                -293.17
No. Observations:                 107   AIC:                             590.3
Df Residuals:                     105   BIC:                             595.7
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          8.6008      0.691     12.445      0.000       7.231       9.971
```

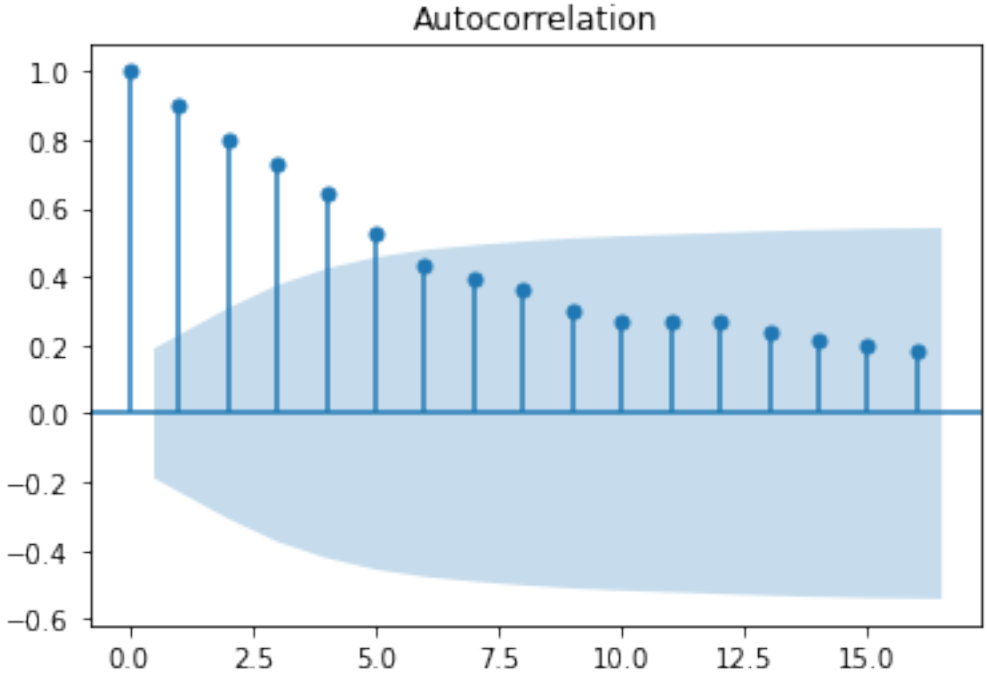
```
Inf             0.2898       0.080       3.608       0.000        0.131        0.449
==============================================================================
Omnibus:                       12.294   Durbin-Watson:                   0.288
Prob(Omnibus):                  0.002   Jarque-Bera (JB):                5.767
Skew:                           0.347   Prob(JB):                       0.0559
Kurtosis:                       2.098   Cond. No.                         16.4
==============================================================================

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

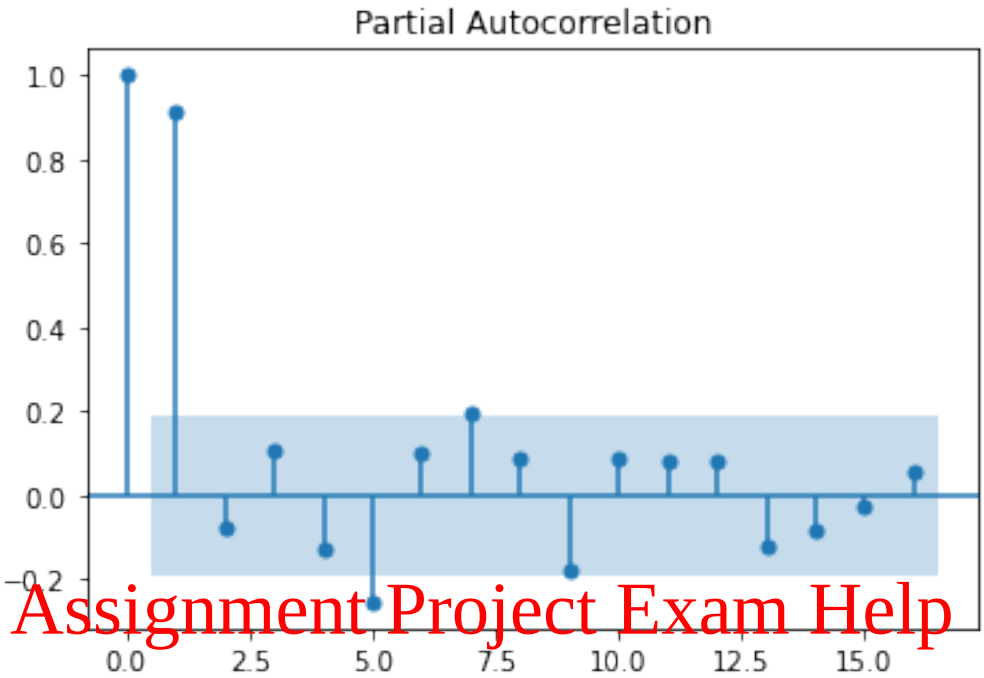
(c) The beta of 0.2898 is statistically significant, given the p-value is less than 5%. The intercept (constant in the above) is 8.690, which indicates the real interest rate.

(d) The ACF and PACF charts show strong auto-correlations for both interest rates and inflation rates, since they are outside the bands. As per PACF, the first-order autocorrelation coefficient is significant, thus explains that R_{t-2} is not directly correlated with R_t but impacts R_t through R_{t-1}

6 Correlogram: ACF and PACF

[14]:
```
#running ACF and PACF for Interest rates
dta= df1.R
sm.graphics.tsa.plot_acf(dta.values.squeeze(),lags=16)
sm.graphics.tsa.plot_pacf(dta.values.squeeze(),lags=16)
plt.show()
```



[15]:
```
# Generating the Q tables
dta= df1.R
r,q,p = sm.tsa.acf(dta.values.squeeze(), qstat=True)
data = np.c_[range(1,41), r[1:], q, p]
table = pd.DataFrame(data, columns=['lag', "AC", "Q", "Prob(>Q)"])
print (table.set_index('lag'))
```



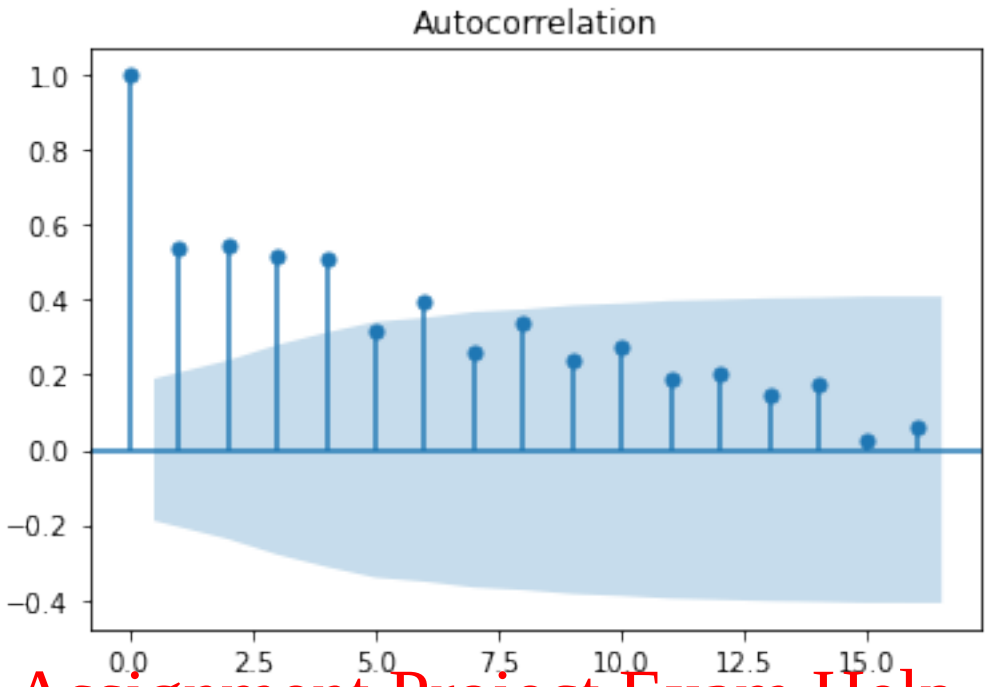
```
            AC           Q      Prob(>Q)
lag
1.0   0.902000   89.519458  3.036373e-21
2.0   0.800714  160.735161  1.249693e-35
3.0   0.727757  220.130163  1.881836e-47
4.0   0.643173  266.971371  1.434100e-56
5.0   0.522747  298.217241  2.420630e-62
6.0   0.432185  319.786128  4.692303e-66
7.0   0.390572  337.577638  5.612705e-69
8.0   0.359094  352.768802  2.322012e-71
9.0   0.296765  363.249933  9.357349e-73
10.0  0.266010  371.758078  9.546965e-74
11.0  0.269571  380.586555  8.045736e-75
12.0  0.271768  389.653956  5.861576e-76
13.0  0.239723  396.784131  1.068394e-76
14.0  0.213190  402.483940  3.799963e-77
```

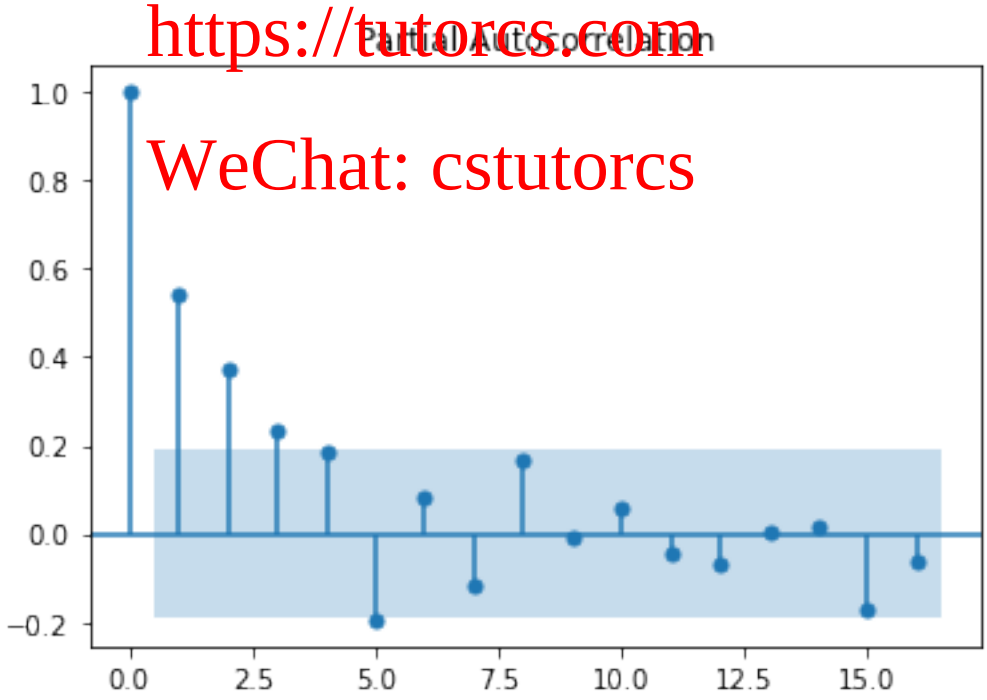
```
15.0  0.201682  407.640445  1.713880e-77
16.0  0.184015  411.980324  1.119549e-77
17.0  0.121160  413.882666  2.312769e-77
18.0  0.069831  414.521694  8.538552e-77
19.0  0.061388  415.021143  3.277967e-76
20.0  0.037563  415.210297  1.421541e-75
21.0 -0.016528  415.247345  6.459737e-75
22.0 -0.049924  415.589329  2.476365e-74
23.0 -0.066728  416.207554  8.136934e-74
24.0 -0.082528  417.164595  2.229276e-73
25.0 -0.116581  419.097697  3.776029e-73
26.0 -0.126028  421.384645  5.324409e-73
27.0 -0.097497  422.770443  1.130317e-72
28.0 -0.077050  423.646894  2.999114e-72
29.0 -0.075916  424.508648  7.881471e-72
30.0 -0.071108  425.274513  2.131973e-71
31.0 -0.052355  425.695155  6.669846e-71
32.0 -0.063438  426.320964  1.867262e-70
33.0 -0.116202  428.449132  2.565716e-70
34.0 -0.168424  432.810623  1.239430e-70
35.0 -0.195566  439.005942  2.546551e-71
36.0 -0.236616  448.202801  1.301903e-72
37.0 -0.287126  461.938753  8.162831e-75
38.0 -0.323681  479.647790  8.151842e-78
39.0 -0.334471  498.835308  4.101115e-81
40.0 -0.351261  520.313409  7.121062e-85
```



```
C:\Users\rluck\anaconda3\lib\site-packages\statsmodels\tsa\stattools.py:657:
FutureWarning: The default number of lags is changing from 40 tomin(int(10 *
np.log10(nobs)), nobs - 1) after 0.12is released. Set the number of lags to an
integer to  silence this warning.
  warnings.warn(
C:\Users\rluck\anaconda3\lib\site-packages\statsmodels\tsa\stattools.py:667:
FutureWarning: fft=True will become the default after the release of the 0.12
release of statsmodels. To suppress this warning, explicitly set fft=False.
  warnings.warn(
```

[16]:
```
#running ACF and PACF for Inflation rates
dt= df1.Inf
sm.graphics.tsa.plot_acf(dt.values.squeeze(),lags=16)
sm.graphics.tsa.plot_pacf(dt.values.squeeze(),lags=16)
plt.show()
```





[17]:
```
# Generating the Q tables
dt= df1.Inf
r,q,p = sm.tsa.acf(dt.values.squeeze(), qstat=True)
data = np.c_[range(1,41), r[1:], q, p]
table = pd.DataFrame(data, columns=['lag', "AC", "Q", "Prob(>Q)"])
print (table.set_index('lag'))
```

```
            AC           Q      Prob(>Q)
lag
1.0   0.538668   31.926214  1.601414e-08
2.0   0.546611   65.113999  7.255626e-15
3.0   0.520620   95.510192  1.434392e-20
4.0   0.512728  125.277975  3.980674e-26
5.0   0.320958  137.056889  7.552935e-28
6.0   0.395231  155.094995  6.469263e-31
7.0   0.263380  163.185481  6.848909e-32
8.0   0.335635  176.456691  5.707403e-34
9.0   0.236423  183.108875  1.136013e-34
10.0  0.276055  192.271705  6.581374e-36
11.0  0.190578  196.664218  3.623768e-36
12.0  0.204584  201.822653  1.371277e-36
13.0  0.144076  204.398177  1.706136e-36
14.0  0.177849  208.364873  1.069236e-36
15.0  0.024400  208.440346  4.078673e-36
16.0  0.057629  208.865988  1.272269e-35
17.0  0.041227  209.086241  4.236566e-35
18.0 -0.013514  209.110173  1.498819e-34
19.0 -0.030582  209.284125  4.903207e-34
20.0  0.145808  212.084192  4.509795e-34
21.0  0.013762  212.109878  1.477257e-33
22.0  0.028119  212.218371  4.546336e-33
23.0  0.070134  212.901328  1.054903e-32
24.0  0.108008  214.540562  1.562105e-32
25.0 -0.003713  214.542523  4.739521e-32
26.0  0.069558  215.239186  1.034074e-31
27.0  0.068827  215.929804  2.224307e-31
28.0  0.084139  216.974961  4.026885e-31
29.0  0.013086  217.000565  1.124209e-30
30.0  0.054102  217.443909  2.569765e-30
31.0  0.039647  217.685134  6.315696e-30
32.0  0.090422  218.956577  9.792370e-30
33.0  0.018652  219.011410  2.534653e-29
34.0  0.045297  219.339228  5.748685e-29
35.0 -0.028321  219.469153  1.399130e-28
36.0 -0.032441  219.642032  3.297061e-28
37.0 -0.074630  220.570003  5.576472e-28
38.0 -0.112505  222.709467  5.615226e-28
39.0 -0.127265  225.487401  4.303657e-28
```

```
40.0 -0.077515  226.533343  6.770094e-28

C:\Users\rluck\anaconda3\lib\site-packages\statsmodels\tsa\stattools.py:657:
FutureWarning: The default number of lags is changing from 40 tomin(int(10 *
np.log10(nobs)), nobs - 1) after 0.12is released. Set the number of lags to an
integer to  silence this warning.
  warnings.warn(
C:\Users\rluck\anaconda3\lib\site-packages\statsmodels\tsa\stattools.py:667:
FutureWarning: fft=True will become the default after the release of the 0.12
release of statsmodels. To suppress this warning, explicitly set fft=False.
  warnings.warn(
```

[ ]: