Depression Classification Project

This notebook includes all steps: data cleaning, EDA, model building, and evaluation.

```
# Import necessary libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix,
accuracy_score

# Load the dataset
df = pd.read_csv(r"C:\Users\WINDOWS 11 PRO\Downloads\ids projet data
set.csv")

df
```

```
           id  Gender  Age           City Profession  Academic
Pressure  \
0           2    Male   33  Visakhapatnam    Student
5
1           8  Female   24      Bangalore    Student
2
2          26    Male   31       Srinagar    Student
3
3          30  Female   28       Varanasi    Student
3
4          32  Female   25         Jaipur    Student
4
...       ...     ...  ...            ...        ...                 ..
.
27896  140685  Female   27          Surat    Student
5
27897  140686    Male   27       Ludhiana    Student
2
27898  140689    Male   31      Faridabad    Student
3
27899  140690  Female   18       Ludhiana    Student
5
27900  140699    Male   27          Patna    Student
4

       Work Pressure  CGPA      Degree Have you ever had suicidal
```

```
thoughts ?  \
0                   0  8.97     B.Pharm
Yes
1                   0  5.90         BSc
No
2                   0  7.03          BA
No
3                   0  5.59         BCA
Yes
4                   0  8.13      M.Tech
Yes
...               ...   ...         ...
...
27896               0  5.75  'Class 12'
Yes
27897               0  9.40         MSc
No
27898               0  6.61          MD
No
27899               0  6.88  'Class 12'
Yes
27900               0  9.24         BCA
Yes

       Work/Study Hours Financial Stress  Depression
0                     3                1           1
1                     3                2           0
2                     9                1           0
3                     4                5           1
4                     1                1           0
...                 ...              ...         ...
27896                 7                1           0
27897                 0                3           0
27898                12                2           0
27899                10                5           1
27900                 2                3           1

[27901 rows x 13 columns]

df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 27901 entries, 0 to 27900
Data columns (total 13 columns):
 #   Column                                 Non-Null Count  Dtype
---  ------                                 --------------  -----
 0   id                                     27901 non-null  int64
 1   Gender                                 27901 non-null  object
 2   Age                                    27901 non-null  int64
 3   City                                   27901 non-null  object
```

```
 4   Profession                                27901 non-null  object
 5   Academic Pressure                         27901 non-null  int64
 6   Work Pressure                             27901 non-null  int64
 7   CGPA                                      27901 non-null  float64
 8   Degree                                    27901 non-null  object
 9   Have you ever had suicidal thoughts ?     27901 non-null  object
 10  Work/Study Hours                          27901 non-null  int64
 11  Financial Stress                          27901 non-null  object
 12  Depression                                27901 non-null  int64
dtypes: float64(1), int64(6), object(6)
memory usage: 2.8+ MB
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 27901 entries, 0 to 27900
Data columns (total 13 columns):
 #   Column                                    Non-Null Count  Dtype
---  ------                                    --------------  -----
 0   id                                        27901 non-null  int64
 1   Gender                                    27901 non-null  object
 2   Age                                       27901 non-null  int64
 3   City                                      27901 non-null  object
 4   Profession                                27901 non-null  object
 5   Academic Pressure                         27901 non-null  int64
 6   Work Pressure                             27901 non-null  int64
 7   CGPA                                      27901 non-null  float64
 8   Degree                                    27901 non-null  object
 9   Have you ever had suicidal thoughts ?     27901 non-null  object
 10  Work/Study Hours                          27901 non-null  int64
 11  Financial Stress                          27901 non-null  object
 12  Depression                                27901 non-null  int64
dtypes: float64(1), int64(6), object(6)
memory usage: 2.8+ MB
```

```
df.describe()
```

```
                  id           Age  Academic Pressure  Work
Pressure  \
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  Work/Study Hours    Depression
count  27901.000000      27901.000000  27901.000000
mean       7.656104          7.156984      0.585499
std        1.470707          3.707642      0.492645
min        0.000000          0.000000      0.000000
25%        6.290000          4.000000      0.000000
50%        7.770000          8.000000      1.000000
75%        8.920000         10.000000      1.000000
max       10.000000         12.000000      1.000000
```

```
df.iloc[0]
```

```
Gender                                  1.00
Age                                    33.00
City                                   51.00
Profession                             12.00
Academic Pressure                       5.00
Work Pressure                           0.00
CGPA                                    8.97
Degree                                  4.00
Have you ever had suicidal thoughts ?   1.00
Work/Study Hours                        3.00
Financial Stress                        0.00
Depression                              1.00
Name: 0, dtype: float64
```

```
df.iloc[1]
```

```
Gender                                  0.0
Age                                    24.0
City                                    5.0
Profession                             12.0
Academic Pressure                       2.0
Work Pressure                           0.0
CGPA                                    5.9
Degree                                 11.0
Have you ever had suicidal thoughts ?   0.0
Work/Study Hours                        3.0
Financial Stress                        1.0
Depression                              0.0
Name: 1, dtype: float64
```

```
# Example: Create a new feature for 'CGPA per Hour of Study'
df['CGPA_per_Hour'] = df['CGPA'] / df['Work/Study Hours']
df.head()
```

```
   Gender  Age  City  Profession  Academic Pressure  Work Pressure
CGPA  \
0       1   33    51          12                  5              0
8.97
1       0   24     5          12                  2              0
5.90
2       1   31    44          12                  3              0
7.03
3       0   28    49          12                  3              0
5.59
4       0   25    18          12                  4              0
8.13

   Degree  Have you ever had suicidal thoughts ?  Work/Study Hours  \
0       4                                      1                 3
1      11                                      0                 3
2       6                                      0                 9
3       8                                      1                 4
4      17                                      1                 1

   Financial Stress  Depression  CGPA_per_Hour
0                 0           1       2.990000
1                 1           0       1.966667
2                 0           0       0.781111
3                 4           1       1.397500
4                 0           0       8.130000
```

```
df.dropna(inplace=True)
print("After dropna (inplace=True):")
df.head()
```

```
After dropna (inplace=True):

   Gender  Age  City  Profession  Academic Pressure  Work Pressure
CGPA  \
0       1   33    51          12                  5              0
8.97
1       0   24     5          12                  2              0
5.90
2       1   31    44          12                  3              0
7.03
3       0   28    49          12                  3              0
5.59
4       0   25    18          12                  4              0
8.13

   Degree  Have you ever had suicidal thoughts ?  Work/Study Hours  \
0       4                                      1                 3
1      11                                      0                 3
```

```
2       6                                          0                  9
3       8                                          1                  4
4      17                                          1                  1

   Financial Stress  Depression  CGPA_per_Hour
0                 0           1       2.990000
1                 1           0       1.966667
2                 0           0       0.781111
3                 4           1       1.397500
4                 0           0       8.130000
```

```
df.dropna(inplace=False)
print("After dropna (inplace=False):")
df.head()
```

```
After dropna (inplace=False):

   Gender  Age  City  Profession  Academic Pressure  Work Pressure
CGPA  \
0       1   33    51          12                  5              0
8.97
1       0   24     5          12                  2              0
5.90
2       1   31    44          12                  3              0
7.03
3       0   28    49          12                  3              0
5.59
4       0   25    18          12                  4              0
8.13

   Degree  Have you ever had suicidal thoughts ?  Work/Study Hours  \
0       4                                       1                 3
1      11                                       0                 3
2       6                                       0                 9
3       8                                       1                 4
4      17                                       1                 1

   Financial Stress  Depression  CGPA_per_Hour
0                 0           1       2.990000
1                 1           0       1.966667
2                 0           0       0.781111
3                 4           1       1.397500
4                 0           0       8.130000
```

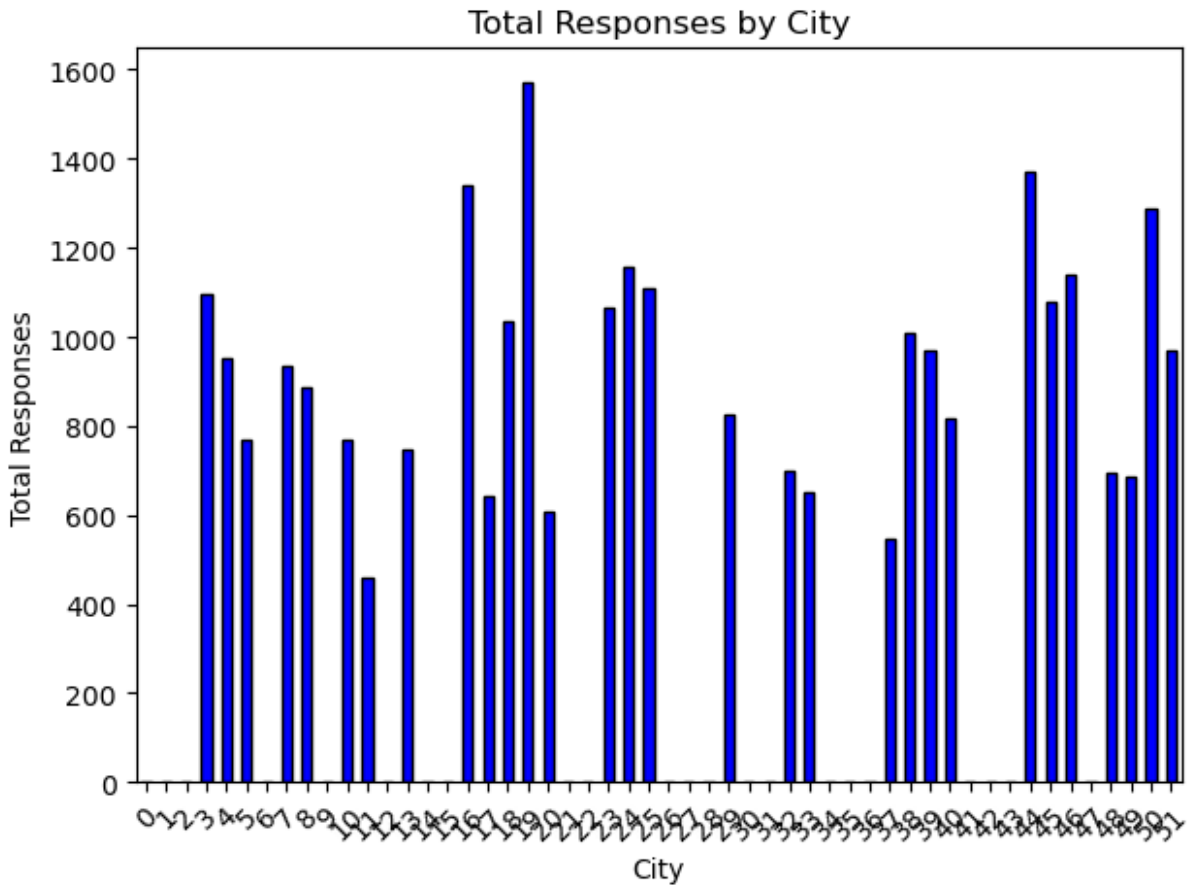
Data Cleaning & Preprocessing

```
# Drop ID column if not needed
df.drop("id", axis=1, inplace=True)
```

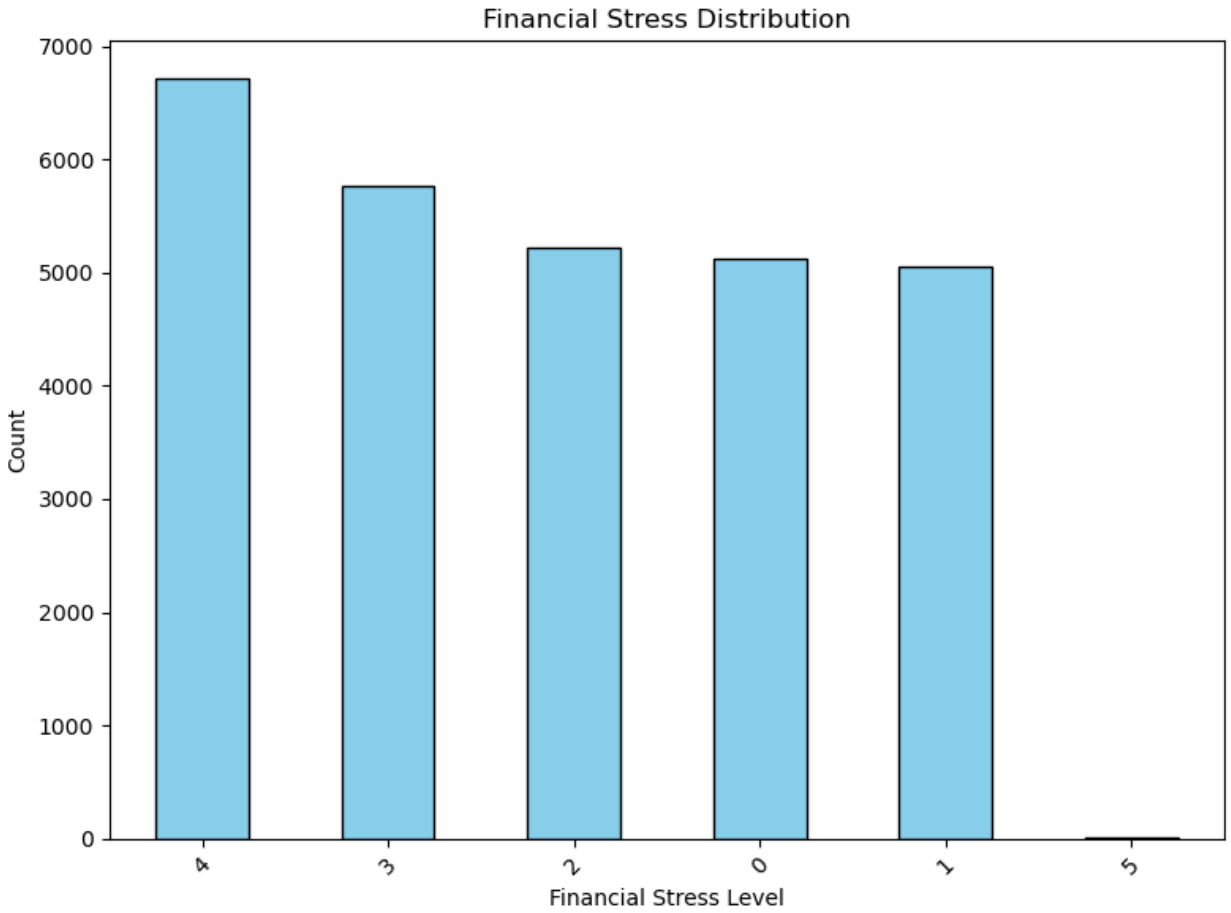
```python
# Check for missing values
print(df.isnull().sum())

# Encode categorical variables
le = LabelEncoder()
categorical_cols = ["Gender", "City", "Profession", "Degree",
                    "Have you ever had suicidal thoughts ?",
"Financial Stress"]

for col in categorical_cols:
    df[col] = le.fit_transform(df[col])
```

```
Gender                                   0
Age                                      0
City                                     0
Profession                               0
Academic Pressure                        0
Work Pressure                            0
CGPA                                     0
Degree                                   0
Have you ever had suicidal thoughts ?    0
Work/Study Hours                         0
Financial Stress                         0
Depression                               0
dtype: int64
```

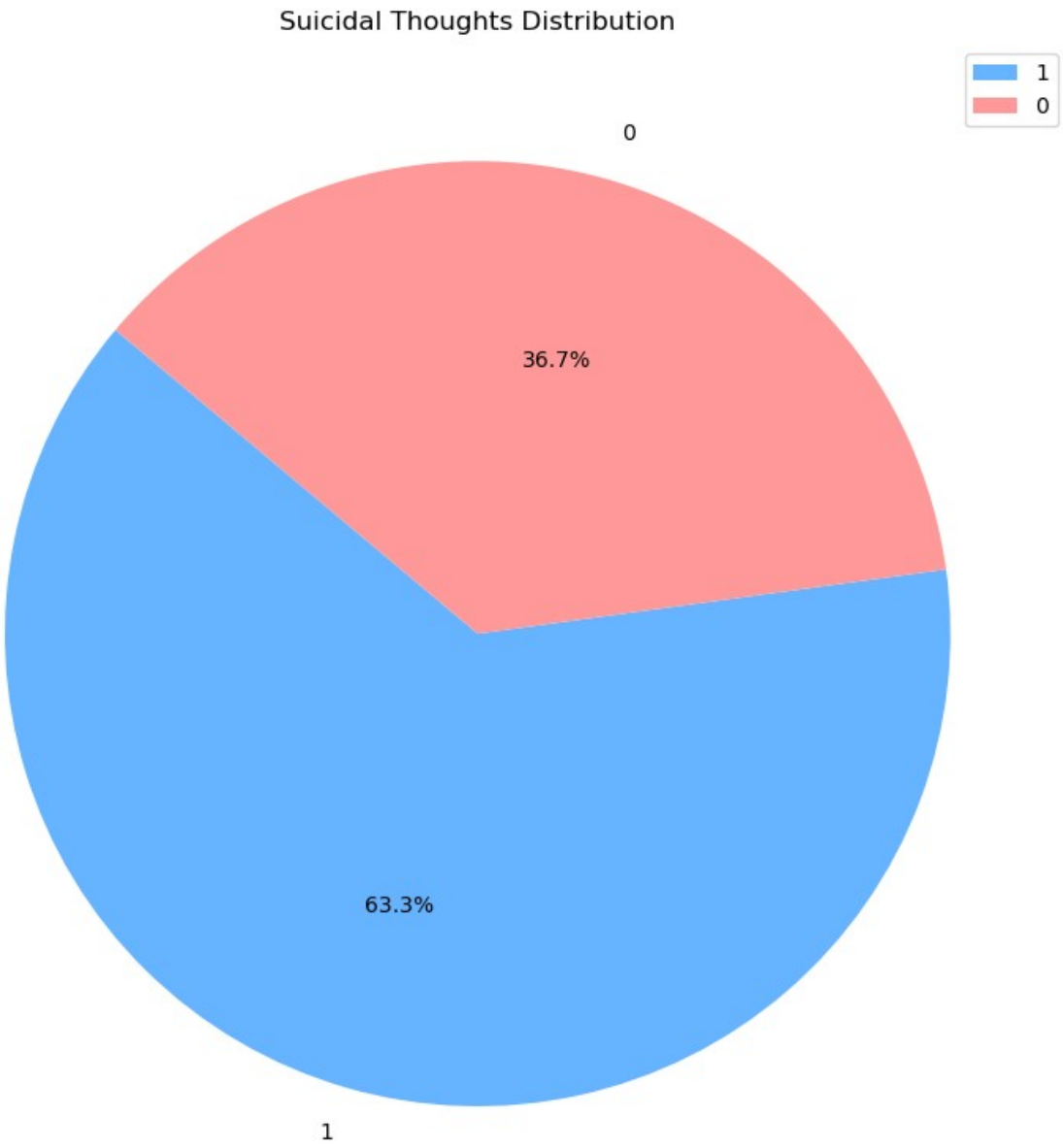
```python
import matplotlib.pyplot as plt

# Group by 'City' and count the number of entries
city_summary = df.groupby("City")["Depression"].count()

# Plotting
city_summary.plot(kind="bar", color="blue", edgecolor="black")
plt.title("Total Responses by City")
plt.xlabel("City")
plt.ylabel("Total Responses")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```




```
import matplotlib.pyplot as plt

# Count of each Financial Stress level
financial_stress_counts = df['Financial Stress'].value_counts()

# Plot
plt.figure(figsize=(8, 6))
financial_stress_counts.plot(kind="bar", color="skyblue",
edgecolor="black")
plt.title("Financial Stress Distribution")
plt.xlabel("Financial Stress Level")
plt.ylabel("Count")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



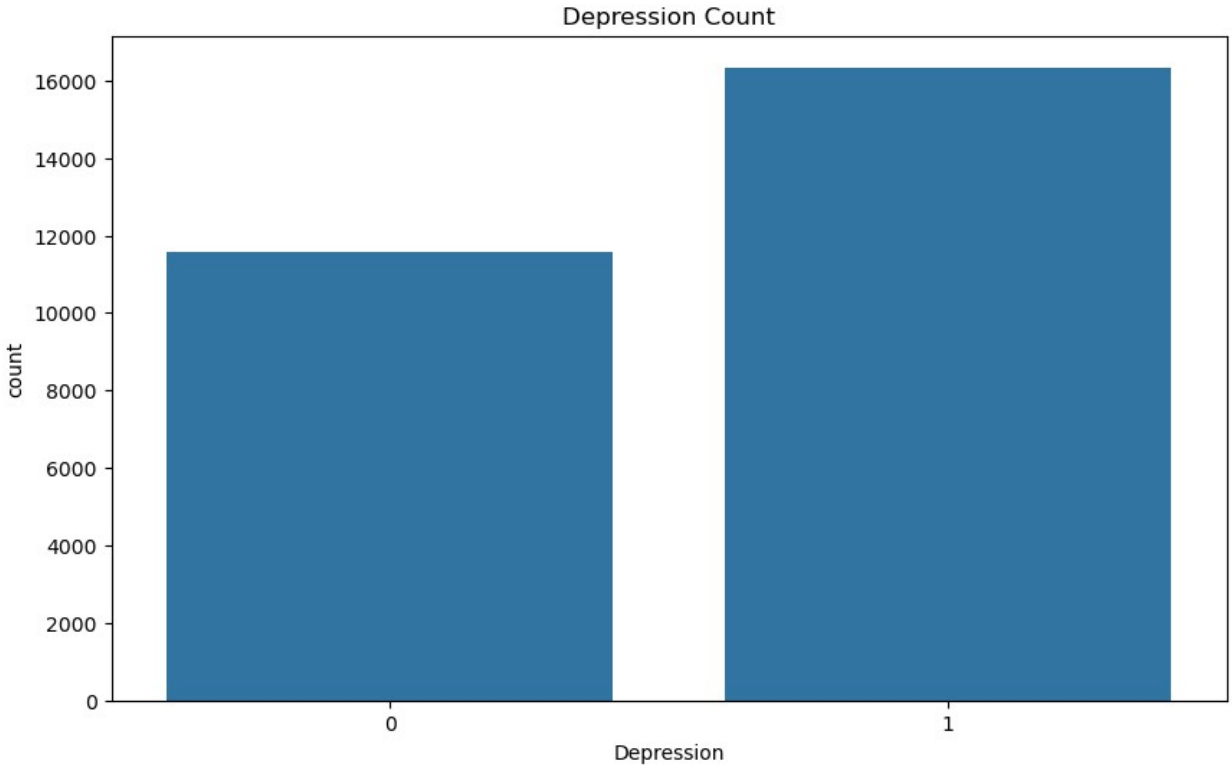

```
import matplotlib.pyplot as plt

# Count values for 'Have you ever had suicidal thoughts ?'
suicidal_thoughts_counts = df['Have you ever had suicidal
thoughts ?'].value_counts()

# Plot pie chart
plt.figure(figsize=(8, 8))
suicidal_thoughts_counts.plot(
    kind='pie',
    autopct='%1.1f%%',
    startangle=140,
    colors=['#66b3ff', '#ff9999'],
    legend=True,
    title='Suicidal Thoughts Distribution'
)

plt.ylabel('')  # Remove y-axis label for clean look
plt.tight_layout()
plt.show()
```

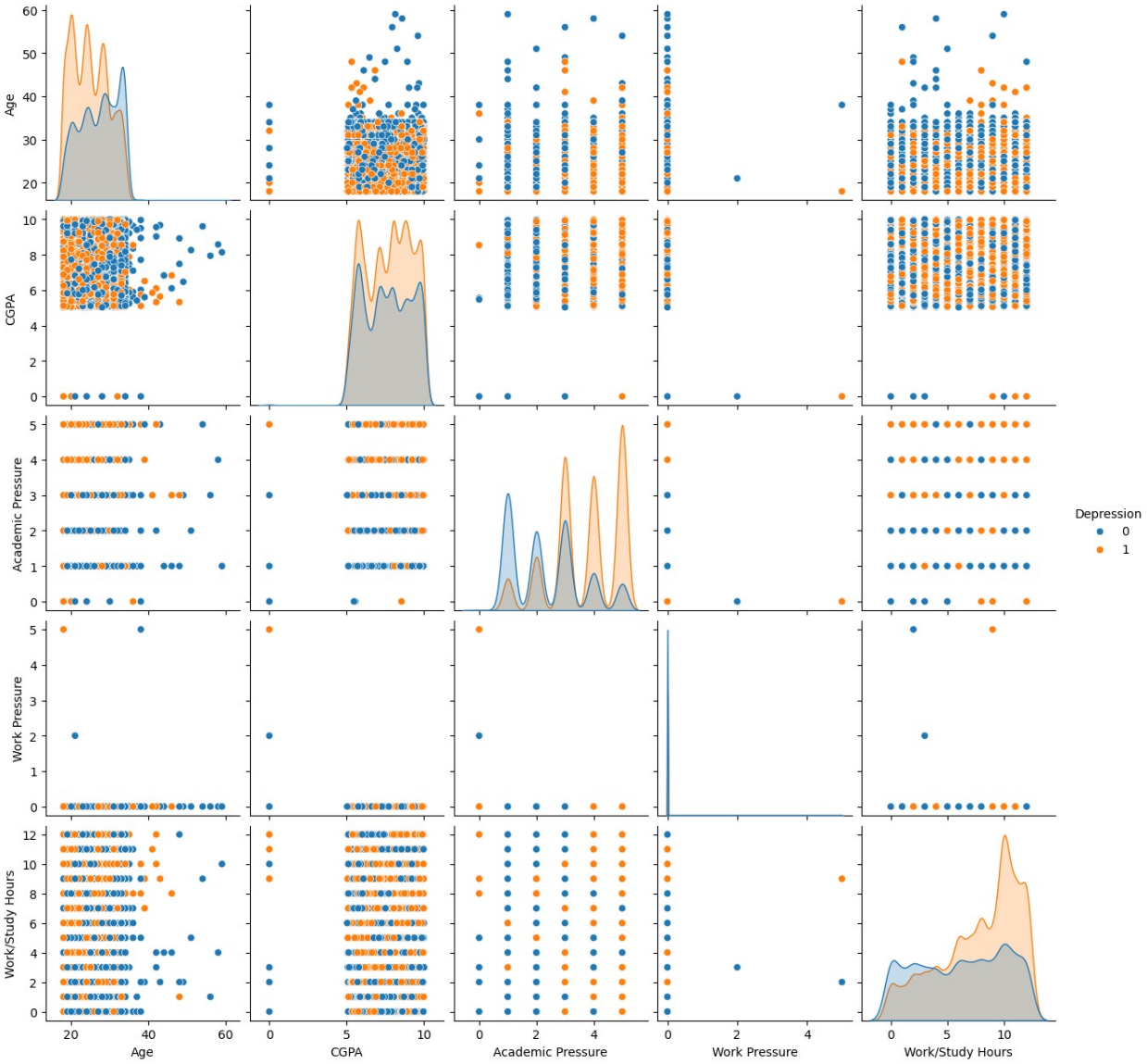


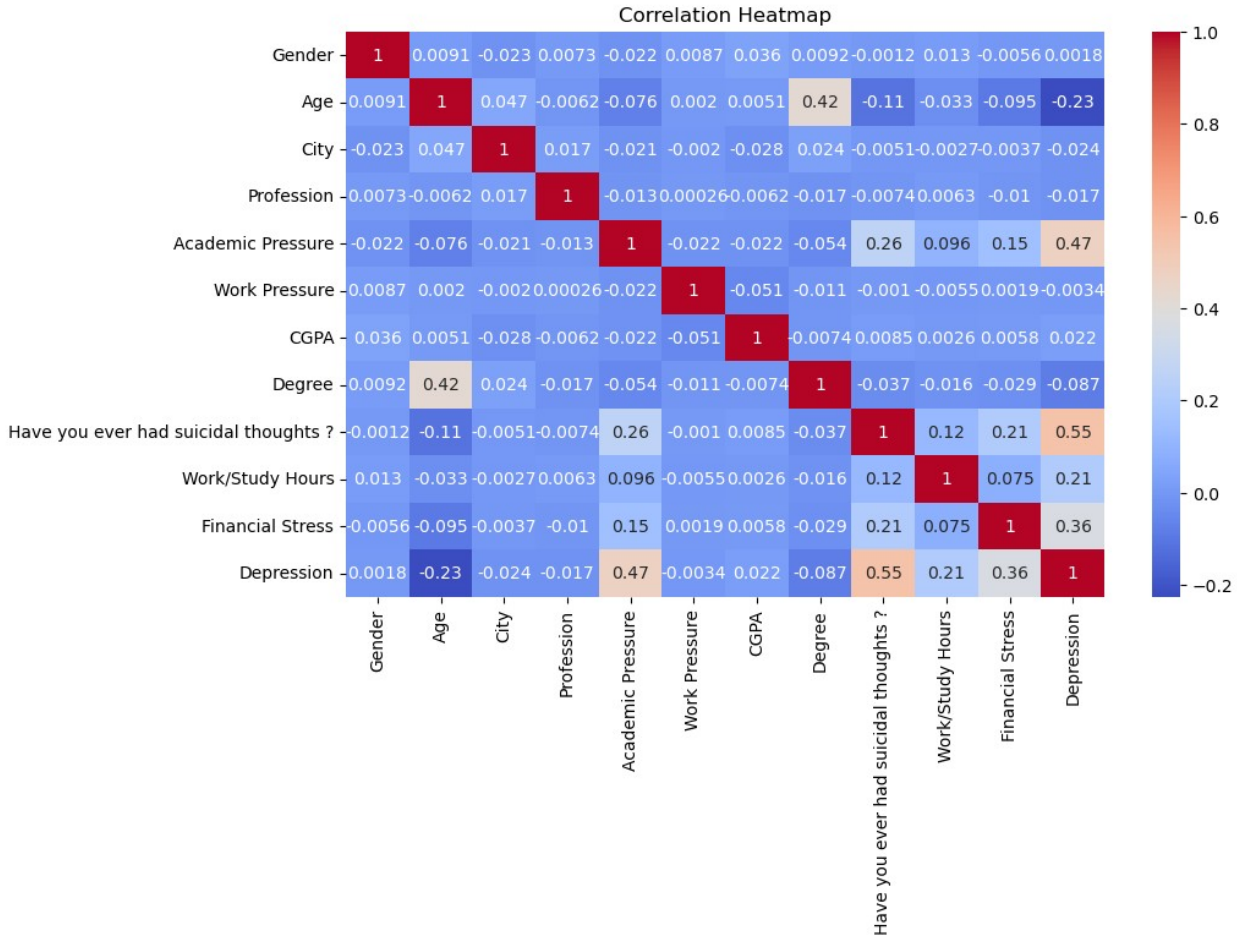
Exploratory Data Analysis (EDA)

```
plt.figure(figsize=(10, 6))
sns.countplot(x="Depression", data=df)
plt.title("Depression Count")
plt.show()

sns.pairplot(df[['Age', 'CGPA', 'Academic Pressure', 'Work Pressure',
'Work/Study Hours', 'Depression']], hue='Depression')
plt.show()
```

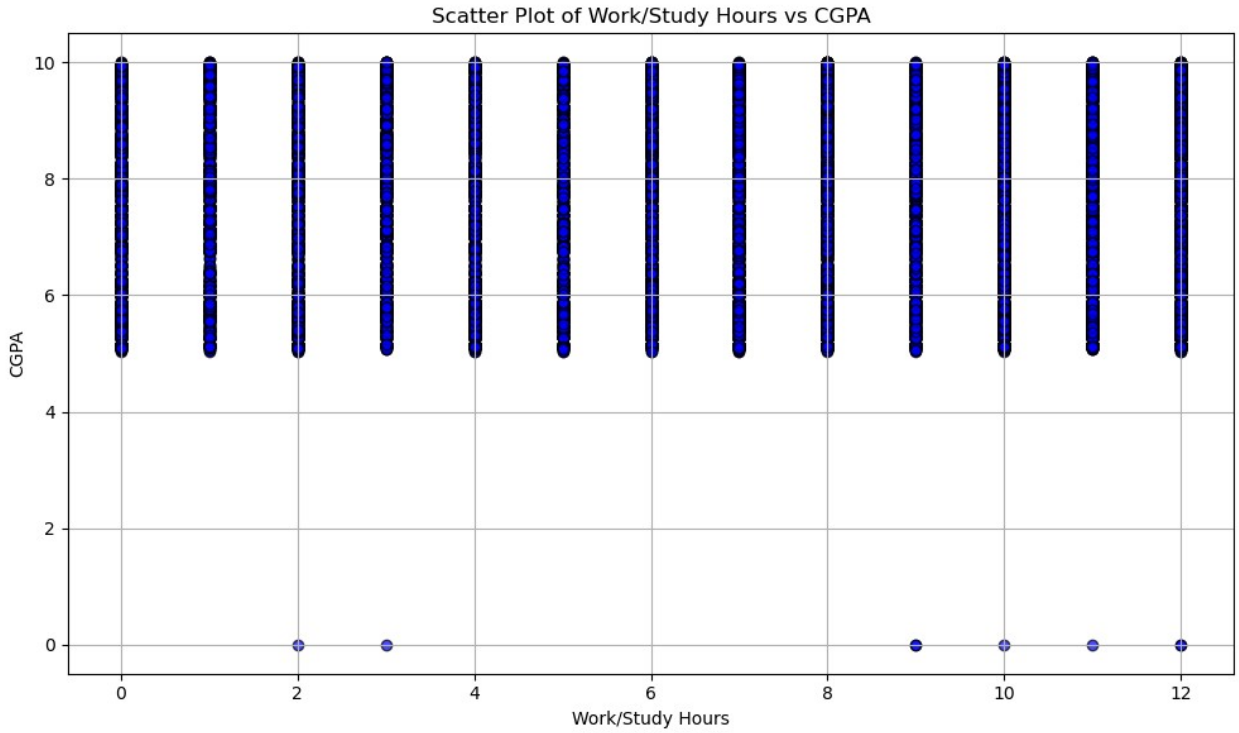
```
plt.figure(figsize=(10, 6))
sns.heatmap(df.corr(), annot=True, cmap="coolwarm")
plt.title("Correlation Heatmap")
plt.show()
```



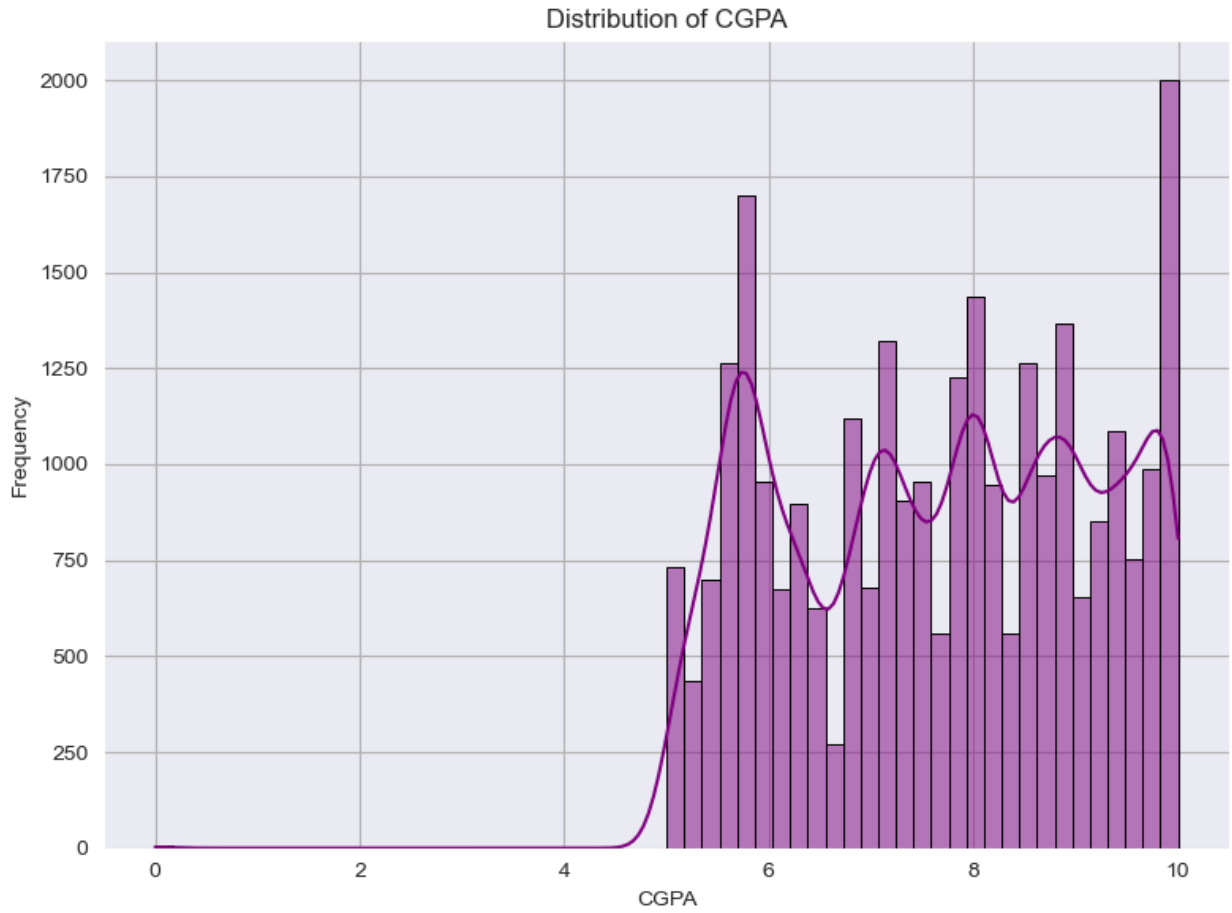

Age
CGPA
Academic Pressure
Work Pressure
Work/Study Hours
Depression
0
1




```python
import pandas as pd
import matplotlib.pyplot as plt

# Create a scatter plot of Work/Study Hours vs CGPA
plt.figure(figsize=(10, 6))
plt.scatter(df['Work/Study Hours'], df['CGPA'], alpha=0.7, c='blue',
edgecolor='k')

# Add labels and title
plt.xlabel("Work/Study Hours")
plt.ylabel("CGPA")
plt.title("Scatter Plot of Work/Study Hours vs CGPA")
plt.grid(True)
plt.tight_layout()
plt.show()
```




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

# Ignore warnings
warnings.filterwarnings('ignore')

# Set Seaborn style with custom grid
sns.set_style('darkgrid', {
    "grid.color": "0.7",
    "grid.linestyle": "-"
})

# Plot histogram of CGPA
plt.figure(figsize=(8, 6))
sns.histplot(df['CGPA'], kde=True, color='purple', edgecolor='black')

plt.xlabel("CGPA")
plt.ylabel("Frequency")
plt.title("Distribution of CGPA")
plt.tight_layout()
plt.show()
```



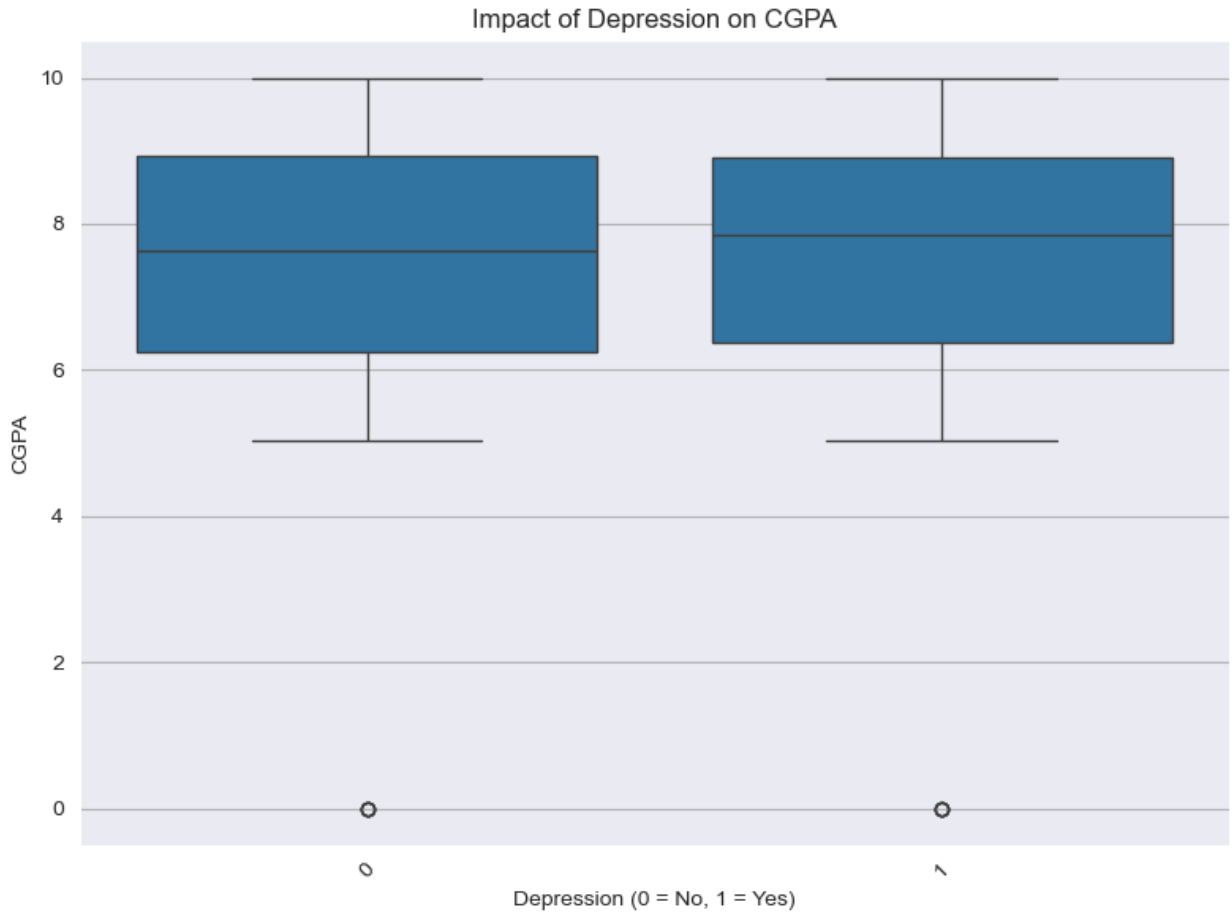

```
df.isna()
```

```
       Gender    Age   City  Profession  Academic Pressure  Work
Pressure  \
0       False  False  False       False              False
False
1       False  False  False       False              False
False
2       False  False  False       False              False
False
3       False  False  False       False              False
False
4       False  False  False       False              False
False
...       ...    ...    ...         ...                ...
...
27896   False  False  False       False              False
False
27897   False  False  False       False              False
False
27898   False  False  False       False              False
False
```

```
27899   False  False  False       False               False
False
27900   False  False  False       False               False
False

        CGPA  Degree  Have you ever had suicidal thoughts ?
Work/Study Hours  \
0      False   False                                False
False
1      False   False                                False
False
2      False   False                                False
False
3      False   False                                False
False
4      False   False                                False
False
...      ...     ...                                  ...
...
27896  False   False                                False
False
27897  False   False                                False
False
27898  False   False                                False
False
27899  False   False                                False
False
27900  False   False                                False
False

       Financial Stress  Depression  CGPA_per_Hour
0                 False       False          False
1                 False       False          False
2                 False       False          False
3                 False       False          False
4                 False       False          False
...                 ...         ...            ...
27896             False       False          False
27897             False       False          False
27898             False       False          False
27899             False       False          False
27900             False       False          False

[27900 rows x 13 columns]
```

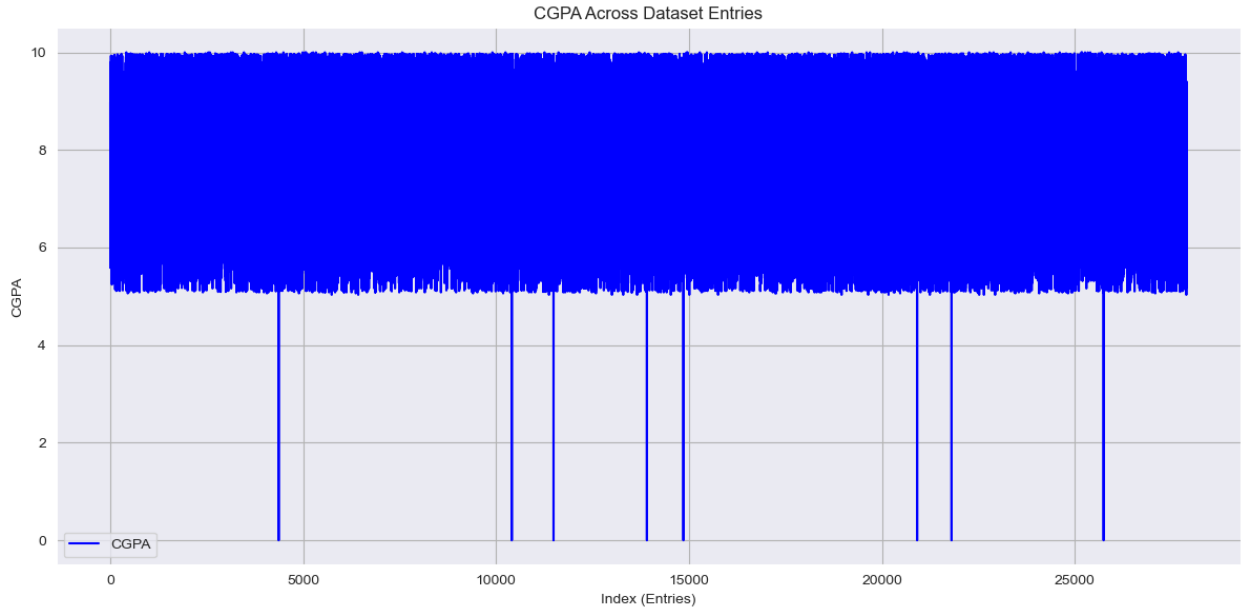
```python
import seaborn as sns
import matplotlib.pyplot as plt

# Create a boxplot of CGPA vs Depression status
plt.figure(figsize=(8, 6))
```

```
sns.boxplot(x="Depression", y="CGPA", data=df)

plt.title("Impact of Depression on CGPA")
plt.xlabel("Depression (0 = No, 1 = Yes)")
plt.ylabel("CGPA")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

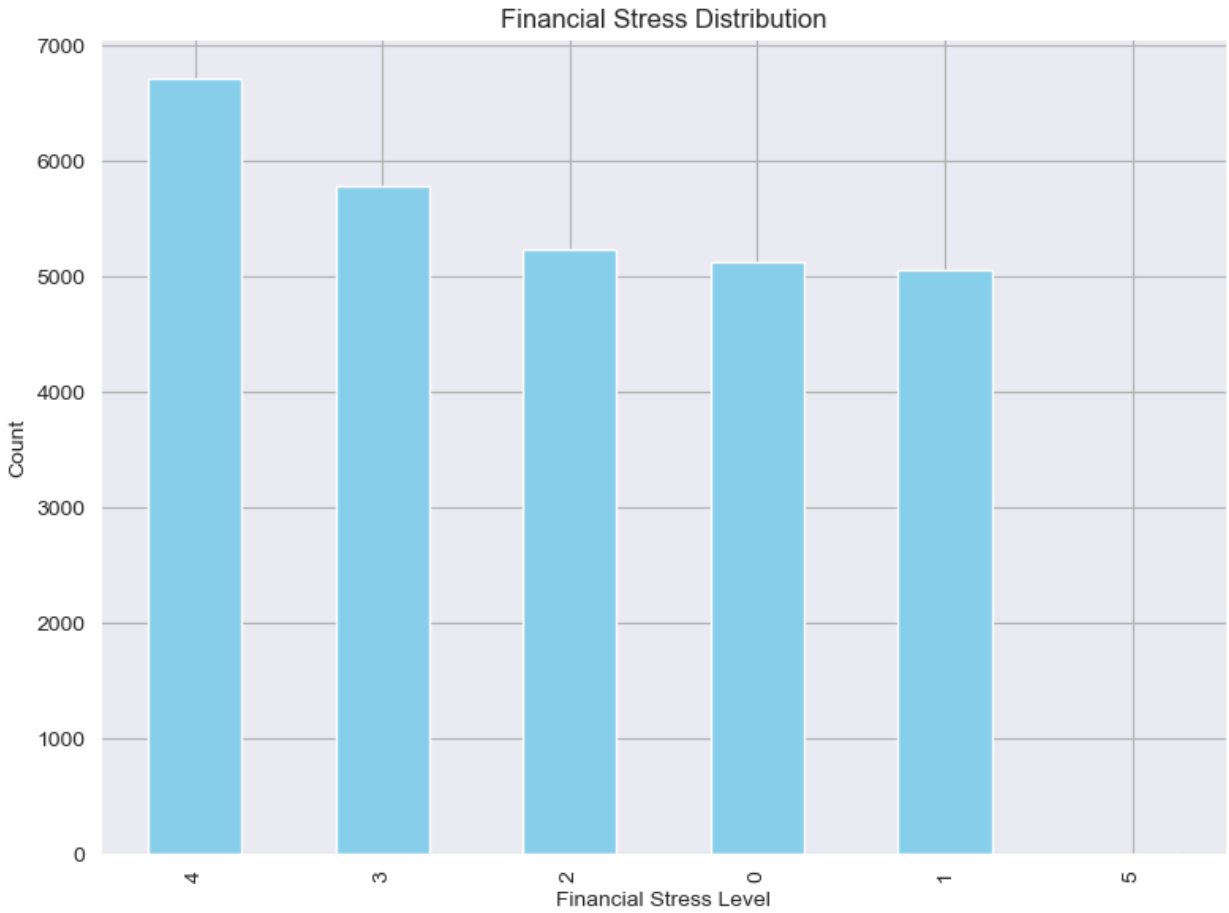



```
import matplotlib.pyplot as plt
import pandas as pd

# Load dataset
# Plot CGPA over index (like a time series)
plt.figure(figsize=(12, 6))
plt.plot(df.index, df['CGPA'], color="blue", label="CGPA")

# Add labels and title
plt.xlabel("Index (Entries)")
plt.ylabel("CGPA")
plt.title("CGPA Across Dataset Entries")
```

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




```
import matplotlib.pyplot as plt

# Plot distribution of Financial Stress
df['Financial Stress'].value_counts().plot(
    kind="bar",
    color="skyblue",
    figsize=(8, 6)
)

plt.title("Financial Stress Distribution")
plt.xlabel("Financial Stress Level")
plt.ylabel("Count")
plt.tight_layout()
plt.show()
```



Feature Selection & Model Building

```
X = df.drop("Depression", axis=1)
y = df["Depression"]

# Scale features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Split data
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y,
test_size=0.2, random_state=42)

# Train model
model = RandomForestClassifier(random_state=42)
model.fit(X_train, y_train)

# Predict
y_pred = model.predict(X_test)
```

```
---------------------------------------------------------------------
-----
ValueError                                Traceback (most recent call
last)
Cell In[83], line 6
      4 # Scale features
      5 scaler = StandardScaler()
----> 6 X_scaled = scaler.fit_transform(X)
      8 # Split data
      9 X_train, X_test, y_train, y_test = train_test_split(X_scaled,
y, test_size=0.2, random_state=42)

File ~\anaconda3\Lib\site-packages\sklearn\utils\_set_output.py:313,
in _wrap_method_output.<locals>.wrapped(self, X, *args, **kwargs)
    311 @wraps(f)
    312 def wrapped(self, X, *args, **kwargs):
--> 313     data_to_wrap = f(self, X, *args, **kwargs)
    314     if isinstance(data_to_wrap, tuple):
    315         # only wrap the first output for cross decomposition
    316         return_tuple = (
    317             _wrap_data_with_container(method, data_to_wrap[0],
X, self),
    318             *data_to_wrap[1:],
    319         )

File ~\anaconda3\Lib\site-packages\sklearn\base.py:1098, in
TransformerMixin.fit_transform(self, X, y, **fit_params)
   1083         warnings.warn(
   1084             (
   1085                 f"This object ({self.__class__.__name__}) has
a `transform`"
   (...)
   1093             UserWarning,
   1094         )
   1096 if y is None:
   1097     # fit method of arity 1 (unsupervised transformation)
-> 1098     return self.fit(X, **fit_params).transform(X)
   1099 else:
   1100     # fit method of arity 2 (supervised transformation)
   1101     return self.fit(X, y, **fit_params).transform(X)

File ~\anaconda3\Lib\site-packages\sklearn\preprocessing\_data.py:878,
in StandardScaler.fit(self, X, y, sample_weight)
    876 # Reset internal state before fitting
    877 self._reset()
--> 878 return self.partial_fit(X, y, sample_weight)

File ~\anaconda3\Lib\site-packages\sklearn\base.py:1473, in
_fit_context.<locals>.decorator.<locals>.wrapper(estimator, *args,
**kwargs)
```

```
   1466     estimator._validate_params()
   1468 with config_context(
   1469     skip_parameter_validation=(
   1470         prefer_skip_nested_validation or
global_skip_validation
   1471     )
   1472 ):
-> 1473     return fit_method(estimator, *args, **kwargs)

File ~\anaconda3\Lib\site-packages\sklearn\preprocessing\_data.py:914,
in StandardScaler.partial_fit(self, X, y, sample_weight)
    882 """Online computation of mean and std on X for later scaling.
    883
    884 All of X is processed as a single batch. This is intended for
cases
   (...)
    911     Fitted scaler.
    912 """
    913 first_call = not hasattr(self, "n_samples_seen_")
--> 914 X = self._validate_data(
    915     X,
    916     accept_sparse=("csr", "csc"),
    917     dtype=FLOAT_DTYPES,
    918     force_all_finite="allow-nan",
    919     reset=first_call,
    920 )
    921 n_features = X.shape[1]
    923 if sample_weight is not None:

File ~\anaconda3\Lib\site-packages\sklearn\base.py:633, in
BaseEstimator._validate_data(self, X, y, reset, validate_separately,
cast_to_ndarray, **check_params)
    631         out = X, y
    632 elif not no_val_X and no_val_y:
--> 633     out = check_array(X, input_name="X", **check_params)
    634 elif no_val_X and not no_val_y:
    635     out = _check_y(y, **check_params)

File ~\anaconda3\Lib\site-packages\sklearn\utils\validation.py:1064,
in check_array(array, accept_sparse, accept_large_sparse, dtype,
order, copy, force_writeable, force_all_finite, ensure_2d, allow_nd,
ensure_min_samples, ensure_min_features, estimator, input_name)
   1058     raise ValueError(
   1059         "Found array with dim %d. %s expected <= 2."
   1060         % (array.ndim, estimator_name)
   1061     )
   1063 if force_all_finite:
-> 1064     _assert_all_finite(
   1065         array,
   1066         input_name=input_name,
```

```
   1067         estimator_name=estimator_name,
   1068         allow_nan=force_all_finite == "allow-nan",
   1069     )
   1071 if copy:
   1072     if _is_numpy_namespace(xp):
   1073         # only make a copy if `array` and `array_orig` may
share memory`

File ~\anaconda3\Lib\site-packages\sklearn\utils\validation.py:123, in
_assert_all_finite(X, allow_nan, msg_dtype, estimator_name,
input_name)
    120 if first_pass_isfinite:
    121     return
--> 123 _assert_all_finite_element_wise(
    124     X,
    125     xp=xp,
    126     allow_nan=allow_nan,
    127     msg_dtype=msg_dtype,
    128     estimator_name=estimator_name,
    129     input_name=input_name,
    130 )

File ~\anaconda3\Lib\site-packages\sklearn\utils\validation.py:172, in
_assert_all_finite_element_wise(X, xp, allow_nan, msg_dtype,
estimator_name, input_name)
    155 if estimator_name and input_name == "X" and has_nan_error:
    156     # Improve the error message on how to handle missing
values in
    157     # scikit-learn.
    158     msg_err += (
    159         f"\n{estimator_name} does not accept missing values"
    160         " encoded as NaN natively. For supervised learning,
you might want"
   (...)
    170         "#estimators-that-handle-nan-values"
    171     )
--> 172 raise ValueError(msg_err)

ValueError: Input X contains infinity or a value too large for
dtype('float64').
```

```
 df_cleaned = df.dropna()
 print(df_cleaned)
```

```
       Gender  Age  City  Profession  Academic Pressure  Work Pressure
CGPA  \
0           1   33    51          12                  5              0
8.97
1           0   24     5          12                  2              0
5.90
```

```
2           1   31    44          12                  3               0
7.03
3           0   28    49          12                  3               0
5.59
4           0   25    18          12                  4               0
8.13
...       ...  ...   ...         ...                ...             ...
...
27896       0   27    45          12                  5               0
5.75
27897       1   27    25          12                  2               0
9.40
27898       1   31    11          12                  3               0
6.61
27899       0   18    25          12                  5               0
6.88
27900       1   27    38          12                  4               0
9.24

       Degree  Have you ever had suicidal thoughts ?  Work/Study Hours
\
0           4                                       1                 3

1          11                                       0                 3

2           6                                       0                 9

3           8                                       1                 4

4          17                                       1                 1

...       ...                                     ...               ...

27896       0                                       1                 7

27897      25                                       0                 0

27898      22                                       0                12

27899       0                                       1                10

27900       8                                       1                 2


       Financial Stress  Depression  CGPA_per_Hour
0                     0           1       2.990000
1                     1           0       1.966667
2                     0           0       0.781111
3                     4           1       1.397500
4                     0           0       8.130000
```

```
...                  ...          ...             ...
27896                  0            0        0.821429
27897                  2            0             inf
27898                  1            0        0.550833
27899                  4            1        0.688000
27900                  2            1        4.620000

[27900 rows x 13 columns]
```

```
 print("Missing values in each column before cleaning:")
 print(df.isnull().sum())
```

```
Missing values in each column before cleaning:
Gender                                   0
Age                                      0
City                                     0
Profession                               0
Academic Pressure                        0
Work Pressure                            0
CGPA                                     0
Degree                                   0
Have you ever had suicidal thoughts ?    0
Work/Study Hours                         0
Financial Stress                         0
Depression                               0
CGPA_per_Hour                            0
dtype: int64
```

Evaluation

```
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test,
y_pred))
print("\nAccuracy:", accuracy_score(y_test, y_pred))
```

```
Confusion Matrix:
 [[1782  561]
 [ 457 2781]]

Classification Report:
              precision    recall  f1-score   support

           0       0.80      0.76      0.78      2343
           1       0.83      0.86      0.85      3238

    accuracy                           0.82      5581
   macro avg       0.81      0.81      0.81      5581
weighted avg       0.82      0.82      0.82      5581
```

```
Accuracy: 0.8175954130084214
```

Feature Importance

```
feature_importances = pd.Series(model.feature_importances_,
index=X.columns)
feature_importances.nlargest(10).plot(kind='barh')
plt.title("Top 10 Feature Importances")
plt.show()
```