Marketing

February 21, 2025

1 Marketing Exercise

1.1 Import notebook functions

[1]:

```
from notebookfuncs import *
```

1.2 Import standard libraries

[2]:

```
import numpy as np
import pandas as pd
from matplotlib.pyplot import subplots
```

1.3 New imports

[3]:

```
import statsmodels.api as sm
```

1.4 Import statsmodel.objects

[4]:

```
from statsmodels.stats.outliers_influence import variance_inflation_factor as␣
  ↪VIF
from statsmodels.stats.outliers_influence import summary_table
from statsmodels.stats.anova import anova_lm
```

1.5 Import ISLP objects

[5]:

```
import ISLP
from ISLP import models
from ISLP import load_data
from ISLP.models import ModelSpec as MS, summarize, poly
```

Inspecting objects and namespaces

[6]:

```
dir()
```

[6]:

```
['Audio',
 'ISLP',
 'In',
 'InteractiveShell',
'Latex',
'MS',
'Markdown',
'Math',
'Out',
'VIF',
'_',
'__',
'___',
'__builtin__',
'__builtins__',
'__doc__',
'__loader__',
'__name__',
'__package__',
'__spec__',
'_dh',
'_i',
'_i1',
'_i2',
'_i3',
'_i4',
'_i5',
'_i6',
'_ih',
'_ii',
'_iii',
'_oh',
'allDone',
'anova_lm',
'display',
'exit',
'get_ipython',
'load_data',
'models',
'np',
'open',
'pd',
'poly',
'printlatex',
'printmd',
'quit',
'sm',
'subplots',
'summarize',
'summary_table']
```

[7]:

```
Advertising = pd.read_csv("Advertising.csv")
# Drop first column
Advertising = Advertising.iloc[:, 1:]
Advertising.head()
```

[7]:

```
      TV  Radio  Newspaper  Sales
0  230.1   37.8       69.2   22.1
1   44.5   39.3       45.1   10.4
2   17.2   45.9       69.3    9.3
3  151.5   41.3       58.5   18.5
4  180.8   10.8       58.4   12.9
```

[8]:

```
Advertising.describe()
```

[8]:

```
               TV       Radio   Newspaper       Sales
count  200.000000  200.000000  200.000000  200.000000
mean   147.042500   23.264000   30.554000   14.022500
std     85.854236   14.846809   21.778621    5.217457
min      0.700000    0.000000    0.300000    1.600000
25%     74.375000    9.975000   12.750000   10.375000
50%    149.750000   22.900000   25.750000   12.900000
75%    218.825000   36.525000   45.100000   17.400000
max    296.400000   49.600000  114.000000   27.000000
```

1.6 Is there a relationship between sales and advertising budget?

[9]:

```
y = Advertising["Sales"]
cols = list(Advertising.columns)
cols.remove("Sales")
X = MS(cols).fit_transform(Advertising)
model = sm.OLS(y, X)
results = model.fit()
print("F-value", results.fvalue)
print("F-pvalue", results.f_pvalue)
summarize(results)
```

```
F-value 570.2707036590944
F-pvalue 1.575227256092416e-96
```

[9]:

```
             coef  std err       t  P>|t|
intercept  2.9389    0.312   9.422   0.00
TV         0.0458    0.001  32.809   0.00
Radio      0.1885    0.009  21.893   0.00
Newspaper -0.0010    0.006  -0.177   0.86
```

[10]:

```
dir(models)
```

```
[10]: ['Column',
       'Feature',
       'FeatureSelector',
       'ModelSpec',
       'Stepwise',
       'StringIO',
       '__builtins__',
       '__cached__',
       '__doc__',
       '__file__',
       '__loader__',
       '__name__',
       '__package__',
       '__path__',
       '__spec__',
       'bs',
       'build_columns',
       'columns',
       'contrast',
       'derived_feature',
       'generic_selector',
       'min_max_strategy',
       'model_spec',
       'np',
       'ns',
       'pca',
       'pd',
       'poly',
       'sklearn_selected',
       'sklearn_selection_path',
       'sklearn_sm',
       'sklearn_wrap',
       'strategy',
       'summarize']
```

- The p-value corresponding to the F-statistic is very low. Thus, clear evidence of a relationship between sales and advertising budget.

```
[11]: dir(results)
```

```
[11]: ['HC0_se',
       'HC1_se',
       'HC2_se',
       'HC3_se',
       '_HCCM',
       '__class__',
       '__delattr__',
       '__dict__',
       '__dir__',
       '__doc__',
       '__eq__',
       '__format__',
       '__ge__',
       '__getattribute__',
       '__getstate__',
       '__gt__',
       '__hash__',
       '__init__',
       '__init_subclass__',
       '__le__',
       '__lt__',
       '__module__',
       '__ne__',
       '__new__',
       '__reduce__',
       '__reduce_ex__',
       '__repr__',
       '__setattr__',
       '__sizeof__',
       '__str__',
       '__subclasshook__',
       '__weakref__',
       '_abat_diagonal',
       '_cache',
       '_data_attr',
       '_data_in_cache',
       '_get_robustcov_results',
       '_get_wald_nonlinear',
       '_is_nested',
       '_transform_predict_exog',
       '_use_t',
       '_wexog_singular_values',
       'aic',
       'bic',
       'bse',
       'centered_tss',
       'compare_f_test',
       'compare_lm_test',
       'compare_lr_test',
       'condition_number',
       'conf_int',
       'conf_int_el',
       'cov_HC0',
       'cov_HC1',
       'cov_HC2',
       'cov_HC3',
       'cov_kwds',
       'cov_params',
       'cov_type',
       'df_model',
       'df_resid',
       'diagn',
       'eigenvals',
       'el_test',
       'ess',
       'f_pvalue',
       'f_test',
       'fittedvalues',
       'fvalue',
       'get_influence',
       'get_prediction',
       'get_robustcov_results',
       'info_criteria',
       'initialize',
       'k_constant',
       'llf',
       'load',
       'model',
       'mse_model',
       'mse_resid',
       'mse_total',
       'nobs',
       'normalized_cov_params',
       'outlier_test',
       'params',
       'predict',
       'pvalues',
       'remove_data',
       'resid',
       'resid_pearson',
       'rsquared',
       'rsquared_adj',
       'save',
       'scale',
       'ssr',
       'summary',
       'summary2',
       't_test',
       't_test_pairwise',
       'tvalues',
       'uncentered_tss',
       'use_t',
       'wald_test',
       'wald_test_terms',
       'wresid']
```

1.7 How strong is the relationship?

```
[12]: results.summary()
```

```
[12]:
Dep. Variable:           Sales          R-squared:              0.897
Model:                   OLS            Adj. R-squared:         0.896
Method:                  Least Squares  F-statistic:            570.3
Date:                    Fri, 21 Feb 2025  Prob (F-statistic):  1.58e-96
Time:                    19:19:32       Log-Likelihood:         -386.18
No. Observations:        200            AIC:                    780.4
Df Residuals:            196            BIC:                    793.6
Df Model:                3
Covariance Type:         nonrobust

                coef    std err    t       P> |t|   [0.025   0.975]
intercept      2.9389   0.312     9.422    0.000    2.324    3.554
TV             0.0458   0.001    32.809    0.000    0.043    0.049
Radio          0.1885   0.009    21.893    0.000    0.172    0.206
Newspaper     -0.0010   0.006    -0.177    0.860   -0.013    0.011

    Omnibus:          60.414   Durbin-Watson:       2.084
    Prob(Omnibus):     0.000   Jarque-Bera (JB):  151.241
    Skew:             -1.327   Prob(JB):         1.44e-33
    Kurtosis:          6.332   Cond. No.             454.
```

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

```
[13]: y.mean()
```

```
[13]: 14.0225
```

```
[14]: results.resid.std()
```

```
[14]: 1.6727572743844117
```

```
[15]: (results.resid.std() / y.mean()) * 100
```

```
[15]: 11.929094486606608
```

- The residual standard error (RSE) is 1.67 and the mean value of the response is 14.023 which translates to a percentage error of roughly 11.93%

```
[16]: ("R-squared", results.rsquared, "Adjusted R-squared", results.rsquared_adj)
```

```
[16]: ('R-squared', 0.8972106381789522, 'Adjusted R-squared', 0.8956373316204668)
```

- The R2 explains about 90% of the variance in Sales.

1.8 Which media are associated with Sales?

- The low p-values for Radio and TV suggest that only they are related to Sales.

1.9 How large is the association between each medium and sales?

[17]:

```
results.conf_int(alpha=0.05)
```

[17]:

```
                  0         1
intercept  2.323762  3.554016
TV         0.043014  0.048516
Radio      0.171547  0.205513
Newspaper -0.012616  0.010541
```

- The confidence intervals for TV and Radio are narrow and far from zero. This provides evidence that these media are related to sales.
- The interval for Newspaper includes zero indicating that it is not statistically significant given values of TV and Radio.

[18]:

```
vals = [VIF(X, i) for i in range(1, X.shape[1])]
print(vals)
```

```
[1.00461078493965, 1.1449519171055353, 1.1451873787239288]
```

- The VIF scores are 1.005, 1.145 and 1.145 respectively for TV, radio and newspaper. These suggest no evidence of collinearity as an explnation for wide standard errors for newspaper.
- In order to assess the association of each medium individually on sales, we can perform three separate linear regressions.

[19]:

```
TV = MS(["TV"]).fit_transform(Advertising)
model = sm.OLS(y, TV)
results = model.fit()
print(summarize(results))
Radio = MS(["Radio"]).fit_transform(Advertising)
model = sm.OLS(y, Radio)
results = model.fit()
print(summarize(results))
Newspaper = MS(["Newspaper"]).fit_transform(Advertising)
model = sm.OLS(y, Newspaper)
results = model.fit()
print(summarize(results))
```

```
             coef  std err       t  P>|t|
intercept  7.0326    0.458  15.360    0.0
TV         0.0475    0.003  17.668    0.0
             coef  std err       t  P>|t|
intercept  9.3116    0.563  16.542    0.0
```

```
Radio      0.2025    0.020   9.921    0.0
              coef  std err       t  P>|t|
intercept  12.3514    0.621  19.876  0.000
Newspaper   0.0547    0.017   3.300  0.001
```

Looking at the p-values, there is evidence of a strong association b/w TV and sales and radio and sales. There is evidence of a mild association between Newspaper and sales when TV and radio are ignored.

1.10 How accurately can we predict future sales?

- Given that $100,000 is spent on TV advertising, and $20,000 is spent on Radio advertising, we need to compute the 95% Confidence intervals for each city (i.e., the mean) and the prediction interval for a particular city (also at 95% confidence intervals).

1.10.1 Fit the regression dropping the Newspaper column as insignificant

[20]:

```
y = Advertising["Sales"]
cols = list(Advertising.columns)
cols.remove("Sales")
cols.remove("Newspaper")
X = MS(cols).fit_transform(Advertising)
model = sm.OLS(y, X)
results = model.fit()
print("F-value", results.fvalue)
print("F-pvalue", results.f_pvalue)
summarize(results)
```

```
F-value 859.6177183058211
F-pvalue 4.8273618513354486e-98
```

[20]:

```
             coef  std err       t  P>|t|
intercept  2.9211    0.294   9.919    0.0
TV         0.0458    0.001  32.909    0.0
Radio      0.1880    0.008  23.382    0.0
```

[21]:

```
results.summary()
```

[21]:

Dep. Variable:	Sales	**R-squared:**	0.897
Model:	OLS	**Adj. R-squared:**	0.896
Method:	Least Squares	**F-statistic:**	859.6
Date:	Fri, 21 Feb 2025	**Prob (F-statistic):**	4.83e-98
Time:	19:19:33	**Log-Likelihood:**	-386.20
No. Observations:	200	**AIC:**	778.4
Df Residuals:	197	**BIC:**	788.3
Df Model:	2		
Covariance Type:	nonrobust		

	coef	std err	t	P> |t|	[0.025	0.975]
intercept	2.9211	0.294	9.919	0.000	2.340	3.502
TV	0.0458	0.001	32.909	0.000	0.043	0.048
Radio	0.1880	0.008	23.382	0.000	0.172	0.204

Omnibus:	60.022	**Durbin-Watson:**	2.081
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	148.679
Skew:	-1.323	**Prob(JB):**	5.19e-33
Kurtosis:	6.292	**Cond. No.**	425.

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

[22]:
```
design = MS(["TV", "Radio"])
new_df = pd.DataFrame({"TV": [100], "Radio": [20]})
print(new_df)
new_X = design.fit_transform(new_df)
new_predictions = results.get_prediction(new_X)
new_predictions.predicted_mean
```

```
    TV  Radio
0  100     20
```

[22]:
```
array([11.25646595])
```

1.10.2 We predict the confidence intervals at 95% as follows:

[23]:
```
new_predictions.conf_int(alpha=0.05)
```

[23]:
```
array([[10.98525445, 11.52767746]])
```

1.10.3 We predict the prediction interval for a particular city as follows:

[24]:
```
new_predictions.conf_int(alpha=0.05, obs=True)
```

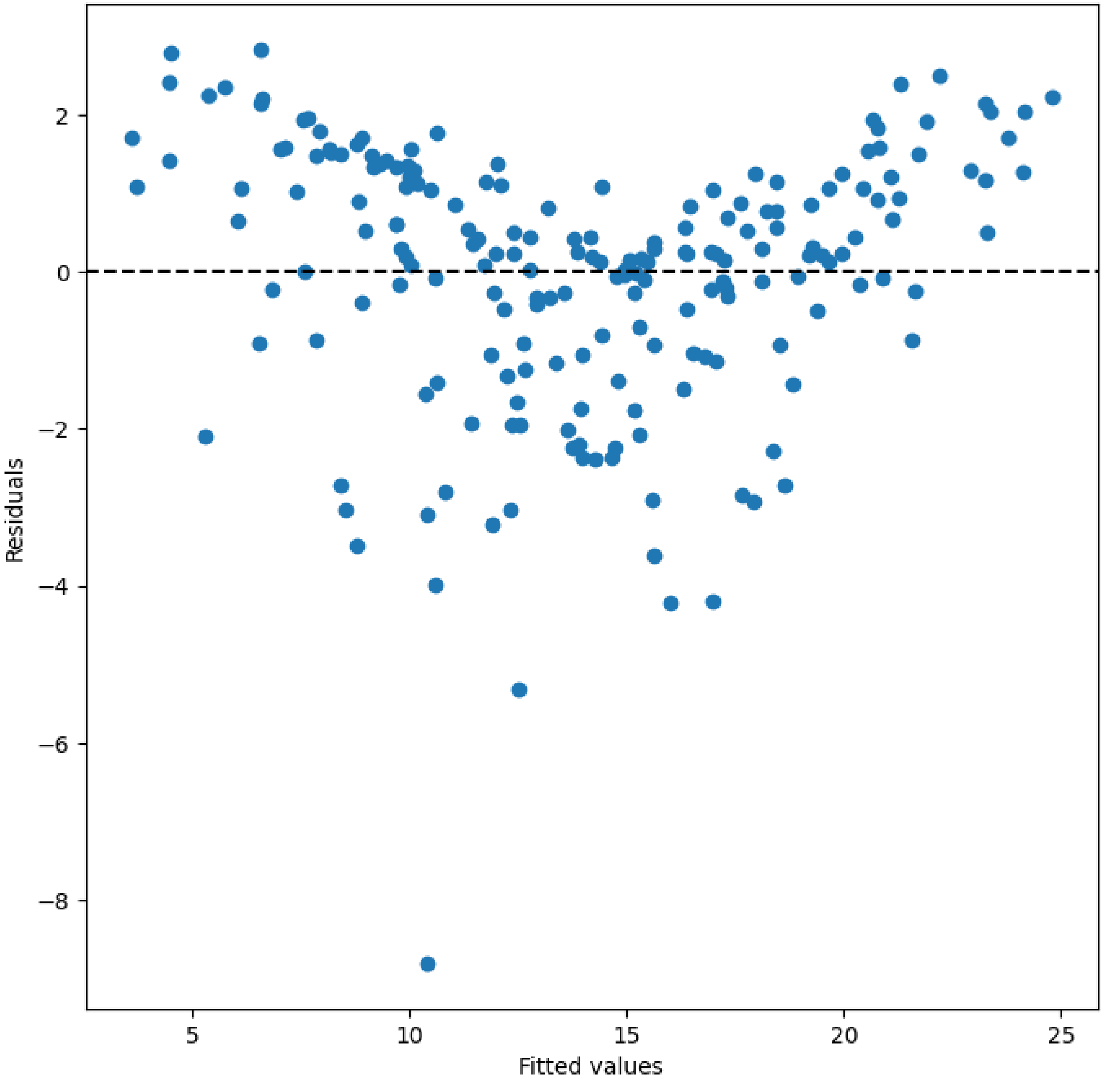
[24]:
```
array([[ 7.92961607, 14.58331584]])
```

- Both intervals are centered at 11,256 but the prediction intervals are wider reflecting the additional uncertainty around sales for a particular city as against the average sales for many locations.

1.10.4 Is the relationship linear?

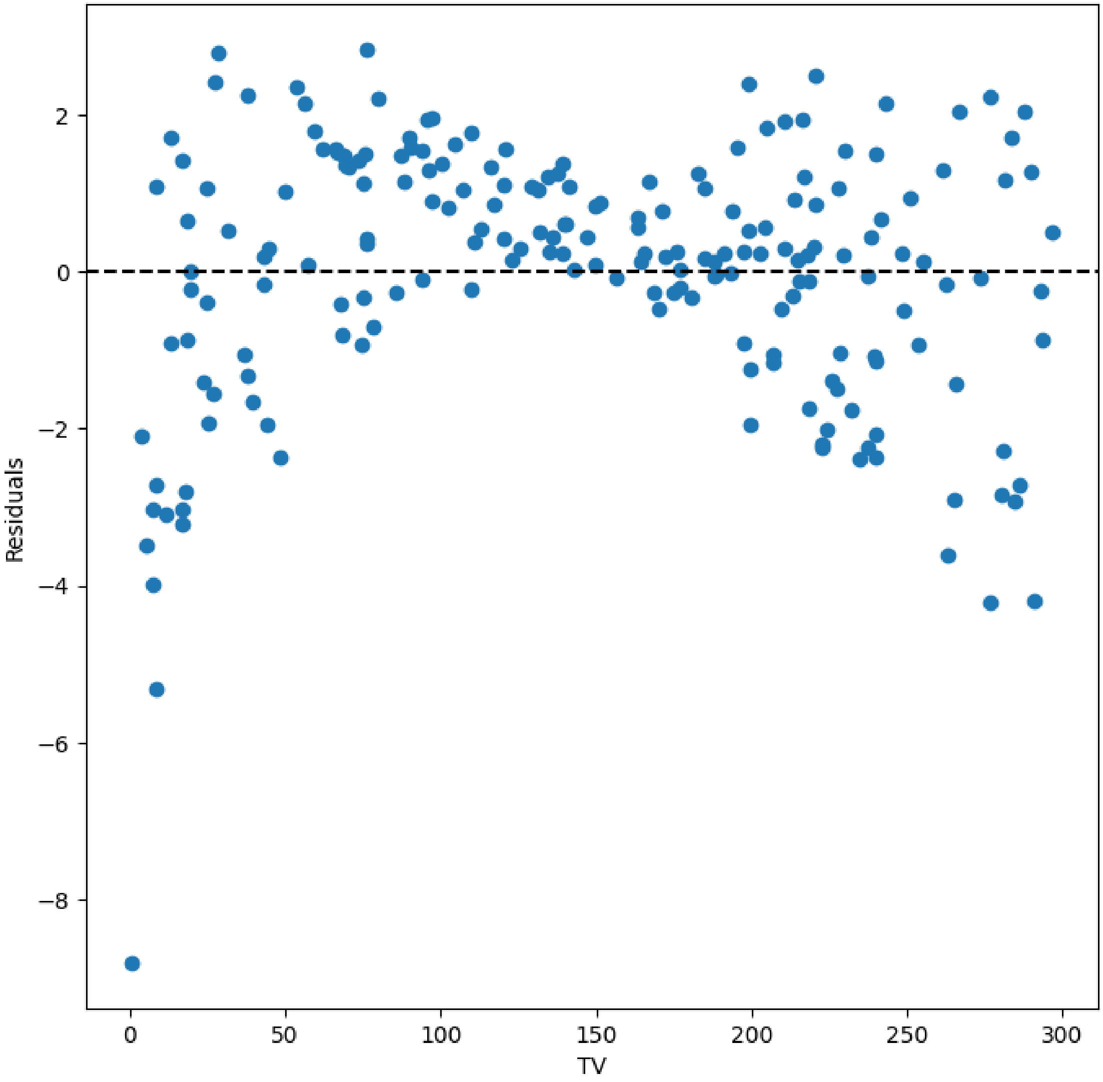
[25]:
```
_, ax = subplots(figsize=(8, 8))
ax.scatter(results.fittedvalues, results.resid)
ax.set_xlabel("Fitted values")
ax.set_ylabel("Residuals")
ax.axhline(0, c="k", ls="--")
```

```
[25]: <matplotlib.lines.Line2D at 0x7bd0434ad790>
```



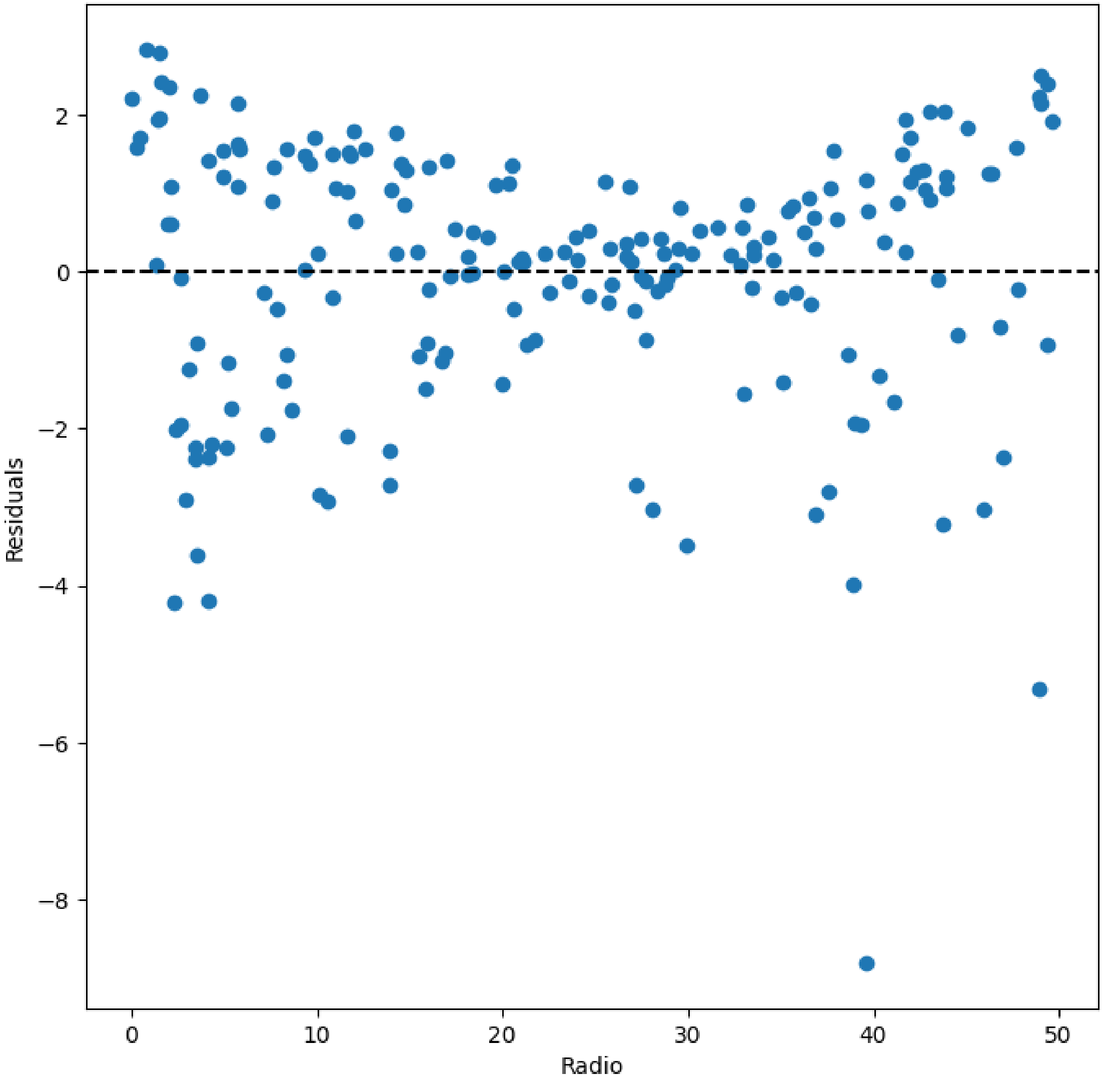
```
[26]: _, ax = subplots(figsize=(8, 8))
      ax.scatter(Advertising["TV"], results.resid)
      ax.set_xlabel("TV")
      ax.set_ylabel("Residuals")
      ax.axhline(0, c="k", ls="--")
```

```
[26]: <matplotlib.lines.Line2D at 0x7bd040ba93d0>
```



```
[27]: _, ax = subplots(figsize=(8, 8))
      ax.scatter(Advertising["Radio"], results.resid)
      ax.set_xlabel("Radio")
      ax.set_ylabel("Residuals")
      ax.axhline(0, c="k", ls="--")
```

```
[27]: <matplotlib.lines.Line2D at 0x7bd0434aee10>
```



- There is evidence of non-linearity in the model from the residuals plotted against the fitted values. Looking at the residuals versus predictors plots, it appears that TV is a better candidate for quadratification.

[28]:

```
X = MS([poly("TV", degree=2, raw=True), "Radio"]).fit_transform(Advertising)
model = sm.OLS(y, X)
results = model.fit()
summarize(results)
```

[28]:

```
                                   coef   std err       t  P>|t|
intercept                        1.2876  0.359000   3.588    0.0
poly(TV, degree=2, raw=True)[0]  0.0784  0.005000  15.736    0.0
poly(TV, degree=2, raw=True)[1] -0.0001  0.000017  -6.775    0.0
Radio                            0.1930  0.007000  26.465    0.0
```

[29]:

```python
results.summary()
```

[29]:

```
                      Dep. Variable:        Sales             R-squared:             0.917
                      Model:                OLS               Adj. R-squared:        0.915
                      Method:               Least Squares     F-statistic:           719.0
                      Date:                 Fri, 21 Feb 2025  Prob (F-statistic):    1.80e-105
                      Time:                 19:19:33          Log-Likelihood:        -365.16
                      No. Observations:     200               AIC:                   738.3
                      Df Residuals:         196               BIC:                   751.5
                      Df Model:             3
                      Covariance Type:      nonrobust

                                        coef      std err     t       P> |t|   [0.025     0.975]
intercept                              1.2876     0.359      3.588   0.000    0.580      1.995
poly(TV, degree=2, raw=True)[0]        0.0784     0.005     15.736   0.000    0.069      0.088
poly(TV, degree=2, raw=True)[1]       -0.0001     1.68e-05  -6.775   0.000   -0.000     -8.05e-05
Radio                                  0.1930     0.007     26.465   0.000    0.179      0.207

                Omnibus:           19.524    Durbin-Watson:       2.136
                Prob(Omnibus):      0.000    Jarque-Bera (JB):    44.712
                Skew:              -0.413    Prob(JB):            1.95e-10
                Kurtosis:           5.164    Cond. No.            1.29e+05
```

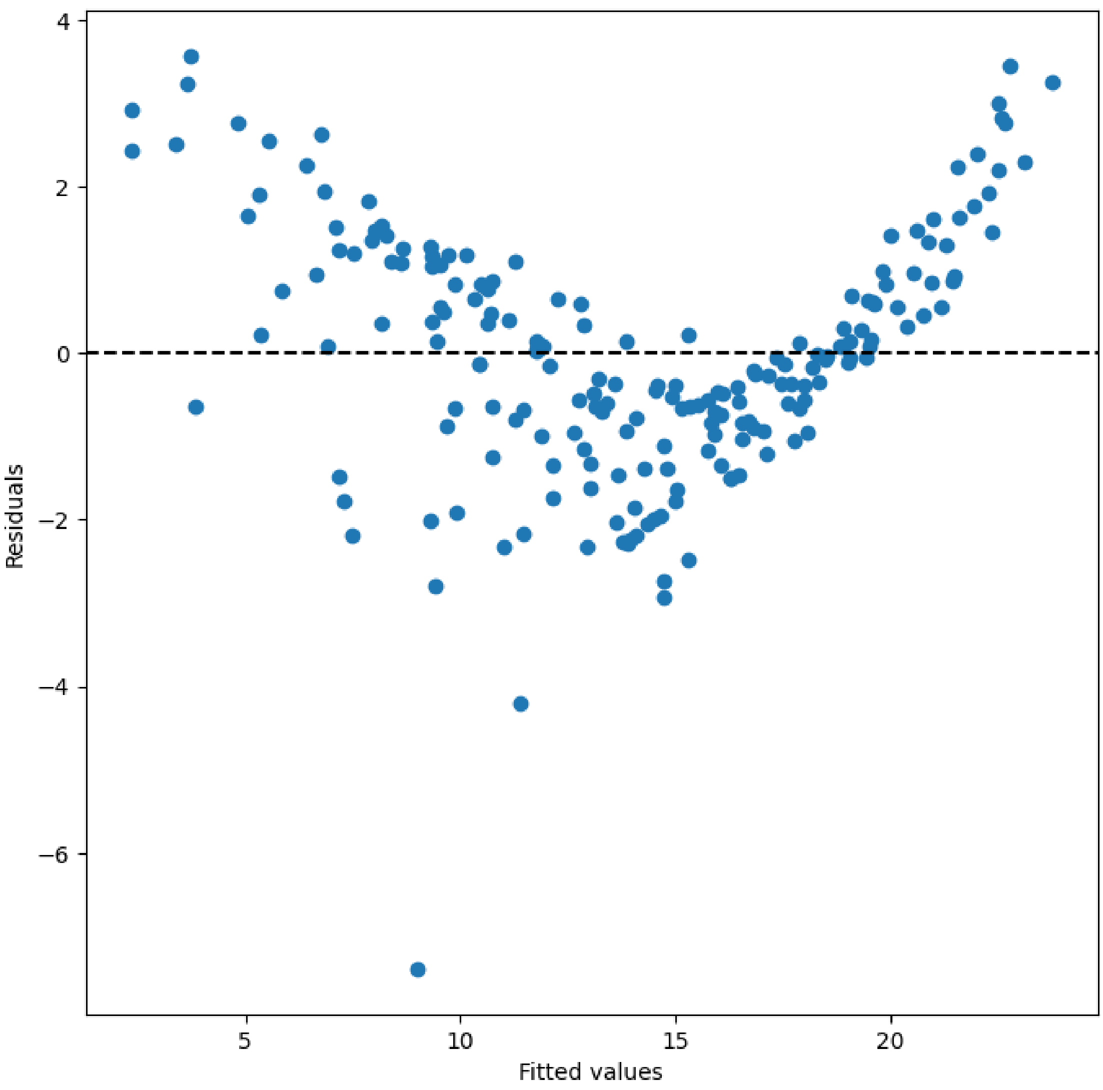
Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.29e+05. This might indicate that there are
strong multicollinearity or other numerical problems.

[30]:

```python
_, ax = subplots(figsize=(8, 8))
ax.scatter(results.fittedvalues, results.resid)
ax.set_xlabel("Fitted values")
ax.set_ylabel("Residuals")
ax.axhline(0, c="k", ls="--")
```

[30]:

```
<matplotlib.lines.Line2D at 0x7bd04098eba0>
```



While the fit has improved as seen from the R2 increasing by 2 percentage points, there is still some non-linearity visible in the residuals plot against fitted values.

References:

https://www.kellogg.northwestern.edu/faculty/weber/emp/_session_3/nonlinearities.htm

https://online.stat.psu.edu/stat462/node/120/

1.10.5 Is there synergy among the advertising media?

Synergy implies an interaction effect. That's what we test out now.

[31]:

```
X = MS([poly("TV", raw=True, degree=2), "Radio", ("TV", "Radio")]).
 ↪fit_transform(
    Advertising
)
model = sm.OLS(y, X)
results = model.fit()
summarize(results)
```

[31]:

```
                                   coef   std err        t  P>|t|
intercept                        5.1371  0.193000   26.663    0.0
poly(TV, degree=2, raw=True)[0]  0.0509  0.002000   22.810    0.0
poly(TV, degree=2, raw=True)[1] -0.0001  0.000007  -15.920    0.0
Radio                            0.0352  0.006000    5.959    0.0
TV:Radio                         0.0011  0.000035   31.061    0.0
```

[32]:

```
results.summary()
```

[32]:

Dep. Variable:	Sales	**R-squared:**	0.986
Model:	OLS	**Adj. R-squared:**	0.986
Method:	Least Squares	**F-statistic:**	3432.
Date:	Fri, 21 Feb 2025	**Prob (F-statistic):**	1.79e-179
Time:	19:19:34	**Log-Likelihood:**	-186.86
No. Observations:	200	**AIC:**	383.7
Df Residuals:	195	**BIC:**	400.2
Df Model:	4		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
intercept	5.1371	0.193	26.663	0.000	4.757	5.517
poly(TV, degree=2, raw=True)[0]	0.0509	0.002	22.810	0.000	0.047	0.055
poly(TV, degree=2, raw=True)[1]	-0.0001	6.89e-06	-15.920	0.000	-0.000	-9.61e-05
Radio	0.0352	0.006	5.959	0.000	0.024	0.047
TV:Radio	0.0011	3.47e-05	31.061	0.000	0.001	0.001

Omnibus:	169.759	**Durbin-Watson:**	2.204
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	4031.167
Skew:	-2.988	**Prob(JB):**	0.00
Kurtosis:	24.166	**Cond. No.**	1.70e+05

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.7e+05. This might indicate that there are
strong multicollinearity or other numerical problems.

- Finally, when we add an interaction term TV * Radio to the model, we can see that the residual fit exhibits no pattern. And the R2 is 98.6%.

1.10.6 Compute VIFs and List Comprehension

```
[33]: vals = [VIF(X, i) for i in range(1, X.shape[1])]
      print(vals)
```

```
[18.787830609925035, 15.885268501061871, 3.9253174186837008, 6.940088238444382]
```

```
[34]: vif = pd.DataFrame({"vif": vals}, index=X.columns[1:])
      print(vif)
      ("VIF Range:", np.min(vif), np.max(vif))
```

```
                                      vif
poly(TV, degree=2, raw=True)[0]  18.787831
poly(TV, degree=2, raw=True)[1]  15.885269
Radio                             3.925317
TV:Radio                          6.940088
```

```
[34]: ('VIF Range:', 3.9253174186837008, 18.787830609925035)
```

- The VIF ranges are high. These can be reduced by transforming variables to mean 0.
- https://stats.stackexchange.com/questions/23538/quadratic-term-and-variance-inflation-fac

```
[35]: Advertising["TV"] = Advertising["TV"] - Advertising["TV"].mean()
      Advertising["Radio"] = Advertising["Radio"] - Advertising["Radio"].mean()
```

```
[36]: X = MS([poly("TV", raw=True, degree=2), "Radio", ("TV", "Radio")]).
        ↪fit_transform(
          Advertising
      )
      model = sm.OLS(y, X)
      results = model.fit()
      summarize(results)
```

```
[36]:                                     coef   std err        t  P>|t|
      intercept                        14.7525  0.067000  219.634    0.0
      poly(TV, degree=2, raw=True)[0]   0.0437  0.001000   84.111    0.0
      poly(TV, degree=2, raw=True)[1]  -0.0001  0.000007  -15.920    0.0
      Radio                             0.1935  0.003000   64.526    0.0
      TV:Radio                          0.0011  0.000035   31.061    0.0
```

```
[37]: results.summary()
```

```
[37]:
```

Dep. Variable:	Sales	**R-squared:**	0.986
Model:	OLS	**Adj. R-squared:**	0.986
Method:	Least Squares	**F-statistic:**	3432.
Date:	Fri, 21 Feb 2025	**Prob (F-statistic):**	1.79e-179
Time:	19:19:34	**Log-Likelihood:**	-186.86
No. Observations:	200	**AIC:**	383.7
Df Residuals:	195	**BIC:**	400.2
Df Model:	4		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P> |t|**	**[0.025**	**0.975]**
intercept	14.7525	0.067	219.634	0.000	14.620	14.885
poly(TV, degree=2, raw=True)[0]	0.0437	0.001	84.111	0.000	0.043	0.045
poly(TV, degree=2, raw=True)[1]	-0.0001	6.89e-06	-15.920	0.000	-0.000	-9.61e-05
Radio	0.1935	0.003	64.526	0.000	0.188	0.199
TV:Radio	0.0011	3.47e-05	31.061	0.000	0.001	0.001

Omnibus:	169.759	**Durbin-Watson:**	2.204
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	4031.167
Skew:	-2.988	**Prob(JB):**	0.00
Kurtosis:	24.166	**Cond. No.**	1.49e+04

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.49e+04. This might indicate that there are
strong multicollinearity or other numerical problems.

[38]:

```
vals = [VIF(X, i) for i in range(1, X.shape[1])]
print(vals)
```

```
[1.0172717815970211, 1.017084612216564, 1.013513326764562, 1.0075840215785734]
```

[39]:

```
vif = pd.DataFrame({"vif": vals}, index=X.columns[1:])
print(vif)
("VIF Range:", np.min(vif), np.max(vif))
```

```
                                     vif
poly(TV, degree=2, raw=True)[0]  1.017272
poly(TV, degree=2, raw=True)[1]  1.017085
Radio                            1.013513
TV:Radio                         1.007584
```

[39]:

```
('VIF Range:', 1.0075840215785734, 1.0172717815970211)
```

[40]:

```
allDone()
```

```
<IPython.lib.display.Audio object>
```