

```
import pandas as pd
df = pd.read_csv(r"C:\Users\ROYCE\OneDrive\Documents\
student_depression_dataset.csv")
df.head(10)
```

	id	Gender	Age	City	Profession	Academic Pressure \
0	2	Male	33.0	Visakhapatnam	Student	5.0
1	8	Female	24.0	Bangalore	Student	2.0
2	26	Male	31.0	Srinagar	Student	3.0
3	30	Female	28.0	Varanasi	Student	3.0
4	32	Female	25.0	Jaipur	Student	4.0
5	33	Male	29.0	Pune	Student	2.0
6	52	Male	30.0	Thane	Student	3.0
7	56	Female	30.0	Chennai	Student	2.0
8	59	Male	28.0	Nagpur	Student	3.0
9	62	Male	31.0	Nashik	Student	2.0

	Work Pressure	CGPA	Study Satisfaction	Job Satisfaction \
0	0.0	8.97	2.0	0.0
1	0.0	5.90	5.0	0.0
2	0.0	7.03	5.0	0.0
3	0.0	5.59	2.0	0.0
4	0.0	8.13	3.0	0.0
5	0.0	5.70	3.0	0.0
6	0.0	9.54	4.0	0.0
7	0.0	8.04	4.0	0.0
8	0.0	9.79	1.0	0.0
9	0.0	8.38	3.0	0.0

	Sleep Duration	Dietary Habits	Degree \
0	'5-6 hours'	Healthy	B.Pharm
1	'5-6 hours'	Moderate	BSc
2	'Less than 5 hours'	Healthy	BA
3	'7-8 hours'	Moderate	BCA
4	'5-6 hours'	Moderate	M.Tech
5	'Less than 5 hours'	Healthy	PhD
6	'7-8 hours'	Healthy	BSc
7	'Less than 5 hours'	Unhealthy	'Class 12'
8	'7-8 hours'	Moderate	B.Ed
9	'Less than 5 hours'	Moderate	LLB

	Have you ever had suicidal thoughts ?	Work/Study Hours	Financial Stress \
0	Yes	3.0	
1.0			
1	No	3.0	
2.0			
2	No	9.0	
1.0			
3	Yes	4.0	

5.0		
4	Yes	1.0
1.0		
5	No	4.0
1.0		
6	No	1.0
2.0		
7	No	0.0
1.0		
8	Yes	12.0
3.0		
9	Yes	2.0
5.0		

	Family History of Mental Illness	Depression
0	No	1
1	Yes	0
2	Yes	0
3	Yes	1
4	No	0
5	No	0
6	No	0
7	Yes	0
8	No	1
9	No	1

df.tail(5)

	id	Gender	Age	City	Profession	Academic Pressure \
27896	140685	Female	27.0	Surat	Student	5.0
27897	140686	Male	27.0	Ludhiana	Student	2.0
27898	140689	Male	31.0	Faridabad	Student	3.0
27899	140690	Female	18.0	Ludhiana	Student	5.0
27900	140699	Male	27.0	Patna	Student	4.0

	Work Pressure	CGPA	Study Satisfaction	Job Satisfaction \
27896	0.0	5.75	5.0	0.0
27897	0.0	9.40	3.0	0.0
27898	0.0	6.61	4.0	0.0
27899	0.0	6.88	2.0	0.0
27900	0.0	9.24	1.0	0.0

	Sleep Duration	Dietary Habits	Degree \
27896	'5-6 hours'	Unhealthy	'Class 12'

27897	'Less than 5 hours'	Healthy	MSc
27898	'5-6 hours'	Unhealthy	MD
27899	'Less than 5 hours'	Healthy	'Class 12'
27900	'Less than 5 hours'	Healthy	BCA

	Have you ever had suicidal thoughts ?	Work/Study Hours \
27896	Yes	7.0
27897	No	0.0
27898	No	12.0
27899	Yes	10.0
27900	Yes	2.0

	Financial Stress	Family History of Mental Illness	Depression
27896	1.0	Yes	0
27897	3.0	Yes	0
27898	2.0	No	0
27899	5.0	No	1
27900	3.0	Yes	1

df.describe()

	id	Age	Academic Pressure	Work
count	27901.000000	27901.000000	27901.000000	27901.000000
mean	70442.149421	25.822300	3.141214	0.000430
std	40641.175216	4.905687	1.381465	0.043992
min	2.000000	18.000000	0.000000	0.000000
25%	35039.000000	21.000000	2.000000	0.000000
50%	70684.000000	25.000000	3.000000	0.000000
75%	105818.000000	30.000000	4.000000	0.000000
max	140699.000000	59.000000	5.000000	5.000000

	CGPA	Study Satisfaction	Job Satisfaction	Work/Study
Hours \				
count	27901.000000	27901.000000	27901.000000	27901.000000
mean	7.656104	2.943837	0.000681	7.156984
std	1.470707	1.361148	0.044394	3.707642
min	0.000000	0.000000	0.000000	0.000000
25%	6.290000	2.000000	0.000000	

4.000000			
50%	7.770000	3.000000	0.000000
8.000000			
75%	8.920000	4.000000	0.000000
10.000000			
max	10.000000	5.000000	4.000000
12.000000			

	Depression
count	27901.000000
mean	0.585499
std	0.492645
min	0.000000
25%	0.000000
50%	1.000000
75%	1.000000
max	1.000000

```
len(df)
```

```
27901
```

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

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
Financial Stress	0
Family History of Mental Illness	0
Depression	0

```
dtype: int64
```

```
missing_percentage = df.isnull().mean()*100
```

```
threshold = 30
```

```
#Filter columns with more than the threshold percentage of missing values
```

```
columns_to_drop = missing_percentage[missing_percentage >
```

```

threshold].index

#Drop the columns
df_cleaned = df.drop(columns=columns_to_drop)

#Display the resulting dataframe after removing columns with too many
missing values
print(f"columns dropped: {list(columns_to_drop)}")
print(f"Remaining columns: {df_cleaned.columns}")

columns dropped: []
Remaining columns: Index(['id', 'Gender', 'Age', 'City', 'Profession',
'Academic Pressure',
'Work Pressure', 'CGPA', 'Study Satisfaction', 'Job
Satisfaction',
'Sleep Duration', 'Dietary Habits', 'Degree',
'Have you ever had suicidal thoughts ?', 'Work/Study Hours',
'Financial Stress', 'Family History of Mental Illness',
'Depression'],
dtype='object')

from sklearn.preprocessing import LabelEncoder
label_encoder = LabelEncoder()
for column in df.select_dtypes(include=['object']).columns:
    df[column] = label_encoder.fit_transform(df[column])
df.head()
{"type": "dataframe", "variable_name": "df"}
#fill missing values in numeric columns with mean
for column in df.select_dtypes(include=['float64']).columns:
    mean_value = df[column].mean()
    df[column] = df[column].fillna(mean_value)

#fill missing values in numeric columns with mode
for column in df.select_dtypes(include=['object']).columns:
    mode_value = df[column].mode()[0]#Get the mode(most frequent
value)
    df[column] = df[column].fillna(mode_value)
df.isnull().sum()

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
Financial Stress        0
Family History of Mental Illness      0
Depression              0
dtype: int64

```

```

import numpy as np
from matplotlib import pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')

```

Matplotlib is building the font cache; this may take a moment.

#REMOVE OUTLIERS

```
data=df.copy()
```

```

def detect_outliers_iqr(column):
    Q1 = column.quantile(0.25)
    Q3 = column.quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR
    #Return a boolean series indicating if the value is an outlier
    return (column < lower_bound) | (column > upper_bound)

```

#Apply the IQR method to each feature in the Dataframe

```

for col in data.columns:
    data[f'{col}_outlier'] = detect_outliers_iqr(data[col])

```

#Filter the Dataframe to display rows where any outlier flag is True

```
outliers_only = data[data.filter(like='outlier').any(axis=1)]
```

#Display the filtered dataframe with outliers

```
outliers_only
```

	id	Gender	Age	City	Profession	Academic Pressure \
113	609	1	21.0	4	0	5.0
292	1428	1	29.0	44	12	3.0
422	2105	0	31.0	48	5	5.0
1845	9483	0	32.0	19	4	3.0
1963	10030	0	30.0	25	12	3.0
...	...	...	...	...	...	...
26289	132550	1	25.0	51	13	5.0
26719	134830	0	24.0	29	12	0.0

27109	136642	1	29.0	25	11	4.0
27334	137827	0	48.0	45	12	1.0
27873	140522	0	22.0	50	5	3.0
	Work Pressure	CGPA	Study Satisfaction	Job Satisfaction	...	
\						
113	0.0	7.83		1.0	0.0	...
292	0.0	9.63		3.0	3.0	...
422	0.0	6.95		3.0	0.0	...
1845	0.0	9.60		2.0	0.0	...
1963	0.0	5.42		2.0	0.0	...
...	...	...		...	...	...
26289	0.0	7.25		1.0	0.0	...
26719	0.0	0.00		0.0	2.0	...
27109	0.0	5.77		4.0	0.0	...
27334	0.0	8.92		4.0	0.0	...
27873	0.0	9.13		4.0	0.0	...
	Study Satisfaction_outlier	Job Satisfaction_outlier		\		
113		False		False		
292		False		True		
422		False		False		
1845		False		False		
1963		False		False		
...		...		...		
26289		False		False		
26719		False		True		
27109		False		False		
27334		False		False		
27873		False		False		
	Sleep Duration_outlier	Dietary Habits_outlier	Degree_outlier			
\						
113	False		False		False	
292	False		False		False	
422	False		False		False	
1845	False		False		False	

1963	True	False	False
...	...	...	...
26289	False	False	False
26719	False	False	False
27109	False	False	False
27334	False	False	False
27873	False	False	False
Have you ever had suicidal thoughts ?_outlier \			
113		False	
292		False	
422		False	
1845		False	
1963		False	
...		...	
26289		False	
26719		False	
27109		False	
27334		False	
27873		False	
Work/Study Hours_outlier Financial Stress_outlier \			
113	False	False	
292	False	False	
422	False	False	
1845	False	False	
1963	False	False	
...	...	...	
26289	False	False	
26719	False	False	
27109	False	False	
27334	False	False	
27873	False	False	
Family History of Mental Illness_outlier Depression_outlier			
113		False	False
292		False	False
422		False	False
1845		False	False
1963		False	False
...		...	...
26289		False	False

26719	False	False
27109	False	False
27334	False	False
27873	False	False

[72 rows x 36 columns]

```
plt.figure(figsize=(12, 6))
```

```
for i, col in enumerate(df.columns[:3]): #only plotting the first 3 features
```

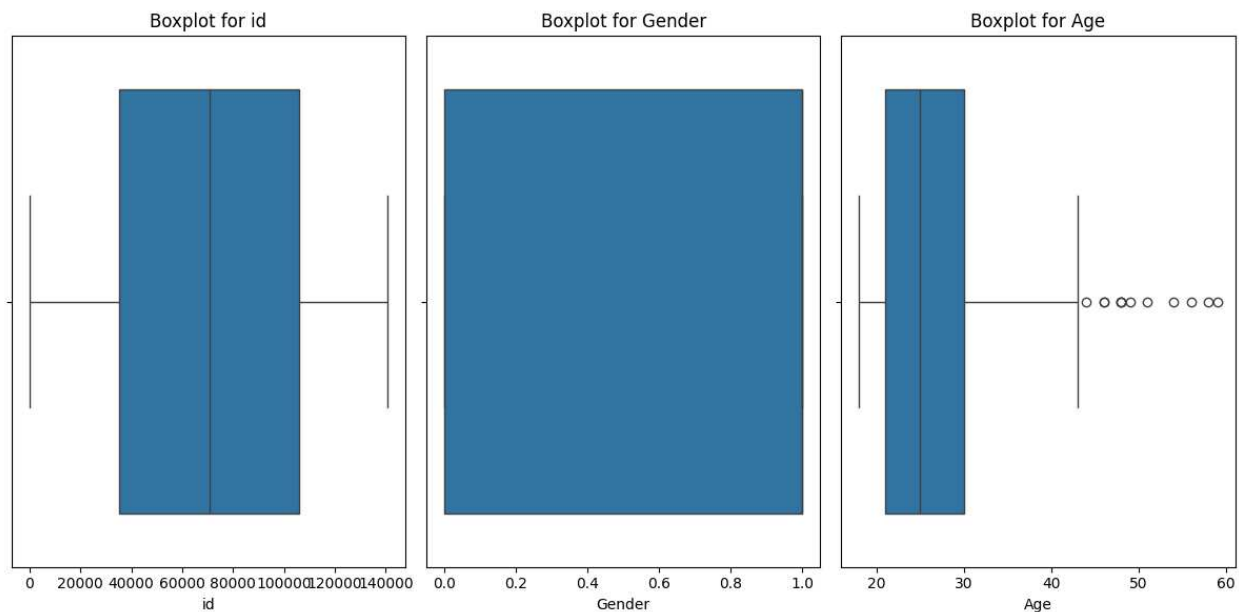
```
    plt.subplot(1, 3, i+1) #create subplots for each feature
```

```
    sns.boxplot(x=df[col])
```

```
    plt.title(f"Boxplot for {col}")
```

```
plt.tight_layout()
```

```
plt.show()
```



```
#function to detect outliers in a column using Z-score
```

```
data=df.copy()
```

```
def detect_outliers_zscore(column, threshold=3):
```

```
    mean = np.mean(column)
```

```
    std_dev = np.std(column)
```

```
    z_scores = (column - mean) / std_dev
```

```
    #Return a boolean series where True indicates an outlier
```

```
    return np.abs(z_scores) > threshold
```

```
#Apply the Z-score method to each feature in the Dataframe
```

```
for col in data.columns:
```

```
    data[f'{col}_outlier'] = detect_outliers_zscore(data[col])
```

```
#filter the dataframe to display rows where any outlier flag is True
outliers_only = data[data.filter(like='_outlier').any(axis=1)]
```

```
outliers_only
```

	id	Gender	Age	City	Profession	Academic Pressure	\
113	609	1	21.0	4	0	5.0	
292	1428	1	29.0	44	12	3.0	
422	2105	0	31.0	48	5	5.0	
1076	5569	0	42.0	40	12	2.0	
1845	9483	0	32.0	19	4	3.0	
...	...	...	...	...	...	...	
26289	132550	1	25.0	51	13	5.0	
26719	134830	0	24.0	29	12	0.0	
27109	136642	1	29.0	25	11	4.0	
27334	137827	0	48.0	45	12	1.0	
27873	140522	0	22.0	50	5	3.0	
	Work Pressure	CGPA	Study Satisfaction	Job Satisfaction	...		
\							
113	0.0	7.83		1.0	0.0	...	
292	0.0	9.63		3.0	3.0	...	
422	0.0	6.95		3.0	0.0	...	
1076	0.0	9.03		5.0	0.0	...	
1845	0.0	9.60		2.0	0.0	...	
...	...	...		...	...	...	
26289	0.0	7.25		1.0	0.0	...	
26719	0.0	0.00		0.0	2.0	...	
27109	0.0	5.77		4.0	0.0	...	
27334	0.0	8.92		4.0	0.0	...	
27873	0.0	9.13		4.0	0.0	...	
	Study Satisfaction_outlier	Job Satisfaction_outlier	\				
113	False	False					
292	False	True					
422	False	False					
1076	False	False					
1845	False	False					
...	...	...					

26289	False	False
26719	False	True
27109	False	False
27334	False	False
27873	False	False

	Sleep Duration_outlier	Dietary Habits_outlier	Degree_outlier
\			
113	False	False	False
292	False	False	False
422	False	False	False
1076	False	False	False
1845	False	False	False
...	...	...	...
26289	False	False	False
26719	False	False	False
27109	False	False	False
27334	False	False	False
27873	False	False	False

	Have you ever had suicidal thoughts ?_outlier	\
113	False	
292	False	
422	False	
1076	False	
1845	False	
...	...	
26289	False	
26719	False	
27109	False	
27334	False	
27873	False	

	Work/Study Hours_outlier	Financial Stress_outlier	\
113	False	False	
292	False	False	
422	False	False	
1076	False	False	
1845	False	False	
...	...	...	

26289	False	False
26719	False	False
27109	False	False
27334	False	False
27873	False	False

	Family History of Mental Illness_outlier	Depression_outlier
113	False	False
292	False	False
422	False	False
1076	False	False
1845	False	False
...	...	...
26289	False	False
26719	False	False
27109	False	False
27334	False	False
27873	False	False

[61 rows x 36 columns]

```
data=df.copy()
def replace_outliers_with_nan(column):
    Q1 = column.quantile(0.25)
    Q3 = column.quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR
    #Replace outliers with Nan
    return column.where((column >= lower_bound) & (column <=
upper_bound), np.nan)

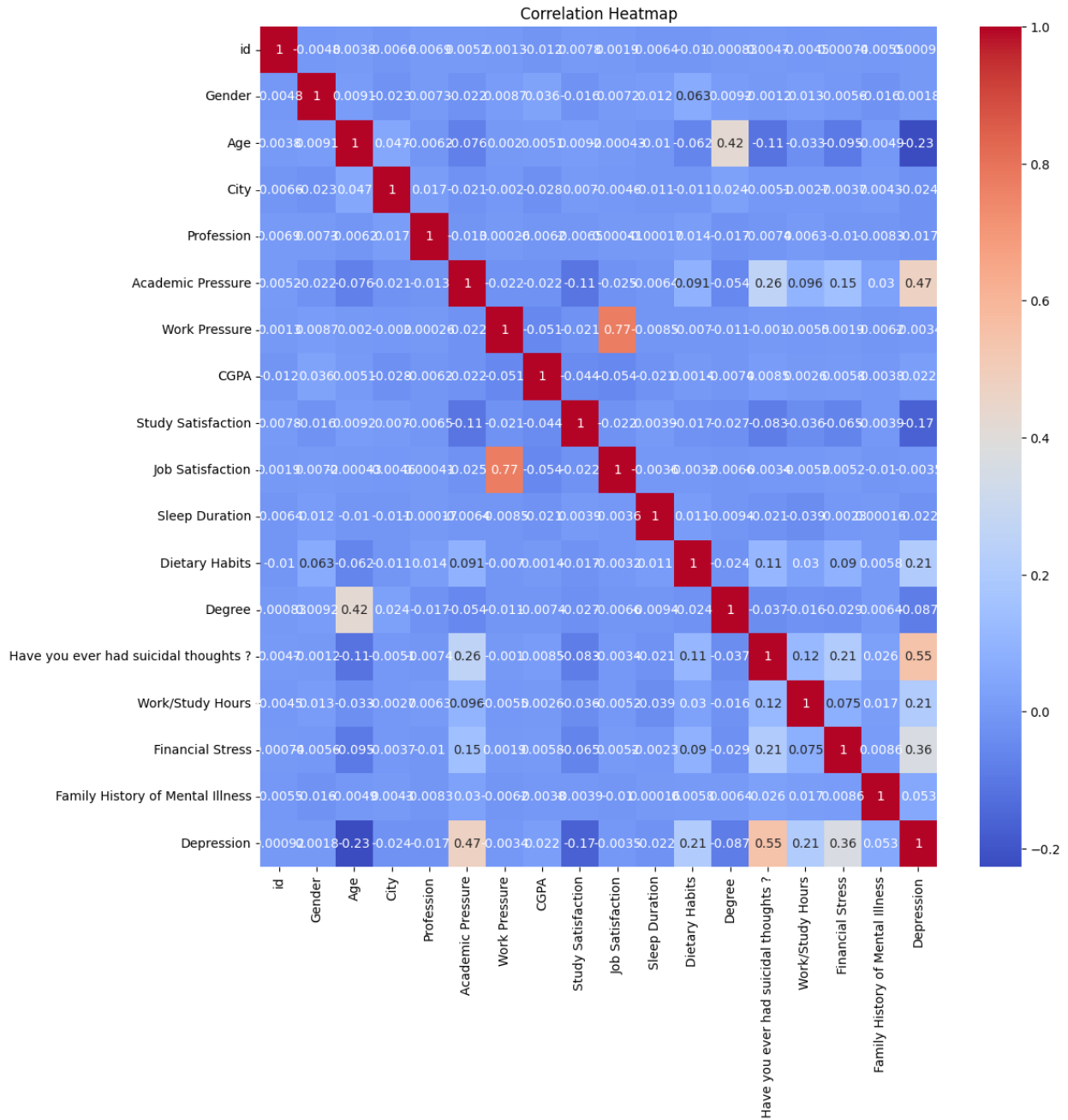
#Replace outliers with Nan for each feature
for col in data.columns:
    data[col] = replace_outliers_with_nan(data[col])

data=df.copy()
def remove_outliers_iqr(data):
    #calculate Q1 (25th percentile) and Q3 (75th percentile) for each
feature
    Q1 = data.quantile(0.25)
    Q3 = data.quantile(0.75)
    IQR = Q3 - Q1

    #define upper and lower bounds
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR

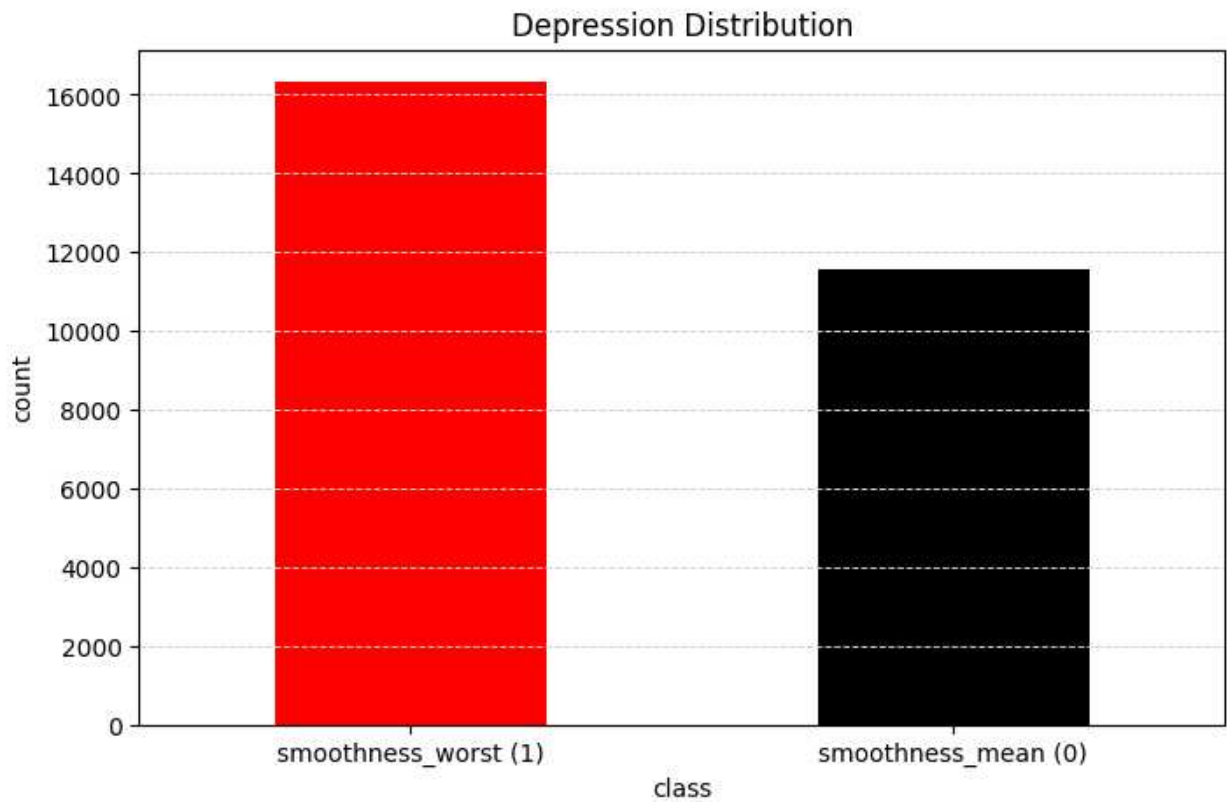
    #filter the dataset to remove rows with outliers in any column
    data_cleaned = data[~((data < lower_bound) | (data >
```

```
upper_bound)).any(axis=1)]  
  
    return data_cleaned  
  
cleaned_data = remove_outliers_iqr(data)  
  
print("Original Data:")  
print(len(data))  
print("\nCleaned Data(without outliers):")  
print(len(cleaned_data))  
  
Original Data:  
27901  
  
Cleaned Data(without outliers):  
27829  
  
plt.figure(figsize=(12, 12))  
sns.heatmap(df.corr(),annot=True, cmap='coolwarm')  
plt.title('Correlation Heatmap')  
plt.show()
```

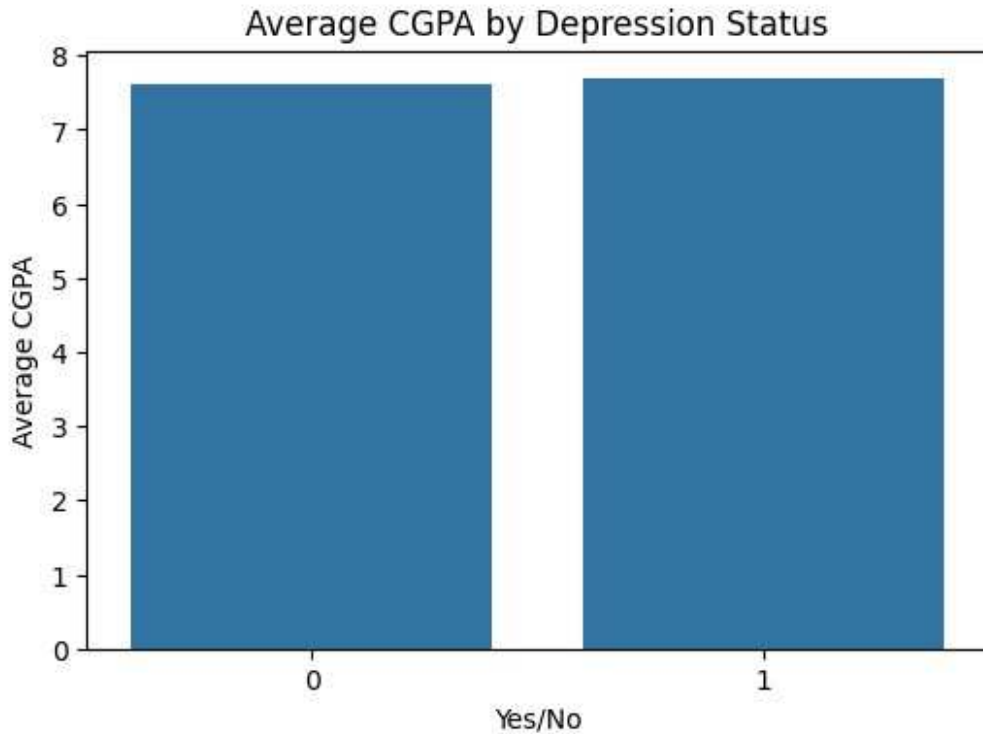


```
df['Depression'].value_counts()
plt.figure(figsize=(8, 5))
df['Depression'].value_counts().plot(kind='bar', color=['red' ,
'black'])
plt.xticks([0,1], ['smoothness_worst (1)', 'smoothness_mean
(0)'],rotation=0)
plt.title('Depression Distribution')
plt.xlabel('class')
plt.ylabel('count')
```

```
plt.grid(axis='y', color='lightgray', linestyle='--', linewidth=0.7)
plt.show()
```



```
import seaborn as sns
plt.figure(figsize=(6, 4))
sns.barplot(x='Depression', y='CGPA', data=df, estimator='mean',
ci=None)
plt.title("Average CGPA by Depression Status")
plt.xlabel("Yes/No")
plt.ylabel("Average CGPA")
plt.show()
```

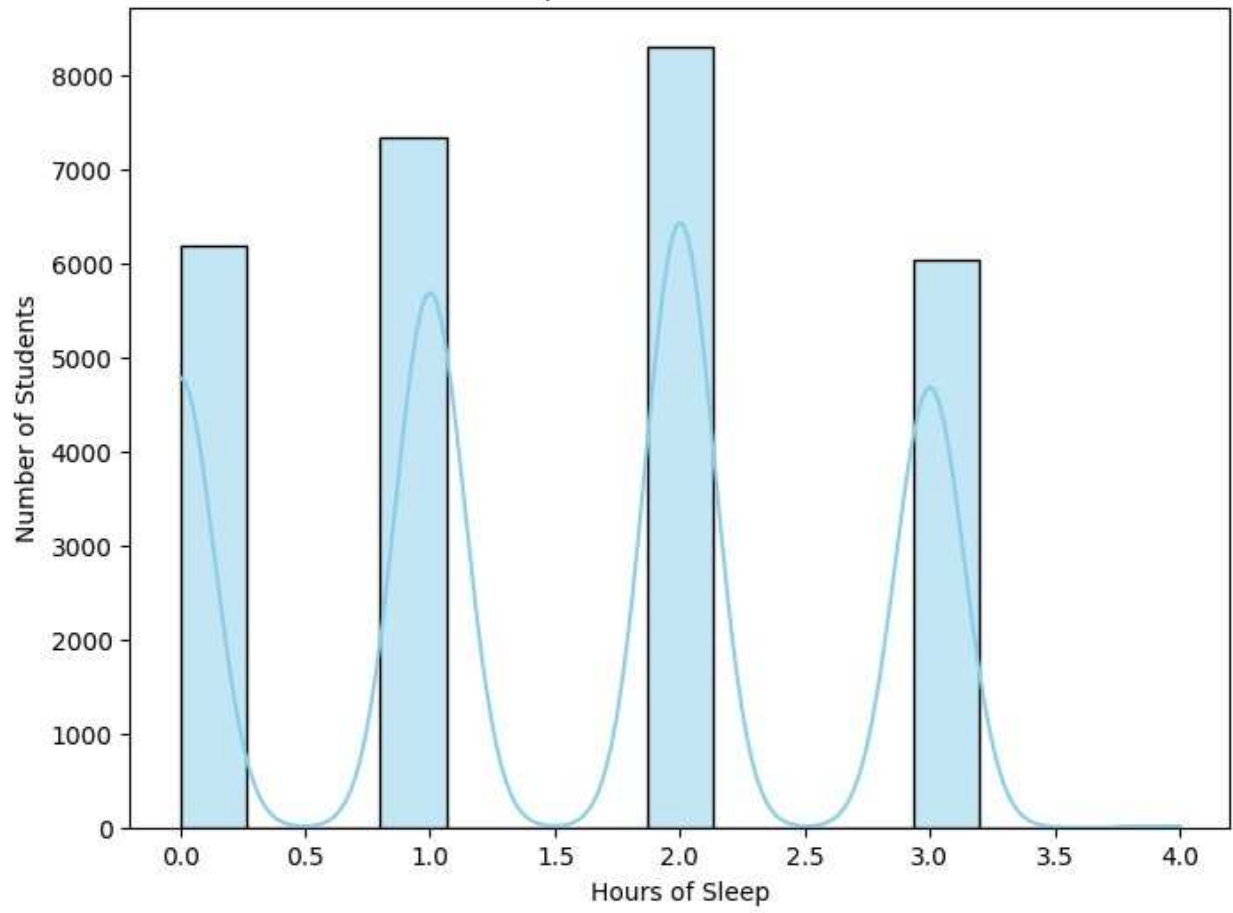


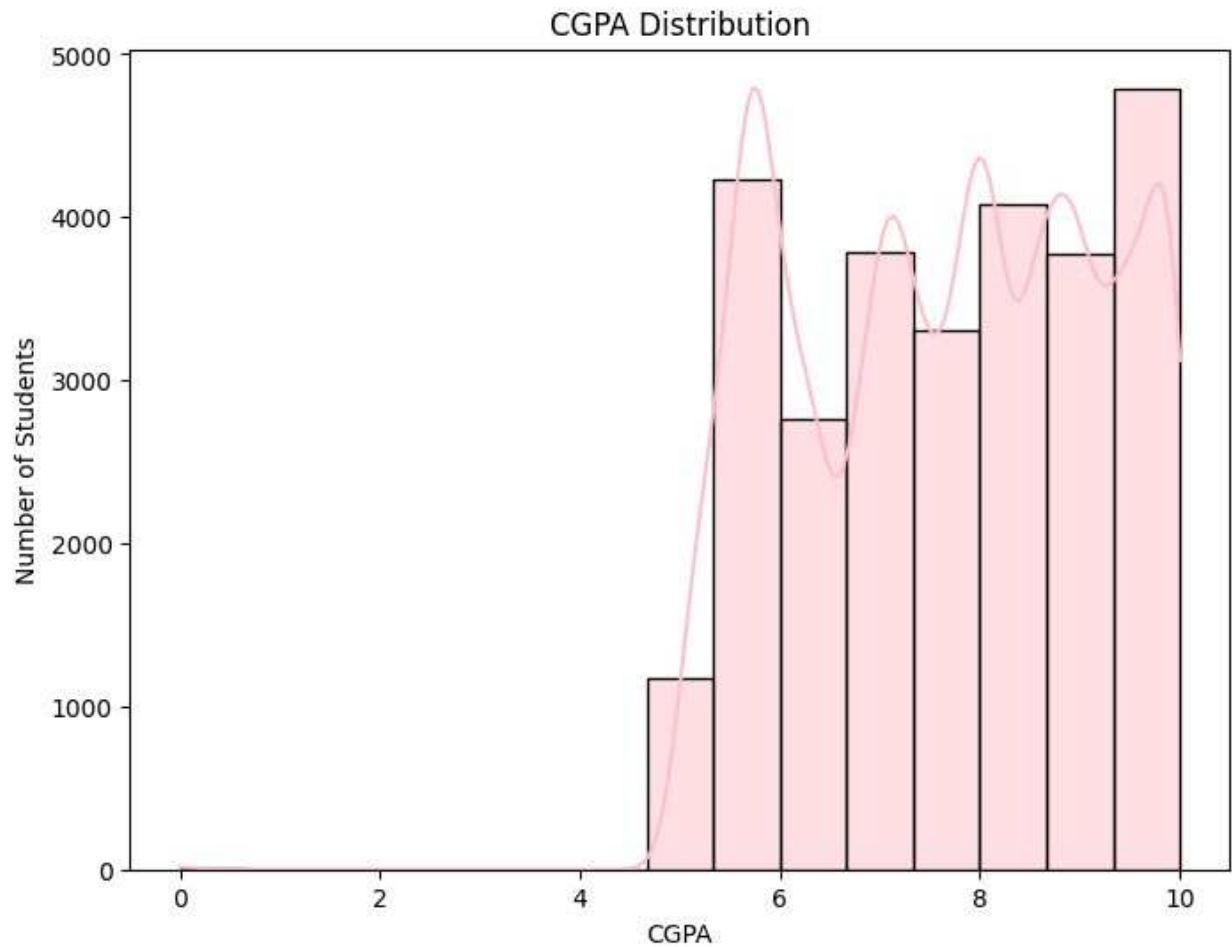
```
cat_cols = [col for col in df.columns if df[col].dtype == 'object']
num_cols = [col for col in df.columns if df[col].dtype != 'object']

for col in cat_cols:
    print(f"{col} has {df[col].unique()}")
plt.figure(figsize=(8,6))
sns.histplot(df['Sleep Duration'], bins=15, kde=True, color='skyblue')
plt.title('Sleep Duration Distribution')
plt.xlabel('Hours of Sleep')
plt.ylabel('Number of Students')
plt.show()

plt.figure(figsize=(8, 6))
sns.histplot(df['CGPA'], bins=15, kde=True, color='pink')
plt.title('CGPA Distribution')
plt.xlabel('CGPA')
plt.ylabel('Number of Students')
plt.show()
```

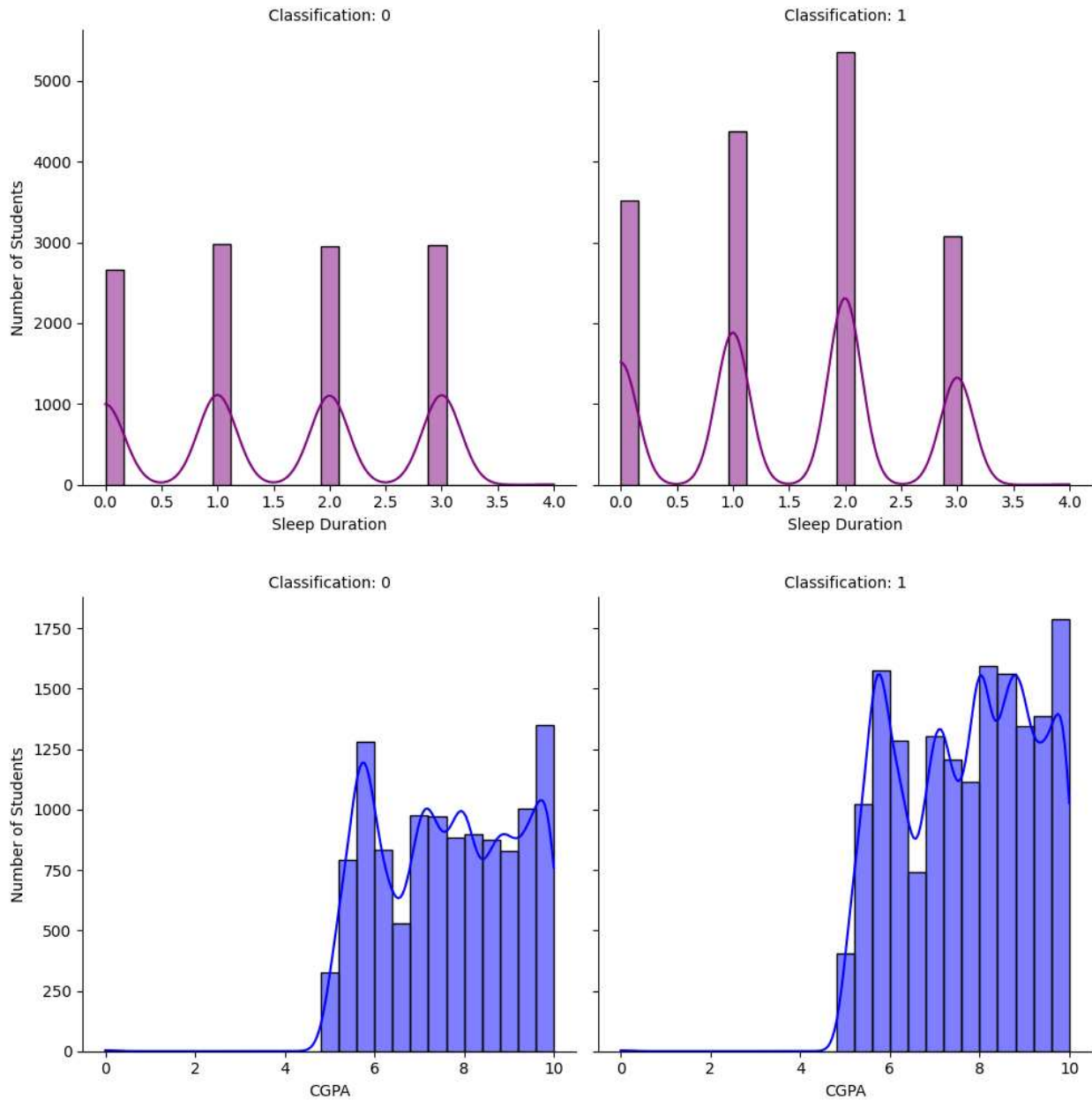
Sleep Duration Distribution





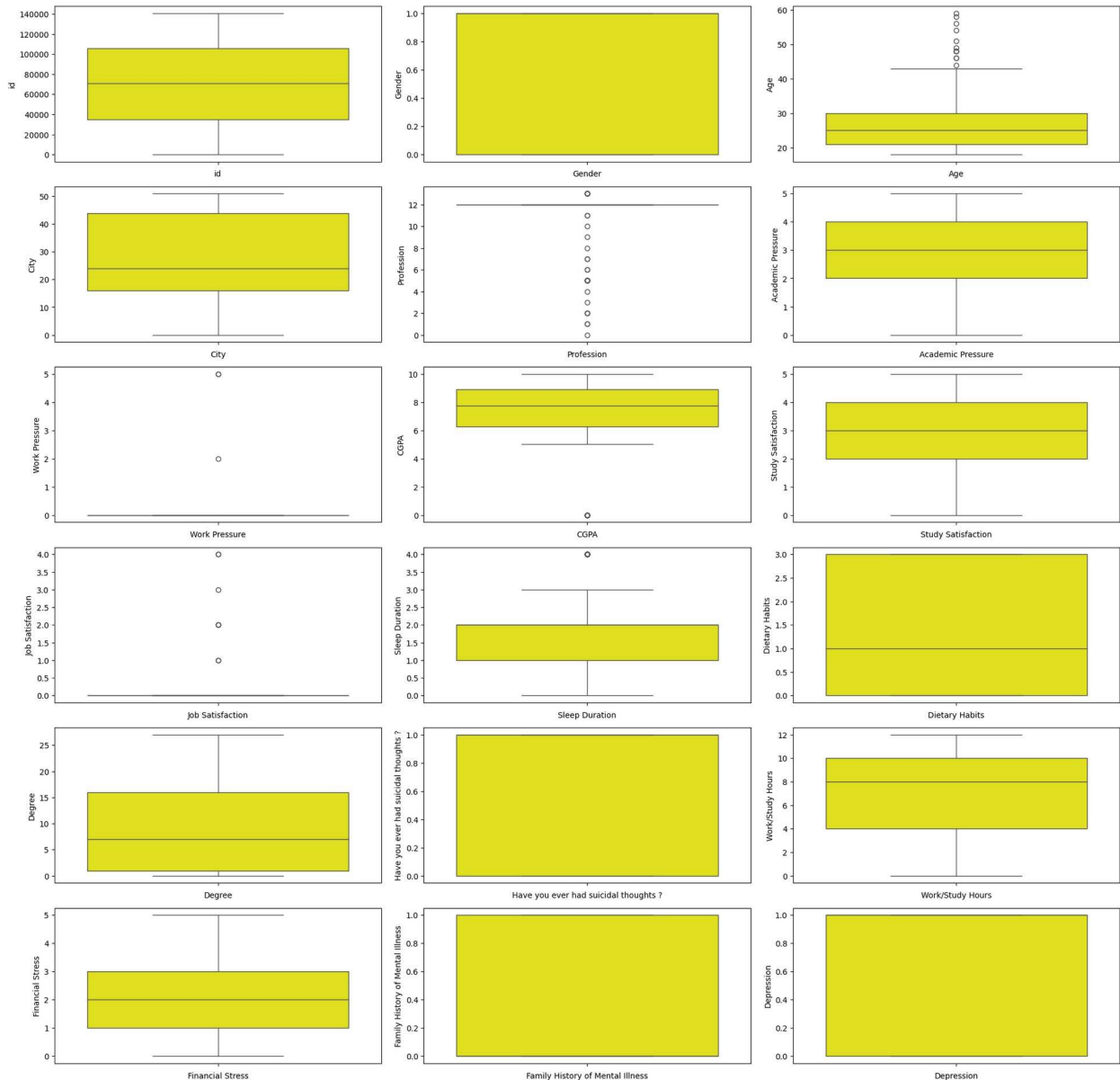
```
g = sns.FacetGrid(df, col='Depression', height=5, aspect=1)
g.map(sns.histplot, 'Sleep Duration', bins=25, kde=True,
color='purple')
g.set_axis_labels('Sleep Duration', "Number of Students")
g.set_titles("Classification: {col_name}")
plt.show()
```

```
g = sns.FacetGrid(df, col='Depression', height=5, aspect=1)
g.map(sns.histplot, 'CGPA', bins=25, kde=True, color='blue')
g.set_axis_labels('CGPA', "Number of Students")
g.set_titles("Classification: {col_name}")
plt.show()
```



```
plt.figure(figsize=(20, 35))
plotnumber = 1
num_cols = [col for col in df.columns if df[col].dtype != 'object']

for column in num_cols:
    ax = plt.subplot(11, 3, plotnumber) #Changed 10 to 11 accomodate more plots
    sns.boxplot(y=df[column], color='yellow')
    plt.xlabel(column)
    plotnumber += 1
plt.tight_layout()
plt.show()
```



```

colors = ['lightblue', 'turquoise']
labels = ['Depression_Yes (0)', 'Depression_No (1)']

order = df['Depression'].value_counts().index

plt.figure(figsize=(16, 8))
plt.suptitle('Depression Distribution', fontweight='bold',
fontsize=16)

plt.subplot(1, 2, 1)
plt.title('Pie Chart', fontsize=14)
plt.pie(df['Depression'].value_counts(), labels=labels, colors=colors,
autopct='%.2f%%', startangle=90)
centre = plt.Circle((0, 0), 0.45, color='white')

```

```

plt.gca().add_artist(centre)

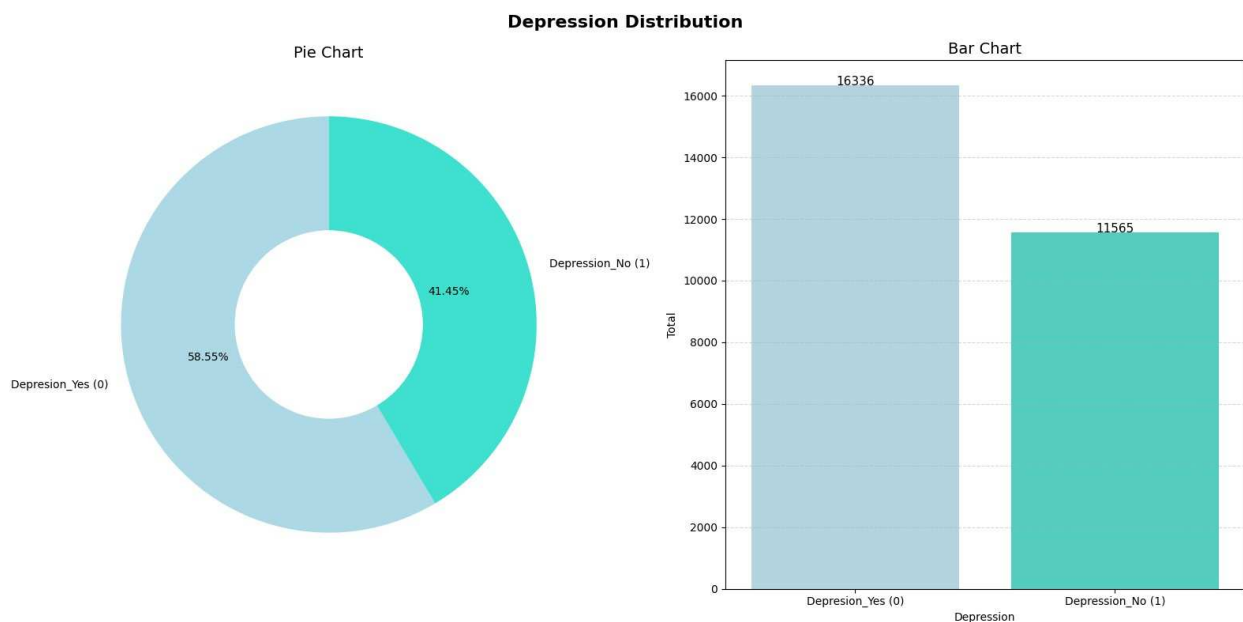
plt.subplot(1, 2, 2)
plt.title('Bar Chart', fontsize=14)
ax = sns.countplot(x='Depression', data=df, palette=colors,
order=order)

for bar in ax.patches:
    height = bar.get_height()
    ax.text(bar.get_x() + bar.get_width()/2, height + 1, int(height),
ha='center', fontsize=11)

plt.xlabel('Depression')
plt.ylabel('Total')
plt.xticks([0, 1], labels)
plt.grid(axis='y', linestyle='--', alpha=0.5)

plt.tight_layout()
plt.show()

```

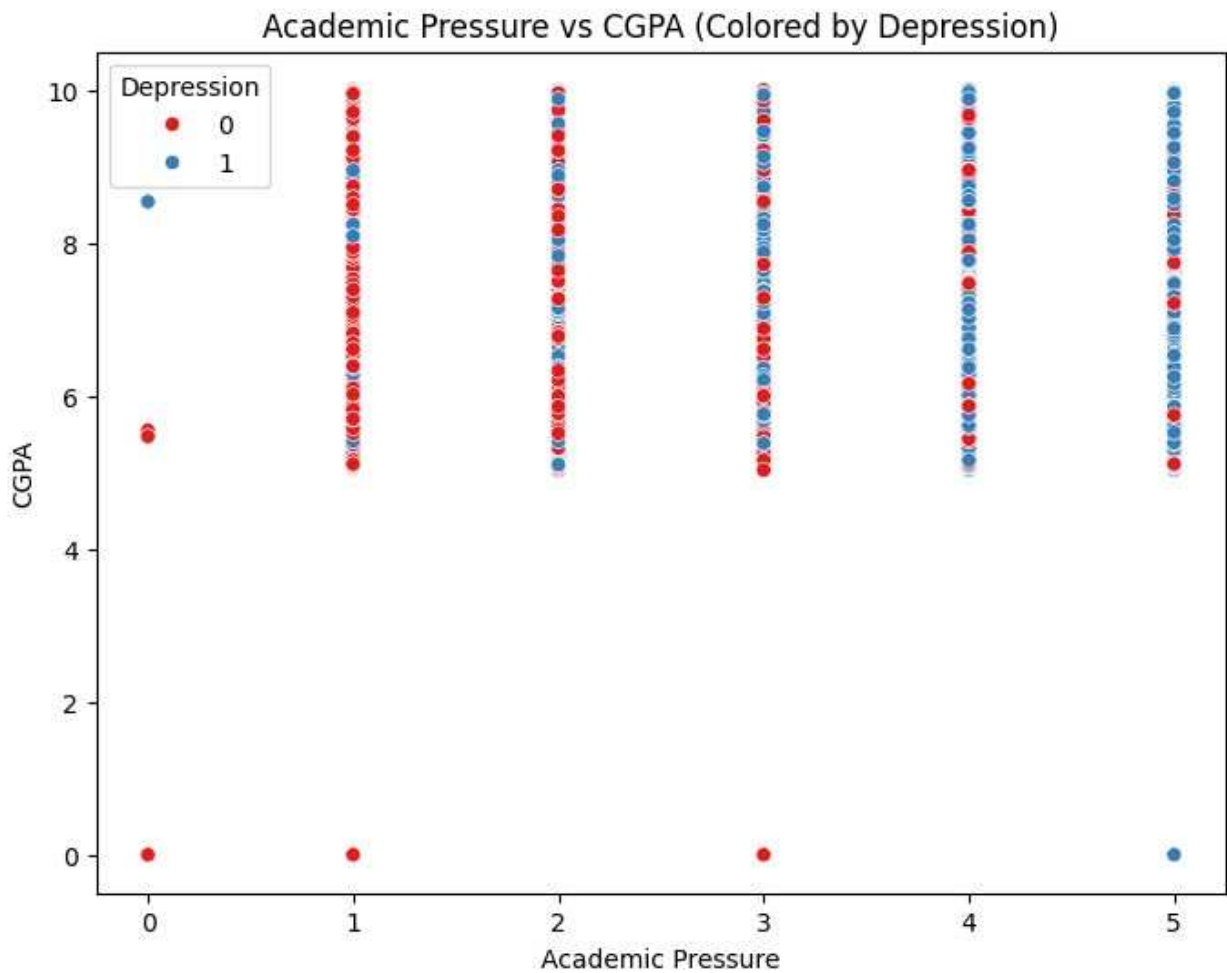


```

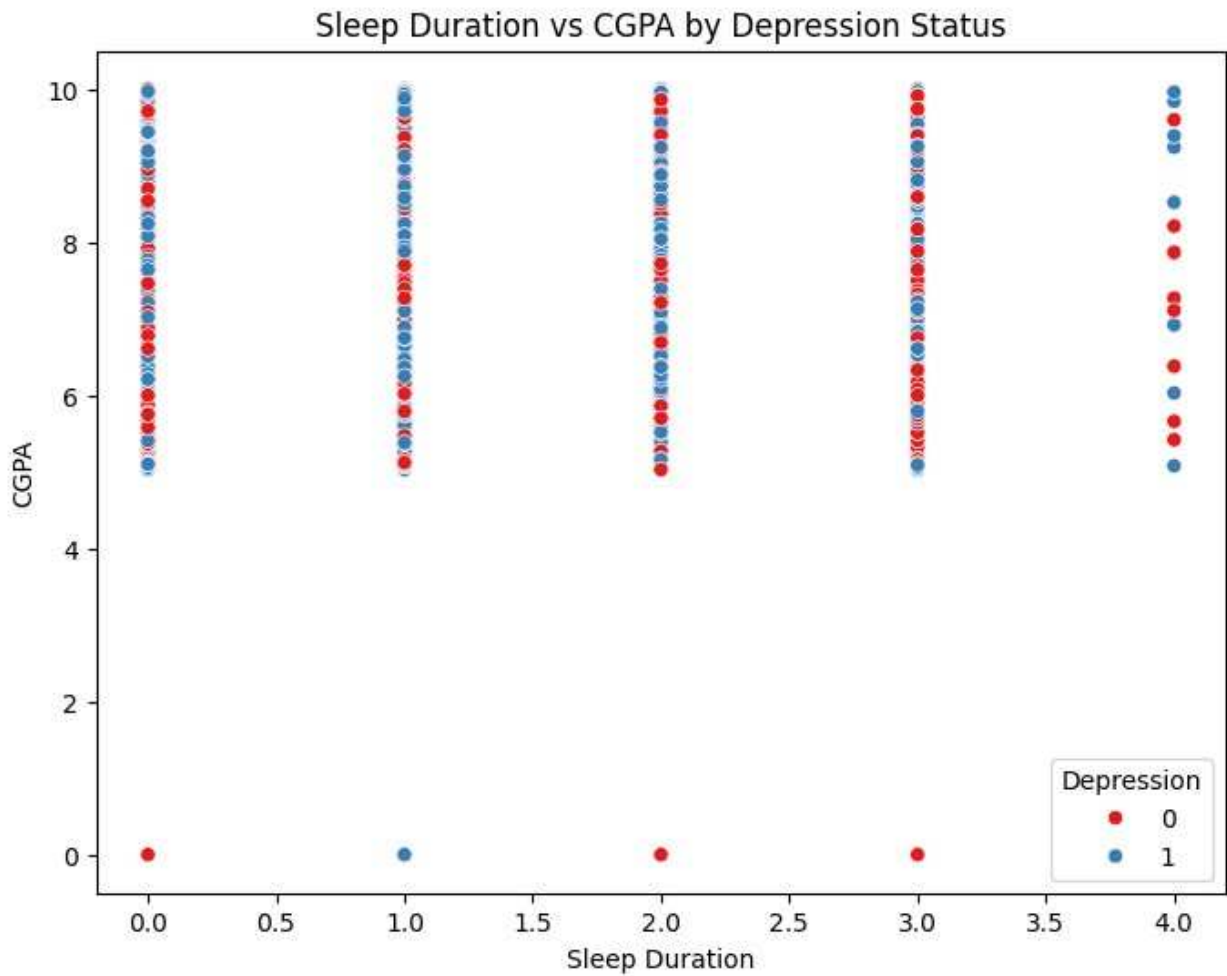
plt.figure(figsize=(8, 6))
sns.scatterplot(
    x='Academic Pressure',
    y='CGPA',
    data=df,
    hue='Depression',
    palette='Set1'
)
plt.title('Academic Pressure vs CGPA (Colored by Depression)')
plt.xlabel('Academic Pressure')

```

```
plt.ylabel('CGPA')
plt.show()
```



```
plt.figure(figsize=(8, 6))
sns.scatterplot(
    x='Sleep Duration',
    y='CGPA',
    data=df,
    hue='Depression',
    palette='Set1'
)
plt.title('Sleep Duration vs CGPA by Depression Status')
plt.xlabel('Sleep Duration')
plt.ylabel('CGPA')
plt.show()
```



```
plt.figure(figsize=(8, 6))
sns.boxplot(
    x='Depression',
    y='CGPA',
    data=df,
    palette='Set1'
)
plt.title('Depression levels by CGPA')
plt.xlabel('Depression')
plt.ylabel('CGPA')
plt.show()
```

