Grading

The final score that you will receive for your programming assignment is generated in relation to the total points set in your programming assignment item—not the total point value in the nbgrader notebook.
When calculating the final score shown to learners, the programming assignment takes the percentage of earned points vs. the total points provided by nbgrader and returns a score matching the equivalent percentage of the point value for the programming assignment.
DO NOT CHANGE VARIABLE OR METHOD SIGNATURES The autograder will not work properly if your change the variable or method signatures.

Validate Button

Please note that this assignment uses nbgrader to facilitate grading. You will see a **validate button** at the top of your Jupyter notebook. If you hit this button, it will run tests cases for the lab that aren't hidden. It is good to use the validate button before submitting the lab. Do know that the labs in the course contain hidden test cases. The validate button will not let you know whether these test cases pass. After submitting your lab, you can see more information about these hidden test cases in the Grader Output.
Cells with longer execution times will cause the validate button to time out and freeze. Please know that if you run into Validate time-outs, it will not affect the final submission grading.

In [1]:
```
%matplotlib inline
import numpy as np
import scipy as sp
import scipy.stats as stats
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
# Set color map to have light blue background
sns.set()
import statsmodels.formula.api as smf
import statsmodels.api as sm
```

N.B.: I recommend that you use the `statsmodel` library to do the regression analysis as opposed to *e.g.* `sklearn`. The `sklearn` library is great for advanced topics, but it's easier to get lost in a sea of details and it's not needed for these problems.

1. Polynomial regression using MPG data [25 pts, Peer Review]

We will be using Auto MPG data from UCI datasets (https://archive.ics.uci.edu/ml/datasets/Auto+MPG) to study polynomial regression.

Processing math: 42%

In [51]:
```python
columns = ['mpg','cylinders','displacement','horsepower','weight','acceleration','model_year','origin','car_name']
df = pd.read_csv("data/auto-mpg.data", header=None, delimiter=r"\s+", names=columns)
print(df.info())
df.describe()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 398 entries, 0 to 397
Data columns (total 9 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   mpg           398 non-null    float64
 1   cylinders     398 non-null    int64  
 2   displacement  398 non-null    float64
 3   horsepower    398 non-null    object 
 4   weight        398 non-null    float64
 5   acceleration  398 non-null    float64
 6   model_year    398 non-null    int64  
 7   origin        398 non-null    int64  
 8   car_name      398 non-null    object 
dtypes: float64(4), int64(3), object(2)
memory usage: 28.1+ KB
None
```

Out[51]:

	mpg	cylinders	displacement	weight	acceleration	model_year	origin
count	398.000000	398.000000	398.000000	398.000000	398.000000	398.000000	398.000000
mean	23.514573	5.454774	193.425879	2970.424623	15.568090	76.010050	1.572864
std	7.815984	1.701004	104.269838	846.841774	2.757689	3.697627	0.802055
min	9.000000	3.000000	68.000000	1613.000000	8.000000	70.000000	1.000000
25%	17.500000	4.000000	104.250000	2223.750000	13.825000	73.000000	1.000000
50%	23.000000	4.000000	148.500000	2803.500000	15.500000	76.000000	1.000000
75%	29.000000	8.000000	262.000000	3608.000000	17.175000	79.000000	2.000000
max	46.600000	8.000000	455.000000	5140.000000	24.800000	82.000000	3.000000

1a) Clean the data (fix data types and remove null or undefined values) and drop the column car_name. [5 pts]

Replace the data frame with the cleaned data frame. Do not change the column names, and do not add new columns.

In [3]:
```python
# replace data frame with cleaned data frame
# fix data types, remove null or undefined values, drop the column car_name
# NOTE: do not change the column names or add new columns
# your code here
#url = 'http://archive.ics.uci.edu/ml/machine-learning-databases/auto-mpg/auto-mpg.data'
```

```
#columns = ['mpg','cylinders','displacement','horsepower','weight','accele
ration','model_year','origin','car_name']
#correctdf = pd.read_csv(url, names = columns, )
df = df.dropna(how = 'any', axis = 0)
a = [np.nan, None, [], {}, 'NaN', 'Null','NULL','None','NA','?','-', '.','
', ' ', '    ']

df = df[~df.isin(['?']).any(axis = 1)]
df.drop("car_name", axis = 1, inplace = True)
df['horsepower'] = df['horsepower'].astype(float, errors = 'raise')
print(df.info())
print(df['horsepower'])
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 392 entries, 0 to 397
Data columns (total 8 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   mpg           392 non-null    float64
 1   cylinders     392 non-null    int64  
 2   displacement  392 non-null    float64
 3   horsepower    392 non-null    float64
 4   weight        392 non-null    float64
 5   acceleration  392 non-null    float64
 6   model_year    392 non-null    int64  
 7   origin        392 non-null    int64  
dtypes: float64(5), int64(3)
memory usage: 27.6 KB
None
0      130.0
1      165.0
2      150.0
3      150.0
4      140.0
       ...  
393     86.0
394     52.0
395     84.0
396     79.0
397     82.0
Name: horsepower, Length: 392, dtype: float64
```

In [4]:
```
# this cell will test that you properly cleaned the dataframe
```

1b) Fit a simple linear regression model with a feature that maximizes R^2. [5 pts]

Which feature is the best predictor, and the resulting r-squared value? Update your answer below.

In [5]:
```
# your code here
#print(sm.add_constant(X))
fit_d = {}
for columns in [x for x in df.columns if x != 'mpg']:

    Y = df['mpg']
```

```python
    X = df[columns]

    X = sm.add_constant(X)

    model = sm.OLS(Y,X, missing = 'drop').fit()

    fit_d[columns] = model.rsquared

print(fit_d)

best_predictor='weight'
best_r_squared=0.6926304331206254
```

```
{'cylinders': 0.6046889889441246, 'displacement': 0.6482294003193044, 'hor
sepower': 0.6059482578894348, 'weight': 0.6926304331206254, 'acceleration'
: 0.1792070501562546, 'model_year': 0.33702781330962284, 'origin': 0.31946
09386689675}
```

In [6]:
```python
# this cell will test best_predictor and best_r_squared
```

1c) Using the feature found above (without normalizing), fit polynomial regression up to N=10 and report R^2. Which polynomial degree gives the best result? [10 pts]

Hint: For N-degree polynomial fit, you may have to include all orders upto N. Use a for loop instead of running it manually. The `statsmodels.formula.api` formula string can understand `np.power(x,n)` function to include a feature representing x^n.

In [7]:
```python
# return updated best_degree and best_r_squared
best_degree = 3
best_r_squared = 0.7151495948129549
# your code here
fit_c = {}


Y = df['mpg']

X = df['weight']

X = sm.add_constant(X)
formula = 'mpg ~ X'
for i in range(1, 10):
    j = i+1
    model = smf.ols(formula, data = df, missing = 'drop').fit()
    fit_c[j] = (model.rsquared)
    formula = formula + '+np.power(X,{})'.format(j)

```

In [8]:
```python
# this cell tests best_degree and best_r_squared
```

1d) Now, let's make a new feature called 'weight_norm' which is weight

normalized by the mean value. [5 pts]

Run training with polynomial models with polynomial degrees up to 20. Print out each polynomial degree and R^2 value. What do you observe from the result? What are the best_degree and best_r_qaured just based on R^2 value? Inspect model summary from each model. What is the highest order model that makes sense (fill the value for the sound_degree)?

In [9]:
```python
best_degree = 20
best_r_squared = 0.7244280571721451
sound_degree = 2

df['weight_norm'] = df['weight']/df['weight'].mean()
# your code here
fit_1d = {}
fit_fstat = {}

Y = df['mpg']

X = df['weight_norm']

X = sm.add_constant(X)
formula = 'mpg ~ X'
for i in range(1, 20):
    j = i+1
    model = smf.ols(formula, data = df, missing = 'drop').fit()
    print(model.summary())
    formula = formula + '+np.power(X,{})'.format(j)
```

```
                            OLS Regression Results                            
==============================================================================
====
Dep. Variable:                    mpg   R-squared:                          0
.693
Model:                            OLS   Adj. R-squared:                     0
.691
Method:                 Least Squares   F-statistic:                        4
38.3
Date:                Mon, 20 Jun 2022   Prob (F-statistic):             2.24e
-100
Time:                        19:48:14   Log-Likelihood:                   -11
30.0
No. Observations:                 392   AIC:                                2
266.
Df Residuals:                     389   BIC:                                2
278.
Df Model:                           2                                         

Covariance Type:            nonrobust                                         

==============================================================================
====
                 coef    std err          t      P>|t|      [0.025      0.
975]
------------------------------------------------------------------------------
----
```

```
Intercept     23.1083      0.400     57.792      0.000      22.322      23
.894
X[0]          23.1083      0.400     57.792      0.000      22.322      23
.894
X[1]         -22.7706      0.769    -29.607      0.000     -24.283     -21
.259
==========================================================================
====
Omnibus:                      41.682   Durbin-Watson:                    0
.808
Prob(Omnibus):                 0.000   Jarque-Bera (JB):                60
.039
Skew:                          0.727   Prob(JB):                      9.18
e-14
Kurtosis:                      4.251   Cond. No.                      1.15
e+15
==========================================================================
====

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is cor
rectly specified.
[2] The smallest eigenvalue is 8.98e-28. This might indicate that there ar
e
strong multicollinearity problems or that the design matrix is singular.
                            OLS Regression Results

==========================================================================
====
Dep. Variable:                   mpg   R-squared:                        0
.715
Model:                           OLS   Adj. R-squared:                   0
.714
Method:                Least Squares   F-statistic:                      4
88.3
Date:               Mon, 20 Jun 2022   Prob (F-statistic):           8.39e
-107
Time:                       19:48:14   Log-Likelihood:                 -11
15.1
No. Observations:                392   AIC:                              2
236.
Df Residuals:                    389   BIC:                              2
248.
Df Model:                          2

Covariance Type:           nonrobust

==========================================================================
===========
                        coef    std err          t      P>|t|      [0.025
     0.975]
--------------------------------------------------------------------------
-----------
Intercept            20.7518      0.998     20.800      0.000      18.790
     22.713
X[0]                 20.7518      0.998     20.800      0.000      18.790
     22.713
```

```
X[1]                 -55.0722      5.872     -9.379      0.000     -66.617
    -43.527
np.power(X, 2)[0]    20.7518      0.998     20.800      0.000      18.790
     22.713
np.power(X, 2)[1]    15.0418      2.713      5.545      0.000       9.709
     20.375
==========================================================================
====
Omnibus:                       53.804   Durbin-Watson:                   0
.770
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               93
.923
Skew:                           0.809   Prob(JB):                     4.03
e-21
Kurtosis:                       4.770   Cond. No.                     1.13
e+19
==========================================================================
====

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is cor
rectly specified.
[2] The smallest eigenvalue is 1.64e-35. This might indicate that there ar
e
strong multicollinearity problems or that the design matrix is singular.
                            OLS Regression Results

==========================================================================
====
Dep. Variable:                    mpg   R-squared:                       0
.715
Model:                            OLS   Adj. R-squared:                  0
.713
Method:                 Least Squares   F-statistic:                     3
24.7
Date:                Mon, 20 Jun 2022   Prob (F-statistic):          2.09e
-105
Time:                        19:48:14   Log-Likelihood:                -11
15.1
No. Observations:                 392   AIC:                             2
238.
Df Residuals:                     388   BIC:                             2
254.
Df Model:                           3

Covariance Type:            nonrobust

==========================================================================
===========
                        coef    std err          t      P>|t|      [0.025
     0.975]
--------------------------------------------------------------------------
-----------
Intercept            15.4238      2.761      5.587      0.000       9.996
     20.852
X[0]                 15.4238      2.761      5.587      0.000       9.996
     20.852
```

```
X[1]                -53.3874     32.500     -1.643      0.101    -117.285
     10.510
np.power(X, 2)[0]    15.4238      2.761      5.587      0.000       9.996
     20.852
np.power(X, 2)[1]    13.4357     30.592      0.439      0.661     -46.710
     73.582
np.power(X, 3)[0]    15.4238      2.761      5.587      0.000       9.996
     20.852
np.power(X, 3)[1]     0.4874      9.247      0.053      0.958     -17.692
     18.667
==========================================================================
====
Omnibus:                       53.711   Durbin-Watson:                   0
.770
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               93
.724
Skew:                           0.808   Prob(JB):                     4.45
e-21
Kurtosis:                       4.768   Cond. No.                     2.77
e+20
==========================================================================
====

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is cor
rectly specified.
[2] The smallest eigenvalue is 4.34e-38. This might indicate that there ar
e
strong multicollinearity problems or that the design matrix is singular.
                            OLS Regression Results

==========================================================================
====
Dep. Variable:                    mpg   R-squared:                       0
.715
Model:                            OLS   Adj. R-squared:                  0
.713
Method:                 Least Squares   F-statistic:                     2
43.3
Date:                Mon, 20 Jun 2022   Prob (F-statistic):          3.27e
-104
Time:                        19:48:15   Log-Likelihood:                -11
14.8
No. Observations:                 392   AIC:                             2
240.
Df Residuals:                     387   BIC:                             2
260.
Df Model:                           4

Covariance Type:            nonrobust

==========================================================================
===========
                        coef    std err          t      P>|t|      [0.025
     0.975]
--------------------------------------------------------------------------
-----------
```

```
Intercept              7.3708      7.727      0.954      0.341      -7.822
     22.563
X[0]                   7.3708      7.727      0.954      0.341      -7.822
     22.563
X[1]                  46.0811    151.766      0.304      0.762    -252.309
    344.471
np.power(X, 2)[0]      7.3708      7.727      0.954      0.341      -7.822
     22.563
np.power(X, 2)[1]  -130.3063    216.399     -0.602      0.547    -555.771
    295.158
np.power(X, 3)[0]      7.3708      7.727      0.954      0.341      -7.822
     22.563
np.power(X, 3)[1]    89.4576    132.917      0.673      0.501    -171.872
    350.788
np.power(X, 4)[0]      7.3708      7.727      0.954      0.341      -7.822
     22.563
np.power(X, 4)[1]   -19.9560     29.741     -0.671      0.503     -78.430
     38.518
==========================================================================
====
Omnibus:                       53.250   Durbin-Watson:                   0
.767
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               93
.572
Skew:                           0.799   Prob(JB):                     4.80
e-21
Kurtosis:                       4.782   Cond. No.                     7.87
e+31
==========================================================================
====

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is cor
rectly specified.
[2] The smallest eigenvalue is 8.72e-61. This might indicate that there ar
e
strong multicollinearity problems or that the design matrix is singular.
                            OLS Regression Results

==========================================================================
====
Dep. Variable:                    mpg   R-squared:                       0
.716
Model:                            OLS   Adj. R-squared:                  0
.712
Method:                 Least Squares   F-statistic:                     1
94.7
Date:                Mon, 20 Jun 2022   Prob (F-statistic):          3.59e
-103
Time:                        19:48:15   Log-Likelihood:                -11
14.4
No. Observations:                 392   AIC:                             2
241.
Df Residuals:                     386   BIC:                             2
265.
Df Model:                           5
```

```
Covariance Type:            nonrobust
==============================================================================
===========
                      coef    std err          t      P>|t|      [0.025
     0.975]
------------------------------------------------------------------------------
-----------
Intercept          -13.4724     22.382     -0.602      0.548     -57.478
     30.533
X[0]               -13.4724     22.382     -0.602      0.548     -57.478
     30.533
X[1]               639.8446    666.387      0.960      0.338    -670.358
   1950.047
np.power(X, 2)[0]   -13.4724     22.382     -0.602      0.548     -57.478
     30.533
np.power(X, 2)[1] -1291.1196   1286.875     -1.003      0.316   -3821.281
   1239.041
np.power(X, 3)[0]   -13.4724     22.382     -0.602      0.548     -57.478
     30.533
np.power(X, 3)[1]  1189.1977   1209.132      0.984      0.326   -1188.111
   3566.506
np.power(X, 4)[0]   -13.4724     22.382     -0.602      0.548     -57.478
     30.533
np.power(X, 4)[1]  -525.5006    553.262     -0.950      0.343   -1613.284
    562.283
np.power(X, 5)[0]   -13.4724     22.382     -0.602      0.548     -57.478
     30.533
np.power(X, 5)[1]    90.3757     98.763      0.915      0.361    -103.805
    284.556
==============================================================================
====
Omnibus:                       52.577   Durbin-Watson:                   0
.764
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               92
.901
Skew:                           0.788   Prob(JB):                     6.71
e-21
Kurtosis:                       4.790   Cond. No.                     9.50
e+23
==============================================================================
====

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is cor
rectly specified.
[2] The smallest eigenvalue is 1.07e-44. This might indicate that there ar
e
strong multicollinearity problems or that the design matrix is singular.
                            OLS Regression Results

==============================================================================
====
Dep. Variable:                    mpg   R-squared:                       0
.717
Model:                            OLS   Adj. R-squared:                  0
.712
```

```
Method:                 Least Squares   F-statistic:                     1
62.2
Date:                Mon, 20 Jun 2022   Prob (F-statistic):          3.84e
-102
Time:                        19:48:15   Log-Likelihood:                -11
14.1
No. Observations:                 392   AIC:                             2
242.
Df Residuals:                     385   BIC:                             2
270.
Df Model:                           6

Covariance Type:            nonrobust

==========================================================================
===========
                         coef    std err          t      P>|t|      [0.025
     0.975]
--------------------------------------------------------------------------
-----------
Intercept            -61.3147     65.344     -0.938      0.349    -189.790
     67.161
X[0]                 -61.3147     65.344     -0.938      0.349    -189.790
     67.161
X[1]                2766.6708   2751.333      1.006      0.315   -2642.849
   8176.191
np.power(X, 2)[0]   -61.3147     65.344     -0.938      0.349    -189.790
     67.161
np.power(X, 2)[1] -6555.5555   6731.550     -0.974      0.331   -1.98e+04
   6679.646
np.power(X, 3)[0]   -61.3147     65.344     -0.938      0.349    -189.790
     67.161
np.power(X, 3)[1]  7954.0686   8576.189      0.927      0.354   -8907.961
   2.48e+04
np.power(X, 4)[0]   -61.3147     65.344     -0.938      0.349    -189.790
     67.161
np.power(X, 4)[1] -5289.0658   6004.217     -0.881      0.379   -1.71e+04
   6516.094
np.power(X, 5)[0]   -61.3147     65.344     -0.938      0.349    -189.790
     67.161
np.power(X, 5)[1]  1835.1712   2192.083      0.837      0.403   -2474.782
   6145.124
np.power(X, 6)[0]   -61.3147     65.344     -0.938      0.349    -189.790
     67.161
np.power(X, 6)[1]  -260.0591    326.395     -0.797      0.426    -901.798
    381.680
==========================================================================
====
Omnibus:                       51.551   Durbin-Watson:                   0
.769
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               90
.102
Skew:                           0.779   Prob(JB):                     2.72
e-20
Kurtosis:                       4.758   Cond. No.                     2.15
e+33
==========================================================================
```

```
====

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is cor
rectly specified.
[2] The smallest eigenvalue is 4.17e-63. This might indicate that there ar
e
strong multicollinearity problems or that the design matrix is singular.
                            OLS Regression Results
==========================================================================
====
Dep. Variable:                    mpg   R-squared:                       0
.718
Model:                            OLS   Adj. R-squared:                  0
.713
Method:                 Least Squares   F-statistic:                     1
39.5
Date:                Mon, 20 Jun 2022   Prob (F-statistic):          2.23e
-101
Time:                        19:48:15   Log-Likelihood:                -11
13.2
No. Observations:                 392   AIC:                             2
242.
Df Residuals:                     384   BIC:                             2
274.
Df Model:                           7

Covariance Type:            nonrobust
==========================================================================
===========
                        coef    std err          t      P>|t|      [0.025
     0.975]
--------------------------------------------------------------------------
-----------
Intercept          -294.3489    195.056     -1.509      0.132    -677.860
     89.162
X[0]               -294.3489    195.056     -1.509      0.132    -677.860
     89.162
X[1]               1.651e+04    1.1e+04      1.501      0.134   -5111.991
   3.81e+04
np.power(X, 2)[0]  -294.3489    195.056     -1.509      0.132    -677.860
     89.162
np.power(X, 2)[1] -4.757e+04   3.25e+04     -1.464      0.144   -1.11e+05
   1.63e+04
np.power(X, 3)[0]  -294.3489    195.056     -1.509      0.132    -677.860
     89.162
np.power(X, 3)[1]  7.436e+04   5.22e+04      1.426      0.155   -2.82e+04
   1.77e+05
np.power(X, 4)[0]  -294.3489    195.056     -1.509      0.132    -677.860
     89.162
np.power(X, 4)[1] -6.833e+04   4.92e+04     -1.388      0.166   -1.65e+05
   2.84e+04
np.power(X, 5)[0]  -294.3489    195.056     -1.509      0.132    -677.860
     89.162
np.power(X, 5)[1]  3.695e+04   2.73e+04      1.354      0.177   -1.67e+04
```

```
   9.06e+04
np.power(X, 6)[0]  -294.3489    195.056     -1.509      0.132    -677.860
     89.162
np.power(X, 6)[1]  -1.09e+04   8247.471     -1.321      0.187   -2.71e+04
   5320.029
np.power(X, 7)[0]  -294.3489    195.056     -1.509      0.132    -677.860
     89.162
np.power(X, 7)[1]  1353.0270   1048.380      1.291      0.198    -708.256
   3414.310
==========================================================================
====
Omnibus:                       51.813   Durbin-Watson:                   0
.781
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               89
.789
Skew:                           0.786   Prob(JB):                     3.18
e-20
Kurtosis:                       4.740   Cond. No.                     2.09
e+35
==========================================================================
====

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is cor
rectly specified.
[2] The smallest eigenvalue is 9.75e-67. This might indicate that there ar
e
strong multicollinearity problems or that the design matrix is singular.
                            OLS Regression Results

==========================================================================
====
Dep. Variable:                    mpg   R-squared:                       0
.718
Model:                            OLS   Adj. R-squared:                  0
.712
Method:                 Least Squares   F-statistic:                     1
21.8
Date:                Mon, 20 Jun 2022   Prob (F-statistic):          2.72e
-100
Time:                        19:48:15   Log-Likelihood:                -11
13.2
No. Observations:                 392   AIC:                             2
244.
Df Residuals:                     383   BIC:                             2
280.
Df Model:                           8

Covariance Type:            nonrobust

==========================================================================
===========
                        coef    std err          t      P>|t|      [0.025
     0.975]
--------------------------------------------------------------------------
-----------
Intercept          -329.8289    576.364     -0.572      0.567   -1463.062
```

```
    803.404
X[0]               -329.8289    576.364     -0.572      0.567   -1463.062
    803.404
X[1]               2.153e+04    4.2e+04      0.513      0.608    -6.1e+04
   1.04e+05
np.power(X, 2)[0]  -329.8289    576.364     -0.572      0.567   -1463.062
    803.404
np.power(X, 2)[1]  -6.52e+04   1.46e+05     -0.447      0.655   -3.52e+05
   2.22e+05
np.power(X, 3)[0]  -329.8289    576.364     -0.572      0.567   -1463.062
    803.404
np.power(X, 3)[1]   1.09e+05   2.84e+05      0.384      0.701   -4.49e+05
   6.67e+05
np.power(X, 4)[0]  -329.8289    576.364     -0.572      0.567   -1463.062
    803.404
np.power(X, 4)[1] -1.099e+05   3.39e+05     -0.324      0.746   -7.76e+05
   5.56e+05
np.power(X, 5)[0]  -329.8289    576.364     -0.572      0.567   -1463.062
    803.404
np.power(X, 5)[1]  6.827e+04   2.54e+05      0.269      0.788   -4.31e+05
   5.68e+05
np.power(X, 6)[0]  -329.8289    576.364     -0.572      0.567   -1463.062
    803.404
np.power(X, 6)[1] -2.536e+04   1.17e+05     -0.217      0.828   -2.55e+05
   2.04e+05
np.power(X, 7)[0]  -329.8289    576.364     -0.572      0.567   -1463.062
    803.404
np.power(X, 7)[1]  5097.9850   3.02e+04      0.169      0.866   -5.43e+04
   6.45e+04
np.power(X, 8)[0]  -329.8289    576.364     -0.572      0.567   -1463.062
    803.404
np.power(X, 8)[1]  -416.7309   3358.984     -0.124      0.901   -7021.089
   6187.627
==========================================================================
====
Omnibus:                       52.013   Durbin-Watson:                   0
.783
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               90
.251
Skew:                           0.788   Prob(JB):                     2.52
e-20
Kurtosis:                       4.744   Cond. No.                     8.95
e+37
==========================================================================
====

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is cor
rectly specified.
[2] The smallest eigenvalue is 1.23e-71. This might indicate that there ar
e
strong multicollinearity problems or that the design matrix is singular.
                            OLS Regression Results

==========================================================================
====
Dep. Variable:                    mpg   R-squared:                       0
```

```
.718
Model:                            OLS   Adj. R-squared:                  0
.712
Method:                 Least Squares   F-statistic:                     1
08.2
Date:                Mon, 20 Jun 2022   Prob (F-statistic):           2.36
e-99
Time:                        19:48:15   Log-Likelihood:                -11
12.9
No. Observations:                 392   AIC:                             2
246.
Df Residuals:                     382   BIC:                             2
286.
Df Model:                           9

Covariance Type:            nonrobust

==========================================================================
===========
                         coef    std err          t      P>|t|      [0.025
     0.975]
--------------------------------------------------------------------------
-----------
Intercept            989.6135   1803.907      0.549      0.584   -2557.217
   4536.444
X[0]                 989.6114   1803.904      0.549      0.584   -2557.213
   4536.436
X[1]               -9.741e+04   1.65e+05     -0.590      0.556   -4.22e+05
   2.27e+05
np.power(X, 2)[0]   989.6103   1803.903      0.549      0.584   -2557.211
   4536.432
np.power(X, 2)[1]  4.141e+05    6.6e+05      0.627      0.531   -8.84e+05
   1.71e+06
np.power(X, 3)[0]   989.6103   1803.903      0.549      0.584   -2557.211
   4536.432
np.power(X, 3)[1] -9.961e+05   1.51e+06     -0.659      0.510   -3.97e+06
   1.97e+06
np.power(X, 4)[0]   989.6103   1803.903      0.549      0.584   -2557.211
   4536.432
np.power(X, 4)[1]  1.497e+06   2.18e+06      0.685      0.494    -2.8e+06
   5.79e+06
np.power(X, 5)[0]   989.6103   1803.903      0.549      0.584   -2557.211
   4536.432
np.power(X, 5)[1] -1.461e+06   2.07e+06     -0.706      0.481   -5.53e+06
   2.61e+06
np.power(X, 6)[0]   989.6103   1803.903      0.549      0.584   -2557.211
   4536.432
np.power(X, 6)[1]  9.275e+05   1.29e+06      0.722      0.471    -1.6e+06
   3.45e+06
np.power(X, 7)[0]   989.6103   1803.903      0.549      0.584   -2557.211
   4536.432
np.power(X, 7)[1]   -3.7e+05   5.05e+05     -0.733      0.464   -1.36e+06
   6.22e+05
np.power(X, 8)[0]   989.6103   1803.903      0.549      0.584   -2557.211
   4536.432
np.power(X, 8)[1]  8.429e+04   1.14e+05      0.741      0.459   -1.39e+05
   3.08e+05
```

```
np.power(X, 9)[0]   989.6096   1803.902      0.549      0.584   -2557.210
   4536.429
np.power(X, 9)[1] -8365.4797   1.12e+04     -0.745      0.457   -3.05e+04
   1.37e+04
==============================================================================
====
Omnibus:                       50.843   Durbin-Watson:                   0
.777
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               86
.844
Skew:                           0.779   Prob(JB):                     1.39
e-19
Kurtosis:                       4.700   Cond. No.                     7.10
e+37
==============================================================================
====

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is cor
rectly specified.
[2] The smallest eigenvalue is 4.73e-71. This might indicate that there ar
e
strong multicollinearity problems or that the design matrix is singular.
                            OLS Regression Results

==============================================================================
====
Dep. Variable:                    mpg   R-squared:                       0
.720
Model:                            OLS   Adj. R-squared:                  0
.713
Method:                 Least Squares   F-statistic:                     9
7.92
Date:                Mon, 20 Jun 2022   Prob (F-statistic):           8.13
e-99
Time:                        19:48:15   Log-Likelihood:                -11
11.8
No. Observations:                 392   AIC:                             2
246.
Df Residuals:                     381   BIC:                             2
289.
Df Model:                          10

Covariance Type:            nonrobust

==============================================================================
============
                         coef    std err          t      P>|t|      [0.025
      0.975]
------------------------------------------------------------------------------
------------
Intercept           9515.6584   5925.364      1.606      0.109   -2134.851
    2.12e+04
X[0]                9515.4045   5925.206      1.606      0.109   -2134.795
    2.12e+04
X[1]               -1.071e+06   6.64e+05     -1.612      0.108   -2.38e+06
    2.35e+05
```

```
np.power(X, 2)[0]   9515.3785   5925.190      1.606      0.109   -2134.789
    2.12e+04
np.power(X, 2)[1]   4.839e+06       3e+06      1.614      0.107   -1.06e+06
    1.07e+07
np.power(X, 3)[0]   9515.3781   5925.190      1.606      0.109   -2134.789
    2.12e+04
np.power(X, 3)[1]   -1.27e+07   7.88e+06     -1.611      0.108   -2.82e+07
     2.8e+06
np.power(X, 4)[0]   9515.3780   5925.190      1.606      0.109   -2134.789
    2.12e+04
np.power(X, 4)[1]   2.147e+07   1.34e+07      1.605      0.109   -4.84e+06
    4.78e+07
np.power(X, 5)[0]   9515.3780   5925.190      1.606      0.109   -2134.789
    2.12e+04
np.power(X, 5)[1]  -2.444e+07   1.53e+07     -1.594      0.112   -5.46e+07
     5.7e+06
np.power(X, 6)[0]   9515.3780   5925.190      1.606      0.109   -2134.789
    2.12e+04
np.power(X, 6)[1]   1.899e+07    1.2e+07      1.582      0.115   -4.62e+06
    4.26e+07
np.power(X, 7)[0]   9515.3780   5925.190      1.606      0.109   -2134.789
    2.12e+04
np.power(X, 7)[1]  -9.949e+06   6.35e+06     -1.566      0.118   -2.24e+07
    2.54e+06
np.power(X, 8)[0]   9515.3780   5925.190      1.606      0.109   -2134.789
    2.12e+04
np.power(X, 8)[1]   3.367e+06   2.17e+06      1.550      0.122   -9.05e+05
    7.64e+06
np.power(X, 9)[0]   9515.3780   5925.190      1.606      0.109   -2134.789
    2.12e+04
np.power(X, 9)[1]  -6.651e+05   4.34e+05     -1.532      0.126   -1.52e+06
    1.89e+05
np.power(X, 10)[0]  9515.4286   5925.221      1.606      0.109   -2134.800
    2.12e+04
np.power(X, 10)[1]  5.828e+04   3.85e+04      1.513      0.131   -1.75e+04
    1.34e+05
==========================================================================
====
Omnibus:                       51.509   Durbin-Watson:                   0
.772
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               88
.045
Skew:                           0.787   Prob(JB):                     7.61
e-20
Kurtosis:                       4.706   Cond. No.                     8.06
e+38
==========================================================================
====

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is cor
rectly specified.
[2] The smallest eigenvalue is 9.13e-73. This might indicate that there ar
e
strong multicollinearity problems or that the design matrix is singular.
                            OLS Regression Results
```

```
==============================================================================
====
Dep. Variable:                    mpg   R-squared:                       0
.721
Model:                            OLS   Adj. R-squared:                  0
.713
Method:                 Least Squares   F-statistic:                     8
9.23
Date:                Mon, 20 Jun 2022   Prob (F-statistic):           4.16
e-98
Time:                        19:48:15   Log-Likelihood:                -11
11.1
No. Observations:                 392   AIC:                             2
246.
Df Residuals:                     380   BIC:                             2
294.
Df Model:                          11

Covariance Type:            nonrobust

==============================================================================
============
                         coef    std err          t      P>|t|      [0.025
      0.975]
------------------------------------------------------------------------------
------------
Intercept           3.178e+04   2.03e+04      1.564      0.119   -8167.123
    7.17e+04
X[0]                3.181e+04   2.03e+04      1.564      0.119   -8180.735
    7.18e+04
X[1]               -4.206e+06   2.74e+06     -1.533      0.126    -9.6e+06
    1.19e+06
np.power(X, 2)[0]    3.18e+04   2.03e+04      1.564      0.119   -8176.964
    7.18e+04
np.power(X, 2)[1]   2.071e+07   1.38e+07      1.500      0.134   -6.43e+06
    4.79e+07
np.power(X, 3)[0]    3.18e+04   2.03e+04      1.564      0.119   -8176.945
    7.18e+04
np.power(X, 3)[1]  -6.017e+07   4.11e+07     -1.465      0.144   -1.41e+08
    2.06e+07
np.power(X, 4)[0]    3.18e+04   2.03e+04      1.564      0.119   -8176.945
    7.18e+04
np.power(X, 4)[1]   1.146e+08   8.02e+07      1.429      0.154   -4.31e+07
    2.72e+08
np.power(X, 5)[0]    3.18e+04   2.03e+04      1.564      0.119   -8176.945
    7.18e+04
np.power(X, 5)[1]  -1.505e+08   1.08e+08     -1.392      0.165   -3.63e+08
    6.21e+07
np.power(X, 6)[0]    3.18e+04   2.03e+04      1.564      0.119   -8176.945
    7.18e+04
np.power(X, 6)[1]   1.389e+08   1.03e+08      1.355      0.176   -6.27e+07
    3.41e+08
np.power(X, 7)[0]    3.18e+04   2.03e+04      1.564      0.119   -8176.945
    7.18e+04
np.power(X, 7)[1]  -9.031e+07   6.85e+07     -1.318      0.188   -2.25e+08
    4.44e+07
np.power(X, 8)[0]    3.18e+04   2.03e+04      1.564      0.119   -8176.945
```

```
    7.18e+04
np.power(X, 8)[1]   4.052e+07   3.16e+07      1.282      0.201   -2.16e+07
    1.03e+08
np.power(X, 9)[0]    3.18e+04   2.03e+04      1.564      0.119   -8176.945
    7.18e+04
np.power(X, 9)[1]  -1.196e+07    9.6e+06     -1.246      0.214   -3.08e+07
    6.91e+06
np.power(X, 10)[0]   3.18e+04   2.03e+04      1.564      0.119   -8176.945
    7.18e+04
np.power(X, 10)[1]   2.09e+06   1.73e+06      1.211      0.227    -1.3e+06
    5.48e+06
np.power(X, 11)[0]  3.181e+04   2.03e+04      1.564      0.119   -8179.332
    7.18e+04
np.power(X, 11)[1]  -1.64e+05   1.39e+05     -1.178      0.240   -4.38e+05
     1.1e+05
==========================================================================
====
Omnibus:                       54.966   Durbin-Watson:                   0
.770
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               96
.904
Skew:                           0.820   Prob(JB):                     9.07
e-22
Kurtosis:                       4.800   Cond. No.                     2.63
e+41
==========================================================================
====

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is cor
rectly specified.
[2] The smallest eigenvalue is 2.17e-77. This might indicate that there ar
e
strong multicollinearity problems or that the design matrix is singular.
                            OLS Regression Results

==========================================================================
====
Dep. Variable:                    mpg   R-squared:                       0
.721
Model:                            OLS   Adj. R-squared:                  0
.712
Method:                 Least Squares   F-statistic:                     8
1.59
Date:                Mon, 20 Jun 2022   Prob (F-statistic):           3.98
e-97
Time:                        19:48:16   Log-Likelihood:                -11
11.0
No. Observations:                 392   AIC:                             2
248.
Df Residuals:                     379   BIC:                             2
300.
Df Model:                          12

Covariance Type:            nonrobust

==========================================================================
```

```
============
                          coef    std err          t      P>|t|      [0.025
      0.975]
--------------------------------------------------------------------------------
------------
Intercept            1.861e+04    7.2e+04      0.259      0.796   -1.23e+05
     1.6e+05
X[0]                 1.862e+04   7.23e+04      0.257      0.797   -1.24e+05
    1.61e+05
X[1]                -2.479e+06   1.15e+07     -0.215      0.830   -2.52e+07
    2.02e+07
np.power(X, 2)[0]    1.862e+04   7.22e+04      0.258      0.797   -1.23e+05
    1.61e+05
np.power(X, 2)[1]    1.106e+07   6.42e+07      0.172      0.863   -1.15e+08
    1.37e+08
np.power(X, 3)[0]    1.862e+04   7.22e+04      0.258      0.797   -1.23e+05
    1.61e+05
np.power(X, 3)[1]   -2.793e+07   2.13e+08     -0.131      0.896   -4.47e+08
    3.91e+08
np.power(X, 4)[0]    1.862e+04   7.22e+04      0.258      0.797   -1.23e+05
    1.61e+05
np.power(X, 4)[1]      4.3e+07   4.72e+08      0.091      0.927   -8.85e+08
    9.71e+08
np.power(X, 5)[0]    1.862e+04   7.22e+04      0.258      0.797   -1.23e+05
    1.61e+05
np.power(X, 5)[1]   -3.887e+07   7.33e+08     -0.053      0.958   -1.48e+09
     1.4e+09
np.power(X, 6)[0]    1.862e+04   7.22e+04      0.258      0.797   -1.23e+05
    1.61e+05
np.power(X, 6)[1]     1.39e+07   8.18e+08      0.017      0.986    -1.6e+09
    1.62e+09
np.power(X, 7)[0]    1.862e+04   7.22e+04      0.258      0.797   -1.23e+05
    1.61e+05
np.power(X, 7)[1]    1.124e+07   6.63e+08      0.017      0.986   -1.29e+09
    1.31e+09
np.power(X, 8)[0]    1.862e+04   7.22e+04      0.258      0.797   -1.23e+05
    1.61e+05
np.power(X, 8)[1]   -1.883e+07   3.87e+08     -0.049      0.961   -7.79e+08
    7.41e+08
np.power(X, 9)[0]    1.862e+04   7.22e+04      0.258      0.797   -1.23e+05
    1.61e+05
np.power(X, 9)[1]    1.239e+07   1.58e+08      0.078      0.938   -2.99e+08
    3.24e+08
np.power(X, 10)[0]   1.862e+04   7.22e+04      0.258      0.797   -1.23e+05
    1.61e+05
np.power(X, 10)[1] -4.569e+06   4.33e+07     -0.106      0.916   -8.97e+07
    8.05e+07
np.power(X, 11)[0]   1.862e+04   7.22e+04      0.258      0.797   -1.23e+05
    1.61e+05
np.power(X, 11)[1]   9.263e+05   7.08e+06      0.131      0.896    -1.3e+07
    1.48e+07
np.power(X, 12)[0]   1.861e+04   7.22e+04      0.258      0.797   -1.23e+05
    1.61e+05
np.power(X, 12)[1] -8.087e+04   5.25e+05     -0.154      0.878   -1.11e+06
    9.52e+05
================================================================================
====
```

```
Omnibus:                       54.732   Durbin-Watson:                   0
.769
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               96
.284
Skew:                           0.818   Prob(JB):                     1.24
e-21
Kurtosis:                       4.794   Cond. No.                     8.33
e+42
==========================================================================
====

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is cor
rectly specified.
[2] The smallest eigenvalue is 5.61e-80. This might indicate that there ar
e
strong multicollinearity problems or that the design matrix is singular.
                            OLS Regression Results

==========================================================================
====
Dep. Variable:                    mpg   R-squared:                       0
.723
Model:                            OLS   Adj. R-squared:                  0
.713
Method:                 Least Squares   F-statistic:                     7
5.82
Date:                Mon, 20 Jun 2022   Prob (F-statistic):           1.05
e-96
Time:                        19:48:16   Log-Likelihood:                -11
09.7
No. Observations:                 392   AIC:                             2
247.
Df Residuals:                     378   BIC:                             2
303.
Df Model:                          13

Covariance Type:            nonrobust

==========================================================================
============
                         coef    std err          t      P>|t|      [0.025
      0.975]
--------------------------------------------------------------------------
------------
Intercept          -4.826e+05   3.22e+05     -1.497      0.135   -1.12e+06
    1.51e+05
X[0]               -2.358e+06   1.51e+06     -1.563      0.119   -5.32e+06
    6.08e+05
X[1]                7.361e+07   4.91e+07      1.499      0.135   -2.29e+07
     1.7e+08
np.power(X, 2)[0]  -2.553e+05    1.8e+05     -1.419      0.157   -6.09e+05
    9.84e+04
np.power(X, 2)[1]  -4.534e+08   2.98e+08     -1.520      0.129   -1.04e+09
    1.33e+08
np.power(X, 3)[0]  -2.765e+05   1.93e+05     -1.432      0.153   -6.56e+05
    1.03e+05
```

```
np.power(X, 3)[1]   1.682e+09   1.09e+09      1.538      0.125   -4.68e+08
    3.83e+09
np.power(X, 4)[0]  -2.816e+05   1.96e+05     -1.435      0.152   -6.67e+05
    1.04e+05
np.power(X, 4)[1]  -4.195e+09    2.7e+09     -1.554      0.121    -9.5e+09
    1.11e+09
np.power(X, 5)[0]  -2.815e+05   1.96e+05     -1.435      0.152   -6.67e+05
    1.04e+05
np.power(X, 5)[1]   7.426e+09   4.74e+09      1.567      0.118   -1.89e+09
    1.67e+10
np.power(X, 6)[0]  -2.815e+05   1.96e+05     -1.435      0.152   -6.67e+05
    1.04e+05
np.power(X, 6)[1]  -9.603e+09   6.09e+09     -1.578      0.115   -2.16e+10
    2.37e+09
np.power(X, 7)[0]  -2.815e+05   1.96e+05     -1.435      0.152   -6.67e+05
    1.04e+05
np.power(X, 7)[1]   9.189e+09   5.79e+09      1.586      0.114    -2.2e+09
    2.06e+10
np.power(X, 8)[0]  -2.815e+05   1.96e+05     -1.435      0.152   -6.67e+05
    1.04e+05
np.power(X, 8)[1]  -6.508e+09   4.09e+09     -1.592      0.112   -1.45e+10
    1.53e+09
np.power(X, 9)[0]  -2.815e+05   1.96e+05     -1.435      0.152   -6.67e+05
    1.04e+05
np.power(X, 9)[1]   3.371e+09   2.11e+09      1.596      0.111   -7.83e+08
    7.52e+09
np.power(X, 10)[0] -2.815e+05   1.96e+05     -1.435      0.152   -6.67e+05
    1.04e+05
np.power(X, 10)[1] -1.242e+09   7.77e+08     -1.598      0.111   -2.77e+09
    2.86e+08
np.power(X, 11)[0] -2.815e+05   1.96e+05     -1.435      0.152   -6.67e+05
    1.04e+05
np.power(X, 11)[1]  3.081e+08   1.93e+08      1.598      0.111    -7.1e+07
    6.87e+08
np.power(X, 12)[0] -2.815e+05   1.96e+05     -1.435      0.152   -6.67e+05
    1.04e+05
np.power(X, 12)[1] -4.617e+07   2.89e+07     -1.597      0.111   -1.03e+08
    1.07e+07
np.power(X, 13)[0]  4.713e+05   2.86e+05      1.650      0.100   -9.03e+04
    1.03e+06
np.power(X, 13)[1]  3.158e+06   1.98e+06      1.594      0.112   -7.37e+05
    7.05e+06
==========================================================================
====
Omnibus:                       58.401   Durbin-Watson:                   0
.779
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              106
.896
Skew:                           0.850   Prob(JB):                     6.14
e-24
Kurtosis:                       4.912   Cond. No.                     8.79
e+16
==========================================================================
====

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is cor
```

```
rectly specified.
[2] The smallest eigenvalue is 1.33e-27. This might indicate that there ar
e
strong multicollinearity problems or that the design matrix is singular.
                            OLS Regression Results

==========================================================================
====
Dep. Variable:                    mpg   R-squared:                       0
.724
Model:                            OLS   Adj. R-squared:                  0
.714
Method:                 Least Squares   F-statistic:                     7
0.64
Date:                Mon, 20 Jun 2022   Prob (F-statistic):           4.13
e-96
Time:                        19:48:17   Log-Likelihood:                -11
08.9
No. Observations:                 392   AIC:                             2
248.
Df Residuals:                     377   BIC:                             2
307.
Df Model:                          14

Covariance Type:            nonrobust

==========================================================================
============
                         coef    std err          t      P>|t|      [0.025
      0.975]
--------------------------------------------------------------------------
------------
Intercept          -1.402e+06   8.44e+05     -1.660      0.098   -3.06e+06
    2.59e+05
X[0]               -3.823e+06   2.45e+06     -1.558      0.120   -8.65e+06
       1e+06
X[1]                 3.02e+08   1.84e+08      1.640      0.102   -6.01e+07
    6.64e+08
np.power(X, 2)[0]  -1.109e+06   6.53e+05     -1.697      0.090   -2.39e+06
    1.76e+05
np.power(X, 2)[1]  -1.965e+09   1.21e+09     -1.621      0.106   -4.35e+09
    4.18e+08
np.power(X, 3)[0]  -1.249e+06   7.45e+05     -1.676      0.095   -2.71e+06
    2.16e+05
np.power(X, 3)[1]   7.764e+09   4.85e+09      1.600      0.110   -1.77e+09
    1.73e+10
np.power(X, 4)[0]  -1.248e+06   7.44e+05     -1.676      0.094   -2.71e+06
    2.16e+05
np.power(X, 4)[1]  -2.082e+10   1.32e+10     -1.577      0.116   -4.68e+10
    5.13e+09
np.power(X, 5)[0]  -1.248e+06   7.44e+05     -1.676      0.094   -2.71e+06
    2.16e+05
np.power(X, 5)[1]   4.009e+10   2.58e+10      1.552      0.121   -1.07e+10
    9.09e+10
np.power(X, 6)[0]  -1.248e+06   7.44e+05     -1.676      0.094   -2.71e+06
    2.16e+05
np.power(X, 6)[1]  -5.718e+10   3.75e+10     -1.526      0.128   -1.31e+11
```

```
    1.65e+10
np.power(X, 7)[0]  -1.248e+06   7.44e+05     -1.676      0.094   -2.71e+06
    2.16e+05
np.power(X, 7)[1]   6.139e+10    4.1e+10      1.498      0.135   -1.92e+10
    1.42e+11
np.power(X, 8)[0]  -1.248e+06   7.44e+05     -1.676      0.094   -2.71e+06
    2.16e+05
np.power(X, 8)[1]  -4.986e+10   3.39e+10     -1.469      0.143   -1.17e+11
    1.69e+10
np.power(X, 9)[0]  -1.248e+06   7.44e+05     -1.676      0.094   -2.71e+06
    2.16e+05
np.power(X, 9)[1]    3.05e+10   2.12e+10      1.439      0.151   -1.12e+10
    7.22e+10
np.power(X, 10)[0] -1.248e+06   7.44e+05     -1.676      0.094   -2.71e+06
    2.16e+05
np.power(X, 10)[1] -1.384e+10   9.82e+09     -1.409      0.160   -3.32e+10
    5.47e+09
np.power(X, 11)[0] -1.248e+06   7.44e+05     -1.676      0.094   -2.71e+06
    2.16e+05
np.power(X, 11)[1]  4.518e+09   3.28e+09      1.379      0.169   -1.93e+09
     1.1e+10
np.power(X, 12)[0] -1.248e+06   7.44e+05     -1.676      0.094   -2.71e+06
    2.16e+05
np.power(X, 12)[1] -1.003e+09   7.44e+08     -1.348      0.179   -2.47e+09
     4.6e+08
np.power(X, 13)[0] -1.248e+06   7.44e+05     -1.676      0.094   -2.71e+06
    2.16e+05
np.power(X, 13)[1]  1.357e+08   1.03e+08      1.317      0.189   -6.69e+07
    3.38e+08
np.power(X, 14)[0] -1.205e+06   7.16e+05     -1.683      0.093   -2.61e+06
    2.03e+05
np.power(X, 14)[1] -8.442e+06   6.56e+06     -1.287      0.199   -2.13e+07
    4.46e+06
===========================================================================
====
Omnibus:                       60.942   Durbin-Watson:                    0
.781
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               114
.984
Skew:                           0.870   Prob(JB):                      1.08
e-25
Kurtosis:                       5.003   Cond. No.                      1.95
e+17
===========================================================================
====

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is cor
rectly specified.
[2] The smallest eigenvalue is 7.2e-28. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.
                            OLS Regression Results

===========================================================================
====
Dep. Variable:                    mpg   R-squared:                        0
.724
```

```
Model:                            OLS   Adj. R-squared:                  0
.714
Method:                 Least Squares   F-statistic:                     7
0.58
Date:                Mon, 20 Jun 2022   Prob (F-statistic):           4.65
e-96
Time:                        19:48:18   Log-Likelihood:                -11
09.0
No. Observations:                 392   AIC:                             2
248.
Df Residuals:                     377   BIC:                             2
308.
Df Model:                          14

Covariance Type:            nonrobust

==========================================================================
============
                         coef    std err          t      P>|t|      [0.025
      0.975]
--------------------------------------------------------------------------
------------
Intercept          -7.173e+05   4.24e+05     -1.693      0.091   -1.55e+06
    1.16e+05
X[0]               -9.456e+05   5.66e+05     -1.672      0.095   -2.06e+06
    1.66e+05
X[1]                1.493e+08   8.85e+07      1.687      0.092   -2.47e+07
    3.23e+08
np.power(X, 2)[0]  -6.906e+05   4.07e+05     -1.696      0.091   -1.49e+06
     1.1e+05
np.power(X, 2)[1]  -8.848e+08   5.27e+08     -1.678      0.094   -1.92e+09
    1.52e+08
np.power(X, 3)[0]  -6.888e+05   4.06e+05     -1.696      0.091   -1.49e+06
     1.1e+05
np.power(X, 3)[1]   3.088e+09   1.85e+09      1.665      0.097   -5.59e+08
    6.74e+09
np.power(X, 4)[0]  -6.888e+05   4.06e+05     -1.696      0.091   -1.49e+06
     1.1e+05
np.power(X, 4)[1]  -6.963e+09   4.22e+09     -1.648      0.100   -1.53e+10
    1.34e+09
np.power(X, 5)[0]  -6.888e+05   4.06e+05     -1.696      0.091   -1.49e+06
     1.1e+05
np.power(X, 5)[1]   1.031e+10   6.33e+09      1.627      0.104   -2.15e+09
    2.28e+10
np.power(X, 6)[0]  -6.888e+05   4.06e+05     -1.696      0.091   -1.49e+06
     1.1e+05
np.power(X, 6)[1]  -9.219e+09   5.78e+09     -1.596      0.111   -2.06e+10
    2.14e+09
np.power(X, 7)[0]  -6.888e+05   4.06e+05     -1.696      0.091   -1.49e+06
     1.1e+05
np.power(X, 7)[1]   2.442e+09   1.67e+09      1.466      0.144   -8.34e+08
    5.72e+09
np.power(X, 8)[0]  -6.888e+05   4.06e+05     -1.696      0.091   -1.49e+06
     1.1e+05
np.power(X, 8)[1]   5.886e+09   3.59e+09      1.639      0.102   -1.18e+09
    1.29e+10
np.power(X, 9)[0]  -6.888e+05   4.06e+05     -1.696      0.091   -1.49e+06
```

```
     1.1e+05
np.power(X, 9)[1]  -1.008e+10   6.36e+09     -1.585      0.114   -2.26e+10
    2.42e+09
np.power(X, 10)[0] -6.888e+05   4.06e+05     -1.696      0.091   -1.49e+06
     1.1e+05
np.power(X, 10)[1]  8.715e+09   5.61e+09      1.553      0.121   -2.32e+09
    1.98e+10
np.power(X, 11)[0] -6.888e+05   4.06e+05     -1.696      0.091   -1.49e+06
     1.1e+05
np.power(X, 11)[1] -4.885e+09   3.21e+09     -1.524      0.128   -1.12e+10
    1.42e+09
np.power(X, 12)[0] -6.888e+05   4.06e+05     -1.696      0.091   -1.49e+06
     1.1e+05
np.power(X, 12)[1]  1.843e+09   1.23e+09      1.496      0.135   -5.79e+08
    4.27e+09
np.power(X, 13)[0] -6.888e+05   4.06e+05     -1.696      0.091   -1.49e+06
     1.1e+05
np.power(X, 13)[1] -4.552e+08    3.1e+08     -1.469      0.143   -1.06e+09
    1.54e+08
np.power(X, 14)[0] -6.888e+05   4.06e+05     -1.696      0.091   -1.49e+06
     1.1e+05
np.power(X, 14)[1]  6.679e+07   4.63e+07      1.441      0.150   -2.43e+07
    1.58e+08
np.power(X, 15)[0] -6.849e+05   4.04e+05     -1.696      0.091   -1.48e+06
    1.09e+05
np.power(X, 15)[1]  -4.43e+06   3.13e+06     -1.414      0.158   -1.06e+07
    1.73e+06
==============================================================================
====
Omnibus:                       60.948   Durbin-Watson:                      0
.780
Prob(Omnibus):                  0.000   Jarque-Bera (JB):                 114
.859
Skew:                           0.871   Prob(JB):                        1.14
e-25
Kurtosis:                       5.000   Cond. No.                        4.28
e+17
==============================================================================
====

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is cor
rectly specified.
[2] The smallest eigenvalue is 4.01e-28. This might indicate that there ar
e
strong multicollinearity problems or that the design matrix is singular.
                            OLS Regression Results

==============================================================================
====
Dep. Variable:                    mpg   R-squared:                          0
.724
Model:                            OLS   Adj. R-squared:                     0
.714
Method:                 Least Squares   F-statistic:                        7
0.74
Date:                Mon, 20 Jun 2022   Prob (F-statistic):              3.43
```

```
e-96
Time:                        19:48:18   Log-Likelihood:                -11
08.7
No. Observations:                 392   AIC:                             2
247.
Df Residuals:                     377   BIC:                             2
307.
Df Model:                          14

Covariance Type:            nonrobust

=====================================================================================
============
                          coef    std err          t      P>|t|      [0.025
      0.975]
-------------------------------------------------------------------------------------
------------
Intercept            6.874e+06   7.17e+06      0.958      0.338   -7.23e+06
     2.1e+07
X[0]                -8.197e+07    8.1e+07     -1.012      0.312   -2.41e+08
    7.72e+07
X[1]                 4.051e+08   3.34e+08      1.215      0.225   -2.51e+08
    1.06e+09
np.power(X, 2)[0]   -1.314e+08    1.3e+08     -1.011      0.313   -3.87e+08
    1.24e+08
np.power(X, 2)[1]   -2.556e+09   2.16e+09     -1.182      0.238   -6.81e+09
    1.69e+09
np.power(X, 3)[0]   -1.123e+08   1.11e+08     -1.011      0.312   -3.31e+08
    1.06e+08
np.power(X, 3)[1]    9.608e+09   8.37e+09      1.148      0.252   -6.85e+09
    2.61e+10
np.power(X, 4)[0]   -1.123e+08   1.11e+08     -1.011      0.312    -3.3e+08
    1.06e+08
np.power(X, 4)[1]   -2.375e+10   2.13e+10     -1.112      0.267   -6.57e+10
    1.82e+10
np.power(X, 5)[0]   -1.123e+08   1.11e+08     -1.011      0.312   -3.31e+08
    1.06e+08
np.power(X, 5)[1]    3.978e+10    3.7e+10      1.074      0.284   -3.31e+10
    1.13e+11
np.power(X, 6)[0]   -1.123e+08   1.11e+08     -1.011      0.312   -3.31e+08
    1.06e+08
np.power(X, 6)[1]   -4.361e+10   4.24e+10     -1.029      0.304   -1.27e+11
    3.98e+10
np.power(X, 7)[0]   -1.123e+08   1.11e+08     -1.011      0.312   -3.31e+08
    1.06e+08
np.power(X, 7)[1]    2.468e+10   2.59e+10      0.952      0.342   -2.63e+10
    7.57e+10
np.power(X, 8)[0]   -1.123e+08   1.11e+08     -1.011      0.312   -3.31e+08
    1.06e+08
np.power(X, 8)[1]    8.975e+09   6.94e+09      1.293      0.197   -4.67e+09
    2.26e+10
np.power(X, 9)[0]   -1.123e+08   1.11e+08     -1.011      0.312   -3.31e+08
    1.06e+08
np.power(X, 9)[1]   -3.526e+10   3.46e+10     -1.020      0.308   -1.03e+11
    3.27e+10
np.power(X, 10)[0] -1.123e+08   1.11e+08     -1.011      0.312   -3.31e+08
    1.06e+08
```

```
np.power(X, 10)[1]  3.994e+10   4.14e+10      0.965      0.335   -4.14e+10
    1.21e+11
np.power(X, 11)[0] -1.123e+08   1.11e+08     -1.011      0.312   -3.31e+08
    1.06e+08
np.power(X, 11)[1] -2.808e+10   3.03e+10     -0.927      0.355   -8.77e+10
    3.15e+10
np.power(X, 12)[0] -1.123e+08   1.11e+08     -1.011      0.312   -3.31e+08
    1.06e+08
np.power(X, 12)[1]  1.353e+10   1.52e+10      0.893      0.372   -1.63e+10
    4.33e+10
np.power(X, 13)[0] -1.123e+08   1.11e+08     -1.011      0.312   -3.31e+08
    1.06e+08
np.power(X, 13)[1] -4.506e+09   5.23e+09     -0.862      0.389   -1.48e+10
    5.77e+09
np.power(X, 14)[0] -1.123e+08   1.11e+08     -1.011      0.312   -3.31e+08
    1.06e+08
np.power(X, 14)[1]  9.988e+08    1.2e+09      0.833      0.405   -1.36e+09
    3.36e+09
np.power(X, 15)[0] -1.123e+08   1.11e+08     -1.011      0.312   -3.31e+08
    1.06e+08
np.power(X, 15)[1] -1.331e+08   1.65e+08     -0.806      0.421   -4.58e+08
    1.92e+08
np.power(X, 16)[0]  1.637e+09   1.62e+09      1.008      0.314   -1.56e+09
    4.83e+09
np.power(X, 16)[1]  8.088e+06   1.04e+07      0.780      0.436   -1.23e+07
    2.85e+07
==========================================================================
====
Omnibus:                       60.728   Durbin-Watson:                   0
.783
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              114
.595
Skew:                           0.867   Prob(JB):                     1.31
e-25
Kurtosis:                       5.001   Cond. No.                     1.83
e+17
==========================================================================
====

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is cor
rectly specified.
[2] The smallest eigenvalue is 5.93e-27. This might indicate that there ar
e
strong multicollinearity problems or that the design matrix is singular.
                            OLS Regression Results

==========================================================================
====
Dep. Variable:                    mpg   R-squared:                       0
.724
Model:                            OLS   Adj. R-squared:                  0
.714
Method:                 Least Squares   F-statistic:                     7
0.78
Date:                Mon, 20 Jun 2022   Prob (F-statistic):           3.18
e-96
```

```
Time:                        19:48:19   Log-Likelihood:                -11
08.6
No. Observations:                 392   AIC:                             2
247.
Df Residuals:                     377   BIC:                             2
307.
Df Model:                          14

Covariance Type:            nonrobust

==========================================================================
============
                         coef    std err          t      P>|t|      [0.025
      0.975]
--------------------------------------------------------------------------
------------
Intercept           -5.011e+05   2.85e+05     -1.760      0.079   -1.06e+06
    5.89e+04
X[0]                 6.158e+05   6.13e+05      1.005      0.316   -5.89e+05
    1.82e+06
X[1]                 2.498e+08   1.69e+08      1.479      0.140   -8.24e+07
    5.82e+08
np.power(X, 2)[0]   -1.196e+06   8.09e+05     -1.478      0.140   -2.79e+06
    3.95e+05
np.power(X, 2)[1]   -1.456e+09      1e+09     -1.450      0.148   -3.43e+09
    5.18e+08
np.power(X, 3)[0]   -1.204e+06   8.15e+05     -1.477      0.140   -2.81e+06
    3.99e+05
np.power(X, 3)[1]    4.922e+09   3.47e+09      1.419      0.157    -1.9e+09
    1.17e+10
np.power(X, 4)[0]   -1.205e+06   8.16e+05     -1.477      0.140   -2.81e+06
    3.99e+05
np.power(X, 4)[1]   -1.046e+10   7.55e+09     -1.385      0.167   -2.53e+10
     4.4e+09
np.power(X, 5)[0]   -1.205e+06   8.16e+05     -1.477      0.140   -2.81e+06
    3.99e+05
np.power(X, 5)[1]    1.375e+10   1.02e+10      1.342      0.180    -6.4e+09
    3.39e+10
np.power(X, 6)[0]   -1.205e+06   8.16e+05     -1.477      0.140   -2.81e+06
    3.99e+05
np.power(X, 6)[1]   -8.939e+09   7.05e+09     -1.267      0.206   -2.28e+10
    4.93e+09
np.power(X, 7)[0]   -1.205e+06   8.16e+05     -1.477      0.140   -2.81e+06
    3.99e+05
np.power(X, 7)[1]   -2.838e+09   1.78e+09     -1.597      0.111   -6.33e+09
    6.57e+08
np.power(X, 8)[0]   -1.205e+06   8.16e+05     -1.477      0.140   -2.81e+06
    3.99e+05
np.power(X, 8)[1]    1.108e+10   8.44e+09      1.314      0.190   -5.51e+09
    2.77e+10
np.power(X, 9)[0]   -1.205e+06   8.16e+05     -1.477      0.140   -2.81e+06
    3.99e+05
np.power(X, 9)[1]   -7.404e+09   6.12e+09     -1.210      0.227   -1.94e+10
    4.62e+09
np.power(X, 10)[0] -1.205e+06   8.16e+05     -1.477      0.140   -2.81e+06
    3.99e+05
np.power(X, 10)[1] -3.868e+09   2.65e+09     -1.457      0.146   -9.09e+09
```

```
    1.35e+09
np.power(X, 11)[0] -1.205e+06   8.16e+05     -1.477      0.140   -2.81e+06
    3.99e+05
np.power(X, 11)[1]  1.166e+10   9.17e+09      1.271      0.204   -6.38e+09
    2.97e+10
np.power(X, 12)[0] -1.205e+06   8.16e+05     -1.477      0.140   -2.81e+06
    3.99e+05
np.power(X, 12)[1] -1.126e+10   9.23e+09     -1.220      0.223   -2.94e+10
    6.89e+09
np.power(X, 13)[0] -1.205e+06   8.16e+05     -1.477      0.140   -2.81e+06
    3.99e+05
np.power(X, 13)[1]  6.512e+09   5.51e+09      1.182      0.238   -4.32e+09
    1.73e+10
np.power(X, 14)[0] -1.205e+06   8.16e+05     -1.477      0.140   -2.81e+06
    3.99e+05
np.power(X, 14)[1] -2.458e+09   2.14e+09     -1.149      0.251   -6.66e+09
    1.75e+09
np.power(X, 15)[0] -1.205e+06   8.16e+05     -1.477      0.140   -2.81e+06
    3.99e+05
np.power(X, 15)[1]  5.978e+08   5.34e+08      1.119      0.264   -4.53e+08
    1.65e+09
np.power(X, 16)[0] -1.205e+06   8.16e+05     -1.477      0.140   -2.81e+06
    3.99e+05
np.power(X, 16)[1] -8.566e+07   7.85e+07     -1.091      0.276    -2.4e+08
    6.87e+07
np.power(X, 17)[0] -1.124e+06   7.53e+05     -1.492      0.137   -2.61e+06
    3.58e+05
np.power(X, 17)[1]   5.52e+06   5.19e+06      1.064      0.288   -4.68e+06
    1.57e+07
==========================================================================
====
Omnibus:                       61.058   Durbin-Watson:                   0
.784
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              115
.690
Skew:                           0.870   Prob(JB):                     7.56
e-26
Kurtosis:                       5.014   Cond. No.                     3.63
e+17
==========================================================================
====

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is cor
rectly specified.
[2] The smallest eigenvalue is 4.14e-27. This might indicate that there ar
e
strong multicollinearity problems or that the design matrix is singular.
                            OLS Regression Results

==========================================================================
====
Dep. Variable:                    mpg   R-squared:                       0
.724
Model:                            OLS   Adj. R-squared:                  0
.713
Method:                 Least Squares   F-statistic:                     6
```

```
5.89
Date:                  Mon, 20 Jun 2022   Prob (F-statistic):           2.68
e-95
Time:                          19:48:20   Log-Likelihood:                -11
08.6
No. Observations:                   392   AIC:                             2
249.
Df Residuals:                       376   BIC:                             2
313.
Df Model:                            15

Covariance Type:              nonrobust

==========================================================================
============
                         coef    std err          t      P>|t|      [0.025
      0.975]
--------------------------------------------------------------------------
------------
Intercept          -5.852e+05   3.47e+05     -1.687      0.092   -1.27e+06
     9.7e+04
X[0]                6.824e+06   4.59e+06      1.486      0.138    -2.2e+06
    1.59e+07
X[1]                1.461e+08   8.82e+07      1.656      0.098   -2.73e+07
     3.2e+08
np.power(X, 2)[0]  -8.838e+05   5.44e+05     -1.625      0.105   -1.95e+06
    1.86e+05
np.power(X, 2)[1]  -7.875e+08   4.82e+08     -1.634      0.103   -1.73e+09
     1.6e+08
np.power(X, 3)[0]  -1.008e+06   6.26e+05     -1.609      0.108   -2.24e+06
    2.23e+05
np.power(X, 3)[1]   2.394e+09   1.49e+09      1.609      0.109   -5.32e+08
    5.32e+09
np.power(X, 4)[0]   -1.01e+06   6.27e+05     -1.609      0.108   -2.24e+06
    2.24e+05
np.power(X, 4)[1]   -4.34e+09   2.75e+09     -1.577      0.116   -9.75e+09
    1.07e+09
np.power(X, 5)[0]   -1.01e+06   6.27e+05     -1.609      0.108   -2.24e+06
    2.24e+05
np.power(X, 5)[1]   4.258e+09   2.79e+09      1.529      0.127   -1.22e+09
    9.74e+09
np.power(X, 6)[0]   -1.01e+06   6.27e+05     -1.609      0.108   -2.24e+06
    2.24e+05
np.power(X, 6)[1]  -6.971e+08   5.68e+08     -1.228      0.220   -1.81e+09
    4.19e+08
np.power(X, 7)[0]   -1.01e+06   6.27e+05     -1.609      0.108   -2.24e+06
    2.24e+05
np.power(X, 7)[1]  -3.302e+09   2.11e+09     -1.564      0.119   -7.45e+09
    8.48e+08
np.power(X, 8)[0]   -1.01e+06   6.27e+05     -1.609      0.108   -2.24e+06
    2.24e+05
np.power(X, 8)[1]    2.89e+09   1.98e+09      1.456      0.146   -1.01e+09
    6.79e+09
np.power(X, 9)[0]   -1.01e+06   6.27e+05     -1.609      0.108   -2.24e+06
    2.24e+05
np.power(X, 9)[1]   1.394e+09   8.32e+08      1.675      0.095   -2.42e+08
    3.03e+09
```

```
np.power(X, 10)[0]  -1.01e+06   6.27e+05     -1.609      0.108   -2.24e+06
    2.24e+05
np.power(X, 10)[1] -3.686e+09   2.49e+09     -1.483      0.139   -8.57e+09
     1.2e+09
np.power(X, 11)[0]  -1.01e+06   6.27e+05     -1.609      0.108   -2.24e+06
    2.24e+05
np.power(X, 11)[1]  1.107e+09   8.64e+08      1.281      0.201   -5.92e+08
     2.8e+09
np.power(X, 12)[0]  -1.01e+06   6.27e+05     -1.609      0.108   -2.24e+06
    2.24e+05
np.power(X, 12)[1]  2.876e+09    1.9e+09      1.518      0.130    -8.5e+08
     6.6e+09
np.power(X, 13)[0]  -1.01e+06   6.27e+05     -1.609      0.108   -2.24e+06
    2.24e+05
np.power(X, 13)[1] -4.219e+09   2.92e+09     -1.443      0.150   -9.97e+09
    1.53e+09
np.power(X, 14)[0]  -1.01e+06   6.27e+05     -1.609      0.108   -2.24e+06
    2.24e+05
np.power(X, 14)[1]  2.945e+09    2.1e+09      1.402      0.162   -1.19e+09
    7.08e+09
np.power(X, 15)[0]  -1.01e+06   6.27e+05     -1.609      0.108   -2.24e+06
    2.24e+05
np.power(X, 15)[1] -1.252e+09   9.14e+08     -1.369      0.172   -3.05e+09
    5.46e+08
np.power(X, 16)[0]  -1.01e+06   6.27e+05     -1.609      0.108   -2.24e+06
    2.24e+05
np.power(X, 16)[1]  3.314e+08   2.47e+08      1.339      0.181   -1.55e+08
    8.18e+08
np.power(X, 17)[0]  -1.01e+06   6.27e+05     -1.609      0.108   -2.24e+06
    2.24e+05
np.power(X, 17)[1] -5.066e+07   3.86e+07     -1.312      0.190   -1.27e+08
    2.53e+07
np.power(X, 18)[0] -2.064e+06   1.33e+06     -1.554      0.121   -4.68e+06
    5.47e+05
np.power(X, 18)[1]  3.438e+06   2.67e+06      1.286      0.199   -1.82e+06
     8.7e+06
==========================================================================
====
Omnibus:                        61.392   Durbin-Watson:                  0
.784
Prob(Omnibus):                   0.000   Jarque-Bera (JB):             116
.752
Skew:                            0.873   Prob(JB):                    4.44
e-26
Kurtosis:                        5.025   Cond. No.                    6.21
e+17
==========================================================================
====

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is cor
rectly specified.
[2] The smallest eigenvalue is 3.91e-27. This might indicate that there ar
e
strong multicollinearity problems or that the design matrix is singular.
                            OLS Regression Results
```

```
==============================================================================
====
Dep. Variable:                    mpg   R-squared:                       0
.724
Model:                            OLS   Adj. R-squared:                  0
.713
Method:                 Least Squares   F-statistic:                     6
5.90
Date:                Mon, 20 Jun 2022   Prob (F-statistic):           2.66
e-95
Time:                        19:48:21   Log-Likelihood:                -11
08.6
No. Observations:                 392   AIC:                             2
249.
Df Residuals:                     376   BIC:                             2
313.
Df Model:                          15

Covariance Type:            nonrobust

==============================================================================
============
                         coef    std err          t      P>|t|      [0.025
      0.975]
------------------------------------------------------------------------------
------------
Intercept          -3.512e+05      2e+05     -1.757      0.080   -7.44e+05
    4.19e+04
X[0]               -1.906e+07   4.98e+07     -0.383      0.702   -1.17e+08
    7.88e+07
X[1]                2.108e+08   3.37e+08      0.625      0.532   -4.53e+08
    8.74e+08
np.power(X, 2)[0]   1.761e+05   1.46e+06      0.121      0.904   -2.69e+06
    3.04e+06
np.power(X, 2)[1]  -1.183e+09   2.02e+09     -0.585      0.559   -5.16e+09
    2.79e+09
np.power(X, 3)[0]   1.809e+05   1.47e+06      0.123      0.902   -2.71e+06
    3.07e+06
np.power(X, 3)[1]   3.799e+09   6.99e+09      0.544      0.587   -9.94e+09
    1.75e+10
np.power(X, 4)[0]   1.856e+05   1.48e+06      0.125      0.900   -2.73e+06
     3.1e+06
np.power(X, 4)[1]  -7.469e+09    1.5e+10     -0.498      0.619   -3.69e+10
     2.2e+10
np.power(X, 5)[0]   1.859e+05   1.48e+06      0.125      0.900   -2.73e+06
     3.1e+06
np.power(X, 5)[1]   8.559e+09   1.93e+10      0.443      0.658   -2.94e+10
    4.65e+10
np.power(X, 6)[0]   1.859e+05   1.48e+06      0.125      0.900   -2.73e+06
     3.1e+06
np.power(X, 6)[1]  -3.635e+09   1.09e+10     -0.333      0.739   -2.51e+10
    1.78e+10
np.power(X, 7)[0]   1.859e+05   1.48e+06      0.125      0.900   -2.73e+06
     3.1e+06
np.power(X, 7)[1]  -4.085e+09   7.11e+09     -0.575      0.566   -1.81e+10
    9.89e+09
np.power(X, 8)[0]   1.859e+05   1.48e+06      0.125      0.900   -2.73e+06
```

```
     3.1e+06
np.power(X, 8)[1]    6.22e+09   1.59e+10      0.390      0.697   -2.51e+10
    3.76e+10
np.power(X, 9)[0]   1.859e+05   1.48e+06      0.125      0.900   -2.73e+06
     3.1e+06
np.power(X, 9)[1]  -4.307e+08    4.5e+09     -0.096      0.924   -9.29e+09
    8.43e+09
np.power(X, 10)[0]  1.859e+05   1.48e+06      0.125      0.900   -2.73e+06
     3.1e+06
np.power(X, 10)[1] -5.409e+09   1.27e+10     -0.425      0.671   -3.04e+10
    1.96e+10
np.power(X, 11)[0]  1.859e+05   1.48e+06      0.125      0.900   -2.73e+06
     3.1e+06
np.power(X, 11)[1]   4.06e+09   1.34e+10      0.303      0.762   -2.22e+10
    3.04e+10
np.power(X, 12)[0]  1.859e+05   1.48e+06      0.125      0.900   -2.73e+06
     3.1e+06
np.power(X, 12)[1]  1.967e+09   2.57e+09      0.766      0.444   -3.08e+09
    7.01e+09
np.power(X, 13)[0]  1.859e+05   1.48e+06      0.125      0.900   -2.73e+06
     3.1e+06
np.power(X, 13)[1] -5.808e+09   1.64e+10     -0.353      0.724   -3.81e+10
    2.65e+10
np.power(X, 14)[0]  1.859e+05   1.48e+06      0.125      0.900   -2.73e+06
     3.1e+06
np.power(X, 14)[1]  5.221e+09   1.75e+10      0.298      0.766   -2.93e+10
    3.97e+10
np.power(X, 15)[0]  1.859e+05   1.48e+06      0.125      0.900   -2.73e+06
     3.1e+06
np.power(X, 15)[1] -2.764e+09   1.06e+10     -0.261      0.794   -2.36e+10
     1.8e+10
np.power(X, 16)[0]  1.859e+05   1.48e+06      0.125      0.900   -2.73e+06
     3.1e+06
np.power(X, 16)[1]  9.442e+08   4.07e+09      0.232      0.817   -7.07e+09
    8.95e+09
np.power(X, 17)[0]  1.859e+05   1.48e+06      0.125      0.900   -2.73e+06
     3.1e+06
np.power(X, 17)[1] -2.059e+08      1e+09     -0.206      0.837   -2.17e+09
    1.76e+09
np.power(X, 18)[0]  1.859e+05   1.48e+06      0.125      0.900   -2.73e+06
     3.1e+06
np.power(X, 18)[1]  2.625e+07   1.44e+08      0.182      0.855   -2.57e+08
    3.09e+08
np.power(X, 19)[0] -2.854e+05   3.04e+05     -0.938      0.349   -8.84e+05
    3.13e+05
np.power(X, 19)[1] -1.493e+06   9.27e+06     -0.161      0.872   -1.97e+07
    1.67e+07
==========================================================================
====
Omnibus:                       61.309   Durbin-Watson:                   0
.783
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              116
.341
Skew:                           0.873   Prob(JB):                     5.46
e-26
Kurtosis:                       5.019   Cond. No.                     1.08
e+18
```

```
================================================================================
====

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is cor
rectly specified.
[2] The smallest eigenvalue is 3.59e-27. This might indicate that there ar
e
strong multicollinearity problems or that the design matrix is singular.
```

In [10]:
```
# tests best_degree, best_r_squared, and sound_degree
```

TODO:

Open the Peer Review assignment for this week to answer a question for section 1d.

2. Multi-Linear Regression [15 pts, Peer Review]

In the following problem, you will construct a simple multi-linear regression model, identify interaction terms and use diagnostic plots to identify outliers in the data. The original problem is as described by John Verzani in the excellent tutorial 'SimplR' on the R statistics language and uses data from the 2000 presidential election in Florida. The problem is interesting because it contains a small number of highly leveraged points that influence the model.

In [11]:
```
votes = pd.read_csv('data/fl2000.txt', delim_whitespace=True, comment='#')
votes = votes[['county', 'Bush', 'Gore', 'Nader', 'Buchanan']]
votes.describe(include='all')
```

Out[11]:

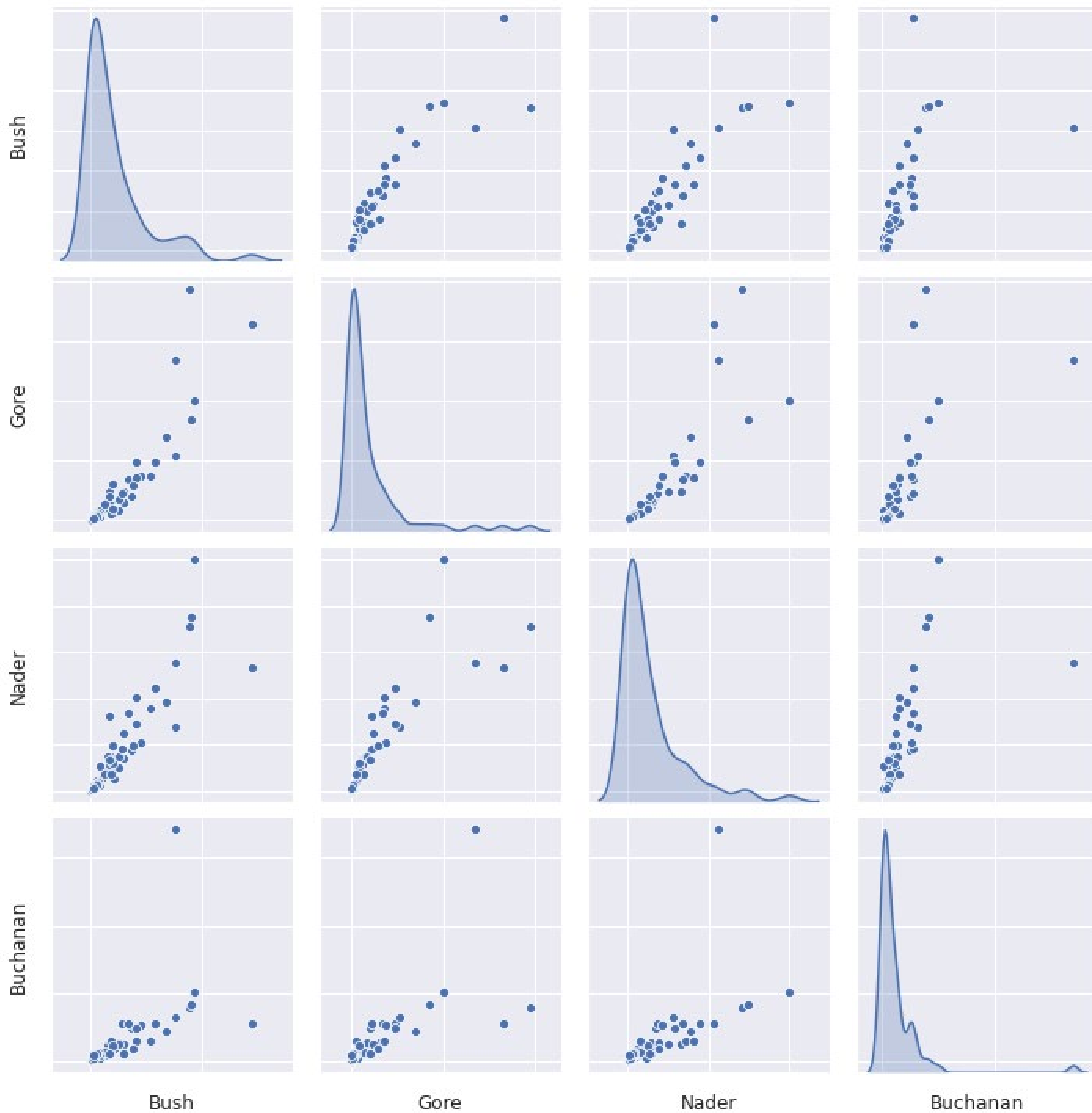
	county	Bush	Gore	Nader	Buchanan
count	67	67.000000	67.000000	67.000000	67.000000
unique	67	NaN	NaN	NaN	NaN
top	Jefferson	NaN	NaN	NaN	NaN
freq	1	NaN	NaN	NaN	NaN
mean	NaN	43450.970149	43453.985075	1454.119403	260.880597
std	NaN	57182.620266	75070.435056	2033.620972	450.498092
min	NaN	1317.000000	789.000000	19.000000	9.000000
25%	NaN	4757.000000	3058.000000	95.500000	46.500000
50%	NaN	20206.000000	14167.000000	562.000000	120.000000
75%	NaN	56546.500000	46015.000000	1870.500000	285.500000
max	NaN	289533.000000	387703.000000	10022.000000	3411.000000

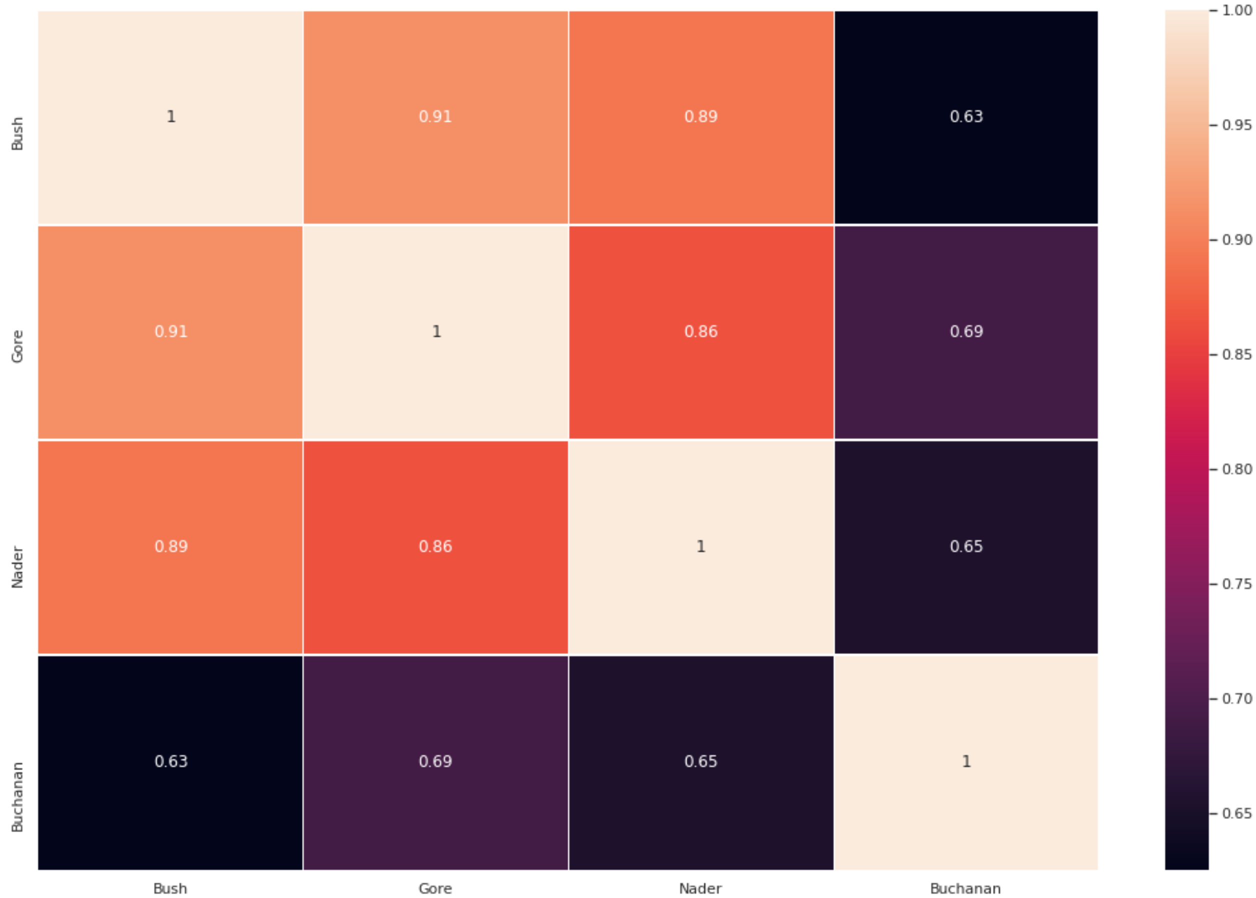
2a. Plot a pair plot of the data using the `seaborn` library. [Peer Review]

Upload a screenshot or saved copy of your plot for this week's Peer Review assignment.
Note: your code for this section may cause the Validate button to time out. If you want to run the Validate button prior to submitting, you could comment out the code in this section after completing the Peer Review.

In [12]:
```
# plot a pair plot of the data using the seaborn library
# possible way to save the image
# plt.savefig('pair_plot.png', dpi = 300, bbox_inches = 'tight')
# your code here
g = sns.pairplot(votes, diag_kind = 'kde')
g.set(xticklabels = [], yticklabels = [])
plt.savefig('pair_plot.png', dpi = 300, bbox_inches = 'tight')

corrg = votes.corr()
plt.figure(figsize = (18,12))
sns.heatmap(corrg, annot = True, linewidths = 0.5)
plt.savefig('corr_plot.png', dpi = 300, bbox_inches = 'tight')
```


Bush
Gore
Nader
Buchanan
Bush
Gore
Nader
Buchanan



2b. Comment on the relationship between the quantiative datasets. Are they correlated? Collinear? [Peer Review]

You will answer this question in this week's Peer Review assignment.

Solution
All the pairwise relationships show collinearity to some degree, exhibiting a positive linear distribution initially and spreading out as we move further right with respect to the x-axis. Relatively speaking, we can observe a skinnier linear shape for the pairs Bush-Buchanan and Nader-Buchanan.
Taking a look at the correlation heatmap, there are strong correlations for the pairs Bush-Gore, Bush-Nader, and Nader-Gore.

2c. Multi-linear [5 pts, Peer Review]

Construct a multi-linear model called `model` without interaction terms predicting the Bush column on the other columns and print out the summary table. You should name your model's object as `model` in order to pass the autograder. Use the full data (not train-test split for now) and do not scale features.

In [13]:
```
# uncomment and construct a multi-linear model
model = (smf.ols(formula = 'Bush ~ Gore + Nader + Buchanan', data = votes)
).fit()
# your code here
fit1 = smf.ols(formula = 'Bush ~ Gore + Nader + Buchanan', data = votes)
```

```
modelfit = fit1.fit()
print(modelfit.summary())
```

```
                            OLS Regression Results

==============================================================================
====
Dep. Variable:                   Bush   R-squared:                       0
.877
Model:                            OLS   Adj. R-squared:                  0
.871
Method:                 Least Squares   F-statistic:                     1
49.5
Date:                Mon, 20 Jun 2022   Prob (F-statistic):           1.35
e-28
Time:                        19:48:37   Log-Likelihood:                -75
8.33
No. Observations:                  67   AIC:                             1
525.
Df Residuals:                      63   BIC:                             1
533.
Df Model:                           3

Covariance Type:            nonrobust

==============================================================================
====
                 coef    std err          t      P>|t|      [0.025      0.
975]
------------------------------------------------------------------------------
----
Intercept   8647.6837   3133.545      2.760      0.008    2385.793    1.49
e+04
Gore           0.4475      0.071      6.305      0.000       0.306       0
.589
Nader         11.8533      2.503      4.735      0.000       6.851      16
.855
Buchanan      -7.2033      7.864     -0.916      0.363     -22.917       8
.511
==============================================================================
====
Omnibus:                       20.698   Durbin-Watson:                   1
.969
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              128
.017
Skew:                           0.383   Prob(JB):                     1.59
e-28
Kurtosis:                       9.728   Cond. No.                     1.08
e+05
==============================================================================
====

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is cor
rectly specified.
[2] The condition number is large, 1.08e+05. This might indicate that ther
e are
```

```
strong multicollinearity or other numerical problems.
```

In [14]:
```
# tests model
```

Is there any insignificant feature(s)? Explain your answer in this week's Peer Review assignment.

Solution
Based on the summary table, the p-values for Gore and Nader are 0, indicating that these variables are statistically significant.

2d. Multi-linear with interactions [Peer Review]

Construct a multi-linear model with interactions that are statistically significant at the $p = 0.05$ level. You can start with full interactions and then eliminate interactions are do not meet the $p = 0.05$ threshold. Name this model object as `model_multi`. You will share you solution in this week's Peer Review assignment.

In [75]:
```
# uncomment and construct multi-linear model
model_multi = (smf.ols(formula = 'Bush ~ Gore + Nader + Buchanan + Gore:Na
der + Gore:Buchanan + Nader:Buchanan', data = votes)).fit()
# your code here
multifit1 = (smf.ols(formula = 'Bush ~ Gore + Nader + Gore:Nader + Gore:Bu
chanan + Nader:Buchanan', data = votes)).fit()
print(multifit1.summary())
```

```
                            OLS Regression Results
                            
==========================================================================
====
Dep. Variable:                   Bush   R-squared:                       0
.948
Model:                            OLS   Adj. R-squared:                  0
.944
Method:                 Least Squares   F-statistic:                     2
23.3
Date:                Mon, 20 Jun 2022   Prob (F-statistic):           7.63
e-38
Time:                        20:46:43   Log-Likelihood:                -72
9.31
No. Observations:                  67   AIC:                             1
471.
Df Residuals:                      61   BIC:                             1
484.
Df Model:                           5                                     
                                                                          
Covariance Type:            nonrobust                                     
                                                                          
==========================================================================
========
                     coef    std err          t      P>|t|      [0.025    
  0.975]
--------------------------------------------------------------------------
--------
```

```
Intercept         592.5059   2384.941      0.248      0.805   -4176.477
5361.489
Gore                1.7895      0.179      9.978      0.000       1.431
   2.148
Nader             -11.4015      5.022     -2.270      0.027     -21.443
  -1.360
Gore:Nader         -0.0001   1.55e-05     -8.451      0.000      -0.000
  -0.000
Gore:Buchanan      -0.0008      0.000     -5.814      0.000      -0.001
  -0.001
Nader:Buchanan      0.0371      0.007      5.399      0.000       0.023
   0.051
==========================================================================
====
Omnibus:                        6.010   Durbin-Watson:                   1
.986
Prob(Omnibus):                  0.050   Jarque-Bera (JB):                9
.326
Skew:                          -0.083   Prob(JB):                      0.0
0944
Kurtosis:                       4.820   Cond. No.                     7.91
e+08
==========================================================================
====

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is cor
rectly specified.
[2] The condition number is large, 7.91e+08. This might indicate that ther
e are
strong multicollinearity or other numerical problems.
```

After including all the interaction terms and conducting backward selection, Buchanan had a p-value of 0.702 and was removed. The result shows all terms including interactions with p-values at the statistically significant level.

In [76]:
```
# tests model_multi
model_multi = (smf.ols(formula = 'Bush ~ Gore + Nader + Gore:Nader + Gore:
Buchanan + Nader:Buchanan', data = votes)).fit()
# your code here
model_multi = (smf.ols(formula = 'Bush ~ Gore + Nader + Gore:Nader + Gore:
Buchanan + Nader:Buchanan', data = votes)).fit()
```

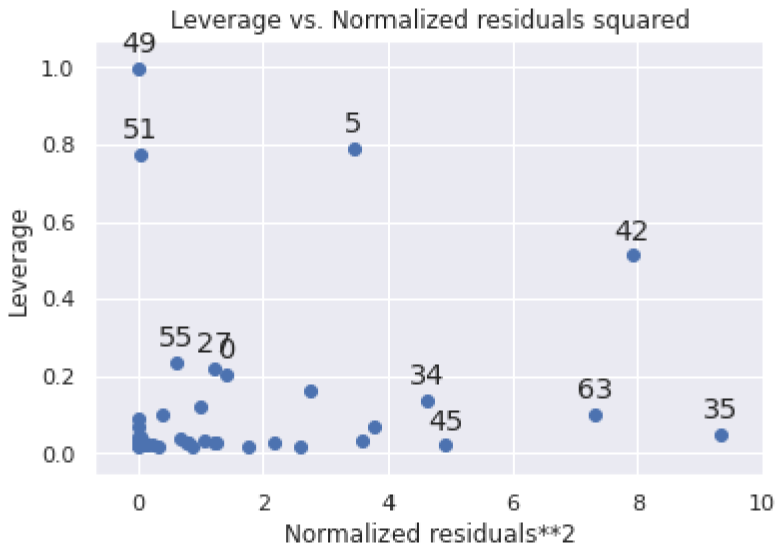
2e. Leverage [Peer Review]

Plot the *leverage* vs. the square of the residual.

These resources might be helpful

- https://rpubs.com/Amrabdelhamed611/669768
- https://www.statsmodels.org/dev/generated/statsmodels.graphics.regressionplots.plot_leverage_re

In [77]:
```
# plot the leverage vs. the square of the residual
# your code here
model_multi = (smf.ols(formula = 'Bush ~ Gore + Nader + Gore:Nader + Gore:
Buchanan + Nader:Buchanan', data = votes)).fit()
sm.graphics.plot_leverage_resid2(model_multi)
plt.show()
```



In [46]:
```
# you can use this cell to try different plots
# your code here
```

Upload your plot for this week's Peer Review assignment. If you tried out multiple models, upload a single model.

2f. Identify and Clean [5pts]

The leverage *vs* residual plot indicates that some rows have high leverage but small residuals and others have high residual. The R^2 of the model is determined by the residual. The data is from the disputed 2000 election [where one county](https://) caused significant issues.

Display the *3 or more* rows for the points indicated having high leverage and/or high residual squared. You will use this to improve the model R^2.

Name the list of indices for those high-leverage and/or high-residual points as `unusual`.

In [104]:
```
# uncomment and fill unusual with list of indices for high-leverage and/or
 high-residual points
unusual = [0, 5, 27, 34, 35, 42, 45, 49, 51, 55, 63]
# your code here
```

In [105]:
```
# tests your list of indices for high-leverage and/or high-residual points
```

2g. Final model [5 pts]

Develop your final model by dropping *one or more* of the troublesome data points indicated in the leverage *vs* residual plot and insuring any interactions in your model are still significant at $p = 0.05$. Your model should have an R^2 great than 0.95. Call your model `model_final`.

In [109]:
```
# develop your model_final here
votes2 = votes.drop(unusual, axis = 0, inplace = False)
model_final = (smf.ols(formula = 'Bush ~ Nader + Buchanan + Gore:Buchanan
+ Nader:Buchanan', data = votes2)).fit()
print(model_final.summary())
# your code here

#votes2 = votes.drop(unusual, axis = 0, inplace = False)
#votes2.head(10)
#fin_model = (smf.ols(formula = 'Bush ~ Nader + Buchanan + Gore:Buchanan +
 Nader:Buchanan', data = votes2)).fit()
#print(fin_model.summary())
```

```
                            OLS Regression Results

==============================================================================
====
Dep. Variable:                   Bush   R-squared:                       0
.958
Model:                            OLS   Adj. R-squared:                  0
.954
Method:                 Least Squares   F-statistic:                     2
89.3
Date:                Mon, 20 Jun 2022   Prob (F-statistic):           2.26
e-34
Time:                        21:10:27   Log-Likelihood:                -57
5.06
No. Observations:                  56   AIC:                             1
160.
Df Residuals:                      51   BIC:                             1
170.
Df Model:                           4

Covariance Type:            nonrobust

==============================================================================
========
                     coef    std err          t      P>|t|      [0.025
  0.975]
------------------------------------------------------------------------------
--------
Intercept       -563.0635   1721.290     -0.327      0.745   -4018.697
2892.570
Nader             24.3157      2.618      9.287      0.000      19.059
  29.572
Buchanan          60.2106     13.892      4.334      0.000      32.320
  88.101
Gore:Buchanan      0.0021      0.000      9.855      0.000       0.002
   0.003
Nader:Buchanan    -0.0611      0.008     -7.528      0.000      -0.077
  -0.045
==============================================================================
```

```
====
Omnibus:                       11.895   Durbin-Watson:                   1
.598
Prob(Omnibus):                  0.003   Jarque-Bera (JB):               13
.823
Skew:                           0.843   Prob(JB):                     0.00
0996
Kurtosis:                       4.756   Cond. No.                     3.06
e+07
==========================================================================
====

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is cor
rectly specified.
[2] The condition number is large, 3.06e+07. This might indicate that ther
e are
strong multicollinearity or other numerical problems.
```

In [22]:
```
# tests model_final
```

3. Body Mass Index Model [20 points, Peer Review]

In this problem, you will first clean a data set and create a model to estimate body fat based on the common BMI measure. Then, you will use the **forward stepwise selection** method to create more accurate predictors for body fat.

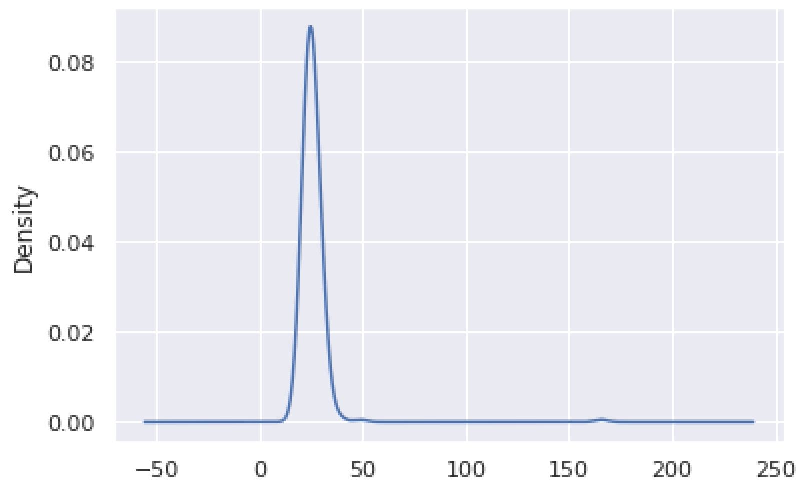
The body density dataset in file `bodyfat` includes the following 15 variables listed from left to right:

- Density : Density determined from underwater weighing
- Fat : Percent body fat from Siri's (1956) equation
- Age : Age (years)
- Weight : Weight (kg)
- Height : Height (cm)
- Neck : Neck circumference (cm)
- Chest: Chest circumference (cm)
- Abdomen : Abdomen circumference (cm)
- Hip : Hip circumference (cm)
- Thigh : Thigh circumference (cm)
- Knee : Knee circumference (cm)
- Ankle : Ankle circumference (cm)
- Biceps : Biceps (extended) circumference (cm)
- Forearm : Forearm circumference (cm)
- Wrist : Wrist circumference (cm)

The `Density` column is the "gold standard" -- it is a measure of body density obtained by dunking people in water and measuring the displacement. The `Fat` column is a prediction using another statistical model. The body mass index (BMI) is [calculated as Kg/m^2](#) and is used to classify people into different weight categories with a [BMI over 30 being 'obese'](#). You will find that BMI is a poor predictor of the `Density` information it purports to predict. You will try to find better models using measurements and regression.

Unfortunately for us, the dataset we have has imperial units for weight and height, so we will convert those to metric and then calculate the BMI and plot the KDE of the data.

In [277]:
```
fat = pd.read_csv('data/bodyfat.csv')
fat = fat.drop('Unnamed: 0', axis=1)
fat.Weight = fat.Weight * 0.453592 # Convert to Kg
fat.Height = fat.Height * 0.0254 # convert inches to m
fat['BMI'] = fat.Weight / (fat.Height**2)
fat.BMI.plot.kde();
```



3a. [5 pts]

The BMI has at least one outlier since it's unlikely anyone has a BMI of 165, even Arnold Schwarzenegger.

Form a new table `cfat` (cleaned fat) that removes any rows with a BMI greater than 40 and calculate the regression model predicting the `Density` from the `BMI`. Display the summary of the regression model. Call your model as `bmi`. You should achieve an R^2 of at least 0.53.

In [278]:
```
# form new table cfat and model bmi
cfat = fat.drop(fat.index[fat['BMI'] > 40], inplace = False)
bmi = (smf.ols(formula = 'Density ~ BMI', data = cfat)).fit()
# your code here
fat.head(10)
mod = (smf.ols(formula = 'Density ~ BMI', data = cfat)).fit()
print(mod.summary())
```

```
                            OLS Regression Results

===========================================================================
====
Dep. Variable:                Density   R-squared:                        0
.536
Model:                            OLS   Adj. R-squared:                   0
.534
Method:                 Least Squares   F-statistic:                      2
86.2
Date:                Tue, 21 Jun 2022   Prob (F-statistic):            3.25
```

```
e-43
Time:                        02:45:49   Log-Likelihood:                    73
4.17
No. Observations:                 250   AIC:                               -1
464.
Df Residuals:                     248   BIC:                               -1
457.
Df Model:                           1

Covariance Type:            nonrobust

==============================================================================
====
                 coef    std err          t      P>|t|      [0.025      0.
975]
------------------------------------------------------------------------------
----
Intercept      1.1602      0.006    186.410      0.000       1.148       1
.172
BMI           -0.0041      0.000    -16.918      0.000      -0.005      -0
.004
==============================================================================
====
Omnibus:                        2.262   Durbin-Watson:                     1
.576
Prob(Omnibus):                  0.323   Jarque-Bera (JB):                  2
.259
Skew:                           0.229   Prob(JB):                          0
.323
Kurtosis:                       2.916   Cond. No.
195.
==============================================================================
====

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is cor
rectly specified.
```

In [279]:

```
# tests your bmi model
```

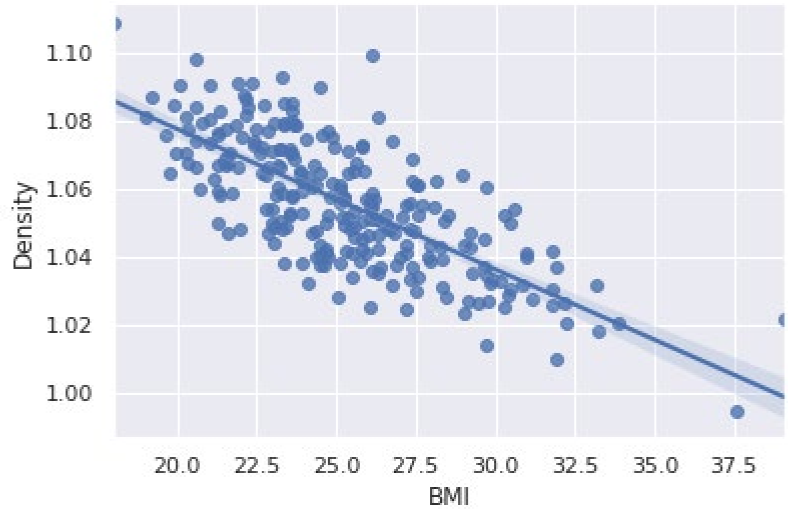
3b. [Peer Review]

Plot your regression model against the BMI measurement, properly labeling the scatterplot axes and showing the regression line. In subsequent models, you will not be able to plot the Density *vs* your predictors because you will have too many predictors, but it's useful to visually understand the relationship between the BMI predictor and the `Density` because you should find that the regression line goes through the data but there is too much variability in the data to achieve a good R^2. Upload a copy or screensho of your plot for this week's Peer Review assignment.

In [280]:

```
# plot regression model against BMI measurement
# properly label the scatterplot axs and show the regression line
# your code here
sns.regplot(x = "BMI", y = "Density", data = cfat)
```

Out[280]: <matplotlib.axes._subplots.AxesSubplot at 0x7fbe765ebfd0>



The `BMI` model uses easy-to-measure predictors, but has a poor $R^2 \sim 0.54$. We will use structured subset selection methods from ISLR Chapter 6.1 to derive two better predictors. That chapter covers *best subset*, *forward stepwise* and *backware stepwise* selection. I have implemented the *best subset* selection which searches across all combinations of $1, 2, \ldots, p$ predictors and selects the best predictor based on the $\text{adjusted } R^2$ metric. This method involved analyzing $2^{13} = 8192$ regression models (programming and computers for the win). The resulting $\text{adjusted } R^2$ plot is shown below (Since the data split can be different, your result may look slightly different):

In this plot, `test_fat` and `train_fat` datasets each containing 200 randomly selected samples were derived from the `cfat` dataset using `np.random.choice` over the `cfat.index` and selected using the Pandas `loc` method. Then, following the algorithm of ISLR Algorithm 6.1 *Best Subset Selection*, all $\binom{p}{k}$ models with k predictors were evaluated on the training data and the model returning the best \textit{Adjusted}~R^2 was selected. These models are indicated by the data points for the solid blue line. As the text indicates, other measures (AIC, BIC, C_p) would be better than the \textit{Adjusted}~R^2, but we use it becuase because you've already seen the R^2 and should have an understanding of what it means.

Then, the best models for each k were evaluated for the `test_fat` data. These results are shown as the red dots below the blue line. Note that because the test and train datasets are randomly selected subsets, the results vary from run-to-run and it may that your test data produces better R^2 than your training data.

In the following exercises, you can not use the `Density`, `Fat` or `BMI` columns in your predictive models. You can only use the 13 predictors in the `allowed_factors` list.

In [281]:
```
allowed_factors = ['Age', 'Weight', 'Height', 'Neck', 'Chest',
          'Abdomen', 'Hip', 'Thigh', 'Knee', 'Ankle', 'Biceps', 'Forearm',
          'Wrist']
```

Forward Stepwise Refinement

You will manually perform the steps of the *forward stepwise selection* method for four parameters. You will do this following Algorithm 6.2 from ISLR. For k = 1\ldots 4:

- Set up a regression model with k factors that involves the fixed predictors from the previous step k-1
- Try all p predictors in the new kth position
- Select the best parameter using \textit{Adjusted}-R^2 (e.g. `model.rsquared_adj`) given your training data
- Fix the new parameter and continue the process for k+1

Then, you will construct a plot similar to the one above, plotting the \textit{Adjusted}-R^2 for each of your k steps and plotting the \textit{Adjusted}-R^2 from the test set using that model.

3c. [5 pts]

First, construct your training and test sets from your `cfat` dataset. Call the resulting data frame to train_fat and test_fat. train_fat includes randomly selected 125 observations and the test_fat has the rest.

In [305]:
```
# construct train_fat and test_fat from cfat dataset
# your code here
from sklearn.model_selection import train_test_split
train_fat, test_fat = train_test_split(cfat, test_size = 0.5)
```

In [306]:
```
# tests your training and test sets
```

3d. Conduct the algorithm above for k=1, leaving your best solution as the answer [5 pts]

Call your resulting model `train_bmi1` .

In [329]:
```
best = ['',0]
for p in allowed_factors:
    model  = smf.ols(formula='Density~'+p, data=train_fat).fit()
    print(p, model.rsquared)
    if model.rsquared>best[1]:
        best = [p, model.rsquared]
print('best:',best)
```

```
Age 0.056156009804181006
Weight 0.47272208994879195
Height 0.023905014171772487
Neck 0.34891542273413856
Chest 0.581691613482783
Abdomen 0.7113570126776247
Hip 0.43687966113631316
Thigh 0.3354506796789819
Knee 0.33192832717313114
Ankle 0.111789910117295
Biceps 0.3097536169434165
```

```
Forearm 0.23716964519406114
Wrist 0.16503438357618605
best: ['Abdomen', 0.7113570126776247]
```

In [357]:

```python
# uncomment and update your solution
def findbestmod(k, data):

    allowed_factors = ['Age', 'Weight', 'Height', 'Neck', 'Chest',
       'Abdomen', 'Hip', 'Thigh', 'Knee', 'Ankle', 'Biceps', 'Forearm',
       'Wrist']

    temp = []
    best = ['', 0]
    bestmodels = {}

    for p in allowed_factors:
        model = smf.ols(formula='Density ~ '+p, data=data).fit()
        if model.rsquared > best[1]:
            best = [p, model.rsquared]

    temp.append(best[0])

    if k == 1:
        bestmodels['features'] = best[0]
        bestmodels['R2'] = best[1]


    else:
        for i in range(k-1):

            for p in allowed_factors:
                param = temp[i] + '+' + p
                model = (smf.ols(formula = 'Density ~ ' + param, data = da
ta).fit())
                if model.rsquared > best[1]:
                    best = [param, model.rsquared]
            temp.append(best[0])

    bestmodels['features'] = best[0]
    bestmodels['R2'] = best[1]

    return bestmodels, model
# your code here
feature_selection1, train_bmi1 = findbestmod(1, train_fat)[0], findbestmod
(1, train_fat)[1]
feature_test1, test_bmi1 = findbestmod(1, test_fat)[0], findbestmod(1, tes
t_fat)[1]
print(feature_selection1, train_bmi1, test_bmi1)
```

```
{'features': 'Abdomen', 'R2': 0.7113570126776247} <statsmodels.regression.
linear_model.RegressionResultsWrapper object at 0x7fbe7695ea50> <statsmode
ls.regression.linear_model.RegressionResultsWrapper object at 0x7fbe767fda
10>
```

In [337]:

```python
# tests train_bmi1 model
```

3e. Conduct the algorithm above for k=2, leaving your best solution as the answer [Peer Review]

Name your model object as `train_bmi2`.
Look at this week's Peer Review assignment for questions about k=2 through k=5.

In [346]:

```
# your code here
def findbestmod(k, data):

    allowed_factors = ['Age', 'Weight', 'Height', 'Neck', 'Chest',
       'Abdomen', 'Hip', 'Thigh', 'Knee', 'Ankle', 'Biceps', 'Forearm',
       'Wrist']

    temp = []
    best = ['', 0]
    bestmodels = {}

    for p in allowed_factors:
        model = smf.ols(formula='Density ~ '+p, data=data).fit()
        if model.rsquared > best[1]:
            best = [p, model.rsquared]

    temp.append(best[0])

    if k == 1:
        return best

    else:
        for i in range(k-1):

            for p in allowed_factors:
                param = temp[i] + '+' + p
                model = (smf.ols(formula = 'Density ~ ' + param, data = da
ta).fit())
                if model.rsquared > best[1]:
                    best = [param, model.rsquared]
            temp.append(best[0])

    bestmodels['features'] = best[0]
    bestmodels['R2'] = best[1]

    return bestmodels, model
# your code here
feature_selection2, train_bmi2 = findbestmod(2, train_fat)[0], findbestmod
(2,train_fat)[1]
feature_test2, test_bmi2 = findbestmod(2, test_fat)[0], findbestmod(2, tes
t_fat)[1]
print(feature_selection2, train_bmi2, test_bmi2)
```

```
{'features': 'Abdomen+Ankle', 'R2': 0.7583638814937845} <statsmodels.regre
ssion.linear_model.RegressionResultsWrapper object at 0x7fbe795698d0> <sta
tsmodels.regression.linear_model.RegressionResultsWrapper object at 0x7fbe
78eea090>
0.7583638814937845
```

3f. Conduct the algorithm above for k=3, leaving your best solution as the answer [Peer Review]

In [339]:

```
# your code here
def findbestmod(k, data):

    allowed_factors = ['Age', 'Weight', 'Height', 'Neck', 'Chest',
       'Abdomen', 'Hip', 'Thigh', 'Knee', 'Ankle', 'Biceps', 'Forearm',
       'Wrist']

    temp = []
    best = ['', 0]
    bestmodels = {}

    for p in allowed_factors:
        model = smf.ols(formula='Density ~ '+p, data=data).fit()
        if model.rsquared > best[1]:
            best = [p, model.rsquared]

    temp.append(best[0])

    if k == 1:
        return best

    else:
        for i in range(k-1):

            for p in allowed_factors:
                param = temp[i] + '+' + p
                model = (smf.ols(formula = 'Density ~ ' + param, data = da
ta).fit())
                if model.rsquared > best[1]:
                    best = [param, model.rsquared]
            temp.append(best[0])

    bestmodels['features'] = best[0]
    bestmodels['R2'] = best[1]

    return bestmodels, model
# your code here
feature_selection3, train_bmi3 = findbestmod(3, train_fat)[0], findbestmod
(3, train_fat)[1]
feature_test3, test_bmi3 = findbestmod(3, test_fat)[0], findbestmod(3, tes
t_fat)[1]
print(feature_selection3, train_bmi3, test_bmi3)
```

```
{'features': 'Abdomen+Ankle+Forearm', 'R2': 0.7656745901351671} <statsmode
ls.regression.linear_model.RegressionResultsWrapper object at 0x7fbe793a31
50> <statsmodels.regression.linear_model.RegressionResultsWrapper object a
t 0x7fbe76a66ed0>
```

3g. Conduct the algorithm above for k=4, leaving your best solution as the answer [Peer Review]

In [340]:

```
# your code here
```

```
def findbestmod(k, data):

    allowed_factors = ['Age', 'Weight', 'Height', 'Neck', 'Chest',
       'Abdomen', 'Hip', 'Thigh', 'Knee', 'Ankle', 'Biceps', 'Forearm',
       'Wrist']

    temp = []
    best = ['', 0]
    bestmodels = {}

    for p in allowed_factors:
        model = smf.ols(formula='Density ~ '+p, data=data).fit()
        if model.rsquared > best[1]:
            best = [p, model.rsquared]

    temp.append(best[0])

    if k == 1:
        return best

    else:
        for i in range(k-1):

            for p in allowed_factors:
                param = temp[i] + '+' + p
                model = (smf.ols(formula = 'Density ~ ' + param, data = da
ta).fit())
                if model.rsquared > best[1]:
                    best = [param, model.rsquared]
            temp.append(best[0])

    bestmodels['features'] = best[0]
    bestmodels['R2'] = best[1]

    return bestmodels, model
# your code here
feature_selection4, train_bmi4 = findbestmod(4, train_fat)[0], findbestmod
(4, train_fat)[1]
feature_test4, test_bmi4 = findbestmod(4, test_fat)[0], findbestmod(4, tes
t_fat)[1]
print(feature_selection4, train_bmi4, test_bmi4)
```

```
{'features': 'Abdomen+Ankle+Forearm+Neck', 'R2': 0.7737305308938128} <stat
smodels.regression.linear_model.RegressionResultsWrapper object at 0x7fbe7
9453690> <statsmodels.regression.linear_model.RegressionResultsWrapper obj
ect at 0x7fbe796bab10>
```

3h. Conduct the algorithm above for k=5, leaving your best solution as the answer [Peer Review]

In [341]:

```
# your code here
def findbestmod(k, data):

    allowed_factors = ['Age', 'Weight', 'Height', 'Neck', 'Chest',
       'Abdomen', 'Hip', 'Thigh', 'Knee', 'Ankle', 'Biceps', 'Forearm',
```

```
        'Wrist']

    temp = []
    best = ['', 0]
    bestmodels = {}

    for p in allowed_factors:
        model = smf.ols(formula='Density ~ '+p, data=data).fit()
        if model.rsquared > best[1]:
            best = [p, model.rsquared]

    temp.append(best[0])

    if k == 1:
        return best

    else:
        for i in range(k-1):

            for p in allowed_factors:
                param = temp[i] + '+' + p
                model = (smf.ols(formula = 'Density ~ ' + param, data = da
ta).fit())
                if model.rsquared > best[1]:
                    best = [param, model.rsquared]
            temp.append(best[0])

    bestmodels['features'] = best[0]
    bestmodels['R2'] = best[1]

    return bestmodels, model
# your code here
feature_selection5, train_bmi5 = findbestmod(5, train_fat)[0], findbestmod
(5, train_fat)[1]
feature_test5, test_bmi5 = findbestmod(5, test_fat)[0], findbestmod(5, tes
t_fat)[1]
print(feature_selection5, train_bmi5, test_bmi5)
print(feature_selection5["R2"])
```

```
{'features': 'Abdomen+Ankle+Forearm+Neck+Hip', 'R2': 0.7780431666108075} <
statsmodels.regression.linear_model.RegressionResultsWrapper object at 0x7
fbe7662f850> <statsmodels.regression.linear_model.RegressionResultsWrapper
 object at 0x7fbe76963f10>
0.7780431666108075
```

3i. Plot [5 pts]

Plot your resulting \textit{adjusted}~R^2 *vs* number of predictors (k=1,2,3,4,5) and overlay the \textit{adjusted}~R^2 for the test data. Call the list of the five adjusted r-squared values from the five train_bmi# models as `adjr2_train` and the one from the test data as `adjr2_test`.

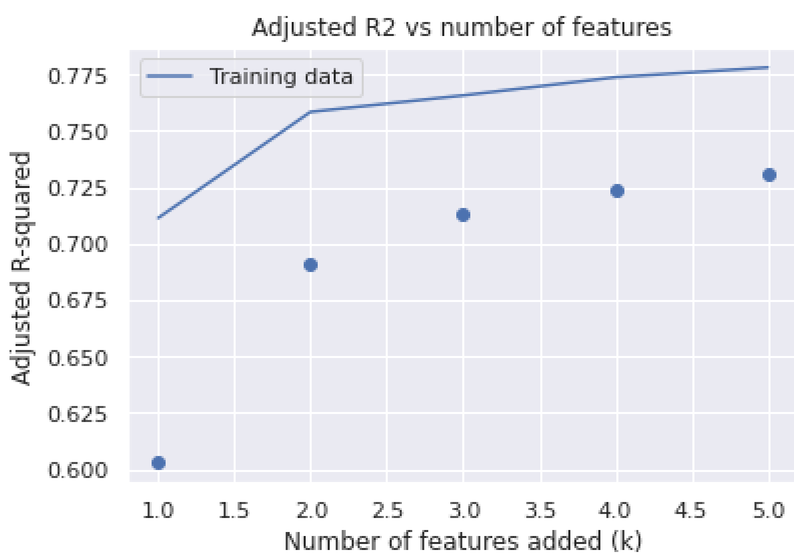
In [359]:

```
# plot resulting adjusted rsquared vs number of predictors (k=1,2,3,4,5)
# overlay the adjusted rsquared for the test data
# your code here
adjr2_train = [feature_selection1["R2"], feature_selection2["R2"], feature
```

```
_selection3["R2"], feature_selection4["R2"], feature_selection5["R2"]]
print(type(feature_selection1))
adjr2_test = [feature_test1["R2"], feature_test2["R2"], feature_test3["R2"
], feature_test4["R2"], feature_test5["R2"]]
print(adjr2_train)
print(adjr2_test)
x = range(1, 6, 1)
#plt.scatter(x, adjr2_train)
plt.scatter(x, adjr2_test)
plt.plot(x, adjr2_train, label = 'Training data')
#plt.plot(x, adjr2_test, label = 'Test data')
plt.xlabel('Number of features added (k)')
plt.ylabel('Adjusted R-squared')
plt.title('Adjusted R2 vs number of features')
plt.legend()
plt.show()

#plt.plot(x, adjr2_train)
```

```
<class 'dict'>
[0.7113570126776247, 0.7583638814937845, 0.7656745901351671, 0.77373053089
38128, 0.7780431666108075]
[0.6028799131561996, 0.6909372707485798, 0.7134568594224162, 0.72410883321
01564, 0.7305904404059493]
```



In [241]:
```
# tests adjusted r-squared plot vs. number of factors
```

3j. Discussion [Peer Review]

The BMI model has the benefit being simple (two measurements, height and wright). Looking at your resulting regression model, how many parameters would you suggest to use for your enhanced BMI model? Justify your answer using your models. Submit your answer with this week's Peer Review assignment.

Solution Looking at the enhanced BMI models, it is clear that as we add features to the model we get a better adjusted rsquared value. More specifically, as we go from k=1 to k=5, the adjusted rsquared increases at smaller k value and starts to shallow down in slope, resembling a plateau shape around k=4 and k=5. This suggests that there may be a "sweet spot" in producing the best BMI model, which in this case appears to be about five parameters.

In [ ]:

```

```