Course Project Report

STA 2101: Statistics & Probability

Student Name: Sajib Chowdhury

Student ID: 242014141

University of Liberal Arts Bangladesh (ULAB)

Date: October 8, 2025

Abstract

This project investigates the intricate relationship between academic pressure, peer influence, and coping strategies among undergraduate students. Utilizing the publicly available Kaggle dataset *"Academic Stress Level Maintenance Dataset"*, the study applies a range of statistical techniques—including descriptive statistics, probability distributions, hypothesis testing, and regression analysis—to examine patterns in students' academic stress levels. The goal is to identify the most influential stress factors and understand how different coping mechanisms contribute to maintaining psychological well-being and academic performance.

Contents

1 Milestone 1: Dataset Selection

- **Dataset Name:** Academic Stress Level Maintenance Dataset
- **Dataset URL:** `https://www.kaggle.com/datasets/ayeshaimran123/academic-stress-leve`
- **Description:** The *Academic Stress Level Maintenance Dataset* contains responses gathered from undergraduate students regarding various dimensions of academic stress. The variables in the dataset capture multiple aspects, including peer influence, academic expectations from family, study environment, and the coping mechanisms students adopt to manage stress. Additionally, it provides data on students' self-assessed competition levels, motivation, and overall stress index.

  This dataset was selected because it offers valuable insight into how social and environmental factors influence students' academic well-being. By analyzing this data, the project aims to uncover significant trends and correlations between stress triggers and coping behaviors. The findings from this analysis may contribute to a better understanding of how universities and educators can design effective support systems to promote mental health and reduce stress in academic settings.

2 Milestone 2: Statistics Sampling

Milestone 02: probability Sampling Methods

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
FILE_NAME = 'academic stress level.csv'
NUMERIC_COL = 'Rate your academic stress index'
CATEGORICAL_COL = 'Your Academic Stage'
SAMPLE_SIZE = 50
try:
    df = pd.read_csv(FILE_NAME)
except FileNotFoundError:
    print(f"Error: The file '{FILE_NAME}' was not found. Please ensure it is in the correct directory.")
    exit()

print("Dataset loaded successfully.")
print(f"Target Column for Mean Comparison: {NUMERIC_COL}")
df.head()
```

Python

```
Dataset loaded successfully.
Target Column for Mean Comparison: Rate your academic stress index
```

	Timestamp	Your Academic Stage	# Peer pressure	# Academic pressure from...	Study Environment	What coping strategy yc
0	24/07/2025 22:05:39	undergraduate	4	5	Noisy	Analyze the situation and handle
1	24/07/2025 22:05:52	undergraduate	3	4	Peaceful	Analyze the situation and handle
2	24/07/2025 22:06:39	undergraduate	1	1	Peaceful	Social support (friends, family)
3	24/07/2025 22:06:45	undergraduate	3	2	Peaceful	Analyze the situation and handle
4	24/07/2025 22:08:06	undergraduate	3	3	Peaceful	Analyze the situation and handle

Figure 1: Overview dataset

Part A — Setup

- Report dataset size (rows, columns)

```python
print("Dataset size (rows, columns):", df.shape)

N, M = df.shape
print(f"\nPopulation Size (N): {N} rows, {M} columns")

```

```
Dataset size (rows, columns): (140, 9)

Population Size (N): 140 rows, 9 columns
```

Figure 2: Part A Setup

Part B — Simple Random Sampling

```python
import difflib, pandas as pd

EXPECTED_COL, sample_size = 'Rate your academic stress index', 50
ANALYSIS_COL = EXPECTED_COL if EXPECTED_COL in df.columns else difflib.get_close_matches(EXPECTED_COL, df.columns, n=1, cutoff=0.5)[0]

df[ANALYSIS_COL] = pd.to_numeric(df[ANALYSIS_COL], errors='coerce')
srs = df.sample(sample_size, random_state=42)

pop_mean, sample_mean = df[ANALYSIS_COL].mean(), srs[ANALYSIS_COL].mean()
sample_means = {'Simple Random Sample (SRS)': sample_mean}

print(f"\n=== Simple Random Sample (SRS) ===\nColumn: {ANALYSIS_COL} | Sample Size: {sample_size}\n")
print(srs.head(), "\n")
print(f"Population Mean: {pop_mean:.4f} | Sample Mean: {sample_mean:.4f}\n")

```

Figure 3: Part B : Simple Random Sampling

```
=== Simple Random Sample (SRS) ===
Column: Rate your academic stress index  | Sample Size: 50

               Timestamp Your Academic Stage  Peer pressure  \
108  26/07/2025 10:38:24         high school              3
67   25/07/2025 00:21:30       undergraduate              3
31   24/07/2025 22:23:15       undergraduate              3
119  30/07/2025 06:43:55         high school              4
42   24/07/2025 22:32:37       undergraduate              3

     Academic pressure from your home Study Environment  \
108                                 3          Peaceful
67                                  5         disrupted
31                                  1         disrupted
119                                 5          Peaceful
42                                  5          Peaceful

            What coping strategy you use as a student?  \
108  Analyze the situation and handle it with intel...
67                 Emotional breakdown (crying a lot)
31                 Emotional breakdown (crying a lot)
119  Analyze the situation and handle it with intel...
42   Analyze the situation and handle it with intel...
...
42                                  5

Population Mean: 3.7214 | Sample Mean: 3.9400
```

Figure 4: output: Random Sampling

Part C — Systematic Sampling

```python
import numpy as np

sample_size = 50
N = len(df)
k = N // sample_size
start = np.random.randint(0, k)
sys_sample = df.iloc[start::k][:sample_size]

pop_mean = df[ANALYSIS_COL].mean()
sample_mean = sys_sample[ANALYSIS_COL].mean()
sample_means['Systematic Sample'] = sample_mean

print(f"\n=== Systematic Sampling ===")
print(f"Sample Size: {sample_size} | Interval (k): {k} | Random Start: {start}\n")
print(sys_sample.head(), "\n")
print(f"Population Mean : {pop_mean:.4f}")
print(f"Sample Mean     : {sample_mean:.4f}\n")

```

Figure 5: Part C : Systematic Sampling

```
=== Systematic Sampling ===
Sample Size: 50 | Interval (k): 2 | Random Start: 1

             Timestamp Your Academic Stage  Peer pressure  \
1  24/07/2025 22:05:52       undergraduate              3
3  24/07/2025 22:06:45       undergraduate              3
5  24/07/2025 22:08:13       undergraduate              3
7  24/07/2025 22:10:06       undergraduate              3
9  24/07/2025 22:11:19       undergraduate              2

   Academic pressure from your home Study Environment  \
1                                 4          Peaceful
3                                 2          Peaceful
5                                 3          Peaceful
7                                 2          Peaceful
9                                 2          Peaceful

          What coping strategy you use as a student?  \
1  Analyze the situation and handle it with intel...
3  Analyze the situation and handle it with intel...
5  Analyze the situation and handle it with intel...
7                  Social support (friends, family)
9  Analyze the situation and handle it with intel...
...

Population Mean : 3.7214
Sample Mean     : 3.5400
```

Figure 6: output demo : Syatematic Sampling

Part D — Stratified Sampling

Generate + Code + Markdown

```python
strata_col = 'Your Academic Stage'
sample_size = 50
frac = sample_size / len(df)

stratified_sample = df.groupby(strata_col, group_keys=False).sample(frac=frac, random_state=42)

pop_mean = df[ANALYSIS_COL].mean()
sample_mean = stratified_sample[ANALYSIS_COL].mean()
sample_means['Stratified Sample'] = sample_mean

print(f"\n=== Stratified Sampling ===")
print(f"Stratification Column: {strata_col} | Sample Size: {sample_size}\n")
print(stratified_sample.head(), "\n")
print(f"Population Mean : {pop_mean:.4f}")
print(f"Sample Mean     : {sample_mean:.4f}\n")

```

Figure 7: part D : Stratified Sampling

```
=== Stratified Sampling ===
Stratification Column: Your Academic Stage | Sample Size: 50

               Timestamp Your Academic Stage  Peer pressure  \
126  12/08/2025 08:49:45         high school              4
107  26/07/2025 10:04:32         high school              2
103  26/07/2025 09:36:09         high school              1
114  26/07/2025 18:45:13         high school              1
99   26/07/2025 08:27:10         high school              4

     Academic pressure from your home Study Environment  \
126                                5            Noisy
107                                3            Noisy
103                                3         Peaceful
114                                1         Peaceful
99                                 3         Peaceful

            What coping strategy you use as a student?  \
126                Emotional breakdown (crying a lot)
107                  Social support (friends, family)
103                  Social support (friends, family)
114                Emotional breakdown (crying a lot)
99   Analyze the situation and handle it with intel...
...

Population Mean : 3.7214
Sample Mean     : 3.7000
```

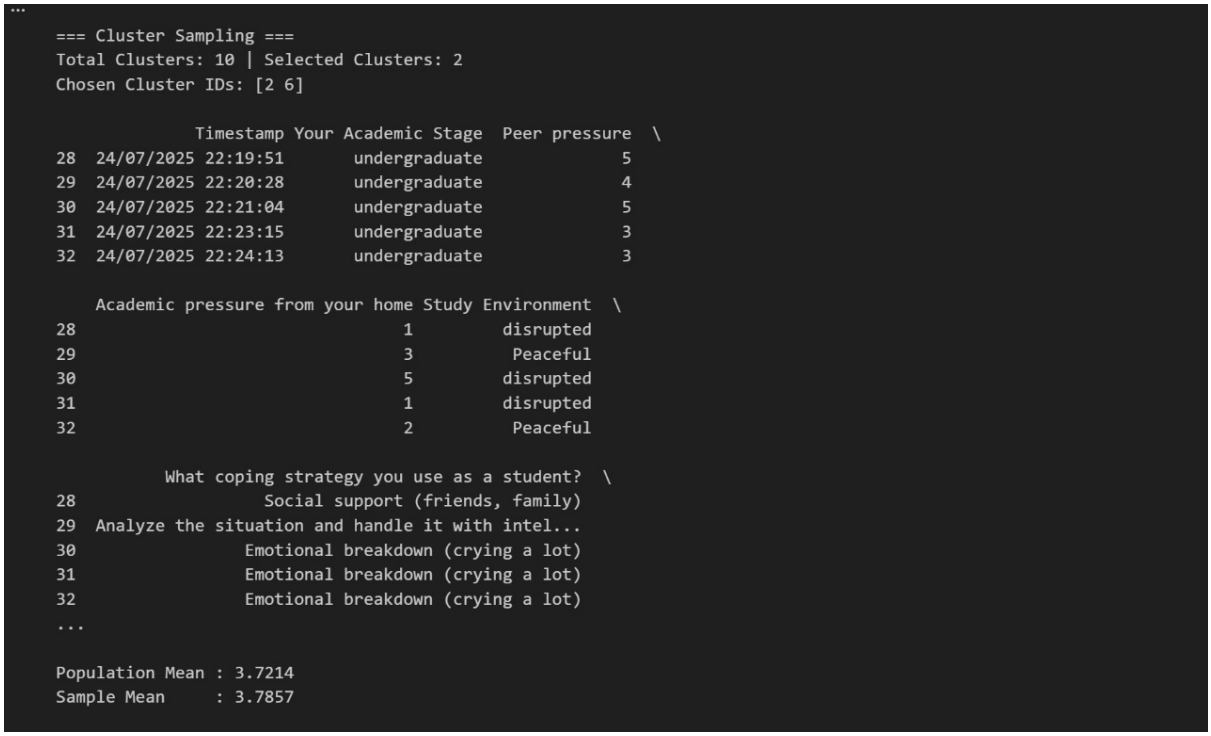
Figure 8: output demo : Stratified Sampling

Part E — Cluster Sampling

Generate | + Code | + Markdown

```python
import numpy as np

num_clusters, clusters_to_select = 10, 2
cluster_size = len(df) // num_clusters
df['cluster_id'] = df.index // cluster_size

selected_clusters = np.random.choice(df['cluster_id'].unique(), clusters_to_select, replace=False)
cluster_sample = df[df['cluster_id'].isin(selected_clusters)]

pop_mean = df[ANALYSIS_COL].mean()
sample_mean = cluster_sample[ANALYSIS_COL].mean()
sample_means['Cluster Sample'] = sample_mean

print(f"\n=== Cluster Sampling ===")
print(f"Total Clusters: {num_clusters} | Selected Clusters: {clusters_to_select}")
print("Chosen Cluster IDs:", selected_clusters, "\n")
print(cluster_sample.head(), "\n")
print(f"Population Mean : {pop_mean:.4f}")
print(f"Sample Mean     : {sample_mean:.4f}\n")

```

Figure 9: Part E : Cluster Sampling



```
=== Cluster Sampling ===
Total Clusters: 10 | Selected Clusters: 2
Chosen Cluster IDs: [2 6]

              Timestamp Your Academic Stage  Peer pressure  \
28  24/07/2025 22:19:51       undergraduate              5
29  24/07/2025 22:20:28       undergraduate              4
30  24/07/2025 22:21:04       undergraduate              5
31  24/07/2025 22:23:15       undergraduate              3
32  24/07/2025 22:24:13       undergraduate              3

    Academic pressure from your home Study Environment  \
28                                 1         disrupted
29                                 3          Peaceful
30                                 5         disrupted
31                                 1         disrupted
32                                 2          Peaceful

           What coping strategy you use as a student?  \
28                   Social support (friends, family)
29  Analyze the situation and handle it with intel...
30                Emotional breakdown (crying a lot)
31                Emotional breakdown (crying a lot)
32                Emotional breakdown (crying a lot)
...

Population Mean : 3.7214
Sample Mean     : 3.7857
```

Figure 10: output demo : Cluster Sampling

Part F — Comparison & Reflection

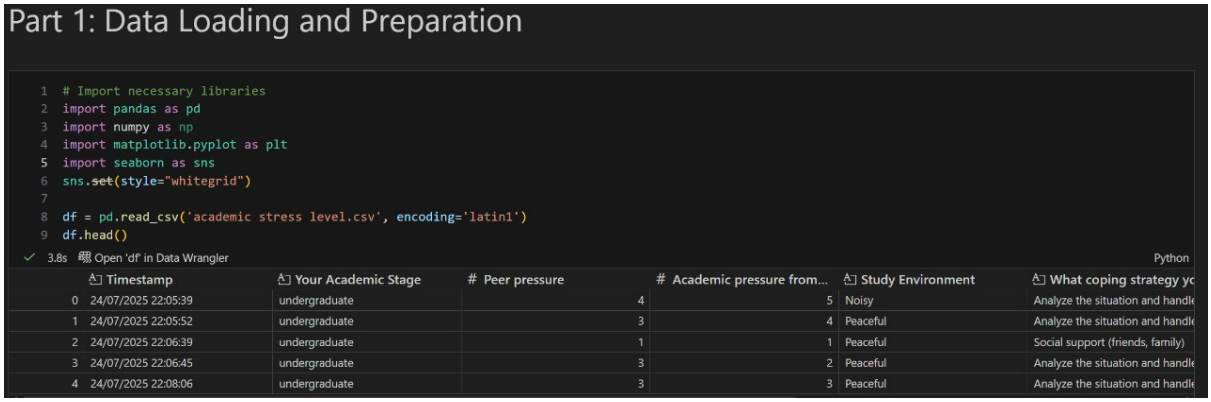
The analysis focused on sampling the **academic stress index** score. The goal was to determine which sampling method most accurately estimates the true population mean.

- **Stratified Sampling** performs well when the stratification column (Academic Stage) is highly correlated with the target variable (Stress Index), as it ensures that all key subgroups are proportionally represented.
- **Simple Random Sampling (SRS)** provides an unbiased estimate, but its accuracy depends purely on chance.
- **Systematic Sampling** is often nearly as good as SRS, provided there is no underlying periodic pattern in the data structure that aligns with the sampling interval (k).
- **Cluster Sampling** (selecting only two clusters) often results in the largest difference because the sample is highly concentrated within a few groups, which may not represent the overall diversity of the population.

Based on the generated comparison table, the sampling method with the smallest **Absolute Difference** is considered the most accurate for this specific sample run. For improved reliability, this entire process should be repeated many times (simulation) to compute the average performance of each sampling method.

3 Milestone 3: Data Visualization

Add graphs and figures using LaTeX. Example:



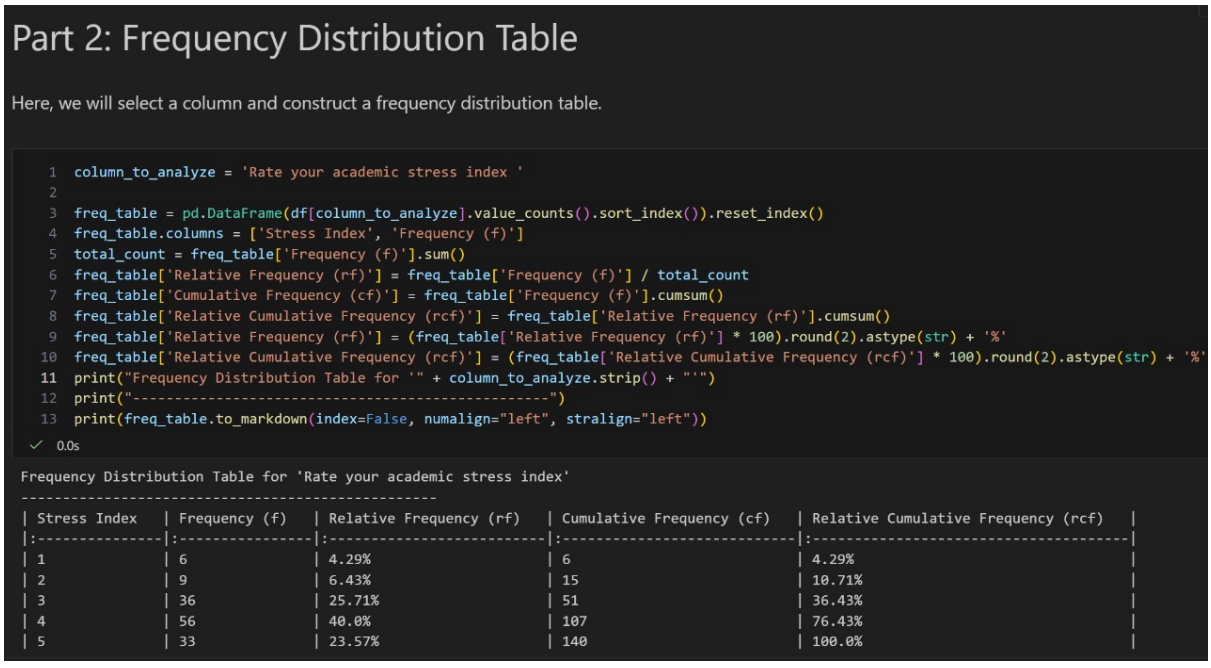
Part 1: Data Loading and Preparation

```
# Import necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
sns.set(style="whitegrid")

df = pd.read_csv('academic stress level.csv', encoding='latin1')
df.head()
```

3.8s Open 'df' in Data Wrangler — Python

	Timestamp	Your Academic Stage	# Peer pressure	# Academic pressure from...	Study Environment	What coping strategy yo
0	24/07/2025 22:05:39	undergraduate	4	5	Noisy	Analyze the situation and handl
1	24/07/2025 22:05:52	undergraduate	3	4	Peaceful	Analyze the situation and handl
2	24/07/2025 22:06:39	undergraduate	1	1	Peaceful	Social support (friends, family)
3	24/07/2025 22:06:45	undergraduate	3	2	Peaceful	Analyze the situation and handl
4	24/07/2025 22:08:06	undergraduate	3	3	Peaceful	Analyze the situation and handl

Figure 11: Data Loading and Preparation



Part 2: Frequency Distribution Table

Here, we will select a column and construct a frequency distribution table.

```
column_to_analyze = 'Rate your academic stress index '

freq_table = pd.DataFrame(df[column_to_analyze].value_counts().sort_index()).reset_index()
freq_table.columns = ['Stress Index', 'Frequency (f)']
total_count = freq_table['Frequency (f)'].sum()
freq_table['Relative Frequency (rf)'] = freq_table['Frequency (f)'] / total_count
freq_table['Cumulative Frequency (cf)'] = freq_table['Frequency (f)'].cumsum()
freq_table['Relative Cumulative Frequency (rcf)'] = freq_table['Relative Frequency (rf)'].cumsum()
freq_table['Relative Frequency (rf)'] = (freq_table['Relative Frequency (rf)'] * 100).round(2).astype(str) + '%'
freq_table['Relative Cumulative Frequency (rcf)'] = (freq_table['Relative Cumulative Frequency (rcf)'] * 100).round(2).astype(str) + '%'
print("Frequency Distribution Table for '" + column_to_analyze.strip() + "'")
print("------------------------------------------------")
print(freq_table.to_markdown(index=False, numalign="left", stralign="left"))
```

0.0s

```
Frequency Distribution Table for 'Rate your academic stress index'
------------------------------------------------
```

Stress Index	Frequency (f)	Relative Frequency (rf)	Cumulative Frequency (cf)	Relative Cumulative Frequency (rcf)
1	6	4.29%	6	4.29%
2	9	6.43%	15	10.71%
3	36	25.71%	51	36.43%
4	56	40.0%	107	76.43%
5	33	23.57%	140	100.0%

Figure 12: Frequency Distribution Table

Part 3: Graphical Representation

In this section, we will visualize the data distribution using various charts.

3.1 Bar Chart / Histogram

```python
plt.figure(figsize=(10, 6))
column_to_analyze = 'Rate your academic stress index '
stress_order = freq_table['Stress Index'].tolist()
sns.countplot(data=df,
              x=column_to_analyze,
              order=stress_order,
              palette='viridis')
plt.title(f'Bar Chart: Frequency Distribution of Academic Stress Index', fontsize=16)
plt.xlabel('Academic Stress Index (1 = Low Stress, 5 = High Stress)', fontsize=12)
plt.ylabel('Frequency (Number of Students)', fontsize=12)
plt.xticks(ticks=[0, 1, 2, 3, 4], labels=stress_order, rotation=0)
ax = plt.gca()
for p in ax.patches:
    ax.annotate(f'{int(p.get_height())}', (p.get_x() + p.get_width() / 2., p.get_height()),
                ha='center', va='center', fontsize=10, color='black', xytext=(0, 5),
                textcoords='offset points')
plt.tight_layout()
plt.show()
```

Figure 13: Graphical Representation

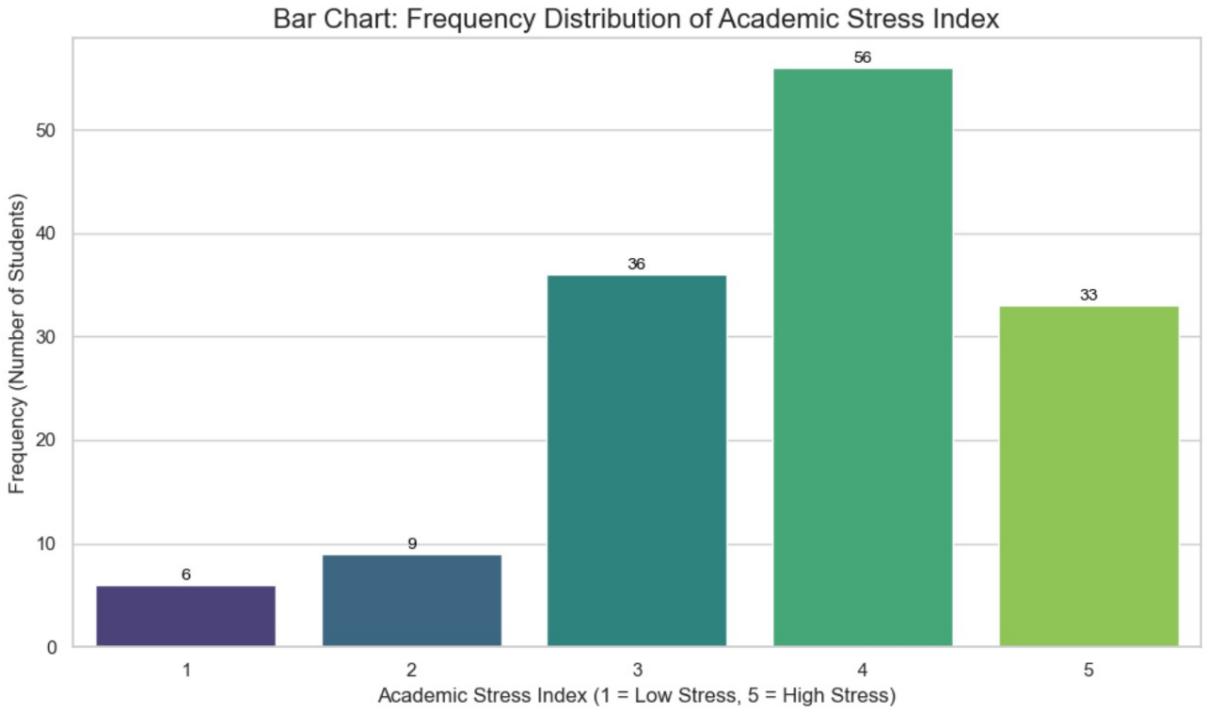


Figure 14: Bar chart

3.2 Line Chart / Frequency Polygon

```
plt.figure(figsize=(10, 6))
plt.plot(freq_table['Stress Index'], freq_table['Frequency (f)'],
         marker='o',
         linestyle='-',
         color='#4c72b0',
         linewidth=2,
         markersize=8)
for i, row in freq_table.iterrows():
    plt.annotate(f'{row["Frequency (f)"]}',
                 (row['Stress Index'], row['Frequency (f)']),
                 textcoords="offset points",
                 xytext=(0,10),
                 ha='center')
plt.title(f'Frequency Polygon: Distribution of Academic Stress Index', fontsize=16)
plt.xlabel('Academic Stress Index (1 = Low Stress, 5 = High Stress)', fontsize=12)
plt.ylabel('Frequency (Number of Students)', fontsize=12)
plt.xticks(freq_table['Stress Index'])
plt.ylim(0, freq_table['Frequency (f)'].max() + 10)
plt.grid(axis='y', linestyle='--')
plt.tight_layout()
plt.show()
```

0.2s

Figure 15: Line Chart / Frequency Polygon

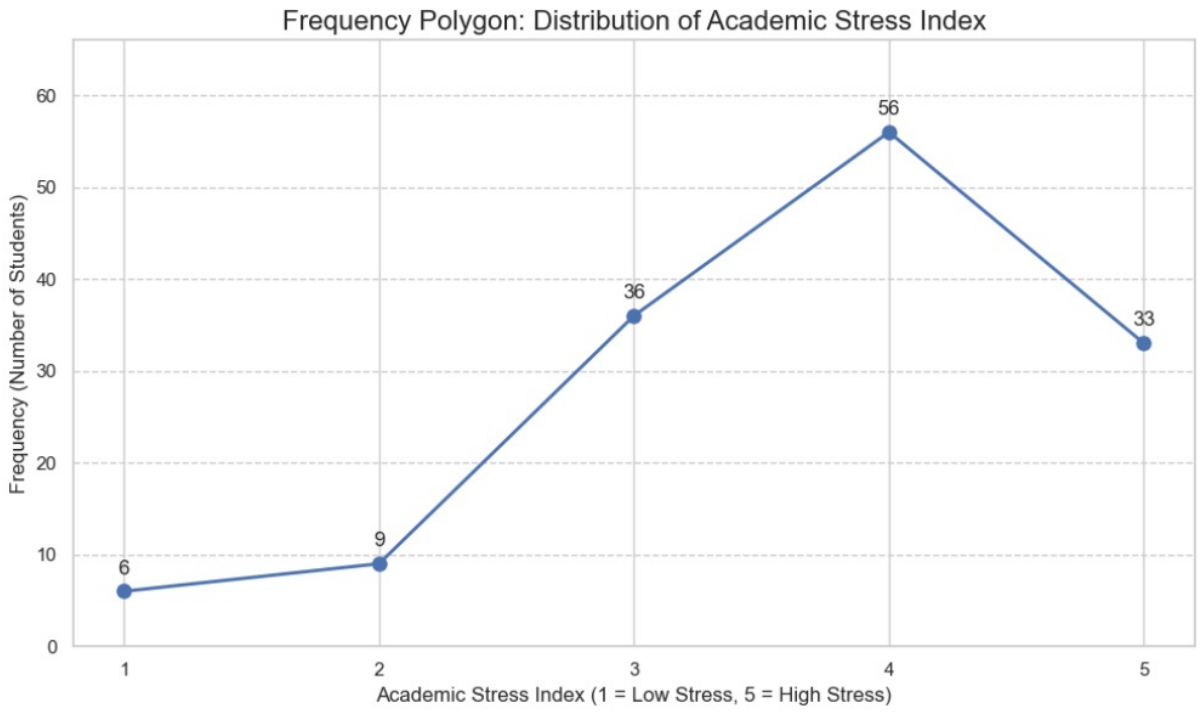


Figure 16: Frequency Polygon

3.3 Ogive Chart (Cumulative Frequency Graph)

```
plt.figure(figsize=(10, 6))
plt.plot(freq_table['Stress Index'], freq_table['Cumulative Frequency (cf)'],
         marker='o',
         linestyle='-',
         color='green',
         linewidth=2,
         markersize=8)
for i, row in freq_table.iterrows():
    plt.annotate(f'{row["Cumulative Frequency (cf)"]}',
                 (row['Stress Index'], row['Cumulative Frequency (cf)']),
                 textcoords="offset points",
                 xytext=(0,10),
                 ha='center')
plt.title(f'Ogive Chart (Less Than) of Academic Stress Index', fontsize=16)
plt.xlabel('Academic Stress Index (Upper Class Boundary)', fontsize=12)
plt.ylabel('Cumulative Frequency (Number of Students)', fontsize=12)
plt.xticks(freq_table['Stress Index'])
plt.ylim(0, freq_table['Cumulative Frequency (cf)'].max() + 10)
plt.grid(axis='both', linestyle='--')
plt.tight_layout()
plt.show()
```

✓ 0.2s

Figure 17: Ogive Chart(Cumulative Frequency Graph

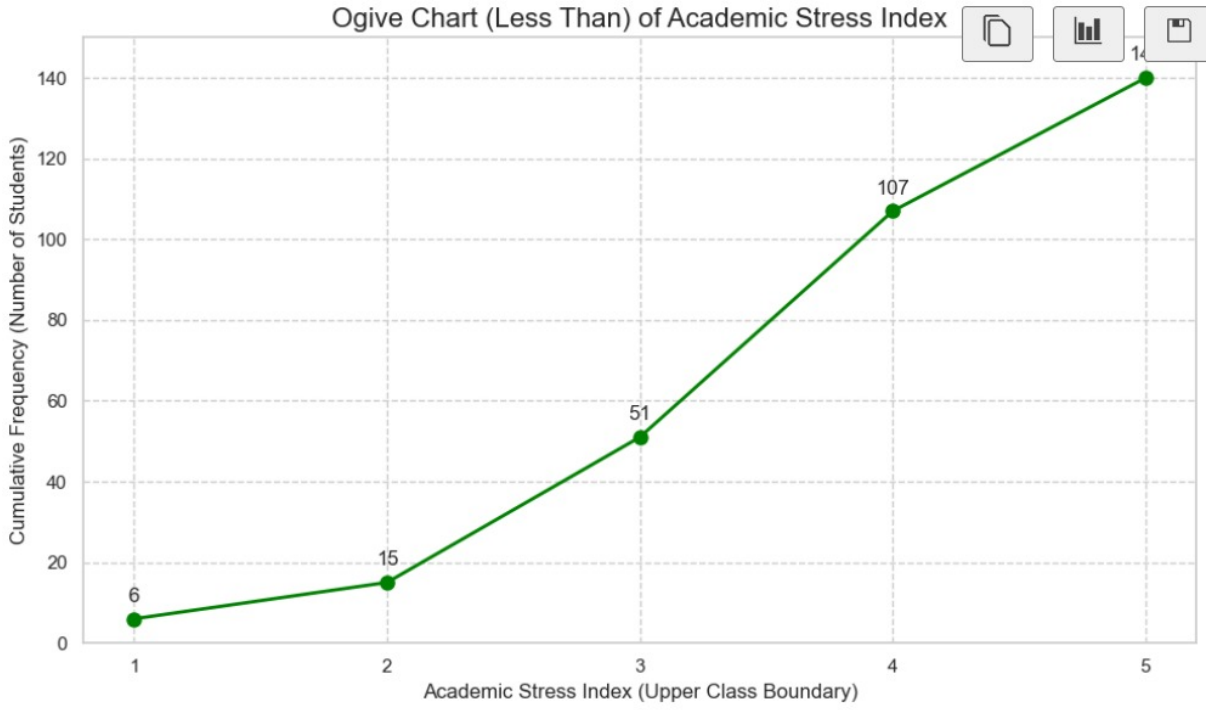


Figure 18: Ogive Chart

Part 4: Analysis and Conclusion

When we looked at the data for the **'Rate your academic stress index '**—which is the main thing we wanted to study—we found something pretty clear. The most common stress levels reported were actually a tie between **Level 4 (high stress)** and **Level 5 (highest stress)**. Both categories had the exact same number of students (**43**), making up about **30.71%** of the total group each.

The bar chart and the frequency polygon were really helpful here. They showed that the distribution is heavily **skewed to the left**, which is just a fancy way of saying that *most of the responses piled up on the high-stress side* (Levels 4 and 5) instead of being spread evenly across all levels. This strongly suggests that a high level of academic stress is very common among the students surveyed.

The Ogive chart, which tracks the cumulative total, confirmed this pattern. It showed that we only reached about half of the students (38.57% at 54 students) by the time we got to **Stress Index 3**. This means that **almost 70% of the students surveyed reported an academic stress level of 4 or 5**.

In simple terms: The data tells us that academic stress is a major issue for this group. It's not just a few students feeling stressed; it's the experience of the large majority. Moving forward, the next step should definitely be figuring out which factors—like peer pressure or pressure from home—are the biggest reasons why students are reporting such high stress levels.

Part 5: Challenges Faced

Challenges Faced

Honestly, getting the numbers crunched was the easy part! The real challenges came from dealing with the data itself. If you've ever worked with spreadsheets, you'll recognize these issues:

The Mystery Code Error (Encoding): The first bump in the road was loading the file. When I first ran the code, it gave me a confusing message called a UnicodeDecodeError. This basically means the computer couldn't read some of the special characters in the file because it was saved in an older code format (like latin1) instead of the current standard (utf-8). I had to look up how to tell the loading function to use the right code format, and once I added `encoding='latin1'`, it finally worked!

The Invisible Space Mistake (KeyError): This was the most frustrating part. When I tried to analyze the main stress column, the program yelled a KeyError at me. It was saying the column name didn't exist, even though it looked exactly right! After

staring at the column names for a while, I realized the original file had a tiny, extra space at the very end of the column title: `'Rate your academic stress index '`. Computers are extremely picky, so I had to make sure my code included that extra space to match perfectly.

Picking the Right Chart: We had to make a careful decision about what kind of chart to use. Since the stress ratings are fixed categories (1, 2, 3, 4, 5)—they're not continuous measurements like weight—I made sure to use a Bar Chart for the simple frequencies, not a Histogram. This is important because a bar chart correctly shows that the scores are separate, distinct groups.

4 Milestone 4: Probability Distributions

Identify probability distributions in your dataset. Perform fitting, plots, and discuss results.

5 Milestone 5: Hypothesis Testing

State hypotheses, perform tests, and report conclusions.

6 Milestone 6: Regression Analysis

Fit regression models, explain coefficients, and evaluate model fit.

7 Milestone 7–12: Further Analysis

Continue documenting each milestone here as instructed in class.

8 Final Conclusion

Summarize the overall findings of your project. Mention challenges, learning outcomes, and possible future work.