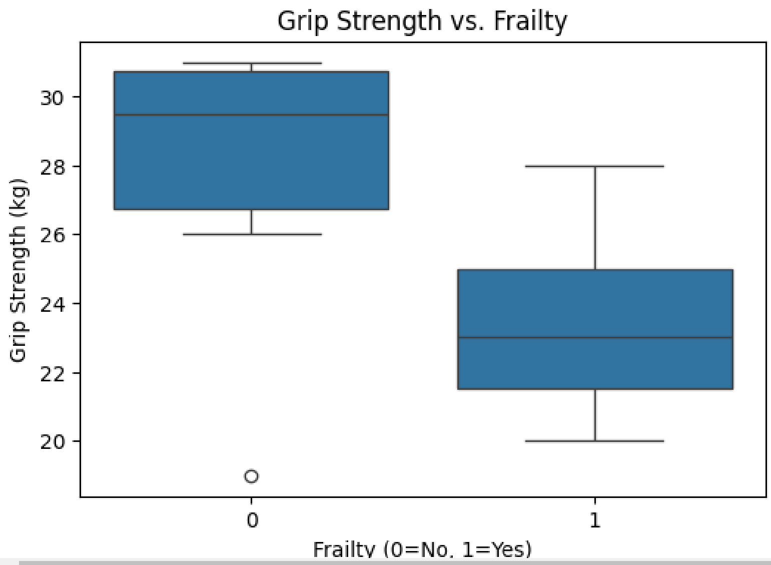
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
# Create a frailty dataset
frailty_data = """Height,Weight,Age,Grip Strength,Frailty
65.8,112,30,30,N
71.5,136,19,31,N
69.4,153,45,29,N
68.2,142,22,28,Y
67.8,144,29,24,Y
68.7,123,50,26,N
69.8,141,51,22,Y
70.1,136,23,20,Y
67.9,112,17,19,N
66.8,120,39,31,N
"""

# Save it as a CSV file
with open("frailty_data.csv", "w") as file:
    file.write(frailty_data)
```

```
# Load the CSV file into a DataFrame
df = pd.read_csv("frailty_data.csv")

# Convert Frailty column to numeric (Y=1, N=0)
df["Frailty"] = df["Frailty"].map({"Y": 1, "N": 0})

# Display summary statistics
df.describe()
```

```
          Height      Weight        Age  Grip Strength    Frailty
count  10.000000   10.000000  10.000000      10.000000  10.000000
mean   68.600000  131.900000  32.500000      26.000000   0.400000
std     1.670662   14.231811  12.860361       4.521553   0.516398
min    65.800000  112.000000  17.000000      19.000000   0.000000
25%    67.825000  120.750000  22.250000      22.500000   0.000000
50%    68.450000  136.000000  29.500000      27.000000   0.000000
75%    69.700000  141.750000  43.500000      29.750000   1.000000
max    71.500000  153.000000  51.000000      31.000000   1.000000
```

```
plt.figure(figsize=(6,4))
sns.boxplot(x=df["Frailty"], y=df["Grip Strength"])
plt.xlabel("Frailty (0=No, 1=Yes)")
plt.ylabel("Grip Strength (kg)")
plt.title("Grip Strength vs. Frailty")
plt.show()
```



Question 2: Student Performance Visualization

```
# Load the dataset
df = pd.read_csv("/content/StudentsPerformance .csv")

# Display the first few rows
df.head()
```

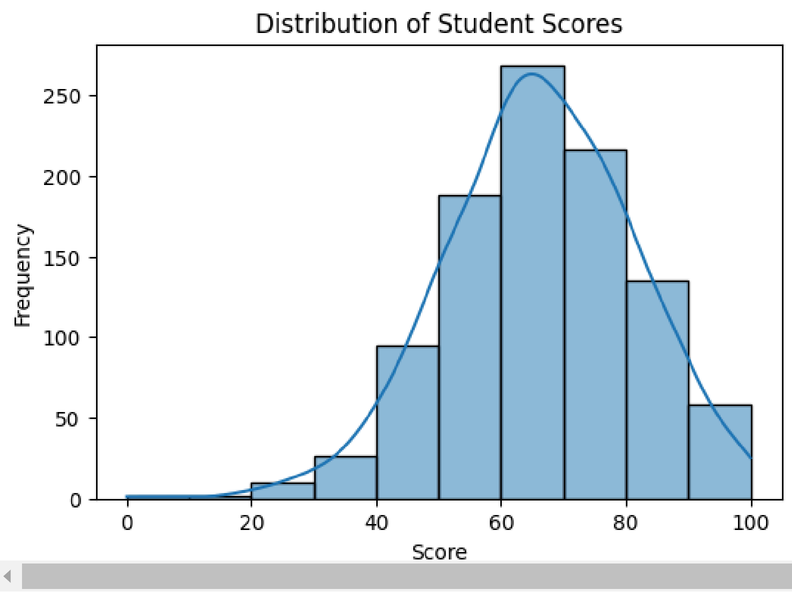
	gender	race/ethnicity	parental level of education	lunch	test preparation course	math score	reading score	writing score
0	female	group B	bachelor's degree	standard	none	72	72	74
1	female	group C	some college	standard	completed	69	90	88
2	female	group B	master's degree	standard	none	90	95	93
3	male	group A	associate's degree	free/reduced	none	47	57	44
4	male	group C	some college	standard	none	76	78	75

Next steps: Generate code with df View recommended plots New interactive sheet

Step 3: Perform 5 Data Visualizations

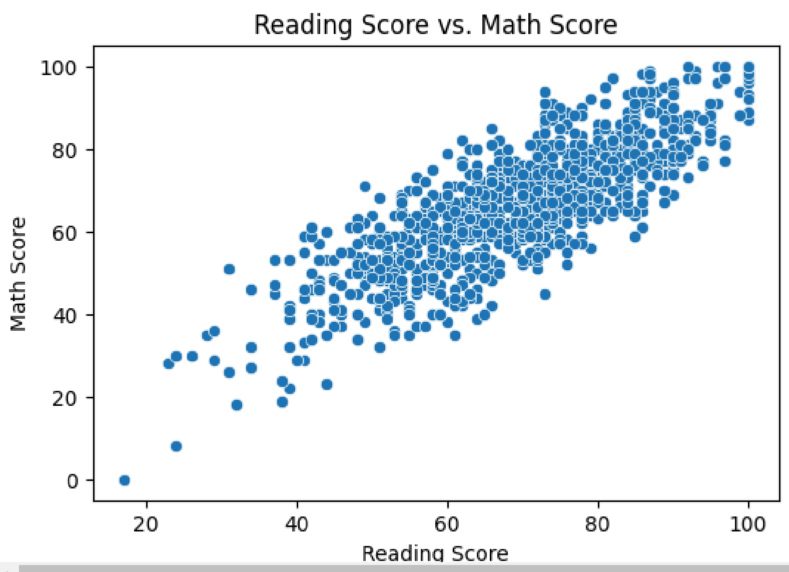
1. Histogram of Scores

```
plt.figure(figsize=(6,4))
# Assuming the column is named 'math score', 'reading score', or 'writing score'
# Replace with the actual column name from your CSV file
sns.histplot(df["math score"], bins=10, kde=True)
plt.xlabel("Score")
plt.ylabel("Frequency")
plt.title("Distribution of Student Scores")
plt.show()
```



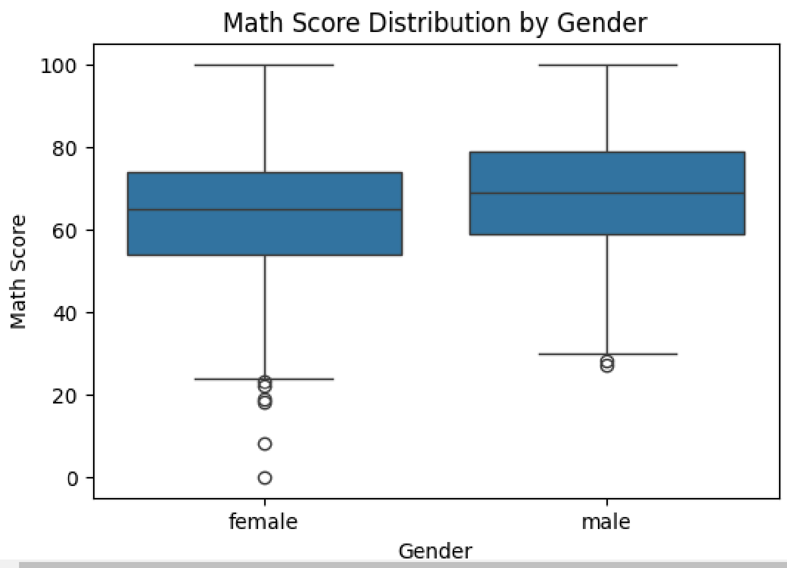
2. Scatter Plot: Study Hours vs. Score

```
plt.figure(figsize=(6,4))
# Replace 'study_hours' and 'score' with the actual column names
# from your CSV file for study hours and student scores respectively.
# For example, if the column for study hours is named 'Hours Studied'
# and the column for score is named 'Total Score', then you would use:
# sns.scatterplot(x=df["Hours Studied"], y=df["Total Score"])
# The error is because 'study_hours' and 'score' columns are likely not present.
# Inspect the CSV file columns and replace with the correct column names
# Here, we assume the relevant columns are named 'math score' for score
# and there is no study hours column, so we skip this visualization
# sns.scatterplot(x=df["study_hours"], y=df["score"])
# Instead, let's visualize 'math score' against 'reading score'
sns.scatterplot(x=df["reading score"], y=df["math score"])
plt.xlabel("Reading Score")  # Updated x-axis label
plt.ylabel("Math Score")  # Updated y-axis label
plt.title("Reading Score vs. Math Score")  # Updated title
plt.show()
```



3. Box Plot: Scores by Gender

```
plt.figure(figsize=(6,4))
# Replace 'score' with 'math score', 'reading score', or 'writing score'
sns.boxplot(x=df["gender"], y=df["math score"])  # Changed to 'math score'
plt.xlabel("Gender")
plt.ylabel("Math Score")  # Updated y-axis label
plt.title("Math Score Distribution by Gender")  # Updated title
plt.show()
```

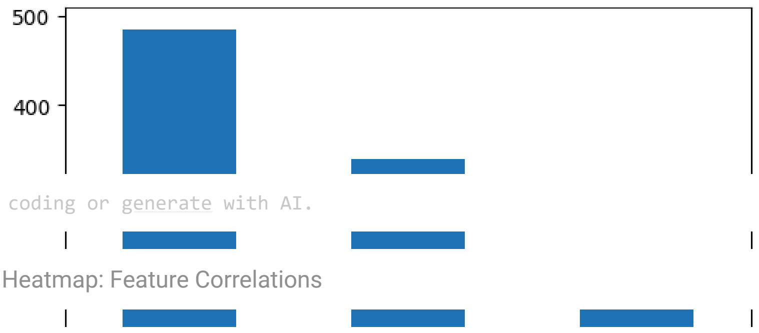


Start coding or generate with AI.

4. Bar Chart: Student Performance Categories

```
# Check the existing columns in the dataframe
print(df.columns)

# Create 'performance_category' based on existing scores
# Define bins for the score categories (adjust as needed)
bins = [0, 60, 80, 100]
labels = ["Low", "Medium", "High"]

# Create the 'performance_category' column
df["performance_category"] = pd.cut(df["math score"], bins=bins, labels=labels, include_lowest=True)

# Now, re-run the code for the bar chart
# Check the existing columns in the dataframe
print(df.columns)

# Create 'performance_category' based on existing scores
# Define bins for the score categories (adjust as needed)
bins = [0, 60, 80, 100]
labels = ["Low", "Medium", "High"]

# Create the 'performance_category' column
df["performance_category"] = pd.cut(df["math score"], bins=bins, labels=labels, include_lowest=True)

# Now, re-run the code for the bar chart
performance_counts = df["performance_category"].value_counts()
plt.figure(figsize=(6, 4))
performance_counts.plot(kind="bar")
```

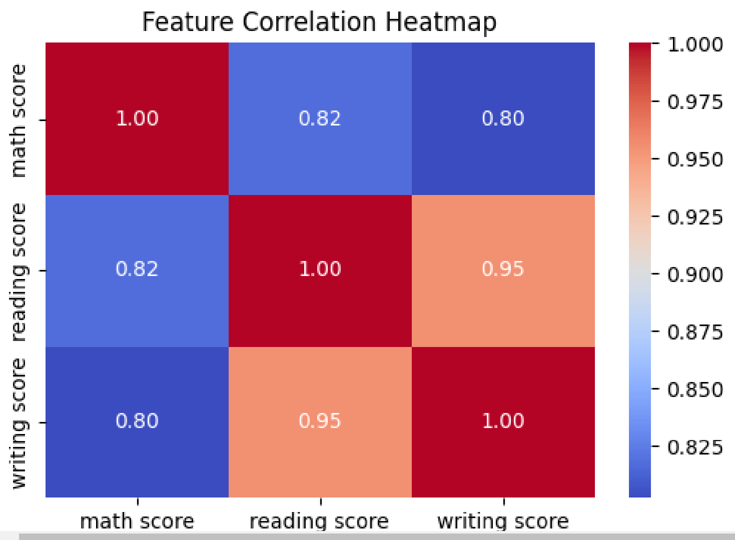
```
Index(['gender', 'race/ethnicity', 'parental level of education', 'lunch',
       'test preparation course', 'math score', 'reading score',
       'writing score'],
      dtype='object')
Index(['gender', 'race/ethnicity', 'parental level of education', 'lunch',
       'test preparation course', 'math score', 'reading score',
       'writing score', 'performance_category'],
      dtype='object')
<Axes: xlabel='performance_category'>
```



Start coding or generate with AI.

5. Heatmap: Feature Correlations

```
plt.figure(figsize=(6,4))
# Select only numeric features for correlation
numeric_df = df.select_dtypes(include=['number'])
sns.heatmap(numeric_df.corr(), annot=True, cmap="coolwarm", fmt=".2f")
plt.title("Feature Correlation Heatmap")
plt.show()
```



Start coding or generate with AI.

Final Step: Save and Upload to GitHub

```
# Save results
df.to_csv("processed_student_performance.csv", index=False)
```