```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from google.colab import files
import io

plt.style.use('seaborn-v0_8-whitegrid')
sns.set_palette("viridis")
plt.rcParams['figure.figsize'] = [12, 8]
plt.rcParams['font.size'] = 12

"""
# Upload your data files
uploaded = files.upload(/content/Student_performance_data _.xlsx)

# VISUALIZATION 1: Impact of Study Time on Academic Performance
fig, ax1 = plt.subplots(figsize=(14, 8))

# Bar chart for student count
ax1.bar(study_time_data['study_time_category'], study_time_data['student_count'],
        alpha=0.7, color='lightblue', label='Student Count')
ax1.set_xlabel('Weekly Study Time', fontsize=14)
ax1.set_ylabel('Number of Students', fontsize=14, color='navy')
ax1.tick_params(axis='y', labelcolor='navy')

# Add student count and percentage labels
for i, (count, avg) in enumerate(zip(study_time_data['student_count'], study_time_data['average_gpa'])):
    percentage = count / total_students * 100
    ax1.text(i, count + 20, f"{count:,}",
             horizontalalignment='center', fontsize=11)
    ax1.text(i, count/2, f"{percentage:.1f}%",
             horizontalalignment='center', fontsize=12, color='white', fontweight='bold')

# Secondary y-axis for GPA
ax2 = ax1.twinx()
ax2.plot(study_time_data['study_time_category'], study_time_data['average_gpa'], 'o-',
         color='darkred', linewidth=3, markersize=10, label='Avg GPA')
ax2.fill_between(range(len(study_time_data)),
                 study_time_data['min_gpa'],
                 study_time_data['max_gpa'],
                 color='salmon', alpha=0.3, label='GPA Range')
ax2.set_ylabel('GPA', fontsize=14, color='darkred')
ax2.tick_params(axis='y', labelcolor='darkred')
ax2.set_ylim(0, 4.5)

# Add GPA values
for i, avg in enumerate(study_time_data['average_gpa']):
    ax2.text(i, avg + 0.2, f"{avg:.2f}",
             horizontalalignment='center', fontsize=12, color='darkred')

# Add legend
lines_1, labels_1 = ax1.get_legend_handles_labels()
lines_2, labels_2 = ax2.get_legend_handles_labels()
ax2.legend(lines_1 + lines_2, labels_1 + labels_2, loc='upper left')

plt.title('Impact of Study Time on Academic Performance', fontsize=16)
plt.grid(False)
plt.tight_layout()
plt.savefig('study_time_impact.png', dpi=300, bbox_inches='tight')
plt.show()

# VISUALIZATION 2: Relationship Between Absences and GPA
plt.figure(figsize=(14, 8))
bars = plt.bar(absence_data['absence_category'], absence_data['average_gpa'],
               color=sns.color_palette("RdYlGn_r", len(absence_data)))
plt.title('Relationship Between Absences and GPA', fontsize=16)
plt.xlabel('Number of Absences', fontsize=14)
plt.ylabel('Average GPA', fontsize=14)
plt.ylim(0, 4.0)
plt.grid(axis='y', linestyle='--', alpha=0.7)

# Add GPA values on top of bars
for i, v in enumerate(absence_data['average_gpa']):
    plt.text(i, v + 0.1, f"{v:.2f}", ha='center', fontsize=12)
```

```
# Add student count below x-axis
for i, count in enumerate(absence_data['student_count']):
    plt.text(i, -0.2, f"n={count}", ha='center', fontsize=10)

plt.tight_layout()
plt.savefig('absences_vs_gpa.png', dpi=300, bbox_inches='tight')
plt.show()

# VISUALIZATION 3: Effect of Tutoring and Parental Support
# Prepare data for grouped bar chart
support_levels = tutoring_support_data['parental_support'].unique()
tutoring_statuses = tutoring_support_data['tutoring_status'].unique()

# Define the desired order for parental support
support_order = ['Very High', 'High', 'Moderate', 'Low', 'None']

# Create a pivot for plotting
pivot_data = tutoring_support_data.pivot_table(
    index='parental_support',
    columns='tutoring_status',
    values='average_gpa'
).reindex(support_order)

# Plot as grouped bar chart
plt.figure(figsize=(14, 8))
pivot_data.plot(kind='bar', figsize=(14, 8), width=0.8)
plt.title('Effect of Tutoring and Parental Support on GPA', fontsize=16)
plt.xlabel('Parental Support Level', fontsize=14)
plt.ylabel('Average GPA', fontsize=14)
plt.ylim(0, 4.0)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.legend(title='Tutoring Status', fontsize=12)


for container in plt.gca().containers:
    plt.bar_label(container, fmt='%.2f', fontsize=10)

plt.tight_layout()
plt.savefig('tutoring_support_effect.png', dpi=300, bbox_inches='tight')
plt.show()
```

Analyze files with Gemini