Jacob Miller - Homework 4

March 2, 2020

[1]:
```python
import pandas as pd
import numpy as np
import patsy
import statsmodels.api as sm
import scipy
import matplotlib.pyplot as plt
from astropy.table import Table
```

0.1 Problem 1

[2]:
```python
# Problem setup
SS_reg = 5550.8166
SS_tot = 5784.5426

table = Table([['Regression', 'Residual', 'Total'], # Source of Variation
               [SS_reg, '[  ]', SS_tot], # Sum of Squares
               ['[  ]', '[  ]', '[  ]'], # Degrees of Freedom
               ['[  ]', '[  ]', '    '], # Mean Square
               ['[  ]', '    ', '    '], # F0
               ['[  ]', '    ', '    ']], # P-value
              names = ('Source of Variation', 'Sum of Squares',
                       'Degrees of Freedom', 'Mean Square', 'F0', 'P-value'))

# Sum of Squares
SS_res = round(SS_tot - SS_reg, 4)

# Degrees of Freedom
DoF_reg = 2
DoF_tot = 25 - 1
DoF_res = round(DoF_tot - DoF_reg, 4)

# Mean Squares
MS_reg = round(SS_reg / DoF_reg, 4)
MS_res = round(SS_res / DoF_res, 4)

# F0
```

```
F0 = round(MS_reg / MS_res, 4)

# P-value
P = 1 - scipy.stats.f.cdf(F0, DoF_reg, DoF_res)

# Final table
table = Table([['Regression', 'Residual', 'Total'], # Source of Variation
               [SS_reg, SS_res, SS_tot], # Sum of Squares
               [DoF_reg, DoF_res, DoF_tot], # Degrees of Freedom
               [MS_reg, MS_res, ''], # Mean Square
               [F0, '', ''], # F0
               [P, '', '']], # P-value
              names = ('Source of Variation', 'Sum of Squares',
                       'Degrees of Freedom', 'Mean Square', 'F0', 'P-value'))
```

[3]:
```
print(table.to_pandas().to_string())
```

```
  Source of Variation  Sum of Squares  Degrees of Freedom Mean Square        F0
P-value
0          Regression       5550.8166                   2   2775.4083  261.2419
4.440892098500626e-16
1            Residual        233.7260                  22     10.6239
2               Total       5784.5426                  24
```

[4]:
```
print('--> Small P-value [{:e}] | Reject null hypothesis <--'.format(P))
```

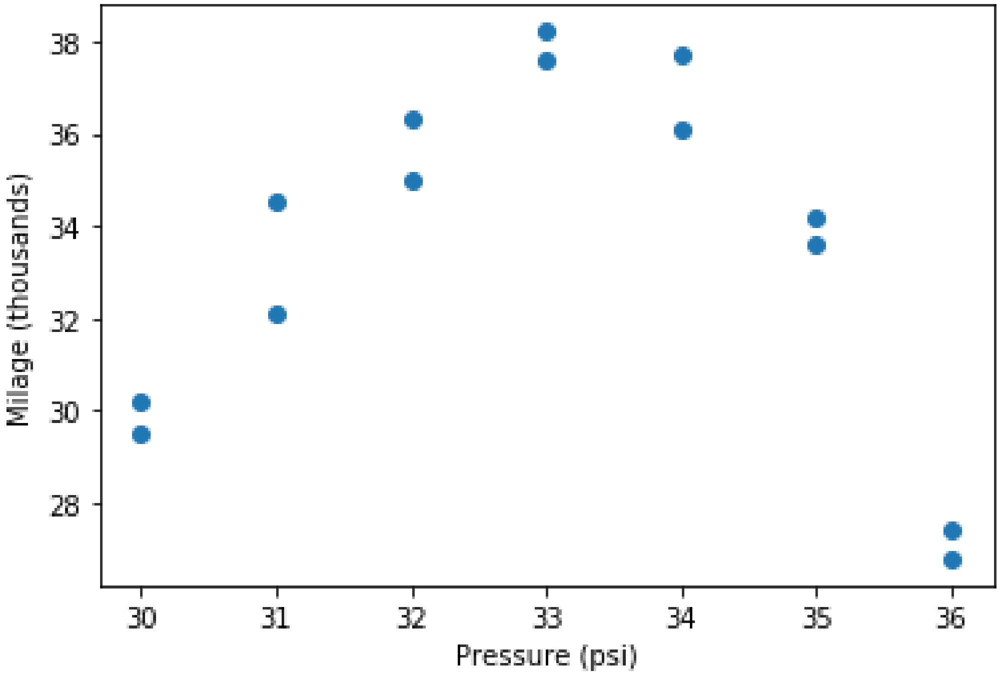
```
--> Small P-value [4.440892e-16] | Reject null hypothesis <--
```

1 Problem 2

[5]:
```
df = pd.DataFrame(data = {'Pressure_(x)': [30, 31, 32, 33, 34, 35, 36],
                          'Mileage_(y1)' : [29.5, 32.1, 36.3, 38.2,
                                            37.7, 33.6, 26.8],
                          'Mileage_(y2)' : [30.2, 34.5, 35.0, 37.6,
                                            36.1, 34.2, 27.4]}).\
                          set_index('Pressure_(x)')
```

1.0.1 Problem 2.a

[6]:
```
df_com = df[df.columns[0]].append(df[df.columns[1]])
plt.scatter(x = df_com.index.values, y = df_com)
plt.xlabel('Pressure (psi)')
plt.ylabel('Milage (thousands)')
plt.show()
print('--> Optimal tire pressure appears to be between 32 - 34 psi <--')
```



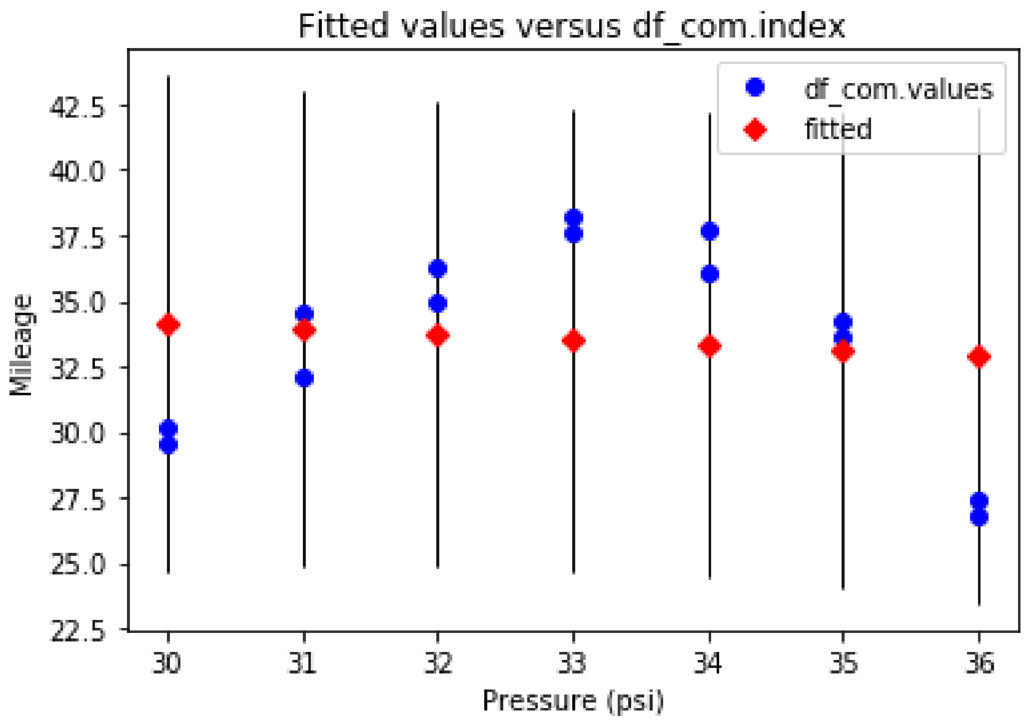

```
--> Optimal tire pressure appears to be between 32 - 34 psi <--
```

1.0.2 Problem 2.b

[7]:
```
y, X = patsy.dmatrices('df_com.values ~ df_com.index', df_com)

model = sm.OLS(y, X)
results = model.fit()
results.model.data.design_info = X.design_info

print('--> y = {} + {} * x <--'.format(*np.round((results.params[0],
                                                  results.params[1]),
                                                  2)))

fig, ax = plt.subplots()
fig = sm.graphics.plot_fit(results, 1, ax = ax)
ax.set_xlabel('Pressure (psi)')
```

```
ax.set_ylabel('Mileage')
plt.show()
```

```
--> y = 40.35 + -0.21 * x <--
```



1.0.3 Problem 2.c

[8]:
```
# Setup - because each X has exactly 2 y, can just append to dataframe
df['Mean'] = df.apply(lambda x: np.mean(x), axis = 1)
df['PE Sums'] = df.apply(lambda x: ((x['Mileage_(y1)'] - x['Mean']) ** 2 +
                                    (x['Mileage_(y2)'] - x['Mean']) ** 2),
                         axis = 1)

# Get beta-hat, start to calculate SS_reg
b_hat = np.matmul(np.matmul(np.linalg.inv(np.matmul(X.T, X)), X.T), y)
first_term = np.matmul(np.matmul(b_hat.T, X.T), y)
second_term = sum(y)**2 / len(y)

# Sum of Squares
SS_tot = np.round(float(sum(y**2) - sum(y)**2 / len(y)), 3)
SS_reg = np.round(float(first_term - second_term), 3)
SS_res = np.round(float(SS_tot - SS_reg), 3)
```

```python
SS_pe = np.round(float(sum(df['PE Sums'])), 3)
SS_lof = np.round(float(SS_res - SS_pe), 3)

# Degrees of Freedom
DoF_reg = 1
DoF_tot = len(df_com) - 1
DoF_res = DoF_tot - DoF_reg
DoF_pe = len(df) # because each X has exactly 2 y, this is just num of X's
DoF_lof = DoF_res - DoF_pe

# Mean Squares
MS_reg = np.round(float(SS_reg / DoF_reg), 3)
MS_res = np.round(float(SS_res / DoF_res), 3)
MS_pe = np.round(float(SS_pe / DoF_pe), 3)
MS_lof = np.round(float(SS_lof / DoF_lof), 3)

# F0
F0 = round(MS_lof / MS_pe, 3)

# P-value
P = round(1 - scipy.stats.f.cdf(F0, DoF_reg, DoF_res), 10)

table = Table([['Regression', 'Error', 'Lack of Fit', 'Pure Error', 'Total'],
               [SS_reg, SS_res, SS_lof, SS_pe, SS_tot], # Sum of Squares
               [DoF_reg, DoF_res, DoF_lof, DoF_pe, DoF_tot], # Degrees of␣
 ↪Freedom
               [MS_reg, MS_res, MS_lof, MS_pe, ''], # Mean Square
               ['', '', F0, '', ''], # F0
               ['', '', P, '', '']], # P-value
              names = ('Source of Variation', 'Sum of Squares',
                       'Degrees of Freedom', 'Mean Square', 'F0', 'P-value'))
```

[9]:
```python
print(table.to_pandas().to_string())
```

```
  Source of Variation  Sum of Squares  Degrees of Freedom Mean Square      F0
P-value
0          Regression           2.403                   1       2.403
1               Error         183.434                  12      15.286
2         Lack of Fit         177.644                   5      35.529  42.961
2.71169e-05
3          Pure Error           5.790                   7       0.827
4               Total         185.837                  13
```

1.0.4 Problem 2.d

[10]:
```
print('--> R = {} | Low R-squared suggests model does not fit data well <--'.\
      format(round(results.rsquared, 5)))
```

```
--> R = 0.01293 | Low R-squared suggests model does not fit data well <--
```

1.0.5 Problem 2.e

[11]:
```
print('--> P = {} | Reject hypothesis that model describes data <--'.format(P))
```

```
--> P = 2.71169e-05 | Reject hypothesis that model describes data <--
```

1.0.6 Problem 2.f

[12]:
```
print('--> Assumed first order linear regression, bad assumption <--')
```

```
--> Assumed first order linear regression, bad assumption <--
```

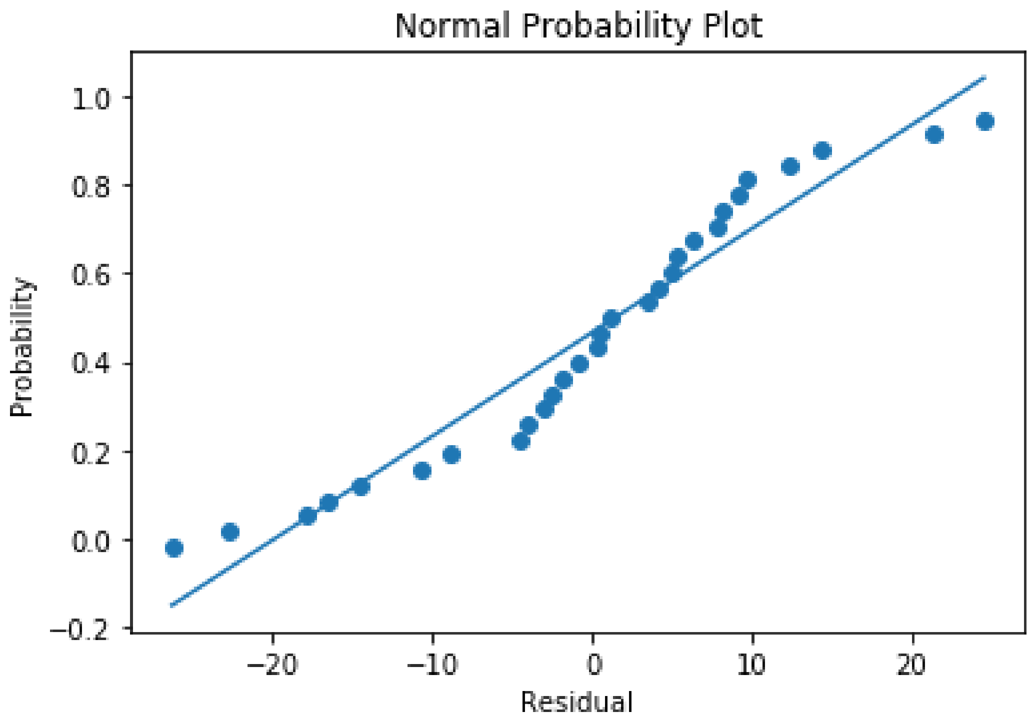
2 Problem 4.3

[13]:
```
df = pd.read_excel('Data/data-table-B2.xlsx')
y, X = patsy.dmatrices('y ~ x4', df)
model = sm.OLS(y, X)
results = model.fit()
results.model.data.design_info = X.design_info
```

2.0.1 Problem 4.3.a

[14]:
```
# Get residuals and probability for plot
residuals = results.resid
Prob = [(i - 1/2) / len(y) for i in range(len(y))]

# Calculate OLS using residuals to plot straight line. Get y values from model
resid_results = sm.OLS(Prob, sm.add_constant(sorted(residuals))).fit()
X_range = np.linspace(min(residuals), max(residuals), len(residuals))

# Normal Probability Plot + straight line
fig, ax = plt.subplots()
ax.scatter(sorted(residuals), Prob)
ax.plot(X_range, resid_results.params[0] + resid_results.params[1] * X_range)
ax.set_xlabel('Residual')
ax.set_ylabel('Probability')
```

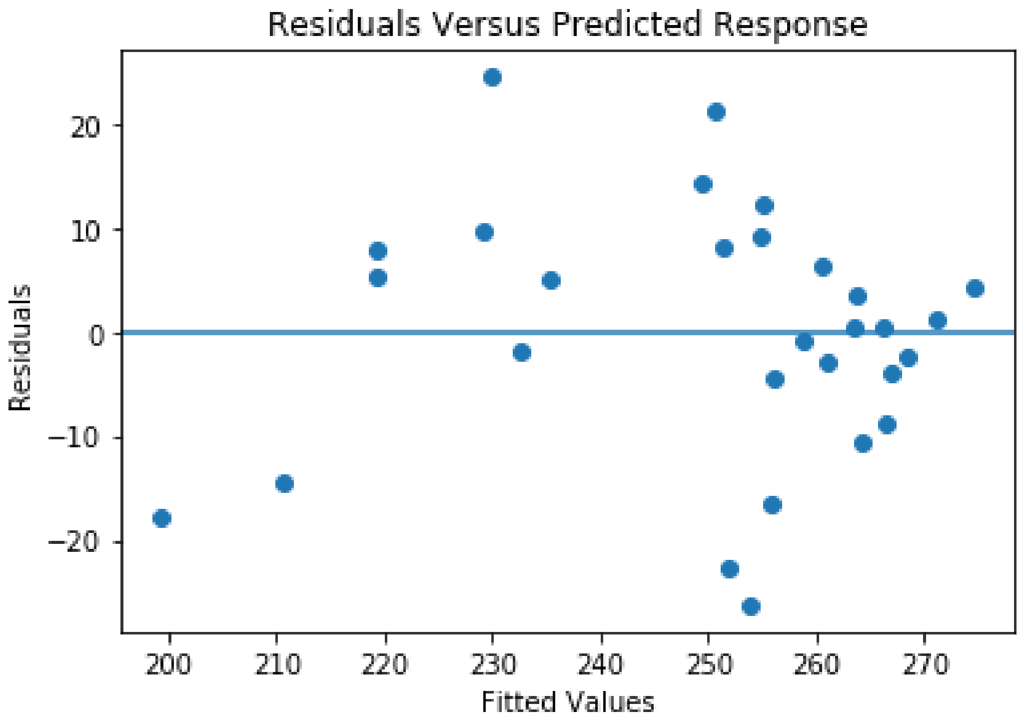
```
plt.title('Normal Probability Plot')
plt.show()

print('--> Minimal fluctuation suggests normality is ok <--')
```




```
--> Minimal fluctuation suggests normality is ok <--
```

2.0.2 Problem 4.3.b

[15]:
```
fig, ax = plt.subplots()
ax.scatter(results.fittedvalues, residuals)
ax.axhline(0)
ax.set_xlabel('Fitted Values')
ax.set_ylabel('Residuals')
plt.title('Residuals Versus Predicted Response')
plt.show()

print('--> Appears to be non-constant variance (funnel or double-bow) <--')
```




```
--> Appears to be non-constant variance (funnel or double-bow) <--
```

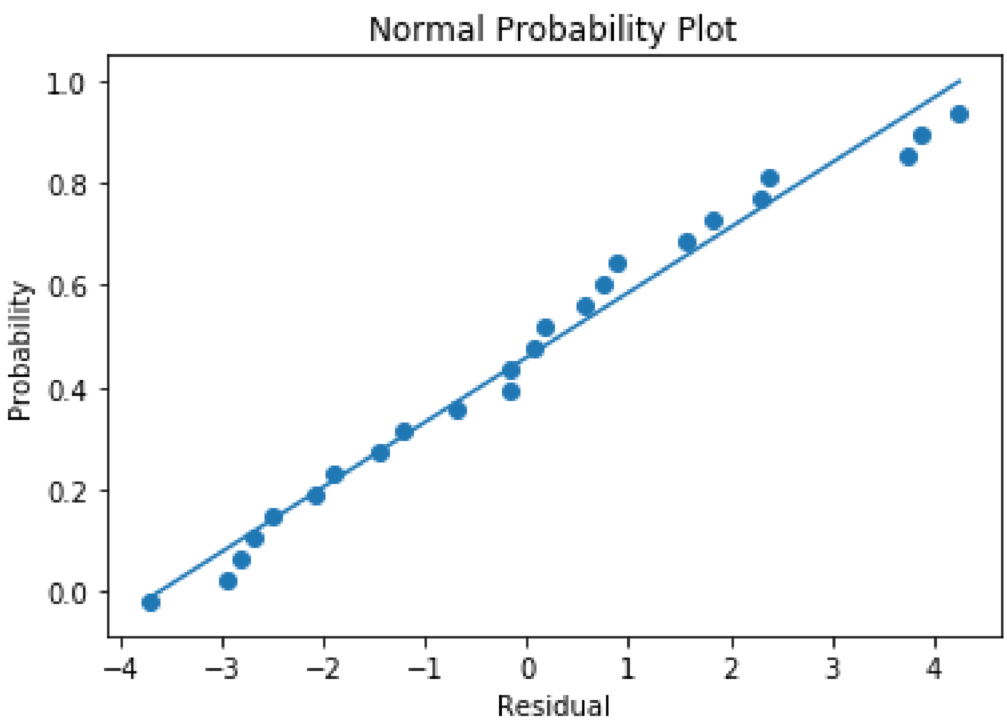
3 Problem 4.5

[16]:
```
df = pd.read_excel('Data/data-table-B4.xlsx')
y, X = patsy.dmatrices('y ~ x1 + x2 + x3 + x4 + x5 + x6 + x7 + x8 + x9', df)
model = sm.OLS(y, X)
results = model.fit()
results.model.data.design_info = X.design_info
```

3.0.1 Problem 4.5.a

[17]:
```
# Get residuals and probability for plot
residuals = results.resid
Prob = [(i - 1/2) / len(y) for i in range(len(y))]

# Calculate OLS using residuals to plot straight line. Get y values from model
resid_results = sm.OLS(Prob, sm.add_constant(sorted(residuals))).fit()
X_range = np.linspace(min(residuals), max(residuals), len(residuals))

# Normal Probability Plot + straight line
```

```
fig, ax = plt.subplots()
ax.scatter(sorted(residuals), Prob)
ax.plot(X_range, resid_results.params[0] + resid_results.params[1] * X_range)
ax.set_xlabel('Residual')
ax.set_ylabel('Probability')
plt.title('Normal Probability Plot')
plt.show()

print('--> Minimal fluctuation suggests normality is ok <--')
```

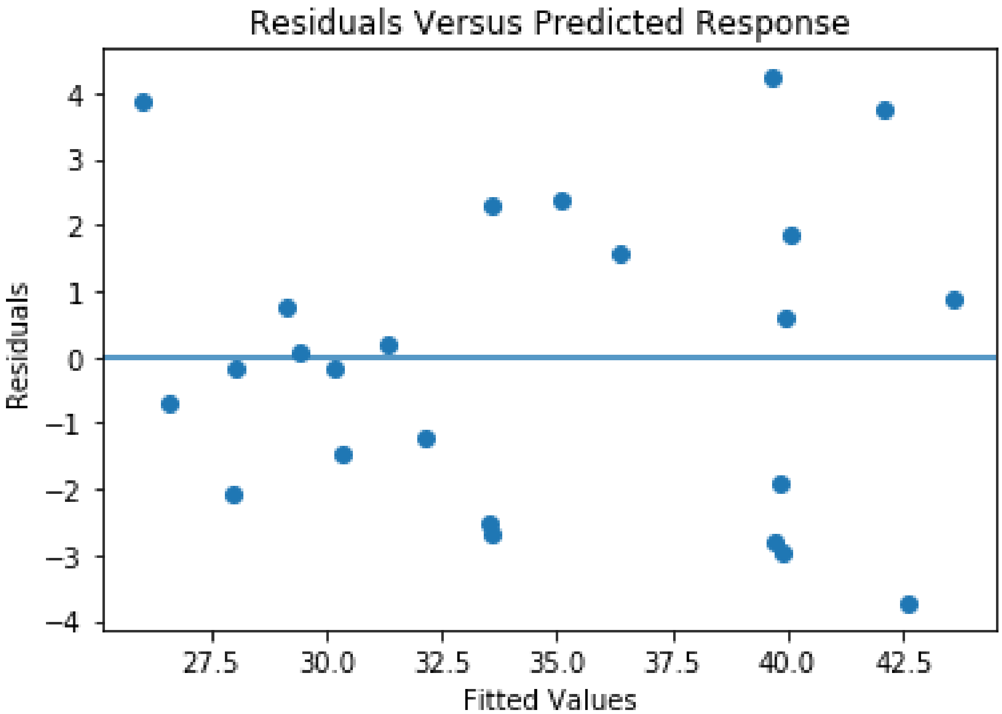



```
--> Minimal fluctuation suggests normality is ok <--
```

3.0.2 Problem 4.5.b

[18]:
```
fig, ax = plt.subplots()
ax.scatter(results.fittedvalues, residuals)
ax.axhline(0)
ax.set_xlabel('Fitted Values')
ax.set_ylabel('Residuals')
plt.title('Residuals Versus Predicted Response')
plt.show()

print('--> If outlier in top left corner is removed, plot has incline <--')
```

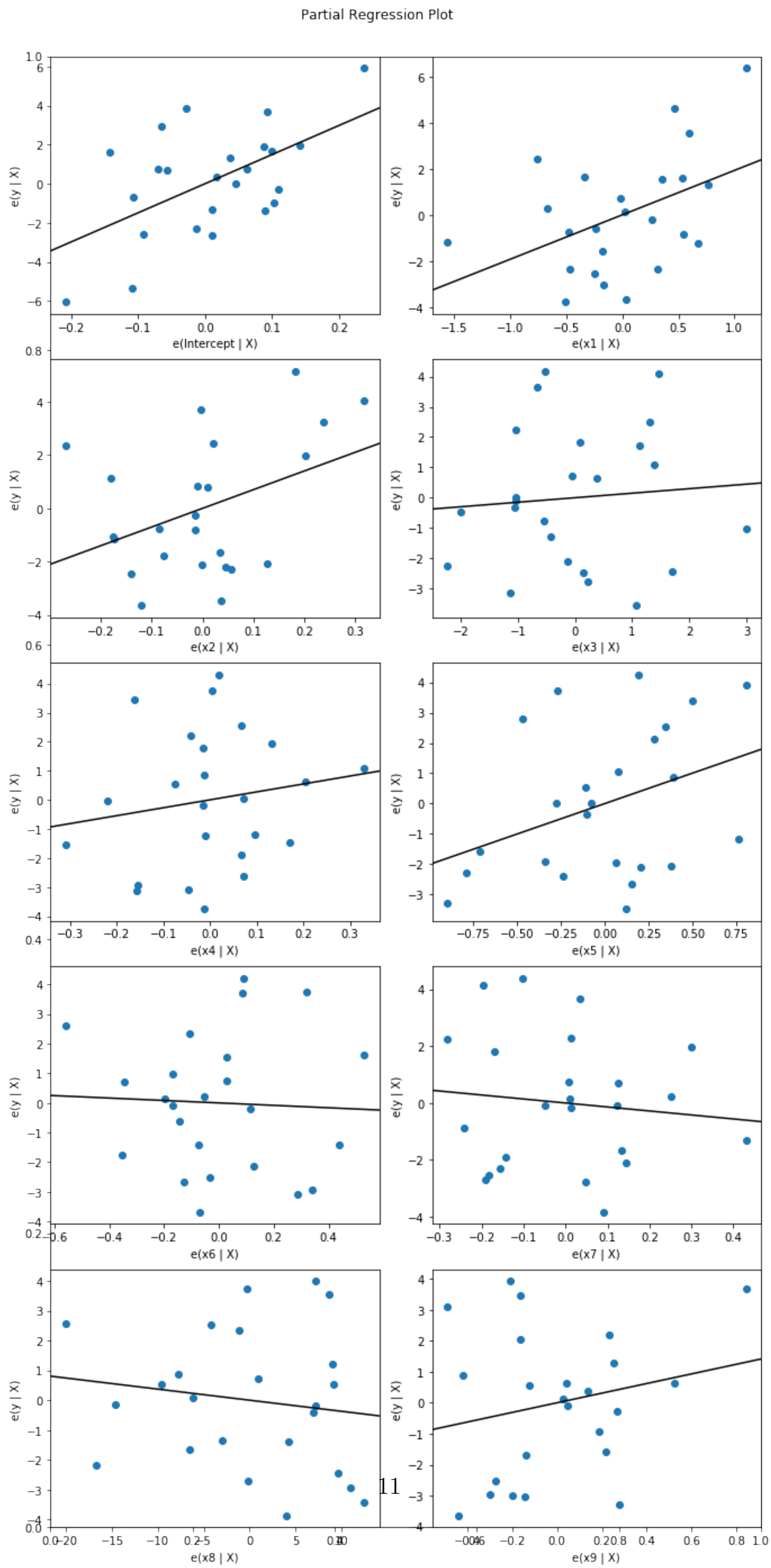
```
print('--> from lower left to top right. Otherwise no significant pattern <--')
```




```
--> If outlier in top left corner is removed, plot has incline <--
--> from lower left to top right. Otherwise no significant pattern <--
```

3.0.3 Problem 4.5.c

[19]:
```
fig, ax = plt.subplots(figsize = (10, 20))
fig = sm.graphics.plot_partregress_grid(results, fig = fig)
plt.show()
print('--> Intercept, x1 have large influence. x2 minimal influence. <--')
print('--> x3 and above appear to not have any influence <--')
```


Partial Regression Plot
e(y | X)
e(Intercept | X)
e(x1 | X)
e(x2 | X)
e(x3 | X)
e(x4 | X)
e(x5 | X)
e(x6 | X)
e(x7 | X)
e(x8 | X)
e(x9 | X)

```
--> Intercept, x1 have large influence. x2 minimal influence. <--
--> x3 and above appear to not have any influence <--
```

3.0.4 Problem 4.5.d

[20]:
```python
infl = results.get_influence()
print(infl.summary_table())
print('--> "Obs 15" (i.e. observation 16 because table is 0-based) <--')
print('--> has larger absolute value than others, likely outlier <--')
```

```
================================================================================
==================
       obs      endog     fitted     Cook's   student.   hat diag    dffits
ext.stud.     dffits
                           value          d   residual              internal
residual
--------------------------------------------------------------------------------
------------------
         0     29.500     29.429      0.000      0.032      0.441      0.029
0.031      0.028
         1     27.900     28.061      0.000     -0.063      0.244     -0.036
-0.060     -0.034
         2     25.900     27.989      0.014     -0.783      0.181     -0.368
-0.771     -0.363
         3     29.900     29.160      0.001      0.265      0.099      0.088
0.256      0.085
         4     29.900     26.028      0.142      1.627      0.349      1.190
1.741      1.274
         5     30.900     32.120      0.018     -0.532      0.395     -0.430
-0.518     -0.418
         6     28.900     30.362      0.232     -0.943      0.723     -1.525
-0.939     -1.518
         7     35.900     33.595      0.029      0.909      0.260      0.539
0.903      0.535
         8     31.500     31.321      0.000      0.076      0.362      0.057
0.073      0.055
         9     31.000     33.512      0.082     -1.102      0.403     -0.905
-1.111     -0.913
        10     30.900     33.599      0.057     -1.107      0.316     -0.753
-1.117     -0.760
        11     30.000     30.165      0.001     -0.082      0.538     -0.089
-0.079     -0.085
        12     36.900     39.861      0.127     -1.320      0.421     -1.127
-1.360     -1.160
```

```
         13     41.900     40.071      0.283      1.119      0.693      1.681
1.130      1.698
         14     40.500     39.921      0.005      0.263      0.442      0.234
0.254      0.226
         15     43.900     39.647      0.372      2.002      0.481      1.929
2.284      2.201
         16     37.500     35.116      0.156      1.179      0.530      1.250
1.197      1.270
         17     37.900     39.811      0.062     -0.872      0.448     -0.785
-0.864     -0.778
         18     44.500     43.609      0.010      0.389      0.398      0.316
0.377      0.307
         19     37.900     36.340      0.035      0.695      0.420      0.591
0.681      0.580
         20     38.900     42.620      0.107     -1.525      0.316     -1.036
-1.609     -1.094
         21     36.900     39.705      0.210     -1.380      0.525     -1.450
-1.431     -1.503
         22     45.800     42.066      0.097      1.511      0.297      0.983
1.591      1.035
         23     25.900     26.591      0.049     -0.441      0.717     -0.703
-0.428     -0.682
================================================================================
==================
--> "Obs 15" (i.e. observation 16 because table is 0-based) <--
--> has larger absolute value than others, likely outlier <--
```

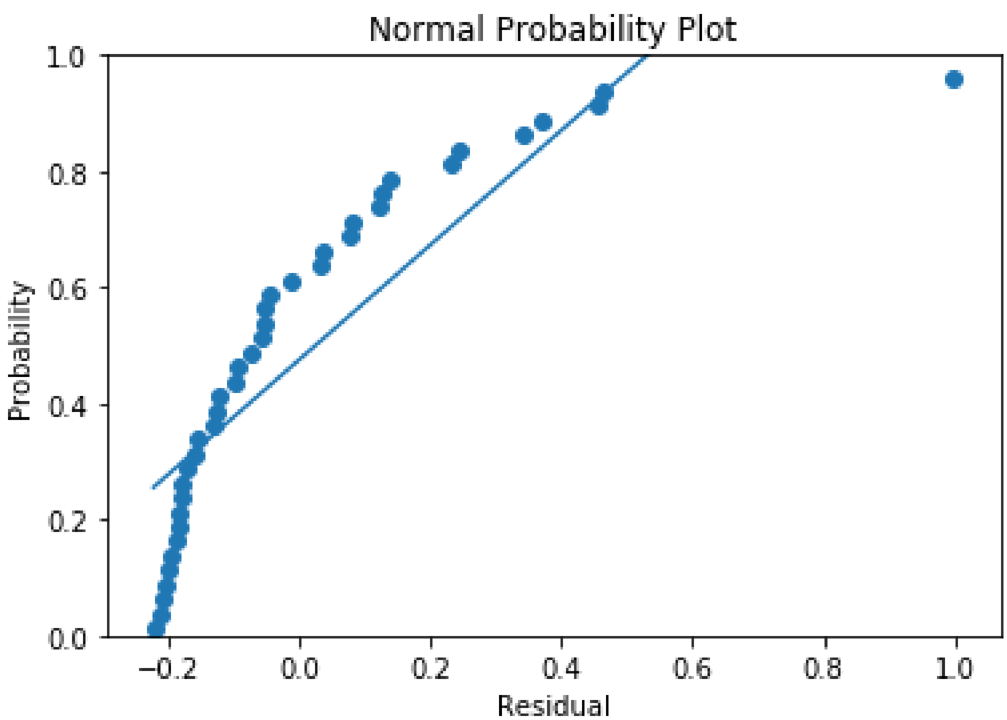
4 Problem 4.17

[21]:
```
df = pd.read_excel('Data/data-table-B10.xlsx')
y, X = patsy.dmatrices('y ~ x1 + x2', df)
model = sm.OLS(y, X)
results = model.fit()
results.model.data.design_info = X.design_info
```

4.0.1 Problem 4.17.a

[22]:
```
# Get residuals and probability for plot
residuals = results.resid
Prob = [(i - 1/2) / len(y) for i in range(len(y))]

# Calculate OLS using residuals to plot straight line. Get y values from model
resid_results = sm.OLS(Prob, sm.add_constant(sorted(residuals))).fit()
X_range = np.linspace(min(residuals), max(residuals), len(residuals))
```

```
# Normal Probability Plot + straight line
fig, ax = plt.subplots()
ax.scatter(sorted(residuals), Prob)
ax.plot(X_range, resid_results.params[0] + resid_results.params[1] * X_range)
ax.set_xlabel('Residual')
ax.set_ylabel('Probability')
ax.set_ylim(0, 1)
plt.title('Normal Probability Plot')
plt.show()

print('--> Normality plot exhibits negative skew, therefore problems <--')
```

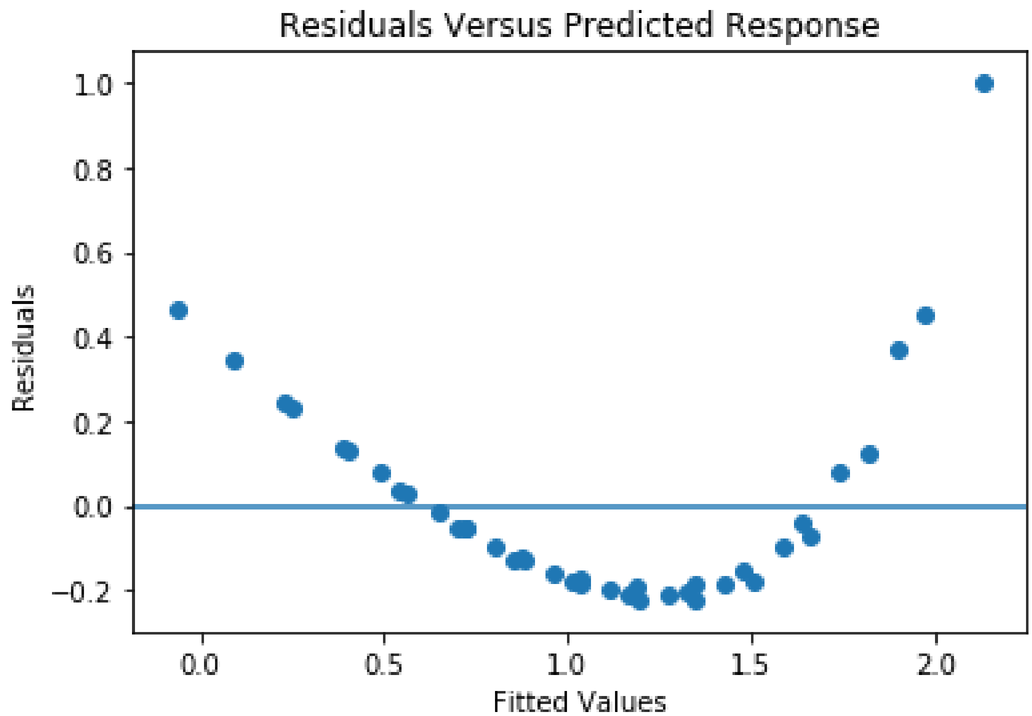



```
--> Normality plot exhibits negative skew, therefore problems <--
```

4.0.2 Problem 4.17.b

[23]:
```
fig, ax = plt.subplots()
ax.scatter(results.fittedvalues, residuals)
ax.axhline(0)
ax.set_xlabel('Fitted Values')
ax.set_ylabel('Residuals')
plt.title('Residuals Versus Predicted Response')
plt.show()
```

```
print('--> Definite non-linear pattern for residual vs. predicted plot <--')
```




```
--> Definite non-linear pattern for residual vs. predicted plot <--
```

4.0.3 Problem 4.17.c

[24]:
```
y2, X2 = patsy.dmatrices('y ~ x2', df)
model2 = sm.OLS(y2, X2)
results2 = model2.fit()
results2.model.data.design_info = X2.design_info

infl1 = results.get_influence()
infl2 = results2.get_influence()
press1 = infl1.resid_press
press2 = infl2.resid_press
r_squared_pred1 = 1 - \
    infl1.ess_press/sm.stats.anova_lm(results, typ = 1).sum_sq.sum()
r_squared_pred2 = 1 - \
    infl2.ess_press/sm.stats.anova_lm(results2, typ = 1).sum_sq.sum()

comp_press = pd.DataFrame({'Ordinary Residuals': residuals,
                           'Full Model (PRESS_1)': press1,
```

```python
                    'Viscosity to Temp Only (PRESS_2)': press2})
```

[25]:
```python
print(comp_press)
```

```
    Ordinary Residuals  Full Model (PRESS_1)  Viscosity to Temp Only (PRESS_2)
0             0.999076              1.145488                          1.506526
1             0.454365              0.506862                          0.886794
2             0.123653              0.135198                          0.524019
3            -0.074058             -0.079914                          0.313249
4            -0.178770             -0.191651                          0.203821
5            -0.221481             -0.237440                          0.159980
6            -0.221793             -0.239329                          0.160660
7            -0.187604             -0.205120                          0.198456
8            -0.130516             -0.145596                          0.263173
9            -0.055227             -0.063321                          0.352562
10            0.372160              0.410135                          0.567421
11            0.077448              0.083130                          0.238523
12           -0.096263             -0.101347                          0.052291
13           -0.182975             -0.190211                         -0.037924
14           -0.210686             -0.217648                         -0.066133
15           -0.200398             -0.207019                         -0.055572
16           -0.159609             -0.165922                         -0.013791
17           -0.094721             -0.099723                          0.053905
18           -0.013032             -0.013988                          0.142049
19            0.080256              0.088446                          0.247925
20           -0.042795             -0.047041                         -0.173574
21           -0.155507             -0.166500                         -0.289264
22           -0.205218             -0.215530                         -0.335759
23           -0.209330             -0.217084                         -0.335802
24           -0.180441             -0.185957                         -0.304061
25           -0.126153             -0.130009                         -0.248337
26           -0.051064             -0.052956                         -0.172336
27            0.036624              0.038465                         -0.082802
28            0.136413              0.146056                          0.021991
29            0.244301              0.268542                          0.140661
30           -0.182774             -0.211246                         -0.646409
31           -0.194985             -0.219217                         -0.642721
32           -0.171097             -0.188507                         -0.605512
33           -0.122608             -0.133304                         -0.547849
34           -0.052320             -0.056512                         -0.472294
35            0.031669              0.034206                         -0.386085
36            0.127757              0.138902                         -0.289256
37            0.230646              0.254115                         -0.185306
38            0.342634              0.385215                         -0.069492
39            0.464422              0.536768                          0.061965
```

[26]:

```
print('Full Model: PRESS Statistic = {} | R^2 (Pred) = {}%'.\
      format(round(infl1.ess_press, 3),
             round(100 * r_squared_pred1, 2)))
print('Partial Model: PRESS Statistic = {} | R^2 (Pred) = {}%'.\
      format(round(infl2.ess_press, 3),
             round(100 * r_squared_pred2, 2)))
print('--> Full Model: Lower PRESS, higher R^2 implies better prediction <--')
```

```
Full Model: PRESS Statistic = 3.112 | R^2 (Pred) = 77.75%
Partial Model: PRESS Statistic = 6.777 | R^2 (Pred) = 51.54%
--> Full Model: Lower PRESS, higher R^2 implies better prediction <--
```