Brian Tenneson

ADS 534

08/05/2024

Statement regarding academic integrity: Large Language Models (LLMs) were utilized alongside human oversight to perform data analysis and generate insights across various contexts and datasets.

1

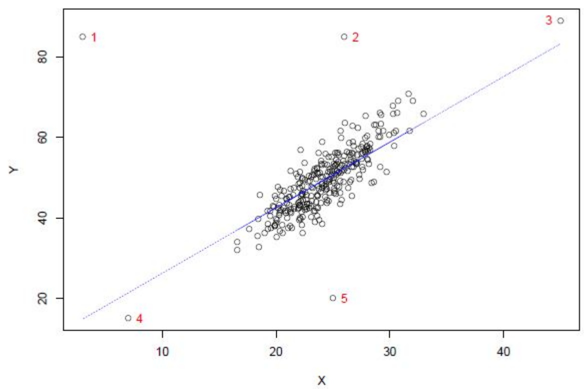


Figure 1: scatter plot of Y on X

Definitions:

Outlier: A point with a large residual, meaning it is far from the fitted regression line.

High Leverage Point: A point with an extreme X (far from $\overline{X}$).

Influential Point: A point that significantly changes the parameters of the regression model when it is removed.

Point 1

Type: Outlier and Influential Point

Justification: This point is far from the regression line and the majority of other points, indicating it is an outlier. Due to its distance from the regression line and its effect on the slope of the line, it can be considered an influential point.

Point 2

Type: Outlier and Influential Point

Justification: This point is significantly below the regression line, making it an outlier. Its deviation from the line means it has a substantial influence on the slope and intercept of the regression line, thus it is also an influential point.

Point 3

Type: High Leverage Point

Justification: This point has a high value of X far from the mean $\overline{X} = 24.2$ which gives it high leverage. However, it lies close to the regression line, so it is not an outlier or influential point.

Point 4

Type: High Leverage Point

Justification: Similar to Point 3, this point has an extreme X value, making it a high leverage point. It is close to the regression line and thus does not significantly influence the slope or intercept.

Point 5

Type: Outlier

Justification: This point is significantly below the regression line, making it an outlier. However, it does not appear to have a large influence on the regression line, so it is not considered an influential point.

2

(a) The error terms have constant variance (Homoscedasticity)

Residuals vs. Fitted Values Plot: This plot helps to check if the residuals (errors) have constant variance across all levels of the fitted values. If the variance of residuals increases or decreases with fitted values, it indicates heteroscedasticity.

(b) The error terms are normally distributed

Normal Q-Q Plot (Quantile-Quantile Plot): This plot compares the distribution of the residuals to a normal distribution. If the points lie approximately along the diagonal line,

Histogram of Residuals: A histogram of the residuals can help to visually assess whether the residuals follow a normal distribution.

(c) There is a linear relationship between the response and predictor variables

Scatter Plot: A scatter plot of the response variable against each predictor variable can show if there is a linear relationship between them.

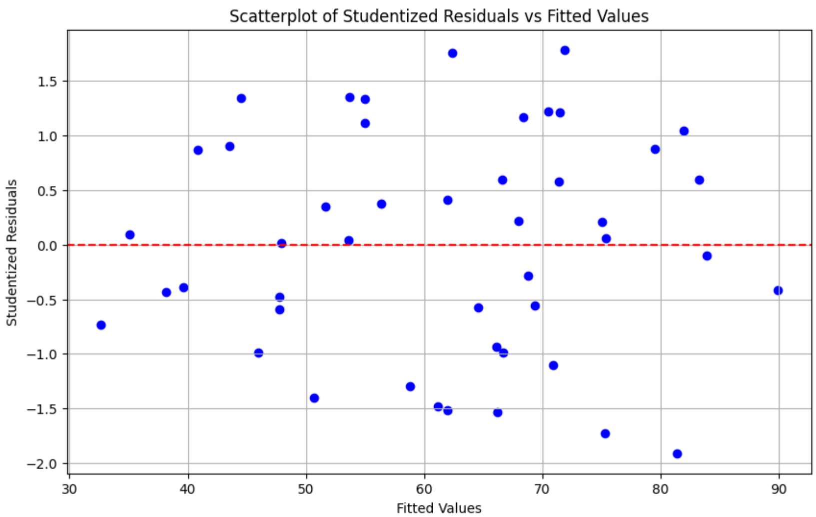
Residuals vs. Fitted Values Plot: This plot can also be used to check for non-linearity. If the plot shows a random scatter without any pattern, it suggests a linear relationship. If there is a systematic pattern (e.g., a curve), it indicates non-linearity.

3(a)

Using the following python code, we obtain a scatterplot of studentized residuals versus fitted values graph:

```
import pandas as pd

# Load the dataset
data_path = "G:\\My Drive\\Summer - 2 -2024\\ADS 534\\homework 5\\satisfaction.csv"
data = pd.read_csv(data_path)

# Display the first few rows of the dataset and the structure of the data
data.head(), data.info()
import statsmodels.api as sm

# Prepare the independent and dependent variables
X = data[['X1', 'X2', 'X3']]
X = sm.add_constant(X)  # adding a constant for the intercept
y = data['Y']

# Fit the regression model
model = sm.OLS(y, X).fit()

# Calculate studentized residuals
data['studentized_residuals'] = model.get_influence().resid_studentized_internal

# Fitted values
data['fitted_values'] = model.fittedvalues

# Scatterplot of studentized residuals against fitted values
import matplotlib.pyplot as plt

plt.figure(figsize=(10, 6))
plt.scatter(data['fitted_values'], data['studentized_residuals'], color='blue')
plt.axhline(y=0, color='r', linestyle='--') # Add a horizontal line at zero for reference
plt.xlabel('Fitted Values')
plt.ylabel('Studentized Residuals')
plt.title('Scatterplot of Studentized Residuals vs Fitted Values')
```

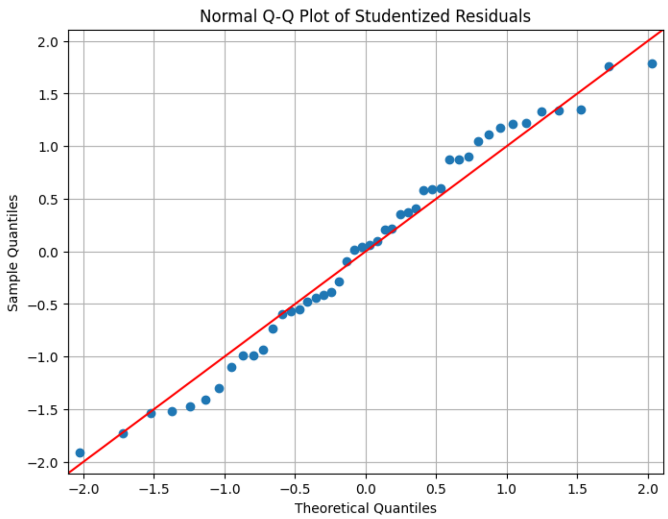
```
plt.grid(True)
plt.show()

# Output model summary for further insights
model.summary()
```



Variable	Coefficient	Standard Error	t-value	p-value	CI Lower	CI Upper
const	158.49	18.13	8.74	5.26e-11	121.91	195.07
X1	-1.14	0.215	-5.31	3.81e-06	-1.58	-0.71
X2	-0.44	0.49	-0.89	0.37	-1.4	0.551
X3	-13.47	7.1	-1.9	0.065	-27.8	0.858

No, there do not appear to be any outliers in the studentized vs fitted plot.

3(b)



The normal Q-Q plot generally indicates that the residuals from the model align well with a normal distribution, which affirms the reliability of the statistical conclusions derived from the analysis. Nonetheless, the minor irregularities observed at the extremes of the plot warrant a thoughtful review and possibly additional examination of any outliers or high leverage points that could affect the model's accuracy.

3(c)

Here, I obtain a list of all 46 points along with whether they are high leverage points. The rule of thumb for determining this is given by

$$\frac{2\,(k+1)}{n}$$

where k is the number of predictors including the constant and n is the number of observations. In our case, this value is 0.2174.

leverage high_leverage

0 0.078197 False

1 0.067068 False

2 0.037171 False

3 0.153611 False

4 0.096737 False

5 0.128577 False

6 0.034485 False

7 0.075244 False

8 0.184259 False

9 0.057979 False

10 0.087592 False

11 0.030875 False

12 0.090321 False

13 0.033238 False

14 0.142890 False

15 0.047133 False

16 0.119542 False

17 0.062417 False

18 0.033508 False

19 0.128929 False

20 0.077696 False

21 0.136901 False

22 0.032881 False

23 0.135751 False

24 0.043367 False

25 0.102946 False

26 0.086823 False

27 0.186019 False

28 0.059442 False

29 0.089981 False

30 0.117105 False

31 0.109631 False

32 0.045045 False

33 0.037171 False

34 0.103040 False

35 0.027232 False

36 0.121221 False

37 0.070589 False

38 0.180960 False

39 0.086896 False

40 0.037976 False

41 0.153859 False

42 0.061019 False

43 0.050910 False

44 0.072616 False

45 0.083152 False.

Case 11:

Cook's Distance: 0.077

DFFITS: 0.569

Case 17:

Cook's Distance: 0.105

DFFITS: 0.666

Case 27:

Cook's Distance: 0.087

DFFITS: -0.609

All the Cook's distances are below 0.1 but are on the higher end within the context of this dataset. For DFFITS, a common rule of thumb is that values larger than $2\sqrt{\frac{p+1}{n}}$ (where p is the number of predictors and n is the number of observations) might indicate influential observations. For this dataset with 3 predictors and 46 observations, the threshold would be approximately $\approx$0.588. The DFFITS values for cases 17 and 27 are around this threshold, indicating potential influence.

4

(a) Age, Income, and Price, came up as the best three predictor variables. R-squared: 0.303

Adjusted R-squared: 0.259

F-statistic: 6.818 (p-value: 0.000657)

Coefficients:

Intercept (const): 64.2482

Age: 4.1559 (p-value: 0.065)

Income: 0.0193 (p-value: 0.007)

Price: -3.3992 (p-value: 0.001)

code:

```
import itertools
import statsmodels.api as sm
import pandas as pd

# Load the provided CSV file
file_path ="G:\\My Drive\\Summer - 2 -2024\\ADS 534\\homework 5\\cigarette.csv"
data = pd.read_csv(file_path)

# Strip any leading/trailing spaces from the column names in the DataFrame
data.columns = data.columns.str.strip()

# Define the list of feature names (excluding the target variable 'Sales')
feature_names = ['Age', 'HS', 'Income', 'Black', 'Female', 'Price']
target = 'Sales'

# Ensure feature names are correctly stripped
feature_names = [name.strip() for name in feature_names]

# Verify feature names against DataFrame columns
missing_features = [feature for feature in feature_names if feature not in data.columns]
if missing_features:
    raise KeyError(f"The following features are not in the DataFrame columns: {missing_features}")
```

```
# Function to calculate adjusted R-squared for a given combination of features
def calculate_adj_r2(features):
    X = data[list(features)]
    X = sm.add_constant(X)
    y = data[target]
    model = sm.OLS(y, X).fit()
    return model.rsquared_adj

# Generate all possible combinations of features
combinations_of_features = []
for r in range(1, len(feature_names) + 1):
    combinations_of_features.extend(itertools.combinations(feature_names, r))

# Calculate adjusted R-squared for each combination of features
results = []
for combination in combinations_of_features:
    adj_r2 = calculate_adj_r2(combination)
    results.append((combination, adj_r2))

# Find the combination with the highest adjusted R-squared
best_combination, best_adj_r2 = max(results, key=lambda x: x[1])

# Fit the final model with the best combination of features
X_final = data[list(best_combination)]
X_final = sm.add_constant(X_final)
y = data[target]

final_model = sm.OLS(y, X_final).fit()
final_model_summary = final_model.summary()

best_combination, final_model_summary
```

(b)

I used the following code to find that Age, Income, and Price:

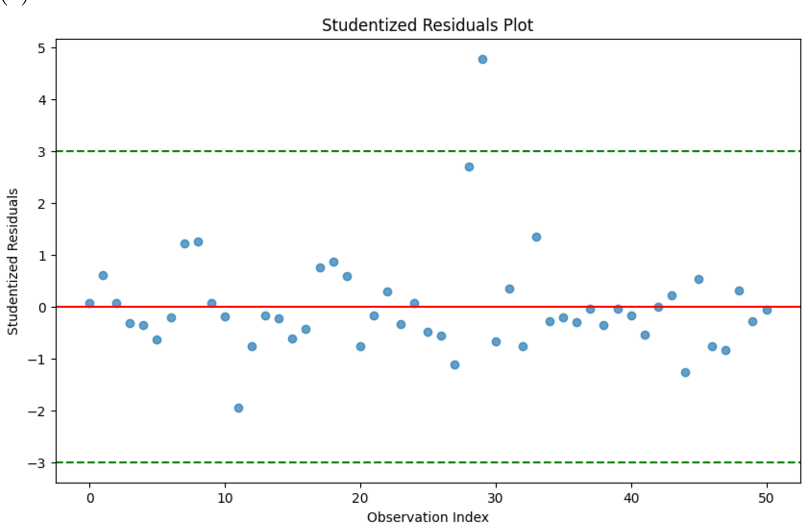
```
import itertools
import statsmodels.api as sm
import pandas as pd
```

```
# Load the provided CSV file
file_path =  "G:\\My Drive\\Summer - 2 -2024\\ADS 534\\homework 5\\cigarette.csv"
data = pd.read_csv(file_path)

# Strip any leading/trailing spaces from the column names in the DataFrame
data.columns = data.columns.str.strip()

# Define the list of feature names (excluding the target variable 'Sales')
feature_names = ['Age', 'HS', 'Income', 'Black', 'Female', 'Price']
target = 'Sales'

# Ensure feature names are correctly stripped
feature_names = [name.strip() for name in feature_names]

# Verify feature names against DataFrame columns
missing_features = [feature for feature in feature_names if feature not in data.columns]
if missing_features:
    raise KeyError(f"The following features are not in the DataFrame columns: {missing_features}")

# Function to calculate AIC for a given combination of features
def calculate_aic(features):
    X = data[list(features)]
    X = sm.add_constant(X)
    y = data[target]
    model = sm.OLS(y, X).fit()
    return model.aic

# Generate all possible combinations of features
combinations_of_features = []
for r in range(1, len(feature_names) + 1):
    combinations_of_features.extend(itertools.combinations(feature_names, r))

# Calculate AIC for each combination of features
results = []
for combination in combinations_of_features:
    aic = calculate_aic(combination)
    results.append((combination, aic))
```

```
# Find the combination with the lowest AIC
best_combination, best_aic = min(results, key=lambda x: x[1])

# Fit the final model with the best combination of features
X_final = data[list(best_combination)]
X_final = sm.add_constant(X_final)
y = data[target]

final_model = sm.OLS(y, X_final).fit()
final_model_summary = final_model.summary()

best_combination, final_model_summary
```

I found that the AIC for this model is 487.

(c)



As we can see, there is a point near the outlier threshold of 3 and a point above the threshold of 3 indicating an outlier in Sales. Here's how we can find out which state the outlier is:

```
# Identify the indices of the outliers
outlier_indices = [i for i, residual in enumerate(studentized_residuals) if abs(residual) > 3]
```

```
# Print the details of the outliers
outliers = data.iloc[outlier_indices]
outliers
```

index	state	Age	HS	Income	Black	Female	Price	Sales
29	NH	28.0	57.6	3737	0.3	51.1	34.1	265.7

(d) After removing NH from the dataset, the best linear regression model that predicts the per capita sale of cigarettes in a given state, the variables that come into play with most prevalence are Age, Income, Black, and Price. The adjusted R^2_{adj} is 0.408 which is notably higher than when the state of NH is included. In the revised model, $AIC = 441.7$.

(e) The state of NV is an outlier presenting with an above three in the studentized versus observation index in the model with NH removed.

(f) Utilizing the following code, we will get the desired result:

```
import itertools
import statsmodels.api as sm
import pandas as pd
import numpy as np

# Load the provided CSV file from the new location
file_path = 'G:\\My Drive\\Summer - 2 -2024\\ADS 534\\homework 5\\cigarette3.csv'
data = pd.read_csv(file_path)

# Strip any leading/trailing spaces from the column names in the DataFrame
data.columns = data.columns.str.strip()

# Remove NV and NH from the dataset
data_filtered = data[~data['State'].isin(['NV', 'NH'])]

# Define the list of feature names (excluding the target variable 'Sales')
feature_names = ['Age', 'HS', 'Income', 'Black', 'Female', 'Price']
target = 'Sales'

# Ensure feature names are correctly stripped
feature_names = [name.strip() for name in feature_names]
```

```
# Verify feature names against DataFrame columns
missing_features = [feature for feature in feature_names if feature not in data_filtered.columns]
if missing_features:
    raise KeyError(f"The following features are not in the DataFrame columns: {missing_features}")

# Function to calculate adjusted R-squared for a given combination of features
def calculate_adj_r2(features):
    X = data_filtered[list(features)]
    X = sm.add_constant(X)
    y = data_filtered[target]
    model = sm.OLS(y, X).fit()
    return model.rsquared_adj

# Generate all possible combinations of features
combinations_of_features = []
for r in range(1, len(feature_names) + 1):
    combinations_of_features.extend(itertools.combinations(feature_names, r))

# Calculate adjusted R-squared for each combination of features
results = []
for combination in combinations_of_features:
    adj_r2 = calculate_adj_r2(combination)
    results.append((combination, adj_r2))

# Find the combination with the highest adjusted R-squared
best_combination, best_adj_r2 = max(results, key=lambda x: x[1])

# Fit the final model with the best combination of features
X_final = data_filtered[list(best_combination)]
X_final = sm.add_constant(X_final)
y = data_filtered[target]

final_model = sm.OLS(y, X_final).fit()

# Calculate SSE
sse = np.sum(final_model.resid**2)
```

```
# Calculate AIC manually
n = len(y)
k = len(best_combination) + 1  # Number of parameters + 1 for the intercept
aic_manual = 2*k + n * np.log(sse/n)

# Get the AIC from the model summary
aic_model = final_model.aic

best_combination, sse, aic_manual, aic_model
```

(('Age', 'HS', 'Income', 'Female', 'Price'), <–best combination

10507.094130448271, <–SSE

275.031296427133, <–AIC manual

414.08727268119094) <–AIC model

Notice that AIC manual differs quite a bit from AIC model.

(g) After we remove NH from the dataset, the best predictors are given by ('Age', 'Income', 'Black', 'Price') with corresponding coefficients (with first coefficient being the reference predictor) are:

$$(39.8, 3.44, 0.019, 0.591, -2.47)\,.$$

Coefficient of Age (3.44):

Meaning: For each additional year in the average age of the population, the per capita sale of cigarettes increases by 3.44 units, assuming all other variables in the model are held constant.

Implication: This positive coefficient suggests that older populations tend to have higher cigarette sales per capita. It implies a direct relationship between the average age and cigarette consumption.

Coefficient of Black (0.591):

Meaning: For each 1 percentage point increase in the proportion of the Black population, the per capita sale of cigarettes increases by 0.591 units, assuming all other variables in the model are held constant.

Implication: This scaled score can be translated into how many more per capita sales in cigarettes if were there 1 unit more of black population. This number is 0.591.

(h) After we remove NV and NH from the dataset as variables that resulted in studentized scores less than 3 in absolute value are 'Age', 'HS', 'Income', 'Female', and 'Price'. I obtain the following:

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  Sales   R-squared:                       0.671
Model:                            OLS   Adj. R-squared:                  0.632
Method:                 Least Squares   F-statistic:                     17.12
Date:                Sun, 04 Aug 2024   Prob (F-statistic):           3.30e-09
Time:                        11:29:39   Log-Likelihood:                -191.66
No. Observations:                  48   AIC:                             395.3
Df Residuals:                      42   BIC:                             406.5
Df Model:                           5
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept   -355.4880    154.575     -2.300      0.026    -667.433     -43.543
Age            1.9427      1.404      1.383      0.174      -0.892       4.777
Price         -3.5339      0.531     -6.650      0.000      -4.606      -2.461
Female        10.0975      3.192      3.164      0.003       3.657      16.538
Income         0.0191      0.005      4.209      0.000       0.010       0.028
HS            -0.6460      0.340     -1.899      0.064      -1.333       0.041
==============================================================================
Omnibus:                        1.737   Durbin-Watson:                   2.158
Prob(Omnibus):                  0.420   Jarque-Bera (JB):                1.577
Skew:                           0.429   Prob(JB):                        0.454
Kurtosis:                       2.767   Cond. No.                     2.88e+05
==============================================================================
```

(i) Using the following code, I didn't find any more outliers, but using backward elimination and after removing the reference predictor:

OLS Regression Results

Dep. Variable:	Sales	R-squared (uncentered):	0.985
Model:	OLS	Adj. R-squared (uncentered):	0.983
Method:	Least Squares	F-statistic:	714.6
Date:	Sun, 04 Aug 2024	Prob (F-statistic):	4.43e-40
Time:	11:39:55	Log-Likelihood:	-201.77
No. Observations:	49	AIC:	411.5
Df Residuals:	45	BIC:	419.1
Df Model:	4		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Female	3.7008	0.475	7.786	0.000	2.743	4.658
HS	-0.9940	0.317	-3.137	0.003	-1.632	-0.356
Income	0.0258	0.005	5.719	0.000	0.017	0.035
Price	-3.0379	0.556	-5.461	0.000	-4.158	-1.917

Omnibus:	0.429	Durbin-Watson:	1.907
Prob(Omnibus):	0.807	Jarque-Bera (JB):	0.046
Skew:	-0.014	Prob(JB):	0.977
Kurtosis:	3.147	Cond. No.	1.18e+03

Note the high $R^2$and R^2_{adj} which are both close to their maximum.

(j) Using the stepwise selection procedure with p-value < 0.10 as the entry criterion and p-value < 0.10 as the staying criterion, we get the following OLS report:

OLS Regression Results

Dep. Variable:	Sales	R-squared:	0.571
Model:	OLS	Adj. R-squared:	0.532
Method:	Least Squares	F-statistic:	14.64
Date:	Sun, 04 Aug 2024	Prob (F-statistic):	1.12e-07
Time:	12:34:55	Log-Likelihood:	-201.66
No. Observations:	49	AIC:	413.3
Df Residuals:	44	BIC:	422.8
Df Model:	4		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	-55.3780	122.604	-0.452	0.654	-302.469	191.713
Price	-3.0090	0.565	-5.327	0.000	-4.148	-1.871
Income	0.0254	0.005	5.502	0.000	0.016	0.035
HS	-0.9096	0.370	-2.457	0.018	-1.656	-0.163
Female	4.7046	2.273	2.069	0.044	0.123	9.286

Omnibus:	0.306	Durbin-Watson:	1.943
Prob(Omnibus):	0.858	Jarque-Bera (JB):	0.083
Skew:	0.101	Prob(JB):	0.959
Kurtosis:	3.016	Cond. No.	2.08e+05