Step 1: Libraries Import

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
# Graphs notebook mein show
%matplotlib inline

# Plot style
sns.set(style='whitegrid')
```

Step 2: Dataset Load

```
df = pd.read_csv('/content/enhanced_student_habits_performance_dataset.csv')
```

Step 3: Dataset Size or Column Names check

```
df.shape
```

```
(80000, 31)
```

```
df.columns
```

```
Index(['student_id', 'age', 'gender', 'major', 'study_hours_per_day',
       'social_media_hours', 'netflix_hours', 'part_time_job',
       'attendance_percentage', 'sleep_hours', 'diet_quality',
       'exercise_frequency', 'parental_education_level', 'internet_quality',
       'mental_health_rating', 'extracurricular_participation', 'previous_gpa',
       'semester', 'stress_level', 'dropout_risk', 'social_activity',
       'screen_time', 'study_environment', 'access_to_tutoring',
       'family_income_range', 'parental_support_level', 'motivation_level',
       'exam_anxiety_score', 'learning_style', 'time_management_score',
       'exam_score'],
      dtype='object')
```

Step 4: Data Type Check

```
df.dtypes
```

	0
student_id	int64
age	int64
gender	category
major	category
study_hours_per_day	float64
social_media_hours	float64
netflix_hours	float64
part_time_job	category
attendance_percentage	float64
sleep_hours	float64
diet_quality	float64
exercise_frequency	int64
parental_education_level	category
internet_quality	category
mental_health_rating	float64
extracurricular_participation	category
previous_gpa	float64
semester	int64
stress_level	float64
dropout_risk	category
social_activity	int64
screen_time	float64
study_environment	category
access_to_tutoring	category
family_income_range	category
parental_support_level	int64
motivation_level	int64
exam_anxiety_score	int64
learning_style	category

```
time_management_score    float64
           exam_score      int64

dtype: object
```

```
# Convert categorical columns to 'category' type
df['gender'] = df['gender'].astype('category')
df['major'] = df['major'].astype('category')
df['part_time_job'] = df['part_time_job'].astype('category')
df['parental_education_level'] = df['parental_education_level'].astype('category')
df['internet_quality'] = df['internet_quality'].astype('category')
df['extracurricular_participation'] = df['extracurricular_participation'].astype('category')
df['dropout_risk'] = df['dropout_risk'].astype('category')
df['study_environment'] = df['study_environment'].astype('category')
df['access_to_tutoring'] = df['access_to_tutoring'].astype('category')
df['family_income_range'] = df['family_income_range'].astype('category')
df['learning_style'] = df['learning_style'].astype('category')

# Convert 'diet_quality' to float64
df['diet_quality'] = pd.to_numeric(df['diet_quality'], errors='coerce')
```

Step 5: View the First 5 Rows

```
df.head()
```

	student_id	age	gender	major	study_hours_per_day	social_media_hours	netflix_hours	part_time_job	attendance_percentage	sleep_hours	...	scr
0	100000	26	Male	Computer Science	7.645367	3.0	0.1	Yes	70.3	6.2	...	
1	100001	28	Male	Arts	5.700000	0.5	0.4	No	88.4	7.2	...	
2	100002	17	Male	Arts	2.400000	4.2	0.7	No	82.1	9.2	...	
3	100003	27	Other	Psychology	3.400000	4.6	2.3	Yes	79.3	4.2	...	
4	100004	25	Female	Business	4.700000	0.8	2.7	Yes	62.9	6.5	...	

5 rows × 31 columns

Step 6: Check for Missing Values

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

	0
student_id	0
age	0
gender	0
major	0
study_hours_per_day	0
social_media_hours	0
netflix_hours	0
part_time_job	0
attendance_percentage	0
sleep_hours	0
exercise_frequency	0
parental_education_level	0
internet_quality	0
mental_health_rating	0
extracurricular_participation	0
previous_gpa	0
semester	0
stress_level	0
dropout_risk	0
social_activity	0
screen_time	0
study_environment	0
access_to_tutoring	0
family_income_range	0
parental_support_level	0
motivation_level	0
exam_anxiety_score	0
learning_style	0
time_management_score	0

```
exam_score    0

dtype: int64
```

```
df.drop(columns=['diet_quality'], inplace=True)
```

Step 7: Statistical Summary of Numerical Columns

```
df.describe()
```

	student_id	age	study_hours_per_day	social_media_hours	netflix_hours	attendance_percentage	sleep_hours	exercise_frequency	mental
count	80000.000000	80000.000000	80000.000000	80000.000000	80000.000000	80000.000000	80000.000000	80000.000000	
mean	139999.500000	22.004288	4.174388	2.501366	1.997754	69.967884	7.017417	3.516587	
std	23094.155105	3.745570	2.004135	1.445441	1.155992	17.333015	1.467377	2.291575	
min	100000.000000	16.000000	0.000000	0.000000	0.000000	40.000000	4.000000	0.000000	
25%	119999.750000	19.000000	2.800000	1.200000	1.000000	55.000000	6.000000	2.000000	
50%	139999.500000	22.000000	4.125624	2.500000	2.000000	69.900000	7.000000	4.000000	
75%	159999.250000	25.000000	5.500000	3.800000	3.000000	84.900000	8.000000	6.000000	
max	179999.000000	28.000000	12.000000	5.000000	4.000000	100.000000	12.000000	7.000000	

Step 8: List of Numerical and Categorical Columns

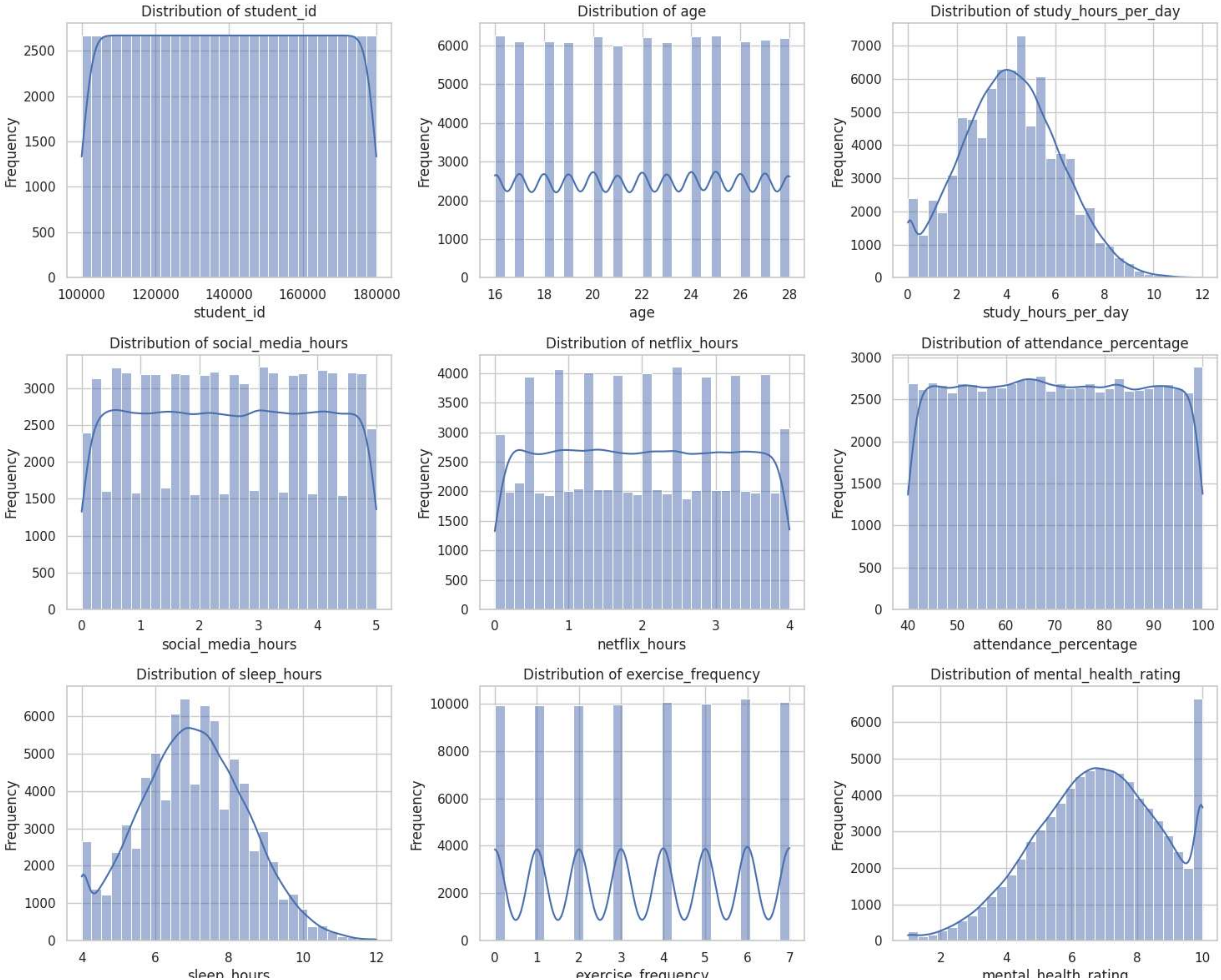
```
# List numerical columns
num_cols = df.select_dtypes(include=['int64', 'float64']).columns.tolist()

# List categorical columns
cat_cols = df.select_dtypes(include=['object']).columns.tolist()

print("\nNumerical Columns:")
print(num_cols)

print("\nCategorical Columns:")
print(cat_cols)
```

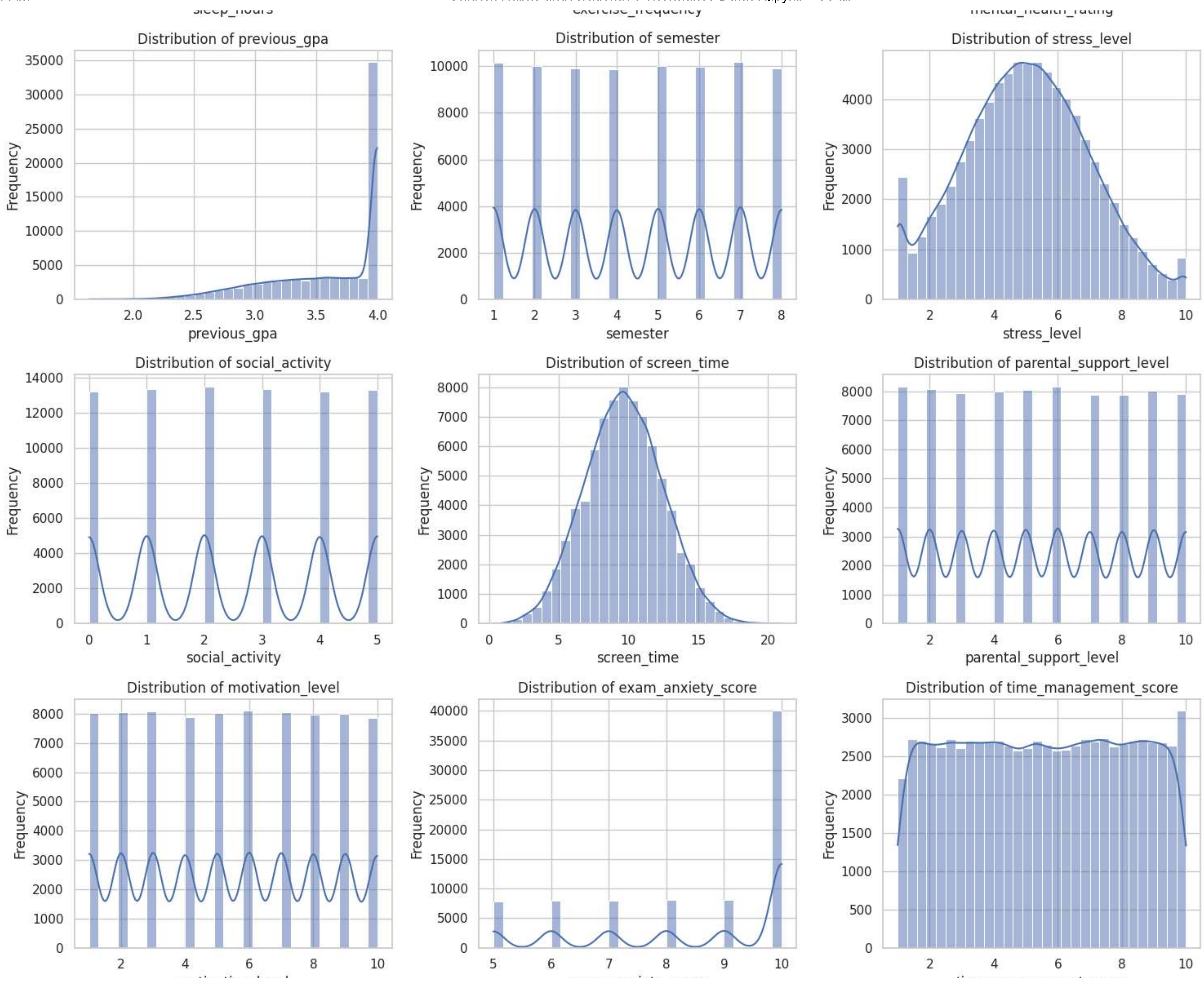
```
Numerical Columns:
```

```
['student_id', 'age', 'study_hours_per_day', 'social_media_hours', 'netflix_hours', 'attendance_percentage', 'sleep_hours', 'exercise_frequency', 'mental

Categorical Columns:
[]
```

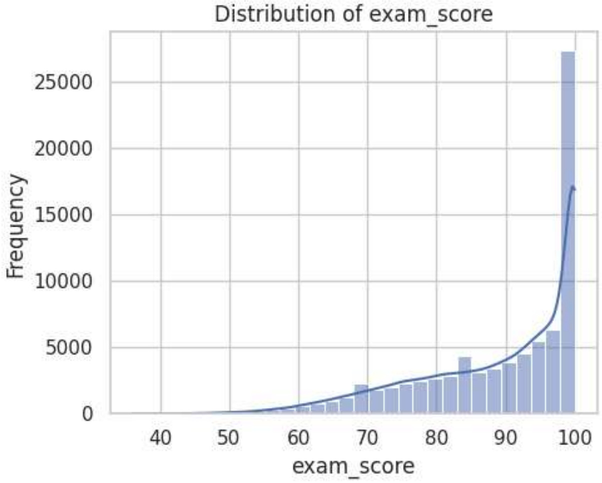
Step 9: Plot Distribution of Numerical Columns

```
import math
# Numerical columns list (agar already defined nahi hai to define kar lo)
num_cols = df.select_dtypes(include=['int64', 'float64']).columns.tolist()

n = len(num_cols)  # Number of numerical columns
cols = 3           # Number of subplot columns (aap change kar sakte hain)
rows = math.ceil(n / cols)  # Rows calculated dynamically

plt.figure(figsize=(15, rows * 4))  # Figure height dynamically adjusted

for i, col in enumerate(num_cols, 1):
    plt.subplot(rows, cols, i)
    sns.histplot(df[col], kde=True, bins=30)
    plt.title(f'Distribution of {col}')
    plt.xlabel(col)
    plt.ylabel('Frequency')

plt.tight_layout()
plt.show()
```


Distribution of student_id
Frequency
0
500
1000
1500
2000
2500
100000
120000
140000
160000
180000
student_id
Distribution of age
Frequency
0
1000
2000
3000
4000
5000
6000
16
18
20
22
24
26
28
age
Distribution of study_hours_per_day
Frequency
0
1000
2000
3000
4000
5000
6000
7000
0
2
4
6
8
10
12
study_hours_per_day
Distribution of social_media_hours
Frequency
0
500
1000
1500
2000
2500
3000
0
1
2
3
4
5
social_media_hours
Distribution of netflix_hours
Frequency
0
500
1000
1500
2000
2500
3000
3500
4000
0
1
2
3
4
netflix_hours
Distribution of attendance_percentage
Frequency
0
500
1000
1500
2000
2500
3000
40
50
60
70
80
90
100
attendance_percentage
Distribution of sleep_hours
Frequency
0
1000
2000
3000
4000
5000
6000
4
6
8
10
12
sleep_hours
Distribution of exercise_frequency
Frequency
0
2000
4000
6000
8000
10000
0
1
2
3
4
5
6
7
exercise_frequency
Distribution of mental_health_rating
Frequency
0
1000
2000
3000
4000
5000
6000
2
4
6
8
10
mental_health_rating

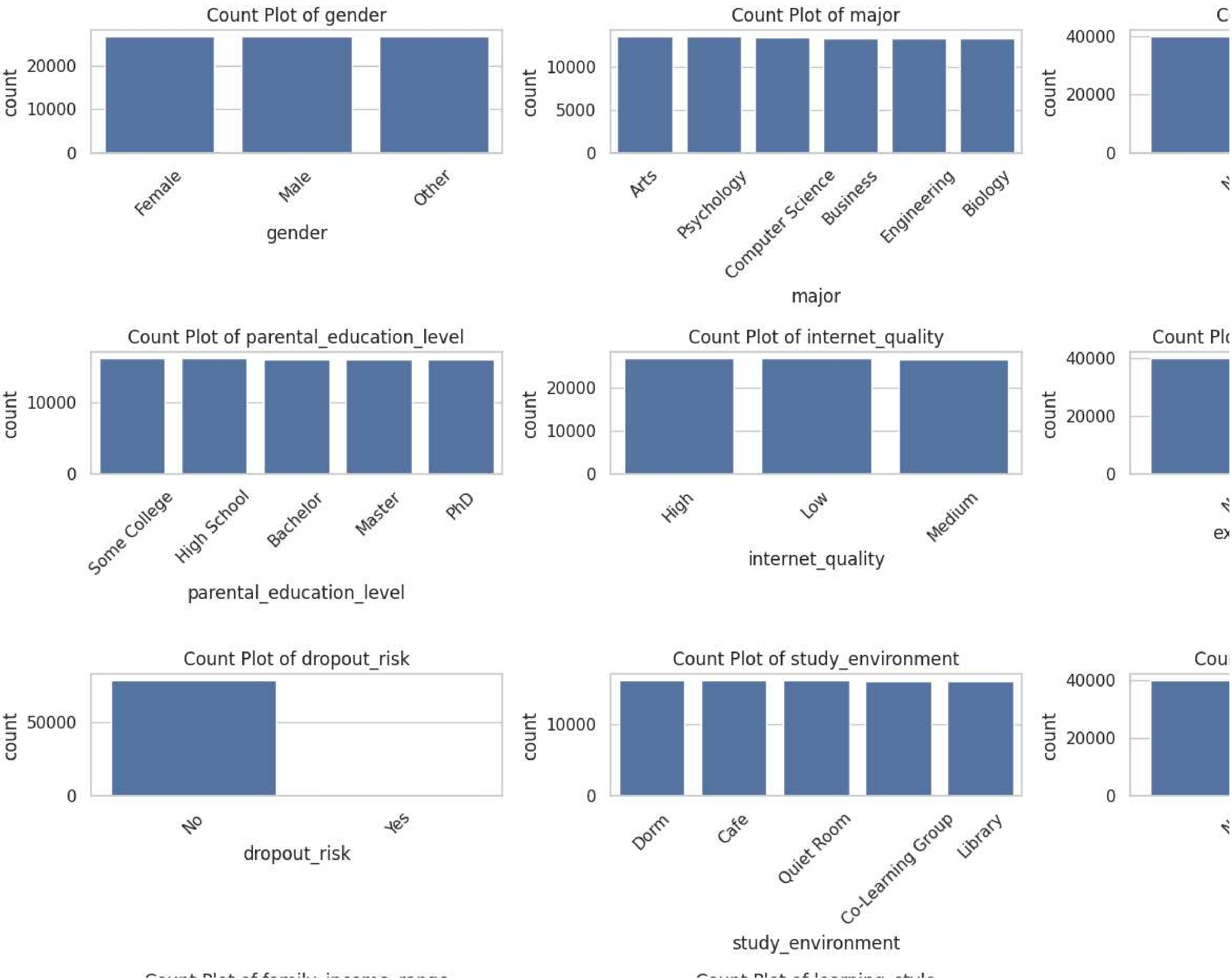

sleep_hours
exercise_frequency
mental_health_rating
Distribution of previous_gpa
Frequency
0
5000
10000
15000
20000
25000
30000
35000
2.0
2.5
3.0
3.5
4.0
previous_gpa
Distribution of semester
Frequency
0
2000
4000
6000
8000
10000
1
2
3
4
5
6
7
8
semester
Distribution of stress_level
Frequency
0
1000
2000
3000
4000
2
4
6
8
10
stress_level
Distribution of social_activity
Frequency
0
2000
4000
6000
8000
10000
12000
14000
0
1
2
3
4
5
social_activity
Distribution of screen_time
Frequency
0
1000
2000
3000
4000
5000
6000
7000
8000
0
5
10
15
20
screen_time
Distribution of parental_support_level
Frequency
0
1000
2000
3000
4000
5000
6000
7000
8000
2
4
6
8
10
parental_support_level
Distribution of motivation_level
Frequency
0
1000
2000
3000
4000
5000
6000
7000
8000
2
4
6
8
10
Distribution of exam_anxiety_score
Frequency
0
5000
10000
15000
20000
25000
30000
35000
40000
5
6
7
8
9
10
Distribution of time_management_score
Frequency
0
500
1000
1500
2000
2500
3000
2
4
6
8
10

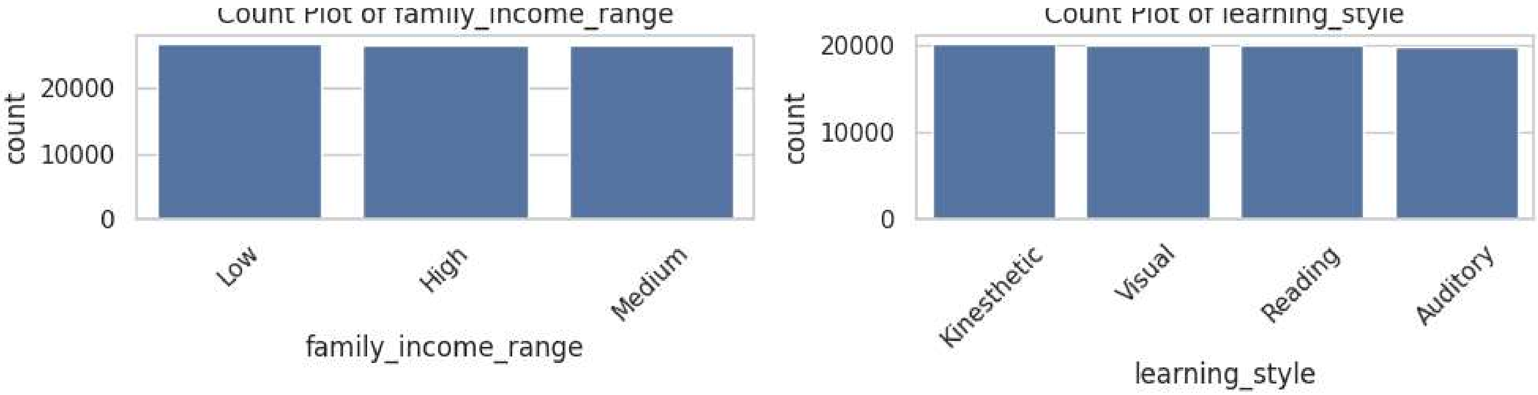
motivation_level exam_anxiety_score time_management_score


Distribution of exam_score

Step 10: Plot Count of Categorical Columns

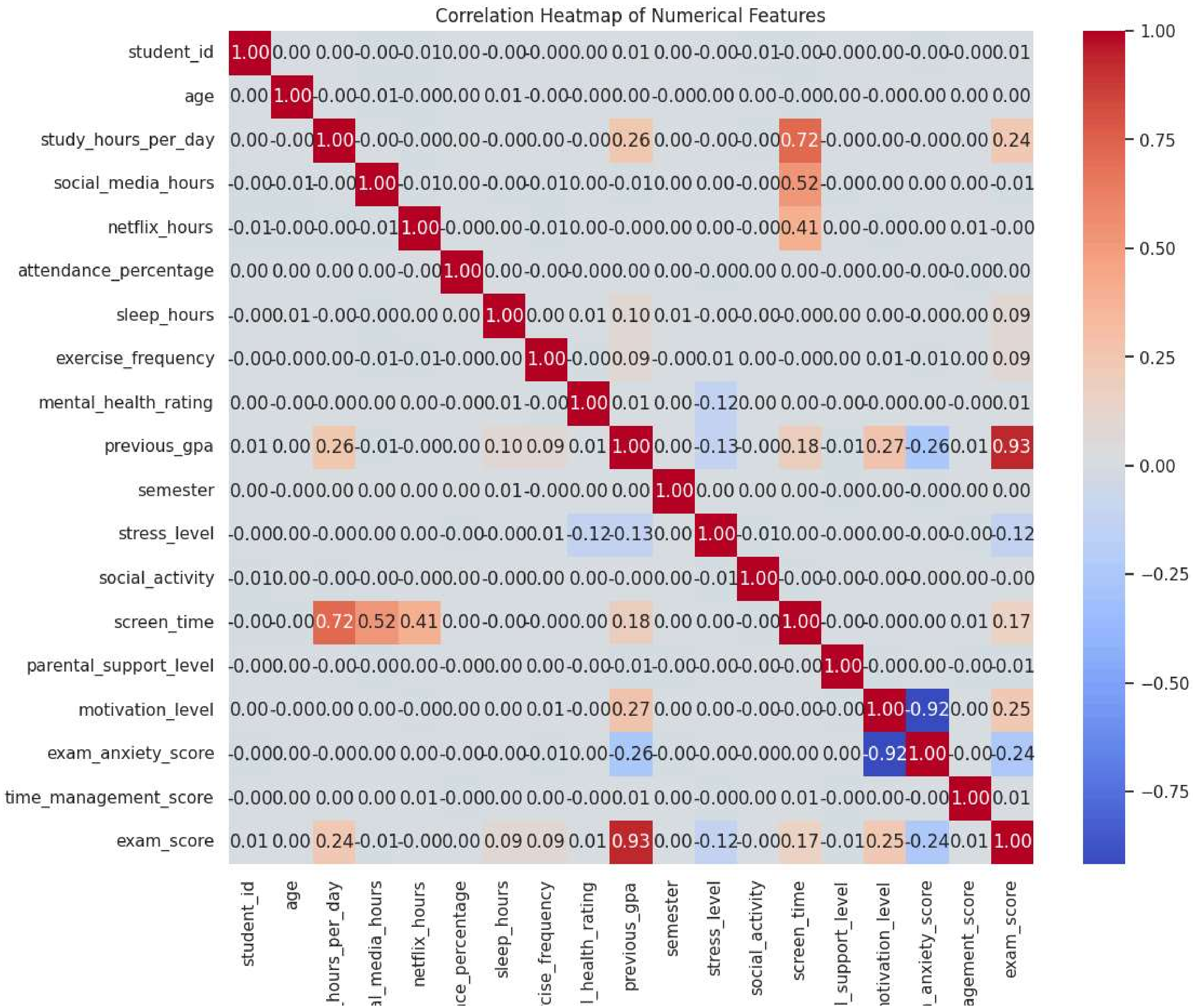
```
# Categorical columns list banayein (agar define nahi kiya hai to)
cat_cols = df.select_dtypes(include=['category']).columns.tolist()

plt.figure(figsize=(15, 12))

for i, col in enumerate(cat_cols, 1):
    plt.subplot(4, 3, i)  # Adjust grid size (4 rows x 3 cols)
    sns.countplot(data=df, x=col, order=df[col].value_counts().index)
    plt.xticks(rotation=45)
    plt.title(f'Count Plot of {col}')

plt.tight_layout()
plt.show()
```


Count Plot of gender
count
20000
10000
0
Female
Male
Other
gender
Count Plot of major
count
10000
5000
0
Arts
Psychology
Computer Science
Business
Engineering
Biology
major
C
40000
20000
0
count
N
Count Plot of parental_education_level
count
10000
0
Some College
High School
Bachelor
Master
PhD
parental_education_level
Count Plot of internet_quality
count
20000
10000
0
High
Low
Medium
internet_quality
Count Plo
40000
20000
0
count
N
ex
Count Plot of dropout_risk
count
50000
0
No
Yes
dropout_risk
Count Plot of study_environment
count
10000
0
Dorm
Cafe
Quiet Room
Co-Learning Group
Library
study_environment
Cou
40000
20000
0
count
N
Count Plot of family_income_range
Count Plot of learning_style



Step 11: Correlation Heatmap for Numerical Columns

```
# Correlation heatmap for numerical columns
plt.figure(figsize=(12,10))
corr = df[num_cols].corr()
sns.heatmap(corr, annot=True, cmap='coolwarm', fmt='.2f')
plt.title("Correlation Heatmap of Numerical Features")
plt.show()
```


Correlation Heatmap of Numerical Features
student_id 1.00 0.00 0.00 -0.00 -0.01 0.00 -0.00 -0.00 0.00 0.01 0.00 -0.00 -0.01 -0.00 -0.00 0.00 -0.00 -0.00 0.01
age 0.00 1.00 -0.00 -0.01 -0.00 0.00 0.01 -0.00 -0.00 0.00 -0.00 0.00 0.00 -0.00 0.00 -0.00 0.00 0.00 0.00
study_hours_per_day 0.00 -0.00 1.00 -0.00 -0.00 0.00 -0.00 0.00 -0.00 0.26 0.00 -0.00 -0.00 0.72 -0.00 0.00 -0.00 0.00 0.24
social_media_hours -0.00 -0.01 -0.00 1.00 -0.01 0.00 -0.00 -0.01 0.00 -0.01 0.00 0.00 -0.00 0.52 -0.00 0.00 0.00 0.00 -0.01
netflix_hours -0.01 -0.00 -0.00 -0.01 1.00 -0.00 0.00 -0.01 0.00 -0.00 0.00 0.00 -0.00 0.41 0.00 -0.00 0.00 0.01 -0.00
attendance_percentage 0.00 0.00 0.00 0.00 -0.00 1.00 0.00 -0.00 -0.00 0.00 0.00 -0.00 0.00 0.00 -0.00 0.00 -0.00 -0.00 0.00
sleep_hours -0.00 0.01 -0.00 -0.00 0.00 0.00 1.00 0.00 0.01 0.10 0.01 -0.00 -0.00 -0.00 0.00 0.00 -0.00 0.00 0.09
exercise_frequency -0.00 -0.00 0.00 -0.01 -0.01 -0.00 0.00 1.00 -0.00 0.09 -0.00 0.01 0.00 -0.00 0.00 0.01 -0.01 0.00 0.09
mental_health_rating 0.00 -0.00 -0.00 0.00 0.00 -0.00 0.01 -0.00 1.00 0.01 0.00 -0.12 0.00 0.00 -0.00 -0.00 0.00 -0.00 0.01
previous_gpa 0.01 0.00 0.26 -0.01 -0.00 0.00 0.10 0.09 0.01 1.00 0.00 -0.13 -0.00 0.18 -0.01 0.27 -0.26 0.01 0.93
semester 0.00 -0.00 0.00 0.00 0.00 0.00 0.01 -0.00 0.00 0.00 1.00 0.00 0.00 0.00 -0.00 0.00 -0.00 0.00 0.00
stress_level -0.00 0.00 -0.00 0.00 0.00 -0.00 -0.00 0.01 -0.12 -0.13 0.00 1.00 -0.01 0.00 -0.00 0.00 -0.00 -0.00 -0.12
social_activity -0.01 0.00 -0.00 -0.00 -0.00 0.00 -0.00 0.00 0.00 -0.00 0.00 -0.01 1.00 -0.00 -0.00 -0.00 -0.00 0.00 -0.00
screen_time -0.00 -0.00 0.72 0.52 0.41 0.00 -0.00 -0.00 0.00 0.18 0.00 0.00 -0.00 1.00 -0.00 -0.00 0.00 0.01 0.17
parental_support_level -0.00 0.00 -0.00 -0.00 0.00 -0.00 0.00 0.00 -0.00 -0.01 -0.00 -0.00 -0.00 -0.00 1.00 -0.00 0.00 -0.00 -0.01
motivation_level 0.00 -0.00 0.00 0.00 -0.00 0.00 0.00 0.01 -0.00 0.27 0.00 0.00 -0.00 -0.00 -0.00 1.00 -0.92 0.00 0.25
exam_anxiety_score -0.00 0.00 -0.00 0.00 0.00 -0.00 -0.00 -0.01 0.00 -0.26 -0.00 -0.00 -0.00 0.00 0.00 -0.92 1.00 -0.00 -0.24
time_management_score -0.00 0.00 0.00 0.00 0.01 -0.00 0.00 0.00 -0.00 0.01 0.00 -0.00 0.00 0.01 -0.00 0.00 -0.00 1.00 0.01
exam_score 0.01 0.00 0.24 -0.01 -0.00 0.00 0.09 0.09 0.01 0.93 0.00 -0.12 -0.00 0.17 -0.01 0.25 -0.24 0.01 1.00
student_id
age
_hours_per_day
al_media_hours
netflix_hours
ice_percentage
sleep_hours
cise_frequency
l_health_rating
previous_gpa
semester
stress_level
social_activity
screen_time
l_support_level
otivation_level
_anxiety_score
agement_score
exam_score
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75

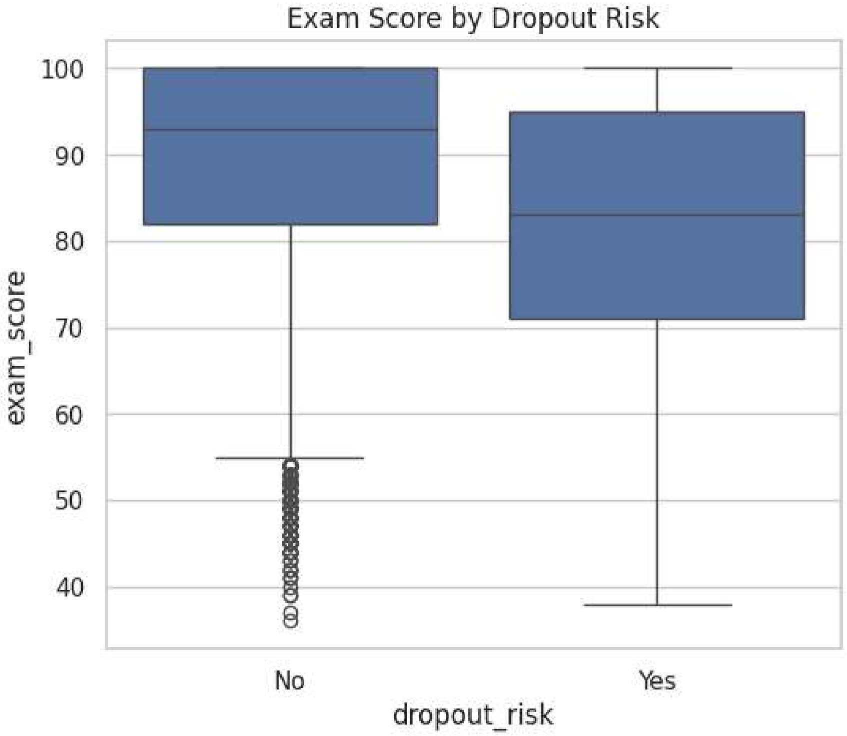
Step 12: Example Analysis - Average Exam Score by Gender

```
avg_score_by_gender = df.groupby('gender', observed=True)['exam_score'].mean()
print("Average Exam Score by Gender:")
print(avg_score_by_gender)
```

```
Average Exam Score by Gender:
gender
Female    89.108444
Male      89.152034
Other     89.163665
Name: exam_score, dtype: float64
```

Step 13: Boxplot of Exam Scores by Dropout Risk

```
# Boxplot for exam score by dropout risk
plt.figure(figsize=(6,5))
sns.boxplot(data=df, x='dropout_risk', y='exam_score')
plt.title("Exam Score by Dropout Risk")
plt.show()
```



Step 14:Feature Engineering

```
df['psychological_distress'] = df['stress_level'] + df['exam_anxiety_score']
```

Step 15: Data Preparation for Modeling Categorical columns encode

Features or target define

```
X = df.drop(['dropout_risk', 'exam_score', 'student_id'], axis=1)
X = pd.get_dummies(X, drop_first=True)
```

Step 16: Split Data into Train and Test Sets

```
# Classification target
```