Practical 8 - JosiahTeh

December 28, 2021

1 First Name: Josiah

2 Last Name: Teh

[1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import statsmodels.formula.api as smf
```

[2]:
```
pd.set_option('display.float_format', lambda x:'%.2f'%x)

gapminder = pd.read_csv('gapminder.csv', low_memory=False)
gapminder.head()
```

[2]:
```
       country   incomeperperson alcconsumption armedforcesrate  \
0  Afghanistan                                .03        .5696534
1      Albania  1914.99655094922           7.29       1.0247361
2      Algeria  2231.99333515006            .69        2.306817
3      Andorra  21943.3398976022          10.17
4       Angola  1381.00426770244           5.57       1.4613288

  breastcancerper100th      co2emissions  femaleemployrate hivrate  \
0                 26.8          75944000  25.6000003814697
1                 57.4  223747333.333333  42.0999984741211
2                 23.5  2932108666.66667  31.7000007629394      .1
3
4                 23.1         248358000  69.4000015258789       2

    internetuserate lifeexpectancy     oilperperson polityscore  \
0  3.65412162280064         48.673                             0
1  44.9899469578783         76.918                             9
2  12.5000733055148         73.131  .42009452521537            2
3                81
4  9.99995388324075         51.093                            -2

  relectricperperson   suicideper100th        employrate urbanrate
```

```
0                    6.68438529968262  55.7000007629394      24.04
1   636.341383366604  7.69932985305786  51.4000015258789      46.72
2   590.509814347428   4.8487696647644              50.5      65.22
3                    5.36217880249023                        88.92
4   172.999227388199  14.5546770095825  75.6999969482422       56.7
```

[3]:
```
gapminder['oilperperson'] = pd.
 ↪to_numeric(gapminder['oilperperson'],errors='coerce')
gapminder['relectricperperson'] = pd.
 ↪to_numeric(gapminder['relectricperperson'],errors='coerce')
gapminder['co2emissions'] = pd.
 ↪to_numeric(gapminder['co2emissions'],errors='coerce')
```

3 Scenario 1 - Linear & Multiple

4 sub1

[4]:
```
sub1 = gapminder[['oilperperson', 'relectricperperson', 'co2emissions']].
 ↪dropna()
sub1.head()
```

[4]:
```
    oilperperson  relectricperperson   co2emissions
2           0.42              590.51  2932108666.67
6           0.64              768.43  5872119000.00
9           1.91             2825.39 12970092666.67
10          1.55             2068.12  4466084333.33
11          0.36              921.56   511107666.67
```

5 Centre oilperperson, relectricperperson and co2emissions

6 use sub1

[8]:
```
# hint lecture cell 5
# center quantitative variables for regression analysis
sub1['oilperperson_c'] = (sub1['oilperperson'] -sub1['oilperperson'].mean())
sub1['relectricperperson_c'] = (sub1['relectricperperson'] -␣
 ↪sub1['relectricperperson'].mean())
sub1['co2emissions_c'] = (sub1['co2emissions'] - sub1['co2emissions'].mean())

sub1.head()
```

[8]:
```
    oilperperson  relectricperperson   co2emissions  oilperperson_c  \
2           0.42              590.51  2932108666.67           -1.06
6           0.64              768.43  5872119000.00           -0.85
9           1.91             2825.39 12970092666.67            0.43
10          1.55             2068.12  4466084333.33            0.06
```

```
11          0.36              921.56   511107666.67          -1.12

    relectricperperson_c  co2emissions_c
2               -1145.94 -12353375047.62
6                -968.02  -9413364714.29
9                1088.94  -2315391047.62
10                331.68 -10819399380.95
11               -814.89 -14774376047.62
```

7 Multi variable linear regression

8 predict co2emission(y) using relectricperperson(x1) and oilperperson(x2)

9 use sub1

[9]:
```
# hint lecture cell 6
reg1 = smf.ols('co2emissions_c ~ relectricperperson_c + oilperperson_c',␣
 ↪data=sub1).fit()
print (reg1.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:         co2emissions_c   R-squared:                       0.020
Model:                            OLS   Adj. R-squared:                 -0.012
Method:                 Least Squares   F-statistic:                    0.6205
Date:                Tue, 28 Dec 2021   Prob (F-statistic):              0.541
Time:                        13:49:53   Log-Likelihood:                -1632.7
No. Observations:                  63   AIC:                             3271.
Df Residuals:                      60   BIC:                             3278.
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
========
                             coef    std err          t      P>|t|      [0.025
0.975]
------------------------------------------------------------------------------
--------
Intercept            -2.533e-06   5.62e+09   -4.5e-16      1.000   -1.12e+10
1.12e+10
relectricperperson_c  3.434e+06   3.24e+06      1.058      0.294   -3.06e+06
9.92e+06
oilperperson_c        -9.47e+08   3.65e+09     -0.260      0.796   -8.25e+09
6.35e+09
==============================================================================
Omnibus:                      116.246   Durbin-Watson:                   1.762
```

```
Prob(Omnibus):                  0.000   Jarque-Bera (JB):             4225.198
Skew:                           5.891   Prob(JB):                         0.00
Kurtosis:                      41.351   Cond. No.                     2.04e+03
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 2.04e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
```

10 Scenario 2 - Linear

11 sub2

[10]:
```
# convert to numeric format
gapminder['employrate'] = pd.to_numeric(gapminder['employrate'],␣
 ↪errors='coerce')
sub2 = gapminder[['relectricperperson', 'employrate']].dropna()
sub2.head()
```

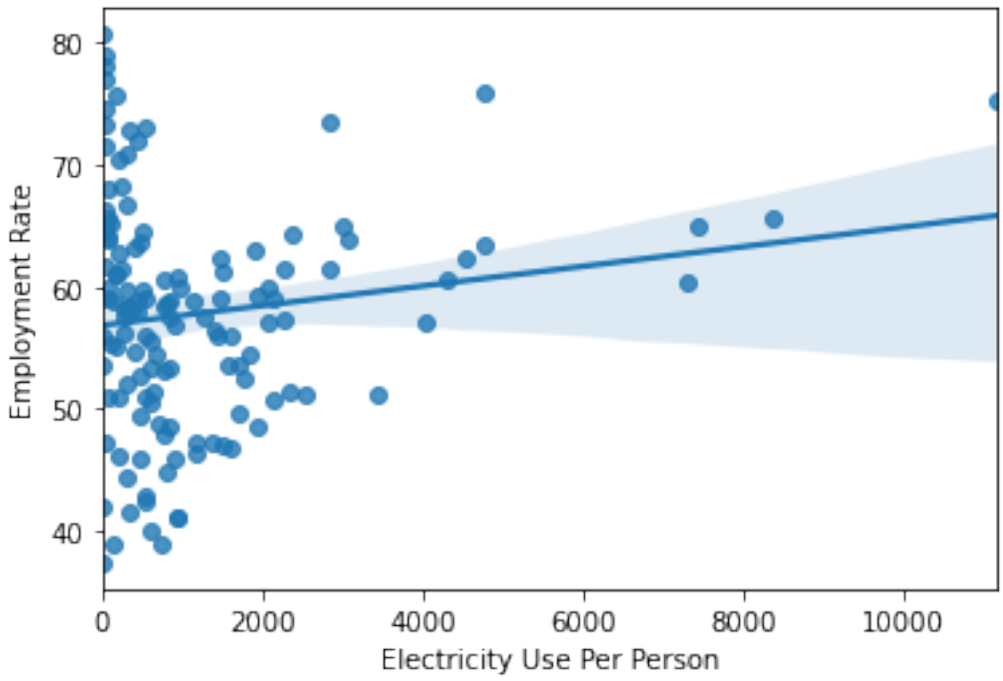
[10]:

	relectricperperson	employrate
1	636.34	51.40
2	590.51	50.50
4	173.00	75.70
6	768.43	58.40
7	603.76	40.10

12 scatter plot to show relationship between employment rate (x) and electricity use per person (y)

[13]:
```
# hint lecture cell 8
%matplotlib inline
plt.figure()
scat1 = sns.regplot(x='relectricperperson', y='employrate', fit_reg=True,␣
 ↪data=sub2)

plt.xlabel('Electricity Use Per Person')
plt.ylabel('Employment Rate')
```

[13]:
```
Text(0, 0.5, 'Employment Rate')
```



13 Centre relectricperperson and employrate

14 use sub2

[16]:
```
# hint lecture cell 9
sub2['relectricperperson_c'] = (sub2['relectricperperson'] -␣
↪sub2['relectricperperson'].mean())
sub2['employrate_c'] = (sub2['employrate'] - sub2['employrate'].mean())

sub2.head()
```

[16]:

	relectricperperson	employrate	relectricperperson_c	employrate_c
1	636.34	51.40	-543.99	-6.41
2	590.51	50.50	-589.82	-7.31
4	173.00	75.70	-1007.33	17.89
6	768.43	58.40	-411.90	0.59
7	603.76	40.10	-576.57	-17.71

15 Linear regression between relectricperperson (x) and employrate (y)

16 use sub2

[17]:
```
# hint lecture cell 10
reg2 = smf.ols('employrate_c ~ relectricperperson_c', data=sub2).fit()
print (reg2.summary())
```

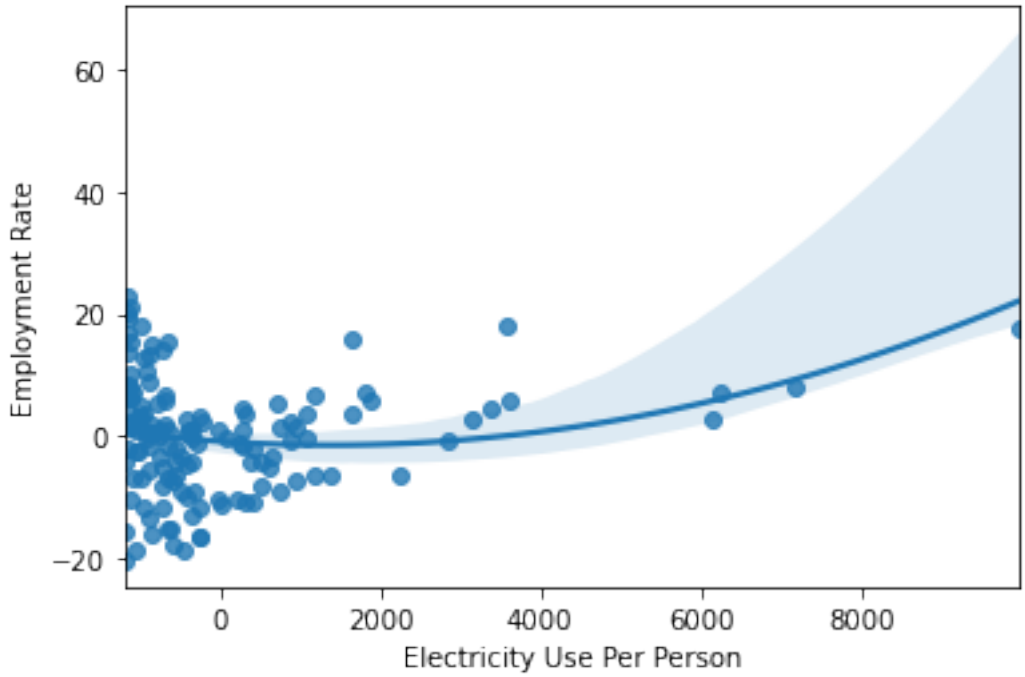
```
                            OLS Regression Results
==============================================================================
Dep. Variable:           employrate_c   R-squared:                       0.021
Model:                            OLS   Adj. R-squared:                  0.014
Method:                 Least Squares   F-statistic:                     2.877
Date:                Tue, 28 Dec 2021   Prob (F-statistic):             0.0922
Time:                        13:55:12   Log-Likelihood:                -487.37
No. Observations:                 134   AIC:                             978.7
Df Residuals:                     132   BIC:                             984.5
Df Model:                           1
Covariance Type:            nonrobust
================================================================================
========
                           coef    std err          t      P>|t|      [0.025
0.975]
--------------------------------------------------------------------------------
--------
Intercept             4.927e-15      0.800   6.16e-15      1.000      -1.582
1.582
relectricperperson_c     0.0008      0.000      1.696      0.092      -0.000
0.002
==============================================================================
Omnibus:                        1.259   Durbin-Watson:                   2.002
Prob(Omnibus):                  0.533   Jarque-Bera (JB):                1.253
Skew:                           0.228   Prob(JB):                        0.534
Kurtosis:                       2.874   Cond. No.                     1.68e+03
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 1.68e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
```

17 Scenario 3 - Polynomial

18 scatter plot to show polynomial (order 2) relationship between employment rate (x) and electricity use per person (y)

[18]:
```
# hint lecture cell 11
#fit second order polynomial
# run the 2 scatterplots together to get second order fit lines
plt.figure()
scat1 = sns.regplot(x='relectricperperson_c', y='employrate_c', order=2,␣
 ↪data=sub2)
plt.xlabel('Electricity Use Per Person')
plt.ylabel('Employment Rate')
```

[18]:
```
Text(0, 0.5, 'Employment Rate')
```



19 Polynomial regression between relectricperperson (x - order 2) and employrate (y)

20 use sub2

[20]:
```
# hint lecture cell 12
reg2 = smf.ols('employrate_c ~ I(relectricperperson_c**2)', data=sub2).fit()
print (reg2.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:           employrate_c   R-squared:                       0.054
Model:                            OLS   Adj. R-squared:                  0.047
Method:                 Least Squares   F-statistic:                     7.606
Date:                Tue, 28 Dec 2021   Prob (F-statistic):            0.00664
Time:                        13:57:20   Log-Likelihood:                -485.06
No. Observations:                 134   AIC:                             974.1
Df Residuals:                     132   BIC:                             979.9
Df Model:                           1
Covariance Type:            nonrobust
================================================================================
================
                                   coef    std err          t      P>|t|
[0.025      0.975]
--------------------------------------------------------------------------------
----------------
Intercept                       -0.5780      0.814     -0.710      0.479
-2.187       1.032
I(relectricperperson_c ** 2)  2.037e-07   7.39e-08      2.758      0.007
5.76e-08     3.5e-07
==============================================================================
Omnibus:                        0.919   Durbin-Watson:                   2.029
Prob(Omnibus):                  0.632   Jarque-Bera (JB):                0.872
Skew:                           0.194   Prob(JB):                        0.647
Kurtosis:                       2.924   Cond. No.                     1.14e+07
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 1.14e+07. This might indicate that there are
strong multicollinearity or other numerical problems.
```

21 Scenario 4 - Multiple & poly

22 sub3

[21]:
```
sub3 = gapminder[['oilperperson', 'relectricperperson',␣
↪'co2emissions','employrate']].dropna()
sub3.head()
```

[21]:

	oilperperson	relectricperperson	co2emissions	employrate
2	0.42	590.51	2932108666.67	50.50
6	0.64	768.43	5872119000.00	58.40
9	1.91	2825.39	12970092666.67	61.50
10	1.55	2068.12	4466084333.33	57.10
11	0.36	921.56	511107666.67	60.90

23 Centre employrate, oilperperson, relectricperperson and co2emissions

24 use sub3

[22]:
```
# hint lecture cell 14
sub3['employrate_c'] = (sub3['employrate'] - sub3['employrate'].mean())
sub3['oilperperson_c'] = (sub3['oilperperson'] - sub3['oilperperson'].mean())
sub3['relectricperperson_c'] = (sub3['relectricperperson'] -␣
↪sub3['relectricperperson'].mean())
sub3['co2emissions_c'] = (sub3['co2emissions'] - sub3['co2emissions'].mean())
```

25 Multiple and polynomial regression between oilperperson(x1) + co2emissions(x2) relectricperperson (x3 - order 2) and employrate (y)

26 use sub3

[26]:
```
# hint lecture cell 15
reg3 = smf.ols('employrate_c ~ oilperperson_c + co2emissions_c +␣
↪I(relectricperperson_c**2)', data=sub3).fit()
print (reg3.summary())
```

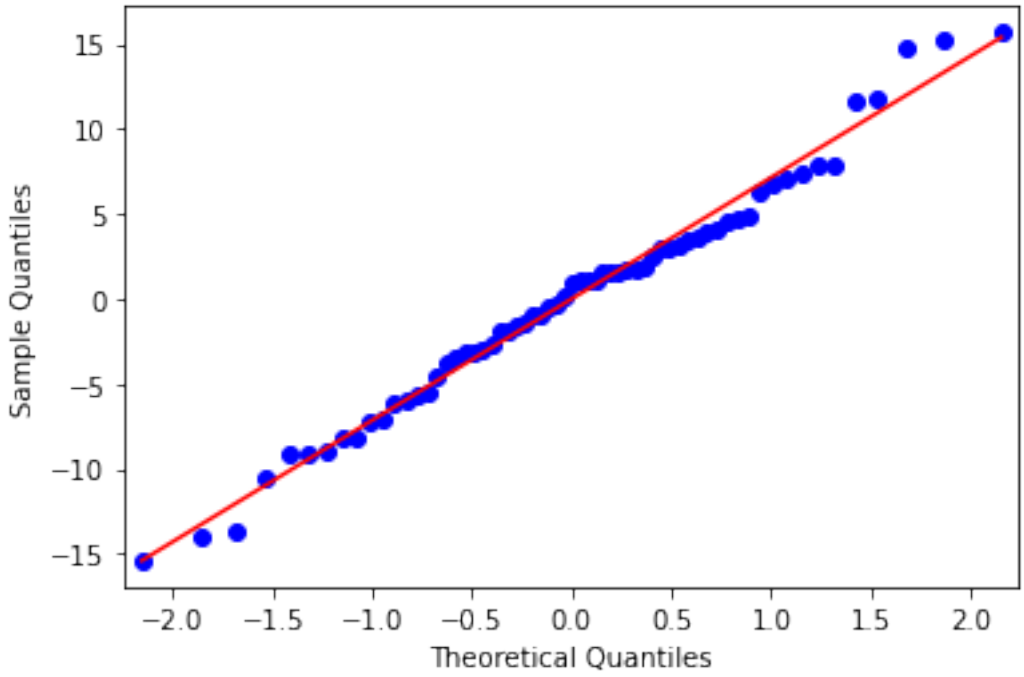
```
                            OLS Regression Results
==============================================================================
Dep. Variable:           employrate_c   R-squared:                       0.186
Model:                            OLS   Adj. R-squared:                  0.144
Method:                 Least Squares   F-statistic:                     4.481
Date:                Tue, 28 Dec 2021   Prob (F-statistic):           0.00670
Time:                        14:22:02   Log-Likelihood:                -210.13
```

```
No. Observations:                    63   AIC:                               428.3
Df Residuals:                        59   BIC:                               436.8
Df Model:                             3
Covariance Type:            nonrobust
================================================================================
================
                                   coef    std err          t      P>|t|
[0.025      0.975]
--------------------------------------------------------------------------------
---------------
Intercept                        -0.8489      0.937     -0.906      0.369
-2.724       1.026
oilperperson_c                    0.6155      0.522      1.179      0.243
-0.429       1.660
co2emissions_c                1.533e-11   2.01e-11      0.761      0.450
-2.5e-11    5.56e-11
I(relectricperperson_c ** 2)  2.047e-07   7.43e-08      2.755      0.008
5.6e-08    3.53e-07
==============================================================================
Omnibus:                        0.228   Durbin-Watson:                   2.324
Prob(Omnibus):                  0.892   Jarque-Bera (JB):                0.068
Skew:                           0.080   Prob(JB):                        0.967
Kurtosis:                       2.998   Cond. No.                     4.67e+10
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 4.67e+10. This might indicate that there are
strong multicollinearity or other numerical problems.
```

27 Evaluating model

28 Plot qqplot for the above regression (reg3)

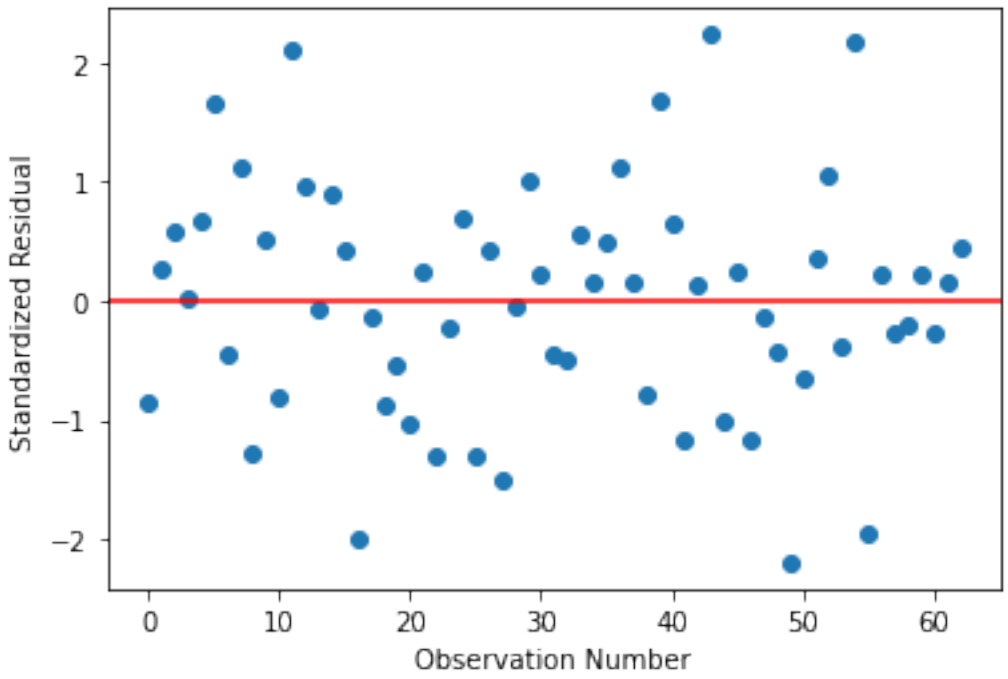
[27]:
```
# hint lecture cell 16
import statsmodels.api as sm
fig4=sm.qqplot(reg3.resid, line='r')
```



29 Residual plot for the above regression (reg3)

[28]:
```
# hint lecture cell 17
# simple plot of residuals
stdres=pd.DataFrame(reg3.resid_pearson)

plt.figure()
plt.plot(stdres, 'o', ls='None')
l = plt.axhline(y=0, color='r')
plt.ylabel('Standardized Residual')
plt.xlabel('Observation Number')
```

[28]:
```
Text(0.5, 0, 'Observation Number')
```



30 Calculate percentage of observations over 2 standardized deviation

[29]:
```
# hint lecture cell 18
percentage_over2sd = (np.count_nonzero( stdres[0] > 2) + np.count_nonzero(␣
 ↪stdres[0] < -2))
print (percentage_over2sd)
```

```
5
```

31 Calculate percentage of observations over 2.5 standardized deviation

[30]:
```
# hint lecture cell 19
percentage_over2_5sd = (np.count_nonzero( stdres[0] > 2.5) + np.count_nonzero(␣
 ↪stdres[0] < -2.5))
print (percentage_over2_5sd)
```

```
0
```

32 On your own

33 experiment with explanatory variable (oilperperson, co2emissions, relectricperperson) and their order to predict employrate

34 use sub3

[35]:
```
# hint lecture cell 15
reg4 = smf.ols('employrate_c ~ co2emissions_c + relectricperperson_c +␣
 ↪oilperperson_c', data=sub3).fit()
print (reg4.summary())
```

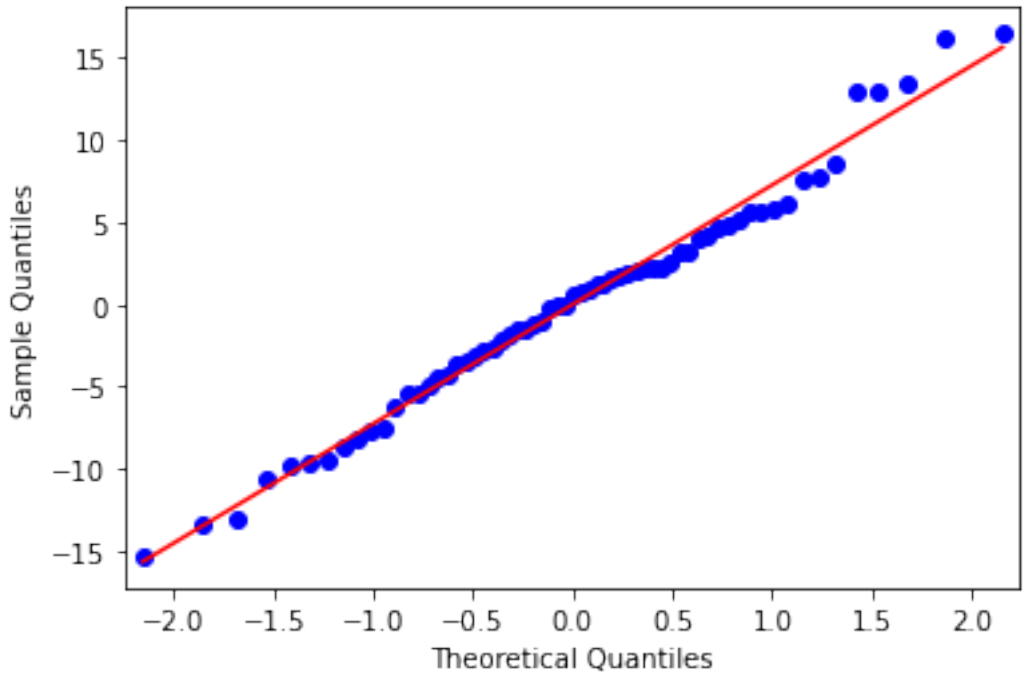
```
                            OLS Regression Results
==============================================================================
Dep. Variable:           employrate_c   R-squared:                       0.158
Model:                            OLS   Adj. R-squared:                  0.116
Method:                 Least Squares   F-statistic:                     3.703
Date:                Tue, 28 Dec 2021   Prob (F-statistic):             0.0165
Time:                        14:46:07   Log-Likelihood:                -211.16
No. Observations:                  63   AIC:                             430.3
Df Residuals:                      59   BIC:                             438.9
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
========
                             coef    std err          t      P>|t|      [0.025
0.975]
------------------------------------------------------------------------------
--------
Intercept              8.882e-16      0.900   9.87e-16      1.000      -1.800
1.800
co2emissions_c         7.613e-12   2.07e-11      0.369      0.714   -3.37e-11
4.89e-11
relectricperperson_c      0.0012      0.001      2.333      0.023       0.000
0.002
oilperperson_c            0.4017      0.584      0.688      0.494      -0.767
1.570
==============================================================================
Omnibus:                        0.417   Durbin-Watson:                   2.249
Prob(Omnibus):                  0.812   Jarque-Bera (JB):                0.238
Skew:                           0.150   Prob(JB):                        0.888
Kurtosis:                       2.990   Cond. No.                     4.40e+10
==============================================================================

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

```
specified.
[2] The condition number is large, 4.4e+10. This might indicate that there are
strong multicollinearity or other numerical problems.
```

35 Evaluate your model

36 Use qqplot

[36]:
```
# hint lecture cell 16
import statsmodels.api as sm
fig5=sm.qqplot(reg4.resid, line='r')
```



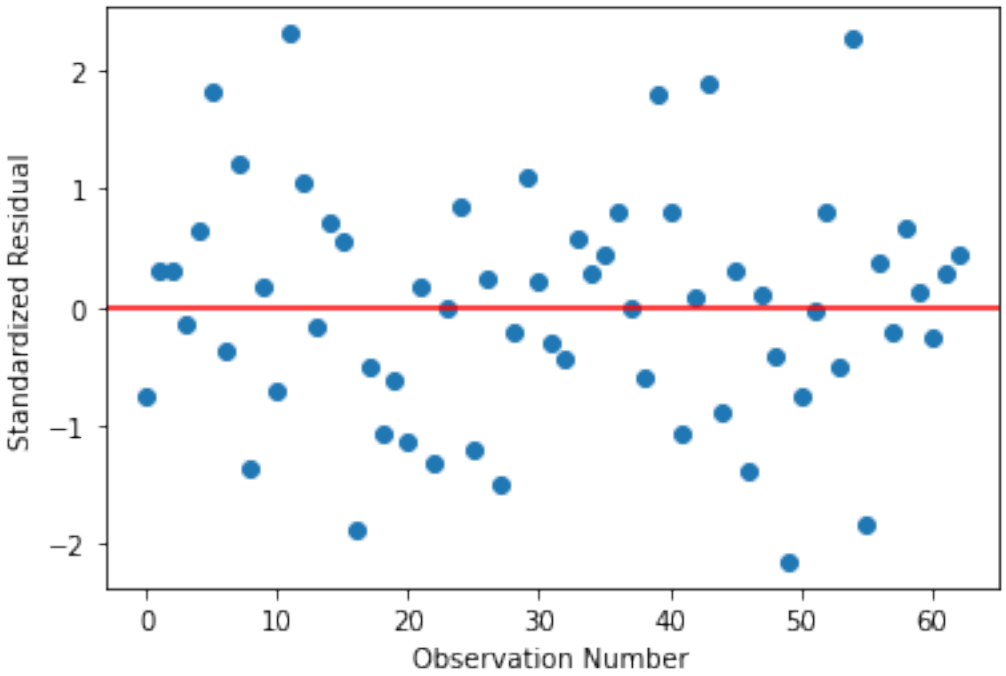
37 Evaluate your model

38 Use residual plot

[37]:
```
# hint lecture cell 17
# simple plot of residuals
stdres=pd.DataFrame(reg4.resid_pearson)

plt.figure()
plt.plot(stdres, 'o', ls='None')
l = plt.axhline(y=0, color='r')
```

```
plt.ylabel('Standardized Residual')
plt.xlabel('Observation Number')
```

[37]: Text(0.5, 0, 'Observation Number')



39 Calculate percentage of observations over 2 standardized deviation

[38]:
```
# hint lecture cell 18
percentage_over2sd = (np.count_nonzero( stdres[0] > 2) + np.count_nonzero(␣
 ↪stdres[0] < -2))
print (percentage_over2sd)
```

3

40 Calculate percentage of observations over 2.5 standardized deviation

[39]:
```
# hint lecture cell 19
percentage_over2_5sd = (np.count_nonzero( stdres[0] > 2.5) + np.count_nonzero(␣
 ↪stdres[0] < -2.5))
print (percentage_over2_5sd)
```

0