

Multilinear Regression: CarSeats dataset

# Table of contents

|   |    |
|---|----|
| Import notebook funcs . . . . .   | 1  |
| Import ISLP objects . . . . .   | 2  |
| Import User Funactions . . . . .  | 2  |
| Load dataset . . . . .  | 2  |
| Display dataset stats . . . . .   | 2  |
| Set categorical types . . . . .   | 3  |
| Standardize variables . . . . .   | 3  |
| (a) Fit a multiple regression model to predict Sales using Price, Urban, and US. . . . .  | 3  |
| (b) Provide an interpretation of each coefficient in the model. Be careful—some of the<br>variables in the model are qualitative! . . . . .   | 4  |
| (c) Write out the model in equation form, being careful to handle the qualitative vari-<br>ables properly. . . . .  | 4  |
| (d) For which of the predictors can you reject the null hypothesis $H_0 : \beta_j = 0$ ? . . . . .  | 4  |
| (e) On the basis of your response to the previous question, fit a smaller model that only<br>uses the predictors for which there is evidence of association with the outcome. . . . . | 5  |
| (f) How well do the models in (a) and (e) fit the data? . . . . .   | 5  |
| (g) Using the model from (e), obtain 95 % confidence intervals for the coefficient(s). . . . .  | 6  |
| (h) Is there evidence of outliers or high leverage observations in the model from (e)? . . . . .  | 6  |
| For a more conservative cutoff values for hat_diag, we have the following influence point(s):   | 14 |
| Using DFFITS cutoff, we have the following influential points . . . . .   | 14 |
| Using Cooks Distance, we have the following influential points . . . . .  | 14 |
| Using Cooks Distance p-values, we have the following influential points . . . . .   | 14 |
| Using DFBeta for intercept, we have the following influential points . . . . .  | 14 |
| Using DFBeta for US, we have the following influential points . . . . .   | 15 |
| Using DFBeta for Price, we have the following influential points . . . . .  | 15 |

## Import notebook funcs

```
from notebookfuncs import *
```

## Import ISLP objects

```
from ISLP import load_data
```

## Import User Functions

```
from userfuncs import *
```

## Load dataset

```
Carseats = load_data("Carseats")  
Carseats.head()
```

|   | Sales | CompPrice | Income | Advertising | Population | Price | ShelveLoc | Age | Education | Urban | US  |
|---|-------|-----------|--------|-------------|------------|-------|-----------|-----|-----------|-------|-----|
| 0 | 9.50  | 138       | 73     | 11          | 276        | 120   | Bad       | 42  | 17        | Yes   | Yes |
| 1 | 11.22 | 111       | 48     | 16          | 260        | 83    | Good      | 65  | 10        | Yes   | Yes |
| 2 | 10.06 | 113       | 35     | 10          | 269        | 80    | Medium    | 59  | 12        | Yes   | Yes |
| 3 | 7.40  | 117       | 100    | 4           | 466        | 97    | Medium    | 55  | 14        | Yes   | Yes |
| 4 | 4.15  | 141       | 64     | 3           | 340        | 128   | Bad       | 38  | 13        | Yes   | No  |

```
Carseats.shape
```

```
(400, 11)
```

```
Carseats = Carseats.dropna()  
Carseats.shape
```

```
(400, 11)
```

## Display dataset stats

```
Carseats.describe()
```

|       | Sales      | CompPrice  | Income     | Advertising | Population | Price      | Age        | Education  |
|-------|------------|------------|------------|-------------|------------|------------|------------|------------|
| count | 400.000000 | 400.000000 | 400.000000 | 400.000000  | 400.000000 | 400.000000 | 400.000000 | 400.000000 |
| mean  | 7.496325   | 124.975000 | 68.657500  | 6.635000    | 264.840000 | 115.795000 | 53.322500  | 13.900000  |
| std   | 2.824115   | 15.334512  | 27.986037  | 6.650364    | 147.376436 | 23.676664  | 16.200297  | 2.620528   |
| min   | 0.000000   | 77.000000  | 21.000000  | 0.000000    | 10.000000  | 24.000000  | 25.000000  | 10.000000  |
| 25%   | 5.390000   | 115.000000 | 42.750000  | 0.000000    | 139.000000 | 100.000000 | 39.750000  | 12.000000  |

|     | Sales     | CompPrice  | Income     | Advertising | Population | Price      | Age       | Education |
|-----|-----------|------------|------------|-------------|------------|------------|-----------|-----------|
| 50% | 7.490000  | 125.000000 | 69.000000  | 5.000000    | 272.000000 | 117.000000 | 54.500000 | 14.000000 |
| 75% | 9.320000  | 135.000000 | 91.000000  | 12.000000   | 398.500000 | 131.000000 | 66.000000 | 16.000000 |
| max | 16.270000 | 175.000000 | 120.000000 | 29.000000   | 509.000000 | 191.000000 | 80.000000 | 18.000000 |

## Set categorical types

```
Carseats["US"] = Carseats["US"].astype("category")
Carseats["Urban"] = Carseats["Urban"].astype("category")
```

## Standardize variables

```
Carseats["Sales"] = standardize(Carseats["Sales"])
Carseats["Price"] = standardize(Carseats["Price"])
```

(a) Fit a multiple regression model to predict Sales using Price, Urban, and US.

```
formula = "Price + Urban + US"
perform_analysis("Sales", formula, Carseats)
```

```

=====
                        OLS Regression Results
=====
Dep. Variable:          Sales      R-squared:                0.239
Model:                  OLS       Adj. R-squared:            0.234
Method:                 Least Squares   F-statistic:           41.52
Date:                  Tue, 25 Feb 2025   Prob (F-statistic):    2.39e-23
Time:                  14:40:07         Log-Likelihood:        -512.88
No. Observations:      400             AIC:                  1034.
Df Residuals:          396             BIC:                  1050.
Df Model:               3
Covariance Type:       nonrobust
=====

```

|              | coef    | std err | t       | P> t  | [0.025 | 0.975] |
|--------------|---------|---------|---------|-------|--------|--------|
| Intercept    | -0.2691 | 0.098   | -2.734  | 0.007 | -0.463 | -0.076 |
| Urban[T.Yes] | -0.0078 | 0.096   | -0.081  | 0.936 | -0.197 | 0.182  |
| US[T.Yes]    | 0.4256  | 0.092   | 4.635   | 0.000 | 0.245  | 0.606  |
| Price        | -0.4566 | 0.044   | -10.389 | 0.000 | -0.543 | -0.370 |

```

=====
Omnibus:                0.676   Durbin-Watson:                1.912

```

|                |       |                   |       |
|----------------|-------|-------------------|-------|
| Prob(Omnibus): | 0.713 | Jarque-Bera (JB): | 0.758 |
| Skew:          | 0.093 | Prob(JB):         | 0.684 |
| Kurtosis:      | 2.897 | Cond. No.         | 4.27  |

=====

Notes:

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

|          | df    | sum_sq     | mean_sq   | F          | PR(>F)       |
|----------|-------|------------|-----------|------------|--------------|
| Urban    | 1.0   | 0.095104   | 0.095104  | 0.123767   | 7.251713e-01 |
| US       | 1.0   | 12.675983  | 12.675983 | 16.496407  | 5.877444e-05 |
| Price    | 1.0   | 82.939070  | 82.939070 | 107.936143 | 1.609917e-22 |
| Residual | 396.0 | 304.289843 | 0.768409  | NaN        | NaN          |

<statsmodels.regression.linear\_model.RegressionResultsWrapper at 0x7fe79017d160>

**(b) Provide an interpretation of each coefficient in the model. Be careful—some of the variables in the model are qualitative!**

- The coefficient of -0.0078 for Urban (True) indicates that -0.0078 of the SD of Sales can be explained by the level urban store as compared to a rural store. However, the p-value of 0.936 indicates that this difference is not significant and can be discounted or discarded.
- The coefficient of 0.4256 for US (True) indicates that 0.4256 of the typical deviation of Sales are explained by a US store as compared to a non-US store.
- The coefficient of -0.4566 for Price indicates that -0.4566 of the typical deviation of Sales is explained by one SD of change in the Price variable.
- We can also conclude that Price has the highest effect on Sales, the response variable, since the absolute value of its coefficient 0.4566 is the highest amongst all the coefficients.
- <https://blogs.sas.com/content/iml/2023/07/17/standardize-reg-coeff-class.html>
- <https://www.statlect.com/fundamentals-of-statistics/linear-regression-with-standardized-variables>

**(c) Write out the model in equation form, being careful to handle the qualitative variables properly.**

- The equation can be written out as follows:
- Sales = -0.0078 \* Urban + 0.4256 \* US -0.4566 \* Price (Standardized) - 0.2691

**(d) For which of the predictors can you reject the null hypothesis  $H_0 : \beta_j = 0$ ?**

- The p-value for the Urban predictor is 0.936 which is much higher than our chosen level of significance 0.01. So we cannot reject the null Hypothesis in this case that its coefficient is zero.
- The p-values for US, Price and Intercept are zero. Hence, we reject the null hypothesis for them that their coefficients are zero.

(e) On the basis of your response to the previous question, fit a smaller model that only uses the predictors for which there is evidence of association with the outcome.

```
formula = "Price + US"
results = perform_analysis("Sales", formula, Carseats)
```

```

=====
                        OLS Regression Results
=====
Dep. Variable:          Sales    R-squared:                0.239
Model:                  OLS      Adj. R-squared:            0.235
Method:                 Least Squares    F-statistic:          62.43
Date:                  Tue, 25 Feb 2025    Prob (F-statistic):    2.66e-24
Time:                  14:40:07    Log-Likelihood:        -512.88
No. Observations:      400      AIC:                  1032.
Df Residuals:          397      BIC:                  1044.
Df Model:               2
Covariance Type:       nonrobust
=====

```

|           | coef    | std err | t       | P> t  | [0.025 | 0.975] |
|-----------|---------|---------|---------|-------|--------|--------|
| Intercept | -0.2743 | 0.074   | -3.730  | 0.000 | -0.419 | -0.130 |
| US[T.Yes] | 0.4253  | 0.092   | 4.641   | 0.000 | 0.245  | 0.605  |
| Price     | -0.4567 | 0.044   | -10.416 | 0.000 | -0.543 | -0.371 |

```

=====
Omnibus:                0.666    Durbin-Watson:            1.912
Prob(Omnibus):          0.717    Jarque-Bera (JB):        0.749
Skew:                   0.092    Prob(JB):                0.688
Kurtosis:               2.895    Cond. No.:               3.12
=====

```

Notes:

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

|          | df    | sum_sq     | mean_sq   | F          | PR(>F)       |
|----------|-------|------------|-----------|------------|--------------|
| US       | 1.0   | 12.544810  | 12.544810 | 16.366658  | 6.273751e-05 |
| Price    | 1.0   | 83.160345  | 83.160345 | 108.495617 | 1.272157e-22 |
| Residual | 397.0 | 304.294845 | 0.766486  | NaN        | NaN          |

<statsmodels.regression.linear\_model.RegressionResultsWrapper at 0x7fe74d1bf8c0>

(f) How well do the models in (a) and (e) fit the data?

- Model(a) has an explanatory value R<sup>2</sup> adjusted value of 0.234
- Model(e) has an explanatory value R<sup>2</sup> adjusted value of 0.235

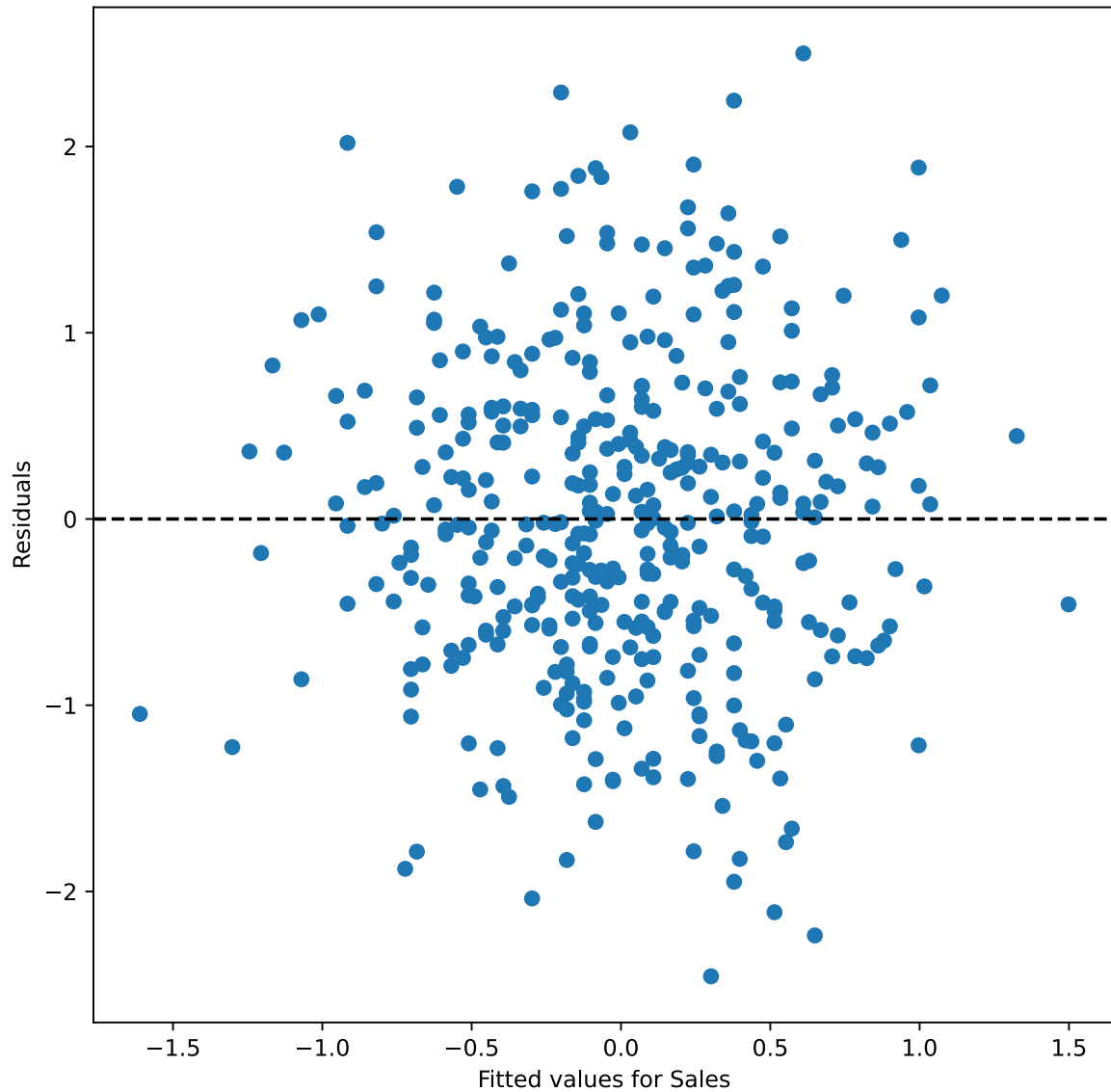
**(g) Using the model from (e), obtain 95 % confidence intervals for the coefficient(s).**

From the summary analysis, it can be seen that the 95% confidence limits for the three terms are as follows: + Intercept (-0.419, -0.130) + US[T.Yes] (0.245, 0.605) + Price (-0.543, -0.371) + None of them include zero in their range unlike that for Urban[T.Yes] in Model(a) which is another indicator that the coefficient is not significant.

**(h) Is there evidence of outliers or high leverage observations in the model from (e)?**

We can check for presence of outliers by plotting the residuals plot and seeing if there are any outliers.

```
display_residuals_plot(results)
```

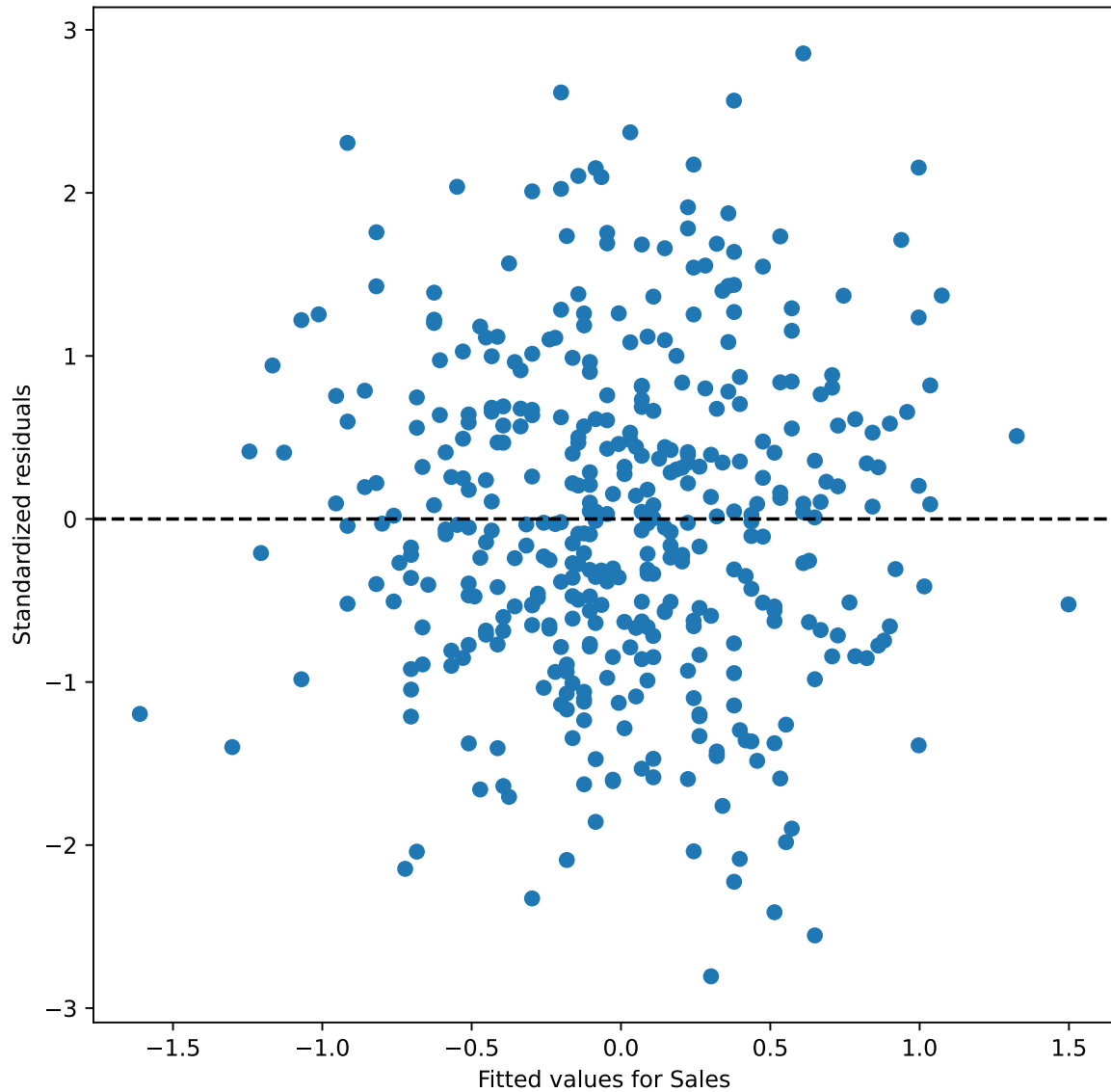


- From the plot above, there doesn't appear to be any obvious outliers.

**We can plot studentized residuals to see whether there are any visible there.**

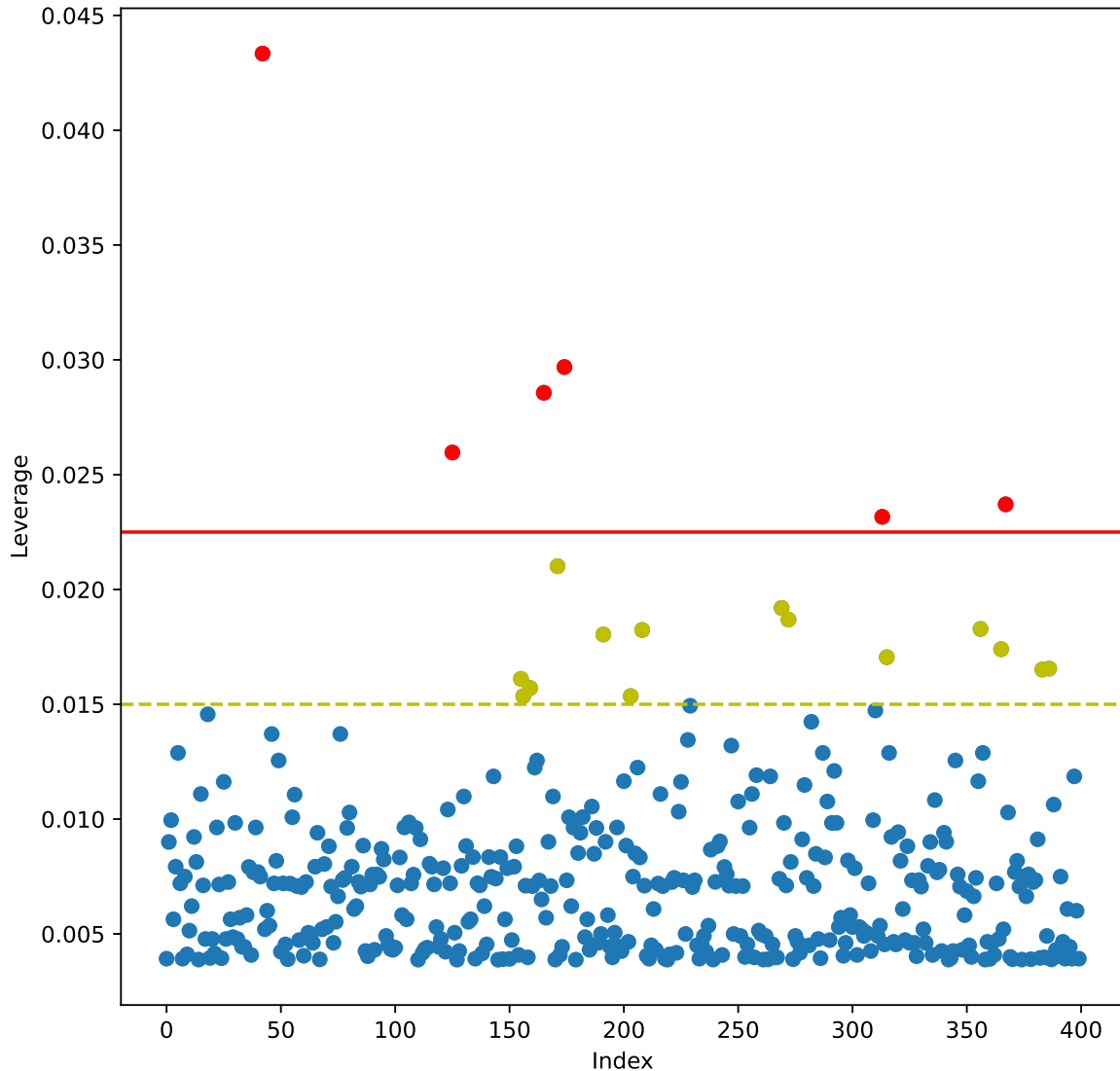
```
display_studentized_residuals(results)
```





- From the above plot, no observation lies outside the  $(-3,3)$  range. Hence, we can safely conclude that there are no evident outliers in the dataset.

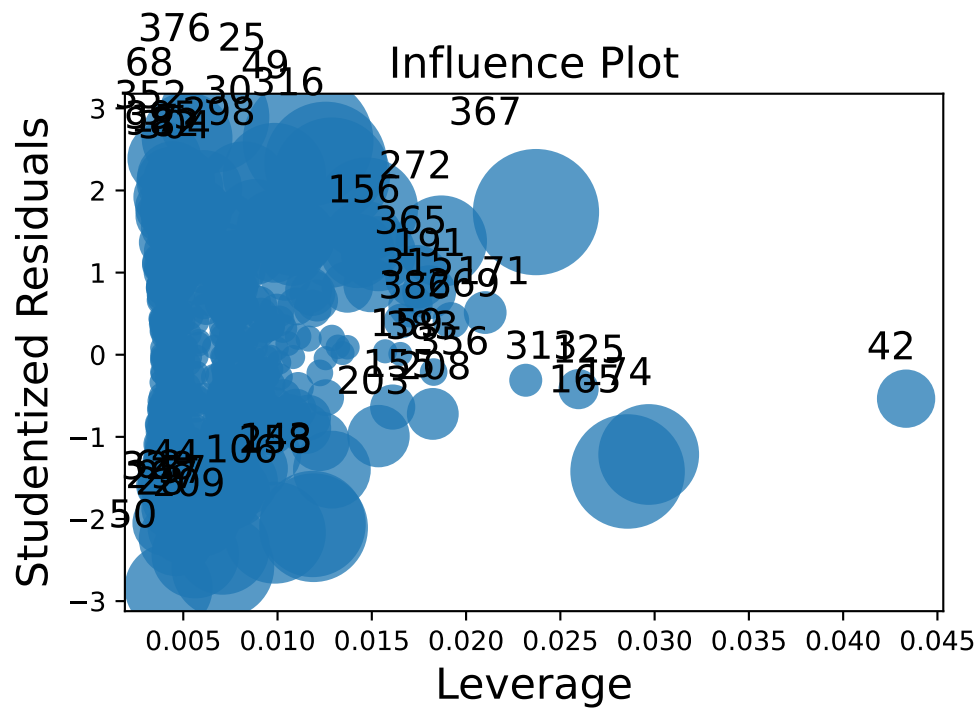
```
display_hat_leverage_cutoffs(results)
```

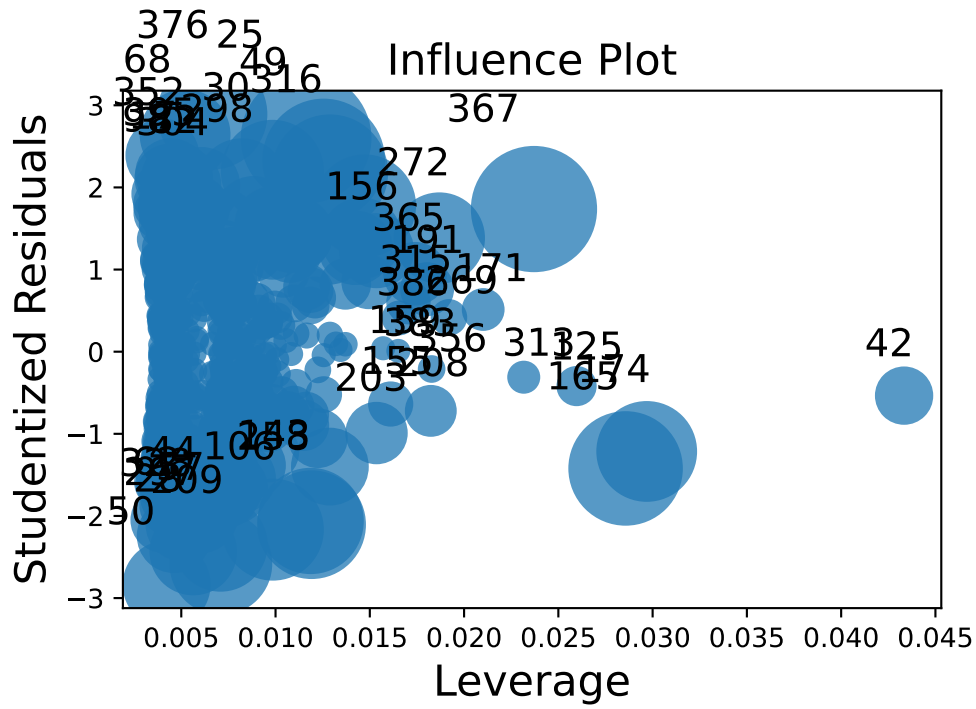


- We can see from the above graph that we have a few leverage points that exceed the cutoff of  $3 \times$  average leverage value. These are plotted in red.
- The ones in yellow exceed the less conservative estimate of  $2 \times$  average leverage value
- We could also use more conservative estimates for the cutoff of either  $4 \times$  average leverage value or  $5 \times$  average cutoff value

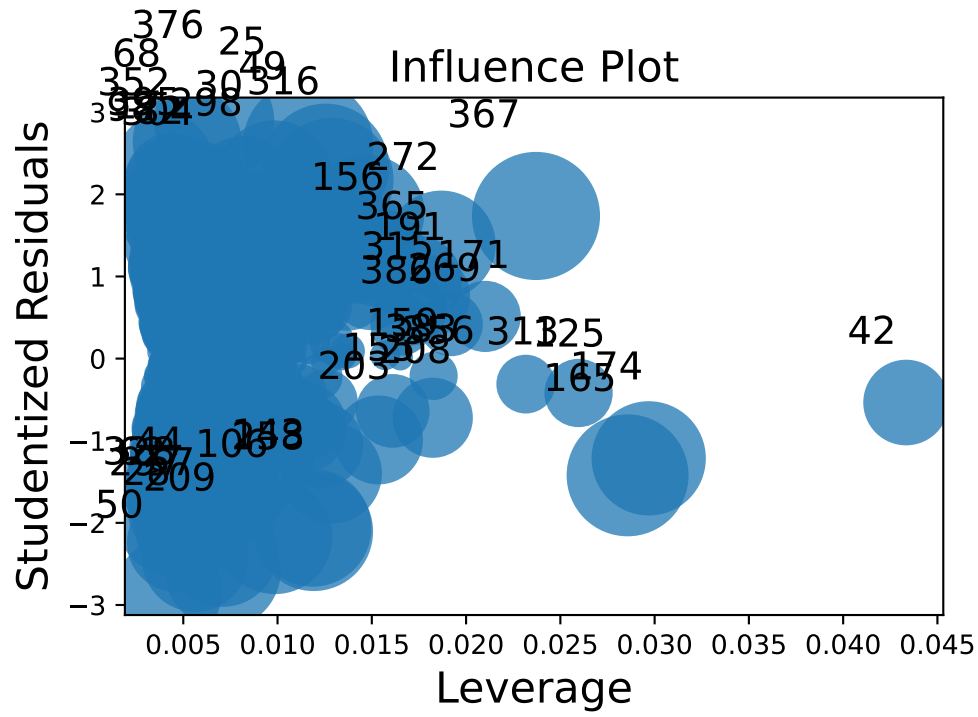
References: - <https://online.stat.psu.edu/stat501/lesson/11/11.2>

```
display_cooks_distance_plot(results)
```





```
display_DFFITS_plot(results)
```





```
{'n': 400.0, 'p': 3, 'average_hat': 0.0075000000000000015, 'hat_leverage_cutoff': 0.015000000000000003, 'dfbetas_cutoff': 0.15, 'dffits_cutoff': 0.17320508075688773, 'studentized_residuals_cutoff': 3.0, 'studentized_residuals_pvalue_cutoff': 0.01, 'cooks_d_cutoff': 1.0, 'cooks_d_pvalue_cutoff': 0.05}
```

For a more conservative cutoff values for `hat_diag`, we have the following influence point(s):

```
inf_df[inf_df["hat_diag"] > (3 * np.mean(inf_df["hat_diag"]))]
```

|  | dfb_Intercept | dfb_US[T.Yes] | dfb_Price | cooks_d | hat_diag | student_resid | dffits | student_resid_pvalue |
|--|---------------|---------------|-----------|---------|----------|---------------|--------|----------------------|
|--|---------------|---------------|-----------|---------|----------|---------------|--------|----------------------|

Using DFFITS cutoff, we have the following influential points

```
inf_df[inf_df["dffits"] > 2 * np.sqrt(len(results.params) / results.nobs)]
```

|     | dfb_Intercept | dfb_US[T.Yes] | dfb_Price | cooks_d  | hat_diag | student_resid | dffits   | student_resid_pvalue |
|-----|---------------|---------------|-----------|----------|----------|---------------|----------|----------------------|
| 25  | 0.210972      | -0.165700     | -0.176952 | 0.026109 | 0.011622 | 2.599652      | 0.281894 | 0.004840             |
| 68  | 0.004485      | 0.092526      | 0.096454  | 0.011988 | 0.005202 | 2.642364      | 0.191069 | 0.004280             |
| 376 | -0.007088     | 0.116257      | -0.152448 | 0.018282 | 0.006637 | 2.891521      | 0.236355 | 0.002023             |

Using Cooks Distance, we have the following influential points

```
inf_df[inf_df["cooks_d"] > 1.0]
```

|  | dfb_Intercept | dfb_US[T.Yes] | dfb_Price | cooks_d | hat_diag | student_resid | dffits | student_resid_pvalue |
|--|---------------|---------------|-----------|---------|----------|---------------|--------|----------------------|
|--|---------------|---------------|-----------|---------|----------|---------------|--------|----------------------|

Using Cooks Distance p-values, we have the following influential points

```
inf_df[inf_df["cooks_d_pvalue"] < 0.05]
```

|  | dfb_Intercept | dfb_US[T.Yes] | dfb_Price | cooks_d | hat_diag | student_resid | dffits | student_resid_pvalue |
|--|---------------|---------------|-----------|---------|----------|---------------|--------|----------------------|
|--|---------------|---------------|-----------|---------|----------|---------------|--------|----------------------|

Using DFBeta for intercept, we have the following influential points

```
inf_df[inf_df["dfb_Intercept"] > (3 / np.sqrt(results.nobs))]
```

|    | dfb_Intercept | dfb_US[T.Yes] | dfb_Price | cooks_d  | hat_diag | student_resid | dffits   | student_resid_pvalue |
|----|---------------|---------------|-----------|----------|----------|---------------|----------|----------------------|
| 25 | 0.210972      | -0.1657       | -0.176952 | 0.026109 | 0.011622 | 2.599652      | 0.281894 | 0.00484              |

Using DFBeta for US, we have the following influential points

```
inf_df[inf_df["dfb_US[T.Yes]"] > (3 / np.sqrt(results.nobs))]
```

|  | dfb_Intercept | dfb_US[T.Yes] | dfb_Price | cooks_d | hat_diag | student_resid | dffits | student_resid_pvalue |
|--|---------------|---------------|-----------|---------|----------|---------------|--------|----------------------|
|--|---------------|---------------|-----------|---------|----------|---------------|--------|----------------------|

Using DFBeta for Price, we have the following influential points

```
inf_df[inf_df["dfb_Price"] > (3 / np.sqrt(results.nobs))]
```

|  | dfb_Intercept | dfb_US[T.Yes] | dfb_Price | cooks_d | hat_diag | student_resid | dffits | student_resid_pvalue |
|--|---------------|---------------|-----------|---------|----------|---------------|--------|----------------------|
|--|---------------|---------------|-----------|---------|----------|---------------|--------|----------------------|

```
allDone()
```

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