



Fundamentals of Data Science Project

Predictive Analysis of Depression in Students

Prepared by :

Name	Batch	Sap ID	Specialisation
Siddarth Valasubramanian	5	500121887	B.Tech CSE Data Science

Predictive Modeling for Student Depression: A Data-Driven Approach

Objective

The goal of this project is to identify the **most influential factors** contributing to **depression among students** using a combination of **feature selection techniques** and exploratory data analysis. The insights gained lay the foundation for building predictive models that could assist educational institutions and mental health professionals in **early detection and intervention**.

Dataset Overview

- **Dataset:** Student Depression Dataset
- **Source:** Kaggle
- **Total Records:** 27,901
- **Features:** 18 variables, including demographic, academic, psychological, and lifestyle aspects
- **Target Variable:** Depression (Binary: 0 = No Depression, 1 = Depression)



Key Features:

- **Demographic:** Age, Gender, City, Degree
- **Academic:** CGPA, Academic Pressure, Study/Work Hours
- **Lifestyle:** Sleep Duration, Dietary Habits
- **Psychological:** Study Satisfaction, Financial Stress, Suicidal Thoughts, Family Mental Health History

Data Preprocessing & Cleaning

- **Missing Values:** Dataset was complete with no missing values.
- **Encoding:** Categorical features (e.g., Sleep Duration, Financial Stress) encoded appropriately.
- **Outlier Detection:** Analyzed via standard deviation and boxplots.
- **Standardization:** Applied to numerical variables where necessary.

Feature Selection Techniques

A multi-method approach was adopted to ensure **robust feature relevance**:

1. Filter Methods:

- **Correlation Matrix** and **Mutual Information Scores** identified statistically significant variables.

2. Wrapper Methods:

- **Recursive Feature Elimination (RFE)** was used with logistic regression to iteratively rank features.

3. Embedded Methods:

- **Lasso Regression** highlighted variables with the strongest coefficients.
- **Decision Tree Classifier** ranked features by Gini-based importance.

Key Predictive Features Identified

After consolidating results from all three selection methods, the most impactful predictors of depression are:

Rank	Feature	Insight
1	Academic Pressure	High academic burden strongly correlates with depression.
2	Study Satisfaction	Lower satisfaction levels are indicative of psychological
3	Sleep Duration	Students sleeping less than 6 hours show higher depression
4	Financial Stress	Economic strain is a consistent stressor impacting mental
5	Suicidal Thoughts	The presence of suicidal ideation is a critical indicator.
6	Family History of Mental Illness	A known predictor of hereditary mental health risks.
7	Work/Study Hours	Extended study/work hours are linked to fatigue and mental burnout.

Evaluation and Visualization Highlights

- **Heatmaps** demonstrated strong correlations between academic pressure, financial stress, and depression.
- **Bar plots** from decision tree feature importances ranked Suicidal Thoughts, Academic Pressure, and Sleep Duration highest.
- **Boxplots** revealed that depressed students typically had:
 - Lower CGPA
 - Higher academic/work pressure
 - Shorter sleep durations

Insights and Implications

This analysis uncovers several important conclusions:

1. Multifactorial Nature of Depression:

Depression in students isn't triggered by a single cause—it results from a **combination of academic, psychological, and lifestyle stressors**.

2. Psychological Indicators Matter:

Variables like **suicidal thoughts** and **family history of mental illness** are critical red flags for intervention.

3. Academic Environment Plays a Major Role:

High academic pressure, long working hours, and low satisfaction with studies contribute significantly to mental fatigue and depression.

4. Lifestyle Factors Are Often Overlooked:

Poor sleep hygiene and unhealthy dietary habits, while less strongly correlated, still contribute to the overall risk.

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder

# Step 2: Load the dataset
df = pd.read_csv('student_depression_dataset.csv') # change path if needed

# Step 3: Drop unnecessary ID column
df.drop(columns=['id'], inplace=True)

# Step 4: Convert Yes/No columns to binary
df['Suicidal_Thoughts'] = df['Have you ever had suicidal thoughts?'].map({'Yes': 1, 'No': 0})
df['Family_History'] = df['Family History of Mental Illness'].map({'Yes': 1, 'No': 0})
df.drop(columns=['Have you ever had suicidal thoughts?', 'Family History of Mental Illness'], inplace=True)

# Step 5: Map ordinal values to numbers
sleep_map = {
    'Less than 5 hours': 1,
    '5-6 hours': 2,
    '7-8 hours': 3,
    'More than 8 hours': 4,
    'Irregular': 0
}
df['Sleep_Duration'] = df['Sleep Duration'].map(sleep_map)
diet_map = {
    'Healthy': 2,
    'Moderate': 1,
    'Unhealthy': 0,
    'Irregular': 0
}
df['Dietary_Habits'] = df['Dietary Habits'].map(diet_map)

# Convert Financial Stress to numeric
df['Financial_Stress'] = pd.to_numeric(df['Financial Stress'], errors='coerce')
df.drop(columns=['Sleep Duration', 'Financial Stress'], inplace=True)

# Step 6: Label Encode remaining categorical columns
label_cols = ['Gender', 'City', 'Profession', 'Degree']
le = LabelEncoder()
for col in label_cols:
    df[col] = le.fit_transform(df[col])
```

CODE

```
# Step 7: Handle missing values (if any)
for col in df.columns:
    if df[col].isnull().sum() > 0:
        if df[col].dtype in ['float64', 'int64']:
            df[col].fillna(df[col].median(), inplace=True)
        else:
            df[col].fillna(df[col].mode()[0], inplace=True)

# Step 8: Final check
print(df.info())
print(df.head())
```

SNIPPETS :

```
Data columns (total 18 columns):
#   Column                Non-Null Count  Dtype
---  -
0   Gender                27901 non-null    int64
1   Age                   27901 non-null    float64
2   City                  27901 non-null    int64
3   Profession             27901 non-null    int64
4   Academic Pressure      27901 non-null    float64
5   Work Pressure          27901 non-null    float64
6   CGPA                   27901 non-null    float64
7   Study Satisfaction      27901 non-null    float64
8   Job Satisfaction       27901 non-null    float64
9   Dietary Habits         27901 non-null    object
10  Degree                 27901 non-null    int64
11  Work/Study Hours       27901 non-null    float64
12  Depression             27901 non-null    int64
13  Suicidal_Thoughts      27901 non-null    int64
14  Family_History         27901 non-null    int64
15  Sleep_Duration         27901 non-null    float64
16  Dietary_Habits         27901 non-null    float64
17  Financial_Stress       27901 non-null    float64
dtypes: float64(10), int64(7), object(1)
memory usage: 3.8+ MB
None
   Gender  Age  City  Profession  Academic Pressure  Work Pressure  CGPA  \
0      1  33.0   51         12             5.0         0.0  8.97
1      0  24.0    5         12             2.0         0.0  5.90
2      1  31.0   44         12             3.0         0.0  7.03
3      0  28.0   49         12             3.0         0.0  5.59
4      0  25.0   18         12             4.0         0.0  8.13

   Study Satisfaction  Job Satisfaction  Dietary Habits  Degree  \
0                2.0                0.0      Healthy     4
1                5.0                0.0      Moderate    11
2                5.0                0.0      Healthy     6
3                2.0                0.0      Moderate     8
4                3.0                0.0      Moderate    17

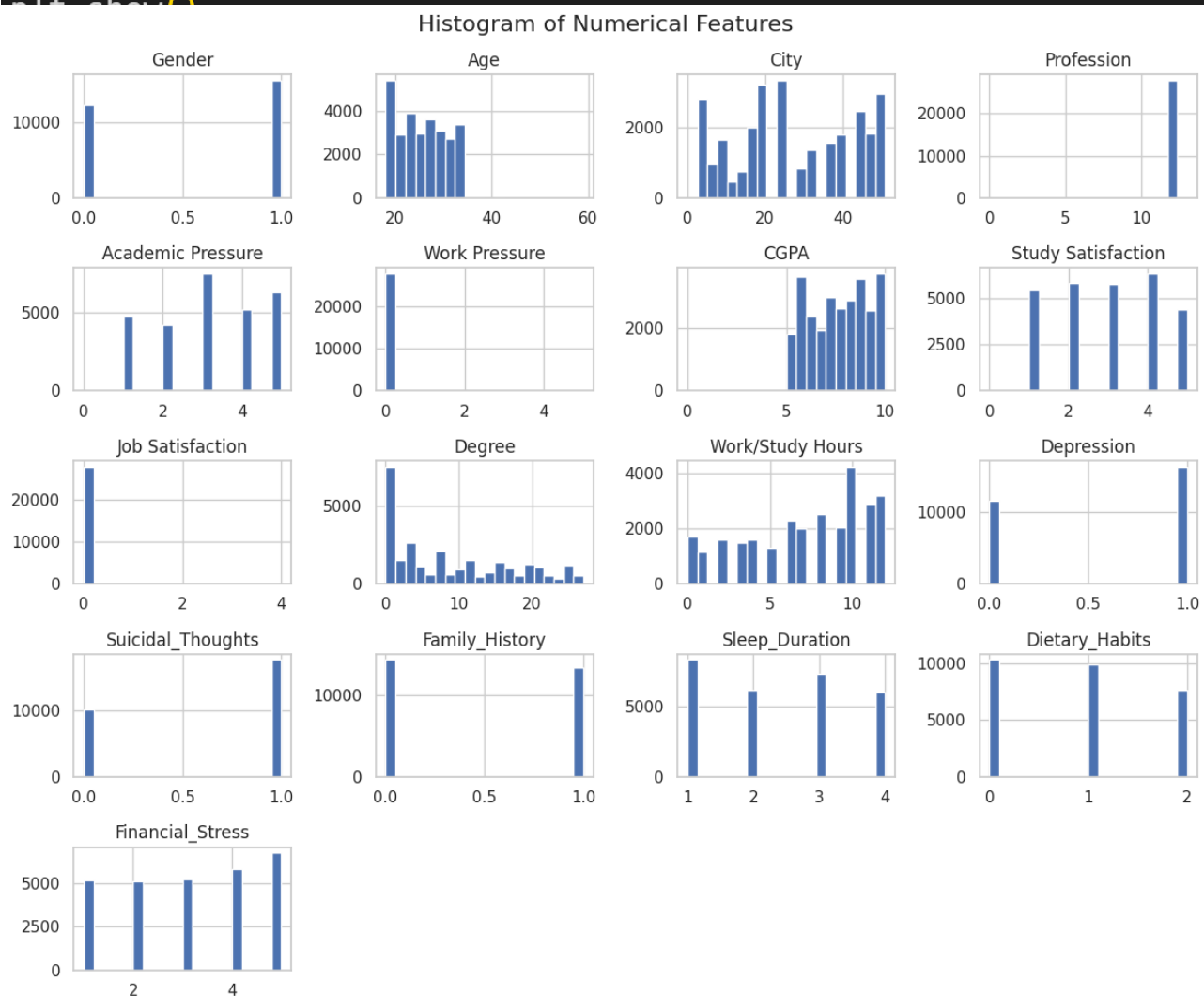
   Work/Study Hours  Depression  Suicidal_Thoughts  Family_History  \
0                3.0            1                1                0
1                3.0            0                0                1
2                9.0            0                0                1
3                4.0            1                1                1
4                1.0            0                1                0

   Sleep_Duration  Dietary_Habits  Financial_Stress
0                2.0            2.0            1.0
1                2.0            1.0            2.0
2                1.0            2.0            1.0
3                3.0            1.0            5.0
4                2.0            1.0            1.0
<ipython-input-3-0a0977cc9d71>:47: FutureWarning: A value is trying
The behavior will change in pandas 3.0. This inplace method will ne
```

Visualisation

I) Histogram

```
df.hist(figsize=(12, 10), bins=20)
plt.suptitle('Histogram of Numerical Features', fontsize=16)
plt.tight_layout()
plt.show()
```



ii) Box Plot (Outlier Detection)

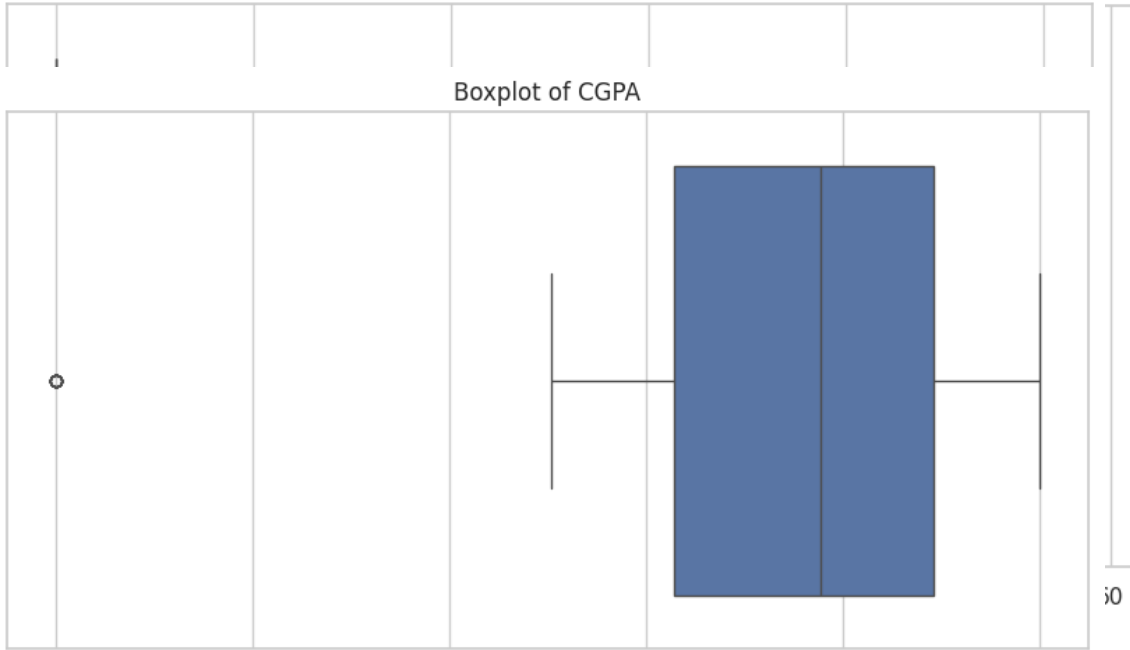
```
[ ] import matplotlib.pyplot as plt
import seaborn as sns

sns.set(style="whitegrid")

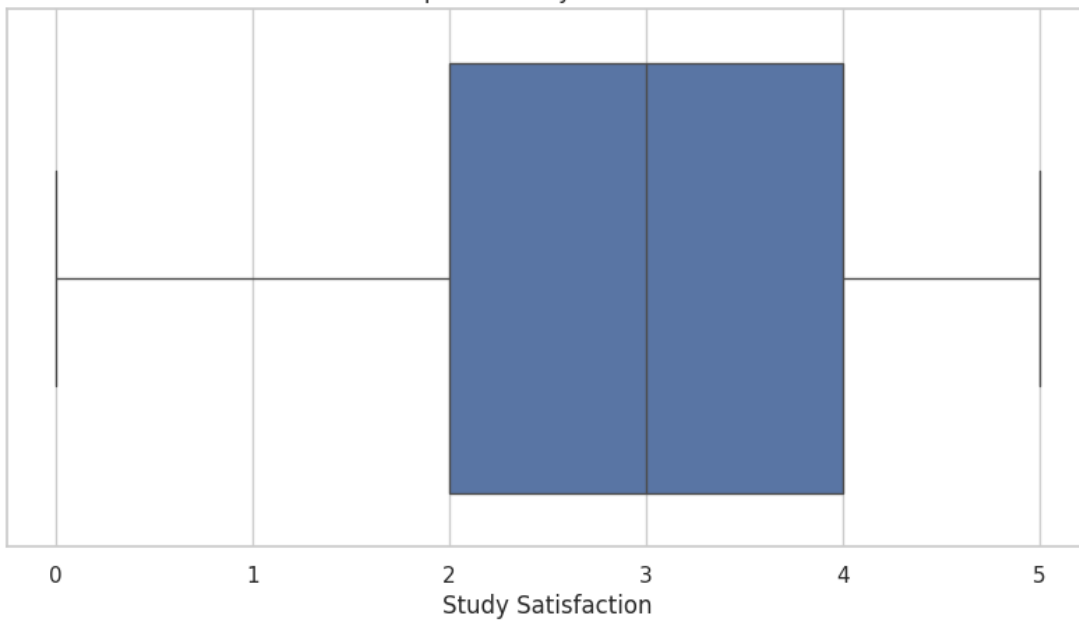
numeric_columns = [
    'Age',
    'Academic Pressure',
    'Work Pressure',
    'CGPA',
    'Study Satisfaction',
    'Job Satisfaction',
    'Work/Study Hours',
    'Financial_Stress'
]

for col in numeric_columns:
    plt.figure(figsize=(10, 5))
    sns.boxplot(x=df[col])
    plt.title(f'Boxplot of {col}')
    plt.xlabel(col)
    plt.show()
```

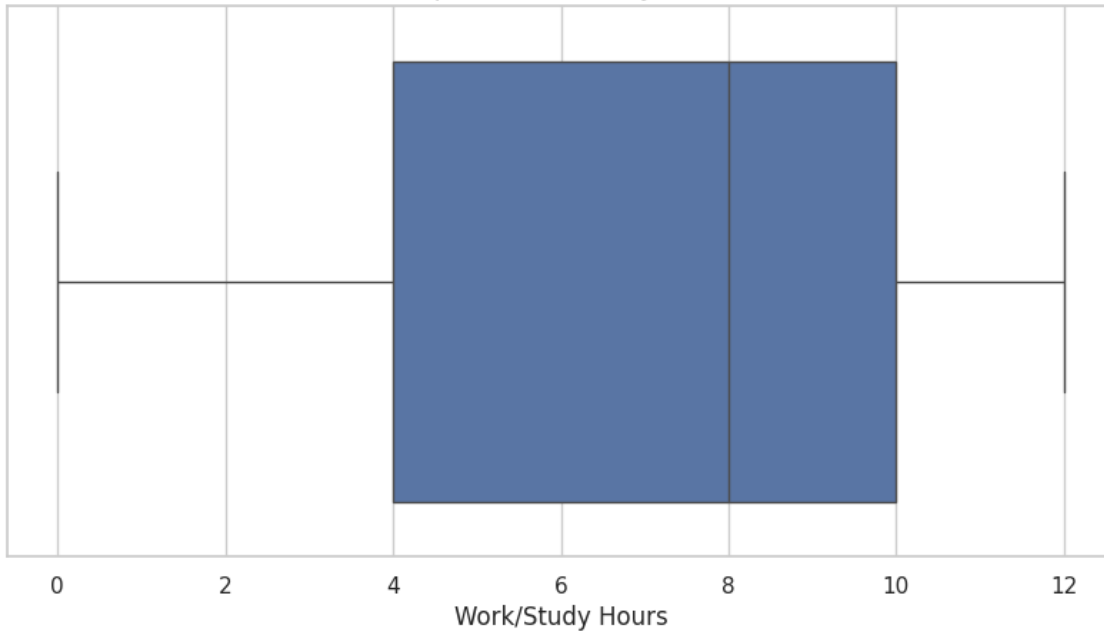

Boxplot of Work Pressure



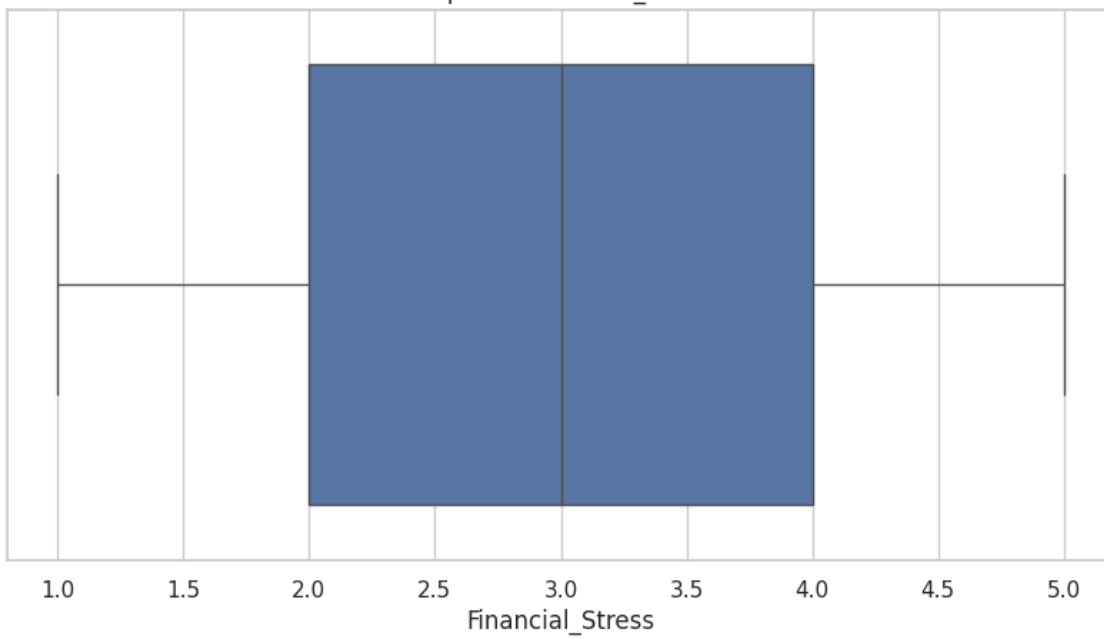
Boxplot of Study Satisfaction



Boxplot of Work/Study Hours

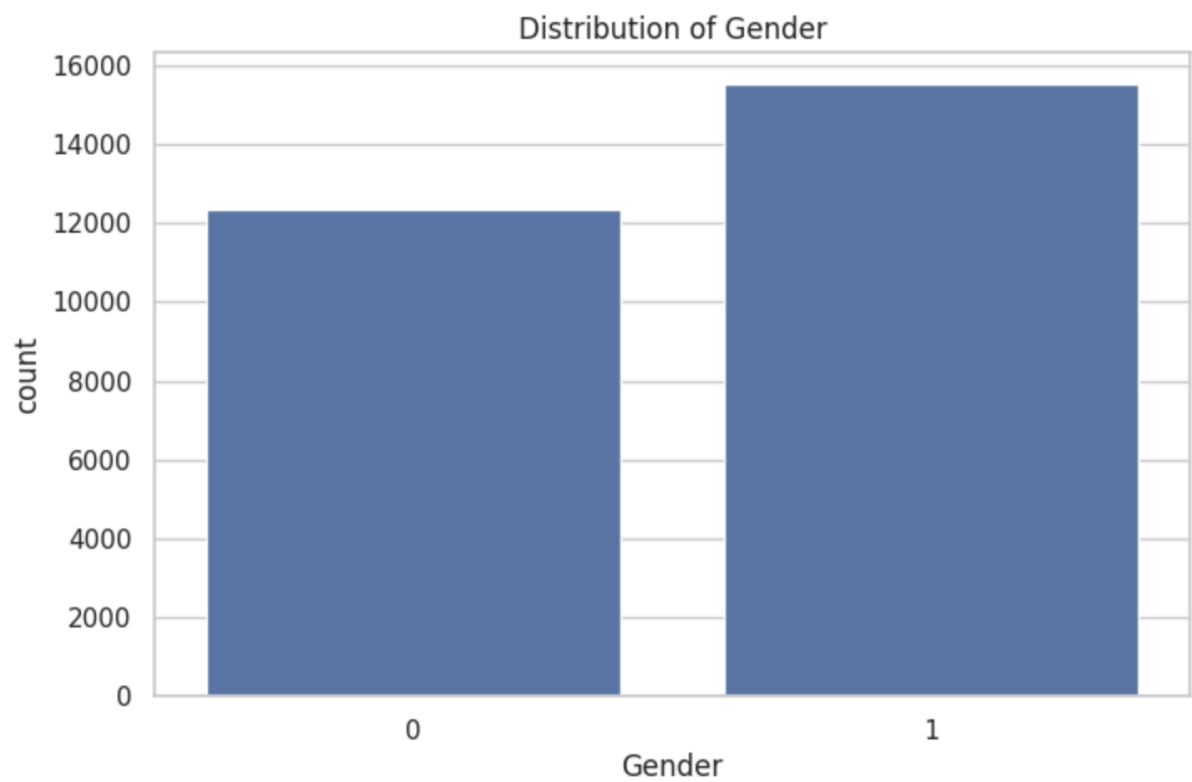


Boxplot of Financial_Stress



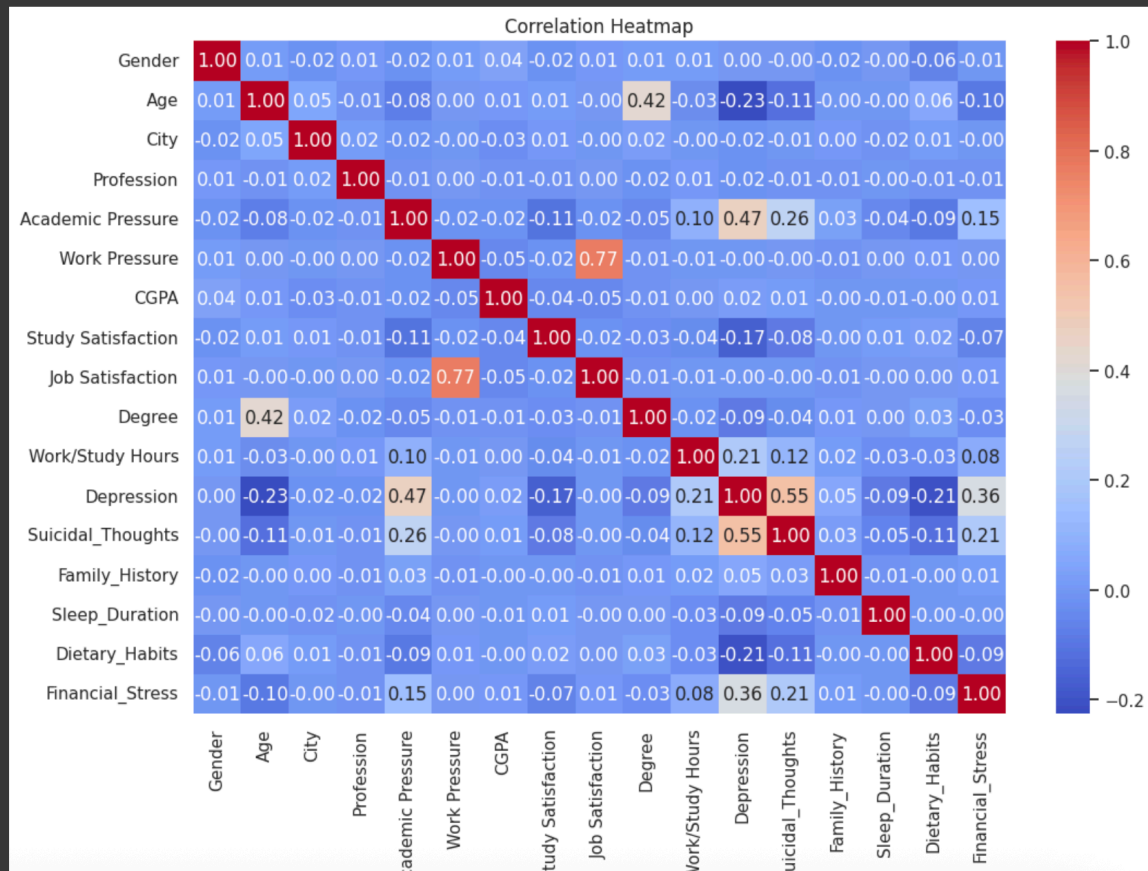
iii) Bar Charts for Categorical Data

```
[ ] plt.figure(figsize=(8, 5))  
    sns.countplot(x='Gender', data=df)  
    plt.title('Distribution of Gender')  
    plt.show()
```



iv) Correlation Heatmap

```
plt.figure(figsize=(12, 8))
sns.heatmap(df.corr(numeric_only=True), annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Heatmap')
plt.show()
```



V) Remove Outliers Using Z-Score

```
from scipy.stats import zscore

# Select numerical columns
num_cols = ['Age', 'CGPA', 'Work/Study Hours', 'Academic Pressure', 'Work Pressure']

# Compute Z-scores
z_scores = df[num_cols].apply(zscore)

# Filter rows where all Z-scores are within [-3, 3]
df_no_outliers = df[(z_scores.abs() <= 3).all(axis=1)]

# Display shape before and after
print("Original shape:", df.shape)
print("Shape after removing outliers:", df_no_outliers.shape)

# Optionally, save the cleaned dataset
df_no_outliers.to_csv("cleaned_student_depression_dataset.csv", index=False)
```

```
Original shape: (27901, 18)
Shape after removing outliers: (27873, 18)
```

VI) Encode Categorical Variables using OneHot Encoder.

```
[ ] import pandas as pd

cat_cols = df.select_dtypes(include='object').columns

# One-Hot Encoding
df_encoded = pd.get_dummies(df, columns=cat_cols, drop_first=True)

print("Original shape:", df.shape)
print("Encoded shape:", df_encoded.shape)
```

```
⇒ Original shape: (27901, 18)
  Encoded shape: (27901, 20)
```

Feature Selection

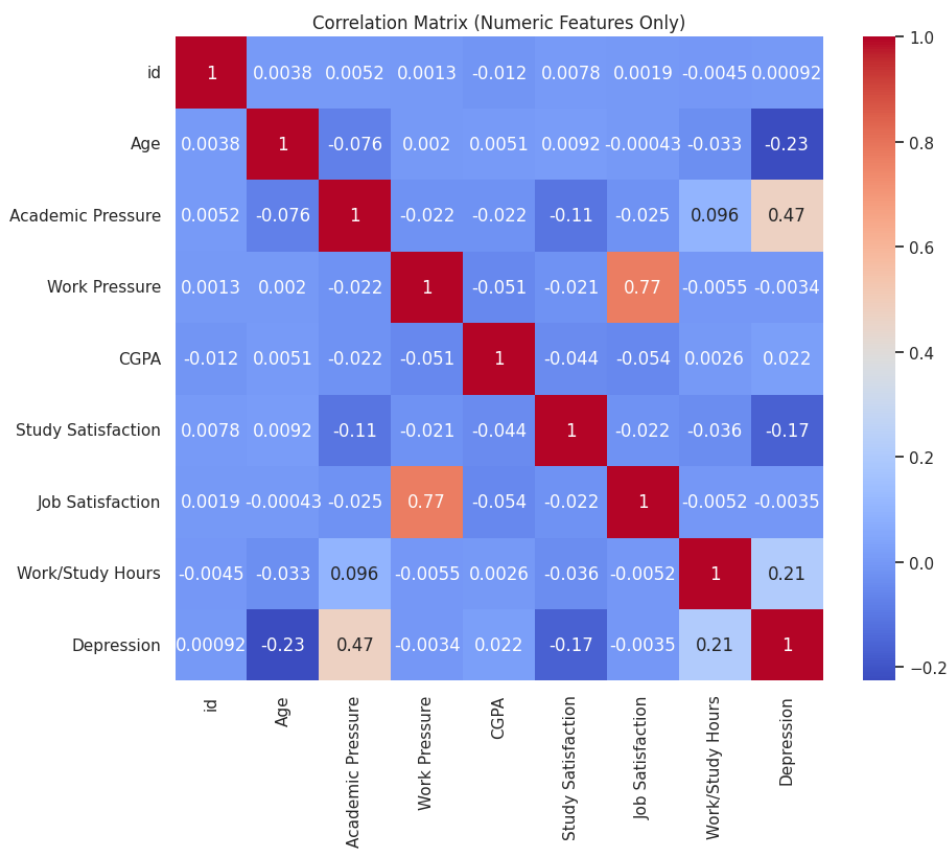
i. Correlation Matrix - for numerical features

```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

df = pd.read_csv('student_depression_dataset.csv')

#only numeric columns
numeric_df = df.select_dtypes(include=['number'])

plt.figure(figsize=(10, 8))
sns.heatmap(numeric_df.corr(), annot=True, cmap='coolwarm')
plt.title('Correlation Matrix (Numeric Features Only)')
plt.show()
```



II) Chi-Square Test

```
[ ] from sklearn.feature_selection import chi2
    from sklearn.preprocessing import LabelEncoder
    import pandas as pd

    # Encode categorical features
    df_encoded = df.copy()
    for col in df_encoded.select_dtypes(include='object').columns:
        df_encoded[col] = LabelEncoder().fit_transform(df_encoded[col])

    # Define X (features) and y (target)
    X = df_encoded.drop('Depression', axis=1)
    y = df_encoded['Depression']

    # Apply chi-square test
    chi_scores = chi2(X, y)

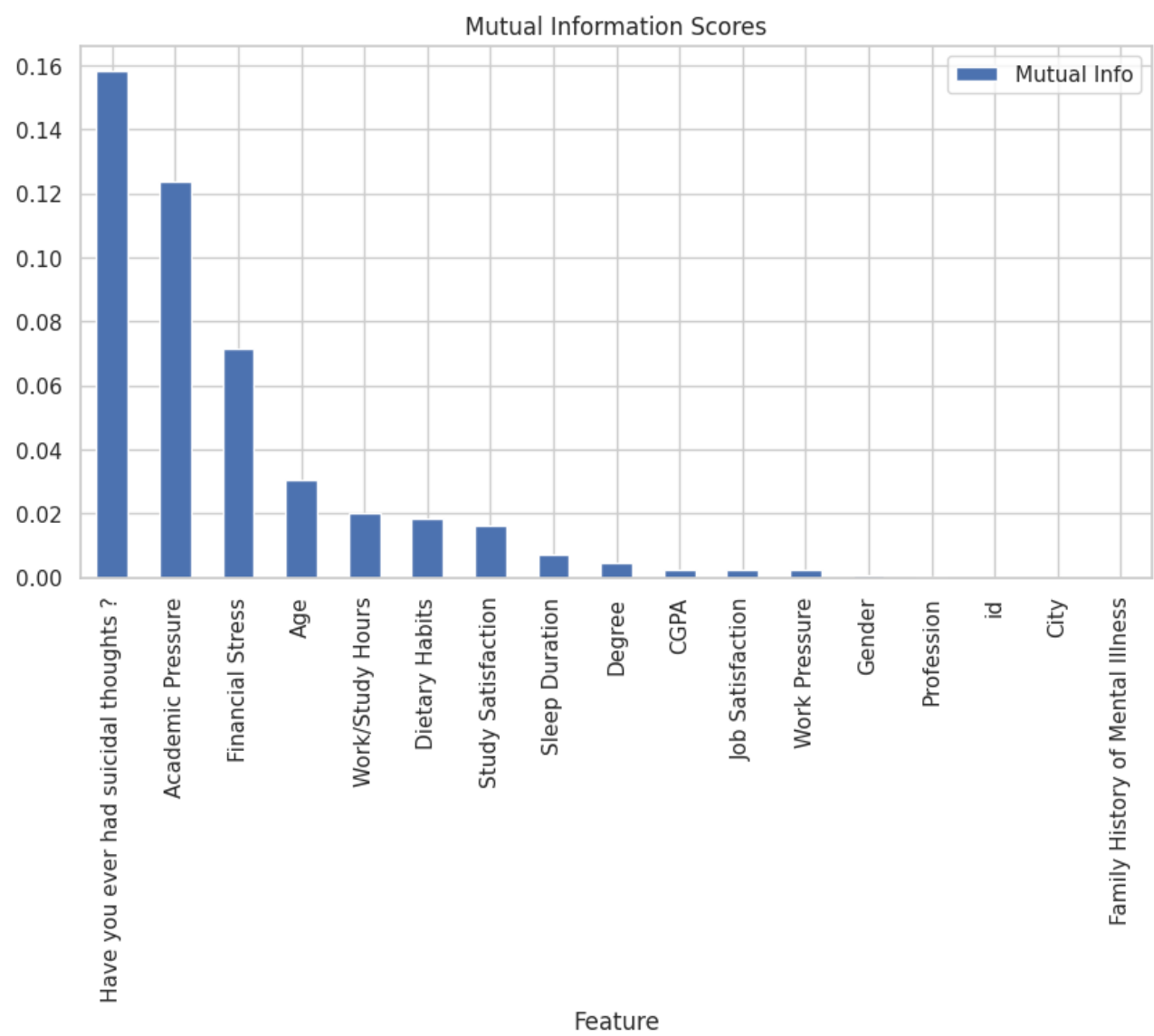
    # Display results
    chi2_results = pd.DataFrame({'Feature': X.columns, 'Chi2 Score': chi_scores[0], 'p-value': chi_scores[1]})
    chi2_results.sort_values(by='Chi2 Score', ascending=False, inplace=True)
    print(chi2_results)
```

	Feature	Chi2 Score	p-value
5	Academic Pressure	3821.833425	0.000000e+00
15	Financial Stress	3557.476357	0.000000e+00
13	Have you ever had suicidal thoughts ?	3057.294614	0.000000e+00
14	Work/Study Hours	2331.007175	0.000000e+00
12	Degree	1629.322633	0.000000e+00
2	Age	1333.049964	7.431387e-292
11	Dietary Habits	1248.711851	1.581353e-273
0	id	557.480304	2.969886e-123
8	Study Satisfaction	495.417900	9.439564e-110
3	City	146.570269	9.742568e-34
16	Family History of Mental Illness	41.103170	1.444020e-10
10	Sleep Duration	10.478948	1.207425e-03
7	CGPA	3.888342	4.862242e-02
6	Work Pressure	1.409416	2.351533e-01
9	Job Satisfaction	0.978809	3.224929e-01
1	Gender	0.039780	8.419114e-01
4	Profession	0.033266	8.552766e-01

iii) Mutual Information

```
from sklearn.feature_selection import mutual_info_classif

mutual_info = mutual_info_classif(X, y)
mi_df = pd.DataFrame({'Feature': X.columns, 'Mutual Info': mutual_info})
mi_df.sort_values(by='Mutual Info', ascending=False).plot(x='Feature', y='Mutual Info', kind='bar', figsize=(10,5))
plt.title('Mutual Information Scores')
plt.show()
```



iv) Feature Selection Method

A. Filter Method - Mutual information

```
from sklearn.feature_selection import mutual_info_classif

mi_scores = mutual_info_classif(X, y)
mi_df = pd.DataFrame({'Feature': X.columns, 'MI Score': mi_scores})
mi_df = mi_df.sort_values(by='MI Score', ascending=False)
mi_df.head()
```



	Feature	MI Score
13	Have you ever had suicidal thoughts ?	0.157499
5	Academic Pressure	0.124283
15	Financial Stress	0.066204
2	Age	0.034810
11	Dietary Habits	0.025127

B. Wrapper method - RFE

```
[ ] from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Apply RFE on scaled data
model = LogisticRegression(max_iter=1000)
rfe = RFE(model, n_features_to_select=5)
rfe.fit(X_scaled, y)

# Get top features
rfe_features = X.columns[rfe.support_]
print("Top RFE Features:", list(rfe_features))
```

Top RFE Features: ['Age', 'Academic Pressure', 'Have you ever had suicidal thoughts ?', 'Work/Study Hours', 'Financial Stress']

C. Embedded Method - Decision Tree

```
[ ] from sklearn.tree import DecisionTreeClassifier

tree = DecisionTreeClassifier()
tree.fit(X, y)
tree_importance = tree.feature_importances_
tree_df = pd.DataFrame({'Feature': X.columns, 'Importance': tree_importance})
tree_df = tree_df.sort_values(by='Importance', ascending=False)
tree_df.head()
```



	Feature	Importance
13	Have you ever had suicidal thoughts ?	0.298418
5	Academic Pressure	0.132755
0	id	0.093564
7	CGPA	0.077204
15	Financial Stress	0.069449

Compare and count frequently Selected features

```
# Combine selected top features from each method
top_mi = list(mi_df['Feature'].head(5))
top_rfe = list(rfe_features)
top_tree = list(tree_df['Feature'].head(5))

# Count feature frequency
from collections import Counter
all_selected = top_mi + top_rfe + top_tree
feature_counts = Counter(all_selected)

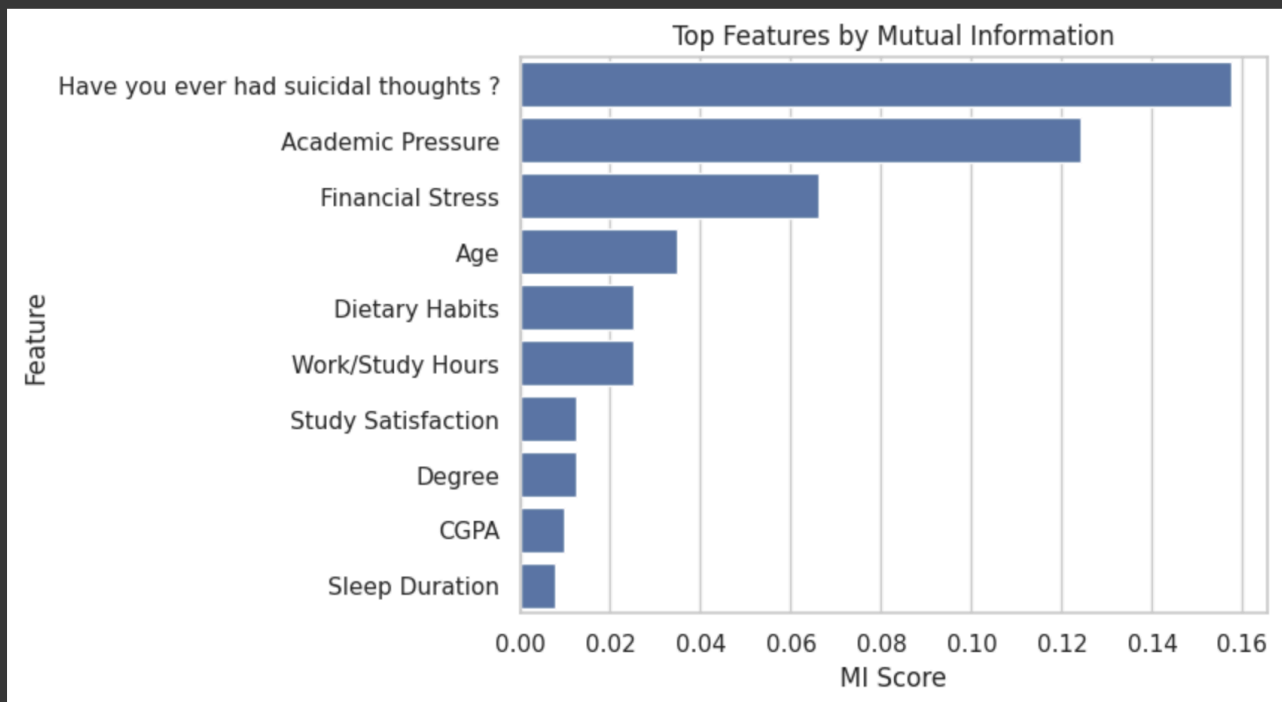
# Display features sorted by frequency
final_features = pd.DataFrame.from_dict(feature_counts, orient='index', columns=['Count'])
final_features = final_features.sort_values(by='Count', ascending=False)
print(final_features)
```

	Count
Have you ever had suicidal thoughts ?	3
Academic Pressure	3
Financial Stress	3
Age	2
Dietary Habits	1
Work/Study Hours	1
id	1
CGPA	1

Evaluation and Visualisation

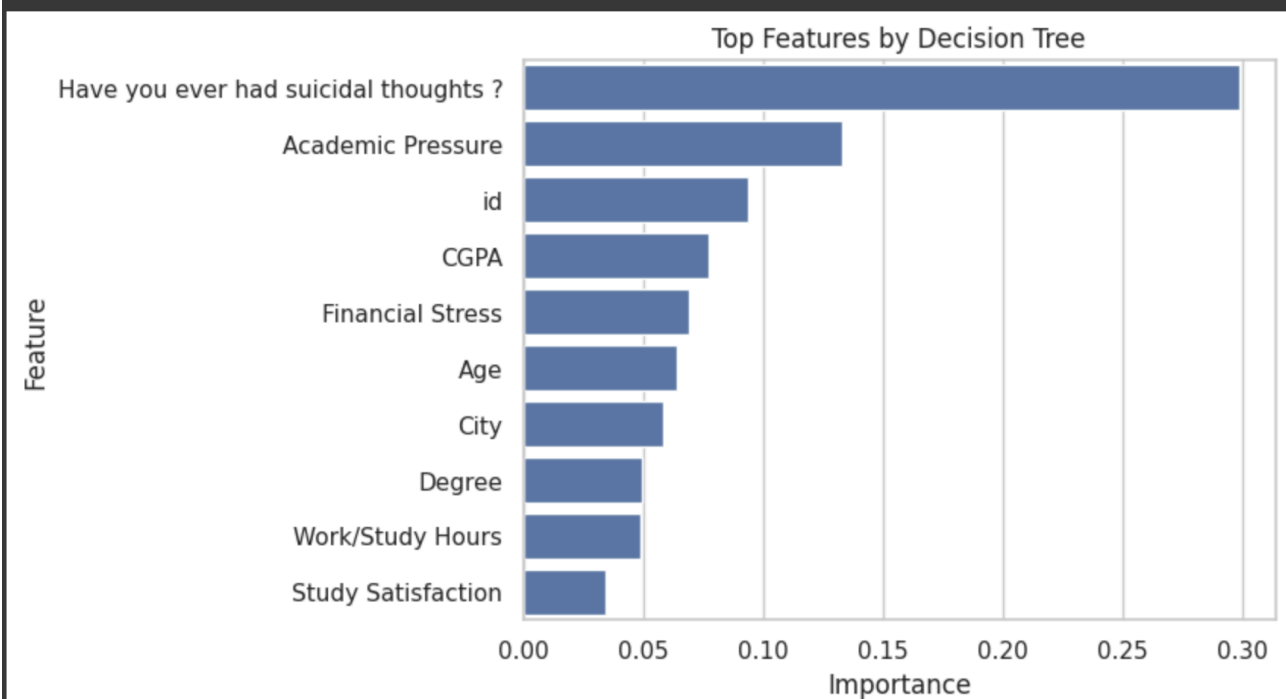
I. Bar Chart

```
sns.barplot(x='MI Score', y='Feature', data=mi_df.head(10))  
plt.title('Top Features by Mutual Information')  
plt.show()
```



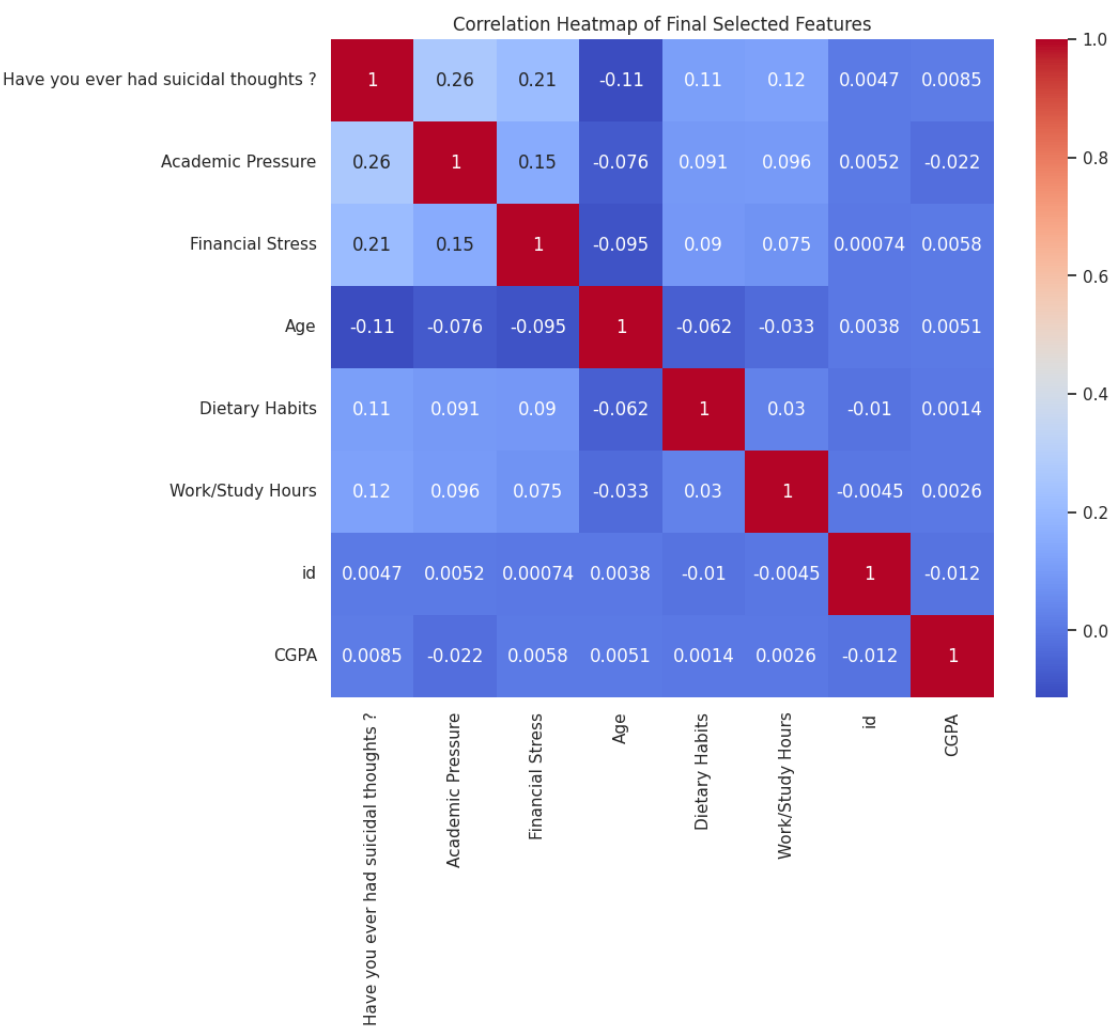
II. Bar Chart - Decision Tree Importance

```
sns.barplot(x='Importance', y='Feature', data=tree_df.head(10))  
plt.title('Top Features by Decision Tree')  
plt.show()
```



III) Correlation Heatmap - Feature Relationships

```
plt.figure(figsize=(10,8))
sns.heatmap(X[final_features['Feature']].corr(), annot=True, cmap='coolwarm')
plt.title('Correlation Heatmap of Final Selected Features')
plt.show()
```



Conclusion

The project successfully identified a set of **core features** that can be used to predict student depression with high confidence. These insights can drive the creation of **machine learning models** to identify at-risk students and support proactive mental health strategies in academic settings.

This work not only emphasizes but also underlines the importance of **early detection, empathy, and systemic support** in tackling student depression.

Google Colab Link :

<https://colab.research.google.com/drive/1T4wswuQBpdfmjiTB7g3FGLKeIS4ks4QT>