MAHATMA EDUCATION SOCIETY'S

PILLAI COLLEGE OF ARTS, COMMERCE AND SCIENCE

(Autonomous)

NEW PANVEL PROJECT REPORT ON

"Student Depression Dataset"

IN PARTIAL FULFILLMENT OF BACHELOR OF

Computer Science

SEMESTER VI – 2024-25

PROJECT GUIDE: Sanjana Bhangale

SUBMITTED BY: Mehta Amitkumar Ravindra

ROLL NO: 9139

Evaluation sheet for continuous assessment with rubrics

Class: Computer Science

Subject: Machine Learning

Details about the continuous Assessment 2/Project work

Name of the Student: Mehta Amitkumar Ravindra

Roll Number: 9139

Class / Division: TYCS-A

Name of Evaluator: Sanjana Bhangale

Project name: Student Depression Dataset

Please circle appropriate score

Grading Criteria	Fair	Good	Excellent	Total
Introduction/ Description of the Case	1	2	3	/3
SWOT Analysis of the company used for case analysis pertaining to the case (**strength** of CA2 topic: e.g main important feature of CA1, **Weakness:** limitations of the project, **Opportunities:** in carrier in the future, **Threat:** obstacles that can cause failure to project CA2	3	4	5	/5
Learnings from the case	2	3	4	/4
Delivery/presentation skills	1	2	3	/3
Total				/15

- **Inform the class the rubric format and the method of evaluation.**

Co-ordinator,

Shubhangi Pawar

Student Depression Prediction using Machine Learning

1. Introduction

The mental health of students is a growing concern in the education system. Depression among students can impact academic performance, social behavior, and overall well-being. This project aims to leverage machine learning techniques to predict depression among students based on various factors such as academic pressure, work stress, sleep patterns, and dietary habits. By building a predictive model, we aim to facilitate early intervention and support for students at risk of depression.

2. Problem Statement

The goal of this project is to analyze various factors influencing student depression and build a machine learning model that predicts whether a student is experiencing depression based on given features.

3. Dataset Description

The dataset contains multiple features related to students' demographics, academic performance, lifestyle habits, and mental health indicators. It includes:

- Demographics: Age, Gender, City, Degree
- Academic & Work Factors: CGPA, Academic Pressure, Work Pressure, Study Satisfaction, Job Satisfaction
- Lifestyle Factors: Sleep Duration, Dietary Habits, Work/Study Hours, Financial Stress
- Mental Health Indicators: Family History of Mental Illness, Suicidal Thoughts, Depression (Target Variable)

The dataset was preprocessed to handle missing values, outliers, and categorical data before applying machine learning models.

4. Methodology

- ✓ Step 1: Business Understanding.
- ✓ Step 2: Data Loading
- ✓ Step 3: Data Preparation.
- ✓ Step 4: Exploratory Data Analysis (EDA)
- ✓ Step 5: Analytical & Visualization Questions
- ✓ Step 6: Model Building
- ✓ Step 7: Model Evaluation

5. Model Building & Evaluation

The Random Forest model was trained on the dataset with the Depression column as the target variable. Performance evaluation was conducted using:

- Accuracy: Measures the percentage of correct predictions.
- Precision & Recall: Analyzed the model's effectiveness in identifying students with depression.
- Confusion Matrix: Provided insight into false positives and false negatives.

Load the Dataset

```
[62] import numpy as np
     import pandas as pd
     import matplotlib.pyplot as plt
     import seaborn as sns
     from scipy.stats import ttest_ind, ttest_ind_from_stats
     from scipy.special import stdtr
     from mpl_toolkits.mplot3d import Axes3D
     from scipy.stats import pointbiserialr, ttest_ind
     from sklearn.model_selection import train_test_split
     from sklearn.ensemble import RandomForestClassifier
     from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
     df = pd.read_csv('/content/Student Depression Dataset (1).csv')
     df.head()
```

	id	Gender	Age	City	Profession	Academic Pressure	Work Pressure	CGPA	Study Satisfaction	Job Satisfaction	Sleep Duration	Dietary Habits	Degree	ever had suicidal thoughts ?	Work/S H
0	2	Male	33.0	Visakhapatnam	Student	5.0	0.0	8.97	2.0	0.0	5-6 hours	Healthy	B.Pharm	Yes	
1	8	Female	24.0	Bangalore	Student	2.0	0.0	5.90	5.0	0.0	5-6 hours	Moderate	BSc	No	
2	26	Male	31.0	Srinagar	Student	3.0	0.0	7.03	5.0	0.0	Less than 5 hours	Healthy	BA	No	
3	30	Female	28.0	Varanasi	Student	3.0	0.0	5.59	2.0	0.0	7-8 hours	Moderate	BCA	Yes	
4	32	Female	25.0	Jaipur	Student	4.0	0.0	8.13	3.0	0.0	5-6 hours	Moderate	M.Tech	Yes	

Data Understanding

```
df.shape
```

```
(27901, 18)
```

```python
unique_counts = df.nunique()
print("\nUnique Values Count:\n", unique_counts)
```

```
Unique Values Count:
 id                                       27901
Gender                                        2
Age                                          34
City                                         52
Profession                                   14
Academic Pressure                             6
Work Pressure                                 3
CGPA                                        332
Study Satisfaction                            6
Job Satisfaction                              5
Sleep Duration                                5
Dietary Habits                                4
Degree                                       28
Have you ever had suicidal thoughts ?         2
Work/Study Hours                             13
Financial Stress                              5
Family History of Mental Illness              2
Depression                                    2
dtype: int64
```

```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 27901 entries, 0 to 27900
Data columns (total 18 columns):
 #   Column                                 Non-Null Count  Dtype
---  ------                                 --------------  -----
 0   id                                     27901 non-null  int64
 1   Gender                                 27901 non-null  object
 2   Age                                    27901 non-null  float64
 3   City                                   27901 non-null  object
 4   Profession                             27901 non-null  object
 5   Academic Pressure                      27901 non-null  float64
 6   Work Pressure                          27901 non-null  float64
 7   CGPA                                   27901 non-null  float64
 8   Study Satisfaction                     27901 non-null  float64
 9   Job Satisfaction                       27901 non-null  float64
 10  Sleep Duration                         27901 non-null  object
 11  Dietary Habits                         27901 non-null  object
 12  Degree                                 27901 non-null  object
 13  Have you ever had suicidal thoughts ?  27901 non-null  object
 14  Work/Study Hours                       27901 non-null  float64
 15  Financial Stress                       27898 non-null  float64
 16  Family History of Mental Illness       27901 non-null  object
 17  Depression                             27901 non-null  int64
dtypes: float64(8), int64(2), object(8)
memory usage: 3.8+ MB
```

```
df.describe()
```

	id	Age	Academic Pressure	Work Pressure	CGPA	Study Satisfaction	Job Satisfaction	Work/Study Hours	Financial Stress	Depression
count	27901.000000	27901.000000	27901.000000	27901.000000	27901.000000	27901.000000	27901.000000	27901.000000	27898.000000	27901.000000
mean	70442.149421	25.822300	3.141214	0.000430	7.656104	2.943837	0.000681	7.156984	3.139867	0.585499
std	40641.175216	4.905687	1.381465	0.043992	1.470707	1.361148	0.044394	3.707642	1.437347	0.492645
min	2.000000	18.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000	0.000000
25%	35039.000000	21.000000	2.000000	0.000000	6.290000	2.000000	0.000000	4.000000	2.000000	0.000000
50%	70684.000000	25.000000	3.000000	0.000000	7.770000	3.000000	0.000000	8.000000	3.000000	1.000000
75%	105818.000000	30.000000	4.000000	0.000000	8.920000	4.000000	0.000000	10.000000	4.000000	1.000000
max	140699.000000	59.000000	5.000000	5.000000	10.000000	5.000000	4.000000	12.000000	5.000000	1.000000

```
df.dtypes
```

	0
id	int64
Gender	object
Age	float64
City	object
Profession	object
Academic Pressure	float64
Work Pressure	float64
CGPA	float64
Study Satisfaction	float64
Job Satisfaction	float64
Sleep Duration	object
Dietary Habits	object
Degree	object
Have you ever had suicidal thoughts ?	object
Work/Study Hours	float64
Financial Stress	float64
Family History of Mental Illness	object

```
df.count()
```

	0
id	27898
Gender	27898
Age	27898
City	27898
Profession	27898
Academic Pressure	27898
Work Pressure	27898
CGPA	27898
Study Satisfaction	27898
Job Satisfaction	27898
Sleep Duration	27898
Dietary Habits	27898
Degree	27898
Have you ever had suicidal thoughts ?	27898
Work/Study Hours	27898

Data processing and EDA

```
[68] df.dropna(inplace=True)
```

```
df.isnull().sum()
```

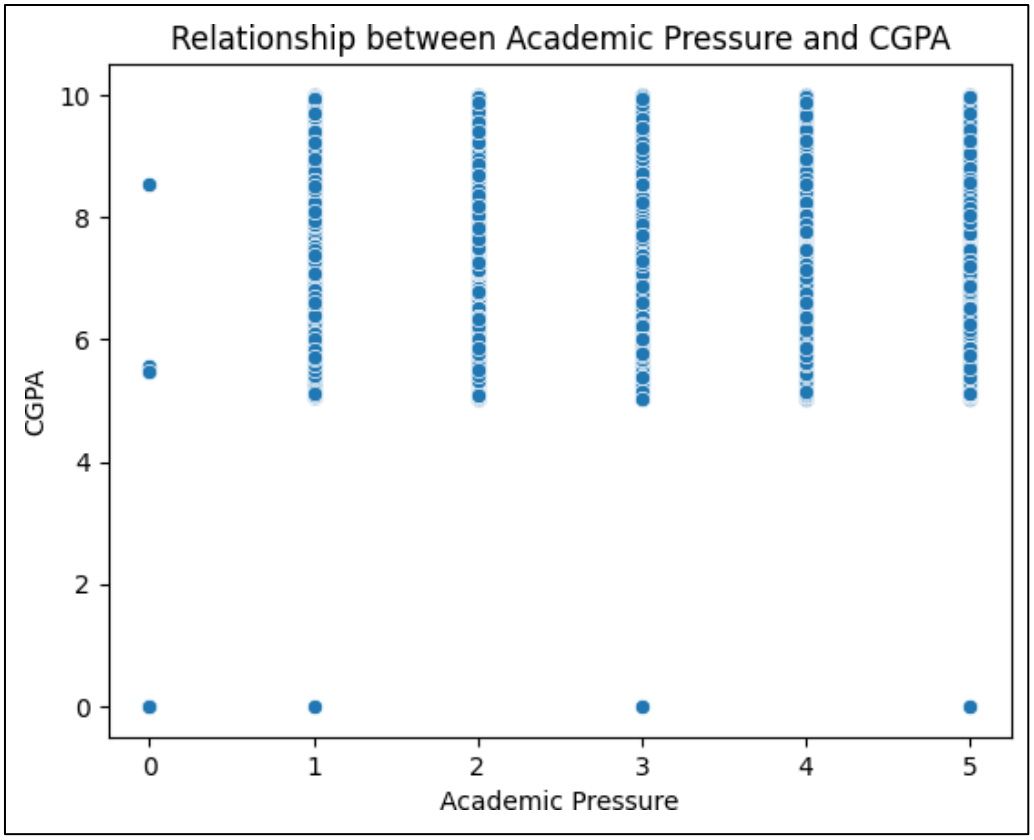
	0
id	0
Gender	0
Age	0
City	0
Profession	0
Academic Pressure	0
Work Pressure	0
CGPA	0
Study Satisfaction	0
Job Satisfaction	0
Sleep Duration	0
Dietary Habits	0
Degree	0
Have you ever had suicidal thoughts ?	0
Work/Study Hours	0

Analytical Questions:

1. What is the relationship between academic pressure and CGPA?

o Investigate if students with higher academic pressure tend to have higher or lower CGPA values.

```
# Correlation
correlation = df['Academic Pressure'].corr(df['CGPA'])
print("Correlation between Academic Pressure and CGPA:", correlation)

sns.scatterplot(x=df['Academic Pressure'], y=df['CGPA'])
plt.title('Relationship between Academic Pressure and CGPA')
plt.show()
```



2. Does sleep duration affect CGPA and study satisfaction?

o Analyze if there is a correlation between the sleep duration and both CGPA and study satisfaction.

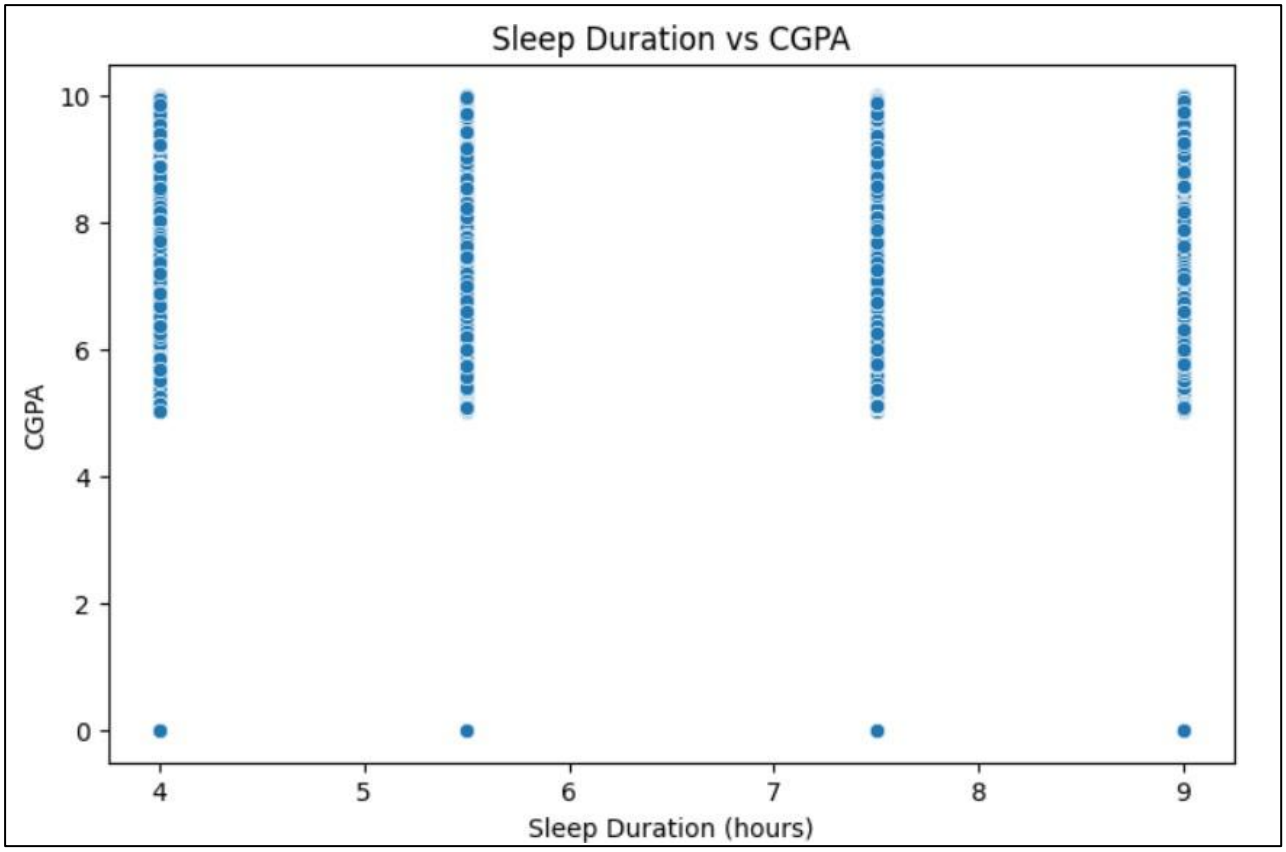
```
df = df[df['Sleep Duration'] != 'Others']
```

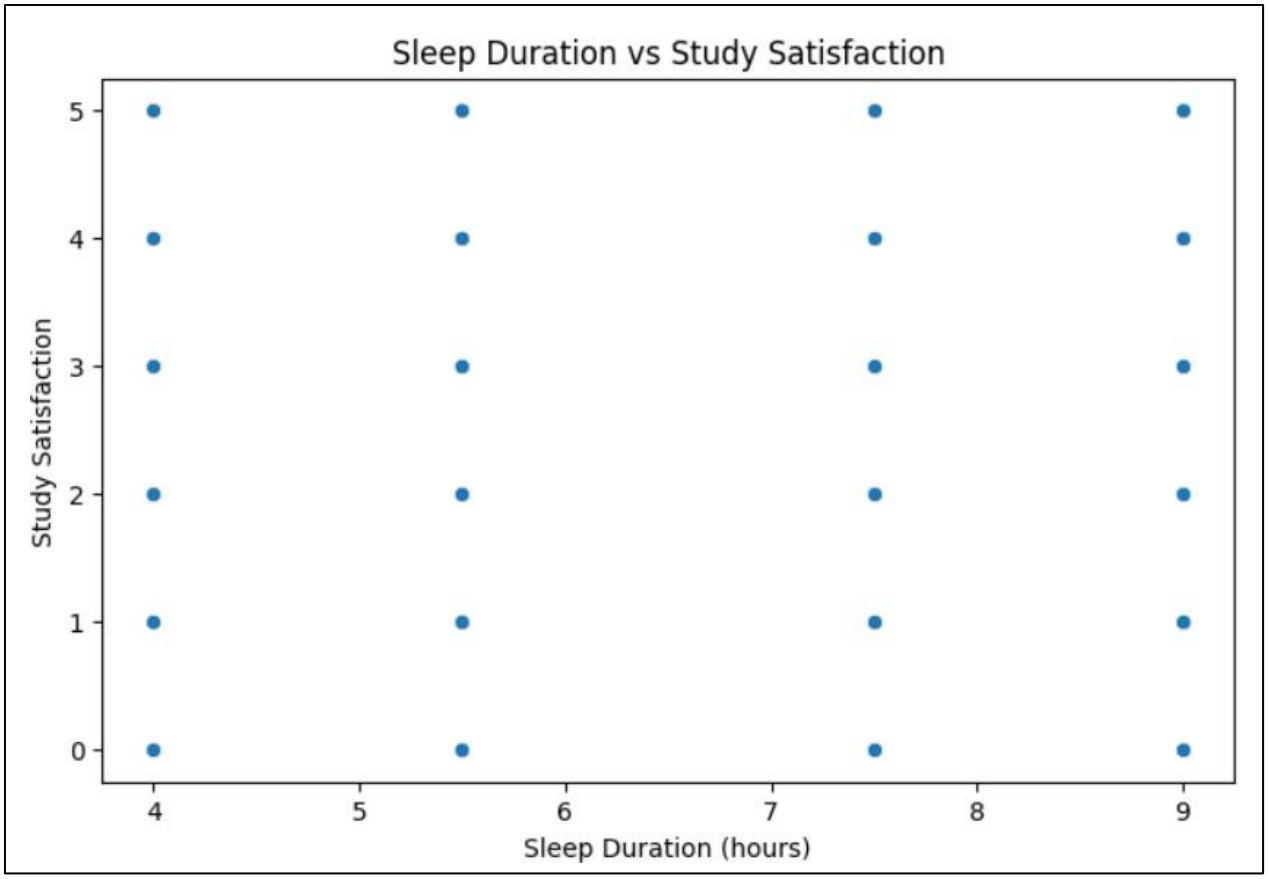
```
[81] print(df['Sleep Duration'].unique())
```

```
['5-6 hours' 'Less than 5 hours' '7-8 hours' 'More than 8 hours']
```

```
<ipython-input-75-dc5945b1d9d6>:4: FutureWarning: Downcasting behavior in `replace
  df.replace('More than 8 hours', 9, inplace=True)
Correlation between Sleep Duration and CGPA: -0.004884627074855316
Correlation between Sleep Duration and Study Satisfaction: 0.012095094907399535
```

```
df.replace('Less than 5 hours', 4, inplace=True)
df.replace('5-6 hours', 5.5, inplace=True)
df.replace('7-8 hours', 7.5, inplace=True)
df.replace('More than 8 hours', 9, inplace=True)

# Check correlation between Sleep Duration and CGPA
corr_sleep_cgpa = df['Sleep Duration'].corr(df['CGPA'])
print("Correlation between Sleep Duration and CGPA:", corr_sleep_cgpa)

# Check correlation between Sleep Duration and Study Satisfaction
corr_sleep_study_satisfaction = df['Sleep Duration'].corr(df['Study Satisfaction'])
print("Correlation between Sleep Duration and Study Satisfaction:", corr_sleep_study_satisfaction)

# Scatter plot: Sleep Duration vs CGPA
plt.figure(figsize=(8,5))
sns.scatterplot(x=df['Sleep Duration'], y=df['CGPA'])
plt.title('Sleep Duration vs CGPA')
plt.xlabel('Sleep Duration (hours)')
plt.ylabel('CGPA')
plt.show()

# Scatter plot: Sleep Duration vs Study Satisfaction
plt.figure(figsize=(8,5))
sns.scatterplot(x=df['Sleep Duration'], y=df['Study Satisfaction'])
plt.title('Sleep Duration vs Study Satisfaction')
plt.xlabel('Sleep Duration (hours)')
plt.ylabel('Study Satisfaction')
plt.show()
```



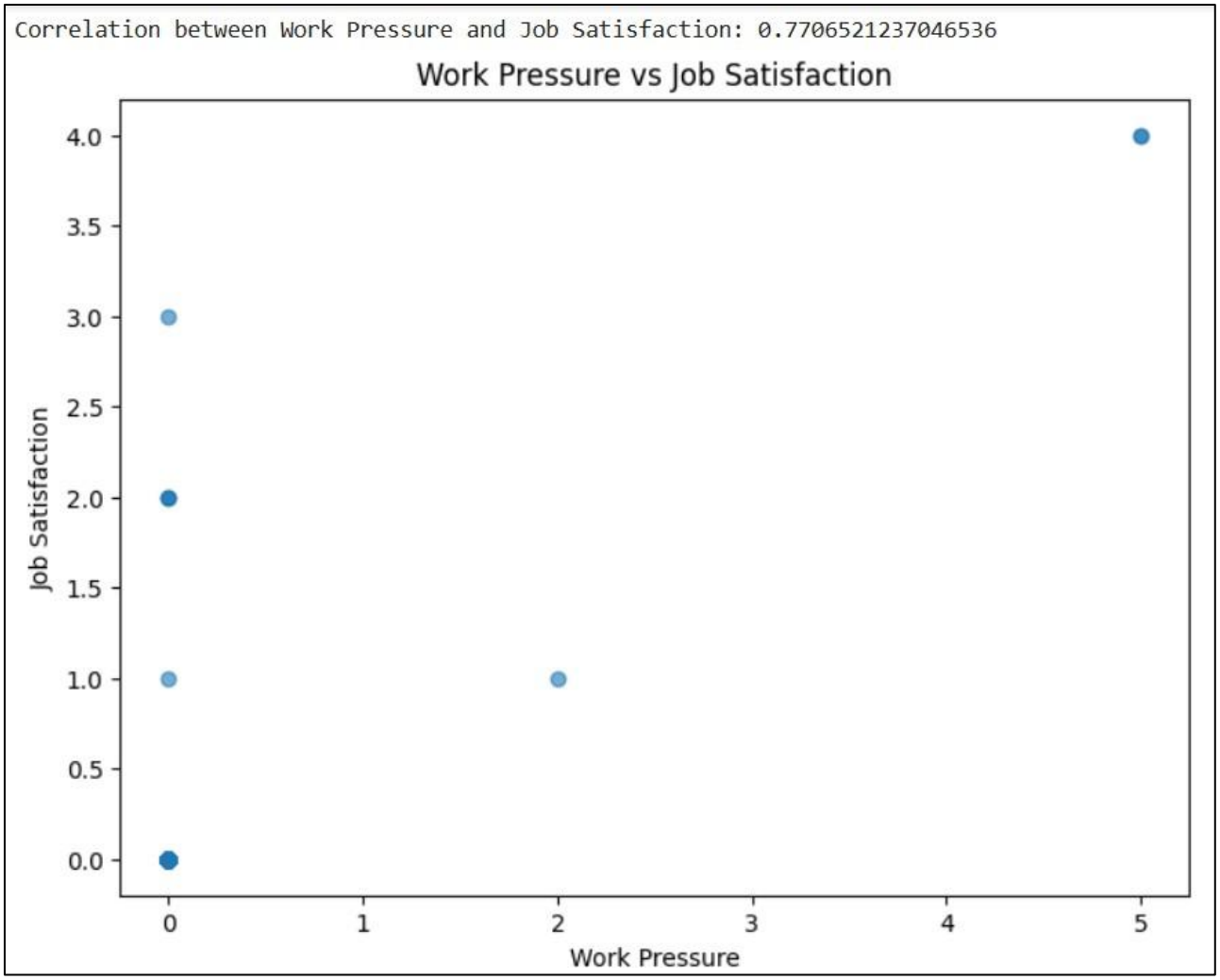


3. How does work pressure influence job satisfaction?

o Explore the relationship between the level of work pressure and job satisfaction among students who are working part-time or involved in internships.

```
# Filter only students
df_students = df[df['Profession'] == 'Student']

# Calculate Correlation
correlation = df_students['Work Pressure'].corr(df_students['Job Satisfaction'])
print(f"Correlation between Work Pressure and Job Satisfaction: {correlation}")

plt.figure(figsize=(8,6))
plt.scatter(df['Work Pressure'], df['Job Satisfaction'], alpha=0.6)
plt.title('Work Pressure vs Job Satisfaction')
plt.xlabel('Work Pressure')
plt.ylabel('Job Satisfaction')
plt.show()
```

```
Correlation between Work Pressure and Job Satisfaction: 0.7706521237046536
```



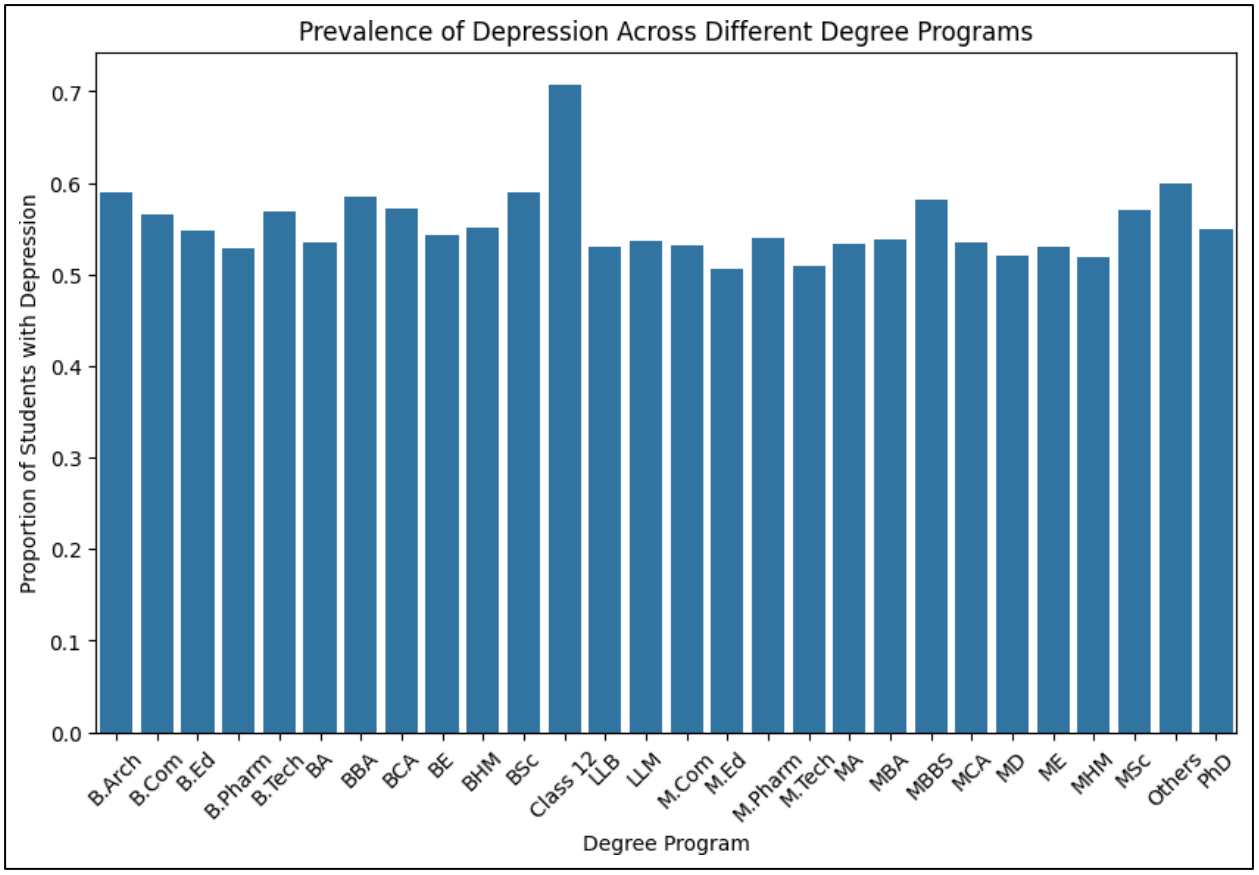
5. What is the distribution of depression across different degrees (B.Pharm, B.Sc., M.Tech, etc.)?

o Compare the prevalence of depression across students pursuing different degrees to see if certain programs have higher levels of depression.

```python
# Calculate the depression rate per degree program
depression_by_degree = df.groupby('Degree')['Depression'].mean()
print(depression_by_degree)

# Visualize the depression rate by degree program
plt.figure(figsize=(10, 6))
sns.barplot(x=depression_by_degree.index, y=depression_by_degree.values)
plt.title('Prevalence of Depression Across Different Degree Programs')
plt.xlabel('Degree Program')
plt.ylabel('Proportion of Students with Depression')
plt.xticks(rotation=45)
plt.show()
```

Degree	
B.Arch	0.590108
B.Com	0.566113
B.Ed	0.547453
B.Pharm	0.528395
B.Tech	0.568576
BA	0.534338
BBA	0.584770
BCA	0.571229
BE	0.543372
BHM	0.550270
BSc	0.589628
Class 12	0.707730
LLB	0.530551
LLM	0.537344
M.Com	0.531335
M.Ed	0.506098
M.Pharm	0.539519
M.Tech	0.508824
MA	0.533088
MBA	0.538324
MBBS	0.581295
MCA	0.534995
MD	0.520979
ME	0.529730
MHM	0.518325
MSc	0.571068
Others	0.600000
PhD	0.550000

Name: Depression, dtype: float64

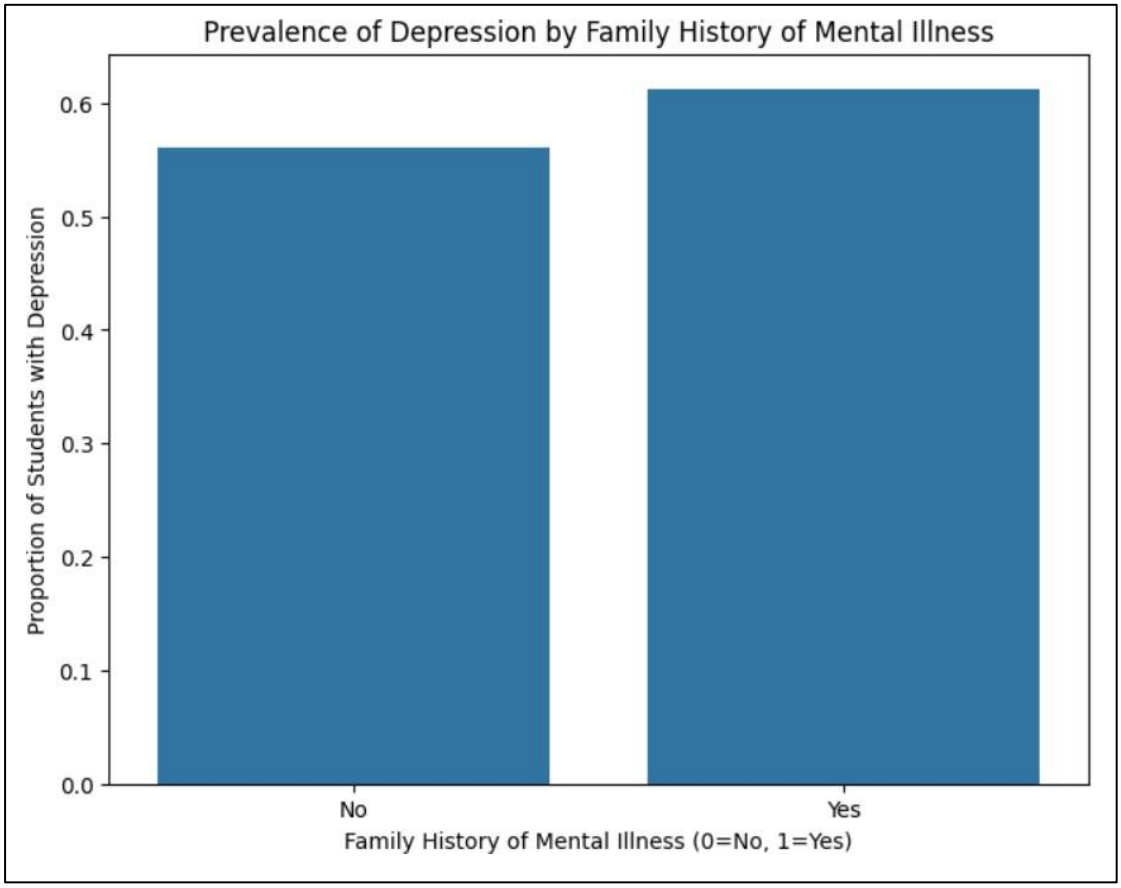


6. Do students with family histories of mental illness have a higher likelihood of depression?

o Explore whether students with a family history of mental illness are more likely to experience depression.

```
# Ensure 'Family History of Mental Illness' is binary (Yes = 1, No = 0)
df['Family History of Mental Illness'] = df['Family History of Mental Illness'].replace({'No': 0, 'Yes': 1})

# Calculate the depression rate by family history of mental illness
depression_by_family_history = df.groupby('Family History of Mental Illness')['Depression'].mean()
print(depression_by_family_history)

# Visualize the depression rate by family history of mental illness
plt.figure(figsize=(8, 6))
sns.barplot(x=depression_by_family_history.index, y=depression_by_family_history.values)
plt.title('Prevalence of Depression by Family History of Mental Illness')
plt.xlabel('Family History of Mental Illness (0=No, 1=Yes)')
plt.ylabel('Proportion of Students with Depression')
plt.xticks([0, 1], ['No', 'Yes'])
plt.show()
```

```
Family History of Mental Illness
0    0.560158
1    0.612688
Name: Depression, dtype: float64
```



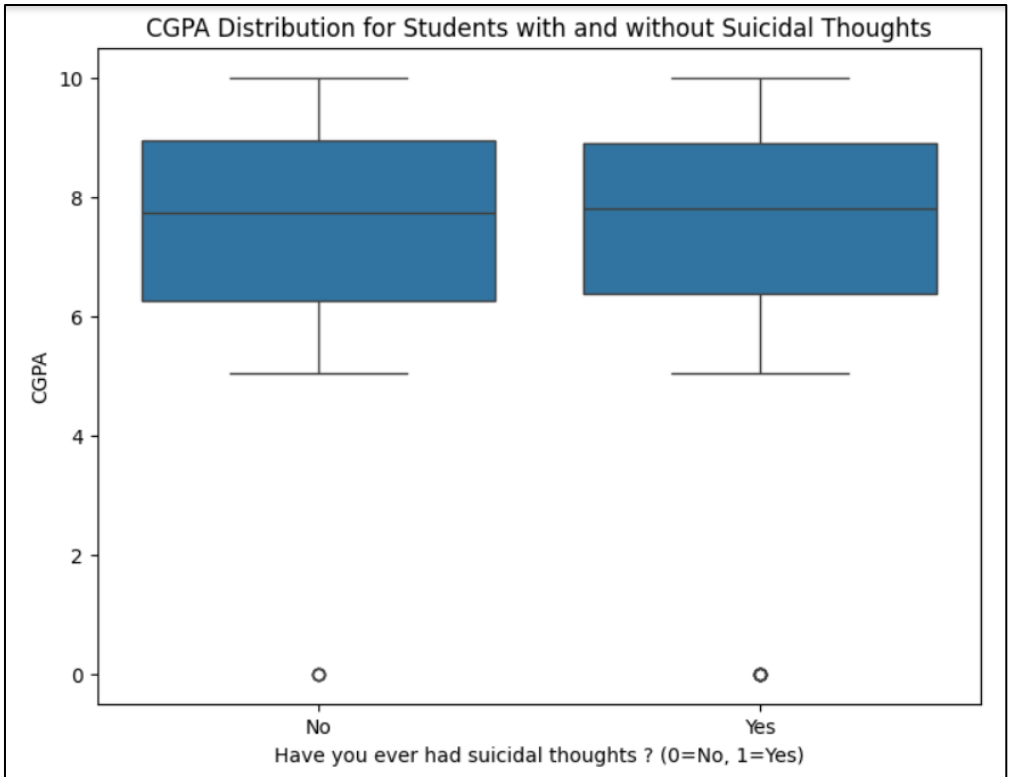
7. What is the average CGPA for students who have experienced suicidal thoughts versus those who have not?

o Analyze if there is a significant difference in CGPA between students who have had suicidal thoughts and those who have not.

```
[123] print(df['Have you ever had suicidal thoughts ?'].unique())

      [1 0]
```

```
# Ensure that 'Have you ever had suicidal thoughts ?' is binary (0 for No, 1 for Yes)
df['Have you ever had suicidal thoughts ?'] = df['Have you ever had suicidal thoughts ?'].replace({'No': 0, 'Yes': 1})

# Calculate the average CGPA for students with and without suicidal thoughts
cgpa_by_suicidal_thoughts = df.groupby('Have you ever had suicidal thoughts ?')['CGPA'].mean()
print(cgpa_by_suicidal_thoughts)

# Visualize the CGPA distribution for students with and without suicidal thoughts
plt.figure(figsize=(8, 6))
sns.boxplot(x='Have you ever had suicidal thoughts ?', y='CGPA', data=df)
plt.title('CGPA Distribution for Students with and without Suicidal Thoughts')
plt.xlabel('Have you ever had suicidal thoughts ? (0=No, 1=Yes)')
plt.ylabel('CGPA')
plt.xticks([0, 1], ['No', 'Yes'])
plt.show()

# Optional: Independent t-test for CGPA difference between groups
group_no_suicide = df[df['Have you ever had suicidal thoughts ?'] == 0]['CGPA']
group_yes_suicide = df[df['Have you ever had suicidal thoughts ?'] == 1]['CGPA']

# Perform t-test
t_stat, p_value = ttest_ind(group_no_suicide, group_yes_suicide, equal_var=False)
print(f"t-statistic: {t_stat}, p-value: {p_value}")
```

```
Have you ever had suicidal thoughts ?
0    7.640045
1    7.665569
Name: CGPA, dtype: float64
```



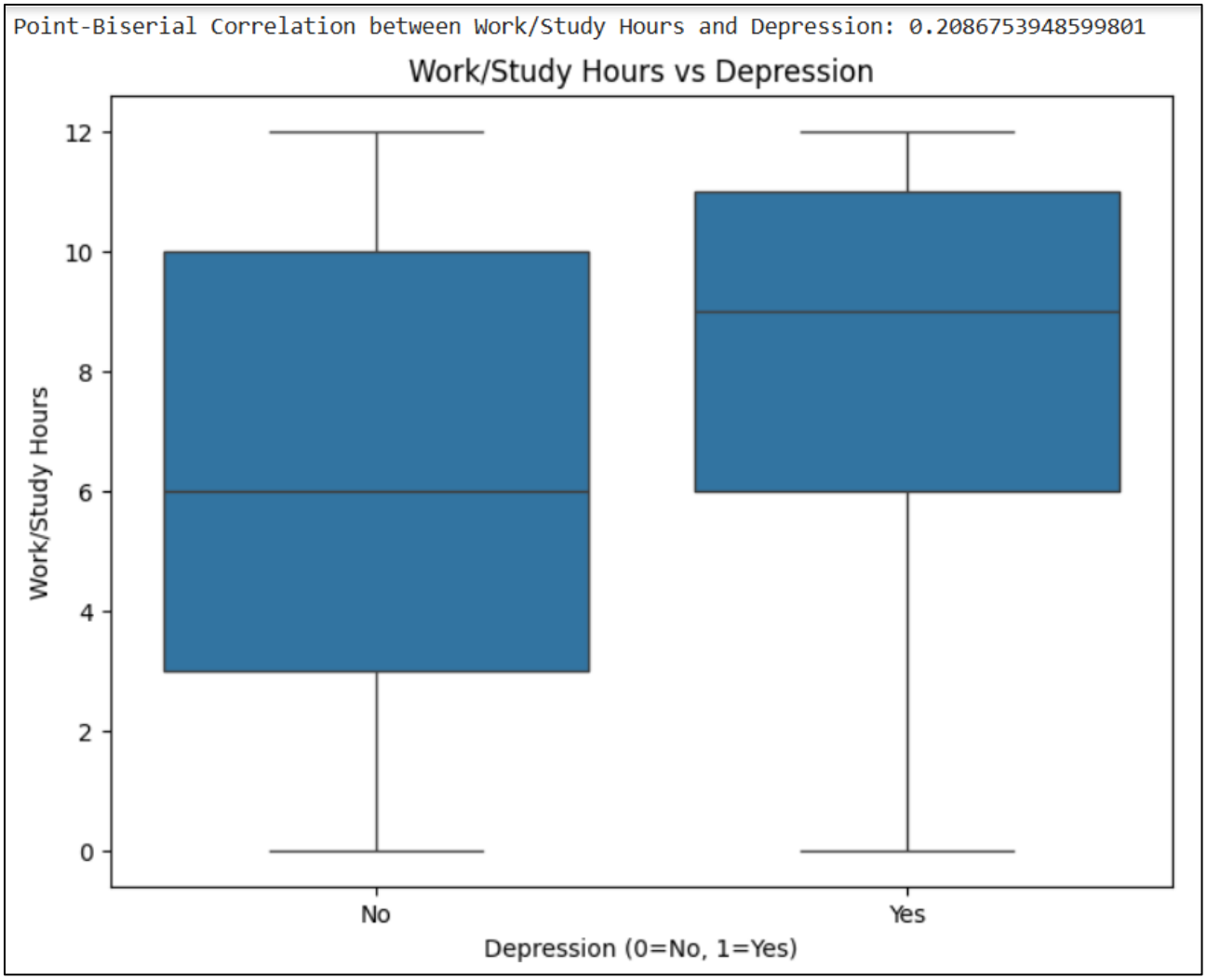

```
t-statistic: -1.3898647258158883, p-value: 0.16458465737160202
```

8. How do work/study hours correlate with depression?

o Explore whether students who work/study for longer hours are more prone to experiencing depression.

```python
correlation, _ = pointbiserialr(df['Work/Study Hours'], df['Depression'])
print(f"Point-Biserial Correlation between Work/Study Hours and Depression: {correlation}")

# Visualize the relationship between work/study hours and depression
plt.figure(figsize=(8, 6))
sns.boxplot(x='Depression', y='Work/Study Hours', data=df)
plt.title('Work/Study Hours vs Depression')
plt.xlabel('Depression (0=No, 1=Yes)')
plt.ylabel('Work/Study Hours')
plt.xticks([0, 1], ['No', 'Yes'])
plt.show()

# Optional: t-test to check if the difference in work/study hours is significant
group_no_depression = df[df['Depression'] == 0]['Work/Study Hours']
group_yes_depression = df[df['Depression'] == 1]['Work/Study Hours']

# Perform t-test
t_stat, p_value = ttest_ind(group_no_depression, group_yes_depression, equal_var=False)
print(f"t-statistic: {t_stat}, p-value: {p_value}")
```

```
Point-Biserial Correlation between Work/Study Hours and Depression: 0.2086753948599801
```

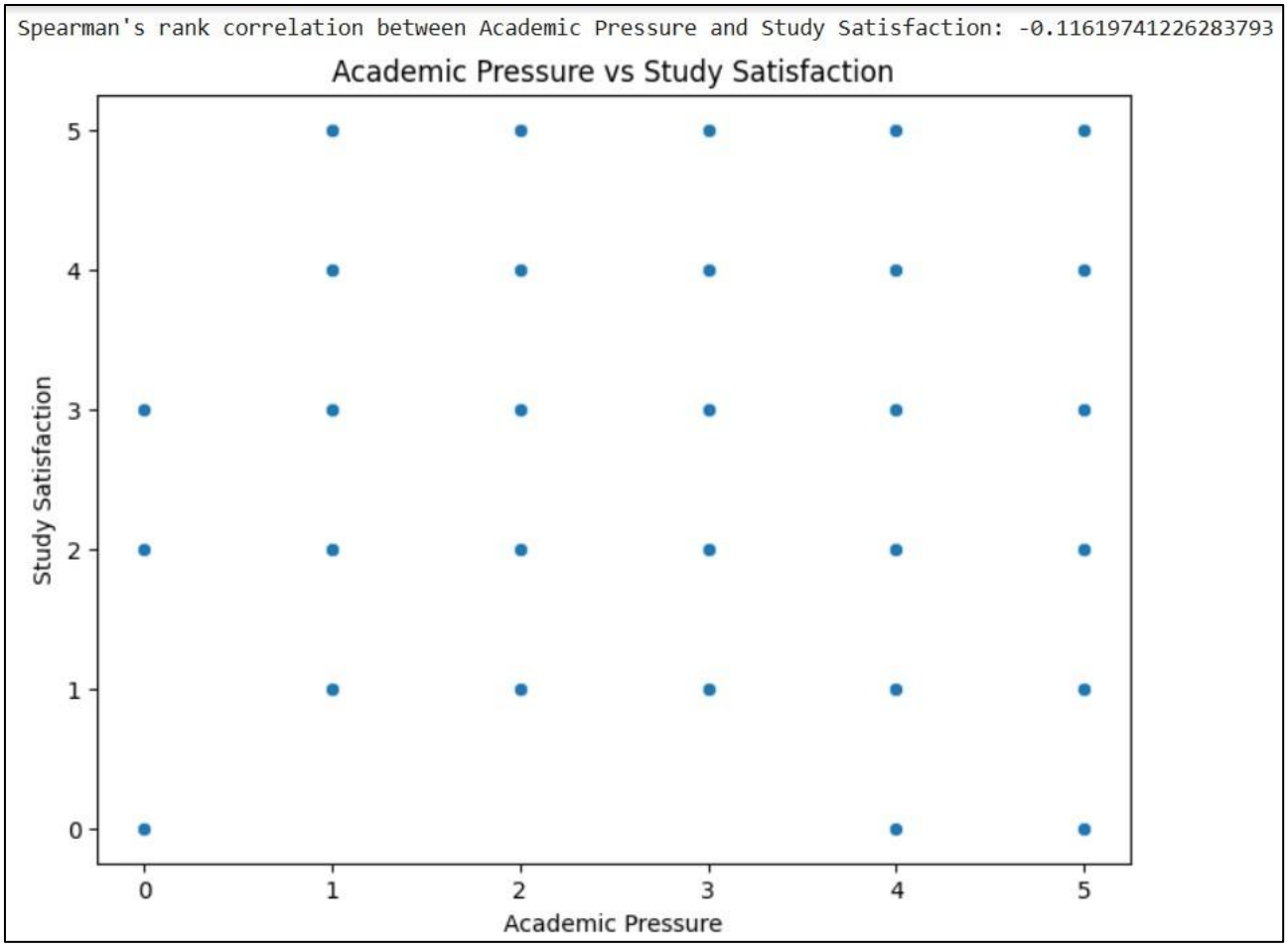


9. What is the relationship between academic pressure and study satisfaction?

o Investigate if higher academic pressure leads to lower study satisfaction.

```python
# Spearman's rank correlation
correlation_spearman = df['Academic Pressure'].corr(df['Study Satisfaction'], method='spearman')
print("Spearman's rank correlation between Academic Pressure and Study Satisfaction:", correlation_spearman)

plt.figure(figsize=(8, 6))
sns.scatterplot(x='Academic Pressure', y='Study Satisfaction', data=df)
plt.title('Academic Pressure vs Study Satisfaction')
plt.xlabel('Academic Pressure')
plt.ylabel('Study Satisfaction')
plt.show()
```

```
Spearman's rank correlation between Academic Pressure and Study Satisfaction: -0.11619741226283793
```

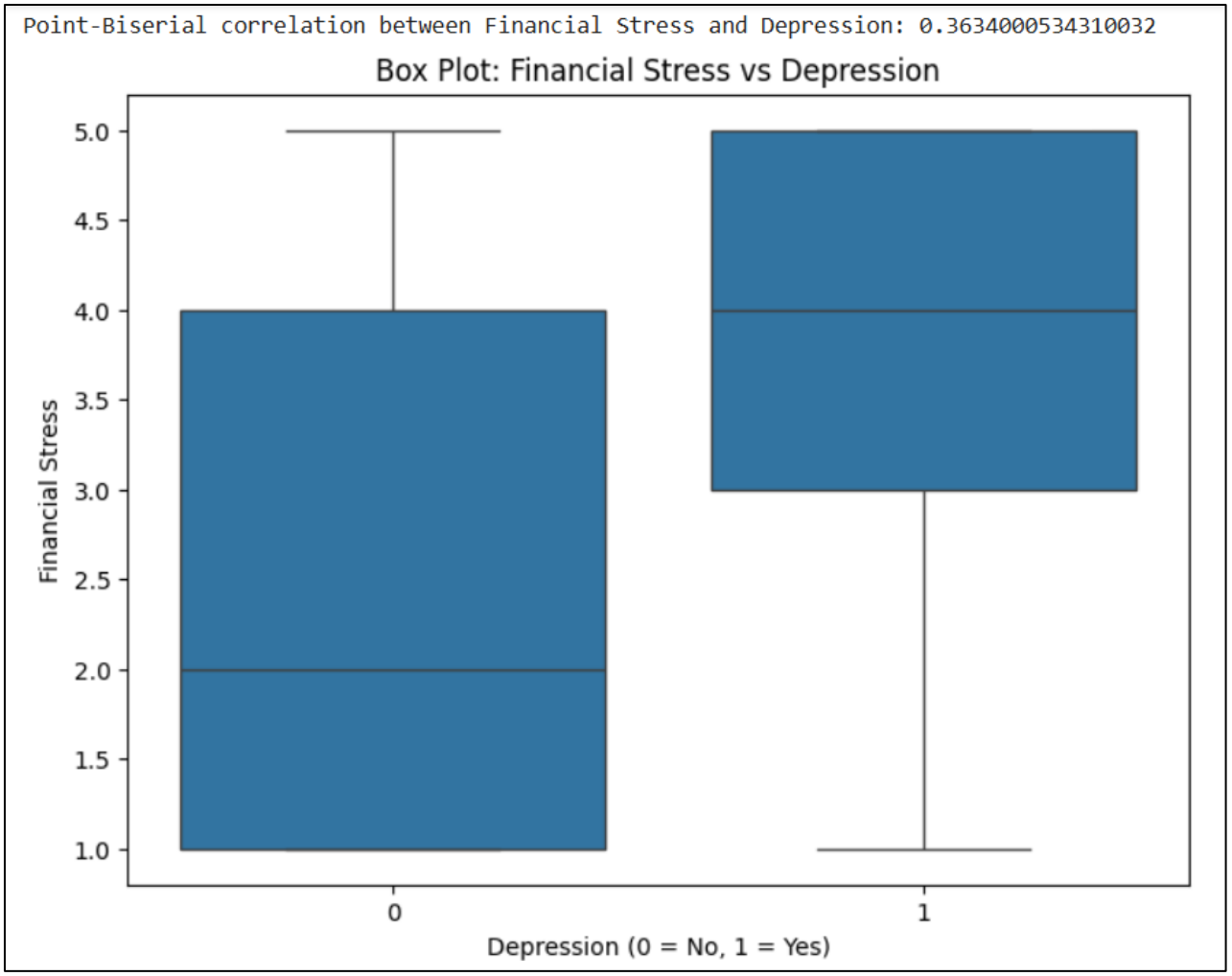


10. How does financial stress correlate with depression among students?

• Analyze if students who report higher financial stress are more likely to experience depression.

```python
# Calculate Point-Biserial correlation
correlation, p_value = pointbiserialr(df['Financial Stress'], df['Depression'])
print("Point-Biserial correlation between Financial Stress and Depression:", correlation)

# Box plot for Financial Stress vs Depression
plt.figure(figsize=(8,6))
sns.boxplot(x='Depression', y='Financial Stress', data=df)
plt.title('Box Plot: Financial Stress vs Depression')
plt.xlabel('Depression (0 = No, 1 = Yes)')
plt.ylabel('Financial Stress')
plt.show()
```

```
Point-Biserial correlation between Financial Stress and Depression: 0.3634000534310032
```

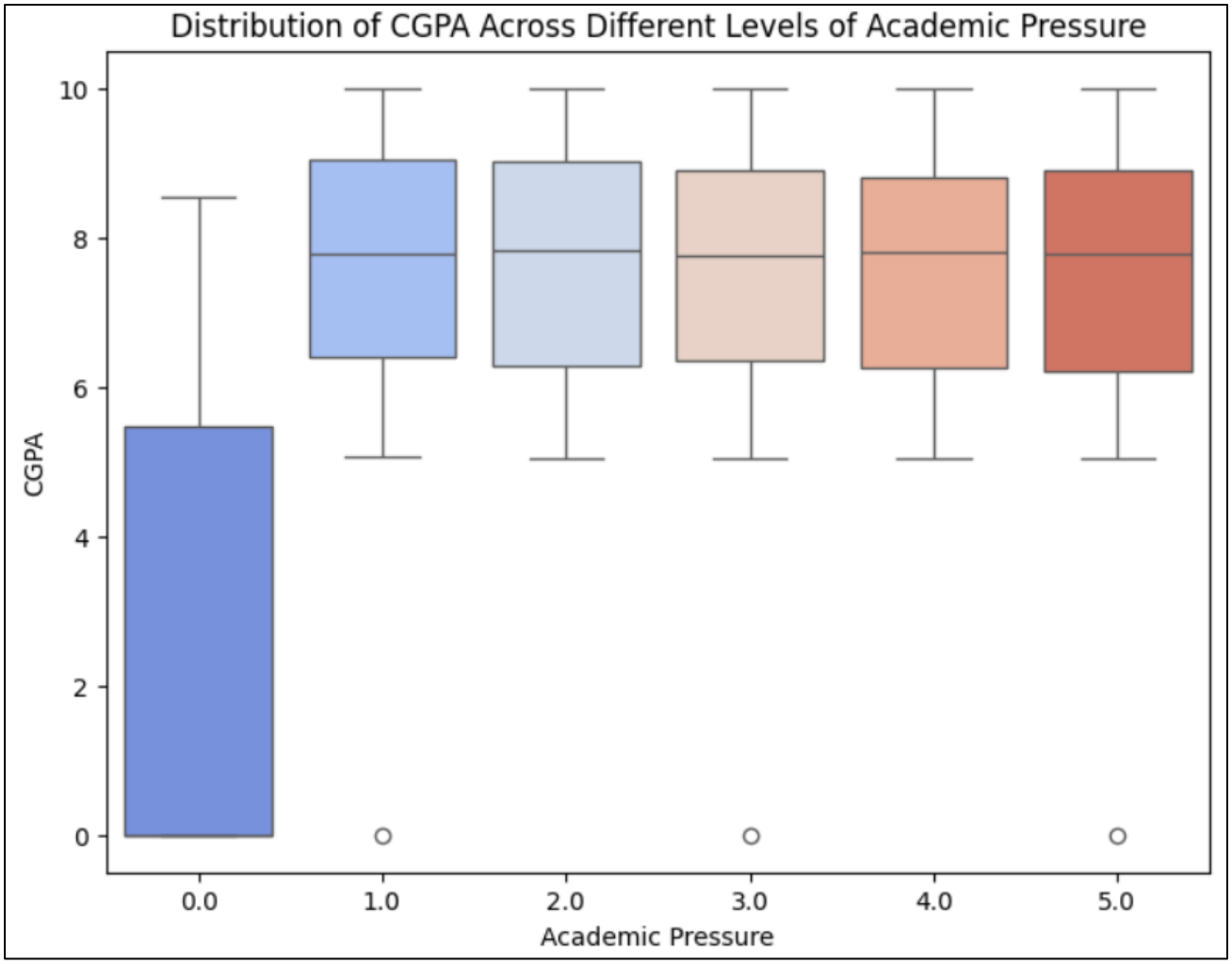


Visualization Questions:

1. What is the distribution of CGPA across different levels of academic pressure?

o Create a box plot or bar chart to visualize how CGPA differs across various levels of academic pressure.

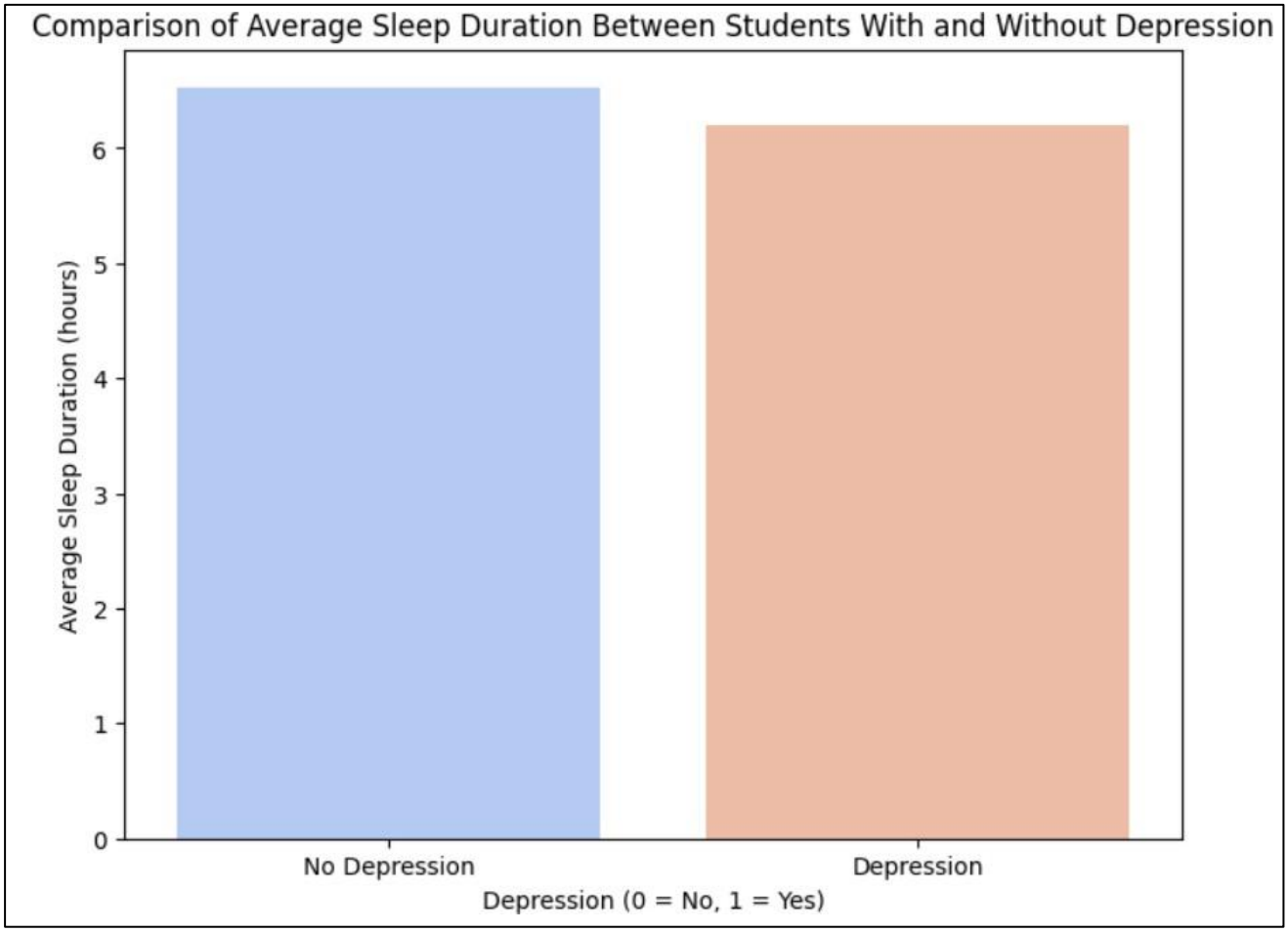
```
# Create a box plot
plt.figure(figsize=(8,6))
sns.boxplot(x=df['Academic Pressure'], y=df['CGPA'], palette='coolwarm')
plt.xlabel('Academic Pressure')
plt.ylabel('CGPA')
plt.title('Distribution of CGPA Across Different Levels of Academic Pressure')
plt.show()
```



2. How does sleep duration vary among students with and without depression?

o Use a bar chart to compare the average sleep duration of students who have depression vs. those who do not.

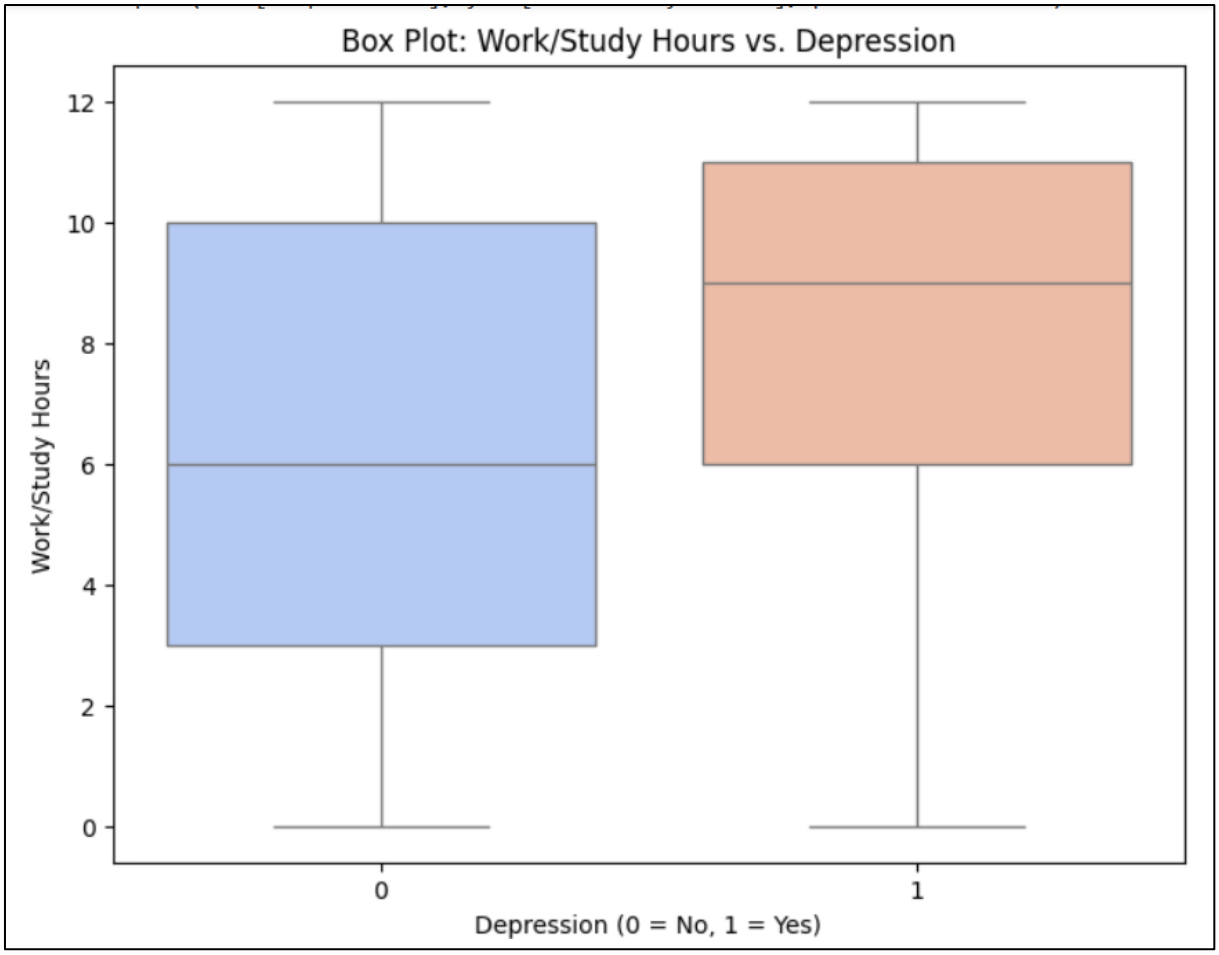
```
# Calculate the mean sleep duration for each depression category
sleep_means = df.groupby('Depression')['Sleep Duration'].mean().reset_index()

# Create a bar plot
plt.figure(figsize=(8,6))
sns.barplot(x='Depression', y='Sleep Duration', data=sleep_means, palette='coolwarm')
plt.xlabel('Depression (0 = No, 1 = Yes)')
plt.ylabel('Average Sleep Duration (hours)')
plt.title('Comparison of Average Sleep Duration Between Students With and Without Depression')
plt.xticks(ticks=[0,1], labels=['No Depression', 'Depression'])  # Rename x-axis labels
plt.show()
```



4. What is the relationship between work/study hours and depression?

o Use a scatter plot or line chart to show the relationship between the number of work/study hours and depression.

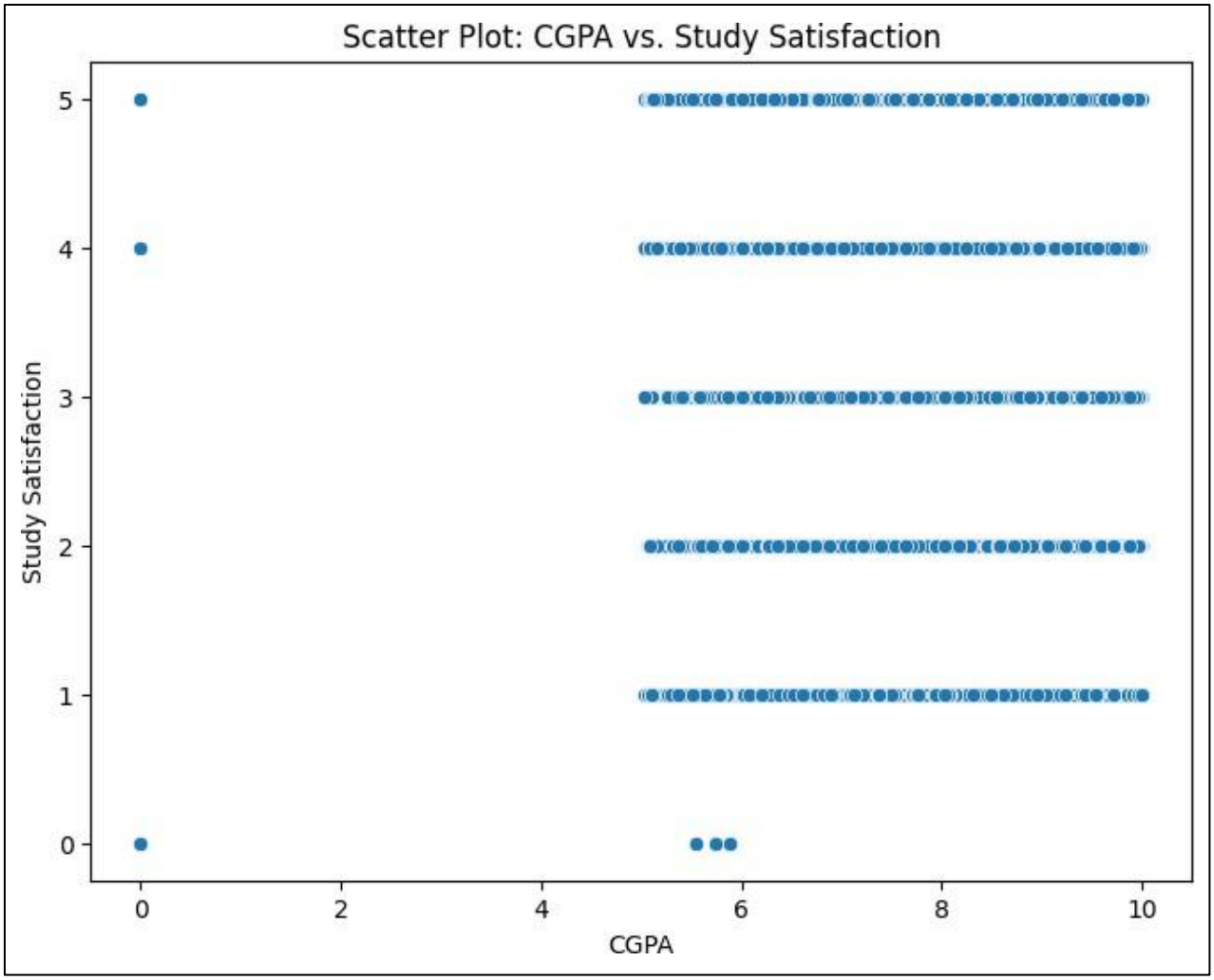
```
plt.figure(figsize=(8,6))
sns.boxplot(x=df['Depression'], y=df['Work/Study Hours'], palette="coolwarm")

# Labels and title
plt.xlabel("Depression (0 = No, 1 = Yes)")
plt.ylabel("Work/Study Hours")
plt.title("Box Plot: Work/Study Hours vs. Depression")
plt.show()
```



5. How does study satisfaction vary by CGPA?

o Create a scatter plot to explore whether students with higher CGPA also report higher study satisfaction.

```
plt.figure(figsize=(8,6))
sns.scatterplot(x=df['CGPA'], y=df['Study Satisfaction'])
plt.xlabel("CGPA")
plt.ylabel("Study Satisfaction")
plt.title("Scatter Plot: CGPA vs. Study Satisfaction")
plt.show()
```

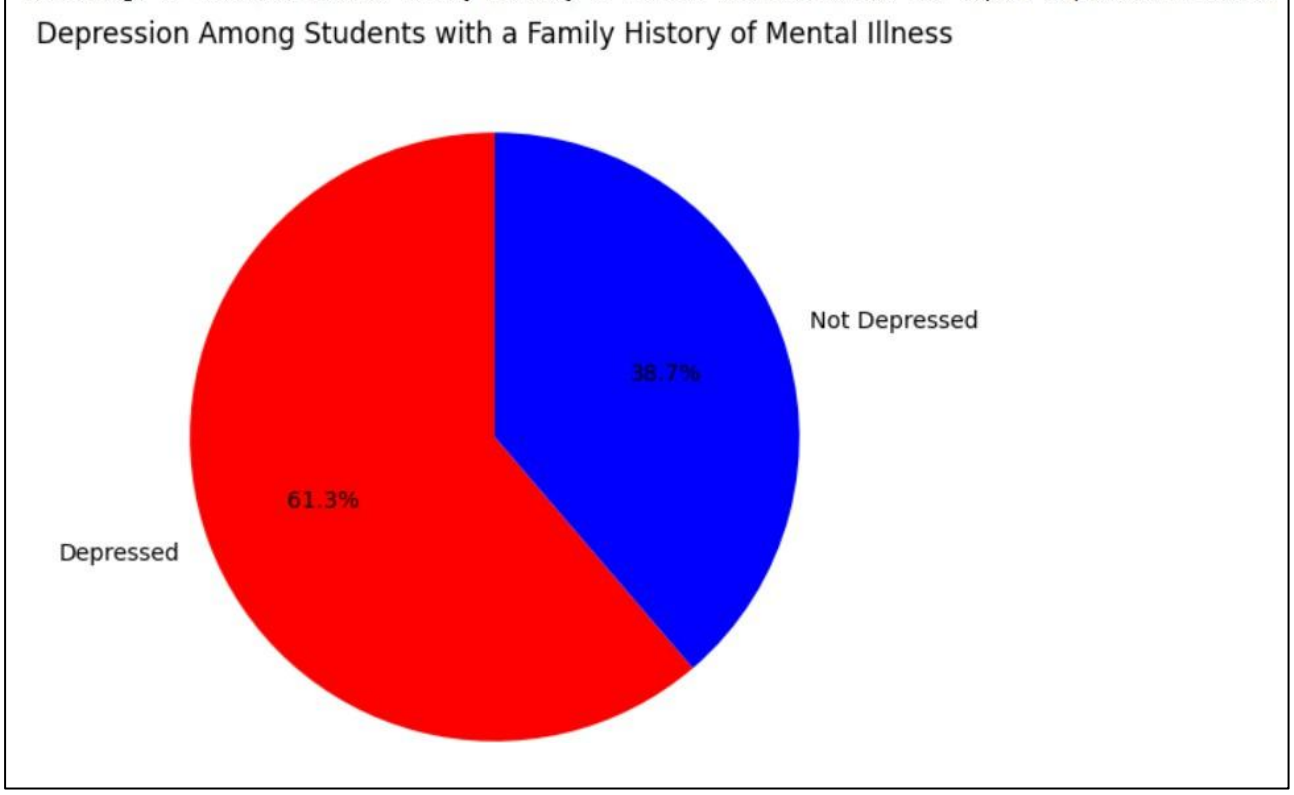


6. What percentage of students with a family history of mental illness report depression?

o Use a pie chart or bar chart to visualize the percentage of students with a family history of mental illness who report depression.

```python
# Filter students with family history of mental illness
family_history_df = df[df['Family History of Mental Illness'] == 1]

# Count students with depression (1) and without depression (0)
depression_counts = family_history_df['Depression'].value_counts()

# Calculate percentage
total_students = len(family_history_df)
percentage_depressed = (depression_counts[1] / total_students) * 100
percentage_not_depressed = (depression_counts[0] / total_students) * 100

print(f"Percentage of students with a family history of mental illness who report depression: {percentage_depressed:.2f}%")
print(f"Percentage of students with a family history of mental illness who do NOT report depression: {percentage_not_depressed:.2f}%")


# Labels and data
labels = ['Depressed', 'Not Depressed']
sizes = [percentage_depressed, percentage_not_depressed]
colors = ['red', 'blue']

# Create pie chart
plt.figure(figsize=(6,6))
plt.pie(sizes, labels=labels, autopct='%1.1f%%', colors=colors, startangle=90)
plt.title("Depression Among Students with a Family History of Mental Illness")
plt.show()
```

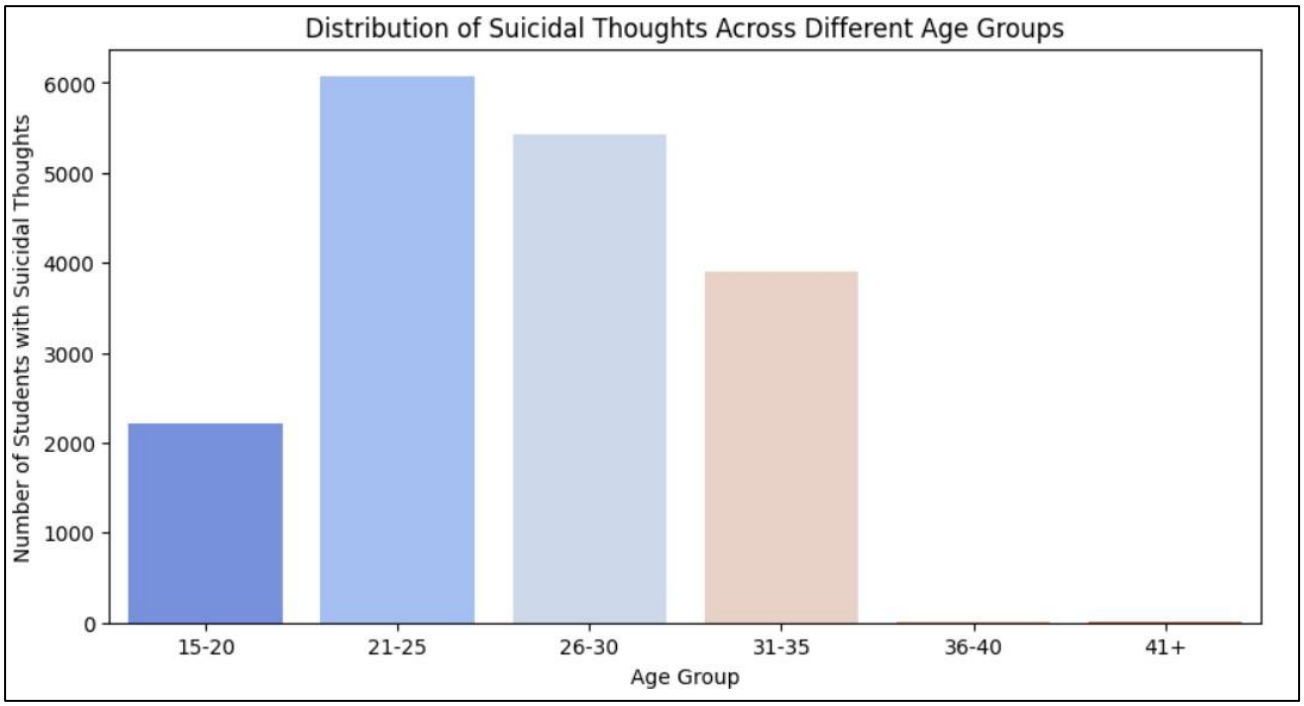
```
Percentage of students with a family history of mental illness who report depression: 61.27%
Percentage of students with a family history of mental illness who do NOT report depression: 38.73%
```



8. What is the distribution of suicidal thoughts across different age groups?

o Use a histogram or bar chart to compare the prevalence of suicidal thoughts among students of different ages.

```
# Define age bins (customize as needed)
bins = [15, 20, 25, 30, 35, 40, 50]  # Adjust based on your dataset
labels = ["15-20", "21-25", "26-30", "31-35", "36-40", "41+"]

# Create an Age Group column
df['Age Group'] = pd.cut(df['Age'], bins=bins, labels=labels, right=False)

# Count the number of students with suicidal thoughts in each age group
suicidal_counts = df.groupby('Age Group')['Have you ever had suicidal thoughts ?'].sum().reset_index()

# Display the counts
print(suicidal_counts)

# Set figure size
plt.figure(figsize=(10,5))
sns.barplot(x='Age Group', y='Have you ever had suicidal thoughts ?', data=suicidal_counts, palette='coolwarm')
plt.xlabel("Age Group")
plt.ylabel("Number of Students with Suicidal Thoughts")
plt.title("Distribution of Suicidal Thoughts Across Different Age Groups")
plt.show()
```

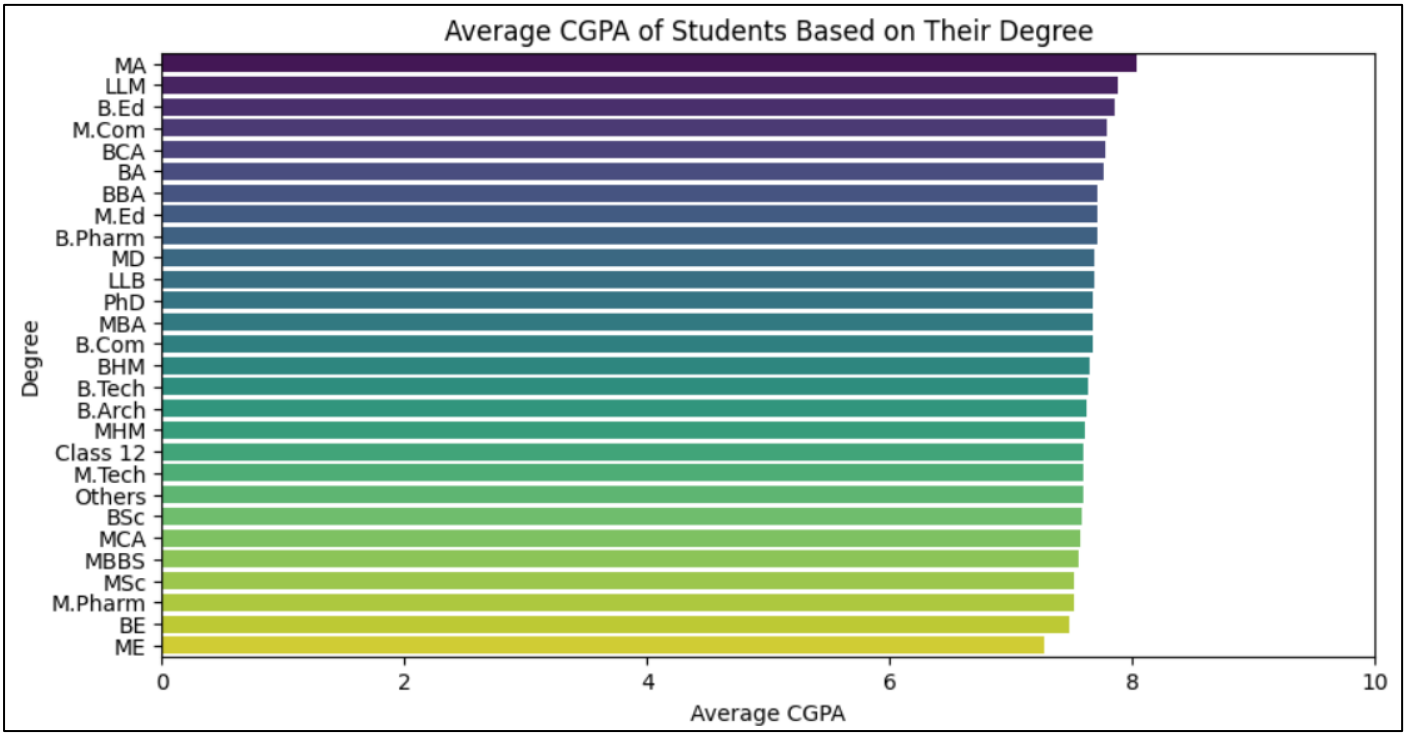


9. What is the average CGPA of students based on their degree?

o Create a bar chart to show the average CGPA for students pursuing different degrees (e.g., B.Pharm, M.Tech, B.Ed, etc.).

```
# Group by 'Degree' and calculate the average CGPA
degree_cgpa = df.groupby('Degree')['CGPA'].mean().reset_index()

# Sort by CGPA for better visualization
degree_cgpa = degree_cgpa.sort_values(by='CGPA', ascending=False)

# Display the average CGPA values
print(degree_cgpa)

# Set figure size
plt.figure(figsize=(10,5))
sns.barplot(x='CGPA', y='Degree', data=degree_cgpa, palette='viridis'
plt.xlabel("Average CGPA")
plt.ylabel("Degree")
plt.title("Average CGPA of Students Based on Their Degree")
plt.xlim(0, 10)  # Assuming CGPA is on a 10-point scale
plt.show()
```

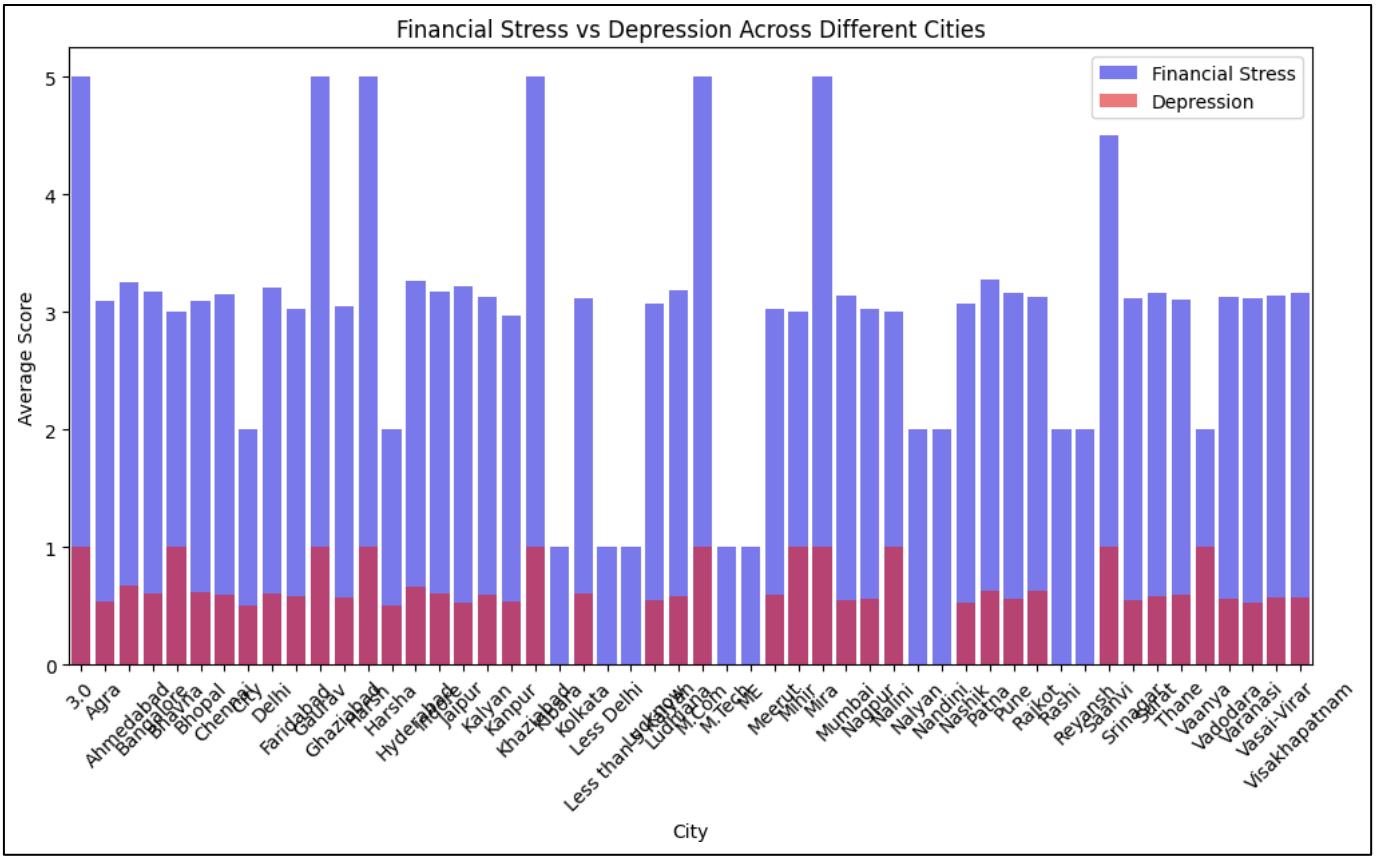
	Degree	CGPA
18	MA	8.028364
13	LLM	7.879419
2	B.Ed	7.851630
14	M.Com	7.788978
7	BCA	7.770768
5	BA	7.763300
6	BBA	7.714468
15	M.Ed	7.709110
3	B.Pharm	7.707753
22	MD	7.691381
12	LLB	7.686602
27	PhD	7.677808
19	MBA	7.676649
1	B.Com	7.670306
9	BHM	7.649859
4	B.Tech	7.636513
0	B.Arch	7.615393
24	MHM	7.606440
11	Class 12	7.594184
17	M.Tech	7.593598
26	Others	7.588857
10	BSc	7.578185
21	MCA	7.563471
20	MBBS	7.552806
25	MSc	7.514819
16	M.Pharm	7.513574
8	BE	7.479067
23	ME	7.269676

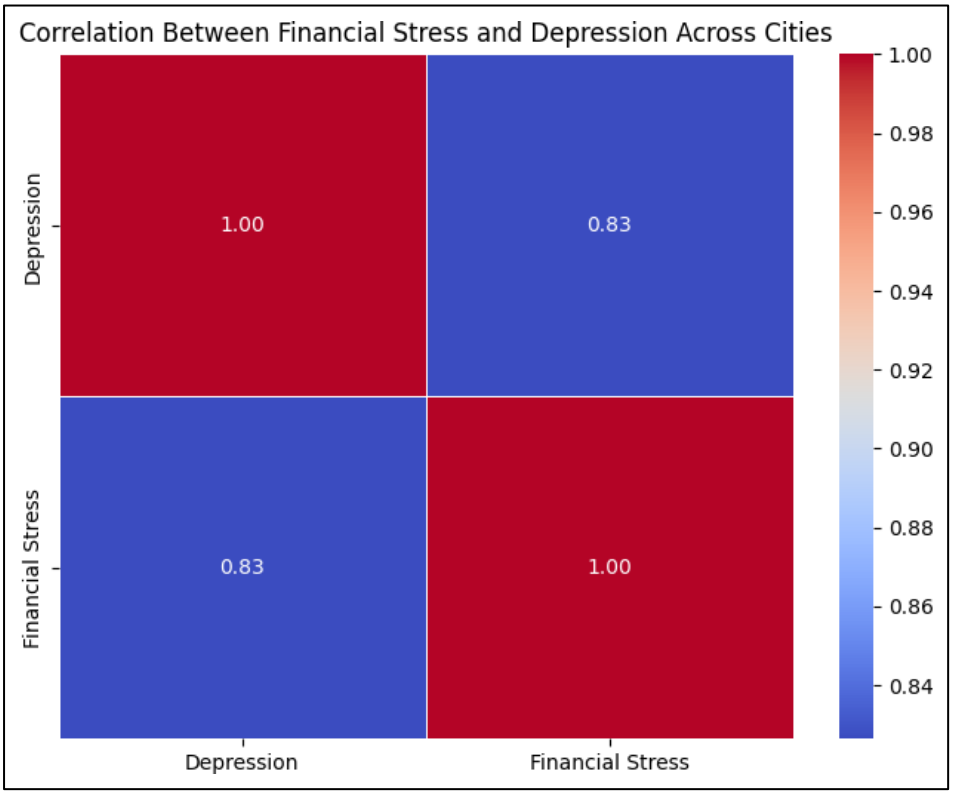


10. How does financial stress affect the likelihood of depression across different cities?

• Create a heatmap or grouped bar chart to compare the correlation between financial stress and depression across various cities.

```
# Group data by city and calculate the mean for financial stress and depression
city_grouped = df.groupby('City')[['Financial Stress', 'Depression']].mean().reset_index()

# Create a grouped bar chart
plt.figure(figsize=(12, 6))
sns.barplot(x='City', y='Financial Stress', data=city_grouped, color='blue', alpha=0.6, label='Financial Stress')
sns.barplot(x='City', y='Depression', data=city_grouped, color='red', alpha=0.6, label='Depression')
plt.xlabel('City')
plt.ylabel('Average Score')
plt.title('Financial Stress vs Depression Across Different Cities')
plt.xticks(rotation=45)
plt.legend()
plt.show()

# Pivot the data to calculate mean financial stress and depression by city
pivot_data = df.pivot_table(values=['Financial Stress', 'Depression'], index='City', aggfunc='mean')

# Compute the correlation matrix between financial stress and depression across cities
correlation_matrix = pivot_data.corr()

# Set up the heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt='.2f', linewidths=0.5)
plt.title("Correlation Between Financial Stress and Depression Across Cities")
plt.show()
```

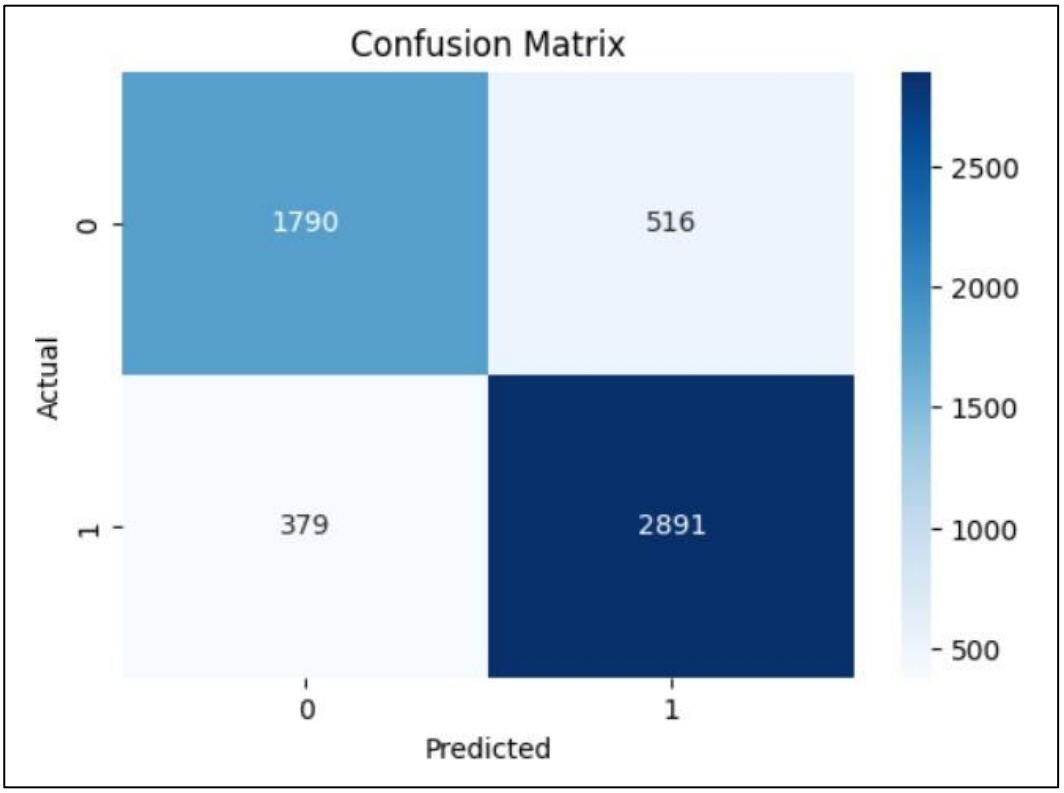

Financial Stress vs Depression Across Different Cities
Financial Stress
Depression
Average Score
City
0
1
2
3
4
5
3.0
Agra
Ahmedabad
Bangalore
Bhavna
Bhopal
Chennai
City
Delhi
Faridabad
Gaurav
Ghaziabad
Harsh
Harsha
Hyderabad
Indore
Jaipur
Kalyan
Kanpur
Khaziabad
Kolkata
Less Delhi
Less than 5 Lakhs
Lucknow
Ludhiana
M.Com
M.Tech
ME
Meerut
Mihir
Mira
Mumbai
Nagpur
Nalini
Nalyan
Nandini
Nashik
Patna
Pune
Rajkot
Rashi
Reyansh
Saanvi
Srinagar
Surat
Thane
Vaanya
Vadodara
Varanasi
Vasai-Virar
Visakhapatnam


Correlation Between Financial Stress and Depression Across Cities
Depression
Financial Stress
1.00
0.83
0.83
1.00
Depression
Financial Stress
1.00
0.98
0.96
0.94
0.92
0.90
0.88
0.86
0.84

Model building

```python
# Convert categorical variables to numeric (if any)
df = pd.get_dummies(df, drop_first=True)

# Define Features (X) and Target (y)
X = df.drop(columns=["Depression"])
y = df["Depression"]

# Split Data into Training (80%) and Testing (20%)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Initialize and Train Random Forest Model
rf_model = RandomForestClassifier(n_estimators=100, random_state=42)
rf_model.fit(X_train, y_train)

# Make Predictions
y_pred = rf_model.predict(X_test)

# Evaluate Model Performance
accuracy = accuracy_score(y_test, y_pred)
print(f"Model Accuracy: {accuracy:.2f}")

# Classification Report
print("\nClassification Report:\n", classification_report(y_test, y_pred))

# Confusion Matrix
plt.figure(figsize=(6, 4))
sns.heatmap(confusion_matrix(y_test, y_pred), annot=True, fmt='d', cmap="Blues")
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix")
plt.show()
```

```
Model Accuracy: 0.84

Classification Report:
              precision    recall  f1-score   support

           0       0.83      0.78      0.80      2306
           1       0.85      0.88      0.87      3270

    accuracy                           0.84      5576
   macro avg       0.84      0.83      0.83      5576
weighted avg       0.84      0.84      0.84      5576
```

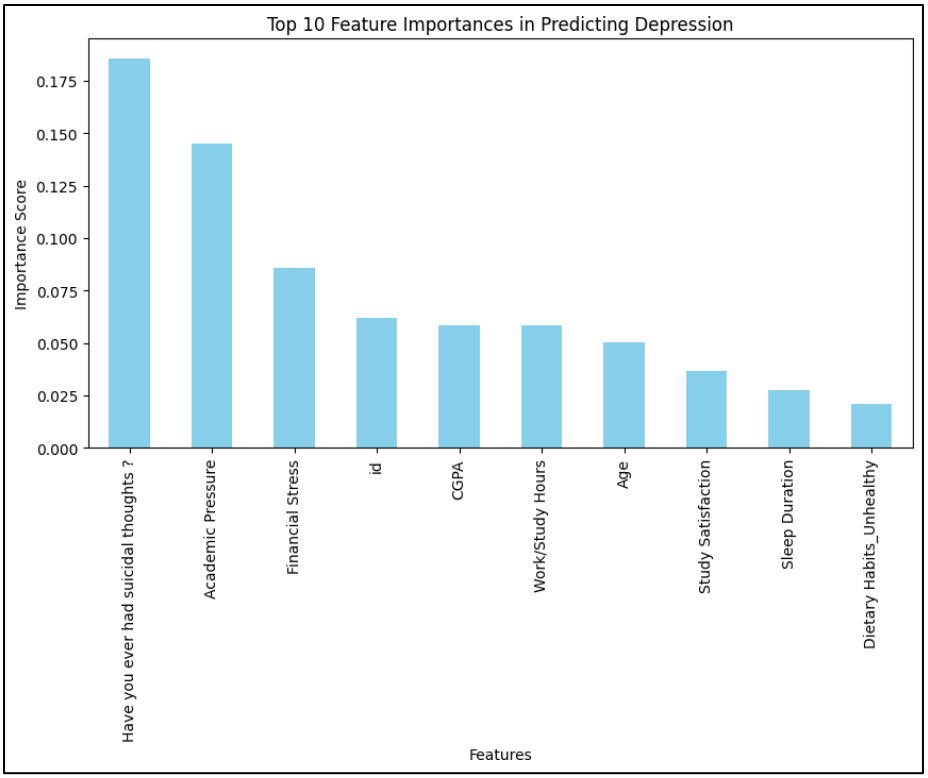



```
cm = confusion_matrix(y_test, y_pred)
print("\nConfusion Matrix:\n", cm)
```

```
Confusion Matrix:
 [[1790  516]
 [ 379 2891]]
```

Model Evaluation & Insights

```
# Get feature importance scores
feature_importances = pd.Series(rf_model.feature_importances_, index=X.columns)

# Sort and visualize the most important features
plt.figure(figsize=(10, 5))
feature_importances.sort_values(ascending=False).head(10).plot(kind="bar", color="skyblue")
plt.title("Top 10 Feature Importances in Predicting Depression")
plt.xlabel("Features")
plt.ylabel("Importance Score")
plt.show()
```



Conclusion:

This project successfully demonstrates how machine learning can be applied to predict student depression using behavioral, academic, and lifestyle data. The **Random Forest model** proved to be effective in identifying students at risk of depression, allowing for potential early intervention. The analysis showed that **academic and work pressure, sleep duration, financial stress, and mental health history** are crucial factors influencing depression. Understanding these key indicators can help educators, parents, and institutions take proactive measures to support student well-being.

SWOT Analysis of Student Depression Prediction Project

Strengths:

- **Data-Driven Approach** – Uses machine learning to analyze large datasets for meaningful insights.
- **Early Detection** – Helps identify students at risk of depression, enabling early intervention.
- **Feature Importance Analysis** – Highlights key factors contributing to student mental health issues.
- **Scalability** – Can be integrated into educational institutions for large-scale monitoring.
- **Interpretability** – The Random Forest model provides clear feature importance insights.

Weaknesses:

+ **Data Quality Issues** – Missing values, biases, or inaccuracies in survey-based data can affect predictions.
+ **Ethical Concerns** – Handling sensitive mental health data requires strict privacy measures.
+ **Generalization** – Model effectiveness might vary across different student populations and educational settings.
+ **Limited Psychological Context** – Machine learning alone cannot replace mental health professionals.

Opportunities:

- **Collaboration with Psychologists** – Combining AI with expert psychological analysis for more accurate predictions.
- **Integration with Educational Platforms** – Implementing this model in universities and schools for real-time monitoring.
- **Enhancing the Model** – Using advanced techniques like deep learning or reinforcement learning for better predictions.
- **Mobile & Web Applications** – Developing a user-friendly interface for students to assess their mental health.
- **Personalized Recommendations** – Providing actionable insights to improve student well-being.

Threats:

⚠ **Data Privacy Risks** – Handling sensitive student information requires strong security protocols.

⚠ **Ethical & Legal Challenges** – Regulatory issues may arise regarding the use of AI in mental health diagnostics.

⚠ **Misinterpretation of Results** – A false positive/negative prediction might lead to unnecessary panic or neglect of real cases.

⚠ **Resistance to Adoption** – Educational institutions and students may be hesitant to trust AI-driven mental health predictions.

⚠ **Continuous Model Maintenance** – Requires regular updates to adapt to changing student behaviors and trends.