Sales Prediction Analysis using Advertising Dataset

This notebook contains the code for a sales prediction analysis using a linear regression model. The analysis covers exploratory data analysis, model training, prediction, and evaluation.

```
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.preprocessing import StandardScaler
from sklearn.metrics import r2_score, mean_squared_error

# --- Step 1: Load Dataset (Supports Excel or CSV) ---
# Note: The original file path is a local path on your machine.
# Please upload 'advertising_sales_data.xlsx' to your Google Colab
# environment and use a relative path like the one below.
file_path = 'advertising_sales_data.xlsx'

try:
    if file_path.endswith('.csv'):
        df = pd.read_csv(file_path, encoding='latin1')
    else:
        df = pd.read_excel(file_path)
    print("Dataset loaded successfully.")
except FileNotFoundError:
    print(f"Error: The file '{file_path}' was not found. Please upload the file to your Colab environment.")
    exit()

print("Initial Data Preview:")
print(df.head())
print(df.info())

# --- Step 2: Clean Dataset ---
# Keep only numeric columns relevant to the analysis and drop rows with missing values
numeric_cols = ['TV', 'Radio', 'Newspaper', 'Sales']
df = df[numeric_cols].apply(pd.to_numeric, errors='coerce')
df = df.dropna()

print("\nCleaned Data Preview:")
print(df.head())
print(df.describe())
```

Question 1: What is the average TV advertising spend?

This code calculates and prints the average value of the `TV` column, providing a basic statistical insight into the advertising budget for television.

```
# --- Q1: Average TV Advertising ---
avg_tv = df['TV'].mean()
print(f"Average TV Advertising Spend: {avg_tv:.2f}")
```

Question 2: What is the correlation between `Radio` and `Sales`?

Here, the Pearson correlation coefficient between the `Radio` and `Sales` columns is calculated. The result indicates the strength and direction of the linear relationship between spending on radio advertising and the resulting sales.

```
# --- Q2: Correlation between Radio and Sales ---
radio_sales_corr = df['Radio'].corr(df['Sales'])
print(f"Correlation between Radio and Sales: {radio_sales_corr:.2f}")
```

Question 3: Which advertising medium has the highest impact on sales?

This section computes the correlation of each advertising medium (`TV`, `Radio`, `Newspaper`) with `Sales` and identifies the one with the strongest correlation. A heatmap is also generated to visualize the correlations between all the numerical variables.

```
# --- Q3: Highest Impact Advertising Medium ---
corr_with_sales = df.corr()['Sales'].drop('Sales')
```

```
print("\nCorrelation of each advertising medium with Sales:")
print(corr_with_sales)
max_corr_medium = corr_with_sales.idxmax()
print(f"Highest impact on sales: {max_corr_medium}")

sns.heatmap(df.corr(), annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Heatmap')
plt.show()
```

Question 4: What is the performance of the Linear Regression model, and how do the actual sales compare to the predicted sales?

The code below first splits the data into training and testing sets, then trains a linear regression model. It then uses the trained model to predict sales on the test set. Finally, it visualizes the actual vs. predicted sales with a scatter plot and prints key performance metrics (R^2 score and Mean Squared Error) to evaluate the model's accuracy.

```
# --- Step 3: Train-Test Split ---
X = df[['TV', 'Radio', 'Newspaper']]
y = df['Sales']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# --- Train Linear Regression Model ---
model = LinearRegression()
model.fit(X_train, y_train)

# --- Predictions and Q4 Visualization ---
y_pred = model.predict(X_test)

plt.figure(figsize=(8,6))
plt.scatter(y_test, y_pred, color='blue', edgecolors='k')
plt.xlabel('Actual Sales')
plt.ylabel('Predicted Sales')
plt.title('Actual vs Predicted Sales')
plt.grid(True, linestyle='--', alpha=0.5)
plt.show()

r2 = r2_score(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
print(f"Model Performance:\nR2 Score: {r2:.2f}\nMSE: {mse:.2f}")
```

Question 5: Predict sales for new advertising data where TV=200, Radio=40, and Newspaper=50.

This section uses the trained linear regression model to make a prediction for a new, unseen data point with specific advertising budget values for TV, Radio, and Newspaper.

```
# --- Q5: Predict sales for new data ---
new_data = np.array([[200, 40, 50]])
new_prediction = model.predict(new_data)
print(f"Predicted Sales for TV=200, Radio=40, Newspaper=50: {new_prediction[0]:.2f}")
```

Question 6: How does normalization of the features impact the model's performance (R^2 score)?

This code block applies `StandardScaler` to normalize the features, then retrains a linear regression model on the scaled data. The R^2 score is calculated and compared to the previous model to assess the impact of normalization.

```
# --- Q6: Normalization Impact ---
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

X_train_s, X_test_s, y_train_s, y_test_s = train_test_split(X_scaled, y, test_size=0.2, random_state=42)

model_scaled = LinearRegression()
model_scaled.fit(X_train_s, y_train_s)
y_pred_s = model_scaled.predict(X_test_s)

r2_scaled = r2_score(y_test_s, y_pred_s)
print(f"R2 Score after Normalization: {r2_scaled:.2f}")
```

Question 7: How does a model trained with only Radio and Newspaper advertising data compare in performance (R² score) to the full model?

A new linear regression model is trained using only `Radio` and `Newspaper` as features. Its performance is then evaluated using the R² score, allowing for a comparison of the predictive power of a model with a subset of the original features.

```
# --- Q7: Model with only Radio and Newspaper ---
X_rn = df[['Radio', 'Newspaper']]
X_train_rn, X_test_rn, y_train_rn, y_test_rn = train_test_split(X_rn, y, test_size=0.2, random_state=42)

model_rn = LinearRegression()
model_rn.fit(X_train_rn, y_train_rn)
y_pred_rn = model_rn.predict(X_test_rn)

r2_rn = r2_score(y_test_rn, y_pred_rn)
print(f"R2 Score using only Radio & Newspaper: {r2_rn:.2f}")
```