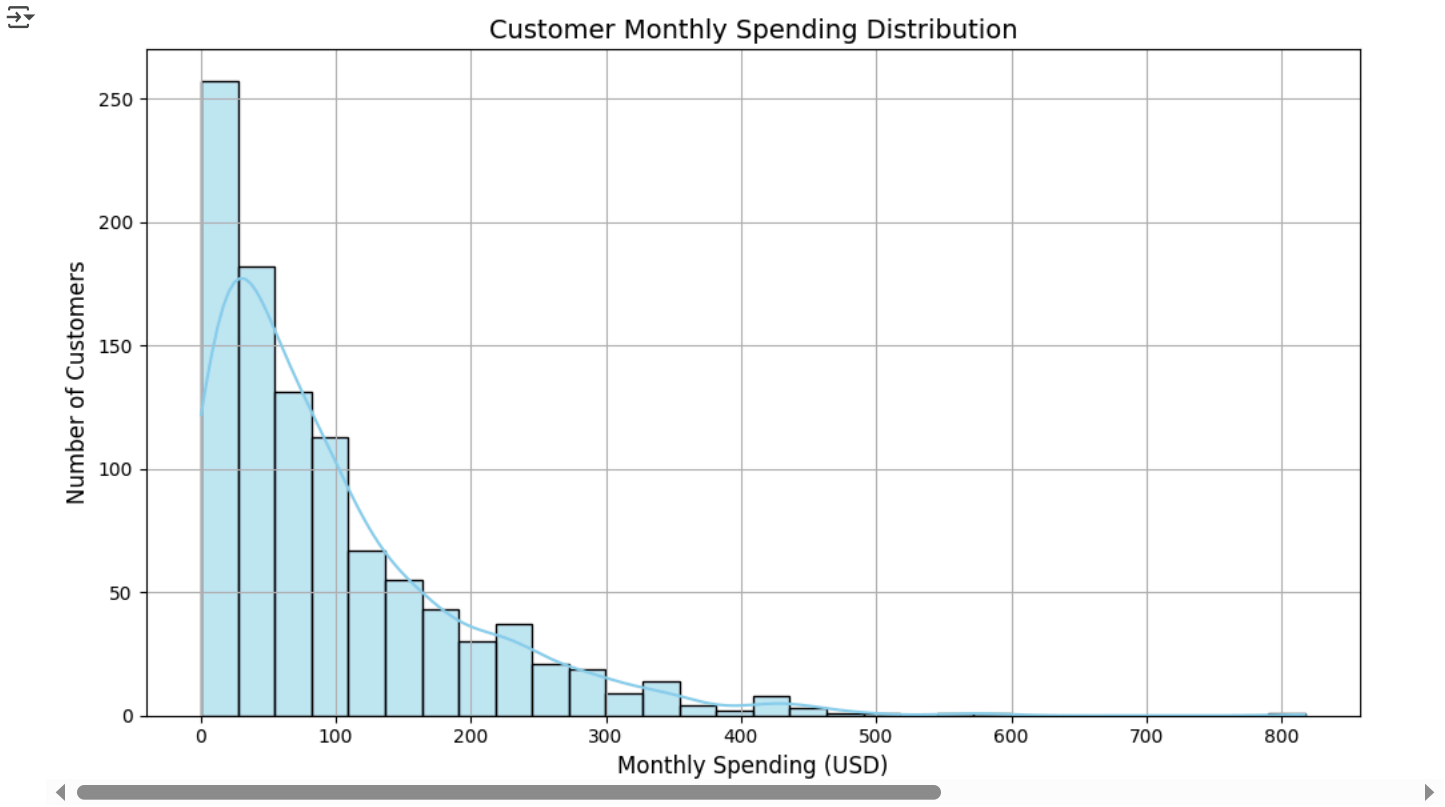
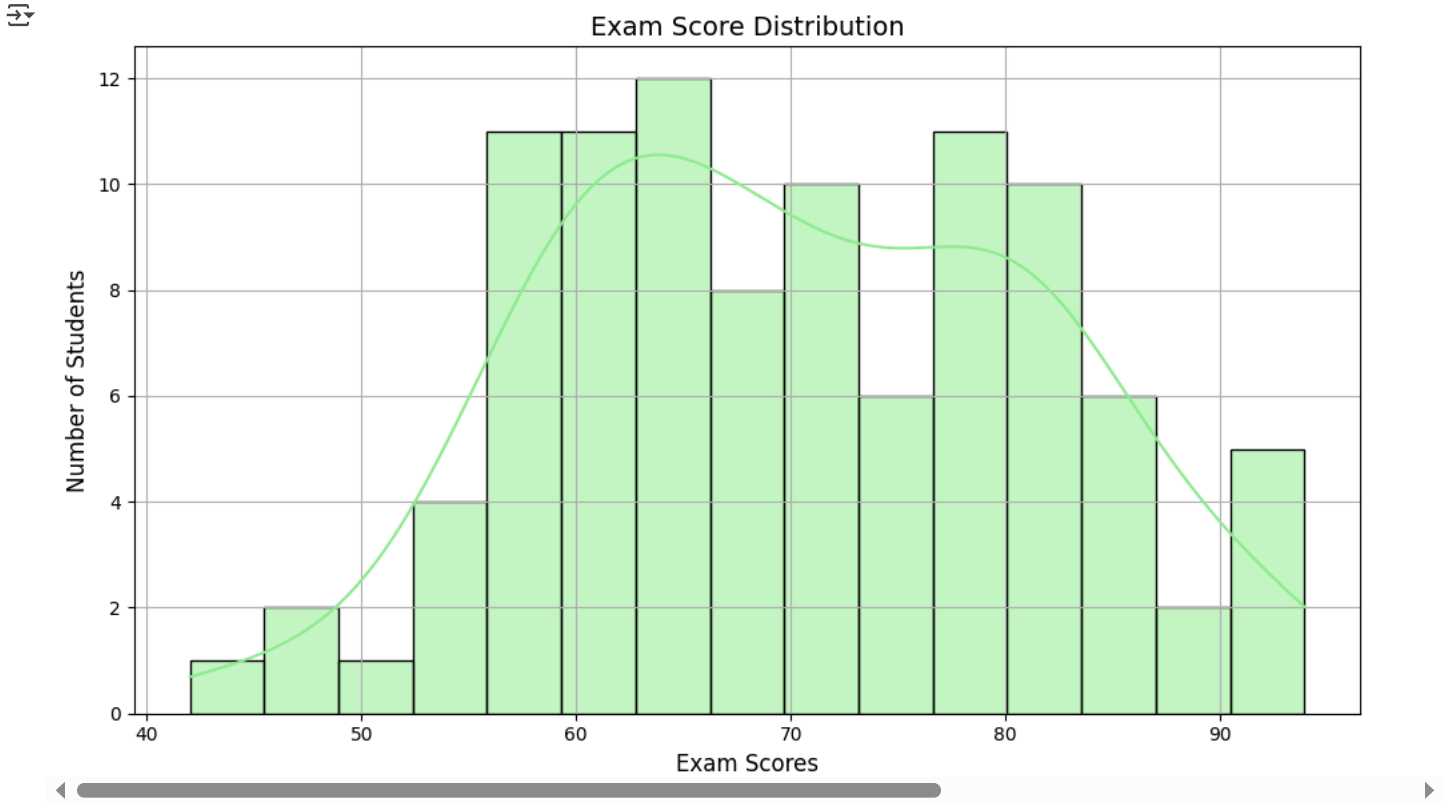
```
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np

# Set random seed for reproducibility
np.random.seed(42)

# Generate synthetic monthly spending data (right-skewed)
monthly_spending = np.random.exponential(scale=100, size=1000)

# Create a histogram with KDE
plt.figure(figsize=(10, 6))
sns.histplot(monthly_spending, kde=True, bins=30, color='skyblue', edgecolor='black')

# Add titles and labels
plt.title("Customer Monthly Spending Distribution", fontsize=14)
plt.xlabel("Monthly Spending (USD)", fontsize=12)
plt.ylabel("Number of Customers", fontsize=12)
plt.grid(True)
plt.tight_layout()

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



Start coding or generate with AI.

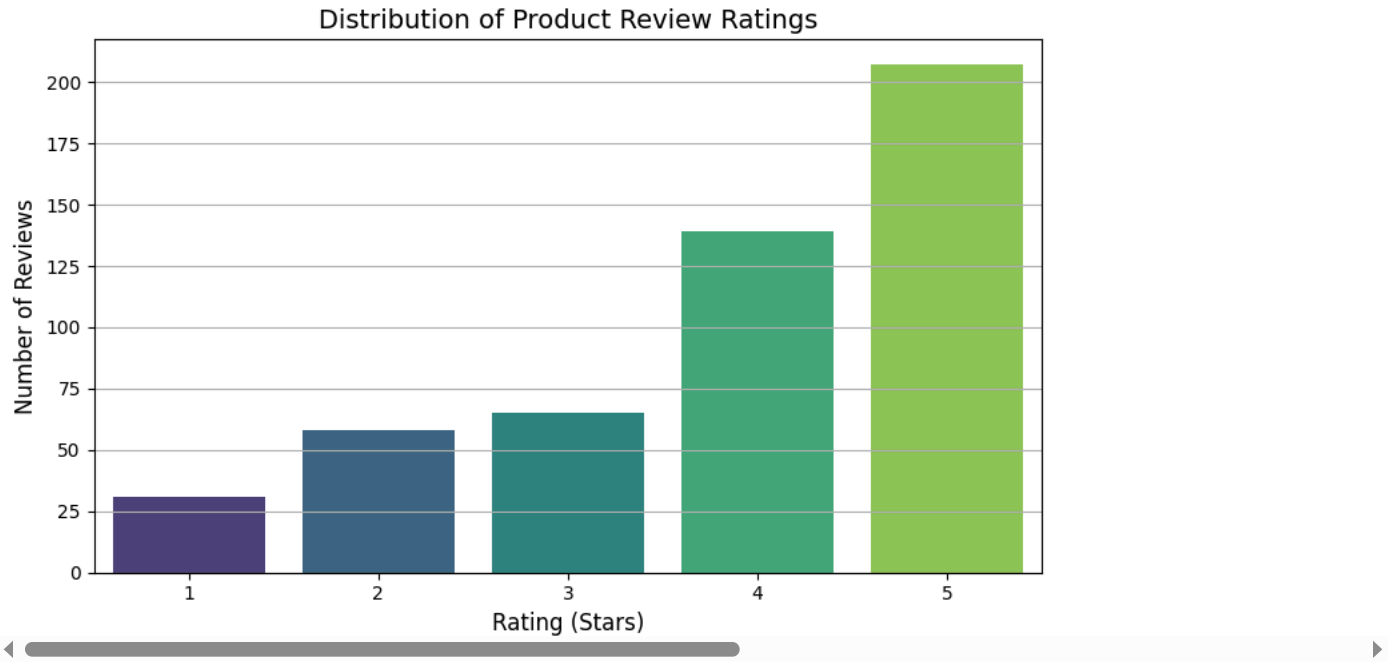
```
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np

# Set seed for reproducibility
np.random.seed(123)

# Generate synthetic exam scores (normally distributed)
exam_scores = np.random.normal(loc=70, scale=10, size=100)

# Create a histogram with KDE
plt.figure(figsize=(10, 6))
sns.histplot(exam_scores, kde=True, bins=15, color='lightgreen', edgecolor='black')

# Add titles and axis labels
```

```
plt.title("Exam Score Distribution", fontsize=14)
plt.xlabel("Exam Scores", fontsize=12)
plt.ylabel("Number of Students", fontsize=12)
plt.grid(True)
plt.tight_layout()

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

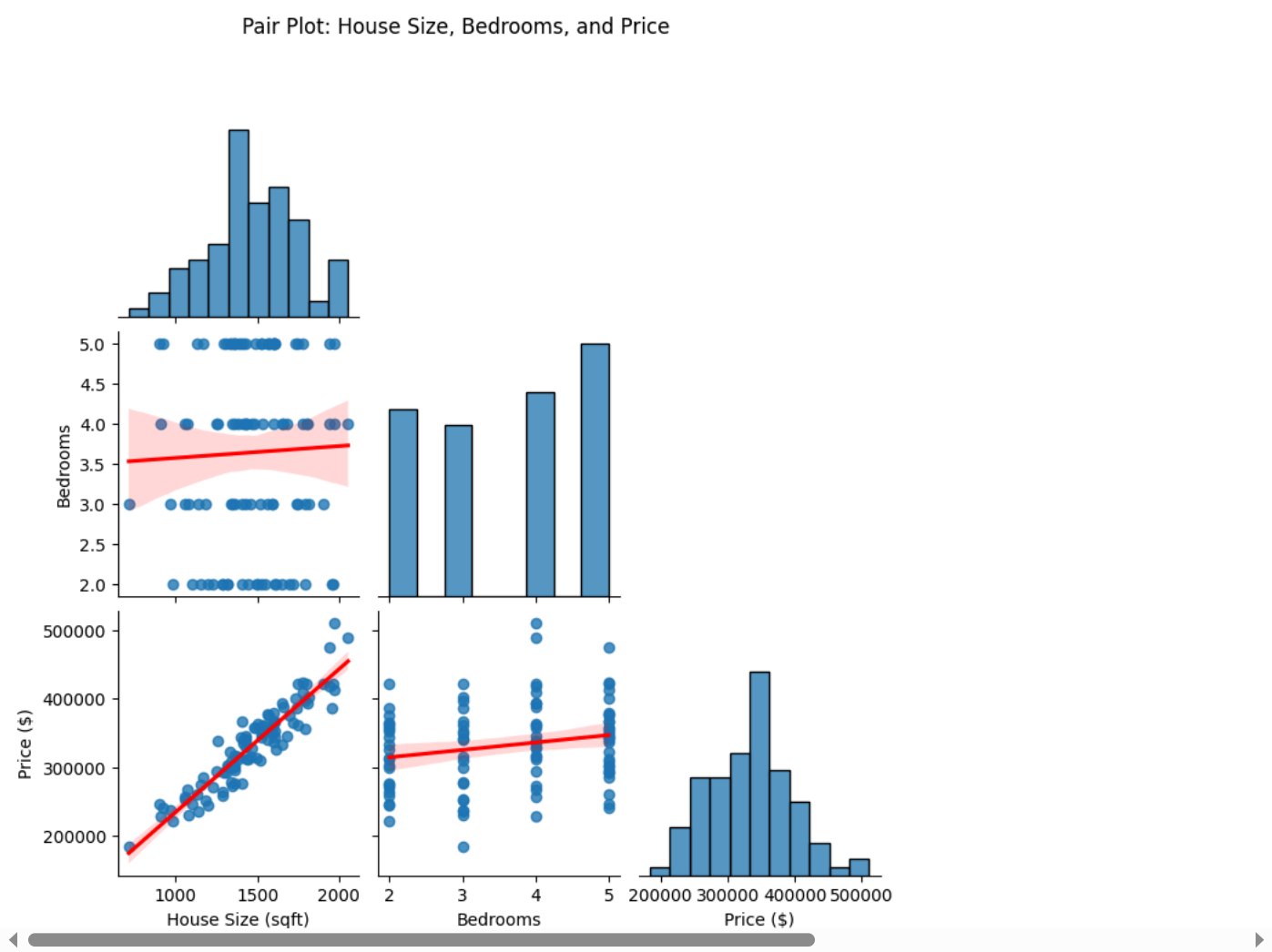



```
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
import pandas as pd

# Simulated product review ratings (1 to 5 stars)
np.random.seed(42)
# Generate 500 ratings with a bias toward higher satisfaction (more 4s and 5s)
ratings = np.random.choice([1, 2, 3, 4, 5], size=500, p=[0.05, 0.10, 0.15, 0.30, 0.40])

# Count frequency of each rating
rating_counts = pd.Series(ratings).value_counts().sort_index()

# Create a bar plot
plt.figure(figsize=(8, 5))
sns.barplot(x=rating_counts.index, y=rating_counts.values, palette='viridis')

# Add titles and labels
plt.title("Distribution of Product Review Ratings", fontsize=14)
plt.xlabel("Rating (Stars)", fontsize=12)
plt.ylabel("Number of Reviews", fontsize=12)
plt.xticks(ticks=[0, 1, 2, 3, 4], labels=[1, 2, 3, 4, 5])
plt.grid(axis='y')
plt.tight_layout()

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

```
<ipython-input-4-508f90dd96b7>:16: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend

  sns.barplot(x=rating_counts.index, y=rating_counts.values, palette='viridis')
```

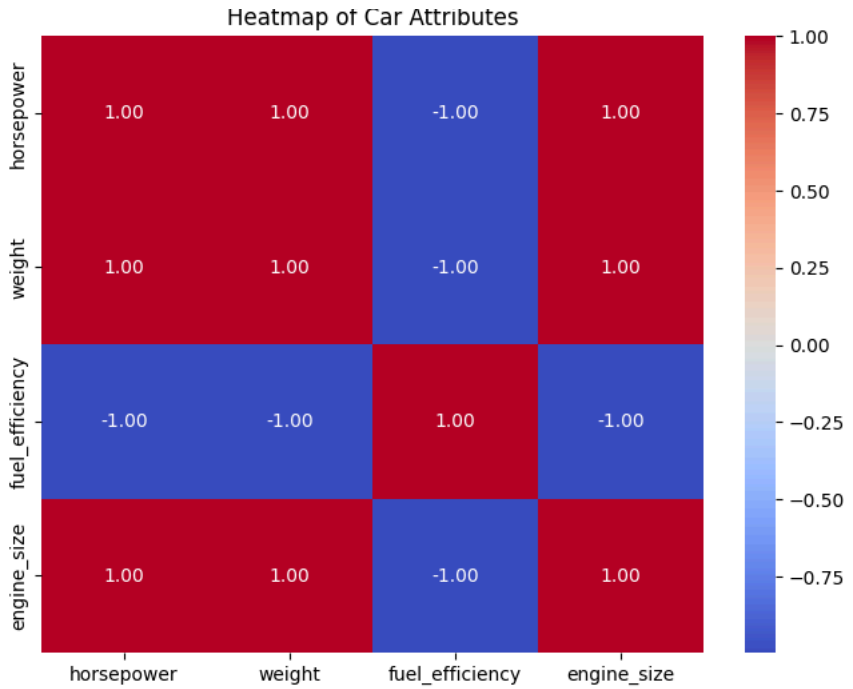


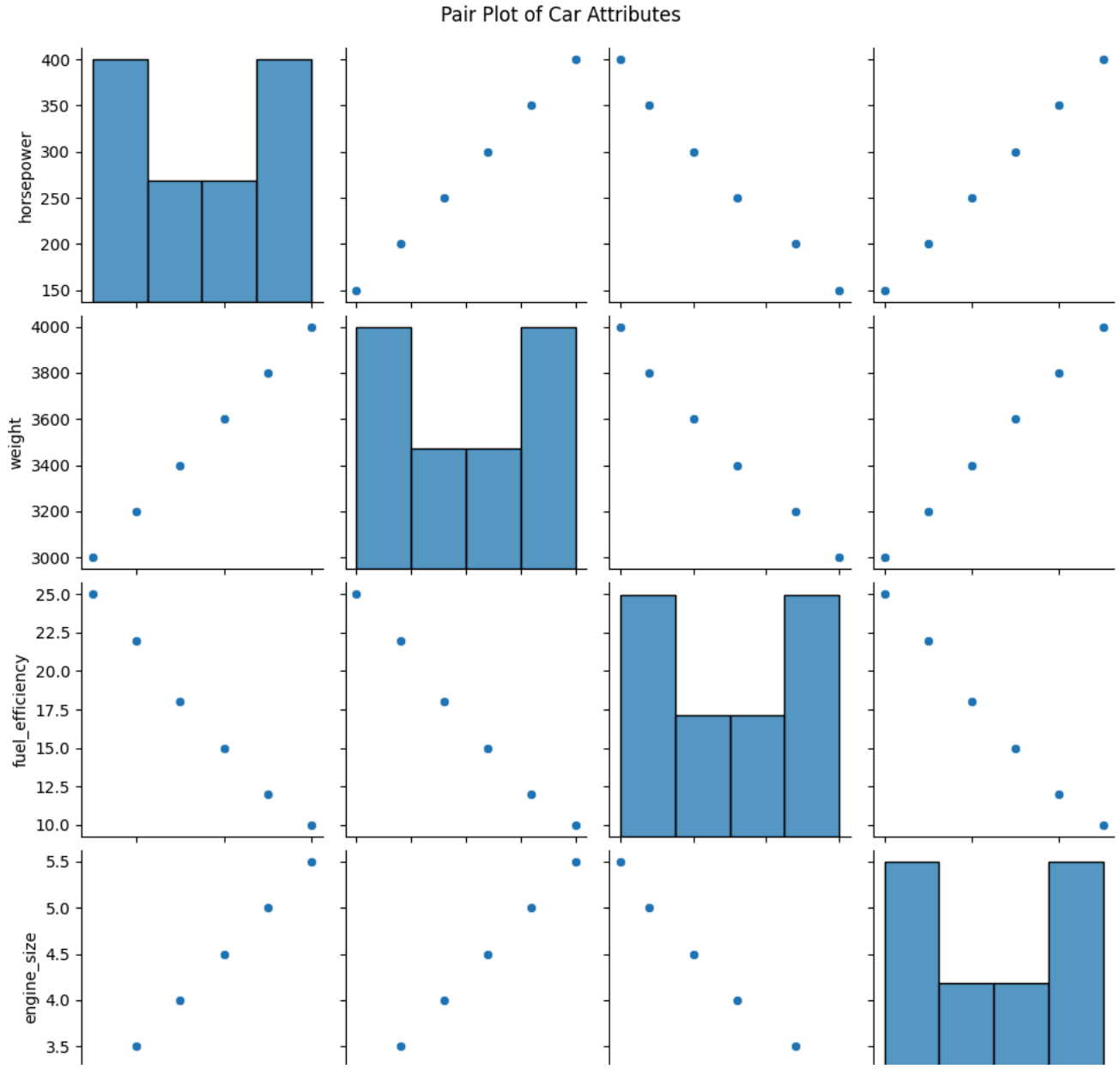

```
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np

# Simulate house dataset
np.random.seed(42)
size = np.random.normal(1500, 300, 100)  # house size in sq ft
bedrooms = np.random.randint(2, 6, size=100)  # number of bedrooms
price = size * 200 + bedrooms * 10000 + np.random.normal(0, 20000, 100)  # price in $

# Create DataFrame
df = pd.DataFrame({
    'House Size (sqft)': size,
    'Bedrooms': bedrooms,
    'Price ($)': price
})

# Pair plot
sns.pairplot(df, corner=True, kind='reg', plot_kws={'line_kws': {'color': 'red'}})
plt.suptitle("Pair Plot: House Size, Bedrooms, and Price", y=1.02)
plt.show()
```

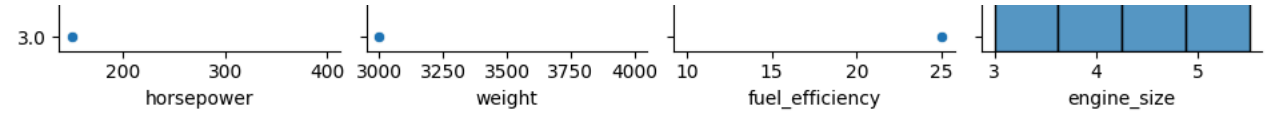



```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Example data (replace this with your actual data)
data = {
    'horsepower': [150, 200, 250, 300, 350, 400],
    'weight': [3000, 3200, 3400, 3600, 3800, 4000],
    'fuel_efficiency': [25, 22, 18, 15, 12, 10],
    'engine_size': [3.0, 3.5, 4.0, 4.5, 5.0, 5.5]
}

# Creating a DataFrame
df = pd.DataFrame(data)

# Heatmap: correlation matrix
plt.figure(figsize=(8, 6))
corr_matrix = df.corr()  # Calculate correlation matrix
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', fmt='.2f', cbar=True)
plt.title('Heatmap of Car Attributes')
plt.show()

# Pair plot: relationships between all variables
sns.pairplot(df)
plt.suptitle('Pair Plot of Car Attributes', y=1.02)
plt.show()

# Analysis: Correlation interpretation
# You can print the correlation matrix for further interpretation
print(corr_matrix)
```

[+]

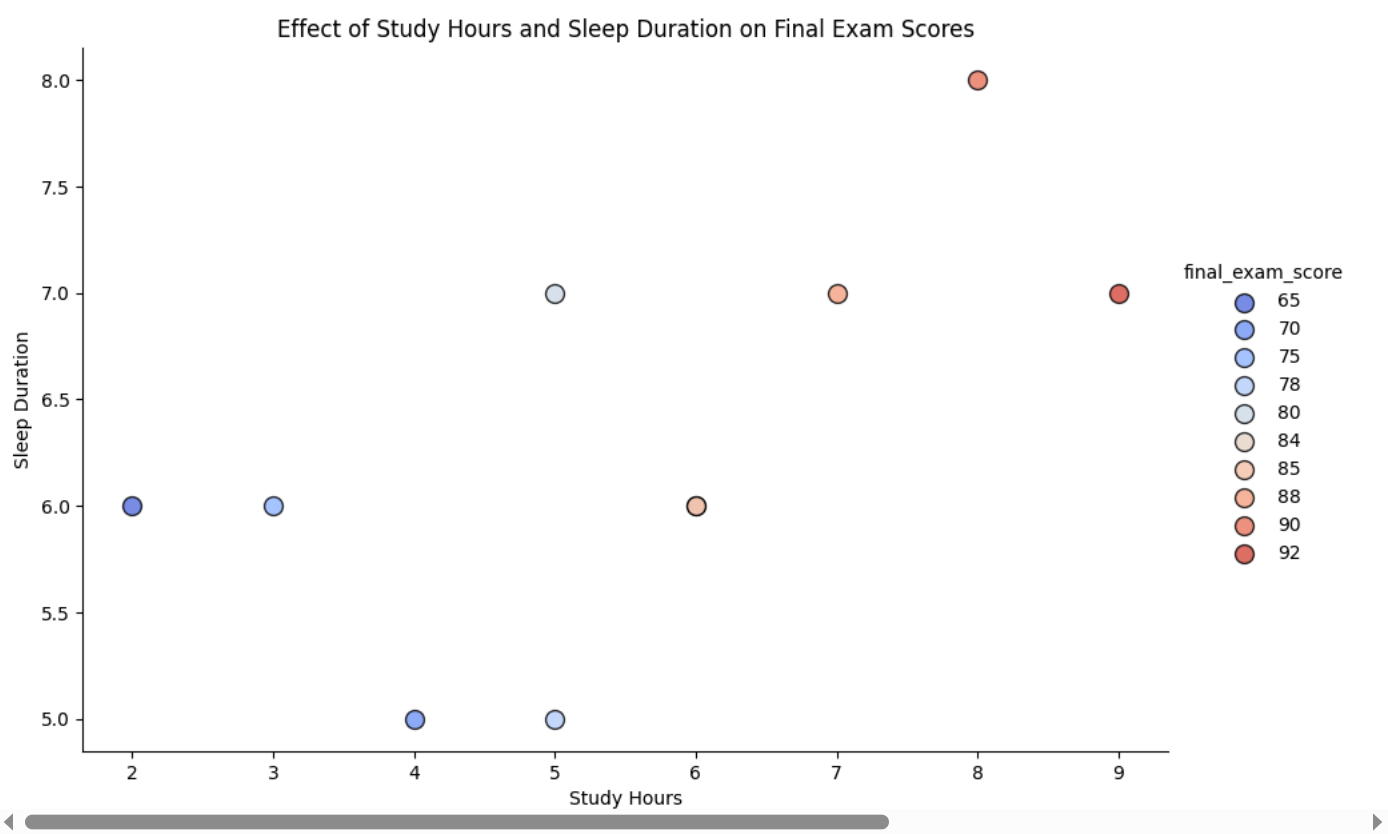

Heatmap of Car Attributes
horsepower
weight
fuel_efficiency
engine_size
1.00
1.00
-1.00
1.00
1.00
1.00
-1.00
1.00
-1.00
-1.00
1.00
-1.00
1.00
1.00
-1.00
1.00
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75


Pair Plot of Car Attributes
horsepower
weight
fuel_efficiency
engine_size



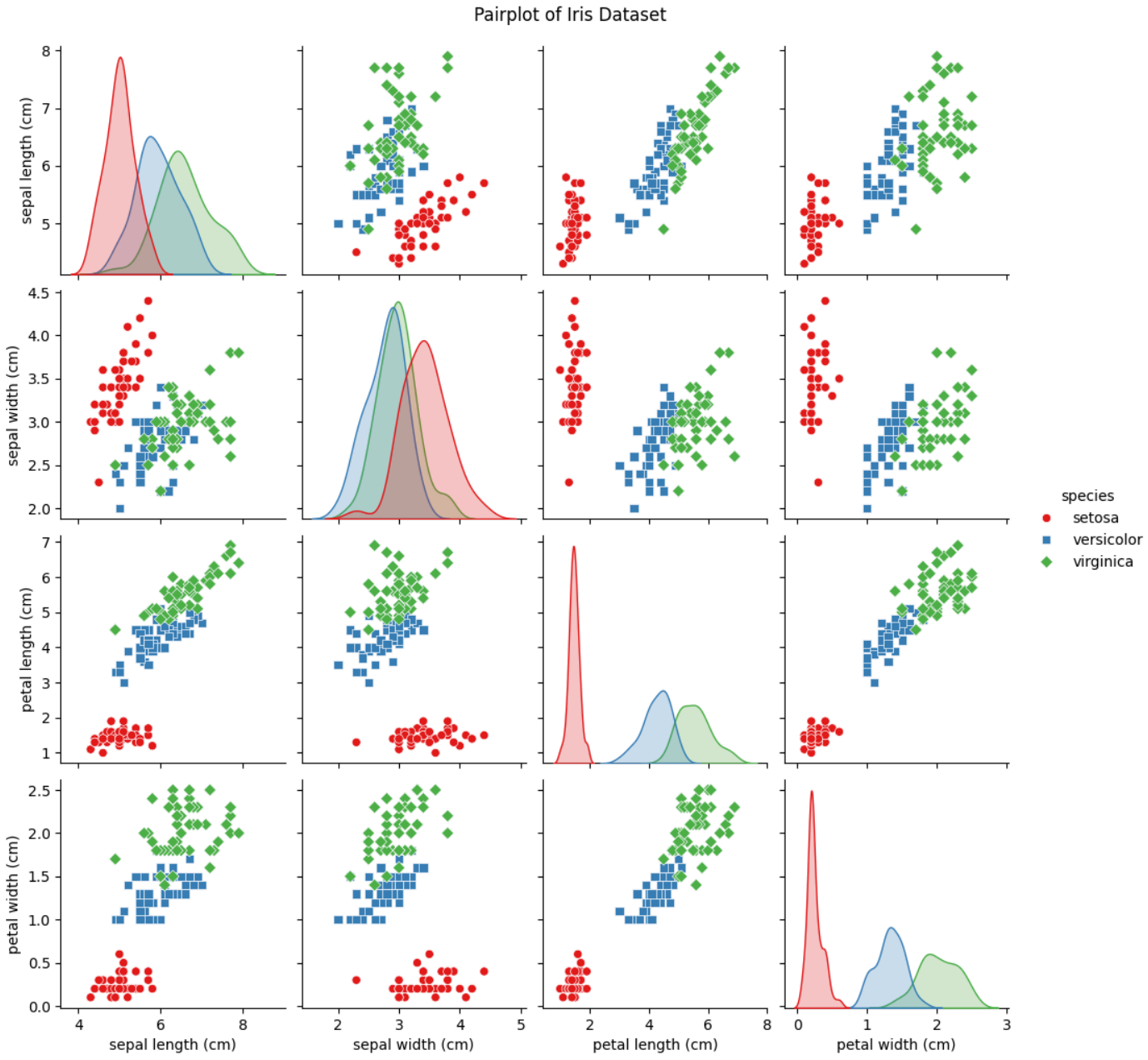
	horsepower	weight	fuel_efficiency	engine_size
horsepower	1.00000	1.00000	-0.99591	1.00000
weight	1.00000	1.00000	-0.99591	1.00000
fuel_efficiency	-0.99591	-0.99591	1.00000	-0.99591
engine_size	1.00000	1.00000	-0.99591	1.00000

```
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd

# Example data (you can replace this with your actual dataset)
data = {
    'study_hours': [5, 3, 8, 6, 4, 7, 2, 9, 5, 6],
    'sleep_duration': [7, 6, 8, 6, 5, 7, 6, 7, 5, 6],
    'final_exam_score': [80, 75, 90, 85, 70, 88, 65, 92, 78, 84]
}

df = pd.DataFrame(data)

# Scatter plot with 'study_hours' and 'sleep_duration' affecting 'final_exam_score'
sns.lmplot(data=df, x='study_hours', y='sleep_duration', hue='final_exam_score',
           palette='coolwarm', markers='o', height=6, aspect=1.5,
           scatter_kws={'s': 100, 'edgecolor': 'black'})

plt.title('Effect of Study Hours and Sleep Duration on Final Exam Scores')
plt.xlabel('Study Hours')
plt.ylabel('Sleep Duration')
plt.show()
```



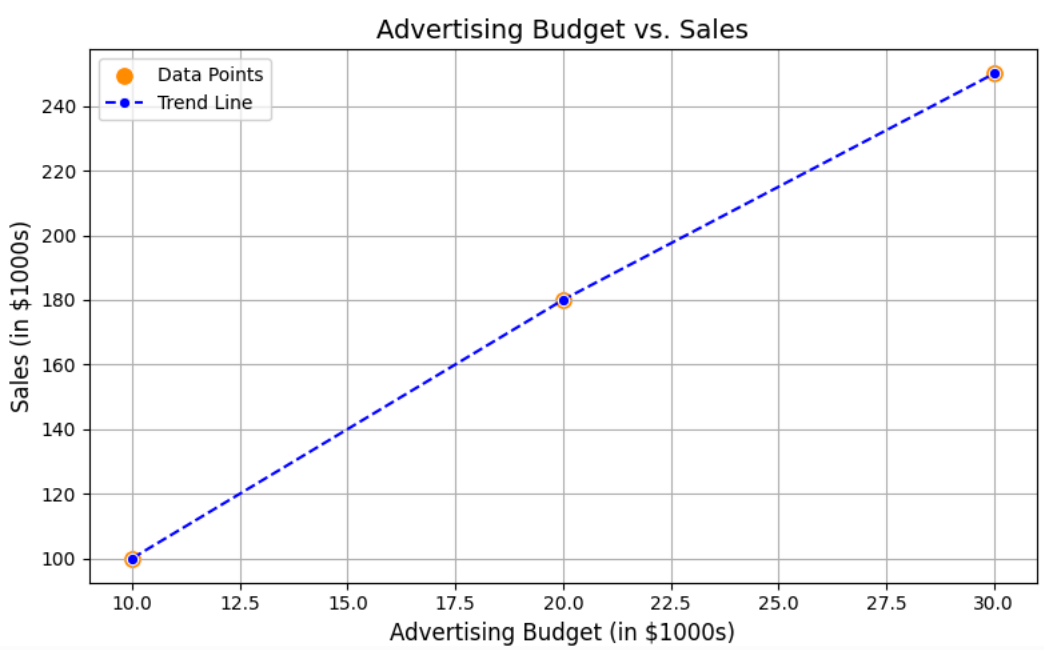

```
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.datasets import load_iris
import pandas as pd

# Load the Iris dataset from sklearn
iris = load_iris()
df = pd.DataFrame(iris.data, columns=iris.feature_names)
```

```
# Add species column to the DataFrame
df['species'] = iris.target
df['species'] = df['species'].map({0: 'setosa', 1: 'versicolor', 2: 'virginica'})

# Create a pairplot
sns.pairplot(df, hue='species', markers=["o", "s", "D"], palette="Set1")

plt.suptitle('Pairplot of Iris Dataset', y=1.02)
plt.show()
```




```
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
import pandas as pd

# Simulated data for 3 months
# Advertising budget in $1000s
ad_budget = [10, 20, 30]  # in thousands
sales = [100, 180, 250]   # in thousands
```

```
# Create a DataFrame
data = pd.DataFrame({
    'Advertising Budget ($1000s)': ad_budget,
    'Sales ($1000s)': sales
})

# Plotting
plt.figure(figsize=(8, 5))
sns.scatterplot(x='Advertising Budget ($1000s)', y='Sales ($1000s)', data=data, s=100, color='darkorange', label='Data Points')
sns.lineplot(x='Advertising Budget ($1000s)', y='Sales ($1000s)', data=data, color='blue', linestyle='--', marker='o', label='Trend Line')

# Add titles and labels
plt.title("Advertising Budget vs. Sales", fontsize=14)
plt.xlabel("Advertising Budget (in $1000s)", fontsize=12)
plt.ylabel("Sales (in $1000s)", fontsize=12)
plt.grid(True)
plt.legend()
plt.tight_layout()

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




```
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np

# Set random seed for reproducibility
np.random.seed(202)

# Simulated data for 100 patients
ages = np.random.randint(20, 80, size=100)  # ages between 20 and 80
# Simulate blood sugar level with some relation to age + random noise
blood_sugar = 70 + (ages * 0.5) + np.random.normal(0, 10, size=100)

# Create scatter plot
plt.figure(figsize=(9, 6))
sns.scatterplot(x=ages, y=blood_sugar, color='mediumseagreen', edgecolor='black')

# Add regression line to visualize trend
sns.regplot(x=ages, y=blood_sugar, scatter=False, color='red', line_kws={'linestyle': '--'})

# Add titles and labels
plt.title("Age vs. Blood Sugar Level", fontsize=14)
plt.xlabel("Age (years)", fontsize=12)
plt.ylabel("Blood Sugar Level (mg/dL)", fontsize=12)
plt.grid(True)
plt.tight_layout()

# Show the plot
```

```
plt.show()
```


```
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np

# Set seed for reproducibility
np.random.seed(101)

# Simulate data: study hours (between 0 to 10)
study_hours = np.random.uniform(0, 10, size=100)

# Simulate scores based on study hours with some noise
# Assuming a positive correlation
scores = 50 + (study_hours * 5) + np.random.normal(0, 5, size=100)

# Create a scatter plot
plt.figure(figsize=(8, 6))
sns.scatterplot(x=study_hours, y=scores, color='dodgerblue', edgecolor='black')

# Add regression line for better visualization of trend
sns.regplot(x=study_hours, y=scores, scatter=False, color='red', line_kws={"linestyle":"--"})

# Add titles and labels
plt.title("Study Hours vs. Exam Scores", fontsize=14)
plt.xlabel("Study Hours", fontsize=12)
plt.ylabel("Exam Scores", fontsize=12)
plt.grid(True)
plt.tight_layout()

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