```
# @title Import Required Libraries
import pandas as pd
import numpy as np

import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

# @title Load the Dataset
df = pd.read_csv("advertising.csv")
df.head()
```


```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 200,\n  \"fields\": [\
n    {\n      \"column\": \"TV\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 85.8542363149081,\n
\"min\": 0.7,\n        \"max\": 296.4,\n        \"num_unique_values\":
190,\n        \"samples\": [\n          287.6,\n          286.0,\n
78.2\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Radio\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 14.846809176168723,\n        \"min\": 0.0,\n        \"max\":
49.6,\n        \"num_unique_values\": 167,\n        \"samples\": [\n
8.2,\n          36.9,\n          44.5\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Newspaper\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
21.778620838522833,\n        \"min\": 0.3,\n        \"max\": 114.0,\n
\"num_unique_values\": 172,\n        \"samples\": [\n          22.3,\n
5.7,\n          17.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Sales\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 5.283892252561874,\n        \"min\": 1.6,\n        \"max\":
27.0,\n        \"num_unique_values\": 121,\n        \"samples\": [\n
19.8,\n          22.6,\n          17.9\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    }\n  ]\n}","type":"dataframe","variable_name":"df"}
```


```
# @title Understand the Dataset
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200 entries, 0 to 199
Data columns (total 4 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   TV         200 non-null    float64
 1   Radio      200 non-null    float64
```

```
 2   Newspaper  200 non-null    float64
 3   Sales      200 non-null    float64
dtypes: float64(4)
memory usage: 6.4 KB
```

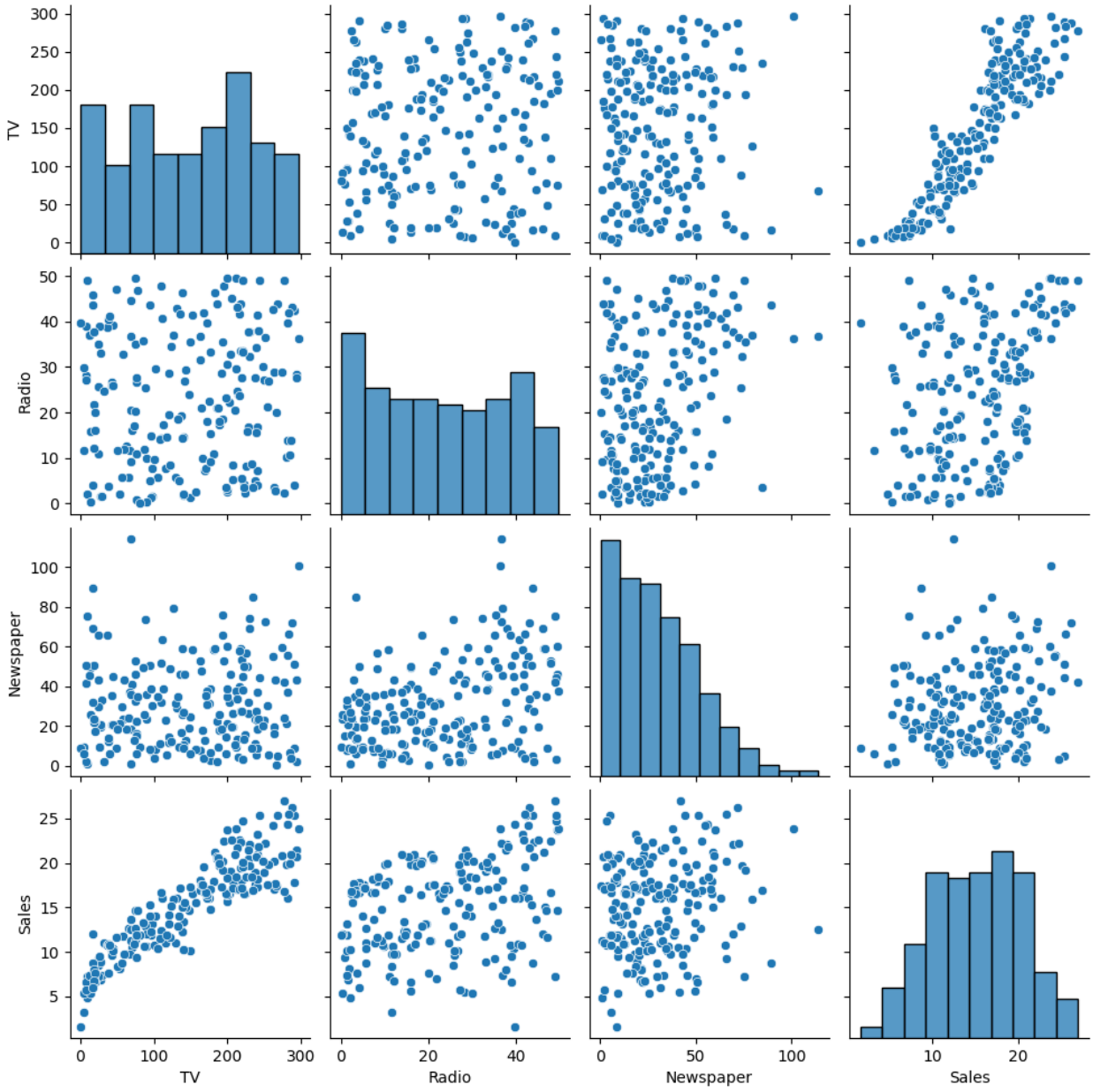
```
df.describe()
```


```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 8,\n  \"fields\": [\n{\n      \"column\": \"TV\",\n      \"properties\": {\n\"dtype\": \"number\",\n        \"std\": 93.12930693433862,\n\"min\": 0.7,\n        \"max\": 296.4,\n        \"num_unique_values\": 8,\n        \"samples\": [\n          147.0425,\n          149.75,\n200.0\n        ],\n        \"semantic_type\": \"\",\n\"description\": \"\"\n      }\n    },\n    {\n      \"column\": \"Radio\",\n      \"properties\": {\n        \"dtype\": \"number\",\n\"std\": 64.62946191825954,\n        \"min\": 0.0,\n        \"max\": 200.0,\n        \"num_unique_values\": 8,\n        \"samples\": [\n23.264000000000006,\n          22.9,\n          200.0\n        ],\n\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\n    },\n    {\n      \"column\": \"Newspaper\",\n\"properties\": {\n        \"dtype\": \"number\",\n        \"std\": 67.53295876114069,\n        \"min\": 0.3,\n        \"max\": 200.0,\n\"num_unique_values\": 8,\n        \"samples\": [\n30.553999999999995,\n          25.75,\n          200.0\n        ],\n\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\n    },\n    {\n      \"column\": \"Sales\",\n      \"properties\": {\n        \"dtype\": \"number\",\n        \"std\": 66.381408327359,\n\"min\": 1.6,\n        \"max\": 200.0,\n        \"num_unique_values\": 8,\n        \"samples\": [\n          15.130500000000001,\n16.0,\n          200.0\n        ],\n        \"semantic_type\": \"\",\n\"description\": \"\"\n      }\n    }\n  ]\n}","type":"dataframe"}
```

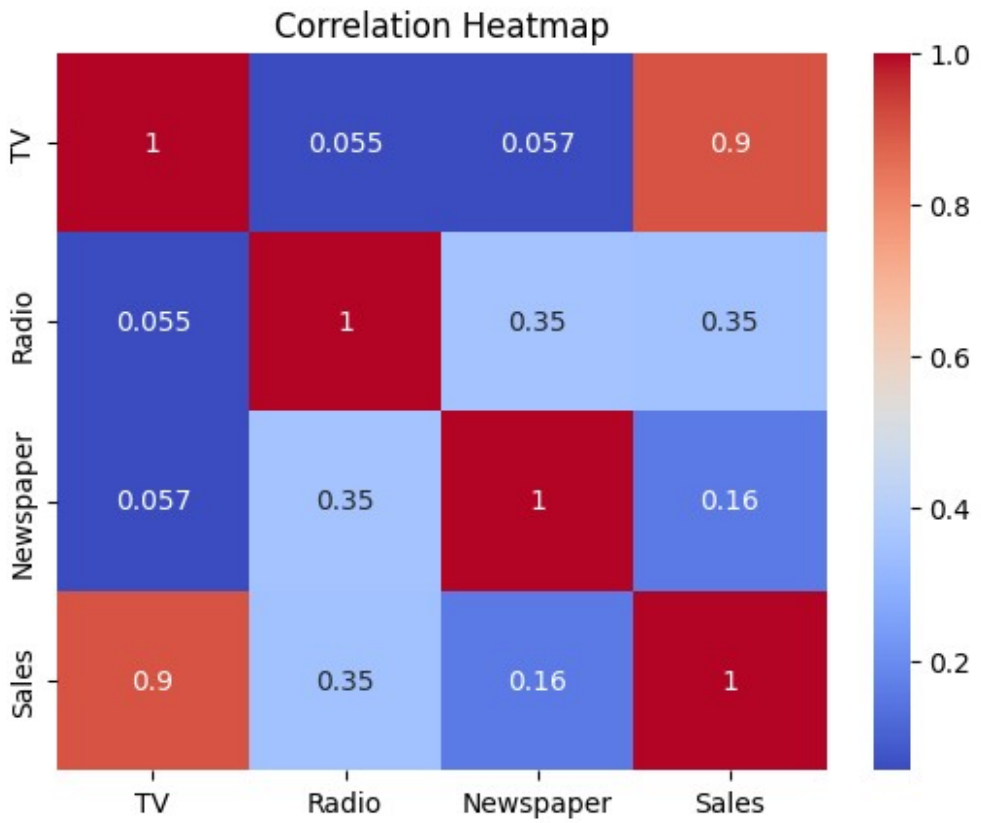

```
df.isnull().sum()
```

```
TV           0
Radio        0
Newspaper    0
Sales        0
dtype: int64
```

```
# @title Exploratory Data Analysis (EDA)
sns.pairplot(df)
plt.show()
```




```
sns.heatmap(df.corr(), annot=True, cmap='coolwarm')
plt.title("Correlation Heatmap")
plt.show()
```




```
# @title Feature Selection
X = df[['TV', 'Radio', 'Newspaper']]
y = df['Sales']

# @title Train-Test Split
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)

# @title Build Regression Model
model = LinearRegression()
model.fit(X_train, y_train)
```

```
LinearRegression()
```

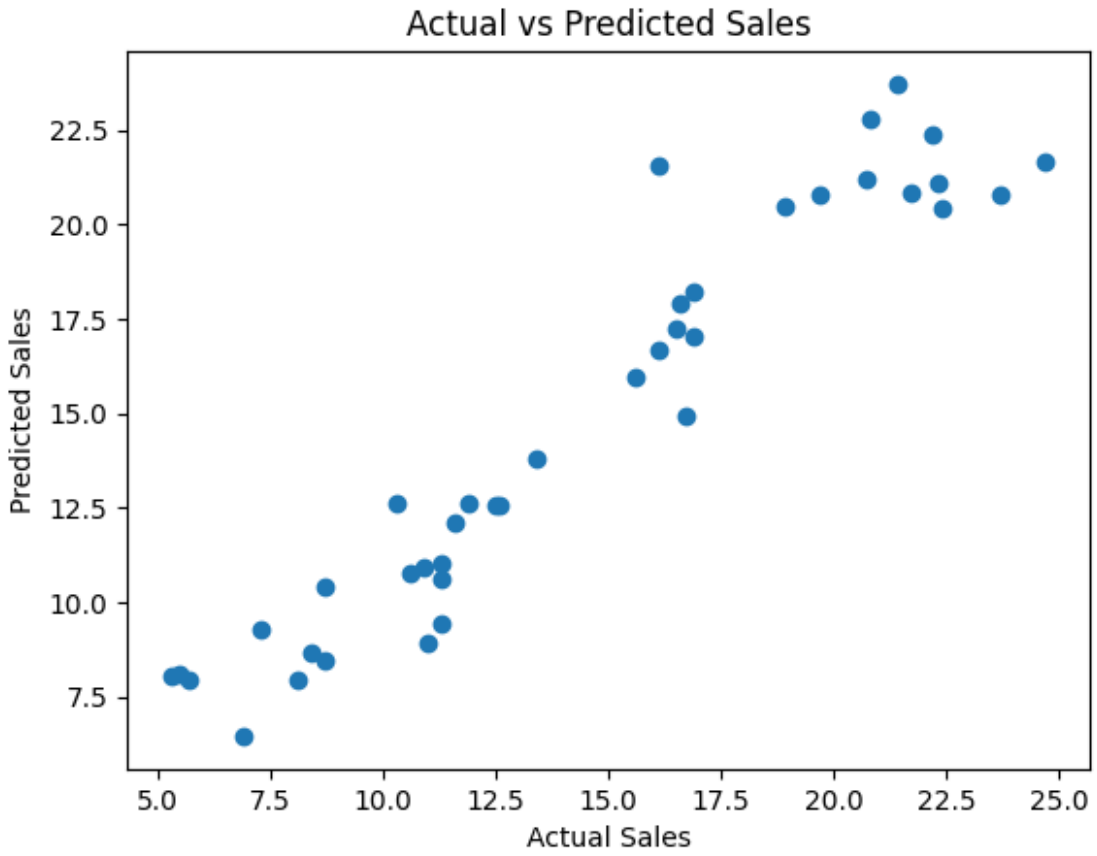
```
# @title Make Predictions
y_pred = model.predict(X_test)

# @title Model Evaluation
mse = mean_squared_error(y_test, y_pred)
print("Mean Squared Error:", mse)
```

```
Mean Squared Error: 2.9077569102710896
```

```
r2 = r2_score(y_test, y_pred)
print("R2 Score:", r2)
```

```
R2 Score: 0.9059011844150826
```

```
# @title Actual vs Predicted Visualization
plt.scatter(y_test, y_pred)
plt.xlabel("Actual Sales")
plt.ylabel("Predicted Sales")
plt.title("Actual vs Predicted Sales")
plt.show()
```




```
# @title Model Coefficients Interpretation
coeff_df = pd.DataFrame({
    'Feature': X.columns,
    'Coefficient': model.coef_
})

coeff_df
```


```
{"summary":"{\n  \"name\": \"coeff_df\",\n  \"rows\": 3,\n
\"fields\": [\n    {\n      \"column\": \"Feature\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
```

\"num_unique_values\": 3,\n   \"samples\": [\n   \"TV\",\n
\"Radio\",\n   \"Newspaper\"\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"Coefficient\",\n
\"properties\": {\n   \"dtype\": \"number\",\n   \"std\":
0.04831639943091294,\n   \"min\": 0.0043366468220340446,\n
\"max\": 0.10094536239295579,\n   \"num_unique_values\": 3,\n
\"samples\": [\n   0.05450927083721978,\n
0.10094536239295579,\n   0.0043366468220340446\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   }\n]\n}","type":"dataframe","variable_name":"coeff_df"}

```
# @title Conclusion
print("Sales Prediction model built successfully.")
print("TV advertising has the highest impact on sales.")
```

```
Sales Prediction model built successfully.
TV advertising has the highest impact on sales.
```