In [218…
```python
# Import libraries
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from statsmodels.formula.api import ols
import statsmodels.api as sm
```

In [286…
```python
data = pd.read_csv(r'C:\Users\HP\Desktop\Advance Data Analyst\5. Simplify Complex Data Relationships\3. Module 3\4. Variable selection and model evaluation\Files\marketing_sales_data.csv')
data.head(10)
```

Out[286…

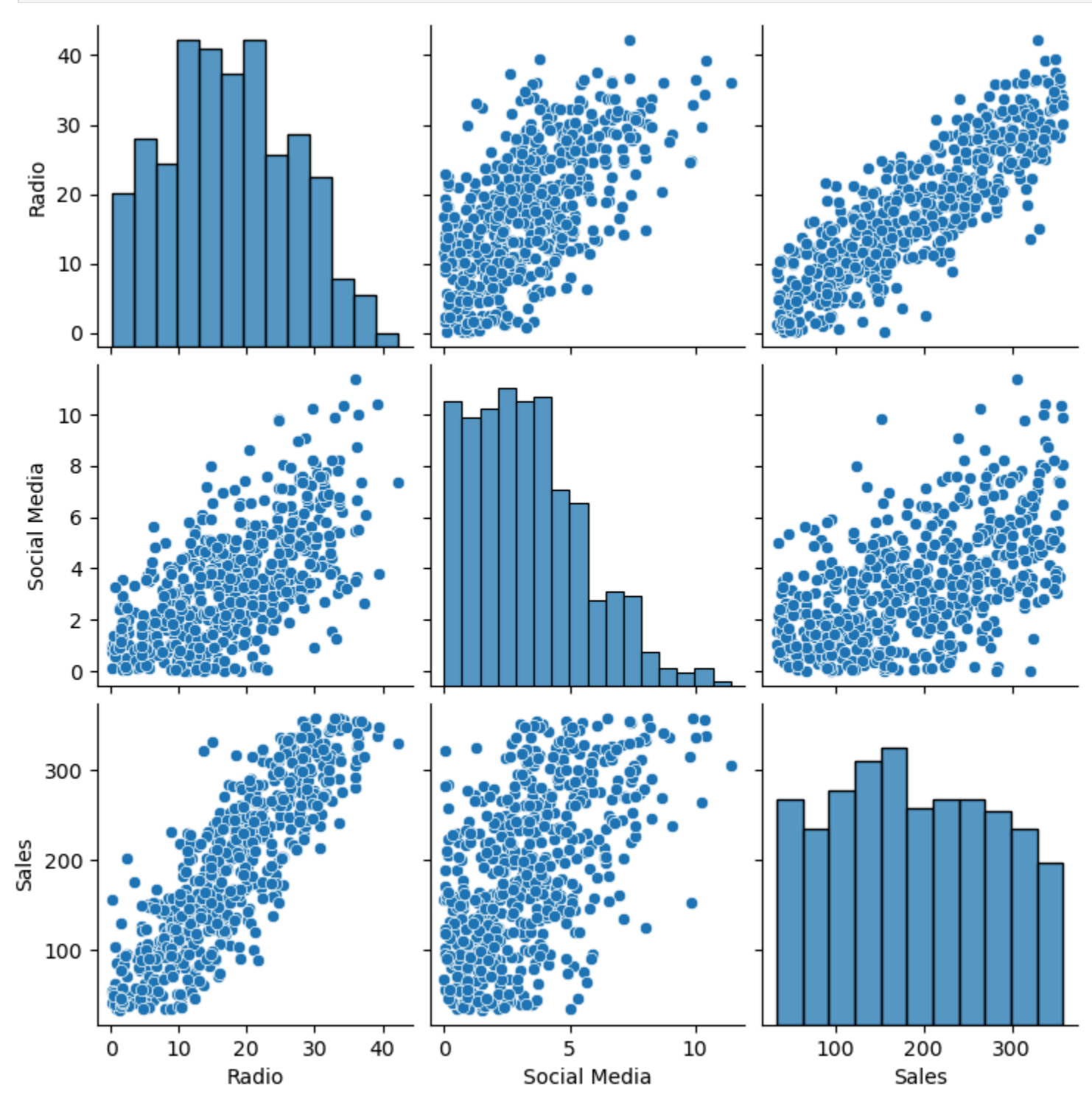
	TV	Radio	Social Media	Influencer	Sales
0	Low	3.518070	2.293790	Micro	55.261284
1	Low	7.756876	2.572287	Mega	67.574904
2	High	20.348988	1.227180	Micro	272.250108
3	Medium	20.108487	2.728374	Mega	195.102176
4	High	31.653200	7.776978	Nano	273.960377
5	Low	5.561586	3.530209	Nano	39.992087
6	Medium	13.951808	4.477142	Micro	147.150178
7	Medium	28.352592	4.695376	Mega	229.141912
8	Medium	15.322175	4.379750	Macro	222.696768
9	High	26.914287	6.074165	Mega	322.466797

In [288…
```python
# Display the first 10 rows of the data

data.head(10)
```

Out[288…

	TV	Radio	Social Media	Influencer	Sales
0	Low	3.518070	2.293790	Micro	55.261284
1	Low	7.756876	2.572287	Mega	67.574904
2	High	20.348988	1.227180	Micro	272.250108
3	Medium	20.108487	2.728374	Mega	195.102176
4	High	31.653200	7.776978	Nano	273.960377
5	Low	5.561586	3.530209	Nano	39.992087
6	Medium	13.951808	4.477142	Micro	147.150178
7	Medium	28.352592	4.695376	Mega	229.141912
8	Medium	15.322175	4.379750	Macro	222.696768
9	High	26.914287	6.074165	Mega	322.466797

In [290…
```python
# Create a pairplot of the data.

sns.pairplot(data);
```



In [292…
```python
# Calculate the mean sales for each TV category.

print(data.groupby('TV')['Sales'].mean())

print('')

# Calculate the mean sales for each Influencer category .

print(data.groupby('Influencer')['Sales'].mean())
```

```
TV
High      300.853195
Low        90.984101
Medium    195.358032
Name: Sales, dtype: float64

Influencer
Macro    181.670070
Mega     194.487941
Micro    188.321846
Nano     191.874432
Name: Sales, dtype: float64
```

In [294…
```python
# Drop rows that contain missing data and update the DataFrame.

data = data.dropna(axis=0)
```

In [296…
```python
# Rename all columns in data that contain a space.

data = data.rename(columns={'Social Media': 'Social_Media'})
```

In [298…
```python
# Define the OLS formula.

ols_formula = 'Sales ~ C(TV) + Radio'

# Create an OLS model.

OLS = ols(formula = ols_formula, data = data)

# Fit the model.

model = OLS.fit()

# Save the results summary.

model_results = model.summary()

# Display the model results.

model_results
```

Out[298…

OLS Regression Results

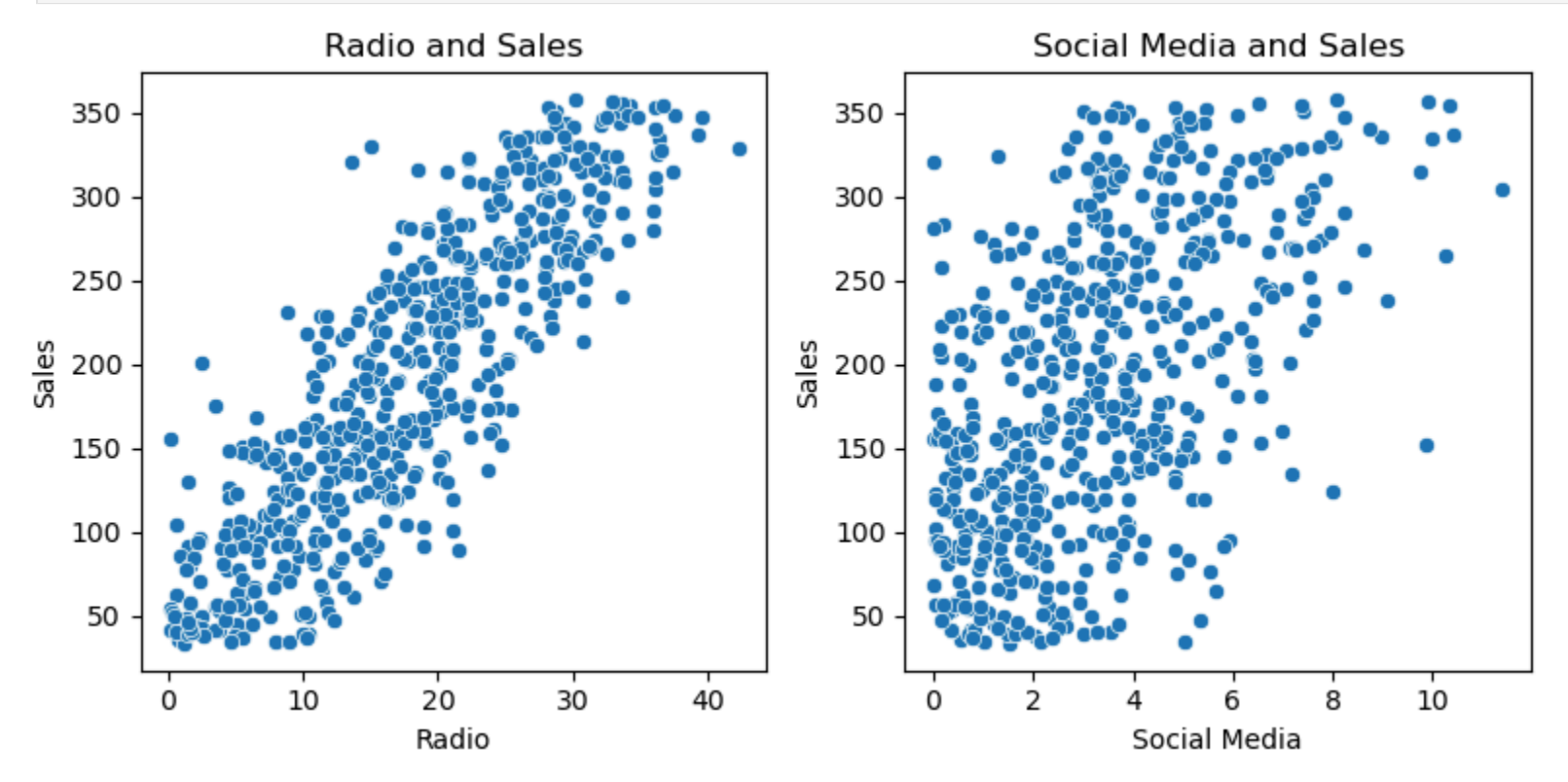
Dep. Variable:	Sales	R-squared:	0.904
Model:	OLS	Adj. R-squared:	0.904
Method:	Least Squares	F-statistic:	1783.
Date:	Fri, 27 Sep 2024	Prob (F-statistic):	1.63e-288
Time:	17:14:01	Log-Likelihood:	-2714.0
No. Observations:	572	AIC:	5436.
Df Residuals:	568	BIC:	5453.
Df Model:	3		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	218.5261	6.261	34.902	0.000	206.228	230.824
C(TV)[T.Low]	-154.2971	4.929	-31.303	0.000	-163.979	-144.616
C(TV)[T.Medium]	-75.3120	3.624	-20.780	0.000	-82.431	-68.193
Radio	2.9669	0.212	14.015	0.000	2.551	3.383

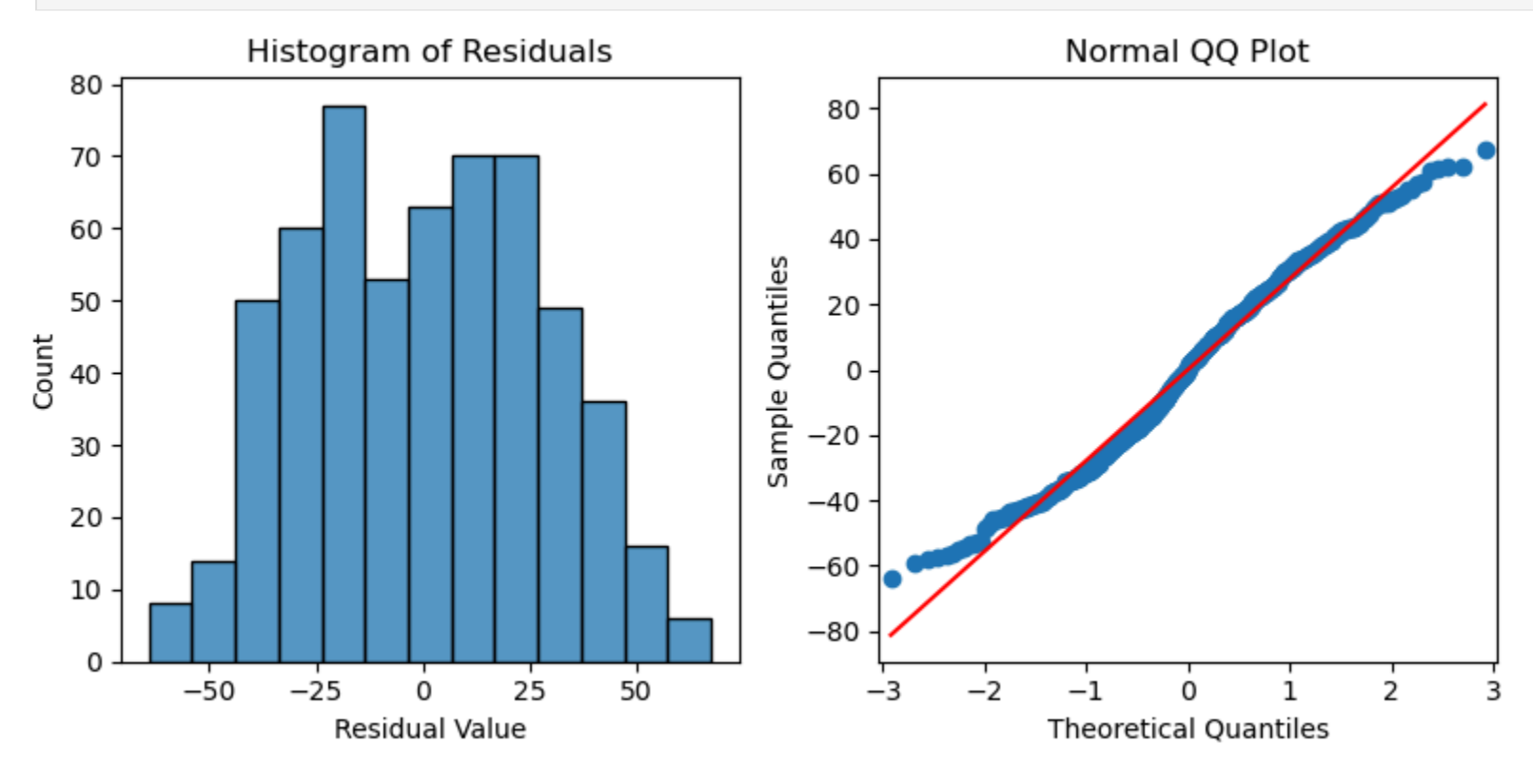
Omnibus:	61.244	Durbin-Watson:	1.870
Prob(Omnibus):	0.000	Jarque-Bera (JB):	18.077
Skew:	0.046	Prob(JB):	0.000119
Kurtosis:	2.134	Cond. No.	142.

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

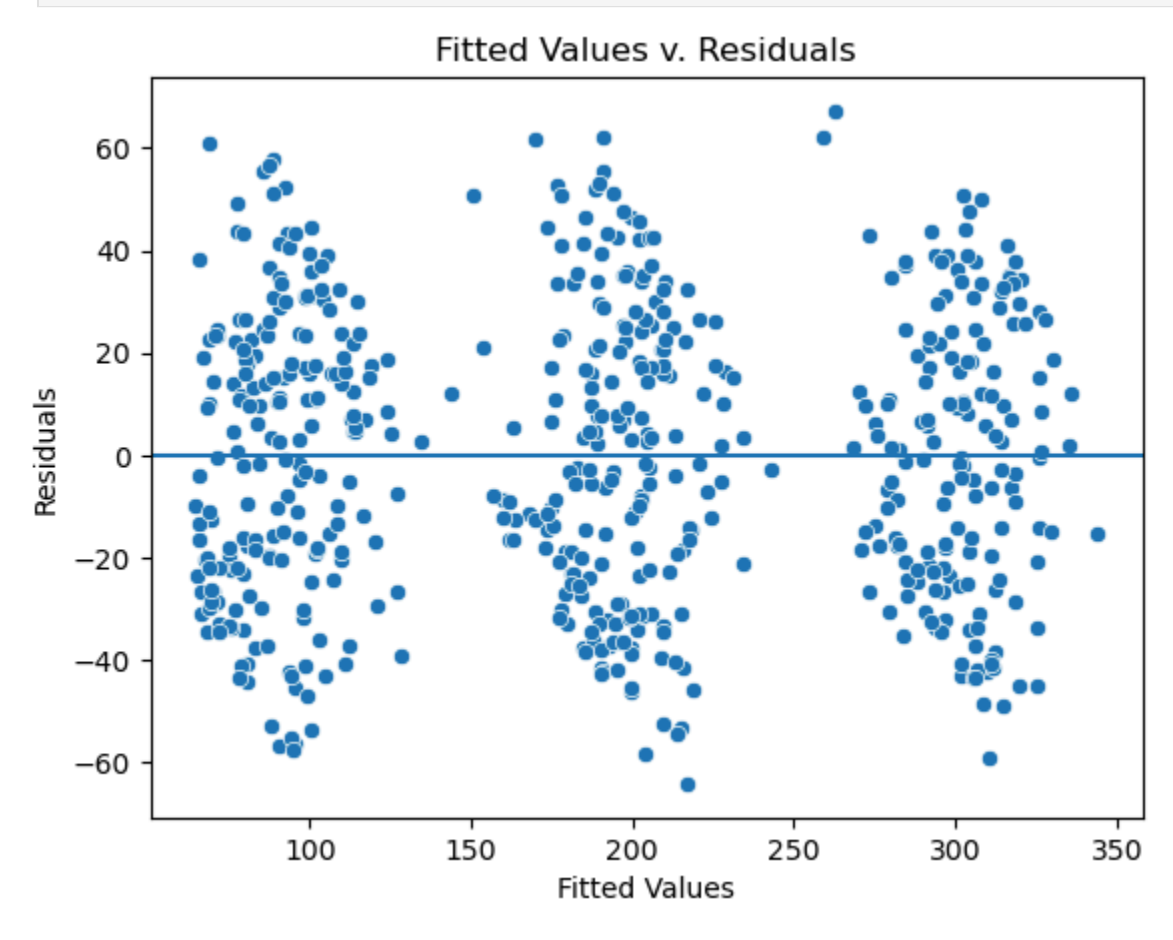
In [300…
```python
# Create a scatterplot for each independent variable and the dependent variable.

# Create a 1x2 plot figure.
fig, axes = plt.subplots(1, 2, figsize = (8,4))

# Create a scatterplot between Radio and Sales.
sns.scatterplot(x = data['Radio'], y = data['Sales'],ax=axes[0])

# Set the title of the first plot.
axes[0].set_title("Radio and Sales")

# Create a scatterplot between Social Media and Sales.
sns.scatterplot(x = data['Social_Media'], y = data['Sales'],ax=axes[1])

# Set the title of the second plot.
axes[1].set_title("Social Media and Sales")

# Set the xlabel of the second plot.
axes[1].set_xlabel("Social Media")

# Use matplotlib's tight_layout() function to add space between plots for a cleaner appearance.
plt.tight_layout()
```



In [302…
```python
# Calculate the residuals.

residuals = model.resid

# Create a 1x2 plot figure.
fig, axes = plt.subplots(1, 2, figsize = (8,4))

# Create a histogram with the residuals.

sns.histplot(residuals, ax=axes[0])

# Set the x label of the residual plot.
axes[0].set_xlabel("Residual Value")

# Set the title of the residual plot.
axes[0].set_title("Histogram of Residuals")

# Create a Q-Q plot of the residuals.

sm.qqplot(residuals, line='s',ax = axes[1])

# Set the title of the Q-Q plot.
axes[1].set_title("Normal QQ Plot")

# Use matplotlib's tight_layout() function to add space between plots for a cleaner appearance.
plt.tight_layout()

# Show the plot.
plt.show()
```

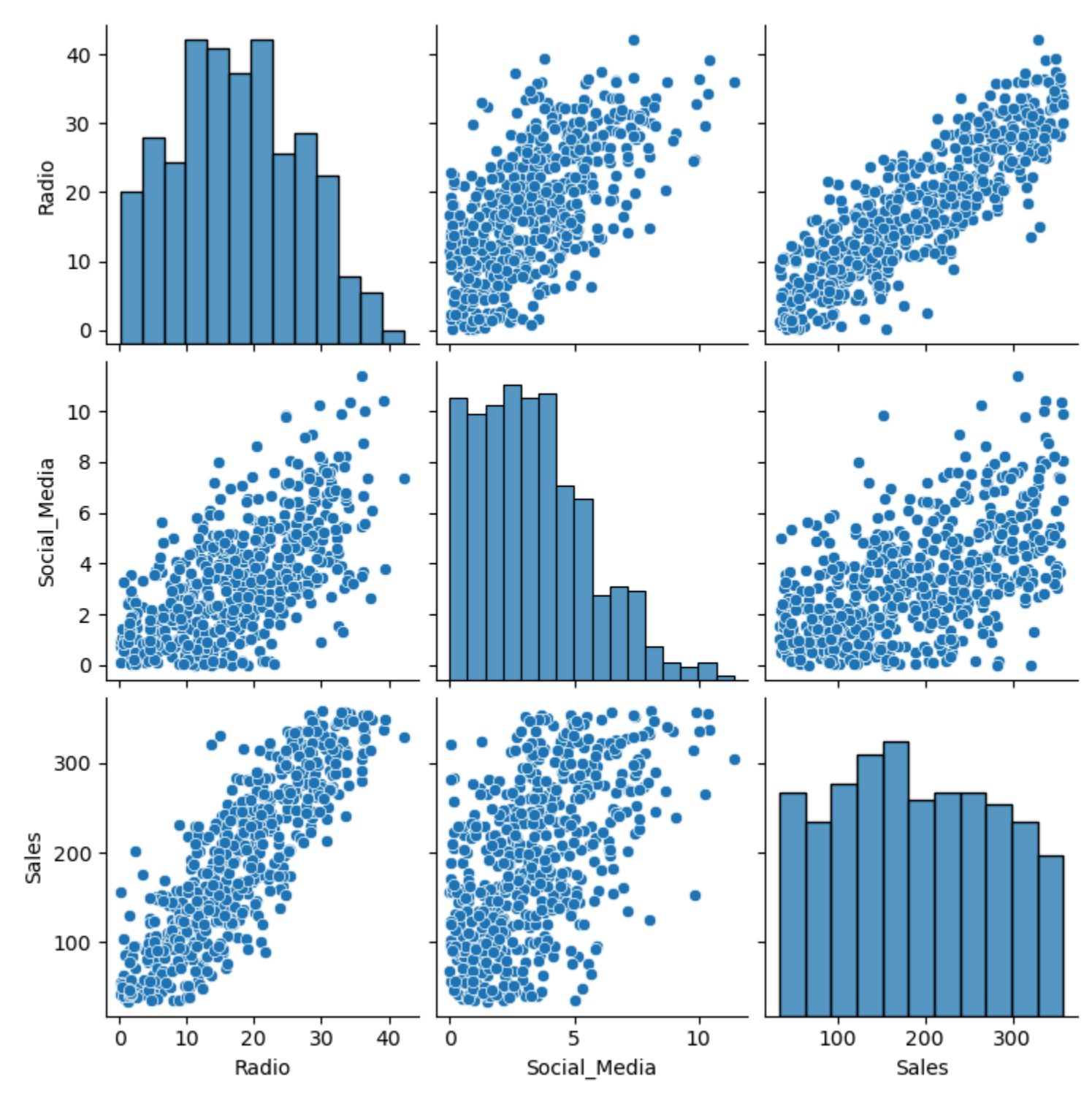


In [304…
```python
# Create a scatterplot with the fitted values from the model and the residuals.

fig = sns.scatterplot(x = model.fittedvalues, y = model.resid)

# Set the x axis label.
fig.set_xlabel("Fitted Values")

# Set the y axis label.
fig.set_ylabel("Residuals")

# Set the title.
fig.set_title("Fitted Values v. Residuals")

# Add a line at y = 0 to visualize the variance of residuals above and below 0.

fig.axhline(0)

# Show the plot.
plt.show()
```



In [306…
```python
# Create a pairplot of the data.

sns.pairplot(data)
```

Out[306… <seaborn.axisgrid.PairGrid at 0x27bf6dca360>



In [308…
```python
# Calculate the variance inflation factor (optional).

# Import variance_inflation_factor from statsmodels.
from statsmodels.stats.outliers_influence import variance_inflation_factor

# Create a subset of the data with the continous independent variables.
X = data[['Radio','Social_Media']]

# Calculate the variance inflation factor for each variable.
vif = [variance_inflation_factor(X.values, i) for i in range(X.shape[1])]

# Create a DataFrame with the VIF results for the column names in X.
df_vif = pd.DataFrame(vif, index=X.columns, columns = ['VIF'])

# Display the VIF results.
df_vif
```

Out[308…

	VIF
Radio	5.170922
Social_Media	5.170922

In [310…
```python
# Display the model results summary.

model_results
```

Out[310…

OLS Regression Results

Dep. Variable:	Sales	R-squared:	0.904
Model:	OLS	Adj. R-squared:	0.904
Method:	Least Squares	F-statistic:	1783.
Date:	Fri, 27 Sep 2024	Prob (F-statistic):	1.63e-288
Time:	17:14:01	Log-Likelihood:	-2714.0
No. Observations:	572	AIC:	5436.
Df Residuals:	568	BIC:	5453.
Df Model:	3		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	218.5261	6.261	34.902	0.000	206.228	230.824
C(TV)[T.Low]	-154.2971	4.929	-31.303	0.000	-163.979	-144.616
C(TV)[T.Medium]	-75.3120	3.624	-20.780	0.000	-82.431	-68.193
Radio	2.9669	0.212	14.015	0.000	2.551	3.383

Omnibus:	61.244	Durbin-Watson:	1.870
Prob(Omnibus):	0.000	Jarque-Bera (JB):	18.077
Skew:	0.046	Prob(JB):	0.000119
Kurtosis:	2.134	Cond. No.	142.

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

```
In [ ]: # some key takeaways

        # Multiple linear regression is a powerful tool to estimate a dependent continuous variable from several independent variables.
        # Exploratory data analysis is useful for selecting both numeric and categorical features for multiple linear regression.
        # Fitting multiple linear regression models may require trial and error to select variables that fit an accurate model while maintaining model assumptions.
```